AVEVA™ Historian

2023 R2

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of AVEVA Group Limited. No liability is assumed with respect to the use of the information contained herein.

Although precaution has been taken in the preparation of this documentation, AVEVA assumes no responsibility for errors or omissions. The information in this documentation is subject to change without notice and does not represent a commitment on the part of AVEVA. The software described in this documentation is furnished under a license agreement. This software may be used or copied only in accordance with the terms of such license agreement. AVEVA, the AVEVA logo and logotype, OSIsoft, the OSIsoft logo and logotype, ArchestrA, Avantis, Citect, DYNSIM, eDNA, EYESIM, InBatch, InduSoft, InStep, IntelaTrac, InTouch, Managed PI, OASyS, OSIsoft Advanced Services, OSIsoft Cloud Services, OSIsoft Connected Services, OSIsoft EDS, PIPEPHASE, PI ACE, PI Advanced Computing Engine, PI AF SDK, PI API, PI Asset Framework, PI Audit Viewer, PI Builder, PI Cloud Connect, PI Connectors, PI Data Archive, PI DataLink, PI DataLink Server, PI Developers Club, PI Integrator for Business Analytics, PI Interfaces, PI JDBC Driver, PI Manual Logger, PI Notifications, PI ODBC Driver, PI OLEDB Enterprise, PI OLEDB Provider, PI OPC DA Server, PI OPC HDA Server, PI ProcessBook, PI SDK, PI Server, PI Square, PI System, PI System Access, PI Vision, PI Visualization Suite, PI Web API, PI WebParts, PI Web Services, PRiSM, PRO/II, PROVISION, ROMeo, RLINK, RtReports, SIM4ME, SimCentral, SimSci, Skelta, SmartGlance, Spiral Software, WindowMaker, WindowViewer, and Wonderware are trademarks of AVEVA and/or its subsidiaries. All other brands may be trademarks of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the license agreement with AVEVA Group Limited or its subsidiaries and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12-212, FAR 52.227-19, or their successors, as applicable.

AVEVA Legal Resources: https://www.aveva.com/en/legal/

AVEVA Third Party Software Notices and Licenses: https://www.aveva.com/en/legal/third-party-software-license/

Contents

Welcome to AVEVA Historian 69
Legal Information 70
Contact information 70

Historian Client User Guides 71
Historian Client User Guide 71
Welcome 71
Legal Information 71
Contact information 72
Introduction 72
About the AVEVA Historian Client Software 72
Desktop Applications 73
Microsoft Office Add-Ins 73
ActiveX and .NET Controls 73
About AVEVA Historian Client Licensing 73
About the AVEVA Historian 74
Client/Server Architecture 74
Integration with AVEVA Application Server 75
Naming Conventions 76
Analyzing Process Data 77
Common Client Components 77
Authenticate using AVEVA Identity Manager 77
AVEVA Connect Profile 81
Sign Out from AVEVA Applications 81
Server Connection Configuration 82
Creating a New Server Connection 82
Editing a Server Connection 84
Reconnecting to a Server 84
Setting the Reconnection Time 85
About Using a Redundant Historian Server 85
Removing a Server Connection 85
Considerations for VPN Access 86
Tag Picker 86
Servers Pane 87
Editing Groups 88
Viewing Server Details 88
Tags Pane 89
Filter Pane 89
Expression Pane 91
Showing/Hiding the Tag Picker 91
Showing/Hiding Panes in the Tag Picker 91
Viewing the Hierarchical Name 92

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag Picker Views . . . 92
Status Bar . . . 92
Time Picker . . . 93
Viewing Program and License Information . . . 94
Client Trend . . . 94
AVEVA Historian Client Trend . . . 94
Viewing License Information . . . 95
Working with Trend Files . . . 96
Creating a New Trend . . . 96
Configuring Default Settings for a Trend File . . . 97
Opening an Existing Trend . . . 97
Saving a Trend . . . 98
Closing a Trend . . . 98
Undoing/Redoing Actions . . . 98
Working with Trend Sets . . . 99
Creating a Trend Set . . . 99
Editing a Trend Set . . . 100
Deleting Files in a Trend Set . . . 101
Configure a Trend . . . 101
Configuring a Trend to Use a Summary Tag . . . 102
Working with Replicated Tags . . . 102
Adding a Source Tag or Replicated Tag . . . 103
Finding a Source Tag or Replicated Tag . . . 103
Replacing a Source Tag or Replicated Tag . . . 104
Viewing Tag Definition Information . . . 105
Viewing the Hierarchical Name in a Trend . . . 107
Viewing Data in the Trend Chart . . . 107
Refreshing the Trend Chart . . . 108
Deleting a Tag . . . 109
Configuring Trend Options for a Tag . . . 109
Configuring Display Options . . . 109
Configuring Displayed Engineering Units for a Tag . . . 111
Defining a Target Region for a Tag . . . 111
Configuring Retrieval Options for a Tag . . . 115
Editing the Expression for an Expression Tag . . . 117
Scrolling through Tags in a Trend . . . 117
Highlighting a Tag . . . 118
Showing a Single Tag in the Trend . . . 118
Stacking Traces . . . 118
Showing Live Data . . . 119
Showing Historical Data in "Replay" Mode . . . 119
Configuring Predictive Data Retrieval . . . 120
Scaling Tags . . . 121
Showing No Scales on the Value Axis . . . 122
Showing Single Scale on the Value Axis . . . 122
Showing Multiple Scales on the Value Axis . . . 124
Showing Cursor Values on the Value Axis . . . 126
Scaling Tags Up or Down . . . 126
Automatically Scaling Tags . . . 127

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Returning Tags to Their Original Scale . . . 128
Moving Tags Up or Down in the Chart . . . 129
Using "Rubber Band" Scaling . . . 129
Panning in the Trend Chart . . . 130
Using Axis Cursors . . . 131
Moving a Cursor . . . 131
Showing/Hiding the Axis Cursors . . . 132
Showing/Hiding the Cursor Difference . . . 132
Zooming . . . 132
Showing/Hiding the Chart Grid . . . 132
Viewing Alarms and Events . . . 133
Viewing Trend Data in a Table Format . . . 136
Viewing the Data Log in a "Narrow" Format . . . 136
Viewing the Data Log in a "Wide" Format . . . 137
Viewing Statistics . . . 139
Using Annotations . . . 140
Adding an Annotation . . . 141
Viewing the Annotation List . . . 141
Editing an Annotation . . . 142
Deleting an Annotation . . . 143
Saving the Annotations List as a .CSV File . . . 143
Printing Annotations . . . 143
Trending Event Tags . . . 145
Using Absolute or Relative Times . . . 146
Using Absolute Time . . . 146
Using Relative Time . . . 147
Switching Between Absolute and Relative Time: Example . . . 148
Time Offset Formats . . . 148
Using Time Offsets to Compare Data in Absolute Time . . . 149
Configuring Trend Application Options . . . 152
Configuring Retrieval Options . . . 153
Configuring Color Options . . . 154
Configuring Time Zone Options . . . 155
Configuring Miscellaneous Options . . . 156
Configuring Other Options . . . 157
Configuring Trend File Properties . . . 158
Configuring General Properties . . . 159
Configuring Color Properties . . . 159
Configuring Axis Properties . . . 161
Configuring Limit Properties . . . 162
Configuring Annotation Properties . . . 163
Configuring Target Region Properties . . . 164
Working with Scatter Plots . . . 165
Viewing Data in a Scatter Plot . . . 167
Scaling Tags in a Scatter Plot . . . 168
Configuring Axes in a Scatter Plot . . . 168
How Are Value Pairs Matched? . . . 168
Quality Calculation for Data Points . . . 169
Panning and Zooming in a Scatter Plot . . . 169

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Defining a Target Region for a Scatter Plot . . . 170
Examples for Target Regions in Scatter Plots . . . 170
Configuring Scatter Plot Properties . . . 172
Other Considerations for Working with Scatter Plots . . . 174
Outputting Trend Data . . . 174
Printing Trend Data . . . 175
Printing Trend Sets . . . 175
Printing a Trend Set . . . 175
Saving Trend Data to a .CSV File . . . 175
Saving the Trend Chart to an Image File . . . 176
E-mailing a Trend File . . . 176
Copying a Trend Chart to the Windows Clipboard . . . 176
Publishing Trends to the AVEVA Information Server . . . 177
Publishing a Static Trend Report . . . 177
Publishing a Dynamic Trend Report . . . 178
Using Trend with a Tablet PC . . . 179
Annotating a Chart . . . 179
Making Chart Annotations . . . 180
Selecting, Copying, and Deleting Chart Annotations . . . 180
Saving, Printing, and E-Mailing an Annotated Chart . . . 180
Importing .CRV Data . . . 181
Client Query . . . 181
Getting Started with Query . . . 181
Viewing License Information . . . 182
Query Menu . . . 183
Columns Pane . . . 184
Results Pane . . . 184
Viewing the Hierarchical Name in a Query . . . 184
Finding a Source Tag or Replicated Tag . . . 184
Status Bar . . . 186
Working with Query Files . . . 186
Opening an Existing Query File . . . 186
Saving a Query File . . . 186
Creating a Query . . . 187
Query Types . . . 188
Query Type: Aggregate Values . . . 188
Criteria Tab . . . 189
Calculations Tab . . . 189
Query Type: Alarm History (Data blocks) . . . 190
Columns Tab . . . 190
Format Tab . . . 191
Query Type: Alarm Summary . . . 191
Alarm summary options Tab . . . 192
Query Type: Analog Statistics . . . 193
Columns Tab . . . 193
Query Type: Annotations . . . 194
Criteria Tab . . . 195
Query Type: Custom . . . 195
Query Type: Favorites . . . 195

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Query Type: History Values . . . 196
Columns Tab . . . 196
Criteria Tab . . . 198
Retrieval Tab . . . 199
Other Tab . . . 199
Query Type: IO Server . . . 200
Query Type: Live Values . . . 201
Columns Tab . . . 201
Query Type: Number of Tags . . . 202
Query Type: Server Version . . . 202
Query Type: Slice By . . . 202
Columns Tab . . . 203
Criteria Tab . . . 204
Query Type: State Statistics . . . 204
Columns Tab . . . 204
Criteria Tab . . . 206
Query Type: Storage . . . 206
Query Type: Storage Size Available . . . 206
Query Type: Storage Start Date . . . 207
Query Type: Tag Details . . . 207
Query Type: Tag Search . . . 208
Search Tab . . . 208
Query Type: Time Running . . . 209
Common Tabs for Query Types . . . 209
Time Tab . . . 209
Format Tab . . . 210
Retrieval Tab . . . 210
Source Tab . . . 211
Order Tab . . . 211
Client Workbook . . . 211
Getting Started . . . 211
Managing Server Connections . . . 212
Opening an Existing Workbook File . . . 212
Manually Loading/Unloading the Excel Add-In . . . 213
Creating a Report: Overview . . . 213
Working with Functions, Formulas, and Cells . . . 214
Refreshing a Function or Array Formula . . . 214
Editing a Function . . . 215
Converting a Function to Values . . . 215
Refreshing a Sheet . . . 215
Converting a Sheet to Values . . . 215
Manually Inserting a Function . . . 215
Manually Editing a Function . . . 216
Copying a Function . . . 216
Selecting Cells . . . 216
Verifying the Date/Time Format in Microsoft Excel . . . 217
Selecting Tags for Reports . . . 217
Selecting Analog, Discrete, String, Summary, or Event Tags . . . 217
Viewing the Hierarchical Name in a Sheet . . . 218

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Selecting Summary Tags . . . 218
Finding a Source Tag or Replicated Tag . . . 219
Selecting Event Snapshot Tags . . . 220
Retrieving Tag Configuration Information . . . 220
Retrieving Configuration Details for a Tag . . . 220
Viewing the Hierarchical Name in a Sheet . . . 222
Retrieving Analog Tag Alarm Limits . . . 222
Retrieving Tag Values . . . 222
Retrieving Live Values . . . 223
Retrieving History Values . . . 224
Display Options Tab . . . 225
Retrieval Tab . . . 226
Retrieving Aggregate Values . . . 226
Calculations Tab . . . 227
Resolution Tab . . . 227
Retrieving Values for Summarized Tags . . . 227
Summary Options Tab . . . 228
Retrieving Values for Event Snapshot Tags . . . 228
Common Properties for Tag Values . . . 229
Display Options Tab . . . 229
Format Tab . . . 230
Criteria Tab . . . 230
Order Tab . . . 231
Analyzing Tag Data . . . 232
Analog Tag Analysis . . . 232
Batch Analysis . . . 235
Scatter Analysis . . . 237
Discrete Tag Analysis . . . 239
Analog Values at Discrete Transition Analysis . . . 241
Analog/Discrete Pair Analysis . . . 243
Creating a Direct Query . . . 245
Configuring Workbook Options . . . 245
Configuring Global Formatting Options . . . 245
Referencing Formatting Options in a Query . . . 246
Using a Named Range for Formatting Options . . . 247
Changing Formatting Options in Named Range . . . 247
Configuring Time Zone Options . . . 247
Configuring Data Source Options . . . 248
Configuring General Options . . . 249
Setting the Base Date and Base Time Parameters . . . 249
Using "Binding" Tags to a Query at Run Time . . . 249
Creating a Bound Report . . . 250
Considerations for Changing Binding Values . . . 250
Time Options for Queries . . . 251
Bound Times . . . 251
Relative Time . . . 251
Absolute Time . . . 251
Publishing Reports . . . 252
Publishing a Static Workbook Report . . . 252

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Publishing a Dynamic Workbook Report . . . 253
Migrate data from Historian Client Workbook to the Excel Add-in . . . 253
Use the Excel Migration tool . . . 254
Supported formulas for migration . . . 254
Live formula migration mapping details . . . 255
Tag details formula migration mapping details . . . 258
History formula migration mapping details . . . 260
Tag analysis formula mapping details . . . 265
Aggregate formula migration mapping details . . . 266
AVEVA Historian Client Workbook Function Reference . . . 267
Function Arguments . . . 274
ActionType . . . 275
AggCalc . . . 276
AnalogFilter . . . 276
DataSource . . . 277
DateTime . . . 277
Description . . . 277
DescriptionFilter . . . 277
DetectDatetime . . . 277
DetectorType . . . 277
DisplayAsWide . . . 277
DisplayDatetime . . . 277
DisplayFlags . . . 278
DisplayMilliseconds . . . 279
DisplayOPCQuality . . . 279
DisplayQuality . . . 279
DisplaySourceServer . . . 279
DisplaySourceTag . . . 279
DisplayTagName . . . 279
EdgeDetection . . . 279
EndDate . . . 280
EngUnit . . . 280
EURange . . . 280
EventTagRange . . . 280
HistoryVersion . . . 280
Interpolation . . . 281
Logged . . . 281
MaxLength . . . 281
Messages . . . 282
OptionRange . . . 282
OrderBy . . . 282
QualityRule . . . 282
RawRange . . . 282
ReplacePoorQuality . . . 283
Reset . . . 283
RetrievalMode . . . 283
RowLimit . . . 283
RowOrRes . . . 283
ScanRate . . . 284

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SnapshotTagRange . . . 284
SnapshotTagType . . . 284
SourceServer . . . 284
SourceTag . . . 284
StartDate . . . 284
State . . . 284
StateCalculation . . . 284
Status . . . 285
SQLQuery . . . 285
Storage . . . 285
SummaryPeriod . . . 285
SummaryType . . . 285
TagCriteria . . . 285
TagFilter . . . 285
TagRange . . . 285
Time1 . . . 285
Time2 . . . 286
TimeDeadband . . . 286
TimestampRule . . . 286
UseStringHistory . . . 286
ValueCriteria . . . 287
ValueDeadband . . . 287
Error Messages for Functions . . . 287
Migrating History Data Retrieval Functions . . . 287
Viewing the AVEVA Historian Details . . . 288
Client Report . . . 288
About Add-ins and Templates . . . 288
Getting Started . . . 288
Manually Loading/Unloading the Word Add-In . . . 291
Managing Server Connections . . . 292
About Field Codes . . . 292
Showing/Hiding Field Codes . . . 294
Opening an Existing Report Document . . . 294
Running a Report Document . . . 294
Saving Report Documents . . . 294
Saving a Report Document . . . 295
Saving a Configured Report Document as a Report Template . . . 295
Saving a Run Report Document as an HTML File . . . 295
Inserting a SQL Query . . . 295
Editing a Query . . . 296
Using Date and Time Options . . . 297
Inserting Date and Time Field Codes . . . 297
About Date and Time Wildcards . . . 297
#time Wildcard . . . 298
#date Wildcard . . . 298
#ReportTime Wildcard . . . 299
Inserting Date and Time Wildcards . . . 299
Configuring Report Options . . . 300
Controls and Objects . . . 300

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About the AVEVA Historian Client Controls and Objects . . . 300
About Properties, Methods, and Events . . . 301
Getting Started with the Controls . . . 301
Using the Controls in Different Environments . . . 302
Using the Controls within InTouch HMI Software . . . 302
Using the Controls in Microsoft Office . . . 302
Mapping for Numerical Data Types . . . 302
Controls . . . 303
aaHistClientTrend Control . . . 303
Using aaHistClientTrend at Runtime . . . 303
Using aaHistClientTrend in an Application . . . 303
Adding aaHistClientTrend to an InTouch Window . . . 304
Using aaHistClientTrend in a Multi-Monitor Environment . . . 304
aaHistClientTrend Unsupported Objects . . . 305
aaHistClientTrend Properties . . . 305
AddMultipleTags . . . 309
AllowContextMenu . . . 310
AllowGridEditing . . . 310
AlwaysUseFullForXYScatterPlots . . . 311
AnalogPlottingAlgorithm . . . 311
ApplyRubberBandToAllTags . . . 311
AutoRefreshMode . . . 312
BackColor . . . 313
BackGradientEndColor . . . 313
BackGradient . . . 313
BackImage . . . 314
BorderColor . . . 314
BorderStyle . . . 314
BorderWidth . . . 315
ChartType . . . 315
CurrentServerName . . . 315
CurrentTagColor . . . 316
CurrentTagCycleCount . . . 316
CurrentTagEffectiveRetrievalMode . . . 317
CurrentTagFormat . . . 317
CurrentTagHistoryVersion . . . 317
CurrentTagIndex . . . 318
CurrentTagInterpolationType . . . 318
CurrentTagName . . . 319
CurrentTagNumStyles . . . 319
CurrentTagOffsetMS . . . 319
CurrentTagPenStyle . . . 320
CurrentTagPenWidth . . . 320
CurrentTagPrecision . . . 321
CurrentTagQualityRule . . . 321
CurrentTagResolution . . . 321
CurrentTagRetrievalMode . . . 322
CurrentTagRetrievalStyle . . . 322
CurrentTagRowLimit . . . 323

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentTagStartDate . . . 323
CurrentTagState . . . 324
CurrentTagStateCalculation . . . 324
CurrentTagTargetRegionVisible . . . 325
CurrentTagTimeDeadband . . . 325
CurrentTagTimeStampRule . . . 325
CurrentTagTrendType . . . 326
CurrentTagUseAutoCycles . . . 326
CurrentTagUseResolution . . . 327
CurrentTagValAtX1 . . . 327
CurrentTagValAtX2 . . . 328
CurrentTagValueDeadband . . . 328
CurrentTrendItem . . . 328
- Visible . . . 329
- PenWidth . . . 329
- Style . . . 330
- ValueFormat . . . 330
- DecimalPlaces . . . 331
- BottomY . . . 331
- TopY . . . 332
- TrendType . . . 332
- Name . . . 332

CurrentValOfX1 . . . 333
CurrentValOfX2 . . . 333
CurrentValOfY1 . . . 334
CurrentValOfY2 . . . 334
CurrentXAxisTagIndex . . . 334
CurrentXAxisTagName . . . 335
CurrentXAxisTagServerName . . . 335
CyclicRows . . . 335
DataPointLabelType . . . 336
DateMode . . . 336
DatePickerFormatString . . . 337
DefaultTagFormat . . . 337
DefaultTagPrecision . . . 338
EnableDeltaRetrieval . . . 338
EnableSummaryData . . . 339
EnableTimeOffsets . . . 339
EndDate . . . 339
FileName . . . 340
GridColor . . . 340
GridHorizontal . . . 340
GridVertical . . . 341
GridVisible . . . 341
HideCurrentTag . . . 341
HighlightCurrentTag . . . 342
HistorySource . . . 342
LiveModeRate . . . 342
LoginTimeout . . . 343

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MaxDeltaSamples . . . 343
LockDown . . . 344
MaxMinutesForDeltaAnalog . . . 344
MaxMinutesForDeltaDiscrete . . . 345
MaxSamplesPerTag . . . 345
MovingAverageMode . . . 345
MovingAverageSamples . . . 346
NumDataPointLabels . . . 346
NumTimeAxisGridPerValue . . . 346
NumTimeAxisValues . . . 347
NumXValueAxisGridLinesPerLabel . . . 347
NumXValueAxisLabels . . . 347
NumYAxisGridPerValue . . . 348
NumYAxisValues . . . 348
PanPercentage . . . 348
PlaybackSpeed . . . 349
PlotColor . . . 349
PlotGradient . . . 349
PlotGradientEndColor . . . 350
PlotImage . . . 350
PrintShowActiveTag . . . 350
PrintShowMarkers . . . 351
PrintShowTitle . . . 351
PrintTitle . . . 352
PublicAnnotations . . . 352
QueryTimeout . . . 352
RealTimeMode . . . 353
RealTimeRate . . . 353
RetrievalOptionsCycleCount . . . 353
RetrievalOptionsHistoryVersion . . . 354
RetrievalOptionsInterpolationType . . . 354
RetrievalOptionsNumStyles . . . 355
RetrievalOptionsQualityRule . . . 355
RetrievalOptionsResolution . . . 355
RetrievalOptionsRetrievalMode . . . 356
RetrievalOptionsRetrievalStyle . . . 356
RetrievalOptionsRowLimit . . . 357
RetrievalOptionsState . . . 357
RetrievalOptionsStateCalculation . . . 358
RetrievalOptionsTimeDeadband . . . 358
RetrievalOptionsTimeStampRule . . . 358
RetrievalOptionsUseAutoCycles . . . 359
RetrievalOptionsUseResolution . . . 359
RetrievalOptionsValueDeadband . . . 360
RetrieveAnnotations . . . 360
RetrieveExtensionData . . . 361
RetrieveManualData . . . 361
RTRate . . . 361
Rubberband . . . 362

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RubberbandAll . . . 362
RubberBandScaling . . . 362
Servers . . . 363
ShowLimits . . . 363
ShowValuesAtCursor . . . 363
ShowWaitCursor . . . 364
ShowXAxisCursors . . . 364
ShowYAxisCursor . . . 365
SingleTagMode . . . 365
StartDate . . . 365
SummaryDataMode . . . 366
SupressErrors . . . 366
TagGridOrientation . . . 366
TagListRows . . . 367
TagPicker . . . 367
TagPickerVisible . . . 367
TargetRegionExcursionType . . . 368
TargetRegionOpacity . . . 368
TimeAxisLabelColor . . . 368
TimeBarVisible . . . 369
TimeBarVisible2 . . . 369
TimeSelector . . . 369
ToolBarVisible . . . 370
ToolbarVisible2 . . . 370
ToolTipText . . . 370
TraceGradientEndingPercentage . . . 371
TraceGradientStartingPercentage . . . 371
TraceGradientType . . . 371
TrendFontSize . . . 372
UpdateToCurrentTimeState . . . 372
UseIniFile . . . 372
ValueAxisLabel . . . 373
XCursor1Color . . . 373
XCursor1Pos . . . 373
XCursor2Color . . . 374
XCursor2Pos . . . 374
YCursor1Color . . . 374
YCursor2Color . . . 375
ZoomOutPercentage . . . 375
aaHistClientTrend Methods . . . 375
AboutBox . . . 378
AddAnyTag . . . 378
AddServer . . . 378
AddServerEx . . . 379
AddTag . . . 380
ClearTags . . . 380
CurrentTagGetStyle . . . 381
DeleteCurrentTag . . . 381
FileNew . . . 381

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

FileOpen . . . 382
FileOpenEx . . . 382
FileSave . . . 383
FileSaveEx . . . 383
GetMenuItemEnabled . . . 384
GetTagColor . . . 384
GetTagFormat . . . 385
GetTagOffsetMS . . . 385
GetTagPenStyle . . . 386
GetTagPenWidth . . . 387
GetTagPrecision . . . 387
GetTagValAtX1 . . . 388
GetTagValAtX2 . . . 388
GetTagVisible . . . 389
GetToolbarButtonEnabled . . . 390
GraphStack . . . 390
LoadCRVString . . . 390
LoadTargetRegionFromFile . . . 391
ManualConnect . . . 391
MoveNextTag . . . 391
MovePrevTag . . . 392
PanLeft . . . 392
PanRight . . . 392
PrintGraph . . . 393
PrintGraphDlg . . . 393
PropertiesDlg . . . 393
RefreshData . . . 393
RemoveServer . . . 394
RemoveServerEx . . . 394
RemoveTag . . . 395
RetrievalOptionsGetStyle . . . 395
SaveData . . . 396
SaveImage . . . 397
SaveSettings . . . 397
ScaleAllTags . . . 397
ScaleAllTagsDlg . . . 398
ScaleAutoAllTags . . . 398
ScaleAutoTag . . . 398
ScaleDownAllTags . . . 399
ScaleDownTag . . . 399
ScaleMoveAllTagsDown . . . 399
ScaleMoveAllTagsUp . . . 400
ScaleMoveTagDown . . . 400
ScaleMoveTagUp . . . 400
ScaleTag . . . 401
ScaleTagDlg . . . 401
ScaleUpAllTags . . . 401
ScaleUpTag . . . 402
SetCurrentTag . . . 402

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SetCurrentTagXAxisTag . . . 403
SetCurrentTagXAxisTagIndex . . . 403
SetDates . . . 404
SetDuration . . . 405
SetMenuItemEnabled . . . 405
SetTagColor . . . 407
SetTagFormat . . . 408
SetTagColorDlg . . . 408
SetTagOffsetMS . . . 409
SetTagPenStyle . . . 409
SetTagPenWidth . . . 410
SetTagPrecision . . . 411
SetTagVisible . . . 411
SetTimeSpan . . . 412
SetToolbarButtonEnabled . . . 413
ShowStatistics . . . 414
UnsetCurrentTagXAxisTag . . . 414
ZoomIn . . . 415
ZoomOut . . . 415
aaHistClientTrend Events . . . 415
CurrentTagChanged . . . 416
DatesChanged . . . 416
StateChanged . . . 417
TagDisplayChanged . . . 417
TaglistChanged . . . 418
aaHistClientTrend Enumerations . . . 418
aaChartType Enumeration . . . 418
aaDashStyle Enumeration . . . 419
aaDataPointLabelingType Enumeration . . . 419
aaDateModeEnumeration Enumeration . . . 419
aaInterpolationType Enumeration . . . 419
aaQualityRules Enumeration . . . 420
aaRetrievalMode Enumeration . . . 420
aaRetrievalVersion Enumeration . . . 421
aaStateCalculation Enumeration . . . 422
aaTargetRegionExcursionType Enumeration . . . 423
aaTimeStampRules Enumeration . . . 423
aaTraceGradientType Enumeration . . . 424
aaTrendGradientType Enumeration . . . 424
aaTrendType Enumeration . . . 425
aaTrendValueFormat Enumeration . . . 425
aaUpdateToCurrentTimeState Enumeration . . . 425
aaValueAxisLabelEnumeration Enumeration . . . 426
aaHistClientQuery Control . . . 426
Using aaHistClientQuery at Runtime . . . 426
Using aaHistClientQuery in an Application . . . 426
Adding aaHistClientQuery to an InTouch Window . . . 427
aaHistClientQuery Properties . . . 427
ActiveServer . . . 428

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AllowQueryTypeChange . . . 429
CurrentServer . . . 429
EnableAllQueriesTab . . . 430
FavoriteQueriesFolder . . . 430
FontBold . . . 431
FontCharset . . . 431
FontItalic . . . 432
FontName . . . 433
FontSize . . . 433
LockDown . . . 434
QueryFont . . . 434
QueryString . . . 435
Recordset . . . 435
Servers . . . 436
ToolbarConnectVisible . . . 436
ToolbarEditVisible . . . 436
ToolbarRequeryVisible . . . 437
ToolBarVisible . . . 437
UsePersistedServers . . . 438
aaHistClientQuery Methods . . . 438
AddServer . . . 439
AddServerEx . . . 440
AddTag . . . 440
ClearTags . . . 441
CopyQuery . . . 441
CutQuery . . . 442
FileOpen . . . 442
ManualConnect . . . 443
OpenQuery . . . 443
PasteQuery . . . 443
Refresh . . . 443
RemoveTag . . . 444
SaveQuery . . . 444
SaveResults . . . 445
SetDates . . . 445
SetDuration . . . 446
SetQueryType . . . 446
SetQueryType2 . . . 447
SetTimeSpan . . . 448
ShowAbout . . . 448
aaHistClientQuery Events . . . 449
ModeChanged . . . 449
QueryChanged . . . 450
ServerChanged . . . 450
aaQueryTypeEnumeration . . . 450
aaHistClientTagPicker Control . . . 452
Using aaHistClientTagPicker at Runtime . . . 452
Using aaHistClientTagPicker in an Application . . . 452
Adding aaHistClientTagPicker to an InTouch Window . . . 453

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaHistClientTagPicker Properties . . . 454
CurrentServer . . . 454
DescriptionFilter . . . 455
ExactMatchFilter . . . 455
FilterVisible . . . 455
HideCaption . . . 456
IOAddressFilter . . . 456
SelectedPath . . . 457
SelectedTagCount . . . 457
Servers . . . 458
SingleSelectMode . . . 458
SplitterOrientation . . . 459
TabSelectedIndex . . . 459
TagNameFilter . . . 460
TagSelectedIndex . . . 460
TreeVisible . . . 460
TreeWidth . . . 461
UseHierarchicalName . . . 461
Visible . . . 461
aaHistClientTagPicker Methods . . . 462
ApplyFilter . . . 462
LogOn . . . 462
OpenAndSelectGroup . . . 463
RefreshTags . . . 464
SelectedTag . . . 464
SetFocusOnSelectedTag . . . 465
aaHistClientTagPicker Events . . . 466
OnFilterChanged . . . 466
OnGroupChanged . . . 466
OnTagsPicked . . . 467
OnTagsSelected . . . 467
OnServerChanged . . . 467
OnSelectedTabChanged . . . 468
OnTagNameChanged . . . 468
aaHistClientTagPickerSplitterOrientation Enumeration . . . 468
aaHistClientTimeRangePicker Control . . . 469
Using aaHistClientTimeRangePicker at Runtime . . . 469
Using aaHistClientTimeRangePicker in an Application . . . 469
Adding aaHistClientTimeRangePicker to an InTouch Window . . . 469
aaHistClientTimeRangePicker Properties . . . 469
DurationMS . . . 470
EndDate . . . 470
EndDateUTC . . . 470
Format . . . 471
StartDate . . . 472
StartDateUTC . . . 473
TimeDuration . . . 473
UpdateToCurrentTimeState . . . 474
aaHistClientTimeRangePicker Methods . . . 474

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GetStartAndEndTimes . . . 474
GetStartAndEndTimesUTC . . . 475
RefreshTimes . . . 476
SetStartAndEndTimes . . . 476
SetStartAndEndTimesUTC . . . 477
aaHistClientTimeRangePicker Events . . . 477
OnChange . . . 478
aaHistClientActiveDataGrid Control . . . 478
Using aaHistClientActiveDataGrid at Runtime . . . 478
Data Grid . . . 478
Navigating through Records . . . 479
Configuring the Database Connection . . . 479
Creating or Editing SQL Statements . . . 480
Refreshing the Data Grid . . . 480
Using aaHistClientActiveDataGrid in an Application . . . 480
Adding aaHistClientActiveDataGrid to an InTouch Window . . . 480
aaHistClientActiveDataGrid Properties . . . 481
AllowUserConfiguration . . . 481
AutoRefresh . . . 482
BOF . . . 482
ColumnCount . . . 483
Connected . . . 483
DatabaseName . . . 484
DefaultColumnWidth . . . 484
Domain . . . 485
Enabled . . . 485
EnableShortcutMenu . . . 486
EOF . . . 486
Handle . . . 486
Password . . . 487
RefreshFrequency . . . 487
Row . . . 488
RowCount . . . 488
ServerName . . . 489
ShowErrorDlgs . . . 489
ShowNavigatorBar . . . 490
SQLString . . . 490
UserName . . . 491
VirtualDirectoryName . . . 491
BusinessObjectServer . . . 491
aaHistClientActiveDataGrid Methods . . . 492
ClearGrid . . . 492
ColumnName . . . 493
ColumnValue . . . 493
ColumnValueByName . . . 494
Execute . . . 494
MoveFirst . . . 495
MoveLast . . . 495
MoveNext . . . 496

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MovePrevious . . . 496
RowColumnValue . . . 496
RowColumnValueByName . . . 497
ShowPropertiesDialog . . . 498
SQLAppend . . . 498
aaHistClientActiveDataGrid Events . . . 499
OnClick . . . 499
OnDblClick . . . 500
OnError . . . 500
Script Examples for aaHistClientActiveDataGrid . . . 501
InTouch Example: History Data Over a LAN . . . 501
InTouch Example: Retrieving Data from the Grid . . . 502
aaHistClientActiveDataGrid Error Messages . . . 503
aaHistClientSingleValueEntry Control . . . 504
Using the aaHistClientSingleValueEntry Control at Runtime . . . 504
Adding a Tag Value . . . 504
Using the aaHistClientSingleValueEntry Control in an Application . . . 504
Adding the aaHistClientSingleValueEntry Control to an InTouch Window . . . 505
aaHistClientSingleValueEntry Control Properties . . . 505
AnalogValue . . . 506
CurrentServerName . . . 507
DateTime . . . 507
DateTimeFieldDisable . . . 508
DateTimeFieldVisible . . . 508
DateTimeString . . . 508
DisableTagEntry . . . 509
DisplayErrorMessages . . . 510
FieldLabelPosition . . . 510
FieldLayoutHorizontal . . . 511
HideDateTimeModeTabs . . . 511
HideFieldLabels . . . 512
HideStatusBar . . . 512
InsertButtonVisible . . . 513
InTouchDateTime . . . 513
LastErrorDetails . . . 514
LastErrorMessage . . . 514
LastOperationResult . . . 515
LastOperationSuccessful . . . 515
Pwd . . . 516
Quality . . . 516
QualityDetail . . . 517
QualityDetailFieldDisable . . . 517
QualityDetailFieldVisible . . . 518
QualityFieldDisable . . . 518
QualityFieldVisible . . . 518
RememberEnteredTags . . . 519
Servers . . . 519
StringValue . . . 520
TagName . . . 520

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagNameFieldDisable . . . 521
TagNameFieldVisible . . . 521
TagPickerButtonDisable . . . 521
TagPickerButtonVisible . . . 522
Tags . . . 522
TagType . . . 523
TagValid . . . 523
User . . . 524
UseTimezone . . . 524
Value . . . 525
ValueEx . . . 525
ValueFieldDisable . . . 526
aaHistClientSingleValueEntry Control Methods . . . 526
AddServerEx . . . 526
AddServer . . . 527
AddTag . . . 528
Connect . . . 529
CreateManualTag . . . 529
Disconnect . . . 530
Insert . . . 530
InsertValue . . . 531
Refresh . . . 532
Reset . . . 532
aaHistClientSingleValueEntry Control Events . . . 532
Change . . . 533
InsertComplete . . . 533
InsertFail . . . 534
TagNameChanged . . . 534
ValueChanged . . . 534
aaFieldLabelPositionEnumeration Enumeration . . . 535
aaUseTimeZoneEnumeration Enumeration . . . 535
Server Objects . . . 535
aaServer Object . . . 535
aaServer Properties . . . 535
BaseURLAddress . . . 536
Build . . . 536
Domain . . . 537
LoginID . . . 537
LoggedOn . . . 538
LoginTimeout . . . 538
MachineName . . . 538
Name . . . 539
Password . . . 539
PatchLevel . . . 540
QueryTimeout . . . 540
RetainPassword . . . 540
RevisionNumber . . . 541
SchemaVersion . . . 541
ServerName . . . 542

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ServerType . . . 542
State . . . 543
TrustedConnection . . . 543
UseHttp . . . 543
VirtualDirectoryName . . . 544
aaServer Methods . . . 544
LogOff . . . 544
LogOn . . . 544
aaServers Object . . . 545
aaServers Properties . . . 545
ApplicationName . . . 545
Count . . . 546
Items . . . 546
aaServers Methods . . . 547
Add . . . 547
GetServer . . . 547
Remove . . . 548
Update . . . 548
aaServers Events . . . 549
OnServerAdded . . . 549
OnServerUpdated . . . 550
OnServerRemoved . . . 550
OnServerStateChange . . . 551
Instantiating an aaServers Object . . . 551
aaServerListChangeArgs Object . . . 551
Properties . . . 552
Server . . . 552
aaServerStateChangeArgs Object . . . 552
Properties . . . 552
Server . . . 552
State . . . 553
When . . . 553
Message . . . 554
aaServerState Enumeration . . . 554
aaServerType Enumeration . . . 554
Other Objects . . . 555
aaTag Object . . . 555
Using aaTag in an Application . . . 555
aaTag Properties . . . 555
DateCreated . . . 556
Description . . . 556
IOAddress . . . 557
MaxRaw . . . 557
MinRaw . . . 558
MinEU . . . 558
MaxEU . . . 559
Message0 . . . 559
Message1 . . . 560
Mode . . . 560

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name . . . 561
RawType . . . 561
Server . . . 562
Type . . . 562
TypeAsTagType . . . 563
Units . . . 563
aaHistClientWorkbookRunner and aaHistClientReportRunner Objects . . . 563
aaHistClientWorkbookRunner Object . . . 564
aaHistClientWorkbookRunner Object Properties . . . 564
ErrDescription . . . 564
ErrNumber . . . 565
OutputFile . . . 565
SourceFile . . . 566
ExcelVisible . . . 566
aaHistClientWorkbookRunner Methods . . . 566
Run . . . 567
RunReport . . . 567
RunReport2 . . . 570
aaHistClientReportRunner Object . . . 572
aaHistClientReportRunner Object Properties . . . 572
ErrDescription . . . 573
ErrNumber . . . 573
OutputFile . . . 574
SourceFile . . . 574
WordVisible . . . 575
aaHistClientReportRunner Object Methods . . . 575
Run . . . 575
Workbook and Report Automation Objects . . . 576
AVEVA Historian Client Workbook Object . . . 576
AVEVA Historian Client Workbook Object Methods . . . 576
AddServer . . . 576
Auto_Close . . . 577
Auto_Open . . . 577
GetLastError . . . 578
RunReport . . . 579
AVEVA Historian Client Workbook Menu Methods . . . 581
AVEVA Historian Client Workbook Functions . . . 582
AVEVA Historian Client Workbook Automation Example . . . 582
AVEVA Historian Client Report Object . . . 585
Report Object Properties . . . 586
ReportDate . . . 586
ReportTime . . . 586
Report Object Methods . . . 587
AutoExec . . . 587
AutoExit . . . 587
RunReport . . . 587
aaHistClientGlobalFunctions Object . . . 588
Using aaHistClientGlobalFunctions Object in an Application . . . 588
aaHistClientGlobalFunctions Methods . . . 588

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GetDictionaryPath . . . 588
GetInstallPath . . . 589
GetAFVersion . . . 589
GetWorkstationName . . . 590
MDACOk . . . 590
Common Properties, Methods, Events, Enums, and Data Types . . . 590
Common Properties . . . 591
BackColor . . . 592
BackStyle . . . 592
BorderStyle . . . 592
CausesValidation . . . 593
Container . . . 593
ContextMenuEnabled . . . 593
DataBindings . . . 594
DragIcon . . . 594
DragMode . . . 594
Enabled . . . 594
Font . . . 595
ForeColor . . . 595
Height . . . 595
HelpContextID . . . 595
Index . . . 596
Language . . . 596
Left . . . 596
Name . . . 597
Object . . . 597
Parent . . . 597
TabIndex . . . 597
TabStop . . . 598
Tag . . . 598
ToolTipText . . . 598
Top . . . 598
Transparent . . . 599
Visible . . . 599
WhatsThisHelpID . . . 599
Width . . . 600
Common Methods . . . 600
Drag . . . 600
Move . . . 601
SetFocus . . . 601
ShowWhatsThis . . . 601
ZOrder . . . 601
Common Events . . . 601
Click . . . 602
DblClick . . . 602
DragDrop . . . 602
DragOver . . . 602
GotFocus . . . 603
KeyDown . . . 603

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

KeyPress . . . 603
KeyUp . . . 603
LostFocus . . . 603
MouseDown . . . 604
MouseMove . . . 604
MouseUp . . . 604
Validate . . . 604
Common Enumerations . . . 604
aaRetrievalSource Enumeration . . . 604
aaTagType Enumeration . . . 605
aaTimeRangeEnumeration Enumeration . . . 605
Common Data Types . . . 607
DateTime . . . 607
Color . . . 607
DataSet . . . 608
Font . . . 608
Object . . . 608
Data Retrieval Modes and Options . . . 608
Understanding Retrieval Modes . . . 609
Cyclic Retrieval . . . 609
Cyclic Retrieval - How It Works . . . 610
Cyclic Retrieval - Supported Value Parameters . . . 610
Cyclic Retrieval - Query Example . . . 611
Cyclic Retrieval - Initial Values . . . 612
Cyclic Retrieval - Handling NULL Values . . . 612
Delta Retrieval . . . 612
Delta Retrieval - How It Works . . . 612
Delta Retrieval - Supported Value Parameters . . . 613
Delta Retrieval - Query Examples . . . 613
Delta Retrieval - Query 1 . . . 613
Delta Retrieval - Query 2 . . . 615
Delta Retrieval - Query 3 . . . 615
Delta Retrieval - Query 4 . . . 616
Delta Retrieval - Initial Values . . . 617
Delta Retrieval - Handling NULL Values . . . 617
Full Retrieval . . . 618
Full Retrieval - How It Works . . . 619
Full Retrieval - Supported Value Parameters . . . 619
Full Retrieval - Query Example . . . 619
Full Retrieval - Initial Values . . . 620
Interpolated Retrieval . . . 620
Interpolated Retrieval - How It Works . . . 620
. . . 0
Interpolated Retrieval - Query Examples . . . 621
Interpolated Retrieval - Query 1 . . . 621
Interpolated Retrieval - Query 2 . . . 622
Interpolated Retrieval - Query 3 . . . 624
Interpolated Retrieval - Initial and Final Values . . . 625
Interpolated Retrieval - Handling NULL Values . . . 625

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Best Fit Retrieval . . . 625
Best Fit Retrieval - How It Works . . . 626
Best Fit Retrieval - Supported Value Parameters . . . 627
Best Fit Retrieval - Query Example . . . 628
Best Fit Retrieval - Initial and Final Values . . . 629
Best Fit Retrieval - Handling NULL Values . . . 629
Average Retrieval . . . 629
Average Retrieval - How It Works . . . 630
Average Retrieval - Supported Value Parameters . . . 632
Average Retrieval - Query Examples . . . 632
Average Retrieval - Query 1 . . . 632
Average Retrieval - Query 2 . . . 633
Average Retrieval - Initial and Final Values . . . 634
Average Retrieval - Handling NULL Values . . . 634
Minimum Retrieval . . . 634
Minimum Retrieval - How It Works . . . 634
Minimum Retrieval - Supported Value Parameters . . . 635
Minimum Retrieval - Query Examples . . . 636
Minimum Retrieval - Query 1 . . . 636
Minimum Retrieval - Query 2 . . . 637
Minimum Retrieval - Query 3 . . . 637
Minimum Retrieval - Initial and Final Values . . . 638
Minimum Retrieval - Handling NULL Values and Incomplete Cycles . . . 638
Maximum Retrieval . . . 639
Maximum Retrieval - How It Works . . . 639
Maximum Retrieval - Supported Value Parameters . . . 640
Maximum Retrieval - Query Examples . . . 641
Maximum Retrieval - Query 1 . . . 641
Maximum Retrieval - Query 2 . . . 642
Maximum Retrieval - Query 3 . . . 642
Maximum Retrieval - Initial and Final Values . . . 643
Maximum Retrieval - Handling NULL Values and Incomplete Cycles . . . 643
Integral Retrieval . . . 644
Integral Retrieval - How It Works . . . 644
Integral Retrieval - Supported Value Parameters . . . 645
Integral Retrieval - Query Example . . . 645
Integral Retrieval - wwExpression Query Example . . . 646
Integral Retrieval - Initial and Final Values . . . 647
Integral Retrieval - Handling NULL Values . . . 647
Slope Retrieval . . . 647
Slope Retrieval - How It Works . . . 647
Slope Retrieval - Supported Value Parameters . . . 648
Slope Retrieval - Query Example . . . 649
Slope Retrieval - wwExpression Query Example . . . 650
Slope Retrieval - Initial and Final Values . . . 651
Slope Retrieval - Handling NULL Values . . . 651
Counter Retrieval . . . 651
Counter Retrieval - How It Works . . . 652
Counter Retrieval - Calculations for a Manually Reset Counter . . . 653

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Counter Retrieval - Using a Counter Deadband . . . 653
Counter Retrieval - Supported Value Parameters . . . 654
Counter Retrieval - Initial and Final Values . . . 654
Counter Retrieval - Handling NULL Values . . . 654
Counter Retrieval - Handling Illegal Values . . . 654
Counter Retrieval - Query Example . . . 654
ValueState Retrieval . . . 655
ValueState Retrieval - How It Works . . . 656
ValueState Retrieval - Supported Value Parameters . . . 657
ValueState Retrieval - Query Examples . . . 657
ValueState Retrieval Query 1: Minimum Time in State . . . 658
ValueState Retrieval Query 2: Minimum Time in State for a Single Tag . . . 659
ValueState Retrieval Query 3 . . . 659
ValueState Retrieval Query 4 . . . 660
ValueState Retrieval Query 5 . . . 660
ValueState Retrieval Query 6: Querying State Summary Values . . . 661
ValueState Retrieval - Initial and Final Values . . . 661
ValueState Retrieval - Handling NULL Values . . . 662
RoundTrip Retrieval . . . 662
RoundTrip Retrieval - How It Works . . . 662
RoundTrip Retrieval - Supported Value Parameters . . . 663
RoundTrip Retrieval - Query Examples . . . 663
RoundTrip Retrieval - Initial and Final Values . . . 664
RoundTrip Retrieval - Handling NULL Values . . . 664
Edge Detection for Events (wwEdgeDetection) . . . 665
Edge Detection for Analog Tags . . . 665
Leading Edge Detection for Analog Tags . . . 667
Trailing Edge Detection for Analog Tags . . . 667
Both Leading and Trailing Edge Detection for Analog Tags . . . 668
Edge Detection for Discrete Tags . . . 669
Leading Edge Detection for Discrete Tags . . . 670
Trailing Edge Detection for Discrete Tags . . . 671
Both Leading and Trailing Edge Detection for Discrete Tags . . . 672
Predictive Filter . . . 673
Bounding Value Retrieval . . . 674
Bounding Value Retrieval - How It Works . . . 674
Bounding Value Retrieval - Query Examples . . . 675
Understanding Retrieval Options . . . 676
Which Options Apply to Which Retrieval Modes? . . . 676
Using Retrieval Options in a Transact-SQL Statement . . . 677
Cycle Count (X Values over Equal Time Intervals) (wwCycleCount) . . . 678
Cycle Count - Query Examples . . . 678
Cycle Count - Query 1: Using a Single Tag . . . 679
Cycle Count - Query 2: Using Multiple Tags . . . 679
Resolution (Values Spaced Every X ms) (wwResolution) . . . 680
Resolution - Query Example . . . 681
About Phantom Cycles . . . 682
Value Deadband (wwValueDeadband) . . . 684
Value Deadband - Query Examples . . . 685

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value Deadband - Query 1 . . . 685
Value Deadband - Query 2 . . . 686
Time Deadband (wwTimeDeadband) . . . 688
Time Deadband - Query Examples . . . 689
Time Deadband - Query 1 . . . 689
Time Deadband - Query 2 . . . 690
Time Deadband - Query 3 . . . 692
History Version (wwVersion) . . . 693
History Version - Query Example . . . 693
Interpolation Type (wwInterpolationType) . . . 694
TimeStamp Rule (wwTimeStampRule) . . . 696
Time Zone (wwTimeZone) . . . 698
Quality Rule (wwQualityRule) . . . 699
Quality Rule - Query Examples . . . 700
Quality Rule - Query 1 . . . 700
Quality Rule - Query 2 . . . 701
Quality Rule - Query 3 . . . 702
Quality Rule - Query 4 . . . 703
Quality Rule - Query 5 . . . 704
Quality Rule - Query 6 . . . 705
Quality Rule - Query 7 . . . 706
Quality Rule - Query 8 . . . 706
State Calculation (wwStateCalc) . . . 707
Analog Value Filtering (wwFilter) . . . 709
Statistically Removing Outliers (SigmaLimit) . . . 709
Converting Analog Values to Discrete Values (ToDiscrete) . . . 710
"Zeroing" Around a Base Value (SnapTo) . . . 712
Selecting Values for Analog Summary Tags (wwValueSelector) . . . 713
Expressions (wwExpression) . . . 716
Implicit unit conversion . . . 718
Interpolation with Expressions . . . 719
Supported arithmetic operators . . . 719
Arithmetic Operator Examples . . . 720
Supported logical operators . . . 721
Supported logical functions . . . 721
AND . . . 721
OR . . . 722
NOT . . . 722
Supported scalar functions . . . 722
ABS . . . 723
ABS Examples . . . 724
CASTUOM . . . 725
IF . . . 726
IF Examples . . . 727
IFBAD . . . 728
IFBAD Examples . . . 729
LOG . . . 729
LOG Examples . . . 730
ROUND . . . 730

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ROUND Examples . . . 731
SIGN . . . 732
SIGN Examples . . . 732
SQRT . . . 733
SQRT Examples . . . 734
TIMESHIFT . . . 734
TIMESHIFT Examples . . . 735
TRUNCATE . . . 735
TRUNCATE Examples . . . 736
UOM (Unit of Measure conversion) . . . 737
UOM Examples . . . 738
Supported time-series functions . . . 738
AVERAGE . . . 739
AVERAGE Examples . . . 740
AVERAGES . . . 741
COUNT . . . 741
COUNT Examples . . . 742
COUNTALL . . . 742
COUNTALL Examples . . . 743
DURATION (Time in State) . . . 743
DURATION Examples . . . 744
IFLONGER . . . 746
IFSTALE . . . 747
METER . . . 748
METER Example . . . 749
MIN/MAX . . . 749
MIN/MAX Examples . . . 750
PREV/PREVGOOD . . . 751
PREV/PREVGOOD Examples . . . 752
TOTAL . . . 752
TOTAL Examples . . . 753
TOTALS . . . 754
Retrieval Styles for Trend . . . 755
Working with Retrieval Styles . . . 755
Location and Structure of Retrieval Styles . . . 755
Structure of the Retrieval Styles File . . . 756
Creating and Editing Retrieval Styles . . . 757
Retrieval Style XML Elements . . . 757
styleCollection XML Element . . . 757
retrievalStyle XML Element . . . 758
duration XML Element . . . 758
retrieval XML Element . . . 759
Using the Standard Retrieval Styles . . . 760
Retrieval Styles, Application Settings, and Tag Settings . . . 762
Historian Client Web User Guides . . . 762
Historian Client Web Classic . . . 762
Logging in to Historian Client Web . . . 763
Chart your data . . . 763
Search for data or content . . . 764

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Chart data with expressions . . . 765
Edit an expression . . . 768
Implicit unit conversion . . . 768
Interpolation with Expressions . . . 770
Selecting an array of streams . . . 770
Supported arithmetic operators . . . 772
Arithmetic Operator Examples . . . 773
Supported logical operators . . . 773
Supported logical functions . . . 774
AND . . . 774
OR . . . 774
NOT . . . 775
Supported scalar functions . . . 775
ABS . . . 776
ABS Examples . . . 776
CASTUOM . . . 778
IF . . . 778
IF Examples . . . 780
IFBAD . . . 781
IFBAD Examples . . . 782
IFLONGER . . . 782
IFLONGER Examples . . . 783
IFSHORTER . . . 784
IFSHORTER Examples . . . 785
IFSTALE . . . 786
IFSTALE Examples . . . 786
LOG . . . 787
LOG Examples . . . 787
ROUND . . . 788
ROUND Examples . . . 788
SIGN . . . 789
SIGN Examples . . . 790
SQRT . . . 791
SQRT Examples . . . 791
TIMESHIFT . . . 792
TIMESHIFT Examples . . . 792
TRUNCATE . . . 793
TRUNCATE Examples . . . 793
UOM (Unit of Measure conversion) . . . 795
UOM Examples . . . 795
Supported time-series functions . . . 795
AVERAGE . . . 796
AVERAGE Examples . . . 797
AVERAGES . . . 798
AVERAGES Examples . . . 798
COUNT . . . 799
COUNT Examples . . . 799
COUNTALL . . . 800
COUNTALL Examples . . . 800

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DURATION (Time in State) . . . 801
DURATION Examples . . . 802
METER . . . 804
METER Example . . . 805
MIN/MAX . . . 805
MIN/MAX Examples . . . 806
PREV/PREVGOOD . . . 807
PREV/PREVGOOD Examples . . . 808
TOTAL . . . 808
TOTAL Examples . . . 809
TOTALS . . . 810
TOTALS Examples . . . 811
Choose a tag category from Search Results . . . 811
About the All Tags category . . . 812
Choose tags to display in your chart . . . 813
View tag details . . . 814
Discover alias and location for a tag . . . 814
Remove a tag . . . 815
Choose a chart style . . . 816
Available chart types . . . 816
Navigate your charted data . . . 819
Scroll through data values (right and left) . . . 819
Pan, zoom, or expand details in a line chart . . . 819
Zoom in or out (line chart) . . . 821
About touch screen settings for Edge . . . 823
Pan left or right (line chart) . . . 823
Expand detail left or right (line chart) . . . 824
Apply chart options . . . 825
Change retrieval mode and interval for your chart . . . 825
Understand retrieval mode . . . 826
Understand interpolation type . . . 827
Select a preset timeframe . . . 828
Define a customized timeframe . . . 829
Stack or merge charted tag values . . . 830
Display or hide a tag . . . 832
Engineering Unit Conversion . . . 833
Add a limit line to a chart . . . 834
Adjust Y-axis autoscaling for line charts . . . 836
Track comments and events . . . 836
Add comments to a chart . . . 837
Edit your comment . . . 838
Delete your comment . . . 839
View events on a chart . . . 840
Work with dashboards . . . 843
Create a dashboard . . . 844
About keywords . . . 845
Add a chart to a dashboard . . . 845
Remove a chart from a dashboard . . . 846
Save a dashboard . . . 846

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Share a link to a dashboard . . . 847
Customize dashboard appearance . . . 847
Save and share content . . . 849
Save your current content . . . 850
Share content link with others . . . 851
Embed content . . . 852
Download content to a file . . . 853
Export Data to Excel . . . 855
Minimum Supported Versions of Microsoft Excel . . . 855
Configuring Internet Security Options for the Excel Add-In . . . 856
Registering and Installing the Excel Add-In . . . 858
Applying the Add-In to a Workbook . . . 860
Using the Excel Add-In . . . 861
About date/time formatting . . . 863
Searching for Tags . . . 864
Retrieving Tag Information . . . 868
Using the HISTORIAN.INFO Function . . . 869
Retrieving Single Values . . . 872
Using the HISTORIAN.SINGLEVALUE Function . . . 874
Retrieving Periodic Numeric Values . . . 877
Using the HISTORIAN.PERIODIC Function . . . 880
Retrieving Time in State . . . 882
Using the HISTORIAN.TIMEINSTATE Function . . . 886
Retrieving Detailed Values . . . 889
Using the HISTORIAN.DETAIL Function . . . 892
Using Smart Names for Tags . . . 894
Troubleshooting Issues using the Excel Add-In . . . 895
Issue: Network connectivity error when loading the add-in . . . 895
Migrate data from Historian Client Workbook to the Excel Add-in . . . 896
Use the Excel Migration tool . . . 896
Supported formulas for migration . . . 897
Live formula migration mapping details . . . 897
Tag details formula migration mapping details . . . 900
Manage content . . . 902
Open saved content . . . 903
Copy saved content . . . 903
Show or contact the content owner . . . 904
View alarms . . . 904
API References . . . 907
Historian Data REST API: Customize data retrieval . . . 907
Supported versions . . . 907
iHistory and Account Authentication . . . 908
Data retrieval . . . 909
Retrieval resources . . . 912
ProcessValues . . . 912
AnalogSummary . . . 919
Using SliceBy . . . 924
StateSummary . . . 925
Events . . . 929

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tags . . . 935
TagProperties . . . 940
TagPropertyValues . . . 941
TagGroups . . . 942
TagSuggest . . . 944
TagSearch . . . 945
TagExtendedProperties . . . 946
Summary (v1 only) . . . 947
Daily (v1 only) . . . 949
Hourly (v1 only) . . . 955
Minutely (v1 only) . . . 957
SystemParameters (on-premises only) . . . 958
Retrieval examples . . . 959
Retrieve data via browser query . . . 959
Retrieve data using Postman . . . 960
Excel query examples . . . 961
PowerBI query example . . . 964
Chart creation with URL-based query . . . 966
Content Scripting REST API . . . 969
PostData Resource . . . 969
LayoutInfo . . . 973
Layout . . . 974
LayoutProperties . . . 975
selectedOptions . . . 976
TagDetails . . . 980
Examples . . . 980
Use Chrome Developer tool to capture content . . . 980
Create content with minimum required properties . . . 985
Create content using all properties . . . 985
Create content with PowerShell . . . 986
Create a column chart . . . 987
Create a column chart with sliceBy support . . . 988
Create a composite chart . . . 989
Create a Gantt chart . . . 990
Create a line chart . . . 990
Create a rolling sum chart . . . 991
Create an XY plot . . . 992
Errors . . . 993
Historian Client Web Modern . . . 993
Use Historian Client Web . . . 993
Logging in to Historian Client Web . . . 993
About the Home page . . . 993
Search for Data or Content . . . 994
Search for tags from the Data panel . . . 995
Create and manage visuals . . . 997
Create visuals . . . 997
Select which data to include on visuals . . . 997
Choose a visual type . . . 998
Visualization content types . . . 998

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column Chart . . . 999
Data Grid . . . 1000
Line Charts . . . 1000
Pie Charts . . . 1000
Status Boards . . . 1001
Summary Chart . . . 1001
XY Plot . . . 1001
Interact with Charts . . . 1002
Adjust the range of time for your data set . . . 1002
Pan on line charts . . . 1002
Zoom on Line Charts or Stacked Line Charts . . . 1002
Change engineering units . . . 1003
Use Dual Cursors on Line Charts . . . 1005
Scale the Y-Axis . . . 1006
Manage existing content . . . 1008
Chart data with expressions . . . 1008
Add an expression . . . 1009
Edit the expression fields . . . 1011
Expression error indicator . . . 1014
Implicit unit conversion . . . 1015
Interpolation with Expressions . . . 1016
Supported arithmetic operators . . . 1017
Examples . . . 1017
Supported logical operators . . . 1018
Supported logical functions . . . 1018
AND . . . 1019
OR . . . 1019
NOT . . . 1019
Supported scalar functions . . . 1019
ABS . . . 1020
ABS Examples . . . 1021
CASTUOM . . . 1022
IF . . . 1023
IF Examples . . . 1024
IFBAD . . . 1025
IFBAD Examples . . . 1026
IFLONGER . . . 1026
IFLONGER Examples . . . 1028
IFSHORTER . . . 1028
IFSHORTER Examples . . . 1030
IFSTALE . . . 1030
IFSTALE Examples . . . 1031
LOG . . . 1031
LOG Examples . . . 1032
ROUND . . . 1032
ROUND Examples . . . 1033
SIGN . . . 1034
SIGN Examples . . . 1034
SQRT . . . 1035

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQRT Examples . . . 1036
TIMESHIFT . . . 1036
TIMESHIFT Examples . . . 1037
TRUNCATE . . . 1037
TRUNCATE Examples . . . 1038
UOM (Unit of Measure conversion) . . . 1039
UOM Examples . . . 1040
Supported time-series functions . . . 1040
AVERAGE . . . 1041
AVERAGE Examples . . . 1042
AVERAGES . . . 1043
AVERAGES Examples . . . 1043
COUNT . . . 1044
COUNT Examples . . . 1044
COUNTALL . . . 1045
COUNTALL Examples . . . 1045
DURATION (Time in State) . . . 1046
DURATION Examples . . . 1047
METER . . . 1049
METER Example . . . 1050
MIN/MAX . . . 1050
MIN/MAX Examples . . . 1051
PREV/PREVGOOD . . . 1052
PREV/PREVGOOD Examples . . . 1053
TOTAL . . . 1053
TOTAL Examples . . . 1054
TOTALS . . . 1055
TOTALS Examples . . . 1056
Work with dashboards . . . 1056
Create a dashboard . . . 1056
Add content to a dashboard . . . 1057
Save and share your dashboard . . . 1057
Manage existing content . . . 1060
View alarms . . . 1061
Use Historian Client Web APIs . . . 1063
Introduction to Historian Client Web APIs . . . 1063
Historian Data REST API: Customize data retrieval . . . 1063
Supported versions . . . 1064
Data Retrieval . . . 1065
Retrieval resources . . . 1068
ProcessValues . . . 1069
Using Startbound and Endbound Retrieval Modes . . . 1075
Returning multiple records with MaxRowCount . . . 1078
AnalogSummary . . . 1081
Using SliceBy . . . 1086
EventSummary . . . 1087
StateSummary . . . 1088
Events . . . 1092
Tags . . . 1098

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagProperties . . . 1103
TagPropertyValues . . . 1104
TagGroups . . . 1105
TagSuggest . . . 1106
TagSearch . . . 1108
TagExtendedProperties . . . 1109
Comments . . . 1110
Retrieval examples . . . 1112
Retrieve data via browser query . . . 1112
Retrieve data using Postman . . . 1113
Retrieve data using Excel . . . 1114
Retrieve data using PowerBI . . . 1117
Retrieval errors . . . 1119
Use a URL-based query to create a chart . . . 1120
Content Scripting REST API . . . 1123
PostData Resource . . . 1124
LayoutInfo . . . 1127
Layout . . . 1128
LayoutProperties . . . 1129
selectedOptions . . . 1130
TagDetails . . . 1134
Examples . . . 1135
Use Chrome Developer tool to capture content . . . 1135
Create content with minimum required properties . . . 1139
Create content using all properties . . . 1140
Create content with PowerShell . . . 1141
Create a column chart . . . 1142
Create a column chart with sliceBy support . . . 1142
Create a composite chart . . . 1144
Create a Gantt chart . . . 1144
Create a line chart . . . 1145
Create a map chart . . . 1145
Create a rolling sum chart . . . 1146
Create an XY plot . . . 1147
Errors . . . 1148

Historian Server Administrator Guides . . . 1149
Historian Administrator Guide . . . 1149
Welcome . . . 1149
AVEVA Historian Documentation Set . . . 1149
Getting Started . . . 1150
About AVEVA Historian Licensing . . . 1150
Viewing License Information . . . 1150
Refreshing the License Information . . . 1151
Registering AVEVA Historian Servers . . . 1151
Registering an AVEVA Historian . . . 1152
Editing Registration Properties . . . 1155
Deleting a Registered Historian . . . 1155

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Moving a Registered Server to a Different Group . . . 1155
About Administrative Tools . . . 1155
About the Operations Control Management Console . . . 1156
About the Management Console . . . 1157
About the Configuration Editor . . . 1158
Configuration Editor Toolbar Buttons . . . 1159
Determining the Configuration Editor Version . . . 1159
Operations Control Management Console Menu Commands . . . 1160
Microsoft SQL Server Management Studio . . . 1161
Registering a Server in Microsoft SQL Server Management Studio . . . 1162
Navigating in Microsoft SQL Server Management Studio . . . 1162
About AVEVA Historian Client Web . . . 1162
Starting AVEVA Historian Insight . . . 1163
Starting and Stopping AVEVA Historian . . . 1163
About the Startup Process . . . 1163
Starting the AVEVA Historian . . . 1163
About Connecting to SQL Server . . . 1164
Manually Starting SQL Server . . . 1164
Ports used by AVEVA Historian . . . 1165
Stopping the AVEVA Historian . . . 1166
Starting and Stopping Modules . . . 1167
Closing the Operations Control Management Console . . . 1168
Configuring General Startup Options . . . 1168
Shutting Down the Entire AVEVA Historian . . . 1168
Creating Server Groups . . . 1169
Adding a Server Group . . . 1170
Renaming a Server Group . . . 1170
Deleting a Server Group . . . 1170
About System-Level Consistency . . . 1170
Time Handling . . . 1171
System Parameters . . . 1171
System Messages . . . 1176
AVEVA Historian Processes . . . 1177
About System Driver and System Tags . . . 1179
Error Count Tags . . . 1179
Date Tags . . . 1179
Time Tags . . . 1180
Storage Space Tags . . . 1180
I/O Statistics Tags . . . 1180
System Monitoring Tags . . . 1181
Miscellaneous (Other) Tags . . . 1183
Classic Event Subsystem Tags . . . 1184
Replication Subsystem Tags . . . 1184
Performance Monitoring Tags . . . 1185
Defining Tags . . . 1188
About Tags . . . 1188
Tag Naming Conventions . . . 1188
Tag Properties (Tag Metadata) . . . 1189
Tag Configuration Versioning . . . 1190

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About Floating-Point Values . . . 1190
Viewing and Configuring Tags . . . 1190
Configuring Analog Tags . . . 1191
Adding an Analog Tag . . . 1191
Editing General Information for an Analog Tag . . . 1195
Editing Acquisition Information for a Tag . . . 1196
Editing Storage Information for an Analog Tag . . . 1198
Editing Limit Information for an Analog Tag . . . 1199
Configuring Limits . . . 1200
Configuring Context Definitions . . . 1201
Configuring Limit Names . . . 1201
Editing Summary Information for an Analog Tag . . . 1202
Editing Extended Properties for an Analog Tag . . . 1202
Configuring the Engineering Units Catalog . . . 1204
Adding a Catalog Unit . . . 1205
Editing a Catalog Unit . . . 1206
Example: Adding a Catalog Unit . . . 1208
Managing Dimensions, Unit Systems, and Catalog Unit Visibility . . . 1210
Adding or Editing a Dimension . . . 1212
Adding a Unit System . . . 1213
Configuring Engineering Units . . . 1215
Viewing Defined Engineering Units . . . 1215
Adding an Engineering Unit . . . 1216
Editing an Engineering Unit . . . 1217
Configuring Discrete Tags . . . 1217
Adding a Discrete Tag . . . 1218
Editing General Information for a Discrete Tag . . . 1218
Editing Storage Information for a Discrete Tag . . . 1219
Editing Extended Properties for a Discrete Tag . . . 1220
Configuring Message Pairs . . . 1222
Viewing the Current Message Pairs for a Server . . . 1222
Editing a Message Pair . . . 1222
Adding a Message Pair . . . 1223
Configuring String Tags . . . 1224
Adding a String Tag . . . 1224
Editing General Information for a String Tag . . . 1224
Editing Storage Information for a String Tag . . . 1225
Editing Extended Properties for a String Tag . . . 1226
Configuring Event Tags . . . 1228
Adding an Event Tag . . . 1228
Copying Tag Definitions . . . 1231
Deleting a Tag . . . 1231
Organizing Tags into Groups . . . 1232
Adding a Group . . . 1232
Renaming a Group . . . 1233
Adding a Tag to a Group . . . 1233
Deleting a Group or Tag Reference . . . 1234
Filtering Tags in the OCMC Details Pane . . . 1234
Applying a Filter . . . 1234

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Disabling or Removing a Filter . . . 1235
Importing and Exporting Tag Configurations . . . 1236
Importing an InTouch Data Dictionary . . . 1237
Before You Import . . . 1237
Determining Import Order . . . 1238
Duplicate Tags or Tag Addresses . . . 1238
Import Limitations for Topic Names . . . 1239
Editing Computer Names . . . 1239
Reimporting . . . 1239
Importing Information for DDE I/O Servers . . . 1240
Holding Database . . . 1240
Importing or Reimporting a Dictionary . . . 1240
Viewing Properties for an Imported InTouch Node . . . 1246
Viewing Tags Associated with an InTouch Node . . . 1247
Importing or Exporting Tag Information . . . 1248
Encoding Formats for Configuration Exports . . . 1249
Configuration Exporter Error Log . . . 1250
Exporting a Configuration . . . 1250
Importing a Configuration . . . 1255
Editing the Configuration Text File . . . 1258
Mode Indicators . . . 1259
Line Entries . . . 1260
Adding New Tag Extended Property Values . . . 1260
Configuring Data Acquisition . . . 1261
About the Data Acquisition Subsystem . . . 1261
Data Acquisition Components . . . 1261
I/O Server Addressing . . . 1262
I/O Server Redundancy . . . 1263
Redirecting I/O Servers to InTouch HMI Software . . . 1263
Time Synchronization for Data Acquisition . . . 1263
Viewing Data Acquisition Information . . . 1265
Configuring IDASs . . . 1266
About IDASs . . . 1266
IDAS Configuration . . . 1266
IDAS Data Processing . . . 1267
IDAS Security and Firewalls . . . 1267
IDAS Error Logging . . . 1269
IDAS Store-and-Forward Capability . . . 1269
IDAS Redundancy . . . 1270
IDAS Autonomous Startup . . . 1270
Configuring IDAS on a Remote Node . . . 1270
Troubleshooting IDAS Connections . . . 1272
Adding an IDAS . . . 1273
Editing General Information for an IDAS . . . 1274
Editing Advanced Information for an IDAS . . . 1276
Setting a remote IDAS to "Classic" . . . 1277
Deleting an IDAS . . . 1277
Configuring I/O Server Types . . . 1277
Adding an I/O Server Type . . . 1278

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Editing I/O Server Type Properties . . . 1278
Deleting an I/O Server Type . . . 1279
Configuring I/O Servers . . . 1279
Adding an I/O Server . . . 1279
Editing General Information for an I/O Server . . . 1280
Editing Storage Rule Information for an I/O Server . . . 1281
Deleting an I/O Server . . . 1283
Configuring Topics . . . 1283
Adding a Topic . . . 1283
Editing General Information for a Topic . . . 1284
Editing Storage Rules for a Topic . . . 1286
Deleting a Topic . . . 1288
Reinitializing I/O Topics . . . 1288
Managing Data Storage . . . 1289
About Data Storage . . . 1289
About Data Storage Subsystem Processes . . . 1289
Integration with Microsoft SQL Server . . . 1290
About Delta Storage Mode . . . 1290
Time and Value Deadbands for Delta Storage . . . 1291
Swinging Door Deadband for Delta Storage . . . 1292
Benefits of the Swinging Door Deadband . . . 1292
Additional Options that Affect the Swinging Door Deadband . . . 1294
Swinging Door Deadband Examples . . . 1295
Swinging Door Deadband: Rate Only . . . 1295
Swinging Door Deadband: Rate and Value . . . 1297
Swinging Door Deadband: Rate, Value, and Deadband Override Period . . . 1297
Managing the AVEVA Historian Runtime Database . . . 1298
Changing the Properties for the Runtime Database . . . 1299
Managing the Runtime Database . . . 1300
Backing Up the Runtime Database . . . 1300
Backing Up the Database . . . 1300
Restoring the Database . . . 1302
Managing a Runtime Database Object . . . 1303
Space Management for Event and Summary History . . . 1304
Managing Partitions and History Blocks . . . 1304
Storage Partition Locations . . . 1304
Circular Storage . . . 1305
Alternate Storage . . . 1305
Permanent Storage . . . 1305
Buffer Storage . . . 1306
About the Auto-Summary Partition . . . 1306
About Block Gaps . . . 1306
Viewing Storage and Auto-Summary Partitions . . . 1307
Editing Storage Partition Properties . . . 1308
Editing Storage Location Properties . . . 1309
Viewing History Blocks . . . 1310
History Block Notation and Creation . . . 1311
Automatic Deletion of History Blocks . . . 1312
Backing Up History Blocks . . . 1313

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About VSS-Aware Backups . . . 1313
Adding Replication or Auto-Summary Values for a Defined Timeframe . . . 1313
Adding History Blocks from Prior Versions to the System . . . 1314
Importing, Inserting, or Updating History Data . . . 1315
Ways to Acquire History Data . . . 1315
Guidelines for Importing, Inserting, and Updating History Data . . . 1315
Importing History Data . . . 1316
Importing Data from an InTouch History File . . . 1318
Importing Data from CSV Files . . . 1319
Predefined CSV File Import Folders . . . 1320
About Normal CSV File Imports . . . 1320
About Fast Load CSV File Imports . . . 1321
General File Format for a CSV Import . . . 1321
Formatting the CSV File for a Normal Import . . . 1321
Example CSV Files for a Normal Import . . . 1322
Formatting the CSV File for a Fast Load Import . . . 1323
Example CSV Files for a Fast Load Import . . . 1324
Handling of NULL Values in CSV Files . . . 1324
Copying a CSV File into an Import Folder . . . 1325
Running the Historian Data Importer from a Command Prompt . . . 1325
Inserting or Updating Data with Transact-SQL Statements . . . 1326
INSERT ... VALUES Syntax . . . 1326
Using the wwVersion Parameter for INSERTs . . . 1328
Inserting Real-time Original Data . . . 1328
Inserting Original Non-Streamed Data . . . 1328
Inserting Latest Revision Data . . . 1329
UPDATE Syntax . . . 1329
Renaming Tags . . . 1330
About Historian Tag Ownership . . . 1331
Preparing to Rename Application Server Tags . . . 1331
Renaming Tags Using the Tag Rename Utility . . . 1332
Updating Replicated Data . . . 1334
Managing and Configuring Replication . . . 1334
About Replication . . . 1334
Replication Schedules . . . 1335
Replication Schedules and Daylight Savings Time . . . 1335
Replication Groups . . . 1336
How Replication is Handled for Different Types of Data . . . 1337
Streaming Replication . . . 1338
Queued Replication . . . 1338
Tag Configuration Synchronization between Tiered Historians . . . 1339
Replication Run-time Operations . . . 1339
Replication Latency . . . 1339
Replication Delay for "Old" Data . . . 1339
Continuous Operation . . . 1340
Overflow Protection . . . 1340
Security for Data Replication . . . 1340
Adding a Replication Server . . . 1340
Adding AVEVA Historian as a Replication Server . . . 1341

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding AVEVA Insight as a Replication Server . . . 1344
Adding AVEVA Data Hub as a Replication Server . . . 1347
Adding AVEVA PI Server as a Replication Server . . . 1350
Prepare your PI Server for Receiving Replication Data . . . 1350
Configure Replication to AVEVA PI Server . . . 1352
Trusting a Certificate . . . 1354
Data Buffer Configuration . . . 1356
Special Considerations for Tag Names . . . 1357
Specifying Naming Schemes for Replication . . . 1358
Editing Replication Server Properties . . . 1361
Deleting a Replication Server . . . 1362
Configuring Tags to Be Replicated . . . 1363
Adding a Single Tag for Simple Replication . . . 1363
Adding Multiple Tags for Simple Replication . . . 1364
Editing Simple Replication Tag Properties . . . 1365
Deleting a Simple Replication Tag . . . 1366
Adding a Replication Schedule . . . 1367
Editing Replication Schedule Properties . . . 1368
Deleting a Replication Schedule . . . 1369
Adding a Replication Group . . . 1369
Editing Replication Group Properties . . . 1370
Deleting a Replication Group . . . 1371
Creating a Replication Group for Multiple Servers . . . 1371
About Summary Replication . . . 1372
About Analog Summary Replication . . . 1372
About State Summary Replication . . . 1373
Adding a Summary Tag . . . 1374
Finding Source Tags . . . 1375
Adding Multiple Summary Tags . . . 1377
Creating a Summary Tag Quickly Using Default Settings . . . 1378
Editing Summary Tag Properties . . . 1379
Deleting Replication for a Summary Tag . . . 1380
Adding Replication or Auto-Summary Values for a Defined Timeframe . . . 1381
Viewing Source Details for a Summary Tag . . . 1382
Viewing the List of Associated Replicated Tags for a Tag . . . 1383
Managing Security . . . 1384
About Security . . . 1384
Windows Operating System Security . . . 1385
Default Windows User Account for AVEVA Historian Services . . . 1385
SQL Server Security . . . 1386
Authentication . . . 1386
Default Windows Security Groups . . . 1386
AVEVA Historian Default Logins . . . 1387
Database Authorization . . . 1388
AVEVA Historian Default Users and Roles . . . 1388
Default SQL Server Login for AVEVA Historian Services . . . 1390
Management Console Security . . . 1390
Verifying the Authentication Mode for a SQL Server . . . 1390
Managing Logins . . . 1391

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing Login Properties . . . 1392
Adding a Login . . . 1392
Local Times and System Times . . . 1395
Managing Users and Roles using the Configurator . . . 1396
Viewing All Users and Role Assignments . . . 1396
Adding Users and Assigning Roles . . . 1397
Managing Users and Roles using SQL Server Management Studio . . . 1400
Viewing All Users and Roles for a Database . . . 1400
Adding a New Database User . . . 1400
Adding a User to a Role . . . 1401
Managing Permissions . . . 1402
Setting Object Permissions . . . 1402
Setting Statement Permissions . . . 1404
Managing Passwords . . . 1405
Adding a User to a Windows Operating System Group . . . 1406
Viewing or Changing System-Wide Properties . . . 1408
About the Configuration Subsystem . . . 1408
Configuration Subsystem Components . . . 1409
About the Runtime and Holding Databases . . . 1409
Runtime Database . . . 1409
Holding Database . . . 1410
About the Configuration Service . . . 1410
Dynamic Configuration . . . 1410
Effects of Configuration Changes on the System . . . 1411
Cases in Which Configuration Changes Are Not Committed . . . 1412
Viewing Properties for System Parameters . . . 1412
Editing System Parameters . . . 1412
Adding a System Parameter . . . 1413
Committing Configuration Changes . . . 1413
Tracking Modifications . . . 1415
About Modification Tracking for Configuration Changes . . . 1415
About Modification Tracking for Historical Data Changes . . . 1416
Turning Modification Tracking On/Off . . . 1417
Viewing Database Modifications . . . 1417
Viewing the Runtime Database Report . . . 1419
Using a Redundant Historian . . . 1420
Changing the Default Network Protocol . . . 1421
Configuring a Custom TCP Port . . . 1421
Historian Client Web Customization . . . 1422
White Labeling . . . 1423
Configuring Customizable White Label Settings . . . 1423
CORS Whitelisting . . . 1424
Configuring the CORS Whitelist . . . 1424
Using HTTPS Instead of HTTP for Historian Client, Historian Client Web, and REST APIs . . . 1425
Enabling Trust for a Self-Signed Certificate . . . 1427
Acquiring a Copy of the Self-Signed Certificate . . . 1428
Trusting a Self-Signed Certificate . . . 1431
Monitoring the System . . . 1434
Monitoring the General Status of AVEVA Historian . . . 1434

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing the Current System Status . . . 1434
Resetting Error Counts . . . 1436
Viewing the Status of System Modules . . . 1436
Viewing System Status Messages . . . 1437
Viewing Status Information . . . 1438
Monitoring Data Acquisition . . . 1438
Monitoring Replications . . . 1439
Monitoring Client Connections . . . 1440
Monitoring System Messages . . . 1441
Viewing Errors in the Windows Event Viewer . . . 1441
Monitoring System Tags from within InTouch HMI Software . . . 1442
Using Windows Performance Logs and Alerts . . . 1443
Browsing the ArchestrA Model View Using Historian Clients . . . 1443
Model View Representation in the Historian Namespace . . . 1443
Model View Replication to the Historian . . . 1444
Replication Configuration using the IDE . . . 1445
Configuring Replication for a WinPlatform . . . 1445
Configuring Replication for an AppEngine . . . 1446
Enabling Replication at Runtime . . . 1447
Viewing Historized Attributes in the AVEVA Historian Configuration Editor . . . 1447
Browsing the Model Hierarchy in a Historian Client . . . 1448
Legacy Features . . . 1450
Classic Storage Subsystem . . . 1451
Memory Management for Retrieval of Classic Storage Data . . . 1451
About the Real-Time Data Window . . . 1452
Determining If the Real-Time Window Is Configured Appropriately for a Swinging Door Deadband . . . 1453
Classic Event Subsystem . . . 1454
Classic Event Subsystem Components . . . 1454
Uses for the Classic Event Subsystem . . . 1455
Classic Event Subsystem Features and Benefits . . . 1456
Classic Event Subsystem Performance Factors . . . 1456
Event Tags . . . 1457
Event Detectors . . . 1457
SQL-Based Detectors . . . 1458
Generic SQL Detectors . . . 1458
Specific Value Detectors . . . 1458
Time Intervals for SQL-Based Detectors . . . 1458
Schedule Detectors . . . 1459
External Detectors . . . 1460
Event Actions . . . 1460
Generic SQL Actions . . . 1460
Snapshot Actions . . . 1460
E-mail Actions . . . 1461
Deadband Actions . . . 1461
Summary Actions . . . 1461
Event Action Priorities . . . 1462
Classic Event Subsystem Resource Management . . . 1462
Detector Thread Pooling . . . 1463

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Action Thread Pooling . . . 1464
Classic Event Subsystem Database Connections . . . 1464
Handling of Event Overloads and Failed Queries . . . 1465
Classic Event Subsystem Variables . . . 1465
Classic Event Subsystem Tags . . . 1466
Configuring Classic Events . . . 1467
Accessing Event Information . . . 1467
Adding an Event Tag . . . 1467
Editing General Information for an Event Tag . . . 1470
Configuring Detectors . . . 1471
Configuring a Specific Value Detector . . . 1471
Configuring a Schedule Detector . . . 1472
Configuring a Generic SQL Detector . . . 1473
Configuring an External Detector . . . 1474
Configuring Actions . . . 1474
Configuring a Deadband Action . . . 1474
Configuring a Snapshot Action . . . 1475
Configuring a Generic SQL Action . . . 1476
Generic SQL Action Template for Executing a Command . . . 1476
Generic SQL Action Templates for E-mail . . . 1477
Configuring an E-mail Action . . . 1478
Setting Up Microsoft SQL Server . . . 1478
Determining the Microsoft Outlook Mail Profile . . . 1479
Setting Up SQL Mail in SQL Management Studio . . . 1479
Configuring the E-mail Action . . . 1480
Configuring a Summary Action . . . 1480
Adding a Summary Operation . . . 1481
Assigning a Tag to a Summary Operation . . . 1482
Using the Tag Finder . . . 1484
Using the Form Query Tab . . . 1484
Using the SQL Query Tab . . . 1485
Retrieving Logged Event Data . . . 1485
Viewing Summary Information . . . 1486
Viewing Summary Tag Properties . . . 1487
Viewing Data for a Summary Tag . . . 1487
Viewing History for a Summary Operation . . . 1488
Using ActiveEvent . . . 1489
Configuring Security Attributes for ActiveEvent . . . 1490
ActiveEvent Methods . . . 1492
AddEventTag() . . . 1493
InitializeEx() . . . 1494
InvokeEventAtTimeEx() . . . 1495
InvokeEventEx() . . . 1495
IsConnected . . . 1496
RemoveEventTag() . . . 1496
Scripting Example: Triggering Events within an InTouch Application . . . 1497
Scripting Example: Triggering Multiple Events within Visual Basic . . . 1497
History Block Storage for Alarms and Events . . . 1498
A2ALMDB Database . . . 1498

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring Purge or Archive Settings . . . 1499
Configuring the Database Connection . . . 1499
Configuring How Much Data to Purge from the Server . . . 1500
Configuring the Archive of Purged Data . . . 1501
Configuring Log File Settings . . . 1502
Manually Purging and Archiving the Database . . . 1503
Setting a Schedule for Automatic Purging . . . 1505
Restoring the Alarm Database . . . 1505
Configuring the Database Connection . . . 1506
Configuring Which Files to Restore . . . 1507
Starting a Database Restore Operation . . . 1508
Migrating Data from the A2ALMDB Database to History Blocks . . . 1509
Historian Concepts . . . 1510
Welcome to AVEVA Historian . . . 1510
Process data: About tags and values . . . 1511
Three-dimensional data: Value, time, and quality (VTQ) . . . 1512
Types of tags . . . 1514
Data acquisition: Getting data into Historian . . . 1516
Sources of data . . . 1516
About AVEVA Historian data security . . . 1518
Store-and-forward safeguards . . . 1519
Data categories . . . 1519
Data storage: Preserving huge amounts of data over time . . . 1520
Storage modes . . . 1521
History blocks and partitions . . . 1523
Auto-summarization . . . 1523
Data retrieval: Transforming data into information . . . 1524
Retrieval modes . . . 1525
Data query tools: Accessing your data . . . 1528
Data replication: Delivering information to people who need it . . . 1530
Replication tiers . . . 1532
Historian Database Guide . . . 1534
Welcome . . . 1534
Table Categories . . . 1534
History Tables and Views . . . 1534
History Table Format . . . 1535
"Wide" History Table Format . . . 1537
"Live" Table Format . . . 1538
Auto-Summary Replication Tables . . . 1538
Tag Definition Tables . . . 1539
Tag Definition Specialty Tables . . . 1539
I/O Data Acquisition Tables . . . 1539
Insight Client Content Tables . . . 1540
InTouch Node Detail Tables . . . 1540
Legacy Event and Summary Tables . . . 1540
Manual Data Tables . . . 1541
Modification Tracking Tables . . . 1541
Namespace and User-SpecificTables . . . 1541
Standard Replication Tables . . . 1542

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Storage Tables . . . 1542
System Configuration Tables . . . 1543
ArchestrA Browsing Tables . . . 1543
Tables . . . 1544
_AutoIntervalReplicationSchedule . . . 1544
_AutoReplicationGroup . . . 1544
_AutoReplicationRule . . . 1545
_AutoReplicationSchedule . . . 1546
_AutoReplicationServer . . . 1547
_AutoReplicationTagEntity . . . 1549
_AutoTag . . . 1551
_AutoTagHistory . . . 1558
_DeletedReplicationTagExtendedProperty . . . 1565
_EventTag . . . 1565
_IODriver . . . 1568
_IOServer . . . 1573
_ReplicationTagExtendedProperty . . . 1574
_Tag . . . 1575
_TagExtendedProperty . . . 1582
_Topic . . . 1583
aaAreaData . . . 1584
aaAreaXML . . . 1585
aaAttributeData . . . 1585
aaAttributeDataPending . . . 1585
aaObjectData . . . 1586
aaObjectDataPending . . . 1586
ActionType . . . 1587
AnalogSnapshot . . . 1588
AnalogSummaryHistory (INSQL.Runtime.dbo.AnalogSummaryHistory) . . . 1588
AttributeType . . . 1593
CalcType . . . 1594
ChangeNotification . . . 1594
ChannelStatus . . . 1594
ChartConfiguration . . . 1595
ChartConfigurationAuditLog . . . 1596
ChartConfigurationKeyword . . . 1596
ChartConfigurationProperty . . . 1596
ChartConfigurationStatistics . . . 1597
ChartConfigurationTag . . . 1597
Comments . . . 1598
CommentsType . . . 1599
ConfigStatusPending . . . 1599
ConfigStatusSnapshot . . . 1600
Context . . . 1600
CurrentEditor . . . 1601
CustomReplicationSchedule . . . 1602
DashboardConfiguration . . . 1603
DeletedReplicationTagEntity . . . 1603
DeletedTag . . . 1603

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DetectorType . . . 1603
Deviation . . . 1604
DiscreteSnapshot . . . 1606
EngineeringUnit . . . 1606
EngineeringUnitCatalog . . . 1607
EngineeringUnitDimension . . . 1609
EngineeringUnitSystem . . . 1610
ErrorLog . . . 1610
EventHistory . . . 1611
EventTagPendingDelete . . . 1612
Frequency . . . 1612
GroupTagList . . . 1612
History (INSQL.Runtime.dbo.History) . . . 1613
HistoryBlock (INSQL.Runtime.dbo.HistoryBlock) . . . 1620
HistorianSysObjects . . . 1622
IntervalReplicationSchedule . . . 1623
InTouchNode . . . 1623
InTouchSpecific . . . 1625
IOServerType . . . 1625
Limit . . . 1626
LimitName . . . 1627
Live (INSQL.Runtime.dbo.Live) . . . 1627
LocalizedText . . . 1630
ManualAnalogHistory . . . 1630
ManualDiscreteHistory . . . 1631
ManualStringHistory . . . 1631
Message . . . 1632
ModLogColumn . . . 1632
ModLogTable . . . 1633
NameSpaceIcons . . . 1635
OPCQualityMap . . . 1635
PrivateGroupTag . . . 1635
PrivateNameSpace . . . 1636
PublicGroupTag . . . 1637
PublicNameSpace . . . 1637
QualityMap . . . 1638
RateOfChange . . . 1639
ReplicationGroup . . . 1639
ReplicationRule . . . 1641
ReplicationSchedule . . . 1641
ReplicationScheduleType . . . 1642
ReplicationServer . . . 1642
ReplicationShard . . . 1644
ReplicationSyncRequest . . . 1645
ReplicationSyncRequestPending . . . 1646
ReplicationTagEntity . . . 1646
ReplicationType . . . 1648
SearchMessageSyncRequest . . . 1649
ServerList . . . 1649

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ShardAssignmentRule . . . 1650
ShareMode . . . 1650
SnapshotDetail . . . 1651
SnapshotTag . . . 1652
SQLTemplate . . . 1652
StateSummaryHistory (INSQL.Runtime.dbo.StateSummaryHistory) . . . 1653
StateWideHistory (INSQL.Runtime.dbo.StateWideHistory) . . . 1657
StorageLocation . . . 1661
StorageShard . . . 1663
StorageType . . . 1665
StringSnapshot . . . 1665
StructureAttributes . . . 1666
StructureType . . . 1666
SummaryData . . . 1666
SummaryHistory . . . 1667
SummaryOperation . . . 1669
SummaryTagList . . . 1670
SystemParameter . . . 1671
TagExtendedPropertyName . . . 1671
TagGroup . . . 1672
TagHistory . . . 1673
TagRef . . . 1681
TagType . . . 1681
TimeDetectorDetail . . . 1682
TimeDetectorDetailPendingDelete . . . 1683
TimeUnit . . . 1683
TimeZone . . . 1683
TopicImportInfo . . . 1683
UserDetail . . . 1685
WideHistory (INSQL.Runtime.dbo.WideHistory) . . . 1686
WideTableDictionary . . . 1692
Views . . . 1693
History Table Views . . . 1693
Annotation . . . 1693
Events . . . 1694
IODriver . . . 1700
IOServer . . . 1704
ReplicationSyncRequestInfo . . . 1706
ReplicationTagExtendedPropertyInfo . . . 1710
Tag . . . 1711
TagExtendedPropertyInfo . . . 1720
TagExtendedPropertyNameInfo . . . 1721
Topic . . . 1721
v_EventSnapshot . . . 1723
v_EventStringSnapshot . . . 1723
v_ModTracking . . . 1724
v_SnapshotData . . . 1726
Stored Procedures . . . 1727
Stored Procedures . . . 1727

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaActionStringSelect . . . 1727
aaAddAnalogSummaryTag . . . 1727
aaAddReplicationGroup . . . 1732
aaAddReplicationRule . . . 1733
aaAddReplicationSchedule . . . 1733
aaAddReplicationServer . . . 1735
aaAddReplicationTagEntity . . . 1736
aaAddReplicationTagExtendedProperty . . . 1738
aaAddStateSummaryTag . . . 1738
aaAddStorageLocation . . . 1741
aaAddStorageShard . . . 1742
aaAddStorageShardAssignmentRule . . . 1743
aaAddStructureTag . . . 1744
aaAnalogDetail . . . 1748
aaAnalogTagDelete . . . 1748
aaAnalogTagInsert . . . 1749
aaAnalogTagSelect . . . 1755
aaAnalogTagUpdate . . . 1755
aaAnnotationDelete . . . 1756
aaAnnotationInsert . . . 1757
aaAnnotationRetrieve . . . 1757
aaAnnotationSelect . . . 1758
aaAnnotationUpdate . . . 1758
aaArchestrANSClear . . . 1759
aaCheckChartConfigurationNameExists . . . 1759
aaCleanupAfterCommit . . . 1759
aaCleanupSystemNotRunning . . . 1760
aaClearDeletedTags . . . 1760
aaClearDeletedReplicationTagEntities . . . 1761
aaCommitAllowed . . . 1761
aaCommitChanges . . . 1761
aaCommitChangesAtStartup . . . 1762
aaContextDelete . . . 1762
aaContextInsert . . . 1763
aaContextSelect . . . 1763
aaContextUpdate . . . 1764
CreateReplicationServerDefaultGroups . . . 1764
CreateReplicationServerSystemTags . . . 1764
aaDBChangesPending . . . 1765
aaDBConfig . . . 1765
aaDeleteChartConfiguration . . . 1765
aaDeleteComment . . . 1766
aaDeleteOlderEvents . . . 1766
aaDeleteOlderSummaries . . . 1767
aaDeleteReplicationGroup . . . 1767
aaDeleteReplicationSchedule . . . 1768
aaDeleteReplicationServer . . . 1768
DeleteReplicationServerSystemTags . . . 1769
aaDeleteReplicationTagEntity . . . 1769

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaDeleteTag . . . 1770
aaDeleteTagExtendedProperty . . . 1770
aaDetectorStringSelect . . . 1770
aaDiscreteDetail . . . 1771
aaDiscreteTagDelete . . . 1771
aaDiscreteTagInsert . . . 1772
aaDiscreteTagSelect . . . 1776
aaDiscreteTagUpdate . . . 1776
aaEngineeringUnitDelete . . . 1777
aaEngineeringUnitInsert . . . 1778
aaEngineeringUnitSelect . . . 1779
aaEngineeringUnitUpdate . . . 1779
aaEventDetection . . . 1780
aaEventHistoryInsert . . . 1781
aaEventHistorySelect . . . 1782
aaEventSnapshotInsert . . . 1782
aaEventSnapshotSelect . . . 1783
aaEventTagDelete . . . 1784
aaEventTagDetail . . . 1784
aaEventTagInsert . . . 1785
aaEventTagSelect . . . 1788
aaEventTagSelectAll . . . 1789
aaEventTagSelectDeleted . . . 1789
aaEventTagSelectDisabled . . . 1789
aaEventTagSelectInserted . . . 1790
aaEventTagSelectUpdated . . . 1790
aaEventTagUpdate . . . 1790
aaGetAnalogSummaryTags . . . 1791
aaGetChangeNotification . . . 1791
aaGetChartConfigurationLayout . . . 1792
aaGetChartConfigurations . . . 1792
aaGetChartConfigurationsForDashboard . . . 1792
aaGetChartConfigurationsForKeywords . . . 1793
aaGetDbRevision . . . 1793
aaGetDeletedReplicationTagEntities . . . 1793
aaGetDeletedTags . . . 1794
aaGetHistorianPartners . . . 1794
aaGetLastTagKey . . . 1794
aaGetReplicationGroups . . . 1795
aaGetReplicationNamingParameters . . . 1796
aaGetReplicationRule . . . 1797
aaGetReplicationSchedules . . . 1797
aaGetReplicationServers . . . 1798
aaGetReplicationShard . . . 1798
aaGetReplicationTagEntities . . . 1799
aaGetReplicationTags . . . 1800
aaGetRowCount . . . 1800
aaGetStateSummaryTags . . . 1801
aaGetStorageShard . . . 1801

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaGetStorageShardAssignmentRule . . . 1802
aaGetTagExtendedProperties . . . 1802
aaGetUserKey . . . 1802
aaHistorianConfigNSExpand . . . 1802
aaHistorianNSExpand . . . 1803
aaHistorianStatusSelect . . . 1803
aaHistorianStatusSet . . . 1804
aaHistoryBlockSelect . . . 1805
aaInTouchNodeTagList . . . 1806
aaIODriverDelete . . . 1806
aaIODriverInsert . . . 1807
aaIODriverSelect . . . 1810
aaIODriverUpdate . . . 1810
aaIOServerDelete . . . 1811
aaIOServerInsert . . . 1811
aaIOServerSelect . . . 1813
aaIOServerTypeDelete . . . 1813
aaIOServerTypeInsert . . . 1814
aaIOServerTypeSelect . . . 1815
aaIOServerTypeUpdate . . . 1815
aaIOServerUpdate . . . 1815
aaLimitDelete . . . 1816
aaLimitInsert . . . 1816
aaLimitNameDelete . . . 1817
aaLimitNameInsert . . . 1818
aaLimitNameSelect . . . 1818
aaLimitNameUpdate . . . 1819
aaLimitSelect . . . 1819
aaLimitUpdate . . . 1819
aaMessageDelete . . . 1820
aaMessageInsert . . . 1820
aaMessageSelect . . . 1821
aaMessageUpdate . . . 1821
aaModLogStatus . . . 1822
aaNotifyChange . . . 1822
aaPrivateNSAddGroup . . . 1823
aaPrivateNSAddLeaf . . . 1823
aaPrivateNSDeleteGroup . . . 1824
aaPrivateNSDeleteLeaf . . . 1824
aaPrivateNSExpand . . . 1825
aaPrivateNSSelect . . . 1825
aaPrivateNSUpdateGroup . . . 1826
aaPublicNSAddGroup . . . 1826
aaPublicNSAddLeaf . . . 1827
aaPublicNSDeleteGroup . . . 1828
aaPublicNSDeleteLeaf . . . 1828
aaPublicNSExpand . . . 1829
aaPublicNSSelect . . . 1829
aaPublicNSUpdateGroup . . . 1830

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaRedirectToInTouch . . . 1831
aaSaveChartConfiguration . . . 1831
aaSearchMessageInsert . . . 1832
aaSetCalculatedAISamples . . . 1833
aaSetServerTimeStamp . . . 1833
aaSetStorageRule . . . 1834
aaSetTagStorage . . . 1837
aaSnapshotDetailSelect . . . 1838
aaSnapshotDetailUpdate . . . 1838
aaSnapToSummary . . . 1839
aaSpaceManager . . . 1840
aaStorageLocationSelect . . . 1840
aaStorageLocationUpdate . . . 1841
aaStringDetail . . . 1842
aaStringTagDelete . . . 1843
aaStringTagInsert . . . 1843
aaStringTagSelect . . . 1848
aaStringTagUpdate . . . 1848
aaSummaryActionInsert . . . 1849
aaSummaryDetail . . . 1849
aaSummaryOperationDelete . . . 1850
aaSummaryOperationInsert . . . 1851
aaSummaryOperationSelect . . . 1852
aaSummaryOperationUpdate . . . 1852
aaSummaryTagListDelete . . . 1853
aaSummaryTagListInsert . . . 1853
aaSummaryTagListSelect . . . 1854
aaSummaryTagListUpdate . . . 1855
aaSystemConfigNSExpand . . . 1855
aaSystemNSExpand . . . 1856
aaSystemNSExpand2 . . . 1857
aaSystemParameterSelect . . . 1857
aaSystemParameterUpdate . . . 1858
aaTagConfig . . . 1858
aaTagConfigModified . . . 1859
aaTagConfigSelect . . . 1859
aaTagInfo . . . 1860
aaTagType . . . 1860
aaTimeDetectorDetailInsert . . . 1860
aaTimeDetectorDetailSelect . . . 1862
aaTimeDetectorDetailUpdate . . . 1862
aaTopicDelete . . . 1862
aaTopicInsert . . . 1863
aaTopicSelect . . . 1864
aaTopicUpdate . . . 1864
aaUpdateCalculatedAISamples . . . 1865
aaUpdateChartConfigurationStatistics . . . 1866
aaUserAccessLevelSelect . . . 1866
aaUserDetailUpdate . . . 1867

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Stored Procedures for Internal Use . . . 1867
Creating Stored Procedures . . . 1867
User-Defined Functions . . . 1867
faaCheckLicenseViolation . . . 1868
faaContainedName . . . 1868
faaGetHierarchicalAttributeNames . . . 1868
faaGetHistorianTagNames . . . 1869
faaLicensedTagDetails . . . 1869
faaLicensedTagTotal . . . 1870
faaObjectTagName . . . 1870
faaTagsInLicenseViolation . . . 1871
faaTZgetdate . . . 1872
faaUser_ID . . . 1872
fww_GetLocalizedText . . . 1873
Backward Compatibility Entities . . . 1874
Backward Compatibility Views . . . 1874
History Table Views (Backward Compatible) . . . 1874
Tag Table Views . . . 1875
AnalogSummaryTag . . . 1875
AnalogTag . . . 1875
DiscreteTag . . . 1878
EventTag . . . 1878
ReplicationTag . . . 1881
StringTag . . . 1881
StructureTag . . . 1882
Alarm and Event Views (Backward Compatible) . . . 1882
Summary Views . . . 1883
v_SummaryData . . . 1884
NamedSystemParameter . . . 1885
SystemNameSpace . . . 1885
InSQLSysObjects . . . 1886
v_ErrorLog . . . 1887
Backward Compatibility Tables . . . 1888
AnalogHistory (INSQL.Runtime.dbo.AnalogHistory) . . . 1888
AnalogLive (INSQL.Runtime.dbo.AnalogLive) . . . 1889
AnalogWideHistory . . . 1890
DiscreteHistory (INSQL.Runtime.dbo.DiscreteHistory) . . . 1891
DiscreteLive (INSQL.Runtime.dbo.DiscreteLive) . . . 1892
DiscreteWideHistory . . . 1892
GroupTagList . . . 1893
ManualAnalogHistory . . . 1893
ManualDiscreteHistory . . . 1894
ManualStringHistory . . . 1895
NameSpaceIcons . . . 1895
StringHistory (INSQL.Runtime.dbo.StringHistory) . . . 1896
StringLive (INSQL.Runtime.dbo.StringLive) . . . 1897
StringWideHistory . . . 1897
TagGroup . . . 1898
WideTableDictionary . . . 1899

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Renamed Tables . . . 1899
Backward Compatibility Stored Procedures . . . 1899
aaAnalogDetail . . . 1900
aaDiscreteDetail . . . 1900
aaStringDetail . . . 1901
ww_CheckClientVersion . . . 1901
ww_CheckWhichDb . . . 1902
ww_dbCheck . . . 1902
ww_DBConfig . . . 1903
ww_LoadInSQLProcedureBody . . . 1903
ww_MDASAnalogTagInsert . . . 1903
ww_MDASAnalogTagUpdate . . . 1903
ww_MDASDiscreteTagInsert . . . 1904
ww_MDASDiscreteTagUpdate . . . 1904
ww_MDASStringTagInsert . . . 1904
ww_MDASStringTagUpdate . . . 1904
Renamed Stored Procedures . . . 1904
Extended Stored Procedure Arguments . . . 1910
Literal Date Expressions . . . 1911
GetDate() Expressions . . . 1911
DateAdd(...) Expressions . . . 1911
Backward Compatibility Functions . . . 1912
Historian Retrieval Guide . . . 1913
Welcome . . . 1913
AVEVA Historian Documentation Set . . . 1913
About Data Retrieval . . . 1914
Data Retrieval Subsystem Features . . . 1914
History Blocks: A SQL Server Remote Data Source . . . 1915
Retrieval subsystem . . . 1915
About the AVEVA Historian OLE DB Provider . . . 1915
Extension Tables for History Data . . . 1916
Linking the AVEVA Historian OLE DB Provider to the Microsoft SQL Server . . . 1917
AVEVA Historian I/O Server . . . 1918
AVEVA Historian I/O Server (classic) . . . 1918
Using SELECT to Retrieve Data . . . 1919
Using the Four-Part Naming Convention . . . 1919
Using an AVEVA Historian OLE DB Provider View . . . 1920
Using the OPENQUERY Function . . . 1921
Using the OPENROWSET Function . . . 1921
Supported Syntax Options . . . 1922
Unsupported or Limited Syntax Options . . . 1922
No Notion of Client Context . . . 1922
Limitations on Wide Tables . . . 1923
LIKE Clause Limitations . . . 1923
IN Clause Limitations . . . 1923
OR Clause Limitations . . . 1923
Using Joins within an OPENQUERY Function . . . 1923
Using Complicated Joins . . . 1924
Using a Sub-SELECT with a SQL Server Table and an Extension Table . . . 1924

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

WHERE Clause Anomalies . . . 1925
CONVERT Function Limitations . . . 1925
SQL Server Optimization of Complex Queries . . . 1926
Using Columns of a Variant Type with Functions . . . 1927
Using StartDateTime in the Query Criteria . . . 1927
Comparison Statements and NULL Values . . . 1927
OPENQUERY and Microsoft Query . . . 1928
AVEVA Historian Time Domain Extensions . . . 1928
Data Retrieval Modes and Options . . . 1929
Understanding Retrieval Modes . . . 1929
Cyclic Retrieval . . . 1930
Cyclic Retrieval - How It Works . . . 1930
Cyclic Retrieval - Supported Value Parameters . . . 1931
Cyclic Retrieval - Query Example . . . 1931
Cyclic Retrieval - Initial Values . . . 1932
Cyclic Retrieval - Handling NULL Values . . . 1932
Daily Retrieval . . . 1932
Daily Retrieval - Supported Value Parameters . . . 1933
Daily Retrieval - Query Example . . . 1933
Daily Retrieval - Initial Values . . . 1934
Daily Retrieval - Handling NULL Values . . . 1934
Delta Retrieval . . . 1934
Delta Retrieval - How It Works . . . 1934
Delta Retrieval - Supported Value Parameters . . . 1934
Delta Retrieval - Query Examples . . . 1935
Delta Retrieval - Query 1 . . . 1935
Delta Retrieval - Query 2 . . . 1936
Delta Retrieval - Query 3 . . . 1937
Delta Retrieval - Query 4 . . . 1938
Delta Retrieval - Initial Values . . . 1939
Delta Retrieval - Handling NULL Values . . . 1939
Full Retrieval . . . 1940
Full Retrieval - How It Works . . . 1940
Full Retrieval - Supported Value Parameters . . . 1941
Full Retrieval - Query Example . . . 1941
Full Retrieval - Initial Values . . . 1941
Interpolated Retrieval . . . 1941
Interpolated Retrieval - How It Works . . . 1942
. . . 0
Interpolated Retrieval - Query Examples . . . 1942
Interpolated Retrieval - Query 1 . . . 1943
Interpolated Retrieval - Query 2 . . . 1944
Interpolated Retrieval - Query 3 . . . 1945
Interpolated Retrieval - Initial and Final Values . . . 1946
Interpolated Retrieval - Handling NULL Values . . . 1946
Best Fit Retrieval . . . 1946
Best Fit Retrieval - How It Works . . . 1948
Best Fit Retrieval - Supported Value Parameters . . . 1949
Best Fit Retrieval - Query Example . . . 1949

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Best Fit Retrieval - Initial and Final Values . . . 1951
Best Fit Retrieval - Handling NULL Values . . . 1951
Average Retrieval . . . 1951
Average Retrieval - How It Works . . . 1952
Average Retrieval - Supported Value Parameters . . . 1953
Average Retrieval - Query Examples . . . 1953
Average Retrieval - Query 1 . . . 1954
Average Retrieval - Query 2 . . . 1955
Average Retrieval - Initial and Final Values . . . 1955
Average Retrieval - Handling NULL Values . . . 1955
Minimum Retrieval . . . 1955
Minimum Retrieval - How It Works . . . 1956
Minimum Retrieval - Supported Value Parameters . . . 1957
Minimum Retrieval - Query Examples . . . 1957
Minimum Retrieval - Query 1 . . . 1957
Minimum Retrieval - Query 2 . . . 1958
Minimum Retrieval - Query 3 . . . 1958
Minimum Retrieval - Initial and Final Values . . . 1959
Minimum Retrieval - Handling NULL Values and Incomplete Cycles . . . 1959
Maximum Retrieval . . . 1960
Maximum Retrieval - How It Works . . . 1961
Maximum Retrieval - Supported Value Parameters . . . 1962
Maximum Retrieval - Query Examples . . . 1962
Maximum Retrieval - Query 1 . . . 1962
Maximum Retrieval - Query 2 . . . 1963
Maximum Retrieval - Query 3 . . . 1963
Maximum Retrieval - Initial and Final Values . . . 1964
Maximum Retrieval - Handling NULL Values and Incomplete Cycles . . . 1964
Integral Retrieval . . . 1965
Integral Retrieval - How It Works . . . 1966
Integral Retrieval - Supported Value Parameters . . . 1966
Integral Retrieval - Query Example . . . 1966
Integral Retrieval - wwExpression Query Example . . . 1967
Integral Retrieval - Initial and Final Values . . . 1968
Integral Retrieval - Handling NULL Values . . . 1968
Slope Retrieval . . . 1968
Slope Retrieval - How It Works . . . 1969
Slope Retrieval - Supported Value Parameters . . . 1969
Slope Retrieval - Query Example . . . 1970
Slope Retrieval - wwExpression Query Example . . . 1971
Slope Retrieval - Initial and Final Values . . . 1972
Slope Retrieval - Handling NULL Values . . . 1972
Counter Retrieval . . . 1972
Counter Retrieval - How It Works . . . 1973
Counter Retrieval - Calculations for a Manually Reset Counter . . . 1974
Counter Retrieval - Using a Counter Deadband . . . 1974
Counter Retrieval - Supported Value Parameters . . . 1975
Counter Retrieval - Initial and Final Values . . . 1975
Counter Retrieval - Handling NULL Values . . . 1975

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Counter Retrieval - Handling Illegal Values . . . 1975
Counter Retrieval - Query Example . . . 1975
ValueState Retrieval . . . 1976
ValueState Retrieval - How It Works . . . 1977
ValueState Retrieval - Supported Value Parameters . . . 1978
ValueState Retrieval - Query Examples . . . 1978
ValueState Retrieval Query 1: Minimum Time in State . . . 1979
ValueState Retrieval Query 2: Minimum Time in State for a Single Tag . . . 1980
ValueState Retrieval Query 3 . . . 1980
ValueState Retrieval Query 4 . . . 1981
ValueState Retrieval Query 5 . . . 1981
ValueState Retrieval Query 6: Querying State Summary Values . . . 1982
ValueState Retrieval - Initial and Final Values . . . 1982
ValueState Retrieval - Handling NULL Values . . . 1983
RoundTrip Retrieval . . . 1983
RoundTrip Retrieval - How It Works . . . 1983
RoundTrip Retrieval - Supported Value Parameters . . . 1984
RoundTrip Retrieval - Query Examples . . . 1984
RoundTrip Retrieval - Initial and Final Values . . . 1985
RoundTrip Retrieval - Handling NULL Values . . . 1985
Edge Detection for Events (wwEdgeDetection) . . . 1986
Edge Detection for Analog Tags . . . 1986
Leading Edge Detection for Analog Tags . . . 1988
Trailing Edge Detection for Analog Tags . . . 1988
Both Leading and Trailing Edge Detection for Analog Tags . . . 1989
Edge Detection for Discrete Tags . . . 1990
Leading Edge Detection for Discrete Tags . . . 1991
Trailing Edge Detection for Discrete Tags . . . 1992
Both Leading and Trailing Edge Detection for Discrete Tags . . . 1993
Predictive Filter . . . 1994
Bounding Value Retrieval . . . 1995
Bounding Value Retrieval - How It Works . . . 1995
Bounding Value Retrieval - Query Examples . . . 1996
Understanding Retrieval Options . . . 1997
Which Options Apply to Which Retrieval Modes? . . . 1997
Using Retrieval Options in a Transact-SQL Statement . . . 1998
Cycle Count (X Values over Equal Time Intervals) (wwCycleCount) . . . 1999
Cycle Count - Query Examples . . . 1999
Cycle Count - Query 1: Using a Single Tag . . . 2000
Cycle Count - Query 2: Using Multiple Tags . . . 2000
Resolution (Values Spaced Every X ms) (wwResolution) . . . 2001
Resolution - Query Example . . . 2002
About Phantom Cycles . . . 2003
Value Deadband (wwValueDeadband) . . . 2005
Value Deadband - Query Examples . . . 2006
Value Deadband - Query 1 . . . 2006
Value Deadband - Query 2 . . . 2007
Time Deadband (wwTimeDeadband) . . . 2009
Time Deadband - Query Examples . . . 2010

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time Deadband - Query 1 . . . 2010
Time Deadband - Query 2 . . . 2011
Time Deadband - Query 3 . . . 2013
History Version (wwVersion) . . . 2014
History Version - Query Example . . . 2014
Interpolation Type (wwInterpolationType) . . . 2015
TimeStamp Rule (wwTimeStampRule) . . . 2017
Time Zone (wwTimeZone) . . . 2019
Quality Rule (wwQualityRule) . . . 2020
Quality Rule - Query Examples . . . 2021
Quality Rule - Query 1 . . . 2021
Quality Rule - Query 2 . . . 2022
Quality Rule - Query 3 . . . 2023
Quality Rule - Query 4 . . . 2024
Quality Rule - Query 5 . . . 2025
Quality Rule - Query 6 . . . 2026
Quality Rule - Query 7 . . . 2027
Quality Rule - Query 8 . . . 2027
State Calculation (wwStateCalc) . . . 2028
Analog Value Filtering (wwFilter) . . . 2030
Statistically Removing Outliers (SigmaLimit) . . . 2030
Converting Analog Values to Discrete Values (ToDiscrete) . . . 2031
"Zeroing" Around a Base Value (SnapTo) . . . 2033
Selecting Values for Analog Summary Tags (wwValueSelector) . . . 2034
Expressions (wwExpression) . . . 2037
Implicit unit conversion . . . 2039
Interpolation with Expressions . . . 2040
Supported arithmetic operators . . . 2040
Arithmetic Operator Examples . . . 2041
Supported logical operators . . . 2042
Supported logical functions . . . 2042
AND . . . 2042
OR . . . 2043
NOT . . . 2043
Supported scalar functions . . . 2043
ABS . . . 2044
ABS Examples . . . 2045
CASTUOM . . . 2046
IF . . . 2047
IF Examples . . . 2048
IFBAD . . . 2049
IFBAD Examples . . . 2050
LOG . . . 2050
LOG Examples . . . 2051
ROUND . . . 2051
ROUND Examples . . . 2052
SIGN . . . 2053
SIGN Examples . . . 2053
SQRT . . . 2054

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQRT Examples . . . 2055
TIMESHIFT . . . 2055
TIMESHIFT Examples . . . 2056
TRUNCATE . . . 2056
TRUNCATE Examples . . . 2057
UOM (Unit of Measure conversion) . . . 2058
UOM Examples . . . 2059
Supported time-series functions . . . 2059
AVERAGE . . . 2060
AVERAGE Examples . . . 2061
AVERAGES . . . 2062
COUNT . . . 2062
COUNT Examples . . . 2063
COUNTALL . . . 2063
COUNTALL Examples . . . 2064
DURATION (Time in State) . . . 2064
DURATION Examples . . . 2065
IFLONGER . . . 2067
IFSTALE . . . 2068
METER . . . 2069
METER Example . . . 2070
MIN/MAX . . . 2070
MIN/MAX Examples . . . 2071
PREV/PREVGOOD . . . 2072
PREV/PREVGOOD Examples . . . 2073
TOTAL . . . 2073
TOTAL Examples . . . 2074
TOTALS . . . 2075
Bounding . . . 2076
Bounding - Query Example . . . 2076
SQL Query Examples . . . 2077
Querying the History Table . . . 2077
Querying the Live Table . . . 2078
Querying the WideHistory Table . . . 2078
Querying Wide Tables in Delta Retrieval Mode . . . 2080
Querying the AnalogSummaryHistory View . . . 2081
Querying the StateSummaryHistory View . . . 2081
Using SliceBy . . . 2083
Querying Metadata with the Latest Value . . . 2087
Using an Unconventional Tagname in a Wide Table Query . . . 2087
Using an INNER REMOTE JOIN . . . 2088
Setting Both a Time and Value Deadband for Retrieval . . . 2088
Using wwResolution, wwCycleCount, and wwRetrievalMode in the Same Query . . . 2091
Determining Cycle Boundaries . . . 2093
Mixing Tag Types in the Same Query . . . 2093
Using a Criteria Condition on a Column of Variant Data . . . 2094
Using DateTime Functions . . . 2094
Using the GROUP BY Clause . . . 2096
Using the COUNT() Function . . . 2096

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using an Arithmetic Function . . . 2097
Using an Aggregate Function . . . 2097
Making and Querying Annotations . . . 2099
Using Comparison Operators with Delta Retrieval . . . 2100
Specifying the Start Date with ">=" . . . 2100
Specifying the Start Date with ">" . . . 2102
Specifying the End Date with "<=" . . . 2104
Specifying the End Date with "<" . . . 2104
Using Comparison Operators with Cyclic Retrieval and Cycle Count . . . 2105
Specifying Cycle Count with Two Equity Operators . . . 2105
Specifying Cycle Count with One Equity Operator . . . 2106
Specifying Cycle Count with No Equity Operators . . . 2107
Using Comparison Operators with Cyclic Retrieval and Resolution . . . 2108
Using Two Equality Operators for Comparison with Cyclic Retrieval and Resolution . . . 2108
Using One Equality Operator for Comparison with Cyclic Retrieval and Resolution . . . 2109
Using No Equality Operators for Comparison with Cyclic Retrieval and Resolution . . . 2111
Returning Time Between Value Changes . . . 2112
Example 1: Cyclic Retrieval . . . 2112
Example 2: Delta and Full Retrieval . . . 2112
Example 3: Querying the WideHistory Table . . . 2114
Example 4: Querying the History Table with the wwValueSelector Parameter . . . 2115
Example 5: Calculating Total Time Between Value Changes . . . 2116
SELECT INTO from a History Table . . . 2117
Moving Data from a SQL Server Table to an Extension Table . . . 2119
Using Server-Side Cursors . . . 2120
Using Stored Procedures in OLE DB Queries . . . 2121
Getting Data from the OPCQualityMap Table . . . 2121
Using Variables with the Wide Table . . . 2122
Retrieval Across a Data Gap in Classically Stored Data . . . 2122
Returned Values for Non-Valid Start Times . . . 2123
Querying Aggregate Data in Different Ways . . . 2124
Bitwise Retrieval for Process Data . . . 2126
SQL Queries for Alarms and Events . . . 2127
Querying Alarms and Events . . . 2128
Example: Listing all events . . . 2128
Example: How often alarms occur . . . 2128
Example: Most frequent alarm per hour . . . 2129
Example: Pinpointing where alarms occur . . . 2130
Example: Showing average time to clearing an alarm . . . 2131
Example: Evaluating response time for alarms . . . 2132
Browser-Friendly Data Retrieval . . . 2134
Historian Data REST API: Customize data retrieval . . . 2134
Supported versions . . . 2134
iHistory and Account Authentication . . . 2135
Data retrieval . . . 2136
Retrieval resources . . . 2139
ProcessValues . . . 2139
AnalogSummary . . . 2145
StateSummary . . . 2151

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Events . . . 2155
Tags . . . 2160
TagProperties . . . 2165
TagPropertyValues . . . 2166
TagGroups . . . 2167
TagSuggest . . . 2169
TagSearch . . . 2170
TagExtendedProperties . . . 2171
Summary (v1 only) . . . 2172
Daily (v1 only) . . . 2174
Hourly (v1 only) . . . 2180
Minutely (v1 only) . . . 2181
SystemParameters (on-premises only) . . . 2183
Retrieval examples . . . 2183
Retrieve data via browser query . . . 2184
Retrieve data using Postman . . . 2184
Excel query examples . . . 2186
PowerBI query example . . . 2189
Querying History Blocks via SQL Server Reporting Services Extension . . . 2191
Retrieval errors . . . 2191
Historian Scenarios Guide . . . 2193
Welcome . . . 2193
Scenario 1: Use hourly and daily reports . . . 2193
Scenario 2: Track inventory and production with packing line counters . . . 2195
Scenario 3: Track liquid and gas use with custody meters . . . 2197
Scenario 4: Analyze cycle time for your process . . . 2198
Scenario 5: Calculate the total flow . . . 2199
Scenario 6: Calculate down time statistics . . . 2201
Scenario 7: Pinpoint reasons for downtime . . . 2202
Scenario 8: Fill in data gaps . . . 2205
Scenario 9: Understand and use data . . . 2207
Scenario 10: Get summary statistics when a certain condition applies . . . 2210
Glossary . . . 2212
Welcome . . . 2212
AVEVA Historian Documentation Set . . . 2212
Documentation Conventions . . . 2213
Glossary . . . 2213
action . . . 2213
action queue . . . 2213
active image . . . 2214
aggregate functions . . . 2214
alarm . . . 2214
alias . . . 2214
alternate storage location . . . 2214
analog summary replication . . . 2214
annotation . . . 2214
application programming interface (API) . . . 2214
architecture . . . 2215
archiving . . . 2215

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

argument . . . 2215
attribute . . . 2215
authentication . . . 2215
back end . . . 2215
bandwidth . . . 2215
block . . . 2215
browser . . . 2215
buffer storage location . . . 2216
byte . . . 2216
cache . . . 2216
circular storage location . . . 2216
client . . . 2216
client/server . . . 2216
Component Object Model (COM) . . . 2216
configuration tables . . . 2217
console . . . 2217
context . . . 2217
CSV . . . 2217
cyclic retrieval . . . 2217
cyclic storage . . . 2217
data . . . 2217
data acquisition . . . 2217
data dictionary . . . 2217
data integrity . . . 2218
data source . . . 2218
data store . . . 2218
data type . . . 2218
database . . . 2218
database name . . . 2218
database object . . . 2218
database object owner . . . 2218
database owner (dbo) . . . 2218
database query . . . 2219
DDE tags . . . 2219
deadband . . . 2219
delta retrieval . . . 2219
delta storage . . . 2219
detector . . . 2219
deviation . . . 2219
discrete . . . 2219
Distributed Component Object Model (DCOM) . . . 2220
domain . . . 2220
dynamic configuration . . . 2220
Dynamic Data Exchange (DDE) . . . 2220
Dynamic Link Library (DLL) . . . 2220
edge detection . . . 2220
engineering unit . . . 2220
event . . . 2220
event block . . . 2221

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

event tag . . . 2221
extension table . . . 2221
failover . . . 2221
field . . . 2221
foreign key (FK) . . . 2221
front end . . . 2221
frozen snapshot . . . 2222
function . . . 2222
group . . . 2222
HCAL . . . 2222
HCAP . . . 2222
heterogeneous query . . . 2222
history block . . . 2222
history block duration . . . 2222
history tables . . . 2222
Holding database . . . 2223
Human-Machine Interface (HMI) . . . 2223
Hypertext Transfer Protocol (HTTP) . . . 2223
I/O . . . 2223
I/O Driver . . . 2223
I/O Server . . . 2223
IDAS . . . 2223
IDENTITY column . . . 2223
index . . . 2223
information system (IS) network . . . 2224
initial value . . . 2224
initialization . . . 2224
in-memory snapshot . . . 2224
integer . . . 2224
IP address . . . 2224
IPX/SPX . . . 2224
item . . . 2224
join . . . 2225
key . . . 2225
latency . . . 2225
leading edge . . . 2225
limit . . . 2225
linked server . . . 2225
live . . . 2225
live tables . . . 2225
local . . . 2225
local area network (LAN) . . . 2226
log file . . . 2226
logical operators . . . 2226
logical tables . . . 2226
logical view . . . 2226
login ID . . . 2226
login identification . . . 2226
logon . . . 2226

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MDAS . . . 2226
memory tag . . . 2227
message pair . . . 2227
Microsoft Management Console (MMC) . . . 2227
millisecond . . . 2227
modification tracking . . . 2227
multi-protocol . . . 2227
named pipe . . . 2227
namespace . . . 2227
network . . . 2228
network address . . . 2228
network card . . . 2228
node . . . 2228
node identifier . . . 2228
NULL . . . 2228
object . . . 2228
Object Linking and Embedding for Databases (OLE DB) . . . 2228
object owner . . . 2229
object permission . . . 2229
OLE DB provider . . . 2229
OPC quality . . . 2229
parameter . . . 2229
partial cycle . . . 2229
password . . . 2229
permanent storage location . . . 2229
permission . . . 2229
phantom cycle . . . 2230
poll rate . . . 2230
port number . . . 2230
primary key (PK) . . . 2230
priority . . . 2230
private namespace . . . 2230
process . . . 2230
process identifier . . . 2230
process network . . . 2231
protocol . . . 2231
public namespace . . . 2231
purging . . . 2231
quality . . . 2231
query . . . 2231
RAID . . . 2231
Random Access Memory (RAM) . . . 2231
rate of change . . . 2232
raw value . . . 2232
real . . . 2232
real-time . . . 2232
record . . . 2232
redundant historian . . . 2232
referential integrity (RI) . . . 2232

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

registry . . . 2232
reinitialization . . . 2232
relational database . . . 2233
remote . . . 2233
remote table . . . 2233
replication delay . . . 2233
replication group . . . 2233
replication latency . . . 2233
replication schedule . . . 2233
replication server . . . 2234
replication tag . . . 2234
resolution . . . 2234
restoring . . . 2234
result . . . 2234
row . . . 2234
row count . . . 2234
rowset . . . 2234
rule . . . 2235
run time . . . 2235
runtime database . . . 2235
scaling . . . 2235
script . . . 2235
server . . . 2235
server name . . . 2235
service . . . 2235
simple replication . . . 2236
Small Computer Systems Interface (SCSI) . . . 2236
snapshot . . . 2236
socket . . . 2236
sort order . . . 2236
SQL . . . 2236
stand-alone installation . . . 2236
state summary replication . . . 2236
statement . . . 2237
storage by exception . . . 2237
storage location . . . 2237
storage path . . . 2237
storage rate . . . 2237
store-and-forward . . . 2237
stored procedure . . . 2237
string . . . 2237
Structured Query Language (SQL) . . . 2238
SuiteLink . . . 2238
summary . . . 2238
summary calculation queue . . . 2238
summary data . . . 2238
summary replication . . . 2238
summary tag . . . 2238
system administrator (sa) . . . 2238

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

system namespace . . . 2239
system parameter . . . 2239
system tags . . . 2239
table . . . 2239
tag . . . 2239
tagname . . . 2239
TCP/IP . . . 2239
thread . . . 2239
tier-1 historian . . . 2240
tier-2 historian . . . 2240
time deadband . . . 2240
time interval . . . 2240
time synchronization . . . 2240
topic . . . 2240
trailing edge . . . 2240
transaction . . . 2240
transaction log . . . 2241
Universal Naming Convention (UNC) . . . 2241
Universal Time Coordinate (UTC) . . . 2241
update . . . 2241
user group . . . 2241
user name . . . 2241
user-defined data type . . . 2241
value deadband . . . 2241
view . . . 2242
wide area network (WAN) . . . 2242
wide tables . . . 2242
wildcard character . . . 2242

Release Notes . . . 2243
Historian Release Notes . . . 2243
Historian 2023 R2 Patch 01 Readme . . . 2243
About this Readme - 2023 R2 Patch 01 . . . 2243
Resolved Issues - 2023 R2 Patch 01 . . . 2244
Historian 2023 R2 Readme . . . 2246
About this Readme - 2023 R2 . . . 2246
New Features in Historian 2023 R2 . . . 2248
Resolved Issues in Historian 2023 R2 . . . 2249
Historian 2023 R2 Readme . . . 2253
About this Readme - 2023 R2 . . . 2253
New Features in Historian 2023 R2 . . . 2255
Resolved Issues in Historian 2023 R2 . . . 2256
Historian Client Release Notes . . . 2260
Historian Client 2023 R2 Patch 01 Readme . . . 2260
About this Readme - 2023 R2 Patch 01 . . . 2260
Resolved Issues - 2023 R2 Patch 01 . . . 2263
Historian Client 2023 R2 Readme . . . 2266
Historian Client 2023 R2 . . . 2266

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

New features in Historian Client 2023 R2 2268
Resolved issues in Historian Client 2023 R2 2269
Historian Client 2023 R2 Readme 2272
Historian Client 2023 R2 2272
New features in Historian Client 2023 R2 2275
Resolved issues in Historian Client 2023 R2 2276

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Welcome to AVEVA Historian

AVEVA Historian is a powerful, high-performance data historian. It can handle the immense volumes of data that modern industrial facilities generate.

Historian can move all of that data fast, acquiring your data for storage and retrieving it when you need it for process analysis and reporting.

AVEVA Historian combines the power and flexibility of Microsoft SQL Server with high speed acquisition and efficient data compression to store time-series data.

With AVEVA Historian, you can:

- **Acquire data from a various sources**
  Historian's Data Acquisition subsystem offers high-speed data capture to acquire plant data from AVEVA I/O Servers, DAServers, InTouch HMI software, Application Server, and other devices.

  AVEVA Historian Historian optimally acquires and stores analog, discrete, and string data. In addition, the SuiteLink protocol used by AVEVA I/O Servers and DAServers provides time and quality stamps as data is added to the system.
- **Store lots of data efficiently**
  The Storage subsystem compresses the data for maximum storage efficiency. In addition to your plant's data, AVEVA Historian stores alarms and events, summary data, configurations, security information, backups, and system monitoring data.
- **Retrieve data when you want it**
  The Retrieval subsystem responds to SQL requests for plant data and allows you to retrieve large amounts of data very quickly. It also supports REST and SDK data retrieval.
- **Replicate your stored data for security**
  The Replication subsystem replicates tags' values to other AVEVA Historian servers with high fidelity or calculates and replicates summaries of those values.
- **Query data using time domain extensions to SQL**
  AVEVA Historian builds on the capabilities of the SQL query language by supporting time-series data and controlling the resolution of returned data in SQL (for example, to give an evenly spaced sampling of data over a period of time).
- **Create, save, and share graphical content**
  AVEVA Historian includes a tool called Insight that lets you quickly search for data and generate clean, clear graphical content that you can save and share.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Legal Information

© 2015-2024 by AVEVA Group Limited or its subsidiaries. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of AVEVA Group Limited. No liability is assumed with respect to the use of the information contained herein.

Although precaution has been taken in the preparation of this documentation, AVEVA assumes no responsibility for errors or omissions. The information in this documentation is subject to change without notice and does not represent a commitment on the part of AVEVA. The software described in this documentation is furnished under a license agreement. This software may be used or copied only in accordance with the terms of such license agreement. AVEVA, the AVEVA logo and logotype, OSIsoft, the OSIsoft logo and logotype, ArchestrA, Avantis, Citect, DYNSIM, eDNA, EYESIM, InBatch, InduSoft, InStep, IntelaTrac, InTouch, Managed PI, OASyS, OSIsoft Advanced Services, OSIsoft Cloud Services, OSIsoft Connected Services, OSIsoft EDS, PIPEPHASE, PI ACE, PI Advanced Computing Engine, PI AF SDK, PI API, PI Asset Framework, PI Audit Viewer, PI Builder, PI Cloud Connect, PI Connectors, PI Data Archive, PI DataLink, PI DataLink Server, PI Developers Club, PI Integrator for Business Analytics, PI Interfaces, PI JDBC Driver, PI Manual Logger, PI Notifications, PI ODBC Driver, PI OLEDB Enterprise, PI OLEDB Provider, PI OPC DA Server, PI OPC HDA Server, PI ProcessBook, PI SDK, PI Server, PI Square, PI System, PI System Access, PI Vision, PI Visualization Suite, PI Web API, PI WebParts, PI Web Services, PRiSM, PRO/II, PROVISION, ROMeo, RLINK, RtReports, SIM4ME, SimCentral, SimSci, Skelta, SmartGlance, Spiral Software, WindowMaker, WindowViewer, and Wonderware are trademarks of AVEVA and/or its subsidiaries. All other brands may be trademarks of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the license agreement with AVEVA Group Limited or its subsidiaries and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12-212, FAR 52.227-19, or their successors, as applicable.

Publication date: Wednesday, October 23, 2024

Publication ID: 1390214

Contact information

AVEVA Group Limited
High Cross
Madingley Road
Cambridge
CB3 0HB. UK

https://sw.aveva.com/

For information on how to contact sales and customer training, see https://sw.aveva.com/contact.

For information on how to contact technical support, see https://sw.aveva.com/support.

To access the AVEVA Knowledge and Support center, visit https://softwaresupport.aveva.com.

Historian Client User Guides

This section includes guides for users of Historian Client and Historian Client Web.

Historian Client User Guide

Welcome

You can use the AVEVA Historian Client software to retrieve data from an AVEVA Historian. Before you can use the AVEVA Historian Client software, the AVEVA Historian must be correctly installed and configured and must be running.

You can view this document online or you can print it, in part or whole, by using the Print feature in Adobe Acrobat Reader.

This guide assumes you know how to use Microsoft Windows, including navigating menus, moving from application to application, and moving objects on the screen. If you need help with these tasks, see the Microsoft online help.

In some areas of the AVEVA Historian Client software, you can also right-click to open a menu. The items listed on this menu change, depending on where you are in the product. All items listed on this menu are available as items on the main menus.

Legal Information

© 2015-2024 by AVEVA Group Limited or its subsidiaries. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of AVEVA Group Limited. No liability is assumed with respect to the use of the information contained herein.

Although precaution has been taken in the preparation of this documentation, AVEVA assumes no responsibility for errors or omissions. The information in this documentation is subject to change without notice and does not represent a commitment on the part of AVEVA. The software described in this documentation is furnished under a license agreement. This software may be used or copied only in accordance with the terms of such license agreement. AVEVA, the AVEVA logo and logotype, OSIsoft, the OSIsoft logo and logotype, ArchestrA, Avantis, Citect, DYNSIM, eDNA, EYESIM, InBatch, InduSoft, InStep, IntelaTrac, InTouch, Managed PI, OASyS, OSIsoft Advanced Services, OSIsoft Cloud Services, OSIsoft Connected Services, OSIsoft EDS, PIPEPHASE, PI ACE, PI Advanced Computing Engine, PI AF SDK, PI API, PI Asset Framework, PI Audit Viewer, PI Builder, PI Cloud Connect, PI Connectors, PI Data Archive, PI DataLink, PI DataLink Server, PI Developers Club, PI Integrator for Business Analytics, PI Interfaces, PI JDBC Driver, PI Manual Logger, PI Notifications, PI ODBC Driver, PI OLEDB Enterprise, PI OLEDB Provider, PI OPC DA Server, PI OPC HDA Server, PI ProcessBook, PI SDK, PI Server, PI Square, PI System, PI System Access, PI Vision, PI Visualization Suite, PI Web API, PI WebParts, PI Web Services, PRiSM, PRO/II, PROVISION, ROMeo, RLINK, RtReports, SIM4ME, SimCentral, SimSci, Skelta, SmartGlance, Spiral Software, WindowMaker, WindowViewer, and Wonderware are trademarks of AVEVA and/or its subsidiaries. All other brands may be trademarks of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the license agreement with AVEVA Group Limited or its subsidiaries and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12-212, FAR 52.227-19, or their successors, as applicable.

Publication date: Wednesday, October 23, 2024

Publication ID: 1390214

Contact information

AVEVA Group Limited
High Cross
Madingley Road
Cambridge
CB3 0HB. UK

https://sw.aveva.com/

For information on how to contact sales and customer training, see https://sw.aveva.com/contact.

For information on how to contact technical support, see https://sw.aveva.com/support.

To access the AVEVA Knowledge and Support center, visit https://softwaresupport.aveva.com.

Introduction

The AVEVA Historian Client software provides a number of client tools to address specific data representation and analysis requirements. These tools remove the requirement to be familiar with the SQL and provide intuitive point-and-click interfaces to access, analyze, and graph both current and historically acquired time-series data.

About the AVEVA Historian Client Software

Whether you are an operator, process engineer, or manager, the AVEVA Historian Client software can help you to organize, explore, analyze, present, and disseminate your process data in a wide variety of formats. All of this can be performed from your desktop computer.

You can also connect and retrieve data from a managed historian server with AVEVA Historian Client.

Important: Only the Trend application can leverage this functionality. Although a managed historian-configured server will appear in other Historian Client applications, it will not be functional in those applications.

The AVEVA Historian Client software is a full-featured suite of applications that maximize the value of the data in the AVEVA Historian. The AVEVA Historian Client software integrates tightly with the most popular Microsoft Office tools. With the AVEVA Historian Client software, you can:

- Explore your data graphically to find important information.
- Analyze the data to produce relevant information.
- Develop and execute ad hoc queries against any data stored in the AVEVA Historian database.
- Visualize the current process state.
- Produce rich automated reports.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Desktop Applications

The AVEVA Historian Client software includes the following stand-alone applications:

- AVEVA Historian Client **Trend.** Enables trending of historical and real time data over time. Powerful features allow data to be compared with other data from different periods. Alarms and limit excursions are readily visible. It is also possible to add and view annotations in your trends.

  For more information, see Aveva Historian Client Trend.
- AVEVA Historian Client **Query**. This point-and-click tool enables complex queries to be created and executed against any AVEVA Historian. Knowledge of the database structure or SQL is not required.

  For more information, see Aveva Historian Client Query.

Microsoft Office Add-Ins

- **AVEVA Historian Client Workbook.** This add-in to Microsoft Excel allows almost any type of analysis and display of data from an AVEVA Historian using the Excel spreadsheet format (.xlsx).

  For more information, see Client Workbook.
- **AVEVA Historian Client Report**. This add-in to Microsoft Word allows sophisticated reporting from an AVEVA Historian using the Word document format (.docx).

  For more information, see AVEVA Historian Client Report.

ActiveX and .NET Controls

aaHistClientTrend and aaHistClientQuery are controls that provide essential functionality of AVEVA Historian Client Trend and AVEVA Historian Client Query for use in container applications such as InTouch® HMI . You can also use AVEVA Historian Client "building block" controls (such as aaHistClientTagPicker, aaHistClientTimeRangePicker, and so on) in your custom applications.

For more information, see Introduction to Controls and Objects.

About AVEVA Historian Client Licensing

AVEVA Historian and AVEVA Historian Client are licensed by AVEVA. You must have a valid license to run AVEVA Historian Client.

When AVEVA Historian is installed on a server, it is associated with a set of AVEVA Historian Client licenses. When you use AVEVA Historian Client to connect to that server, your computer automatically uses one of those client licenses. The license is released when you exit the application.

Your administrator manages these licenses using AVEVA License Manager on the licensing server. Any necessary configuration is automatic.

If you have any issues related to licensing, contact your administrator.

For information about product licensing and activation, see the *AVEVA Licensing Guide* or *Licensing Help*. The *AVEVA Licensing Guide* is available on the AVEVA License Manager node as a PDF file, under the AVEVA start directory.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About the AVEVA Historian

The AVEVA Historian is a real-time relational database for plant data. The historian acquires and stores process data at full resolution and provides real-time and historical plant data together with configuration, event, summary, and associated production data to client applications on the desktop. The historian combines the power and flexibility of Microsoft SQL Server with the high speed acquisition and efficient data compression characteristics of a real-time system.

The AVEVA Historian appears to client applications as a Microsoft SQL Server. The AVEVA Historian database server receives SQL queries and then locates, processes, and returns the data.

In the AVEVA Historian, plant data is stored in special history "extension" tables. The historian surpasses the functionality of Microsoft Transact-SQL by providing time domain extensions that allow for more useful retrieval of time-series data from these tables.

For example, the extension tables support cyclic and delta retrieval. For cyclic retrieval, evenly spaced data at a specified resolution is returned. For delta retrieval, data is returned for each time the value of a tag changed.

The combination of normal SQL Server tables and the extension tables provides a powerful way to access meaningful data stored in the database. Since the historian is a relational database, queries can join data across multiple tables to retrieve data efficiently.

Some examples of database queries possible with the historian are:

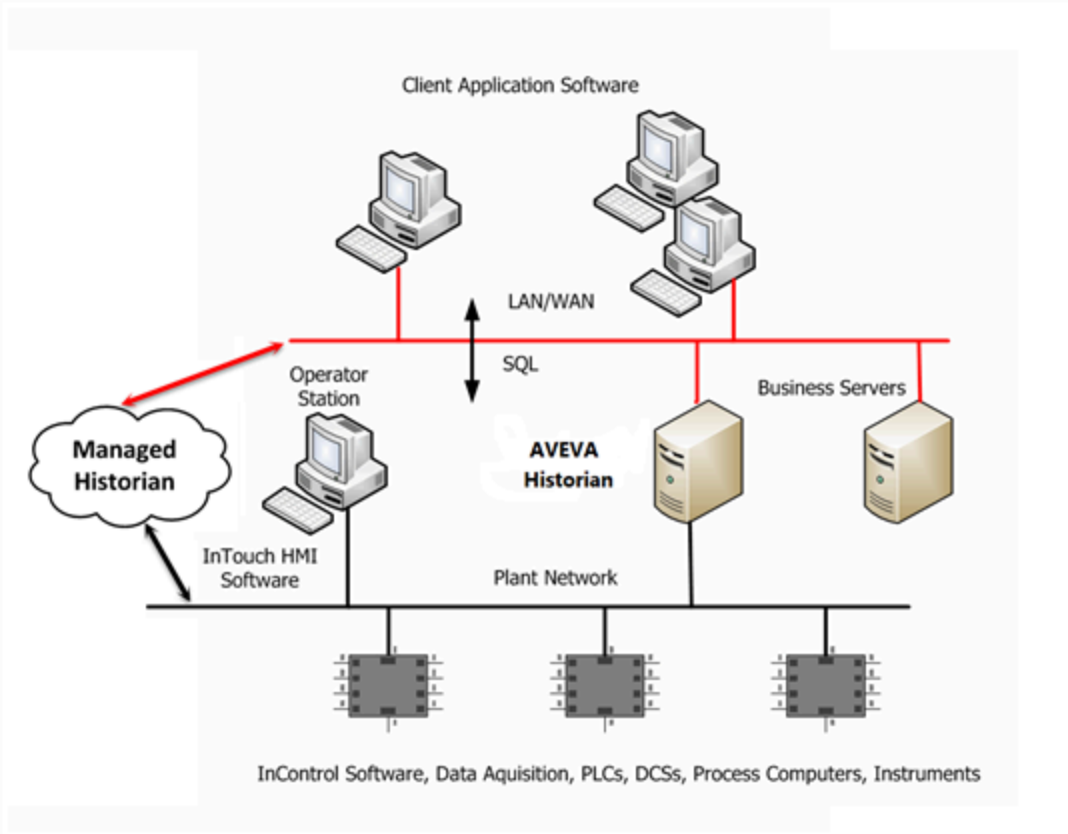
- Average vibration of a motor each day over the last month.
- Annotation for a discrete tag that was made six months ago.
- The limit of an analog tag in the context of a normal production mode. The limit of the same analog tag in the context of an accelerated production mode.
- The values for 50 specified analog tags at a point in time when the value of *x* was greater than 10.
- The path to the storage location for a specific tag.
- 20 evenly distributed data values from the total values stored for an analog tag between 8:00 and 8:30 a.m. on September 12, 2004.
- All data values at 20 minute intervals from the total values stored for an analog tag between 8:00 and 8:30 a.m. on September 12, 2004.
- All values of an analog tag stored on January 8, 2004, where the value of the analog tag changed by 10 engineering units. The data for this analog tag was stored if the value changed by 5 engineering units.
- All values for tags associated with an event boiler trip on January 8, 2004.

Client/Server Architecture

The AVEVA Historian client/server architecture allows for flexible and easy-to-use client applications on the desktop, while ensuring the integrity and security of the data on the server. The computing power of both the client and the server are exploited by optimizing processor intensive operations on the server and minimizing data to be transmitted on the network to improve system performance.

The following illustration shows one possible network architecture where the AVEVA Historian is used as a link between the process network and the business LAN/WAN:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Integration with AVEVA Application Server

The AVEVA Application Server software provides a unified environment for development, deployment, and maintenance of the distributed automation objects.

AVEVA Application Server facilitates:

- Real-time data acquisition
- Data manipulation
- Device communication
- Alarm and event management
- System-wide security
- Remote deployment of objects
- Collaborative engineering

In AVEVA Application Server, externally acquired data items are called attributes. You can configure AVEVA Application Server to store the attributes to an AVEVA Historian server that allows you to analyze and process stored data.

You can use the System Platform IDE to replicate the Galaxy Model View to an AVEVA Historian. Galaxies and objects are represented as groups and attributes are represented as tags in the AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Naming Conventions

The following naming conventions are used in the AVEVA software :

- **Tag name:** The name of an object. For example, Reactor_002.
- **Attribute name:** The name of a variable exposed by an object. For example, ReactLevel.
- **Attribute reference:** The combination of a tag name and an attribute name. The format is:

  <TagName>.<AttributeName>

  For example, Reactor_002.ReactLevel
- **Contained name:** The name of an object with considerations to its place within the overall object hierarchy. By default, the contained name for an object is the same as its tag name. However, if you use the same object within other objects, you can change the contained name at each instance in the object hierarchy to reflect the unique context in which the object is used. For example, you might decide that the contained name for the object called "Reactor_002" should be "Reactor".

  In the Model and Deployment view of the System Platform IDE, the contained name is shown in brackets to the right of the object's tag name in the following format:

  <TagName> [<ContainedName>]

  For example:

  Reactor_002 [Reactor]
- **Hierarchical name:** The contained name for an object, preceded by the tag names of the containing objects in the hierarchy, in the following format:

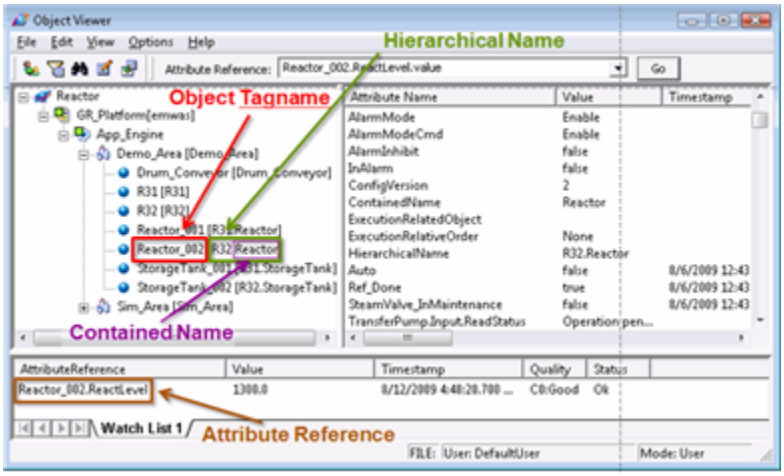
  <ContainerNameN>.<ContainerName2>.<ContainerName1>.<ContainedName>

  For example, if an object whose contained name is Reactor is contained within an object whose tag name is R32, then the hierarchical name of the object is:

  R32.Reactor.

  In the Derivation view of the System Platform IDE, the hierarchical name is shown within brackets to the right of the object's tag name. For example:

  Reactor_002 [R32.Reactor]



You can then use the AVEVA Historian Client software to browse the replicated Model view hierarchy for groups and tags. For more information on replicating the object hierarchy, see the AVEVA Historian documentation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Analyzing Process Data

Process data is any type of information that is relevant to the execution of a process. The following types of information are considered part of process data:

- Real-time data - What is the current value of this tag?
- Historical data - What was the value of this tag every second last Monday?
- Summary data - What is the average of each of these five tags?
- Business data - How much does this particular material cost?
- Event data - When did that boiler trip?
- Configuration data - How many I/O Servers am I using and what are their types?

To improve performance and quality, while reducing cost, all of this acquired information must be able to be analyzed. Plant data is typically analyzed to determine:

- Process analysis, diagnostics, and optimization.
- Materials management, such as raw materials usage.
- Predictive and preventive maintenance of equipment.
- Product and process quality (SPC/SQC).
- Health and safety; environmental impact (EPA/FDA).
- Production reporting.
- Failure analysis.

Common Client Components

Some of the AVEVA Historian Client applications and controls use a common set of components.

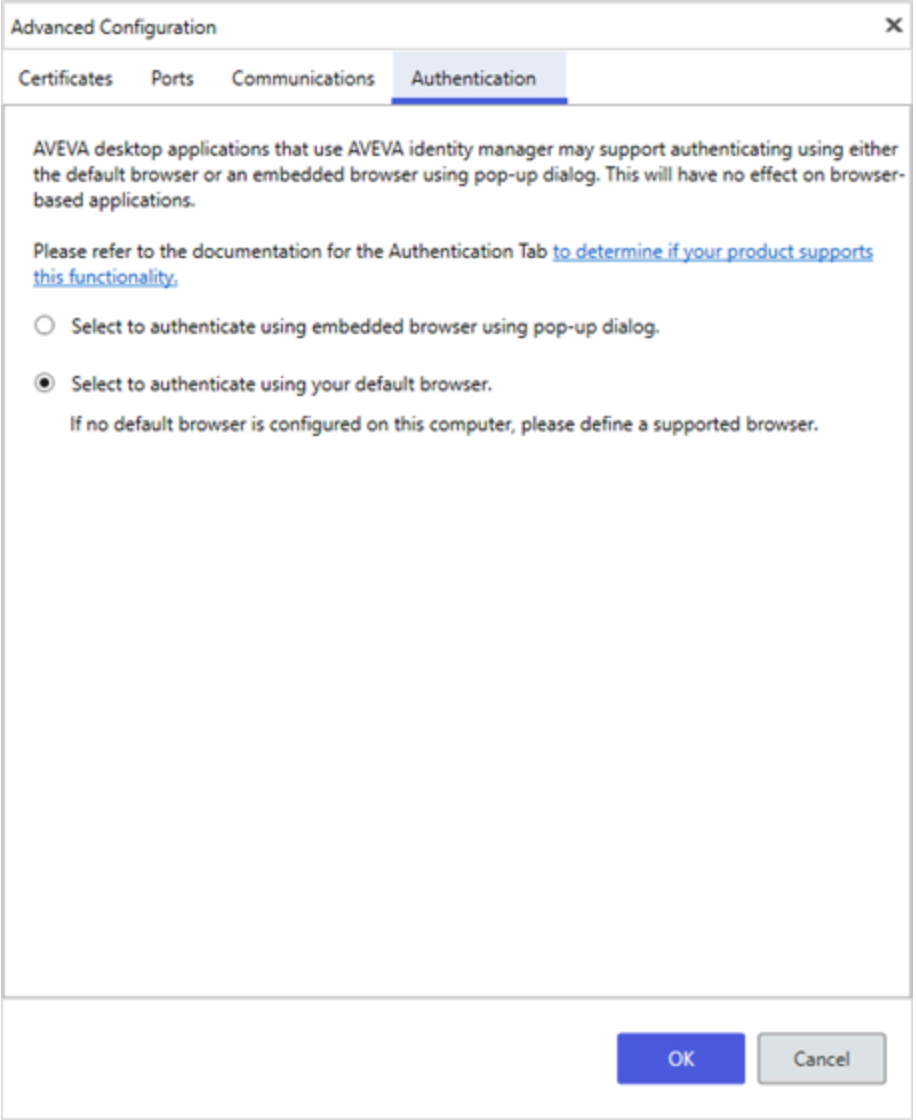
Authenticate using AVEVA Identity Manager

When the Operations Control connected experience is enabled in the Configurator, you will be prompted to authenticate with AVEVA Identity Manager when Historian Client Trend or Historian Client Query are started for the first time. After that, you will be authenticated with SSO.

To authenticate using AVEVA Identity Manager on a web browser:

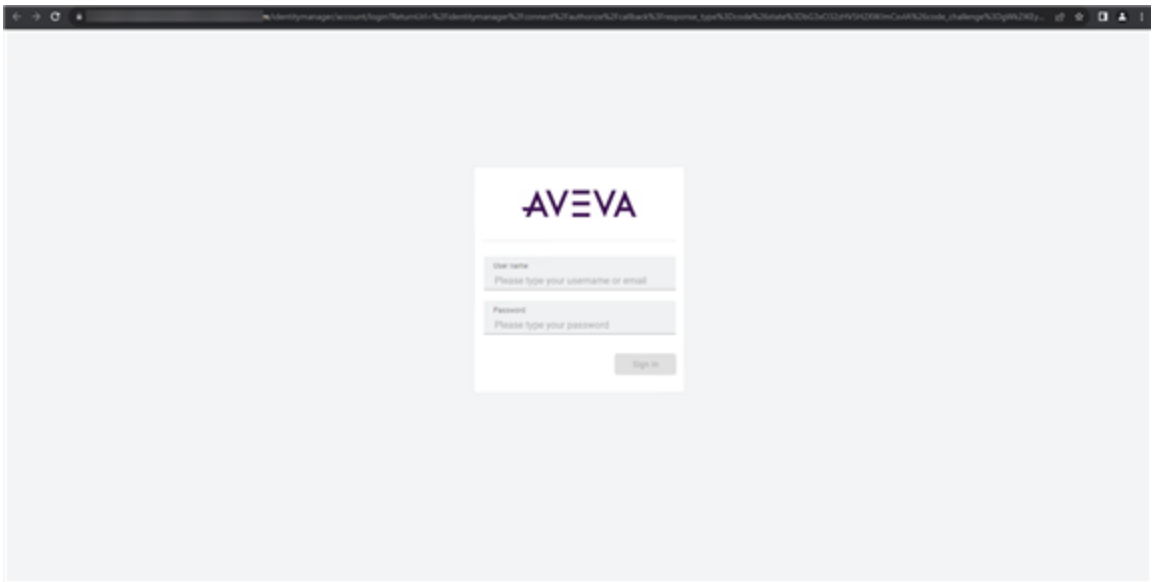
1. To open the AVEVA Identity Manager sign in screen on a web browser, select the **Select to authenticate using your default browser** option under **Common Platform** > **System Management Server** > **Advanced** > **Authentication** in the Configurator.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. When you launch any AVEVA Operations Control application, the AVEVA Identity Manager login page opens in your default browser.
3. On the AVEVA Identity Manager login page, enter a **User name** and **Password** that is registered with an AVEVA Connect account.
4. Select **Sign In**.

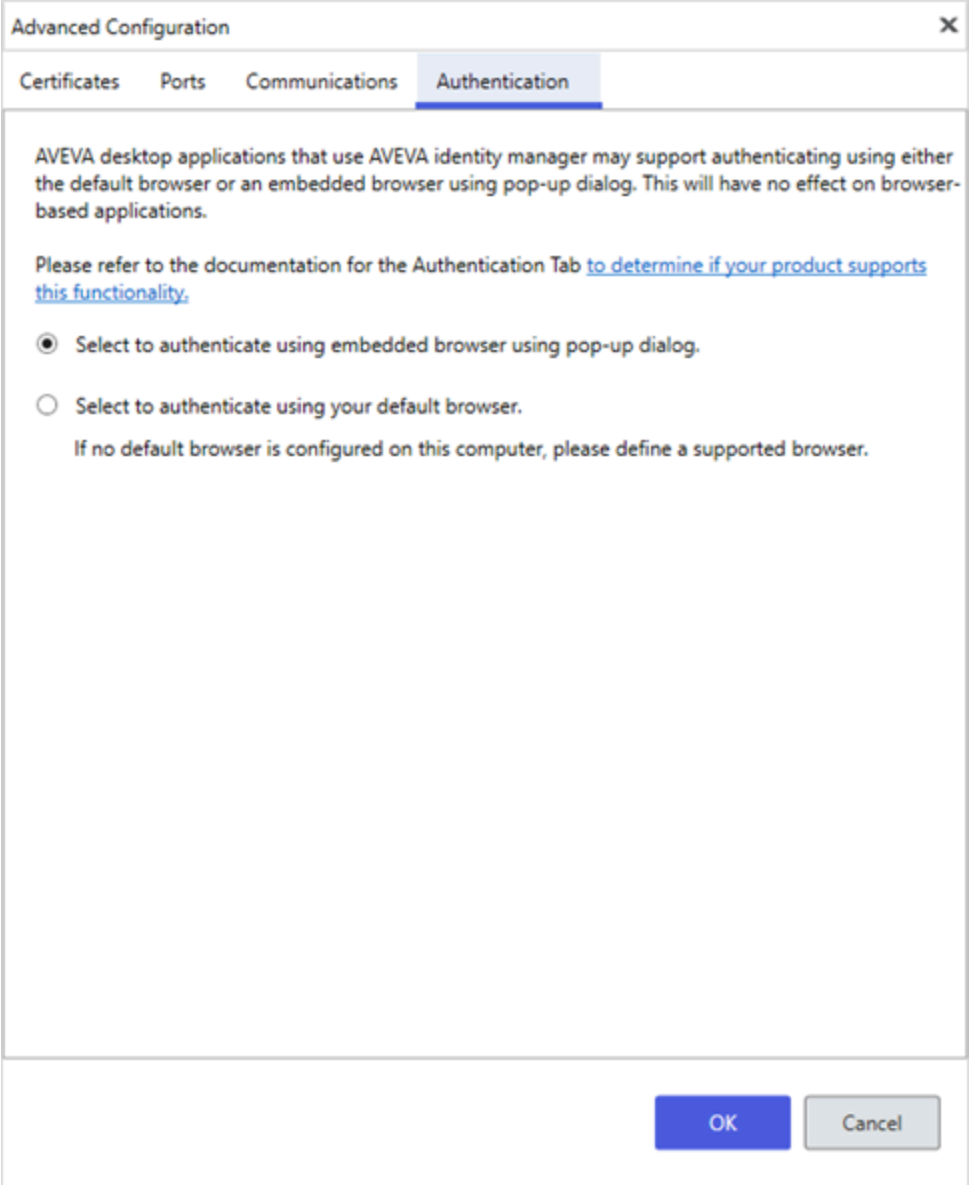
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



To authenticate using AVEVA Identity Manager on an embedded browser:

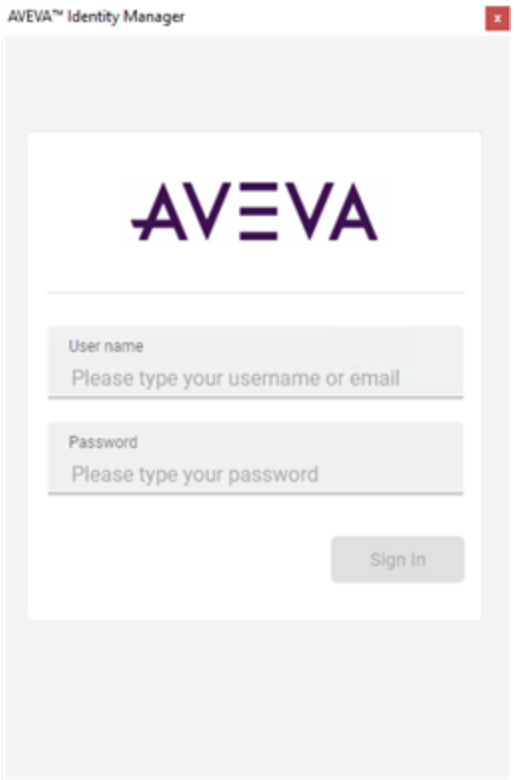
1. To open the AVEVA Identity Manager sign in screen on a embedded browser, select the **Select to authenticate using embedded browser using pop-up dialog** option under **Common Platform** > **System Management Server** > **Advanced** > **Authentication** of the Configurator.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



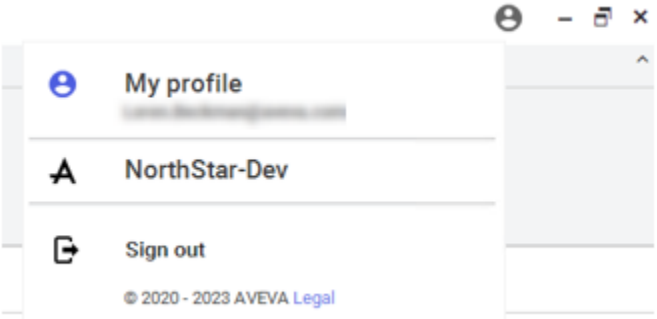
2. When you launch any AVEVA Operations Control application, the AVEVA Identity Manager login page opens in an embedded browser.
3. On the AVEVA Identity Manager login page, enter a **User name** and **Password** that is registered with an AVEVA Connect account.
4. Select **Sign In**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



AVEVA Connect Profile

After successful authentication, the AVEVA Connect Profile displays in the top right-hand corner. It displays your profile name, e-mail address, and the organization name from AVEVA Connect. In the Logger, the e-mail ID, account name, and the products that you are entitled to are displayed.



When you sign out by using the **Sign out** option, the profile icon no longer displays your profile. Select the **Sign in** option to sign in again.

Sign Out from AVEVA Applications

You can sign out from the AVEVA Operations Control connected experience applications by clicking the **Sign Out** option under the **Profile** icon. When you sign out of an application:

- The AVEVA Connect session closes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- You are only signed out of that specific desktop application. Other desktop applications will continue to run with their current signed in user.
  - If you launch the application again, you will be prompted to authenticate.

Closing an application does not end the session. Only that instance of the application closes. The session is still on the AVEVA Connect server. When you open that application again, your session is available and you will be authenticated with single sign-on.

Server Connection Configuration

To use the AVEVA Historian Client applications, you must first connect to an AVEVA Historian using a valid user account that has the right to retrieve data.

You can either use your Windows user account (integrated security) or a valid SQL Server user account or a managed Historian account, depending on how the AVEVA Historian is configured.

Historian Client Trend supports connectivity to Managed Historian, running on Microsoft Azure. Managed Historian is an OData feed which can be accessed using Managed Accounts. The Managed Account credentials come with your AVEVA Cloud subscription and allow you to connect from Historian Client Trend using the OData feed option to the Managed Historian.

Ask your administrator what type of user account you must use to access the server. Server connections are shared among the AVEVA Historian Client applications. For example, once you have configured a server connection in the Trend application, you can use it in the Query application as well.

Note: Managed Historian connection and retrieval is only available for the Trend Client. The Managed Historian server registration will be visible in other Historian Client tools, but is not supported and should not be used.

When you start an AVEVA Historian Client application, you are not automatically logged on to every server that you configured before. You are only logged on to a server when you do the following:

- Open a file that causes data to be retrieved for a tag on that server.
- Expand a server in the Tag Picker to view its Tag List.
- Manually log on to the server.

  Note: Make sure you always log on to the server manually in case you are using a Managed Historian connection.

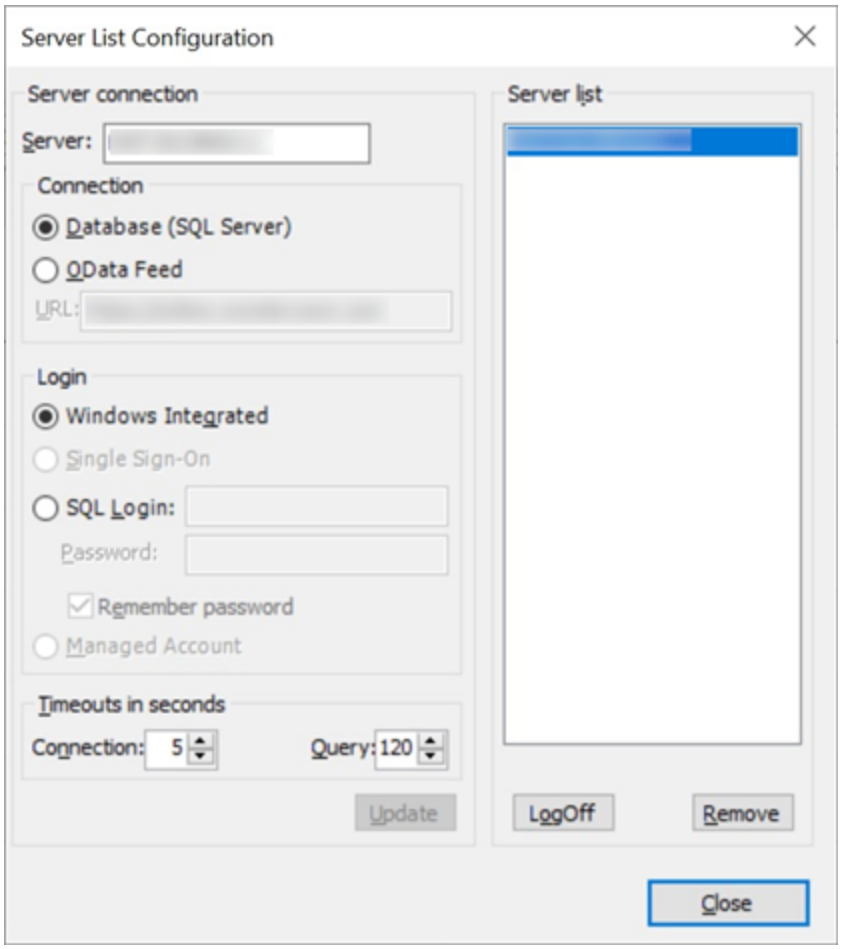
Creating a New Server Connection

The first time you start a Historian Client application, you must create a server connection to the Historian. To create a connection, you need your assigned Historian username and password.

To create a Historian connection

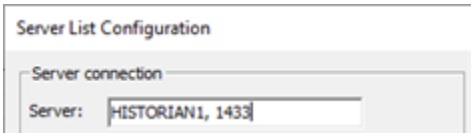
1. Start the Historian Client application.
2. On the **Tools** menu, select **Servers**. The **Server List Configuration** dialog box appears, which shows a list of connected servers or servers that were configured in the past.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. In the **Server** field, enter the name of the server to which you want to connect.
   - For an On-Premises Historian, enter the name of the server where it is hosted.
   - For a Managed Historian, use a name of your choice.
   - For a SQL Server named instance, use the format <server_name>\<instance_name>.

   Note: If the default port for the SQL Server is changed, the Historian Client server configuration should be given as *<server_name>*[*<instance_name>*], *<port_number>*. For example:

4. In the **Connection** section, specify the type of connection you want to use:
   - **Database (SQL Server)**: Choose this option to connect to the SQL Server database.
   - **OData Feed**: Choose this option to connect to the open data protocol feed. Then specify the URL and continue to step 7. You can connect using this option only if you are using a Managed Historian connection.

   Notes: If you select OData Feed to connect to the Historian Server, all the fields related to Login section become unavailable and the Managed Account option becomes deselected automatically.

   By default, the Historian's OData server supports unencrypted HTTP connections. To use an encrypted (HTTPS) OData connection, you will first need to install and configure an SSL client certificate. Contact

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVEVA Technical Support for details.

5. In the **Login** section, select one of the following:
   - **Windows Integrated:** Authenticate with Windows integrated security.
   - **Single Sign-On:** Authenticate with AVEVA Identity Manager. You must be using the connected experience for this option to be enabled. See the C*onfigure Operations Control with connected experience* section in the *AVEVA Service Platform Installation Guide* for details.
   - **SQL Login:** Authenticate with SQL Server credentials, click **SQL Login** and configure the following login details. Then go to step 7.
     - **Login ID:** Type your assigned Historian username. If your system administrator has not assigned you a username and password, you may use one of the default user accounts, which are automatically configured during a typical Historian installation.
     - **Password:** Type the password that is associated with the username. Select the **Remember password** check box to specify for the system to remember your password.
6. In the **Timeouts in seconds** area, configure the time allocated for the database connection and the query execution.
   - **Connection:** The connection time-out period in seconds. Valid values are 1 to 600.
   - **Query:** The query time-out in seconds. Valid values are 1 to 600.
7. Select **Add**.

   After several seconds, the server connection should be established.
8. Click **Close**. An error message appears if a connection cannot be made to the Historian server.

Editing a Server Connection

You can edit an existing server connection.

To edit a server connection

1. On the **Tools** menu, click **Servers**. The **Server List Configuration** dialog box appears.
2. In the **Server List** box, select the name of the server to edit.
3. In the **Server connection** area, edit the details for the server. For more information, see Creating a New Server Connection.
4. Select **Update**.
5. Select **Close**.

   If a connection cannot be made to the server, an error message displays. However, the server will still be added to the list.

Reconnecting to a Server

When a Historian Client application is running in live mode and a server is disconnected, it attempts to reconnect to the server periodically.

The server gets disconnected if it is shut down or restarted, the network goes down, or the SQL service is stopped or restarted.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Setting the Reconnection Time

You can set the time required for a trend to reconnect to a server. By default, the reconnection time is 120 seconds. However, you can modify this value in the win.ini file.

To set the reconnection time

1. Browse to locate the win.ini file in the **C:\WINDOWS** folder.
2. Open the **win.ini** file in a text editor.
3. Find the HistClient entry. If the entry is not found, create a new HistClient entry.
4. Specify the reconnection time in seconds against the RECONNECTION_INTERVAL_SECS key as follows:

   ```
   [HistClient]
   RECONNECTION_INTERVAL_SECS=120
   ```

   Be sure that the specified value is in the range of 1 to 9999 seconds.

After the connection is established, the data is read and trend starts plotting tag values.

About Using a Redundant Historian Server

An AVEVA Historian may be configured to have a symmetrical "partner" AVEVA Historian that can be used as a backup if the primary, or main, historian is not available. This is known as a "redundant historian" setup. No client configuration is required to take advantage of a redundant historian.

When the primary historian is unavailable, a Historian Client automatically switches over to the configured partner historian. The client remains connected to the partner historian, even when the primary historian becomes available again. A Historian Client switches back to an available primary historian if it fails to connect to the partner or during a new attempt to connect to the primary historian, such as when restarting Trend.

When a Historian Client successfully connects to either the primary historian or its partner, the following columns are updated with the connected historian server name in the tags list of the selected tags:

- Server
- I/O Address

The historian name shown in the Tag Picker is always the name of the primary historian, even when the client is connected to the partner.

There is no automatic synchronization built in to the redundant historian setup; it is up to the historian server administrator to make sure that the two historians in the pair are symmetrical and synchronized.

If the SQL Server Service is running while Historian Service is not running, this is not recognized by the Historian Client as a scenario in which the Historian Server is unavailable.

Removing a Server Connection

You can remove a server connection you no longer need. Make sure you select the server you want to delete. After you delete a server, you cannot undelete it.

To remove a server connection

1. On the **Tools** menu, click **Servers**. The **Server List Configuration** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. In the **Server List** box, select the name of the server to remove.
3. Select **Remove**.
4. Click **Close**.

Considerations for VPN Access

Check with your system administrator regarding how to access the server over a virtual private network (VPN).

For example, you may need to provide a fully qualified domain name when specifying the server:

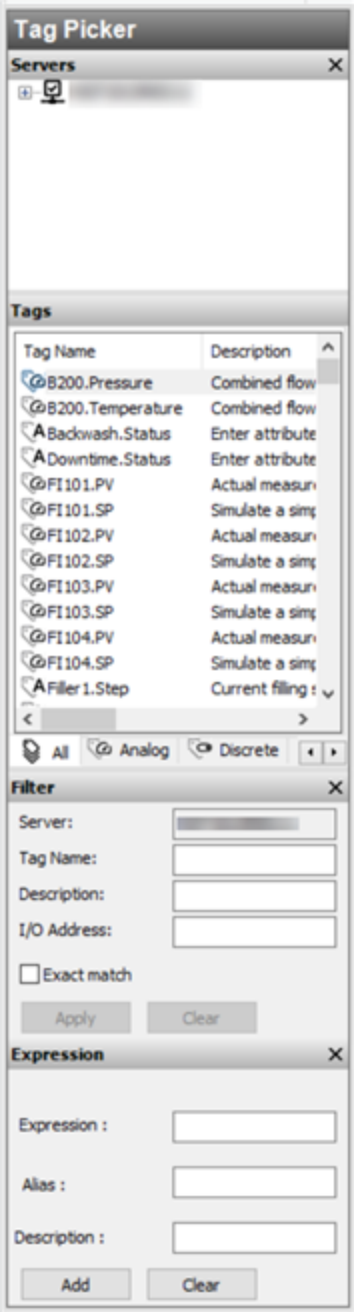
`host-a.example.yourdomain.com.`

Tag Picker

The Tag Picker shows which tag groups and tags exist in the database. It shows all of the tags that are visible to the currently logged on user based on his or her permissions.

Using the Tag Picker, you can quickly search the database for tags of a certain type, for tags that match a particular search pattern, or create an expression. You can then select the items you want to include for the client application or control.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



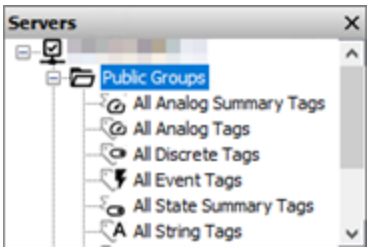
The Tag Picker is comprised of the following panes:

- Servers Pane
- Tags Pane
- Filter Pane
- Expression Pane

Servers Pane

The **Servers** pane shows a list of Historian folders. The **Servers** pane enables you to navigate through the folder structure (namespace) of one or more Historian servers and select a group (folder) of tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Note: When you select a parent item (such as Public Groups in the illustration above), the Tag Pane shows items from all of the child items (such as All Analog Summary Tags and its peers above).

The **Servers** pane shows the following items:

Category	Description
Servers	All objects that make up the basic Historian system, such as tags, I/O Servers, defined engineering units, storage locations, and so on.
Public Groups	All objects that are visible to all clients. If you have administrative permissions, you can create, rename, and delete groups in the public groups folder.
Private Groups	All objects that are visible to the user that is currently logged on. Users can create, rename, and delete groups in the private groups folder.

Editing Groups

You can add group folders as you would add a new folder in Windows Explorer. For example, you can create the "BoilerTags" group under the existing "Private Groups" group folder. You can also delete, cut, copy, paste, and drag objects from one folder to another.

Note: You cannot add, edit, delete, or rename a tag groups when connected to a Managed Historian.

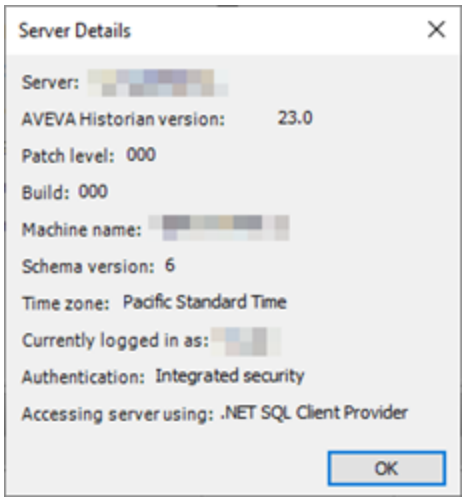
Viewing Server Details

You can view information such as the version number, time zone, and security mode for any server in the **Servers** pane.

To view server details

1. In the **Servers** pane, right-click on a server and then click **Server details**. The **Server Details** dialog box appears.

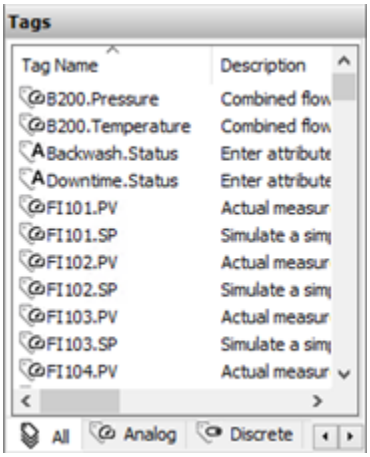
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Click **OK**.

Tags Pane

The **Tags** pane shows all the tags for the currently selected group in the **Servers** pane.

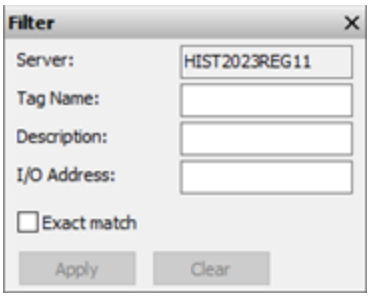


- To select multiple tags from the list, hold the CTRL and/or SHIFT key while clicking.
- To view only tags of a certain type, click the appropriate tab at the bottom of the pane.
- To sort the table by a particular column, click the column heading.

Filter Pane

Use the **Filter** pane to filter the tags listed in the **Tags** pane according to criteria that you specify. You can filter tags by name, description, and I/O address.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The following wildcard characters are supported:

Wildcard Character	Filter Function
%	Any string of zero or more characters.
_	Any single character.
[ ]	Any single character within the specified range or set. For example: • [a-f] • [abcdef]
[^]	Any single character not within the specified range or set. For example: • [^a - f] • [^abcdef]

For example, to find all tag names ending with "level," enter "%level." Filter criteria are not case-sensitive.

Note: Wildcard characters cannot be treated literally in a filter; they will always act as wildcard characters.

When the **Servers** pane and the **Filter** pane are both visible, the filter conditions apply to the selected group in the **Servers** pane. When the **Servers** pane is hidden, the filter applies to all of the tags for the selected Historian.

To apply a filter

1. In the **Server** field, specify or verify the server.

   This field is not available if the **Servers** pane is visible.
2. In the **Tag name** field, enter the string to match for the tagname.
3. In the **Description** field, enter the string to match for the description.
4. In the **I/O Address** field, enter the string to match for the I/O address.
5. Select **Exact match** to search for tags that exactly match the entire string that you provided for the tagname and/or description fields.

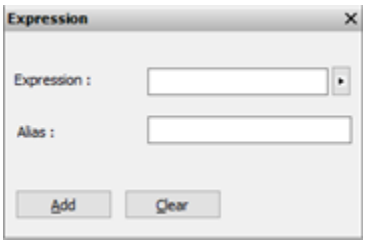
   For example, if you specify "level" as the tagname and do not select **Exact match**, any tagname that contains the string "level" appears. This will include tagnames such as "ReactLevel," "ProdLevel," and "$AccessLevel."
6. Click **Apply** to apply the filter criteria.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

7. Click **Clear** to clear the filter criteria.

Expression Pane

Use the Expression pane to add an expression that will show in your trend. See Expressions (wwExpression) for information about the syntax and usage for expressions.



Add an expression

1. In the **Expression** field, add the expression you wish to display in the trend.

   Select the **Expanded Expression Editor** button to open a larger expression window. This is helpful for entering longer expressions.
2. In the **Alias** field, a default alias is assigned. If it is changed, the revised alias will show as the tag name of the expression in the trend. If the alias is left at default, the value from the Expression field will be used instead.
3. Select **Add** to add the expression to the trend.

   Note: A trend with an expression must use extension tables for the source. If you configure a trend with a source other than **Extension tables** in the options, the trend is automatically processed with extension tables. A message to that effect appears in the log.

Showing/Hiding the Tag Picker

To show or hide the Tag Picker

- From the **View** menu, select **Tag Picker** to toggle the tag picker on or off.

Showing/Hiding Panes in the Tag Picker

The Servers, Filter, and Expression panes in the Tag Picker can be hidden or shown.

To hide a pane

- Right-click in the Tag Picker. Panes that are shown have a checkmark next to them.
- Select a pane to hide it.

To show a pane

- Right-click in the **Tag Picker.**
- Select a pane to show it.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing the Hierarchical Name

A hierarchical name represents the fully qualified name of an object within Historian. For more information on hierarchical names, see Integration with AVEVA Application Server.

To view a hierarchical name

- Right-click in the Tag Picker and click **Use hierarchical name**.

The **Tags** pane shows the hierarchical names instead of tag names.

Tag Picker Views

By default, all Historian Client applications (except for the Workbook add-in) start with the **Tag Picker** in the horizontal view. You can view the **Servers** and **Tags** panes in a vertical view instead.

To change the Tag Picker to the vertical view

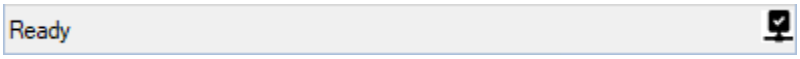
- Right-click in the Tag Picker and then click **Vertical orientation** so that a check mark appears.

To change to the horizontal view

- Right-click in the Tag Picker and then click **Vertical orientation** so that no check mark appears.

Status Bar

The status bar allows you to view the status of the connection to the AVEVA Historian and any other status messages that may be sent by the client.



To show or hide the status bar

- From the **View** menu, select **Status Bar** to toggle the status bar on or off.

The icon at the right side of the status bar indicates the status of the servers that are being used by the AVEVA Historian Client software. The following table describes the icons and their meanings:

Icon	Status Meaning
	No servers are configured.
	Connections (log ons) have been established to all servers in the server configuration list.
	Connections (log ons) have been established to most of the servers in the list.
	No connections (log ons) have been established to most of the servers in the list.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Icon	Status Meaning
	No connections (log ons) have been established to any of the servers in the list.

Click the icon to access the **Server List Configuration** dialog box. For more information, see Server Connection Configuration.

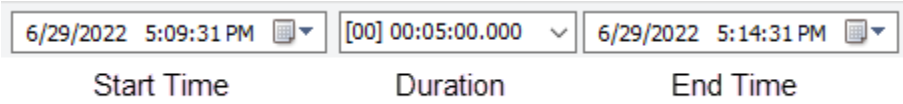
Time Picker

The time picker allows you to select a time range by specifying a start time, end time, and/or duration.

An error appears next to the start or end time controls if you specify an invalid time period. For example, an end time before a start time.

To specify a time period

1. On the Time toolbar, specify the start time, end time, and/or duration. To select a date from a calendar, click the down arrow on the start time or end time field. To select a predefined duration, click the down arrow on the duration field.



   When you change one of the fields, one of the other fields is recalculated automatically. When you select a field, a blue frame appears around the field that will be recalculated as a result of the change.

   The relation between changed and updated times is as follows:

You change...	The time picker updates...
Start time	End time (based on duration)
End time	Start time (based on duration)
Duration	Start time (based on end time)

   If you change multiple time fields in a row, which field is updated depends on which two other fields you changed last. For example, if you change the start time and then the end time, the duration is calculated accordingly. If you change the start time and then the duration, the end time is calculated, and so on.

2. Press ENTER.

To specify a time period relative to the current time

1. Do one of the following:
   - On the **Chart** menu, click **Update to Current Time** so that a check mark appears.
   - Click the **Update to Current Time** toolbar button so that it is highlighted.
2. In the duration list of the Time toolbar, click a duration or type one manually.

   The start time is automatically calculated as the current time minus the duration you selected, and the trend display is updated with the new time period.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing Program and License Information

For each AVEVA Historian Client application, you can view program information, such as the version of the program, copyright information, and licensing information.

To view program information

1. On the **Help** menu, click **About <client name>**. The **About** dialog box appears, showing version and copyright information.
2. Click **OK**.

To view license information

1. On the **Help** menu, click **License Status**. The **License Status** dialog box appears, showing license information.
2. Click **OK**.

Client Trend

AVEVA Historian Client Trend is a client application that lets you query tags from an AVEVA Historian database and plot them on a graphical display. Trend supports two different chart types: a regular trend curve and an XY scatter plot.

After you add tags to a trend chart, you can manipulate the display in a variety of ways, including panning, zooming, and scaling. You can customize any trend by configuring display options and set general options for use with all trends.

Before Trend can be used to query tag information from the database, the server must be running and you must have security access.

This section describes the basic procedures for working with regular trends. Most of this information also applies to XY scatter plots. For information specific to XY scatter plots, see Working with Scatter Plots.

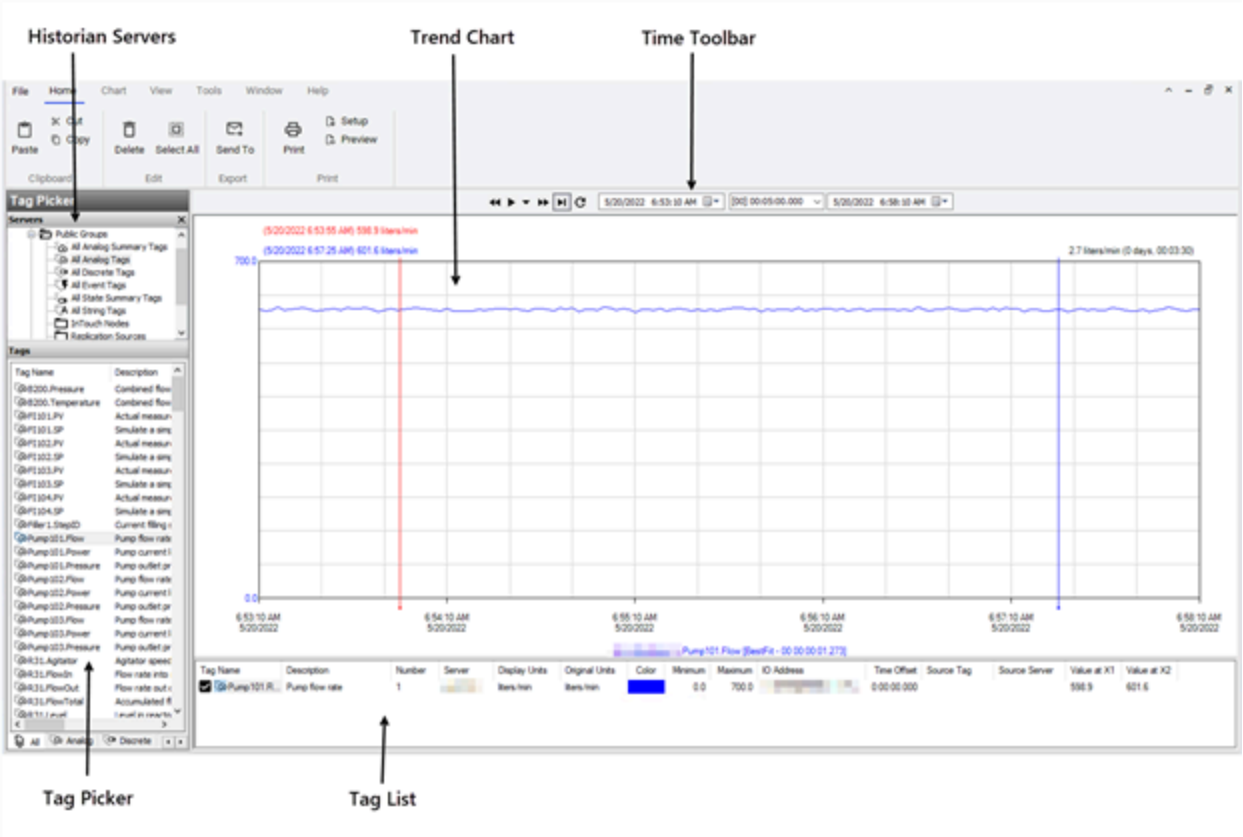
AVEVA Historian Client Trend

AVEVA Historian Client Trend is a stand-alone app which contains a set of properties that can be configured in both design-time and run-time. The design-time properties are set to default values that enable you to show trends in your ViewApps with minimal configuration work. Changes made to a property during run time override the value set to the same property during design time.

When you start the Historian Client Trend for the first time, you must log on and connect to a Historian server. If you are opening an existing Trend file that includes at least one server configuration and the log in was successful, you are not prompted to log on. For more information about connecting to a Historian, see Creating a New Server Connection.

After you establish a connection with the server, the Trend main window appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- For information about assigning values to properties, see Configuring Trend File Properties.
- For information on using the Tag Picker and the Time picker, see Configure a Trend. To show or hide tool bars or components, use the corresponding commands on the **View** menu.

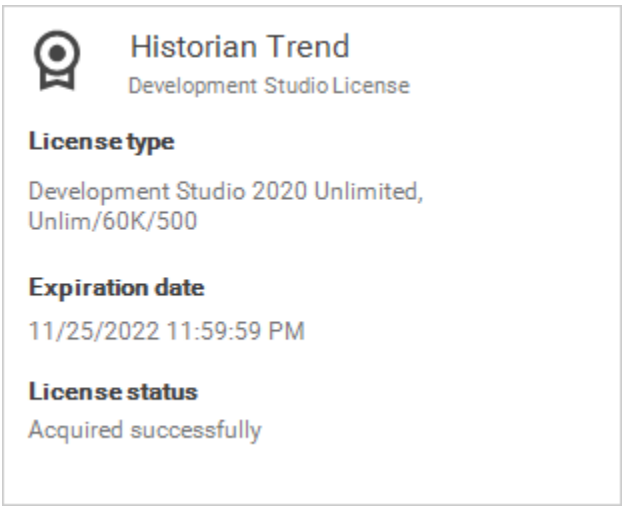
Viewing License Information

You can view information about your AVEVA Historian Client license by selecting the **File** menu, and then selecting License. Details about your license are displayed.

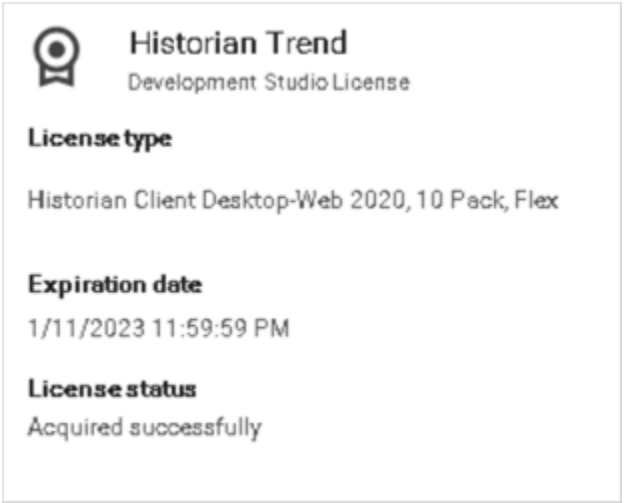
- The **About** section displays version information:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- The **License information** section displays details about your license:



- If your license is a Flex license, the details are slightly different:



Working with Trend Files

This section describes how to create, open, and save trend files. A trend file contains all of the configuration data required to trend one or more tags, such as the tags, time axes, colors, zoom level, and so on.

Any mix of analog, discrete, event, or summary tags is allowed. However, the mix of summary tags with other tags is dependant on the retrieval mode. For example, if the summary tags are supported by Cyclic retrieval mode and that retrieval mode is set, then we can mix the summary tags with the other tags for trending.

There is no pre-set limit to the number of tags that you can have in a trend; however, keep in mind performance limitations for data retrieval for your computer system.

Creating a New Trend

Creating a new trend chart resets all trend settings to the default values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To create a new trend

- On the **File** menu, click **New**.

To configure the new trend, see Configure a Trend. By default, the new trend is in standard trend mode. To create an XY scatter plot, see Viewing Data in a Scatter Plot.

Configuring Default Settings for a Trend File

You can configure default settings for a trend in a "Default.aaTrend" file and use it as a template for your new trend documents.

The trend properties such as border thickness, border style, background color, Tag List pane height and columns, and pens and their properties are saved in the Default.aaTrend file.

When you start the Trend application or Trend control for the first time, the Default.aaTrend file is created in:

C:\Users\<User>\AppData\Local\Wonderware\ActiveFactory\Trend

To configure the default settings

1. Create a new trend file. For more information, see Creating a New Trend.
2. Configure the default settings such as the background color, pen color, and trend chart views.
3. Save the file as Default.aaTrend in the default location.
4. Restart the Trend application.

The default settings are applied to the new trend documents. To restore the default trend settings, delete the Default.aaTrend file and create a new trend document.

Opening an Existing Trend

To open an existing trend

1. On the **File** menu, select **Open**.
2. Select **Trend** to open a single trend file.

   The standard Windows **Open** dialog box appears.
3. Browse to and select the trend file to open. All trend files have the .aaTrend extension.
4. Click **Open**. The trend appears in the chart.

   Note: You can also double-click the .aaTrend file in the Windows Explorer to start up Trend.

To open a recent trend

- From the **File** menu, locate the file in **Recent Files**, and then double-click the name of the file to open.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Saving a Trend

To save a trend

1. From the **File** menu, select **Save**.
2. Click the **Save** button

.

   If you are saving the trend for the first time, the standard Windows **Save As** dialog box appears. Otherwise, the trend is saved to disk using the existing file name.
3. In the **Save As** dialog box, type a name for the trend. All trend files have the .aaTrend extension.

   Note: The trend files with .aaTrend extension are not encrypted.
4. Click **OK**.

To save a trend as another name

1. From the **File** menu, select **Save**.
2. Click **Save As**.

   The standard Windows **Save As** dialog box appears.
3. In the **Save As** dialog box, type a name for the trend. All trend files have the .aaTrend extension.

Note: The trend files with .aaTrend extension are not encrypted.

1. Click **OK**.

Closing a Trend

To close a trend

Do one of the following:

- On the **File** menu, click **Close**.
- Click the trend window's close button ✕.

Undoing/Redoing Actions

The trend application supports 64 levels of undo/redo. You can undo/redo all of the actions for scaling and moving, as well as for adding and removing tags in the trend.

To undo an action

- Click the **Undo** toolbar button

To redo an action

- Click the **Redo** toolbar button

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Working with Trend Sets

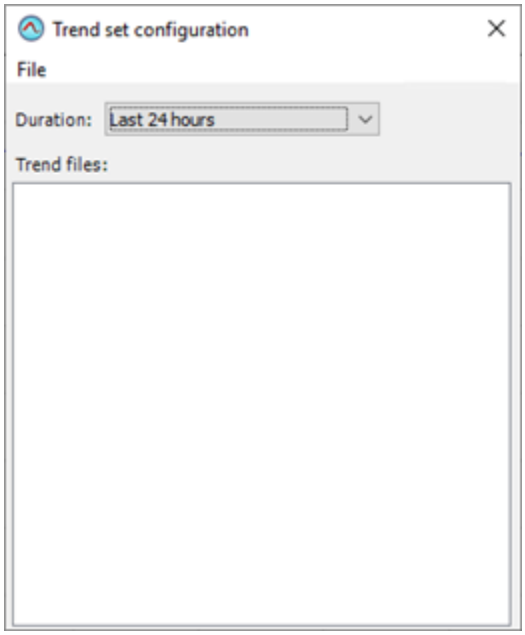
This section describes how to create, open, and save a trend set. A trend set is a saved grouping of trend files. You can either specify a common trend duration (for example, the last 24 hours) to apply to all of the files in the set or specify the time period for each individual trend file in the set.

Common trend duration allows you to easily print or open information for the same duration from multiple trend files at the same time, whereas the varied time period for each individual trend file in the set allows you to open multiple trend files in a trend set with their respective time periods.

Creating a Trend Set

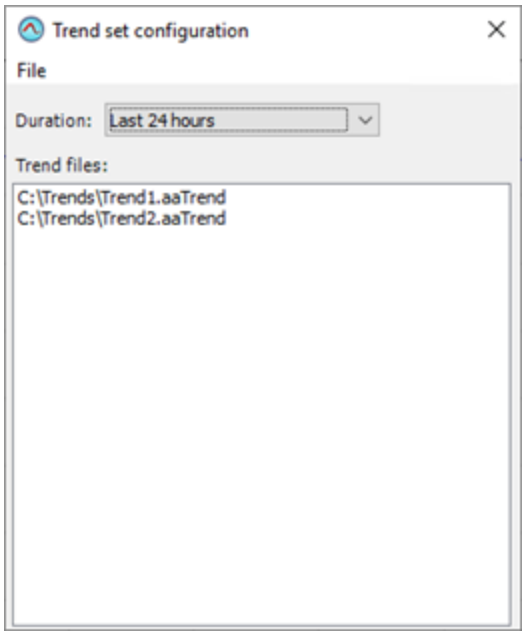
To create a trend set

1. From the **File** menu, select **New**, then click **Trend Set**. The **Trend set configuration** dialog box appears.



2. On the **File** menu, click **Add**. The standard Windows **Open** dialog box appears.
3. Select the trend file(s) to add to the set.
4. Click **Open**. The added files appear in the **Trend set configuration** dialog box.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



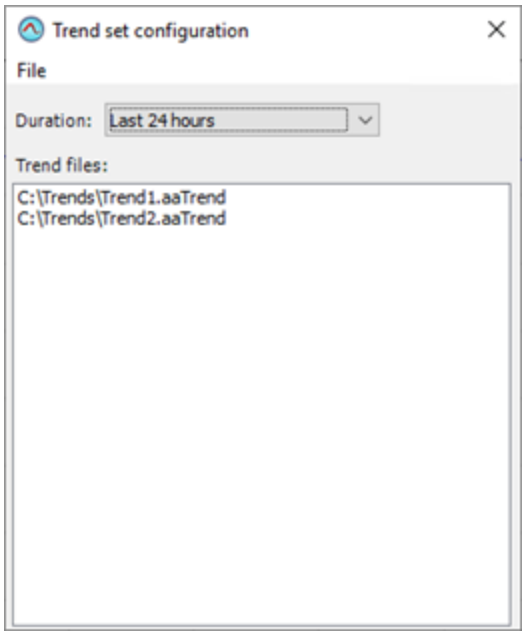
5. In the **Duration** list, select the time period for the trend set. The duration is applied to all of the files in the set. Data that is outside of the scope of the trend set duration is ignored.
6. To save the trend set, on the **File** menu, click **Save As**. The standard Windows **Save As** dialog box appears.
7. In the **File name** box, type a name for the file.
8. Click **Save**.

Editing a Trend Set

To edit a trend set

1. On the **File** menu, click **Open Trend Set**. The standard Windows **Open** dialog box appears.
2. Select the .aaTrendSet file.
3. Click **Open**. The **Trend set configuration** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. Add or delete trend files as required.
5. To exit, on the **File** menu, click **Exit**.

Deleting Files in a Trend Set

To delete a file

1. Select the file in the **Trend files** window.
2. On the **File** menu, click **Delete**.

Configure a Trend

When you configure a trend, you must select the tag(s) for which you want to query the trend data. This data is queried from the Historian database(s) to which you are currently logged on. After you select tags for the trend, you can set the start date and end date for the trend. The default settings in the Trend app are generally adequate to enable you to begin tracking trends.

The following list summarizes the key steps to configure the HistoricalTrendApp during run time.

1. Place the HistoricalTrendApp on a layout pane.
2. Configure the Trend app properties. You can use the default settings, or change them as needed. Default settings are as follows:
   - **Follow Current Context:** Enabled
   - **Max Context Tags to Show:** Up to 10 context tags are shown by default. If the asset you are viewing has more than 10 tags, the first 10 that are listed alphabetically are shown. Showing more than 10 may impact performance.
   - **Pens:** Empty (null). You can add additional context tags to show their trend data by entering the static value or bind to a reference. The pens configured here are in addition to the max context tags setting. List additional pens by entering fully qualified attribute names (for example,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ObjectName.AttributeName). Use a comma to separate attribute names. You can also bind a reference to you pen list. To create a reference:

a. Create a ViewApp Namespace object in the Visualization folder and rename it (for example, "TrendSettings").
b. Create an attribute of string type in the ViewApp Namespace. In the following example, the attribute name is "MyPens."
c. Add the fully qualified attribute names you want to add to the trend. Multiple names must be separated by commas.
d. For the Pens property, create the dynamic binding read-only reference to the ViewApp Namespace (for example, MyViewApp.TrendSettings.MyPens).

3. Deploy and start the ViewApp containing the HistoricalTrendApp.
4. Log in to the Historian Server that contains the data to you want to show in a trend.

Configuring a Trend to Use a Summary Tag

Summary tags consist of summarized data of tags from a Historian server. A summary tag provides an aggregation calculation (minimum, maximum, average, or sum) that is configured on a Historian server. Summary tags are of two types: analog and state.

- Analog summary tags provide summary statistics for analog tags.
- State summary tags provide the summaries of the states of the tag value of analog (integer only), discrete, and string tags.

You can select one or more summary tags from the Tag Picker and drag them to the Tag List pane. When you drag a summary tag to the Tag List, the Trend application plots the value of the aggregate calculation on the Trend chart. The aggregate calculation is performed when you configure the summary tag on the Historian server. For example, if you have configured a ReactTemp_Avg_Hourly summary tag to store the hourly averages of the Reactor temperature, the Trend application shows the hourly average value of the Reactor temperature when you drag and drop the ReactTemp_Avg_Hourly tag to the Trend chart. For more information on the Tag List pane, see Viewing Tag Definition Information.

You can configure trend options for a summary tag. For more information, see Configuring Trend Options for a Tag.

For more information on retrieving summary tags, see Configuring Retrieval Options for a Tag or Configuring Retrieval Options.

Working with Replicated Tags

You can replicate tag information in a Historian from one Historian to another. This allows you to replicate tag data from one or more Historians (known as tier-1 historians) to one or more other historians (known as tier-2 historians). You can replicate tag data to the same server as the tier-1 historian.

You can replicate tag data directly using simple replication, where the tag information is replicated directly to the tier-2 historian. For simple replication, every value for a tag is copied. You can also set up summary tags that receive a summarized version of the tag data.

You can drill down from a source tag to its replicated tag or drill up from a replicated tag to its source tag. You can add a source tag with its replicated tag or a replicated tag with its source tag in the active trend chart. You

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

can also replace a source tag with its replicated tag or a replicated tag with its source tag in the active trend chart.

Adding a Source Tag or Replicated Tag

You can select a source tag or replicated tag from the Tag Picker to add the corresponding replicated tag or source tag to the active trend chart.

To add a source tag or replicated tag

1. Select a tag in the Tag Picker.
2. If the selected tag is a source tag, do the following:
   - In the **Tags** pane, right-click the selected tag, then select **Add - replicated tag**, and then select the tag that you want to add to the trend chart. The corresponding replicated tag is added to the active trend chart.
3. If the selected tag is a replicated tag, do the following:
   - In the **Tags** pane, right-click the selected tag, and then click **Add - source tag**. The corresponding source tag is added to the active trend chart.

The **Add** command is not available if:

- Multiple tags are selected in the Tag Picker.
- A normal tag that is neither a source tag nor a replicated tag is selected in the Tag Picker.

Note: You cannot execute the **Add** command if a source tag is deleted but its replication configuration still exists in the Historian.

The replicated tags are not listed in the context menu if:

- The replicated tags are not committed in the Historian.
- The replication schedule is removed from the Historian. For example, you are connected to a Historian 10.0 server and you create a tag called 'MyTag'. 'MyTag' is replicated as a simple tag called 'MyServer.MyTag'. When you execute the **Add - replicated tag** command, the 'MyServer.MyTag' tag is shown. When you execute the **Add - source tag** command, the 'MyTag' tag is shown. At this instance, if the replication link between 'MyTag' and 'MyServer.MyTag' is removed and if you execute the **Add - replicated tag** command, the 'MyServer.MyTag' tag is not shown in the list of replicated tags.

  However, if you execute the **Add - source tag** command, the 'MyTag' tag is shown as 'MyTag'. If 'MyServer.MyTag' is the only replicated tag, 'MyTag' is considered as a normal tag.

  The above scenario holds true if the entire replication schedule is removed in the Historian. If only one replication is removed, the list shows the remaining replicated tags.

Finding a Source Tag or Replicated Tag

You can select a source tag or replicated tag from the Tag Picker to find the corresponding replicated or source tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To find a source tag or replicated tag

1. Select a tag in the Tag Picker.
2. If the selected tag is a source tag, do the following:
   - In the **Tags** pane, right-click the selected tag, point to **Find - replicated tag**, and then click the tag that you want to find.

     The application navigates within the Tag Picker to find the corresponding replicated tag.
3. If the selected tag is a replicated tag, do the following:
   - In the **Tags** pane, right-click the selected tag, and then click **Find - source tag**.

     The application navigates within the Tag Picker to find the corresponding source tag.

The **Find** command is not available if:

- Multiple tags are selected in the Tag Picker.
- A normal tag that is neither a source tag nor a replicated tag is selected in the Tag Picker.

Note: You cannot execute the **Find** command if a source tag is deleted but its replication configuration still exists in the Historian.

The replicated tags are not listed in the context menu if:

- The replicated tags are not committed in the Historian.
- The replication schedule is removed from the Historian. For example, you are connected to an Historian 10.0 server and you create a tag called 'MyTag'. 'MyTag' is replicated as a simple tag called 'MyServer.MyTag'. When you execute the **Find - replicated tag** command, the 'MyServer.MyTag' tag is shown. When you execute the **Find - source tag** command, the 'MyTag' tag is shown. At this instance, if the replication link between 'MyTag' and 'MyServer.MyTag' is removed and if you execute the **Find - replicated tag** command, the 'MyServer.MyTag' tag is not shown in the list of replicated tags. However, if you execute the **Find - source tag** command, the 'MyTag' tag is shown as 'MyTag'. If 'MyServer.MyTag' is the only replicated tag, 'MyTag' is considered as a normal tag.

  The above scenario holds true if the entire replication schedule is removed in the Historian. If only one replication is removed, the list shows the remaining replicated tags.

Replacing a Source Tag or Replicated Tag

You can replace a source tag with its replicated tag or a replicated tag with its source tag in the active trend chart.

To replace a source tag or replicated tag

1. Select a tag in the Tag List.
2. If the selected tag is a source tag, do the following:
   - Right-click the selected tag, point to **Replace with replicated tag**, and then click the tag that you want to replace with.

     The source tag is replaced by the selected replicated tag in the Tag List and in the active trend chart without changing the pen configuration such as the pen color, width, or retrieval mode.
3. If the selected tag is a replicated tag, do the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Right-click the selected tag and click **Replace with source tag**.

  The replicated tag is replaced by the source tag in the Tag List and in the active trend chart without changing the pen configuration such as the pen color, width, or retrieval mode.

The **Replace** command is not available if a normal tag that is neither a source tag nor a replicated tag is selected in the Tag Picker.

Note: You cannot execute the **Replace** command if a source tag is deleted but its replication configuration still exists in the Historian.

The following happens if you execute the **Replace** command on a tag configured as an x-axis tag in a XY Scatter Plot.

- If a tag is used as the x-axis tag, the **Replace with source tag** or **Replace with replicated tag** command replaces the x-axis tag reference. For example, a tag called 'MyTagY' is configured to have 'MyTagX' as the x-axis tag. Replacing 'MyTagX' with 'MyTagX2' changes the configuration of the 'MyTagY' and the x-axis tag is replaced by 'MyTagX2'.

The replicated tags are not listed in the context menu if:

- The replicated tags are not committed in the Historian.
- The replication schedule is removed from the Historian. For example, you are connected to a Historian 10.0 server and you create a tag called 'MyTag'. 'MyTag' is replicated as a simple tag called 'MyServer.MyTag'. When you execute the **Replace with replicated tag** command, the 'MyServer.MyTag' tag is shown. When you execute the **Replace with source tag** command, the 'MyTag' tag is shown. At this instance, if the replication link between 'MyTag' and 'MyServer.MyTag' is removed and if you execute the **Replace with replicated tag** command, the 'MyServer.MyTag' tag is not shown in the list of replicated tags.

  However, if you execute the **Replace with source tag** command, the 'MyTag' tag is shown as 'MyTag'. If 'MyServer.MyTag' is the only replicated tag, 'MyTag' is considered as a normal tag.

  The above scenario holds true if the entire replication schedule is removed in the Historian. If only one replication is removed, the list shows the remaining replicated tags.

Viewing Tag Definition Information

Use the Tag List to view information for the tags that you have added to the trend chart.

By default, the Minimum Raw, Maximum Raw, Precision, Format, and Date Created columns are not visible in the Trend application's Tag List. To show the columns, drag the right column separator of the Time Offset column heading to the right. Repeat this step until all columns are visible.

To view tag information

- In the Tag List pane, scroll to the name of the tag for which you want to view definition information.

Tag Name	Description	Number	Server	Display Units	Original Units	Color	Minimum	Maximum	IO Address	Time Offset	Source Tag	Source Server	Value at X1	Value at X2
SysTimeHour	System Time : Hours	1		Hour	Hour		0.000	23.000		0:00:00.000			19.000	19.000
SysTimeMn	System Time : Minutes	2		Minute	Minute		0.00	59.00		0:00:00.000			15.00	18.00
SysTimeSec	System Time : Seconds	3		Second	Second		0.00	59.00		0:00:00.000			17.00	47.00

  The grid shows the following information:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description
Tag Name	The name of the tag within the Historian. If the data values are coming from the application server, the attribute reference is shown as the tag name. For application server attributes, you can also choose to show the hierarchical name along with the attribute reference.
Description	The description for the tag.
Number	The serial number of the tag.
Server	The name of the server that contains the tag.
Display Units	The displayed unit of measure of the tag value. Examples: seconds, psi, and lbs. By default, this is the same as the "Original Units", but when licensed and configured, you can select different display units.
Original Units	The configured unit of measure of the tag value. Examples: seconds, psi, and lbs.
Color	The line color of the tag.
Minimum	The minimum value of the raw acquired value.
Maximum	The maximum value of the raw acquired value.
IO Address	The complete I/O address for the tag, including I/O Server name, application, topic, and item name.
Time Offset	The amount of time that the trend curve of the currently selected tag will be shifted from the actual time.
Source Tag	The name of the source tag that provides the source data for the replicated tag.
Source Server	The name of the server that contains the source tag.
Value at X1	The value that is displayed at Timer axis Cursor 1.
Value at X2	The value that is displayed at Timer axis Cursor 2.

- Select or clear the check box to show or hide the tag in the trend chart. This allows you to hide a tag without removing it from the list of tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing the Hierarchical Name in a Trend

A hierarchical name represents the fully qualified name of a contained object within Application Server. For more information, see Integration with AVEVA Application Server.

To view hierarchical names in a trend

- Do one of the following:
  - On the **View** menu, from the **Canvas** group select **Hierarchical Name.**
  - Right-click in the Tag Picker and click **Use hierarchical name**.

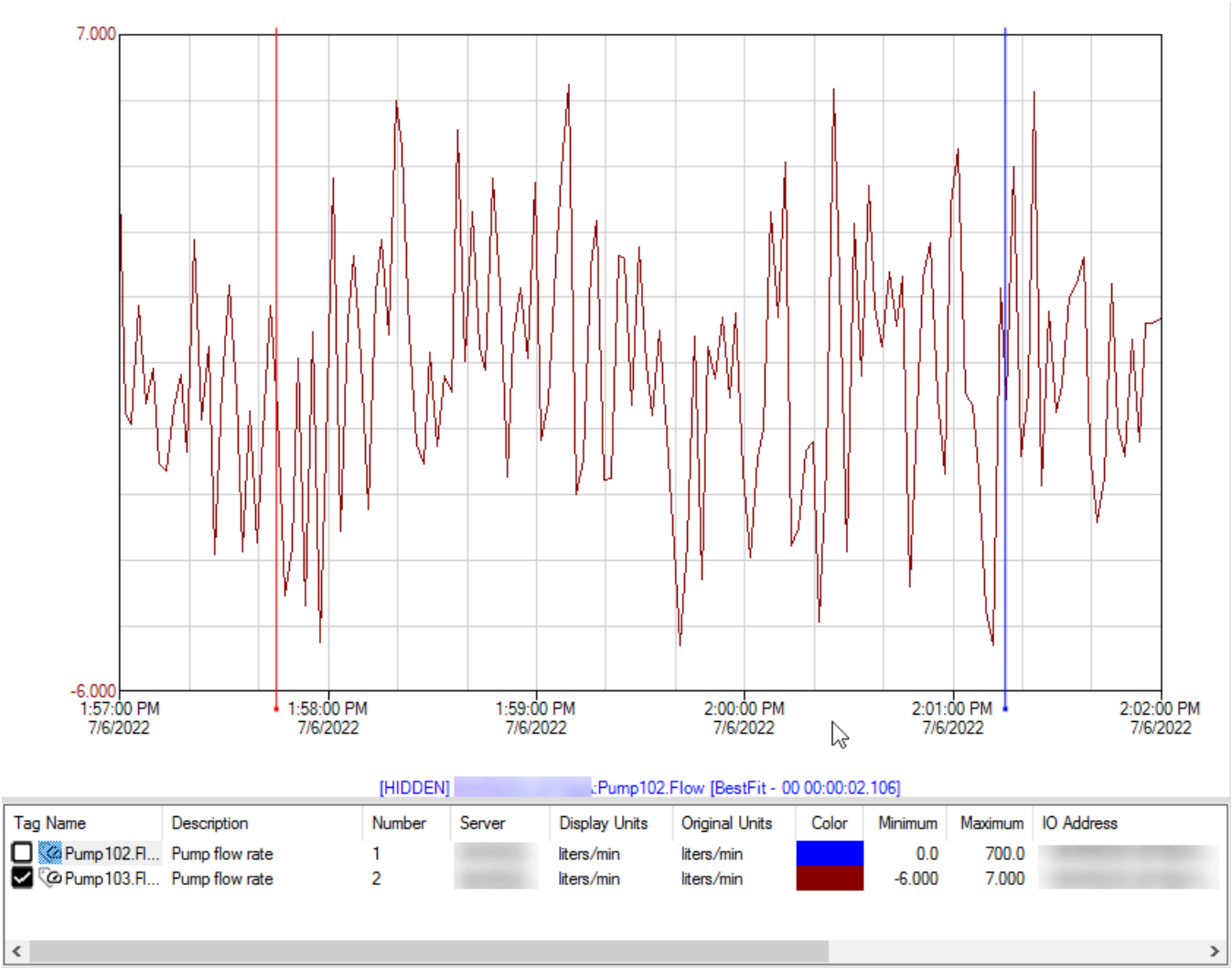
  The Trend application shows the hierarchical name instead of the tag name. For example, the Tag List, **Data Log** dialog box, and the Trend chart area show hierarchical names.

Viewing Data in the Trend Chart

This section describes how to use Trend to show historical and live data for trends, as well as how to manipulate the trend display.

Information for individual tags appears in the Tag List below the chart. The name of the currently selected tag, its retrieval mode and resolution or cycle count (if applicable) appear along the bottom of the chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Refreshing the Trend Chart

You can refresh the Trend Chart to see the most recent information.

To refresh the chart

1. Set the **Update to Current Time** toolbar button depending on whether you want the trend's time period to be updated to the current time when refreshing the chart. If the button is not available, the time period remains the same as before the update. If the button is available, the time period is updated so that it ends with the current time.
2. Do any of the following:
   - Click the **Refresh** toolbar button.
   - Right-click the chart, and select **Refresh**.

     The trend chart is updated with current data from the database.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Deleting a Tag

You can delete a tag. Deleting a tag removes it from the chart.

To delete a tag from the trend

1. In the Tag List, select the tag to delete.
2. Delete the tag by doing any of the following:
   - On the **Chart** menu, click **Delete Tag**. At the prompt, click **Yes**.
   - Right-click on a tag in the Tag List. In the menu that appears, click **Delete**.
   - Press the DELETE key on your keyboard.

Configuring Trend Options for a Tag

You can configure trend options for one or more tags in the Tag List window. Trend options include the appearance of the trend pen, its target region, and the retrieval mode used to retrieve data for the tag.

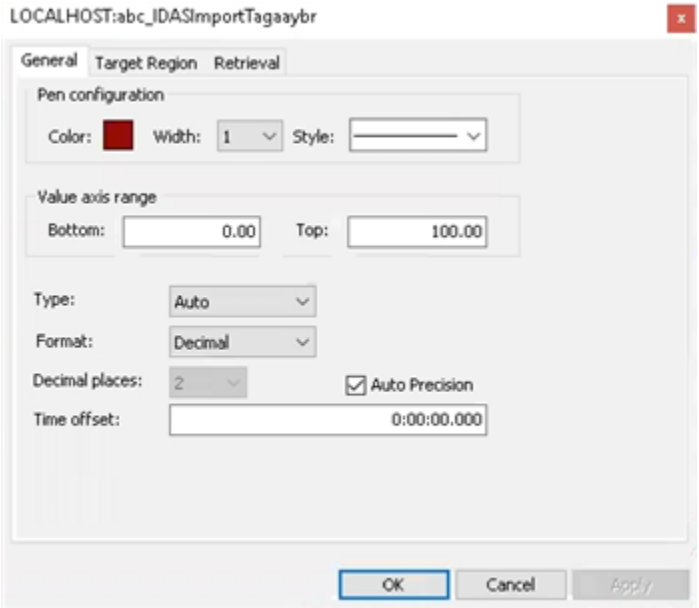
Configuring Display Options

You can configure the pen style, value axis scale, value display options, and time offset for each tag in the chart.

To configure display options for a tag

1. In the Tag List pane, do one of the following:
   - Right-click on the name of the tag and then click **Configure**.
   - Double-click on the name of the tag.

     The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.



2. In the **Pen Configuration** area, configure the appearance of the curve for the selected tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

• Color	The line color of the tag. Click the colored square to select the color from a palette or define a custom color.
• Width	The thickness of the trend curve.
• Style	The style of the trend curve. For example, a solid or dashed line.

3. In the **Value axis range** area, configure the limits for the value axis on the chart.

• Top	The maximum axis value for the tag, in engineering units.
• Bottom	The minimum axis value for the tag, in engineering units.

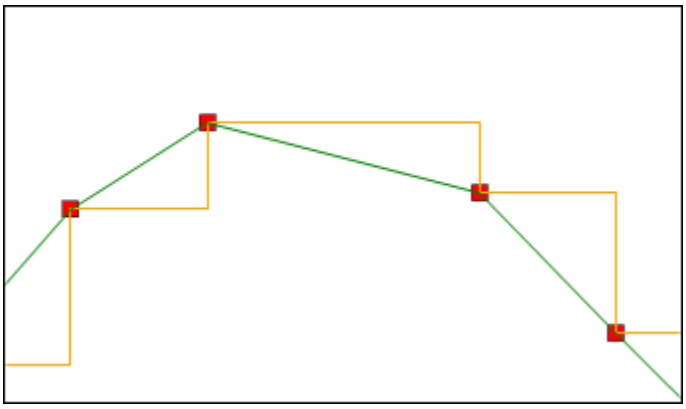
4. In the **Type** list, select the type of trend curve to draw. Options are **Auto**, **Line**, **Step line**, and **Point**.

   A line curve is best suited for continuously-changing analog data. A step-line curve is best suited for discrete data and for analog data that is not continuous. By default, the line curve trend is selected for the summary tags.

   When you select the **Auto** option, the curve type is determined as follows:

   - For tags retrieved from a version 9.0 or a later Historian, the type is based on the tag's effective interpolation setting, which may be specified in the Trend application's retrieval settings or on the Historian. Tags that use stair-step interpolation are trended as a step line, and tags that use linear interpolation are trended as a line.
   - For tags retrieved from earlier Historian versions, the curve type is based on the tag type: step line for integer tags, and line for real tags.

     The following illustration shows the same data drawn using each type of curve. The line curve is shown in green, the step line curve is shown in orange, and the point curve is shown in red.



5. From the **Format** list, select how the values for the tag appear, either in decimal format or scientific format.
6. From the **Decimal places** list, select the number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags. Valid values are 0 through 15.
7. Select **Auto Precision** to have the number of decimal places set automatically based on the value range.

   Note: Auto Precision is enabled by default. When auto precision is enabled, the **Decimal places** field is read-only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. Configure either the time offset or start time for the data.

   For more information on time offsets, see Using Absolute or Relative Times.

• Time offset	Shown only for absolute mode. The amount of time that the trend curve of the currently selected tag will be shifted from the actual time. For information on the offset notation, see Time Offset Formats.
• Start time	Shown only for relative mode. The starting time stamp for the tag data in the chart.

9. Click **OK**.

Configuring Displayed Engineering Units for a Tag

For each analog tag in a trend, you can configure the chart to display the data in related units different from the tag's base unit.

Note: This feature must be licensed and correctly configured before you can change display units. Refer to the *AVEVA Historian Administration Guide* for more information on configuring engineering units, and the engineering units catalog.

To configure display units for an analog trend tag

1. In the Tag List pane, locate and select the tag which you want displayed in different units.

   The **Display Units** column indicates the units used in the chart currently. The **Original Units** column indicates the tag's configured units.
2. Select the unit in the **Display Units** column to open a dropdown list of the tag's related units. This list is composed of all the engineering units assigned to the same dimension as the original units, as configured in the engineering units catalog.
3. Select an engineering unit from the list. The chart will refresh and the tag data will display in the selected unit.

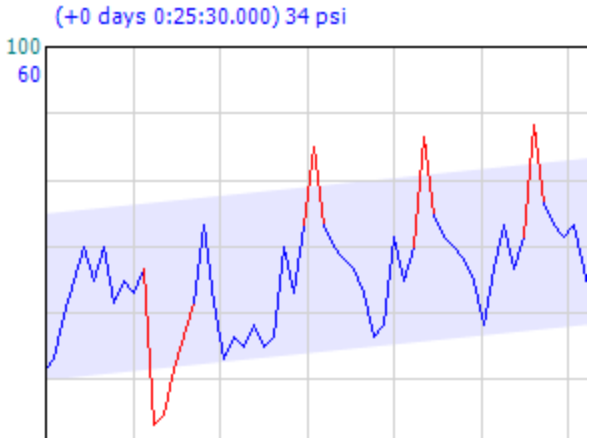
   Note: You may need to scale the tag after conversion if the data displays outside the current chart boundaries.

Defining a Target Region for a Tag

For each analog, discrete, or summary tag in a trend, you can define a "target region." The target region is a highlighted area of the chart into which tag values should fall during normal operation. Values that fall outside these normal limits can be highlighted in a special color, making it easy to detect them.

The following chart shows an example of a target region (the area tinted in blue). The red spikes indicate limit excursions:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



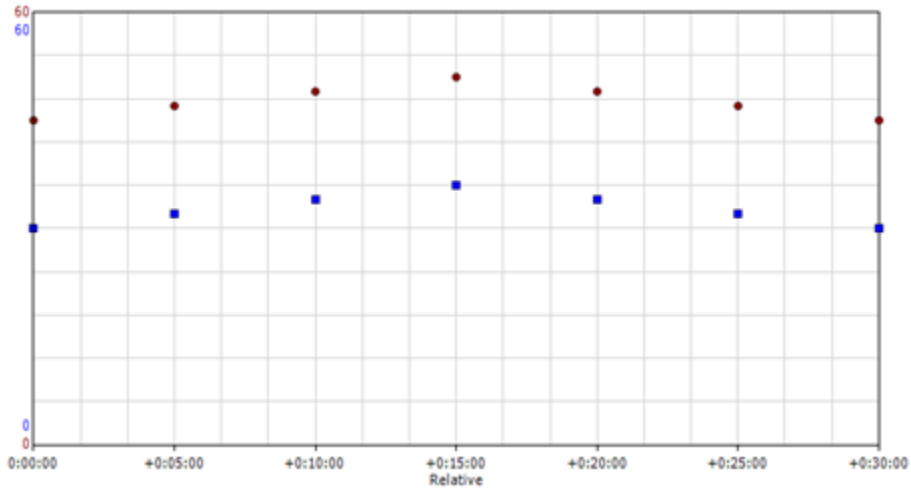
In a regular trend, you can only use target regions in relative time mode. A target region is defined by "region items," that is, pairs of high and low values at specific time offsets.

To determine the target region, a boundary is drawn that connects all of the high values, and another boundary that connects all of the low values. The area between these two boundaries constitutes the target region.

For example, assume that you define the following region items:

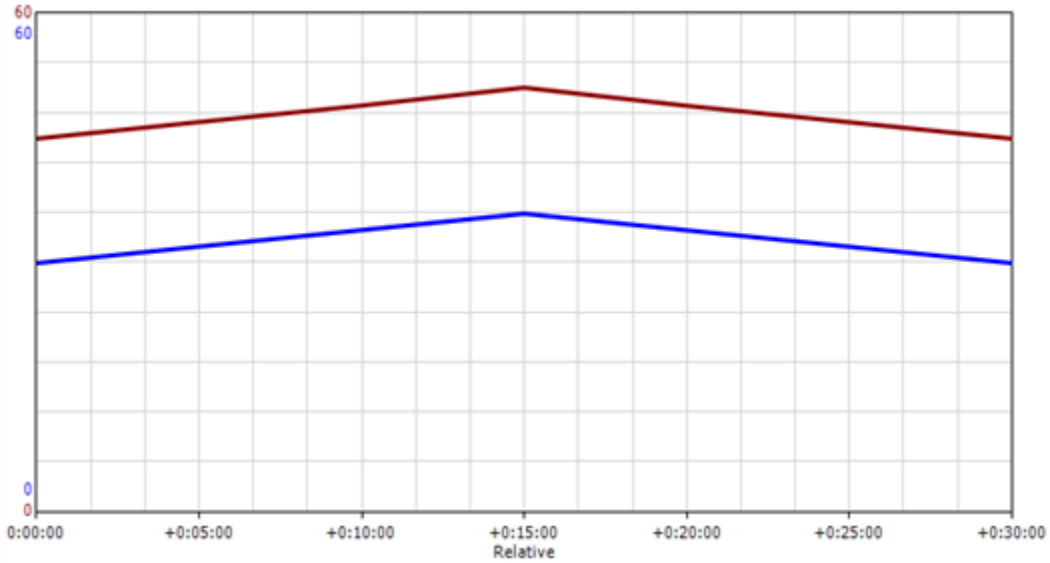
Time	Low	High
0:00:00.00	30	45
+0:05:00.00	32	47
+0:10:00.00	34	49
+0:15:00.00	36	51
+0:20:00.00	34	49
+0:25:00.00	32	47
+0:30:00.00	30	45

On a trend chart in relative time mode, these points look like the following:

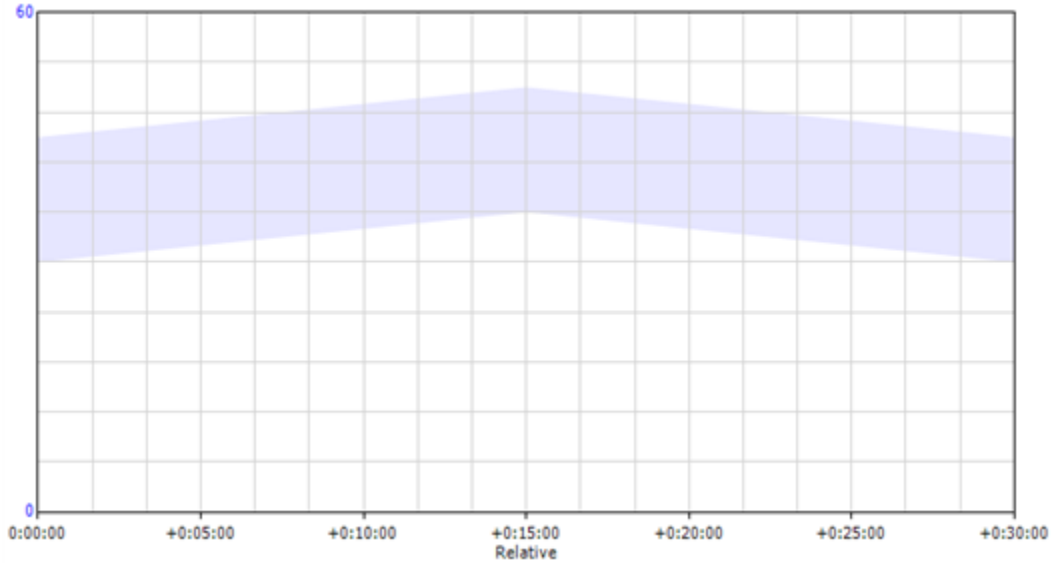


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The connecting boundaries look like the following:



The area between these boundaries constitutes the target region:



The target region has the same color as the tag's trend curve. It is only shown when the tag is currently selected in the Tag List. Also, target regions are not visible if you are using stacked traces.

You can define and save target regions separately for each tag. Target regions are saved in the trend file. If you delete a tag from the trend, its target region is deleted as well. To use the same target region for multiple tags or in different trends, either copy and paste it or create a CSV file with the target region data that you can load for each tag. For more information, see the procedures below.

The following section explains how to set up a target region for a tag. To specify whether and how values outside the target region should be highlighted, and to set the target region's opacity, see Configuring Target Region Properties.

To configure a target region for a trend tag

1. In the Tag List pane, do one of the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Right-click on the name of the tag and then click **Configure**.
- Double-click on the name of the tag.

  The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.

2. Click the **Target Regions** tab.
3. Edit region items by using any of the following procedures:
   - Edit region items manually
   - Load region items from a CSV file
   - Paste region items in CSV format from another application
   - Paste region items from another tag
   - Enable or disable a tag's target region
4. Click **OK**. The target region appears in the trend chart, spanning the time period that you defined using the region items' time offsets.

To edit region items manually

- Do any of the following:
  - To add an item, click **Add**. A new item appears in the list. Double-click each cell to edit it.
  - To delete an item, right-click it and click **Cut**.
  - To delete all items, click **Clear All**.
  - To move an item up or down in the list, select it and then click the up or down arrows.

To load region items from a CSV file

1. Click **Load file**. A standard **Open** dialog box appears.
2. Select the file you want and click **Open**. The list is populated with the entries from the CSV file.

   Note the following format requirements for the CSV file:
   - Each row must contain a region item composed of three values: the time offset, the low value, and the high value. The format of the time offset is the same as for time offsets in relative time mode. For more information, see Time Offset Formats.
   - The Trend application tries to determine the CSV delimiter and the decimal and time separators automatically. If one of the values contains a delimiter or separator character, that value must be enclosed in double quotation marks.

To paste region items in CSV format from another application

1. In the other application, open the file that contains the region items in CSV format. Copy the CSV data to the clipboard.
2. In the Trend application, right-click the list of region items, and then click **Paste**. The list is populated with the pasted entries.

   The format of the copied data must follow the same conventions as content in CSV files. For more information, see the previous section.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To paste region items from another tag

1. In the Tag List pane, double-click the name of the tag whose region items you want to copy. The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.
2. Click the **Target Region** tab.
3. In the list of region items, select the item(s) you want to copy. To select multiple items, hold down SHIFT and/or CTRL while clicking.
4. Right-click the selected items and click **Copy**.
5. Click **OK** to close the dialog box. Repeat steps 1 and 2 for the tag where you want to paste the region items.
6. Right-click the list of region items and click **Paste**. The list is populated with the pasted entries.

To enable or disable a tag's target region

1. In the Tag List pane, double-click the name of the tag. The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.
2. Click the **Target Region** tab. Select or clear the **Show target region** check box to enable or disable the tag's target region.

   Note: Regardless of this setting, the target region for a tag is only shown when that tag is currently selected in the Tag List.
3. Click **OK**.

Configuring Retrieval Options for a Tag

You can configure retrieval options separately for each tag in a trend. Tags can either use the retrieval style specified in the trend options, a different predefined retrieval style, or custom retrieval settings.

Most of the retrieval settings that you configure here only apply if you are retrieving data from a Historian with a version of 9.0 or later.

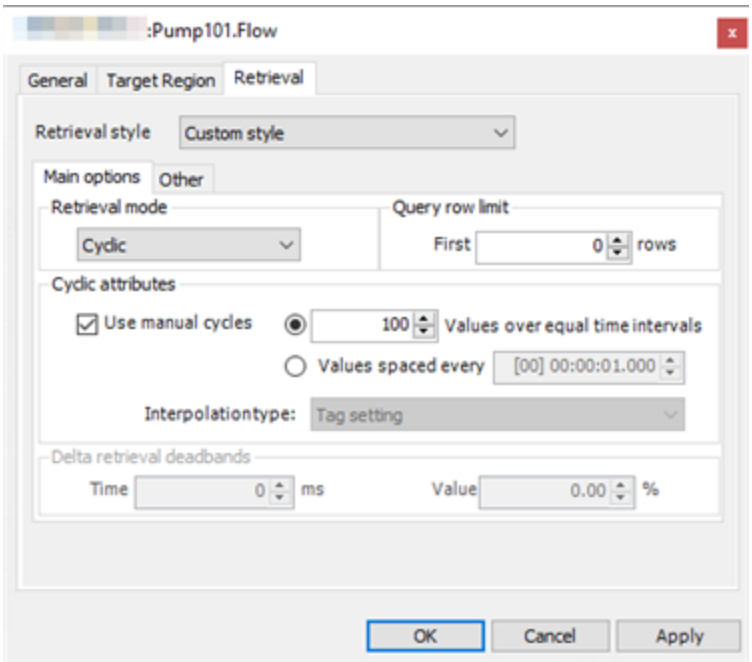
If you are using an earlier Historian version, see Configuring Other Options and Working with Retrieval Styles for details.

To configure retrieval options for a tag

1. In the Tag List pane, do one of the following:
   - Right-click on the name of the tag and then click **Configure**.
   - Double-click on the name of the tag.

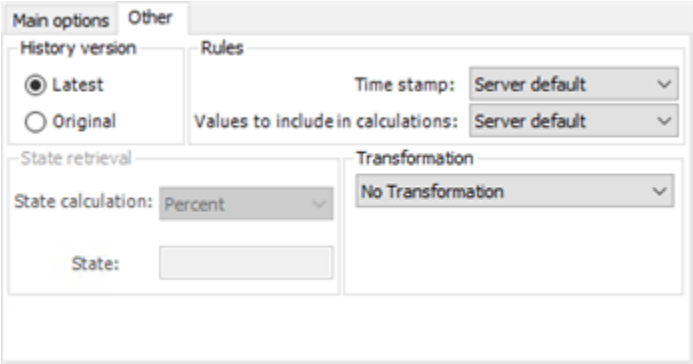
     The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.
2. Click the **Retrieval** tab.
3. Do one of the following:
   - To have the tag use the same retrieval settings as specified in the trend options, click **Style selected at option level** in the **Retrieval style** list. This is the default setting when you add a tag to a trend.
   - To use a predefined retrieval style, click its name in the **Retrieval style** list. For more information on retrieval styles, see Working with Retrieval Styles.
   - To use custom retrieval settings, click **Custom style** in the **Retrieval style** list.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. Specify any additional settings required.
   - If you are using custom retrieval settings, select a retrieval mode and specify all the settings that are relevant to it.
   - If you are using one of the predefined styles, you can edit all settings that are not covered by the style definition. For information on which settings are covered by style definitions, see Working with Retrieval Styles. Because a style definition can contain multiple sets of retrieval settings with different retrieval modes, some of the settings available for editing here may turn out to be irrelevant for the retrieval mode that actually gets used for a given query. However, because there is no way to know in advance which retrieval mode will be used, the settings are still available for editing.

     For more information on the various retrieval options, see Understanding Retrieval Options.
5. Specify any additional settings required on the **Other** tab.



   a. In the **History version** area, click **Latest** or **Original** to overwrite the values of a stored tag. For more information on History version, see History Version (wwVersion).
   b. In the **Rules** area, do the following:
      a. In the **Timestamp** list, click the required timestamp value. For more information on the Time stamp rule, see Timestamp Rule (wwTimestampRule).
      b. In the **Values to include in calculations** list, click the data values that you want to include in the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

result. You can include the following quality rules:

Good and uncertain quality: To include data values with uncertain quality in calculations.

Good quality: To exclude data values with uncertain quality from calculations.

Optimistic quality: To use the optimistic quality when the data values are missing. This option can be configured at either the tag level or the application level.

Server default: To use the default quality rule specified at the Historian level.

For more information on each option, see Quality Rule (wwQualityRule).

c. In the **State retrieval** area, do the following:
   a. In the **State calculation** list, click the state calculation.
   b. In the **State** box, specify the value of the state. For example, you can specify either a open or close state for the SteamValve tag.

   Note: The state calculation settings are applicable only to ValueState and RoundTrip retrieval modes.

   For more information on State calculation, see State Calculation (wwStateCalc).
d. In the **Transformation** list, click the transformations to be applied to the result. You can include the following transformations:
   a. **No Transformation**: If the query does not specify the filter element or if the value is not overridden for the filter element.
   b. **Remove outliers**: To remove outliers from a set of analog points.
   c. **Convert analog values to discrete**: To convert value streams for any analog tag to discrete value streams.
   d. **Snap to base value**: To force values in a well-defined range around one or more base values to "snap to" that base value. For more information on Transformation, see Analog Value Filtering (wwFilter).

Editing the Expression for an Expression Tag

You can edit the expression and alias of any expression tags in a trend. See Expressions (wwExpression) for information about the syntax and usage for expressions.

To edit the expression for an expression tag

1. In the Tag List pane, do one of the following:
   - Right-click on the name of the tag and then click **Configure**.
   - Double-click on the name of the tag.

   The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.
2. Select the **Edit Expression** tab. The fields in the tab are:
   - **Expression:** Edit the expression.
   - **Alias:** Edit the alias.
3. Select **OK**.

Scrolling through Tags in a Trend

You can change which tag is currently selected by using toolbar buttons. When using single tag mode, this allows you to "scroll" through tags without having the Tag List visible.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To change the currently selected tag

- From the **Chart** menu, in the **Tag** group, click either **Next** or **Previous** to scroll to the tag that you want.

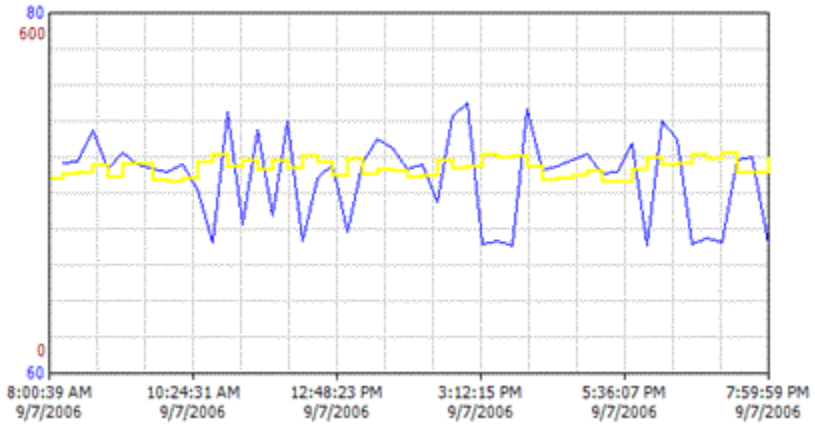
Highlighting a Tag

You can select and highlight a tag in the chart. To remove the highlighting, follow the same procedure so that no check mark or highlighting appears.

To highlight a tag

1. Select the tag that you want in the Tag List.
2. From the **Chart** menu, in the **Tag** group, select **Highlight**.

   The tag is highlighted in the chart.



Showing a Single Tag in the Trend

When you initially create a Tag List for a trend, all the tags are included in the display. Setting the trend to single tag mode allows you to exclude all tags but one from being displayed in the trend chart, without removing them from the Tag List.

To display a single tag in the trend

1. From the **Chart** menu, in the **Layout** group, select **Single** so that it is highlighted.

   To again view multiple tags in the chart, deselect **Single** so that it is not highlighted.

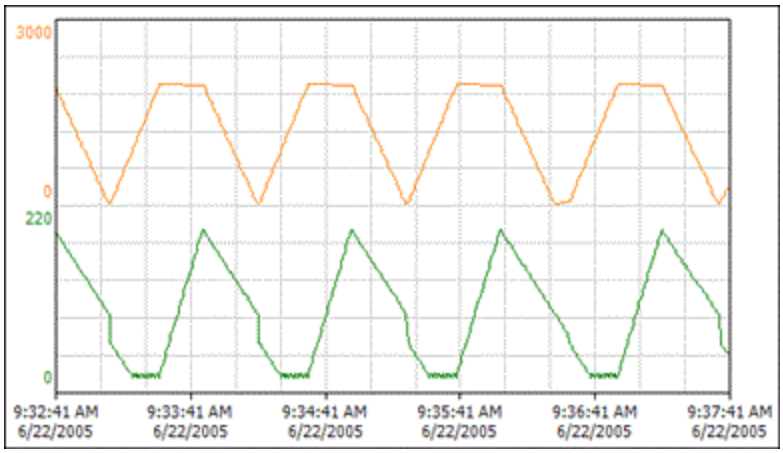
Stacking Traces

You can view individual trends, or "traces," for multiple tags in the chart by stacking them in the display.

To stack tags in the trend

1. From the **Chart** menu, in the Layout group, select **Stacked**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Showing Live Data

A trend can be configured to show live data. Live data is data that is retrieved continuously in real time for a fixed duration that is relative to the current time (for example, the last hour).

When retrieving live data, the Trend application retrieves data incrementally with every update. For example, if you set the update rate to ten seconds, then every ten seconds, the Trend application retrieves data for the last ten seconds and updates the chart with that data. Additionally, it periodically retrieves data for the entire chart time span to refresh the entire chart. You can specify both the update rate and the refresh interval for the entire chart. For more information, see Configuring General Properties.

If the connection to the Historian is lost while retrieving live data, any data that was retrieved up to that point is still shown on the chart until the next full refresh occurs. If the Historian is still unavailable at that time, the old data is cleared from the trend chart.

Notes: When retrieving live data, the time stamp rule for data retrieval is forced to "End." For more information on this setting, see Timestamp Rule (wwTimestampRule).

While the chart is displaying live data, if the trended data includes points outside of the visible plot area, any affected tags are automatically scaled to fit.

To display "live" data

1. Click the **Update to Current Time** toolbar button so that it is highlighted.
2. In the Duration list of the Time toolbar, click a duration or type one manually.
3. Click the **Live Mode** toolbar button.

To display "static" data

- Click the **Stop Live Mode** toolbar button.

Showing Historical Data in "Replay" Mode

When you "replay" historical data, the data is continuously plotted on the chart, starting with the start date. By default, the "replay" mode uses real-time speed. For example, if you set the chart to update every second, the start time advances one second with each update.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

You can accelerate or slow down the playback by specifying a "playback speed." For example, if you select a playback speed of 2 x (that is, twice the normal speed) and set the chart to update every second, the start time advances two seconds with each update, as compared to one second for a playback speed of 1 x (normal speed) or half a second for a playback speed of 1/2 x (half the normal speed).

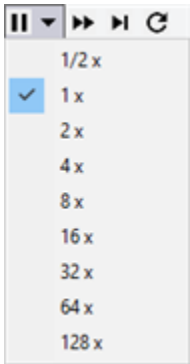
To "replay" historical data

1. Configure a query with a time period whose end time is earlier than the current time.
2. Make sure the **Update to Current Time** toolbar button is *not* highlighted. If it is highlighted, click it so that it isn't highlighted anymore.
3. Click the **Live Mode** toolbar button.

   The trend curve is dynamically drawn on the chart. The **Replay Mode** icon appears at the top center of the chart to indicate replay mode.

To "replay" historical data at a slower or faster speed

1. Configure a query with a time period whose end time is earlier than the current time.
2. Make sure the **Update to Current Time** toolbar button is *not* highlighted. If it is highlighted, click it so that it isn't highlighted anymore.
3. Click the downward arrow next to the Live Mode toolbar button. A list of playback speeds appears.



4. Click the playback speed you want to use.
5. Click the **Live Mode** toolbar button.

   The trend curve is dynamically drawn on the chart at the specified speed. The **Replay Mode** icon and the playback speed appear at the top center of the chart to indicate replay mode. To change the speed while replay mode is active, repeat steps 1 and 2.

Note: When you replay historical data at an accelerated speed, eventually the time period "catches up" with the current time. When that happens, the speed is automatically reset to normal, and the trend effectively goes into live mode. For more information, see Showing Live Data.

Configuring Predictive Data Retrieval

You can define a predictive data retrieval option in the Historian Client. Predictive retrieval means the Historian Client will interpolate future data trend based on the existing data received.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For example, suppose you were using the Historian Client to view a trend for a tank level at your plant. With the data already stored, you can see the tank level values since operations started this morning, but now you would like to know what the value is likely to be in an hour.

Note: Predictive retrieval is supported for the best-fit, full, and delta retrieval modes. If this option is applied to other retrieval modes, it is ignored.

By using the Pan Right (">>") toolbar button, you can advance the trend and, with the Historian Client's new predictive data retrieval feature, you can see a prediction for that future value based on current data. Historian Client interpolates the trend based on the last value was received.

Note: When you save the trend to a CSV file, you can also see predictive data in an Excel Workbook. See Saving Trend Data to a .CSV File for more information.

Using the Transformation Option at application level

1. From the **Tools** menu, click **Options**.
2. In the **Options** dialog box, click the **Other** tab.
3. From the **Transformation** dropdown, choose **Extrapolate using linear regression**.

Using the Transformation Option at tag level

1. From the **Trend Tag List** grid, double-click a tag.

   The **Tag Properties** for that tag display.
2. From the **Retrieval** tab, click the **Other** tab.
3. From the **Transformation** dropdown, choose **Extrapolate using linear regression**.

Scaling Tags

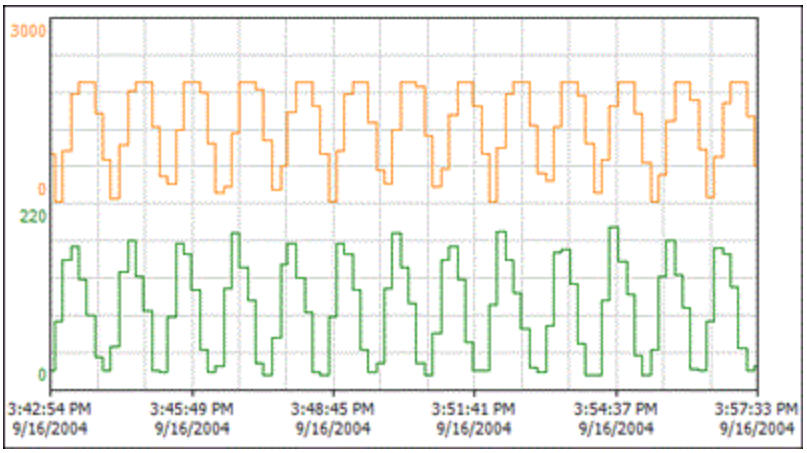
The scale is the minimum and maximum range of values for the tag. Each tag has its own scale, which is usually quite different from other tags in the chart. Scales for tags on the chart are always displayed along the value axis.

Only discrete, analog, and summary tags can be scaled; event and string tags cannot be scaled. For discrete tags, the message associated with the 1 value is used as the maximum scale value, and the message associated with the 0 value is used as the minimum scale value.

In the following chart, two tags are trended in stacked mode. The scale for the ReactLevel tag is from 0 to 3000. The scale for the other tag, ReactTemp, is from 0 to 220.

The minimum and maximum values of each scale appear on the value axis.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The initial scale of a tag is determined by its Min/Max EU settings in the Historian. To adjust tag scales, you have two options:

- Edit the value axis range individually for each tag. For more information, see Configuring Trend Options for a Tag.
- Use the scaling commands to adjust the scale of single tag or all tags. For more information, see Scaling Tags Up or Down and subsequent sections.

You can also change the way in which scale values appear on the value axis. The following sections describe the available options.

Showing No Scales on the Value Axis

You can configure the chart to show no chart label, X and Y axes scales and cursor information. This makes the entire chart area margins smaller and you get more area to plot and view the trend chart.

To show no scales on the value axis

- From the **View** menu, in the **Scale** group, select **None**.

  The following chart is configured to show no chart label, X and Y axes scales and cursor information. Stacked mode is applied.

Showing Single Scale on the Value Axis

You can configure the chart to show single value scale along the value axis.

If the chart includes multiple tags, the scale of the currently selected tag in the chart is shown along the value axis. The scale label color matches the pen color of the selected tag. As you scroll through tags in the chart, the value axis always shows the scale of the selected tag. The labels along the value axis are shown even if the current tag is hidden.

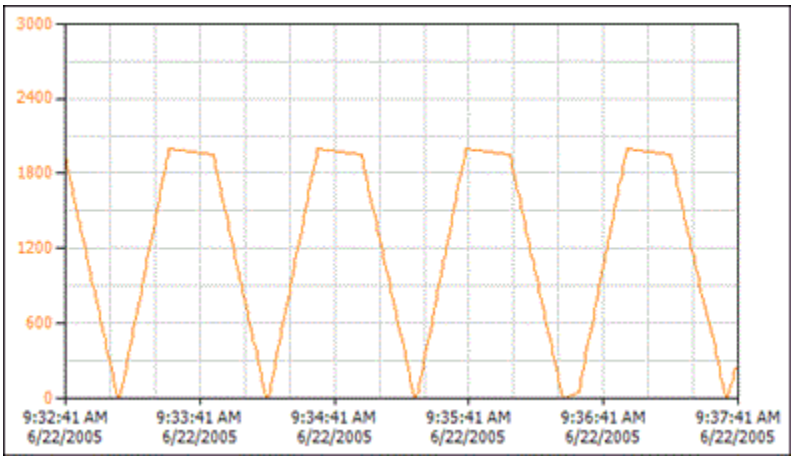
To show single scale

- From the **View** menu, in the **Scale** group, select **Single**.

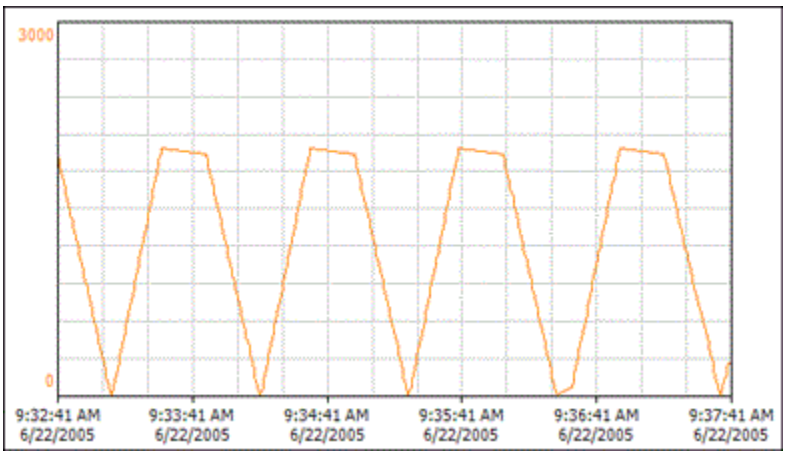
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

How a chart looks when single scale is applied depends on the number of tags in the chart and whether the chart is in stacked mode.

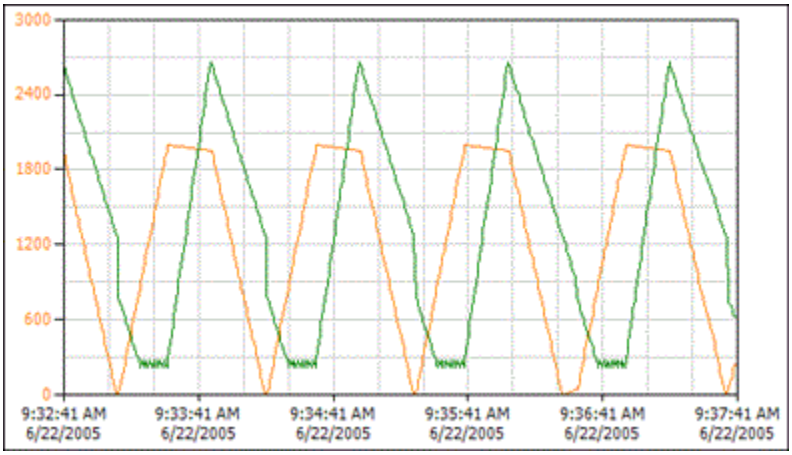
The following chart includes single tag and is configured for single scale. Stacked mode is not applied.



The following chart includes single tag and is configured for single scale. Stacked mode is applied. Only the minimum and maximum values are shown.

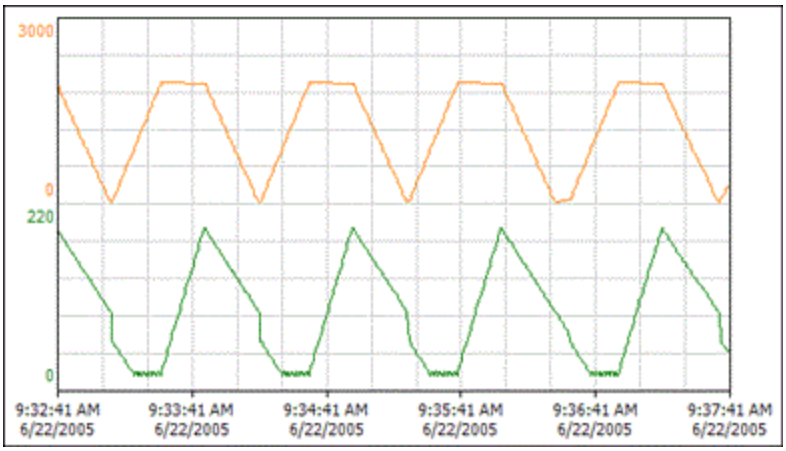


The following chart includes multiple tags and is configured for single scale. Stacked mode is not applied.



The following chart includes multiple tags and is configured for single scale. Stacked mode is applied. The minimum and maximum values for each tag are shown for the corresponding trend curve.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Showing Multiple Scales on the Value Axis

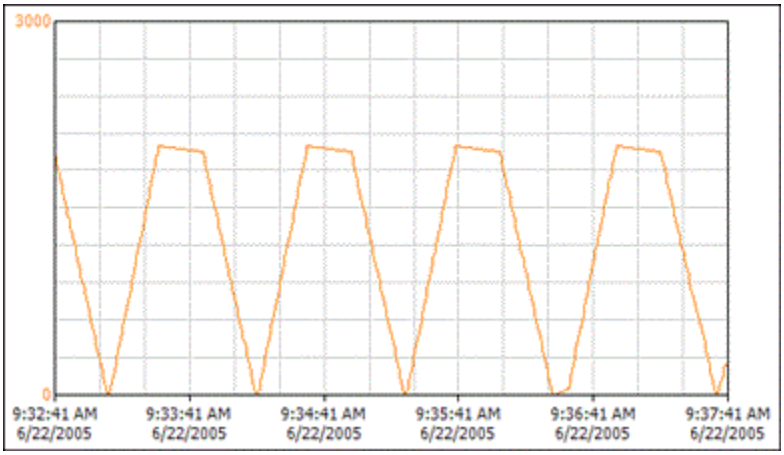
You can configure the chart to show multiple value scales. For multiple scales, only the minimum and maximum values are shown on the value axis. The scale label colors match the pen colors of the corresponding tags. The values of hidden tags are not shown.

To show multiple scales

- From the **View** menu, in the **Scale** group, select **Multiple**.

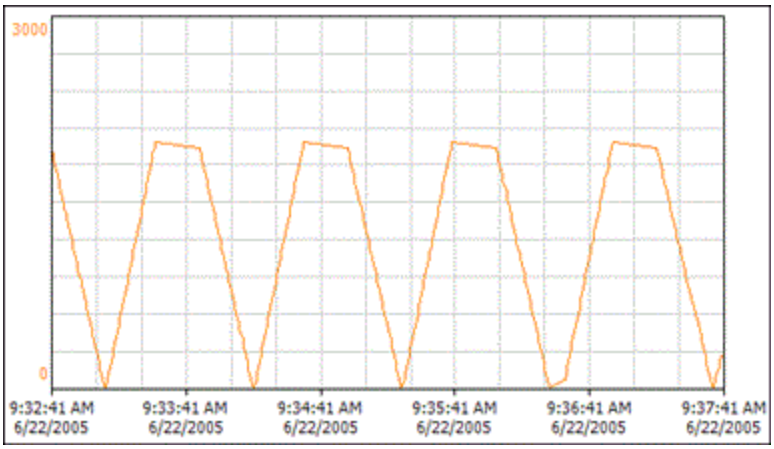
How a chart looks when multiple scales are applied depends on the number of tags in the chart and whether the chart is in stacked mode. When stacked mode is applied, there is no difference between using single scale or multiple scales.

The following chart includes single tag and is configured to use multiple scales. Stacked mode is not applied.

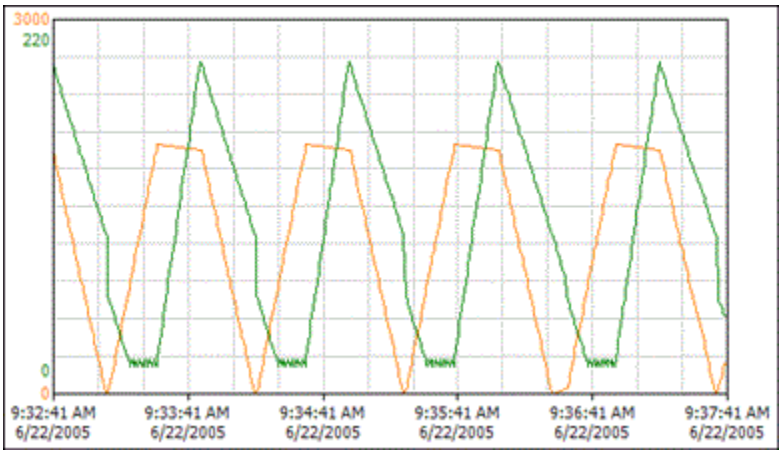


The following chart includes single tag and is configured to use multiple scales. Stacked mode is applied.

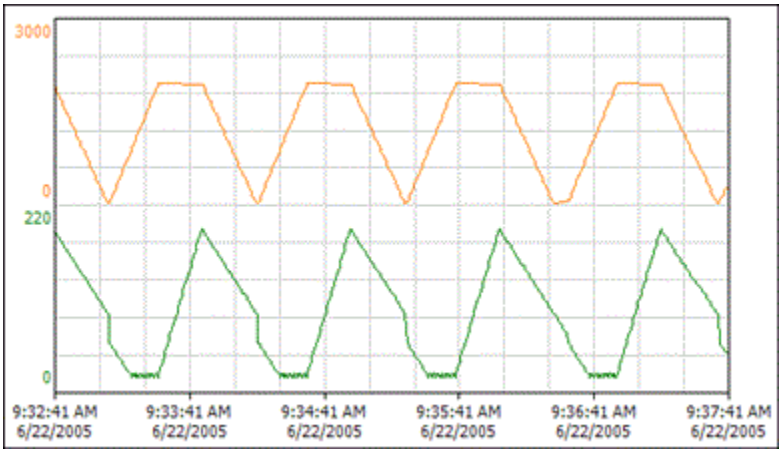
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The following chart includes multiple tags and is configured to use multiple scales. Stacked mode is not applied. The top and bottom labels show the scale for the first tag in the Tag List that is included in the chart. For the second tag in the Tag List, the scale labels are shown as second from the top and second from the bottom. As you add tags to the chart, the addition of scale labels continues to progress inward toward the middle of the chart. If there is not enough space on the chart to show all of the scale labels, then the innermost values are not shown.



The following chart includes multiple tags and is configured to use multiple scales. Stacked mode is applied.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

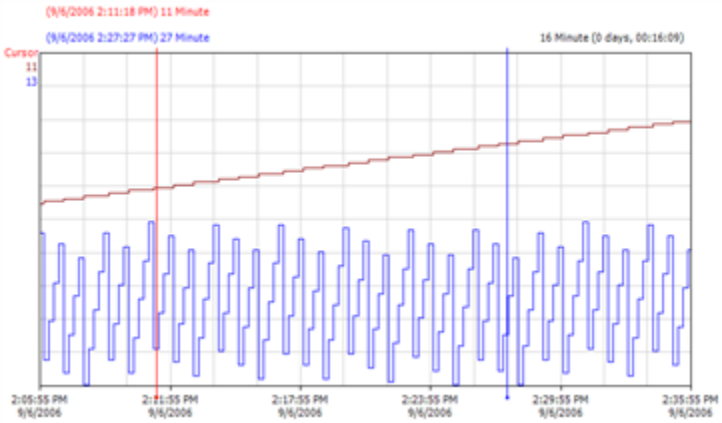
Showing Cursor Values on the Value Axis

You can configure the chart so that the value axis shows the value of each tag at the position of the first X axis cursor. The axis label colors match the pen colors of the corresponding tags. The values of hidden tags are not shown.

To show cursor values on the value axis

- From the **View** menu, in the **Data** group, select **Cursor Values**.

  The following chart is configured to show cursor values on the value axis. Stacked mode is applied.



Scaling Tags Up or Down

You can scale a single tag or all of the tags in a trend up or down. If you scale a tag down, the range of values increases by one third. For example, if the scale is 10 to 70, it becomes 0 to 80. If you scale a tag up, the range of values decreases by one fourth. For example, if the scale is 0 to 80, it becomes 10 to 70.

To scale up a single tag

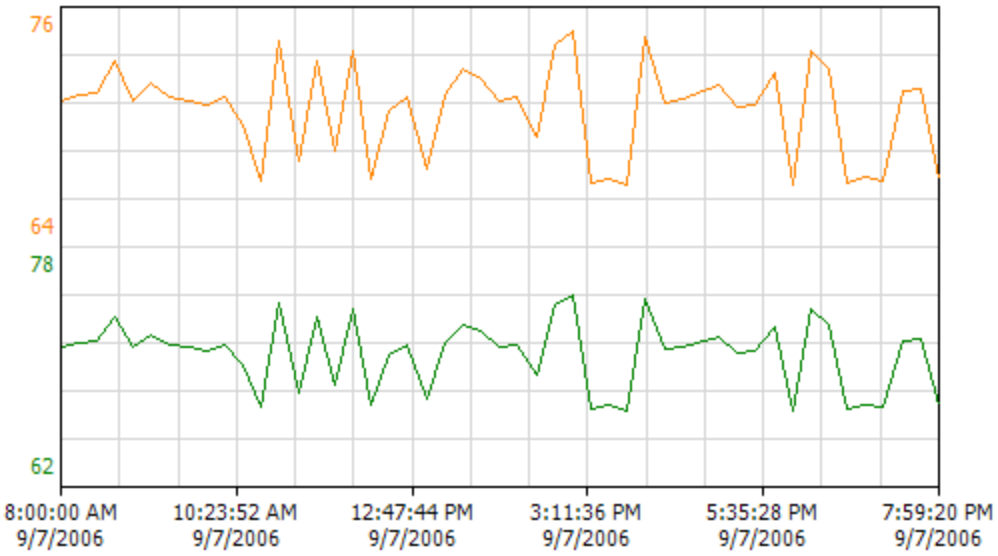
1. Select a tag from the tag list.
2. From the **Chart** menu, in the **Single Tag Scale** group, click **Scale Up**.

To scale all tags up

- From the **Chart** menu, in the **All Tags Scale** group, click **Scale Up**.

The following example shows a single tag scaled up:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



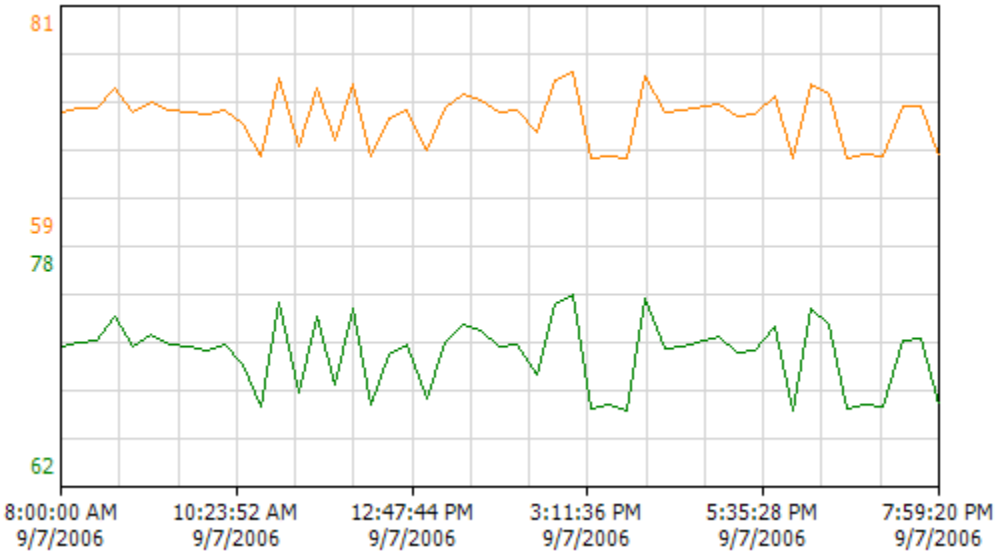
To scale down a single tag

1. Select a tag from the tag list.
2. From the **Chart** menu, in the **Single Tag Scale** group, click **Scale Down**.

To scale all tags down

- From the **Chart** menu, in the **All Tags Scale** group, click **Scale Down**.

The following example shows a single tag scaled down:



Automatically Scaling Tags

When a tag is automatically scaled, the value axis range is automatically adjusted to reflect the actual data currently being displayed for the trend. For example, if the default value axis range is 0 to 3000, and the data

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ranged from 1827 to 2059, the scale might be automatically adjusted to a range of 1800 to 2100.

The adjusted scale does not exactly match the actual minimum and maximum data values for the chart. The calculation rounds the values so as to make the chart easier to read. Also, a percentage allocation is added to the final values so that the adjusted scale fits within the boundaries of the trend chart. Therefore, the adjusted scale is a round number slightly above the actual data values.

You can automatically scale a single tag or all of the tags in a trend. For information on resetting scales back to the original default, see Returning Tags to Their Original Scale.

Tag scaling is performed automatically for a trend while it is retrieving live data. If the trended data includes points outside of the scale region, any affected tags are scaled automatically.

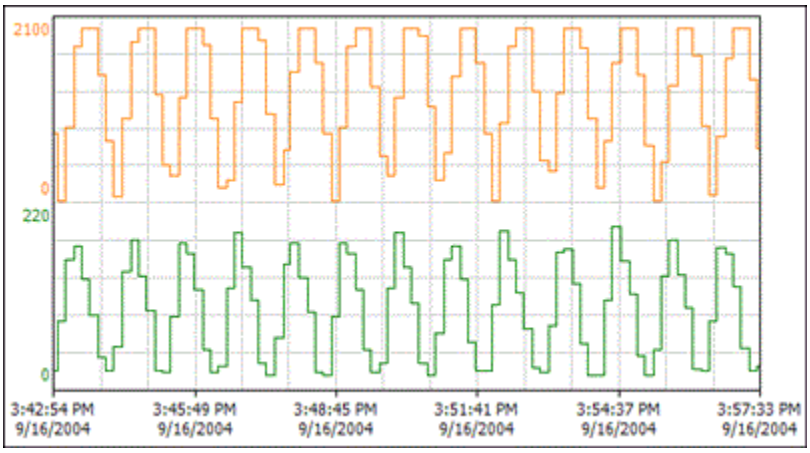
To automatically scale a single tag

1. Select a tag from the tag list.
2. From the **Chart** menu, in the Single Tag Scale group, click **Auto**.

To automatically scale all tags

- From the **Chart** menu, in the All Tags Scale group, click **Auto**.

The following example shows a tag automatically scaled:



Returning Tags to Their Original Scale

You can return the value axis scale for a single tag or all of the tags in a trend to the original scale.

To return a single tag to original scale

1. Select a tag from the tag list.
2. From the **Chart** menu, in the **Single Tag Scale** group, click **Original**.

To return all tags to original scale

- From the **Chart** menu, in the **All Tags Scale** group, click **Original**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Moving Tags Up or Down in the Chart

You can move a single tag or all of the tags in a trend up or down in the trend chart. The scale is adjusted to reflect the move.

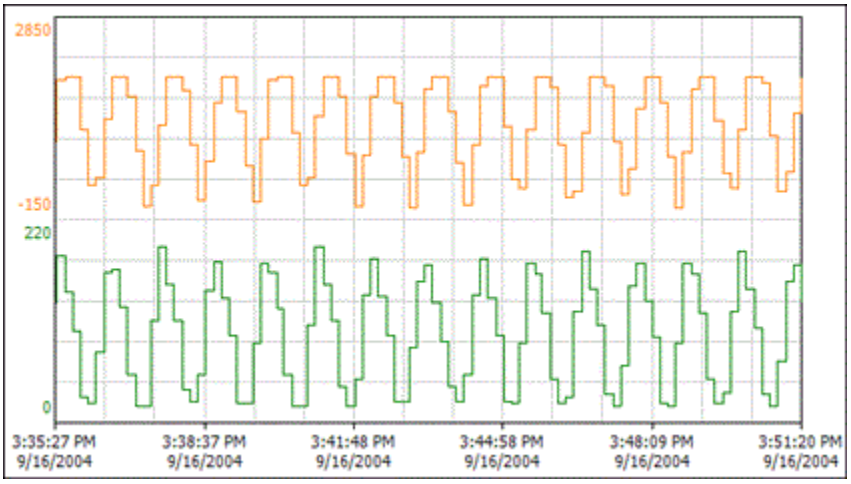
To move a single tag up

1. Select a tag from the tag list.
2. From the **Chart** menu, in the Single Tag Scale group, click **Move Up**.

To move all tags up

- From the **Chart** menu, in the **All Tags Scale** group, click **Move Up**.

The following example shows a single tag moved up in the trend chart:



Using "Rubber Band" Scaling

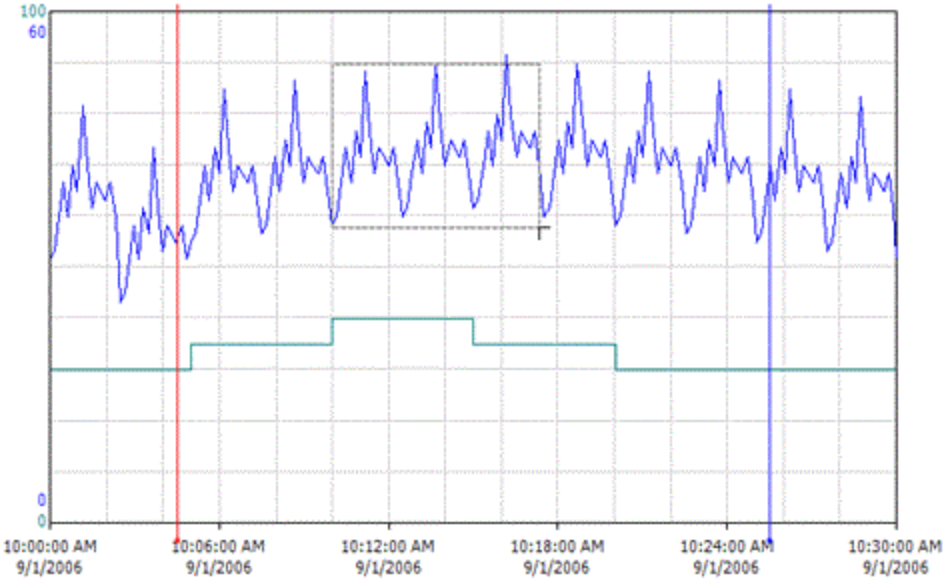
Rubber band scaling allows you to "lasso" an area of the trend chart with the mouse cursor to automatically adjust the time and value axis scales based on the area that you lassoed. If you are using stacked traces, rubber band scaling is limited to the time axis.

To use rubber band scaling

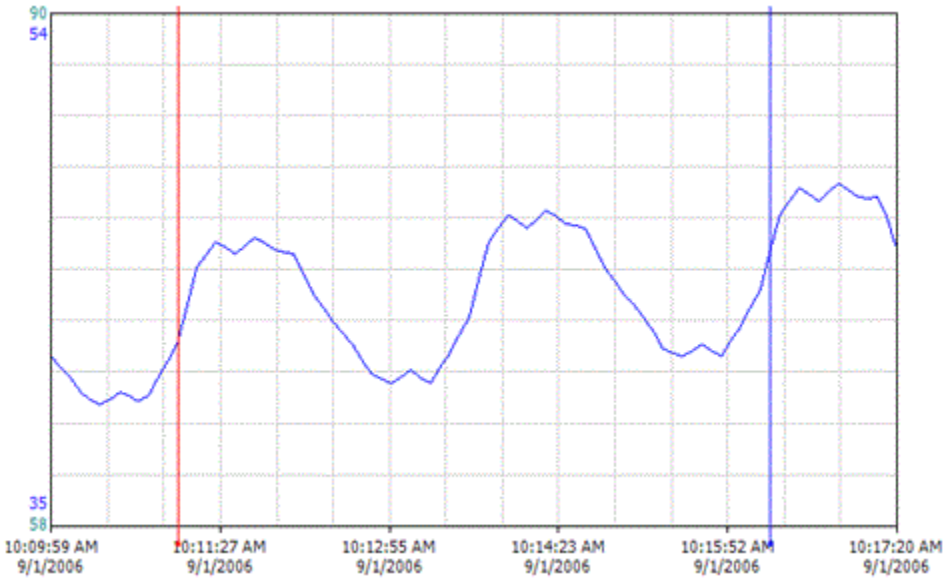
1. From the **Chart** menu, in the **Zoom** group, click **Selection** so that it is highlighted.
2. Unless you are using stacked traces, rubber band scaling affects both the time and the value axes. Time axis scaling always applies to all tags in the chart. Value axis scaling can apply to all tags or the currently selected tag only. If you want value axis scaling to apply to all tags, from the **Chart** menu, in the **Zoom** group, click **Select All** so that it is highlighted.

   If you are using stacked traces, rubber band scaling affects the time axis only, and this setting has no effect.
3. Drag a box around the area you want to use for the new scale.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The trend chart is automatically redrawn using the new zooming values that you selected with the mouse.



Panning in the Trend Chart

By default, the chart is panned to the left or right by the time span percentage set for the chart. This time span applies to both left and right panning and is a percentage of existing data coverage on the chart. The default time span is 75 percent; that is, when you pan right or left, the chart pans by three quarters of the total time span. For example, if the time axis for the chart spans one hour, the chart is panned to the left or right by 45 minutes.

For more information on configuring the panning scale, see Configuring Axis Properties.

To pan left

- Click the **Pan Left** toolbar button.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To pan right

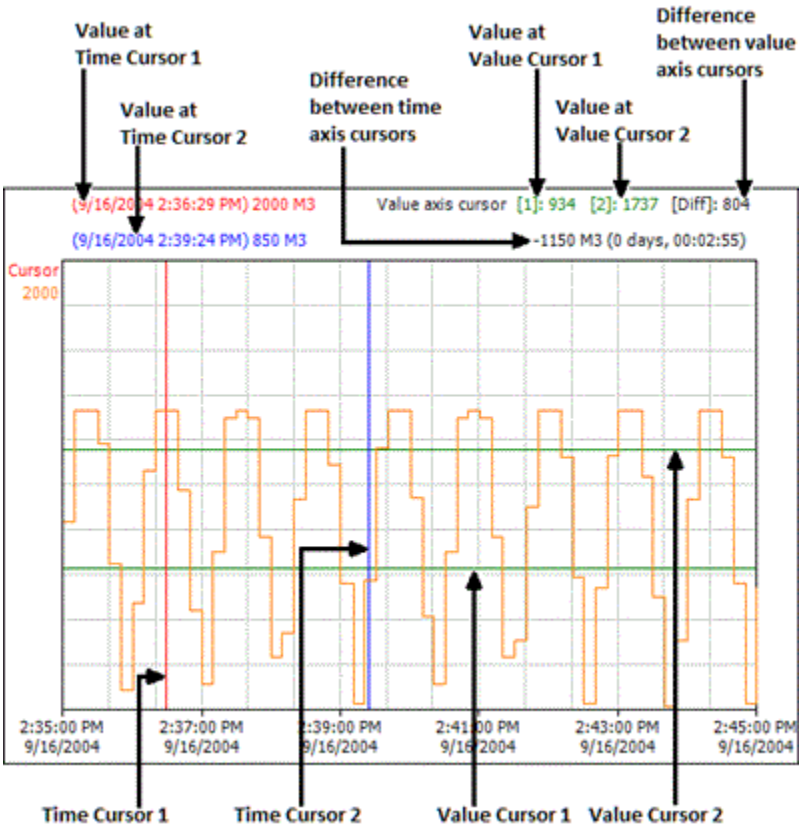
- Click the **Pan Right** toolbar button.

If the time scale is set into the future, then white space appears.

During a pan, the time picker changes to reflect the currently displayed selection.

Using Axis Cursors

Each trend chart has two value cursors and two time cursors. These cursors pinpoint tag values in the chart. The values shown for the axis cursors are updated continuously as the cursors are moved or as the trend curve moves in live mode.



You can show or hide the value and time cursors, as well as the values at the cursors. You can also show or hide the value cursor difference.

To configure the line and font colors for the cursors and cursor value displays, see Configuring Axis Properties.

Moving a Cursor

To move a cursor

1. Select the cursor with your mouse.
2. Drag the cursor to the spot on the chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

As you move the cursor in the trend chart, the value for the tag where the cursor and the tag curve meet appears.

Showing/Hiding the Axis Cursors

To show the time axis cursors

- From the **View** menu, in the **Data** group, select **Time Axis**.

To show the value axis cursors

- From the **View** menu, in the **Data** group, select **Value Axis**.

To hide the cursors, follow the same procedure so that no highlighting appears.

Showing/Hiding the Cursor Difference

To show the cursor difference

- From the **View** menu, in the **Data** group, select **Cursor Difference**.

To hide the cursor difference, follow the same procedure so that no highlighting appears.

Zooming

When you use zooming in the trend chart, the zoom value depends on whether you are using time axis cursors.

If you are not using time axis cursors, zooming is based on the total value for the time axis. The trend chart is zoomed in or out based on the current percentage set for the zooming scale. All zooms are positioned along the middle line of the trend chart.

If you are using time axis cursors, zooming in sets the time period to the period between the cursors. Zooming out works as described above.

For information on setting the zooming percentage, see Configuring Axis Properties.

To zoom in

- From the **Chart** menu, in the **Zoom** group, click **In**.

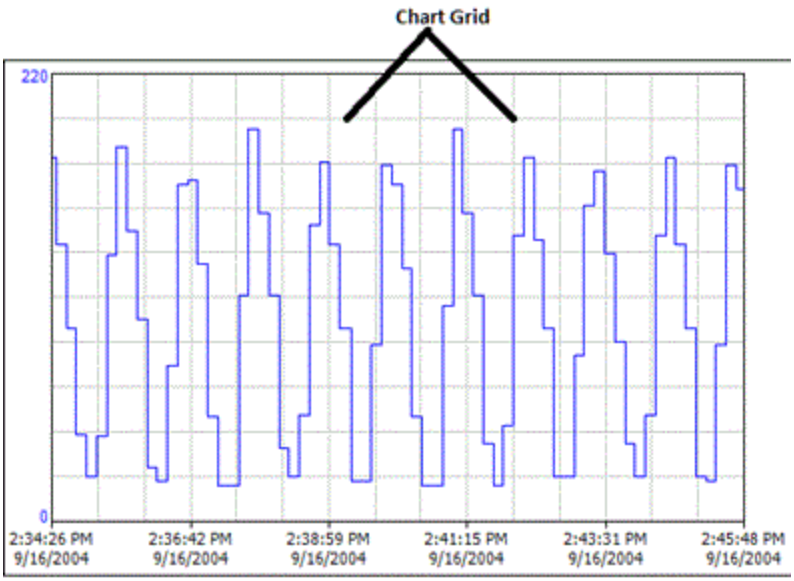
To zoom out

- From the **Chart** menu, in the **Zoom** group, click **Out**.

Showing/Hiding the Chart Grid

You can show/hide the horizontal and vertical chart lines.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



To show the horizontal lines

- From the **View** menu, in the **Grid** group, select **Horizontal**.

To show the vertical lines

- From the **View** menu, in the **Grid** group, select **Vertical**.

To hide the lines, deselect both **Horizontal** and **Vertical**.

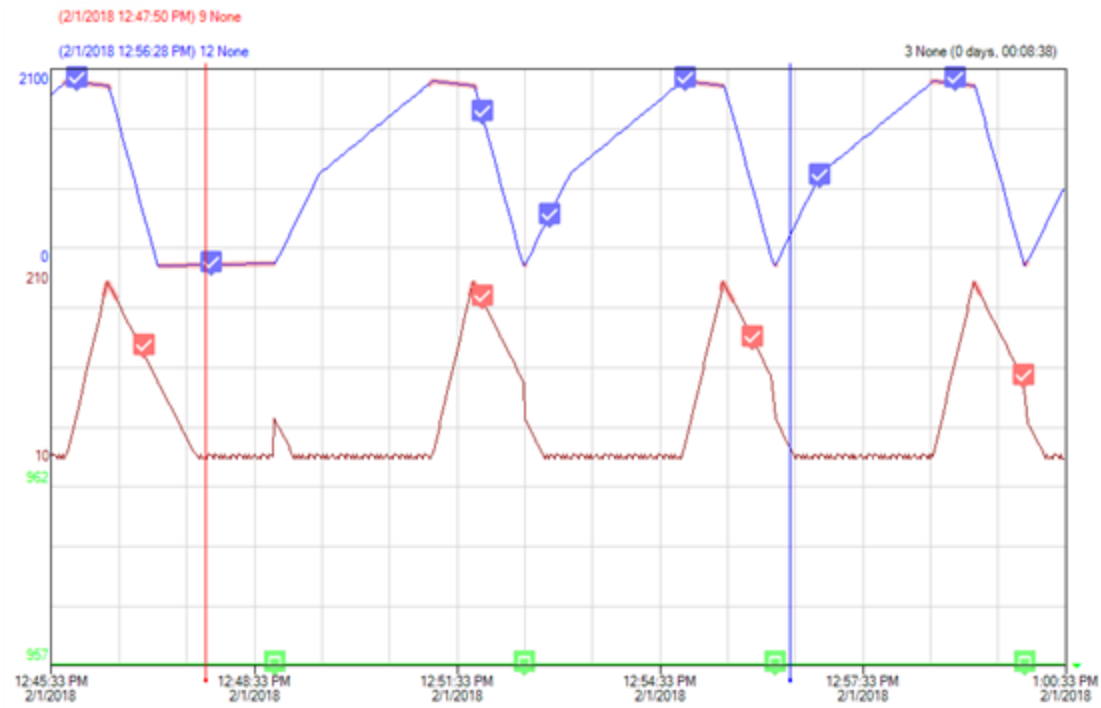
Viewing Alarms and Events

AVEVA Historian Client supports AVEVA Application Server alarms and events captured through AVEVA Historian (including replication) or the AVEVA Historian SDK. This feature does not support alarms and events from the Microsoft SQL databases, WWALMDB, A2ALMDB, and AVEVA Insight.

For more information on Application Server events, see the AVEVA System Platform documentation.

When you include tags with associated alarms and events in a trend, those alarms and events are displayed as part of your trend.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Note: Alarm and events for discrete and string tags are not supported in Trend.

In a trend, AVEVA Historian Client can display:

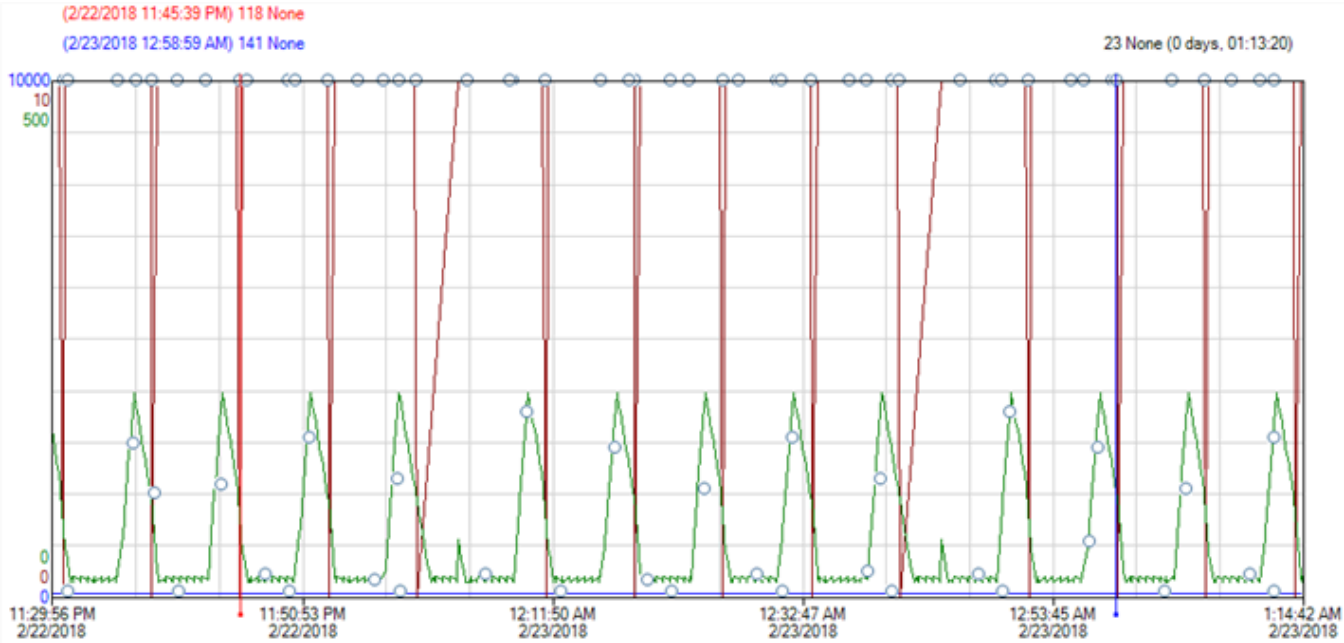
	Alarm state The alarm state is displayed as a highlighted line that begins with an Alarm.Set event and ends with an Alarm.Clear event. **Note:** If either the Alarm.Set or Alarm.Clear for an alarm occurs outside the displayed timeframe, the highlighted are will still display. However, if both Alarm.Set and Alarm.Clear for the same alarm occur outside of the displayed timeframe, the highlight will not be displayed.
	Alarm Acknowledge event When an alarm is acknowledged, that point in time is marked by an Alarm Acknowledge icon (left).
	Application Write event When the application records an event, it is marked with an Application Write icon (shown left).
	User Write event When a user records an event, it is marked with a User Write icon (shown left).
	User Write Secured event When a User.Write.Secured event occurs, it is marked with a User Write Secured icon (shown left)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	User Write Verified event When a User.Write.Verified event occurs, it is marked with a User Write Verified icon (shown left)
	Multiple events To reduce clutter, if there are too many events to display neatly for the timeframe presented, InSight uses the Multiple Events icon (left). This occurs if any tag displayed has more than 30 events for the timeframe shown. If you zoom in, you will see individual event and alarm icons. If you zoom out far enough, the Multiple Events icon disappears.

Note: Events can appear in the display area for a chart even if no tag data is available.

When the displayed timeframe includes one or more tags with more than 30 events, individual event icons are replaced with Multiple Events icons.



If more than 100 events per tag occur within the displayed timeframe, you'll see no icons Instead, you'll see a message in the status bar saying there are too many events to display. In this case, you can zoom in to see more events details.

To enable Alarms and Events in your trend

1. From the **Tools** menu, click **Options**.
2. Mark the **Enable Alarms and Events (from Historian data blocks only)** check box.

To disable Alarms and Events in your trend

1. From the **Tools** menu, click **Options**.
2. Clear the **Enable Alarms and Events (from Historian data blocks only)** check box.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

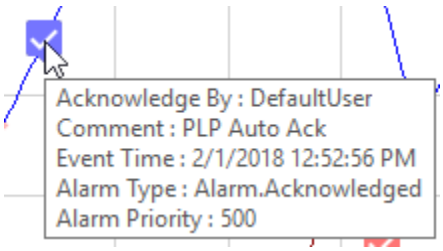
To view event icons when none display

- To see individual event icons, select a smaller timeframe from the Time toolbar.

  For example, if you see Multiple Events icons when the Time toolbar is set to 1 hour, try setting it to 30 minutes to view individual event icons.

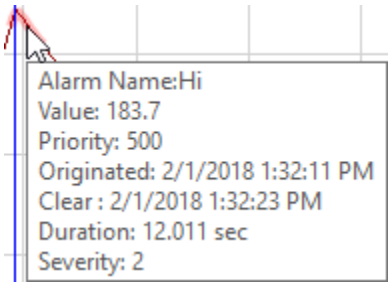
To view details about a single event

- Hover over the event icon to see its details.



To view details about an alarm state

- Click anywhere on the highlighted area.



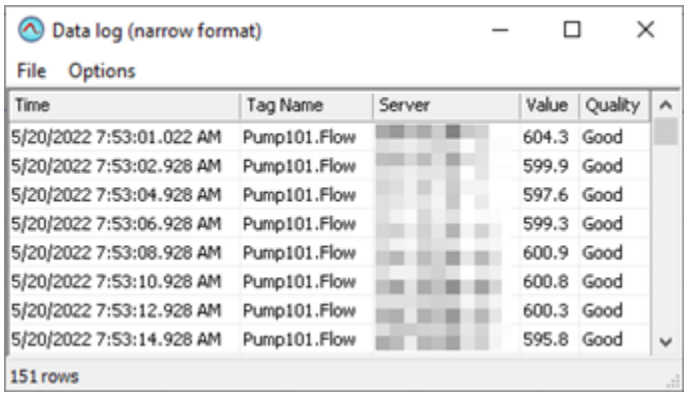
Viewing Trend Data in a Table Format

You can view a table of all data points used in a chart. This data log can be either in a "narrow" or "wide" format. In both cases, the log only shows values for tags that aren't hidden.

Viewing the Data Log in a "Narrow" Format

To view the data log

1. On the **View** menu, click **Logs**, and then select **Narrow**. The **Data log** dialog box appears.



Time	Tag Name	Server	Value	Quality
5/20/2022 7:53:01.022 AM	Pump101.Flow		604.3	Good
5/20/2022 7:53:02.928 AM	Pump101.Flow		599.9	Good
5/20/2022 7:53:04.928 AM	Pump101.Flow		597.6	Good
5/20/2022 7:53:06.928 AM	Pump101.Flow		599.3	Good
5/20/2022 7:53:08.928 AM	Pump101.Flow		600.9	Good
5/20/2022 7:53:10.928 AM	Pump101.Flow		600.8	Good
5/20/2022 7:53:12.928 AM	Pump101.Flow		600.3	Good
5/20/2022 7:53:14.928 AM	Pump101.Flow		595.8	Good

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Data appears for the following columns:

• Time	The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
• Tag Name	The name of the tag within the Historian server. The attribute reference is displayed as a tag name for data values from application server. For application server attributes, you can also choose to show the hierarchical name along with the attribute reference.
• Server	The server from which data is being retrieved.
• Value	The value of the tag at the time stamp.
• Quality	The basic data quality indicator associated with the data value.

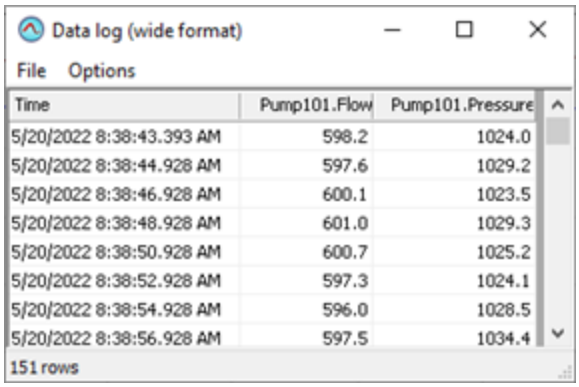
2. To include only the data that is between the time axis cursors on the chart, on the **Options** menu, click **Data From Between Cursors**.
3. To include all of the data on the chart, on the **Options** menu, click **Data From Between Graph Start/End**.
4. To show actual values for discrete tags (for example, 1 or 0), on the **Options** menu, click **Show Actual Values For Discretes**. When retrieving data for discrete tags in ValueState mode, you must select this option to see correct time-in-state information.
5. To show messages for discrete tags (for example, ON or OFF), on the **Options** menu, click **Show Messages For Discretes**.
6. You can copy and paste data to the Windows clipboard by right-clicking in the data and selecting the appropriate option from the menu that appears.
7. To save the data as a .csv file, on the **File** menu, click **Save As**.
8. To set up a printout of the data, on the **File** menu, click **Page Setup**. Setting up the page works like in any other Windows application.
9. To preview a printout of the data, on the **File** menu, click **Print Preview**. Using the preview window works like in any other Windows application.
10. To print the data, on the **File** menu, click **Print**. Specifying printing options works like in any other Windows application.
11. To exit the dialog box, on the **File** menu, click **Exit**. Or, click the **Close** button.

Viewing the Data Log in a "Wide" Format

To view the data log

1. On the **View** menu, click **Logs**, and then select **Wide**. The **Data log** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Time	Pump101.Flow	Pump101.Pressure
5/20/2022 8:38:43.393 AM	598.2	1024.0
5/20/2022 8:38:44.928 AM	597.6	1029.2
5/20/2022 8:38:46.928 AM	600.1	1023.5
5/20/2022 8:38:48.928 AM	601.0	1029.3
5/20/2022 8:38:50.928 AM	600.7	1025.2
5/20/2022 8:38:52.928 AM	597.3	1024.1
5/20/2022 8:38:54.928 AM	596.0	1028.5
5/20/2022 8:38:56.928 AM	597.5	1034.4

Data appears for the following columns:

• Time	The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
• <Tag Name>	The name of the tag within the Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. For Application Server attributes, you can also choose to show the hierarchical name along with the attribute reference.

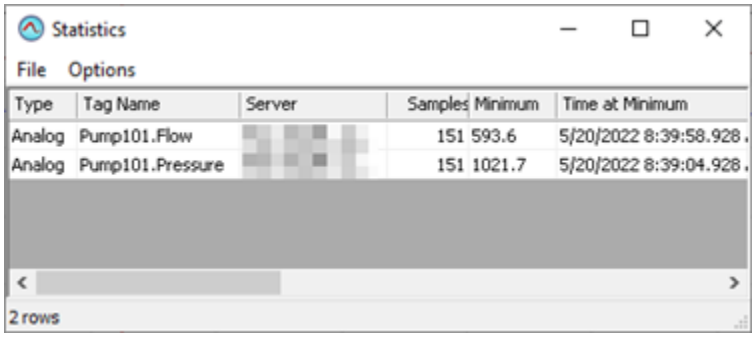
2. To include only the data that is between the time axis cursors on the chart, on the **Options** menu, click **Data From Between Cursors**.
3. To include all of the data on the chart, on the **Options** menu, click **Data From Between Graph Start/End**.
4. To show actual values for discrete tags (for example, 1 or 0), on the **Options** menu, click **Show Actual Values For Discretes**. When retrieving data for discrete tags in ValueState mode, you must select this option to see correct time-in-state information.
5. To show messages for discrete tags (for example, ON or OFF), on the **Options** menu, click **Show Messages For Discretes**.
6. You can copy and paste data to the Windows clipboard by right-clicking in the data and selecting the appropriate option from the menu that appears.
7. To save the data as a .csv file, on the **File** menu, click **Save As**.
8. To set up a printout of the data, on the **File** menu, click **Page Setup**. Setting up the page works like in any other Windows application.
9. To preview a printout of the data, on the **File** menu, click **Print Preview**. Using the preview window works like in any other Windows application.
10. To print the data, on the **File** menu, click **Print**. Specifying printing options works like in any other Windows application.
11. To exit the dialog box, on the **File** menu, click **Exit**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing Statistics

You can view statistics for the data that is retrieved and displayed for a trend. Display statistics include range, average, minimum, maximum, sum, standard deviation, and query properties. Examples of query properties are query time range, start time, end time, and number of rows returned. To display data statistics

1. On the **View** menu, click **Statistics**. The **Statistics** dialog box appears.



Statistics appear in columns as follows.

• Type	The type of tag.
• Tag Name	The name of the tag within the Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. For Application Server attributes, you can also choose to show the hierarchical name along with the attribute reference.
• Server	The server that contains the tag.
• Samples	The number of samples in the trend.
• Minimum	Minimum value for the data that is plotted in the chart.
• Time at Minimum	The time stamp of the minimum value.
• Maximum	Maximum value for the data that is plotted in the chart.
• Time at Maximum	The time stamp of the maximum value.
• Average	Average value for the data.
• Standard Deviation	Standard deviation for the data.
• Range	Value range for the data.
• Timespan	The total amount of time that is spanned by the data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

• From	The starting date for the data.
• To	The ending data of the data.

2. To include only the data that is between the time axis cursors on the chart, on the **Options** menu, click **Data From Between Cursors**.
3. To include all of the data on the chart, on the **Options** menu, click **Data From Between Graph Start/End**.
4. To show actual values for discrete tags (for example, 1 or 0), on the **Options** menu, click **Show Actual Values For Discretes**.
5. To show messages for discrete tags (for example, ON or OFF), on the **Options** menu, click **Show Messages For Discretes**.
6. You can copy and paste data to the Windows clipboard by right-clicking in the log and selecting the appropriate option from the menu that appears.
7. To save the data as a .csv file, on the **File** menu, click **Save As**.
8. To set up a printout of the data, on the **File** menu, click **Page Setup**. Setting up the page works like in any other Windows application.
9. To preview a printout of the data, on the **File** menu, click **Print Preview**. Using the preview window works like in any other Windows application.
10. To print the data, on the **File** menu, click **Print**. Specifying printing options works like in any other Windows application.
11. To exit the dialog box, on the **File** menu, click **Exit**.

Using Annotations

You can use Trend to make an annotation for a tag at any point in time. An annotation is a user comment about a tag. For example, you might want to save a comment about a very high spike in a trend. You can create an annotation for the value of the tag at the spike. All annotations are saved to the database and can be retrieved again at a later time.

You can create a private annotation (that only you can see) or a public annotation (which is viewable by all trend users). Private annotations are only available to the users who created them and have suitable access.

For each annotation, an annotation mark (solid circle) is added to the trend. This annotation mark can be viewed on the trend if the trend properties are set to allow it.

When you make an annotation, the following information is stored in the Annotation table in the Runtime database of the Historian:

- Name of the tag for which the annotation is associated.
- The date/time of the annotation. The time of the annotation is based on the position of where it was created on the time axis.
- The value of the tag at the time of the annotation.
- The annotation text.

Note: You cannot use the Annotations functionality when connected to a Managed Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

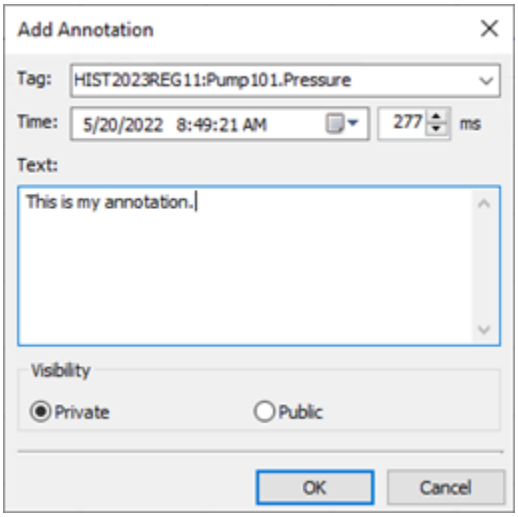
Adding an Annotation

Annotations are inserted in the chart at the location where the mouse button is clicked and are associated with the selected tag's value where the mouse button is clicked.

To add an annotation

1. Select the tag for which you want to add an annotation. You can do this by selecting the tag in the Tag List pane.
2. Do one of the following:
   - On the **Chart** menu, click **Add Annotation**.
   - Right-click in the chart. In the shortcut menu that appears, click **Add Annotation**.

     The **Add Annotation** dialog box appears.



3. In the **Tag** List, click the name of the tag for which you want to add the annotation.
4. In the **Time** list, click the time stamp of the tag value for which you want to add the annotation.
5. In the **Text** window, type in your comment.
6. In the **Visibility** area, specify if you want the annotation to be visible to others. Click **Private** to have annotations only visible to you.Click **Public** to have annotations visible to anyone who is able to log on to the database.
7. Click **OK**.

   An annotation marker (dot) appears on the chart at the point on the chart where the annotation was made.

   If you hover with the mouse on the marker, the details for the annotation will appear on the chart.

Viewing the Annotation List

To view a list of annotations

1. On the **View** menu, click **Annotations**. The **Annotations** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The table shows the following information:

• Type	Specifies whether the annotation is public.
• Tag Name	The name of the tag within the Historian server. If the data values are coming from application server, the attribute reference is shown as the tag name. For application server attributes, you can also choose to show the hierarchical name along with the attribute reference.
• Server	The name of the server that stores the tag values.
• Content	The annotation text.
• User	The name of the database user. This is the user that created the annotation.
• Time	The timestamp of the tag value for which the user has made an annotation.
• Created On	The date that the annotation was created.

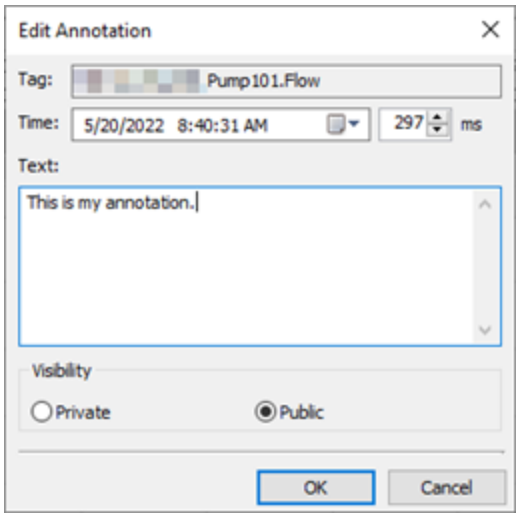
2. To sort the table according to the information in a particular column, click the column heading. Click again to reverse the sorting order.

Editing an Annotation

To edit an annotation

1. On the **View** menu, click **Annotations**. The **Annotations** dialog box appears.
2. Select the annotation in the list.
3. On the **Annotations** menu, click **Edit**. The **Edit Annotation** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. Edit the annotation.
5. Click **OK**.

Deleting an Annotation

Deleting an annotation removes the annotation from the trend.

To delete an annotation

1. On the **View** menu, click **Annotations**. The **Annotations** dialog box appears.
2. Select the annotation in the list.
3. On the **Annotations** menu, click **Delete**. Confirm the deletion.
4. Click **OK**.

Saving the Annotations List as a .CSV File

To save the list of annotations as a .csv (text) file

1. On the **View** menu, click **Annotations**. The **Annotations** dialog box appears.
2. On the **File** menu, click **Save As**. The standard Windows **Save As** dialog box appears.
3. In the **File name** box, type a name for the .csv file.
4. Browse to the location in which to save the file.
5. Click **Save**.

The .csv file contains the same information that appears in the **Annotations** dialog box.

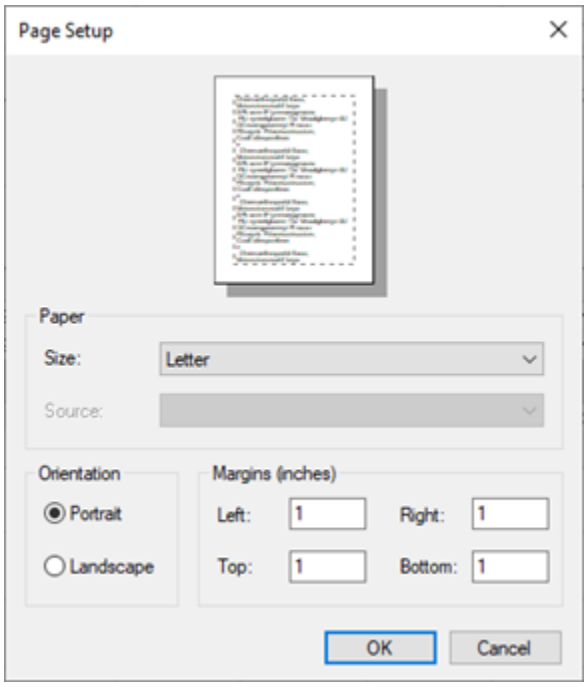
Printing Annotations

To print the list of annotations

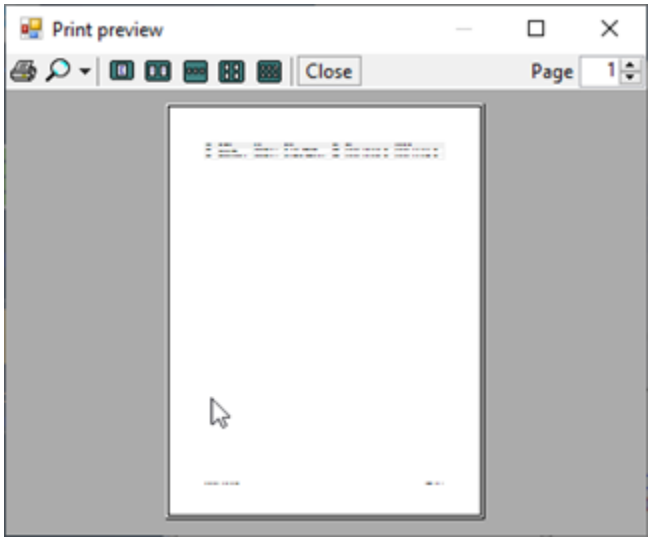
1. On the **View** menu, click **Annotations**. The **Annotations** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. To configure the printing options, on the **File** menu, click **Page Setup**. The **Page Setup** dialog box appears.

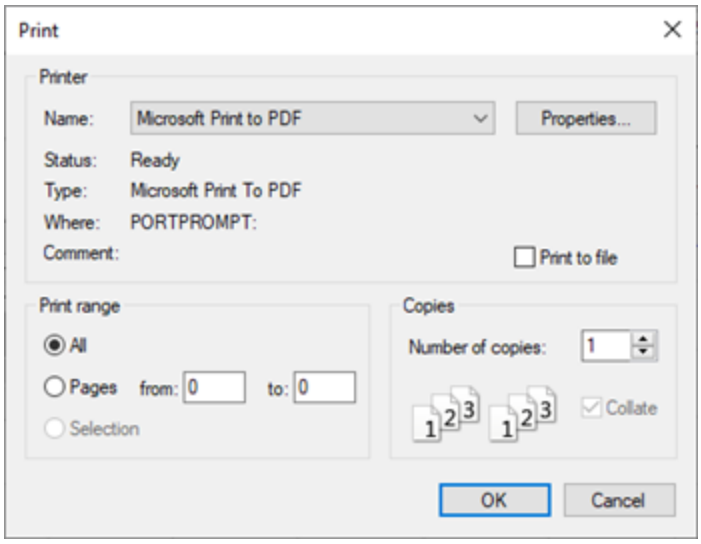


3. Configure the setup options and then click **OK**.
4. To preview the printout, on the **File** menu, click **Print Preview**. The **Print preview** dialog box appears.



5. Verify the preview and then click **Close**.
6. To print the annotations, on the **File** menu, click **Print**. The **Print** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. Configure the printing options and then click **OK**.

Trending Event Tags

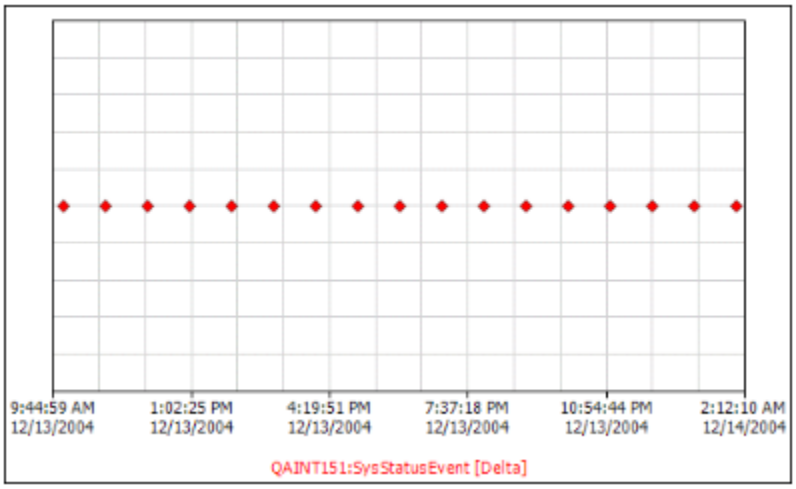
A trend can be configured to show event data from the internal Historian Event Detection/Action subsystem.

Note: This is different than Alarms and Events, which can be associated with tags received and stored in the Historian. See Viewing Alarms and Events for details.

An event is the set of attributes describing the moment of satisfaction of a set of criteria on historical tag values in the AVEVA Historian. Attributes of an event include the date and time that the event occurred and the criteria that were satisfied.

An event tag is a name for an event definition in the system. Whereas these tag types are the definitions of types of variables to be stored, an event tag is a named reference for the description of how a specific change is detected and what to do if it is detected.

You can select and trend event tags like any other tag in the system. Events can also be displayed along with analog and discrete tags.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using Absolute or Relative Times

The following time modes are available for the trend chart:

- Absolute time
- Relative time

The time mode selection is saved as part of the chart definition when you save the .aaTrend file.

Using Absolute Time

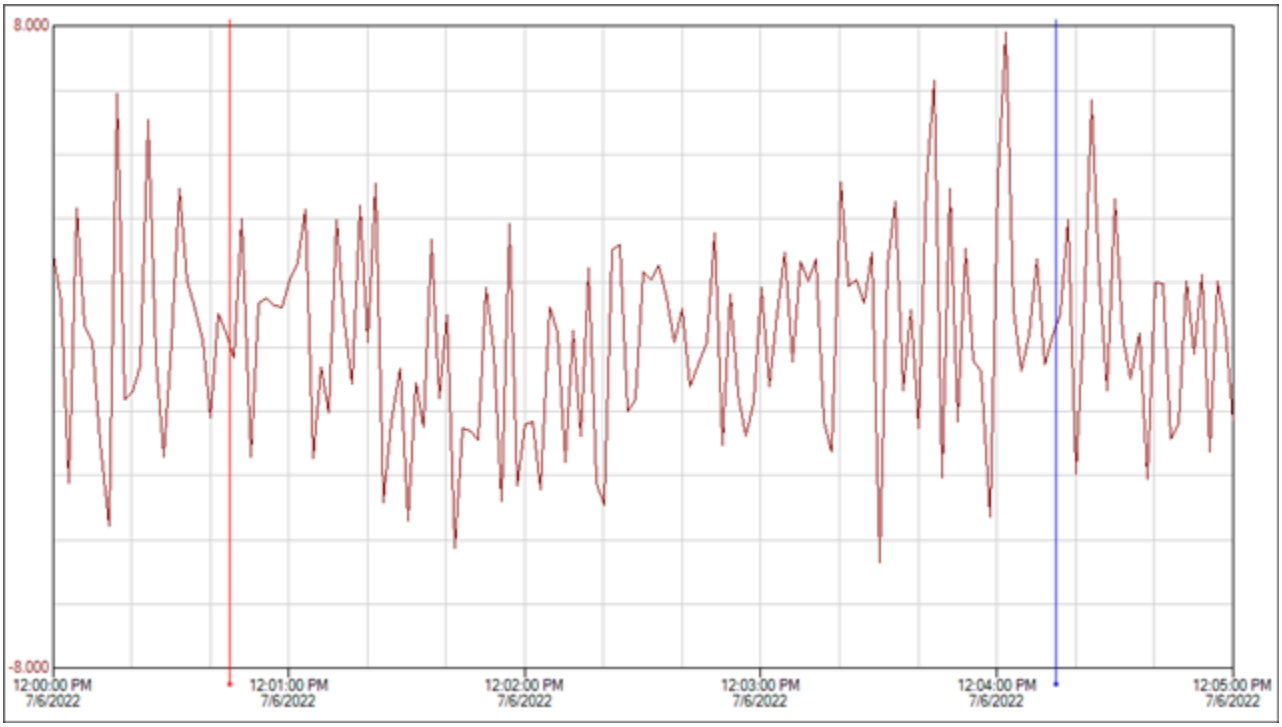
In absolute time mode, you determine the time period for the query by setting a combination of the start time, duration, and end time.

To use absolute time

On the **View** menu, in the **Time** group, select **Absolute**.

For example, the following chart shows five minutes of data in absolute time.

The query for the data starts at 12:00:00 and ends at 12:05:00, and the time axis values reflect these times.



In absolute time mode:

- The times shown for the time axis cursors, if enabled, are absolute times.
- The times shown in the Time Bar are absolute times.
- The time offset for each tag can be configured in the Tag List. For more information, see Using Time Offsets to Compare Data in Absolute Time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

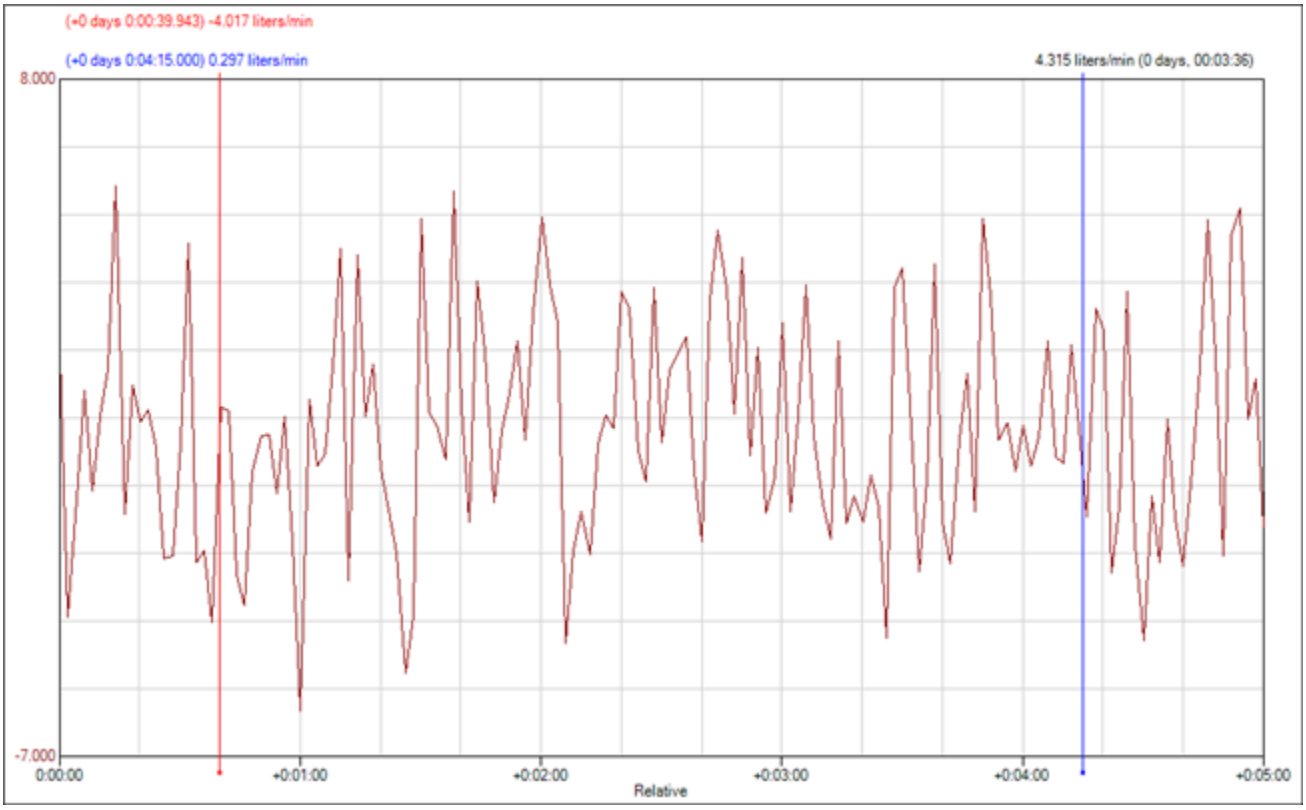
Using Relative Time

For the relative time mode, a base value (such as 0:00:00.0) is used for the start time of the chart, and the end time is calculated based on the time span for the query. Switching to relative mode does not change the data shown in the chart nor the actual start and end time of the trend query. Only the time axis is updated.

To use relative time

On the **View** menu, in the **Time** group, select **Relative**.

For example, the following chart shows five minutes of data in relative mode. The query for the data starts at 12:00:00 and ends at 12:05:00, but the time axis shows a start time of 0:00:00 and an end time of +0:05:00.



In relative time mode:

- The times shown for the time axis cursors, if enabled, are relative times. For details on the display format for times, see Time Offset Formats.
- The time fields in the Time Bar are relative times. The first field is the base time (for example, 0:00:00.000), and the second field is the timespan of the query. The third field is the end time of the query (the base time plus the timespan). To offset the time for the query, adjust the base time (for example, 1:00:00.000 to offset it by +1 hour).
  The first field (the base time) defaults to 0:00:00 when you switch to relative mode.
- The Tag List shows the actual start time for the tag data in the Start Time column.
- The **Tag Properties** dialog box shows the start time option. For more information, see Configuring Trend Options for a Tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Switching Between Absolute and Relative Time: Example

When you change the time mode, the Time Bar and individual tag time settings convert between absolute times and relative offsets.

The following table summarizes the states for the Time Bar and three tag offsets/dates for some example data. Tag1 is the currently selected tag. In this example, the actions performed for the different steps are:

1. At exactly 2005-07-04 10:00, "2 hours" is selected from the Time Bar while the **Update to Current Time** option is enabled. The chart is in absolute mode.
2. The chart is switched to relative time mode.
3. The start and end values are changed in the Time Bar.
4. The chart is switched back to absolute time mode.

Step:	**1**	**2**	**3**	**4**
Mode:	Absolute	Relative	Relative	Absolute
Time Bar Start:	2005-07-04 8:00	0:00	-0:15	2005-07-04 7:45
Time Bar End:	2005-07-04 10:00	2:00	0:45	2005-07-04 8:45
Tag1:	- 24:00	2005-07-03 8:00	2005-07-03 8:00	- 24:00
Tag2:	0:00	2005-07-04 8:00	2005-07-04 8:00	0:00
Tag3:	+3:30	2005-07-04 11:30	2005-07-04 11:30	+3:30
Time Axis Start:	2005-07-03 8:00	0:00	-0:15	2005-07-03 7:45
Time Axis End:	2005-07-03 10:00	2:00	0:45	2005-07-03 8:45

Time Offset Formats

The application displays time offsets in the following formats:

[±][d]hh:mm[:ss[.fff]]

-OR-

[±]HH:mm[:ss[.fff]]

-OR-

[±]d[.FF]

Items in square brackets ([and]) are optional. Colons and periods (: and .) are literal characters.

The notation variables are as follows.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Item	Description
±	Minus sign indicating a negative time. Positive time is assumed.
d	Days
hh	Hours, ranging from 0 to 23.
HH	Hours of 24 or greater.
mm	Minutes, ranging from 0 to 59.
ss	Seconds, ranging from 0 to 59.
fff	Fractional seconds, from 1 to 7 decimal digits.
FF	Fractional days.

Time offsets are shown in the application in either a short or long form:

Short form:	HH:mm:ss.ff or d hh:mm:ss.fff
Long form:	d <label> hh:mm:ss.fff

In the short form:

- "d" is omitted for offsets less than or equal to 48 hours
- ".fff" is omitted for offsets greater than 60 seconds
- ":ss.fff" is omitted for offsets greater than 24 hours

Thus, for periods of less than 60 seconds, the short form is never longer than 11 characters. For the short form, the hours in HH format (rollover at 48 instead of 24) are shown only if days are not to be displayed anywhere on the time axis.

The "<label>" is the localized word for "days" or "day." The period (.) and colon (:) are replaced with the appropriate characters from the regional settings.

Using Time Offsets to Compare Data in Absolute Time

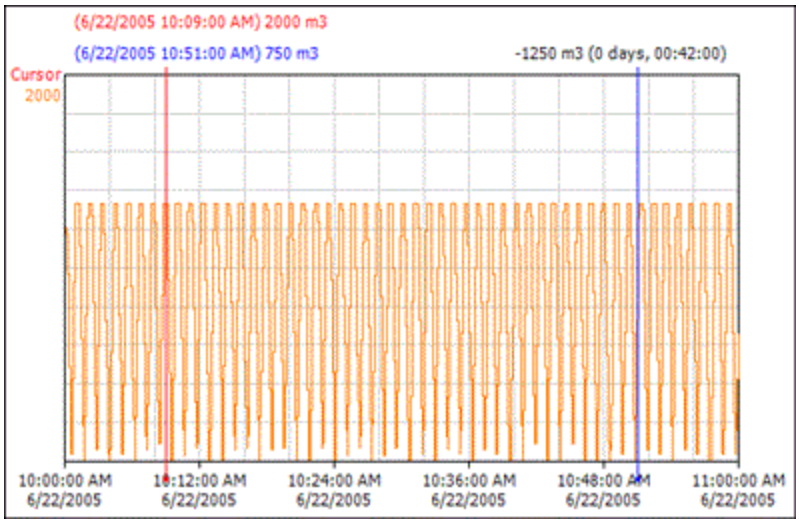
In absolute time mode, you can use a time offset to compare the same data from different time periods. For example, you may want to compare data from a shift at 10:00 a.m. to data from a shift at 11:00 a.m. The time offset feature enables you to adjust the time period for one of the shifts so that the data appears as if it occurred during the same time period as the other shift. Using a time offset enables you to easily see the differences between the data on the trend chart.

To use a time offset

1. Create a trend for batch of data that you want to use as the basis for comparison.

   In this example, the chart is configured to show data for the ReactLevel tag between 6/22/2005 10:00:00 AM and 6/22/2005 11:00:00 AM.

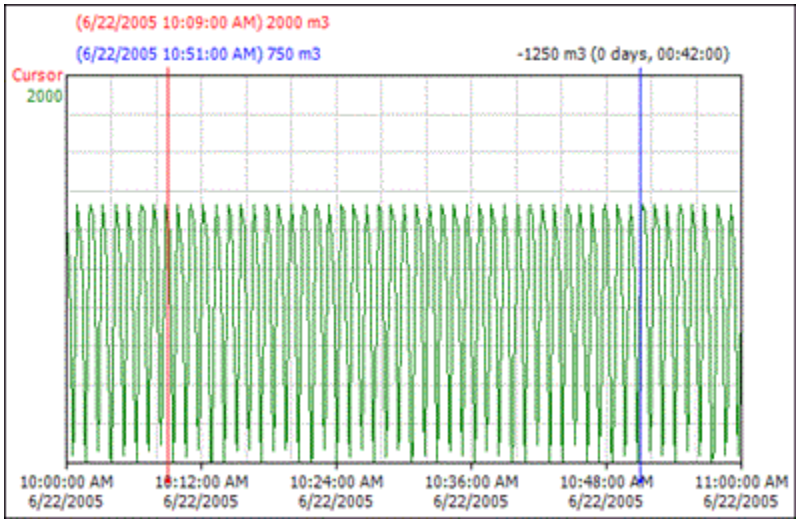
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Add the same tag again to the trend chart for the same time period.

   In this example, the ReactLevel tag was added again to the chart.

3. Because the data is identical, you only see single trend curve in the chart.



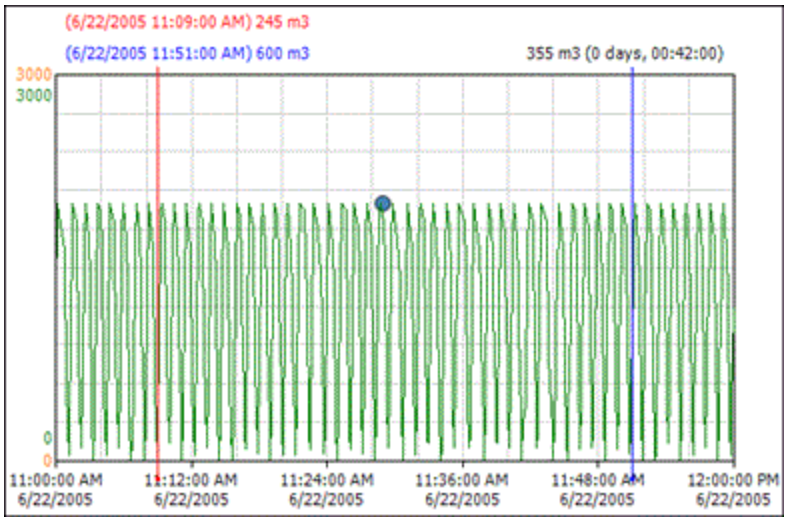
4. Determine the time span for the data you want to compare with the base batch of data.
5. To specify the time offset for the data to compare, double-click on the tag in the Tag List. The **Tag Properties** dialog box appears.
6. In the **Time offset** box, configure the amount of time that the data shown in the chart is to be offset from the actual query time. For more information on the format, see Time Offset Formats.

   In this example, this data is to be compared with the base batch that occurred an hour before, so the time offset is set to one hour.

7. Click **OK**.

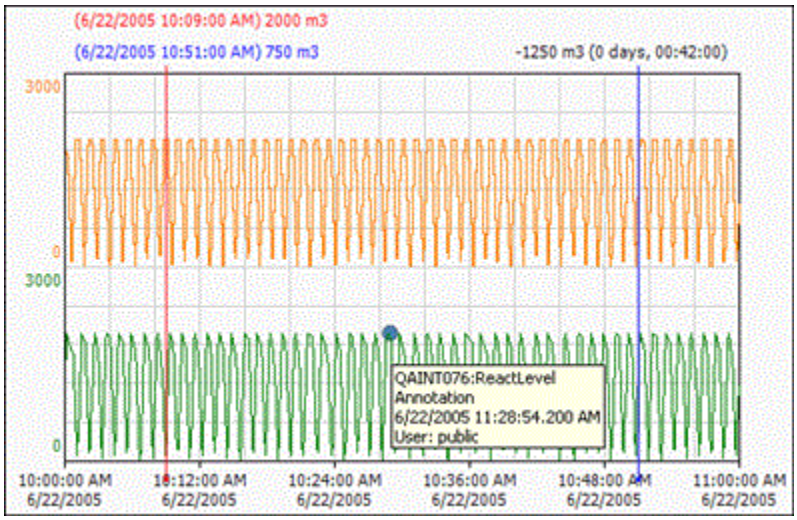
   In this example, the data for the ReactLevel tag between 6/22/2005 11:00:00 AM and 6/22/2005 12:00:00 AM includes an annotation made around 11:30:00 AM.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



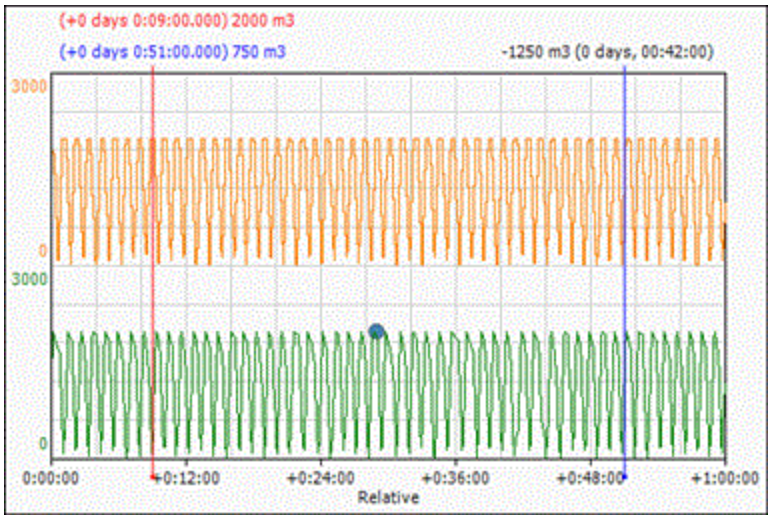
8. Stack the traces so that you can see both sets of data separately and then select the first tag that you added to the chart.

   In this example, the trend curve for the later set of data (shown in green) appears on the chart, even though the time axis reflects the time of the base batch of data (shown in orange).



9. To view the chart in relative mode, on the **View** menu, click **Relative Time**. The time axis now shows the time span for the base batch starting at 0:00:00 instead of the actual time.

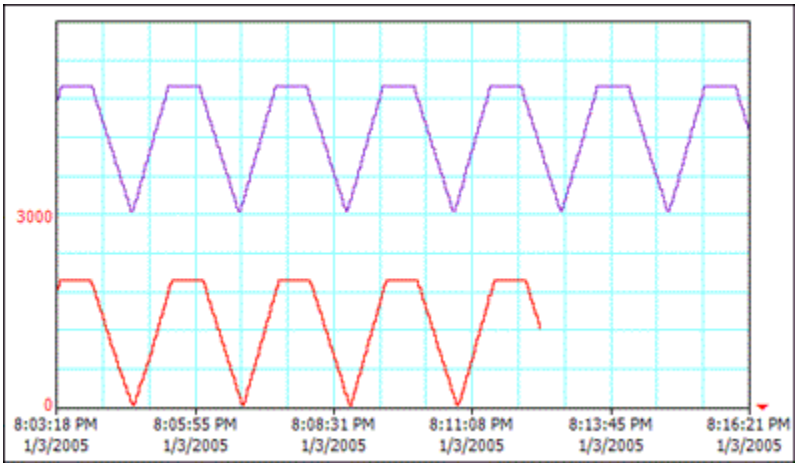
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



You can also use the offset to compare a trend curve against another curve either forward or backward in time. To do this, set the time offset of the "master" batch of data so that the start time is the same as the start time for the batch of data you want to compare.

In the following example, the time offset for the complete batch is set to a value of -01.00.44. The complete batch appears as the top curve in the chart.

The incomplete curve at the bottom of the chart is plotted in live mode next to the complete curve at the top.



Configuring Trend Application Options

The trend options allow the user to configure the trend application. These options apply to all saved trend files. Categories of trend options that can be set include:

- Configuring Retrieval Options
- Configuring Color Options
- Configuring Time Zone Options
- Configuring Miscellaneous Options
- Configuring Other Options

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring Retrieval Options

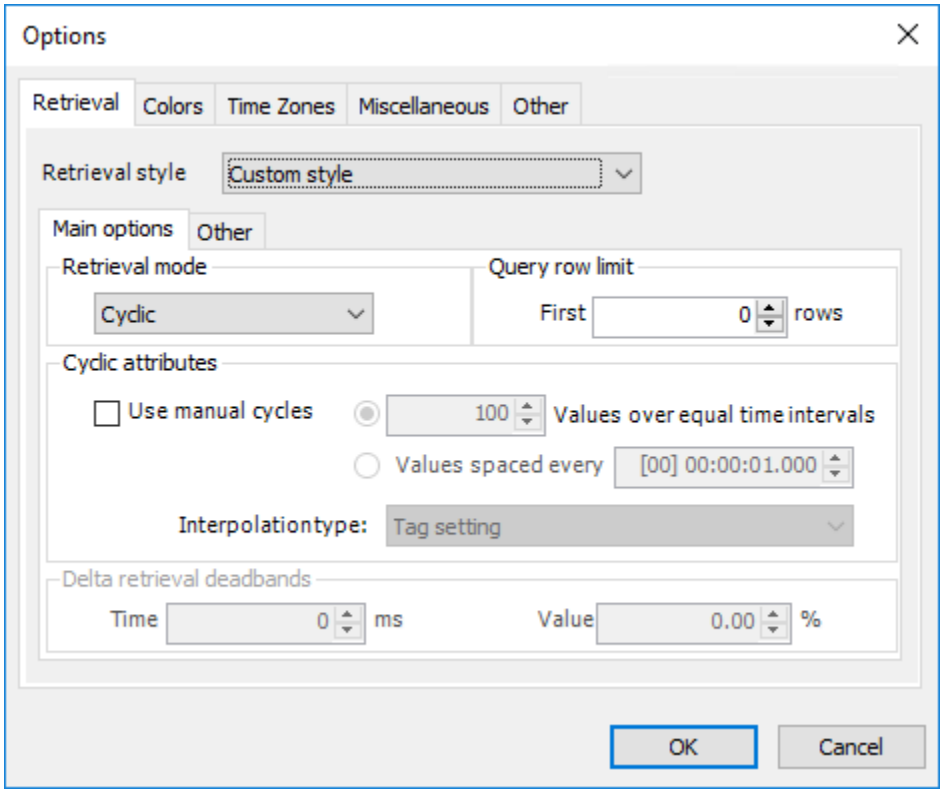
You can define data retrieval options at the application level. These options are used for all tags that do not have their own settings defined.

Application-level retrieval options are not saved in trend files. Therefore, trend files with tags that rely on application-level retrieval settings may look different depending on the retrieval options that are configured in the Trend application that they are opened in. To make sure that your tags are using specific retrieval options, define these options individually for each tag. For more information, see Configuring Trend Options for a Tag.

Most retrieval settings that you configure here only apply if you are retrieving data from a Historian with a version of 9.0 or later. If you are using an earlier Historian version, see Configuring Other Options and Working with Retrieval Styles for details.

To configure retrieval options

1. On the **Tools** menu, click **Options**. The **Options** dialog box appears with the **Retrieval** tab selected.



2. Do one of the following:
   - To use a predefined retrieval style, click its name in the **Retrieval style** list. For more information on retrieval styles, see Working with Retrieval Styles.
   - To use custom retrieval settings, click **Custom style** in the **Retrieval style** list.
3. Specify any additional settings required.
   - If you are using custom retrieval settings, select a retrieval mode and specify all the settings that are relevant to it. For more information, see Understanding Retrieval Modes.
   - If you are using one of the predefined styles, you can edit all settings that are not covered by the style definition. For information on which settings are covered by style definitions, see Working with Retrieval Styles.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Because a style definition can contain multiple sets of retrieval settings with different retrieval modes, some of the settings available for editing here may turn out to be irrelevant for the retrieval mode that actually gets used for a given query. However, because there is no way to know in advance whether this will be the case, the settings are still available for editing.

For more information on the various retrieval options, see Understanding Retrieval Options.

By default, the retrieval settings that you specify here are used for all tags on all trend charts. However, you can override these settings individually for each tag. For more information, see Configuring Trend Options for a Tag.

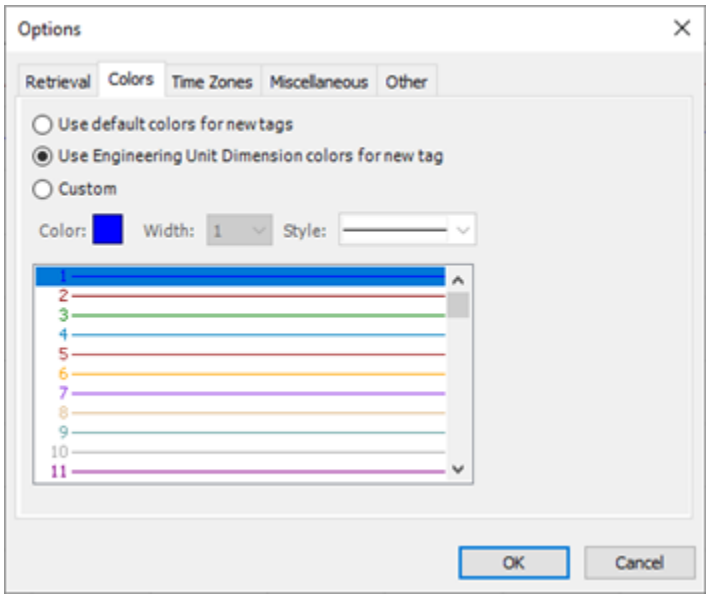
Configuring Color Options

The trend color options control how the trend pen looks for each new tag as it is added to the chart. By default, Trend includes 256 different pen styles, which are numbered from 1 to 256. An unused style is applied each time you add a tag to the trend chart. The trend assigns the lowest pen style that is available. For example, the first pen style is a solid red line, so the first tag you place in a chart has this style. You can change the default pen styles.

Changing the options does not affect tags that are already in the trend chart.

To configure color options

1. On the **Tools** menu, click **Options**. The **Options** dialog box appears.
2. Click the **Colors** tab.



3. To use the default pen styles for the tags in a trend, select either **Use default colors for new tags** or **Use Engineering Unit Dimension colors for new tag**. Go to step 10.
4. To configure one or more pen styles, select **Custom**.
5. Select a pen number from the list.
6. Click the **Color** box and select or configure a color for the pen line.
7. In the **Width** list, select the width, in pixels, of the pen line.
8. In the **Style** box, select the style of the pen, either a solid line or one of a variety of dashes.
9. Repeat steps 5 through 8 for each pen style you want to configure.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

10. Click **OK**.

Configuring Time Zone Options

You can configure Trend so that data appears with time stamps that reflect any time zone. For example, you may want to configure the Trend to the same time as the location of the server.

To configure time zone options:

1. On the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
2. Select the **Time Zone** tab. The grid displays the current time zone and daylight saving time settings for the following entities:

Entity	Description
Application	The Historian Client Workbook application. You can select the time zone for the data as it appears in the Workbook application.
Client	The physical computer on which the Workbook application is installed. The time zone displayed for the client is for informational purposes only and cannot be changed using the Workbook application.
<Server>	The Historian(s) to which the Workbook application is currently connected. The time zone displayed for the server(s) is for informational purposes only and cannot be changed using the Workbook application.

3. In the **Time zone** list, click the name of the time zone to use for the Workbook application.

   The time zone for the Workbook application in the grid displays the new time zone picked.

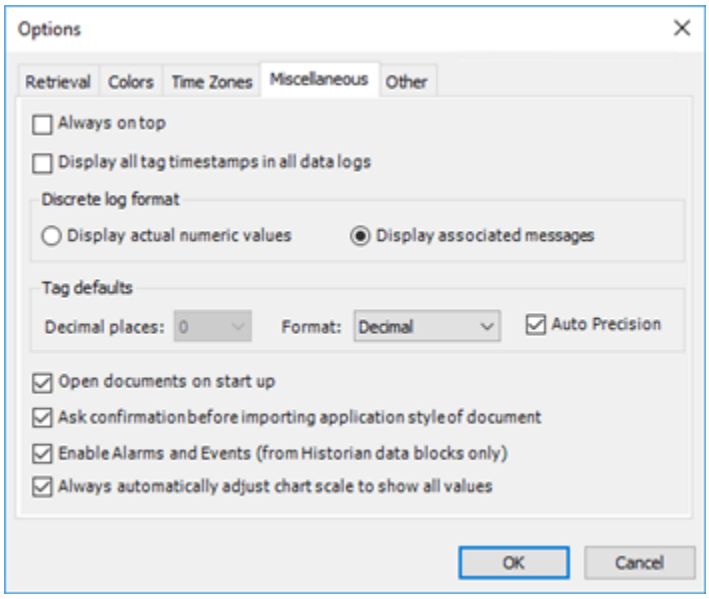
   For example, consider a SCADA application that monitors a pipeline between Houston, Texas and Lake Forest, California. The Workbook application is installed on a computer located in Houston, Texas. Therefore, the time zone entry for the Client entity displays Central Standard Time. The server is also located in Houston, Texas. The time zone entry for the Server entity also displays Central Standard Time. You want to send a Workbook file to an engineer located at the start of the pipeline in Lake Forest to aid in troubleshooting a problem. You can set the time zone of the Workbook application to reflect the time of Lake Forest, California (Pacific Standard Time), so that the workbook that you send to the engineer displays data in a time zone that is relevant to him/her.
4. Click **OK.**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring Miscellaneous Options

To configure miscellaneous options

1. On the **Tools** menu, click **Options**. The **Options** dialog box appears.
2. Click the **Miscellaneous** tab.



3. Select **Always on top** to always display Trend as the top-most program on the computer desktop.
4. Select **Display all tag timestamps in all data logs** to include the time stamps for all tags in the data log.
5. In the **Discrete log format** area, configure how the values for discrete tags appear in the data log. Select **Display actual numeric values** to show the numeric value for the discrete tag, either 1 for the TRUE state or 0 for the FALSE state. Select **Display associated messages** to show the text associated with the TRUE or FALSE state of the discrete tag. For example, "On" or "Off," "Started" or "Stopped."
6. In the **Tag defaults** area, configure how tag values appear in the chart. Changes to these settings are not applied until the next tag is added to the chart.

• Decimal places	The number of values that appear to the right of the decimal period.
• Format	The format for tag values, either decimal format or scientific format. For the scientific format, the value appears with an E denoting the exponent.
• Auto Precision	If selected, the number of decimal places is automatically set based on the value range. When Auto Precision is selected, the Decimal places field is read-only.

7. Configure the following additional options:

• Open documents on start up	If selected, when you start the Trend application, it automatically reopens the trend files that were

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

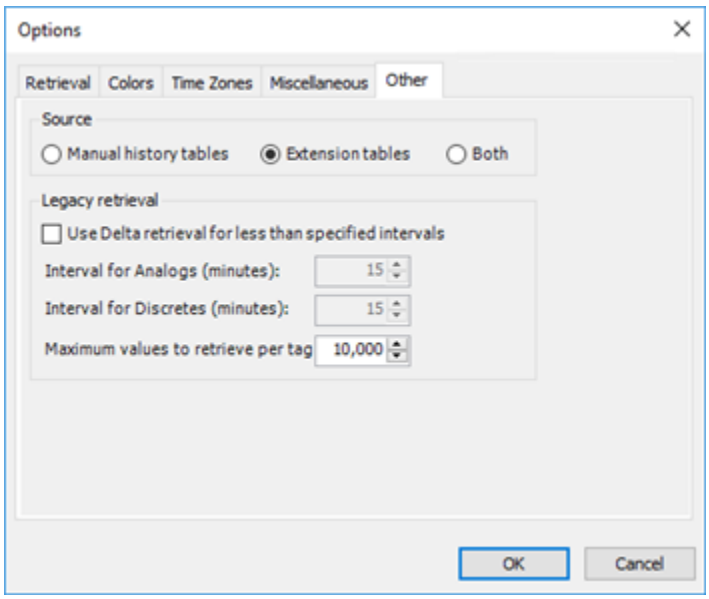
	open when you closed it. Otherwise, a new trend file is created.
• Ask confirmation before importing application style of document	Some style information is saved in trend files, and this may differ from the default styles configured in the Trend application. If this option is selected, when you open a trend file you will be prompted to confirm whether or not to override the default settings with the settings from the trend file. Otherwise, the settings from the trend file will override the default settings automatically, with no confirmation required.
• Enable Alarms and Events (from Historian data blocks only)	Select this option to enable the alarms and events overlay in the trend chart.
• Always automatically adjust chart scale to show all values	If this option is selected, when trend values cross the chart boundaries, the Trend application will automatically adjust the chart scale so that all values are displayed.

8. Click **OK**.

Configuring Other Options

To configure other options

1. On the **Tools** menu, click **Options**. The **Options** dialog box appears.
2. Click the **Other** tab.



3. In the **Source** area, specify the Historian tables from which data will be retrieved.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

• Manual history tables	Normal SQL Server tables that are used to store data. These are the ManualAnalogHistory and ManualDiscreteHistory tables.
• Extension tables	Logical tables that are populated from the Historian data files. These tables support the Historian time domain extensions for handling data.
• Both	Select this option to retrieve data from both the manual and extension tables.

4. In the **Legacy retrieval** area, specify the retrieval mode for data that is retrieved from the Historians with a version earlier than 9.0.

   For information on how these settings interact with a retrieval style that you may have selected, see Working with Retrieval Styles.

• Use Delta retrieval for less than specified intervals	Select this check box to use delta retrieval mode for query time periods that are less than a specified amount.
• Interval for Analogs	The time period, in minutes, for which delta values are retrieved for analog tags. For greater time periods, cyclic retrieval is used instead. Valid values are 0 to 250,000. The default value is 15.
• Interval for Discretes	The time period, in minutes, for which delta values are retrieved for analog tags. For greater time periods, cyclic retrieval is used instead. Valid values are 0 to 10,000. The default value is 15.
• Maximum values to retrieve per tag	The maximum number of values to return per tag. Valid values are 0 to 30,000. The default value is 10,000.

5. Click **OK**.

Configuring Trend File Properties

The trend properties enable you to configure a trend file. Trend file properties are saved with the trend file. Categories of trend properties that can be configured include:

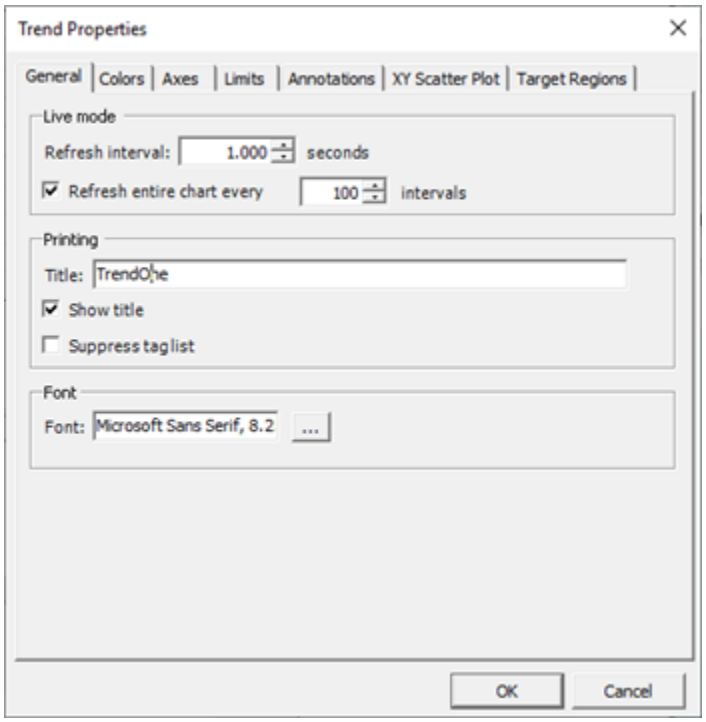
- Configuring General Properties
- Configuring Color Properties
- Configuring Axis Properties
- Configuring Limit Properties
- Configuring Annotation Properties
- Configuring Target Region Properties

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring General Properties

To configure general properties

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Select the **General** tab.



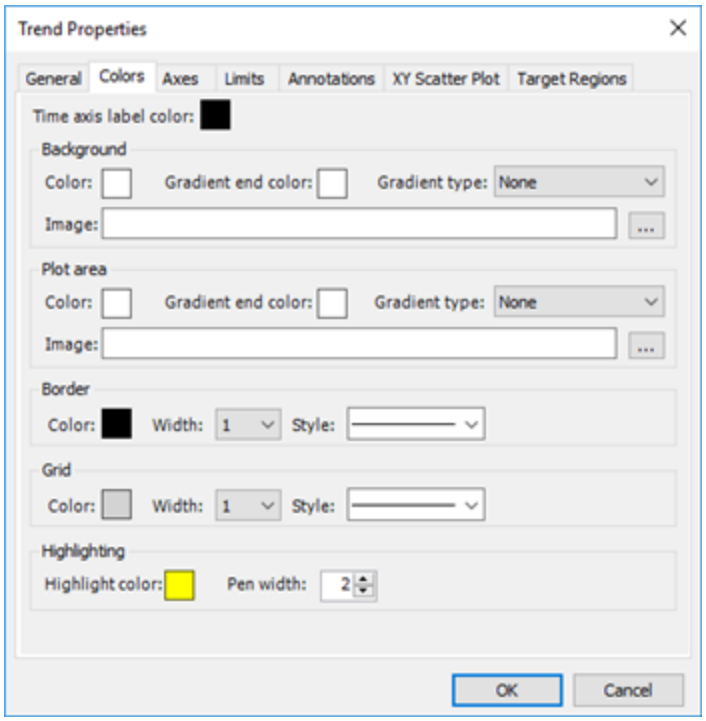
3. In the **Refresh interval** box, specify the time period, in seconds, at which the chart is refreshed if set to live mode. Valid values are 0.25 to 300. The default value is 1.
4. In the **Refresh entire chart every XX intervals** box, specify the number of refresh intervals after which the entire chart is refreshed. The chart is not only refreshed with the new live data, but all the data in the chart is refreshed. Valid values are 1 to 100,000. The default is 100.
5. In the **Printing** area, configure options for chart printing.
   - **Title:** The title of the chart.
   - **Show title:** Show the title in the printout.
   - **Suppress tag list:** Do not include the tag list in the printout.
6. In the **Font** area, click the **Font** icon to select the name, style, and size of the font that is to be displayed on the chart and Tag List.
7. Click **OK**.

Configuring Color Properties

To configure color properties

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Click the **Colors** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Click the **Time axis label color** box to select or configure the color for the time labels that appear at the bottom of the chart.
4. In the **Background** area, configure the colors or image to use for the background of the entire chart area.
   - **Color:** Click to select or configure a main color. If you are using a gradient fill, this is the starting color for the gradient.
   - **Gradient end color:** Click to select the ending color for the gradient. The gradient starts with the main color and fade to the gradient end color.
   - **Gradient type:** The starting point for the flow of the gradient. Valid values are Center, Diagonal Left, Diagonal Right, Horizontal Center, Left Right, Top Bottom, and Vertical Center. For example, if you select green as main color, white as the gradient end color, and center as the gradient type, the center of the chart is green and fades to white towards the surrounding edges.
   - **Image:** The name of the image to use as the background. The image is resized to fit within the chart area. The color of the pixel in the lower left corner of the image is used as the transparency mask for the image. Click the ellipses button to browse for and select an existing image.
5. In the **Plot** area, configure the colors or image to use for the chart plotting area. Options are the same as for the **Background** colors.
6. In the **Border** area, configure the appearance of the chart's border.
   - **Color:** Click to select or configure a color.
   - **Width:** The width, in pixels, of the border line.
   - **Type:** The style of the border line.
7. In the **Grid** area, configure the color for the grid lines of the chart. Options are the same as for **Border**.
8. In the **Highlighting** area, configure the color and pen width to be used for tag highlighting.
   - **Highlight color:** Click to select or configure a color for highlighting the tag curve.
   - **Pen width:** Specify how wide (in pixels) a highlighted curve should be.

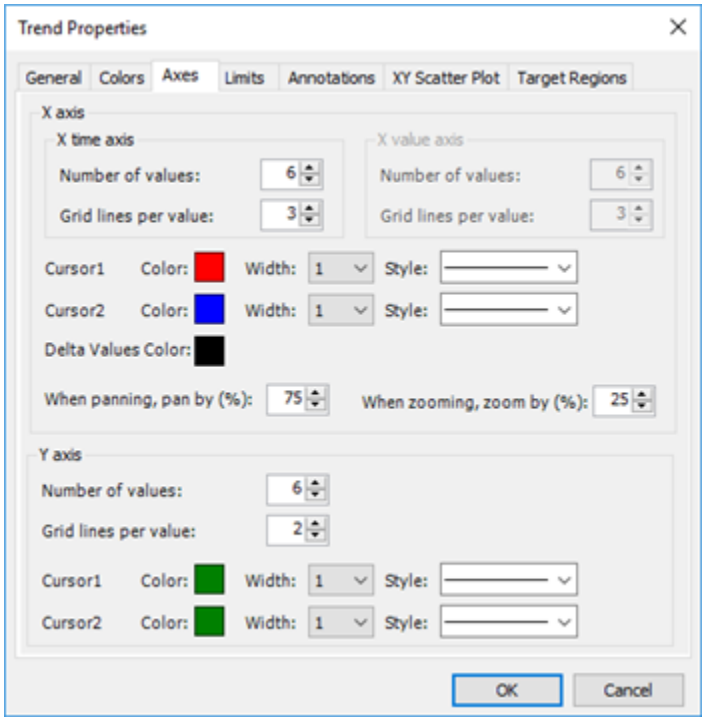
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

9. Click **OK**.

Configuring Axis Properties

To configure axis properties

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Select the **Axes** tab.



3. In the **X axis** area, configure the properties for the horizontal axis.
   - **Number of values:** The number of values that are shown along the time axis. The values are shown at evenly-spaced points along the axis. The number of values remain the same even if you zoom in and out. The valid range is from 2 to 15, with a default value of 6.
   - **Grid lines per value:** The number of grid lines that appear between each tag value plotted on the chart. The valid range is from 1 to 20, with a default value of 3.
   - **Cursor1 Color, Cursor2 Color:** Click to select or configure the color for each time axis cursor.
   - **Delta Values Color:** Click to select or configure the color of the Delta cursor label displayed above the plot area in the upper-right corner.
   - **Width:** The width of each time axis cursor.
   - **Style:** The line style for each time axis cursor.
   - **When panning, pan by:** The percentage used for the panning scale. The panning scale range is from 1 to 100.
   - **When zooming, zoom by:** The percentage used for the zoom. The zoom factor range is from 1 to 100.
4. In the **Y axis** area, configure the properties for the vertical axis.
   - **Number of values:** The number of values that are shown along the value axis. The time stamps are shown at evenly-spaced points along the axis. The number of values remain the same even if you zoom

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

in and out. The valid range is from 2 to 15, with a default value of 6.

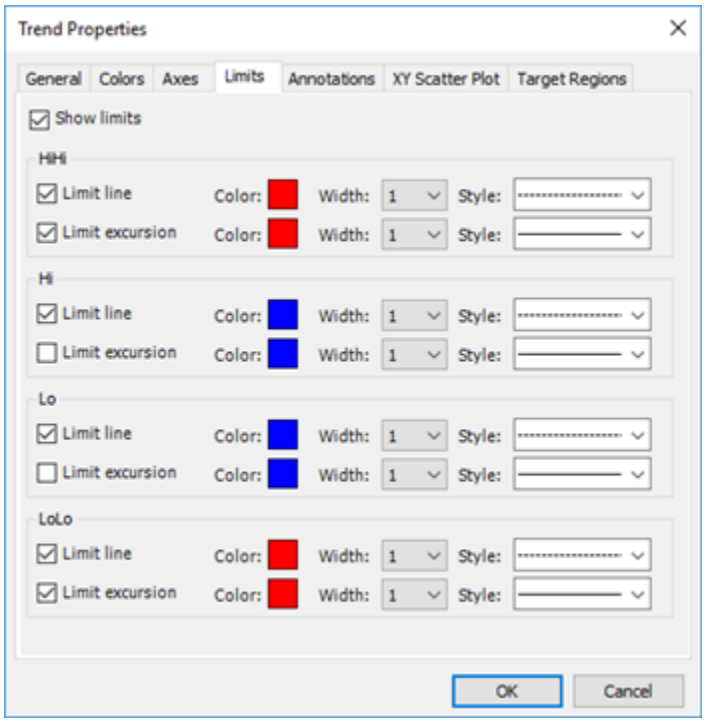
- **Grid lines per value:** The number of grid lines appearing between each tag value that is plotted on the chart. The valid range is from 1 to 20, with a default value of 3.
- **Color:** Click to select or configure the color for each value axis cursor.
- **Width:** The width of each value axis cursor.
- **Style:** The line style for each value axis cursor.

5. Click **OK**.

Configuring Limit Properties

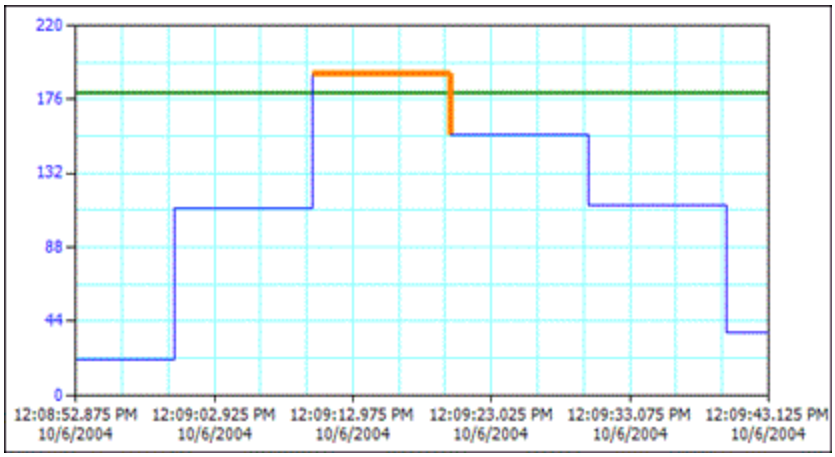
To configure limit properties

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Select the **Limits** tab.



3. Select the **Show Limits** check box to show horizontal lines on the chart at the limit values configured for analog tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



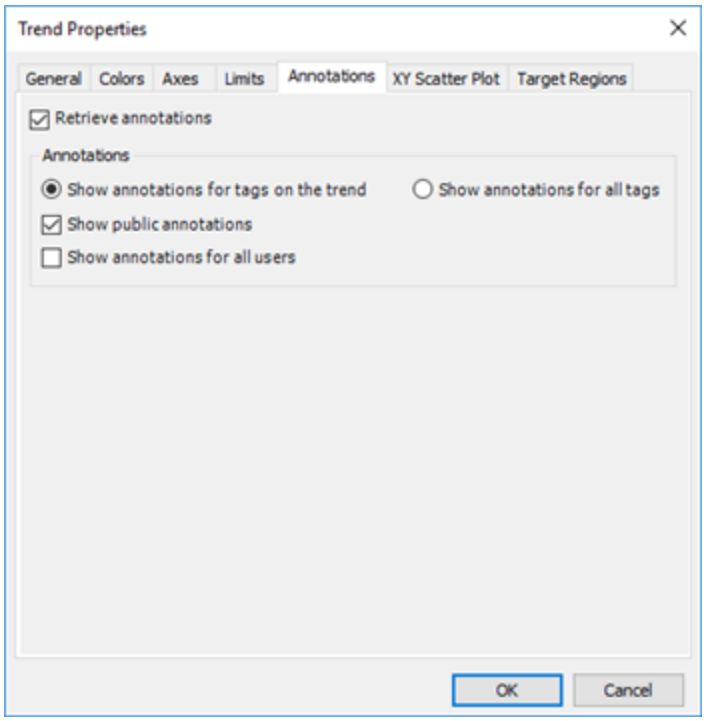
4. For each type of limit (HiHi, Hi, Lo, and LoLo), configure the properties of the line.
   - **Limit line:** Select to include a line on the chart for the limit value. For example, if an analog tag has a Hi limit of 1800, a horizontal line is drawn at the 1800 mark on the vertical axis.
   - **Limit excursion:** Select to indicate the portion of the trace that is outside of the limit.
   - **Color:** The color of the line.
   - **Width:** The width of the line.
   - **Style:** The style of the line.
5. Click **OK**.

Configuring Annotation Properties

To configure annotation properties

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Select the **Annotations** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Select the **Retrieve annotations** check box to retrieve annotation information and show them on the chart.
4. In the **Annotations** area, configure how annotations are shown on the chart.
   - **Show annotations for tags on the trend:** Show only the annotations for the tags currently charted in the trend.
   - **Show annotations for all tags:** Show all annotations for all tags. For those tags not currently charted on the trend, the annotation marker appears at the top of the chart at the point in time on chart at which the annotation was made.
   - **Show public annotations:** Show only public annotations. You can see your private annotations and the public annotations of other Historian users.
   - **Show annotations for all users:** Show both public and private annotations. You can see your private annotations, as well as both the public annotations and private annotations of others.
5. Click **OK**.

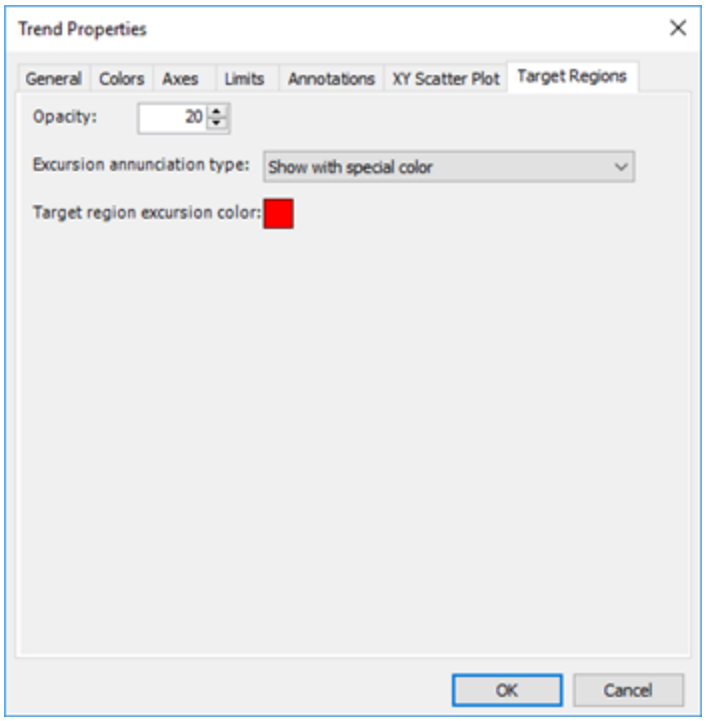
For information on the **XY Scatter Plot** tab, see Configuring Scatter Plot Properties.

Configuring Target Region Properties

To configure target region properties

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Select the **Target Regions** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



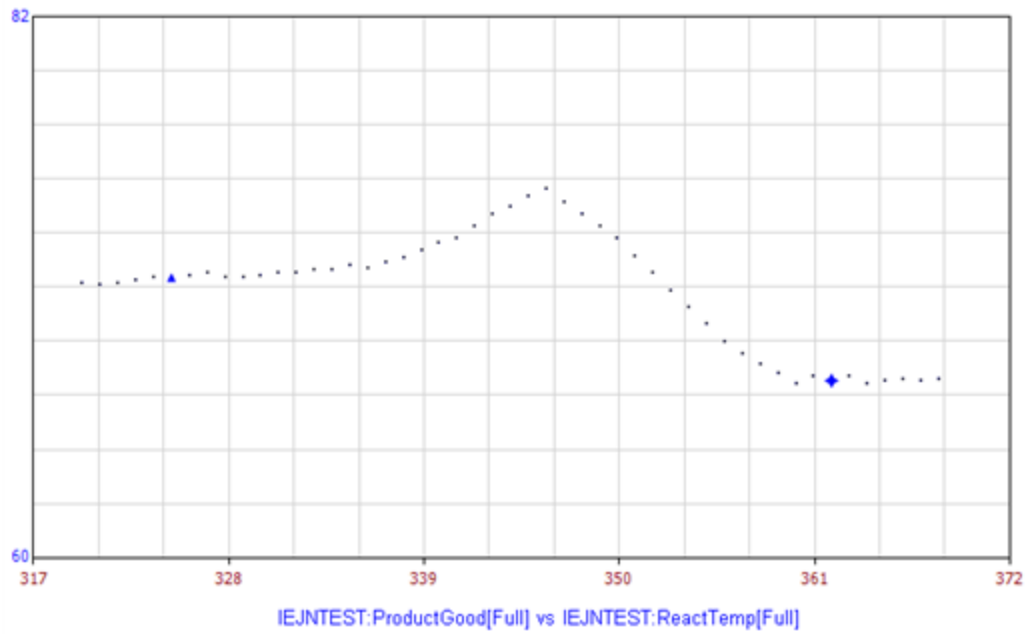
3. In the **Opacity** box, enter the opacity with which you want the target region to appear on the trend chart.
4. In the **Excursion annunciation type** list, specify whether values that fall outside the target region should be highlighted. Select **Show with special color** to highlight parts of the trend graph that are outside the target region in a special color. To select the color, click the color box next to **Target region excursion color**. Select **None** if you do not want any special highlighting.
5. Click **OK**.

Working with Scatter Plots

In addition to regular trends, you can display data in XY scatter plots. While a regular trend shows the variation of a tag's value over time, a scatter plot shows the variation of a tag's value over the variation of another tag's value. This allows you to see correlations between the two tags.

For example, you could show how product yield varies depending on the reactor temperature in a manufacturing process, and use this information to determine the optimum temperature:

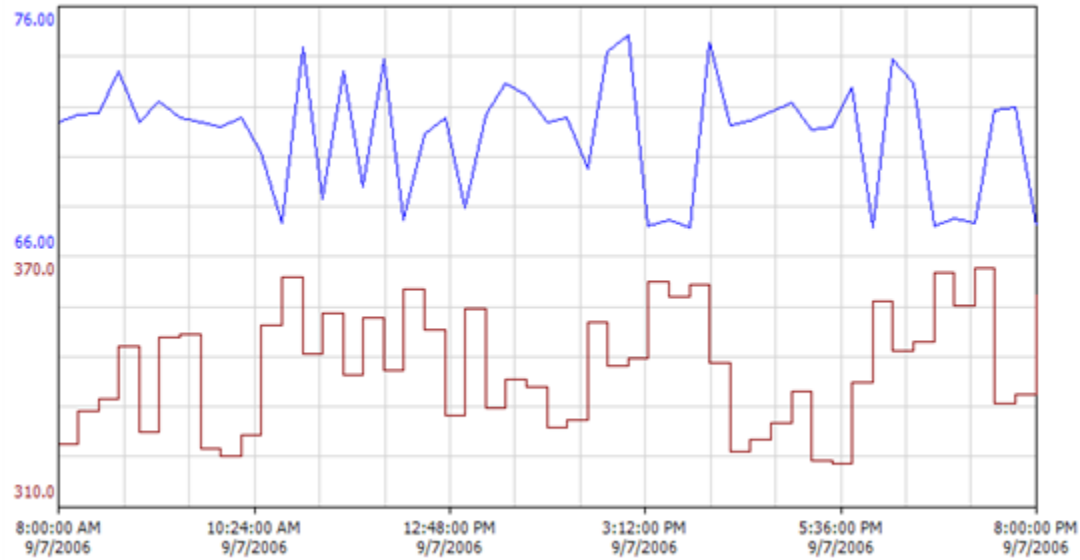
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In this example, the X axis represents the reactor temperature as historized by the ReactTemp tag (the "X axis tag"). The Y axis represents the product yield as historized by the ProductGood tag (the "Y axis tag"). For each available data sample of either tag during the chosen time period, a corresponding value for the other tag is matched or interpolated and plotted on the chart.

For more information, see How Are Value Pairs Matched?.

Plotted over time, the two tags look like this:



Compared to this type of display, the scatter plot shows the correlation much more clearly.

The following sections show you how to configure a scatter plot and manipulate the display. Many of these features work in a regular trend. Therefore, these sections mainly explain the specific differences when working with scatter plots.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Viewing Data in a Scatter Plot

Scatter plots show value pairs. As in a geometric coordinate system, every data point in scatter plot must have an X value that determines its horizontal position as well as a Y value that determines the vertical position. On a regular trend, there is no such thing as an "X value" that corresponds to the Y value of a tag; instead, the horizontal position of a tag's value on the chart is determined by the value's time stamp.

On a scatter plot, however, both the X and the Y values must be supplied as tag data. Therefore, you must assign an X axis tag to every tag that you want to view in the scatter plot.

Tags without a corresponding X axis tag are visible in the Tag List, but not in the chart. For more information on how X and Y values are matched, see How Are Value Pairs Matched?

To configure a scatter plot

1. Click the **New Chart** toolbar button. A new trend chart appears.
2. On the **Chart** menu, in the **Layout** group, select **Scatter**.

   The chart switches into scatter plot mode.
3. Add tags to the chart by double-clicking them in the Tag Picker or dragging them onto the Tag List. For more information on the Tag Picker, see Tag Picker.

   You must add all tags that you want to use as X or Y axis tags. Note the following:
   - One tag can serve as the X axis tag for multiple other tags.
   - If you want to view the same tag against different X axis tags, add it to the Tag List multiple times.
   - While you can add string or event tags, they serve no purpose in a scatter plot. Therefore, these tags are automatically marked as hidden.
4. Specify a time period for the chart using the time toolbar.
5. Assign an X axis tag to every tag that you want to view in the scatter plot:
   a. Double-click the tagname in the Tag List. The **<ServerName:Tagname>** dialog box appears with the **General** tab selected.
   b. In the **X axis** Tag List, click the name of the tag that you want to use as the X axis tag for this tag. To remove an existing X axis tag association, click the blank entry instead.
   c. Configure other tag options as required. For more information, see Configuring Trend Options for a Tag.
   d. Click **OK**.

   Data for the X/Y tag pairs is retrieved for the specified time period and plotted in the chart. The oldest value pair appears as a triangle-shaped point, and the latest value pair as a diamond-shaped point.

   Tags that do not have an X axis tag assigned to them are shown in italics at the end of the Tag List.

To quickly assign an X axis tag to a tag

If you do not need to configure any other tag settings, you can use the following steps to quickly assign an X axis tag to a tag that you want to display in a scatter plot (the Y axis tag).

1. Add the Y axis tag to the chart.
2. With the Y axis tag selected in the Tag List, drag the X axis tag from the Tag Picker onto the X axis of the chart.

Alternatively, use these steps:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

1. Add the X and Y axis tags to the chart.
2. In the Tag List, click the X axis tag's name and drag it onto the **X Axis Tag** column of the Y axis tag.

Scaling Tags in a Scatter Plot

Scaling tags in a scatter plot works much like scaling tags in a regular trend. For more information, see Scaling Tags. Note the following:

- Scaling a tag affects the display of all tags that use it as their X axis tag. The display of all other tags remains unchanged. If you want to plot multiple tags against the same X axis tag, but with different X axis scales, you must add the X axis tag to the chart multiple times and assign each tag a different instance of the X axis tag. You can then scale the various instances of the X axis tag individually.
- Rubber band scaling always affects all tags in the chart. It applies to both X axis and Y axis tags. It is not possible to use rubber band scaling for single tag. Rubber band scaling does not affect the chart's time period.
- The scale of the X axis changes as you select different tags in the Tag List. It reflects the scale of the X axis tag associated with the selected tag, or the scale of the tag itself if it does not have an X axis tag. The "multiple scales" option has no effect on the X axis.
- It is not possible to use cursor values as axis labels in a scatter plot.
- Stacking traces is not possible in a scatter plot.

Configuring Axes in a Scatter Plot

Configuring the axes of a scatter plot works much like configuring the axes of a regular trend. For more information, see Configuring Axis Properties. Note the following:

- On the **Axes** tab of the **Trend Properties** dialog box, the **X time axis** area only applies to regular trends. For scatter plots, use the **X value axis** area instead.

How Are Value Pairs Matched?

To plot a data point, the scatter plot must determine which Y value belongs to a given value of the X axis tag and vice versa. This is easy if there are data samples available with the same time stamp for both the X axis tag and the Y axis tag. If there is a sample available for one tag (Tag 1) at time T, but not for the other tag (Tag 2), the missing value is calculated based on the following rules:

- If Tag 2 uses a curve type of "Point" or "Step Line", then the data point uses the latest sample of Tag 2 that is earlier than T.
- If Tag 2 uses a curve type of "Line", then the data point uses the result of a linear interpolation between the two samples of Tag 2 that surround T.

For example, assume you have the following samples available for two tags. Tag 1 uses a trace type of "Step Line." Tag 2 uses a trace type of "Line." A dash indicates that there is no sample at that point in time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time	Value of Tag 1	Value of Tag 2
t_1	x_{t1}	y_{t1}
t_2	—	y_{t2}
t_3	x_{t3}	—
t_4	x_{t4}	y_{t4}

According to the rules above, the missing value of Tag 1 at t_2 is assumed to be x_{t1}. The missing value of Tag 2 at t_3 is calculated using a linear interpolation between y_{t2} and y_{t4}.

If either tag has a NULL sample at a given point in time, the data point is considered "empty," which may result in a gap in the curve.

Quality Calculation for Data Points

In the chart display, data points of uncertain, bad, or unknown quality are visually highlighted with special indicators. The overall quality of a data point in a scatter plot depends on the quality of the two tag values of which it is composed. The following table shows the overall quality that results from each possible combination of tag qualities, assuming that both tag values aren't NULL. The top row contains the quality of the first tag, the left column contains the quality of the other tag.

	Unknown	**Uncertain**	**Bad**
Good	Unknown	Uncertain	Bad
Unknown	Unknown	Uncertain	Bad
Uncertain	Uncertain	Uncertain	Bad
Bad	Bad	Bad	Bad

For example, if one tag has good quality and the other tag has bad quality, the data point is highlighted with the indicator for bad quality.

Panning and Zooming in a Scatter Plot

Panning and zooming affect the time period used in a chart. For a scatter plot, this means that panning or zooming moves, enlarges or reduces the time period for which data is retrieved. This may result in more or fewer data points being available for display. Depending on the nature of the data, this may or may not change the visual appearance of the chart—unlike in a regular chart, where panning or zooming inevitably changes the display.

To reflect this, the panning commands in the **Chart** menu are called **Pan Earlier** and **Pan Later** in a scatter plot, as opposed to **Pan Left** and **Pan Right** in a regular trend. However, they still work the same way. The zooming options are identical. For more information, see Panning in the Trend Chart and Zooming.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Defining a Target Region for a Scatter Plot

You can configure a target region for each tag displayed in a scatter plot as you configure in a regular trend. For an overview of what a target region does, see Defining a Target Region for a Tag.

Configuring a target region for a scatter plot tag is very similar to configuring one for a regular trend. The main difference is that the target region isn't defined by high and low boundaries at certain points in time, but by a series of X/Y value pairs. The target region is determined by connecting the X/Y points in the order they are given. For some examples, see Examples for Target Regions in Scatter Plots.

To configure a target region for a scatter plot tag

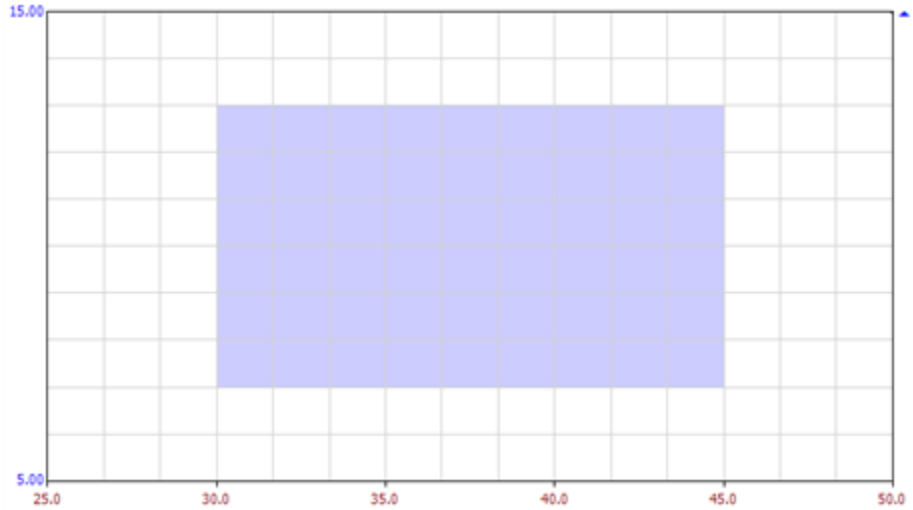
- Follow the procedure given under To configure a target region for a trend tag. The only difference is that when importing a CSV file or pasting clipboard data, each row must contain a region item that is composed of two items instead of three. The first item is the X value, the second item is the Y value.

Examples for Target Regions in Scatter Plots

When defining a scatter plot target region, listing the same X/Y points in different order can result in very different target regions. For example, assume that you define the following X/Y points:

X	Y
30	7
30	13
45	13
45	7

The resulting target region looks like this:

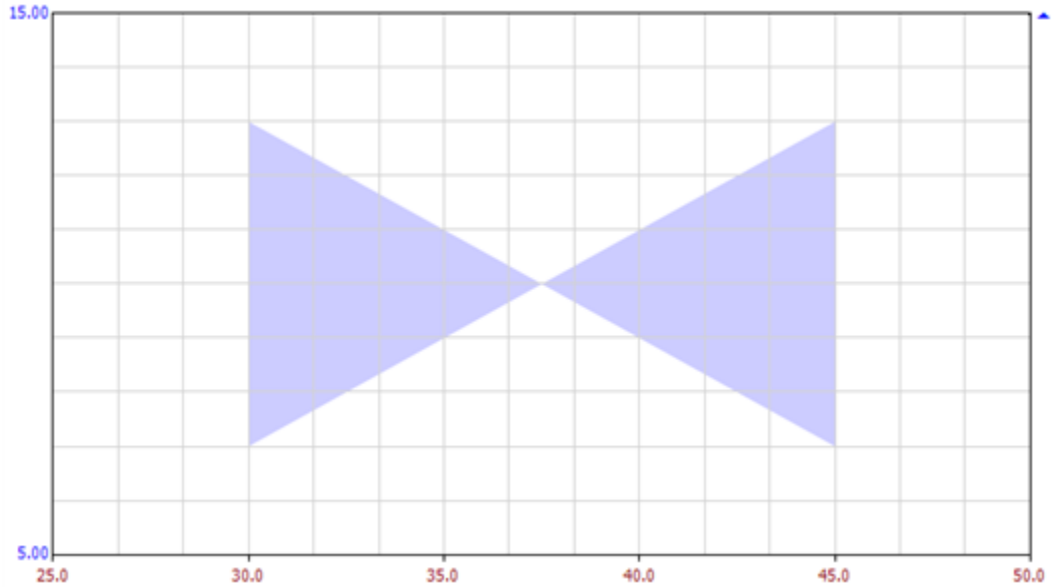


Because the points are connected in the order they are defined, reordering the points results in a different target region. Assume that you reorder the same points like this:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

X	Y
30	7
30	13
45	7
45	13

The resulting target region looks like this:

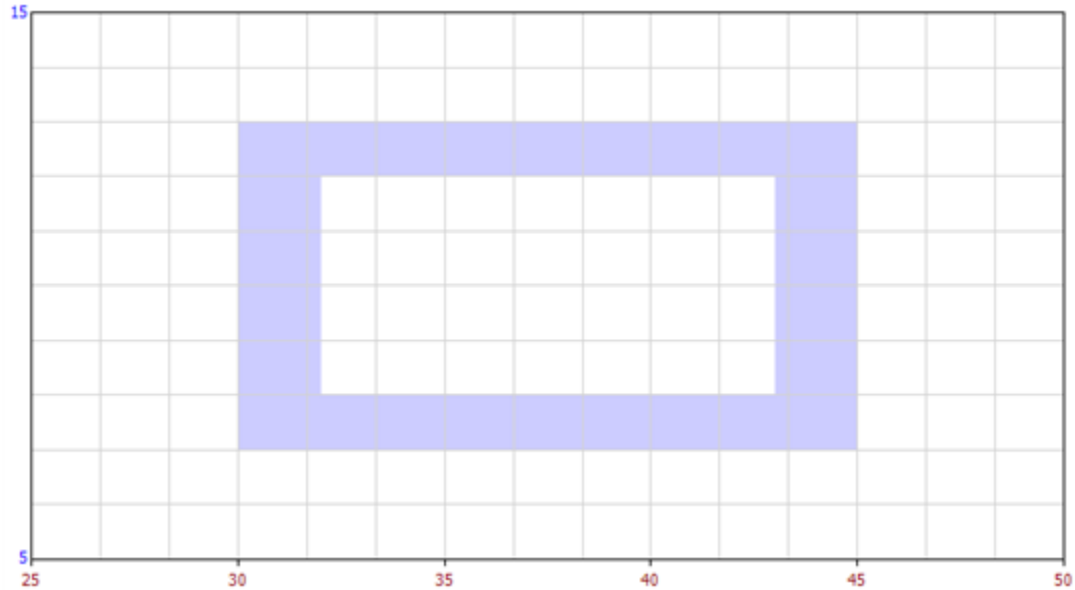


You can also create target regions with a "hole" in the middle. For example, use the following points:

X	Y
30	7
30	13
45	13
45	7
30	7
32	8
32	12
43	12
43	8
32	8

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The resulting target region looks like this:



Configuring Scatter Plot Properties

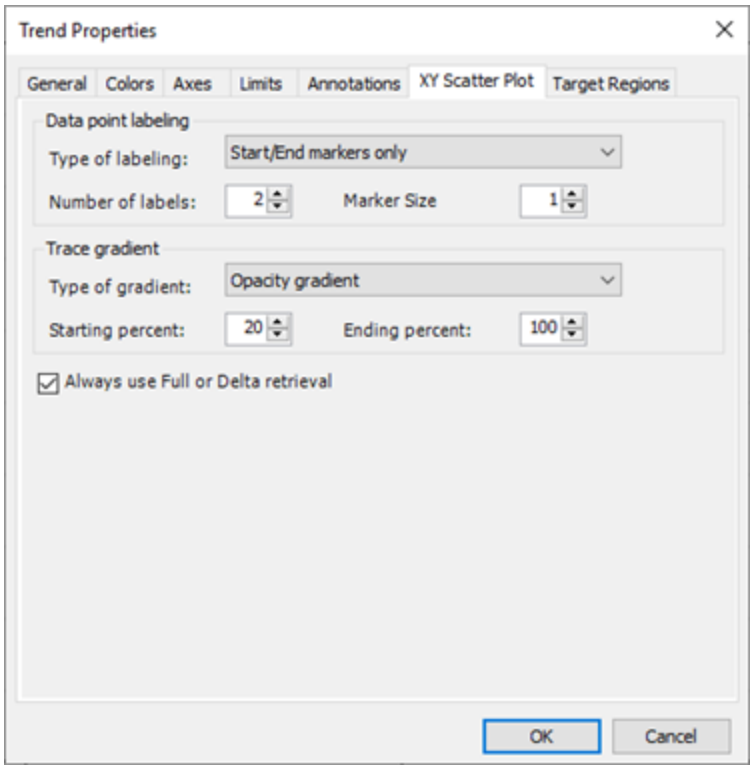
You can set up a scatter plot to show time labels on the data points and use an opacity gradient for line traces to indicate the sequence of data points in time.

Also, you can override the tags' data retrieval settings so that full or delta mode retrieval is always used in scatter plots.

To configure scatter plot properties:

1. On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
2. Click the **XY Scatter Plot** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. In the **Data point labeling** area, configure the following options:

• Type of labeling	Selected Option	Result
	Start/End markers only	Start/end markers are displayed, with no time labels on the data points
	Start/End markers with time labels on current tag	Start/end markers are displayed, along with time labels on the data points
	Start marker only	Start markers are displayed, but end markers are not displayed
	End marker only	End markers are displayed, but start markers are not displayed
	No marker and time labels	No markers or time labels are displayed
• Number of labels	The number of time labels that appear on the chart. The valid range is from 2 to 15, with a default value of 6. The labels are spaced evenly over the time period between the earliest and the latest data point in the chart. (Therefore, they may not be spaced evenly in distance.)	
• Marker Size	The size to display the start/end markers in the chart. The valid range increases in size from from 1 to 5, with a default value of 1.	

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

4. In the **Trace gradient** area, configure the following options:

• Type of gradient	Select **Opacity gradient** if you want the opacity of the line trace in a scatter plot to change based on the time stamp of the data points that it connects. For example, the trace could be fully opaque at the most recent data point and almost transparent at the oldest data point. Select **None** if you want the trace to have the same opacity at all data points.
• Starting percent	The opacity of the trace at the oldest data point. 0 means transparent, 100 means fully opaque.
• Ending percent	The opacity of the trace at the most recent data point.

5. Select the **Always use Full or Delta retrieval** check box if you always want to use Full or Delta mode retrieval for all tags in a scatter plot regardless of the retrieval settings that are configured at the application or tag level. Full retrieval is used when retrieving data from a Historian with a version of 9.0 or higher. Delta retrieval is used for earlier server versions. We recommend to keep this option enabled unless the nature of your data makes full retrieval impractical.
6. Click **OK**.

Other Considerations for Working with Scatter Plots

Also note the following when working with scatter plots:

- **Retrieval:** If a tag is neither associated with an X axis tag nor acting as an X axis tag itself, no data is retrieved for it. Therefore, the data logs do not show any data for such tags.
- **Cursors:** You can work with cursors similar to regular trend. However, the cursor commands in the **View** menu are called **X Value Axis Cursors** and **Y Value Axis Cursors** instead of **Time Axis Cursors** and **Value Axis Cursors**.
- **Curve type:** If a tag has a curve type of "Line" or "Step Line," its data points are connected in chronological order. Depending on the nature of the data, changing the curve type to "Point" may result in a clearer display.
- **Switching between chart types:** Trend options that aren't applicable to a scatter plot are disabled and/or ignored when you switch the chart type from regular trend to scatter plot. However, their values are generally retained so that they are still available if you switch the chart type back to regular trend.

Outputting Trend Data

You can output trend data to a printer, a .csv file, or to an image file, such as .bmp, jpeg, and .gif. You can also copy and paste the trend graph and associated Tag List to the Windows Clipboard.

Note: The capability to output a trend to an image file is available only in System Platform 2020 R2 SP1 and later.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Printing Trend Data

Before you print a chart, you can specify print options and preview the printout. Use the following commands:

- To set up the print page: From the **Home** menu, in the **Print** group, click **Setup**.
- To preview the print output: From the **Home** menu, in the **Print** group, click **Preview**.
- To print the chart: From the **Home** menu, in the **Print** group, click **Print**.

The available options for these commands work like in any other Windows application.

Note: When you print a chart, only the data that is currently displayed in the application appears in the printout. For example, if part of the Tag List is not displayed in the application, then that portion does not appear in the printout.

Printing Trend Sets

A trend set is a saved grouping of trend files. You can specify a common trend duration (for example, the last 24 hours) to apply to all of the files in the set.

This allows you to easily print information for the same duration from multiple trend files at the same time.

Printing a Trend Set

To print a trend set

- From the **File** menu, in the **Print** group, click **Print**.

If you double-click a trend set in Windows Explorer, the trend set opens in the Historian Client Trend application, the associated trends are printed, and then the application closes automatically.

You can also print a trend set from a command prompt by executing the trend set filename, including the extension:

```
aatrend /s <fully qualified filename>
```

To have the Trend application automatically close after the trend set is printed, omit the `/s` parameter.

Saving Trend Data to a .CSV File

When you save trend data, all data is exported to comma separated values (.csv) format. The .csv file includes all time stamps and values for the tags on the current trend chart at the time of the save.

To save trend data

1. From the **File** menu, select **Save**.
2. Select **Save data**.

   The standard Windows **Save As** dialog box appears.
3. Browse to the location in which you want to save the file.
4. In the **File name** box, type a name for the trend data file.
5. Click **Save**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

You can view the data in any spreadsheet application that can open .csv files, such as Microsoft Excel.

If you do not want the data values to be blank unless the value is NULL, use the Cyclic retrieval style and add the following entry into the win.ini file:

```
[HistClient]
KeepAllTrendPoints=1
```

Saving the Trend Chart to an Image File

You can save a trend chart to a .bmp, .gif, .jpeg, .svg, or .png image file.

Note: The capability to output a trend to an image file is available only in System Platform 2020 R2 SP1 and later.

To save the trend chart

1. From the **File** menu, select **Save**.
2. Click **Save Image**.
   The standard Windows **Save As** dialog box appears.
3. Browse to the location in which you want to save the file.
4. In the **File name** box, type a name for the trend image file.
5. In the **Save as Type** box, select an image type.
6. Click **Save**.

E-mailing a Trend File

To e-mail a trend file, you must have a valid SMTP server and account configured and an e-mail application correctly installed and configured with a suitable e-mail account.

Before you e-mail a trend, make sure to save the trend file on your computer.

To e-mail a trend file

1. From the **Home** menu, in the **Export** group, select **Send To**. The e-mail program starts up and a new message appears.

   Note: By default, the trend file that you want to send does not have a *.aaTrend file extension.
2. Remove the attachment and then browse to the location and attach the trend file that includes the trend *.aaTrend file as an extension.
3. Use the e-mail program to send the trend file.

Copying a Trend Chart to the Windows Clipboard

When you copy a trend chart, only the data that is currently displayed in the application is copied. For example, if part of the associated Tag List is not displayed in the application, then that portion is not copied.

To copy a trend chart

1. Click in the trend chart so that it has focus.
2. From the **Home** menu, in the **Clipboard** group, click **Copy**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Open the target application (for example, Microsoft Word).
4. Paste the chart.

   The trend chart is pasted as a graphic in the target application.

Publishing Trends to the AVEVA Information Server

You can publish trends to the AVEVA Information Server. When you publish a trend, the trend report information is stored in special tables in the AVEVA Historian, and the file is copied to a folder on the AVEVA Information Server. When you publish a trend, AVEVA Information Server users can view the trend you published with only an Internet browser.

Published trends are of two types:

- **Static**. For a static trend report, AVEVA Information Server users see the same trend, but cannot alter the trend configuration in any way. They can, however, perform some basic navigation functions, like panning and zooming.
- **On Demand**. For an "on demand" report, AVEVA Information Server users see the same trend, but can fully manipulate the trend, including changing the configuration. However, any changes made to the original trend are not saved.

Published trend files contain the configuration information for the trend, but not the actual data values that are trended. For both types of reports, when the trend appears on the AVEVA Information Server, data is retrieved from the AVEVA Historian database and appears in the trend chart.

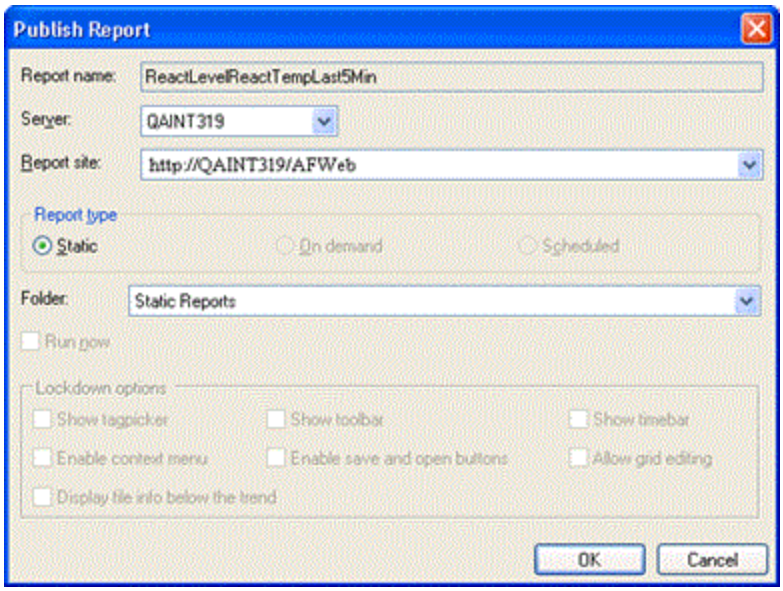
The AVEVA Information Server must be associated with the same AVEVA Historian as the trend you want to publish. If a trend references multiple AVEVA Historians, the AVEVA Information Server must be associated with at least one of the AVEVA Historians to publish.

Publishing a Static Trend Report

To publish a static trend report

1. Create a trend and save it as an .aaTrend file.
2. On the **File** menu, point to **Publish**, and then click **Static Trend**. The **Publish Report** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The **Report name** box shows the name of the trend report as it appears on the AVEVA Information Server. This name is automatically created based on the name of the file that you are publishing.

3. In the **Server** list, click the name of the AVEVA Historian on which to store the report publishing information.
4. In the **Report site** list, select the URL of the AVEVA Information Server to which you want to publish the trend.

   The AVEVA Information Server may or may not be physically located on the AVEVA Historian computer.
5. In the **Report type** area, click **Static**.
6. In the **Folder** list, click the name of the physical folder on the report site where the static report is posted.
7. Click **OK**. The **Report successfully published** dialog box appears.

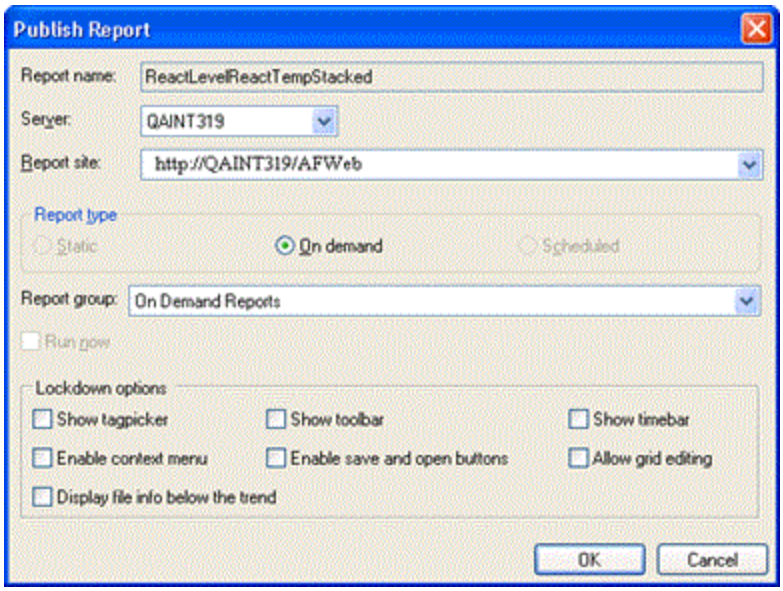
   Note: The AVEVA Information Server periodically scans the publishing folders for changes. A short delay may occur prior to the published report being shown on the AVEVA Information Server.
8. To view the AVEVA Information Server, click **Browse**. Otherwise, click **Done**.

Publishing a Dynamic Trend Report

To publish a dynamic trend report

1. Create a trend and save it as an .aaTrend file.
2. On the **File** menu, point to **Publish**, and then click **Dynamic Trend**. The **Publish Report** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The **Report name** box shows the name of the trend report as it appears on the AVEVA Information Server. This name is automatically created based on the name of the file that you are publishing.

3. In the **Server** list, click the name of the AVEVA Historian on which to store the report publishing information.
4. In the **Report site** list, select the URL of the AVEVA Information Server to which you want to publish the trend.

   The report site may or may not be physically located on the AVEVA Historian computer.
5. In the **Report type** area, click **On demand**.
6. In the **Report group** list, click the name of the physical folder on the report site where the static report is posted.
7. In the **Lockdown options** area, select the check boxes for the functionality you want to allow in the published trend report.

   For example, if you want AVEVA Information Server users to be able to change the report using the Tag Picker, select the **Show tagpicker** check box.
8. Click **OK**. The **Report successfully published** dialog box appears.

   Note: AVEVA Information Server periodically scans the publishing folders for changes. A short delay may occur prior to the published report being shown in the server.
9. To view the AVEVA Information Server, click **Browse**. Otherwise, click **Done**.

Using Trend with a Tablet PC

When you run the Trend application on a Tablet PC, you can capture an image of the application window or the chart area, annotate the image using various drawing tools, and save, print or e-mail the results.

Annotating a Chart

To annotate a chart or the application window

1. Create a trend chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. On the **Tools** menu, click **Annotate Chart** or **Annotate Application**. The **Annotate Layout** dialog box appears.
3. Use the stylus to write annotations on the image. For more information, see Making Chart Annotations.
4. Save or e-mail the file. For more information, see Saving, Printing, and E-Mailing an Annotated Chart.

Making Chart Annotations

To make annotations to the chart, use the following tools:

- **Pen:** To draw and write comments.
- **Highlighter:** To highlight areas of the chart using a semi-transparent color.
- **Eraser:** To delete parts of an annotation.

Each of these tools has certain options such as size, color, or transparency.

- To set these options, click the downward arrow next to each tool's icon and then click the command for the option.
- To restore these options to their default settings, on the **Tools** menu, click **Restore Defaults**.

Selecting, Copying, and Deleting Chart Annotations

To select annotations

1. Click the **Lasso** icon in the toolbar.
2. While holding down the stylus button, draw an area around the annotation(s) that you want to select.
   You can now cut, copy or delete the selected annotations.

To cut, copy, and paste annotations

- Use the standard Windows Cut, Copy, and Paste commands.

To delete annotations

- Do one of the following:
  - To delete all annotations on a chart, on the **Edit** menu, point to **Clear** and then click **All**.
  - To delete annotations that you selected using the lasso, on the **Edit** menu, point to **Clear** and then click **Selection**.

Saving, Printing, and E-Mailing an Annotated Chart

Once you have made annotations to a chart, you can save it as an image file, print it, or e-mail it.

To save an annotated chart

1. From the **File** menu, select **Save**.
   a. Click **Save as..** to save the chart as a trend file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

b. Click **Save data..** to save a text file containing the chart's data. Select a file type from the list.
c. Click **Save image..** to save the chart as an image file. Select an image type from the list.

2. Type a name for the file and click **Save**.

To print an annotated chart

1. From the **Home** menu, click **Print**. A standard Windows Print dialog box appears.
2. Specify any printing options and click **OK**.

To e-mail an annotated chart

Note: AVEVA Historian uses the system's default e-mail application to send e-mail. For more information about configuring default applications, refer to your Microsoft Windows documentation. If you do not have a default e-mail application configured, you cannot e-mail charts.

1. From the **Home** menu, click **Send To**. The default application for sending e-mail launches.
2. Send the e-mail according to the directions for your default e-mail application.

Importing .CRV Data

You can import data from a .crv file and display it in the trend chart. This allows you to migrate trend files from versions before ActiveFactory 9.0.

To import .crv data

1. On the **Tools** menu, click **Import**. The **Open** dialog box appears.
2. Select the .crv file to open and then click **Open**.

Client Query

The AVEVA Historian Client Query is an application that allows you to retrieve data from an AVEVA Historian database or any SQL Server database and return the results in a table format. If you are querying an AVEVA Historian, you can choose from a number of predefined query types and easily select the options for each type, eliminating the need to know SQL syntax. The SQL query is created for you.

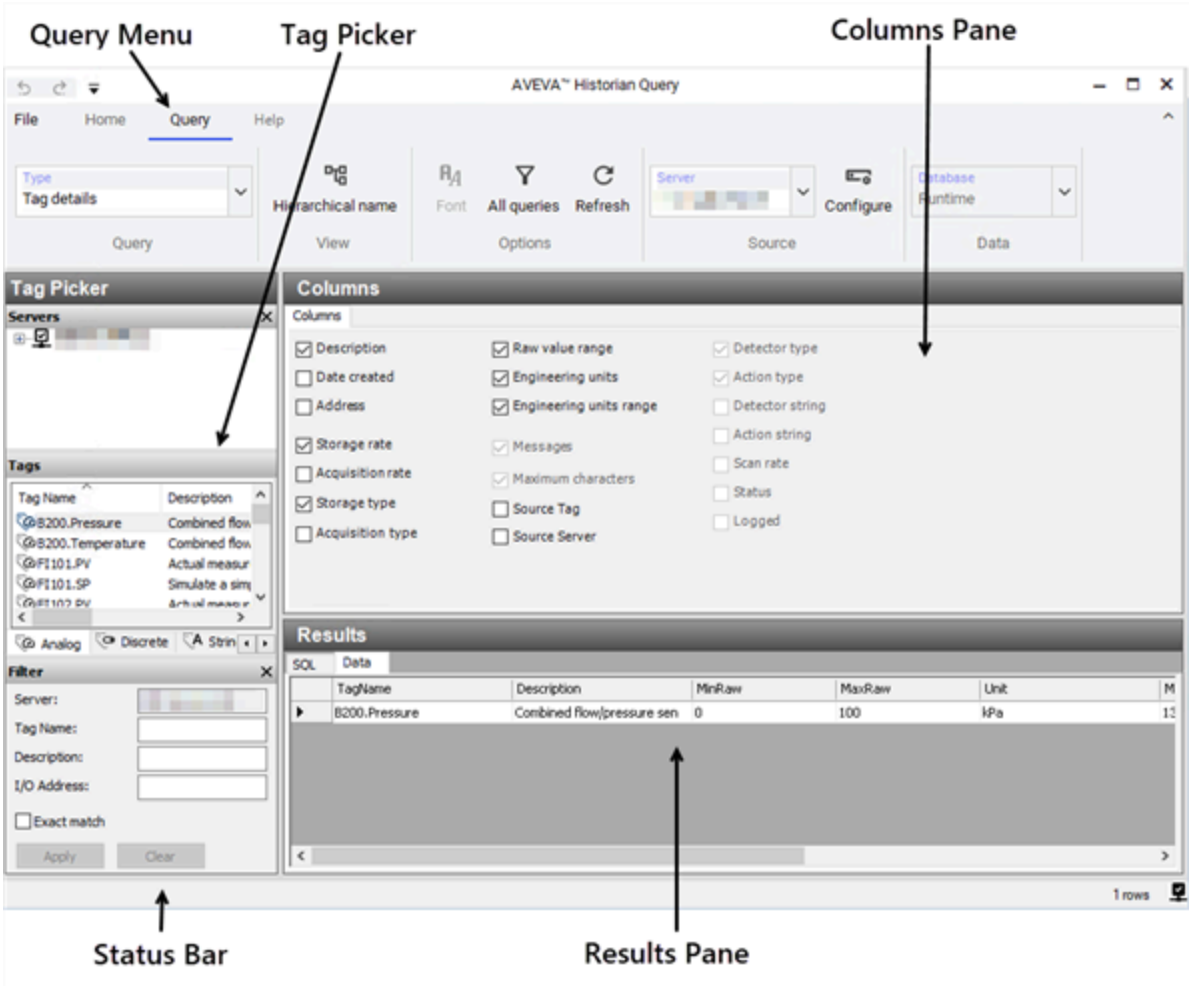
You can also write custom queries if you know SQL syntax and the schema of database you are using.

Getting Started with Query

When you start up the Query application, you are immediately prompted to connect to the server. However, if you are opening an existing Query file that includes at least one server configuration and the log in was successful, you are not prompted to log in. For more information, see Server Connection Configuration.

After you establish a connection with the server, the Query program appears, showing the main window:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The Query application user interface consists of the following components:

- Query Toolbar
- Tag Picker (may not appear depending on the query type)
- Columns Pane
- Results Pane
- Status Bar

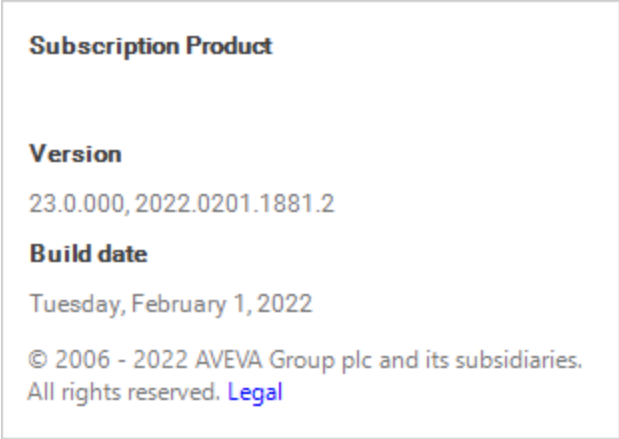
For information on using the common toolbars and the Tag Picker, see Common Client Components.

Viewing License Information

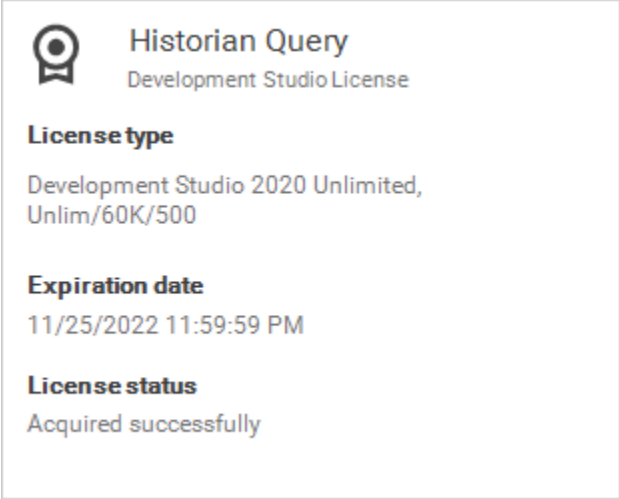
You can view information about your AVEVA Historian Client license by selecting the **File** menu, and then selecting License. Details about your license are displayed.

- The **About** section displays version information:

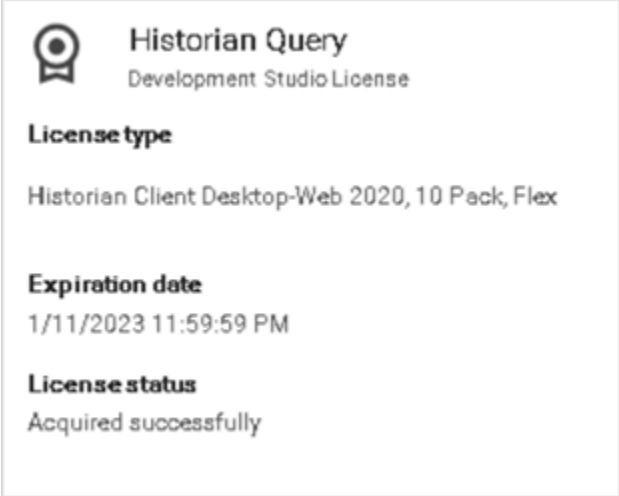
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- The **License information** section displays details about your license:



- If your license is a Flex license, the details are slightly different:

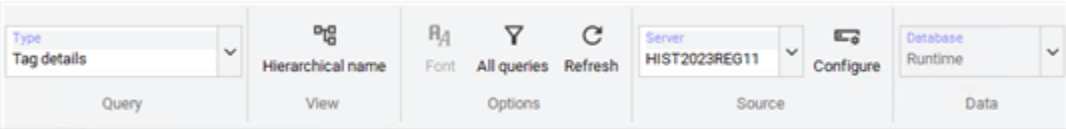


Query Menu

Use the query menu to select the query type, server, and database for the query.

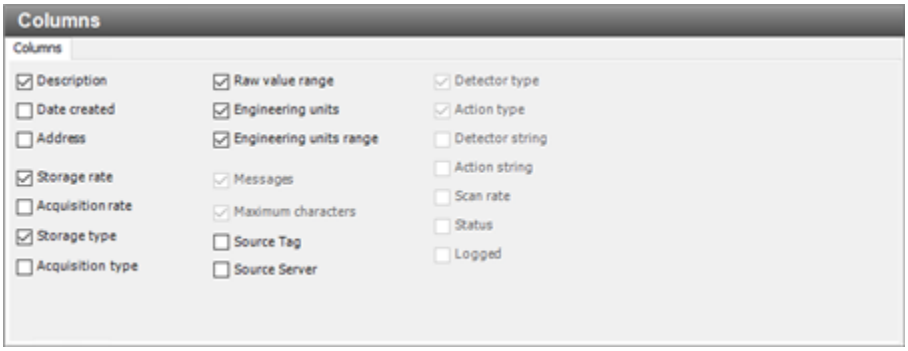
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The **Servers** list contains the list of connected servers. The **Database** list is only available if the query type is Custom.



Columns Pane

Use the **Columns** pane to select details for the query.



Results Pane

Use the **Results** pane to view the results of the query that you have created. The Results pane includes three tabs:

- **SQL tab:** Shows the complete SQL statement that is sent to the server.
- **Data tab:** Shows the data returned by the SQL statement.
- **All Queries tab:** Shows all of the SQL statements that have been created for the selected tag type for the current query. To view all the SQL statements, select **All queries** on the **Options** menu.

Viewing the Hierarchical Name in a Query

You can view the hierarchical name in a query. For more information on hierarchical names, see Integration with AVEVA Application Server.

To view the hierarchical names in a query

- Do one of the following:
  - From the **Query** menu, in the **View** group, select **Hierarchical Name**.
  - Right-click in the Tag Picker and click **Use hierarchical name**.

    The Query application shows the hierarchical names instead of the tag names. For example, **Results** pane and the **Columns** tab show hierarchical names.

Finding a Source Tag or Replicated Tag

You can replicate tag information in a AVEVA Historian from one historian to another. This allows you to replicate

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tag data from one or more historians (known as tier-1 historians) to one or more other historians (known as tier-2 historians). You can replicate tag data to the same server as the tier-1 historian.

You can replicate tag data directly using simple replication, where the tag information is replicated directly to the tier-2 historian. For simple replication, every value for a tag is copied. You can also set up summary tags that receive a summarized version of the tag data.

Use the Tag Picker to find a source tag or a replicated tag. You can drill down from a source tag to its replicated tag or drill up from a replicated tag to its source tag.

To find a source tag or replicated tag

1. Select a tag in the Tag Picker.
2. If the selected tag is a source tag, in the **Tags** pane, right-click the selected tag in the Tags pane. Point to **Find - replicated tag,** and then click the tag that you want to find.
   - The application navigates within the Tag Picker to find the corresponding replicated tag. The **SQL** tab of the **Results** pane is updated with the modified query and the **Data** tab shows the corresponding data.
3. If the selected tag is a replicated tag, in the **Tags** pane, right-click the selected tag. Then click **Find - source tag**.
   - The application navigates within the Tag Picker to find the corresponding source tag. The **SQL** tab of the **Results** pane is updated with the modified query and the **Data** tab shows the corresponding data.

The **Find** command is not available if:

- Multiple tags are selected in the Tag Picker.
- A normal tag that is neither a source tag nor a replicated tag is selected in the Tag Picker.

Note: You cannot execute the **Find** command if a source tag is deleted but its replication configuration still exists in the Historian.

The **Find** command does not navigate to the tag when the target tag is not of the type that can be shown in the current Tag Picker configuration.

For example, in the **Aggregate values** query type, the Tag Picker shows only **Analog**, **Discrete**, and **Analog Summary** tabs and you want to find a replicated state summary tag, the relevant tag is not navigated as the **State Summary** tab is not available. For example, a tag ‘Watervalve’ is replicated as a state summary tag called ‘WaterValve.S1M’ and as a simple tag called ‘MyServer.WaterValve.’ If you execute the command to find the ‘WaterValve.S1M’ tag, the application does not navigate and find the tag as the **State Summary** tab is not available.

However, if you execute the command to find the ‘MyServer.WaterValve’ tag, the application navigates and finds the tag as the **Discrete** tab is available.

The replicated tags are not listed in the context menu if:

- The replicated tags are not committed in the Historian.
- The replication schedule is removed from the Historian. For example, you are connected to a Historian 10.0 server and you create a tag called ‘MyTag’. ‘MyTag’ is replicated as a simple tag called ‘MyServer.MyTag’. When you execute the **Find - replicated tag** command, the ‘MyServer.MyTag’ tag is shown. When you execute the **Find - source tag** command, the ‘MyTag’ tag is shown. At this instance, if the replication link between ‘MyTag’ and ‘MyServer.MyTag’ is removed and if you execute the **Find - replicated tag** command, the ‘MyServer.MyTag’ tag is not shown in the list of replicated tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

However, if you execute the **Find - source tag** command, the 'MyTag' tag is shown as 'MyTag'. If 'MyServer.MyTag' is the only replicated tag, 'MyTag' is considered as a normal tag.

The above scenario holds true if the entire replication schedule is removed in the Historian. If only one replication is removed, the list shows the remaining replicated tags.

Note: If you find a source tag or a replicated tag in a server to which you are logged on but the server is currently disconnected from the network, the **Find** command finds the server but the **Tags** pane will not list the tags from that server.

Status Bar

The right side of the status bar shows the status of the AVEVA Historian. If the **Data** tab in the **Results** pane is selected, then the number of rows of result data is also shown in the status bar.

For more information on the status bar, see Status Bar.

Working with Query Files

This section describes how to open and save query files. A query file contains all of the configuration data required to execute a SQL statement against the server.

Opening an Existing Query File

To open an existing query

1. From the **File** menu, select **Open**.
2. Click the **Open** button

.

   The standard Windows **Open** dialog box appears.
3. Browse to and select the query file to open. All query files have the .sql extension.
4. Click **Open**.

Saving a Query File

The contents of a saved query file depends on which tab is currently selected in the **Results** pane. If you select the **SQL** or **All queries** tab, you can save the contents of the tab (the SQL statement) with either an .sql or .txt extension.

If you select the **Data** tab, you can save the contents of this tab (the query results) with either an .csv or .txt extension. The .csv file conforms to the locale settings of the computer and the dates are localized.

To save a query

1. From the **File** menu, select **Save**.
2. Click the **Save** button

.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The standard Windows **Save As** dialog box appears.

3. In the **Save As** dialog box, type a name for the query.

   You can select to save the query as a SQL script file or as a text file.

4. Click **OK**.

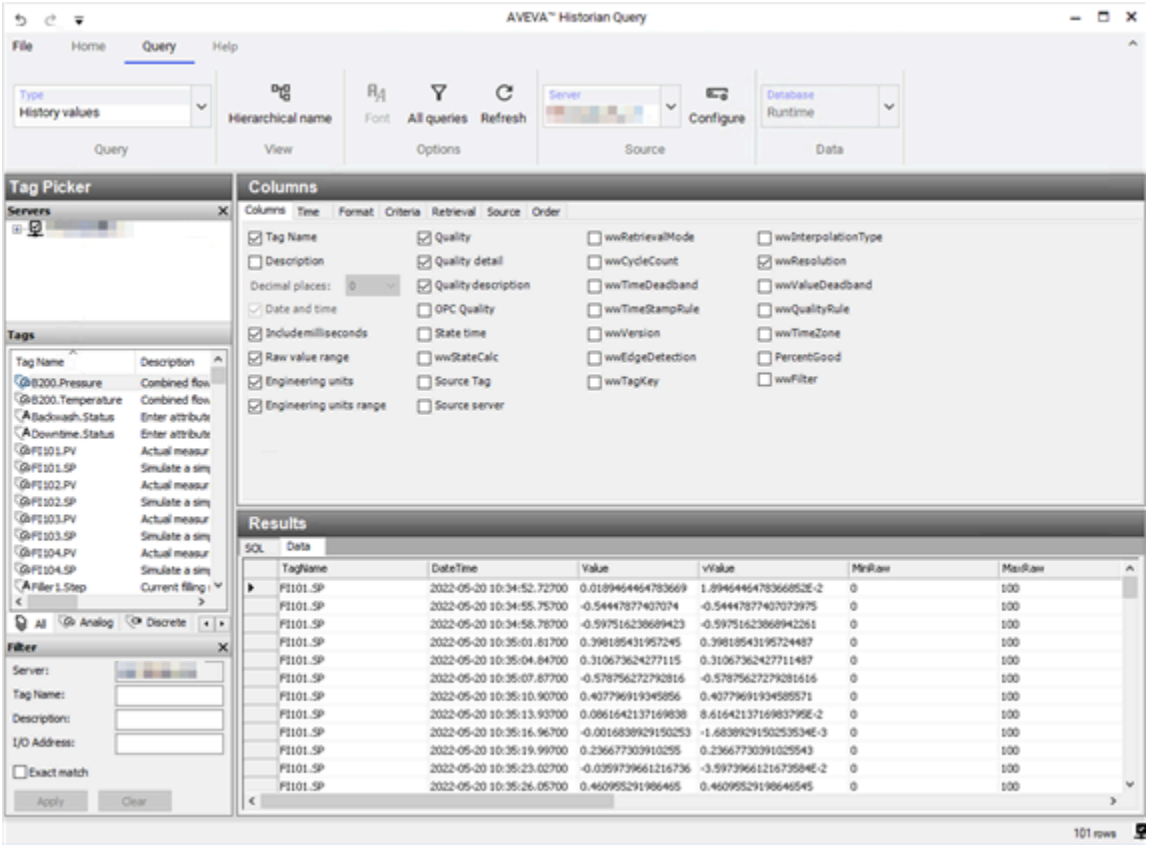
Creating a Query

When you configure a query, you must select the tag or tags for which you want to retrieve data, the type of query, and the server(s) from which to retrieve the data. The data is queried from the database to which you are currently logged on. You can also configure additional parameters that are specific to each query type.

There is no limit to the number of tags in a query; you can include as many as your system allows.

To create a query

1. Select the **Query** tab.
2. From the **Type** list in the toolbar, select the type of query you want to use as a starting point. For more information on the possible types, see Query Types.
3. From the **Server** list, select the name of the server from which you want to retrieve data.
4. Use the Tag Picker to find tags in the AVEVA Historian database that you want to include in your query. For more information on the Tag Picker, see Tag Picker.



5. In the **Columns** pane, click on each tab and configure the details for the query. The tabs that appear depend on what query type you have selected. For more information on configuring the details for a particular query type, see Query Types.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

6. In the **Results** pane, click the **Data** tab to view the resulting data.

Note: You do not have to execute the query. The Query application automatically executes the query after you switch to the **Data** tab, or if you make any changes while the **Data** tab is shown.

Query Types

The following types of queries are supported. For each query type, a set of relevant tabs appear in the **Columns** pane so that you can configure the details for the query. Some of the tabs are the same for multiple query types.

- Query Type: Aggregate Values
- Query Type: Alarm History (Data blocks)
- Query Type: Alarm Summary
- Query Type: Analog Statistics
- Query Type: Annotations
- Query Type: Custom
- Query Type: Favorites
- Query Type: History Values
- Query Type: IO Server
- Query Type: Live Values
- Query Type: Number of Tags
- Query Type: Server Version
- Query Type: Slice By
- Query Type: State Statistics
- Query Type: Storage
- Query Type: Storage Size Available
- Query Type: Storage Start Date
- Query Type: Tag Details
- Query Type: Tag Search
- Query Type: Time Running

Query Type: Aggregate Values

You can view aggregated values for specified tags. Aggregations supported are count, minimum, maximum, sum, average, and standard deviation. Aggregations are calculated using the standard SQL Server aggregation functions.

To view aggregate values

1. In the **Query Type** list in the toolbar, click **Aggregate values**.
2. Use the Tag Picker to select one or more tags.
3. In the **Columns** pane, click on each tab and configure the parameters for the query.
   - See Format Tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- See Time Tab.
- See Criteria Tab.
- See Calculations Tab.
- See Retrieval Tab.
- See Source Tab.

4. To view the results, click the **Data** tab in the **Results** pane.

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.

To configure value criteria

1. To configure criteria for a discrete tag, select the first **Value** check box and set the criteria to be either a 1 or 0. Go to Step 4.
2. To configure criteria for an analog tag:
   a. Select the first **Value** check box and set the criteria for the data value. For example, the value must be greater than (>) 1500.
   b. (Optional) Select the second **Value** check box and set another criteria for the data value. For example, the value must be less than (<) 2000.
   c. Go to Step 4.
3. (Optional) Select the **Value not null** check box to filter out NULL values from the results.
4. (Optional) In the **Quality** list, click the quality criteria for the data. Only data values that match the quality you specify (Good, Bad, Doubtful) are returned.
5. (Optional) In the **Criteria applicability** list, select the moment at which the edge detection criteria is met.
   - **first true:** Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false). This is also known as “leading” edge detection.
   - **no longer true:** Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true). This is also known as “trailing” edge detection.
   - **true:** Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
   - **first true or no longer true:** All rows satisfying both the leading and trailing conditions are returned.

Calculations Tab

Use the **Calculations** tab to configure the aggregations to perform on the values for the selected tag(s).

To configure calculations

- **Display calculated values for each tag separately:** If selected, one row of calculated values is returned for each tag. If this check box is not selected, then all values for all specified tags are included for a single aggregation.
- **Count:** The total number of values for the tag.
- **Minimum:** The minimum value for the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Maximum:** The maximum value for the tag.
- **Average:** The average value for the tag.
- **Sum:** The sum of all values for the tag.
- **Standard deviation:** The statistical standard deviation of all values for the tag.
- **Decimal places:** The number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.

Query Type: Alarm History (Data blocks)

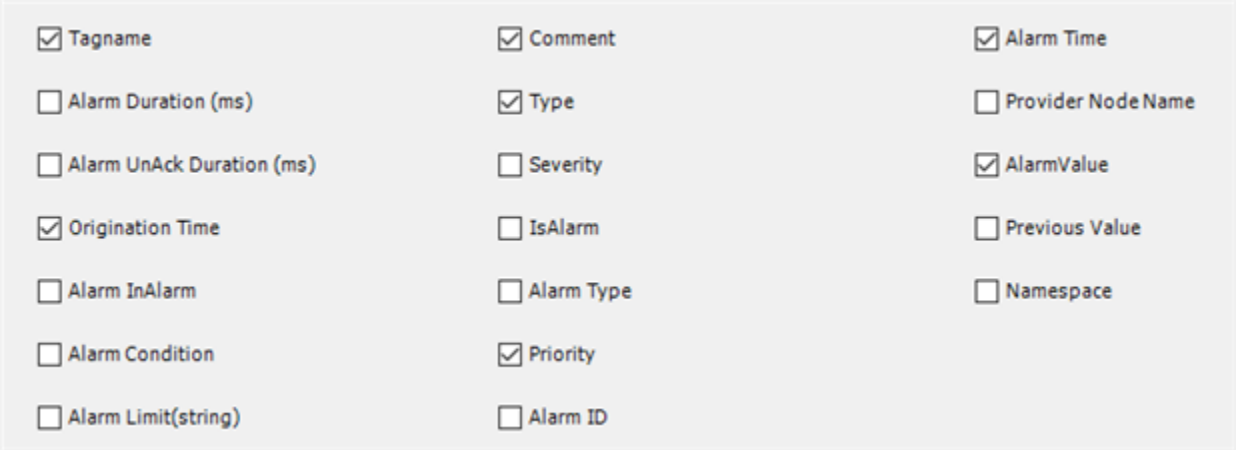
You can query alarms and events to return the alarm history for a tag. You can further scope the query to only return the tag values that are beyond an alarm limit. For example, if the Hi alarm limit for the ReactLevel tag is 1800, the alarm history can include all values that were above 1800 Hi limit.

To view alarm history

1. In the **Query Type** list in the toolbar, click **Alarm history**.
2. If you want to only retrieve alarm history for particular tag(s), use the Tag Picker to select one or more tags. For example, if you want to search for alarm history for all analog tags, select the **All Analog Tags** public group and then select all analog tags in the **Tags** pane.
3. In the **Columns** pane, click on each tab and configure the parameters for the query:
   - See Columns Tab.
   - See Time Tab.
   - See Format Tab.
4. To view the results, click the **Data** tab in the **Results** pane.

Columns Tab

Use the **Columns** tab to configure the columns that are returned for the results.



Options are as follows:

- **Tagname:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming

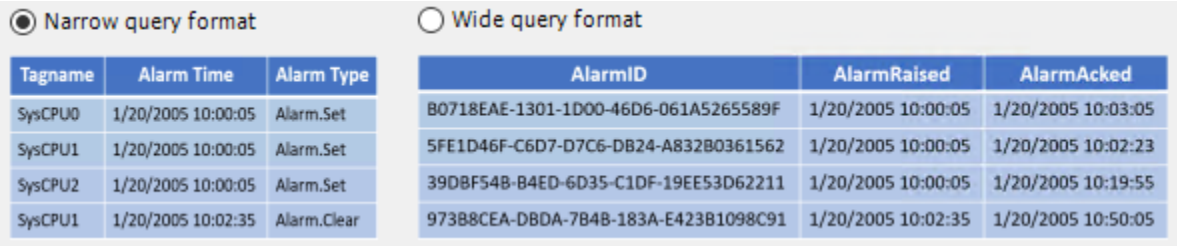
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Conventions.

- **Alarm Duration (ms):** The amount of time the alarm was active.
- **Alarm UnAck Duration (ms):** The amount of time that the alarm was unacknowledged.
- **Origination Time:** For the current alarm event, the time when the associated alarm began.
- **Alarm InAlarm:** The amount of time the tag was in alarm.
- **Alarm Condition:** The type of alarm (major/minor, high/low, rate of change (ROC), etc.)
- **Alarm Limit (string):** The alarm limit at which an alarm condition is triggered for this tag.
- **Comment:** Additional comments for the event.
- **Type:** The type of alarm event (set, clear, acknowledged).
- **Severity:** The severity of the alarm (where 1 is "critical").
- **IsAlarm:** Indicates whether the message is an alarm type message or not.
- **Alarm Type:** The event category. For example, Alarm.Set, Alarm.Clear, Alarm.Acknowledge.
- **Priority:** Indicates the importance of the event. Values range from 1 to 999, with lower numbers indicating higher importance.
- **Alarm ID:** The unique identifier for the alarm.
- **Alarm Time:** The time when the alarm began.
- **Provider NodeName:** Name of the node that generated the event.
- **Alarm Value:** The tag value that triggered the alarm.
- **Previous Value:** The value before the alarm is raised or cleared.
- **Namespace:** The hierarchical name.

Format Tab

Use the **Format** tab to specify how the results of the query are presented.



Tagname	Alarm Time	Alarm Type
SysCPU0	1/20/2005 10:00:05	Alarm.Set
SysCPU1	1/20/2005 10:00:05	Alarm.Set
SysCPU2	1/20/2005 10:00:05	Alarm.Set
SysCPU1	1/20/2005 10:02:35	Alarm.Clear

AlarmID	AlarmRaised	AlarmAcked
B0718EAE-1301-1D00-46D6-061A5265589F	1/20/2005 10:00:05	1/20/2005 10:03:05
5FE1D46F-C6D7-D7C6-DB24-A832B0361562	1/20/2005 10:00:05	1/20/2005 10:02:23
39DBF54B-B4ED-6D35-C1DF-19EE53D62211	1/20/2005 10:00:05	1/20/2005 10:19:55
973B8CEA-DBDA-7B4B-183A-E423B1098C91	1/20/2005 10:02:35	1/20/2005 10:50:05

Options are as follows:

- **Narrow query format:** In this format, there is one row per alarm event.
- **Wide query format:** In this format, there is one row per alarm. The results contain a column for the alarm ID, a comment, timestamps for when the alarm was raised, acknowledged, and cleared, the length of time the alarm went unacknowledged, and the total duration of the alarm.

Query Type: Alarm Summary

You can query alarms and events to view various alarm summary information.

Note: These queries only work when the Historian is configured to use high-speed alarms and events storage.

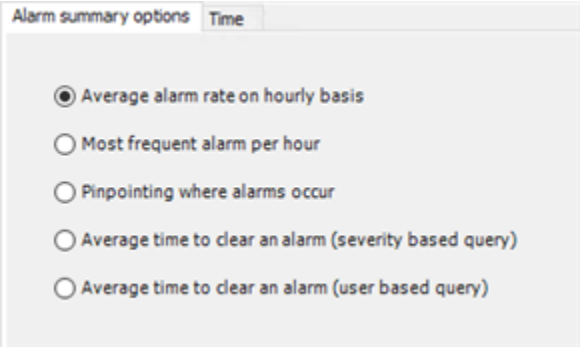
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To view alarm summary information

1. In the **Query Type** list in the toolbar, select **Alarm Summary**.
2. In the **Columns** pane, click on each tab and configure the parameters for the query:
   - See Alarm summary options Tab.
   - See Time Tab.
3. To view the results, click the **Data** tab in the **Results** pane.

Alarm summary options Tab

Use the **Alarm summary options** tab to specify how the results of the query are presented.



Options are as follows:

- **Average alarm rate on hourly basis**: This query calculates the average number of alarm events raised per hour for the selected time range.

	Average Alarm Rate on Hourly Basis
▶	11

- **Most frequent alarm per hour**: This query calculates the frequency of alarm events raised per hour for the selected time range, listed in order from highest to lowest frequency.

	ForDate	OnHour	Count per Hour	Alarm Attribute
▶	3/11/2022 12:00:00 AM	7	5	EMI_MinOfDayEMI_MinOfDa
	3/11/2022 12:00:00 AM	7	1	Reactor_001Reactor_001.Re
	3/11/2022 12:00:00 AM	7	1	Reactor_001Reactor_001.Re
	3/11/2022 12:00:00 AM	7	1	Reactor_001Reactor_001.Re

- **Pinpointing where alarms occur**: This query lists all alarm events raised for the selected time range, grouped by area, object, and alarm type.

	Source_HierarchicalArea	Source_Object	Source_ProcessVariable	Alarm_Type	AlarmCount
▶	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.AlarmHeartB	Hi	5
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	Lo	1
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Hi	1
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	HiHi	1
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Lo	1
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactTemp	Hi	1

- **Average time to clear an alarm (severity based query)**: This query shows you the average amount of time each alarm event remained active before being cleared, grouped by alarm severity.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Source_HierarchicalArea	Source_Object	Source_ProcessVariable	Alarm_Type	Severity	SecsInAlarm	AlarmCount	AvgSecsInAlarm	SecsInAlarmUnAck	AvgSecsInAlarmUnack
▶	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.AlarmHeartB	Hi	2	200.004000	10	40.000800	196.163000	39.232600
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	Hi	4	446.001000	1	446.001000	(null)	(null)
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	Lo	4	(null)	1	(null)	7.428000	7.428000
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	ROCHi	2	3979.999000	2	3979.999000	46.950000	46.950000
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Hi	2	120.000000	2	120.000000	49.415000	49.415000
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	HiHi	2	84.001000	2	84.001000	19.416000	19.416000
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Lo	2	25.999000	2	25.999000	37.426000	37.426000
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactTemp	Hi	2	23.999000	2	23.999000	4.951000	4.951000

- **Average time to clear an alarm (user based query)**: This query shows you the average amount of time each alarm event remained active before being cleared, grouped by Historian user.

	Source_HierarchicalArea	Source_Object	Source_ProcessVariable	Alarm_Type	User_Name	SecsInAlarm	AlarmCount	AvgSecsInAlarm	SecsInAlarmUnAck	AvgSecsInAlarmUnack
▶	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.AlarmHeartB	Hi	DefaultUser	200.002000	9	40.000400	157.221000	39.305250
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	Hi	DefaultUser	446.001000	1	446.001000	(null)	(null)
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	Lo	DefaultUser	(null)	1	(null)	7.428000	7.428000
	Enterprise.EMI_Area	EMI_MinOfDay	EMI_MinOfDay.ROCAlarm	ROCHi	DefaultUser	3979.999000	2	3979.999000	46.950000	46.950000
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Hi	DefaultUser	120.000000	1	120.000000	(null)	(null)
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	HiHi	DefaultUser	84.001000	1	84.001000	(null)	(null)
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Lo	(null)	25.999000	1	25.999000	(null)	(null)
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactLevel	Lo	DefaultUser	(null)	1	(null)	37.426000	37.426000
	Enterprise.Site.Process_Area	Reactor_001	Reactor_001.ReactTemp	Hi	DefaultUser	23.999000	1	23.999000	(null)	(null)

Query Type: Analog Statistics

You can retrieve statistics for analog or analog summary tags. These statistics include the minimum, maximum, time weighted average, standard deviation, and integral calculations. For more information on summary tags, see Configuring a Trend to Use a Summary Tag.

To view analog statistics

1. In the **Query Type** list in the toolbar, click **Analog statistics**.
2. Use the Tag Picker to select one or more analog summary or analog tags.
3. In the **Columns** pane, click each tab and configure the parameters for the query.
   - See Columns Tab.
   - See Time Tab.
   - See Retrieval Tab.
4. Click the **Data** tab in the **Results** pane to view results.

	TagName	Description	OPCQu	Minimum	Average	StdDev	Integral
▶	AvgRegular	Changes regularly--time wieghted avg won't matter	192	(null)	30	0	90
	AvgRegular	Changes regularly--time wieghted avg won't matter	192	(null)	30	0	90
	AvgRegular	Changes regularly--time wieghted avg won't matter	192	(null)	30	0	90

Columns Tab

Use the **Columns** tab to select the columns that you want to include in the query results.

The column options are:

- **Tag Name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information see, ArchestrA Naming Conventions.
- **Description:** The description of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **SourceTag:** The source tag of the tag.
- **SourceServer:** The source server of the tag.
- **StartDateTime:** The start time of the retrieval cycle.
- **EndDateTime:** The end time of the retrieval cycle.
- **OPCQuality:** The quality value received from the data source.
- **PercentGood:** The percentage of rows with good quality in relation to the total number of rows in the retrieval cycle.
- **First:** First value within the retrieval cycle or the most recent value prior to the cycle.
- **FirstDateTime:** The time stamp associated with the first value of the retrieval cycle.
- **Last:** The last value within the retrieval cycle or the most recent value prior to the cycle.
- **LastDateTime:** The time stamp associated with the Last value, which can be earlier than the StartDateTime if this is the initial value for the retrieval cycle.
- **Minimum:** The minimum value that occurred within the retrieval cycle.
- **MinDateTime:** The time stamp associated with the minimum value.
- **Maximum:** The maximum value that occurred within the retrieval cycle.
- **MaxDateTime:** The time stamp associated with the maximum value.
- **Average:** The time weighted average value of the retrieval cycle.
- **Standard Deviation:** The time weighted standard deviation value of the retrieval cycle.
- **Integral:** The area under the value curve of the retrieval cycle.
- **ValueCount:** The number of values contributing to the summary.
- **wwCycleCount:** The number of retrieval cycles (sub-intervals) for the specified time period. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).
- **wwResolution:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode. For more information, see "Resolution (Values Spaced Every X ms) (wwResolution).
- **wwTimeZone:** The time zone for retrieval is specified.
- **wwRetrievalMode:** The processing of retrieved data is specified before it is returned to the client. For more information, see Understanding Retrieval Modes.
- **wwVersion:** The version of data to be used if the original data value is changed.For more information, see History Version (wwVersion).

Query Type: Annotations

You can view annotations that were made by database users regarding data values of tags.

To view annotations

1. In the **Query Type** list in the toolbar, click **Annotations**.
2. If you want to only retrieve annotations for particular tag(s), use the Tag Picker to select one or more tags. For example, if you want to search for annotations for all analog tags, select the **All Analog Tags** public group and then select all analog tags in the **Tags** pane.
3. In the **Columns** pane, click on each tab and configure the parameters for the query:
   - See Criteria Tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- See Time Tab.

4. To view the results, click the **Data** tab in the **Results** pane.

	TagName	DateTime	Content	UserName	DateCreated
▶	ReactLevel	10/6/2004 12:04:39.130 PM	Needs investigation.	wwUser	10/6/2004 12:07:10.14
	ReactTemp	10/6/2004 12:04:44.617 PM	Check out the cause of this	public	10/6/2004 12:07:43.57
	ProdLevel	10/6/2004 12:04:46.953 PM	There may be a problem	dbo	10/6/2004 12:07:57.71

Criteria Tab

Use the Criteria tab to specify the type of annotations to be retrieved and which columns to show in the results. The Tagname column always appears.

To configure the annotation criteria

1. Select the columns to show in the results:
   - **Date and time:** The timestamp of the tag value for which the user has made an annotation.
   - **Date created:** The date that the annotation was created.
   - **Content:** The annotation text.
   - **User name:** The name of the database user.
2. Select the type of annotations to show:
   - **Public:** Show only public annotations. You can see your private annotations and the public annotations of other AVEVA Historian users.
   - **All users:** Show both public and private annotations. You can see your private annotations, as well as both the public annotations and private annotations of others.
   - **All tags on server:** Show all annotations for all tags.

Query Type: Custom

You can write custom SQL queries to execute against the database.

To create a custom query

1. In the **Query Type** list in the toolbar, click **Custom**.
2. In the **Results** pane, type the SQL query in the **SQL** tab.
3. To view the results, click the **Data** tab in the **Results** pane.

You can use the Custom query type to retrieve data from any database. For example, the following query retrieves from the Northwind database the list of employees who live in London. (The Northwind database is a sample database that is available from Microsoft.)

```
USE Northwind
SELECT * FROM Employees
   WHERE City = 'London';
```

Query Type: Favorites

You can load a saved SQL query file (.sql) and execute it against the database.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To execute a saved query

1. In the **Query Type** list in the toolbar, click **Favorites**.
2. In the **Columns** pane, click the **Favorites** tab.
3. In the **Favorite queries folder** box, type the path to the query file. To browse to the folder, click the ellipsis button.

   All .sql files in the folder appear in the **Favorite queries** window.
4. Select the query to execute in the window.
5. To view the query, click the **SQL** tab in the **Results** pane.
6. To view the results, click the **Data** tab in the **Results** pane.

Query Type: History Values

You can retrieve history data for specified tags. You can retrieve data for multiple types of tags in the same query. However, if you want to use a string value criterion, you can only retrieve string tags in the query. For more information, see Criteria Tab.

To view history data

1. In the **Query Type** list in the toolbar, click **History values**.
2. Use the Tag Picker to select one or more tags.
3. In the **Columns** pane, click on each tab and configure the parameters for the query.
   - See Columns Tab.
   - See Time Tab.
   - See Format Tab.
   - See Criteria Tab.
   - See Retrieval Tab.
   - See Source Tab.
   - See Order Tab.
4. Click the **Data** tab in the **Results** pane to view results.

Columns Tab

Use the **Columns** tab to configure the columns to show in the results. The **Value** (numeric value) and **vValue** (string value) columns are always shown.

Note: Some of the columns are not available when you select **Wide query format** on the **Format** tab.

The column options are:

- **Tag name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
- **Description:** The description of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Decimal places:** The number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.
- **Date and time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
- **Include milliseconds:** The time period used to include milliseconds in the timestamp.
- **Raw value range:** The minimum value of the raw acquired value. Also, the maximum value of the raw acquired value.
- **Engineering units:** The unit of measure.For example mph, grams, and pounds.
- **Engineering units range:** The minimum value of the tag, measured in engineering units. Also, the maximum value of the tag, measured in engineering units.
- **Quality:** The basic data quality indicator associated with the data value.
- **Quality description:** A text string that describes the quality detail value.
- **OPC Quality:** The quality value received from the data source.
- **State time:** The time that the tag remains in the specified value state (when using ValueState retrieval).
- **wwStateCalc:** The state calculation type used to calculate the state time when using ValueState retrieval (for example, average time or total time). For more information, see State Calculation (wwStateCalc).
- **SourceTag**: The source tag of the tag.
- **SourceServer**: The source server of the tag.
- **wwRetrievalMode:** The processing of retrieved data is specified before it is returned to the client.For more information, see Understanding Retrieval Modes.
- **wwCycleCount:** The number of retrieval cycles (sub-interavls) for the specified time period. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).
- **wwTimeDeadband:** The time deadband used in data retrieval. For more information, see Time Deadband (wwTimeDeadband).
- **wwTimeStampRule:** The time stamp rule used in data retrieval. For more information, see Timestamp Rule (wwTimestampRule).
- **wwVersion:** The version of data to be used if the original data value is changed. For more information, see History Version (wwVersion).
- **wwEdgeDetection:** The type of edge detection used in the query.
- **wwTagKey:** The unique identifier of the tag in the AVEVA Historian server.
- **InterpolationType:** The interpolation type used to calculate the value. For more information, see Interpolation Type (wwInterpolationType).
- **wwResolution:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode. For more information, see Resolution (Values Spaced Every X ms) (wwResolution).
- **wwValueDeadband:** The value deadband used in data retrieval. For more information, see Value Deadband (wwValueDeadband).
- **wwQualityRule:** The quality rule used in data retrieval. For more information, see Quality Rule (wwQualityRule).
- **wwTimeZone:** The time zone for retrieval is specified.
- **PercentGood:** The percentage of rows with good quality in relation to the total number of rows in the retrieval cycle.

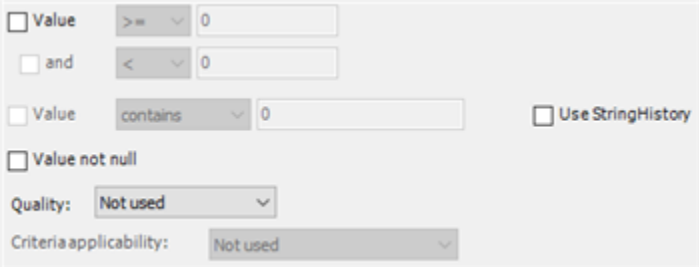
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **wwFilter:** The analog filter used to filter data during retrieval. For more information, see Analog Value Filtering (wwFilter).

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.

To configure criteria to retrieve data values (narrow format)



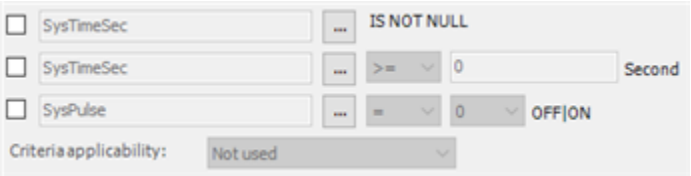
1. To configure criteria for a discrete tag, select the first **Value** check box and set the criteria to be either a 1 or a 0, then go to Step 4.
2. To configure criteria for an analog tag
   - Select the first **Value** check box and set the criteria for the data value. For example, the value must be greater than (>) 1500.
   - (Optional) Select the second **Value** check box and set another criteria for the data value. For example, the value must be less than (<) 2000.
   - Go to Step 4.
3. To configure criteria for a string tag:

   Note: If you use a string criterion, you can only retrieve data for string tags in the query. No data is returned for tags of other types that you may have selected.

   - Select the **Use StringHistory** check box.
   - Select the third **Value** check box and specify text that the returned string value must match. You can specify whether the returned value must equal, start, end, or contain the specified text. For example, you can specify that the value must contain the text “alert.”
   - Go to Step 4.
4. (Optional) Select the **Value not null** check box to filter out null values from the results.
5. (Optional) In the **Quality** list, click the quality criteria for the data. Only data values that match the quality you specify (Good, Bad, or Doubtful) are returned.
6. (Optional) In the **Criteria applicability** list, select the moment at which the edge detection criteria is met.
   - **first true:** Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false). This is also known as “leading” edge detection.
   - **no longer true:** Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true). This is also known as “trailing” edge detection.
   - **true:** Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
   - **first true or are no longer true:** All rows satisfying both the leading and trailing conditions are returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To configure criteria to retrieve data values (wide format)



1. To filter out null values from the results, select the first checkbox, then click ... to select a tag.
2. To configure criteria for an analog tag
   - Select the second checkbox, then click ... to select a tag. Set the criteria for the data value. For example, the value must be greater than (>) 50.
   - (Optional) Select the second check box, then click ... to select a tag. Set another criteria for the data value. For example, the value must be less than (<) 20.
3. (Optional) In the **Criteria applicability** list, select the moment at which the edge detection criteria is met.
   - **first true:** Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false). This is also known as “leading” edge detection.
   - **no longer true:** Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true). This is also known as “trailing” edge detection.
   - **true:** Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
   - **first true or are no longer true:** All rows satisfying both the leading and trailing conditions are returned.

Retrieval Tab

Use the Retrieval tab to configure data retrieval options.

To configure data retrieval options

1. In the **Retrieval mode** list, select the retrieval mode that allows you to access the data stored in an AVEVA Historian in different ways.

   For more information on the retrieval options, see Understanding Retrieval Modes and Understanding Retrieval Options.
2. In the **Query row limit** list, select the maximum number of rows for the data retrieval to avoid excessively large result sets. For example, if you set a row limit of 200, the historian only returns the first 200 rows of a query’s results. The row limit applies to each query.

Other Tab

Use the **Other** tab to configure the other retrieval options.

1. In the **History version** area, click **Latest** or **Original** to overwrite the values of a stored tag. For more information on History version, see History Version (wwVersion).
2. In the **Rules** area, do the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- In the **Timestamp** list, click the required timestamp value. For more information on the Time stamp rule, see Timestamp Rule (wwTimestampRule).
- In the **Values to include in calculations** list, click the data values that you want to include in the result. You can include the following quality rules:

  Good and uncertain quality: To include data values with uncertain quality in calculations.

  Good quality: To exclude data values with uncertain quality from calculations.

  Estimate when values are missing: To use the optimistic quality when the data values are missing.

  Note: The **Estimate when values are missing** quality rule is applicable only for Integral and Counter retrieval modes.

  Server default: To use the default quality rule specified at the AVEVA Historian level.

  For more information on each option, see Quality Rule (wwQualityRule).

3. In the **State retrieval** area, do the following:
   - In the **State calculation** list, click the the state calculation.
   - Select the **Specify state** check box to set the value of the state. For example, you can specify either a open or close statefor the SteamValve tag.

     Note: The state calculation settings are applicable only to ValueState and RoundTrip retrieval modes.

     For more information on State calculation, see State Calculation (wwStateCalc).
4. In the **Transformation** list, click the transformations to be applied to the result. You can include the following transformations:
   - **No Transformation**: If the query does not specify the filter element or if the value is not overridden for the filter element.
   - **Remove outliers**: To remove outliers from a set of analog points.
   - **Convert analog values to discrete**: To convert value streams for any analog tag to discrete value streams.
   - **Snap to base value**: To force values in a well-defined range around one or more base values to "snap to" that base value.

     For more information on Transformation, see Analog Value Filtering (wwFilter).
5. In the **Phantom cycle** area, select the **Do not include boundary values** check box to remove boundary values from the result.

   For more information on Phantom cycle filter, see About "Phantom" Cycles.

Query Type: IO Server

You can retrieve basic configuration information for all I/O Servers configured for use with the AVEVA Historian.

To retrieve I/O Server information

1. In the **Query Type** list in the toolbar, click **IO server**.
2. In the **Columns** pane, click the **IO Server** tab.
3. Select the columns to show in the results:
   - **Description:** The description of the I/O Server.
   - **Application name:** The application name of the I/O Server. This name is usually the same as the executable file name.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Topic name:** The name of the topic.
- **Topic timeout:** The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic will be considered "dead." The AVEVA Historian will disconnect and then attempt to reconnect to the topic.

4. In the **Computer** list, click the name of the computer on which the I/O Servers run.

Query Type: Live Values

You can retrieve real-time data values for specified tags.

To view live data

1. In the **Query Type** list in the toolbar, click **Live values**.
2. Use the Tag Picker to select one or more tags.
3. In the **Columns** pane, select each tab and configure the parameters for the query.
   - See Columns Tab.
   - See Time Tab.
4. To view the results, click the **Data** tab in the **Results** pane.

Columns Tab

Use the **Columns** tab to configure the columns to show in the results.

The column options are:

- **Tag name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
- **Description:** The description of the tag.
- **Decimal places:** The number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.
- **Date and time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
- **Include milliseconds:** Used to include milliseconds in the timestamp.
- **Raw value range:** The minimum value of the raw acquired value. Also, the maximum value of the raw acquired value.
- **Engineering units:** The unit of measure.For example mph, grams, and pounds.
- **Engineering units range:** The minimum value of the tag, measured in engineering units. Also, the maximum value of the tag, measured in engineering units.
- **Quality:** The basic data quality indicator associated with the data value.
- **Quality description:** The text string that describes the quality detail value.
- **OPC Quality:** The quality value received from the data source.
- **SourceTag:** The source tag of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **SourceServer:** The source server of the tag.

Query Type: Number of Tags

You can retrieve the total number of tags of a certain type for the currently selected AVEVA Historian.

To retrieve the number of tags

1. In the **Query Type** list in the toolbar, click **Number of tags**.
2. In the **Columns** pane, click the **Count** tab.
3. In the **Tag type** list, click the type of tag for which you want to return the total number.
4. To view the results, click the **Data** tab in the **Results** pane.

Query Type: Server Version

You can retrieve the version number for the currently selected AVEVA Historian.

To retrieve the server version

1. In the **Query Type** list in the toolbar, click **Server version**.
2. To view the results, click the **Data** tab in the **Results** pane.

Query Type: Slice By

You can view the values of one or more tags based on the timestamps of the value changes for a selected tag, referred to as the "slice by" tag.

To view slice by data

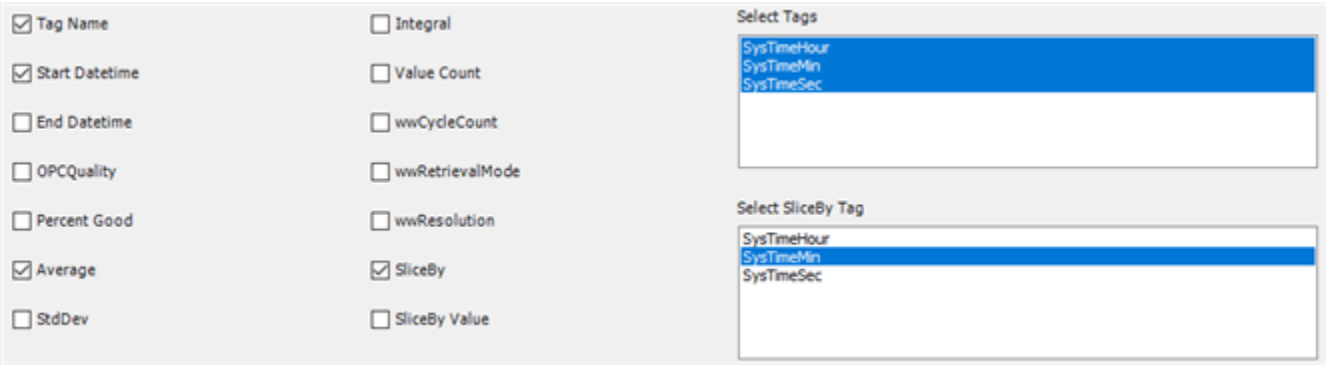
1. In the **Query Type** list in the toolbar, click **Slice By**.
2. Use the Tag Picker to select one or more tags.
3. In the **Columns** pane, click on each tab and configure the parameters for the query.
   - See Columns Tab.
   - See Time Tab.
   - See Criteria Tab.
4. Click the **Data** tab in the **Results** pane to view results.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagName	StartDateTime	Average	SliceBy
SysTimeHour	11/5/2021 1:59:00 PM	13	SysTimeMin
SysTimeMin	11/5/2021 1:59:00 PM	59	SysTimeMin
SysTimeSec	11/5/2021 1:59:00 PM	29.5	SysTimeMin
SysTimeHour	11/5/2021 2:00:00 PM	14	SysTimeMin
SysTimeMin	11/5/2021 2:00:00 PM	0	SysTimeMin
SysTimeSec	11/5/2021 2:00:00 PM	29.5	SysTimeMin
SysTimeHour	11/5/2021 2:01:00 PM	14	SysTimeMin
SysTimeMin	11/5/2021 2:01:00 PM	1	SysTimeMin
SysTimeSec	11/5/2021 2:01:00 PM	29.5	SysTimeMin
SysTimeHour	11/5/2021 2:02:00 PM	14	SysTimeMin
SysTimeMin	11/5/2021 2:02:00 PM	2	SysTimeMin
SysTimeSec	11/5/2021 2:02:00 PM	29.5	SysTimeMin

Columns Tab

Use the **Columns** tab to configure the columns that are returned for the results.



Options are as follows:

- **Tag Name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
- **Start Datetime:** The start time of the retrieval cycle.
- **End Datetime:** The end time of the retrieval cycle.
- **OPCQuality:** The quality value received from the data source.
- **Percent Good:** The percentage of rows with good quality in relation to the total number of rows in the retrieval cycle.
- **Average:** The time weighted average value of the retrieval cycle.
- **StdDev:** The time weighted standard deviation value of the retrieval cycle.
- **Integral:** The area under the value curve of the retrieval cycle.
- **Value Count:** The number of values contributing to the summary.
- **wwCycleCount:** The number of retrieval cycles (sub-intervals) for the specified time period. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).
- **wwRetrievalMode:** The processing of retrieved data is specified before it is returned to the client. For more information, see Understanding Retrieval Modes.
- **wwResolution:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode. For more

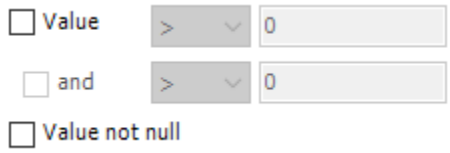
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

information, see "Resolution (Values Spaced Every X ms) (wwResolution).

- **SliceBy:** The name of the slice by tag.
- **SliceBy Value:** The value of the slice by tag.
- **Select Tags:** Click to select one or more tags for which data will be retrieved.
- **Select SliceBy Tag:** Click to select the slice by tag.

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.



To configure value criteria

1. To configure criteria for a discrete tag, select the **Value** check box and set the criteria to be either a 1 or 0.
2. To configure criteria for an analog tag:
   a. Select the **Value** check box and set the criteria for the data value. For example, the value must be greater than (>) 1500.
   b. (Optional) Select the **and** check box and set another criteria for the data value. For example, the value must be less than (<) 2000.
   c. (Optional) Select the **Value not null** check box to filter out NULL values from the results.

Query Type: State Statistics

You can retrieve state statistics for state summary, analog, discrete, or string tags. These statistics include the total time, shortest time, longest time, average time, OPC Quality, percent of the cycle, and value. For more information on state summary tags, see Configuring a Trend to Use a Summary Tag.

To view state statistics

1. In the **Query Type** list in the toolbar, click **State statistics.**
2. Use the Tag Picker to select one or more state summary, analog, discrete, or string tag.
3. In the **Columns** pane, click each tab and configure the parameters for the query.
   - See Columns Tab.
   - See Time Tab.
   - See Criteria Tab.
   - See Retrieval Tab.
4. To view the results, click the **Data** tab in the **Results** pane.

Columns Tab

Use the **Columns** tab to select which columns to include in the query results.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The column options are:

- **Tag Name**: The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
- **Description:** The description of the tag.
- **SourceTag**: The source tag of the tag.
- **SourceServer**: The source server of the tag.
- **StartDateTime**: The start time of the retrieval cycle.
- **EndDateTime**: The end time of the retrieval cycle.
- **OPC Quality**: The quality value received from the data source.
- **State Count**: The number of times the state occurred within the retrieval cycle.
- **Contained State Count**: The number of times the state occurred fully contained within the retrieval cycle.
- **Minimum**: The minimum time in the current state amongst all its occurrences during the current retrieval cycle, including state occurrences that fall partially within the period.
- **MinContained**: The minimum time in the current state amongst all its occurrences during the current retrieval cycle, excluding state occurrences that fall partially within the period.
- **Maximum**: The maximum time in the current state amongst all its occurrences during the current retrieval cycle, including state occurrences that fall partially within the period.
- **MaxContained**: The maximum time in the current state amongst all its occurrences during the current retrieval cycle, excluding state occurrences that fall partially within the period.
- **Average**: The average time in the current state amongst all its occurrences during the current retrieval cycle, including state occurrences that fall partially within the period.
- **AverageContained**: The average time in the current state amongst all its occurrences during the current retrieval cycle, excluding state occurrences that fall partially within the period.
- **Total**: The total time in the current state during the current retrieval cycle, including state occurrences that fall partially within the period.
- **TotalContained**: The total time in the current state during the current retrieval cycle, excluding state occurrences that fall partially within the period.
- **Percent**: The percentage of time during the current retrieval cycle that the tag was in the current state, including state occurrences that fall partially within the period.
- **PercentContained**: The percentage ratio between StateTimeTotalContained and StateTimeTotal.
- **wwMaxState:**The maximum number of states allowed in the same reporting period. The default number of maximum states is 10.
- **wwCycleCount**: The number of retrieval cycles (sub-interavls) for the specified time period.
- **wwResolution**: The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
- **wwTimeZone**: The time zone for retrieval is specified.
- **wwRetrievalMode**: The processing of retrieved data is specified before it is returned to the client.
- **wwVersion**: The version of data to be used if the original data value is changed

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.

To configure value criteria

1. To configure criteria for a discrete tag, select the first **Value** check box and set the criteria to be either a 1 or 0.
2. To configure criteria for an analog tag:
   a. Select the first **Value** check box and set the criteria for the data value. For example, the value must be greater than (>) 1500.
   b. (Optional) Select the second **Value** check box and set another criteria for the data value. For example, the value must be less than (<) 2000.
   c. (Optional) Select the **Value not null** check box to filter out NULL values from the results.

Query Type: Storage

You can retrieve configuration information regarding the directories in which a selected AVEVA Historian is storing history files. The different storage types are circular, alternate, buffer, and permanent.

To retrieve storage information

1. In the **Query Type** list in the toolbar, click **Storage**.
2. In the **Columns** pane, click the **Storage** tab.
3. Select the columns to show in the results:
   - **Path:** The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). The alternate, buffer, and permanent storage locations can be anywhere on the network, provided that the AVEVA Historian service user has full access to those network locations.
   - **Maximum storage size:** The limit, in megabytes, for the amount of data that will be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location will be used.
   - **Minimum storage size threshold:** The minimum amount of disk space, in megabytes, at which the system will attempt to start freeing up space. The threshold applies to circular and alternate storage only. Typically, the minimum threshold should be the size of the average history block (before any compression) multiplied by 1.5.
4. In the **and computer** list, click the name of the computer on which the storage node resides.
5. To view the results, click the **Data** tab in the **Results** pane.

   Note that the **Storage Type** column always appears.

Query Type: Storage Size Available

You can retrieve the amount of space, in MB, that remains for each of the storage locations. The amount of space remaining is monitored by system tags on the server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To retrieve the storage size

1. In the **Query Type** list in the toolbar, click **Storage Size**.
2. To view the results, click the **Data** tab in the **Results** pane.

Query Type: Storage Start Date

You can retrieve the start date for the oldest history block in the system.

To retrieve the storage start date

1. In the **Query Type** list in the toolbar, click **Storage Start Date**.
2. To view the results, click the **Data** tab in the **Results** pane.

Query Type: Tag Details

You can view the configuration details for specified tags.

To view tag details

1. In the **Query Type** list in the toolbar, select **Tag details**.
2. Use the Tag Picker to select one or more tags.
3. In the **Columns** pane, select the **Columns** tab.
4. Select which columns to include in the query results. The options that are available in this tab depend upon the type of tag you have selected. For example, a detector type only applies to event tags.

   The **Tagname** column is always shown.

   - **Description:** The description of the tag.
   - **Date created:** The date that the tag was created.
   - **Address:** The tag address, which is made up of the application name of the I/O Server, the name of the topic, and the address string of the tag.
   - **Storage rate:** The rate at which the tag is stored if the storage type is cyclic.
   - **Acquisition rate:** For polled tags of acquisition type 1, the poll rate in milliseconds.
   - **Storage type:** The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored."
   - **Acquisition type:** The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, the name of the I/O Server, topic, and item must be specified. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via MDAS or a manual update; 3 = System driver.
   - **Messages:** The message associated with the FALSE state of the discrete tag. A discrete tag set to 0 is in the FALSE state. Also, the message associated with the TRUE state of the discrete tag. A discrete tag set to 1 is in the TRUE state.
   - **Maximum characters:** The maximum number of characters for the string.
   - **Raw value range:** The minimum value of the raw acquired value. Also, the maximum value of the raw acquired value.
   - **Engineering units:** The unit of measure.For example mph, grams, and pounds.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Engineering units range:** The minimum value of the tag, measured in engineering units. Also, the maximum value of the tag, measured in engineering units.
- **SourceTag:** The source tag of the tag.
- **SourceServer:** The source server of the tag.
- **Detector type:** The name given to the type of detector.
- **Action type:** The name given to the type of action.
- **Detector string:** The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian.
- **Action string:** The script that specifies the event action. Action scripts are executed on the local AVEVA Historian.
- **Scan rate:** The interval, in milliseconds, at which the system will check to see if the event conditions specified by the detector have occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).
- **Status:** The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.
- **Logged:** Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured.

5. To view the results, select the **Data** tab in the **Results** pane.

Query Type: Tag Search

You can search for tags by name or criteria for the names.

To search for tags

1. In the **Query Type** list in the toolbar, click **Tag search**.
2. In the Tag Picker, select the type of tag that you want to search for. For example, if you want to search for an analog tag, select the **All Analog Tags** public group.
3. In the **Columns** pane, select the tab and configure the parameters for the query:
   - See Search Tab.
4. To view the results, select the **Data** tab in the **Results** pane.

Search Tab

Use the **Search** tab to search for a tag in the database.

1. In the **Tag type** list, click the type of tag to search for, either Analog, Discrete, Event, String, or Summary.
2. For summary tags, further restrict the search by specifying a particular calculation type or frequency.
   - **Calculation type:** The type of calculation. Sum, Maximum, Minimum, or Average.
   - **Calculation frequency:** The time duration, in seconds, for which the calculation is performed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Select the **Show** check boxes to show the calculation type and/or frequency in the result set.

Query Type: Time Running

You can retrieve the amount of time, in minutes, that the AVEVA Historian has been running since the last startup.

To retrieve the time

1. In the **Query Type** list in the toolbar, click **Time running**.
2. To view the results, click the **Data** tab in the **Results** pane.

Common Tabs for Query Types

This section describes the configuration tabs that are common to multiple query types.

Time Tab

Use the **Time** tab to specify the time options for the query.

The grid shows the time zone and daylight savings time settings for the following entities:

Entity	Description
Application	The AVEVA Historian Client Query application. The timestamps of the returned data reflect this time zone. To change this time zone, see the procedure below.
Client	The client computer on which the Query application is installed.
<ServerName>	The AVEVA Historian to which the Query application is currently connected. You can be connected to more than one server.

To configure the time period and time zone

1. In the Time area, use the time picker to select the start and end times for the query. For more information, see Time Picker.
2. To return the data with a timestamp that reflects the time zone setting of the AVEVA Historian, select the **Use time zone of the server** check box.
3. To return the data with a timestamp that reflects a time zone setting, different than that of the local client computer, click the name of the appropriate time zone to use in the **Time Zone** list.

For example, consider a SCADA application that monitors a pipeline between Houston, Texas and Lake Forest, and California. The Query application is installed on a computer in Houston and Texas. You want to send a query file to an engineer located at the start of the pipeline in Lake Forest to aid in troubleshooting a problem. You can set the time zone of the Query application to reflect the time of Lake Forest, California (Pacific Standard Time), so

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

that the query that you send to the engineer displays data in a time zone that is relevant to him/her.

Format Tab

Use the **Format** tab to specify how the results of the query are presented.

Options are as follows:

- **Narrow query format:** In this format, there is one row for a single tag's value for a particular timestamp.
- **Wide query format:** In this format, there is one row for one or more tag values for a single timestamp, thus providing a "wide" view of the data. To use the wide query format, you must specify the timestamp and one or more tagnames as the column names in the query syntax. The results will contain a column for the timestamp and columns for the value of each specified tag at that timestamp.

Retrieval Tab

Use the **Retrieval** tab to specify the "granularity" of the data to be returned.

To configure retrieval mode options

1. In the **Retrieval mode** list, select the retrieval mode that allows you to access the data stored in a AVEVA Historian in different ways.
2. In the **Main options** area, do the following:
   - **Cyclic**: Cyclic based retrieval is the retrieval of stored data for the given time period based on a specified cyclic retrieval resolution, regardless of whether or not the value of the tag(s) has changed.
   - **Full:** Full retrieval mode returns one row per stored value for tags at a given time interval, including duplicate values with different timestamps. You can use full retrieval for all tag types.
3. In the **Query row limit** list, select the maximum number of rows for the data retrieval to avoid excessively large result sets. For example, if you set a row limit of 200, the historian only returns the first 200 rows of a query’s results. The row limit applies to each query.
4. If you select cyclic retrieval mode, configure additional options in the **Cyclic attributes** area.
   - **XX values over equal time intervals:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query.
   - **Values spaced every XX ms:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
   - **Interpolation type:** The interpolation type for data retrieval.
5. If you select delta retrieval mode, configure additional options in the **Delta retrieval deadbands** area.
   - **Time:** The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
   - **Value:** The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage will not be returned. Applies only to delta retrieval. The default is 0.

For more information on configuring the other data retrieval options, see Other Tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Source Tab

Use the **Source** tab to specify the data version and type of table for the query.

To configure the source

- In the **Source** area, specify the AVEVA Historian tables from which data will be retrieved.
  - **Manual history tables:** Normal SQL Server tables that are used to store data. These are the ManualAnalogHistory and ManualDiscreteHistory tables.
  - **Extension tables:** Logical tables that are populated from the AVEVA Historian data files. These tables support the AVEVA Historian time domain extensions for handling data.
  - **Both:** Select this option to retrieve data from both the manual and extension tables.

Order Tab

Use the **Order** tab to specify how the results are ordered.

To configure the ordering

1. In the left window, select a column to add to the ordering criteria and then click the arrow button to move the column to the right column. Repeat to add all of the columns you want to the ordering criteria.
2. To move a column up or down in the ordering, select the column in the right window, and then click the up or down buttons. The results are first ordered according to the column that is listed first in the window, then ordered according to the column that is listed second, and so on.
3. In the **Order** area, select whether you want the results to be ordered in ascending or descending order.

Client Workbook

The AVEVA Historian Client Workbook is an add-in to Microsoft Excel that allows you to query one or more AVEVA Historian or SQL Server databases and return results to a spreadsheet. Using the AVEVA Historian Client Workbook, you can easily create reports using AVEVA Historian data without needing in-depth knowledge of SQL scripting. The reports that you create with the AVEVA Historian Client Workbook can be saved, allowing you to run a report again at any time.

Note: Historian data can also be used in Excel via the task pane add-in. See the *Historian Client Web Help* for details.

Getting Started

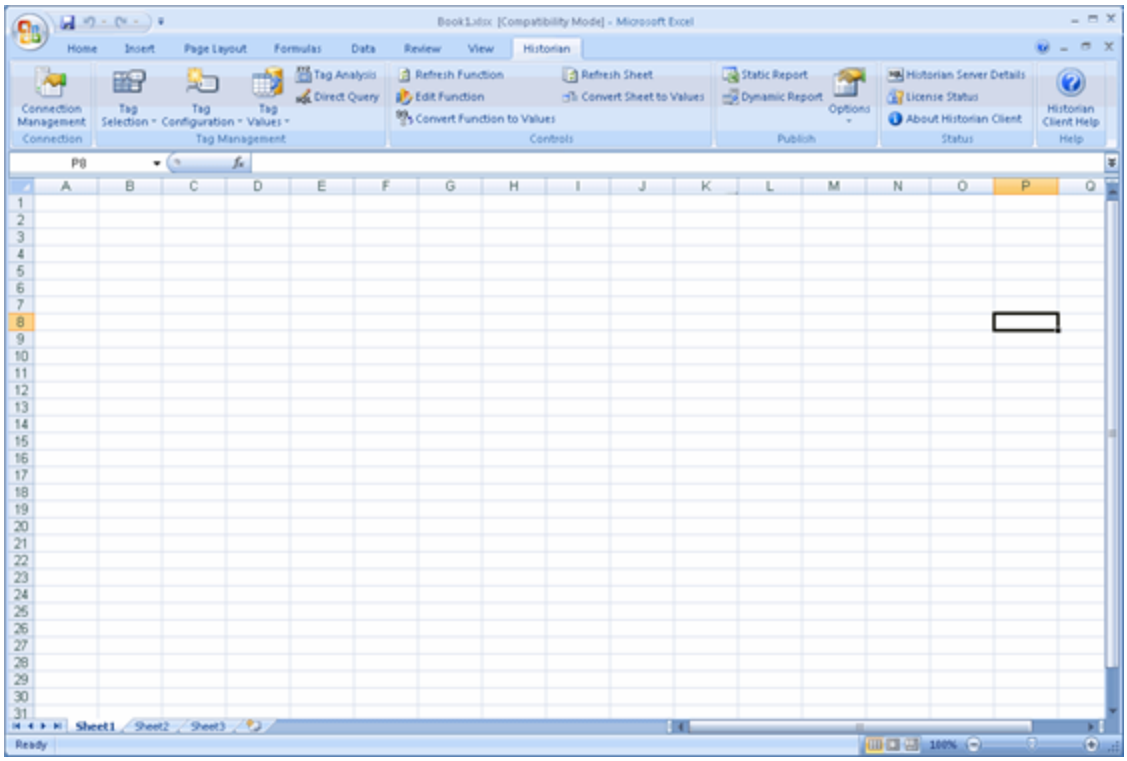
If the AVEVA Historian Client Workbook is installed, an additional menu called **Historian** is added in Microsoft Excel.

The **Historian** menu contains all of the commands you use to create a report using AVEVA Historian data.

- The **Historian** menu is a part of the Ribbon Bar.

  Note: If you don't see the Historian menu, you may need to manually load the Excel add-in.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Managing Server Connections

You must specify one or more AVEVA Historians and/or SQL Servers as data sources for the AVEVA Historian Client Report.

To manage server connections

1. On the **Historian** tab, in the **Connection** group, click **Connection Management**. The **Server List Configuration** dialog box appears.
2. Configure the server(s) and then click **Close**.

   For more information, see Server Connection Configuration.

Opening an Existing Workbook File

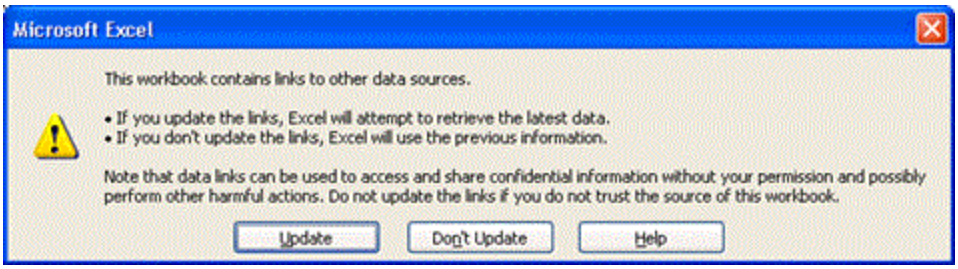
Within a Workbook file, referenced links may be different than the instance of Excel you are currently using if:

- The file you are opening was saved using a previous version of the AVEVA Historian Client Workbook.
- The file you are opening was saved using a different computer.

The AVEVA Historian Client Workbook add-in automatically updates only the AVEVA Historian Client Workbook reference within the file to use the current add-in location.

If Microsoft Excel detects that links need updating, a message box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



From this dialog box, you can choose to update the links or keep them the same.

To open an existing Workbook file

1. In Excel, from the **File** menu, click **Open**.
2. Select the name of the file to open and click **Open**.

Manually Loading/Unloading the Excel Add-In

Generally, when you install AVEVA Historian Client on a computer with Microsoft Excel , the Excel add-in is automatically loaded into Excel. The **Historian** tab appears in the Ribbon bar.

However, if you need to manually load or unload the add-in, use the following procedure.

To manually load the Excel add-in

1. Click **File** and then click **Options**.
2. In the Excel Options dialog box, click **Add-Ins**.
3. In the Manage list, select **Excel Add-ins**, and then click **Go**.

   The **Add-Ins** dialog box will appear.
4. Click **Browse**.
5. Browse and select the HistClient.xla and HistClient.xlam.

   By default, they are installed here:

   `C:\Program Files (x86)\Common Files\ArchestrA`
6. Check the **ActiveFactory Workbook** and **Historian Client Workbook** check boxes.
7. Click **OK**.

To manually unload the Excel add-in

1. Click the **Microsoft Office** button, and then click **Excel Options.**
2. Click **Add-Ins**.
3. In the Manage list, select **Word Add-ins** (or **Excel Add-ins**), and then click **Go**.
4. Unmark the **Historian Client Workbook** check box.
5. Click **OK**.

Creating a Report: Overview

Follow these general steps to create reports using the AVEVA Historian Client Workbook.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

1. Configure a connection to one or more servers. For more information, see Managing Server Connections.
2. Understand how functions, formulas, and array formulas work. For more information, see Working with Functions, Formulas, and Cells.
3. Determine how you want to set up or use Workbook options. For more information, see Configuring Workbook Options.
4. Configure tags for which you want to return data. For more information, see Selecting Tags for Reports.
5. Create a data report for the selected tags.
   - For information on retrieving configuration data using wizards, see Retrieving Tag Configuration Information.
   - For information on retrieving current and historical data values using wizards, see Retrieving Tag Values.
   - For information on generating analysis graphs and data using wizards, see Retrieving Tag Values.
   - For information on retrieving data using a manually created SQL query, see Creating a Direct Query.
6. Configure other advanced optional features. For more information, see Configuring Workbook Options.
7. Save the report.
8. Optionally publish the report to the AVEVA Information Server. For more information, see Publishing Reports.

Working with Functions, Formulas, and Cells

An Excel function is a predefined formula that performs a calculation. For example, a function can add two numbers and return the results:

```
=SUM(number1,number2, ...)
```

An array formula is a type of function that can perform multiple calculations and then return either single or multiple results. Array formulas act on two or more sets of values called array arguments. The arguments are the inputs to the function and are required to be in a particular order. Most of the formulas created using the AVEVA Historian Client Workbook are array formulas. For example:

```
=wwAnalogTagDetails(DataSource, TagRange, Description, EngUnit, EURange, RawRange, Storage, OptionRange)
```

When the specific inputs are provided and the array formula is executed, the results appear in one or more cells. You can click anywhere in the array results to see the associated formula.

You can manually create or edit array formulas in the same way that you create or edit other formulas, except you press CTRL+SHIFT+ENTER to enter or update the array formula.

TagDetails_Sheet

Refreshing a Function or Array Formula

You can refresh any function or array formula in the worksheet.

To refresh a function or array formula:

1. Select the function to refresh. If you want to refresh an array formula, select any cell in the array.
2. On the **Historian** tab, in the **Controls** group, click **Refresh Function**.

   The function is executed and the results are returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Editing a Function

To edit a function:

1. Select the function to edit. If you want to edit an array formula, select any cell in the array.
2. On the **Historian** tab, in the **Controls** group, click **Edit Function.**
   If applicable, the appropriate wizard opens, allowing you to edit the query.

Converting a Function to Values

To convert a function to values:

1. Select the function to convert. If you want to convert an array formula, select any cell in the array.
2. On the **Historian** tab, in the **Controls** group, click **Convert Function to Values.**

Refreshing a Sheet

You can refresh all of the formulas for a selected worksheet.

To refresh a worksheet:

1. Select any cell in the sheet.
2. On the **Historian** tab, in the **Controls** group, click **Refresh Sheet**.
   The query is executed and the worksheet is updated with the returned results.

Converting a Sheet to Values

To convert all of the functions in a sheet to values:

1. Select the sheet to convert.
2. On the **Historian** tab, in the **Controls** group, click **Convert Sheet to Values.**

Manually Inserting a Function

You can manually insert functions instead of using the function wizards.

In Excel, functions are not automatically inserted as array formulas. By default, only single cell contains a value from the result set. You must type the formula as an array formula (by pressing CTRL+SHIFT+ENTER) so that all values in the result set appear.

To manually insert a function:

1. In the worksheet, type values to use for the function arguments. For example, you might type "EMINSQL10" for the data source name and "ReactLevel" and "ReactTemp" as the tags for which to retrieve live values.
2. On the **Formulas** tab, in the **Function Library** group, click **Insert Function**. The **Insert Function** dialog box appears.
3. In the **Or select a category** list, click **User Defined**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

4. In the **Select a function** list, select any of the AVEVA Historian Client Workbook functions.

   All of these functions are prefixed with "ww." For more information regarding these functions and their arguments, see AVEVA Historian Client Workbook Function Reference.
5. Click **OK**. The **Function Arguments** dialog box appears.
6. For each of the arguments, assign a cell value that contains the input.

   For example, assigning A1 to the DataSource argument causes "MyInSQL" to be used for the data source.
7. Click **OK**. The function is inserted into the spreadsheet.

   Note: If the function returns a date/time value, the date/time appear in the Julian format, unless a different format is configured for the cell.
8. Select the returned value.
9. On the **Historian** tab, in the **Controls** group, click **Refresh Function**. The formula is converted to an array and you can see all of the return values.

Manually Editing a Function

To manually edit a function:

1. In your worksheet, select the function to edit so that it appears in the formula bar.
2. In the formula bar, edit the argument value(s) for the function.
3. Press CTRL+SHIFT+ENTER on your keyboard to type the array formula.
4. On the **Historian** tab, in the **Controls** group, click **Refresh Function** to resize the results. You can then see all the return values.

Copying a Function

You can copy functions to different locations in the worksheet. This is useful when creating additional functions that are only slightly different than existing functions.

To copy a function:

1. In the worksheet, select the range of cells that contains the array formula. To select all of the array cells, insert the mouse cursor in the array and then press CTRL+/ on your keyboard, where / is the forward slash.
2. Press CTRL+C to copy the function.
3. Insert the mouse cursor in the new location for the function.
4. Press CTRL+V to paste the function.

For information on manually editing a function, see Manually Editing a Function.

Selecting Cells

Various option boxes require you to specify a worksheet cell for either input or output. You can easily select the cell or range of cells that you want to use, eliminating the need to type the formula for the cell location.

Note: The Workbook supports selection of contiguous cells only. For example, if you have tags in the A1, A2, and A3 cells then you should select all three cells. You cannot select A1 and A3 only by leaving A2.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To select a cell in the worksheet:

1. Select the button to the right of the option box that requires a cell location. The cell selector dialog box appears.
2. In the spreadsheet, select the cell(s). The cell notation appears in the dialog box.
3. Select the **Notation** button to insert the notation into the option box.

Verifying the Date/Time Format in Microsoft Excel

When you query the database for history values, you must specify the time range for the query. If you choose to use specific dates, you must make sure that the date/time format that you specify match the Microsoft Excel date/time format settings for the result cells.

To verify the date/time format:

1. In the spreadsheet, select the cells to contain the time stamps for the returned data.
2. On the **Home** tab, in the **Cells** group, click **Format** menu, and then click **Format Cells**. The **Format Cells** dialog box appears.
3. Click the **Number** tab.
4. In the **Category** window, click **Date.**
5. In the **Type** list, verify the date format.
6. Click **OK**.

Selecting Tags for Reports

When you configure a report, you can either type the tagname(s) directly in the worksheet or pick the tag and have it inserted for you.

You can include the following types of tags in your worksheet:

- Analog, discrete, string, summary, and event tags. For more information, see Selecting Analog, Discrete, String, Summary, or Event Tags.
- Summary tags. A summary tag is a tag for which an aggregation calculation (minimum, maximum, average, or sum) is configured on the server. For more information, see Selecting Summary Tags.
- Event snapshot tags. A snapshot tag is a tag for which a snapshot action has been configured on the server. A snapshot action logs into dedicated SQL Server tables the data values for selected analog, discrete, or string tags that have the same timestamp as the detected event. For more information, see Selecting Event Snapshot Tags.

Selecting Analog, Discrete, String, Summary, or Event Tags

To select analog, discrete, string, summary, or event tags:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.

   For instructions on how to use most of the options in this dialog box, see Tag Picker.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Select the **Include description** check box to include tag descriptions in the results.
3. In the **Select cell range to insert tags** list, click the name of the workbook cell into which you want to insert the tags. For more information, see Selecting Cells.
4. Click **OK**.

   The tag is inserted into the selected cell.

Viewing the Hierarchical Name in a Sheet

You can view the hierarchical name in a sheet. For more information on hierarchical names, see Integration with AVEVA Application Server.

To view the hierarchical name in a sheet:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
2. Right-click in the Tag Picker and click **Use hierarchical name**.

   The Workbook application shows the hierarchical names instead of the tag names. For example, the **Filter** pane, the **Tag Selection** dialog box, and the **Tag Configuration** dialog box show hierarchical names. Any query generated after enabling the hierarchical name option shows hierarchical names in the worksheet.

Selecting Summary Tags

To select summary tags:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Summary Tag Selection**. The **Summary Tag Selection** dialog box appears.
2. In the **Servers** list, click the name of the server to use.
3. In the **Filter** area, configure the criteria by which the summary tags are filtered and displayed in the **Select tags to insert into workbook** window. The summary tags have one or more of the following summary operations configured for them:
   - **Summarization frequency:** The time duration, in seconds, for which the calculation is performed.
   - **Calculation type:** The type of calculation. Sum, Maximum, Minimum, or Average.
   - **Tag name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
   - **Description:** The description of the tag.
4. Select the **Include description** check box to include tag descriptions in the results.
5. In the **Select cell range to insert tags** list, click the name of the workbook cell into which you want to insert the tags. For more information, see Selecting Cells.
6. Click **OK**.

   The summary tags are inserted into the selected cell.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Finding a Source Tag or Replicated Tag

You can replicate tag information in an AVEVA Historian from one historian to another. This allows you to replicate tag data from one or more historians (known as tier-1 historians) to one or more other historians (known as tier-2 historians). You can replicate tag data to the same server as the tier-1 historian.

You can replicate tag data directly using simple replication, where the tag information is replicated directly to the tier-2 historian. For simple replication, every value for a tag is copied. You can also set up summary tags that receive a summarized version of the tag data.

Use the Tag Picker to find a source tag or a replicated tag. You can drill down from a source tag to its replicated tag or drill up from a replicated tag to its source tag.

To find a source tag or replicated tag:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
2. Select a tag in the Tag Picker.
3. If the selected tag is a source tag, do the following:
   - In the Tags pane, right-click the selected tag, point to **Find - replicated tag**, and then click the tag that you want to find.

     The application navigates within the Tag Picker to find the corresponding replicated tag.
4. If the selected tag is a replicated tag, do the following:
   - In the **Tags** pane, right-click the selected tag, and then click **Find - source tag**.

     The application navigates within the Tag Picker to find the corresponding source tag.

The **Find** command is not available if:

- Multiple tags are selected in the Tag Picker.
- A normal tag that is neither a source tag nor a replicated tag is selected in the Tag Picker.
- You use the **Select tag** dialog box when performing tag analysis.

Note: You cannot execute the **Find** command if a source tag is deleted but its replication configuration still exists in the Historian.

The replicated tags are not listed in the context menu if:

- The replicated tags are not committed in the Historian.
- The replication schedule is removed from the Historian. For example, you are connected to a Historian 10.0 server and you create a tag called ‘MyTag’. ‘MyTag’ is replicated as a simple tag called ‘MyServer.MyTag’.

  When you execute the **Find - replicated tag** command, the ‘MyServer.MyTag’ tag is shown. When you execute the **Find - source tag** command, the ‘MyTag’ tag is shown. At this instance, if the replication link between ‘MyTag’ and ‘MyServer.MyTag’ is removed and if you execute the **Find - replicated tag** command, the ‘MyServer.MyTag’ tag is not shown in the list of replicated tags.

  However, if you execute the **Find - source tag** command, the ‘MyTag’ tag is shown as ‘MyTag’. If ‘MyServer.MyTag’ is the only replicated tag, ‘MyTag’ is considered as a normal tag.

  The above scenario holds true if the entire replication schedule is removed in the Historian. If only one replication is removed, the list shows the remaining replicated tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Selecting Event Snapshot Tags

To select event snapshot tags:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Even snapshot Tag Selection**. The **Event Snapshot Tag Selection** dialog box appears.
2. In the **Servers** list, click the name of the server to use.
3. In the **Filter** area, configure the criteria by which the tags are filtered and displayed in the **Select tags to insert into workbook** window.
   - **Event tag:** The name of the event tag to which the snapshot tag is related.
   - **Snapshot tag type:** The type of snapshot, either analog, discrete, or string.
4. Select the **Include description** check box to include tag descriptions in the results.
5. In the **Select cell range to insert tags** list, click the name of the workbook cell into which you want to insert the tags. For more information, see Selecting Cells.
6. Click **OK**.

   The tags are inserted into the selected cell.

Retrieving Tag Configuration Information

You can retrieve configuration information for analog, discrete, string, summary, and event tags. You can also retrieve alarm limit information for analog tags.

Retrieving Configuration Details for a Tag

Yo can retrieve configuration details for tags, such as a description. The configuration details that can be retrieved depend on the type of tag. For example, the minimum and maximum values are only applicable for an analog tag.

To retrieve tag details:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Configuration**, and then click **Tag Details**. The **Tag Details - Step 1 of 3** dialog box appears.
2. In the **Server** list, click the name of the server to use.
3. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.
4. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
5. Click **Next**. The **Tag Details - Step 2 of 3** dialog box appears.
6. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
7. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
9. Click **Next**. The **Tag Details - Step 3 of 3** dialog box appears. This dialog box displays different options, depending on the type of tag you have selected.
10. For analog and summary tags, configure the following options:
    - **Description:** The description of the tag.
    - **Raw value range:** The minimum value of the raw acquired value. Also, the maximum value of the raw acquired value.
    - **Engineering units:** The unit of measure.For example mph, grams, and pounds.
    - **Engineering units range:** The minimum value of the tag, measured in engineering units. Also, the maximum value of the tag, measured in engineering units.
    - **Storage rate and type:** The type of storage defined for the tag, either cyclic or delta. The storage rate is the rate at which the tag is stored if the storage type is cyclic.
    - **Source tag:** The source tag of the tag.
    - **Source server:** The source server of the tag.
11. For discrete tags, configure the following options:
    - **Description:** The description of the tag.
    - **Storage rate and type:** The type of storage defined for the tag, either cyclic or delta. The storage rate is the rate at which the tag is stored if the storage type is cyclic.
    - **Messages:** The messages associated with the TRUE/FALSE or ON/OFF state of the tag.
12. For string tags, configure the following options:
    - **Description:** The description of the tag.
    - **Maximum tag name length permitted:** The maximum number of characters for the string.
13. For event tags, configure the following options:
    - **Description:** The description of the tag.
    - **Time deadband:** The minimum time, in milliseconds, between stored events. If more than one event occurs during the deadband, only the most recent are stored. The system does not store another event until the specified time has elapsed. A time deadband of 0 indicates that the system stores all events.
    - **Detector type:** The name given to the type of detector.
    - **Action type:** The name given to the type of action.
    - **Status:** The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.
    - **Logged:** Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured.
    - **Scan rate:** The interval, in milliseconds, at which the system will check to see if the event conditions specified by the detector have occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

14. Click **Finish**. The details appear in the spreadsheet.

Viewing the Hierarchical Name in a Sheet

You can view the hierarchical name in a sheet. For more information, see Integration with AVEVA Application Server.

To view the hierarchical names in a sheet:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
2. Right-click a group or the tag, and click **Use hierarchical name**.

   The **Tags** pane shows the hierarchical names of the selected attributes.

Retrieving Analog Tag Alarm Limits

If a tag is configured to have alarm limits, you can retrieve that information. Examples of limits are Hi, HiHi, Lo, and LoLo alarm limits.

To retrieve analog tag alarm limits:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Configuration**, and then click **Analog Tag Alarm Limits**. The **Alarm Values - Step 1 of 2** dialog box appears.
2. In the **Server** list, click the name of the server to use.
3. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.
4. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
5. Click **Next**. The **Alarm Values - Step 2 of 2** dialog box appears.
6. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
7. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
8. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
9. Click **Finish**. The details appear in the spreadsheet.

Retrieving Tag Values

You can retrieve the following types of values for tags:

- "Live" values
- History values

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Aggregate values
- Summary System values
- Event Shapshot values

Retrieving Live Values

You can retrieve the current data values for specified tags.

To retrieve live values:

1. In cells in your worksheet, enter one or more tagnames (one tagname per cell). For more information, see Selecting Tags for Reports.
2. On the **Historian** tab, in the **Tag Management** group, click **Tag Values**, and then click **Live Values**. The **Live Values - Step 1 of 3** dialog box appears.
3. In the **Server** list, click the name of the server to use.
4. Select the **Support multiple data types** check box to allow for the selection of dissimilar data types for the same query. That is, a mix of analog, discrete, string, and/or event tags.
5. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.
6. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
7. Click **Next**. The **Live Values - Step 2 of 3** dialog box appears.
8. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
9. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
10. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
11. Click **Next**. The **Live Values - Step 3 of 3** dialog box appears.
12. Configure the criteria for the query.
    - **Tag name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
    - **Date time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
    - **Include milliseconds:** Used to include milliseconds in the timestamp.
    - **Quality:** The basic data quality indicator associated with the data value.
    - **Replace poor quality values:** The text string of "poor" will replace the current value if the value has a quality <> 0 or 133.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Detect date time:** Only applicable to event tags. The timestamp reflecting when the event was detected by the event system.
- **OPC Quality:** The quality value received from the data source. Only available if you selected the **Support multiple data types** check box in the **Live Values - Step 1 of 3** dialog box (see step 4 above).
- **Source Tag**: The source tag of the tag.
- **Source Server**: The source server of the tag.

13. Click **Finish**.

Retrieving History Values

You can retrieve history data for specified analog, discrete, string, summary, and/or event tags. However, you cannot retrieve data for event tags and other types of tags in the same query. To retrieve data for event tags, create a separate query that only includes event tags.

To retrieve history values:

1. In cells in your worksheet, enter one or more tagnames (one tagname per cell).
2. On the **Historian** tab, in the **Tag Management** group, click **Tag Values**, and then click **History Values**. The **History Values - Step 1 of 4** dialog box appears.
3. In the **Servers** list, click the name of the server to use.
4. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.
5. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
6. Click **Next**. The **History Values - Step 2 of 4** dialog box appears.
7. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
8. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
9. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
10. Click **Next**. The **History Values - Step 3 of 4** dialog box appears.
11. Configure the criteria for the query.

    See Display Options Tab.

    See Format Tab.

    See Retrieval Tab.

    See Order Tab.

    See Criteria Tab.
12. Click **Next**. The **History Values - Step 4 of 4** dialog box appears.
13. Configure the time for the query. For more information on configuring these options, see Time Options for

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Queries.

14. Click **Finish**.

Display Options Tab

Use the **Display Options** tab to configure the columns to display in the results.

By default, the **Display Options** tab only shows basic display options. For a description of these options, see Display Options Tab.

To see additional options, click the **More >>** button. The following options appear:

- **wwRetrievalMode:** The retrieval mode used for the tag. For more information, see Understanding Retrieval Modes.
- **wwCycleCount:** The cycle count used in data retrieval. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).

  When retrieving data from a AVEVA Historian with a version earlier than 9.0, the wwRowCount column is returned instead of the wwCycleCount column.
- **wwTimeDeadband:** The time deadband used in data retrieval. For more information, see Time Deadband (wwTimeDeadband).
- **wwTimeStampRule:** The timestamp rule used in data retrieval. For more information, see Timestamp Rule (wwTimestampRule).
- **wwVersion:** The history version of the value. For more information, see History Version (wwVersion).
- **wwEdgeDetection:** The type of edge detection used in the query.
- **wwTagKey:** The unique identifier of the tag on the AVEVA Historian.
- **SourceTag:** The source tag of the tag.
- **SourceServer:** The source server of the tag.
- **wwInterpolationType:** The interpolation type used to calculate the value. For more information, see Interpolation Type (wwInterpolationType).
- **wwResolution:** The resolution used in data retrieval. For more information, see Resolution (Values Spaced Every X ms) (wwResolution).
- **wwValueDeadband:** The value deadband used in data retrieval. For more information, see Value Deadband (wwValueDeadband).
- **wwQualityRule:** The quality rule used in data retrieval. For more information, see Quality Rule (wwQualityRule).
- **wwTimeZone:** The time zone that the value's timestamp refers to.
- **PercentGood:** The percentage of rows with good quality in relation to the total number of rows in the retrieval cycle.
- **wwStateCalc:** The state calculation type used to calculate the state time when using ValueState retrieval (for example, average time or total time). For more information, see State Calculation (wwStateCalc).
- **wwFilter:** The analog filter used to filter data during retrieval. For more information, see Analog Value Filtering (wwFilter).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval Tab

Use the **Retrieval** tab to configure the data retrieval mode and additional retrieval options. For a detailed description of retrieval modes and options, see Understanding Retrieval Modes and Understanding Retrieval Options.

Retrieving Aggregate Values

You can retrieve aggregated values for one or more analog tags. Values are calculated using the standard SQL Server aggregation functions.

To retrieve aggregated values from the AVEVA Historian's summary tables, see Retrieving Values for Summarized Tags.

To retrieve aggregate values:

1. In cells in your worksheet, enter one or more tagnames (one tagname per cell).
2. On the **Historian** tab, in the **Tag Management** group, click **Tag Values**, and then click **Aggregate Values**. The **Aggregate Values - Step 1 of 4** dialog box appears.
3. In the **Servers** list, click the name of the server to use.
4. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.
5. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
6. Click **Next**. The **Aggregate Values - Step 2 of 4** dialog box appears.
7. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
8. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
9. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
10. Click **Next**. The **Aggregate Values - Step 3 of 4** dialog box appears.
11. Configure the criteria for the query.

    See Format Tab.

    See Calculations Tab.

    See Resolution Tab.

    See Criteria Tab.
12. Click **Next**. The **Aggregate Values - Step 4 of 4** dialog box appears.
13. Configure the time for the query. For more information on configuring these options, see Time Options for Queries.
14. Click **Finish**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Calculations Tab

Use the **Calculations** tab to specify which calculated values to retrieve from the database.

To specify the calculation

- In the **Calculation type** list, click the type of calculation: Sum, Maximum, Minimum, Average, Range, or Standard deviation. The calculation you choose determines which retrieval mode is used. Delta retrieval is used for the Minimum, Maximum, and Range calculations. Cyclic retrieval is used for the other calculations.

Resolution Tab

Use the **Resolution** tab to specify the "granularity" of the data to be returned.

To configure the resolution:

1. If cyclic retrieval is used for the calculation you selected, configure the following options in the **Cyclic** area.
   - **XX values over equal time intervals:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query.
   - **Values spaced every XX ms:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
   - **Full:** All records between the start and end dates are returned. This option is only available for cyclically-stored tags.
   - **Interpolate:** Linear interpolation is used between stored values. Interpolation only applies for values of cyclically-stored analog tags where no criteria has been specified. Also, the resolution must be set to return all values or to return values spaced according to a time interval.
2. If delta retrieval is used for the calculation you selected, configure the following options in the **Delta** area.
   - **All rows:** Return all rows in the record set.
   - **First XX rows:** The total number of consecutive rows to be returned, starting from the first row in the record set.
   - **Value deadband:** The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval.
   - **Time deadband:** The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.

Retrieving Values for Summarized Tags

You can retrieve summary values for tags that have been configured to be summarized by the AVEVA Historian event subsystem.

To retrieve summary system values:

1. In cells in your worksheet, enter one or more tagnames (one tagname per cell).
2. On the **Historian** tab, in the **Tag Management** group, click **Tag Values**, and then click **Summary System Values**. The **Summary System Values - Step 1 of 4** dialog box appears.
3. In the **Servers** list, click the name of the server to use.
4. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.

5. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
6. Click **Next**. The **Summary Values - Step 2 of 4** dialog box appears.
7. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
8. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
9. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
10. Click **Next**. The **Summary Values - Step 3 of 4** dialog box appears.
11. Configure the criteria for the query.

    See Display Options Tab and Summary Options Tab.
12. Click **Next**. The **Summary Values - Step 4 of 4** dialog box appears.
13. Configure the time for the query. For more information on configuring these options, see Time Options for Queries.
14. Click **Finish**.

Summary Options Tab

Use the **Summary Options** tab to specify the criteria for the type of summary data to return.

To configure the summary criteria:

1. In the **Calculation type** list, select the type of calculation: **Sum**, **Maximum**, **Minimum**, or **Average**.
2. In the **Calculation frequency** list, click the time duration, in seconds, for which the calculation is performed.

Retrieving Values for Event Snapshot Tags

You can retrieve values for snapshot tags associated with a particular event tag.

To retrieve event snapshot values:

1. In cells in your worksheet, enter one or more tagnames (one tagname per cell).

   You must specify both the event tag that is associated with the snapshot action and the snapshot tag.

   For information on selecting an event tag, see Selecting Analog, Discrete, String, Summary, or Event Tags.

   For information on selecting a snapshot tag, see Selecting Event Snapshot Tags. Select the tag(s) associated with the event tag you selected.
2. On the **Historian** tab, in the **Tag Management** group, click **Tag Values**, and then click **Event Snapshot Values**. The **Event Snapshot Values - Step 1 of 4** dialog box appears.
3. In the **Servers** list, click the name of the server to use.
4. In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tag name(s). Click on the button to select the cell(s). For more information, see Selecting Cells.

5. If you want to use a named tag range variable instead, click **Binding Options** and then configure the range. For more information, see Using "Binding" Tags to a Query at Run Time.
6. Click **Next**. The **Event Snapshot Values - Step 2 of 4** dialog box appears.
7. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
8. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
9. Select the **Select cells to specify format options** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
10. Click **Next**. The **Event Snapshot Values - Step 3 of 4** dialog box appears.
11. Configure the criteria for the query.

    For more information, see Display Options Tab.
12. Click **Next**. The **Event Snapshot Values - Step 4 of 4** dialog box appears.
13. Configure the time for the query. For more information on configuring these options, see Time Options for Queries.
14. Click **Finish**.

Common Properties for Tag Values

The data retrieval wizards use some of the same tabs.

Display Options Tab

Use the **Display Options** tab to configure the columns to display in the results.

Note: The availability of options depends on the type of tag(s) selected for the query.

Options are as follows:

- **Tag name:** The name of the tag within the AVEVA Historian server. If the data values are coming from Application Server, the attribute reference is shown as the tag name. You can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions.
- **Date time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
- **Include milliseconds:** Used to include milliseconds in the timestamp.
- **Quality:** The basic data quality indicator associated with the data value.
- **Replace poor quality values:** The text string of "poor" will replace the current value if the value has a quality <> 0 or 133.
- **Detect date time:** Only applicable to event tags. The timestamp reflecting when the event was detected by

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the event system.

- **OPC Quality:** The quality value received from the data source.

For an event tag, if data is returned in the narrow format and the manual history data option is enabled, the **Date time** option is selected by default, and you cannot change it. If the manual history data option is disabled, the **Date time** option is available.

Format Tab

Use the **Format** tab to specify the order in which tags and data are returned and how the results of the query are presented.

The retrieval options you choose determine what appears on the **Criteria** tab. For more information on this tab, see Criteria Tab.

The retrieval options are as follows.

- **Value based criteria (narrow tables):** Data values are returned if they match certain criteria applied to the Value or vValue column. For example, if any possible value > 5000. You can also specify quality criteria for the value. For example, if the data quality for any possible value = Good.
- **Tag based criteria (wide tables):** Data values are returned if they match certain criteria applied to the column for a tagname. For example, if Tagname1 > 5000.

The presentation options are as follows:

- **Narrow query format:** In this format, there is one row for single tag's value for a particular timestamp.
- **Wide query format:** In this format, there is one row for one or more tag values for a single timestamp, thus providing a "wide" view of the data. To use the wide query format, you must specify the timestamp and one or more tagnames as the column names in the query syntax. The results will contain a column for the timestamp and columns for the value of each specified tag at that timestamp.

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.

The filtering criteria options are determined by what you selected for the display format for the returned data, either "narrow" or "wide." For more information, see Format Tab.

For tag based criteria (wide tables), data values are returned if they match certain criteria applied to the column for a tagname. For example, if Tagname1 > 5000.

A NULL value indicates that a column entry has no assigned value. A NULL value is not the same as a numeric value of 0 or an empty string.

For value based criteria (narrow tables), data values are returned if they match certain criteria applied to the Value or vValue column. For example, if any possible value > 5000. You can also specify quality criteria for the value.For example, if the data quality for any possible value = Good.

The value based criteria options that are available in the **Criteria** tab depend upon what types of tags you have selected for the query, either analog, discrete, string, or a mix of these types.

To configure value criteria:

1. To configure criteria for a discrete tag, select the first **Value** check box and set the criteria to be either a 1 or

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

a 0. Go to Step 5.

2. To configure criteria for an analog or summary tag:
   a. Select the first **Value** check box and set the criteria for the data value. For example, the value must be greater than (>) 1500.
   b. (Optional) Select the second **Value** check box and set another criteria for the data value. For example, the value must be less than (<) 2000.
   c. Go to Step 5.
3. To configure criteria for a string tag:
   a. If you are retrieving history values, select the **Use StringHistory** check box. In this case, you can only retrieve data for string tags in the query. No data is returned for tags of other types that you may have selected. This is due to a limitation in the AVEVA Historian.
   b. Select the third **Value** check box and specify text that the returned string value must match. You can specify whether the returned value must equal, start, end, or contain the specified text. For example, you can specify that the value must contain the text "alert."
   c. Go to Step 6.
4. (Optional) Select the **Value not null** check box to filter out NULL values from the results.
5. (Optional) In the **Quality** list, click the quality criteria for the data. Only data values that match the quality you specify (Good, Bad, Doubtful) are returned.
6. (Optional) In the **OPC Quality** list, click the OPC quality criteria for the data. Only data values that match the quality you specify (Good, Bad, Doubtful) are returned.
7. (Optional) In the **Criteria applicability** list, select the moment at which the edge detection criteria is met.
   - **first true:** Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false). This is also known as "leading" edge detection.
   - **no longer true:** Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true). This is also known as "trailing" edge detection.
   - **true:** Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
   - **first true or no longer true:** All rows satisfying both the leading and trailing conditions are returned.

Order Tab

Use the **Order** tab to specify how the results are ordered.

To configure the ordering

1. In the left window, select a column to add to the ordering criteria and then click the arrow button to move the column to the right column. Repeat to add all of the columns you want to the ordering criteria.
2. To move a column up or down in the ordering, select the column in the right window, and then click the up or down buttons. The results are first ordered according to the column that is listed first in the window, then ordered according to the column that is listed second, and so on.
3. In the **Order** area, select whether you want the results to be ordered in ascending or descending order.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Analyzing Tag Data

In addition to creating value reports, you can use the AVEVA Historian Client Workbook to generate statistics, charts, and graphs that are useful for analysis.

- **Analog Tag Analysis.** Create graphs and trends, calculate statistics, and return information regarding configuration and limits.
- **Batch Analysis**. Graph single analog tag over two time periods.
- **Scatter Analysis**. Create a scatter plot of two analog tags.
- **Discrete Tag Analysis**. Create graphs and trends, calculate statistics, and return information regarding configuration.
- **Analog Values at Discrete Transition Analysis**. Graph analog tag values at discrete tag transitions.
- **Analog/Discrete Pair Analysis**. Graph analog vs. discrete tags.

Wizards are provided to guide you though selecting the required options to create the output.

Analog Tag Analysis

Use the Analog Tag Analysis wizard to generate graphs and statistics for an analog tag.

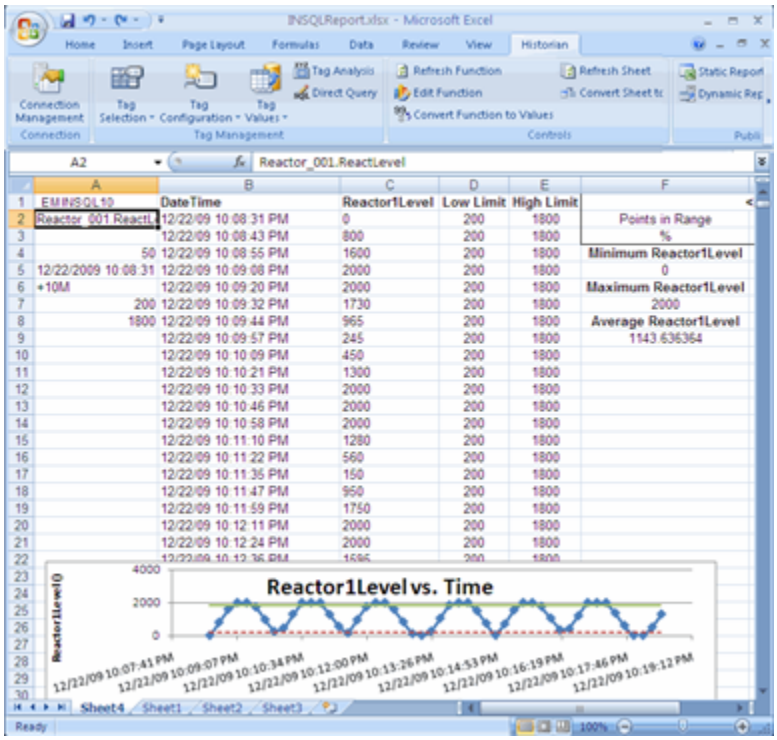
To analyze an analog tag:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
2. Select **Analog tag analysis**.
3. Click **Next**. The **Tag Analysis - Step 2 of 5** dialog box appears.
4. In the **Servers** list, click the name of the server to use.
5. In the **Analog tag** list, specify the name of the tag to analyze. Click the ellipsis button to open the Tag Picker and browse for the tag. For more information, see Tag Picker.
6. Click **Next**. The **Tag Analysis - Step 3 of 5** dialog box appears.
7. In the **Starting time** list, enter the starting time for a query. Click the arrow button to select a date from a calendar.
8. In the **Duration** lists, specify the duration and the duration unit. For example, 10 minutes. The duration is used to calculate the end date for the query.
9. Click **Next**. The **Tag Analysis - Step 4 of 5** dialog box appears.
10. Configure the resolution for the data to be returned.
    - **Number of rows:** The number of rows to be returned for a specified time period using cyclic retrieval. The rows are spaced evenly across the time period, and the default row count is 50 rows. The row count is applied for each tag in a query.
    - **Values spaced every:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
11. Click **Next**. The **Tag Analysis - Step 5 of 5** dialog box appears.
12. Select the analysis options to include.
    - **Plot trend (tag vs. time):** If selected, the value of the tag over time will be plotted in a trend chart.
    - **Statistics:** If selected, tag statistics will be included in the output.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Pie graph:** If selected, a pie graph will be created.
- **Limit 1:** The highest limit as was configured in InTouch. If no InTouch limits are set for the tag, then this value is equal to the maximum engineering unit.
- **Limit 2:** The lowest limit as was configured in InTouch. If no InTouch limits are set for the tag, then this value is equal to the minimum engineering unit.
- **Minimum:** The minimum value for the tag.
- **Maximum:** The maximum value for the tag.
- **Average:** The average value for the tag.
- **Sum:** The sum of all values for the tag.
- **Range:** The difference between the maximum and the minimum value for the tag.
- **Standard deviation:** The statistical standard deviation of all values for the tag.
- **Resolution:** The sampling rate for retrieving the data that is used for calculating the aggregations (Minimum, Maximum, etc.)

13. Click **Finish**. Information that you specified using the wizard is assigned to cells in the worksheet. For example:



In this particular example:

A1: Server

A2: Tag

A4: Row or resolution

A5: Start Time

A6: Duration

A7: Low Limit

A8: High Limit

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Note: The assignments are not absolute and may change in future releases.

14. Click in the Workbook to view the functions that are inserted to create the analysis report.

    In this example, click in the following cells to view the functions.

    Cell C2:

    ```
    =wwAnalogWideHistory(A1,A2,"Row" & A4,"Rel",A6 & "(" & A5 & ")",FALSE," ",TRUE,FALSE)
    ```

    each results cell for the aggregation column (column F):

    For delta stored tags, the resolution you specified in the wizard is used:

    ```
    =wwAggregateWide(A1,A2,"Res60000","Rel",A6 & "(" & A5 & ")","Min","")
    ```

    For cyclic stored tags, the storage rate is used as the retrieval resolution:

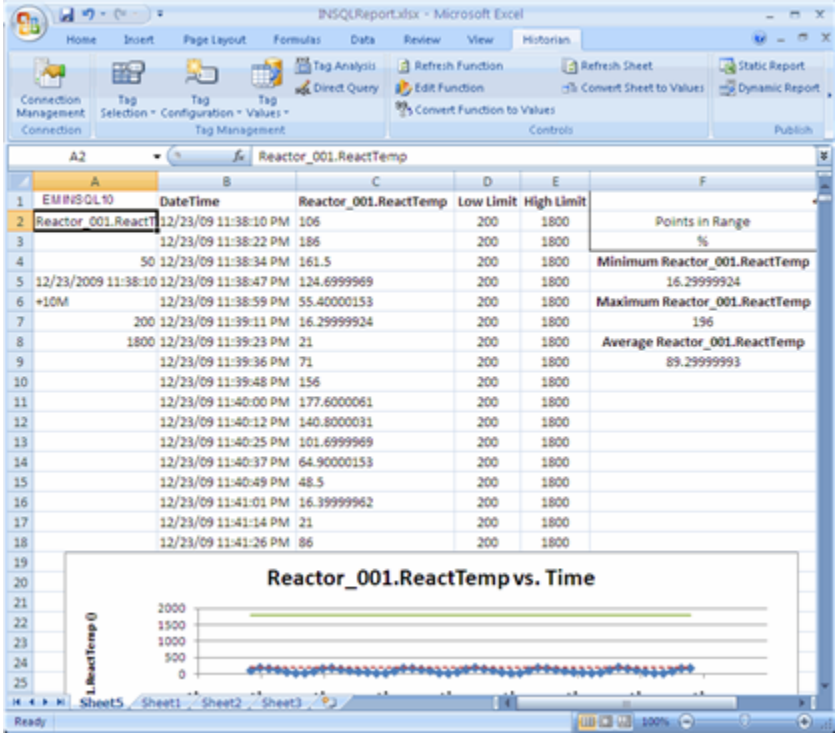
    ```
    =wwAggregateWide(A1,A2,"ResFull","Rel",A6 & "(" & A5 & ")","Min","")
    ```

15. Optionally alter the results by changing values that appear in the first column of the worksheet.

    If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

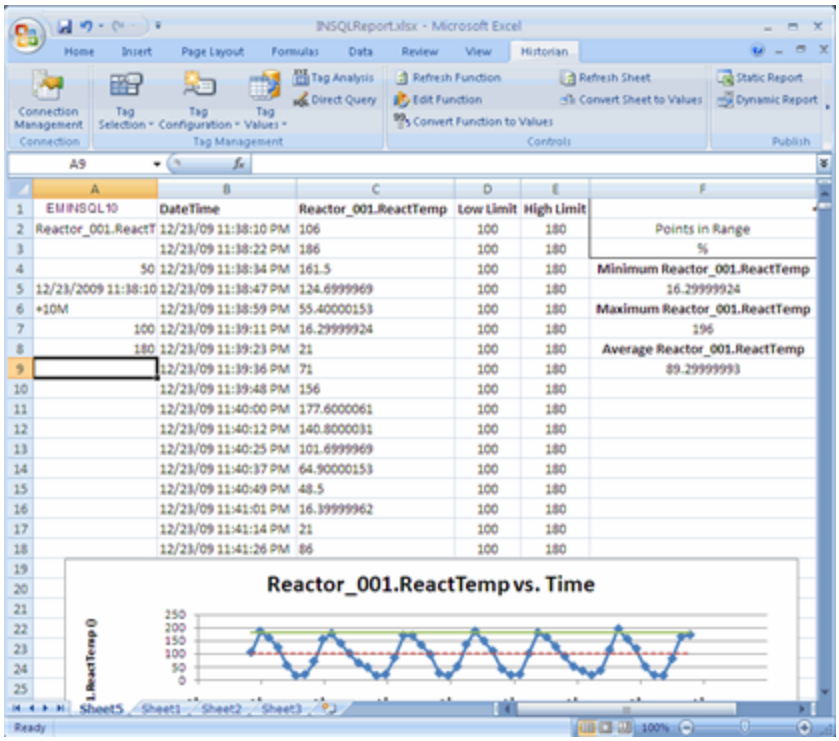
    Note: If you change the resolution and/or duration, this may change the cells that are referenced by the graph and statistics.

    For example, you can change the tagname from ReactLevel to ReactTemp.

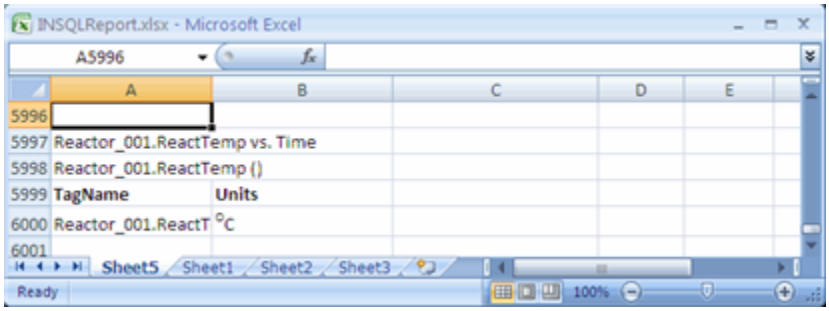


Change the limits to the correct limits so that the statistics and trend chart reflect the data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The information for the analysis headings and units is stored in cells outside of the maximum range for a formula array. To see this information, scroll down in the worksheet to near row 6000.



You can also change the duration in column A1 and watch the data in the report change.

TagAnalysis_AnalogTag_Sheet6

Batch Analysis

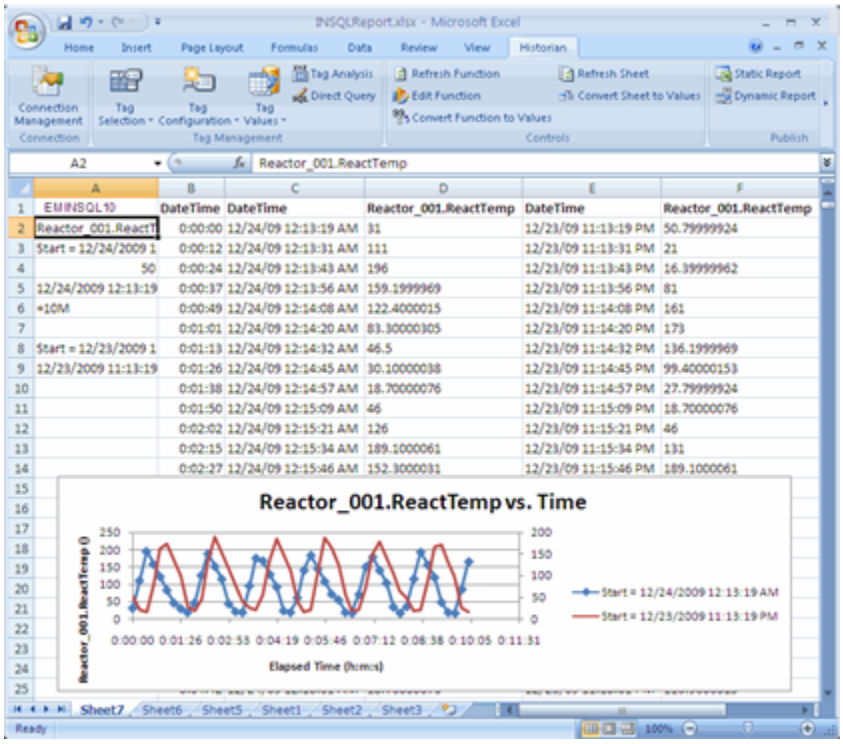
Use the Batch Analysis wizard to graph one analog tag over two time periods.

To create a batch analysis:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
2. Select **Batch analysis**.
3. Click **Next**. The **Tag Analysis - Step 2 of 4** dialog box appears.
4. In the **Servers** list, click the name of the server to use.
5. In the **Analog tag** list, specify the name of the tag to analyze. Click the ellipsis button to open the Tag Picker and browse for the tag. For more information, see Tag Picker.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

6. Click **Next**. The **Tag Analysis - Step 3 of 4** dialog box appears.
7. In the **Starting time** list, enter the starting time for the first time period. Click the arrow button to select a date from a calendar.
8. In the **Starting time for second time period** list, enter the starting time for the second time period. Click the arrow button to select a date from a calendar.
9. In the **Duration** lists, specify the duration and the duration unit. For example, 10 minutes. The duration is used to calculate the end dates for the query.
10. Click **Next**. The **Tag Analysis - Step 4 of 4** dialog box appears.
11. Configure the resolution for the data to be returned.
    - **Number of rows:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 50 rows. For cyclic retrieval, the row count is applied for each tag in a query.
    - **Values spaced every:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
12. Click **Finish**. Information that you specified using the wizard is assigned to cells in the worksheet. For example:



In this example:

A1: Server

A2: Tag

A3: Start Time 1 Chart Legend

A4: Row or resolution

A5: Start Time

A6: Duration

A8: Start Time 2 Chart Legend

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A9: Start Time 2

13. Click in the workbook to view the functions that are inserted to create the analysis report.

    In this example, click in the following cells to view the functions.

    Cell C2:

    ```
    =wwAnalogWideHistory(A1,A2,"Res" & A4,"Rel",A6 & "(" & A5 & ")",FALSE," ",TRUE,FALSE)
    ```

    Cell E2:

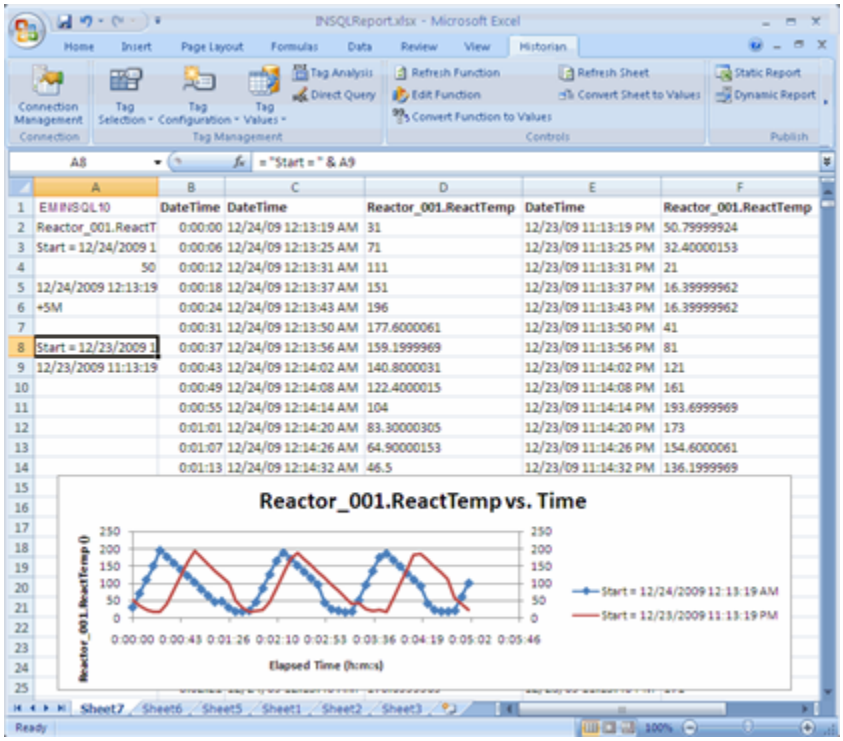
    ```
    =wwAnalogWideHistory(A1,A2,"Res" & A4,"Rel",A6 & "(" & A9 & ")",FALSE," ",TRUE,FALSE)
    ```

14. Optionally alter the results by changing values that appear in the first column of the worksheet.

    If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

    For example, decrease the resolution and time period by 50%.

    Note: If you reduce the resolution by half, but do not adjust the time range (duration), only half of the time is reflected in the trend chart. If you want to make changes to the input parameters that affect the total number of returned rows, you must modify the chart to reference the new cell ranges. You must also refresh the worksheet functions.



Scatter Analysis

Use the Scatter Analysis wizard to create a scatter plot of two analog tags.

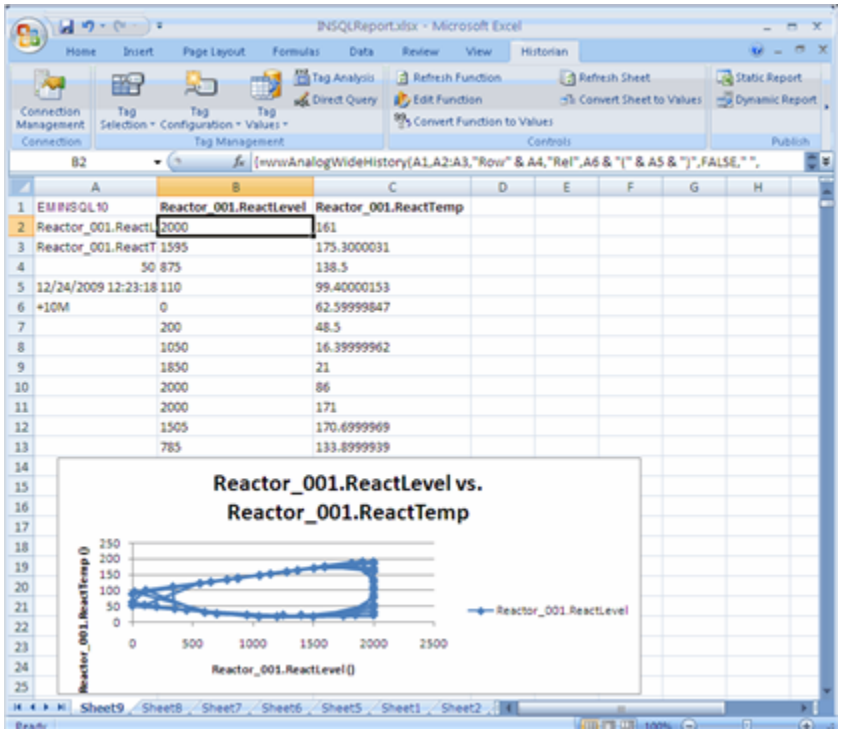
To create a scatter plot:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
2. Select **Scatter analysis**.
3. Click **Next**. The **Tag Analysis - Step 2 of 4** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

4. In the **Servers** list, click the name of the server to use.
5. In the **Analog tags** list, specify the name of the tags to analyze. Click the ellipsis button to open the Tag Picker and browse for the tag. For more information, see Tag Picker.
6. Click **Next**. The **Tag Analysis - Step 3 of 4** dialog box appears.
7. In the **Starting time** list, enter the starting time for the query. Click the arrow button to select a date from a calendar.
8. (optional) To show data for a second time period in the same scatter plot, enter a second starting time in the **Starting time for second time period** list. Click the arrow button to select a date from a calendar.

   Using a second time period allows you to view differences in operation for two time periods.
9. In the **Duration** lists, specify the duration and the duration unit. For example, 10 minutes. The duration is used to calculate the end date for the query.
10. Click **Next**. The **Tag Analysis - Step 4 of 4** dialog box appears.
11. Configure the resolution for the data to be returned.
    - **Number of rows:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 50 rows. For cyclic retrieval, the row count is applied for each tag in a query.
    - **Values spaced every:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
12. Click **Finish**. Information that you specified using the wizard is assigned to cells in the worksheet. For example:



For this example:

A1: Server

A2: Tag 1

A3: Tag 2

A4: Row or resolution

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A5: Start Time 1

A6: Duration

A7: Start Time 2

13. Click in the workbook to view the functions that are inserted to create the analysis report.

    In this example, click in the following cells to view the functions.

    Cell B2:

    ```
    =wwAnalogWideHistory(A1,A2:A3, "Row" & A4,"Rel",A6 & "(" & A5 & ")",FALSE,"
    ",FALSE,FALSE)
    ```

    Cell D2:

    ```
    =wwAnalogWideHistory(A1, A2:A3, "Row" & A4,"Rel",A6 & "(" & A7 & ")",FALSE," ", FALSE,
    FALSE)
    ```

14. Optionally alter the results by changing values that appear in the first column of the worksheet.

    If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

Discrete Tag Analysis

Use the Discrete Tag Analysis wizard to create graphs and trends, calculate statistics, and return configuration information.

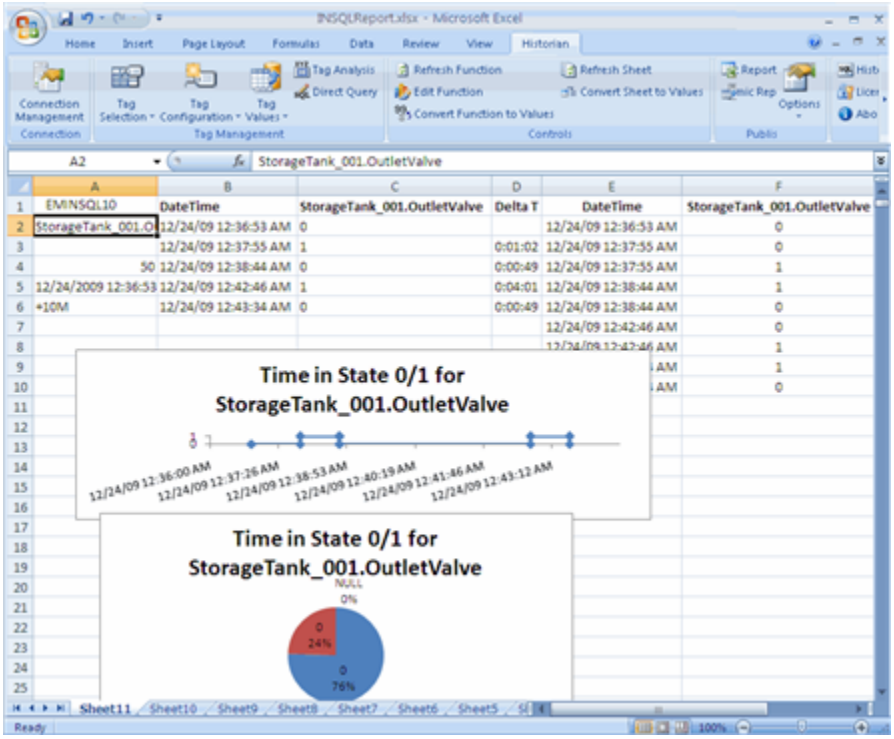
To analyze a discrete tag:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
2. Select **Discrete tag analysis**.
3. Click **Next**. The **Tag Analysis - Step 2 of 5** dialog box appears.
4. In the **Servers** list, click the name of the server to use.
5. In the **Discrete tag** list, specify the name of the tag to analyze. Click the ellipsis button to open the Tag Picker and browse for the tag. For more information, see Tag Picker.
6. Click **Next**. The **Tag Analysis - Step 3 of 5** dialog box appears.
7. In the **Starting time** list, enter the starting time for the query. Click the arrow button to select a date from a calendar.
8. In the **Duration** lists, specify the duration and the duration unit. For example, 10 minutes. The duration is used to calculate the end date for the query.
9. Click **Next**. The **Tag Analysis - Step 4 of 5** dialog box appears.
10. Configure the resolution for the data to be returned.
    - **Number of rows:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 50 rows. For cyclic retrieval, the row count is applied for each tag in a query.
    - **Values spaced every:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
11. Click **Next**. The **Tag Analysis - Step 5 of 5** dialog box appears.
12. Configure the analysis options.
    - **Plot trend (tag vs. time):** If selected, the value of the tag over time will be plotted in a trend chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Statistics:** If selected, tag statistics will be included in the output.
- **Pie graph:** If selected, a pie graph will be created.

13. Click **Finish**. Information that you specified using the wizard is assigned to cells in the worksheet. For example:



In this example:

A1: Server

A2: Tag

A4: Row or resolution

A5: Start Time

A6: Duration

14. Click in the workbook to view the functions that are inserted to create the analysis report.

    In this example, click in the following cells to view the functions.

    Cell B2:

    ```
    =wwDiscreteWideHistory(A1,A2,"Res" & A4,"Rel",A6 & "(" & A5 & ")","",TRUE,FALSE)
    ```

    Cell H2 (time in state)

    ```
    =SUMIF(C2:C11, "0",D3:D12)
    ```

    Cell H4 (number of transitions)

    ```
    =COUNTIF(C3:C12, 0)
    ```

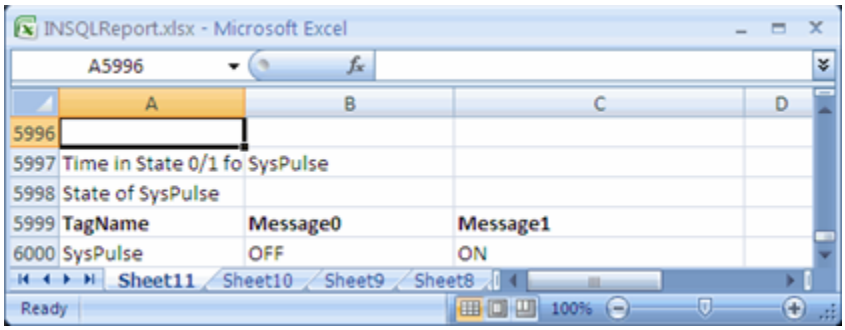
    Columns E and F contain both transitions for each of the dates in Column B.

15. Optionally alter the results by changing values that appear in the first column of the worksheet.

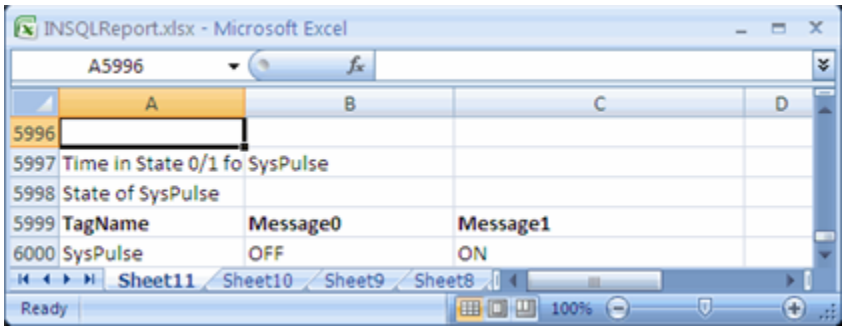
    If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For example, change the tagname to SysPulse. You must refresh the function for columns B and C.



The information for the analysis headings and units is stored in cells outside of the maximum range for a formula array. To see this information, scroll down in the worksheet to near row 6000.



Analog Values at Discrete Transition Analysis

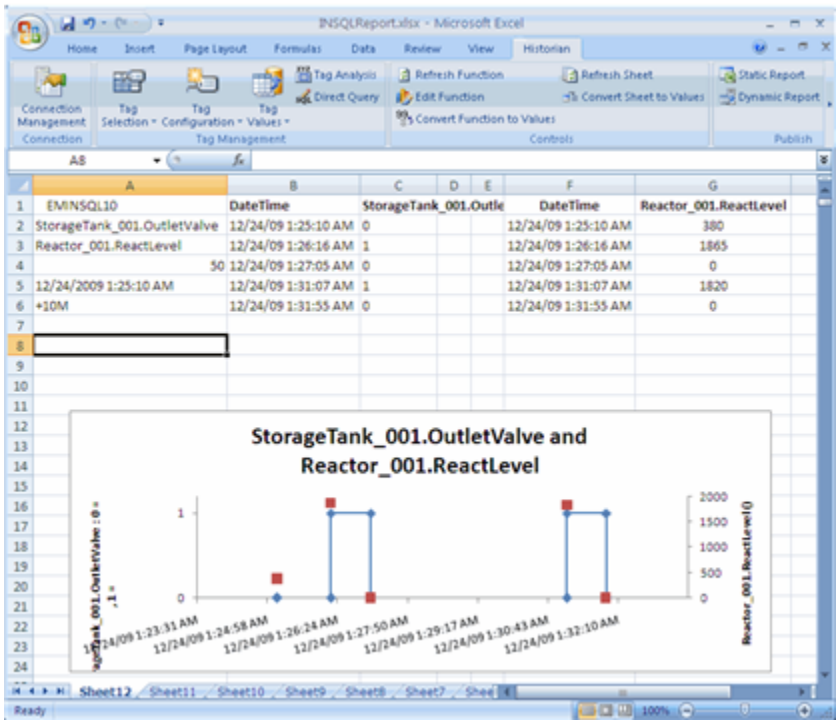
Use the Analog Values at Discrete Transition Analysis wizard to graph analog tag values at discrete tag transitions.

To analyze an analog tag at a discrete transition:

1. On the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
2. Select **Analog values at discrete transition analysis**.
3. Click **Next**. The **Tag Analysis - Step 2 of 4** dialog box appears.
4. In the **Servers** list, click the name of the server to use.
5. In the **Analog tag** and **Discrete tag** lists, specify the names of the tags to analyze. Click the ellipsis button to open the Tag Picker and browse for the tags. For more information, see Tag Picker.
6. Click **Next**. The **Tag Analysis - Step 3 of 4** dialog box appears.
7. In the **Starting time** list, enter the starting time for the query. Click the arrow button to select a date from a calendar.
8. In the **Duration** lists, specify the duration and the duration unit. For example, 10 minutes. The duration is used to calculate the end date for the query.
9. Click **Next**. The **Tag Analysis - Step 4of 4** dialog box appears.
10. Configure the resolution for the data to be returned.
    - **Number of rows:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 50 rows. For cyclic retrieval, the row count is applied for each tag in a query.
    - **Values spaced every:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

11. Click **Finish**. Information that you specified using the wizard is assigned to cells in the worksheet. For example:



In this example:

A1: Server

A2: Discrete Tag

A3: Analog Tag

A4: Row or resolution

A5: Start Time

A6: Duration

12. Click in the workbook to view the functions that are inserted to create the analysis report.

In this example, click in the following cells to view the functions.

Cell B2:

```
=wwDiscreteWideHistory(A1,A2,"Row" & A4,"Rel",A6 & "(" & A5 & ")","",TRUE, FALSE)
```

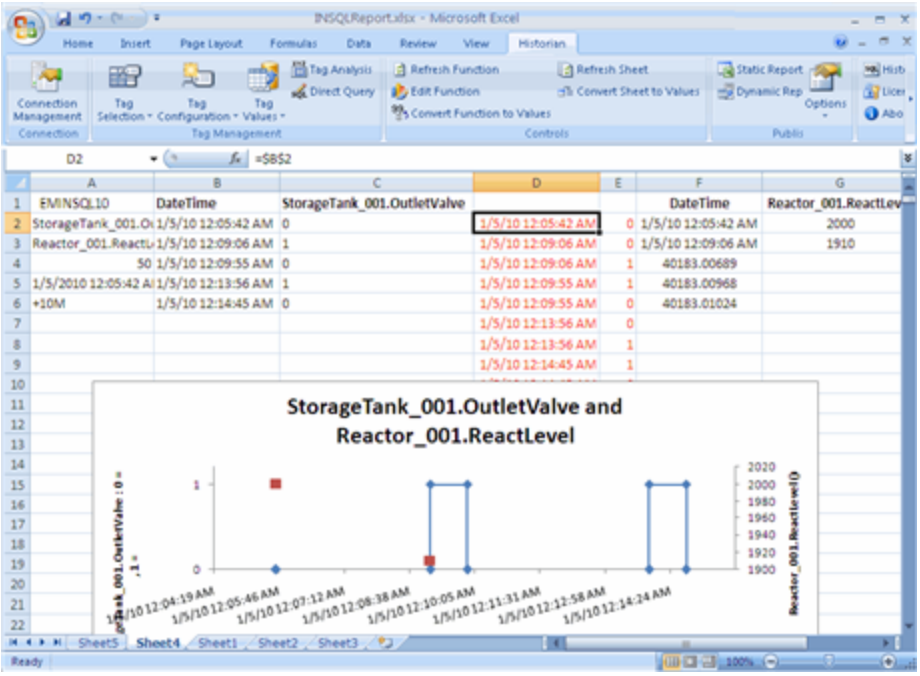
Each cell in F2:

```
=wwAnalogWideHistory(A1,A3,"Row1",B2,B2,FALSE," ",TRUE,FALSE)
```

where the "B2, B2," portion of the function is the associated column used to determine the analog value at the time of the discrete tag transition.

There are "hidden" values that appear in a font that matches the worksheet background. These values are used for the chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



13. Optionally alter the results by changing values that appear in the first column of the worksheet.

    If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

    Keep in mind that changing the time arguments may return a different number of rows in the result set, causing the analysis to be incorrect.

Analog/Discrete Pair Analysis

Use the Analog/Discrete Pair Analysis wizard to graph analog vs. discrete tags.

To analyze an analog-discrete pair:

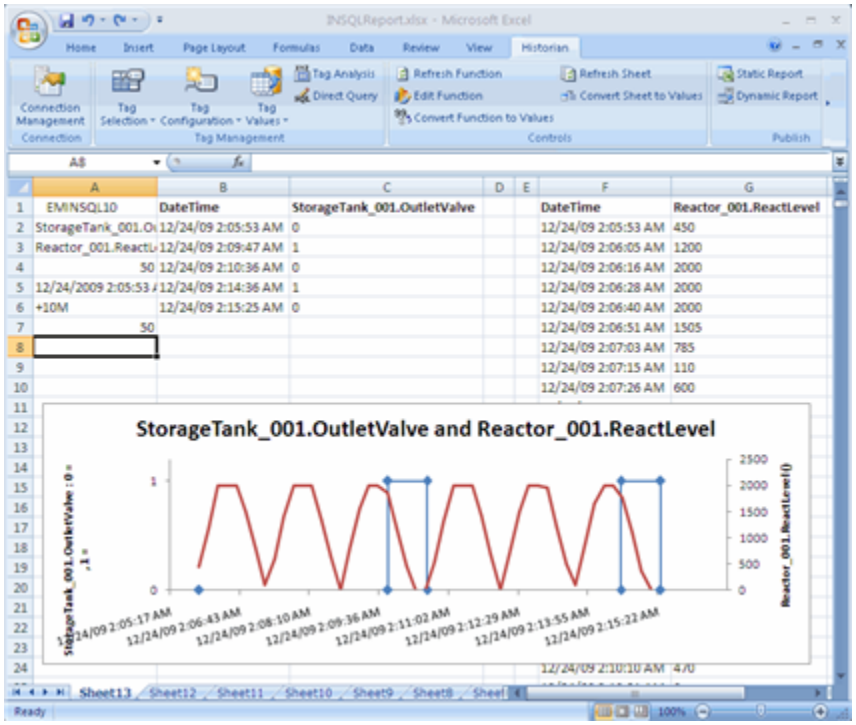
1. On the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
2. Select **Analog-Discrete pair analysis**.
3. Click **Next**. The **Tag Analysis - Step 2 of 4** dialog box appears.
4. In the **Servers** list, click the name of the server to use.
5. In the **Analog tag** and **Discrete tag** lists, specify the names of the tags to analyze. Click the ellipsis button to open the Tag Picker and browse for the tags. For more information, see Tag Picker.
6. Click **Next**. The **Tag Analysis - Step 3 of 4** dialog box appears.
7. In the **Starting time** list, enter the starting time for the query. Click the arrow button to select a date from a calendar.
8. In the **Duration** lists, specify the duration and the duration unit. For example, 10 hours. The duration is used to calculate the end date for the query.
9. Click **Next**. The **Tag Analysis - Step 4 of 4** dialog box appears.
10. Configure the resolution for the data to be returned.
    - **Number of rows:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 50 rows. For cyclic retrieval,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the row count is applied for each tag in a query.

- **Values spaced every:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.

11. Click **Finish**. Information that you specified using the wizard is assigned to cells in the worksheet. For example:



In this example:

A1: Server

A2: Discrete tag

A3: Analog tag

A4: Row or resolution to use for the analog tag

A5: Start time

A6: Duration

A7: Row or resolution to use for the discrete tag

12. Click in the workbook to view the functions that are inserted to create the analysis report.

    In this example, click in the following cells to view the functions.

    Cell B2:

    ```
    =wwDiscreteWideHistory(A1,A2,"Row" & A4,"Rel",A6 & "(" & A5 & ")"," ",TRUE, FALSE)
    ```

    Cell F2:

    ```
    =wwAnalogWideHistory(A1,A3,"Row" & A7,B2,B12,FALSE," ",TRUE, FALSE)
    ```

13. Optionally alter the results by changing values that appear in the first column of the worksheet.

    If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Creating a Direct Query

You can either type in a SQL query (if you know SQL syntax) or use the query builder to create a query. The results are output to the workbook.

To perform a direct query:

1. On the **Historian** tab, in the **Tag Management** group, click **Direct Query**. The **Direct Query** dialog box appears.
2. In the **Servers** list, click the name of the server to use. For more information about this list, see Creating a New Server Connection and Server Connection Configuration.
3. In the **Query** window, type the SQL query to execute against the database.
4. You can also click the query button to start the Query client tool. You can use the Query client to build a query, which is inserted into the **Query** window. For more information, see AVEVA Historian Client Query.
5. In **Select StartDate, EndDate**, specify the workbook cells that show the start date and end date to use in this query.
6. Optionally, in **Select TagName**, specify the workbook cell or range of cells that show the tagname(s) to use in this query.
7. Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells.
8. Select the **Specify format options (select cells)** check box to specify a range of cells that contain formatting information. The formatting information in the cells will be applied to the query results. For more information, see Selecting Cells.
9. In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
10. Click **OK**.
11. To edit the query, click in the cell that contains the red triangle.

Configuring Workbook Options

You can configure global settings related to formatting, time zone usage, data sources, and other general options. You can also set values for formatting and date/time options and then reference them from functions in your workbook. Finally, you can set up custom filters for your reports.

Configuring Global Formatting Options

Formatting is applied to all of the data inserted as a result of using the Workbook wizards.

To configure formatting options:

1. On the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
2. Click the **Format** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Configure the column headings.
   - **Display heading:** Display the column heading for the results in the worksheet.
   - **Bold:** Display the column heading in a bold font.
   - **Italics:** Display the column heading in an italicized font.
4. Select the **Auto-fit results** check box to adjust the worksheet columns so that the entire result text for a column appears.
5. Configure the formatting for numerical values.
   - **Number format:** If set to **General**, the numerical value displayed reflects the original value retrieved from the database. If set to **Fixed**, the retrieved value is rounded to a specified number of decimal places.
   - **Decimal places:** For a number format of **Fixed**, the number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.
6. In the **Date/time format** list, click the formatting for the timestamps.

   The default date format in your workbook is determined by the default language setting of the SQL Server login. To set the date/time format, do either of the following:
   - Change the default language for a SQL Server login. This will also change the default date format of the SQL Server.

     If the default language of the SQL Server login is not set, the language of the SQL Server instance is set as the default. For example, if you install a U.S. English version of the SQL Server, the default language is set to U.S. English.
   - Override the date/time format for the timestamp by using the **Select output cell** list option in Step 10 to contain the formatting settings. Then, reference the Date Format cell in your query to control the timestamp format for the returned data values. You can change the Date Format cell to any format you want, and the timestamp column in the query results will reflect the change after you refresh the sheet.
7. In the **Result alignment** list, click the alignment for the returned data within the worksheet cells.
8. Select the **Check that specified tag(s) exist(s)** check box to validate that the tag exists in the database prior to the function being executed.
9. Select the **Paste format options on worksheet** check box to insert the default formatting options in the worksheet.

   Note: The inserted formatting information is not automatically updated if you change the options.
10. In the **Select output cell** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
11. Click **OK**.

    If you select to output the formatting information, it appears in the sheet.

Referencing Formatting Options in a Query

Before you can reference the formatting options, you must have inserted them into a location in your worksheet. For more information, see Configuring Global Formatting Options.

The following procedure assumes that you are using a wizard to create the query.

To reference formatting options in a query:

1. In the wizard, select the **Select cells to specify format options**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Either type in the cell range that contains the formatting option values displayed in your spreadsheet or select the cells. For more information, see Selecting Cells.

   The query results are formatted according to the option settings. Note that the function references the formatting option cells.

Using a Named Range for Formatting Options

Instead of referencing multiple cells that contain formatting options, you can give the group of cells a name and then reference just that name.

To use a named range:

1. Insert the formatting options into a location in your worksheet. For more information, see Configuring Global Formatting Options.
2. Select the cells that contain the options.
3. In the **Name Box** list, type a name for the cell range.
4. Press Enter on your keyboard.

   You can then specify the named range in your queries.

   The formula references the named range instead of the worksheet cells.

Changing Formatting Options in Named Range

If you have formatting options configured as a named range and are referencing the named range in a query, you can change the value in one or more cells and apply the changes.

To change a format option for a named range:

1. Select one of the cells in the named range and change the value. For example, change the Italic Heading option from FALSE to TRUE.
2. Select the cell containing the function that refers to the format named range.
3. On the **Historian** menu, click **Refresh Function**.

   Note: The **Refresh Sheet** command refreshes the data values, but not the formatting.

   The new formatting options are applied to the results.

Configuring Time Zone Options

You can configure Trend so that data appears with time stamps that reflect any time zone. For example, you may want to configure the Trend to the same time as the location of the server.

To configure time zone options:

1. On the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
2. Select the **Time Zone** tab. The grid displays the current time zone and daylight saving time settings for the following entities:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Entity	Description
Application	The Historian Client Workbook application. You can select the time zone for the data as it appears in the Workbook application.
Client	The physical computer on which the Workbook application is installed. The time zone displayed for the client is for informational purposes only and cannot be changed using the Workbook application.
<Server>	The Historian(s) to which the Workbook application is currently connected. The time zone displayed for the server(s) is for informational purposes only and cannot be changed using the Workbook application.

3. In the **Time zone** list, click the name of the time zone to use for the Workbook application.

   The time zone for the Workbook application in the grid displays the new time zone picked.

   For example, consider a SCADA application that monitors a pipeline between Houston, Texas and Lake Forest, California. The Workbook application is installed on a computer located in Houston, Texas. Therefore, the time zone entry for the Client entity displays Central Standard Time. The server is also located in Houston, Texas. The time zone entry for the Server entity also displays Central Standard Time. You want to send a Workbook file to an engineer located at the start of the pipeline in Lake Forest to aid in troubleshooting a problem. You can set the time zone of the Workbook application to reflect the time of Lake Forest, California (Pacific Standard Time), so that the workbook that you send to the engineer displays data in a time zone that is relevant to him/her.
4. Click **OK.**

Configuring Data Source Options

The data source settings are applied to all functions.

To configure data source options:

1. On the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
2. Click the **Source** tab.
3. In the **Version (Legacy)** area, specify what version of data should be retrieved. This setting is only relevant when retrieving data from an AVEVA Historian with a version earlier than 9.0.
   - **Original:** The original value as it was received from the data source (for example, the I/O Server) to the AVEVA Historian.
   - **Latest:** The latest value that is stored in the AVEVA Historian with the same timestamp as the original value. Multiple versions are created as the result of data inserts and updates.
4. Select the **Retrieve history data from both manual and extension tables** check box to retrieve data from both the manual and extension tables.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Manual history tables:** Normal SQL Server tables that are used to store data. These are the ManualAnalogHistory and ManualDiscreteHistory tables.
- **Extension tables:** Logical tables that are populated from the AVEVA Historian data files. These tables support the AVEVA Historian time domain extensions for handling data.

5. Click **OK**.

Configuring General Options

To configure general options:

1. On the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
2. Click the **General** tab.
3. Select the **Do not update functions when opening worksheet** check box to prevent the functions from being refreshed when the worksheet is opened.
4. Click **OK**.

Setting the Base Date and Base Time Parameters

The base date and time parameters can be used within the history and aggregate functions instead of actual dates/times. The base date and time are stored with the current workbook and affect only the active workbook.

By using these parameters, you can create generic reports that accommodate any date/time; change the base date and base time for the workbook.

To set the base date and time:

1. On the **Historian** tab, in the **Publish** group, click **Options**, and then click **Set Base Date/Time**. The **Set Base Date/Time** dialog box appears.
2. In the **Base date** list, configure the base date.
3. If you want to insert the wwBaseDate() function in a cell, select the **Insert wwBaseDate() function in selected cell** check box and specify location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
4. In the **Base time** list, configure the base time.
5. If you want to insert the wwBaseTime() function in a cell, select the **Insert wwBaseTime() function in selected cell** check box and specify location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see Selecting Cells.
6. Click **OK**. You are prompted to confirm the base date and base time.
7. Click **OK**. If you selected to insert the base date and/or base time, they appear in the spreadsheet.

Using "Binding" Tags to a Query at Run Time

In Excel, a group of cells (a range) can be referenced by single name. A report that you create using the AVEVA Historian Client Workbook can contain the named ranges "AFTagBinding," "AFStartBinding," and "AFEndBinding."

The AFTagBinding range is a placeholder for one or more tags in the query. This range can accept different sets of tags assigned to it, allowing you to programmatically control the tags used for the query without altering the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

actual query string. This is very useful if you are publishing reports on demand; you can "bind" a set of tags or times to the report at runtime.

The AFStartBinding and AFEndBinding ranges work the same as the AFTagBinding range, except that they are used as placeholders for the date and time specification.

The binding values can be used in the following ways:

- Publish the workbook report as a dynamic report to the AVEVA Information Server. The user can select the report and then select a group of tags. When the report is run, the binding ranges are updated with the user-selected information, the queries are executed, and the finished report appears in the browser.
- Programmatically update the ranges using the RunReport method of either the AVEVA Historian Client Workbook add-in or the WorkbookRunner object.

Creating a Bound Report

To create a "bound" report:

1. Start one of the Workbook data retrieval wizards. For example, the **History Values** wizard.
2. Click **Binding Options** to show the binding options.
3. Click **Use bound tags in the range named 'AFTagBinding' of type** and then select the type of tag from the list.

   If the AFTagBinding range does not already exist for the current workbook, a new sheet is added to the workbook. You are prompted to confirm the creation.
4. Click **OK**. The AFBindings sheet is created for the current workbook.
5. You specify to use the AFStartBinding and AFEndBinding ranges when you select the date/time for the query.
6. Click **Finish**. The named ranges are used in the report instead of specific tags and/or starting and ending times.

Considerations for Changing Binding Values

If a graph is configured to display information for *n* number of tags and the tag binding range is programmatically edited to include more tags than that number, the graph does not include the additional tags. The data in the function, however, is updated to include all the tags.

When you develop the report layout, be sure to allow adequate space for additional rows or columns that might appear as the result of changing the binding values or how the add-in RunReport method resizes the functions.

The resizing logic is:

- Check each filled cell in the sheet and determine if the cell contains a AVEVA Historian Client function.
- If a cell contains a function, determine if it is an array function.
- If so, determine if the cell is the first cell in the array.
- If so, determine how many rows the array formula occupies and how many it needs to occupy as the result of the resizing.
- Add additional rows as needed.

If the add-in cannot add rows or additional columns, the function is not updated. The function occupies single

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

cell, indicated by red background as unable to be resized.

Time Options for Queries

You have three choices when setting the starting and ending timestamps for a workbook query: bound times, relative time, and absolute times.

Bound Times

For a "bound" time, the values that are currently assigned to the AFStartBinding and AFEndBinding named ranges are used for the start and end times. For more information, see Using "Binding" Tags to a Query at Run Time.

The **Bound times** option only appears if the **Bound Tags of Type** option is selected in a data retrieval wizard.

Relative Time

You can define a query that uses a relative time in the past for the starting date. For example, the last five minutes from the current time.

To set a relative time:

1. In the **from** list, select the date to use as the start date.
   - **Now:** Use the current date and time.
   - **Today:** Use the current date and a time that you specify.
   - **Specify date:** Use the date and time that you specify.
   - **BaseDate:** Use the date and time as set by the global base date and time options for the workbook. For more information, see Setting the Base Date and Base Time Parameters.
2. If necessary, specify a date and/or time.
3. In the **plus/minus** list, select plus to go forward in time or minus to go backward in time, and then set the duration.

Absolute Time

Absolute time uses fixed start and end dates that you specify. You can either specify a time range or single point in time. If you specify single point in time, only single tag value at that point in time is retrieved.

To set an absolute time range:

- Do one of the following:
  - Select the top option to reference cells in the workbook sheet that contain the start and/or end date. Click the button to the right of the box to select the cell. For more information, see Selecting Cells.
  - Select the bottom option to configure the start and/or end date. Click the button to the right of the box to select a date from a calendar.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To set single point in time:

1. Select the **Single Value** check box.
2. Do one of the following:
   - Select the top option to reference a cell in the workbook sheet that contains the date and time. Click the button to the right of the box to select the cell. For more information, see Selecting Cells.
   - Select the bottom option to specify the date and time directly. Click the button to the right of the box to select a date from a calendar.

Publishing Reports

You can publish spreadsheet reports to the AVEVA Information Server. When you publish a report, the report information is stored in special tables in the AVEVA Historian, and the file is copied to a folder on the AVEVA Information Server. When you publish a report, AVEVA Information Server users can view the report you published with only the browser software.

Published reports are of two types:

- **Static.** For a static report, data is retrieved at the time the report is published to the AVEVA Information Server. After that, its content remains static and does not change when users access it.
- **Dynamic.** For a dynamic report, new data is retrieved from the database every time a user requests the report.

Note the following restrictions when publishing reports:

- Do not publish shared workbooks.
- If you create reports that use the wwHistory2 and wwWideHistory2 data retrieval functions introduced in ActiveFactory 9.2 (for example, reports created using the History Values wizard), do not publish them to a AVEVA Information Server.
- The AVEVA Information Server must be associated with the same AVEVA Historian(s) as the spreadsheet you want to publish.

Publishing a Static Workbook Report

For a static report, data is retrieved at the time the report is published to the AVEVA Information Server. After that, its content remains static and does not change when users access it. If you want current data to be retrieved every time a user requests the report, use a dynamic on-demand report instead.

To publish a static workbook report:

1. Create a workbook sheet and save it as an .xlsx file.
2. On the **Historian** tab, in the **Publish** group, click **Static Report**. The **Publish Report** dialog box appears.

   The **Report name** box displays the name of the workbook report as it appears in AVEVA Information Server. This name is automatically created based on the name of the file that is being published.
3. In the **Server** list, click the name of the AVEVA Historian on which to store the report publishing information.
4. In the **Report site** list, select the URL of the AVEVA Information Server to which you want to publish the report.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The report site may or may not be physically located on the AVEVA Historian computer.

5. In the **Report type** area, click **Static**.
6. In the **Folder** list, click the name of the physical folder on the report site where the static report is posted.
7. Click **OK**. The **Report successfully published** dialog box appears.

Note: The AVEVA Information Server periodically scans the publishing folders for changes; a short delay may occur prior to the published report being displayed in the AVEVA Information Server.

8. To view the AVEVA Information Server, click **Browse**. Otherwise, click **Done**.

Publishing a Dynamic Workbook Report

Dynamic Workbook reports include on-demand and scheduled reports. On-demand reports are executed at a user's request. Scheduled reports are executed automatically on a defined schedule.

To publish a dynamic workbook report:

1. Create a workbook sheet and save it as an .xlsx file.
2. On the **Historian** tab, in the **Publish** group, click **Dynamic Report**. The **Publish Report** dialog box appears.

   The **Report name** box displays the name of the report as it appears on the AVEVA Information Server. This name is automatically created based on the name of the file that is being published.
3. In the **Server** list, click the name of the AVEVA Historian on which to store the report publishing information.
4. In the **Report site** list, select the URL of the AVEVA Information Server to which you want to publish the report.

   The report site may or may not be physically located on the AVEVA Historian computer.
5. In the **Report type** area, click either **On demand** or **Scheduled**, depending on the type of report you want to publish.
6. If you selected **On demand**, in the **Report Group** list, click the name of the physical folder on the report site where the on demand report is posted. Go to Step 9.
7. If you selected **Scheduled**, in the **Schedules** list, click the name of the time period that the scheduled report is run and posted to the AVEVA Information Server.
8. To run the scheduled report immediately without waiting for the selected time period to elapse first, click **Run now**.
9. Click **OK**. The **Report successfully published** dialog box appears.

   Note: The AVEVA Information Server periodically scans the publishing folders for changes. Therefore, a short delay may occur prior to the published report being displayed on the AVEVA Information Server.
10. To view the AVEVA Information Server, click **Browse**. Otherwise, click **Done**.

Migrate data from Historian Client Workbook to the Excel Add-in

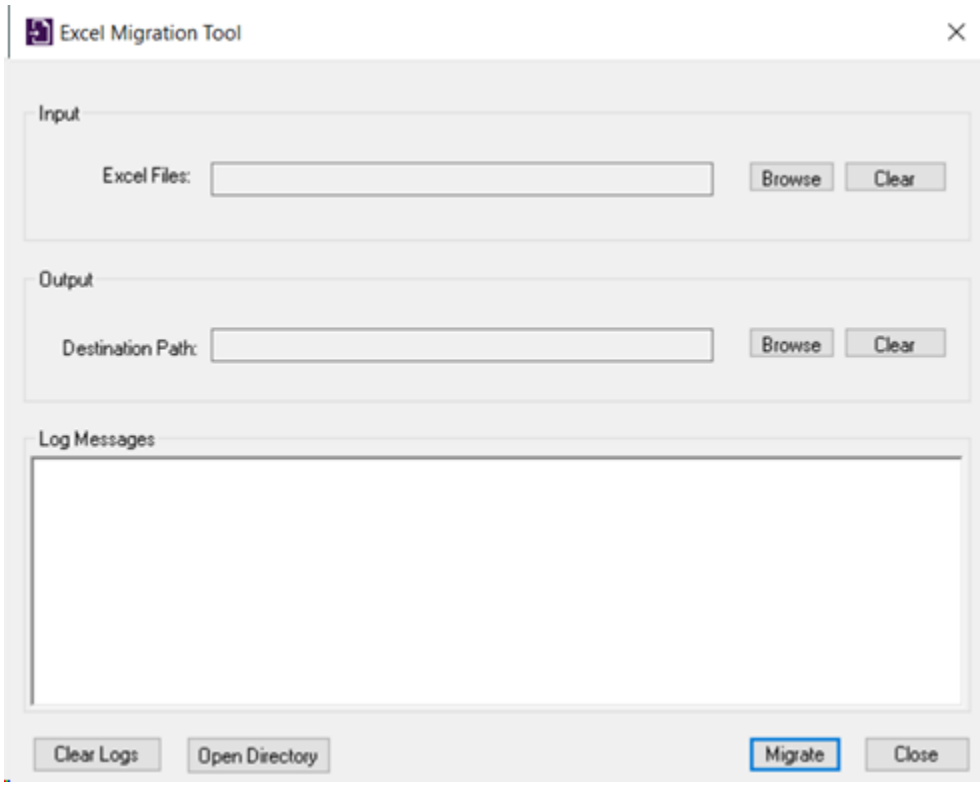
The Excel Migration Tool updates Excel workbooks created in Historian Client Workbook to formulas compatible with the Historian Excel add-in.

This tool is supported to migrate files from Excel 2016 or later and works with Office365 Excel 32/64bit.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Use the Excel Migration tool

1. Open **<Historian installation folder>\Historian\x64\aahExcelMigration.exe**. The default installation path is **C:\Program Files (x86)\Wonderware\Historian\x64\aahExcelMigration.exe**.



2. In the Input area, select **Browse** and select the Excel files that you wish to migrate. (Excel files must be in the .xlsx format.)
3. In the Output area, the Destination Path field shows where the migrated files will be saved. Select **Browse** if you wish to change the path.
4. Select **Migrate**. The migration runs. Information about the status of the migration appears in the Log Messages field. Select **Clear Logs** if you wish to remove the log messages.
5. Select **Open Directory** to open the destination path that the Excel files have been migrated to.

Supported formulas for migration

Formulas	Historian Client Workbook Formulas	Excel Add-in Formula
Live Formulas	wwAnalogLive1, wwDiscreteLive1, wwStringLive1, wwlive1, wwAnalogLive, wwDiscreteLive, wwStringLive, wwReplicatedLive	Historian.SingleValue
Tag Details Formulas	wwAnalogTagDetails1, wwDiscreteTagDetails1, wwStringTagDetails1,	Historian.Info

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Formulas	Historian Client Workbook Formulas	Excel Add-in Formula
	wwReplicatedTagDetails	
History Formulas	wwHistory3, wwWideHistory3, wwStringHistory, wwDiscreteHistory, wwAnalogHistory	Historian.Detail
Aggregate Formulas	wwAggregate, WwwideAggregate. TagAnalysis (Analog Tag Analysis & Discrete Tag Analysis)	Historian.Periodic

Event tags are not supported, so they are omitted during migration.

Live formula migration mapping details

wwlive1 Formula

```
wwLive1 =wwLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality,
DisplaySourceTag, DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplayOPCQuality	OPC Quailty--Details
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

wwAnalogLive1

```
wwAnalogLive1 =wwAnalogLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	**Excel Add-in**
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

wwDiscreteLive1:

```
wwDiscreteLive1 =wwDiscreteLive1( DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	**Excel Add-in**
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

wwStringLive1

```
wwStringLive1 =wwStringLive1( DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

wwReplicatedLive

```
wwReplicatedLive =wwReplicatedLive(DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime,	Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
DisplayMilliseconds	
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

Tag details formula migration mapping details

wwAnalogTagDetails1

`wwAnalogTagDetails1 =wwAnalogTagDetails1( DataSource, TagRange, Description, EngUnit, EURange, RawRange, Storage, SourceTag , SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
Description As Boolean, _	Description
EngUnit As Boolean, _	EngUnit
EURange As Boolean, _	EngUnitMax,EngUnitMin
RawRange As Boolean, _	NA
Storage As Boolean, _	NA
Source Tag	Source Tag
Source Server	NA
Optional OptionRange	NA

wwDiscreteTagDetails1

`wwDiscreteTagDetails1 =wwDiscreteTagDetails1(DataSource, TagRange, Description, Messages, Storage, SourceTag, SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace.

Historian Client Workbook	Excel Add-in
DataSource As String	NA
TagRange As Range	TagList
Description As Boolean	Description
Messages As Boolean	MessageOff,MessageOn
Storage As Boolean	NA
Optional OptionRange	NA
Source Tag	Source Tag
Source Server	NA

wwStringTagDetails1

`wwStringTagDetails1 =wwStringTagDetails1( DataSource, TagRange, Description, MaxLength, SourceTag, SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace.

Historian Client Workbook	Excel Add-in
DataSource As String	NA
TagRange As Range	TagList
Description As Boolean	Description
MaxLength As Boolean	NA
Source Server	NA
Source Tag	Source Tag
Optional OptionRange	NA

wwReplicatedTagDetails

`wwReplicatedTagDetails =wwReplicatedTagDetails(DataSource, TagRange, Description, Storage, SourceTag, SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
Description As Boolean, _	Description
Storage As Boolean, _	NA
SourceTag As Boolean, _	Source Tag
SourceServer As Boolean, _	NA
Optional OptionRange	NA

History formula migration mapping details

Limitations for migrating history formulas

- The Excel add-in only supports Full and Delta Modes. Unsupported modes from Historian Client Workbook are migrated to Full Retrieval as the default.
- Bound time retrievals are not supported.

wwHistory3

`wwHistory3 = wwHistory3(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, RowLimit, TimestampRule, QualityRule, State, StateCalculation, ValueCriteria, EdgeDetection, HistoryVersion, ReplacePoorQuality, DisplayAsWide UseStringHistory, OrderBy, AnalogFilter, DisplayFlags)`

Equivalent formula in the Excel add-in: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
rowOrRes As String, _	NA
time1 As Variant,	StartTime,
time2 As Variant, _	EndTime,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
Optional Interpolation,	NA
Optional RetrievalMode, _	Retrieval(Full and Delta)
Optional TimeDeadband,	NA
Optional ValueDeadband, _	NA
Optional RowLimit, _	NA
Optional TimeStampRule,	NA
Optional QualityRule, _	NA
Optional State,	NA
Optional StateCalculation, _	NA
Optional ValueCriteria,	NA
Optional EdgeDetection, _	NA
Optional HistoryVersion,	NA
Optional ReplacePoorQuality, _	NA
Optional DisplayAsWide,	Format(SingleTag or ALLTags)
Optional UseStringHistory, _	NA
Optional OrderBy,	Ascending,
Optional AnalogFilter,	NA
Optional DisplayFlags, _	Details
Optional OptionRange)	NA

wwWideHistory3

```
wwWideHistory3 = wwWideHistory3( DataSource, TagRange, RowOrRes, Time1, Time2,
InterpolationType, RetrievalMode, TimeDeadband, ValueDeadband, RowLimit, TimestampRule,
QualityRule, State, StateCalculation, TagCriteria, EdgeDetection, HistoryVersion, OrderBy,
AnalogFilter, DisplayFlags )
```

Equivalent formula in the Excel add-in: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
rowOrRes As String, _	NA
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
Optional Interpolation,	NA
Optional RetrievalMode, _	Retrieval(Full and Delta)
Optional TimeDeadband,	NA
Optional ValueDeadband, _	NA
Optional RowLimit, _	NA
Optional TimeStampRule,	NA
Optional QualityRule, _	NA
Optional State,	NA
Optional StateCalculation, _	NA
Optional TagCriteria,	NA
Optional EdgeDetection, _	NA
Optional HistoryVersion,	NA
Optional OrderBy,	Ascending,
Optional AnalogFilter,	NA
Optional DisplayFlags, _	Details
Optional OptionRange)	NA

wwAnalogHistory

```
wwAnalogHistory =wwAnalogHistory( DataSource, TagRange, RowOrRes, Time1, Time2,
Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, ValueCriteria, DetectDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, OptionRange )
```

Equivalent formula in the Excel add-in: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
rowOrRes As String, _	NA
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
Optional Interpolation, _	NA
Optional RetrievalMode, _	Retrieval,
Optional TimeDeadband, _	NA
Optional ValueDeadband, _	NA
Optional ValueCriteria, _	NA
Optional DisplayDateTime, _	Details,
Optional DisplayMilliseconds, _	Details,
Optional DisplayQuality, _	Details,
Optional ReplacePoorQuality, _	NA
Optional OptionRange	NA

wwDiscreteHistory

```
wwDiscreteHistory =wwDiscreteHistory( DataSource, TagRange, RowOrRes, Time1, Time2,
RetrievalMode, TimeDeadband, ValueCriteria, DisplayDatetime, DisplayMilliseconds,
DisplayQuality, ReplacePoorQuality OptionRange )
```

Equivalent formula in the Excel add-in: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
rowOrRes As String, _	NA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
Optional TagCriteria, _	NA
Optional DisplayDateTime, _	Details,
Optional DisplayMilliseconds, _	Details,
Optional OptionRange	NA

wwStringHistory:

```
wwStringHistory =wwStringHistory(DataSource,
TagRange,Time1,Time2,ValueCriteria,DetectDatetime, DisplayMilliseconds, DisplayQuality,
ReplacePoorQuality, OptionRange )
```

Equivalent formula in the Excel add-in: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
Optional ValueCriteria, _	NA
Optional DisplayDateTime, _	Details,
Optional DisplayMilliseconds, _	Details,
Optional DisplayQuality, _	Details,
Optional ReplacePoorQuality, _	NA
Optional OptionRange	NA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag analysis formula mapping details

Tag Analysis includes widehistory and the wide aggregate formulas.

Note that any charts will be ignored.

wwAnalogWideHistory:

`wwAnalogWideHistory =wwAnalogWideHistory( DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, TagCriteria, DisplayDatetime, DisplayMilliseconds, OptionRange )`

Equivalent formula in the Excel add-in: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
rowOrRes As String, _	NA
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
Optional Interpolation, _	NA
Optional TagCriteria, _	Retrieval,
Optional DisplayDateTime, _	NA
Optional DisplayMilliseconds, _	NA
Optional OptionRange	NA

wwDiscreteWideHistory:

`wwDiscreteWideHistory =wwDiscreteWideHistory( DataSource , TagRange, RowOrRes, Time1, Time2, TagCriteria, DisplayDatetime, DisplayMilliseconds, OptionRange )`

Equivalent formula in the current version: `HISTORIAN.DETAIL(TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress)`

Details: DateTime, Resolution, OpcQuality, Unit, and Value.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
rowOrRes As String, _	NA
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
Optional TagCriteria, _	NA
Optional DisplayDateTime, _	Details,
Optional DisplayMilliseconds, _	Details,
Optional OptionRange	NA

Aggregate formula migration mapping details

Limitations for migrating aggregate formulas

- Only Min, Max, Avg, and Sum calculation types can be migrated.
- Single Value option for aggregate formulas is not supported for migration.

wwAggregate

`wwAggregate =wwAggregate( DataSource, TagRange, RowOrRes, Time1, Time2, AggCalc,ValueCriteria, OptionRange )`

Equivalent formula in the Excel add-in: `HISTORIAN.PERIODIC(TagList, StartTime, EndTime, Interval, Details, Down, Ascending, UOMAddress, SliceBy)`

Details:
Average, Minimum, Maximum, Total, Count, Last, StartDateTime, EndDateTime, OpcQuality, PercentGood and Unit.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
rowOrRes As String, _	NA
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
AggCalc As String, _	Details,
Optional ValueCriteria, _	NA
Optional OptionRange	NA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

wwAggregateWide

`wwAggregateWide =wwAggregateWide( DataSource, TagRange, RowOrRes, Time1, Time2, AggCalc, TagCriteria, OptionRange )`

Equivalent formula in the Excel add-in: `HISTORIAN.PERIODIC(TagList, StartTime, EndTime, Interval, Details, Down, Ascending, UOMAddress, SliceBy)`

Details:
Average, Minimum, Maximum, Total, Count, Last, StartDateTime, EndDateTime, OpcQuality, PercentGood and Unit.

Historian Client Workbook	**Excel Add-in**
DataSource As String, _	NA
TagRange As Range, _	TagList
rowOrRes As String, _	NA
time1 As Variant, _	StartTime,
time2 As Variant, _	EndTime,
AggCalc As String, _	Details,
Optional ValueCriteria, _	NA
Optional OptionRange	NA

AVEVA Historian Client Workbook Function Reference

This section describes the AVEVA Historian Client Workbook functions and their arguments. For a description of the arguments, see Function Arguments.

Required arguments are shown in a bold font.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
wwAggregate	=wwAggregate(DataSource, TagRange, RowOrRes, Time1, Time2, AggCalc, ValueCriteria, OptionRange)	Returns an array. This function retrieves values from the AnalogHistory table for specified tags, and then performs aggregate calculations. Single value is returned for each of the selected tags. Quality and value criteria can be specified. Use a row count of 0 to ensure that all stored values are included in the calculations.
wwAggregateWide	=wwAggregateWide(DataSource, TagRange, RowOrRes, Time1, Time2, AggCalc, TagCriteria, OptionRange)	Returns an array. This function retrieves values from the AnalogWideHistory table for specified tags, and then returns an aggregate value for each tag. Use a row count of 0 to ensure that all stored values are included in the calculations.
wwAlarmLimits	=wwAlarmLimits(DataSource, TagRange, OptionRange)	Returns an array.
wwAnalogHistory	=wwAnalogHistory(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, ValueCriteria, DetectDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array. If the retrieval mode is set to delta, then a time deadband and value deadband may be specified.
wwAnalogLive	=wwAnalogLive(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array.
wwAnalogLive1	=wwAnalogLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
wwAnalogTagDetails	=wwAnalogTagDetails(DataSource, TagRange, Description, EngUnit, EURange, RawRange, Storage, OptionRange)	Returns an array.
wwAnalogTagDetails1	=wwAnalogTagDetails1(DataSource, TagRange, Description, EngUnit, EURange, RawRange, Storage, SourceTag , SourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwAnalogWideHistory	=wwAnalogWideHistory(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, TagCriteria, DisplayDatetime, DisplayMilliseconds, OptionRange)	Returns an array. The cyclic retrieval mode is used.
wwBaseDate	=wwBaseDate()	Returns a string. Inserts the current base date into a cell.
wwBaseTime	=wwBaseTime()	Returns a string. Inserts the current base time into a cell. The base time can be used in conjunction with the base date to specify "relative" time periods for queries.
wwDiscreteHistory	=wwDiscreteHistory(DataSource, TagRange, RowOrRes, Time1, Time2, RetrievalMode, TimeDeadband, ValueCriteria, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality OptionRange)	Returns an array. If the retrieval mode is set to delta, then a time deadband may be specified.
wwDiscreteLive	=wwDiscreteLive(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
wwDiscreteLive1	=wwDiscreteLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwDiscreteTagDetails	=wwDiscreteTagDetails(DataSource, TagRange, Description, Messages, Storage, OptionRange)	Returns an array.
wwDiscreteTagDetails1	=wwDiscreteTagDetails1(DataSource, TagRange, Description, Messages, Storage, SourceTag, SourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwDiscreteWideHistory	=wwDiscreteWideHistory(DataSource, TagRange, RowOrRes, Time1, Time2, TagCriteria, DisplayDatetime, DisplayMilliseconds, OptionRange)	Returns an array. The delta retrieval mode is used.
wwEventHistory	=wwEventHistory(DataSource, TagRange, RowOrRes, Time1, Time2, ValueCriteria, DetectDatetime, DateTime, OptionRange)	Returns an array. The first 100 rows are returned.
wwEventSnapshot	=wwEventSnapshot(DataSource, EventTagRange, SnapshotTagRange, SnapshotTagType, Time1, Time2, DetectDatetime, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array.
wwEventTagDetails	=wwEventTagDetails(DataSource, TagRange, Description, TimeDeadband, DetectorType, ActionType, Status, Logged, ScanRate, Reset, OptionRange)	Returns an array.
wwHistory	=wwHistory(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, ValueCriteria, EdgeDetection, HistoryVersion,	Returns an array. Use with IndustrialSQL Server (AVEVA Historian) 8.0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
	DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality, DisplayAsWide, OrderBy, OptionRange)	
wwHistory2	wwHistory2(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, RowLimit, TimestampRule, QualityRule, State, StateCalculation, ValueCriteria, EdgeDetection, HistoryVersion, ReplacePoorQuality, DisplayAsWide UseStringHistory, OrderBy, DisplayFlags)	Returns an array. Use with IndustrialSQL Server (AVEVA Historian) 9.0.
wwHistory3	= wwHistory3(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, RowLimit, TimestampRule, QualityRule, State, StateCalculation, ValueCriteria, EdgeDetection, HistoryVersion, ReplacePoorQuality, DisplayAsWide UseStringHistory, OrderBy, AnalogFilter, DisplayFlags)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwLive	=wwLive(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality, OptionRange)	Returns an array. Use with IndustrialSQL Server (AVEVA Historian) 8.0 or 9.0.
wwLive1	=wwLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
wwQuery	=wwQuery(DataSource, SQLQuery, OptionRange, StartDate, EndDate, TagRange)	StartDate, EndDate, and TagRange are optional. Returns an array. Returns the results of a SQL statement.
wwRefreshFunction	=wwRefreshFunction()	Refreshes the array-formula of the selected cell. Returns a Boolean value indicating the success or failure of the refresh operation.
wwReplicatedLive	=wwReplicatedLive(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwReplicatedTagDetails	=wwReplicatedTagDetails(DataSource, TagRange, Description, Storage, SourceTag, SourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwStringHistory	=wwStringHistory(DataSource, TagRange, Time1, Time2, ValueCriteria, DetectDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array. The first 100 rows are returned.
wwStringLive	=wwStringLive(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array.
wwStringLive1	=wwStringLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwStringTagDetails	=wwStringTagDetails(DataSource, TagRange, Description, MaxLength,	Returns an array.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
	OptionRange)	
wwStringTagDetails1	=wwStringTagDetails1(DataSource, TagRange, Description, MaxLength, SourceTag, SourceServer, OptionRange)	Returns an array. Use with AVEVA Historian 10.0 or later.
wwSummaryTags	=wwSummaryTags(DataSource, TagFilter, DescriptionFilter, Description, SummaryPeriod, SummaryType, OptionRange)	Returns an array. Returns tags that have been configured to generate summary data. Filters can be used to show which tags are configured for different summary durations or type (min, max, avg, sum).
wwSummaryTagValues	=wwSummaryTagValues(DataSource, TagRange, SummaryType, SummaryPeriod, Time1, Time2, DisplayDatetime, DisplayQuality, ReplacePoorQuality, OptionRange)	Returns an array.
wwTagSearch	=wwTagSearch(DataSource, TagRange, TagFilter, DescriptionFilter, Description, OptionRange)	Returns an array. This function is not inserted into a worksheet cell when executed using the Tag Search menu command.
wwWideHistory	=wwWideHistory(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TagCriteria, EdgeDetection, HistoryVersion, DetectDatetime, DisplayMilliseconds, OrderBy, OptionRange)	Returns an array. Use with AVEVA Historian version 8.0.
wwWideHistory2	=wwWideHistory2(DataSource, TagRange, RowOrRes, Time1, Time2, InterpolationType, RetrievalMode, TimeDeadband, ValueDeadband, RowLimit, TimestampRule, QualityRule, State, StateCalculation, TagCriteria, EdgeDetection, HistoryVersion, OrderBy, DisplayFlags)	Returns an array. Use with AVEVA Historian version 9.0 or later.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Function Name	Syntax	Comments
wwWideHistory3	= wwWideHistory3(DataSource, TagRange, RowOrRes, Time1, Time2, InterpolationType, RetrievalMode, TimeDeadband, ValueDeadband, RowLimit, TimestampRule, QualityRule, State, StateCalculation, TagCriteria, EdgeDetection, HistoryVersion, OrderBy, AnalogFilter, DisplayFlags)	Returns an array. Use with AVEVA Historian 9.5 or later.

Function Arguments

The following arguments are used for the AVEVA Historian Client Workbook functions:

- ActionType
- AggCalc
- AnalogFilter
- DataSource
- DateTime
- Description
- DescriptionFilter
- DetectDatetime
- DetectorType
- DisplayAsWide
- DisplayDatetime
- DisplayFlags
- DisplayMilliseconds
- DisplayOPCQuality
- DisplayQuality
- DisplaySourceServer
- DisplaySourceTag
- DisplayTagName
- EdgeDetection
- EngUnit
- EURange
- EventTagRange
- HistoryVersion
- Interpolation

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Logged
- MaxLength
- Messages
- OptionRange
- OrderBy
- QualityRule
- RawRange
- ReplacePoorQuality
- Reset
- RetrievalMode
- RowLimit
- RowOrRes
- ScanRate
- SnapshotTagRange
- SnapshotTagType
- SourceServer
- SourceTag
- State
- StateCalculation
- Status
- SQLQuery
- Storage
- SummaryPeriod
- SummaryType
- TagCriteria
- TagFilter
- TagRange
- Time1
- Time2
- TimeDeadband
- TimestampRule
- UseStringHistory
- ValueCriteria
- ValueDeadband

ActionType

The unique identifier for a particular type of action. Event tags and actions are linked via this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 =

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary.

- TRUE - Includes the action type in the results.
- FALSE - Does not include the action type in the results.

AggCalc

Specifies what aggregate calculation is performed for the specified tagname. Valid values are:

- MIN - Calculates the minimum value. Delta retrieval is used.
- MAX - Calculates the maximum value. Delta retrieval is used.
- AVG - Calculates the average value. Cyclic retrieval is used.
- SUM - Calculates the total value. Cyclic retrieval is used.
- RNG - Calculates the range between MIN and MAX. Delta retrieval is used.
- STD - Calculates the standard deviation. Cyclic retrieval is used.

AnalogFilter

Specifies the analog filters for wwHistory3 or wwWideHistory3 functions. Valid values are:

Value	Description
NoFilter	Accepts empty string or NoFilter.
SigmaLimit	Accepts one parameter of type double. The default value is 2.0.
ToDiscrete	Accepts two parameters: Cutoff value and operator. The cutoff value signifies the boundary between values that are interpreted as ON and OFF. For double value, the default is 0.0 The operator parameter specifies the value range relative to the cutoff value that is to be converted to the ON value and vice versa. The valid operators are:">", ">=", "<" or "<=". The default value is ">".
SnapTo	Accepts two parameters: Tolerance and base value. The tolerance value is of type double. The default value is 0.01. Base values are comma separated double values. The default value is 0.0. When the SnapTo filter is specified, point values falling within the range (Base value - Tolerance) or (Base value + Tolerance) are forced to the base value before the point goes into further retrieval processing.

For more information, see Analog Value Filtering (wwFilter).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DataSource

The name of the server to use. You can also specify the worksheet cell containing the server to use.

DateTime

The date/time stamp.

Description

TRUE = Include the tag description in the results. Only the tags that meet the filtering criteria are included.

FALSE = Do not include the tag description in the results.

DescriptionFilter

Type a description filter (or reference a cell).

Note: Both % and * are valid wild-card characters.

DetectDatetime

TRUE = Include the date and time stamp of the event detection in the results.

FALSE = Do not include the date/time in the results.

DetectorType

The unique identifier for a particular type of detector. Event tags and detectors are linked via this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6 = Time-based (schedule).

TRUE = Include the detector type in the results.

FALSE = Do not include the detector type in the results.

DisplayAsWide

TRUE = Return the results in the "wide" table format. That is, return a column of values for each tag specified in the function.

FALSE = Return the results in the "narrow" table format. That is, return one value for each tag per row in the result set.

DisplayDatetime

TRUE = Include the date and time stamp in the results.

FALSE = Do not include the date/time in the results.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DisplayFlags

Determines which data columns to include in the results. This parameter is an integer representing a bit pattern where each bit represents a data column. If the bit is set to 1, the column is included in the results.

When you use this parameter with the wwHistory2 function, the bits are as follows (bit 0 is the rightmost bit):

Bit	Display option	Integer equivalent
18	wwStateCalc	262144
17	wwInterpolationType	131072
16	Value timestamp	65536
15	Show milliseconds in timestamp	32768
14	Quality	16384
13	OPC quality	8192
12	Quality detail	4096
11	wwRetrievalMode	2048
10	wwTagKey	1024
9	wwCycleCount	512
8	wwResolution	256
7	wwTimeDeadband	128
6	wwValueDeadband	64
5	wwTimestampRule	32
4	wwQualityRule	16
3	wwVersion	8
2	wwTimeZone	4
1	wwEdgeDetection	2
0	PercentGood	1

For additional information on each display option, see Display Options Tab.

To enable multiple display options, add up their integer equivalents and use the result as the value for this parameter. For example, if you want to use the wwHistory2 function and show the value timestamp with milliseconds as well as the history version, add the following values:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Bit	Display option	Integer equivalent
16	Value timestamp	65536
15	Show milliseconds in timestamp	32768
3	wwVersion	8

In this case, the parameter value is 98312 (65536 + 32768 + 8).

DisplayMilliseconds

TRUE = Include the milliseconds in results. By default, Microsoft Excel does not return milliseconds.
FALSE = Do not include the milliseconds in the results.

DisplayOPCQuality

TRUE = Include the quality value in the results.
FALSE = Do not include the quality value in the results.

DisplayQuality

TRUE = Include the quality value in the results.
FALSE = Do not include the quality value in the results.

DisplaySourceServer

TRUE = Include the source server for the tag in the results.
FALSE = Do not include the source server in the results.

DisplaySourceTag

TRUE = Include the source tag for the tag in the results.
FALSE = Do not include the source tag in the results.

DisplayTagName

TRUE = Include the TagName in the results.
FALSE = Do not include the TagName in the results.

EdgeDetection

The moment at which the edge detection criteria is met. Valid values are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Description
-1	No edge detection.
0	None. Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
1	Leading. Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false).
2	Trailing. Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true).
3	Both. All rows satisfying both the leading and trailing conditions are returned.

EndDate

Specifies the ending date and time for the report.

EngUnit

TRUE = Include the engineering units in the results.

FALSE = Do not include the engineering units in the results.

EURange

TRUE = Include the engineering range in the results.

FALSE = Do not include the engineering range in the results.

EventTagRange

Specifies the cell range that contains the event tag names for the query.

HistoryVersion

Specifies the history version for data retrieval. For more information, see History Version (wwVersion).

When used with the wwHistory2 or wwWideHistory2 functions, valid values are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Description
0	Retrieve the latest values available for a tag.
1	Retrieve the original values historized for a tag.

When used with other functions, valid values are:

Value	Description
0	Not specified.
1	Retrieve the original values historized for a tag.
2	Retrieve the latest values available for a tag.

Interpolation

Specifies the interpolation type for data retrieval. When used with the wwHistory2 or wwWideHistory2 functions, valid values are:

Value	Description
0	Use stair-step interpolation.
1	Use linear interpolation.
254	Use the tag's interpolation setting specified at the AVEVA Historian level.

For more information on each option, see Interpolation Type (wwInterpolationType).

When used with other functions, valid values are:

TRUE = Linear interpolation is performed between stored values. Interpolation is only an option for history functions that use cyclic retrieval for analog tag values and where no criteria is specified. If these conditions are not satisfied, interpolation is not performed.

FALSE = Interpolation not performed.

Logged

Determines whether events are logged to the EventHistory table in the AVEVA Historian. 0 = Events are not logged; 1 = Events are logged.

TRUE = Include the log setting in the results.

FALSE = Do not include the log setting in the results.

MaxLength

TRUE = Include the maximum length of the string tag in the results.

FALSE = Do not include the maximum length in the results.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Messages

TRUE = Include the messages associated with the discrete tag in the results.

FALSE = Do not include the messages in the results.

OptionRange

Reference to a range of cells that contains formatting options, which are applied to the query results. The option range must be nine contiguous cells with the following acceptable values:

Option	Valid values
Date Format	Any valid Excel date format. For example: hh:mm:ss.
Display Heading	TRUE/FALSE
Bold Heading	TRUE/FALSE
Italic Heading	TRUE/FALSE
Result Alignment	right/left/center
Number Format	General/Fixed
Decimal Places (only applied if the Number Format = Fixed)	Integer value (0-10)
Fit Results	TRUE/FALSE
Check tags exist	TRUE/FALSE

If a heading currently exists, and the Display Heading option is set to FALSE, the heading is not deleted. Also, none of the other heading options are applied.

OrderBy

Text string that specifies the result order. For example: ORDER BY DateTime Desc.

QualityRule

Specifies the quality rule for data retrieval. Valid values are options 0, 1, and 3 of the aaQualityRules Enumeration. For more information, see aaQualityRules Enumeration.

RawRange

TRUE = Include the raw range for the tag in the results.

FALSE = Do not include the raw range in the results.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ReplacePoorQuality

TRUE = Poor data quality is replaced with word "poor."

FALSE = Data quality not replaced. This is the default value.

Reset

Setting this value has no effect. Provided for backward-compatibility only. Valid values are: TRUE, FALSE.

RetrievalMode

Specifies the type of data retrieval. When used with the wwHistory2 or wwWideHistory2 functions, valid values are options 0 to 11 of the aaRetrievalMode enumeration. For more information, see aaRetrievalMode Enumeration.

When used with other functions, valid values are:

TRUE = Cyclic retrieval. All stored data for the specified time interval is returned. This is the default retrieval mode.

FALSE = Delta retrieval. Only values that changed during the specified time interval are returned.

RowLimit

Specifies the maximum number of rows for the data retrieval to avoid excessively large result sets. For example, if you set a row limit of 200, the historian only returns the first 200 rows of a query's results. The row limit applies to each query. In the Trend application, multiple queries may be used for the tags in a trend depending on the tags' configuration. Therefore, the total number of rows retrieved may be higher than the row limit you configured in the application options.

For example, assume the following scenario:

- The row limit in the application options is set to 200.
- There are five tags in the trend. Four of them use the application options, but one of them is separately configured for a row limit of 100.

In this case, the four tags that use the application options are combined in a single query, but a separate query is created for the tag with the custom row limit. Therefore, you may receive up to 300 rows for all tags combined.

RowOrRes

Specifies whether the number of rows returned are based on data resolution or a row count. A resolution is the sampling interval between rows returned for the specified time period. A row count is the number of rows to be returned . You can either select a cell containing the RowOrRes value or type in the value.

Examples are:

Row50 = 50 evenly spaced rows returned.

Row44 = 44 evenly spaced rows returned.

Res1000 = *n* number of rows at a 1 second resolution.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ResFull = *n* number of rows at a full resolution. The lowest storage rate of the selected tags is used. This is not an option for tags stored by exception (delta storage).

ScanRate

The scan rate is the interval, in milliseconds, at which the system will check to see if the event conditions specified by the detector have occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).

TRUE = Include the scan rate for the event tag in the results.

FALSE = Do not include the scan rate in the results.

SnapshotTagRange

The range of cells that contains the names of snapshot tags for the query.

SnapshotTagType

Specifies the type of snapshot tag. Valid values are: Analog, Discrete, String, and Summary (if applicable).

SourceServer

TRUE = Include the source server for the tag in the results.

FALSE = Do not include the source server in the results.

SourceTag

TRUE = Include the source tag for the tag in the results.

FALSE = Do not include the source tag in the results.

StartDate

Specifies the starting date and time for the report.

State

Specifies the state for which Time-in-State data is retrieved for a tag. This parameter is only relevant for Time-in-State retrieval mode. It specifies the unique tag state for which Time-in-State information is calculated based on the calculation type specified by the StateCalculation parameter.

StateCalculation

Specifies the calculation type for Time-in-State data retrieval. Valid values are options 0 to 4 of the aaStateCalculation Enumeration. For more information, see aaStateCalculation Enumeration.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Status

The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.

TRUE = Include the status for the event tag in the results.

FALSE = Do not include the status in the results.

SQLQuery

The cell address of a custom SQL query.

Storage

TRUE = Include the storage type (and rate, if cyclic) for the tag in the results.

FALSE = Do not include the storage type or rate in the results.

SummaryPeriod

The time period between summary calculations. Valid values are any of the configured summary periods.

SummaryType

Type of aggregation to be performed. Valid values are: Min, Max, Avg, and Sum.

TagCriteria

Enables criteria to be applied to each of the tags for the query. For example, "AND Tag1 > 33"

TagFilter

The filter string for the tag search. You can also specify a cell address that contains the filter string.

Note: Both % and * are valid wild-card characters.

TagRange

Contiguous range of cells containing the tagname(s) for the query. You can also specify a range of cells that contains the tagnames.

Time1

Determines the dates for the query, in conjunction with the Time2 argument.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

(starting time) = Absolute time. You can either type the starting time or reference a cell containing a valid start time. The value of the Time2 argument is used to specify the end time.

REL = Relative time. The value of the Time2 argument are used to specify a reference from the base date.

This argument accepts date/times as real (double) numbers, in addition to dates.

Time2

Determines the dates for the query, in conjunction with the Time1 argument.

If the Time1 argument contains the starting time, specify the end time of the query.

If the Time1 argument is set to REL (relative time), specify the time relative to the base date. The required form is:

###T(##:##:##)

where

- ### = Number of time units from the date/time.
- T = Time unit (d=days, h=hours, m=minutes, and s=seconds).
- (##:##:##) = Time that the time units are relative to. This time and the base date are used together to determine the date/time. To specify the now as the date/time, leave the parentheses empty.

Examples are:

+20m(5:00)	**Starting at 5:00 of the BaseDate, ending 20 minutes later.**
-20m()	Starting 20 minutes ago, ending now.
-20m(#Time)	Uses the configured base time.

This argument accepts date/times as real (double) numbers, in addition to dates.

TimeDeadband

The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.

TimestampRule

Specifies the timestamp rule for data retrieval. Valid values are options 0, 1, and 3 of the aaTimeStampRules Enumeration. For more information, see aaTimeStampRules Enumeration.

UseStringHistory

TRUE = Include the string history table for query.

FALSE = Do not include the string history table for query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueCriteria

Enables a criterion to be specified for the tagname value. The criterion acts as a calculation filter.

For example, if you are performing aggregate calculations on the tag "SysTimeSec" and want the aggregation based only on values > 20, set this parameter to:

"value > 20 AND value IS NOT NULL"

"quality = 0"

"value > 20 AND quality = 0"

ValueDeadband

The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval.

Error Messages for Functions

The following table lists the error messages produced by the AVEVA Historian Client software for functions.

Error Message	Description
NoServer	No server is configured.
NoConnection	There is no connection the server.
QueryError	A general error occurred while data was being retrieved, most likely due to a SQL syntax problem.
NoRecords	No records were returned.
InvalidTags	The tag range argument does not contain valid tags.
CalcType	The function argument "CalcType" is invalid.
InvalidCyclic	The function argument "Cyclic" is invalid.
Resolution	The function argument "Resolution" is invalid.

For unexpected errors, the actual exception message text is returned.

Migrating History Data Retrieval Functions

Workbooks created with earlier versions of the AVEVA Historian Client software may contain calls to the wwHistory and wwWideHistory functions. Instead of manually updating these calls to use the new wwHistory2 and wwWideHistory2 functions, you can automatically update them using the **History Values** wizard.

To migrate history data retrieval functions:

1. Open the workbook containing the function calls you want to migrate.
2. Click the cell containing the call to wwHistory or wwWideHistory.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. On the **Historian** tab, in the **Controls** group, click **Edit Function**. The **History Values - Step 1 of 4** dialog box appears.
4. Select the **Upgrade function to newer format** check box. Click **Next**.
5. Go through the rest of the wizard as described in Retrieving History Values. After you finish the wizard, the function call is updated to use the new function.

Viewing the AVEVA Historian Details

You can view details for all of the AVEVA Historians that have been configured for use within the AVEVA Historian Client Workbook.

For information about configuring servers, see Server Connection Configuration.

To view server details:

1. On the **Historian** tab, in the **Status** group, click **Historian Server Details**. The **Server Details** dialog box appears.
2. In the **Server** list, click the name of the server for which you want to view details.
3. Review the information for the server.
4. Click **OK**.

Client Report

The AVEVA Historian Client Report is an add-in to Microsoft Word that allows you to query one or more AVEVA Historian or SQL Server databases and return results to a Word document.

About Add-ins and Templates

The AVEVA Historian Client Report is an "add-in" to Microsoft Word. An add-in is a supplemental program that runs within the Microsoft Word application and provides custom features and specialized commands.

If the AVEVA Historian Client Report add-in is installed, an additional menu called **Historian** is added to Microsoft Word.

After the add-in is loaded, the **Historian** menu contains all of the commands you use to create a report document or report template using data from an AVEVA Historian or a normal SQL Server database.

Note: You can manually load/unload the AVEVA Historian Client Report Word add-in. For more information, see Manually Loading/Unloading the Word Add-In.

The AVEVA Historian Client Report default template, HistClient.dot or HistClient.dotm, is a blank template to use as the starting point for any report documents or additional templates that you want to create.

Getting Started

Use this section to get started with the AVEVA Historian Client Report.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To get started with the AVEVA Historian Client Report

1. Create a new Word document based on the HistClient.dot or HistClient.dotm template by doing any of the following:
   - From the **Start** menu on the Windows Taskbar, point to **Programs**, point to the **AVEVA** program group, point to the AVEVA Historian Client program group, and then click **Report**.
   - Open Word. On the **Historian** menu, click **Open Report**. In the **New** dialog box that appears, select to create a new blank document, and then click **OK**.
   - Open Word. Click the **Microsoft Office** button, and then click **New**.
   - Right-click on the HistClient.dot or HistClient.dotm file in Windows Explorer, and then click **New**. By default, the HistClient.dot and HistClient.dotm files are installed in the folders: `C:\Program Files (x86)\Common Files\ArchestrA`, and `C:\Program Files (x86)\Microsoft Office\OFFICE##\STARTUP`

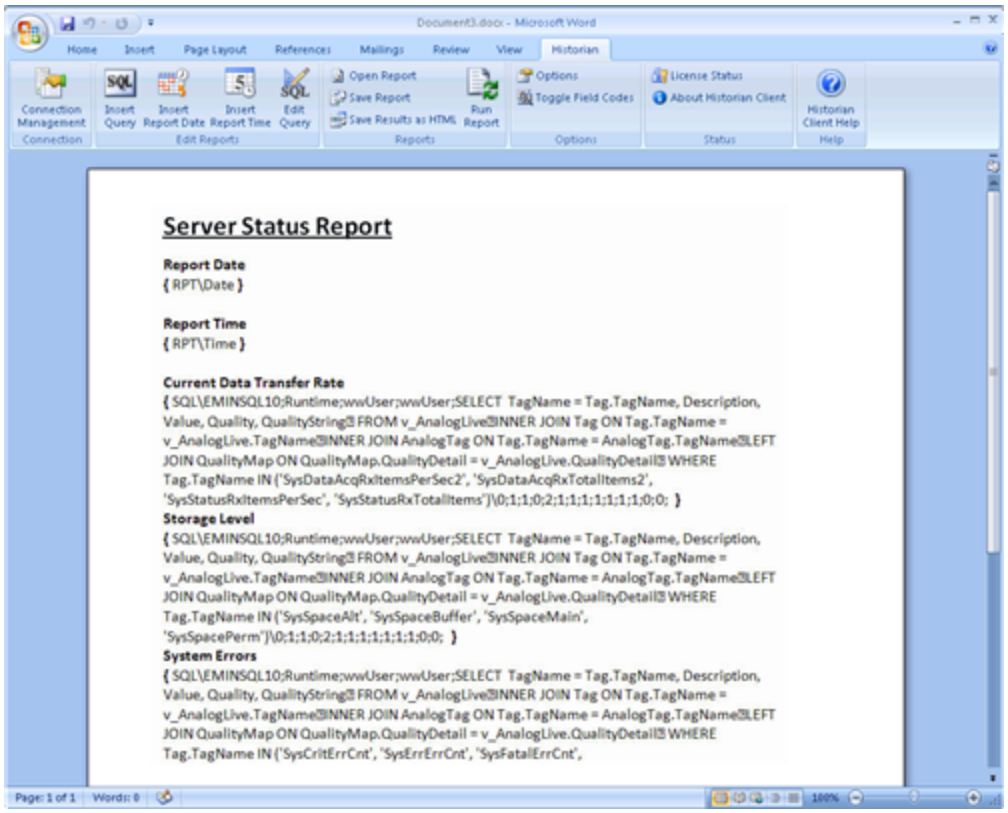
     A new blank document opens in Microsoft Word. The **Historian** menu appears in the Ribbon Bar.

     Note: If you don't see the Historian menu, you may need to manually load the Word add-in.

     Embedded Image (65% Scaling) (LIVE)
2. Configure the connection to one or more servers. For more information, see Managing Server Connections.
3. Create headings, explanatory paragraphs, sections, and so on, similar to a normal Word document.
4. Use the commands on the AVEVA Historian Client menu to insert queries into your report document to retrieve data from the database when the report document is run. The results appear in the final report document. For more information, see Saving a Configured Report Document as a Report Template.
5. Optionally add date and time fields to your report document. For more information, see Inserting Date and Time Field Codes.

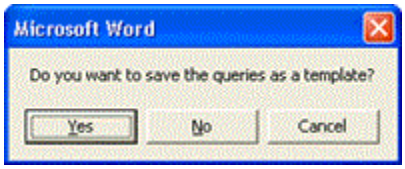
   The following example shows a configured report document that shows status information for an AVEVA Historian, as well as the date and time that the report document was run.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



6. Run the report document. For more information, see Running a Report Document.

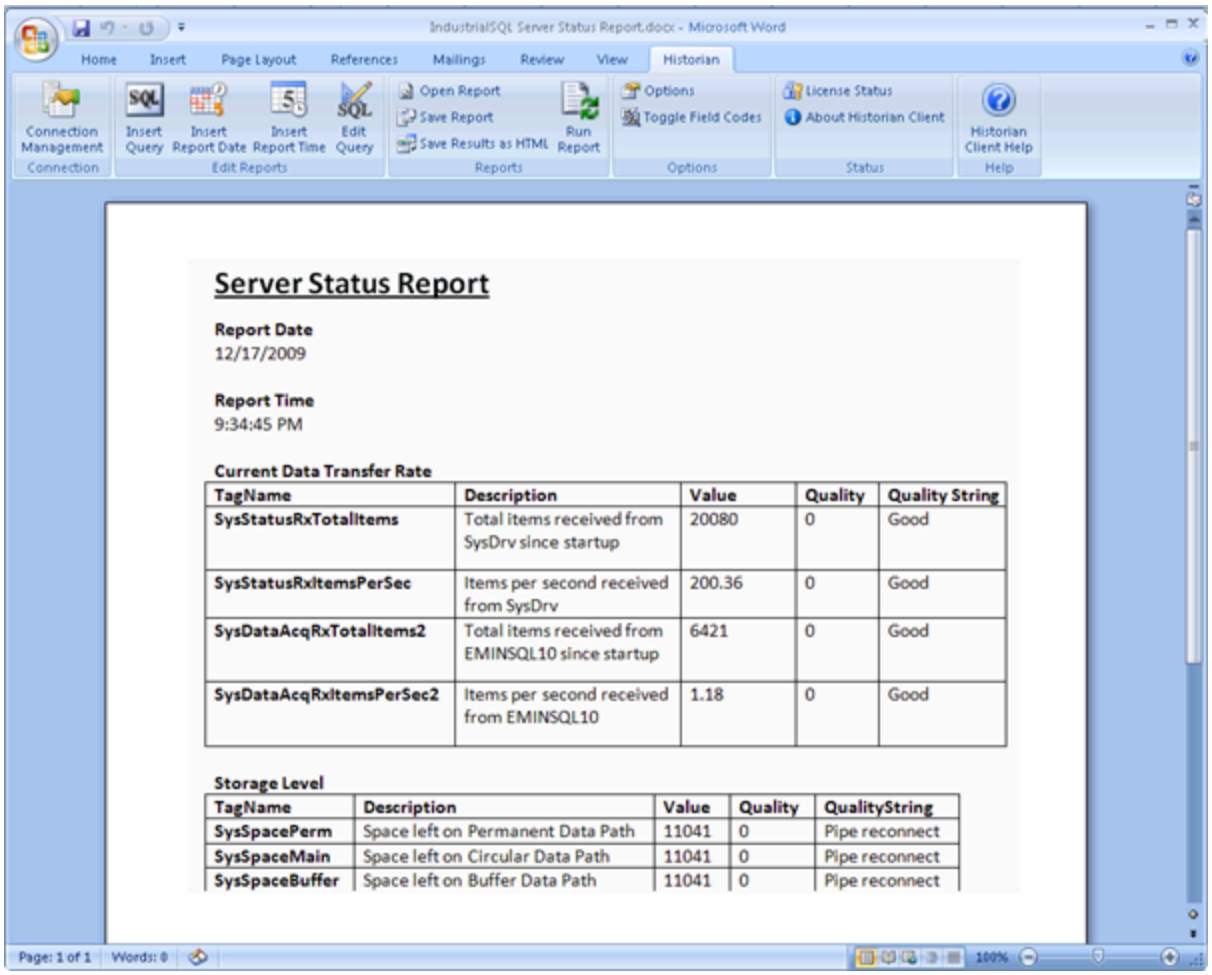
   When you run the report document, you can optionally save the file as a report template, which you can then use as a basis for other report documents, instead of the default HistClient.dot or HistClient.dotm report template.



Note: Running a report document replaces all of the field codes with actual data.

The AVEVA Historian Client Word add-in fills in the report document with the data and the resulting report document appears. For example:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. Save the run report document. For more information, see Saving Report Documents.

Manually Loading/Unloading the Word Add-In

Generally, when you install the AVEVA Historian Client software, the Word add-in is automatically loaded into Microsoft Word. The **Historian** tab appears in the Ribbon bar.

However, if you need to manually load or unload the add-in, use the following procedure.

To manually load the Word add-in

1. Click **File** and then click **Options**.
2. In the Word Options dialog box, click **Add-Ins**.
3. In the Manage list, select **Word Add-ins**, and then click **Go**. The Templates and Add-Ins dialog box appears.
4. Click the **Templates** tab.
5. Click **Add** .
6. Browse and select the the HistClient.dot and HistClient.dotm files.

   By default, they are installed in the following folders:

   `C:\Program Files (x86)\Common Files\ArchestrA`

   `C:\Program Files (x86)\Microsoft Office\OFFICE##\STARTUP`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

7. Mark the **HistClient.dot** and **HistClient.dotm** check boxes.
8. Click **OK**.

To manually unload the Word add-in

1. Click the **Microsoft Office Button,** and then click **Word Options.**
2. Click **Add-Ins**.
3. In the **Manage** list, select **Word Add-ins**, and then click **Go**. The **Templates and Add-Ins** dialog box appears.
4. Clear the **HistClient.dot** and/or **HistClient.dotm** check box in the **Checked items are currently loaded** window.
5. Click **OK**.

Managing Server Connections

You must specify one or more AVEVA Historians and/or SQL Servers as data sources for the AVEVA Historian Client Report.

To manage server connections

1. On the **Historian** tab, in the **Connection** group, click **Connection Management**. The **Server List Configuration** dialog box appears.
2. Configure the server(s) and then click **Close**.

   For more information, see Server Connection Configuration.

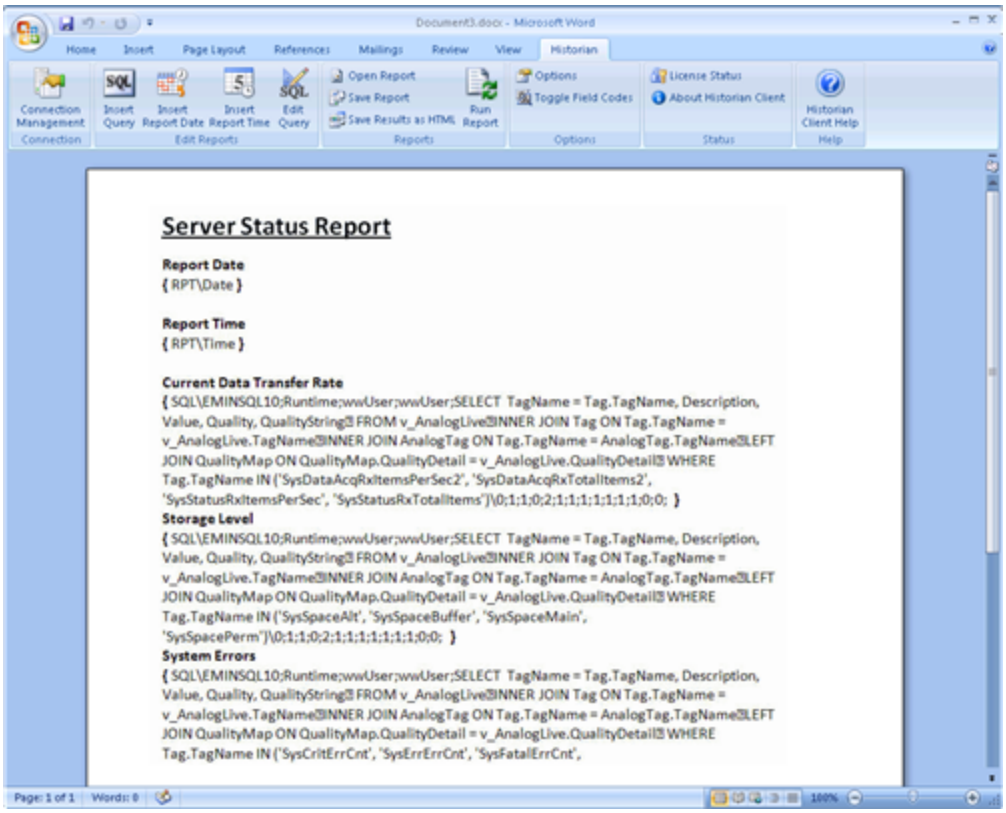
About Field Codes

A field code is a special string of text in a Microsoft Word document that includes instructions for data processing. Field codes can process data from inside the same document or from external sources. For the AVEVA Historian Client Report, field codes are used to contain the instructions for retrieving data from the database and returning the results to the report document.

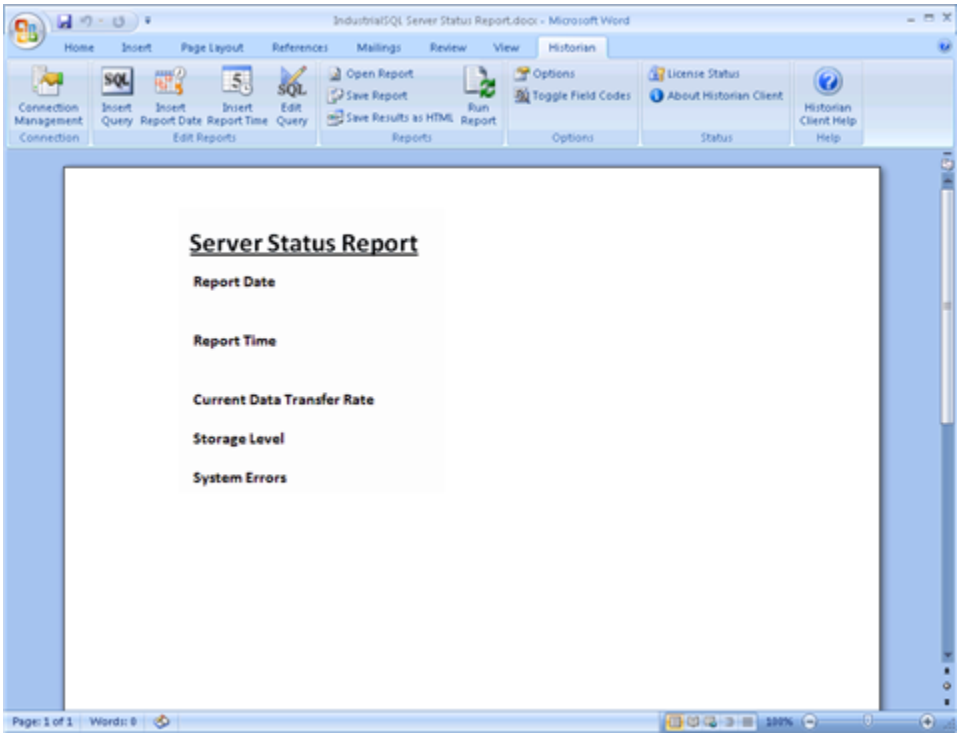
Field codes are present in report templates and report documents that have not yet been run.

The following graphic shows how field codes appear when turned on and before the report document has been run. Field codes appear between the curly brackets { }.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In the following graphic, the field codes are hidden. For more information on showing or hiding the field codes, see Showing/Hiding Field Codes.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Showing/Hiding Field Codes

To show/hide the field codes using the AVEVA Historian Client menu:

1. On the **Historian** tab, in the **Options** group, click **Toggle Field Codes.**

To show/hide the field codes using Word options:

1. Click **Microsoft Office,** and then click **Word Options.** The **Word Options** dialog box appears.
2. Click **Advanced**.
3. In the **Show document content** area, select or clear the **Show field codes instead of their values** check box.
4. Click **OK**.

Opening an Existing Report Document

For information on opening a new report document, see Getting Started.

To open an existing report document:

1. In Word, from the **File** menu**,** lick **Open.**
2. Select your report and click **Open**.

Running a Report Document

To run a report document:

1. Open an existing report.
2. On the **Historian** tab, in the **Reports** group, click **Run Report**.

   When you run the report document, you can optionally save the document as a report template, which you can then use as a basis for other report documents, instead of the default HistClient.dot report template.

   Note: Running a report document replaces all the field codes with actual data.

   The field codes are replaced with data and the resulting report document appears.

Saving Report Documents

You can save a report document as:

- A static .docx file.
- A new report template file.
- An HTML file.

You must save a report document as a report template prior to running the report document.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Saving a Report Document

You can save the report document at any time.

To save a report document:

1. If you want to save a finished report document, run the report document so that data is retrieved and displayed in the report document. For more information, see Running a Report Document.
2. On the **Historian** tab, in the **Reports** group, click **Save Report**. The **Save As** dialog box appears.
3. In the **File name** box, type a name for the report document.
4. In the **Save as type** box, select **Word Document**.
5. Click **Save**. The saved report document appears in Microsoft Word.

Saving a Configured Report Document as a Report Template

You can save a configured report document as a report template. However, the report document must not yet be run so that the field codes are still present in the report document.

To save a report document as a report template:

1. On the **Historian** tab, in the **Reports** group, click **Save Report**. The **Save As** dialog box appears.
2. In the **File name** box, type a name for the report template.
3. In the **Save as type** box, select **Document Template**.
4. Click **Save**. The .dotx file appears in Microsoft Word.
5. You can then copy the report template file into the Microsoft Word template directory and use it to create report documents or as a basis to create new report template files.

Saving a Run Report Document as an HTML File

You can save a run report document as an HTML file so that it can be viewed in a browser. This type of report document is a "static" report document and can be published to a web site such as the AVEVA Information Server.

To save the results as HTML:

1. If you have not done so already, run the report document so that the results appear.
2. On the **Historian** tab, in the **Reports** group, click **Save Results as HTML**. The **Save As** dialog box appears.
3. In the **File name** box, type a name for the report document.
4. In the **Save as type** box, select **Web Page**.
5. Click **Save**. The .htm file appears in Microsoft Word.
6. On the **Quick Access Toolbar,** click **Web Page Preview** to see the report document in a web browser.

Inserting a SQL Query

You can either type in a SQL query or launch the Query application to allow you to build the query using a point-and-click interface.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If you want to use the Query application, you must configure at least one server connection. For more information, see Getting Started with Query.

You can insert a query into either a report document or a report template.

To insert a SQL query:

1. Click in the location in the report document or report template where you want to insert the query.
2. On the **Historian** tab, in the **Edit Reports** group, click **Insert Query**. The **Direct Query** dialog box appears.
3. In the **Servers** list, click the name of the server to use.
4. In the **Query** box, type the SQL query that are executed against the database.

   You can also click the **Query** button to start the Query client tool. You can use the Query client to build a query, which is inserted into the **Query** box. For more information, see AVEVA Historian Client Query.
5. Configure how to display the results in the report document after it has been run.
   - **Show results in a cell:** Displays only the value in the first column of the first record in the returned record set. For example, if you queried the tagname and the description for a tag, only the value for the tagname is returned and displayed.
   - **Show results in a table:** Formats the returned data in a Word table.
6. If you chose to format the results in a table, configure the table options.
   - **Include column headings:** Use the column names for the returned data as column headings in the table.
   - **Show headings on every page:** Allow the column headings to appear automatically on each page of the report document after it has been run, if the data table spans more than one page.
   - **Format table:** Click to pick the table format from a list. The **Table AutoFormat** dialog box appears. For more information on this dialog box, see the Microsoft Word documentation.
7. Click **OK**.

   The query is inserted into the report document or report template.

Editing a Query

You can edit an existing query in a report template or report document that has not yet been run. (After a report document is run, all queries are converted to actual data.)

You can either edit a query manually by typing changes in the query string or by using the **Direct Query** dialog box to select different options.

Both methods require that the field codes for the report document or template are visible. For more information, see About Field Codes.

To edit a query using the Direct Query dialog box:

1. In the report document or template, show the field codes so that you can see the query string.
2. Select the query string so that it is highlighted.
3. On the **Historian** tab, in the **Edit Reports** group, click **Edit Query**. The **Direct Query** dialog box appears.

   The Selected query string appears in the **Query** window.
4. Either edit the query directly, or click the **Query** button to start the Query client tool. For more information, see AVEVA Historian Client Query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. Change any of the options for the results. For more information, see Saving a Configured Report Document as a Report Template.
6. Click **OK**.

Using Date and Time Options

The following date and time options are available for a report.

- Field codes for the report date and time. These codes provide an easy way to display the date and/or time that the report document was run in the body of the finished report document. For more information, see Inserting Date and Time Field Codes.
- Variables for relative date/time parameters in the WHERE clause for queries. For more information, see About Date and Time Wildcards.

Inserting Date and Time Field Codes

You can insert field codes that are replaced with the date and time when the report document runs.

Date and time field codes can be inserted into either a report template or in a report document that has not yet been run. (After a report document is run, all field codes are converted to actual data.)

To insert a date field:

1. On the **Historian** tab, in the **Edit Reports** group, click **Insert Report Date**.

To insert a time field:

1. On the **Historian** tab, in the **Edit Reports** group, click **Insert Report Time**.

   The appropriate field code is added to the report document or template.

   Report Date
   { RPT\Date }

   Report Time
   { RPT\Time }

   When the report document is run, the date and/or time appear. For example:

   Report Date
   26 January 2005

   Report Time
   11:48:19 AM

For more information on viewing field codes, see Showing/Hiding Field Codes.

About Date and Time Wildcards

If you need to create recurring reports that cover the same time period, you can use date and time variables (called "wildcards") in the report template. For example, you might want to produce a daily report that always covers the time period for the first shift, which starts at 8:00 a.m.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The wildcards are:

- #time Wildcard. Used as a placeholder for the current (today's) date, but a specified time.
- #date Wildcard. Used as a placeholder for a specific date and a specified time.
- #ReportTime Wildcard. Used as a placeholder for a defined report time to be used with either #date or #time.

The values used for the #date and #ReportTime wildcards are set for the entire report using the **Report Options** dialog box. For more information, see Configuring Report Options.

These wildcards are handled by the AVEVA Historian Client Report; queries that include them do not work in other query tools, such as the AVEVA Historian Client Query or the Microsoft Query Analyzer.

#time Wildcard

The #time wildcard is used to represent the current date (today) in the query. The use of this wildcard allows you to run the report document on any day and retrieve the data for the same time period.

For example, the WHERE clause for a query for the last eight hours of today's data, starting at 17:00 is as follows:

```
WHERE DateTime >= '#time(17:0:0)-8h'
   AND DateTime <= '#time(17:0:0)'
```

The time specification for the query is indicated in the parentheses.

The valid duration units for the time offset are:

- s = seconds
- mi = minutes
- h = hours
- d = days
- w = weeks
- mm = months

#date Wildcard

The #date wildcard is used to represent a specific date in the query. This wildcard is similar to the #time wildcard. The #time wildcard is a placeholder for the current date, and the #date wildcard is a placeholder for a specific date.

For example:

```
WHERE DateTime >= '#date(17:0:0)-8h'
   AND DateTime <= '#date(17:0:0)'
```

The WHERE clause indicates to use the last eight hours of data, starting at 17:00, for the date that is specified by the **Report Date** option in the **Report Options** dialog box. For more information, see Configuring Report Options.

The time specification for the query is specified within parenthesis.

The valid duration units for the time offset are:

- s = seconds
- mi = minutes

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- h = hours
- d = days
- w = weeks
- mm = months

#ReportTime Wildcard

The #ReportTime wildcard is used to represent the report time in the query. This wildcard can be used with the #time and #date wildcards.

For example:

```
WHERE DateTime >= '#time(#ReportTime)-8h'
   AND DateTime <= '#time(#ReportTime)'
```

This WHERE clause indicates to use the last eight hours of data, for today's date, for the time that is specified by the **Report Time** option in the **Report Options** dialog box.

Another example is:

```
WHERE DateTime >= '#date(#ReportTime)-8h'
   AND DateTime <= '#date(#ReportTime)'
```

This WHERE clause indicates to use the last eight hours of data for the date and time specified by the **Report Date** and **Report Time** options, respectively, in the **Report Options** dialog box. For more information, see Configuring Report Options.

The valid duration units for the time offset are:

- s = seconds
- mi = minutes
- h = hours
- d = days
- w = weeks
- mm = months

Inserting Date and Time Wildcards

For more information on wildcards, see About Date and Time Wildcards.

To use date and time wildcards:

1. On the **Historian** tab, in the **Options** group, click **Options**.
2. Configure the report to use wildcards and set the base date and base time. For more information, see Configuring Report Options.
3. On the **Historian** tab, in the **Edit Reports** group, click **Insert Query**.
4. In the **Query** window, type the SQL query or use the Query client to build the query. For more information, see AVEVA Historian Client Query.
5. Click **OK**. The **Additional Time Options** dialog box appears.
6. In the **Wildcards for time** area, configure which time is substituted for the base time in the query.
   - **(list box):** A base time for the query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Time specified in Report Options:** The time specified in the report options is used as the base time for the query. For more information, see Configuring Report Options.

7. Click **OK**.

   The wildcards are inserted into the query for the date/time parameters and then updated with the appropriate date/time when the report document is run.

Configuring Report Options

To configure general report options:

1. On the **Historian** tab, in the **Options** group, click **Options**. The **Report Options** dialog box appears.
2. In the **Date and time** area, configure the base time and date used for the report wildcards. Every time this report document is run, the same date and time are used.

   For more information about wildcards, see About Date and Time Wildcards.

   - **Report date:** The date to be used as the base date for a relative date/time in the query. Click the arrow button to access a calendar.
   - **Report time:** The time to be used as for a relative date/time in the query.
3. Select the **Suppress messages when report is running** check box to stop dialog box messages from being displayed when the report is running.
4. Select the **Use date/time wildcards** check box to allow for the use of wildcards in a query. You are prompted to specify the wildcards during the query configuration.
5. In the **Maximum rows per query** list, specify the maximum number of rows returned for the query.
6. Click **OK.**

Controls and Objects

The AVEVA Historian Client controls and objects can be run in any application that can function as a .NET or an ActiveX control container, such as InTouch HMI software, Visual Basic, Visual C#, Visual C++, web pages, and so on. For InTouch HMI software, you can select these controls from within WindowMaker when you create your runtime graphical user interface.

The AVEVA Historian Client objects and controls must be installed on the computer running the application that you want to use them in. For example, if you want to use the aaHistClientTrend control in InTouch HMI software, you must install the Trend files on the InTouch computer.

Technically, the ActiveX versions of the controls can be used within Internet Explorer. However, because Internet Explorer is a native .NET control container, use the native controls instead of the ActiveX versions.

About the AVEVA Historian Client Controls and Objects

The AVEVA Historian Client controls can be categorized as either "application" controls, "building block" controls, or "core functionality" controls.

An application-level control runs within the container application, but functions as if it were a stand-alone application. This type of control does not require extensive scripting to function. Application-level controls include:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- aaHistClientTrend Control
- aaHistClientQuery Control

A building block control provides specific functionality for use within an application. Scripting is required to make these controls functional. Building block controls include:

- aaHistClientTimeRangePicker Control
- aaHistClientTagPicker Control
- aaHistClientSingleValueEntry Control
- aaHistClientActiveDataGrid Control

The following low-level controls and objects are used by either an application or building block control. Core functionality controls include:

- aaTag Object
- aaServer Object
- aaServers Object
- aaHistClientWorkbookRunner Object
- aaHistClientReportRunner Object

About Properties, Methods, and Events

There are three main aspects of controls: properties, methods and events.

- Properties are attributes of the control that you can set. For example, a property can control what background color is used for the trend chart in the Trend control.
- A method performs a function for a control. For example, a method can set the time span for the query.
- An event is an occurrence of something within or to the control (such as a mouse click or a data change) that you might want to respond to through scripting (known as an event handler).

Getting Started with the Controls

When you use an AVEVA Historian Client control in a container application (for example, InTouch HMI software), perform the following for each control:

- Name the control.

  When you first place a control in an application, a name is assigned to it by default. You can change this name to something more meaningful to you.

  Also, if you use more than one instance of the same control in your application, you must distinguish them by giving them different names.

  For information on naming a control, see the documentation for your container application (for example, your InTouch User Documentation).
- Configure general properties.

  General properties pertain to how the control appears to the user at runtime. General properties can be

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

configured through a user interface property panel during the design of your application, or at runtime with scripting in the container application.

- Use any of the control's properties, methods, and events in scripts in your application.

Using the Controls in Different Environments

There are two versions for each of the AVEVA Historian Client controls: one for use within a ActiveX control containers and another for within .NET containers. ActiveX control containers include Visual Basic, Visual C++, InTouch HMI software, and so on. .NET containers include VB.NET, C#, ASP.NET, and so on.

The ActiveX versions are named according to the following convention:

aaHistClientXXX Control

The .NET versions are named as follows:

ArchestrA.HistClient.UI.aaXXXControl

For example, the Trend control is implemented as two versions: aaHistClientTrend Control and ArchestrA.HistClient.UI.aaTrendControl.

If a container supports both .NET and ActiveX controls, such as Internet Explorer, use the .NET version, since that is the native form of the controls for the AVEVA Historian Client software.

Using the Controls within InTouch HMI Software

Before any control can be used in the InTouch HMI software, it must be installed. See the InTouch documentation for information on how to install a control and insert it into an application window.

You must assign InTouch tagnames to the properties of a control. Keep in mind that a property must be assigned to the equivalent InTouch tagname type. For example, a message property must be assigned to an InTouch message tagname. Although you can use the **Properties** dialog box to assign tagnames to properties, it is easier to set the properties directly through QuickScripts.

For events, if the window containing a control is closed, its event scripts and any other InTouch QuickScripts containing script functions associated with that control do not execute properly.

Using the Controls in Microsoft Office

To use the AVEVA Historian Client controls on a Visual Basic for Applications (VBA) user form, add them to the form's `Controls` collection dynamically using a call like the following:

```
Set NewControl = UserForm1.Controls.Add( <control's ProgID etc.> )
```

It is not possible to drop them on the user form in the visual editing environment.

Mapping for Numerical Data Types

The following rules explain how data types are handled for the containers in which the AVEVA Historian Client controls can be run.

In C# or .NET environments:

- Byte = 8 bits

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Int = 32 bits
- Long = 64 bits
- Short = 16 bits

In C++ or IDL environments (versions prior to ActiveFactory 9.0 software):

- Byte = 8 bits
- Int = 32 bits
- Long = 32 bits
- Short = 16 bits

The size of `long` in the C++/IDL environment is the same as the size of `int` in C#.

In the InTouch HMI software, an integer value is stored in 32 bits.

Controls

This section describes how to work with the AVEVA Historian Client controls.

aaHistClientTrend Control

The aaHistClientTrend control allows you to run the AVEVA Historian Client Trend program (or a functional subset) from within the AVEVA InTouch HMI software or a .NET container like Visual Basic .NET or Internet Explorer.

For more information on using the AVEVA Historian Client Trend, see AVEVA Historian Client Trend.

Using aaHistClientTrend at Runtime

At runtime, aaHistClientTrend trends configured tag values from the AVEVA Historian in the container application. You can use aaHistClientTrend just as you would use the AVEVA Historian Client Trend application.

For more information on using AVEVA Historian Client Trend, see AVEVA Historian Client Trend.

Using aaHistClientTrend in an Application

aaHistClientTrend is capable of running with all of the functionality of the AVEVA Historian Client Trend application.

You can also use the aaHistClientTrend control's properties, methods, and events in runtime scripts in your application to control the functionality that is available to the runtime user.

For example, maybe you want to limit the functionality of aaHistClientTrend to only allow the runtime operator to connect to an AVEVA Historian, load a set of predefined tags, and then trend them in live mode.

The following InTouch script illustrates how to log on to the server, add a tag to the trend, hide some of the navigation controls for a trend, and start the trend in live mode.

```
#aaHistClientTrend1.AddServer("MyInSQL", "wwUser", "wwUser", 1);
#aaHistClientTrend1.AddTag("MyInSQL", "SysTimeSec");
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
#aaHistClientTrend1.TagPickerVisible = 0;
#aaHistClientTrend1.TimeBarVisible = 0;
#aaHistClientTrend1.RealTimeMode = 1;
```

Adding aaHistClientTrend to an InTouch Window

There are two ways you can add the Trend control to an InTouch window: using the ActiveX Controls wizard, or by adding a Graphic object containing the Trend control to your window.

In both cases, you can edit the properties of the control at runtime with scripting.

If you use the Trend control embedded within a graphic object, you can set the default properties of the control within the IDE with no scripting required.

To add the aaHistClientTrend control using the wizard

1. In WindowMaker, select the **Wizards** button. The **Wizard Selection** dialog box appears.
2. Select the aaHistClientTrend control.
3. Click **OK**.

   The control appears in the window.

To add the Trend control using a graphic object

1. In WindowMaker, create a new graphic, then open it for editing.
2. Select **Trend Control** from the Toolbox, then click on the graphic to add the control.
3. Save and close the new graphic.
4. Drag the new graphic from the toolbox onto the canvas.

   The control appears in the window.
5. To change the initial properties of the Trend control, right-click the control and select **Properties...** to display the Trend properties window. Select the **Properties** tab and set the starting property values as required.

For more information about working with InTouch, refer to the System Platform documentation.

For more details about the Trend control's properties, see aaHistClientTrend Properties.

Using aaHistClientTrend in a Multi-Monitor Environment

By default, dialog boxes shown by the aaHistClientTrend control appear in the middle of the screen. This may be a problem in multi-monitor configurations where multiple screens are combined into one large logical screen. To avoid this, you can specify a screen position where dialog boxes should appear.

To specify a screen position for dialog boxes

1. Open the `C:\Windows\win.ini` file in your Windows folder using a text editor.
2. Look for the `[HistClient]` section. If no such section exists, create one.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Add the following lines to the `[HistClient]` section:

   `UsedFixedWindowPosition=1`

   `FixedWindowPositionX=<XPos>`

   `FixedWindowPositionY=<YPos>`

   where `<XPos>` is the horizontal position (in pixels) where you want dialog boxes to appear, and `<YPos>` is the vertical position. For example, `FixedWindowPositionX=300`.
4. Save the `win.ini` file and restart the Trend application.

   Dialog boxes now appear at the position you specified.

aaHistClientTrend Unsupported Objects

The aaHistClientTrend control contains a few commonly used objects that are not supported.

The following members of the aaTrendControl class are unsupported:

- ArchestrA.HistClient.UI.aaTrend
- ArchestrA.HistClient.UI.aaTrendItemEditor
- Dundas.Charting.WinControl.Chart

aaHistClientTrend Properties

The aaHistClientTrend properties include:

- AddMultipleTags
- AllowContextMenu
- AllowGridEditing
- AlwaysUseFullForXYScatterPlots
- AnalogPlottingAlgorithm
- ApplyRubberBandToAllTags
- AutoRefreshMode
- BackColor
- BackGradient
- BackGradientEndColor
- BackImage
- BorderColor
- BorderStyle
- BorderWidth
- ChartType
- CurrentServerName
- CurrentTagColor
- CurrentTagCycleCount

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- CurrentTagEffectiveRetrievalMode
- CurrentTagFormat
- CurrentTagHistoryVersion
- CurrentTagIndex
- CurrentTagInterpolationType
- CurrentTagName
- CurrentTagNumStyles
- CurrentTagOffsetMS
- CurrentTagPenStyle
- CurrentTagPenWidth
- CurrentTagPrecision
- CurrentTagQualityRule
- CurrentTagResolution
- CurrentTagRetrievalMode
- CurrentTagRetrievalStyle
- CurrentTagRowLimit
- CurrentTagStartDate
- CurrentTagState
- CurrentTagStateCalculation
- CurrentTagTargetRegionVisible
- CurrentTagTimeDeadband
- CurrentTagTimeStampRule
- CurrentTagTrendType
- CurrentTagUseAutoCycles
- CurrentTagUseResolution
- CurrentTagValAtX1
- CurrentTagValAtX2
- CurrentTagValueDeadband
- CurrentTrendItem
- CurrentValOfX1
- CurrentValOfX2
- CurrentValOfY1
- CurrentValOfY2
- CurrentXAxisTagIndex
- CurrentXAxisTagName
- CurrentXAxisTagServerName
- CyclicRows
- DataPointLabelType

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- DateMode
- DatePickerFormatString
- DefaultTagFormat
- DefaultTagPrecision
- Enabled
- EnableDeltaRetrieval
- EnableSummaryData
- EnableTimeOffsets
- EndDate
- FileName
- GridColor
- GridHorizontal
- GridVertical
- GridVisible
- HideCurrentTag
- HighlightCurrentTag
- HistorySource
- LiveModeRate
- LockDown
- LoginTimeout
- MaxDeltaSamples
- MaxMinutesForDeltaAnalog
- MaxMinutesForDeltaDiscrete
- MaxSamplesPerTag
- MovingAverageMode
- MovingAverageSamples
- NumDataPointLabels
- NumTimeAxisGridPerValue
- NumTimeAxisValues
- NumXValueAxisGridLinesPerLabel
- NumXValueAxisLabels
- NumYAxisGridPerValue
- NumYAxisValues
- PanPercentage
- PlaybackSpeed
- PlotColor
- PlotGradient
- PlotGradientEndColor

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- PlotImage
- PrintShowActiveTag
- PrintShowMarkers
- PrintShowTitle
- PrintTitle
- PublicAnnotations
- QueryTimeout
- RealTimeMode
- RealTimeRate
- RetrievalOptionsCycleCount
- RetrievalOptionsHistoryVersion
- RetrievalOptionsInterpolationType
- RetrievalOptionsNumStyles
- RetrievalOptionsQualityRule
- RetrievalOptionsResolution
- RetrievalOptionsRetrievalMode
- RetrievalOptionsRetrievalStyle
- RetrievalOptionsRowLimit
- RetrievalOptionsState
- RetrievalOptionsStateCalculation
- RetrievalOptionsTimeDeadband
- RetrievalOptionsTimeStampRule
- RetrievalOptionsUseAutoCycles
- RetrievalOptionsUseResolution
- RetrievalOptionsValueDeadband
- RetrieveAnnotations
- RetrieveExtensionData
- RetrieveManualData
- RTRate
- Rubberband
- RubberbandAll
- RubberBandScaling
- Servers
- ShowLimits
- ShowValuesAtCursor
- ShowWaitCursor
- ShowXAxisCursors
- ShowYAxisCursor

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- SingleTagMode
- StartDate
- SummaryDataMode
- SupressErrors
- TagGridOrientation
- TagListRows
- TagPicker
- TagPickerVisible
- TargetRegionExcursionType
- TargetRegionOpacity
- TimeAxisLabelColor
- TimeBarVisible
- TimeBarVisible2
- TimeSelector
- ToolBarVisible
- ToolbarVisible2
- ToolTipText
- TraceGradientEndingPercentage
- TraceGradientStartingPercentage
- TraceGradientType
- TrendFontSize
- UpdateToCurrentTimeState
- UseIniFile
- ValueAxisLabel
- Visible
- XCursor1Color
- XCursor1Pos
- XCursor2Color
- XCursor2Pos
- YCursor1Color
- YCursor2Color
- ZoomOutPercentage

AddMultipleTags

The AddMultipleTags property is a read-write property that enables or disables the automatic refresh of the trend chart each time a tag is added.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.AddMultipleTags = discrete;
Result = aaHistClientTrend.AddMultipleTags;
```

Remarks

You can set this property to True and then add multiple tags using a script without refreshing the graph. After adding the final tag, set this property back to False. The graph is automatically refreshed and shows all the tags that you added.

The default value is False.

AllowContextMenu

The AllowContextMenu property is a read-write property that enables or disables the Context menu for the control.

Syntax

```
aaHistClientTrend.AllowContextMenu = discrete;
Result = aaHistClientTrend.AllowContextMenu;
```

Remarks

If this property is set to True, then the context menu appears when the runtime user right-clicks in the control.

The default value is True.

AllowGridEditing

The AllowGridEditing property is a read-write property that enables or disables the editing of the tag list that appears below the trend chart.

Syntax

```
aaHistClientTrend.AllowGridEditing = discrete;
Result = aaHistClientTrend.AllowGridEditing;
```

Remarks

If this property is set to True, then the tag list can be edited.

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AlwaysUseFullForXYScatterPlots

This read-write property determines whether Full or Delta mode retrieval is forced for all tags in a scatter plot regardless of the retrieval settings that are configured at the application or tag level.

Syntax

```
aaHistClientTrend.AlwaysUseFullForXYScatterPlots = discrete;
Result = aaHistClientTrend.AlwaysUseFullForXYScatterPlots;
```

Remarks

If this property is set to True, then full or delta retrieval is forced. Full retrieval is used when retrieving data from an AVEVA Historian with a version of 9.0 or higher. Delta retrieval is used for earlier server versions.

The default value is True. We recommend to keep this option enabled unless the nature of your data makes full retrieval impractical.

AnalogPlottingAlgorithm

The AnalogPlottingAlgorithm property is a read-write property that determines the type of the trend curve for all analog and discrete tags in the trend.

Syntax

```
aaHistClientTrend.AnalogPlottingAlgorithm = integer;
Result = aaHistClientTrend.AnalogPlottingAlgorithm;
```

Remarks

Provided for backward compatibility. Use the CurrentTagTrendType property instead.

Valid values: 0 = Stair-step curve; 1 = Line curve (interpolation).

Return Value

If all analog and discrete tags in the trend use the same curve type, the type is returned; otherwise, a 0 is returned. A return value of 0, therefore, can either mean that all tags use stair-step curves, or that they use different types.

ApplyRubberBandToAllTags

The ApplyRubberBandToAllTags property is a read-write property that indicates whether all tags are scaled by rubber band scaling or just the current tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.ApplyRubberBandToAllTags = discrete;
Result = aaHistClientTrend.ApplyRubberBandToAllTags;
```

Remarks

Provided for backward compatibility. Use the RubberbandAll property instead.

The default value is True.

AutoRefreshMode

The AutoRefreshMode property is a read-write property that gets or sets when the Trend control is refreshed with data from the server.

Syntax

```
aaHistClientTrend.AutoRefreshMode = integer;
Result = aaHistClientTrend.AutoRefreshMode;
```

Remarks

The following table describes the enumerations for this property:

Value	Enumeration	Description
0	Never	No automatic refresh occurs.
1	OnAddTags	Automatic refresh occurs when tags are added.
2	OnTimeSpanChange	Automatic refresh occurs when the time span is changed.
3	OnBoth	Automatic refresh occurs both when tags are added and the time span is changed. This is the default value.

If you are adding multiple tags, set this property to zero to prevent a refresh from occurring as each individual tag is added. When you are finished adding tags, set this property back to a non-zero value.

If the AutoRefreshMode property is zero, you need to call the RefreshData method to refresh the trend.

The default value is 3.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

BackColor

The BackColor property is a read-write property that gets or sets the background color of the chart.

Syntax

```
aaHistClientTrend.BackColor = integer;
Result = aaHistClientTrend.BackColor;
```

Tables

For information on setting the color value, see Color.

The default value is 16777215 (white).

BackGradientEndColor

The BackGradientEndColor property is a read-write property that gets or sets the background gradient end color of the chart.

Syntax

```
aaHistClientTrend.BackGradientEndColor = integer;
Result = aaHistClientTrend.BackGradientEndColor;
```

Remarks

The gradient starts with the main background color and fades to the gradient end color. Use the BackColor property to set the main background color. Use the BackGradient property to set the type of gradient fill.

For information on setting the color value, see Color.

The default value is 16777215 (white).

BackGradient

The BackGradient property is a read-write property that gets or sets the type of background gradient for the chart.

Syntax

```
aaHistClientTrend.BackGradient = integer;
Result = aaHistClientTrend.BackGradient;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The gradient starts with the main background color and fade to the gradient end color. Use the BackColor property to set the main background color. Use the BackGradientEndColor property to set the ending gradient color.

For more information on the values for the back gradient, see the aaTrendGradientType Enumeration.

The default value is 0.

BackImage

The BackImage property is a read-write property that gets or sets the background image for the chart.

Syntax

```
aaHistClientTrend.BackImage = message;
Result = aaHistClientTrend.BackImage;
```

Remarks

The value of this property is the folder path and filename for the image. Supported image types are .jpeg, .gif, .bmp, and .png.

This property has no default value.

BorderColor

The BorderColor property is a read-write property that gets or sets the color for the border of the graph.

Syntax

```
aaHistClientTrend.BorderColor = integer;
Result = aaHistClientTrend.BorderColor;
```

Remarks

For information on setting the color value, see Color.

The default value is 0 (black).

BorderStyle

The BorderStyle property is a read-write property that gets or sets the style of the border line around the trend chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.BorderStyle = aaDashStyle;
Result = aaHistClientTrend.BorderStyle;
```

Remarks

For more information on the values for the border style, see aaDashStyle Enumeration.

The default value is 0, which indicates a solid line.

BorderWidth

The BorderWidth property is a read-write property that gets or sets the width, in pixels, of the border line around the trend chart.

Syntax

```
aaHistClientTrend.BorderWidth = integer;
Result = aaHistClientTrend.BorderWidth;
```

Remarks

Valid values are 0 through 10. The default value is 1.

ChartType

The ChartType is a read-write property that determines the chart type of the current Trend file.

Syntax

```
aaHistClientTrend.ChartType = aaChartType;
Result = aaHistClientTrend.ChartType;
```

Remarks

For information on possible values, see aaChartType Enumeration.

The default value is 0 (regular trend).

CurrentServerName

The CurrentServerName property is a read-only property that gets the name of the server for the tag that is currently selected.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaHistClientTrend.CurrentServerName;
```

Remarks

This property is not visible at design time. This property has no default value.

Return Value

The result is a string value.

CurrentTagColor

The CurrentTagColor property is a read-write property that determines the line color of the currently selected tag's curve in the trend.

Syntax

```
aaHistClientTrend.CurrentTagColor = integer;
Result = aaHistClientTrend.CurrentTagColor;
```

Remarks

This property is not visible at design time.

For information on setting the color value, see Color.

The default value is 0 (black).

CurrentTagCycleCount

This read-write property controls the current tag's number of cycles for cycle-based data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagCycleCount = integer;
Result = aaHistClientTrend.CurrentTagCycleCount;
```

Remarks

This property is only taken into account if both the CurrentTagUseAutoCycles property and the CurrentTagUseResolution property are set to False. Also, it may be overridden by a retrieval style setting. For

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

more information, see Working with Retrieval Styles.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

Valid values: any positive integer or 0. If you specify 0, the cycle count is calculated automatically, just as if the CurrentTagUseAutoCycles property were set to True. The default value is 0.

CurrentTagEffectiveRetrievalMode

This read-only property returns the retrieval mode that is used for the currently selected tag. This helps you to determine the tag's actual retrieval mode when you are using a retrieval style that specifies different retrieval modes depending on tag type or duration.

Syntax

```
Result = aaHistClientTrend.CurrentTagEffectiveRetrievalMode;
```

Remarks

The return value is an integer. For an explanation of each return value, see aaRetrievalMode Enumeration.

CurrentTagFormat

The CurrentTagFormat property is a read-write property that is used to control how the values for the selected tag appear, either in decimal format or scientific format.

Syntax

```
aaHistClientTrend.CurrentTagFormat = integer;
Result = aaHistClientTrend.CurrentTagFormat;
```

Remarks

0 = Decimal; 1 = Scientific. If you use the decimal format, set the number of decimal places using the CurrentTagPrecision property.

The default value is 0.

CurrentTagHistoryVersion

This read-write property determines the current tag's history source for data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.CurrentTagHistoryVersion = aaRetrievalVersion;
Result = aaHistClientTrend.CurrentTagHistoryVersion;
```

Remarks

For information on possible values, see aaRetrievalVersion Enumeration. This property is relevant for all retrieval modes.

The default value is 0 (latest values).

CurrentTagIndex

This read-only property returns the index of the tag that is currently selected.

Syntax

```
Result = aaHistClientTrend.CurrentTagIndex;
```

Return Value

The result is an integer value.

Remarks

The index reflects the order in which the tags were added to the trend. 0 denotes the first tag that was added to the trend, 1 denotes the second, and so on. If no tag is currently selected, -1 is returned.

CurrentTagInterpolationType

This read-write property determines the current tag's interpolation type for data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagInterpolationType = aaInterpolationType;
Result = aaHistClientTrend.CurrentTagInterpolationType;
```

Remarks

For information on possible values, see aaInterpolationType Enumeration. This property is only relevant for the following retrieval modes: Interpolated, Best Fit, Average, and Integral.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The default value is 3 (use the default value of the server).

CurrentTagName

The CurrentTagName property is a read-only property that gets the name of the tag that is currently selected.

Syntax

```
Result = aaHistClientTrend.CurrentTagName;
```

Return Value

The result is a message.

Remarks

This property is not visible at design time. This property has no default value.

CurrentTagNumStyles

This read-only property returns the number of retrieval styles that are available for the current tag.

Syntax

```
Result = aaHistClientTrend.CurrentTagNumStyles;
```

Remarks

The count only includes retrieval styles for which a name is defined for the current locale. If no style names at all are defined for the current locale, the count for the en locale is returned.

To return the name of a style with a specific number, use the CurrentTagGetStyle method.

CurrentTagOffsetMS

The CurrentTagOffsetMS property is a read-write property that gets or sets the amount of time that the trend curve of the currently selected tag will be shifted from the actual time.

Syntax

```
aaHistClientTrend.CurrentTagOffsetMS = integer;
Result = aaHistClientTrend.CurrentTagOffsetMS;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The offset, expressed in milliseconds, can be positive or negative. For more information, see Using Time Offsets to Compare Data on page 78. Setting this property updates the CurrentTagStartDate property accordingly.

Due to the limited range for integer values, the maximum offset you can set using this property is about 29 days. For larger offsets, use the CurrentTagStartDate property.

The default value is 0. This property is only relevant if the DateMode property is set to absolute mode.

CurrentTagPenStyle

The CurrentTagPenStyle property is a read-write property that gets or sets the style of the trend curve for the currently selected tag. For example, a solid or dashed line.

Syntax

```
aaHistClientTrend.CurrentTagPenStyle = integer;
Result = aaHistClientTrend.CurrentTagPenStyle;
```

Remarks

Valid values are:

Value	Style
0	Solid
1	Dashed
2	Dotted
3	DashDot
4	DashDotDot

The default value is 0.

CurrentTagPenWidth

The CurrentTagPenWidth property is a read-write property that gets or sets the thickness of the trend curve for the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTagPenWidth = integer;
Result = aaHistClientTrend.CurrentTagPenWidth;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

Valid values are 0 through 10. The default value is 0.

CurrentTagPrecision

The CurrentTagPrecision property is a read-write property that gets or sets the number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.

Syntax

```
aaHistClientTrend.CurrentTagPrecision = integer;
Result = aaHistClientTrend.CurrentTagPrecision;
```

Remarks

To set the tag to use the decimal format, use the DefaultTagFormat property.

The default value is 0.

CurrentTagQualityRule

This read-write property determines the current tag's quality rule for data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagQualityRule = aaQualityRules;
Result = aaHistClientTrend.CurrentTagQualityRule;
```

Remarks

For information on possible values, see aaQualityRules Enumeration. This property is relevant for all retrieval modes except the following: Cyclic, Delta, and Full.

The default value is 3 (use the default value of the server).

CurrentTagResolution

This read-write property controls the current tag's time interval for calculating the number of cycles in cycle-based data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.CurrentTagResolution = integer;
Result = aaHistClientTrend.CurrentTagResolution;
```

Remarks

This property is only relevant if the CurrentTagUseAutoCycles property is set to False, and the CurrentTagUseResolution property is set to True. Also, it may be overridden by a retrieval style setting. For more information, see Working with Retrieval Styles.

The value of this property is a time interval in milliseconds. The aaHistClientTrend control divides the query duration by this interval and uses the result as the number of cycles for the query.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

Valid values: any positive integer or 0. If you specify 0, the cycle count is calculated automatically, just as if the CurrentTagUseAutoCycles property were set to True. The default value is 0.

CurrentTagRetrievalMode

This read-write property determines the current tag's data retrieval mode. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagRetrievalMode = aaRetrievalMode;
Result = aaHistClientTrend.CurrentTagRetrievalMode;
```

Remarks

This property may be overridden by a retrieval style setting. For more information, see Working with Retrieval Styles. For information on possible values, see aaRetrievalMode Enumeration.

The default value is 0 (cyclic). Make sure that the specified retrieval mode is supported by the AVEVA Historian on which the tag is stored.

CurrentTagRetrievalStyle

This read-write property determines the current tag's retrieval style. This setting overrides the default setting specified at the application level.

Syntax

```
aaHistClientTrend.CurrentTagRetrievalStyle = string;
Result = aaHistClientTrend.CurrentTagRetrievalStyle;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

You must provide the retrieval style name for the current locale as it is defined in the retrieval style document. For more information, see Location and Structure of Retrieval Styles. To find out how many retrieval styles are available for the current tag, use the CurrentTagNumStyles property. To determine the name of a retrieval style if you know its position in the list of available styles, use the CurrentTagGetStyle method.

Valid values: `Custom style` (or the translated equivalent for the current locale), `Style selected at option level` (ditto) and any retrieval style name that is defined for the current locale in the retrieval style document. Values are case-sensitive. If no style names at all are available for the current locale, use the name for the en locale. The default style is the `Style selected at option level` (or the translated equivalent).

CurrentTagRowLimit

This read-write property determines the current tag's row limit for data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagRowLimit = integer;
Result = aaHistClientTrend.CurrentTagRowLimit;
```

Remarks

The row limit applies to each query. For more information, see RowLimit. This property is relevant for all retrieval modes.

Valid values: any positive number or 0 (no row limit). The default value is 0.

CurrentTagStartDate

The CurrentTagStartDate property is a read-write property that gets or sets the trend start date for the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTagStartDate = DateTime;
Result = aaHistClientTrend.CurrentTagStartDate;
```

Return Value

The result is a DateTime value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property has no default. Setting this property updates the CurrentTagOffsetMS property accordingly.

This property is only applicable if the DateMode property is set to relative. It reflects local time.

For information on setting the date/time value, see DateTime.

CurrentTagState

This read-write property determines the state for which Time-in-State and RoundTrip data is retrieved for the current tag. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagState = message;
Result = aaHistClientTrend.CurrentTagState;
```

Remarks

This property is only relevant for the Time-in-State and RoundTrip retrieval modes. It specifies the unique tag state for which Time-in-State and RoundTrip information is calculated based on the calculation type specified by the CurrentTagStateCalculation property.

This property has no default value.

CurrentTagStateCalculation

This read-write property determines the current tag's calculation type for the Time-in-State and RoundTrip data retrievals. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagStateCalculation = aaStateCalculation;
Result = aaHistClientTrend.CurrentTagStateCalculation;
```

Remarks

For information on possible values, see aaStateCalculation Enumeration. This property is only relevant for the Time-in-State and RoundTrip retrieval modes. Also, it may be overridden by a retrieval style setting. For more information, see Working with Retrieval Styles.

The default value is 5 (use application setting).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentTagTargetRegionVisible

This read-write property determines whether a currently selected tag's target region is shown on the chart.

Syntax

```
aaHistClientTrend.CurrentTagTargetRegionVisible = discrete;
Result = aaHistClientTrend.CurrentTagTargetRegionVisible;
```

Remarks

If no target region is defined for a tag, this property has no effect. Regardless of the value of this property, the target region for a tag is only shown when that tag is currently selected in the tag list.

The default value is True.

CurrentTagTimeDeadband

This read-write property determines the current tag's time deadband in milliseconds for Delta data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagTimeDeadband = integer;
Result = aaHistClientTrend.CurrentTagTimeDeadband;
```

Remarks

Valid values: any positive number or 0 (no deadband). This property is only relevant for Delta retrieval mode. For more information on how this setting works, see Time Deadband (wwTimeDeadband).

The default value is 0 (no deadband).

CurrentTagTimeStampRule

This read-write property determines the current tag's timestamp rule for data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagTimeStampRule = aaTimeStampRules;
Result = aaHistClientTrend.CurrentTagTimeStampRule;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

For information on possible values, see aaTimeStampRules Enumeration. This property is only relevant for the following retrieval modes: Cyclic, Interpolated, Time-Weighted Average, Integral, Counter, and Time-in-State.

The default value is 3 (use the default value of the server).

CurrentTagTrendType

This read-write property determines the type of the current tag's trend curve.

Syntax

```
aaHistClientTrend.CurrentTagTrendType = aaTrendType;
Result = aaHistClientTrend.CurrentTagTrendType;
```

Remarks

For information on possible values, see aaTrendType Enumeration.

The default value is 3 (Auto).

CurrentTagUseAutoCycles

This read-write property controls the current tag's auto-calculation setting for cycle-based data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagUseAutoCycles = discrete;
Result = aaHistClientTrend.CurrentTagUseAutoCycles;
```

Remarks

If this property is set to True, the aaHistClientTrend control automatically calculates the number of cycles for a query based on the width of the chart. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).

If it is set to False, you must specify the number of cycles manually. Use the CurrentTagUseResolution property to specify whether you want to provide a number of cycles or a time interval. Then use the CurrentTagCycleCount property to specify the number of cycles, or the CurrentTagResolution property to specify the time interval.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

The default value is False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentTagUseResolution

This read-write property controls the current tag's behavior for determining the number of cycles in cycle-based data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagUseResolution = discrete;
Result = aaHistClientTrend.CurrentTagUseResolution;
```

Remarks

This property is only relevant if the CurrentTagUseAutoCycles property is set to False.

If this property is set to False, the aaHistClientTrend control uses a fixed number of cycles when retrieving data using cycle-based retrieval modes. To specify the number of cycles, use the CurrentTagCycleCount property.

If it is set to True, the aaHistClientTrend control calculates the number of cycles based on the query duration and a time interval. To specify this interval, use the CurrentTagResolution property.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

The default value is False.

CurrentTagValAtX1

The CurrentTagValAtX1 property is a read-only property that gets the value of the current tag at the point at which the curve intersects with the first time axis cursor.

Syntax

```
Result = aaHistClientTrend.CurrentTagValAtX1;
```

Return Value

The result is a real value.

Remarks

For more information on cursors, see Using Axis Cursors.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentTagValAtX2

The CurrentTagValAtX2 property is a read-only property that gets the value of the current tag at the point at which the curve intersects with the second time axis cursor.

Syntax

```
Result = aaHistClientTrend.CurrentTagValAtX2;
```

Return Value

The result is a real value.

Remarks

For more information on cursors, see Using Axis Cursors.

This property has no default value.

CurrentTagValueDeadband

This read-write property determines the current tag's value deadband for Delta data retrieval. This setting overrides the default setting specified at the application level if the CurrentTagRetrievalStyle property is set to an option other than `Style selected at option level`.

Syntax

```
aaHistClientTrend.CurrentTagValueDeadband = real;
Result = aaHistClientTrend.CurrentTagValueDeadband;
```

Remarks

The deadband is a percentage of the full scale in Engineering Units. Valid values are 0 (no deadband) to 100. This property is only relevant for Delta retrieval mode. For more information on how this setting works, see Value Deadband (wwValueDeadband).

The default value is 0 (no deadband).

CurrentTrendItem

The CurrentTrendItem property is a read-only property that refers to the currently selected trend item in the Tag List.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaHistClientTrend.CurrentTrendItem;
```

Remarks

If no items are added or selected in the Tag List, this property contains a null value.

The CurrentTrendItem property supports the following properties:

- Visible
- PenWidth
- Style
- ValueFormat
- DecimalPlaces
- BottomY
- TopY
- TrendType
- Name

Visible

Visible

The Visible property is a read-write property that gets or sets the visibility of the current trend item.

This property has no default value.

Syntax

```
aaHistClientTrend.CurrentTrendItem.Visible = true;
Result = aaHistClientTrend.CurrentTrendItem.Visible;
```

PenWidth

PenWidth

The PenWidth property is a read-write property that gets or sets the thickness of the trend curve for the currently selected tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.CurrentTrendItem.PenWidth = integer;
Result = aaHistClientTrend.CurrentTrendItem.PenWidth;
```

Remarks

Valid values are 0 through 10. The default value is 0.

Style

Style

The Style property is a read-write property that gets or sets the style of the trend curve for the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTrendItem.Style = integer;
Result = aaHistClientTrend.CurrentTrendItem.Style;
```

Remarks

The valid values and curve styles are as follows:

Value	Style
0	Solid
1	Dashed
2	Dotted
3	DashDot
4	DashDotDot

The default value is 0.

ValueFormat

ValueFormat

The ValueFormat property is a read-write property that gets or sets the value format of the currently selected

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tag. The format can be decimal or scientific.

Syntax

```
aaHistClientTrend.CurrentTrendItem.ValueFormat = integer;
Result = aaHistClientTrend.CurrentTrendItem.ValueFormat;
```

Remarks

The value 0 specifies decimal format and 1 specifies scientific format. If you use the decimal format, then set the number of decimal places using the DecimalPlaces property.

The default value is 0.

DecimalPlaces

DecimalPlaces

The DecimalPlaces property is a read-write property that gets or sets the number of decimal places to display for the data value of the currently selected tag. This property is applicable only to the analog tags.

Syntax

```
aaHistClientTrend.CurrentTrendItem.DecimalPlaces = integer;
Result = aaHistClientTrend.CurrentTrendItem.DecimalPlaces;
```

Remarks

The default value is 0.

BottomY

BottomY

The BottomY property is a read-write property that gets or sets the lower value for the y-axis of the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTrendItem.BottomY = double;
Result = aaHistClientTrend.CurrentTrendItem.BottomY;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TopY

TopY

The TopY property is a read-write property that gets or sets the upper value for the y-axis of the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTrendItem.TopY = double;
Result = aaHistClientTrend.CurrentTrendItem.TopY;
```

TrendType

TrendType

The TrendType property is a read-write property that gets or sets the type of the trend curve of the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTrendItem.TrendType = integer;
```

Remarks

For information on possible values, see aaTrendType Enumeration.

The default value is 3.

Name

Name

The Name property is a read-only property that gets the name of the currently selected tag.

Syntax

```
Result = aaHistClientTrend.CurrentTrendItem.Name;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

The result is a string value.

Remarks

This property is not visible at design time. This property has no default value.

CurrentValOfX1

This read-write property gets or sets the position of the first x-axis cursor of the currently selected tag in a scatter plot.

Syntax

```
aaHistClientTrend.CurrentValOfX1 = real;
Result = aaHistClientTrend.CurrentValOfX1;
```

Remarks

This property contains the value at which the cursor intersects with the current x-axis scale. Therefore, the same cursor position may be reflected by different values depending on which tag is selected.

To control the position of the first time axis cursor in a regular trend, use the XCursor1Pos property.

CurrentValOfX2

This read-write property gets or sets the position of the second x-axis cursor of the currently selected tag in a scatter plot.

Syntax

```
aaHistClientTrend.CurrentValOfX2 = real;
Result = aaHistClientTrend.CurrentValOfX2;
```

Remarks

This property contains the value at which the cursor intersects with the current x-axis scale. Therefore, the same cursor position may be reflected by different values depending on which tag is selected.

To control the position of the second time axis cursor in a regular trend, use the XCursor2Pos property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentValOfY1

This read-write property controls the position of the first y-axis cursor of the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentValOfY1 = real;
Result = aaHistClientTrend.CurrentValOfY1;
```

Remarks

This property contains the value at which the cursor intersects with the current y-axis scale. Therefore, the same cursor position may be reflected by different values depending on which tag is selected.

CurrentValOfY2

This read-write property controls the position of the second y-axis cursor of the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentValOfY2 = real;
Result = aaHistClientTrend.CurrentValOfY2;
```

Remarks

This property contains the value at which the cursor intersects with the current y-axis scale. Therefore, the same cursor position may be reflected by different values depending on which tag is selected.

CurrentXAxisTagIndex

This read-only property returns the index of the x-axis tag that is associated with the currently selected tag.

Syntax

```
Result = aaHistClientTrend.CurrentXAxisTagIndex;
```

Return Value

The result is an integer value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The index reflects the order in which the tags were added to the trend. 0 denotes the first tag that was added to the trend, 1 denotes the second, and so on. If no tag is currently selected, or if the currently selected tag isn't associated with an x-axis tag, -1 is returned.

CurrentXAxisTagName

This read-only property returns the name of the x-axis tag that is associated with the currently selected tag.

Syntax

```
Result = aaHistClientTrend.CurrentXAxisTagName;
```

Return Value

The result is a message value.

Remarks

If no x-axis tag is set for the currently selected tag, this property contains an empty string.

CurrentXAxisTagServerName

This read-only property returns the name of the server for the x-axis tag that is associated with the currently selected tag.

Syntax

```
Result = aaHistClientTrend.CurrentXAxisTagServerName;
```

Return Value

The result is a message value.

Remarks

If no x-axis tag is set for the currently selected tag, this property contains an empty string.

CyclicRows

This property is deprecated and included for backward compatibility only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.CyclicRows = integer;
Result = aaHistClientTrend.CyclicRows;
```

Remarks

To specify the number of cycles for cyclic retrieval, use the CurrentTagCycleCount or RetrievalOptionsCycleCount properties instead.

DataPointLabelType

This property determines whether data point labels are shown in a scatter plot.

Syntax

```
aaHistClientTrend.DataPointLabelType = aaDataPointLabelingType;
Result = aaHistClientTrend.DataPointLabelType;
```

Remarks

For information on possible values, see aaDataPointLabelingType Enumeration.

The default value is 0 (no labels).

DateMode

The DateMode property is a read-write property that gets or sets the date mode for the trend.

Syntax

```
aaHistClientTrend.DateMode = aaDateModeEnumeration;
Result = aaHistClientTrend.DateMode;
```

Remarks

The default value is 0 (absolute mode).

For more information on the aaDateModeEnumeration enumeration, see aaDateModeEnumeration Enumeration.

The DateMode property determines the functionality of the Time Bar and how time shifting is anchored as you switch between different time periods.

- In absolute mode, the Time Bar has a start time and an end time. In this mode, each tag has its own time offset. The actual time period used for queries is the sum of the tag's "offset" and the start and end time for the Time Bar. The tag offset is set using the CurrentTagOffsetMS property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- In relative mode, the Time Bar has a starting time offset and an ending time offset. In this mode each tag has its own starting time. The actual time period used for queries is the sum of the tag's start time to the offsets of the Time Bar. If you set the DateMode property to use relative time, specify the start time for the current tag using the CurrentTagStartDate property.

In both modes, zoom and pan operations only manipulate the Time Bar properties, not the tag properties.

DatePickerFormatString

The DatePickerFormatString property is a read-write property that gets or sets the format string for the time range picker.

Syntax

```
aaHistClientTrend.DatePickerFormatString = message;
Result = aaHistClientTrend.DatePickerFormatString;
```

Remarks

This value is determined from the regional settings for the operating system.

For example, the time setting might be:

hh:mm:ss tt

where:

hh = hour, with a leading zero

mm = minute, with a leading zero

ss = second, with a leading zero

tt = AM or PM

For more information, see the regional settings options in Control Panel.

This property only changes the format for the time range picker; it does not change the system-wide value.

The default format is M/d/yyyy h:mm:ss tt.

DefaultTagFormat

The DefaultTagFormat property is a read-write property that gets or sets the format of the trend item for presentation to the client.

Syntax

```
aaHistClientTrend.DefaultTagFormat = integer;
Result = aaHistClientTrend.DefaultTagFormat;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

Valid values are: 0 = Decimal, 1 = Scientific. The default value is 0. If you use the decimal format, use the DefaultTagPrecision property to specify the number of decimal points. Format changes are not applied to trend items already in the chart at the time the format change is made.

The default value is 0.

DefaultTagPrecision

The DefaultTagPrecision property is read-write property that gets or sets the number of decimal places of the trend item for presentation to the client.

Syntax

```
aaHistClientTrend.DefaultTagPrecision = integer;
Result = aaHistClientTrend.DefaultTagPrecision;
```

Remarks

Precision changes are not applied to trend items already in the chart at the time the precision change is made.

The default value is 0.

EnableDeltaRetrieval

The EnableDeltaRetrieval property is a read-write property that enables or disables delta retrieval for the trend control.

Syntax

```
aaHistClientTrend.EnableDeltaRetrieval = discrete;
Result = aaHistClientTrend.EnableDeltaRetrieval;
```

Remarks

The aaHistClientTrend control only takes this property into account when retrieving data from the AVEVA Historians with a version earlier than 9.0. For more information, see Retrieval Styles, Application Settings, and Tag Settings.

Delta retrieval is used for analog and discrete queries that have a time range that are within the settings of the MaxMinutesForDeltaAnalog and MaxMinutesForDeltaDiscrete properties.

Delta retrieval is always used for the "live" retrieval mode. If you set this property to False, this has no effect on live mode.

The default value is False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

EnableSummaryData

This property is included for backward compatibility only. Its value is ignored.

Syntax

```
aaHistClientTrend.EnableSummaryData = discrete;
Result = aaHistClientTrend.EnableSummaryData;
```

Remarks

To retrieve summarized data, use a retrieval style instead. For more information, see Working with Retrieval Styles.

EnableTimeOffsets

Note: This property is included for backward compatibility only. Setting this property has no effect.

The EnableTimeOffsets property is a read-write property that enables or disables time offsets for the trend items.

Syntax

```
aaHistClientTrend.EnableTimeOffsets = discrete;
Result = aaHistClientTrend.EnableTimeOffsets;
```

Remarks

The default value is True.

EndDate

The EndDate property is a read-only property that returns the end date and time of the time range.

Syntax

```
Result = aaHistClientTimeRangePicker.EndDate;
```

Return Value

A message value in a valid date/time format is returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

FileName

The FileName property is a read-only property that gets the name of the current trend file.

Syntax

```
Result = aaHistClientTrend.FileName;
```

Return Value

The result is a message.

Remarks

The default value is an empty message value ("").

GridColor

The GridColor property is a read-write property that gets or sets the color of the trend grid.

Syntax

```
aaHistClientTrend.GridColor = integer;
Result = aaHistClientTrend.GridColor;
```

Remarks

For information on setting the color value, see Color.

The default value is 13882323.

GridHorizontal

The GridHorizontal property is a read-write property that shows or hides the horizontal grid.

Syntax

```
aaHistClientTrend.GridHorizontal = discrete;
Result = aaHistClientTrend.GridHorizontal;
```

Remarks

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GridVertical

The GridVertical property is a read-write property that shows or hides the vertical grid.

Syntax

```
aaHistClientTrend.GridVertical = discrete;
Result = aaHistClientTrend.GridVertical;
```

Remarks

The default value is True.

GridVisible

The GridVisible property is a read-write property that shows or hides the tag list underneath the chart area.

Syntax

```
aaHistClientTrend.GridVisible = discrete;
Result = aaHistClientTrend.GridVisible;
```

Remarks

The default value is True.

HideCurrentTag

The HideCurrentTag property is a read-write property that shows or hides the currently selected trend item (tag).

Syntax

```
aaHistClientTrend.HideCurrentTag = discrete;
Result = aaHistClientTrend.HideCurrentTag;
```

Remarks

The default value is False. If there are no tags on the chart, this property returns True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

HighlightCurrentTag

The HighlightCurrentTag property is a read-write property that controls whether to highlight whichever tag is currently selected.

Syntax

```
aaHistClientTrend.HighlightCurrentTag = discrete;
Result = aaHistClientTrend.HighlightCurrentTag;
```

Remarks

This property is a trend-level setting, not a tag-level setting. If you enable it while a particular tag is selected, that tag is highlighted. Once you select a different tag, that other tag is highlighted, and so on. The default value is False.

HistorySource

The HistorySource property is a read-write property that gets or sets the selection of the source of historical data.

Syntax

```
aaHistClientTrend.HistorySource = aaRetrievalSource;
Result = aaHistClientTrend.HistorySource;
```

Remarks

For more information on the aaRetrievalSource enumeration, see aaRetrievalSource Enumeration.

Remarks

The default value is 2.

LiveModeRate

The LiveModeRate property is a read-write property that gets or sets the refresh interval in milliseconds for live and replay mode.

Syntax

```
aaHistClientTrend.LiveModeRate = integer;
Result = aaHistClientTrend.LiveModeRate;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The lower limit for the LiveModeRate property is set to 250 milliseconds. The default value is 1,000.

Apart from the different unit of measure, this property serves the same purpose as the RealTimeRate property.

LoginTimeout

The LoginTime property is a read-write property that gets or sets the amount of time, in seconds, that the control waits for a connection to the server to be established before determining that the attempt failed.

Syntax

```
aaHistClientTrend.LoginTimeout = integer;
Result = aaHistClientTrend.LoginTimeout;
```

Remarks

This setting only applies to servers that you add or update dynamically using the AddServer method. All other servers continue to use the timeout that you set in the server configuration dialog box.

Remarks

The default value is 120.

MaxDeltaSamples

The MaxDeltaSamples property is a read-write property that gets or sets the maximum number of data values to retrieve for delta retrieval mode.

Syntax

```
aaHistClientTrend.MaxDeltaSamples = integer;
Result = aaHistClientTrend.MaxDeltaSamples;
```

Remarks

The aaHistClientTrend control only takes this property into account when retrieving data from AVEVA Historians with a version earlier than 9.0.

Valid values are 0 to 100,000. The default value is 10,000.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LockDown

This read-write property enables or disables a “lock down” mode in the control.

Syntax

```
aaHistClientTrend.LockDown = discrete;
Result = aaHistClientTrend.LockDown;
```

Remarks

In "lock down" mode, the following features are not available to the run-time user:

- Opening a file, saving a file, saving a file as a different name, and creating a new file
- Deleting a tag
- Adding an annotation
- Viewing or changing properties and options
- Configuring servers
- Viewing the Tag Picker and the main toolbar
- Editing the tag list (grid)

The default value is False.

MaxMinutesForDeltaAnalog

The MaxMinutesForDeltaAnalog property is a read-write property that gets or sets the maximum minutes filter for analog tags for delta retrieval mode.

Syntax

```
aaHistClientTrend.MaxMinutesForDeltaAnalog = integer;
Result = aaHistClientTrend.MaxMinutesForDeltaAnalog;
```

Remarks

The aaHistClientTrend control only takes this property into account when retrieving data from AVEVA Historians with a version earlier than 9.0.

Delta retrieval is used for analog queries that have a time range that are within the setting of the MaxMinutesForDeltaAnalog property. If the query time range is longer, cyclic retrieval is used.

Remarks

The default value is 15.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MaxMinutesForDeltaDiscrete

The MaxMinutesForDeltaDiscrete property is a read-write property that gets or sets the maximum minutes filter for discrete tags for delta retrieval mode.

Syntax

```
aaHistClientTrend.MaxMinutesForDeltaDiscrete = integer;
Result = aaHistClientTrend.MaxMinutesForDeltaDiscrete;
```

Remarks

The aaHistClientTrend control only takes this property into account when retrieving data from an AVEVA Historian with a version earlier than 9.0.

Delta retrieval is used for discrete queries that have a time range that are within the setting of the MaxMinutesForDeltaDiscrete property. If the query time range is longer, cyclic retrieval is used.

The default value is 15.

MaxSamplesPerTag

The MaxSamplesPerTag property is a read-write property that gets or sets the maximum number of samples per tag.

Syntax

```
aaHistClientTrend.MaxSamplesPerTag = integer;
Result = aaHistClientTrend.MaxSamplesPerTag;
```

Remarks

The aaHistClientTrend control only takes this property into account when retrieving data from an AVEVA Historian with a version earlier than 9.0.

The default value is 10,000.

MovingAverageMode

This property is included for backward compatibility only. Its value is ignored.

Syntax

```
aaHistClientTrend.MovingAverageMode = discrete;
Result = aaHistClientTrend.MovingAverageMode;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

To calculate moving averages, use a retrieval style instead. For more information, see Working with Retrieval Styles.

MovingAverageSamples

This property is included for backward compatibility only. Its value is ignored.

Syntax

```
aaHistClientTrend.MovingAverageSamples = integer;
Result = aaHistClientTrend.MovingAverageSamples;
```

Remarks

To calculate moving averages, use a retrieval style instead. For more information, see Working with Retrieval Styles.

NumDataPointLabels

This read-write property determines the number of data point labels in a scatter plot.

Syntax

```
aaHistClientTrend.NumDataPointLabels = integer;
Result = aaHistClientTrend.NumDataPointLabels;
```

Remarks

The valid range is from 2 to 15. The default value is 6. This property is only used if data points are actually shown on the scatter plot. For more information, see DataPointLabelType.

NumTimeAxisGridPerValue

The NumTimeAxisGridPerValue property is a read-write property that gets or sets the number of grid lines that appear between each tag value plotted on the graph.

Syntax

```
aaHistClientTrend.NumTimeAxisGridPerValue = integer;
Result = aaHistClientTrend.NumTimeAxisGridPerValue;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The valid range is from 1 to 20. The default value is 3.

NumTimeAxisValues

The NumTimeAxisValues property is a read-write property that gets or sets the number of values that are shown along the time axis.

Syntax

```
aaHistClientTrend.NumTimeAxisValues = integer;
Result = aaHistClientTrend.NumTimeAxisValues;
```

Remarks

The values are shown at evenly-spaced points along the axis. The number of values remain the same even if you zoom in and out. The valid range is from 2 to 15. The default value is 6.

NumXValueAxisGridLinesPerLabel

This read-write property determines the number of grid lines that appear between each scale value label on the X axis of a scatter plot.

Syntax

```
aaHistClientTrend.NumXValueAxisGridLinesPerLabel = integer;
Result = aaHistClientTrend.NumXValueAxisGridLinesPerLabel;
```

Remarks

The valid range is from 1 to 20. The default value is 3.

NumXValueAxisLabels

This read-write property determines the number of scale value labels that appear on the X axis of a scatter plot.

Syntax

```
aaHistClientTrend.NumXValueAxisLabels = integer;
Result = aaHistClientTrend.NumXValueAxisLabels;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The valid range is from 2 to 15. The default value is 6.

NumYAxisGridPerValue

This read-write property determines the number of grid lines that appear between each scale value label on the Y axis of a chart.

Syntax

```
aaHistClientTrend.NumYAxisGridPerValue = integer;
Result = aaHistClientTrend.NumYAxisGridPerValue;
```

Remarks

The valid range is from 1 to 20. The default value is 2.

NumYAxisValues

This read-write property determines the number of scale value labels that appear on the Y axis of a chart.

Syntax

```
aaHistClientTrend.NumYAxisValues = integer;
Result = aaHistClientTrend.NumYAxisValues;
```

Remarks

The values are shown at evenly-spaced points along the axis. The number of values remains the same even if you zoom in and out. The valid range is from 2 to 15. The default value is 6.

PanPercentage

The PanPercentage property is a read-write property that gets or sets the percentage (1 to 100) by which the time axis (x-axis) pans.

Syntax

```
aaHistClientTrend.PanPercentage = integer;
Result = aaHistClientTrend.PanPercentage;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The default value is 75.

PlaybackSpeed

This read-write property determines the playback speed in "replay" mode.

Syntax

```
aaHistClientTrend.PlaybackSpeed = real;
Result = aaHistClientTrend.PlaybackSpeed;
```

Remarks

For information on replay mode, see Showing Historical Data in "Replay" Mode.

Valid values are 0.5 to 128. The default value is 1 (normal speed).

PlotColor

The PlotColor property is a read-write property that gets or sets the color for the plot area of the graph.

Syntax

```
aaHistClientTrend.PlotColor = integer;
Result = aaHistClientTrend.PlotColor;
```

Remarks

For information on setting the color value, see Color.

The default value is 16777215.

PlotGradient

The PlotGradient property is a read-write property that gets or sets the type of plot gradient for the chart.

Syntax

```
aaHistClientTrend.PlotGradient = aaTrendGradientType;
Result = aaHistClientTrend.PlotGradient;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The gradient starts with the main plot color and fades to the gradient end color. Use the PlotColor property to set the main background color. Use the PlotGradientEndColor property to set the ending gradient color.

For more information on the aaTrendGradientType enumeration, see aaTrendGradientType Enumeration.

The default value is 0.

PlotGradientEndColor

The PlotGradientEndColor property is a read-write property that gets or sets the gradient end color for the plot area of the chart.

Syntax

```
aaHistClientTrend.PlotGradientEndColor = integer;
Result = aaHistClientTrend.PlotGradientEndColor;
```

Remarks

The gradient starts with the main plot color and fades to the gradient end color. Use the PlotColor property to set the main plot color. Use the PlotGradient property to set the type of gradient fill.

For information on setting the color value, see Color.

The default value is 16777215.

PlotImage

The PlotImage property is a read-write property that gets or sets the plot image for the chart.

Syntax

```
aaHistClientTrend.PlotImage = message;
Result = aaHistClientTrend.PlotImage;
```

Remarks

The value of this property is the folder path and filename for the image. Supported image types are .jpeg, .gif, .bmp, and .png.

This property has no default value.

PrintShowActiveTag

The PrintShowActiveTag property is a read-write property that shows or hides the name of the active tag in the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

chart area of printed trends.

Syntax

```
aaHistClientTrend.PrintShowActiveTag = discrete;
Result = aaHistClientTrend.PrintShowActiveTag;
```

Remarks

True = Show the tag; False = Hide the tag.

The default value is True.

PrintShowMarkers

The PrintShowMarkers property is a read-write property that shows or hides the markers in printed trends.

Syntax

```
aaHistClientTrend.PrintShowMarkers = discrete;
Result = aaHistClientTrend.PrintShowMarkers;
```

Remarks

True = Show the markers; False = Hide the markers.

The default value is True.

PrintShowTitle

The PrintShowTitle property is a read-write property that shows or hides the print title in printed trends.

Syntax

```
aaHistClientTrend.PrintShowTitle = discrete;
Result = aaHistClientTrend.PrintShowTitle;
```

Remarks

True = Show the title; False = Hide the title.

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

PrintTitle

The PrintTitle property is a read-write property that gets or sets the print title for the trend.

Syntax

```
aaHistClientTrend.PrintTitle = message;
Result = aaHistClientTrend.PrintTitle;
```

Remarks

This property has no default value.

PublicAnnotations

The PublicAnnotations property is a read-write property that shows or hides all public annotations in the trend chart.

Syntax

```
aaHistClientTrend.PublicAnnotations = discrete;
Result = aaHistClientTrend.PublicAnnotations;
```

Remarks

The default value is True.

QueryTimeout

The QueryTimeout property is a read-write property that gets or sets the amount of time, in seconds, that the control waits for a query to be executed against the server before determining that the query failed.

Syntax

```
aaHistClientTrend.QueryTimeout = integer;
Result = aaHistClientTrend.QueryTimeout;
```

Remarks

This setting only applies to servers that you add or update dynamically using the AddServer method. All other servers continue to use the timeout that you set in the server configuration dialog box.check this

The default value is 20.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RealTimeMode

The RealTimeMode property is a read-write property that enables or disables live or replay mode.

Syntax

```
aaHistClientTrend.RealTimeMode = discrete;
Result = aaHistClientTrend.RealTimeMode;
```

Remarks

Use the LiveModeRate or RealTimeRate properties to set the rate at which the trend is refreshed in live or replay mode.

The default value is False.

RealTimeRate

The RealTimeRate property is a read-write property that gets or sets the refresh interval in seconds for live and replay mode.

Syntax

```
aaHistClientTrend.RealTimeRate = integer;
Result = aaHistClientTrend.RealTimeRate;
```

Remarks

The default value is 1.

Apart from the different unit of measure, this property serves the same purpose as the LiveModeRate property.

RetrievalOptionsCycleCount

This read-write property controls the aaHistClientTrend control's default number of cycles for cycle-based data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsCycleCount = integer;
Result = aaHistClientTrend.RetrievalOptionsCycleCount;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property is only taken into account if both the RetrievalOptionsUseAutoCycles property and the RetrievalOptionsUseResolution property are set to False. Also, it may be overridden by a retrieval style setting. For more information, see Working with Retrieval Styles.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

Valid values: any positive integer or 0. If you specify 0, the cycle count is calculated automatically, just as if the RetrievalOptionsUseAutoCycles property were set to True. The default value is 100.

RetrievalOptionsHistoryVersion

This read-write property determines the aaHistClientTrend control's default history source for data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsHistoryVersion = aaRetrievalVersion;
Result = aaHistClientTrend.RetrievalOptionsHistoryVersion;
```

Remarks

For information on possible values, see aaRetrievalVersion Enumeration. This property is relevant for all retrieval modes.

The default value is 0 (use the latest value).

RetrievalOptionsInterpolationType

This read-write property determines the aaHistClientTrend control's default interpolation type for data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsInterpolationType = aaInterpolationType;
Result = aaHistClientTrend.RetrievalOptionsInterpolationType;
```

Remarks

For information on possible values, see aaInterpolationType Enumeration. This property is only relevant for the following retrieval modes: Interpolated, Best Fit, Average, and Integral.

The default value is 3 (use the default value of the server).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RetrievalOptionsNumStyles

This read-only property returns the number of retrieval styles that are available in the control.

Syntax

```
Result = aaHistClientTrend.RetrievalOptionsNumStyles;
```

Remarks

The count only includes retrieval styles for which a name is defined for the current locale. If no style names at all are defined for the current locale, the count for the `en` locale is returned.

To return the name of a style with a specific number, use the RetrievalOptionsGetStyle method.

RetrievalOptionsQualityRule

This read-write property determines the aaHistClientTrend control's default quality rule for data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsQualityRule = aaQualityRules;
Result = aaHistClientTrend.RetrievalOptionsQualityRule;
```

Remarks

For information on possible values, see aaQualityRules Enumeration. This property is relevant for all retrieval modes except the following: Cyclic, Delta, and Full.

The default value is 3 (use the default value of the server).

RetrievalOptionsResolution

This read-write property controls the aaHistClientTrend control's default time interval for calculating the number of cycles in cycle-based data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsResolution = integer;
Result = aaHistClientTrend.RetrievalOptionsResolution;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property is only relevant if the RetrievalOptionsUseAutoCycles property is set to False, and the RetrievalOptionsUseResolution property is set to True. Also, it may be overridden by a retrieval style setting. For more information, see Working with Retrieval Styles.

The value of this property is a time interval in milliseconds. The aaHistClientTrend control divides the query duration by this interval and uses the result as the number of cycles for the query.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

Valid values: any positive integer or 0. If you specify 0, the cycle count is calculated automatically, just as if the RetrievalOptionsUseAutoCycles property were set to True. The default value is 1000.

RetrievalOptionsRetrievalMode

This read-write property determines the aaHistClientTrend control's default data retrieval mode. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsRetrievalMode = aaRetrievalMode;
Result = aaHistClientTrend.RetrievalOptionsRetrievalMode;
```

Remarks

This property may be overridden by a retrieval style setting.

For more information, see Working with Retrieval Style. For information on possible values, see aaRetrievalMode Enumeration.

The default value is 0 (cyclic). Make sure that the specified retrieval mode is supported by the AVEVA Historian that the tags are stored on.

RetrievalOptionsRetrievalStyle

This read-write property determines the aaHistClientTrend control's default retrieval style. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsRetrievalStyle = string;
Result = aaHistClientTrend.RetrievalOptionsRetrievalStyle;
```

Remarks

You must provide the retrieval style name for the current locale as it is defined in the retrieval style document. For more information, see Location and Structure of Retrieval Styles. To find out how many retrieval styles are

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

available in the control, use the RetrievalOptionsNumStyles property. To determine the name of a retrieval style if you know its position in the list of available styles, use the RetrievalOptionsGetStyle method.

Valid values: `Custom style` (or the translated equivalent for the current locale) and any retrieval style name that is defined for the current locale in the retrieval style document. Values are case-sensitive. If no style names at all are available for the current locale, use the name for the en locale. The default style is BestFit-5 (or the translated equivalent).

RetrievalOptionsRowLimit

This read-write property determines the aaHistClientTrend control's default row limit for data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsRowLimit = integer;
Result = aaHistClientTrend.RetrievalOptionsRowLimit;
```

Remarks

The row limit applies to each query. For more information, see RowLimit. This property is relevant for all retrieval modes.

Valid values: any positive number or 0 (no row limit). The default value is 0.

RetrievalOptionsState

This read-write property determines the aaHistClientTrend control's default state for which Time-in-State data is retrieved for a tag. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsState = message;
Result = aaHistClientTrend.RetrievalOptionsState;
```

Remarks

This property is only relevant for Time-in-State retrieval mode. It specifies the unique tag state for which Time-in-State information is calculated based on the calculation type specified by the RetrievalOptionsStateCalculation property.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RetrievalOptionsStateCalculation

This read-write property determines the aaHistClientTrend control's default calculation type for Time-in-State data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsStateCalculation = aaStateCalculation;
Result = aaHistClientTrend.RetrievalOptionsStateCalculation;
```

Remarks

For information on possible values, see aaStateCalculation Enumeration. This property is only relevant for Time-in-State retrieval mode. Also, it may be overridden by a retrieval style setting. For more information, see Working with Retrieval Styles.

The default value is 4 (percent).

RetrievalOptionsTimeDeadband

This read-write property determines the aaHistClientTrend control's default time deadband in milliseconds for Delta data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsTimeDeadband = integer;
Result = aaHistClientTrend.RetrievalOptionsTimeDeadband;
```

Remarks

Valid values: any positive number or 0 (no deadband). This property is only relevant for Delta retrieval mode. For more information on how this setting works, see Time Deadband (wwTimeDeadband).

The default value is 0 (no deadband).

RetrievalOptionsTimeStampRule

This read-write property determines the aaHistClientTrend control's default timestamp rule for data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsTimeStampRule = aaTimeStampRules;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Result = aaHistClientTrend.RetrievalOptionsTimeStampRule;
```

Remarks

For information on possible values, see aaTimeStampRules Enumeration.

This property is only relevant for the following retrieval modes: Cyclic, Interpolated, Time-Weighted Average, Integral, Counter, and Time-in-State.

The default value is 3 (use the default value of the server).

RetrievalOptionsUseAutoCycles

This read-write property controls the aaHistClientTrend control's default auto-calculation setting for cycle-based data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsUseAutoCycles = discrete;
Result = aaHistClientTrend.RetrievalOptionsUseAutoCycles;
```

Remarks

If this property is set to True, the aaHistClientTrend control automatically calculates the number of cycles for a query based on the width of the chart. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).

If it is set to False, you must specify the number of cycles manually. Use the RetrievalOptionsUseResolution property to specify whether you want to provide a number of cycles or a time interval. Then use the RetrievalOptionsCycleCount property to specify the number of cycles, or the RetrievalOptionsResolution property to specify the time interval.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

The default value is True.

RetrievalOptionsUseResolution

This read-write property controls the aaHistClientTrend control's default behavior for determining the number of cycles in cycle-based data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsUseResolution = discrete;
Result = aaHistClientTrend.RetrievalOptionsUseResolution;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property is only relevant if the RetrievalOptionsUseAutoCycles property is set to False.

If this property is set to False, the aaHistClientTrend control uses a fixed number of cycles when retrieving data using cycle-based retrieval modes. To specify the number of cycles, use the RetrievalOptionsCycleCount property.

If it is set to True, the aaHistClientTrend control calculates the number of cycles based on the query duration and a time interval. To specify this interval, use the RetrievalOptionsResolution property.

This property is relevant for all retrieval modes except the following: Delta, Full, and Slope.

The default value is False.

RetrievalOptionsValueDeadband

This read-write property determines the aaHistClientTrend control's default value deadband for Delta data retrieval. This setting applies to all tags in a trend whose retrieval style is set to `Style selected at option level`.

Syntax

```
aaHistClientTrend.RetrievalOptionsValueDeadband = real;
Result = aaHistClientTrend.RetrievalOptionsValueDeadband;
```

Remarks

The deadband is a percentage of the full scale in Engineering Units. Valid values are 0 (no deadband) to 100. This property is only relevant for Delta retrieval mode. For more information on how this setting works, see Value Deadband (wwValueDeadband).

The default value is 0 (no deadband).

RetrieveAnnotations

The RetrieveAnnotations property is a read-write property that enables or disables the retrieval of annotations.

Syntax

```
aaHistClientTrend.RetrieveAnnotations = discrete;
Result = aaHistClientTrend.RetrieveAnnotations;
```

Remarks

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RetrieveExtensionData

The RetrieveExtensionData property is a read-write property that enables or disables the retrieval of data from the extension tables.

Syntax

```
aaHistClientTrend.RetrieveExtensionData = discrete;
Result = aaHistClientTrend.RetrieveExtensionData;
```

Remarks

The extension data tables are logical tables that are populated from the AVEVA Historian data files. These tables support the historian time domain extensions for handling data.

The default value is True.

RetrieveManualData

The RetrieveManualData property is a read-write property that enables or disables the retrieval of data from the manual data tables.

Syntax

```
aaHistClientTrend.RetrieveManualData = discrete;
Result = aaHistClientTrend.RetrieveManualData;
```

Remarks

The manual data tables are normal SQL Server tables that are used to store data.

The default value is True.

RTRate

The RTRate property is a read-write property that gets or sets the live mode refresh interval, in seconds.

Syntax

```
aaHistClientTrend.RTRate = object;
Result = aaHistClientTrend.RTRate;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

Do not use. Only provided for backward compatibility. Use the RealTimeRate property instead.

Remarks

The default value is 1.

Rubberband

The RubberBand property is a read-write property that enables or disables rubber band scaling.

Syntax

```
aaHistClientTrend.RubberBand = discrete;
Result = aaHistClientTrend.RubberBand;
```

Remarks

Provided for backward compatibility. Use the RubberBandScaling property instead.

Remarks

The default value is False.

RubberbandAll

The RubberbandAll property is a read-write property that indicates whether all tags are scaled by rubber band scaling or just the selected tags.

Syntax

```
aaHistClientTrend.RubberbandAll = discrete;
Result = aaHistClientTrend.RubberbandAll;
```

Remarks

The default value is True.

RubberBandScaling

The RubberBandScaling property is a read-write property that enables or disables rubber band scaling.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.RubberBandScaling = discrete;
Result = aaHistClientTrend.RubberBandScaling;
```

Remarks

The default value is False.

Servers

The Servers property is a read-only property that gets the server list.

Syntax

```
Result = aaHistClientTrend.Servers;
```

Remarks

This property has no default value.

Return Value

The result is an aaServers object. For more information on the aaServers object, see aaServers Object.

ShowLimits

The ShowLimits property is a read-write property that shows or hides the limits for a tag.

Syntax

```
aaHistClientTrend.ShowLimits = discrete;
Result = aaHistClientTrend.ShowLimits;
```

Remarks

The default value is True.

ShowValuesAtCursor

The ShowValuesAtCursor property is a read-write property that shows/hides data values at the trend cursors along the value axis.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.ShowValuesAtCursor = discrete;
Result = aaHistClientTrend.ShowValuesAtCursor;
```

Remarks

The default value is False.

If the ShowValuesAtCursor property is set to True, the ValueAxisLabel property is set to 2, and values at cursors are shown in the chart. If the ShowValuesAtCursor property is set to False, the ValueAxisLabel property is set to 0, and multiple scales are shown in the chart.

ShowWaitCursor

The ShowWaitCursor property is a read-write property that controls the usage of the wait cursor (hourglass).

Syntax

```
aaHistClientTrend.ShowWaitCursor = discrete;
Result = aaHistClientTrend.ShowWaitCursor;
```

Remarks

The default value is False.

If the ShowWaitCursor property is set to true, the wait cursor (hourglass) is shown when you move the pointer over the toolbar, time bar, or the **Servers** pane or the **Filters** pane in the Tag Picker.

The ShowWaitCursor property setting does not override the wait cursor of the Trend. For example, if the ShowWaitCursor property is set to false, the Trend still shows a wait cursor during a refresh. This property only provides an option to force the wait cursor at other times.

ShowXAxisCursors

The ShowXAxisCursors property is a read-write property that shows or hides the time axis (x-axis) cursors.

Syntax

```
aaHistClientTrend.ShowXAxisCursors = discrete;
Result = aaHistClientTrend.ShowXAxisCursors;
```

Remarks

The default value is False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ShowYAxisCursor

The ShowYAxisCursor property is a read-write property that shows or hides the value axis (y-axis) cursors.

Syntax

```
aaHistClientTrend.ShowYAxisCursor = discrete;
Result = aaHistClientTrend.ShowYAxisCursor;
```

Remarks

The default value is False.

SingleTagMode

The SingleTagMode property is a read-write property that controls whether to show only the currently selected tag or all tags.

Syntax

```
aaHistClientTrend.SingleTagMode = discrete;
Result = aaHistClientTrend.SingleTagMode;
```

Remarks

The default value is False.

StartDate

The StartDate property is a read-only property that gets the timestamp at the left edge of the trend.

Syntax

```
Result = aaHistClientTrend.StartDate;
```

ReturnValue

The result is a DateTime data type. For information on the date/time value, see DateTime on page 675.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SummaryDataMode

This property is included for backward compatibility only. Its value is ignored.

Syntax

```
aaHistClientTrend.SummaryDataMode = discrete;
Result = aaHistClientTrend.SummaryDataMode;
```

Remarks

To retrieve summarized data, use a retrieval style instead. For more information, see Working with Retrieval Styles.

SupressErrors

The SupressErrors property is a read-write property that suppresses or allows errors.

Syntax

```
aaHistClientTrend.SupressErrors = discrete;
Result = aaHistClientTrend.SupressErrors;
```

Remarks

The default value is False.

TagGridOrientation

The TagGridOrientation property is a read-write property that orients the tag list vertically or horizontally.

Syntax

```
aaHistClientTrend.TagGridOrientation = integer;
Result = aaHistClientTrend.TagGridOrientation;
```

Remarks

0 = Horizontal; 1 = Vertical.

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagListRows

The TagListRows property is a read-write property that sets the height of the Tag List pane in the Trend control.

Syntax

```
aaHistClientTrend.TagListRows = integer;
Result = aaHistClientTrend.TagListRows;
```

Remarks

If the value of TagListRows is 0, the Tag List pane is not visible.

The default value is 5.

TagPicker

The TagPicker property is a read-only property that gets the TagPicker object used in the Trend control.

Syntax

```
Result = aaHistClientTrend.TagPicker;
```

Return Value

The return value is an aaHistClientTagPicker control. For more information on this control, see aaHistClientTagPicker Control.

TagPickerVisible

The TagPickerVisible property is a read-write property that shows or hides the Tag Picker in the Trend control.

Syntax

```
aaHistClientTrend.TagPickerVisible = discrete;
Result = aaHistClientTrend.TagPickerVisible;
```

Remarks

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TargetRegionExcursionType

The TargetRegionExcursionType property is a read-write property that determines whether the values that fall outside the target region of a tag are highlighted.

Syntax

```
aaHistClientTrend.TargetRegionExcursionType = aaTargetRegionExcursionType;
Result = aaHistClientTrend.TargetRegionExcursionType;
```

Remarks

For information on possible values, see aaTargetRegionExcursionType Enumeration.

The default value is 1 (highlight values in a special color).

TargetRegionOpacity

The TargetRegionOpacity is a read-write property that determines the opacity of a tag's target region.

Syntax

```
aaHistClientTrend.TargetRegionOpacity = integer;
Result = aaHistClientTrend.TargetRegionOpacity;
```

Remarks

A value of 0 means transparent, 100 means fully opaque. The default value is 20.

TimeAxisLabelColor

The TimeAxisLabelColor property is a read-write property that changes the color of the text labels that show the time in the chart area of the Trend control.

Syntax

```
aaHistClientTrend.TimeAxisLabelColor = integer;
Result = aaHistClientTrend.TimeAxisLabelColor;
```

Remarks

When the value of the TimeAxisLabelColor property changes, the color of the time-axis text labels also change. For more information on setting the color value, see Color.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The default value is 0 (black).

TimeBarVisible

The TimeBarVisible property is a read-write property that shows or hides the time and main toolbars in the Trend control.

Syntax

```
aaHistClientTrend.TimeBarVisible = discrete;
Result = aaHistClientTrend.TimeBarVisible;
```

Remarks

The default value is True.

TimeBarVisible2

The TimeBarVisible2 property is a read-write property that shows or hides the time toolbar in the Trend control.

Syntax

```
aaHistClientTrend.TimeBarVisible2 = discrete;
Result = aaHistClientTrend.TimeBarVisible2;
```

Remarks

This property is provided for backward compatibility. Alternatively, you can use the TimeBarVisible property, which shows or hides the main toolbar as well as the time toolbar.

The default value is True.

TimeSelector

The TimeSelector property is a read-only property that gets the Time Range Picker object used in the Trend control.

Syntax

```
Result = aaHistClientTrend.TimeSelector;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

The return value is an aaHistClientTimeRangePicker control. For more information on this control, see aaHistClientTimeRangePicker Control.

ToolBarVisible

The ToolBarVisible property is a read-write property that shows or hides the main toolbar in the Trend control.

Syntax

```
aaHistClientTrend.ToolBarVisible = discrete;
Result = aaHistClientTrend.ToolBarVisible;
```

Remarks

The default value is True.

ToolbarVisible2

The ToolBarVisible2 property is a read-write property that shows or hides the main toolbar in the Trend control.

Syntax

```
aaHistClientTrend.ToolBarVisible2 = discrete;
Result = aaHistClientTrend.ToolBarVisible2;
```

Remarks

This property is provided for backward compatibility only. Use the ToolBarVisible property instead.

The default value is True.

ToolTipText

The ToolTipText property is a read-write property that gets or sets the pop-up text that appears when the mouse cursor is hovered over the control at runtime.

Syntax

```
aaHistClientTrend.ToolTipText = message;
Result = aaHistClientTrend.ToolTipText;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The default is an empty message value ("").

TraceGradientEndingPercentage

This read-write property determines the ending opacity of a scatter plot trace if a gradient is used.

Syntax

```
aaHistClientTrend.TraceGradientEndingPercentage = integer;
Result = aaHistClientTrend.TraceGradientEndingPercentage;
```

Remarks

The ending opacity applies to the latest data point in the scatter plot. A value of 0 means transparent, 100 means fully opaque. The default value is 100. This property is only used if the TraceGradientType property is set to use a gradient.

TraceGradientStartingPercentage

This read-write property determines the starting opacity of a scatter plot trace if a gradient is used.

Syntax

```
aaHistClientTrend.TraceGradientStartingPercentage = integer;
Result = aaHistClientTrend.TraceGradientStartingPercentage;
```

Remarks

The starting opacity applies to the earliest data point in the scatter plot. A value of 0 means transparent, 100 means fully opaque. The default value is 20. This property is only used if the TraceGradientType property is set to use a gradient.

TraceGradientType

This read-write property determines whether a gradient is applied to the trace(s) in a scatter plot.

Syntax

```
aaHistClientTrend.TraceGradientType = aaTraceGradientType;
Result = aaHistClientTrend.TraceGradientType;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

For information on possible values, see aaTraceGradientType Enumeration.

The default value is 1 (opacity gradient).

TrendFontSize

This read-write property gets or sets the font size of the trend.

Syntax

```
aaHistClientTrend.TrendFontSize = integer;
Result = aaHistClientTrend.TrendFontSize;
```

UpdateToCurrentTimeState

This read-write property determines whether the Update to Current Time option is enabled.

Syntax

```
aaHistClientTrend.UpdateToCurrentTimeState = aaUpdateToCurrentTimeState;
Result = aaHistClientTrend.UpdateToCurrentTimeState;
```

Remarks

For information on how this option works in different scenarios, see Time Picker, Refreshing the Trend Chart, and Showing Live Data.

For information on possible values, see aaUpdateToCurrentTimeState Enumeration.

The default value is 1 (option is enabled).

UseIniFile

Do not use. Obsolete.

Syntax

```
aaHistClientTrend.UseIniFile = integer;
Result = aaHistClientTrend.UseIniFile;
```

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueAxisLabel

The ValueAxisLabel property is a read-write property that gets or sets the value axis labeling.

Syntax

```
aaHistClientTrend.ValueAxisLabel = aaValueAxisLabelEnumeration;
Result = aaHistClientTrend.ValueAxisLabel;
```

Remarks

The default value is 0 (MultipleScales).

For more information on value axis labeling, see Scaling Tags. For more information on the aaValueAxisLabelEnumeration enumeration, see aaValueAxisLabelEnumeration Enumeration.

If the ShowValuesAtCursor property is set to True, the ValueAxisLabel property is set to 2, and values at cursors are shown in the chart. If the ShowValuesAtCursor property is set to False, the ValueAxisLabel property is set to 0, and multiple scales are shown in the chart.

XCursor1Color

The XCursor1Color property is a read-write property that gets or sets the color for first time axis cursor.

Syntax

```
aaHistClientTrend.XCursor1Color = integer;
Result = aaHistClientTrend.XCursor1Color;
```

Remarks

For information on setting the color value, see Color.

The default value is 255 (red).

XCursor1Pos

The XCursor1Pos property is a read-write property that controls the position of the first time axis cursor.

Syntax

```
aaHistClientTrend.XCursor1Pos = DateTime;
Result = aaHistClientTrend.XCursor1Pos;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The value is given as a date/time value. For information on the date/time value format, see DateTime.

To control the position of the first X axis cursor in a scatter plot, use the CurrentValOfX1 property instead.

This property has no default value.

XCursor2Color

The XCursor2Color property is a read-write property that gets or sets the color for second time axis cursor.

Syntax

```
aaHistClientTrend.XCursor2Color = integer;
Result = aaHistClientTrend.XCursor2Color;
```

Remarks

For information on setting the color value, see Color.

The default value is 16711680 (blue).

XCursor2Pos

The XCursor2Pos property is a read-write property that controls the position of the second time axis cursor.

Syntax

```
aaHistClientTrend.XCursor2Pos = DateTime;
Result = aaHistClientTrend.XCursor2Pos;
```

Remarks

The value is given as a date/time value. For information on the date/time value format, see DateTime.

To control the position of the second X axis cursor in a scatter plot, use the CurrentValOfX2 property instead.

This property has no default value.

YCursor1Color

The YCursor1Color property is a read-write property that gets or sets the color for first value axis cursor.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.YCursor1Color = integer;
Result = aaHistClientTrend.YCursor1Color;
```

Remarks

For information on setting the color value, see Color.

The default value is 32768 (dark green).

YCursor2Color

The YCursor2Color property is a read-write property that gets or sets the color for second value axis cursor.

Syntax

```
aaHistClientTrend.YCursor2Color = integer;
Result = aaHistClientTrend.YCursor2Color;
```

Remarks

For information on setting the color value, see Color.

The default value is 32768 (dark green).

ZoomOutPercentage

The ZoomOutPercentage property is a read-write property that gets or sets the percentage (1 to 100) to zoom by when zooming out on the trend chart.

Syntax

```
aaHistClientTrend.ZoomOutPercentage = integer;
Result = aaHistClientTrend.ZoomOutPercentage;
```

Remarks

The default value is 25.

aaHistClientTrend Methods

The following are the methods used by the aaHistClientTrend:

- AboutBox

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- AddAnyTag
- AddServer
- AddServerEx
- AddTag
- ClearTags
- CurrentTagGetStyle
- DeleteCurrentTag
- FileNew
- FileOpen
- FileOpenEx
- FileSave
- FileSaveEx
- GetMenuItemEnabled
- GetTagColor
- GetTagFormat
- GetTagOffsetMS
- GetTagPenStyle
- GetTagPenWidth
- GetTagPrecision
- GetTagValAtX1
- GetTagValAtX2
- GetTagVisible
- GetToolbarButtonEnabled
- GraphStack
- LoadCRVString
- LoadTargetRegionFromFile
- ManualConnect
- MoveNextTag
- MovePrevTag
- PanLeft
- PanRight
- PrintGraph
- PrintGraphDlg
- PropertiesDlg
- RefreshData
- RemoveServer
- RemoveServerEx
- RemoveTag

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- RetrievalOptionsGetStyle
- SaveData
- SaveImage
- SaveSettings
- ScaleAllTags
- ScaleAllTagsDlg
- ScaleAutoAllTags
- ScaleAutoTag
- ScaleDownAllTags
- ScaleDownTag
- ScaleMoveAllTagsDown
- ScaleMoveAllTagsUp
- ScaleMoveTagDown
- ScaleMoveTagUp
- ScaleTag
- ScaleTagDlg
- ScaleUpAllTags
- ScaleUpTag
- SetCurrentTag
- SetCurrentTagXAxisTag
- SetCurrentTagXAxisTagIndex
- SetDates
- SetDuration
- SetMenuItemEnabled
- SetTagColor
- SetTagFormat
- SetTagColorDlg
- SetTagOffsetMS
- SetTagPenStyle
- SetTagPenWidth
- SetTagPrecision
- SetTagVisible
- SetTimeSpan
- SetToolbarButtonEnabled
- ShowStatistics
- ZoomIn
- ZoomOut

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AboutBox

The AboutBox method shows the About dialog box for the control.

Syntax

```
[Result=] aaHistClientTrend.AboutBox();
```

AddAnyTag

The AddAnyTag method verifies and adds a tag to the trend.

Syntax

```
[Result=] aaHistClientTrend.AddAnyTag(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

Return Value

Returns True if the tag was added; otherwise returns False.

Remarks

The tag can be on any server. This method first checks if the tag exists before adding it. The AddTag method also adds a tag, but it does not perform the checking and is thus more efficient.

If you specify a server name that is part of the current server list, but is currently disconnected, an attempt is made to connect to the server. If the authentication credentials are correct, the server is logged on, and the tag added.

If you specify a server name that is not part of the current server list, the runtime user is prompted to add the server name to the server list. A False is returned. If you want to suppress the notification, use the SupressErrors property. For more information, see SupressErrors.

AddServer

The AddServer method adds a server to the list.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientTrend.AddServer(message serverName, message loginName, message
password, [discrete bPersistPassword]);
```

Parameters

serverName

The name of the server.

loginName

A valid user name to log on to the server. If no login name is provided, Windows integrated security is used.

password

A valid password for the server.

bPersistPassword

Optional parameter. If set to True, the password is remembered for the subsequent connection. The password is only remembered for single application; the persisted password is not available to all applications. The default value is True.

AddServerEx

The AddServer method adds a server to the list.

Syntax

```
[Result=] aaHistClientTrend.AddServerEx(message serverName, message loginName, message
password, [discrete bPersistPassword]);
```

Parameters

serverName

The name of the server.

loginName

A valid user name to log on to the server. If no login name is provided, Windows integrated security is used. See remarks below.

password

A valid password for the server.

bPersistPassword

If set to True, the password is remembered for the subsequent connection. The password is only remembered for a single application; the persisted password is not available to all applications.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the server can be added; otherwise returns False.

Remarks

All parameters are required. Errors, if any, are reported.

To use Windows integrated security, provide empty values for the login name and password, and set bPersistPassword to true:

```
[Result=] aaHistClientTrend.AddServerEx("MyServer", "", "", true);
```

AddTag

The AddTag method adds the specified tag to the trend.

Syntax

```
[Result=] aaHistClientTrend.AddTag(message serverName, message newTag, integer tType);
```

Parameters

serverName

The name of the server for which to add the tag.

newtag

The name of the tag to add.

tType

The type of tag. This parameter is provided for backward compatibility and does not have any effect on the outcome of the operation. However, you must still specify one of the following valid values: 1, 2, 3, or 5.

Return Value

Returns True if the tag can be added; otherwise returns False.

ClearTags

The ClearTags method removes all tags from the trend.

Syntax

```
[Result=] aaHistClientTrend.ClearTags();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentTagGetStyle

This method returns the name of a retrieval style based on its index in the list of available retrieval styles for the currently selected tag.

Syntax

```
Result = aaHistClientTrend.CurrentTagGetStyle(integer styleNumber);
```

Parameters

styleNumber

The index of the style whose name you want to retrieve. Counting starts at 0.

Return Value

Returns the style's name as defined for the current locale. If no style names are defined for the current locale, the name in the en locale is returned.

Remarks

To find out how many retrieval styles are available for the current tag, use the CurrentTagNumStyles property.

DeleteCurrentTag

The DeleteCurrentTag method deletes the currently selected tag.

Syntax

```
[Result=] aaHistClientTrend.DeleteCurrentTag();
```

Return Value

Returns True if the tag can be deleted; otherwise returns False.

FileNew

The FileNew method creates a new file and then resets the trend to the default properties.

Syntax

```
[Result=] aaHistClientTrend.FileNew();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the file is successfully created; otherwise returns False.

FileOpen

The FileOpen method opens the specified trend file.

Syntax

```
[Result=] aaHistClientTrend.FileOpen([message fileName]);
```

Parameters

fileName
Optional parameter. The full path to the trend file to open.

Return Value

Returns True if the file can be successfully opened; otherwise, returns False.

Remarks

Any errors are reported.

FileOpenEx

The FileOpenEx method opens the specified trend file.

Syntax

```
[Result=] aaHistClientTrend.FileOpenEx([message fileName]);
```

Parameters

fileName
The full path to the trend file to open.

Return Value

Returns True if the file can be successfully opened; otherwise, returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

All parameters are required. Errors, if any, are reported.

FileSave

The FileSave method saves the trend to the specified file.

Syntax

```
[Result=] aaHistClientTrend.FileSave([message fileName]);
```

Parameters

fileName
Optional parameter. The name of the trend file to save.

Return Value

Returns True if the file can be successfully saved; otherwise, returns False.

Remarks

Any Errors are reported.

FileSaveEx

The FileSaveEx method saves the trend to the specified file.

Syntax

```
[Result=] aaHistClientTrend.FileSaveEx([message fileName]);
```

Parameters

fileName
The name of the trend file to save.

Return Value

Returns True if the file can be successfully saved; otherwise, returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

All parameters are required. Errors, if any, are reported.

GetMenuItemEnabled

Use the GetMenuItemEnabled method to check if a specific command in the context menu is enabled.

Syntax

```
[Result=] aaHistClientTrend.GetMenuItemEnabled(integer itemNumber);
```

Parameters

itemNumber
The index number of the command. Numbering starts at 0.

Return Value

Returns True if the menu item is enabled; otherwise, returns False.

Remarks

If you specify -1 as the *itemNumber* parameter, the method checks the status of all items in the menu.

GetTagColor

The GetTagColor method gets the line color of the tag curve in the trend.

Syntax

```
[Result=] aaHistClientTrend.GetTagColor(message serverName, message tagName);
```

Parameters

serverName
The name of the server.
tagName
The name of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns an integer that specifies the color. For information on the color value, see Color.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetTagFormat

The GetTagFormat method gets how the values for the tag appear, either in decimal format or scientific format.

Syntax

```
[Result=] aaHistClientTrend.GetTagFormat(message serverName, message tagName);
```

Parameters

serverName
The name of the server.
tagName
The name of the tag.

Return Value

Returns an integer. 0 = Decimal; 1 = Scientific.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetTagOffsetMS

The GetTagOffsetMS method gets the amount of time that the trend curve is shifted from the actual time.

Syntax

```
[Result=] aaHistClientTrend.GetTagOffsetMS(message serverName, message tagName);
```

Parameters

serverName

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The name of the server.

tagName

The name of the tag.

Return Value

The result is an integer value for the tag offset in milliseconds. For more information, see Using Time Offsets to Compare Data.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetTagPenStyle

The GetTagPenStyle method gets the style of the trend curve for the currently selected tag. For example, a solid or dashed line.

Syntax

```
[Result=] aaHistClientTrend.GetTagPenStyle(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

Return Value

Returns the pen style as an integer value. Valid values are:

0	Solid
1	Dashed
2	Dotted
3	DashDot
4	DashDotDot
5	Alternate

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetTagPenWidth

The GetTagPenWidth method gets the thickness of the trend curve for the selected tag.

Syntax

```
[Result=] aaHistClientTrend.GetTagPenWidth(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

Return Value

The width, in pixels, of the pen as an integer.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetTagPrecision

The GetTagPrecision method gets the number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.

Syntax

```
[Result=] aaHistClientTrend.GetTagPrecision(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

The decimal places (precision) for the tag as an integer.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetTagValAtX1

The GetTagValAtX1 method gets the value of the specified tag at the point at which the curve intersects with the first time axis cursor.

Syntax

```
[Result=] aaHistClientTrend.GetTagValAtX1(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

Return Value

The tag value as a real.

Remarks

For more information on cursors, see Using Axis Cursors. If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

In a scatter plot, this method behaves as if the X axis were a time axis and the X axis cursors were time cursors. For example, if the plot shows data from 3:00 PM to 4:00 PM, and the cursor is exactly at the middle of the X axis, this method returns the value of the tag at 3:30 PM.

GetTagValAtX2

The GetTagValAtX2 method gets the value of the specified tag at the point at which the curve intersects with the second time axis cursor.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientTrend.GetTagValAtX2(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

Return Value

The tag value as a real.

Remarks

For more information on cursors, see Using Axis Cursors. If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

In a scatter plot, this method behaves as if the X axis were a time axis and the X axis cursors were time cursors. For example, if the plot shows data from 3:00 PM to 4:00 PM, and the cursor is exactly at the middle of the X axis, this method returns the value of the tag at 3:30 PM.

GetTagVisible

The GetTagVisible method gets whether the selected tag is visible in the trend chart.

Syntax

```
[Result=] aaHistClientTrend.GetTagVisible(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

Return Value

The visibility as a discrete. False = Not visible; True = Visible.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

GetToolbarButtonEnabled

Use the GetToolbarButtonEnabled method to check if a specific button in the toolbar is enabled.

Syntax

```
[Result=] GetToolbarButtonEnabled(integer buttonNumber);
```

Parameters

buttonNumber

The index number of the toolbar button. Numbering starts at 0.

Return Value

Returns True if the button is enabled; otherwise, returns False.

GraphStack

This method toggles the chart between "stacked" mode (one tag curve on top of the other) and non-stacked mode.

Syntax

```
[Result=] aaHistClientTrend.GraphStack();
```

Return Value

Returns True if the operation was successful.

LoadCRVString

The LoadCRVString method is an obsolete method. Do not use.

Syntax

```
[Result=] aaHistClientTrend.LoadCRVString(message crv);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LoadTargetRegionFromFile

This method sets a target region for the currently selected tag based on values read from a CSV file. It replaces any existing target region that may already be defined for the tag.

Syntax

```
[Result=] aaHistClientTrend.LoadTargetRegionFromFile(message source);
```

Parameters

source

The location of the file containing the target region items. This can be a local file name or a URL.

Return Value

Returns True if the tag's target region was set successfully; otherwise, returns False, and the tag's existing target region is left unchanged.

Remarks

For information on file format requirements, see Defining a Target Region for a Tag for regular trends, and Defining a Target Region for a Scatter Plot for scatter plots.

ManualConnect

The ManualConnect method opens the **Server connection** dialog box.

Syntax

```
[Result=] aaHistClientQuery.ManualConnect();
```

MoveNextTag

The MoveNextTag method sets the current tag to the next tag in the tag list.

Syntax

```
[Result=] aaHistClientTrend.MoveNextTag();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the operation was successful; otherwise, False is returned. If you call this method while the last tag in the list is selected, the current tag is set to the first tag in the list.

MovePrevTag

The MovePrevTag method sets the current tag to the previous tag in the tag list.

Syntax

```
[Result=] aaHistClientTrend.MovePrevTag();
```

Return Value

Returns True if the operation was successful; otherwise, returns False. If you call this method while the first tag in the list is selected, the first tag remains the current tag.

PanLeft

The PanLeft method pans the trend to the left by the amount specified by pan percentage.

Syntax

```
[Result=] aaHistClientTrend.PanLeft();
```

Return Value

Returns True if the time range for the panning can be set; otherwise, returns False.

Remarks

The pan percentage is set using the PanPercentage property.

PanRight

The PanRight method pans the trend to the right by the amount specified by pan percentage.

Syntax

```
[Result=] aaHistClientTrend.PanRight();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the time range for the panning can be set; otherwise, returns False.

Remarks

The pan percentage is set using the PanPercentage property.

PrintGraph

The PrintGraph method prints the trend chart to the default printer.

Syntax

```
[Result=] aaHistClientTrend.PrintGraph();
```

PrintGraphDlg

The PrintGraphDlg method displays the Print dialog box, allowing the runtime user to choose the printer to which to print the trend chart.

Syntax

```
[Result=] aaHistClientTrend.PrintGraphDlg();
```

PropertiesDlg

The PropertiesDlg method opens the **Trend Properties** dialog box.

Syntax

```
[Result=] aaHistClientTrend.PropertiesDlg();
```

RefreshData

The RefreshData method refreshes the trend chart by retrieving new data for all tags.

Syntax

```
[Result=] aaHistClientTrend.RefreshData();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the trend was successfully updated; otherwise, returns False.

Remarks

Data is requested from the databases as necessary. This method ensures that all tags within the trend that can be synchronized are synchronized.

RemoveServer

The RemoveServer method removes the specified server from the servers list. If no server is specified, this method removes the entire server list.

Syntax

```
[Result=] aaHistClientTrend.RemoveServer([message serverName]);
```

Parameters

serverName

Optional parameter. The name of the server to remove.

Return Value

Returns True if the server was successfully removed; otherwise, returns False.

RemoveServerEx

The RemoveServerEx method removes the specified server from the servers list. If no server is specified, this method removes the entire server list.

Syntax

```
[Result=] aaHistClientTrend.RemoveServerEx([message serverName]);
```

Parameters

serverName

The name of the server to remove.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the server was successfully removed; otherwise, returns False.

Remarks

All parameters are required. Errors, if any, are reported.

RemoveTag

The RemoveTag method removes the specified tag from the trend.

Syntax

```
[Result=] aaHistClientTrend.RemoveTag(message serverName, message tagName);
```

Parameters

serverName

The name of the server that the tag is stored on.

tagName

The name of the tag to remove.

Return Value

Returns True if the tag was successfully removed; otherwise, returns False. If a tag is shown in the chart multiple times, the method removes the first instance that was added.

RetrievalOptionsGetStyle

This method returns the name of a retrieval style based on its index in the list of retrieval styles that are available in the control.

Syntax

```
Result = aaHistClientTrend.RetrievalOptionsGetStyle(integer styleNumber);
```

Parameters

styleNumber

The index of the style whose name you want to retrieve. Counting starts at 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns the style's name as defined for the current locale. If no style names are defined for the current locale, the name in the en locale is returned.

Remarks

To find out how many retrieval styles are available in the control, use the RetrievalOptionsNumStyles property.

SaveData

The SaveData method optionally prompts the runtime user and saves the trend data (in the "wide" format) or image to a file or to the clipboard.

Syntax

```
[Result=] aaHistClientTrend.SaveData(integer format, message fileName);
```

Parameters

format

The type of output:

0	Saves trend data in tab-delimited format using the file name specified in the *fileName* parameter.
1	Copies the trend image to the clipboard.
2	Copies the trend image to the clipboard. (Legacy option)
3	Saves the trend image in JPEG format using the file name specified in the *fileName* parameter.
100	Opens the **Save** dialog box to save the trend data in CSV or tab-delimited format.

fileName

The name of the file.

Return Value

Returns True if the operation was successful; otherwise, returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SaveImage

The SaveImage method saves the trend image to a JPEG file.

Syntax

```
[Result=] aaHistClientTrend.SaveImage(message fileName);
```

Parameters

fileName

The name of the file. If you leave this value empty and the current trend has no file name, an error message appears when the method is executed. If you leave this value empty and the current trend has a file name, the file is saved using the trend's file name with a .JPG extension.

Return Value

Returns True if the file was successfully saved; otherwise, returns False.

SaveSettings

The SaveSettings method saves the current file.

Syntax

```
[Result=] aaHistClientTrend.SaveSettings();
```

Return Value

Returns True if the file was successfully saved; otherwise, returns False.

Remarks

If no file name currently exists, the user is prompted to specify a file name.

ScaleAllTags

The ScaleAllTags method sets the y-axis scale for all tags in the chart.

Syntax

```
[Result=] aaHistClientTrend.ScaleAllTags(real min, real max);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

min

The minimum value for the value (y-axis) scale.

max

The maximum value for the value (y-axis) scale.

Return Value

Returns True if the tags were successfully scaled; otherwise, returns False.

ScaleAllTagsDlg

The ScaleAllTagsDlg method opens a dialog box that allows the user to enter new minimum and maximum scale values for all tags in the chart.

Syntax

```
[Result=] aaHistClientTrend.ScaleAllTagsDlg();
```

Return Value

Returns True if the tags were scaled as a result of this operation; otherwise, returns False (for example, if the user clicked **Cancel** in the dialog box).

ScaleAutoAllTags

The ScaleAutoAllTags method sets a suitable y-axis scale for all tags in the chart according to the currently displayed minimum and maximum values.

Syntax

```
[Result=] aaHistClientTrend.ScaleAutoAllTags();
```

Return Value

Returns True if the scale was successfully set; otherwise, returns False.

ScaleAutoTag

The ScaleAutoTag method sets a suitable y-axis scale for the currently selected tag according to the currently

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

displayed minimum and maximum values.

Syntax

```
[Result=] aaHistClientTrend.ScaleAutoTag();
```

Return Value

Returns True if the scale was successfully set; otherwise, returns False.

ScaleDownAllTags

This method increases the value range of all tags in the chart by one third.

Syntax

```
[Result=] aaHistClientTrend.ScaleDownAllTags();
```

Return Value

Returns True if the scaling was successful; otherwise, returns False.

ScaleDownTag

This method increases the value range of the currently selected tag by one third.

Syntax

```
[Result=] aaHistClientTrend.ScaleDownTag();
```

Return Value

Returns True if the scaling was successful; otherwise, returns False.

ScaleMoveAllTagsDown

The ScaleMoveAllTagsDown method moves the value scale down for all tags in the chart.

Syntax

```
[Result=] aaHistClientTrend.ScaleMoveAllTagsDown();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the scaling was successful; otherwise, returns False.

ScaleMoveAllTagsUp

The ScaleMoveAllTagsUp method moves the value scale up for all tags in the chart.

Syntax

```
[Result=] aaHistClientTrend.ScaleMoveAllTagsUp();
```

Return Value

Returns True if the scaling was successful; otherwise, returns False.

ScaleMoveTagDown

The ScaleMoveTagDown method moves the value scale down for the currently selected tag.

Syntax

```
[Result=] aaHistClientTrend.ScaleMoveTagDown();
```

Return Value

Returns True if the scaling was successful; otherwise, returns False.

ScaleMoveTagUp

The ScaleMoveTagUp method moves the value scale up for the currently selected tag.

Syntax

```
[Result=] aaHistClientTrend.ScaleMoveTagUp();
```

Return Value

Returns True if the scaling was successful; otherwise, returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ScaleTag

The ScaleTag method sets the y-axis scale for the currently selected tag.

Syntax

```
[Result=] aaHistClientTrend.ScaleTag(real min, real max);
```

Parameters

min

The minimum value for the value (y-axis) scale.

max

The maximum value for the value (y-axis) scale.

Return Value

Returns True if the tag was successfully scaled; otherwise, returns False.

ScaleTagDlg

The ScaleTagDlg method opens a dialog box that allows the user to enter new minimum and maximum scale values for the currently selected tag.

Syntax

```
[Result=] aaHistClientTrend.ScaleTagDlg();
```

Return Value

Returns True if the tag was scaled as a result of this operation; otherwise, returns False (for example, if the user clicked **Cancel** in the dialog box).

ScaleUpAllTags

This method decreases the value range of all tags in the chart by one fourth.

Syntax

```
[Result=] aaHistClientTrend.ScaleUpAllTags();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the scaling was successful; otherwise, returns False.

ScaleUpTag

This method decreases the value range of the currently selected tag by one fourth.

Syntax

```
[Result=] aaHistClientTrend.ScaleUpTag();
```

Return Value

Returns True if the scaling was successful; otherwise, returns False.

SetCurrentTag

The SetCurrentTag method sets the specified tag to be the current tag.

Syntax

```
[Result=] aaHistClientTrend.SetCurrentTag(message serverName, message tagName);
```

Parameters

serverName
The name of the server.
tagName
The name of the tag.

Return Value

The return value is a discrete. Returns True if successful; otherwise returns False.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SetCurrentTagXAxisTag

This method configures the currently selected tag in a scatter plot to use another tag from the tag list as its X axis tag. The X axis tag is identified by its server and name.

Syntax

```
[Result=] aaHistClientTrend.SetCurrentTagXAxisTag(message serverName, message tagName);
```

Parameters

serverName

The name of the server that the *tagName* tag is stored on.

tagName

The name of the tag that you want to use as the X axis tag for the current tag. The tag must already be contained in the tag list.

Return Value

The return value is a discrete. Returns True if successful; otherwise returns False. Possible causes of failures include:

- No tag is currently selected.
- No tag matches the specified parameters.
- The current tag is not an analog or discrete tag.
- The designated X axis tag is the current tag itself.
- The designated X axis tag is not an analog or discrete tag.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetCurrentTagXAxisTagIndex

This method configures the currently selected tag in a scatter plot to use another tag from the tag list as its X axis tag. The X axis tag is identified by its index.

Syntax

```
[Result=] aaHistClientTrend.SetCurrentTagXAxisTagIndex(integer index);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

index

The index of the tag that you want to use as the X axis tag for the current tag.

Return Value

The return value is a discrete. Returns True if successful; otherwise returns False. Possible causes of failures include:

- No tag is currently selected.
- No tag matches the specified index.
- The current tag is not an analog or discrete tag.
- The designated X axis tag is the current tag itself.
- The designated X axis tag is not an analog or discrete tag.

SetDates

The SetDates method sets the start and end time for the trend.

Syntax

```
[Result=] aaHistClientTrend.SetDates(DateTime startTime, DateTime endTime);
```

Parameters

startTime

The start time for the trend.

endTime

The end time for the trend.

Remarks

For more information on setting the date/time, see DateTime.

In relative time mode, you must still specify an absolute date/time value. For example, if the start time of your tags is 11/13/2006 8:00 AM and you want the trend to start at an offset of one hour to that start time, specify 11/13/2006 9:00 AM for the startTime parameter.

Return Value

Returns True if the dates were set. Returns False in case of an error.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SetDuration

The SetDuration method sets the time period for the trend as a duration relative to the current time.

Syntax

```
[Result=] aaHistClientTrend.SetDuration(real duration);
[Result=] aaHistClientTrend.SetDuration(DateTime duration);
```

Parameters

duration

The duration from the current time.

Remarks

When using the ActiveX version of the control, the duration parameter can be either a number of days or a date/time string.

When using the .NET version of the control, the duration parameter must be a valid DateTime value.

In both cases, when you specify a date/time value, the duration is the difference between the specified date/time and the base date of December 30th, 1899, 12:00:00 AM.

For more information on the format for date/time values, see DateTime.

Examples

In the following example, the duration is set to the past five minutes, relative to the current time.

```
#aaHistClientTrend1.SetDuration("00:05:00");
```

In the following example, the time period is set to the past 36 hours by specifying the number of days.

```
#aaHistClientTrend1.SetDuration(1.5);
```

In the following example, the duration is set to the past 36 hours by specifying a date/time value.

```
#aaHistClientTrend1.SetDuration("12/31/2018 12:00:00");
```

SetMenuItemEnabled

Use the SetMenuItemEnabled method to control if a specific command in the shortcut menu is enabled.

Syntax

```
[Result=] aaHistClientTrend.SetMenuItemEnabled(integer itemNumber, integer bEnabled);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

itemNumber

The index number of the command. Numbering starts at 0.

bEnabled

Specify a non-zero number to enable or zero to disable.

Return Value

Returns True if the menu item is enabled; otherwise, returns False.

Remarks

If you specify -1 as the *itemNumber* parameter, the method sets the status of all items in the menu.

Item numbers are as follows:

Number	Corresponding menu item
0	File
2	Single Tag Mode
3	Highlight Tag
5	Next Tag
6	Previous Tag
7	Add Annotation
8	Delete Tag
10	Color
12	View
13	Show
15	Scale Tag
16	Scale All Tags
18	Rubber Band Scaling
19	Apply Rubber Band To All Tags
21	Pan & Zoom
23	Copy

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Number	Corresponding menu item
24	Save Data
25	Print
26	Properties
28	Chart Type
29	Tools
30	Live Mode
31	Stacked Traces
32	Refresh
33	Update To Current Time

SetTagColor

The SetTagColor method sets the line color of the tag curve in the trend.

Syntax

```
[Result=] aaHistClientTrend.SetTagColor(message serverName, message tagName, integer
color);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

color

The color value for the curve.

Return Value

Returns True if successful; otherwise returns False.

Remarks

For information on setting the color value, see Color.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If the tag is shown multiple times in the chart, this property applies to the first instance of the tag that was added.

SetTagFormat

The SetTagFormat method sets how the values for the tag appear, either in decimal format or scientific format.

Syntax

```
[Result=] aaHistClientTrend.SetTagFormat(message serverName, message tagName, long format);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

format

The format for the tag value. 0 = Decimal; 1 = Scientific.

Return Value

Returns True if successful; otherwise returns False.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetTagColorDlg

This method opens a dialog box where the user can specify a color for the currently selected tag.

Syntax

```
[Result=] aaHistClientTrend.SetTagColorDlg();
```

Return Value

Returns a discrete value. Returns True if the dialog was shown; otherwise, returns False (for example, if there are no tags in the trend).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SetTagOffsetMS

The SetTagOffsetMS method sets the amount of time that the trend curve of the currently selected tag will be shifted from the actual time.

Syntax

```
[Result=] aaHistClientTrend.SetTagOffsetMS(message serverName, message tagName, integer
milliseconds);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

milliseconds

The offset, for the shift in milliseconds. The offset can be positive or negative. For more information, see Using Time Offsets to Compare Data.

Return Value

Returns a discrete value. Returns True if the set was successful; otherwise, returns False.

Due to the limited range for integer values, the maximum offset you can set using this property is about 29 days. For larger offsets, use the CurrentTagStartDate property.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetTagPenStyle

The SetTagPenStyle method sets the style of the trend curve for the currently selected tag. For example, a solid or dashed line.

Syntax

```
[Result=] aaHistClientTrend.SetTagPenStyle(message serverName, message tagName, integer
penStyle);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

serverName

The name of the server.

tagName

The name of the tag.

penStyle

The appearance of the pen. Valid values are:

0	Solid
1	Dashed
2	Dotted
3	DashDot
4	DashDotDot
5	Alternate

Return Value

Returns True if successful; otherwise returns False.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetTagPenWidth

The SetTagPenWidth method sets the thickness of the trend curve.

Syntax

```
[Result=] aaHistClientTrend.SetTagPenWidth(message serverName, message tagName, integer
width);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

width

The width, in pixels, of the pen.

Return Value

Returns True if successful; otherwise returns False.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetTagPrecision

The SetTagPrecision method sets the number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.

Syntax

```
[Result=] aaHistClientTrend.SetTagPrecision(message serverName, message tagName, integer
precision);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

precision

The decimal places (precision) for the tag. Valid values are 0 to 15.

Return Value

Returns True if successful; otherwise returns False.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetTagVisible

The SetTagVisible method sets whether a tag is visible in the trend chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientTrend.SetTagVisible(message serverName, message tagName, discrete
bVisible);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

bVisible

False = Not visible; True = Visible.

Return Value

Returns True if successful; otherwise returns False.

Remarks

If the specified tag is shown in the chart multiple times, the method uses the first instance that was added.

SetTimeSpan

The SetTimeSpan method sets the start and end time for the trend.

Syntax

```
[Result=] aaHistClientTrend.SetTimeSpan(DateTime startTime, DateTime endTime, integer
duration);
```

Parameters

startTime

The start time for the trend. Only considered if the duration is set to Custom. For other durations, the start time is calculated automatically based on the end time and duration.

endTime

The end time for the trend. Only considered if the duration is set to Custom, or if the property is set to False and the duration is set to an option from 17 to 32 (OneMinute to ThreeMonths). Otherwise, the end time is always assumed to be the current time.

duration

The time duration. If the duration is set to Custom, the specified start and end times are used. For other duration options, the time indicated by the duration is used, and the start and/or end times are updated as necessary. For

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

more information on valid values for the duration, see aaTimeRangeEnumeration Enumeration.

Remarks

For information on setting the date/time value, see DateTime.

SetToolbarButtonEnabled

Use the SetToolbarButtonEnabled method to control if a specific button in the toolbar is enabled.

Syntax

```
[Result=] aaHistClientTrend.SetToolbarButtonEnabled(integer buttonNumber, integer
bEnabled);
```

Parameters

buttonNumber

The index number of the toolbar button. Numbering starts at 0.

bEnabled

Specify a non-zero number to enable the button. Set to zero to disable.

Return Value

Returns True if the button can be enabled; otherwise, returns False.

Button numbers are as follows:

Number	Corresponding button
0	Open a trend
1	Save the trend
2	Print the trend
3	Copy
5	Configure the servers
6	Configure the trend properties
7	Configure the trend options
9	Select XY Scatter Plot or Trend chart type
10	Enable or disable single tag mode

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Number	Corresponding button
11	Stack the tag traces
12	Move to the previous tag
13	Move to the next tag
14	Highlight tag
16	Show or hide the time axis cursor
17	Show or hide the value axis cursor
19	Move the current tag up
20	Move the current tag down
21	Scale all tags to their original scale
22	Auto scale all tags
23	Scale all tags up
24	Scale all tags down
26	Enable rubber band scaling
27	Apply rubber band to all tags
28	View license status

ShowStatistics

The ShowStatistics method shows the **Statistics** dialog box.

Syntax

```
[Result=] aaHistClientTrend.ShowStatistics();
Remarks
```

For more information about the **Statistics** dialog box, see Viewing Statistics.

UnsetCurrentTagXAxisTag

This method removes any associated X axis tag from the currently selected tag in a scatter plot.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTrend.UnsetCurrentTagXAxisTag();
```

Remarks

If the current tag is not associated with any X axis tag, this method does nothing.

ZoomIn

The ZoomIn method zooms in on the trend chart.

Syntax

```
[Result=] aaHistClientTrend.ZoomIn();
```

Return Value

Returns True if the time range for the operation can be set; otherwise, returns False.

Remarks

The amount of the zoom is controlled by the ZoomOutPercentage property.

ZoomOut

The ZoomOut method zooms out on the trend chart.

Syntax

```
[Result=] aaHistClientTrend.ZoomOut();
```

Return Value

Returns True if the time range for the operation can be set; otherwise, returns False.

Remarks

The amount of the zoom is controlled by the ZoomOutPercentage property.

aaHistClientTrend Events

The following are the events used by the aaHistClientTrend:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- CurrentTagChanged
- DatesChanged
- StateChanged
- TagDisplayChanged
- TaglistChanged

CurrentTagChanged

The CurrentTagChanged event is triggered when a different tag is selected in the Tag List.

Syntax

```
aaHistClientTrend.CurrentTagChanged(message serverName, message tagName, integer TagType);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

tagType

The type of tag.

Remarks

For more information on the tag type, see aaTagType Enumeration.

To retrieve the value of an event parameter in the InTouch HMI software, refer to `#ThisEvent.<Event Name><Parameter Name>` inside the script for the respective event. For example, to read the value of the *tagName* parameter, use a statement like the following:

```
MyMsgTag = #ThisEvent.CurrentTagChangedtagName;
```

DatesChanged

The DatesChanged event is triggered when the date for the trend changes. It is also triggered once when the live or replay modes are started, but not on the automatic updates that follow.

Syntax

```
aaHistClientTrend.DatesChanged();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

StateChanged

The StateChanged event is triggered when a change has been made to the configuration for a tag in the Tag List.

Syntax

```
aaHistClientTrend.StateChanged();
```

TagDisplayChanged

The TagDisplayChanged event is triggered when the display options for a tag in the Tag List are changed. This includes the following actions:

- Showing or hiding the tag
- Changing the type, color, width, or style of the tag's trend curve
- Changing the tag's value format or precision
- Changing the tag's time offset
- Changing the tag's scale
- Editing the tag's target region

Syntax

```
aaHistClientTrend.TagDisplayChanged(message serverName, message tagName, integer
displayItem);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag.

displayItem

The identifier for the displayed trend item.

Remarks

To retrieve the value of an event parameter in the InTouch HMI software, refer to `#ThisEvent.<Event Name><Parameter Name>` inside the script for the respective event. For example, to read the value of the *tagName* parameter, use a statement like the following:

```
MyMsgTag = #ThisEvent.TagDisplayChangedtagName;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TaglistChanged

The TaglistChanged event is triggered when a tag is added or removed from the Tag List.

Syntax

```
aaHistClientTrend.TaglistChanged();
```

aaHistClientTrend Enumerations

The aaHistClientTrend enumerations include:

- aaChartType Enumeration
- aaDashStyle Enumeration
- aaDataPointLabelingType Enumeration
- aaDateModeEnumeration Enumeration
- aaInterpolationType Enumeration
- aaQualityRules Enumeration
- aaRetrievalMode Enumeration
- aaRetrievalVersion Enumeration
- aaStateCalculation Enumeration
- aaTargetRegionExcursionType Enumeration
- aaTimeStampRules Enumeration
- aaTraceGradientType Enumeration
- aaTrendGradientType Enumeration
- aaTrendType Enumeration
- aaTrendValueFormat Enumeration
- aaValueAxisLabelEnumeration Enumeration

aaChartType Enumeration

An enumeration used to specify the chart type.

Value	Enumeration	Description
0	Trend	Regular trend.
1	XYScatterPlot	XY scatter plot.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaDashStyle Enumeration

An enumeration used to specify the line style.

Value	Enumeration	Description
0	Solid	Specifies a solid line.
1	Dash	Specifies a line consisting of dashes.
2	Dots	Specifies a line consisting of dots.
3	Dash Dot	Specifies a line consisting of a repeating pattern of dash-dot.
4	Dash Dot Dot	Specifies a line consisting of a repeating pattern of dash-dot-dot.

aaDataPointLabelingType Enumeration

An enumeration used to specify the type of labels that are shown next to data points on a chart.

Value	Enumeration	Description
0	None	No labels.
1	TimeLabelsOnCurrentTag	Time labels on the currently selected tag, evenly spaced in time.

aaDateModeEnumeration Enumeration

An enumeration used to specify the time mode for the trend chart.

Value	Enumeration	Description
0	Absolute	Use absolute time.
1	Relative	Use relative time.

aaInterpolationType Enumeration

Specifies the interpolation type for data retrieval.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
0	Stairstep	Use stair-step interpolation.
1	Linear	Use linear interpolation.
2	ApplicationSetting	Use the default interpolation type specified at the control level. This value is only valid at the tag level, but not at the control level itself.
3	ServerDefault	Use the default interpolation type specified at the AVEVA Historian level.

For more information on each option, see Interpolation Type (wwInterpolationType).

aaQualityRules Enumeration

Specifies the quality rule for data retrieval.

Value	Enumeration	Description
0	Good and Uncertain quality	Include data values with uncertain quality in calculations.
1	Good quality	Exclude data values with uncertain quality from calculations.
2	ApplicationSetting	Use the default quality rule specified at the control level. This value is only valid at the tag level, but not at the control level itself.
3	ServerDefault	Use the default quality rule specified at the AVEVA Historian level.
4	Estimate when values are missing	Include some good and some NULL values. Do not cause the overall calculations to return NULL.

For more information on each option, see Quality Rule (wwQualityRule).

aaRetrievalMode Enumeration

Specifies the data retrieval mode.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
0	Cyclic	Use Cyclic retrieval mode.
1	Delta	Use Delta retrieval mode.
2	Full	Use Full retrieval mode.
3	Interpolated	Use Interpolated retrieval mode.
4	BestFit	Use "Best Fit" retrieval mode.
5	Average	Use Time-Weighted Average retrieval mode.
6	Min	Use Minimum retrieval mode.
7	Max	Use Maximum retrieval mode.
8	Integral	Use Integral retrieval mode.
9	Slope	Use Slope retrieval mode.
10	Counter	Use Counter retrieval mode.
11	ValueState	Use ValueState retrieval mode.
12	RoundTrip	Use RoundTrip retrieval mode.
13	ApplicationSetting	Use the default retrieval mode specified at the control level. This value is only valid at the tag level, but not at the control level itself.

For more information on each option, see Understanding Retrieval Modes.

aaRetrievalVersion Enumeration

Specifies the history version to retrieve data from.

Value	Enumeration	Description
0	Latest	Retrieve the latest values available for a tag.
1	Original	Retrieve the original values historized for a tag.

For more information on each option, see History Version (wwVersion).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaStateCalculation Enumeration

Specifies the aggregation type to use in Time-in-State data retrieval.

Value	Enumeration	Description
0	Min	The shortest amount of time that the tag was in each unique state over the query period.
1	Max	The longest amount of time that the tag was in each unique state over the query period.
2	Average	The average amount of time that the tag was in each unique state over the query period.
3	Total	The total amount of time that the tag was in each unique state over the query period.
4	Percent	The total percentage of time that the tag was in each unique state over the query period.
5	AvgContained	The average amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
6	MinContained	The shortest amount of time each tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
7	MaxContained	The longest amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
8	TotalContained	The total amount of time that the tag has been in each unique state

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
		for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
9	PercentContained	The percentage of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
10	ApplicationSetting	Use the default aggregation type specified at the control level. This value is only valid at the tag level, but not at the control level itself.

For more information on each option, see State Calculation (wwStateCalc).

aaTargetRegionExcursionType Enumeration

An enumeration used to specify whether values that fall outside a tag's target region should be highlighted.

Value	Enumeration	Description
0	None	Do not highlight values.
1	ShowWithSpecialColor	Highlight values in a special color.

aaTimeStampRules Enumeration

Specifies the timestamp rule for data retrieval.

Value	Enumeration	Description
0	Start	Query results are timestamped at the beginning of each cycle.
1	End	Query results are timestamped at the end of each cycle.
2	ApplicationSetting	Use the default timestamp rule specified at the control level. This value is only valid at the tag level, but not at the control level itself.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
3	ServerDefault	Use the default timestamp rule specified at the AVEVA Historian level.

For more information on each option, see Timestamp Rule (wwTimestampRule).

aaTraceGradientType Enumeration

An enumeration used to specify the gradient type applied to the trace in a scatter plot.

Value	Enumeration	Description
0	None	No gradient.
1	OpacityGradient	Opacity gradient from the start to the end of the trace.

aaTrendGradientType Enumeration

Specifies the gradient type for the plot area and the background for a trend.

Value	Enumeration	Description
0	None	No gradient.
1	LeftRight	Gradient from left to right.
2	TopBottom	Gradient from top to bottom.
3	Center	Gradient from center outwards.
4	DiagonalLeft	Gradient from top left to bottom right.
5	DiagonalRight	Gradient from top right to bottom left.
6	HorizontalCenter	Gradient from center to left and right edges.
7	VerticalCenter	Gradient from the center to top and bottom edges.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaTrendType Enumeration

Specifies the type of line for the trend curve.

Value	Enumeration	Description
0	Point	No line.
1	Line	A straight line is drawn directly from point to point on the trend.
2	StepLine	The line is drawn horizontally to the next point and then vertically up (if ascending) or down (if descending).
3	Auto	Automatically determine the curve type.

For more information on each option, see Configuring Display Options.

aaTrendValueFormat Enumeration

Specifies the value display format of the trend value.

Value	Enumeration	Description
0	Decimal	The decimal format.
1	Scientific	The scientific format.

aaUpdateToCurrentTimeState Enumeration

Specifies the state of the **Update to Current Time** option.

Value	Enumeration	Description
0	Reset	Option is disabled. The corresponding toolbar button is not highlighted.
1	Set	Option is enabled. The corresponding toolbar button is highlighted.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaValueAxisLabelEnumeration Enumeration

Specifies the value display format of the trend value. For more information on the types of scales, see Scaling Tags.

Value	Enumeration	Description
0	MultipleScales	Show multiple value scales on the chart.
1	SingleScale	Show single value scale on the chart.
2	ValuesAtCursor	Show data values at the point at which the cursor intersects the data.
3	NoScale	Show no chart label, X and Y axes scales and cursor information.

aaHistClientQuery Control

The aaHistClientQuery control allows you to run the AVEVA Historian Client Query program (or a functional subset) from within the AVEVA InTouch HMI software or a .NET container like Visual Basic .NET or Internet Explorer.

For more information on using the AVEVA Historian Client Query, see AVEVA Historian Client Query.

Using aaHistClientQuery at Runtime

At runtime, aaHistClientQuery can retrieve data from the AVEVA Historian database and return the results in a table format. You can use aaHistClientQuery as you do the AVEVA Historian Client Query application.

For more information on using the AVEVA Historian Client Query, see AVEVA Historian Client Query.

Using aaHistClientQuery in an Application

aaHistClientQuery is capable of running with all of the functionality of the AVEVA Historian Client Query application. You can also use the aaHistClientQuery control's properties, methods, and events in runtime scripts in your application to control the functionality that is available to the runtime user.

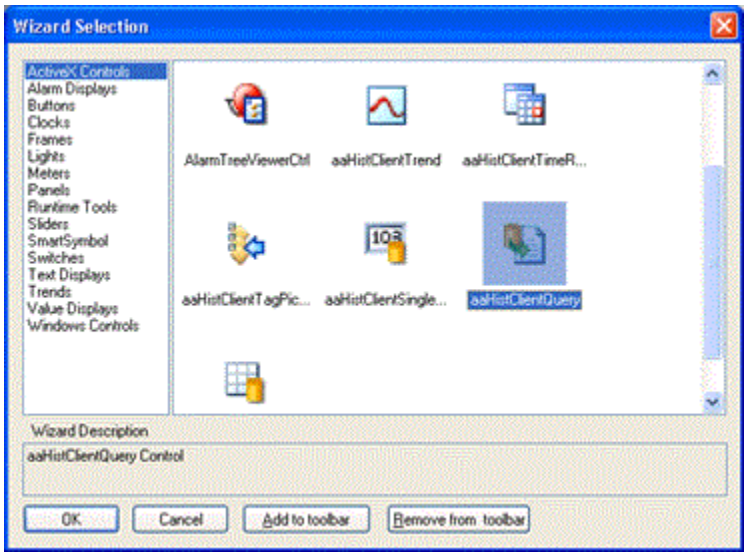
For example, maybe you want to limit the functionality of aaHistClientQuery to only allow the runtime operator to connect to an AVEVA Historian and run a particular query for a specific set of tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

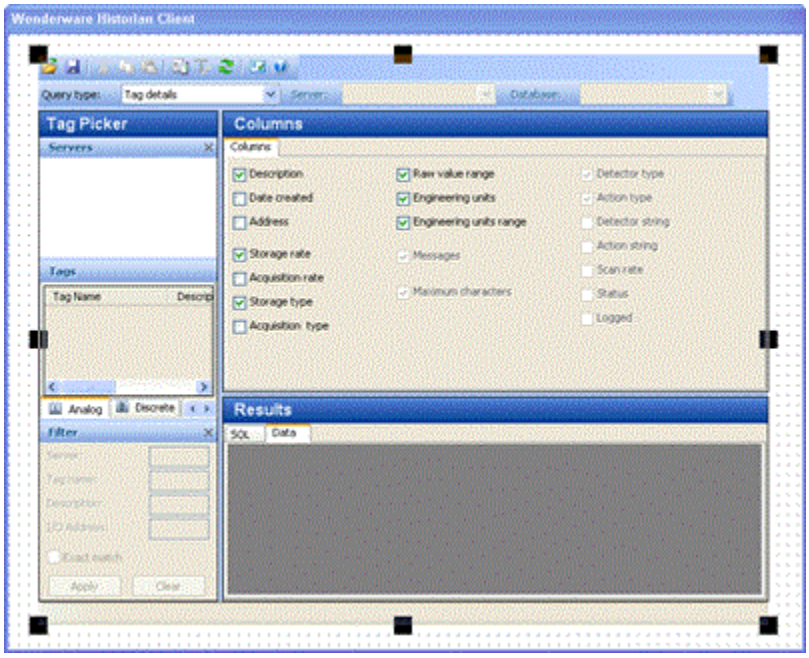
Adding aaHistClientQuery to an InTouch Window

To add the aaHistClientQuery control:

1. In WindowMaker, click the **Wizards** button . The **Wizard Selection** dialog box appears.



2. Select the aaHistClientQuery control.
3. Click **OK**. The control appears in the window.



aaHistClientQuery Properties

The properties for the aaHistClientQuery control are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- ActiveServer
- AllowQueryTypeChange
- CurrentServer
- EnableAllQueriesTab
- FavoriteQueriesFolder
- FontBold
- FontCharset
- FontItalic
- FontName
- FontSize
- LockDown
- QueryFont
- QueryString
- Recordset
- Servers
- ToolbarConnectVisible
- ToolbarEditVisible
- ToolbarRequeryVisible
- ToolbarVisible
- UsePersistedServers

ActiveServer

ActiveServer

The ActiveServer property is a read-write property that sets or gets the name of the server to which the aaHistClientQuery is connected.

Syntax

```
aaHistClientQuery.ActiveServer = message;
Result = aaHistClientQuery.ActiveServer;
```

Return Value

The name of the server as a message. If there are no active servers, this property returns a NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property has no default value.

AllowQueryTypeChange

AllowQueryTypeChange

The AllowQueryTypeChange property is a read-write property that gets or sets whether the run-time user is allowed to change the query type.

Syntax

```
aaHistClientQuery.AllowQueryTypeChange = discrete;
Result = aaHistClientQuery.AllowQueryTypeChange;
```

Remarks

The default value is True.

CurrentServer

CurrentServer

The CurrentServer property is a read-only property that returns the aaServer object of the server to which the aaHistClientQuery is connected.

Syntax

```
Result = aaHistClientQuery.CurrentServer;
```

Remarks

If there are no active servers, this property returns a NULL. For more information on the aaServer object, see aaServer Object.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

EnableAllQueriesTab

EnableAllQueriesTab

The EnableAllQueriesTab property is a read-write property that shows or hides the **All Queries** tab in the **Results** pane.

Syntax

```
aaHistClientQuery.EnableAllQueriesTab = discrete;
Result = aaHistClientQuery.EnableAllQueriesTab;
```

Remarks

The default value is False.

FavoriteQueriesFolder

FavoriteQueriesFolder

The FavoriteQueriesFolder property is a read-write property that gets or sets the location of the favorite queries folder.

Syntax

```
aaHistClientQuery.FavoriteQueriesFolder = message;
Result = aaHistClientQuery.FavoriteQueriesFolder;
```

Remarks

When the FavoriteQueriesFolder property is set, the query file list in the corresponding folder is transferred to the **Favorite Queries** list box.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

FontBold

FontBold

The FontBold property is a read-write property that gets or sets the boldface characteristic for the font used for displaying the query text in the **SQL** and **All Queries** tab in the **Results** pane.

Syntax

```
aaHistClientQuery.FontBold = discrete;
Result = aaHistClientQuery.FontBold;
```

Remarks

True = Use bold; False = Do not use bold.

The default value is False.

FontCharset

FontCharset

The FontCharset property is a read-write property that gets or sets the character set used for the query and result text.

Syntax

```
aaHistClientQuery.FontCharset = integer;
Result = aaHistClientQuery.FontCharset;
```

Remarks

This property is an integer value that specifies the character set used by the font. The following are some common settings for the value:

Value	Description
0	The standard Windows character set (ASCII).
1	The system default character set.
2	The symbol character set.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Description
77	Characters used by Macintosh.
128	The Japanese character set.
129, 130	Korean character set.
134	The Chinese character set used in mainland China (Simplified Chinese)
136	The Chinese character set used mostly in Hong Kong SAR and Taiwan (Traditional Chinese).
161	The Greek character set.
162	The Turkish character set.
163	The Vietnamese character set.
177	The Hebrew character set.
178	The Arabic character set.
204	The Russian character set.
222	The Thai character set.
238	The Eastern European character set.
255	The extended ASCII character set used with DOS and some Microsoft® Windows® fonts.

The default value is 1.

FontItalic

FontItalic

The FontItalic property is a read-write property that gets or sets whether the query text appears in an italicized font.

Syntax

```
aaHistClientQuery.FontItalic = discrete;
Result = aaHistClientQuery.FontItalic;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

True = Use italics; False = Do not use italics.

The default value is False.

FontName

FontName

The FontName property is a read-write property that gets or sets the name of the font family used for the query text.

Syntax

```
aaHistClientQuery.FontName = message;
Result = aaHistClientQuery.FontName;
```

Remarks

The default value is Tahoma.

FontSize

FontSize

The FontSize property is a read-write property that gets or sets the size, in points, of the font used for displaying the query text.

Syntax

```
aaHistClientQuery.FontSize = integer;
Result = aaHistClientQuery.FontSize;
```

Remarks

The default value is 8.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LockDown

LockDown

The LockDown property is a read-write property that enables or disables a "lock down" mode in the control.

Syntax

```
aaHistClientQuery.LockDown = discrete;
Result = aaHistClientQuery.LockDown;
```

Remarks

In the "lock down" mode, the following features are not available to the run-time user:

- Tag Picker
- Main toolbar

The default value is False.

QueryFont

QueryFont

The QueryFont property is a read-write property that gets or sets the font used for displaying the query text.

Syntax

```
aaHistClientQuery.QueryFont = Font;
Result = aaHistClientQuery.QueryFont;
```

Remarks

This property is not accessible in the InTouch HMI software.
For more information on setting the font, see Font.
The default font is Tahoma, 8 point (for English versions).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

QueryString

QueryString

The QueryString property is a read-write property that gets or sets the query string.

Syntax

```
aaHistClientQuery.QueryString = message;
Result = aaHistClientQuery.QueryString;
```

Remarks

If you set the QueryString property, then the query type is automatically set to Custom.

This property has no default.

Recordset

Recordset

The Recordset property is a read-only property that gets the data set for the query.

Syntax

```
DataSet = aaHistClientQuery.Recordset;
```

Return Value

Returns a DataSet object. For more information on data sets, see DataSet.

Remarks

This property is not accessible in the InTouch HMI software.

This property has no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Servers

Servers

The Servers property is a read-write property that gets or sets the list of servers.

Syntax

```
aaHistClientQuery.Servers = aaServers;
Result = aaHistClientQuery.Servers;
```

Remarks

This property uses the aaServers object. For more information on the aaServers object, see aaServers Object on page 610.

This property has no default.

ToolbarConnectVisible

ToolbarConnectVisible

The ToolbarConnectVisible property is a read-write property that shows or hides the server connection toolbar button.

Syntax

```
aaHistClientQuery.ToolbarConnectVisible = discrete;
Result = aaHistClientQuery.ToolbarConnectVisible;
```

Remarks

The default is True.

ToolbarEditVisible

ToolbarEditVisible

The ToolbarEditVisible property is a read-write property that shows or hides the cut, copy, and paste toolbar

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

buttons.

Syntax

```
aaHistClientQuery.ToolbarEditVisible = discrete;
Result = aaHistClientQuery.ToolbarEditVisible;
```

Remarks

The default is True.

ToolbarRequeryVisible

ToolbarRequeryVisible

The ToolbarRequeryVisible property is a read-write property that shows or hides the re-query (refresh) toolbar button.

Syntax

```
aaHistClientQuery.ToolbarRequeryVisible = discrete;
Result = aaHistClientQuery.ToolbarRequeryVisible;
```

Remarks

The default is True.

ToolBarVisible

ToolBarVisible

This read-write property shows or hides the entire toolbar.

Syntax

```
aaHistClientQuery.ToolbarVisible = discrete;
Result = aaHistClientQuery.ToolbarVisible;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The default is True, that is, the toolbar is visible.

UsePersistedServers

UsePersistedServers

This read-write property controls whether changes to the control's server connections are only valid for the current runtime session, or whether they are saved to the global server list shared by the AVEVA Historian Client applications.

Syntax

```
aaHistClientQuery.UsePersistedServers = discrete;
Result = aaHistClientQuery.UsePersistedServers;
```

Remarks

If you set this property to True, changes to the configured server connections are saved in the global server list. If you set it to False, changes do not affect the global server list.

For example, if you add a server while this property is set to True, the server is added to the global list. If you set the property to False and remove the same server, it disappears from the server list for the current runtime session, but it is not deleted from the global list.

The default is False. To initialize the control with the server connections stored in the global list, set the value to True. You can set it back to False afterwards to avoid inadvertent changes by the run-time user.

For more information on managing servers, see Server Connection Configuration.

aaHistClientQuery Methods

The aaHistClientQuery methods are:

- AddServer
- AddServerEx
- AddTag
- ClearTags
- CopyQuery
- CutQuery
- FileOpen
- ManualConnect

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- OpenQuery
- PasteQuery
- RemoveTag
- Refresh
- SaveQuery
- SaveResults
- SetDates
- SetDuration
- SetQueryType
- SetQueryType2
- SetTimeSpan
- ShowAbout

AddServer

AddServer

The AddServer method adds a server to the list.

Syntax

```
[Result=] aaHistClientQuery.AddServer(message serverName, message loginName, message
password, [discrete bPersistPassword]);
```

Parameters

serverName

The name of the server.

loginName

A valid user name to log on to the server. If no login name is provided, Windows integrated security is used.

password

A valid password for the server.

bPersistPassword

Optional parameter. If set to True, the password is remembered for the next time a connection is attempted. The password is only remembered for single application; the persisted password is not available to all applications. The default value is True.

Return Value

Returns True if the server can be added; otherwise returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AddServerEx

AddServerEx

The AddServerEx method adds a server to the list.

Syntax

```
[Result=] aaHistClientSingleValueEntry.AddServerEx(message serverName, message loginName,
message password, [discrete bPersistPassword]);
```

Parameters

serverName

The name of the server to which to connect.

loginName

A valid user name to log on to the server. If no login name is provided, Windows integrated security is used.

password

A valid password for the server.

bPersistPassword

If set to True, the password is remembered for the subsequent connection attempt. The password is only remembered for single application; the persisted password is not available to all applications.

Return Value

Returns True if the server can be added to the list; otherwise returns False.

Remarks

All parameters are required. Errors, if any, are reported.

AddTag

AddTag

The AddTag method adds a tag to the tag collection.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientQuery.AddTag(message serverName, message tagName, integer tagType);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag to add.

tagType

The type of the tag. This parameter is provided for backward compatibility and does not have any effect on the outcome of the operation. However, you must still specify one of the following valid values: 1, 2, 3, or 5.

Return Value

Returns True if the tag can be added; otherwise, returns False.

ClearTags

ClearTags

The ClearTags method removes all of the tags from the query.

Syntax

```
[Result=] aaHistClientQuery.ClearTags();
```

Example

In the following example, all tags from the query are deleted, and the ReactLevel tag is added to the query.

```
#aaHistClientQuery1.ClearTags;
#aaHistClientQuery1.AddTag("MyInSQL", "ReactLevel", 1);
```

CopyQuery

CopyQuery

The CopyQuery method copies the current selection in the query text box to the clipboard.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientQuery.CopyQuery();
```

CutQuery

CutQuery

The CutQuery method deletes the current selection in the query text box and then copies it to the clipboard.

Syntax

```
[Result=] aaHistClientQuery.CutQuery();
```

FileOpen

FileOpen

The FileOpen method opens a specified text file containing a SQL query.

Syntax

```
[Result=] aaHistClientQuery.FileOpen(message fileName);
```

Parameters

fileName
The full path to the file.

Remarks

When this method is called, it automatically sets the query type to **Custom**. If the **SQL** tab is active at the time the method is called, the method loads the SQL query from the file into the **SQL** tab, but does not send it to the server. If the **Data** tab is active, the method loads the query into the **SQL** tab, sends it to the currently selected server, and shows the results on the **Data** tab.

Return Value

Returns True if the file can be opened successfully; otherwise returns False (for example, if no file name is specified or the specified file does not exist).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ManualConnect

ManualConnect

The ManualConnect method opens the **Server connection** dialog box.

Syntax

```
[Result=] aaHistClientQuery.ManualConnect();
```

OpenQuery

OpenQuery

The OpenQuery method opens the **Open** dialog box, so that the runtime user can select an existing query file (.sql) to open.

Syntax

```
[Result=] aaHistClientQuery.OpenQuery();
```

PasteQuery

PasteQuery

The PasteQuery method pastes the current contents of the clipboard to the query text box.

Syntax

```
[Result=] aaHistClientQuery.PasteQuery();
```

Refresh

Refresh

The Refresh method re-executes the query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientQuery.Refresh();
```

Remarks

The focus must be on the Results tab for this method to take effect.

RemoveTag

RemoveTag

The RemoveTag method removes the specified tag from the query.

Syntax

```
[Result=] aaHistClientQuery.RemoveTag(message serverName, message tagName);
```

Parameters

serverName

The name of the server.

tagName

The name of the tag to remove.

Return Value

Returns True if the tag was found and can be removed; otherwise, returns False.

SaveQuery

SaveQuery

The SaveQuery method opens the **Save As** dialog box, so that the runtime user can save the current query to a text file.

Syntax

```
[Result=] aaHistClientQuery.SaveQuery();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SaveResults

SaveResults

The SaveResults method opens the **Save As** dialog box, so that the runtime user can save the current **Data** tab contents to a .txt or .csv file.

Syntax

```
[Result=] aaHistClientQuery.SaveResults();
```

SetDates

SetDates

The SetDates method sets the start and end time for the query.

Syntax

```
[Result=] aaHistClientQuery.SetDates(DateTime startTime, DateTime endTime);
```

Parameters

startTime

The start time for the query.

endTime

The end time for the query.

Remarks

For more information on setting the date/time, see DateTime.

Return Value

Returns True if the dates were set. Returns False in case of an error.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SetDuration

SetDuration

The SetDuration method sets the query period as a duration relative to the current time.

Syntax

```
[Result=] aaHistClientQuery.SetDuration(real duration);
[Result=] aaHistClientQuery.SetDuration(DateTime duration);
```

Parameters

duration

The duration from the current time.

Remarks

When using the ActiveX version of the control (for example, in the InTouch HMI software), the duration parameter can be either a number of days or a date/time string.

When using the .NET version of the control, the duration parameter must be a valid DateTime value.

In both cases, when you specify a date/time value, the duration is the difference between the specified date/time and the base date of December 30th, 1899, 12:00:00 AM.

For more information on the format for date/time values, see DateTime.

Example

In the following example, the time period is set to the past five minutes, relative to the current time.

```
#aaHistClientQuery1.SetDuration("00:05:00");
```

In the following example, the time period is set to the past 36 hours by specifying the number of days.

```
#aaHistClientQuery1.SetDuration(1.5);
```

In the following example, the time period is set to the past 36 hours by specifying a date/time value.

```
#aaHistClientQuery1.SetDuration("12/31/1899 12:00:00");
```

SetQueryType

SetQueryType

The SetQueryType method selects the specified query type and tag type in the Tag Picker.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientQuery.SetQueryType(aaQueryTypeEnumeration queryType, aaTagType
tagType);
```

Parameters

queryType

The type of the query. For information on the valid enumerations, see aaQueryTypeEnumeration.

tagType

The type of the tag. For information on the valid enumerations, see aaTagType Enumeration.

Return Value

Returns True if it can be shown; otherwise, returns False.

Remarks

This method is not accessible in the InTouch HMI software. Use the SetQueryType2 method instead.

SetQueryType2

SetQueryType2

The SetQueryType2 method selects the specified query type and tag type in the Tag Picker.

Syntax

```
[Result=] aaHistClientQuery.SetQueryType2(integer queryType, integer tagType);
```

Parameters

queryType

The type of the query. For information on the valid values, see aaQueryTypeEnumeration.

tagType

The type of the tag. For information on the valid values, see aaTagType Enumeration.

Return Value

Returns True if it can be shown; otherwise, returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

Use this method in the InTouch HMI software instead of the SetQueryType method.

SetTimeSpan

SetTimeSpan

The SetTimeSpan method sets the start and end time for the query.

Syntax

```
[Result=] aaHistClientQuery.SetTimeSpan(DateTime start, DateTime end,
aaTimeRangeEnumeration duration);
```

Parameters

startTime

The start time for the query.

endTime

The end time for the query.

duration

The time duration, either Custom or an enumerated set.

Return Value

Returns True if the time span can be set; otherwise, returns False.

Remarks

The times can be specified as a duration (Last5Minutes, Last24Hours, etc.) or as a pair of start and end values, in which case the duration must be specified as Custom.

For more information on setting the date/time, see DateTime. For more information on setting the duration, see aaTimeRangeEnumeration Enumeration.

ShowAbout

ShowAbout

The ShowAbout method opens the **About** dialog box.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientQuery.ShowAbout();
```

aaHistClientQuery Events

The aaHistClientQuery events are:

- ModeChanged
- QueryChanged
- ServerChanged

ModeChanged

ModeChanged

The ModeChanged event is triggered when the run-time user changes tabs on the **Results** pane in the control.

Syntax

```
aaHistClientQuery.ModeChanged(integer mode);
```

Parameters

mode

The type of tab for which changes are detected. 0 = The focus has changed to the **Query** or **All Queries** tab; 1 = The focus has changed to the **Results** tab.

Remarks

To retrieve the value of an event parameter in the InTouch HMI software, refer to `#ThisEvent.<Event Name><Parameter Name>` inside the script for the respective event. For example, to read the value of the *mode* parameter, use a statement like the following:

```
MyIntTag = #ThisEvent.ModeChangedmode;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

QueryChanged

QueryChanged

The QueryChanged event is triggered when the query is changed.

Syntax

```
aaHistClientQuery.QueryChanged();
```

Remarks

When the query changes as a result of a user action with the control (not as a result of entering text), or as a result of changing the query type, the control triggers a query changed event, unless the query is of Custom type. For a Custom query, the change event is triggered each time the user changes the text. The change event is also triggered when the user sets the QueryString property.

ServerChanged

ServerChanged

The ServerChanged event is triggered when the server is changed.

Syntax

```
aaHistClientQuery.ServerChanged();
```

Remarks

This event is triggered when a logon has successfully completed.

aaQueryTypeEnumeration

Used for specifying the various types of queries for the aaHistClientQuery control.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
0	TagDetails	Retrieve configuration details for the specified tags.
1	LiveValues	Retrieve the real-time value of the specified tags.
2	HistoryValues	Retrieve the history of the tag values over time for the specified tags. Allows control over the format, and all of the time domain extensions for the AVEVA Historian.
3	AggregateValues	Retrieve aggregated values of the specified tags. For example, minimum, maximum, sum, and average.
4	SummaryValues	Retrieve the values calculated by the summary system of the specified tags.
5	EventHistoryValues	Retrieve when specified events have occurred in history.
6	EventSnapshot	Retrieves the values of tags associated with events at the time that the events occurred.
7	AlarmLimits	Retrieve information about the limits configured for analog tags.
8	TagSearch	Search for tags.
9	Custom	Indicates to create a custom query.
10	Annotations	Retrieve comments regarding data points.
11	Favorite	Indicates to use a pre-existing SQL query.
12	AlarmHistory	Retrieve alarm data based on limits configured using the AVEVA Historian.
13	ServerVersion	Retrieve the server version.
14	StorageStartDate	Retrieve the start date of data storage.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
15	TimeRunning	Retrieve the amount of time the server has been running.
16	NumberOfTags	Retrieve a tag count for various kinds of tags.
17	StorageSizeAvailable	Retrieve storage size availability information.
18	IOServer	Retrieve information regarding the specified I/O server(s).
19	Storage	Retrieve storage details.
20	AnalogSummary Values	Retrieves summary values for analog tags.
21	StateSummary Values	Retrieves summary values of the different states of tags.

aaHistClientTagPicker Control

The aaHistClientTagPicker control allows you to view the hierarchy of objects in an AVEVA Historian database (for example, tags, InTouch nodes, events, and so on) in a hierarchical format.

For more information on using the aaHistClientTagPicker, see Tag Picker.

Using aaHistClientTagPicker at Runtime

The aaHistClientTagPicker control functions the same as the Tag Picker that appears in the Trend and Query applications.

For more information on using the Tag Picker, see Tag Picker.

Using aaHistClientTagPicker in an Application

Use the aaHistClientTagPicker control's properties, methods, and events to create scripts that set up a database connection and customize how the aaHistClientTagPicker control behaves during runtime. For example, you can configure the **Filter** pane so that does not appear during runtime.

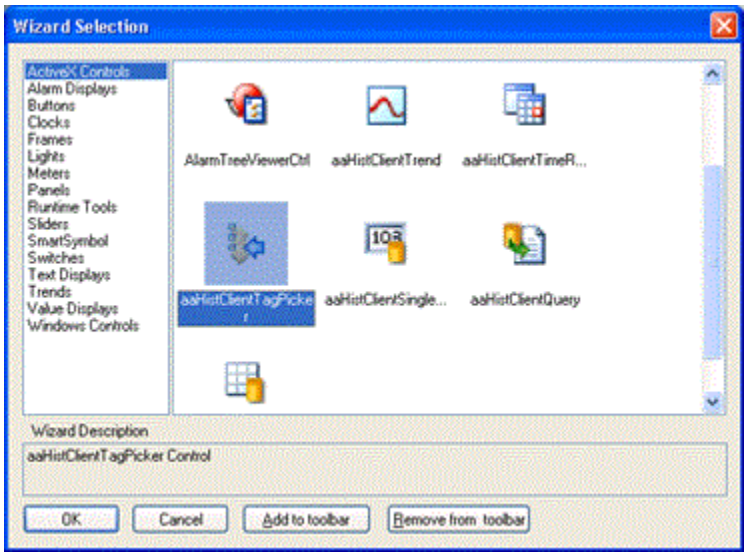
All properties, methods, and events can be controlled through scripting. In addition, some of these properties and methods are exposed through the aaHistClientTagPicker property panel available during application development.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding aaHistClientTagPicker to an InTouch Window

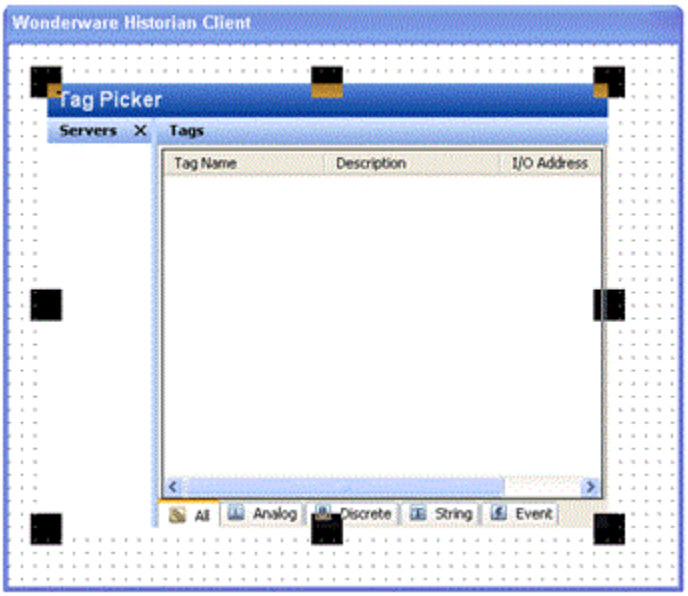
To add the aaHistClientTagPicker control:

1. In WindowMaker, click the **Wizards** button . The **Wizard Selection** dialog box appears.



2. Select the aaHistClientTagPicker control.
3. Click **OK**.

   The control appears in the window.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaHistClientTagPicker Properties

The aaHistClientTagPicker properties are:

- CurrentServer
- DescriptionFilter
- ExactMatchFilter
- FilterVisible
- HideCaption
- IOAddressFilter
- SelectedPath
- SelectedTagCount
- Servers
- SingleSelectMode
- SplitterOrientation
- TabSelectedIndex
- TagNameFilter
- TagSelectedIndex
- TreeVisible
- TreeWidth
- UseHierarchicalName
- Visible

CurrentServer

CurrentServer

The CurrentServer property is a read-write property that gets or sets the selected server in the **Servers** pane.

Syntax

```
aaHistClientTagPicker.CurrentServer = aaServer;
Result = aaHistClientTagPicker.CurrentServer;
```

Remarks

The current server determines the tags that appear in the **Tags** pane. This property uses the aaServer object. For more information, see aaServer Object.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DescriptionFilter

DescriptionFilter

The DescriptionFilter property is a read-write property that gets or sets the description filter criteria.

Syntax

```
aaHistClientTagPicker.DescriptionFilter = message;
Result = aaHistClientTagPicker.DescriptionFilter;
```

Remarks

The description filter criteria is applied when the ApplyFilter method is called or when the **Apply** button is clicked by the run-time user.

The default is an empty message value ("").

ExactMatchFilter

ExactMatchFilter

The ExactMatchFilter property is a read-write property that gets or sets whether or not the filter criteria must be an exact match.

Syntax

```
aaHistClientTagPicker.ExactMatchFilter = discrete;
Result = aaHistClientTagPicker.ExactMatchFilter;
```

Remarks

The default value is False.

FilterVisible

FilterVisible

The FilterVisible property is a read-write property that shows or hides the **Filter** pane.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTagPicker.FilterVisible = discrete;
Result = aaHistClientTagPicker.FilterVisible;
```

Remarks

The default value is False.

HideCaption

HideCaption

The HideCaption property is a read-write property that hides or shows the caption at the top of the Tag Picker.

Syntax

```
aaHistClientTagPicker.HideCaption = discrete;
Result = aaHistClientTagPicker.HideCaption;
```

Remarks

The default value is False, that is, the caption is shown.

IOAddressFilter

IOAddressFilter

The IOAddressFilter property is a read-write property that gets or sets the I/O address filter criteria.

Syntax

```
aaHistClientTagPicker.IOAddressFilter = message;
Result = aaHistClientTagPicker.IOAddressFilter;
```

Remarks

The default is an empty message value ("").

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SelectedPath

SelectedPath

Use this read-write property to return the path of the currently selected folder or to show only a specific part of the folder structure on an AVEVA Historian.

Syntax

```
aaHistClientTagPicker.SelectedPath = message;
Result = aaHistClientTagPicker.SelectedPath;
```

Remarks

This property serves two purposes:

- When you read this property, the path of the currently selected folder in the **Servers** pane is returned. For example, if the "All Analog Tags" folder in the "Public Groups" folder on the Server1 host is selected, this property returns `Server1.Public Groups.All Analog Tags`.
- When you write to this property, the Tag Picker only displays the contents of the specified path for a server. For example, if you set this property to `Server1.Public Groups`, the **Servers** pane only shows the contents of the "Public Groups" folder for the Server1 host. To show all folders on a server again, set the property to the server name. For example, to show all folders on the Server1 host, set this property to `Server1`.

Values are case-sensitive if the AVEVA Historian is installed on a case-sensitive SQL Server.

SelectedTagCount

SelectedTagCount

This read-only property gets the total count of tags that are selected in the Tag Picker.

Syntax

```
Result = aaHistClientTagPicker.SelectedTagCount;
```

Remarks

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Servers

Servers

This read-write property gets or sets the list of servers.

Syntax

```
aaHistClientTagPicker.Servers = aaServers;
Result = aaHistClientTagPicker.Servers;
```

Remarks

This property uses the aaServers object. For more information, see aaServer Object.

This property has no default value.

Example: Login

The following InTouch HMI software example adds the server MyInSQL1 to the Tag Picker and logs on to the server:

```
%NewServer = #aaHistClientTagPicker1.Servers.Add("MYINSQL1");
%NewServer.LoginID = "wwAdmin";
%NewServer.Password = "wwAdmin";
#aaHistClientTagPicker1.LogOn( %NewServer );
```

SingleSelectMode

SingleSelectMode

The SingleSelectMode property is a read-write property that enables or disables only single tag at a time to be selected from the list of tags.

Syntax

```
aaHistClientTagPicker.SingleSelectMode = discrete;
Result = aaHistClientTagPicker.SingleSelectMode;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The default value is False.

SplitterOrientation

SplitterOrientation

The SplitterOrientation property is a read-write property that controls whether the splitter bar that divides the **Tags** pane from the **Servers** pane is vertical or horizontal.

Syntax

```
aaHistClientTagPicker.SplitterOrientation = aaHistClientTagPickerSplitterOrientation;
Result = aaHistClientTagPicker.SplitterOrientation;
```

Remarks

This aaHistClientTagPickerSplitterOrientation enumeration is used for the orientation types. For more information, see aaHistClientTagPickerSplitterOrientation Enumeration.

The default value is 0 (horizontal).

TabSelectedIndex

TabSelectedIndex

The TabSelectedIndex is a read-only property that returns the index of the currently selected tab in the Tag Picker. The index starts from zero.

Syntax

```
aaHistClientTagPicker.TabSelectedIndex = integer;
Result = aaHistClientTagPicker.TabSelectedIndex;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagNameFilter

TagNameFilter

The TagNameFilter property is a read-write property that gets or sets the tagname filter criteria.

Syntax

```
aaHistClientTagPicker.TagNameFilter = message;
Result = aaHistClientTagPicker.TagNameFilter;
```

Remarks

The default is an empty message value ("").

TagSelectedIndex

TagSelectedIndex

The TagSelectedIndex is a read-only property that returns the index of the currently selected tag in the Tag Picker. The index starts from zero.

Syntax

```
aaHistClientTagPicker.TagSelectedIndex = integer;
Result = aaHistClientTagPicker.TagSelectedIndex;
```

Remarks

If multiple tags are selected, this property returns the index of the first selected tag.

TreeVisible

TreeVisible

The TreeVisible property is a read-write property that shows or hides the **Servers** pane.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTagPicker.TreeVisible = discrete;
Result = aaHistClientTagPicker.TreeVisible;
```

Remarks

The default value is True.

TreeWidth

TreeWidth

The TreeWidth property is a read-write property that gets or sets the width of the **Servers** pane when the splitter orientation is vertical or the height of the **Servers** pane when the splitter orientation is horizontal.

Syntax

```
aaHistClientTagPicker.TreeWidth = integer;
Result = aaHistClientTagPicker.TreeWidth;
```

UseHierarchicalName

UseHierarchicalName

The UserHierarchicalName property is a read-write property that sets the option to use the hierarchical name in the Tag Picker.

Syntax

```
aaHistClientTagPicker.UseHierarchicalName = discrete;
Result = aaHistClientTagPicker.UseHierarchicalName;
```

Visible

Visible

The Visible property is a read-write property that shows or hides the Tag Picker.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTagPicker.Visible = discrete;
Result = aaHistClientTagPicker.Visible;
```

Remarks

The default value is True.

aaHistClientTagPicker Methods

The aaHistClientTagPicker methods are:

- ApplyFilter
- LogOn
- OpenAndSelectGroup
- RefreshTags
- SelectedTag
- SetFocusOnSelectedTag

ApplyFilter

ApplyFilter

The ApplyFilter method applies the filter as set up by the properties for the name, description, and I/O address filters.

Syntax

```
[Result=] aaHistClientTagPicker.ApplyFilter();
```

LogOn

LogOn

The LogOn method displays a dialog box for connecting to the specified server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientTagPicker.Logon(aaServer server);
```

Parameter

server

The server to which to connect.

Remarks

This methods uses the aaServer object. For more information, see aaServer Object.

OpenAndSelectGroup

OpenAndSelectGroup

The OpenAndSelectGroup method opens the specified path, and selects the group on a connected Historian server to which you are logged on. You can access this method from the Tag Picker control that is hosted in InTouch as follows:

```
#aaTagPicker.OpenAndSelectGroup(string Path)
```

and from the Tag Picker in Trend control as follows:

```
#aaHistClientTrend1.TagPicker.OpenAndSelectGroup
(string Path)
```

Syntax

```
[Result=] aaHistClientTagPicker.OpenAndSelectGroup (string Path);
```

Parameter

Path

The specific group path which is to be selected. The syntax of the Path parameter is same as that of the SelectedPath property.

Remarks

This method parses the given path and traverses through the tree node collection until the specified group is found. If the group is found, the group is opened and selected with hierarchy.

The specified path is case-sensitive if the AVEVA Historian is installed on a case-sensitive SQL Server. Errors, if any, are logged in the OCMC Logger.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RefreshTags

RefreshTags

The RefreshTags method applies the current filter conditions to all tags from a server.

Syntax

```
[Result=] aaHistClientTagPicker.RefreshTags();
```

Remarks

Use the RefreshTags() method to update the set of filtered tags with any new tags that have been added to the server since the filter was applied. For example, you can add a tag to the server using a script, and then use this method to refresh the Tag Picker so that the new tag is shown.

SelectedTag

SelectedTag

The SelectedTag method gets the selected tag as identified by the index.

Syntax

```
[aaTag=] aaHistClientTagPicker.SelectedTag(integer index);
```

Parameters

index
The numerical identifier for the tag. The identifier is zero-based.

Return Value

Returns the tag or, if it is out of bounds, returns NULL. (It does not return a NULL string.)

Remarks

This method works in conjunction with the SelectedTagCount property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example

The following InTouch HMI software example gets all of the selected tags using a loop:

```
DIM i AS INTEGER;
DIM count AS INTEGER;
Count = #aaHistClientTagPicker3.SelectedTagCount;
FOR i = 0 TO count - 1
%ReturnTag = #aaHistClientTagPicker3.SelectedTag( i );
NEXT;
```

In this example, Count is the number of tags and is retrieved using the SelectedTagCount property.

The index passed to the SelectedTag() method ranges from 0 to
Count - 1. For example:

- If the Count was 0, there are no tags selected.
- If Count is 1, there is one tag selected, and its index is 0.
- If Count is 5, there are 5 tags, and the indices range from 0 to 4.

Therefore, you must first check to see that Count is not 0 and then you can index appropriately to get the tag.

SetFocusOnSelectedTag

SetFocusOnSelectedTag

The SetFocusOnSelectedTag method sets the focus on the selected tag based on the selected path and the index of the selected tab and tag.

Syntax

```
aaHistClientTagPicker.SetFocusOnSelectedTag(string treePath, int tabIndex, int tagIndex);
```

Parameters

treePath

The full path of the tree node from where the tag is to be selected.

tabIndex

The index of the selected tab starting from zero.

tagIndex

The index of the selected tag starting from zero.

Example

The following example sets the focus on the sixth tag from the third tab in the "All Discrete Tags" group available under server MES01.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
#aaHistClientQuery1.SetFocusOnSelectedTag ("MES01.Public Groups.All Discrete Tags", 2, 5);
```

Remarks

This method is specifically used for the aaReports feature in the Input parameter page of the AVEVA Information Server portal.

aaHistClientTagPicker Events

The aaHistClientTagPicker events are:

- OnFilterChanged
- OnGroupChanged
- OnTagsPicked
- OnTagsSelected
- OnServerChanged
- OnSelectedTabChanged
- OnTagNameChanged

OnFilterChanged

OnFilterChanged

The OnFilterChanged event is triggered when the filter is changed.

Syntax

```
aaHistClientTagPicker.OnFilterChanged();
```

OnGroupChanged

OnGroupChanged

The OnGroupChanged event is triggered when the tag group is changed in the navigation tree in the **Servers** pane.

Syntax

```
aaHistClientTagPicker.OnGroupChanged();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

OnTagsPicked

OnTagsPicked

The OnTagsPicked event is triggered when the user double-clicks or picks one or more tags.

Syntax

```
aaHistClientTagPicker.OnTagsPicked();
```

Remarks

A selected tag is a tag that is highlighted (clicked one time) with the mouse by a runtime user. A picked tag is a tag that is double-clicked or selected with the mouse to be dragged. A "picked" tag is always selected, but a selected tag is not always picked.

The application controls whether to also "pick" a tag when it is selected. For example, in the Query client application, selecting a tag causes a change in the query. This is an instance of when the selection of a tag also results in its being picked. In the Trend client application, selecting a tag does not pick and place it on to the trend. However, double-clicking on a tag (picking it) does.

OnTagsSelected

OnTagsSelected

The OnTagsSelected event is triggered when the user selects one or more tags.

Syntax

```
aaHistClientTagPicker.OnTagsSelected();
```

Remarks

For the differences between a "picked" tag and a "selected" tag, see the OnTagsPicked event.

OnServerChanged

OnServerChanged

The OnServerChanged event is triggered when the server is changed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientTagPicker.OnServerChanged();
```

OnSelectedTabChanged

OnSelectedTabChanged

The OnSelectedTabChanged event is triggered when the user changes tabs in the **Tags** pane.

Syntax

```
aaHistClientTagPicker.OnSelectedTabChanged();
```

OnTagNameChanged

OnTagNameChanged

The OnTagNameChanged event is triggered when you set the option to use the hierarchical name or tag name in the Tag Picker.

Syntax

```
aaHistClientTagPicker.OnTagNameChanged();
```

aaHistClientTagPickerSplitterOrientation Enumeration

Specifies the orientation of the **Servers** pane with respect to the **Tags** pane in the Tag Picker.

Value	Enumeration	Description
0	Horizontal	The **Servers** pane is above the **Tags** pane in the Tag Picker.
1	Vertical	The **Servers** pane is to the left of the **Tags** pane in the Tag Picker.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaHistClientTimeRangePicker Control

The aaHistClientTimeRangePicker control allows you to select a time duration based on a start time, duration and/or end time.

Using aaHistClientTimeRangePicker at Runtime

The aaHistClientTimeRangePicker control functions the same as the Time Picker in the time toolbar that appears in the Trend and Query applications.

For more information on using the Time Picker, see Time Picker.

Using aaHistClientTimeRangePicker in an Application

Use the aaHistClientTimeRangePicker control's properties, methods, and events to customize how the time selector behaves during runtime. For example, you can enable the selection of a list of time durations during runtime.

All properties, methods, and events can be controlled through scripting. In addition, some of these properties and methods are exposed through the aaHistClientTimeRangePicker property panel available during application development.

Adding aaHistClientTimeRangePicker to an InTouch Window

To add the aaHistClientTimeRangePicker control:

1. In WindowMaker, click the **Wizards** button . The **Wizard Selection** dialog box appears.
2. Select the aaHistClientTimeRangePicker control.
3. Click **OK**. The control appears in the window.

aaHistClientTimeRangePicker Properties

The properties for the aaHistClientTimeRangePicker are:

- DurationMS
- EndDate
- EndDateUTC
- Format
- StartDate
- StartDateUTC
- TimeDuration
- UpdateToCurrentTimeState

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DurationMS

DurationMS

The DurationMS is a read-write property that controls the duration of the time range in milliseconds.

Syntax

```
aaHistClientTimeRangePicker.DurationMS = integer;
Result = aaHistClientTimeRangePicker.DurationMS;
```

Remarks

When you change this property, the start time is updated based on the new duration and the current end time.

EndDate

EndDate

The EndDate property is a read-only property that returns the end date and time of the time range.

Syntax

```
Result = aaHistClientTimeRangePicker.EndDate;
```

Return Value

A message value in a valid date/time format is returned.

EndDateUTC

EndDateUTC

The EndDateUTC property is a read-only property that returns the end date and time of the time range in the UTC format. The UTC term refers to Coordinated Universal Time. The UTC is a time scale that joins Greenwich Mean Time (GMT).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaHistClientTimeRangePicker.EndDateUTC;
```

Return Value

A message value in a valid date/time format in UTC is returned.

Format

Format

The Format property is a read-write property that gets or sets the date and time formats for the control.

Syntax

```
aaHistClientTimeRangePicker.Format = message;
Result = aaHistClientTimeRangePicker.Format;
```

Remarks

To display the string literals that contain date and time separators or format strings, you must use escape characters in the substring. For example, to display the date and time as 06/15/2001 12:00 PM, this property must be set to:

"dd'/'MM'/'yyyy hh':'mm tt"

The following table lists all the valid format strings and their descriptions.

Format String	Description
d	The one or two-digit day.
dd	The two-digit day. Single digit day values are preceded by a zero.
ddd	The three-character day-of-week abbreviation.
dddd	The full day-of-week name.
h	The one or two-digit hour in 12-hour format.
hh	The two-digit hour in 12-hour format. Single digit values are preceded by a zero.
H	The one or two-digit hour in 24-hour format.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format String	Description
HH	The two-digit hour in 24-hour format. Single digit values are preceded by a zero.
m	The one or two-digit minute.
mm	The two-digit minute. Single digit values are preceded by a zero.
M	The one or two-digit month number.
MM	The two-digit month number. Single digit values are preceded by a zero.
MMM	The three-character month abbreviation.
MMMM	The full month name.
s	The one or two-digit seconds.
ss	The two-digit seconds. Single digit values are preceded by a zero.
t	The one-letter AM/PM abbreviation ("AM" appears as "A").
tt	The two-letter AM/PM abbreviation ("AM" appears as "AM").
y	The one-digit year (2001 appears as "1").
yy	The last two digits of the year (2001 appears as "01").
yyyy	The full year (2001 appears as "2001").

Remarks

The default format is `M/d/yyyy h:mm:ss tt` for English systems.

StartDate

StartDate

The StartDate property is a read-only property that returns the start date and time of the time range.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaHistClientTimeRangePicker.StartDate;
```

Return Value

A message value in a valid date/time format is returned.

StartDateUTC

StartDateUTC

The StartDateUTC property is a read-only property that returns the start date and time of the time range in the UTC format.

Syntax

```
Result = aaHistClientTimeRangePicker.StartDateUTC;
```

Return Value

A message value in a valid date/time format in UTC is returned.

TimeDuration

TimeDuration

The TimeDuration property is a read-write property that controls the duration of the time range as one of the several predefined durations.

Syntax

```
aaHistClientTimeRangePicker.Duration = aaTimeRangeEnumeration;
Result = aaHistClientTimeRangePicker.Duration;
```

Remarks

When you change this property, the start time is updated based on the new duration and the current end time.

For more information on valid values, see aaTimeRangeEnumeration Enumeration.

The default value is 18.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

UpdateToCurrentTimeState

UpdateToCurrentTimeState

The UpdateToCurrentTimeState property is a read-write property that sets the option to update the Time Range Picker to the current time.

Syntax

```
aaHistClientTimeRangePicker.UpdateToCurrentTimeState = integer;
Result = aaHistClientTimeRangePicker.UpdateToCurrentTimeState;
```

Remarks

The valid values are 0 and 1. The default value is 0.

aaHistClientTimeRangePicker Methods

The methods for the aaHistClientTimeRangePicker are:

- GetStartAndEndTimes
- GetStartAndEndTimesUTC
- RefreshTimes
- SetStartAndEndTimes
- SetStartAndEndTimesUTC

GetStartAndEndTimes

GetStartAndEndTimes

The GetStartAndEndTimes method retrieves the start and end times for the query.

Syntax

```
[aaTimeRangeEnumeration=] aaHistClientTimeRangePicker.GetStartAndEndTimes(DateTime
startTime, DateTime endTime);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

startTime

The start time for the query.

endTime

The end time for the query.

Remarks

The date and time formats are set using the Format property.

The container may not allow method parameters to return values. This method is not accessible in the InTouch HMI software. Use the StartDate, EndDate, and TimeDuration properties instead.

Return Value

The time range enumeration (such as Custom, Last5Minutes, and so on) is returned. For more information, see aaTimeRangeEnumeration Enumeration.

GetStartAndEndTimesUTC

GetStartAndEndTimesUTC

The GetStartAndEndTimesUTC method retrieves the start and end times for the query in the UTC format.

Syntax

```
[aaTimeRangeEnumeration=] aaHistClientTimeRangePicker.GetStartAndEndTimesUTC
(DateTime startTimeUTC, DateTime endTimeUTC);
```

Parameters

startTimeUTC

The start time for a query in the UTC format.

endTimeUTC

The end time for a query in the UTC format.

Remarks

The date and time formats are set using the Format property.

The container may not allow method parameters to return values. This method is not accessible in the InTouch HMI software. Use the StartDateUTC, EndDateUTC, and TimeDuration properties instead.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

The time range enumeration (such as Custom, Last5Minutes, and so on) is returned. For more information, see aaTimeRangeEnumeration Enumeration.

RefreshTimes

RefreshTimes

The RefreshTimes method updates the end time to the current time and recalculates the start time based on the new end time and the duration.

Syntax

```
[Result=] aaHistClientTimeRangePicker.RefreshTimes(discrete bFireEvent);
```

Parameters

bFireEvent

When set to True, a change in dates causes the OnChange event to be triggered.

SetStartAndEndTimes

SetStartAndEndTimes

The SetStartAndEndTimes method sets the time period based on a start time, end time, and/or duration.

Syntax

```
[Result=] aaHistClientTimeRangePicker.SetStartAndEndTimes
(DateTime startTime, DateTime endTime, integer duration);
```

Parameters

startTime

The start time for the query. Only considered if the duration is set to Custom. For other durations, the start time is calculated automatically based on the end time and duration.

endTime

The end time for the query. Only considered if the duration is set to Custom or an option from 17 to 32 (OneMinute to ThreeMonths). Otherwise, the end time is set based on the duration.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

duration

The time range duration. If the duration is set to Custom, the specified start and end times are used. For other duration options, the time indicated by the duration is used, and the start and/or end times are updated as necessary. For more information on valid values for the duration, see aaTimeRangeEnumeration Enumeration.

Remarks

The date and time formats are set using the Format property.

SetStartAndEndTimesUTC

SetStartAndEndTimesUTC

The SetStartAndEndTimesUTC method sets the time period based on a start time, end time, and/or duration.

Syntax

```
[Result=] aaHistClientTimeRangePicker.SetStartAndEndTimesUTC
(DateTime startTimeUTC, DateTime endTimeUTC, integer duration);
```

Parameters

startTimeUTC

The start time for a query in the UTC format. Only considered if the duration is set to Custom. For other durations, the start time is calculated automatically based on the end time and duration.

endTimeUTC

The end time for a query in the UTC format. Only considered if the duration is set to Custom or an option from 17 to 32 (OneMinute to ThreeMonths). Otherwise, the end time is set based on the duration.

duration

The time range duration. If the duration is set to Custom, the specified start and end times are used. For other duration options, the time indicated by the duration is used, and the start and/or end times are updated as necessary. For more information on valid values for the duration, see aaTimeRangeEnumeration Enumeration.

Remarks

The date and time formats are set using the Format property.

aaHistClientTimeRangePicker Events

The events for the aaHistClientTimeRangePicker are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- OnChange

OnChange

OnChange

The OnChange event is triggered when the start date and/or end dates are changed.

Syntax

```
aaHistClientTimeRangePicker.OnChange();
```

aaHistClientActiveDataGrid Control

The aaHistClientActiveDataGrid control can execute any SQL query that returns a result set from any AVEVA Historian or Microsoft SQL Server database and returns the results in a grid.

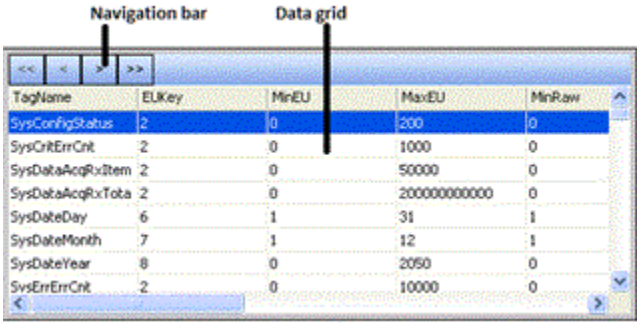
Note: The aaHistClientActiveDataGrid does not support data definition or data manipulation queries.

The aaHistClientActiveDataGrid provides functionality through a user interface and with scripting using properties, methods, and events.

Information is provided on how to configure the aaHistClientActiveDataGrid during application development and describes the aaHistClientActiveDataGrid properties, methods, and events. The runtime functionality of aaHistClientActiveDataGrid is also described.

Using aaHistClientActiveDataGrid at Runtime

The aaHistClientActiveDataGrid provides a user interface that allows you to view record-set data as returned from a specified query during runtime.



Data Grid

Data appears in a tabular format, where each row represents a record and each column represents an attribute (field). The data is read-only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The data grid displays results based on the SQL statement(s) executed and can be used to query different tables and attributes. For example, if the SQL query executes a join on three tables and includes two attributes from each table, the aaHistClientActiveDataGrid shows the records resulting from the join and only the six attributes specified. The number of columns varies dynamically, depending on how many records are returned.

You can resize the columns in the data grid.

Navigating through Records

To navigate through records, do any one of the following:

- Use the arrow keys on your keyboard.
- Right-click the grid, point to **Navigate**, and click one of the navigation commands.
- Use the navigator bar. The navigator bar buttons are as follows:

Button	Description
<<	Moves the current record selection to the first record in the grid.
<	Moves the current selection to the previous record.
>	Moves the current selection to the next record.
>>	Moves the current selection to the last record.

Note: Depending on how the aaHistClientActiveDataGrid was configured during development, the navigator bar may not be available during runtime.

Configuring the Database Connection

You can change the database connection for the aaHistClientActiveDataGrid at runtime.

To configure the database connection

1. Right-click in the aaHistClientActiveDataGrid. In the shortcut menu that appears, click **Properties**.

   The **ActiveDataGrid Properties** dialog box appears.
2. If the **InSQL Connection** tab is not already selected, click the **InSQL Connection** tab.
3. Configure the connection parameters. For more information, see Server Connection Configuration.

   Note: The aaHistClientActiveDataGrid control can only connect to a single server. Multiple servers are not supported.
4. To apply the changes, click **Apply** or **OK**.

   The grid is cleared, and the current SQL statement is re-executed according to the values specified.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Creating or Editing SQL Statements

During runtime, you can create or edit the SQL statement that is executed by the aaHistClientActiveDataGrid. This SQL statement is executed each time the aaHistClientActiveDataGrid is refreshed. If the SQL statement is invalid or if the refresh fails, the data grid is cleared, and an error message appears.

To create or edit a SQL statement

1. Right-click in the aaHistClientActiveDataGrid.
2. In the shortcut menu that appears, click **SQL**. The **ActiveDataGrid Properties** dialog box appears.
3. If the **SQL** tab is not already selected, click the **SQL** tab.
4. In the **SQL statement** window, create or edit the current SQL statement(s) that is executed.
5. To use a pre-configured template, click **Templates**. If not, go to Step 9.

   The **Templates** dialog box appears.
6. In the **Description** list, select the template that you want.
7. Click **Select**.
8. The pre-configured SQL statement syntax appears in the **SQL statement(s)** window. You can then modify the syntax (for example, change the tagname, start date, and end date).
9. To delete all of the text in the **SQL statement** window, click **Clear**.
10. To apply the changes, click **Apply** or **OK**.

Refreshing the Data Grid

When you refresh the data grid , the current contents are cleared and the grid is updated by executing the current SQL query.

To refresh the data grid

- Right-click in the aaHistClientActiveDataGrid. In the shortcut menu that appears, click **Refresh**.

Using aaHistClientActiveDataGrid in an Application

Use aaHistClientActiveDataGrid's properties, methods, and events to create scripts that set up a database connection and customize how the data grid functions during runtime.

Adding aaHistClientActiveDataGrid to an InTouch Window

To add the aaHistClientActiveDataGrid control

1. In WindowMaker, click the **Wizards** button . The **Wizard Selection** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Select the aaHistClientActiveDataGrid control.
3. Click **OK**. The control appears in the window.

aaHistClientActiveDataGrid Properties

The aaHistClientActiveDataGrid properties are:

- AllowUserConfiguration
- AutoRefresh
- BOF
- BusinessObjectServer
- ColumnCount
- Connected
- DatabaseName
- DefaultColumnWidth
- Domain
- Enabled
- EnableShortcutMenu
- EOF
- Handle
- Password
- RefreshFrequency
- Row
- RowCount
- ServerName
- ShowErrorDlgs
- ShowNavigatorBar
- SQLString
- UserName
- VirtualDirectoryName

AllowUserConfiguration

AllowUserConfiguration

The AllowUserConfiguration property is a read-write property that determines whether the user can access the aaHistClientActiveDataGrid **Properties** dialog box at runtime by using the control's shortcut menu.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientActiveDataGrid.AllowUserConfiguration = discrete;
Result = aaHistClientActiveDataGrid.AllowUserConfiguration;
```

Remarks

True = Show the **Properties** and **SQL** menu commands on the shortcut menu; False = Hide the **Properties** and **SQL** menu commands on the shortcut menu.

If this property is disabled, you can use the ShowPropertiesDialog method to let the user access the **Properties** dialog box.

The default value is True.

AutoRefresh

AutoRefresh

The AutoRefresh property is a read-write property that enables or disables automatic refresh of the data in the aaHistClientActiveDataGrid.

Syntax

```
aaHistClientActiveDataGrid.AutoRefresh = discrete;
Result = aaHistClientActiveDataGrid.AutoRefresh;
```

Remarks

True = Automatic refresh on; False = Automatic refresh off.

The default value is False.

Automatic refresh works by periodically calling the Execute method. The time interval is based on the RefreshFrequency property. The default time interval is 60 seconds.

The AutoRefresh property is set to False if the last manual call to the Execute method failed. If the AutoRefresh property is set to True, and for some reason later fails, it is automatically set to False, and the aaHistClientActiveDataGrid is reset (cleared).

BOF

BOF

The BOF property is a read-only property that returns whether the user has attempted to navigate prior to the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

first row in the data grid.

Syntax

```
Result = aaHistClientActiveDataGrid.BOF;
```

Return Value

The result is a discrete. True is returned if an attempt was made to move prior to the first row in the data grid through a call to the MovePrevious method; otherwise False is returned.

ColumnCount

ColumnCount

The ColumnCount property is a read-only property that gets the number of columns in the current result set of the data grid.

Syntax

```
Result = aaHistClientActiveDataGrid.ColumnCount;
```

Return Value

Returns the number of columns as an integer. If the data grid is not connected, 0 is returned.

Remarks

The default value is 0.

Connected

Connected

Use this read-write property to initiate or terminate a connection to the AVEVA Historian and to check whether a connection is currently active.

Syntax

```
aaHistClientActiveDataGrid.Connected = discrete;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Result = aaHistClientActiveDataGrid.Connected;
```

Remarks

If set to True, and the ServerName, DatabaseName, UserName, and Password properties are set, the control tries to connect to the AVEVA Historian and execute the SQL statement specified by the SQLString property. If an error occurs, the Connected property is set to False.

If set to False while a connection is active, the control is disconnected from the server and reset.

The default value is False.

DatabaseName

DatabaseName

The DatabaseName property is a read-write property that specifies the name of the database to connect to. The database must exist on the database server specified by the ServerName property.

Syntax

```
aaHistClientActiveDataGrid.DatabaseName = message;
Result = aaHistClientActiveDataGrid.DatabaseName;
```

Remarks

When working with a AVEVA Historian database, the value for the DatabaseName property must be `Runtime`. However, aaHistClientActiveDataGrid can connect to any database in the Microsoft SQL Server, such as the `master` database.

The default value is `Runtime`.

DefaultColumnWidth

DefaultColumnWidth

The DefaultColumnWidth property is a read-write property that gets or sets the default column width, in pixels, of the columns shown in the data grid.

Syntax

```
aaHistClientActiveDataGrid.DefaultColumnWidth = integer;
Result = aaHistClientActiveDataGrid.DefaultColumnWidth;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The default value is 100.

Domain

Domain

The Domain property is a read-write property that gets or sets the domain string for the connection to the server.

Syntax

```
aaServer.Domain = message;
Result = aaServer.Domain;
```

Remarks

This property is useful in cases where the Windows integrated security requires the domain name to be specified.

The default is an empty message value ("").

Enabled

Enabled

The Enabled property is a read-write property that enables or disables the user interface functionality of the control.

Syntax

```
aaHistClientActiveDataGrid.Enabled = discrete;
Result = aaHistClientActiveDataGrid.Enabled;
```

Remarks

True = User interface functionality enabled; False = User interface functionality disabled.

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

EnableShortcutMenu

EnableShortcutMenu

The EnableShortcutMenu property is a read-write property that enables or disables the right-click shortcut menu of the control.

Syntax

```
aaHistClientActiveDataGrid.EnableShortcutMenu = discrete;
Result = aaHistClientActiveDataGrid.EnableShortcutMenu;
```

Remarks

True = Shortcut menu enabled; False = Shortcut menu disabled.

The default value is True.

EOF

EOF

The EOF property is a read-only property that returns whether the aaHistClientActiveDataGrid user has attempted to navigate beyond the last row in the data grid.

Syntax

```
Result= aaHistClientActiveDataGrid.EOF;
```

Return Value

The result is a discrete. True is returned if an attempt was made to move past the last row in the data grid with a call to the MoveNext method; otherwise False is returned.

Handle

Handle

The Handle property is a read-only property that returns the Window handle for the control.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaHistClientActiveDataGrid.Handle;
```

Return Value

The return value is an integer. Returns the 32-bit Window handle of the main container window.

Remarks

The Window handle is useful when using Windows API functions to manipulate a control.

This property has no default value.

Password

Password

The Password property is a write-only property that specifies the password for the provided username on the specified AVEVA Historian.

Syntax

```
aaHistClientActiveDataGrid.Password = message;
```

Remarks

See the AVEVA Historian documentation for the default passwords associated with the default usernames.

RefreshFrequency

RefreshFrequency

The RefreshFrequency property is a read-write property that specifies how often an automatic refresh of the aaHistClientActiveDataGrid occurs.

Syntax

```
aaHistClientActiveDataGrid.RefreshFrequency = integer;
Result = aaHistClientActiveDataGrid.RefreshFrequency;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property specifies the frequency, in milliseconds, that the SQL statement is re-executed when the AutoRefresh property is set to True. The frequency value must be greater than 0.

The default value is 60,000 milliseconds (1 minute).

Row

Row

The Row property is a read-only property that returns the relative row number of the selected row in the data grid.

Syntax

```
Result = aaHistClientActiveDataGrid.Row;
```

Return Value

The return value is an integer that specifies the number of the selected row. Row numbers start at 1.

Remarks

The default value is -1.

RowCount

RowCount

The RowCount property is a read-only property that returns the total number of rows in the record set that is returned.

Syntax

```
Result= aaHistClientActiveDataGrid.RowCount;
```

Return Value

The return value is an integer that specifies the number of rows in the record set.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

The default value is 0.

ServerName

ServerName

The ServerName property is a read-write property that specifies the name of the AVEVA Historian to which you want to connect.

Syntax

```
aaHistClientActiveDataGrid.ServerName = message;
Result = aaHistClientActiveDataGrid.ServerName;
```

Remarks

The ServerName property must be set to establish a connection to an AVEVA Historian.

This property has no default value.

ShowErrorDlgs

ShowErrorDlgs

The ShowErrorDlgs property is a read-write property that determines whether error messages appear during runtime in an error dialog box.

Syntax

```
aaHistClientActiveDataGrid.ShowErrorDlgs = discrete;
Result = aaHistClientActiveDataGrid.ShowErrorDlgs;
```

Remarks

True = Error messages displayed; False = Error messages suppressed. If the error message display is disabled, you do not see any errors, even if they are critical. Use this option with extreme caution.

The default value is True.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ShowNavigatorBar

ShowNavigatorBar

The ShowNavigatorBar property is a read-write property that shows or hides the Navigator toolbar that is located above the data grid.

Syntax

```
aaHistClientActiveDataGrid.ShowNavigatorBar = discrete;
Result = aaHistClientActiveDataGrid.ShowNavigatorBar;
```

Remarks

True = Shows the Navigator toolbar; False = Hides the Navigator toolbar.

The default value is True.

SQLString

SQLString

The SQLString property is a read-write property that specifies the SQL statement to be executed by the Execute method.

Syntax

```
aaHistClientActiveDataGrid.SQLString = message;
Result = aaHistClientActiveDataGrid.SQLString;
```

Remarks

The aaHistClientActiveDataGrid uses the InSQL OLE DB provider to access the AVEVA Historian historical data. If you are querying data from the analog or discrete history tables, the SQL statement must follow the syntax rules for OLE DB provider queries. Otherwise, you can use any valid Transact-SQL that returns rows.

Remarks

The default is an empty message value ("").

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

UserName

UserName

The UserName property is a read-write property that specifies the username used to logon to the AVEVA Historian specified in the ServerName property.

Syntax

```
aaHistClientActiveDataGrid.UserName = message;
Result = aaHistClientActiveDataGrid.UserName;
```

Remarks

See the AVEVA Historian documentation for information on the default AVEVA Historian users.

Remarks

The default UserName is wwUser.

VirtualDirectoryName

VirtualDirectoryName

The VirtualDirectoryName property is a read-write property that gets or sets the virtual directory name. more

Syntax

```
aaHistClientActiveDataGrid.VirtualDirectoryName = message;
Result = aaHistClientActiveDataGrid.VirtualDirectoryName;
```

Remarks

The default is an empty message value ("").

BusinessObjectServer

This read-write property specifies the path to the HTTP server when using HTTP to access the historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientActiveDataGrid.BusinessObjectServer = message;
Result = aaHistClientActiveDataGrid.BusinessObjectServer;
```

Remarks

If this property is set to a non-empty string value, the control uses HTTP access to the historian. If it is set to an empty string, the control uses regular SQL Server access.

You can obtain a secured connection by specifying `https://<Servername>`. For example:

```
#ActiveDataGrid.BusinessObjectServer ="HTTPS://www.server.com";
```

For more information on using HTTP to access the historian, see Considerations for VPN Access.

To enable HTTP access, you must also specify the virtual directory name using the VirtualDirectoryName property.

aaHistClientActiveDataGrid Methods

The aaHistClientActiveDataGrid methods are:

- ClearGrid
- ColumnName
- ColumnValue
- ColumnValueByName
- Execute
- MoveFirst
- MoveLast
- MoveNext
- MovePrevious
- RowColumnValue
- RowColumnValueByName
- ShowPropertiesDialog
- SQLAppend

ClearGrid

ClearGrid

The ClearGrid method clears the contents of the data grid and sets the Connected property to False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientActiveDataGrid.ClearGrid();
```

ColumnName

ColumnName

The ColumnName method returns the column name that corresponds to the specified column index.

Syntax

```
Result = aaHistClientActiveDataGrid.ColumnName(integer columnIndex);
```

Parameters

columnIndex

Number of the column name for which the string representation is returned. Column names start at 1.

Return Value

The name of the column as a message value.

ColumnValue

ColumnValue

The ColumnValue method returns the string representation of the data for the specified column of the currently selected row.

Syntax

```
Result = aaHistClientActiveDataGrid.ColumnValue(integer Column);
```

Parameters

Column

Number of the column for which the string representation is returned. Column numbers start at 1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

A message representation of the data.

ColumnValueByName

ColumnValueByName

The ColumnValueByName method gets the string representation of the data for the specified column name, for the currently selected row.

Syntax

```
Result = aaHistClientActiveDataGrid.ColumnValueByName(message columnName);
```

Parameters

columnName

The name of the column.

Return Value

The data in the column as a message value.

Execute

Execute

The Execute method executes the SQL query defined in the SQLString property.

Syntax

```
[Result=] aaHistClientActiveDataGrid.Execute();
```

Return Value

True = Execution is successful; False = Execution unsuccessful.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

If the Execute method fails, the data grid is cleared and an error is raised.

The most typical conditions that cause Execute to fail are:

- The specified server is not running or connection to it is not available.
- The server assigned to the ServerName property is invalid or not found.
- The username assigned to the UserName property is invalid or not found.
- The password assigned to the Password property is invalid or not associated with the specified UserName on the specified ServerName.
- There is a syntax error in the SQLString property.
- The DatabaseName property was not assigned or the wrong database was specified.
- The BusinessObjectServer property is set to an HTTP server that does not exist, or the HTTP server specified is not running

MoveFirst

MoveFirst

The MoveFirst method selects the first row in the data grid.

Syntax

```
aaHistClientActiveDataGrid.MoveFirst();
```

MoveLast

MoveLast

The MoveLast method selects the last row in the data grid.

Syntax

```
aaHistClientActiveDataGrid.MoveLast();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MoveNext

MoveNext

The MoveNext method selects the next row in the data grid.

Syntax

```
aaHistClientActiveDataGrid.MoveNext();
```

Remarks

If an attempt is made to move past the last row the EOF property is set to True.

MovePrevious

MovePrevious

The MovePrevious method selects the previous row in the data grid.

Syntax

```
aaHistClientActiveDataGrid.MovePrevious();
```

Remarks

If an attempt is made to move past the last row the BOF property is set to True.

RowColumnValue

RowColumnValue

The RowColumnValue method returns the string representation of the data in the specified row and column in the data grid.

Syntax

```
[Result=] aaHistClientActiveDataGrid.RowColumnValue(integer row, integer column);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

row
Number of the row for which the string representation is returned. Row numbers start at 1.
column
Number of the column for which the string representation is returned. Column numbers start at 1.

Return Value

A message representation of the data.

Remarks

This property does not move the selected row, nor does it require the selected row to be changed.

RowColumnValueByName

RowColumnValueByName

The RowColumnValueByName method gets the value at the indicated row and column (specified by name).

Syntax

```
[Result=] aaHistClientActiveDataGrid.RowColumnValueByName(integer row, message columnName);
```

Parameters

row
Number of the row for which the string representation is returned. Row numbers start at 1.
columnName
Name of the column for which the string representation is returned.

Return Value

A message representation of the data.

Remarks

This property does not move the selected row, nor does it require the selected row to be changed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ShowPropertiesDialog

ShowPropertiesDialog

The ShowPropertiesDialog method shows the **Properties** dialog box for the aaHistClientActiveDataGrid during runtime.

Syntax

```
[Result=] aaHistClientActiveDataGrid.ShowPropertiesDialog( integer Page);
```

Parameters

Page

Specifies which tab should be active when the **Properties** dialog box is opened. 0 = **InSQL Connection** tab is active; 1 = **SQL** tab is active.

SQLAppend

SQLAppend

The SQLAppend method appends a section of a long SQL statement to the end of the existing SQL string in the SQLString property.

Syntax

```
[Result=] aaHistClientActiveDataGrid.SQLAppend(message SQL);
```

Parameters

SQL

Section of SQL to be added to the SQL statement(s) that are to be executed.

Remarks

This method facilitates the scripting of long SQL Statements within the InTouch HMI software. Currently, the InTouch HMI software has a 131 character limitation for strings. To circumvent this limitation, use this method to add SQL statements in sections.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example

The following example demonstrates how to use the SQLAppend method to setup the necessary SQL to retrieve the last 30 minutes of history data for the tag 'SysTimeSec.'

```
#aaHistClientActiveDataGrid.ServerName = "toddm1";
#aaHistClientActiveDataGrid.UserName = "wwUser";
#aaHistClientActiveDataGrid.Password = "wwUser";
#aaHistClientActiveDataGrid.SQLString = "";
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @StartDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @EndDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @StartDate = DateAdd(mi, -30, GetDate())");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @EndDate = GetDate()");
#aaHistClientActiveDataGrid.SQLAppend("SELECT TagName, DateTime, Value");
#aaHistClientActiveDataGrid.SQLAppend("FROM v_AnalogHistory");
#aaHistClientActiveDataGrid.SQLAppend("WHERE TagName IN ('SysTimeMin')");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime >= @StartDate");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime <= @EndDate");
#aaHistClientActiveDataGrid.SQLAppend("AND wwRetrievalMode = 'Delta' ");
#aaHistClientActiveDataGrid.Execute();
```

aaHistClientActiveDataGrid Events

The aaHistClientActiveDataGrid control has the following events:

- OnClick
- OnDblClick
- OnError

For information on ambient events, see Common Events.

OnClick

OnClick

The OnClick event is triggered every time the user clicks on a data row in the control.

Syntax

```
aaHistClientActiveDataGrid.OnClick;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

OnDblClick

OnDblClick

The OnDblClick event is triggered every time the user double-clicks on a data row in the control.

Syntax

```
aaHistClientActiveDataGrid.OnDblClick;
```

OnError

OnError

The OnError event executes each time an error message is to be displayed.

Syntax

```
aaHistClientActiveDataGrid.OnError(integer ErrorNo, ref message ErrStr, ref discrete
ShowErrorDlg);
```

Parameters

ErrorNo

A unique number that corresponds to the error message, which is specified by the ErrStr parameter.

ErrStr

Message to be displayed in the error dialog box.

ShowErrorDlg

Determines whether the error dialog box appears. True = Error dialog box displayed; False = Err dialog box not displayed. The ShowErrorDlg parameter defaults to the value of the ShowErrorDlg property.

Remarks

The OnError event provides a means to intercept an error message and either disable it from showing or change the error message text shown.

For information on error numbers and error text pertaining to each control, see the "Error Messages" section in the chapter for that control.

The OnError event executes prior to the display of any error messages. In your script, you can then capture the error, check the ErrStr parameter, and set the parameter to a new value. You can also translate the same string into a different language. If you want to implement your own error handling, you can suppress the default error

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

dialog by setting the ShowErrorDlg parameter to False.

Example

The following example shows how the event parameter can be set in the InTouch HMI software:

```
TRErrorNo = #ThisEvent.OnErrorerrorNo; {Assign error number from the event to a tag called
TRErrorNo}
IF TRShowErrorDlg == 0 THEN {Checking user preference on showing Error Dialog}
   #ThisEvent.OnErrorshowErrorDlg = TRShowErrorDlg; {Do not show any error dialog. A value
   has been assigned value to the ShowErrorDlg parameter}
ELSE
   #ThisEvent.OnErrorshowErrorDlg = TRShowErrorDlg; {Show the Error dialog}
   IF UserPreferredDialog == 1 THEN {Check whether user wants his/her own dialog}
      IF TRErrorNo == 0 THEN {If the error number from the event is 0}
         #ThisEvent.OnErrorerrorString = "General Error"; {Assigning a value to ErrStr
         parameter of the event.}
      ELSE IF TRErrorNo == 1 THEN
         #ThisEvent.OnErrorerrorString = "Not able to connect to the specified server.";
      ELSE IF TRErrorNo == 2 THEN
         #ThisEvent.OnErrorerrorString = "Set the server name first.";
      ENDIF;
      ENDIF;
      ENDIF;
   ENDIF;
ENDIF;
```

Script Examples for aaHistClientActiveDataGrid

The following sections provide scripting examples for aaHistClientActiveDataGrid.

InTouch Example: History Data Over a LAN

InTouch Example: History Data Over a LAN

The following example demonstrates how to connect to the AVEVA Historian named "maggie" on a LAN. The example retrieves the last 45 minutes of history data for the 'SysPulse' tag.

```
#aaHistClientActiveDataGrid.ServerName ="maggie";
#aaHistClientActiveDataGrid.UserName = "wwUser";
#aaHistClientActiveDataGrid.Password = "wwUser";
#aaHistClientActiveDataGrid.DatabaseName = "Runtime";
#aaHistClientActiveDataGrid.SQLString = "";
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @StartDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @EndDate DateTime");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @StartDate = DateAdd(mi, -45, GetDate())");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @EndDate = GetDate()");
#aaHistClientActiveDataGrid.SQLAppend("SELECT Tagname, DateTime, Value");
#aaHistClientActiveDataGrid.SQLAppend("FROM v_DiscreteHistory");
#aaHistClientActiveDataGrid.SQLAppend("WHERE TagName IN (‘SysPulse’)");
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime >= @StartDate");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime <= @EndDate");
#aaHistClientActiveDataGrid.SQLAppend("AND wwRetrievalMode = ‘Delta’");
#aaHistClientActiveDataGrid.Connected = 1;
```

InTouch Example: Retrieving Data from the Grid

InTouch Example: Retrieving Data from the Grid

The following example script demonstrates how to extract data from the grid using the ColumnValue method.

```
#aaHistClientActiveDataGrid.ServerName = "maggie";
#aaHistClientActiveDataGrid.UserName = "wwUser";
#aaHistClientActiveDataGrid.Password = "wwUser";
#aaHistClientActiveDataGrid.DatabaseName = "Runtime";
#aaHistClientActiveDataGrid.SQLString = "";
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @StartDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @EndDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @StartDate = DateAdd(mi, -30, GetDate())");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @EndDate = GetDate()");
#aaHistClientActiveDataGrid.SQLAppend("SELECT TagName, DateTime, Value");
#aaHistClientActiveDataGrid.SQLAppend("FROM v_AnalogHistory");
#aaHistClientActiveDataGrid.SQLAppend("WHERE TagName IN ('SysTimeSec')");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime >= @StartDate");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime <= @EndDate");
#aaHistClientActiveDataGrid.SQLAppend("AND wwRetrievalMode = 'Cyclic'");
#aaHistClientActiveDataGrid.SQLAppend("AND wwCycleCount = 100");
#aaHistClientActiveDataGrid.Connected = 1;
#aaHistClientActiveDataGrid.MoveFirst();
FOR Row = 1 TO #aaHistClientActiveDataGrid.RowCount
   TagName = #aaHistClientActiveDataGrid.ColumnValue(0);
   DateTime = #aaHistClientActiveDataGrid.ColumnValue(1);
   TagValueText = #aaHistClientActiveDataGrid.ColumnValue(2);
   TagValue = StringToReal( TagValueText );
   EndOfFile = #aaHistClientActiveDataGrid.EOF;
   IF EndOfFile THEN
      EXIT FOR;
   ELSE
      #aaHistClientActiveDataGrid.MoveNext();
   ENDIF;
NEXT;
```

Another slightly different approach is to go through the returned data without actually moving the row selector using the RowColumnValue method. This approach is much more efficient because there is no UI updating.

```
#aaHistClientActiveDataGrid.ServerName = "maggie";
#aaHistClientActiveDataGrid.UserName = "wwUser";
#aaHistClientActiveDataGrid.Password = "wwUser";
#aaHistClientActiveDataGrid.DatabaseName = "Runtime";
#aaHistClientActiveDataGrid.SQLString = "";
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @StartDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("DECLARE @EndDate Datetime");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @StartDate = DateAdd(mi, -30, GetDate())");
#aaHistClientActiveDataGrid.SQLAppend("SELECT @EndDate = GetDate()");
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
#aaHistClientActiveDataGrid.SQLAppend("SELECT TagName, DateTime, Value");
#aaHistClientActiveDataGrid.SQLAppend("FROM v_AnalogHistory");
#aaHistClientActiveDataGrid.SQLAppend("WHERE TagName IN ('SysTimeSec')");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime >= @StartDate");
#aaHistClientActiveDataGrid.SQLAppend("AND DateTime <= @EndDate");
#aaHistClientActiveDataGrid.SQLAppend("AND wwRetrievalMode = 'Cyclic'");
#aaHistClientActiveDataGrid.SQLAppend("AND wwCycleCount = 100");
#aaHistClientActiveDataGrid.Connected = 1;
FOR Row = 1 TO #aaHistClientActiveDataGrid.RowCount - 1
   TagName = #aaHistClientActiveDataGrid.RowColumnValue(Row, 0);
   DateTime = #aaHistClientActiveDataGrid.RowColumnValue(Row, 1);
   TagValueText = #aaHistClientActiveDataGrid.RowColumnValue(Row, 2);
   TagValue = StringToReal ( TagValueText );
NEXT;
```

aaHistClientActiveDataGrid Error Messages

The aaHistClientActiveDataGrid error messages are:

Error Number	Error Message
0	General error. A general error is usually due to a data connectivity error.
1	Failed to connect to server: <ServerName>
2	The ServerName property must be set to a valid AVEVA Historian.
3	The UserName property cannot be blank.
4	Unable to get the Generic SQL view for server name: <ServerName>
5	You must first execute the SQL query before performing this operation.
6	You must type the SQL query you wish to execute before a connection attempt is performed.
7	The 'ActiveDataGrid' ActiveX is not licensed for your use on this workstation. Please contact your Administrator.
8	Row (<Row Index>) does not exist in the current query results.'
9	Column (<Column index>) does not exist in the current query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Error Number	Error Message
10	The column name, <Column Name>, was not found.
11	The RefreshFrequency property must be assigned a positive number.

aaHistClientSingleValueEntry Control

Use the aaHistClientSingleValueEntry control to manually add a tag value to the AVEVA Historian database.

Using the aaHistClientSingleValueEntry Control at Runtime

Use the aaHistClientSingleValueEntry control to manually add single data value for a tag in the AVEVA Historian database. To add a data value for a tag:

- You must log in to the historian with administrative privileges.
- The tag must currently exist in the database.

Values are inserted into the AVEVA Historian's history blocks. Therefore, you can retrieve them using the same views and tables as data that is stored automatically by the AVEVA Historian.

Adding a Tag Value

The available functionality depends on how the application developer configured the aaHistClientSingleValueEntry control.

To add a tag value

1. Log on to the historian using the means provided by the application.

   If the log on is successful, a green indicator appears in the server status icon in the status bar for the control. The name of the logged on user also appears in the status bar.
2. In the **Tagname** list, type the name of the tag for which you want to insert a value. To browse for a tag, click the ellipsis button. The Tag Picker appears, in which you can browse for a tag. For more information on using the Tag Picker, see Tag Picker.
3. In the **Date and time** box, enter the timestamp used for the inserted value. To use the current time, select the check box to the right of the **Date and time** box.
4. In the **Value** box, enter the data value to insert for the tag.
5. Click the arrow button. The status of the insertion appears in the status bar.

Using the aaHistClientSingleValueEntry Control in an Application

You can use the aaHistClientSingleValueEntry control's properties, methods, and events in runtime scripts in your application to control the functionality available to the runtime user.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding the aaHistClientSingleValueEntry Control to an InTouch Window

To add the aaHistClientSingleValueEntry control:

1. In WindowMaker, click the **Wizards** button . The **Wizard Selection** dialog box appears.
2. Select the aaHistClientSingleValueEntry control.
3. Click **OK**. The control appears in the window.

aaHistClientSingleValueEntry Control Properties

The aaHistClientSingleValueEntry control properties include:

- AnalogValue
- ContextMenuEnabled
- CurrentServerName
- DateTime
- DateTimeFieldDisable
- DateTimeFieldVisible
- DateTimeString
- DisableTagEntry
- DisplayErrorMessages
- FieldLabelPosition
- FieldLayoutHorizontal
- HideDateTimeModeTabs
- HideFieldLabels
- HideStatusBar
- InsertButtonVisible
- InTouchDateTime
- LastErrorDetails
- LastErrorMessage
- LastOperationResult
- LastOperationSuccessful
- Pwd
- Quality
- QualityDetail
- QualityDetailFieldDisable
- QualityDetailFieldVisible

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- QualityFieldDisable
- QualityFieldVisible
- RememberEnteredTags
- Servers
- StringValue
- TagName
- TagNameFieldDisable
- TagNameFieldVisible
- TagPickerButtonDisable
- TagPickerButtonVisible
- Tags
- TagType
- TagValid
- Transparent
- User
- UseTimezone
- Value
- ValueEx
- ValueFieldDisable

AnalogValue

AnalogValue

The AnalogValue property is a read-write property that gets or sets the analog value to be inserted.

Syntax

```
aaHistClientSingleValueEntry.AnalogValue = real;
Result = aaHistClientSingleValueEntry.AnalogValue;
```

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

CurrentServerName

CurrentServerName

The CurrentServerName property is a read-write property that gets or sets the name of the AVEVA Historian.

Syntax

```
aaHistClientSingleValueEntry.CurrentServerName = message;
Result = aaHistClientSingleValueEntry.CurrentServerName;
```

Remarks

If the server has already been added, the User property is automatically set to the current username.
This property has no default value.

DateTime

DateTime

The DateTime property is a read-write property that gets or sets the timestamp to be used for the value insert.

Syntax

```
aaHistClientSingleValueEntry.DateTime = DateTime;
Result = aaHistClientSingleValueEntry.DateTime;
```

Remarks

To use the current time, set this property to 0.
Setting this property also updates the DateTimeString and InTouchDateTime properties, and vice-versa.
For more information on the DateTime data type, see DateTime.

Remarks

The default value is 12:00:00 AM.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTimeFieldDisable

DateTimeFieldDisable

The DateTimeFieldDisable property is a read-write property that gets or sets whether the **Date and time** box is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.DateTimeFieldDisable = discrete;
Result = aaHistClientSingleValueEntry.DateTimeFieldDisable;
```

Remarks

The default value is False.

DateTimeFieldVisible

DateTimeFieldVisible

The DateTimeFieldVisible property is a read-write property that gets or sets whether the **Date and time** box is visible in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.DateTimeFieldVisible = discrete;
Result = aaHistClientSingleValueEntry.DateTimeFieldVisible;
```

Remarks

The default value is True.

DateTimeString

DateTimeString

The DateTimeString property is a read-write property that gets and sets the timestamp as a string value to be used for the insert.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientSingleValueEntry.DateTimeString = message;
Result = aaHistClientSingleValueEntry.DateTimeString;
```

Remarks

The DateTimeString property reflects the value of the DateTime property, but it is expressed as a string that uses local regional settings. The DateTime property is expressed in the Date format.

If the DateTime property is set to 0, the current date and time are returned. If this property is set to an empty string (" "), the current date and time are used for the insert.

Setting this property also updates the DateTime and InTouchDateTime properties, and vice-versa.

Remarks

The default is an empty message value (which indicates to use the current time).

DisableTagEntry

DisableTagEntry

The DisableTagEntry property is a read-write property that gets or sets whether the **Tag Name** box can be edited at runtime.

Syntax

```
aaHistClientSingleValueEntry.DisableTagEntry = discrete;
Result = aaHistClientSingleValueEntry.DisableTagEntry;
```

Remarks

If set to True, the runtime user cannot use the **Tag Name** box to type in a tagname. The user needs to use the Tag Picker to select a tag or select a tag from a list of recently used tags. (provided that either functionality is enabled).

The default value is False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DisplayErrorMessages

DisplayErrorMessages

The DisplayErrorMessage property is a read-write property that enables or disables the display of error message dialog boxes.

Syntax

```
aaHistClientSingleValueEntry.DisplayErrorMessages = discrete;
Result = aaHistClientSingleValueEntry.DisplayErrorMessages;
```

Remarks

If set to True, all error message dialog boxes appear. If set to False, no error messages appear, except for server logon failure messages.

The default value is True.

FieldLabelPosition

FieldLabelPosition

The FieldLabelPosition property is a read-write property that gets or sets whether the field labels appear when the control is in the vertical layout mode.

Syntax

```
aaHistClientSingleValueEntry.FieldLabelPosition = aaFieldLabelPositionEnumeration;
Result = aaHistClientSingleValueEntry.FieldLabelPosition;
```

Remarks

For more information on the aaFieldLabelPositionEnumeration enumeration, see aaFieldLabelPositionEnumeration Enumeration.

If the FieldLayoutHorizontal property is set to True, the FieldLabelPosition property has no effect.

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

FieldLayoutHorizontal

FieldLayoutHorizontal

The FieldLayoutHorizontal property is a read-write property that gets or sets whether or not the text boxes (fields) for the control appear next to each other from left to right (horizontally) instead of stacked on top of each other (vertically).

Syntax

```
aaHistClientSingleValueEntry.FieldLayoutHorizontal = discrete;
Result = aaHistClientSingleValueEntry.FieldLayoutHorizontal;
```

Remarks

The default value is True.

HideDateTimeModeTabs

HideDateTimeModeTabs

This read-write property controls whether the check box next to the Date and time box is visible at runtime. If visible, the check box allows the user to toggle between using automatic timestamps and manually specifying a timestamp.

Syntax

```
aaHistClientSingleValueEntry.HideDateTimeModeTabs = discrete;
Result = aaHistClientSingleValueEntry.HideDateTimeModeTabs;
```

Remarks

If set to False, the check box is visible.

The default value is False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

HideFieldLabels

HideFieldLabels

The HideFieldLabels property is a read-write property that gets or sets whether the labels for the text boxes (fields) are visible to the runtime user.

Syntax

```
aaHistClientSingleValueEntry.HideFieldLabels = discrete;
Result = aaHistClientSingleValueEntry.HideFieldLabels;
```

Remarks

If set to True, the field labels are hidden.

The default value is False.

HideStatusBar

HideStatusBar

The HideStatusBar property is a read-write property that gets or sets whether the status bar is visible to the runtime user.

Syntax

```
aaHistClientSingleValueEntry.HideStatusBar = discrete;
Result = aaHistClientSingleValueEntry.HideStatusBar;
```

Remarks

If set to True, the status bar is hidden.

The default value is False.

The status bar appears at the bottom of the control.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

InsertButtonVisible

InsertButtonVisible

The InsertButtonVisible property is a read-write property that gets or sets whether the **Insert** button is visible in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.InsertButtonVisible = discrete;
Result = aaHistClientSingleValueEntry.InsertButtonVisible;
```

Remarks

The default value is True.

InTouchDateTime

InTouchDateTime

The InTouchDateTime property is a read-write property that gets or sets the timestamp for the data insert using the InTouch HMI software Date format.

Syntax

```
aaHistClientSingleValueEntry.InTouchDateTime = real;
Result = aaHistClientSingleValueEntry.InTouchDateTime;
```

Remarks

The InTouchDateTime property reflects the value of the DateTime property, but it is expressed in the InTouch HMI software $DateTime format. The DateTime property is expressed in the Date format. For more information on the $DateTime format, see the InTouch HMI software documentation.

If this property is set -1, the current date and time are used for the insert.

If the DateTime property is set to 0, the current date and time are returned for the InTouchDateTime property.

Setting this property also updates the DateTime and DateTimeString properties, and vice-versa.

The DateTime property supports dates starting from 12/30/1899. The InTouch HMI software supports dates starting from 1/1/1970. Therefore, if the DateTime property is set to a date prior to 1/1/1970, the InTouchDateTime property are set to -1. To support dates prior to 1/1/1970, use the DateTimeString property.

The default value is -1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example

The following example sets the timestamp for the insert to the current time (reflected by the $DateTime system tag in the InTouch HMI software).

```
aaHistClientSingleValueEntry1.InTouchDateTime = $DateTime;
```

LastErrorDetails

LastErrorDetails

The LastErrorDetails property is a read-only property that gets the error code for the error message from SQL Server.

Syntax

```
Result = aaHistClientSingleValueEntry.LastErrorDetails;
```

Remarks

If a SQL error occurred during the last insert, the error is returned to this property. This property contains the long version of the error.

No details are available if the LastOperationResult property contains a value between 0 and -6.

To clear the contents of this property, use the Reset method.

This property has no default value.

LastErrorMessage

LastErrorMessage

The LastErrorMessage property is a read-only property that gets the status of the last data insert.

Syntax

```
Result = aaHistClientSingleValueEntry.LastErrorMessage;
```

Remarks

The status returned is the short version. Use the LastErrorDetails property to return the details.

To clear the contents of this property, use the Reset method.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LastOperationResult

LastOperationResult

The LastOperationResult property is a read-only property that gets the error code for the last insert.

Syntax

```
Result = aaHistClientSingleValueEntry.LastOperationResult;
```

Return Values

Returns one of the following values:

0 = The value was successfully inserted.

-1 = The insert failed.

-2 = The specified server is not in the collection of servers. Be sure that you call the AddServer method first.

-3 = No server name provided. The CurrentServerName property is blank or the *serverName* parameter was not provided for the InsertValue method.

-4 = No tagname provided.

-5 = The date/time is invalid (the date/time string was unable to be converted).

-6 = The tag does not exist on the server.

<*other negative values*> = Error code from Microsoft SQL Server. For more information, check the LastErrorDetails property.

The default value is 0.

Remarks

Before calling this method, call the AddServer method to ensure that the server name is in the server collection for this object.

LastOperationSuccessful

LastOperationSuccessful

The LastOperationSuccessful property is a read-only property that gets the status of the last data value insert.

Syntax

```
Result = aaHistClientSingleValueEntry.LastOperationSuccessful;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

If set to True, the last insert was successful.

To reset this property, use the Reset method.

The default value is False.

Pwd

Pwd

Use this write-only property to specify the password that should be used to log on the current user to the current server.

Syntax

```
aaHistClientSingleValueEntry.Pwd = message;
```

Remarks

This property has no default value.

Quality

Quality

The Quality property is a read-write property that gets or sets the data quality to be used for the inserted value.

Syntax

```
aaHistClientSingleValueEntry.Quality = integer;
Result = aaHistClientSingleValueEntry.Quality;
```

Remarks

This property is only considered if you set it to a value of 1 (Bad). In this case, a NULL value is stored on the historian with a QualityDetail value of 24. In all other cases, the quality of the inserted value is determined by the QualityDetail property.

Valid values are:

-1 = None.

0 = Good

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

1 = Bad

16 = Doubtful

The default value is -1.

QualityDetail

QualityDetail

The QualityDetail property is a read-write property that gets or sets the data quality detail to be used for the inserted value.

Syntax

```
aaHistClientSingleValueEntry.DataQuality = integer;
Result = aaHistClientSingleValueEntry.DataQuality;
```

Remarks

The value must be present in the QualityMap table of the AVEVA Historian. If the value does not exist, any attempt to set the quality detail for the inserted value is ignored, and this property is reset to the default.

The default value is -1. In this case, the value is inserted with a QualityDetail value of 192 (Good quality).

Before you can set this property, you must have a valid server connection.

QualityDetailFieldDisable

QualityDetailFieldDisable

The QualityDetailFieldDisable property is a read-write property that gets or sets whether the **Quality Detail** box is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.QualityDetailFieldDisable = discrete;
Result = aaHistClientSingleValueEntry.QualityDetailFieldDisable;
```

Remarks

The default value is False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

QualityDetailFieldVisible

QualityDetailFieldVisible

The QualityDetailFieldVisible property is a read-write property that gets or sets whether the **Quality Detail** box is visible in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.QualityDetailFieldVisible = discrete;
Result = aaHistClientSingleValueEntry.QualityDetailFieldVisible;
```

Remarks

The default value is False.

QualityFieldDisable

QualityFieldDisable

The QualityFieldDisable property is a read-write property that gets or sets whether the **Quality** box is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.QualityFieldDisable = discrete;
Result = aaHistClientSingleValueEntry.QualityFieldDisable;
```

Remarks

The default value is False.

QualityFieldVisible

QualityFieldVisible

The QualityFieldVisible property is a read-write property that gets or sets whether the **Quality** box is visible in the control at runtime.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientSingleValueEntry.QualityFieldVisible = discrete;
Result = aaHistClientSingleValueEntry.QualityFieldVisible;
```

Remarks

The default value is False.

Any value the user specifies in the **Quality** box is ignored. The quality of the inserted value is determined by the value specified in the **Quality Detail** box.

RememberEnteredTags

RememberEnteredTags

The RememberEnteredTags property is a read-write property that gets or sets whether the control keeps track of previously entered tags and makes them available in the **Tag Name** box at runtime.

Syntax

```
aaHistClientSingleValueEntry.RememberEnteredTags = discrete;
Result = aaHistClientSingleValueEntry.RememberEnteredTags;
```

Remarks

The default value is True.

Servers

Servers

The Servers property is a read-write property that sets or gets the list of servers used by the control.

Syntax

```
aaHistClientSingleValueEntry.Servers = aaServers;
Result = aaHistClientSingleValueEntry.Servers;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This property references the aaServers object. For more information, see aaServer Object.

This property has no default value.

StringValue

StringValue

The StringValue property is a read-write property that sets or gets the value to be inserted for a tag.

Syntax

```
aaHistClientSingleValueEntry.StringValue = message;
Result = aaHistClientSingleValueEntry.StringValue;
```

Remarks

This property is provided for use within the InTouch HMI software, as the InTouch HMI software does not handle variant data types. The Value property is a variant datatype.

Setting this property automatically updates the Value and AnalogValue properties.

This property has no default value.

TagName

TagName

The TagName property is a read-write property that gets or sets the name of the current tag assigned to the control.

Syntax

```
aaHistClientSingleValueEntry.TagName = message;
Result = aaHistClientSingleValueEntry.TagName;
```

Remarks

Use this property to change an existing tag or to add a new tag.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagNameFieldDisable

TagNameFieldDisable

The TagNameFieldDisable property is a read-write property that gets or sets whether the **Tag Name** box is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.TagNameFieldDisable = discrete;
Result = aaHistClientSingleValueEntry.TagNameFieldDisable;
```

Remarks

The default value is False.

TagNameFieldVisible

TagNameFieldVisible

The TagNameFieldVisible property is a read-write property that gets or sets whether the **Tag Name** box is visible in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.TagNameFieldVisible = discrete;
Result = aaHistClientSingleValueEntry.TagNameFieldVisible;
```

Remarks

If you set this property to False, the Tag Picker button is also hidden at runtime.

The default value is True.

TagPickerButtonDisable

TagPickerButtonDisable

The TagPickerButtonDisable property is a read-write property that gets or sets whether the Tag Picker button to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the right of the **Tag Name** box is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.TagPickerButtonDisable = discrete;
Result = aaHistClientSingleValueEntry.TagPickerButtonDisable;
```

Remarks

The default value is False.

TagPickerButtonVisible

TagPickerButtonVisible

The TagPickerButtonVisible property is a read-write property that gets or sets whether the Tag Picker button to the right of the **Tag Name** box is visible in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.TagPickerButtonVisible = discrete;
Result = aaHistClientSingleValueEntry.TagPickerButtonVisible;
```

Remarks

The default value is True.

Tags

Tags

The Tags property is an array of aaTag objects that corresponds to the tags listed in the control's **Tagname** list.

Syntax

```
aaHistClientSingleValueEntry.Tags(n) = aaTag;
Result = aaHistClientSingleValueEntry.Tags(n);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

For more information on the aaTag object, see aaTag Object.

This property is not accessible in the InTouch HMI software.

This property has no default value.

TagType

TagType

The TagType property is a read-only property that returns the tag type for the current tag.

Syntax

```
Result = aaHistClientSingleValueEntry.TagType;
```

Remarks

Valid values are:

-1	The tag type can't be determined. This can occur if the tag is invalid or if there was a failure to connect to the server.
1	Analog
2	Discrete
3	String
4	Complex (not supported)
5	Event

The default value is -1.

TagValid

TagValid

The TagValid property is a read-only property that gets whether the current tag is valid.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaHistClientSingleValueEntry.TagValid;
```

Remarks

This value is set to False if the tag is invalid. The tag is invalid if there was a failure to connect to the server. The default value is False.

User

User

The User property is a read-write property that gets or sets the current user for a AVEVA Historian.

Syntax

```
aaHistClientSingleValueEntry.User = message;
Result = aaHistClientSingleValueEntry.User;
```

Remarks

Important: To insert data for a tag, a user must have wwAdministrator privileges for the AVEVA Historian.

If the value of the CurrentServerName property is changed, this property reflects the current user for the server. The default User is wwUser.

UseTimezone

UseTimezone

The UseTimezone property is a read-write property that is used for the timestamp of the inserted data value.

Syntax

```
aaHistClientSingleValueEntry.UseTimezone = aaUseTimeZoneEnumeration;
Result = aaHistClientSingleValueEntry.UseTimezone;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

For more information on the aaUseTimeZoneEnumeration enumeration, see aaUseTimeZoneEnumeration Enumeration.

The default value is 0.

Value

Value

This read-write property gets or sets the data value to be inserted for a tag.

Syntax

```
aaHistClientSingleValueEntry.Value = object;
Result = aaHistClientSingleValueEntry.Value;
```

Remarks

This property has no default value. It is not available in the .NET version of the control.

ValueEx

ValueEx

This read-write property gets or sets the data value to be inserted for a tag.

Syntax

```
aaHistClientSingleValueEntry.ValueEx = object;
Result = aaHistClientSingleValueEntry.ValueEx;
```

Remarks

This property has no default value. It is not writeable from the InTouch HMI software.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueFieldDisable

ValueFieldDisable

The ValueFieldDisable property is a read-write property that gets or sets whether the **Value** box is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.ValueFieldDisable = discrete;
Result = aaHistClientSingleValueEntry.ValueFieldDisable;
```

Remarks

The default value is False.

aaHistClientSingleValueEntry Control Methods

The aaHistClientSingleValueEntry control properties include:

- AddServer
- AddServerEx
- AddTag
- Connect
- CreateManualTag
- Disconnect
- Insert
- InsertValue
- Refresh
- Reset

AddServerEx

AddServerEx

The AddServer method adds a server to the list.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientSingleValueEntry.AddServerEx(message serverName, message loginName,
message password, [discrete bPersistPassword]);
```

Parameters

serverName

The name of the server to which to connect.

loginName

A valid user name for the server.

password

A valid password for the server.

bPersistPassword

If set to True, the password is remembered for the subsequent connection attempt. The password is only remembered for single application; the persisted password is not available to all applications.

Return Value

Returns True if the server can be added to the list; otherwise returns False.

Remarks

Important: A user must have administrative privileges for the AVEVA Historian to insert data for a tag.

- If the server is already part of the servers collection for the control and the provided log on information matches with the information already available with the server, the control switches to the new server. If the provided log on information does not match, the server is logged off and logged again with the new login credentials.
- This method does not actually attempt to connect to the server. The connection occurs when tags are added.
- All parameters are required. Errors, if any, are reported.

AddServer

AddServer

The AddServer method adds a server to the list.

Syntax

```
[Result=] aaHistClientSingleValueEntry.AddServer(message serverName, message loginName,
message password, [discrete bPersistPassword]);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

serverName

The name of the server to connect.

loginName

A valid user name for the server.

password

A valid password for the server.

bPersistPassword

Optional parameter. If set to True, the password is remembered for the subsequent connection attempts. The password is only remembered for single application; the persisted password is not available to all applications. The default value is True.

Return Value

Returns True if the server can be added to the list; otherwise returns False.

Remarks

Important: A user must have administrative privileges for the AVEVA Historian to insert data for a tag.

- If the server is already a part of the servers collection for the control and the provided log on information matches with the information already available with the server, the control switches to the new server. If the provided log on information does not match, the server is logged off and logged again with the new login credentials.
- This method does not actually attempt to connect to the server. The connection occurs when tags are added.

AddTag

AddTag

The AddTag method adds a tag for the control.

Syntax

```
[Result=] aaHistClientSingleValueEntry.AddTag(message tagName);
```

Parameters

tagName

The name of the tag to add.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the tag can be added; otherwise returns False.

Remarks

Calling this method and assigning a value to the TagName property have the same effect.

Connect

Connect

The Connect method establishes a connection to the current server.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Connect();
```

Return Value

Returns True if the connection can be made; otherwise returns False.

Remarks

This method is not required, since adding tags automatically causes a connection to the server. If the server is already logged on to, then this method prompts a reconnect.

CreateManualTag

CreateManualTag

The CreateManualTag method creates a tag with a manual data acquisition type. The tag is created in the historian database.

Syntax

```
[Result=] aaHistClientSingleValueEntry.CreateManualTag(message tagName, integer tagType);
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

tagName

The name of the tag to create.

tagType

The type of tag. 1 = Analog; 2 = Discrete; 3 = String

Return Value

For a description of return values, see the LastOperationResult property.

Remarks

If the manual tag can be added, it is set to the current tag.

Disconnect

Disconnect

The Disconnect method disconnects the control from the current server.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Disconnect();
```

Return Value

Returns True if the disconnect was successful; otherwise returns False.

Insert

Insert

The Insert method inserts a value for a manual tag.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Insert();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the value was inserted; otherwise returns False.

Remarks

This method has the same effect as a runtime user clicking the Insert button on the control interface.

If this method returns False, use the LastOperationResult, LastErrorMessage, and LastErrorDetails properties to determine the cause of the failure.

InsertValue

InsertValue

The InsertValue method inserts a value for a manual tag.

Syntax

```
[Result=] aaHistClientSingleValueEntry.InsertValue(message tagName, object tagValue,
[object dTime], [integer quality], [integer qualityDetail]);
```

Parameters

tagName

The tag for which the value is inserted.

tagValue

The value to insert.

dTime

The timestamp for the data value. If this parameter is not specified, the current date and time is used. You can use a message value for this parameter in an acceptable date/time format.

quality

The quality value to use.

qualityDetail

The quality detail to use.

Return Value

For a description of return values, see the LastOperationResult property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This method attempts to insert the specified value for the specified tag on the current server, regardless of the user interface settings. Likewise, the current settings for the user interface have no effect on the calling of this method.

If this method returns False, use the LastErrorMessage and LastErrorDetails properties to determine the cause of the failure.

Refresh

Refresh

The Refresh method forces a repaint of the control.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Refresh();
```

Reset

Reset

The Reset method clears the error information and values for the control.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Reset();
```

Remarks

Calling this method clears all of the text boxes in the user interface for the control. Also, any errors or success indicators from a previous operation are cleared.

aaHistClientSingleValueEntry Control Events

The aaHistClientSingleValueEntry control properties include:

- Change
- InsertComplete

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- InsertFail
- TagNameChanged
- ValueChanged

Change

Change

The Change event is triggered when the significant properties for the control are changed.

Syntax

```
aaHistClientSingleValueEntry.Change();
```

Remarks

This event is triggered if any of the following properties change:

- Tags
- Servers
- CurrentServerName
- User
- Pwd
- TagName
- DateTime (DateTimeString and InTouchDateTime)
- Value (AnalogValue and StringValue)
- Quality
- QualityDetail
- LastOperationResult, LastOperationSuccessful, LastErrorMessage, LastErrorDetails

InsertComplete

InsertComplete

The InsertComplete event is triggered when a data value insert operation succeeds.

Syntax

```
aaHistClientSingleValueEntry.InsertComplete();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This event is not triggered by the InsertValue method.

InsertFail

InsertFail

The InsertFail event is triggered when a data value insert operation fails.

Syntax

```
aaHistClientSingleValueEntry.InsertFail();
```

Remarks

This event is not triggered by the InsertValue method.

TagNameChanged

TagNameChanged

The TagNameChanged event is triggered when the TagName property changes.

Syntax

```
aaHistClientSingleValueEntry.TagNameChanged();
```

Remarks

This event is triggered in addition to the Change event.

ValueChanged

ValueChanged

The ValueChanged event is triggered when the Value, StringValue, or AnalogValue property changes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaHistClientSingleValueEntry.ValueChanged();
```

Remarks

This event is triggered in addition to the Change event.

aaFieldLabelPositionEnumeration Enumeration

Specifies where the label appears for text boxes in the control.

Value	Enumeration	Description
0	fldlblTop	The label appears above the boxes.
1	fldlblLeft	The label appears to the left of the boxes.

aaUseTimeZoneEnumeration Enumeration

Specifies the time zone.

Value	Enumeration	Description
0	tzConvertLocalToServer	Convert to the server time zone.
1	tzDoNotConvert	Do not convert to the server time zone.

Server Objects

Use the server-related objects to manage individual servers and the servers in the server list.

aaServer Object

The aaServer object encapsulates a SQL connection to a server. It provides properties for configuring the connection and methods for logging on and off the connection. It also includes read-only properties for obtaining information about the server and methods for working with the connection.

This object is referenced with parameters from other AVEVA Historian Client objects and controls.

aaServer Properties

The aaServer properties are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- BaseURLAddress
- Build
- Domain
- LoggedOn
- LoginID
- LoginTimeout
- MachineName
- Name
- Password
- PatchLevel
- QueryTimeout
- RetainPassword
- RevisionNumber
- SchemaVersion
- ServerName
- ServerType
- State
- TrustedConnection
- UseHttp
- VirtualDirectoryName

BaseURLAddress

The BaseURLAddress property is a read-write property that gets or sets the base URL address for the HTTP connection to the server.

Syntax

```
aaServer.BaseURLAddress = message;
Result = aaServer.BaseURLAddress;
```

Remarks

The default BaseURLAddress is http://localhost/.

Build

The Build property is a read-only property that returns the build number of the AVEVA Historian as a message.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
Result = aaServer.Build;
```

Return Value

Returns the build number as a message.

Remarks

An exception is thrown if no one is currently logged on to the server. Use the LoggedOn property to find out if the server is logged on.

This property has no default value.

Domain

The Domain property is a read-write property that gets or sets the domain string for the connection to the server.

Syntax

```
aaServer.Domain = message;
Result = aaServer.Domain;
```

Remarks

This property is useful in cases where the Windows integrated security requires the domain name to be specified.

The default is an empty message value ("").

LoginID

The LoginID property is a read-write property that gets and sets the login ID for the SQL Server.

Syntax

```
aaServer.LoginID = message;
Result = aaServer.LoginID;
```

Remarks

This login ID is used if Windows integrated security is not used. After a log on has occurred, changing the value of

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

this property has no effect until a log off and subsequent log on occurs.

The default LoginID is wwUser.

LoggedOn

The LoggedOn property is a read-only property that returns True if the server has been logged on.

Syntax

```
Result = aaServer.LoggedOn;
```

Return Value

Returns True if the server has been logged on; otherwise, returns False.

Remarks

The default value is False.

LoginTimeout

The LoginTimeout property is a read-write property that determines how long to wait, in seconds, for the connection to the server to be established before generating an error.

Syntax

```
aaServer.LoginTimeout = integer;
Result = aaServer.LoginTimeout;
```

Remarks

The default value is 5. A value of 0 means no timeout. If you do not use a timeout, the application waits indefinitely when trying to connect to a server, which may cause it to hang if the server is unavailable.

MachineName

The MachineName property is a read-only property that returns the actual computer name of the server.

Syntax

```
Result = aaServer.MachineName;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns the computer name as a message.

Remarks

An exception is thrown if no one is currently logged on to the server. Use the LoggedOn property to find out if the server is logged on.

This property has no default value.

Name

The Name property is a read-only property that returns the name of the server.

Syntax

```
Result = aaServer.Name;
```

Return Value

Returns the name of the server as a message.

Remarks

This property has no default value.

Password

The Password property is a read-write property that gets and sets the password for the connection to the server.

Syntax

```
aaServer.Password = message;
Result = aaServer.Password;
```

Remarks

This property is used if Windows integrated security is not used. After a logon has occurred, changing the value of this property has no effect until a logoff and subsequent logon occurs.

The default Password is wwUser.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

PatchLevel

The PatchLevel property is a read-only property that returns the patch level of the AVEVA Historian.

Syntax

```
Result = aaServer.PatchLevel;
```

Return Value

Returns the patch level as a message value.

Remarks

An exception is thrown if no one is currently logged on to the server. Use the LoggedOn property to find out if the server is logged on.

This property has no default value.

QueryTimeout

The QueryTimeout property is a read-write property that specifies the number of seconds to wait for a query to finish executing before the operation is aborted with a timeout error.

Syntax

```
aaServer.QueryTimeout = integer;
Result = aaServer.QueryTimeout;
```

Remarks

Changing the value of this property after log on has no effect until log off and subsequent log on.

The default value is 120. A value of 0 means no timeout. If you do not use a timeout, the application waits indefinitely when trying to query a server, which may cause it to hang if the server is unavailable.

RetainPassword

The RetainPassword property is a read-write property that indicates whether the password is stored in persistent storage.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaServer.RetainPassword = discrete;
Result = aaServer.RetainPassword;
```

Remarks

The default value is True.

RevisionNumber

The RevisionNumber property is a read-only property that gets the revision number of the AVEVA Historian.

Syntax

```
Result = aaServer.RevisionNumber;
```

Return Value

Returns the revision number as a message.

Remarks

An exception is thrown if no one is currently logged on to the server. Use the LoggedOn property to find out if the server is logged on.

This property has no default value.

SchemaVersion

The SchemaVersion property is a read-only property that gets the AVEVA Historian Client schema version for the server.

Syntax

```
Result = aaServer.SchemaVersion;
```

Return Value

Returns the schema version as a message.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

An exception is thrown if no one is currently logged on to the server. Use the LoggedOn property to find out if the server is logged on.

This property has no default value.

ServerName

The ServerName property is a read-only property that gets the name of the server.

Note: Provided for backward-compatibility only.

Syntax

```
Result = aaServer.ServerName;
```

Return Value

Returns the name of the server as a message.

Remarks

You can use the Name property to return the server name.

This property has no default value.

ServerType

The ServerType property is a read-only property that gets the server type.

Note: Provided for backward-compatibility only. Do not use for new applications.

Syntax

```
Result = aaServer.ServerType;
```

Return Value

Returns the server type as an enumeration. For more information, see aaServerType Enumeration.

Remarks

This property always returns a value of 1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

State

The State property is a read-only property that gets the state of the server.

Syntax

```
Result = aaServer.State;
```

Return Value

Returns the server state as an enumeration. For more information, see aaServerState Enumeration.

The default value is 2.

TrustedConnection

The TrustedConnection property is a read-write property that gets or sets the indication of whether Windows integrated security is used when logging on to the AVEVA Historian.

Syntax

```
aaServer.TrustedConnection = discrete;
Result = aaServer.TrustedConnection;
```

Remarks

True = Windows integrated security is used; False = A SQL Server login ID and password is used.

Changing the value of this property after logon has no effect until logoff and subsequent logon.

The default value is False.

UseHttp

The UseHttp property is a read-write property that controls whether to use HTTP to access the SQL Server.

Syntax

```
aaServer.UseHttp = discrete;
Result = aaServer.UseHttp;
```

Remarks

If set to True, HTTP is used. This property also creates the connection object, if necessary.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The default value is False.

VirtualDirectoryName

The VirtualDirectoryName property is a read-write property that gets or sets the virtual directory name.

Syntax

```
aaServer.VirtualDirectoryName = message;
Result = aaServer.VirtualDirectoryName;
```

Remarks

The default directory name is ActiveFactory.

aaServer Methods

The aaServer methods are:

- LogOff
- LogOn

LogOff

The LogOff method terminates the connection to the server.

Syntax

```
[Result=] aaServer.LogOff();
```

Remarks

Repeated calls to this method are harmless and do not result in further state change events. For more information on state change events, see OnServerStateChange.

LogOn

The LogOn method creates a connection (logs on) to the server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaServer.LogOn(out message statusMessage);
```

Parameters

statusMessage

Information about the result of the log on attempt.

Return Value

Returns True if the log on was successful; otherwise, returns False.

Remarks

The server must be configured before calling the LogOn method. Changes made to the server configuration after a logon do not take effect until after a logoff and subsequent logon.

This method produces state change events. For more information, see OnServerStateChange.

aaServers Object

The aaServers object is a collection of aaServer instances. This object provides methods and properties for maintaining a sorted list of servers. Use the properties to get information regarding the number of servers in the collection. Use the methods to perform basic functions for the collection, such as adding or removing servers. Events for this object indicate when servers are added to the list, removed from the list, updated within the list, or when a server's state changes.

This object is referenced with parameters from other AVEVA Historian Client objects and controls.

aaServers Properties

The aaServers properties are:

- ApplicationName
- Count
- Items

ApplicationName

The ApplicationName property gets or sets the application name to be used in profile logs when making a request to the AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaServers.ApplicationName = message;
[Result=] aaServers.ApplicationName;
```

Remarks

The name must be set prior to a server in the list initiating a log on.

Count

The Count property is a read-only property that gets the number of servers in the server list.

Syntax

```
Result = aaServers.Count;
```

Return Value

Returns the number as an integer.

Remarks

The default value is 0.

Items

The Items property is a read-only property that returns the list of servers in an array.

Syntax

```
Result = aaServers.Items;
```

Return Value

Returns the aaServer object. The same aaServer object instances that are in the server list are in the array. For more information on the aaServer object, see aaServer Object.

Remarks

This property is not supported in the InTouch HMI software.

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaServers Methods

The aaServers methods are:

- Add
- GetServer
- Remove
- Update

Add

The Add method adds a server to the server list.

Syntax

```
[Result=] aaServers.Add(message name);
```

Parameters

name

The name of the server to add.

Return Value

If a server with the given name is already in the list, the aaServer object for that server is returned. Otherwise, a new server with the given name is added to the list and the aaServer object for the new server is returned. For more information on the aaServer object, see aaServer Object.

GetServer

The GetServer method gets the aaServer object for a server from the server list.

Syntax

```
[Result=] aaServers.GetServer(message name);
```

Parameters

name

The name of the server to get.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

If the server exists, the aaServer object is returned; otherwise, a NULL is returned. For more information on the aaServer object, see aaServer Object.

Remove

The Remove method removes the specified server from the list.

Syntax

```
[Result=] aaServers.Remove(aaServer server);
```

Parameters

server

The name of the server to remove.

Return Value

If this method returns True, the instance was removed from the list. This method returns False if the exact instance is not in the list, and the list remains unchanged.

Remarks

The aaServer instance passed as an argument to the OnServerRemoved event is the same instance that was in the server list.

Update

The Update method updates the specified server in the server list.

Syntax

```
[Result=] aaServers.Update(aaServer server);
```

Parameters

server

The name of the server to update.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if the given aaServer instance is currently in the server list; otherwise, False is returned.

Remarks

The Update method serves two purposes:

- It causes the list of servers (which is the list that appears in the **Server Configuration** dialog box) to be persisted, if persistence is in effect. For example, the AVEVA Historian Client Trend and the AVEVA Historian Client Query applications run with persistence in effect; when you start these applications, you see previously-configured servers in the list. Controls, however, do not necessarily run with persistence in effect. When changes are made to properties in an instance of the aaServer object, they are not persisted until the Update method is called.
- It causes an OnServerUpdated event to fire. This allows other parts of the application to respond to changes in any of the servers in the servers list. When changes are made to properties in an instance of aaServer, no event is fired to report the change until the Update method is called.

The aaServer instance must be the exact same instance, not an instance with the same name. If the instance is not in the list, then the list is not updated.

The aaServer instance passed as an argument to the OnServerUpdated event is the exact same instance that is in the list.

aaServers Events

The aaServers events are:

- OnServerAdded
- OnServerUpdated
- OnServerRemoved
- OnServerStateChange

These events are not accessible from the InTouch HMI software.

OnServerAdded

The OnServerAdded event is triggered when a new server is added to the server list.

Syntax

```
aaServers.OnServerAdded(object source, aaServerListChangeArgs args);
```

Parameters

source

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

This parameter is not used.

args

The server state change arguments. For more information on the aaServerListChangeArgs object, see aaServerListChangeArgs Object.

Remarks

This event is not accessible from the InTouch HMI software.

OnServerUpdated

The OnServerUpdated event is triggered when a server that is currently in the server list is updated.

Syntax

```
aaServers.OnServerUpdated(object source, aaServerListChangeArgs args);
```

Parameters

source

The object. For more information on specifying an object, see Object.

args

The server state change arguments. For more information on the aaServerListChangeArgs object, see aaServerListChangeArgs Object.

Remarks

This event is not accessible from the InTouch HMI software.

OnServerRemoved

The OnServerRemoved event is triggered when a server is removed from the server list.

Syntax

```
aaServers.OnServerRemoved(object source, aaServerListChangeArgs args);
```

Parameters

source

The object. For more information on specifying an object, see Object.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

args

The server state change arguments. For more information on the aaServerListChangeArgs object, see aaServerListChangeArgs Object.

Remarks

This event is not accessible from the InTouch HMI software.

OnServerStateChange

The OnServerStateChange event is triggered when the state of a server is changed.

Syntax

```
aaServers.OnServerStateChange(object source, aaServerStateChangeArgs args);
```

Parameters

source

The object. For more information on specifying an object, see Object.

args

The server state change arguments. For more information on the aaServerStateChangeArgs object, see aaServerStateChangeArgs Object.

Remarks

This event is not accessible from the InTouch HMI software.

Instantiating an aaServers Object

The aaServers object is a cocreatable object. That is, it can be instantiated on its own, instead of only being created when used by a method of another object. In the InTouch HMI software, you can use the OLE_CreateObject() function to instantiate the aaServers object. The ProgID for the aaServers object is `ArchestrA.HistClient.Database.aaServers`.

For example:

```
OLE_CreateObject (%Man, "ArchestrA.HistClient.Database.aaServers");
```

aaServerListChangeArgs Object

The aaServerListChangeArgs object is used to return name of the aaServer instance that changed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Properties

The aaServerListChangeArgs object property is:

- Server

Server

The Server property is a read-only property that gets the aaServer instance that was either added, updated, or removed during the operation that produced the event.

Syntax

```
Result = aaServerListChangeArgs.Server;
```

Return Value

The aaServer instance. For more information on the aaServer object, see aaServer Object.

Remarks

This property has no default value.

aaServerStateChangeArgs Object

The aaServerListChangeArgs object is used to return state changes for the server.

Properties

The aaServerStateChangeArgs object properties are:

- Server
- State
- When
- Message

Server

The Server property is a read-only property that gets the server that changed state.

Syntax

```
Result = aaServerStateChangeArgs.Server;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

The aaServer instance. For more information on the aaServer object, see aaServer Object.

Remarks

This property has no default value

State

The State property is a read-only property that gets the state to which the server changed.

Syntax

```
Result = aaServerStateChangeArgs.State;
```

Return Value

The aaServerState enumeration. For more information on the aaServerState enumeration, see aaServerState Enumeration.

Remarks

This property has no default value.

When

The When property is a read-only property that gets the date and time of the state change.

Syntax

```
Result = aaServerStateChangeArgs.When;
```

Return Value

The date/time stamp. For more information on the DateTime data type, see DateTime.

Remarks

This property has no default value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Message

The Message property is a read-only property that gets any message available for the state change, such as a detailed error message.

Syntax

```
Result = aaServerStateChangeArgs.Message;
```

Return Value

Returns the message as a message value.

Remarks

This property has no default value.

aaServerState Enumeration

Specifies the allowed states of a server.

Value	Enumeration	Description
0	LoggedOn	There is a connection to the server.
1	Error	The connection can not be made to the server.
2	LoggedOff	No connection has been attempted to the server or the LogOff method has been called.
3	LoggingOn	An attempt to connect to the server has begun and has not yet succeeded, nor has the attempt yet timed out.

aaServerType Enumeration

Specifies the types of a server.

Note: Provided for backward-compatibility only. Do not use for new applications.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
0	isSvrPdssrv	For backward-compatibility only.
1	isSvrSQLServer	For backward-compatibility only.
2	isSvrNotInSQL	For backward-compatibility only.

Other Objects

This section describes how to work with objects not covered in previous sections.

aaTag Object

When tags are read from an AVEVA Historian database, they are each stored in an instance of the aaTag object. This object provides read-only properties for accessing the information about the tag that was obtained from the AVEVA Historian.

Using aaTag in an Application

You can use the aaTag object's properties in runtime scripts in your application to get configuration information for a tag. Also, this object is referenced with parameters from other AVEVA Historian Client objects and controls.

aaTag Properties

The aaTag properties are:

- DateCreated
- Description
- IOAddress
- MaxRaw
- MinRaw
- MinEU
- MaxEU
- Message0
- Message1
- Mode
- Name
- RawType
- Server
- Type

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- TypeAsTagType
- Units

DateCreated

DateCreated

This read-only property returns the date that the tag was created.

Syntax

```
Result = aaTag.DateCreated;
```

Return Value

The return value is of type DateTime.

Remarks

The default value is the current time.

Description

Description

This read-only property returns the description of the tag.

Syntax

```
Result = aaTag.Description;
```

Return Value

The return value is a message value.

Remarks

The default value is NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IOAddress

IOAddress

This read-only property returns the I/O address of the tag.

Syntax

```
Result = aaTag.IOAddress;
```

Return Value

The return value is a message.

Remarks

The default value is NULL.

MaxRaw

MaxRaw

This read-only property returns the maximum value of the raw acquired value.

Syntax

```
Result = aaTag.MaxRaw;
```

Return Value

The return value is a real.

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MinRaw

MinRaw

This read-only property returns the minimum value of the raw acquired value.

Syntax

```
Result = aaTag.MinRaw;
```

Return Value

The return value is a real.

Remarks

The default value is 0.

MinEU

MinEU

This read-only property returns the minimum value of the tag, measured in engineering units.

Syntax

```
Result = aaTag.MinEU;
```

Return Value

The return value is a real.

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MaxEU

MaxEU

This read-only property returns the maximum value of the tag, measured in engineering units.

Syntax

```
Result = aaTag.MaxEU;
```

Return Value

The return value is a real.

Remarks

The default value is 0.

Message0

Message0

This read-only property returns the message associated with the FALSE state of the discrete tag. A discrete tag set to 0 is in the FALSE state.

Syntax

```
Result = aaTag.Message0;
```

Return Value

The return value is a message.

Remarks

The default value is NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Message1

Message1

This read-only property returns the message associated with the TRUE state of the discrete tag. A discrete tag set to 1 is in the TRUE state.

Syntax

```
Result = aaTag.Message1;
```

Return Value

The return value is a message.

Remarks

The default value is NULL.

Mode

Mode

This read-only property returns the storage mode of this tag as a localized string.

Syntax

```
Result = aaTag.Mode;
```

Return Value

The return value is a message.

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name

Name

This read-only property returns the name of the tag.

Syntax

```
Result = aaTag.Name;
```

Return Value

The return value is a message.

Remarks

The default value is the name that was specified when the tag was created.

RawType

RawType

This read-only property returns the numeric type for the raw value. 1 = Euro Float (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).

Syntax

```
Result = aaTag.RawType;
```

Return Value

The return value is an integer.

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Server

Server

This read-only property returns the server associated with the tag.

Syntax

```
Result = aaTag.Server;
```

Return Value

The return value is an aaServer object. For more information, see aaServer Object. The server cannot be changed after construction.

Remarks

The default value is the name that was specified when the tag was created.

Type

Type

This read-only property returns the type of the tag, converted to a localized string.

Syntax

```
Result = aaTag.Type;
```

Return Value

The return value is a message.

Remarks

The default value is UnknownTag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TypeAsTagType

TypeAsTagType

This read-only property returns the type of the tag.

Syntax

```
Result = aaTag.TypeAsTagType;
```

Return Value

The return value is of type aaTagType. For more information on the aaTagType enumeration, see aaTagType Enumeration.

The default value is 0.

Units

Units

This read-only property returns the unit of measure. For example mph, grams, and pounds.

Syntax

```
Result = aaTag.Units;
```

Return Value

The return value is a message.

Remarks

The default value is NULL.

aaHistClientWorkbookRunner and aaHistClientReportRunner Objects

The aaHistClientWorkbookRunner and aaHistClientReportRunner objects are used when reports are published to the AVEVA Information Server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaHistClientWorkbookRunner Object

The aaHistClientWorkbookRunner object is a control that is used to run reports created with the AVEVA Historian Client Workbook. There is no user interface for this control.

You can use the aaHistClientWorkbookRunner control's properties and methods in runtime scripts in your application to run existing Workbook files (.xlsx) and output the results (.htm).

aaHistClientWorkbookRunner Object Properties

The aaHistClientWorkbookRunner object properties include:

- ErrDescription
- ErrNumber
- OutputFile
- SourceFile
- ExcelVisible

ErrDescription

ErrDescription

The ErrDescription property is a read-only property that returns an error message if the Run method fails.

Syntax

```
Result = aaHistClientWorkbookRunner.ErrDescription;
```

Return Value

The return value is a message. The error message describes the reason for the failure.

Remarks

The default is an empty message value ("").

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ErrNumber

ErrNumber

The ErrNumber property is a read-only property that returns an error code number if the Run method fails.

Syntax

```
Result = aaHistClientWorkbookRunner.ErrNumber;
```

Return Value

The return value is an integer.

Remarks

The default value is 0.

OutputFile

OutputFile

The OutputFile property is a read-write property that is used to specify the file to be created as a result of running the report.

Syntax

```
aaHistClientWorkbookRunner.OutputFile = message;
Result = aaHistClientWorkbookRunner.OutputFile;
```

Remarks

You must specify the entire path and include the .htm extension.

The default is an empty message value ("").

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SourceFile

SourceFile

The SourceFile property is a read-write property that specifies the name of the Word template file (.dotx, .dotm, or .dot) to use to generate the report.

Syntax

```
aaHistClientWorkbookRunner.SourceFile = message;
Result = aaHistClientWorkbookRunner.SourceFile;
```

Remarks

You must specify the entire path and include the extension (.dotx, .dotm, or .dot).

The default is an empty message value ("").

ExcelVisible

ExcelVisible

The ExcelVisible property is a read-write property that specifies whether or not the Excel application user interface is visible when the report is run.

Syntax

```
aaHistClientWorkbookRunner.ExcelVisible = discrete;
Result = aaHistClientWorkbookRunner.ExcelVisible;
```

Remarks

If set to True, Excel is visible. If set to False, Excel is not visible. The default value is False.

Setting this property to True is useful when you are testing the report generation.

The default value is False.

aaHistClientWorkbookRunner Methods

The aaHistClientWorkbookRunner control methods include:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Run
- RunReport
- RunReport2

Run

Run

The Run method processes the Workbook report.

Syntax

```
[Result=] aaHistClientWorkbookRunner.Run();
```

Return Value

Returns True if the report generation was successful; otherwise returns False.

Remarks

When this method is called, the following occurs:

1. Excel starts. Excel is visible only if the ExcelVisible property was set to True.
2. The Workbook file (.xlsx) specified by the SourceFile property opens.
3. The report runs.
4. Excel closes.

If you want to use binding options for the report, use the RunReport method.

RunReport

RunReport

The RunReport method processes the Workbook report. This method uses the date/time binding feature of Workbook.

Syntax

```
[Result=] aaHistClientWorkbookRunner.RunReport(
   message inputFile,
   message outputFile,
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
    message outputPrefix,
    integer outputFormat,
    message tagString,
    integer NSFolderKey,
    message nameSpace,
    integer dateMode,
    message startDate,
    message endDate,
    integer duration);
```

Parameters

inputFile

The name of the source file for the report generation, including the full path. Valid file types are .htm, .xlsx, and .xlt.

outputFile

The name of the output file that is generated, including the full path. If this parameter is set to an empty string (" "), then a file name is generated automatically according to the following formula:
`OutputFile = OutputPrefix + InputFile + year + month + day + _ + hour + minute + second`

outputPrefix

The value that is prepended to the output file name. If you specify an empty string (" "), no prefix is prepended. The outputPrefix parameter is only used if the outputFile parameter is an empty string.

outputFormat

The file type for the output file. Valid values are:
0 = Native. That is, if the source file is an .htm file, the output file is an .htm file. If the source file is an .xlsx or .xlt file, the output file is an .xlsx file.
1 = .htm
2 = .xlsx
3 = .xlt

tagString

A comma separated list of strings to be used for the AFTagBinding named range. Valid formats are:

```
    "<tagname1>,<tagname2>"
    "'<tagname1>','<tagname2>'"
```

For example:
"ReactLevel,ReactTemp"
"'ReactLevel','ReactTemp'"

NSFolderKey

Reserved for future use. This parameter cannot be blank. Specify a value (for example, 0) for this parameter, even though it has no effect.

nameSpace

Reserved for future use. This parameter cannot be blank. Specify an empty string ("") for this parameter, even though it has no effect.

dateMode

Determines the values used for the AFStartBinding and AFEndBinding named ranges. Valid values are:
0 = Use specific start and end times.
1 = Use a duration relative to the current time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2 = Use a duration relative to the specified start time.
3 = Use a duration relative to the specified end time.
Use the *startDate*, *endDate*, and *Duration* parameters to specify the dates.

startDate

A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the *dateMode* parameter is set to 1 or 3, this parameter is ignored.
If the *dateMode* parameter is set to 0, this value indicates the specific date/time to be used for the AFStartBinding range.
If the *dateMode* parameter is set to 2, then "`rel`" is used for the AFStartBinding range and '`+Duration(StartDate)`' is used for the AFEndBinding range.

endDate

A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the *dateMode* parameter is set to 1 or 2, this parameter is ignored.
If the *dateMode* parameter is set to 0, this value indicates the specific date/time to be used for the AFEndBinding range.
If the *dateMode* parameter is set to 3, then "`rel`" is used for the AFStartBinding range and '`+Duration(EndDate)`' is used for the AFEndBinding range.

Duration

The time span, in seconds, used for date/time calculations. This value cannot be a negative number.
If the *dateMode* parameter is set to 0, this value is ignored.
If the *dateMode* parameter is set to 1, "`rel`" is used for the AFStartBinding range and '`-Duration()`' is used for the AFEndBinding range.
If the *dateMode* parameter is set to 2, "`rel`" is used for the AFStartBinding range and '`+Duration(StartDate)`' is used for the AFEndBinding range.
If the *dateMode* parameter is set to 3, "`rel`" is used for the AFStartBinding range and '`-Duration(EndDate)`' is used for the AFEndBinding range.

Return Value

Returns the output file name if the report generation was successful; otherwise, an empty string is returned.

Remarks

When this method is called, the following occurs:

1. Excel starts. Excel is visible only if the ExcelVisible property was set to True.
2. The Workbook file (.xlsx) specified by the SourceFile property opens.
3. The binding information in the workbook file is updated.
4. The report runs and the output is saved as an .htm file as specified in the OutputFile property.
5. Excel closes.

To run a report without using the binding options, use the method. To run a report that only uses additional binding options for custom filters, use the RunReport2 method.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RunReport2

RunReport2

The RunReport2 method processes the Workbook report. This method uses the date/time binding feature of Workbook, plus custom binding filters.

Syntax

```
[Result=] aaHistClientWorkbookRunner.RunReport2(
   message inputFile,
   message outputFile,
   message outputPrefix,
   integer outputFormat,
   message tagString,
   integer NSFolderKey,
   message nameSpace,
   integer dateMode,
   message startDate,
   message endDate,
   integer duration
   message customFilters);
```

Parameters

inputFile

The name of the source file for the report generation, including the full path. Valid file types are .htm, .xlsx, and .xlt.

outputFile

The name of the output file generated, including the full path. If this parameter is set to an empty string (" "), then a file name is generated automatically according to the following formula:
`OutputFile = OutputPrefix + InputFile + year + month+ day + _ + hour + minute + second`

outputPrefix

The value prepended to the output file name. If you specify an empty string (" "), no prefix is prepended.
The outputPrefix parameter is only used if the outputFile parameter is an empty string.

outputFormat

The file type for the output file. Valid values are:
0 = Native. That is, if the source file is an .htm file, the output file is an .htm file. If the source file is an .xlsx or .xlt file, the output file is an .xlsx file.
1 = .htm
2 = .xlsx
3 = .xlt

tagString

A comma separated list of strings to be used for the AFTagBinding named range. Valid formats are:
"*<tagname1>*,*<tagname2>*"

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

"'<*tagname1*>','<*tagname2*>'"
For example:
"ReactLevel,ReactTemp"
"'ReactLevel','ReactTemp'"

NSFolderKey

Reserved for future use. This parameter cannot be blank. Specify a value (for example, 0) for this parameter, even though it has no effect.

nameSpace

Reserved for future use. This parameter cannot be blank. Specify an empty string ("") for this parameter, even though it has no effect.

dateMode

Determines the values used for the AFStartBinding and AFEndBinding named ranges. Valid values are:
0 = Use specific start and end times.
1 = Use a duration relative to the current time.
2 = Use a duration relative to the specified start time.
3 = Use a duration relative to the specified end time.
Use the *startDate*, *endDate*, and *Duration* parameters to specify the dates.

startDate

A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the *dateMode* parameter is set to 1 or 3, this parameter is ignored.
If the *dateMode* parameter is set to 0, this value indicates the specific date/time to be used for the AFStartBinding range.
If the *dateMode* parameter is set to 2, then "`rel`" is used for the AFStartBinding range and '`+Duration(StartDate)`' is used for the AFEndBinding range.

endDate

A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the *dateMode* parameter is set to 1 or 2, this parameter is ignored.
If the *dateMode* parameter is set to 0, this value indicates the specific date/time to be used for the AFEndBinding range.
If the *dateMode* parameter is set to 3, then "`rel`" is used for the AFStartBinding range and '`+Duration(EndDate)`' is used for the AFEndBinding range.

Duration

The time span, in seconds, used for date/time calculations. This value cannot be a negative number.
If the *dateMode* parameter is set to 0, this value is ignored.
If the *dateMode* parameter is set to 1, "`rel`" is used for the AFStartBinding range and '`-Duration()`' is used for the AFEndBinding range.
If the *dateMode* parameter is set to 2, "`rel`" is used for the AFStartBinding range and '`+Duration(StartDate)`' is used for the AFEndBinding range.
If the *dateMode* parameter is set to 3, "`rel`" is used for the AFStartBinding range and '`-Duration(EndDate)`' is used for the AFEndBinding range.

customFilters

A string of name-value pairs used to pass information from the AVEVA Information Server to the workbook file before the report is run.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The format for the string is as follows:
<name>=<value>
To pass more than one name-value pair, join them with ampersands. For example:
<name>=<value>&<name>=<value>
The parameter name that you use must correspond to an existing named range in the workbook that starts with "AFBinding."
The value you specify in the name-value pair is used for the corresponding named range in the workbook. You can specify multiple values if you separate them with commas.
For example, you workbook contains the AFBindingReportValue and AFBindingReportText named ranges. You want to pass a value of 5 for the report value and Line1 and Line2 for the ReportText. The customFilters parameter is:
ReportValue=5&ReportText=Line2,Line2

Return Value

Returns the output file name if the report generation was successful; otherwise, an empty string is returned.

Remarks

When this method is called, the following occurs:

1. Excel starts. Excel is visible only if the ExcelVisible property was set to True.
2. The Workbook file (.xlsx) specified by the SourceFile property opens.
3. The binding information in the workbook file is updated.
4. The report runs and the output is saved as an .htm file as specified in the OutputFile property.
5. Excel closes.

To run a report without using the binding options, use the Run method. To run a report that only uses the date/time binding options, use the RunReport method.

aaHistClientReportRunner Object

The aaHistClientReportRunner object is a control that is used to run reports created with the AVEVA Historian Client Report. There is no user interface for this control.

You can use the aaHistClientReportRunner object's properties and methods in runtime scripts in your application to run existing Report files and output the results (.htm).

aaHistClientReportRunner Object Properties

The aaHistClientReportRunner object properties include:

- ErrDescription
- ErrNumber
- OutputFile

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- SourceFile
- WordVisible

ErrDescription

ErrDescription

The ErrDescription property is a read-only property that returns an error message if the Run method fails.

Syntax

```
Result = aaHistClientReportRunner.ErrDescription;
```

Return Value

The return value is a message. The error message describes the reason for the failure.

Remarks

The default is an empty message value ("").

ErrNumber

ErrNumber

The ErrNumber property is a read-only property that returns an error code number if the Run method fails.

Syntax

```
Result = aaHistClientReportRunner.ErrNumber;
```

Return Value

The return value is an integer.

Remarks

The default value is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

OutputFile

OutputFile

The OutputFile property is a read-write property that is used to specify the file to be created as a result of running the report.

Syntax

```
aaHistClientReportRunner.OutputFile = message;
Result = aaHistClientReportRunner.OutputFile;
```

Remarks

You must specify the entire path and include the .htm extension.

The default is an empty message value ("").

SourceFile

SourceFile

The SourceFile property is a read-write property that specifies the name of the Word template file (.dotx, .dotm, or .dot) to use to generate the report.

Syntax

```
aaHistClientReportRunner.SourceFile = message;
Result = aaHistClientReportRunner.SourceFile;
```

Remarks

You must specify the entire path and include the extension (.dotx, .dotm, or .dot).

The default is an empty message value ("").

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

WordVisible

WordVisible

The WordVisible property is a read-write property that specifies whether or not the Word application user interface is visible when the report is run.

Syntax

```
aaHistClientReportRunner.WordVisible = discrete;
Result = aaHistClientReportRunner.WordVisible;
```

Remarks

If set to True, Word is visible. If set to False, Word is not visible. The default value is False.

Setting this property to True is useful when you are testing the report generation.

The default value is False.

aaHistClientReportRunner Object Methods

The aaHistClientReportRunner object has single method:

- Run

Run

Run

The Run method processes the Word report.

Syntax

```
[Result=] aaHistClientReportRunner.Run();
```

Return Value

Returns True if the report generation was successful; otherwise returns False.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

When this method is called, the following occurs:

1. Word starts. Word is visible only if the WordVisible property was set to True.
2. The report file (.htm) specified by the SourceFile property opens.
3. The report runs and is saved as the .htm file specified by the OutputFile property.
4. Word closes.

Workbook and Report Automation Objects

The AVEVA Historian Client Workbook and the AVEVA Historian Client automation objects allow you to automate the AVEVA Historian Client Workbook and Report from a scripting environment, such as Visual Basic for Applications.

AVEVA Historian Client Workbook Object

To automate the generation of reports from the AVEVA Historian Client Workbook, use the AVEVA Historian Client Workbook object within the scripting environment.

AVEVA Historian Client Workbook Object Methods

The AVEVA Historian Client Workbook object methods are:

- AddServer
- Auto_Close
- Auto_Open
- GetLastError
- RunReport
- AVEVA Historian Client Workbook Menu Methods
- AVEVA Historian Client Workbook Functions

AddServer

AddServer

The AddServer method adds a server to the list of servers for the current workbook.

Syntax

```
ActiveFactoryWorkbook.AddServer(message serverName, message loginName, message password)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

serverName

The name of the server to which to connect.

loginName

A valid user name for the server.

password

A valid password for the server.

Remarks

Use the keyword "CALL" before the method name to invoke the method.

Auto_Close

Auto_Close

The Auto_Close method removes the AVEVA Historian Client toolbar and resets the main menu for Excel.

Syntax

```
ActiveFactoryWorkbook.Auto_Close()
```

Auto_Open

Auto_Open

The Auto_Open method adds the AVEVA Historian Client toolbar and adds the AVEVA Historian Client menu to the main menu for Excel.

Syntax

```
ActiveFactoryWorkbook.Auto_Open();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GetLastError

GetLastError

The GetLastError method returns a message for any error that occurs when the report is run using the RunReport method.

Syntax

```
[Result=] ActiveFactoryWorkbook.GetLastError();
```

Return Value

Returns a message value containing the error for the RunReport method. If an empty string is returned, then an error has occurred. If the output file name is returned, a warning may have occurred.

Remarks

Possible errors are:

- Only available when a server is present, click OK to add a server.
- The input file specified does not exist.
- The output format specified is invalid.
- The DateMode argument must be 0, 1, 2, or 3.
- The specified start date is invalid.
- The specified end date is invalid.
- The specified duration is invalid.
- TagString is not empty and AFTagBinding does not exist.
- Invalid TagString format.
- Warning: The AFTagBinding range in the report contains no tags and no tags have been passed in.
- The <filename> range must be defined.
- An error occurred while attempting to save the file.
- <filename> is unknown.
- No values for <filename>.
- Wizard Error.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

RunReport

RunReport

The RunReport method processes the Workbook report. This method uses the date/time binding feature of Workbook, plus custom binding filters. need x-refs

Syntax

```
[Result=] ActiveFactoryWorkbook.RunReport(
   message inputFile,
   message outputFile,
   message outputPrefix,
   integer outputFormat,
   message tagString,
   integer NSFolderKey,
   message nameSpace,
   integer dateMode,
   message startDate,
   message endDate,
   integer duration
   message customFilters);
```

Parameters

inputFile

The name of the source file for the report generation, including the full path. Valid file types are .htm, .xlsx, and .xlt.

outputFile

The name of the output file generated, including the full path. If this parameter is set to an empty string (""), then a file name is generated automatically according to the following formula:
`OutputFile = OutputPrefix + InputFile + _ + year + month + day + hour + minute + second`

outputPrefix

The value prepended to the output file name. If you specify an empty string (""), no prefix is prepended.
The outputPrefix parameter is only used if the outputFile parameter is an empty string.

outputFormat

The file type for the output file. Valid values are:
0 = Native. That is, if the source file is an .htm file, the output file is an .htm file. If the source file is an .xlsx or .xlt file, the output file is an .xlsx file.
1 = .htm
2 = .xlsx
3 = .xlt

tagString

A comma separated list of strings to be used for the AFTagBinding named range. If the AFTagBinding range does not exist, and this parameter is set to any value other than an empty string (""), an error is raised. Valid formats

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

are:
"<tagname1>,<tagname2>"
"'<tagname1>','<tagname2>'"
For example:
"ReactLevel,ReactTemp"
"'ReactLevel','ReactTemp'"

NSFolderKey

Reserved for future use. This parameter cannot be blank. Specify a value (for example, 0) for this parameter, even though it has no effect.

nameSpace

Reserved for future use. This parameter cannot be blank. Specify an empty string ("") for this parameter, even though it has no effect.

dateMode

Determines the values used for the AFStartBinding and AFEndBinding named ranges. An error is raised if the binding ranges do not exist or if this parameter is blank. Valid values are:
0 = Use specific start and end times.
1 = Use a duration relative to the current time.
2 = Use a duration relative to the specified start time.
3 = Use a duration relative to the specified end time.
Use the *startDate*, *endDate*, and *Duration* parameters to specify the dates.

startDate

A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the *dateMode* parameter is set to 1 or 3, this parameter is ignored.
If the *dateMode* parameter is set to 0, this value indicates the specific date/time to be used for the AFStartBinding range.
If the *dateMode* parameter is set to 2, then "`rel`" is used for the AFStartBinding range and '`+Duration(StartDate)`' is used for the AFEndBinding range.

endDate

A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the *dateMode* parameter is set to 1 or 2, this parameter is ignored.
If the *dateMode* parameter is set to 0, this value indicates the specific date/time to be used for the AFEndBinding range.
If the *dateMode* parameter is set to 3, then "`rel`" is used for the AFStartBinding range and '`+Duration(EndDate)`' is used for the AFEndBinding range.

Duration

The time span, in seconds, used for date/time calculations. This value cannot be a negative number.
If the *dateMode* parameter is set to 0, this value is ignored.
If the *dateMode* parameter is set to 1, "`rel`" is used for the AFStartBinding range and '`-Duration()`' is used for the AFEndBinding range.
If the *dateMode* parameter is set to 2, "`rel`" is used for the AFStartBinding range and '`+Duration(StartDate)`' is used for the AFEndBinding range.
If the *dateMode* parameter is set to 3, "`rel`" is used for the AFStartBinding range and '`-Duration(EndDate)`' is used for the AFEndBinding range.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

customFilters

A string of name-value pairs used to pass information from the AVEVA Information Server to the workbook file before the report is run.
The format for the string is as follows:
<name>=<value>
To pass more than one name-value pair, join them with ampersands. For example:
<name>=<value>&<name>=<value>
The parameter name that you use must correspond to an existing named range in the workbook that starts with "AFBinding."
The value you specify in the name-value pair is used for the corresponding named range in the workbook. You can specify multiple values if you separate them with commas.
For example, your workbook contains the AFBindingReportValue and AFBindingReportText named ranges. You want to pass a value of 5 for the report value and Line1 and Line2 for the ReportText. The customFilters parameter is:
ReportValue=5&ReportText=Line2,Line2

Return Value

Returns the output file name if the report generation was successful; otherwise, an empty string is returned.

AVEVA Historian Client Workbook Menu Methods

AVEVA Historian Client Workbook Menu Methods

The following methods execute Workbook menu commands

Method	Used to
mnuAbout	Open the **About** dialog box.
mnuAddDSN	Open the **Server List Configuration** dialog box.
mnuAggregates	Open the **Aggregate Values** wizard.
mnuAlarm	Open the **Alarm Values** wizard.
mnuAnalysis	Open the **Tag Analysis** wizard.
mnuBaseDate	Open the **Set Base Date/Time** dialog box.
mnuConvert	Convert the function in the selected cell to values.
mnuConvertSheet	Convert the functions in the active sheet to values.
mnuEditFunction	Open the appropriate wizard for the selected function.
mnuHelp	Open the Help file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Method	Used to
mnuHistory	Open the **History Values** wizard.
mnuInSQL	Open the **Server Details** dialog box.
mnuLive	Open the **Live Values** wizard.
mnuOptions	Open the **Options** dialog box.
mnuQuery	Open the **Direct Query** dialog box.
mnuRefreshSelection	Refresh the selected function.
mnuRefreshSheet	Refresh the active worksheet.
mnuSnapSearch	Open the **Event Snapshot Tag Selection** dialog box.
mnuSnapShot	Open the **Event Snapshot Values** wizard.
mnuSumTagSearch	Open the **Summary Tag Selection** dialog box.
mnuSumTagValues	Open the **Summary Values** wizard.
mnuTagDesc	Open the **Tag Details** wizard.
mnuTagSearch	Open the **Tag Selection** dialog box.

AVEVA Historian Client Workbook Functions

AVEVA Historian Client Workbook Functions

For more information see AVEVA Historian Client Workbook Function Reference.

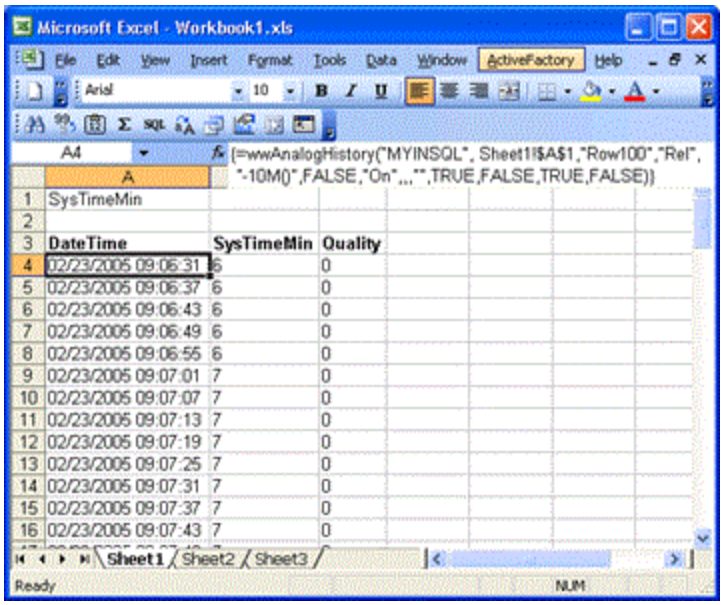
AVEVA Historian Client Workbook Automation Example

The following example illustrates how to automate the AVEVA Historian Client Workbook within Visual Basic for Applications (VBA). In this example, a button is added to the workbook that can be used to convert all of the functions in the sheet to values. This example uses the ActiveFactoryWorkbook.mnuConvertSheet method.

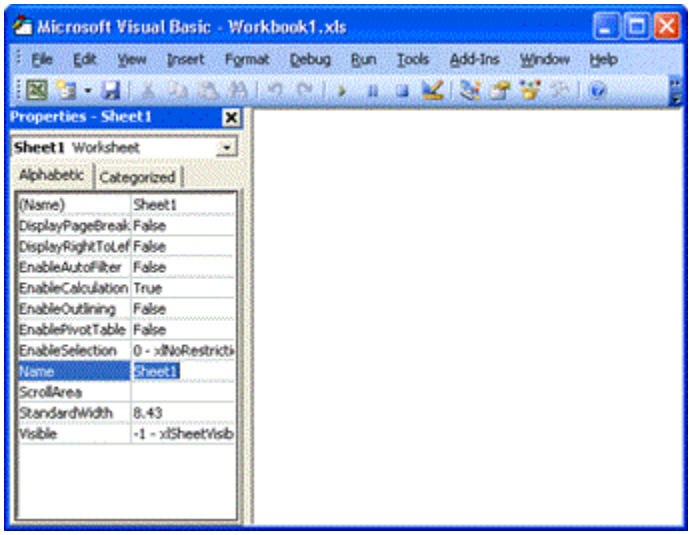
To automate the AVEVA Historian Client Workbook:

1. Start Excel and create an AVEVA Historian Client Workbook spreadsheet.

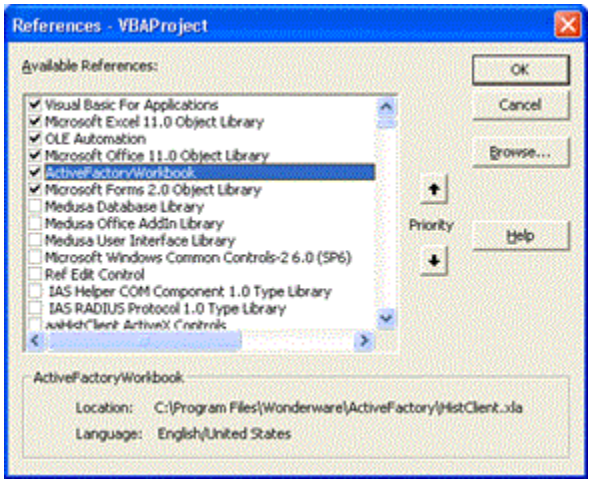
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. On the **Tools** menu, point to **Macro** and then click **Visual Basic Editor**. The **Microsoft Visual Basic** editor appears.

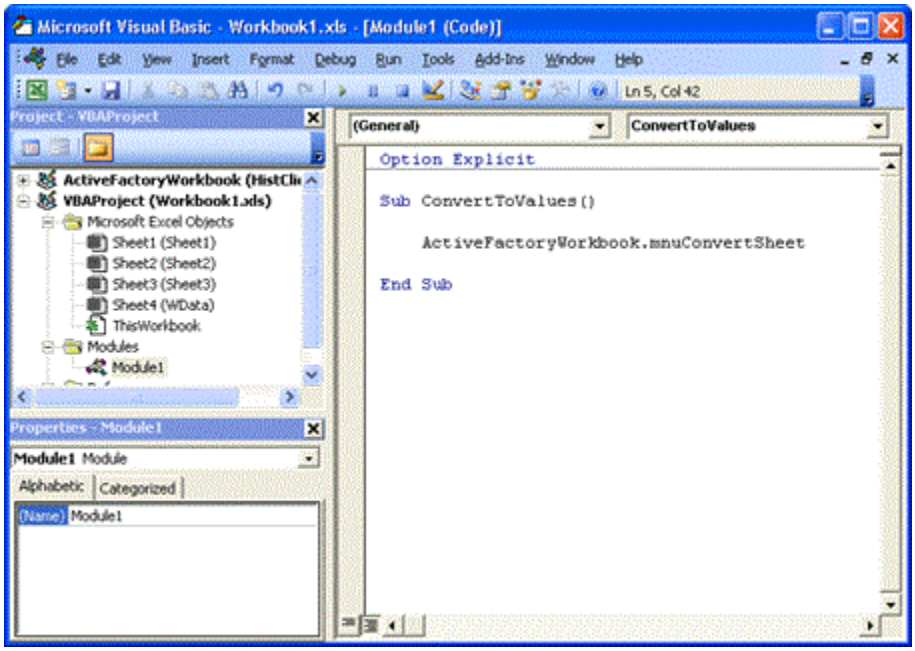


3. On the **Tools** menu, click **References**. The **References - VBAProject** dialog box appears.

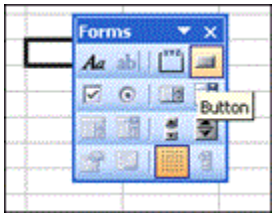


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

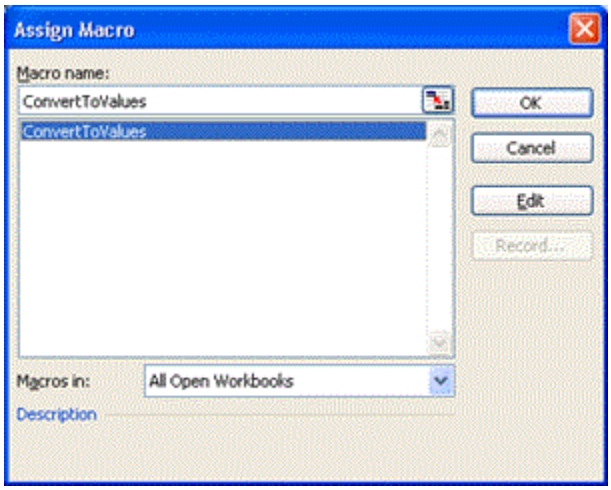
4. Select the **ActiveFactoryWorkbook** check box.
5. Click **OK**.
6. On the **Insert** menu, click **Module** to add a new module to the project.
7. Add a subroutine that executes the mnuConvertSheet method.



8. Switch back to Excel.
9. On the **View** menu, point to **Toolbars** and then click **Forms** to open the **Forms** toolbar.



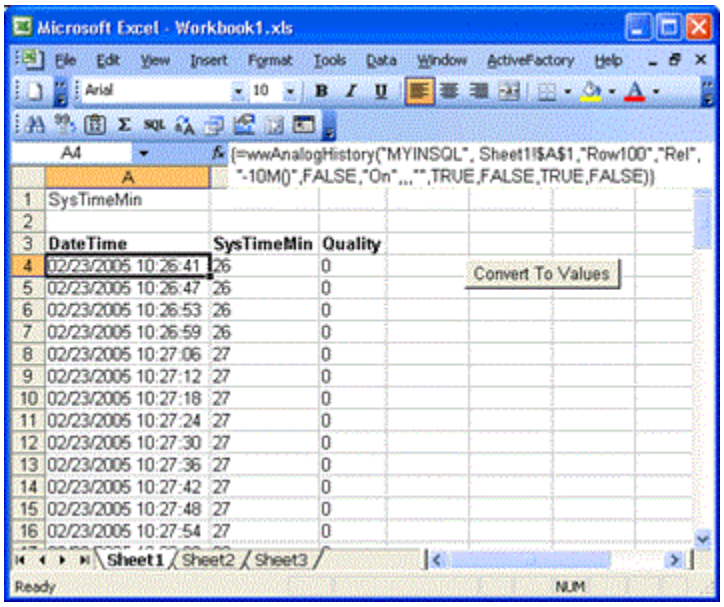
10. Insert a button into the worksheet. The **Assign Macro** dialog box appears.



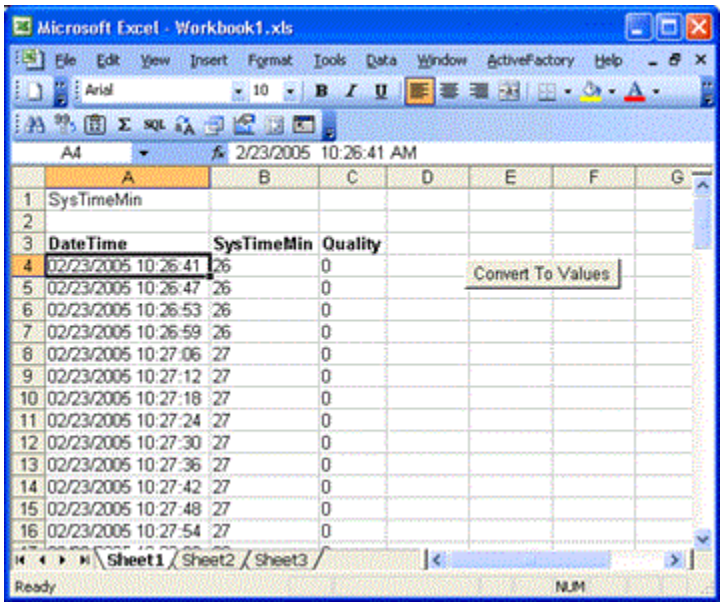
11. In the **Macro name** list, select ConvertToValues, which is the subroutine that you created in Step 7.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

12. Click **OK**.
13. Change the display name for the button and adjust the size, appropriately.



14. Click the **Convert To Values** button to execute the command.
15. All of the functions in the sheet are converted to values.



AVEVA Historian Client Report Object

To automate the generation of reports from the AVEVA Historian Client Report, use the AVEVA Historian Client Report object within the scripting environment.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Report Object Properties

The Report object properties are:

- ReportDate
- ReportTime

ReportDate

ReportDate

The ReportDate property is a read-write property that gets or sets the date that the report was run.

Syntax

```
ActiveFactoryReport.ReportDate = message;
Result = ActiveFactoryReport.ReportDate;
```

Remarks

The value of this property is used for any #Date wildcards used within the report. For more information on the #Date wildcard, see #date Wildcard.

The default value is the current date of when Microsoft Word was launched.

ReportTime

ReportTime

The ReportTime property is a read-write property that gets or sets the time that the report was run.

Syntax

```
ActiveFactoryReport.ReportTime = message;
Result = ActiveFactoryReport.ReportTime;
```

Remarks

The value of this property is used for any #ReportTime wildcards used within the report. For more information on the #ReportTime wildcard, see #ReportTime Wildcard.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The default value is the current date of when Microsoft Word was launched.

Report Object Methods

The Report object methods are:

- AutoExec
- AutoExit
- RunReport

AutoExec

AutoExec

The AutoExec method initializes values.

Syntax

```
ActiveFactoryReport.AutoExec()
```

AutoExit

AutoExit

The AutoExit method removes the AVEVA Historian Client toolbar and resets the main menu for Word.

Syntax

```
ActiveFactoryReport.AutoExit()
```

RunReport

RunReport

The RunReport method processes the Word report.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
ActiveFactoryReport.RunReport()
```

Return Value

Returns False if the report generation was successful; otherwise returns True.

Remarks

Any message dialog boxes are suppressed.

aaHistClientGlobalFunctions Object

This object provides methods for accessing the information about the AVEVA Historian Client software installation.

Using aaHistClientGlobalFunctions Object in an Application

You can use the aaHistClientGlobalFunctions object's methods in runtime scripts in your application to get installation information for the AVEVA Historian Client software.

The ProgID for the GlobalFunctions object is:

ArchestrA.HistClient.Util.aaHistClientGlobalFunctions

aaHistClientGlobalFunctions Methods

The aaHistClientGlobalFunctions methods are:

- GetDictionaryPath
- GetInstallPath
- GetAFVersion
- GetWorkstationName
- MDACOk

GetDictionaryPath

GetDictionaryPath

The GetDictionaryPath method returns the path to the dictionary file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
[Result=] aaHistClientGlobalFunctions.GetDictionaryPath();
```

Return Value

Returns an empty string.

Remarks

This method is provided for backward compatibility only.

GetInstallPath

GetInstallPath

The GetInstallPath method returns the path where the AVEVA Historian Client software is installed.

Syntax

```
[Result=] aaHistClientGlobalFunctions.GetInstallPath();
```

Return Value

Returns a fully-qualified path to the installation folder as a string.

GetAFVersion

GetAFVersion

The GetAFVersion method returns the version number of the Historian Client software.

Syntax

```
[Result=] aaHistClientGlobalFunctions.GetAFVersion();
```

Return Value

Returns a string value containing the version of the Historian Client software.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GetWorkstationName

GetWorkstationName

The GetWorkstationName method returns the name of the computer on which the Historian Client software is running.

Syntax

```
[Result=] aaHistClientGlobalFunctions.GetWorkstationName();
```

Return Value

Returns a string value containing the name of the computer on which the Historian Client software is running.

MDACOk

MDACOk

The MDACOk method returns whether the Microsoft Data Access Components (MDAC) are installed.

Syntax

```
[Result=] aaHistClientGlobalFunctions.MDACOk();
```

Return Value

Returns True if the components are installed; otherwise, returns False.

Remarks

Since MDAC is a prerequisite for the AVEVA Historian Client software, this method always returns True.

Common Properties, Methods, Events, Enums, and Data Types

This section describes the generic properties, methods, and events that are common to one or more controls. Also, descriptions for common data types and enumerations are provided.

Some of the common properties, methods, and events are ambient. Ambient properties, methods, and events are defined by the control container to assist the control in adapting to the particular environment in which it is used.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Notes:
Not all of the common properties, methods, events, enums, and data types are used by all of the controls.

This section includes information about all supported properties and methods on the documented controls . Any properties or methods not described in this section are not supported. We do not test, nor ensure performance from version to version, nor provide customer assistance for non-supported properties and methods.

Common Properties

All of the following properties are ambient properties.

- BackColor
- BackStyle
- BorderStyle
- CausesValidation
- Container
- ContextMenuEnabled
- DataBindings
- DragIcon
- DragMode
- Enabled
- Font
- ForeColor
- Height
- HelpContextID
- Index
- Left
- Name
- Object
- Parent
- TabIndex
- TabStop
- Tag
- ToolTipText
- Top

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Transparent
- Visible
- WhatsThisHelpID
- Width

BackColor

The BackColor property is a read-write property that specifies the background color for the control.

Syntax

```
<objectname>.BackColor = integer;
Result = <objectname>.BackColor;
```

Remarks

The default value of 1 indicates to use the window color.

BackStyle

The BackStyle property is a read-write property that specifies whether the background for a label or shape is opaque or transparent.

Syntax

```
<objectname>.BackStyle = integer;
Result = <objectname>.BackStyle;
```

Remarks

0 = Transparent; 1 = Opaque.

Setting this property to 0 has the same effect as setting the Transparent property to True.

BorderStyle

The BorderStyle property is a read-write property that specifies whether the control has a border line around it or not.

Syntax

```
<objectname>.BorderStyle = integer;
Result = <objectname>.BorderStyle;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

0 = No border; 1 = Single line border.

CausesValidation

The CausesValidation property is a read-write property that specifies whether validation occurs on the control.

Syntax

```
<objectname>.CausesValidation = discrete;
Result = <objectname>.CausesValidation;
```

Remarks

This property is not available in the InTouch HMI software.

Container

The Container property is a read-only property that returns the container of the control.

Syntax

```
<objectname>.Container = object;
Result = <objectname>.Container;
```

Remarks

This property is not available in the InTouch HMI software.

ContextMenuEnabled

The ContextMenuEnabled property is a read-write property that specifies whether the shortcut menu appears when a user right-clicks on the control.

Syntax

```
<objectname>.ContextMenuEnabled = discrete;
Result = <objectname>.ContextMenuEnabled;
```

Remarks

If this property is set to False, the Windows context menu still appears when a user right-clicks on an editable field. The Windows context menu contains editing commands such as Cut, Copy, Paste, and so on.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DataBindings

The DataBindings property is a read-only property that gets the bindable properties that are available to the application developer.

Syntax

```
<objectname>.DataBindings = DataBindings;
Result = <objectname>.DataBindings;
```

DragIcon

The DragIcon property is a read-write property that gets or sets the icon to be displayed for the mouse pointer during a drag-and-drop operation.

Syntax

```
<objectname>.DragIcon = Picture;
Result = <objectname>.DragIcon;
```

Remarks

This property is not available in the InTouch HMI software.

DragMode

The DragMode property is a read-write property that controls whether automatic or manual dragging is used.

Syntax

```
<objectname>.DragMode = integer;
Result = <objectname>.DragMode;
```

Remarks

This property is not available in the InTouch HMI software.

Enabled

The Enabled property is a read-write property that determines whether the control can be acted upon by the runtime user.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
<objectname>.Enabled = discrete;
Result = <objectname>.Enabled;
```

Font

For C# and .NET applications, a Font parameter or result can reference a Font class. For more information, see the documentation on the Font class in the *.NET Framework Class Library*.

ForeColor

The Fore property is a read-write property that gets or sets the color to be used for the foreground color in the control.

Syntax

```
<objectname>.ForeColor = Long;
Result = <objectname>.ForeColor;
```

Remarks

The default value is 1 (the window color).

Height

The Height property is a read-write property that gets or sets the height of the control.

Syntax

```
<objectname>.Height = Single;
Result = <objectname>.Height;
```

Remarks

This property is not available in the InTouch HMI software.

HelpContextID

The HelpContextID property is a read-write property that gets or sets the Help context ID for the object.

Syntax

```
<objectname>.HelpContextID = Long;
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Result = <objectname>.HelpContextID;
```

Remarks

This property is not available in the InTouch HMI software.

Index

The Index property is a read-only property that returns the number identifier for a control in an array.

Syntax

```
<objectname>.Index = integer;
Result = <objectname>.Index;
```

Remarks

The starting value for the identifier is typically 1. This property is not available in the InTouch HMI software.

Language

The Language property is a read-only property that enables reading content from the historian in the specified language, if translated text is available in the specified language. If no translation is available in the specified language, the default text is used.

Syntax

```
<objectname>.Language = $Language;
Result = <objectname>.Language;
```

Left

The Left property is a read-write property that gets or sets the distance between the left edge of the container application and the internal left edge of the control.

Syntax

```
<objectname>.Left = Single;
Result = <objectname>.Left;
```

Remarks

This property is not available in the InTouch HMI software.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name

The Name property is a read-only property that gets the name used to identify an object.

Syntax

```
<objectname>.Name = message;
Result = <objectname>.Name;
```

Remarks

This property is not available in the InTouch HMI software.

Object

For C# and .NET applications, an object parameter or result can reference an Object class. For more information, see the documentation on the Object class in the *.NET Framework Class Library*.

Parent

The Parent property is a read-only property that gets the object on which this object is located.

Syntax

```
<objectname>.Parent = Object;
Result = <objectname>.Parent;
```

Remarks

This property is not available in the InTouch HMI software.

TabIndex

The TabIndex property is a read-write property that gets or sets the tab order of the object within its parent form.

Syntax

```
<objectname>.TabIndex = integer;
Result = <objectname>.TabIndex;
```

Remarks

This property is only available during design time. This property is not available in the InTouch HMI software.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TabStop

The TabStop property is a read-write property that gets or sets whether the TAB key can be used to give focus to an object.

Syntax

```
<objectname>.TabStop = discrete;
Result = <objectname>.TabStop;
```

Remarks

This property is not available in the InTouch HMI software.

Tag

The Tag property is a read-write property that can be used to store extra data needed for the application.

Syntax

```
<objectname>.Tag = message;
Result = <objectname>.Tag;
```

Remarks

This property is not available in the InTouch HMI software.

ToolTipText

The ToolTipText property is a read-write property that gets or sets the text that appears when the mouse pointer hovers over the control at runtime.

Syntax

```
<objectname>.ToolTipText = message;
Result = <objectname>.ToolTipText;
```

Remarks

This property is only available in the InTouch HMI software for the aaHistClientTrend control.

Top

The Top property is a read-write property that gets or sets the distance between the top edge of the object

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

container and the internal top edge of an object.

Syntax

```
<objectname>.Top = Single;
Result = <objectname>.Top;
```

Transparent

The Transparent property is a read-write property that gets or sets the background of the object to be transparent.

Syntax

```
<objectname>.Transparent = discrete;
Result = <objectname>.Transparent;
```

Remarks

If set to True, the underlying form appears through the background of the object. The field (text box) label is not transparent. However, you can hide the field label to achieve total transparency.

The default value is False.

Visible

The Visible property is a read-write property that determines whether an object is visible or hidden.

Syntax

```
<objectname>.Visible = discrete;
Result = <objectname>.Visible;
```

Remarks

This property is not available in the InTouch HMI software. To hide the control, change the object coordinates so that the object appears out of the bounds of the window.

WhatsThisHelpID

The WhatsThisHelpID property is a read-write property that gets or sets the associated context-sensitive Help ID number for an object.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
<objectname>.WhatsThisHelpID = Long;
Result = <objectname>.WhatsThisHelpID;
```

Remarks

This property is not available in the InTouch HMI software.

Width

The Width property is a read-write property that gets or sets the width of an object.

Syntax

```
<objectname>.Width = Single;
Result = <objectname>.Width;
```

Remarks

This property is not available in the InTouch HMI software.

Common Methods

All of the following methods are ambient methods.

- Drag
- Move
- SetFocus
- SetToolbarButtonEnabled
- ShowWhatsThis
- ZOrder

Drag

The Drag method begins, ends, or cancels the drag operation for any object except for Menu, Line, Time, and Shape.

Syntax

```
[Result=] <objectname>.Drag();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Value

Returns True if successful; otherwise returns False.

Move

The Move method moves an object.

Syntax

```
[Result=] <objectname>.Move(single Left, [Top], [Width], [Height]);
```

SetFocus

The SetFocus method sets the focus to the specified object.

Syntax

```
[Result=] <objectname>.SetFocus();
```

ShowWhatsThis

The ShowWhatsThis method displays a particular topic in a Help file.

Syntax

```
[Result=] <objectname>.ShowWhatsThis();
```

Remarks

The What' This? Popup control provided by Windows Help is used.

ZOrder

The ZOrder method locates a specified object at the back or from of the z-order within its graphical level.

Syntax

```
[Result=] <objectname>.ZOrder([Position]);
```

Common Events

All of the following events are ambient events.

- Click

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- DblClick
- DragDrop
- DragOver
- GotFocus
- KeyDown
- KeyPress
- KeyUp
- LostFocus
- MouseDown
- MouseMove
- MouseUp
- Validate

Click

The Click event is triggered when the run-time user clicks on the object at runtime with the mouse.

Syntax

```
<objectname>.Click();
```

DblClick

The DblClick event is triggered when the user double-clicks on the object at runtime with the mouse.

Syntax

```
<objectname>.DblClick();
```

DragDrop

The DragDrop event is triggered when a drag-and-drop operation is completed.

Syntax

```
<objectname>.DragDrop(Control source, single X, single Y);
```

DragOver

The DragOver event is triggered when a drag-and-drop operation is in progress.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
<objectname>.DragOver(Control source, single X, single Y, integer state);
```

GotFocus

The GotFocus event is triggered when an object receives focus.

Syntax

```
<objectname>.GotFocus();
```

KeyDown

The KeyDown event is triggered when a user presses a key on the keyboard while the object has focus.

Syntax

```
<objectname>.KeyDown(integer KeyCode, integer Shift);
```

KeyPress

The KeyPress event is triggered when the runtime user presses and releases an ANSI key.

Syntax

```
<objectname>.KeyPress(integer KeyAscii);
```

KeyUp

The KeyUp event is triggered when a user releases a key on the keyboard while the object has focus.

Syntax

```
<objectname>.KeyUp(integer KeyCode, integer Shift);
```

LostFocus

The LostFocus event is triggered when an object loses focus.

Syntax

```
<objectname>.LostFocus();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

MouseDown

The MouseDown event is triggered when the user presses the mouse key down while an object has focus.

Syntax

```
<objectname>.MouseDown(integer button, integer shift, single X, single Y);
```

MouseMove

The MouseMove event is triggered when the user moves the mouse.

Syntax

```
<objectname>.MouseMove(integer button, integer shift, single X, single Y);
```

MouseUp

The MouseUp event is triggered when the user presses the mouse key up while an object has focus.

Syntax

```
<objectname>.MouseUp(integer button, integer shift, single X, single Y);
```

Validate

The Validate event is triggered when a control loses focus to a control that causes validation.

Syntax

```
<objectname>.Validate(discrete Cancel);
```

Common Enumerations

Common enumerations are:

- aaRetrievalSource Enumeration
- aaTagType Enumeration
- aaTimeRangeEnumeration Enumeration

aaRetrievalSource Enumeration

Specifies the source of process data to retrieve.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value	Enumeration	Description
0	ExtensionTablesOnly	Retrieve data only from a history extension tables (for example, History).
1	ManualHistoryOnly	Retrieve data only from a manual history tables (for example, ManualAnalogHistory).
2	Both	Retrieve data from both the extension table and the manual history table and combine the data from both.

aaTagType Enumeration

Specifies the set of tag types allowed for tags.

Value	Enumeration	Description
0	UnknownTag	Not a valid type. This value indicates a tag that cannot be found on the server.
1	Analog	Analog tag type.
2	Discrete	Discrete tag type.
3	String	String tag type.
4	Complex	Complex tag type.
5	Event	Event tag type.
7	Summary	Summary tag type.

aaTimeRangeEnumeration Enumeration

Specifies which time range is selected.

Enum	Value	Description
0	Custom	The time duration is custom.
1	LastMinute	The last minute.
2	Last5Minutes	The last five minutes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Enum	Value	Description
3	Last10Minutes	The last ten minutes.
4	Last15Minutes	The last fifteen minutes.
5	Last30Minutes	The last 30 minutes.
6	LastHour	The last hour.
7	Last2Hours	The last two hours.
8	Last4Hours	The last four hours.
9	Last8Hours	The last eight hours.
10	Last12Hours	The last twelve hours.
11	Last24Hours	The last twenty-four hours.
12	Last2Days	The last two days.
13	LastWeek	The last week.
14	Last2Weeks	The last two weeks.
15	LastMonth	The last month.
16	Last3Months	The last three months.
17	OneMinute	One minute.
18	FiveMinutes	Five minutes.
19	TenMinutes	Ten minutes.
20	FifteenMinutes	Fifteen minutes.
21	ThirtyMinutes	30 minutes.
22	OneHour	One hour.
23	TwoHours	Two hours.
24	FourHours	Four hours.
25	EightHours	Eight hours.
26	TwelveHours	Twelve hours.
27	TwentyFourHours	Twenty-four hours.
28	TwoDays	Two days.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Enum	Value	Description
29	OneWeek	One week.
30	TwoWeeks	Two weeks.
31	OneMonth	One month.
32	ThreeMonths	Three months.
33	Yesterday	0:00:00 of the previous day to 0:00:00 of the current day.
34	CurrentDay	0:00:00 of the current day to the current time.
35	PreviousHour	The start of the previous hour to the start of the current hour.
36	CurrentHour	The start of the current hour to the current time.
37	FullPeriod	Only applicable in relative mode. Sets the offsets and duration so that the trend shows the same time period as it did before it was switched into relative mode.

Common Data Types

Common data types are:

- DateTime
- Color
- DataSet
- Font
- Object

DateTime

For the InTouch HMI software, use a message value in a valid date/time format.

For C# and .NET applications, a DateTime parameter or result can reference a DateTime structure. For more information, see the documentation on the DateTime structure in the *.NET Framework Class Library*.

Color

To specify a color for a control, you must specify the color as an integer value. The color is an ABGR color, where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A = Transparency

B = Blue

G = Green

R = Red

A BGR color value is made up of 24 bits, with the upper 8 bits always being 0. For example, 0xFF0000 is 0B00 in the BGR convention, which equates to Blue.

An ABGR color value is made up of 32 bits, with the upper 8 bits being 0 by default. It can be set to any opacity:

- 00 (in HEX) in the upper 8 bits means no transparency or full opacity.
- FF (in HEX) in the upper 8 bits means full transparency or no opacity.
- B0 (in HEX) in the upper 8 bits means more transparent than opaque.
- 0A (in HEX) in the upper 8 bits means more opaque than transparent.

In decimal notation, the value for full transparency is 255.

For a color of Blue, the ABGR values are as follows:

A = 0 (full opacity)

B = 255

G = 0

R = 0

The hexidecimal value for this color is 0x00FF0000. The decimal value is 16711680.

0xA0FF0000 is 62.7% transparent blue. The decimal value is 2701066240.

0xFFFF0000 is fully transparent blue, so you do not see it at all. The decimal value is 4294901760.

0xA0000000 is a transparent shade of black (62.7% transparent). The decimal value is 2684354560.

DataSet

For C# and .NET applications, a DataSet parameter or result can reference a DataSet object. For more information, see the documentation on the ADO.NET DataSet object in the *.NET Framework Developer's Guide*.

Font

For C# and .NET applications, a Font parameter or result can reference a Font class. For more information, see the documentation on the Font class in the *.NET Framework Class Library*.

Object

For C# and .NET applications, an object parameter or result can reference an Object class. For more information, see the documentation on the Object class in the *.NET Framework Class Library*.

Data Retrieval Modes and Options

You can use a variety of retrieval modes and options to suit different reporting needs and applications.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Understanding Retrieval Modes

Different retrieval modes allow you to access the data stored in an AVEVA Historian in different ways. For example, if you retrieve data for a long time period, you might want to retrieve only a few hundred evenly spaced data points to minimize response time. For a shorter time period, you might want to retrieve all values that are stored on the server to get more accurate results.

An AVEVA Historian with a version earlier than 9.0 supports two retrieval modes:

- Cyclic Retrieval
- Delta Retrieval

An AVEVA Historian with a version of 9.0 or higher supports various additional modes:

- Full Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Slope Retrieval
- Counter Retrieval
- ValueState Retrieval

An AVEVA Historian with a version of 10.0 or higher supports the following additional mode:

- RoundTrip Retrieval

An AVEVA Historian version 11.6.14101 or higher supports the following additional mode:

- Predictive Retrieval

An AVEVA Historian with a version of 17.3.100 or higher supports the following additional mode:

- Bounding Value Retrieval

Cyclic Retrieval

Cyclic retrieval is the retrieval of stored data for the given time period based on a specified cyclic retrieval resolution, regardless of whether or not the value of the tag(s) has changed. It works with all types of tags. Cyclic retrieval produces a virtual rowset, which may or may not correspond to the actual data rows stored on the AVEVA Historian.

In cyclic retrieval, one row is returned for each "cycle boundary." You specify the number of cycles either directly or by means of a time resolution, that is, the spacing of cycle boundaries in time. If you specify a number of cycles, the AVEVA Historian returns that number of rows, evenly spaced in time over the requested period. The

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

cyclic resolution is calculated by dividing the requested time period by the number of cycle boundaries. If you specify a resolution, the number of cycles is calculated by dividing the time period by the resolution.

If no data value is actually stored at a cycle boundary, the last value before the boundary is returned.

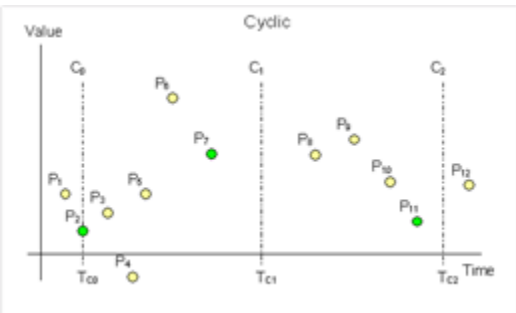
Beginning with AVEVA System Platform 2014 R2 SP1, Historian cyclic storage rules improve the handling of "slow rate change" data tags. Instead of delaying posts to the database if tag values do not arrive in a timely manner, new rules define a cyclic timeout when the database will be updated anyway. That timeout is typically one-half the period for the tag's cycle storage rate or the database server's maximum cyclic storage timeout, whichever is shorter. The time out is controlled by a the system parameter MaxCyclicStorageTimeout. For more information, see System Parameters in the *AVEVA Historian Administration Guide*.

The default retrieval mode is cyclic for retrieval from analog tables, including analog and state summary tables.

Cyclic retrieval is fast and therefore consumes little server resources. However, it may not correctly reflect the stored data because important process values (gaps, spikes, etc.) might fall between cycle boundaries. For an alternative, see Best Fit Retrieval.

Cyclic Retrieval - How It Works

The following illustration shows how values are returned for cyclic retrieval:



Data is retrieved in cyclic mode with a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for three cycle boundaries at T_{C0}, T_{C1}, and T_{C2}. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

- At T_{C0}: P_2, because it falls right on the cycle boundary
- At T_{C1}: P_7, because it is the last point before the cycle boundary
- At T_{C2}: P_{11}, for the same reason

Cyclic Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in cyclic retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule), for AVEVA Historian 9.0 and above
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Cyclic Retrieval - Query Example

To use the cyclic retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Cyclic'
```

For example, the following query returns data values for the analog tag 'ReactLevel'. If you do not specify a wwCycleCount or wwResolution, the query will return 100 rows (the default).

```
SELECT DateTime, Sec = DATEPART(ss, DateTime), TagName, Value
   FROM History
      WHERE TagName = 'ReactLevel'
         AND DateTime >= '2001-03-13 1:15:00pm'
         AND DateTime <= '2001-03-13 2:15:00pm'
         AND wwRetrievalMode = 'Cyclic'
```

The results are:

DateTime	Sec	TagName	Value
2001-03-13 13:15:00.000	0	ReactLevel	1775.0
2001-03-13 13:15:00.000	36	ReactLevel	1260.0
2001-03-13 13:16:00.000	12	ReactLevel	1650.0
2001-03-13 13:16:00.000	49	ReactLevel	1280.0
2001-03-13 13:17:00.000	25	ReactLevel	1525.0
2001-03-13 13:18:00.000	1	ReactLevel	585.0
2001-03-13 13:18:00.000	38	ReactLevel	1400.0
2001-03-13 13:19:00.000	14	ReactLevel	650.0
2001-03-13 13:19:00.000	50	ReactLevel	2025.0
2001-03-13 13:20:00.000	27	ReactLevel	765.0
2001-03-13	3	ReactLevel	2000.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

13:21:00.000			
2001-03-13 13:21:00.000	39	ReactLevel	830.0
2001-03-13 13:22:00.000	16	ReactLevel	1925.0
...			
(100 row(s) affected)			

Cyclic Retrieval - Initial Values

No special handling is done for initial values. The initial value will behave like a normal cycle boundary at the start time. For information on initial values, see Delta Retrieval - Initial Values.

Cyclic Retrieval - Handling NULL Values

No special handling is done for NULL values. They are returned just like any other value.

Delta Retrieval

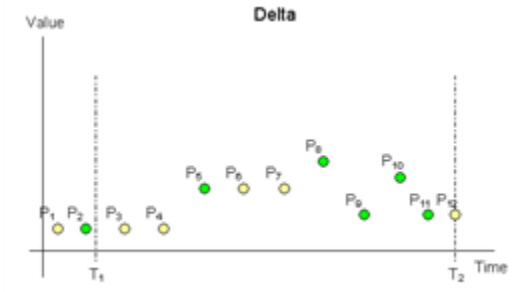
Delta retrieval, or retrieval based on exception, is the retrieval of only the changed values for a tag(s) for the given time interval. That is, duplicate values are not returned. It works with all types of tags.

Delta retrieval always produces a rowset comprised of only rows that are actually stored on the historian; that is, a delta query returns all of the physical rows in history for the specified tags, over the specified period, minus any duplicate values. If there is no actual data point at the start time, the last data point before the start time is returned.

Delta retrieval is the default mode for discrete and string tables and from the History table.

Delta Retrieval - How It Works

The following illustration shows how values are returned for delta retrieval:



Data is retrieved in delta mode with a start time of T_1 and an end time of T_2. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- P_2, because there is no actual data point at T_1
- P_5, P_8, P_9, P_{10}, and P_{11}, because they represent changed values during the time period

For delta retrieval for replicated summary tags on a tier-2 historian, if a point with doubtful quality is returned as the result of a value selection from an input summary point with a contained gap, the same point can be returned again with good quality if the same value is selected again from the next input summary point that has good quality.

Delta Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in delta retrieval mode. For more information, see the following sections:

- Time Deadband (wwTimeDeadband)
- Value Deadband (wwValueDeadband)
- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Delta Retrieval - Query Examples

To use the delta retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Delta'
```

For examples, see the following:

- Delta Retrieval - Query 1
- Delta Retrieval - Query 2
- Delta Retrieval - Query 3
- Delta Retrieval - Query 4

Delta Retrieval - Query 1

As an example of how delta mode works, consider the following query:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwRetrievalMode = 'Delta'
```

This query can be run against the following sample data:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:11	1.0	192
A001	2009-09-12 00:13	1.6	192
A001	2009-09-12 00:16	1.3	192
A001	2009-09-12 00:21	2.0	192
A001	2009-09-12 00:24	1.2	192
A001	2009-09-12 00:27	1.2	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:32	0.6	192
A001	2009-09-12 00:35	0.0	249
A001	2009-09-12 00:37	1.5	192
A001	2009-09-12 00:43	1.3	192

A graphical representation of the data is as follows:

delta_mode_new_example_query

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	1.3	192
A001	2009-09-12 00:21	2.0	192
A001	2009-09-12 00:24	1.2	192
A001	2009-09-12 00:28	NULL	249
A001	2009-09-12 00:32	0.6	192
A001	2009-09-12 00:35	NULL	249
A001	2009-09-12 00:37	1.5	192

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 2009-09-12 00:20 and 2009-09-12 00:40 (shown on the chart as dark vertical lines) are returned by the query.

Because there is no value that matches the start time, an initial value at 2009-09-12 00:20 is returned in the results based on the value of the preceding data point at 2009-09-12 00:16. Because there is no change in the value at 2009-09-12 00:27 from the value at 2009-09-12 00:24, the data point appears on the chart but does not

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

appear in the results. Similarly, the second 0.0 value at 2009-09-12 00:29 is also excluded from the results.

You can further control the number of rows returned by using the wwTimeDeadband, wwValueDeadband, and wwCycleCount extensions. The use of a cycle count returns the first number of rows within the time range of the query. For more information, see -old-Using wwResolution, wwCycleCount, and wwRetrievalMode in the Same Query.

Also, the use of a time deadband and/or value deadband with delta retrieval produces differing results. For more information, see Time Deadband (wwTimeDeadband) and Value Deadband (wwValueDeadband).

Delta Retrieval - Query 2

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeSec','SysTimeMin')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Delta'
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:01.000	SysTimeSec	1
2001-12-09 11:35:02.000	SysTimeSec	2
2001-12-09 11:35:03.000	SysTimeSec	3
2001-12-09 11:35:04.000	SysTimeSec	4
...		
2001-12-09 11:35:58.000	SysTimeSec	58
2001-12-09 11:35:59.000	SysTimeSec	59
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:00.000	SysTimeMin	36

Delta Retrieval - Query 3

```
SELECT * FROM OpenQuery(INSQL,'SELECT DateTime, Value, Quality, QualityDetail
   FROM AnalogHistory
      WHERE TagName = "SysTimeSec"
         AND wwRetrievalMode = "Delta"
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
         AND Value = 10
         AND DateTime >="2001-07-27 03:00:00.000"
         AND DateTime <="2001-07-27 03:05:00.000"
      ')
```

The results are:

DateTime	Value	Quality	QualityDetail
2001-07-27 03:00:10.000	10	0	192
2001-07-27 03:01:10.000	10	0	192
2001-07-27 03:02:10.000	10	0	192
2001-07-27 03:03:10.000	10	0	192
2001-07-27 03:04:10.000	10	0	192

Delta Retrieval - Query 4

For a delta query, if both a wwCycleCount and a Value comparison are specified, the query will return the first number of rows (if available) that meet the value indicated.

```
SELECT * FROM OpenQuery(INSQL,'SELECT DateTime, Value, Quality, QualityDetail
   FROM AnalogHistory
      WHERE TagName = "SysTimeSec"
         AND wwRetrievalMode = "Delta"
         AND Value = 20
         AND wwCycleCount = 10
         AND DateTime >="2001-07-27 03:00:00.000"
         AND DateTime <="2001-07-27 03:20:00.000"
   ')
```

The results are:

DateTime	Value	Quality	QualityDetail
2001-07-27 03:00:20.000	20	0	192
2001-07-27 03:01:20.000	20	0	192
2001-07-27 03:02:20.000	20	0	192
2001-07-27	20	0	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

03:03:20.000			
2001-07-27 03:04:20.000	20	0	192
2001-07-27 03:05:20.000	20	0	192
2001-07-27 03:06:20.000	20	0	192
2001-07-27 03:07:20.000	20	0	192
2001-07-27 03:08:20.000	20	0	192
2001-07-27 03:09:20.000	20	0	192

Delta Retrieval - Initial Values

Initial values are special values that can be returned from queries that lie exactly on the query start time, even if there is not a data point that specifically matches the specified start time. If there is not a value exactly on the query start time, the last point before the start time will be returned with its DateTime set to the query start time and its Quality set to 133. If no value exists at or prior to the query start time, a NULL value will be returned at start time with QualityDetail set to 65536, OPCQuality set to 0, and Quality set to 1.

Querying the start time in exclusive form with the > operator indicates that a value should not be returned for the query start time if one does not exist. Querying the start time in inclusive form with the >= operator indicates that an initial value should be returned.

For example, the following exclusive query statement does not return an initial value for 2009-01-01 02:00:00.

```
   DateTime > '2009-01-01 02:00:00'
```

However, the following inclusive query statement does return an initial value for 2009-01-01 02:00:00.

```
   DateTime >= '2009-01-01 02:00:00'
```

No special final value is returned.

Delta Retrieval - Handling NULL Values

The initial NULL value after a non-NULL is always returned. Multiple NULL values are suppressed. The first non-NULL after a NULL is always returned even if it is the same as the previous non-NULL value.

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwRetrievalMode = 'Delta'
```

This query can be run against the following sample data:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:24	0.0	249
A001	2009-09-12 00:27	0.0	249
A001	2009-09-12 00:28	0.5	192
A001	2009-09-12 00:31	0.0	249
A001	2009-09-12 00:33	0.0	24
A001	2009-09-12 00:35	0.0	24
A001	2009-09-12 00:36	0.5	192

The following is a graphical representation of the data:

delta_mode_new_example_NULL_handling

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	0.8	192
A001	2009-09-12 00:24	NULL	249
A001	2009-09-12 00:28	0.5	192
A001	2009-09-12 00:31	NULL	249
A001	2009-09-12 00:36	0.5	192

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query.

Because there is no value that matches the start time, an initial value at 00:20 is returned in the results based on the value of the preceding data point at 00:16. Because there is no change in the value at 00:27 from the value at 00:24, the data point appears on the chart but does not appear in the results. Similarly, the two 0.0 values at 00:33 and 00:35 are also excluded from the results. However, the non-NULL value at 00:36 is returned, even though it is the same as the value at 00:28, because it represents a delta from the preceding (NULL) value at 00:35.

Full Retrieval

In full retrieval mode, all stored data points are returned, regardless of whether a value or quality has changed since the last value. This mode allows the same value and quality pair (or NULL value) to be returned consecutively with their actual timestamps. It works with all types of tags.

By using full retrieval in conjunction with storage without filtering (that is, no delta or cyclic storage mode is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

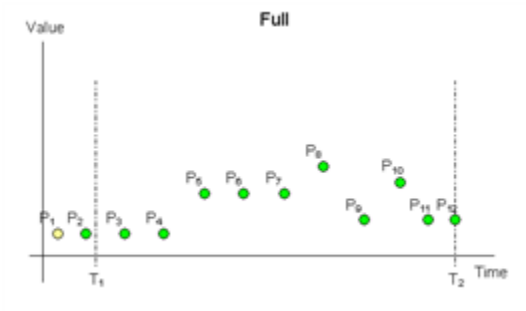
applied at the historian), you can retrieve all values that originated from the plant floor data source or from another application.

Full retrieval best represents the process measurements recorded by the AVEVA Historian. However, it creates a higher load for the server, the network and the client system because a very large number of records may be returned for longer time periods.

For full retrieval for replicated summary tags on a tier-2 historian, if a point with doubtful quality is returned as the result of a value selection from an input summary point with a contained gap, the same point can be returned again with good quality if the same value is selected again from the next input summary point that has good quality.

Full Retrieval - How It Works

The following illustration shows how values are returned for full retrieval:



Data is retrieved in full mode with a start time of T_1 and an end time of T_2. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

- P_2, because there is no actual data point at T_1
- P_3 through P_{12}, because they represent stored data points during the time period

Full Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in full retrieval mode. For more information, see the following sections:

- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Full Retrieval - Query Example

For example, the following query uses full retrieval mode:

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeSec','SysTimeMin')
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND DateTime >= '2001-12-09 11:35'
        AND DateTime <= '2001-12-09 11:36'
        AND wwRetrievalMode = 'Full'
```

Full Retrieval - Initial Values

Full retrieval mode handles initial values the same way as delta mode. For more information on initial values, see Delta Retrieval - Initial Values.

Interpolated Retrieval

Interpolated retrieval works like cyclic retrieval, except that interpolated values are returned if there is no actual data point stored at the cycle boundary.

This retrieval mode is useful if you want to retrieve cyclic data for slow-changing tags. For a trend, interpolated retrieval results in a smoother curve instead of a "stair-stepped" curve. This mode is also useful if you have a slow-changing tag and a fast-changing tag and want to retrieve data for both. Finally, some advanced applications require more evenly spaced values than would be returned if interpolation was not applied.

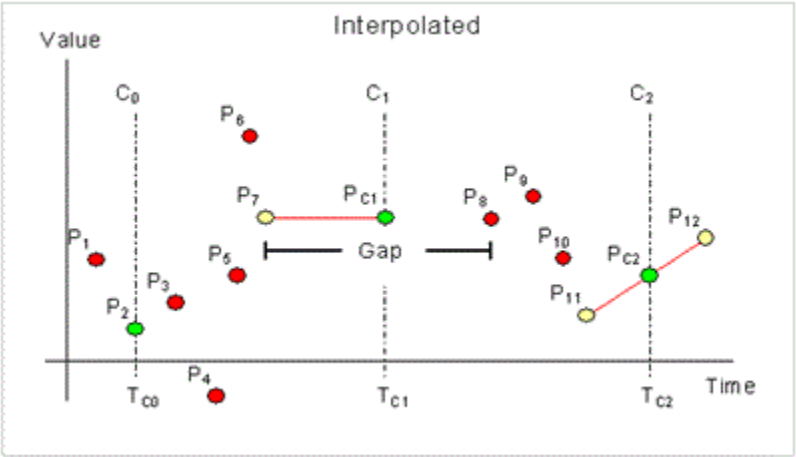
By default, interpolated retrieval uses the interpolation setting specified for the tag in the AVEVA Historian. This means that if a tag is set to use stair-step interpolation, interpolated retrieval gives the same results as cyclic retrieval.

Interpolation is only applied to analog tags. If you retrieve data for other types of tags, stair-step interpolation is used, and the results are the same as for cyclic retrieval.

Interpolated retrieval is a bit slower than cyclic retrieval. It shares the limitations of cyclic retrieval in that it may not accurately represent the stored process data.

Interpolated Retrieval - How It Works

The following illustration shows how the values for an analog tag that is configured for linear interpolation are returned when using interpolated retrieval.



Data is retrieved in interpolated mode with a start time of TC_0 and an end time of TC_2. The resolution has been set in such a way that the historian returns data for three cycle boundaries at TC_0, TC_1, and TC_2. P_1 to P_{12} represent actual data points stored on the historian. Of these points, eleven represent normal analog values, and one, P_7, represents a NULL value due to an I/O Server disconnect, which causes a gap in the data between P_7 and

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

P_8.

The green points (P_2, PC_1, PC_2) are returned. The yellow points (P_7, P_{11}, P_{12}) are used to interpolate the returned value for each cycle. The red points are considered, but not used in calculating the points to return.

Because P_2 is located exactly at the query start time, it is returned at that time without the need for any interpolation. At the following cycle boundary, point PC1 is returned, which is the NULL value represented by P_7 shifted forward to time TC_1. At the last cycle boundary, point PC_2 is returned, which has been interpolated using points P_{11} and P_{12}.

Interpolated Retrieval - Query Examples

To use the interpolated mode, set the following parameter in your query.

```
wwRetrievalMode = 'Interpolated'
```

For examples, see the following:

- Interpolated Retrieval - Query 1
- Interpolated Retrieval - Query 2
- Interpolated Retrieval - Query 3

Interpolated Retrieval - Query 1

Two analog tags and a discrete tag are retrieved from the History table, using linear interpolation. The start and end times are offset to show interpolation of the SysTimeMin tag. The data points at all cycle boundaries are interpolated for the two analog tags, while the values returned for the discrete tag are stair-stepped.

```
SELECT DateTime, TagName, Value, wwInterpolationType FROM History
   WHERE TagName IN ('SysTimeMin', 'ReactTemp', 'SysPulse')
      AND DateTime >= '2005-04-11 12:02:30'
      AND DateTime <= '2005-04-11 12:06:30'
      AND wwRetrievalMode = 'Interpolated'
      AND wwInterpolationType = 'Linear'
      AND wwResolution = 60000
```

The results are:

DateTime	TagName	Value	wwInterpolationType
2005-04-11 12:02:30.000	SysTimeMin	2.5	LINEAR
2005-04-11 12:02:30.000	ReactTemp	23.2	LINEAR
2005-04-11 12:02:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:03:30.000	SysTimeMin	3.5	LINEAR

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2005-04-11 12:03:30.000	ReactTemp	139.96753	LINEAR
2005-04-11 12:03:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:04:30.000	SysTimeMin	4.5	LINEAR
2005-04-11 12:04:30.000	ReactTemp	111.49636	LINEAR
2005-04-11 12:04:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:05:30.000	SysTimeMin	5.5	LINEAR
2005-04-11 12:05:30.000	ReactTemp	17.00238	LINEAR
2005-04-11 12:05:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:06:30.000	SysTimeMin	6.5	LINEAR
2005-04-11 12:06:30.000	ReactTemp	168.99531	LINEAR
2005-04-11 12:06:30.000	SysPulse	1.0	STAIRSTEP

Interpolated Retrieval - Query 2

If you omit the interpolation type in the query, the historian determines which interpolation type to use for an analog tag based on the value of the InterpolationType column in the AnalogTag table, in conjunction with the InterpolationTypeInteger and InterpolationTypeReal system parameters.

In the following query both analog tags are set to use the system default through the AnalogTag table, while the InterpolationTypeInteger and InterpolationTypeReal system parameters are set to 0 and 1, respectively. Because SysTimeMin is defined as a 2-byte integer and ReactTemp is defined as a real we see that only rows for ReactTemp are interpolated.

```
SELECT DateTime, TagName, Value, wwInterpolationType FROM History
   WHERE TagName IN ('SysTimeMin', 'ReactTemp', 'SysPulse')
      AND DateTime >= '2005-04-11 12:02:30'
      AND DateTime <= '2005-04-11 12:06:30'
      AND wwRetrievalMode = 'Interpolated'
      AND wwResolution = 60000
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The results are:

DateTime	TagName	Value	wwInterpolationType
2005-04-11 12:02:30.000	SysTimeMin	2.0	STAIRSTEP
2005-04-11 12:02:30.000	ReactTemp	23.2	LINEAR
2005-04-11 12:02:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:03:30.000	SysTimeMin	3.0	STAIRSTEP
2005-04-11 12:03:30.000	ReactTemp	139.96753	LINEAR
2005-04-11 12:03:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:04:30.000	SysTimeMin	4.0	STAIRSTEP
2005-04-11 12:04:30.000	ReactTemp	111.49636	LINEAR
2005-04-11 12:04:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:05:30.000	SysTimeMin	5.0	STAIRSTEP
2005-04-11 12:05:30.000	ReactTemp	17.00238	LINEAR
2005-04-11 12:05:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:06:30.000	SysTimeMin	6.0	STAIRSTEP
2005-04-11 12:06:30.000	ReactTemp	168.99531	LINEAR
2005-04-11 12:06:30.000	SysPulse	1.0	STAIRSTEP

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Interpolated Retrieval - Query 3

```
SELECT TagName, DateTime, Value, QualityDetail, wwInterpolationType
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwRetrievalMode = 'Interpolated'
         AND wwResolution = '10000'
```

This query can be run against the following sample data:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:09	0.2	192
A001	2009-09-12 00:15	1.3	192
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:22	0.6	249
A001	2009-09-12 00:26	0.9	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:33	1.1	192
A001	2009-09-12 00:35	1.6	192
A001	2009-09-12 00:38	0.5	192
A001	2009-09-12 00:42	0.8	192

The following is a graphical representation of the data:

interpolated_mode_new_second_example

The results are:

Tagname	DateTime	Value	QualityDetail	wwInterpolationType
A001	2009-09-12 00:20	0.8	192	STAIRSTEP
A001	2009-09-12 00:30	NULL	249	STAIRSTEP
A001	2009-09-12 00:40	0.5	192	LINEAR

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Because there is no value that matches the start time, an initial value at 00:20 is returned in the results based on the preceding data point at 00:17 because the following data point at 00:22 is NULL. Because a NULL value precedes the 00:30 cycle boundary at 00:29, the NULL is returned at the cycle boundary. The value at 00:40 is an interpolation of the data points at 00:38 and 00:42.

Interpolated Retrieval - Initial and Final Values

A value is returned at the start time and end time of the query using interpolation of the surrounding points.

Interpolated Retrieval - Handling NULL Values

When a NULL value precedes a cycle boundary, that NULL will be returned at the cycle boundary.

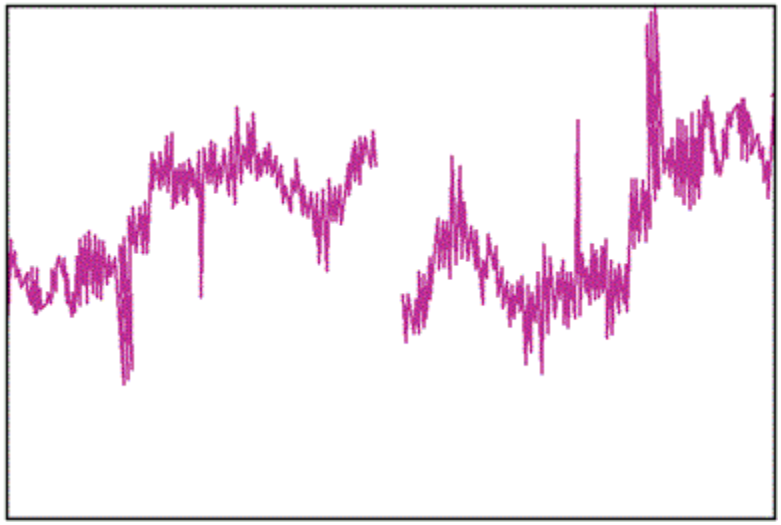
If a valid value precedes a cycle boundary, but is followed by a NULL value after the cycle boundary, no interpolation will be used and wwInterpolationType will be set to STAIRSTEP for that value.

Best Fit Retrieval

For the " best fit" retrieval mode, the total time for the query is divided into even sub-periods, and then up to five values are returned for each sub-period:

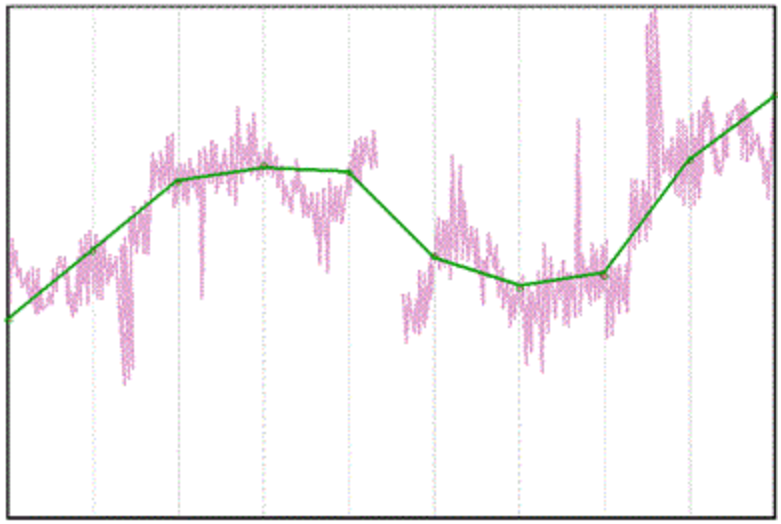
- First value in the period
- Last value in the period
- Minimum value in the period, with its actual time
- Maximum value in the period, with its actual time
- The first "exception" in the period (non-Good quality)

"Best fit" retrieval allows for a compromise between delta retrieval and cyclic retrieval. For example, delta retrieval can accurately represent a process over a long period of time, as shown in the following trend. However, to achieve this representation, a large number of data values must be returned.

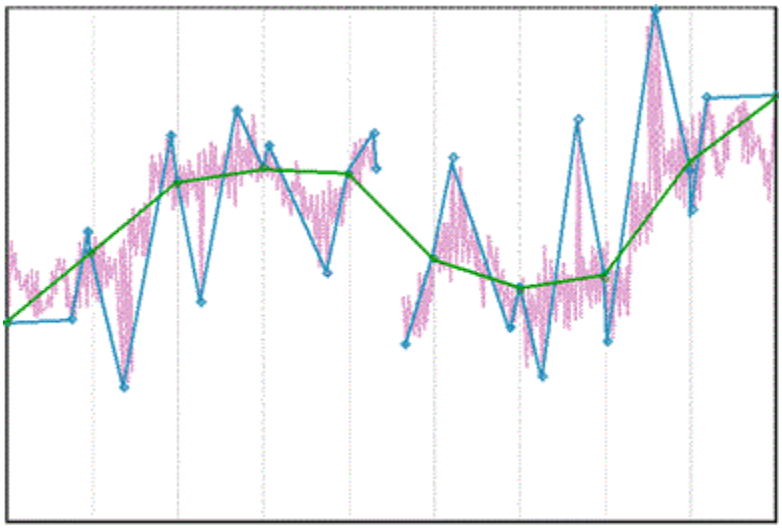


If cyclic retrieval is used to retrieve the data, the retrieval is much more efficient, because fewer values are returned. However, the representation is not as accurate, as the following trend shows.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



"Best fit" retrieval allows for faster retrieval, as typically achieved by using cyclic retrieval, plus the better representation typically achieved by using delta retrieval. This is shown in the following trend.



For example, for one week of five-second data, 120,960 values would be returned for delta retrieval, versus around 300 values for best-fit retrieval.

Best-fit retrieval uses retrieval cycles, but it is not a true cyclic mode. Apart from the initial value, it only returns actual delta points. For example, if one point is both the first value and the minimum value in a cycle, it is returned only one time. In a cycle where a tag has no points, nothing is returned.

As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, the number of values returned is likely to be more than one per cycle.

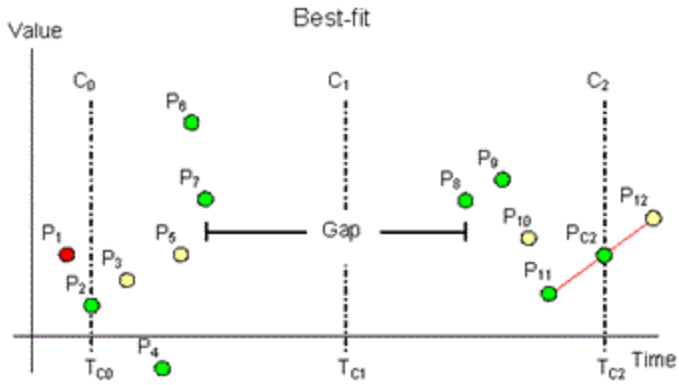
All points are returned in chronological order. If multiple points are to be returned for a particular timestamp, then those points are returned in the order in which the corresponding tags were specified in the query.

The best-fit algorithm is only applied to analog and analog summary tags. For all other tags, delta results are returned.

Best Fit Retrieval - How It Works

The following illustration shows how the best-fit algorithm selects points for an analog tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Data is retrieved in best-fit mode with a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}. P_1 to P_{12} represent actual data points stored on the historian. Of these points, eleven represent normal analog values, and one, P_7, represents a NULL value due to an I/O Server disconnect, which causes a gap in the data between P_7 and P_8.

Because P_2 is located exactly at the start time, no initial value needs to be interpolated at the start time. Therefore, point P_1 is not considered at all. All other points are considered, but only the points indicated by green markers on the graph are returned.

From the first cycle, four points are returned:

- P_2 as the initial value of the query, as well as the first value in the cycle
- P_4 as the minimum value in the cycle
- P_6 as both the maximum value and the last value in the cycle
- P_7 as the first (and only) occurring exception in the cycle

From the second cycle, three points are returned:

- P_8 as the first value in the cycle
- P_9 as the maximum value in the cycle
- P_{11} as both the minimum value and the last value in the cycle
- As no exception occurs in the second cycle, none is returned.

Because the tag does not have a point exactly at the query end time, where an incomplete third cycle starts, the end value P_{C2} is interpolated between P_{11} and P_{12}, assuming that linear interpolation is used.

Best Fit Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in best-fit retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Interpolation Type (wwInterpolationType)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Best Fit Retrieval - Query Example

To use the best fit retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'BestFit'
```

For example, an analog tag is retrieved over a five-minute period using the best-fit retrieval mode. The wwResolution parameter is set to 60000, thus specifying five 1-minute cycles. Within each cycle, the retrieval sub-system returns the first, minimum, maximum, and last data points. There are no exception (NULL) points in the time period. Notice how the points at the query start time and at the query end time are interpolated, while all other points are actual delta points.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 1), Value) AS Value, wwInterpolationType AS
IT FROM History
      WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 12:15:00'
      AND DateTime <= '2005-04-11 12:20:00'
      AND wwRetrievalMode = 'BestFit'
      AND wwResolution = 60000
```

The results are:

	DateTime	TagName	Value	IT
(initial, first, min)	2005-04-11 12:15:00.000	ReactTemp	40.7	LINEAR
(max in interval 1)	2005-04-11 12:15:38.793	ReactTemp	196.0	STAIRSTEP
(last in interval 1)	2005-04-11 12:15:58.810	ReactTemp	159.2	STAIRSTEP
(first, max in interval 2)	2005-04-11 12:16:00.013	ReactTemp	156.9	STAIRSTEP
(last, min in interval 2)	2005-04-11 12:16:58.857	ReactTemp	16.3	STAIRSTEP
(first, min in interval 3)	2005-04-11 12:17:00.060	ReactTemp	14.0	STAIRSTEP
(last, max in interval 3)	2005-04-11 12:17:58.793	ReactTemp	151.0	STAIRSTEP
(first in interval 4)	2005-04-11 12:18:00.107	ReactTemp	156.0	STAIRSTEP

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

(max in interval 4)	2005-04-11 12:18:10.057	ReactTemp	196.0	STAIRSTEP
(last, min in interval 4)	2005-04-11 12:18:58.837	ReactTemp	106.3	STAIRSTEP
(first, max in interval 5)	2005-04-11 12:19:00.040	ReactTemp	104.0	STAIRSTEP
(min in interval 5)	2005-04-11 12:19:31.320	ReactTemp	14.0	STAIRSTEP
(last in interval 5)	2005-04-11 12:19:58.773	ReactTemp	26.0	STAIRSTEP
(end bounding value)	2005-04-11 12:20:00.000	ReactTemp	30.7	LINEAR

Best Fit Retrieval - Initial and Final Values

A point will be returned at the query start time and the query end time for each tag queried, if a point exists for that tag at or after the end time of the query. The values of the initial and final points will be determined by interpolating the points preceding and following the query start or query end time.

Standard interpolation rules will be used to return the initial and final values. For more information, see Interpolated Retrieval.

Best Fit Retrieval - Handling NULL Values

When any of the four good points are returned from a cycle that contains gaps or from an incomplete cycle with the query end time located inside of the calculation cycle the quality detail of each of the non-null points returned is modified to alert the user to this fact. This is done by performing a logical OR operation of the value 4096, which means partial cycle, onto the existing quality detail. (This is the delta point equivalent to the use of PercentGood for cyclic.)

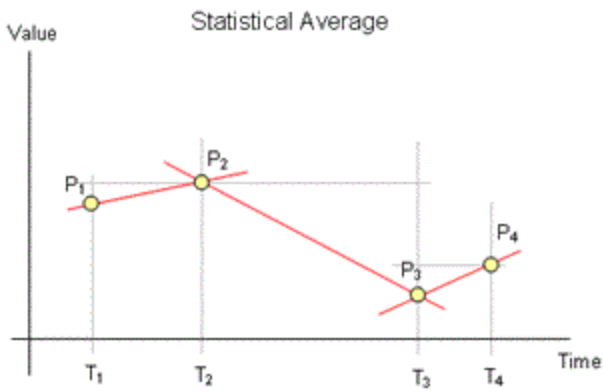
Average Retrieval

For the time-weighted average (in short: "average") retrieval mode, a time-weighted average algorithm is used to calculate the value to be returned for each retrieval cycle.

For a statistical average, the actual data values are used to calculate the average. The average is the sum of the data values divided by the number of data values. For the following data values, the statistical average is computed as:

$(P_1 + P_2 + P_3 + P_4) / 4) = \text{Average}$

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

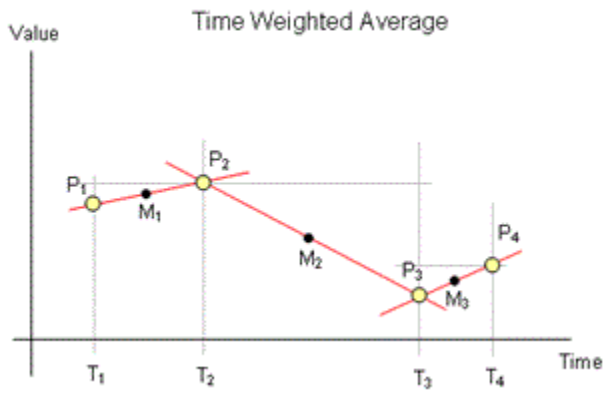


For a time-weighted average, values are multiplied by the time difference between the points to determine the time-weighted value. Therefore, the longer a tag had a particular value, the more weight that value holds in the overall average. The overall average is determined by adding all of the time-weighted values and then dividing that number by the total amount of time.

Which values are weighted depends on the interpolation setting of the tag. For a tag that uses linear interpolation, the midpoints between values are weighted. For a tag that uses stair-step interpolation, the earlier of two values is weighted.

For the following data values of a tag that uses linear interpolation, the time-weighted average is computed as:

$(((P_1 + P_2) / 2) \times (T_2 - T_1)) + (((P_2 + P_3) / 2) \times (T_3 - T_2)) + (((P_3 + P_4) / 2) \times (T_4 - T_3)) / (T_4 - T_1) = \text{Average}$



If the same tag uses stair-step interpolation, the time-weighted average is:

$((P_1 \times (T_2 - T_1)) + (P_2 \times (T_3 - T_2)) + (P_3 \times (T_4 - T_3))) / (T_4 - T_1) = \text{Average}$

The SQL Server AVG aggregate is a simple statistical average. Using the average retrieval mode with a cycle count of 1 returns data much faster than the AVG aggregate, and usually more accurately due to the time weighting. The Event subsystem also returns a simple statistical average.

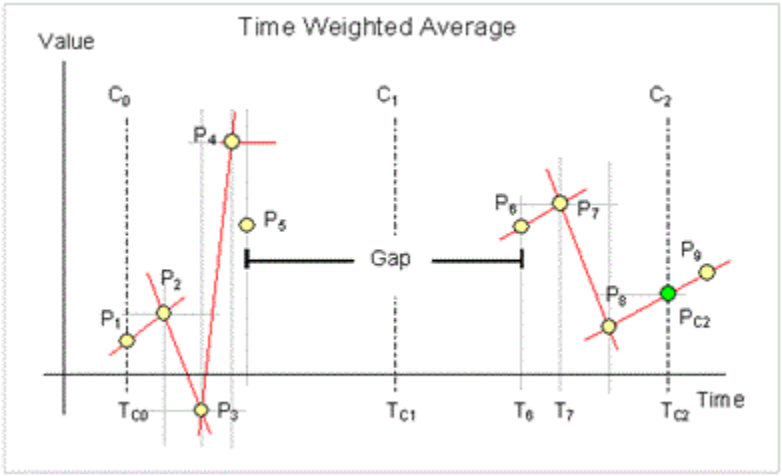
Average retrieval returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

The time-weighted average algorithm is only applied to analog and analog summary tags. If you use average retrieval with other tags, the results are the same as when using regular cyclic retrieval.

Average Retrieval - How It Works

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The following illustration shows how the time-weighted average is calculated for an analog tag that uses linear interpolation.



Data is retrieved in average mode with a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}. P_1 to P_9 represent actual data points stored on the historian. Of these points, eight represent normal analog values, and one, P_5, represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_5 and P_6. Assume that the query calls for timestamping at the end of the cycle.

Results are calculated as follows:

- The "initial value" returned at the query start time (T_{C0}) is the time-weighted average of the points in the last cycle preceding T_{C0}.
- The value returned at T_{C1} is the time-weighted average of the points in the cycle starting at T_{C0}.
- The value returned at the query end time (T_{C2}) is the time-weighted average of the points in the cycle starting at T_{C1}.

To understand how the time-weighted average is calculated, observe the last cycle as an example. First, the area under the curve must be calculated. This curve is indicated by the red line through P_6, P_7, P_8 and P_{C2}, where P_{C2} represents the interpolated value at time T_{C2} using points P_8 and P_9. The data gap caused by the I/O Server disconnect does not contribute anything to this area. If a quality rule of "good" has been specified, then points with doubtful quality will not contribute anything to the area, either.

To understand how the area is calculated, consider points P_6 and P_7. The area contribution between these two points is calculated by multiplying the arithmetic average of value P_6 and value P_7 by the time difference between the two points. The formula is:

$((P_6 + P_7) / 2) \times (T_7 - T_6)$

When the area for the whole cycle has been calculated, the time-weighted average is calculated by dividing that area by the cycle time, less any periods within the cycle that did not contribute anything to the area calculation. The result is returned at the cycle end time.

If you take a closer look at points P_4 and P_5 in the example, you can see that the red line through point P_4 is parallel to the x-axis. This is because P_5 represents a NULL, which cannot be used to calculate an arithmetic average. Instead, the value P_4 is used for the whole time period between points P_4 and P_5.

The area calculation is signed. If the arithmetic average between two points is negative, then the contribution to the area is negative.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Average Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in average retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Interpolation Type (wwInterpolationType)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Average Retrieval - Query Examples

To use the average mode, set the following parameter in your query.

```
   wwRetrievalMode = 'Average'
```

For examples, see the following:

- Average Retrieval - Query 1
- Average Retrieval - Query 2

For an additional example, see Querying Aggregate Data in Different Ways.

Average Retrieval - Query 1

The time-weighted average is computed for each of five 1-minute long cycles.

Note that the wwTimeStampRule parameter is set to "Start" in the query. This means that the value stamped at 11:18:00.000 represents the average for the interval 11:18 to 11:19, the value stamped at 11:19:00.000 represents the average for the interval 11:19 to 11:20, and so on. If no timestamp rule is specified in the query, then the default setting in the TimeStampRule system parameter is used.

In the first cycle there are no points, so a NULL is returned. In the second cycle value points are found covering 77.72 percent of the time as returned in PercentGood. This means that the returned average is calculated based on 77.72 percent of the cycle time. Because the same OPCQuality is not found for all the points in the cycle, OPCQuality is set to Doubtful. In the remaining three cycles, only good points occur, all with an OPCQuality of 192.

Because no quality rule is specified in the query using the wwQualityRule parameter, the query uses the default as specified by the QualityRule system parameter. If a quality rule of Extended is specified, any point stored with doubtful OPCQuality will be used to calculate the average, and the point time will therefore be included in the calculation of PercentGood.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value, OPCQuality, PercentGood
FROM History
   WHERE TagName = 'ReactTemp'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      AND DateTime >= '2005-04-11 11:18:00'
      AND DateTime < '2005-04-11 11:23:00'
      AND wwRetrievalMode = 'Average'
      AND wwCycleCount = 5
      AND wwTimeStampRule = 'Start'
```

The results are:

	DateTime	TagName	Value	OPCQuality	PercentGood
(cycle 1)	2005-04-11 11:18:00.000	ReactTemp	NULL	0	0.0
(cycle 2)	2005-04-11 11:19:00.000	ReactTemp	70.00	64	77.72
(cycle 3)	2005-04-11 11:20:00.000	ReactTemp	153.99	192	100.0
(cycle 4)	2005-04-11 11:21:00.000	ReactTemp	34.31	192	100.0
(cycle 5)	2005-04-11 11:22:00.000	ReactTemp	134.75	192	100.0

Average Retrieval - Query 2

This query demonstrates the use of the average retrieval mode in a wide query. Time-weighted average values are returned for the analog tags ReactTemp and ReactLevel, while regular cyclic points are returned for the discrete tag, WaterValve.

```
SELECT * FROM OpenQuery(INSQL,
   'SELECT DateTime, ReactTemp, ReactLevel, WaterValve FROM WideHistory
      WHERE DateTime >= "2004-06-07 08:00"
         AND DateTime < "2004-06-07 08:05"
         AND wwRetrievalMode = "Average"
         AND wwCycleCount = 5
   ')
```

The results are:

DateTime	ReactTemp	ReactLevel	WaterValve
2004-06-07 08:00:00.000	47.71621	1676.69716	1
2004-06-07 08:01:00.000	157.28076	1370.88097	0
2004-06-07 08:02:00.000	41.33734	797.67296	1
2004-06-07	122.99525	1921.66771	0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

08:03:00.000			
2004-06-07 08:04:00.000	105.28866	606.40488	1

Average Retrieval - Initial and Final Values

If wwTimeStampRule = END, the initial value is calculated by performing an average calculation on the cycle leading up to the query start time. No special handling is done for the final value.

If wwTimeStampRule = START, the final value is calculated by performing an average calculation on the cycle following the query end time. No special handling is done for the initial value.

Average Retrieval - Handling NULL Values

Gaps introduced by NULL values are not included in the average calculations. The average only considers the time ranges with good values. TimeGood indicates the total time per cycle that the tags value was good.

Minimum Retrieval

The minimum value retrieval mode returns the minimum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. NULL is returned if the cycle contains one or more NULL values.

As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, minimum retrieval is not a true cyclic mode. Apart from the initial value, all points returned are delta points.

Minimum retrieval only works with analog tags. For all other tags, normal delta results are returned.

All returned values are in chronological order. If multiple points are to be returned for a particular timestamp, they are returned in the order in which the tags were specified in the query. If the minimum value occurs several times in a cycle, the minimum value with the earliest timestamp is returned.

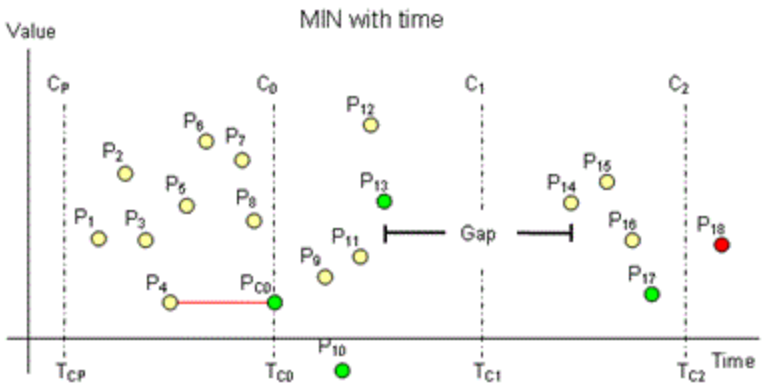
The minimum retrieval mode must use the "<=" operator for the ending date/time.

Using the minimum retrieval mode with a cycle count of 1 returns the same results as the SQL Server MIN aggregate; however, the data is returned much faster.

Minimum Retrieval - How It Works

The following illustration shows how the minimum value is selected for an analog tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



This example has a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1}, a "phantom" cycle starting at T_{CP}, and an incomplete cycle starting at T_{C2}. The phantom cycle has the same duration as the first cycle in the query period, extending back in time from the query start time.

For the queried tag, a total of 18 points are found throughout the cycles, represented by the markers P_1 through P_{18}. Of these points, 17 represent normal analog values. The point P_{13} represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_{13} and P_{14}.

The minimum value for the "phantom" cycle starting at T_{CP} is returned as the initial value at T_{C0}. Point P_{18} is not considered at all because it is outside of the query time frame. All other points are considered, but only the points indicated by green markers on the graph are returned (P_{10}, P_{13}, and P_{17}).

In total, four points are returned:

- P_4 as the minimum value of the "phantom" cycle and the initial point
- P_{10} as the minimum value in the first cycle
- P_{13} as the first and only exception occurring in the first cycle
- P_{17} as the minimum value in the second cycle

No points are returned for the incomplete third cycle starting at the query end time, because the tag does not have a point exactly at that time.

If the minimum value of the first cycle is located exactly at the query start time, both this value and the minimum value of the phantom cycle are returned.

Minimum Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in minimum retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Minimum Retrieval - Query Examples

To use the minimum mode, set the following parameter in your query:

```
wwRetrievalMode = 'Min'
```

or

```
wwRetrievalMode = 'Minimum'
```

For examples, see the following:

- Minimum Retrieval - Query 1
- Minimum Retrieval - Query 2
- Minimum Retrieval - Query 3

Minimum Retrieval - Query 1

In this example, an analog tag is retrieved over a five minute period, using the minimum retrieval mode. Because the wwResolution parameter is set to 60000, each cycle is exactly one minute long. The minimum data value is returned from each of these cycles.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
   AND DateTime >= '2005-04-11 11:21:00'
   AND DateTime <= '2005-04-11 11:26:00'
   AND wwRetrievalMode = 'Minimum'
   AND wwResolution = 60000
```

The initial value at the query start time is the minimum value found in the phantom cycle before the start time of the query.

The results are:

	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:21:00.000	ReactTemp	104.00
(cycle 1)	2005-04-11 11:21:30.837	ReactTemp	14.00
(cycle 2)	2005-04-11 11:22:00.897	ReactTemp	36.00
(cycle 3)	2005-04-11 11:23:59.567	ReactTemp	18.60
(cycle 4)	2005-04-11 11:24:02.083	ReactTemp	14.00
(cycle 5)	2005-04-11 11:25:59.550	ReactTemp	108.60

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Minimum Retrieval - Query 2

In this example, the minimum retrieval mode is used in a manner equivalent to using the SQL Server MIN aggregate. Note that the cycle producing the result is the five-minute phantom cycle just before the query start time.

```
SELECT TOP 1 DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:31:00'
      AND DateTime <= '2005-04-11 11:31:00'
      AND wwRetrievalMode = 'Minimum'
      AND wwResolution = 300000
```

The results are:

	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:31:00.000	ReactTemp	14.00

Minimum Retrieval - Query 3

This example shows how the minimum retrieval mode marks the QualityDetail column to indicate that a minimum value is returned based on an incomplete cycle. In this case, an incomplete cycle is a cycle that either contained periods with no values stored or a cycle that was cut short because the query end time was located inside the cycle. All values returned for the QualityDetail column are documented in the QualityMap table in the Runtime database.

```
SELECT DateTime, TagName, Value, QualityDetail FROM History
   WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2005-04-11 11:18:50'
      AND DateTime <= '2005-04-11 11:20:50'
      AND wwRetrievalMode = 'Minimum'
      AND wwResolution = 60000
```

The results are:

	DateTime	TagName	Value	QualityDetail
(phantom cycle)	2005-04-11 11:18:50.000	SysTimeSec	NULL	65536
(cycle 1)	2005-04-11 11:19:13.000	SysTimeSec	13.0	4140
(cycle 2)	2005-04-11 11:20:00.000	SysTimeSec	0.0	192
(cycle 3)	2005-04-11 11:20:50.000	SysTimeSec	50.0	4288

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Minimum Retrieval - Initial and Final Values

For analog tags, the minimum value of the tag in the cycle leading up to the query start time is returned with its timestamp changed to the query start time. If there is no point exactly at the "phantom" cycle start time, the point leading up to the phantom cycle is also considered for the minimum calculation.(No adjustments are made to the quality of the initial point even though the timestamp may have been altered.) Apart from the initial value, all points returned are delta points. (For more information on initial values, see Delta Retrieval - Initial Values.)

If a point occurs exactly on the query end time, that point will be returned with the partial cycle bit, 4096, set in quality detail. If there is more than one such point, only the first point will be returned.

Minimum Retrieval - Handling NULL Values and Incomplete Cycles

The first NULL value in a cycle is returned.

When a minimum value is returned from a cycle that contains gaps (including a gap extended from the previous cycle) or from an incomplete cycle with the query end time located inside of the calculation cycle, the point's quality detail is modified to flag this. This is done by performing a logical OR operation of the value 4096, which indicates a partial cycle, onto the existing quality detail.

As an example of how minimum retrieval mode handles NULLs, consider the following query:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Minimum'
```

This query can be run against the following sample data:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:09	0.2	192
A001	2009-09-12 00:15	1.3	192
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:22	0.5	192
A001	2009-09-12 00:26	0.9	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:33	1.1	192
A001	2009-09-12 00:35	1.6	192
A001	2009-09-12 00:38	0.5	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A001	2009-09-12 00:42	0.8	192

The following is a graphical representation of the data:

minimum_mode_new_example

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	0.2	192
A001	2009-09-12 00:22	0.5	4288
A001	2009-09-12 00:28	NULL	249
A001	2009-09-12 00:38	0.5	4288

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query. The resolution is set at 10,000 milliseconds.

Because there is no value that matches the start time, an initial value at 00:20 is returned based on the minimum value of the preceding cycle, which is the data point at 00:09. In the two subsequent cycles, the minimum values are at 00:22 and 00:38. The quality for these two values is set to 4288 (4096 + 192). The remaining data points are excluded because they are not minimums. In addition, the first NULL at 00:28 is included, but the second NULL (at 00:29) is not.

Maximum Retrieval

The maximum value retrieval mode returns the maximum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. NULL is returned if the cycle contains one or more NULL values.

As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, maximum retrieval is not a true cyclic mode. Apart from the initial value, all points returned are delta points.

Maximum retrieval only works with analog tags. For all other tags, normal delta results are returned.

All returned values are in chronological order. If multiple points are to be returned for a particular timestamp, they are returned in the order in which the tags were specified in the query. If the maximum value occurs several times in a cycle, the maximum value with the earliest timestamp is returned.

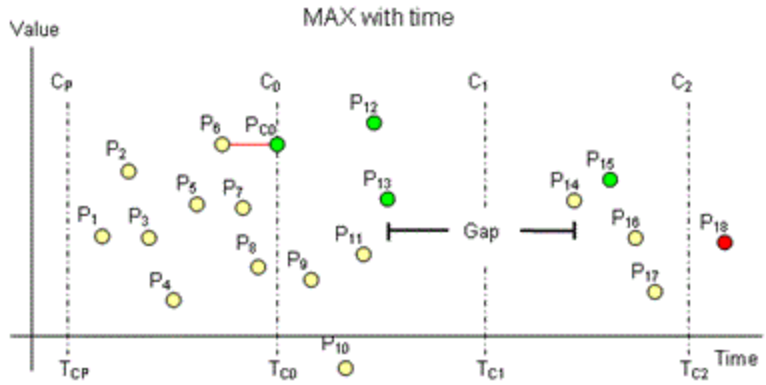
The maximum retrieval mode must use the "<=" operator for the ending date/time.

Using the maximum retrieval mode with a cycle count of 1 returns the same results as the SQL Server MAX aggregate; however, the data is returned much faster.

Maximum Retrieval - How It Works

The following illustration shows how the maximum value is selected for an analog tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



This example has a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1}, a "phantom" cycle starting at T_{CP}, and an incomplete cycle starting at T_{C2}. The phantom cycle has the same duration as the first cycle in the query period, extending back in time from the query start time.

For the queried tag, a total of 18 points are found throughout the cycles, represented by the markers P_1 through P_{18}. Of these points, 17 represent normal analog values. The point P_{13} represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_{13} and P_{14}.

The maximum value for the "phantom" cycle starting at T_{CP} is returned as the initial value at T_{C0}. Point P_{18} is not considered at all because it is outside of the query time frame. All other points are considered, but only the points indicated by green markers on the graph are returned (P_{12}, P_{13}, and P_{15}).

In total, four points are returned:

- P_6 as the maximum value of the "phantom" cycle and the initial point
- P_{12} as the maximum value in the first cycle
- P_{13} as the first and only exception occurring in the first cycle
- P_{15} as the maximum value in the second cycle

No points are returned for the incomplete third cycle starting at the query end time, because the tag does not have a point exactly at that time.

If the maximum value of the first cycle is located exactly at the query start time, this value and the maximum value of the phantom cycle are returned.

Maximum Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in maximum retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Maximum Retrieval - Query Examples

To use the maximum mode, set the following parameter in your query:

```
wwRetrievalMode = 'Max'
```

or

```
wwRetrievalMode = 'Maximum'
```

For examples, see the following:

- Maximum Retrieval - Query 1
- Maximum Retrieval - Query 2
- Maximum Retrieval - Query 3

Maximum Retrieval - Query 1

In this example, an analog tag is retrieved over a five-minute period, using the maximum retrieval mode. Because the wwResolution parameter is set to 60000, each cycle is exactly one minute long. The maximum data value is returned from each of these cycles.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:21:00'
      AND DateTime <= '2005-04-11 11:26:00'
      AND wwRetrievalMode = 'Maximum'
      AND wwResolution = 60000
```

The initial value at the query start time is the maximum value found in the phantom cycle before the start time of the query.

The results are:

Cycle	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:21:00.000	ReactTemp	196.00
(cycle 1)	2005-04-11 11:21:00.853	ReactTemp	101.70
(cycle 2)	2005-04-11 11:22:40.837	ReactTemp	196.00
(cycle 3)	2005-04-11 11:23:00.833	ReactTemp	159.20
(cycle 4)	2005-04-11 11:24:59.613	ReactTemp	146.00
(cycle 5)	2005-04-11 11:25:12.083	ReactTemp	196.00

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Maximum Retrieval - Query 2

In this example, the maximum retrieval mode is used in a manner equivalent to using the SQL Server MIN aggregate. Note that the cycle producing the result is the five-minute phantom cycle just before the query start time.

```
SELECT TOP 1 DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:31:00'
      AND DateTime <= '2005-04-11 11:31:00'
      AND wwRetrievalMode = 'Maximum'
      AND wwResolution = 300000
```

The results are:

	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:31:00.000	ReactTemp	196.00

Maximum Retrieval - Query 3

This example shows how the maximum retrieval mode marks the QualityDetail column to indicate that a maximum value is returned based on an incomplete cycle. In this case, an incomplete cycle is a cycle that either contained periods with no values stored or a cycle that was cut short because the query end time was located inside the cycle. All values returned for the QualityDetail column are documented in the QualityMap table in the Runtime database.

```
SELECT DateTime, TagName, Value, QualityDetail FROM History
   WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2005-04-11 11:19:10'
      AND DateTime <= '2005-04-11 11:21:10'
      AND wwRetrievalMode = 'Maximum'
      AND wwResolution = 60000
```

The results are:

	DateTime	TagName	Value	QualityDetail
(phantom cycle)	2005-04-11 11:19:10.000	SysTimeSec	NULL	65536
(cycle 1)	2005-04-11 11:19:59.000	SysTimeSec	59	4288
(cycle 2)	2005-04-11 11:20:59.000	SysTimeSec	59	192
(cycle 3)	2005-04-11 11:21:10.000	SysTimeSec	10	4288

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Maximum Retrieval - Initial and Final Values

For analog tags, the maximum value of the tag in the cycle leading up to the query start time is returned with its timestamp changed to the query start time. If there is no point exactly at the phantom cycle start time, the point leading up to the phantom cycle is also considered for the maximum calculation. No adjustments are made to the quality of the initial point even though the timestamp may have been altered. Apart from the initial value, all points returned are delta points. (For more information on initial values, see Determining Cycle Boundaries.)

If a point occurs exactly on the query end time, that point is returned with the partial cycle bit, 4096, set in quality detail. If there is more than one such point, only the first point is returned.

Maximum Retrieval - Handling NULL Values and Incomplete Cycles

The first NULL value in a cycle is returned.

When a maximum value is returned from a cycle that contains gaps (including a gap extended from the previous cycle) or from an incomplete cycle with the query end time located inside of the calculation cycle, the point's quality detail is modified to flag this. This is done by performing a logical OR operation of the value 4096, which indicates a partial cycle, onto the existing quality detail.

As an example of how maximum retrieval mode handles NULLs, consider the following query:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Maximum'
```

If you run this query against the following sample data:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:09	0.2	192
A001	2009-09-12 00:15	1.3	192
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:22	0.5	192
A001	2009-09-12 00:26	0.9	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:33	1.1	192
A001	2009-09-12 00:35	1.6	192
A001	2009-09-12 00:38	0.5	192
A001	2009-09-12 00:42	0.8	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	1.3	192
A001	2009-09-12 00:26	0.9	4288
A001	2009-09-12 00:28	NULL	249
A001	2009-09-12 00:35	1.6	4288

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query. The resolution is set at 10,000 milliseconds.

Because there is no value that matches the start time, an initial value at 00:20 is returned based on the maximum value of the preceding cycle, which is the data point at 00:15. In the two subsequent cycles, the maximum values are at 00:26 and 00:35. The quality for these two values is set to 4288 (4096 + 192). The remaining data points are excluded because they are not maximums. In addition, the first NULL at 00:28 is included, but the second NULL (at 00:29) is not.

maximum_mode_new_example

Integral Retrieval

Integral retrieval calculates the values at retrieval cycle boundaries by integrating the graph described by the points stored for the tag. Therefore, it works much like average retrieval, but it additionally applies a scaling factor. This retrieval mode is useful for calculating volume for a particular tag. For example, if one of your tags represents product flow in gallons per second, integral retrieval allows you to retrieve the total product flow in gallons during a certain time period.

Integral retrieval is a true cyclic mode. It returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

Integral retrieval only works with analog tags. For all other tags, normal cyclic results are returned.

Integral Retrieval - How It Works

Calculating values for a cycle in integral retrieval is a two-step process:

- First, the historian calculates the area under the graph created by the data points. This works the same as in average retrieval. For more information, see Average Retrieval.
- After this area has been found, it is scaled using the value of the IntegralDivisor column in the EngineeringUnit table. This divisor expresses the conversion factor from the actual rate to one of units per second.

For example, if the time-weighted average for a tag during a 1-minute cycle is 3.5 liters per second, integral retrieval returns a value of 210 for that cycle (3.5 liters per second multiplied by 60 seconds).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Integral Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in integral retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Interpolation Type (wwInterpolationType)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Integral Retrieval - Query Example

To use the integral retrieval mode, set the following parameter in your query.

```
   wwRetrievalMode = 'Integral'
```

In this example, the integral is computed for each of five 1-minute long cycles. The wwQualityRule parameter is used to ensure that only points with good quality are used in the computation, which means that points with doubtful quality are discarded. The rules used to determine the returned OPCQuality are the same as described for a time weighted average query.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Flow, OPCQuality, PercentGood
FROM History
   WHERE TagName = 'FlowRate'
      AND DateTime >= '2004-06-07 08:00'
      AND DateTime < '2004-06-07 08:05'
      AND wwRetrievalMode = 'Integral'
      AND wwCycleCount = 5
      AND wwQualityRule = 'Good'
```

The results are:

	DateTime	TagName	Flow	OPCQuality	PercentGood
(interval 1)	2004-06-07 08:00:00.000	FlowRate	2862.97	192	100.0
(interval 2)	2004-06-07 08:01:00.000	FlowRate	9436.85	192	100.0
(interval 3)	2004-06-07 08:02:00.000	FlowRate	2480.24	192	100.0
(interval 4)	2004-06-07 08:03:00.000	FlowRate	7379.71	192	100.0
(interval 5)	2004-06-07	FlowRate	6317.32	192	100.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	08:04:00.000				

Also, the "phantom" cycle affects the integral retrieval mode just as it does the average retrieval mode. For examples, see Querying Aggregate Data in Different Ways.

Integral Retrieval - wwExpression Query Example

In this example, the integral of a tag named Speed (measured in m/s), is computed over two 1-minute long intervals (wwResolution = 60000 milliseconds) to measure the distance traveled in meters over the interval.

```
SELECT StartDateTime, DateTime, TagName, Value, Quality, QualityDetail as QD, wwUnit
   FROM History
   WHERE TagName = 'Speed'
      AND DateTime >= '2020-12-03 10:00'
      AND DateTime < '2020-12-03 10:02'
      AND wwRetrievalMode = 'Integral'
      AND wwResolution = 60000
```

The results are:

StartDateTime	DateTime	TagName	Value	Quality	QD	wwUnit
2020-12-03 09:59:00.000	2020-12-03 10:00:00.000	Speed	-2906.88948	0	192	m
2020-12-03 10:00:00.000	2020-12-03 10:01:00.000	Speed	877.57619	0	192	m

To display the results using a different unit of measure, you can use the wwExpression query parameter with the following syntax:

```
   wwExpression = 'UOM(TagName, target unit)'
```

For example, the integral of the Speed tag is computed again over two 1-minute long intervals, but this time the value is converted to the distance in centimeters.

```
SELECT StartDateTime, DateTime, TagName, Value, Quality, QualityDetail as QD, wwUnit
   FROM History
   WHERE TagName = 'Speed'
      AND DateTime >= '2020-12-03 10:00'
      AND DateTime < '2020-12-03 10:02'
      AND wwRetrievalMode = 'Integral'
      AND wwResolution = 60000
      AND wwExpression = 'UOM([Speed],cm)'
```

The results are:

StartDateTime	DateTime	TagName	Value	Quality	QD	wwUnit
2020-12-03 09:59:00.000	2020-12-03 10:00:00.000	Speed	-290688.948	192	192	cm

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2020-12-03 10:00:00.000	2020-12-03 10:01:00.000	Speed	87757.619	192	192	cm

Note: Engineering unit conversion requires the advanced feature license. If your Historian does not have the advanced feature license activated, the results are returned in the original units.

Integral Retrieval - Initial and Final Values

If wwTimeStampRule = END, the initial value is calculated by performing an integral calculation on the cycle leading up to the query start time. No special handling is done for the final value.

If wwTimeStampRule = START, the final value is calculated by performing an integral calculation on the cycle following the query end time. No special handling is done for the initial value.

Integral Retrieval - Handling NULL Values

Gaps introduced by NULL values are not included in the integral calculations. The average only considers the time ranges with good values. TimeGood indicates the total time per cycle that the tags value was good.

Slope Retrieval

Slope retrieval returns the slope of a line drawn through a given point and the point immediately before it, thus expressing the rate at which values change.

This retrieval mode is useful for detecting if a tag is changing at too great a rate. For example, you might have a temperature that should steadily rise and fall by a small amount, and a sharp increase or decrease could point to a potential problem.

The slope retrieval mode can be considered a delta mode. Apart from the initial value and a value at the query end time, all returned points are calculated delta points returned with the timestamp of an actual delta point.

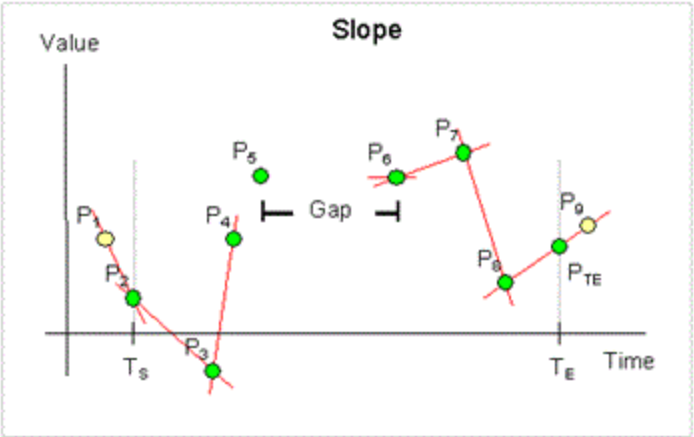
Slope retrieval only works with analog tags. For all other tags, normal delta results are returned.

All returned values are in chronological order. If multiple points are to be returned for a particular timestamp, they are returned in the order in which the tags were specified in the query.

Slope Retrieval - How It Works

The following illustration shows how the slope is calculated for an analog tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



This example has a start time of T_S and an end time of T_E.

For the queried tag, a total of nine points are found, represented by the markers P_1 through P_9. Of these points, eight represent normal analog values. The point P_5 represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_5 and P_6.

For every point in the time period, slope retrieval returns the slope of the line going through that point and the point immediately before it. For two points P_1 and P_2 occurring at times T_1 and T_2, the slope formula is as follows:

$(P_2 - P_1) / (T_2 - T_1)$

The difference between T_1 and T_2 is measured in seconds. Therefore, the returned value represents the change in Engineering Units per second.

In this example, point P_2 is located at the query start time, and because there is a prior value (P_1), the slope of the line through both points is calculated and returned at time T_S. Similarly, slopes are calculated to be returned at times T_3, T_4, T_7, and T_8. The slope is also calculated for the line through P_8 and P_9, but that value is returned as point P_{TE} at the query end time.

For point P_6, there is no prior point with which to perform a slope calculation. Instead, the slope of the flat line going through the point (that is, the value 0) is calculated. At the time of P_5, NULL is returned.

The quality detail and OPC quality returned with a slope point is always directly inherited from the point that also provides the time stamp. In this example, this means that point P_2 provides the qualities for the slope point returned at the query start time, T_S.

Slope Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in slope retrieval mode. For more information, see the following sections:

- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Slope Retrieval - Query Example

To use the slope retrieval mode, set the following parameter in your query.

```
   wwRetrievalMode = 'Slope'
```

For example, the following query calculates and returns the rate of change of the ReactTemp tag in °C/second. The initial value in the Quality column at the query start time shows no value is located exactly at that time, so the slope returned is the same as the one returned at the next delta point. (For more information on initial values, see Determining Cycle Boundaries.)

At 08:01:17.947 the tag has two delta points, so a slope is calculated and returned for the first point, while a NULL is returned at the second one with a special QualityDetail of 17, indicating that no slope can be calculated as it is either plus or minus infinite.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 4), Value) AS Slope, Quality, QualityDetail
FROM History
   WHERE TagName = 'ReactTemp'
         AND DateTime >= '2005-04-17 08:00'
         AND DateTime <= '2005-04-17 08:05'
         AND wwRetrievalMode = 'Slope'
```

The results are:

DateTime	TagName	Slope	Quality	QualityDetail
2005-04-17 08:00:00.000	ReactTemp	3.8110	133	192
2005-04-17 08:00:00.510	ReactTemp	3.8110	0	192
2005-04-17 08:00:01.713	ReactTemp	4.1563	0	192
2005-04-17 08:00:02.917	ReactTemp	4.1563	0	192
2005-04-17 08:00:04.230	ReactTemp	3.8081	0	192
2005-04-17 08:00:05.433	ReactTemp	4.1563	0	192
…	…	…		
2005-04-17 08:01:16.743	ReactTemp	-1.7517	0	192
2005-04-17 08:01:17.947	ReactTemp	-27.0158	0	192
2005-04-17 08:01:17.947	ReactTemp	NULL	1	17

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Slope	Quality	QualityDetail
2005-04-17 08:01:19.260	ReactTemp	-1.7530	0	192
2005-04-17 08:01:20.463	ReactTemp	-1.9119	0	192
2005-04-17 08:01:21.667	ReactTemp	-1.9119	0	192
2005-04-17 08:01:22.977	ReactTemp	-1.7517	0	192
…	…	…		

Slope Retrieval - wwExpression Query Example

The following query calculates and returns the rate of change of the Distance tag. The results are returned in the tag's default unit of measure, which in this case is feet/second.

```
SELECT DateTime, TagName, Value, QualityDetail, wwUnit FROM History
   WHERE TagName = 'Distance'
          AND wwRetrievalMode = 'Slope'
```

The results are:

DateTime	TagName	Value	QualityDetail	wwUnit
2020-07-29 06:50:53.748	Distance	1	192	ft/s
2020-07-29 06:50:54.748	Distance	2	192	ft/s

To display the results using a different unit of measure, you can use the wwExpression query parameter with the following syntax:

```
   wwExpression = 'UOM(TagName, target unit)'
```

For example, the following query calculates and returns the rate of change of the Distance tag in meters/second.

```
SELECT DateTime, TagName, Value, QualityDetail, wwUnit FROM History
   WHERE TagName = 'Distance'
          AND wwExpression = 'UOM(Distance, m/s)'
          AND wwRetrievalMode = 'Slope'
```

The results are:

DateTime	TagName	Value	QualityDetail	wwUnit
2020-07-29 06:50:53.748	Distance	0.3048	192	m/s
2020-07-29	Distance	0.6096	192	m/s

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value	QualityDetail	wwUnit
06:50:54.748				

Note: Engineering unit conversion requires the advanced feature license. If your Historian does not have the advanced feature license activated, the results are returned in the original units.

Slope Retrieval - Initial and Final Values

An initial value is always generated. If a point is stored exactly at the query start time, the slope is returned as the slope between that point and the previous point. Otherwise, the slope is calculated using the slope of the points before and after the query start time.

A final value is always generated. If a point is stored exactly at the query end time, the slope is returned as the slope between that point and the previous point. Otherwise, the slope is calculated using the slope of the points before and after the query end time.

Slope Retrieval - Handling NULL Values

The first NULL following a non-NULL value is returned. Subsequent NULL values are not. If a point is preceded by a NULL, the slope for that point will be zero.

Counter Retrieval

Counter retrieval allows you to accurately retrieve the delta change of a tag's value over a period of time even for tags that are reset upon reaching a "rollover value." The rollover value is defined in the AVEVA Historian for each tag.

This retrieval mode is useful for determining how much of an item was produced during a particular time period. For example, you might have an integer counter that keeps track of how many cartons were produced. The counter has an indicator like this:

The next value after the highest value that can be physically shown by the counter is called the rollover value. In this example, the rollover value is 10,000. When the counter reaches the 9,999th value, the counter rolls back to 0. Therefore, a counter value of 9,900 at one time and a value of 100 at a later time means that you have produced 200 units during that period, even though the counter value has dropped by 9,800 (9,900 minus 100). Counter retrieval allows you to handle this situation and receive the correct value. For each cycle, the counter retrieval mode shows the increase in that counter during the cycle, including rollovers.

Note: If Historian receives a data tag value that is outside of the specified engineering unit boundaries, Historian ignores the received value and instead returns a 0. For example, if a value can be an integer between 1 and 8, and the received value is 9, Historian returns 0 as the value for that tag during that cycle.

Counter retrieval also works with floating point counters, which is useful for flow meter data. Similar to the carton counter, some flow meters "roll over" after a certain amount of flow accumulates. For both examples, the need is to convert the accumulating measure to a "delta change" value over a given period.

Counter retrieval is a true cyclic mode. It returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

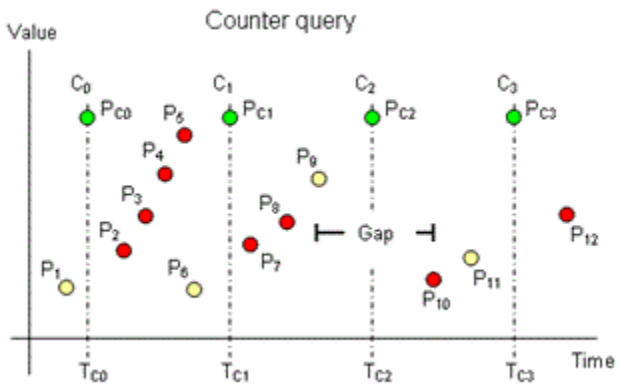
The counter algorithm is only applied to analog tags and to discrete tags. For integer analog tags, the result will be an integer returned as a float data type. For a real analog tag, the rollover value and the result may be real values and can include fractional values. If a query contains tags of other types, then no rows are returned for those tags. For discrete tags, the rollover value is assumed to be 2.

The rules used to determine the OPCQuality returned with a counter value are the same as for a time weighted average query. For more information, see Quality Rule (wwQualityRule). When a rollover has occurred in the calculation cycle, a special quality detail of 212 is returned in all non-NULL cases.

CTU counters will default to "signed integer" tags when imported into the Historian, giving a normal range of -2147483648 to 2147483647 (for a 32-bit integer). In operation, these counters will count up to the upper limit and "rollover" to the lower limit on the next increment. If these tags are changed to be "unsigned integers" the normal range will be 0 to 4294967295 and values will rollover to "0", conforming to the expected behavior of a tag used with "counter" retrieval. When used with a 16-bit CTU counter, the same rules apply, but the range of values is -32768 to 32767 as "signed" and 0 to 65535 for an "unsigned".

Counter Retrieval - How It Works

The following illustration shows how the counter algorithm determines the count for an analog tag.



This example has a start time of T_{C0} and an end time of T_{C3}. The resolution has been set in such a way that the historian returns data for three complete cycles starting at T_{C0}, T_{C1}, and T_{C2}, and an incomplete cycle starting at T_{C3}.

For the queried tag, a total of twelve points are found throughout the cycles represented by the markers P_1 through P_{12}. Of these points, eleven represent normal analog values. The point P_9 represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_9 and P_{10}. Point P_{12} is not considered because it is outside of the query time frame.

All points are considered in the counter calculation, but only the yellow ones are actually used to determine which values to return to the client. The returned points are P_{C0}, P_{C1}, P_{C2} and P_{C3}, shown in green at the top to indicate that there is no simple relationship between them and any of the actual points.

All cycle values are calculated as the delta change between the cycle time in question and the previous cycle time, taking into account the number of rollovers between the two points in time. The counter algorithm assumes that a rollover occurred if the current value is lower than the previous value. The initial value at the query start time (P_{C1}) is calculated the same way, only based on a phantom cycle before the query start time.

For example, the formula to calculate P_{C1} is as follows:

$P_{C1} = n * V_R + P_6 - P_1$

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

where:

- *n* = the number of rollovers that have occurred during the cycle
- V_R = the rollover value for the tag

If either *n* or V_R are equal to zero, P_{C1} is simply the difference between the values P_1 and P_6.

In the case of cycle C_2, there is no value at the cycle time, so the NULL value represented by point P_9 is returned. In the case of cycle C_3, a NULL is again returned, because there is no counter value at the previous cycle boundary to use in the calculation. There must be a full cycle of values in order for the counter to be calculated.

If a gap is fully contained inside a cycle, and if points occur within the cycle on both sides of the gap, then a counter value is returned, even though it may occasionally be erroneous. Zero or one rollovers are assumed, even though the counter might have rolled over multiple times.

Counter Retrieval - Calculations for a Manually Reset Counter

If you have a counter that you typically reset manually before it rolls over, you must set the rollover value for the tag to 0 so that the count is simply how much change occurred since the manual reset.

For example, assume that you have the following data values for five consecutive cycle boundaries, and that the value 0 occurs as the first value within the last cycle:

100, 110, 117, 123, 3

If you set the rollover value to 0, the counter retrieval mode assumes that the 0 following the value 123 represents a manual reset, and returns a value of 3 for the last cycle, which is assumed to be the count after the manual reset. The value 0 itself does not contribute 1 to the counter value in this case.

If the rollover value is instead set to 200, then the counter retrieval mode assumes that the value 0 represents a normal rollover, and a count of 80 is calculated and returned (200 - 123 + 3). In this case, the value 0 contributes 1 to the counter value, and that is the change from the value 199 to the value 200.

Counter Retrieval - Using a Counter Deadband

You can set a deadband for counter retrieval to better handle reporting of rates and quantities. For example, setting a counter deadband can help to:

- Distinguish between counter resets and rollovers.
- Filter out counter reversals.

The counter deadband is the percentage of the full range of the counter.

For example, if you set the counter deadband to be 10, then any counter reset that occurs within the top 10% of the range is assumed to be a rollover and is counted as such.

For a reversal, you might have a conveyor belt carrying boxes that are counted as they pass by a station. If a jam occurs, you might reverse the conveyor belt to clear it, resulting in the counter decrementing and then incrementing again. In this case, a counter deadband of 10 would discard any duplicate counts within 10% of the range starting from the point of reversal. Only one reversal can be detected until the rollover occurs.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Counter Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in integral retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)

Counter Retrieval - Initial and Final Values

An initial value is returned using the period leading up to the query start time.

A data point that has a cycle time is used to generate the counter value for its preceding cycle. A NULL point with cycle time will cause the preceding cycle to end in a gap and the following cycle to start with a gap.

Counter Retrieval - Handling NULL Values

If wwQualityRule is configured as OPTIMISTIC, NULL data points will not be used in calculation. 0.0 will be used as the starting base value for the query unless the query data starts with a NULL. If the query starts with a NULL, the value change for the cycle is calculated from the first actual value in the cycle, rather than 0.

Otherwise, if any points considered in a cycle have UNCERTAIN quality, the result for that row will also have UNCERTAIN quality. Any cycle that starts or ends in a gap will have a quality detail of 65536.

The quality detail of DOUBTFUL will be used with the counter result for the cycles, if a NULL point is considered for the cycle and the counter result is not NULL.

Counter Retrieval - Handling Illegal Values

If the configured rollover value is larger than 0.0, then the data points whose values are greater than or equal to the rollover value causes the counter value for the cycle to be set to 0.0, with qdIO_FILTEREDPOINT applied to the quality detail.

Similarly, if any data point with a value less than 0.0 is found in a cycle, the counter value for the cycle is set to 0.0, with qdIO_FILTEREDPOINT applied to the quality detail.

Counter Retrieval - Query Example

To use the counter mode, set the following parameter in your query.

```
wwRetrievalMode = 'Counter'
```

In the following example, the rollover value for the SysTimeSec system tag is set to 0. In a five-minute time span, the SysTimeSec tag increments a total of 300 times. The following query returns the total count within the five-minute time span, which begins with StartTime and ends with DateTime. The QualityDetail of 212 indicates that a counter rollover occurred during the query time range.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
select StartDateTime, DateTime, TagName, Value, wwUnit, QualityDetail as QD from History
   where TagName = 'SysTimeSec'
      and DateTime > '2020-12-02 1:00'
      and DateTime <= '2020-12-02 1:05'
      and wwRetrievalMode = 'counter'
      and wwCycleCount = 1
```

The results are:

StartTime	DateTime	TagName	Value	wwUnit	QD
2020-12-02 01:00:00.0000000	2020-12-02 01:05:00.0000000	SysTimeSec	300	Second	212

To display the results using a different unit of measure, you can use the wwExpression query parameter with the following syntax:

```
   wwExpression = 'UOM(TagName, target unit)'
```

The following example is similar to the previous example. The following query returns the total count within the five-minute time span, which begins with StartTime and ends with DateTime, and converts the retrieved value from its original units (seconds) to minutes.

```
select StartDateTime, DateTime, TagName, Value, Quality, QualityDetail as QD from History
   where TagName = 'SysTimeSec'
      and DateTime > '2020-12-02 1:00'
      and DateTime <= '2020-12-02 1:05'
      and wwRetrievalMode = 'counter'
      and wwCycleCount = 1
      and wwExpression = 'UOM(SysTimeSec,Minute)'
```

The results are:

StartTime	DateTime	TagName	Value	wwUnit	QD
2020-12-02 01:00:00.0000000	2020-12-02 01:05:00.0000000	SysTimeSec	5	Minute	212

Note: Engineering unit conversion requires the advanced feature license. If your Historian does not have the advanced feature license activated, the results are returned in the original units.

ValueState Retrieval

ValueState retrieval returns information on how long a tag has been in a particular value state during each retrieval cycle. That is, a time-in-state calculation is applied to the tag value.

This retrieval mode is useful for determining how long a machine has been running or stopped, how much time a process spent in a particular state, how long a valve has been open or closed, and so on. For example, you might have a steam valve that releases steam into a reactor, and you want to know the average amount of time the valve was in the open position during the last hour. ValueState retrieval can return the shortest, longest, average, or total time a tag spent in a state, or the time spent in a state as a percentage of the total cycle length.

When you use ValueState retrieval for a tag in a trend chart, you must specify a single value state for which to retrieve information. ValueState retrieval then returns one value for each cycle—for example, the total amount of time that the valve was in the "open" state during each 1-hour cycle. This information is suitable for trend

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

display.

If you *don't* specify a state, ValueState retrieval returns one row of information for each value that the tag was in during each cycle. For example, this would return not only the time a valve was in the "open" state, but also the time it was in the "closed" state. This information is not suitable for meaningful display in a regular trend. You can, however, retrieve this type of information in a query and view it as a table.

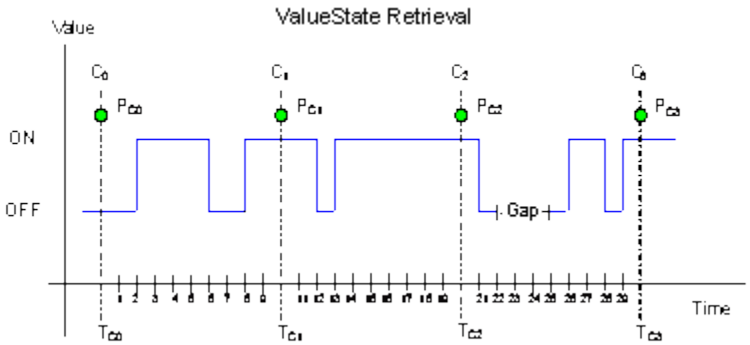
ValueState retrieval works with integer, discrete, string, and state summary tags. For other types of tags, no rows are returned. NULL values are treated like any other distinct state.

The values returned at the query start time are the result of applying the algorithm to a "phantom" cycle preceding the query range. It is assumed that the tag value at the start of the cycle is located at that point in time.

To specify the type of calculation, set the wwStateCalc parameter in the query. For more information, see State Calculation (wwStateCalc).

ValueState Retrieval - How It Works

The following illustration shows how ValueState retrieval returns values for a discrete tag.



This example has a start time of T_{C0} and an end time of T_{C3}. The resolution has been set in such a way that the historian returns data for three complete cycles starting at T_{C0}, T_{C1}, and T_{C2}, and an incomplete cycle starting at T_{C3}. Time is measured seconds.

A gap in the data occurs in the third cycle due to an I/O Server disconnect.

The state calculation is based on each cycle, and the values returned at the query start time are not regular initial values, but are the resulting values that occur after applying the algorithm to the last cycle preceding the query range. The returned points are P_{C0}, P_{C1}, P_{C2} and P_{C3}, shown in green at the top to indicate that there is no simple relationship between the calculated values and any of the actual points.

Assume the query is set so that the total time (wwStateCalc = 'Total')in the two states are returned. The timestamping is set to use the cycle end time.

- For T_{C0}, the query returns two rows (one for the "on" state and one for the "off" state), calculated as a result of the "phantom" cycle that precedes the query start time. The values have a timestamp of the query start time.
- For T_{C1}, one row is returned for the "on" state. The "on" state occurred twice during the cycle--one time for four seconds and again for two seconds before the cycle boundary, and the total time returned is six seconds. The state was "off" twice during the cycle for a total time of four seconds, and one row is returned with that value.
- For T_{C2}, one row is returned for the "on" state, and one row is returned for the "off" state. The "on" state

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

occurred for a total of nine seconds between the cycle boundaries, and the "off" state occurred for a total of one second.

- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The "on" state occurred for a total of four seconds between the cycle boundaries, and the "off" state occurred for a total of three seconds. An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect.

Using the same data, if you queried the **total contained** time for the states, the following is returned:

- For T_{C0}, the query returns two values (one for the "on" state and one for the "off" state), calculated as a result of the "phantom" cycle the precedes the query start time. The value has a timestamp of the query start time.
- For T_{C1}, one row is returned for the "on" state, and one row is returned for the "off" state. The "on" state occurred one time for four seconds within the cycle. The two seconds of "on" time that crosses the cycle boundary does not contribute to the total time. The state was "off" one time during the cycle for two seconds completely within the cycle boundary.
- For T_{C2}, the state was not "on" for any contained time between the cycle. Both occurrences of the "on" state cross over a cycle boundary, so no rows are returned for this state. One row is returned for the "off" state. The state was "off" one time during the cycle for one seconds completely within the cycle boundary.
- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The state was "on" for a single contained time of two seconds between the cycle boundaries. The state was "off" three times during the cycle for three seconds completely within the cycle boundary. An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect. The state was NULL for a total of three seconds. The I/O Server disconnect that "disrupted" the off state is treated as its own state, thereby changing what would have been a single "off" state instance of five seconds into two instances of the "off" state for one second each.

ValueState Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in ValueState retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- State Calculation (wwStateCalc)
- Analog Value Filtering (wwFilter)

ValueState Retrieval - Query Examples

To use the ValueState retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'ValueState'
```

To specify the type of aggregation, set the wwStateCalc parameter in the query, such as:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   wwStateCalc = 'Total'
```

Be sure that you use the "<=" operator for ending date/time.

For examples, see the following:

- ValueState Retrieval Query 1: Minimum Time in State
- ValueState Retrieval Query 2: Minimum Time in State for a Single Tag
- ValueState Retrieval Query 3
- ValueState Retrieval Query 4
- ValueState Retrieval Query 5
- ValueState Retrieval Query 6: Querying State Summary Values

ValueState Retrieval Query 1: Minimum Time in State

The following query finds the minimum time-in-state for the SteamValve discrete tag. Note that minimum times are returned for each state for both the five-minute phantom cycle before the query start time and for the single retrieval cycle between 10:00 and 10:05.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc FROM History
   WHERE TagName IN ('SteamValve')
      AND DateTime >= '2005-04-17 10:00:00'
      AND DateTime <= '2005-04-17 10:05:00'
      AND wwCycleCount = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Min'
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	0	35359.0	MINIMUM
2005-04-17 10:00:00.000	SteamValve	1	43749.0	MINIMUM
2005-04-17 10:05:00.000	SteamValve	0	37887.0	MINIMUM
2005-04-17 10:05:00.000	SteamValve	1	43749.0	MINIMUM

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueState Retrieval Query 2: Minimum Time in State for a Single Tag

The following query finds the minimum time-in-state for the Mixer discrete tag for the "on" state. Note that minimum times are returned for each state for both the five-minute phantom cycle before the query start time and for the single retrieval cycle between 10:00 and 10:05.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc
   FROM History
      WHERE TagName IN ('Mixer')
         AND DateTime >= '2017-12-11 08:00:00'
         AND DateTime < '2017-12-12 08:00:00'
         AND wwCycleCount= '1'
         AND wwRetrievalMode = 'ValueState'
         AND wwStateCalc = 'Min'
         AND vValue = 1
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2017-12-11 08:00:00.0000000	Mixer	1	35906	Min

ValueState Retrieval Query 3

The following query finds the maximum time-in-state for the SteamValve discrete tag in the same time period as in Query 1. Note how both the minimum and maximum values for the "1" state are very similar, while they are very different for the "0" state. This is due to the "cut-off" effect.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc FROM History
   WHERE TagName IN ('SteamValve')
      AND DateTime >= '2005-04-17 10:00:00'
      AND DateTime <= '2005-04-17 10:05:00'
      AND wwCycleCount = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Max'
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	0	107514.0	MAXIMUM
2005-04-17 10:00:00.000	SteamValve	1	43750.0	MAXIMUM
2005-04-17 10:05:00.000	SteamValve	0	107514.0	MAXIMUM

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2005-04-17 10:05:00.000	SteamValve	1	43750.0	MAXIMUM

ValueState Retrieval Query 4

The following query returns the total of time in state for a discrete tag. In this example, the TimeStampRule system parameter is set to "End" (the default setting), so the returned values are timestamped at the end of the cycle. The returned rows represent the time-in-state behavior during the period starting at 2005-04-13 00:00:00.000 and ending at 2005-04-14 00:00:00.000.

```
SELECT DateTime, vValue, StateTime, wwStateCalc FROM History
   WHERE DateTime > '2005-04-13 00:00:00.000'
      AND DateTime <= '2005-04-14 00:00:00.000'
      AND TagName IN ('PumpStatus')
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Total'
      AND wwCycleCount = 1
```

The results are:

DateTime	vValue	StateTime	wwStateCalc
2005-04-14 00:00:00	NULL	1041674.0	TOTAL
2005-04-14 00:00:00	On	56337454.0	TOTAL
2005-04-14 00:00:00	Off	29020872.0	TOTAL

ValueState Retrieval Query 5

The following query returns the percentage of time in state for a discrete tag for multiple retrieval cycles. The TimeStampRule system parameter is set to "End" (the default setting), so the returned values are timestamped at the end of the cycle. Note that the first row returned represents the results for the period starting at 2003-07-03 22:00:00.000 and ending at 2003-07-04 00:00:00.000.

The "Percent" time-in-state retrieval mode is the only mode in which the StateTime column does not return time data. Instead, it returns percentage data (in the range of 0 to 100 percent) representing the percentage of time in state.

```
SELECT DateTime, vValue, StateTime, wwStateCalc FROM History
   WHERE DateTime >= '2003-07-04 00:00:00.000'
      AND DateTime <= '2003-07-05 00:00:00.000'
      AND TagName IN ('PumpStatus')
      AND Value = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Percent'
      AND wwCycleCount = 13
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	vValue	StateTime	wwStateCalc
2003-07-04 00:00:00	1	50.885	PERCENT
2003-07-04 02:00:00	1	82.656	PERCENT
2003-07-04 04:00:00	1	7.082	PERCENT
2003-07-04 06:00:00	1	7.157	PERCENT
2003-07-04 08:00:00	1	55.580	PERCENT
2003-07-04 10:00:00	1	28.047	PERCENT
2003-07-04 12:00:00	1	47.562	PERCENT
2003-07-04 14:00:00	1	74.477	PERCENT
2003-07-04 16:00:00	1	40.450	PERCENT
2003-07-04 18:00:00	1	78.313	PERCENT
2003-07-04 20:00:00	1	54.886	PERCENT
2003-07-04 22:00:00	1	39.569	PERCENT
2003-07-05 00:00:00	1	50.072	PERCENT

ValueState Retrieval Query 6: Querying State Summary Values

If state summary values are queried and the cycle boundaries match the summary periods, the ValueState calculations are supported and return valid results.

If state summary points are queried and the cycle boundaries do not match the summary periods, the ValueState calculations are supported, but they return DOUBTFUL (QualityDetail = 64) results.

State summaries are included in the cycle where the summary end time occurs. This causes results that do not match queries against the source tag and may cause inaccurate results, such as a total state time that is greater than the cycle time.

For example, this can occur if SysTimeSec is summarized with a state summary with one minute resolution, but then queried with 10 second intervals. In most of the retrieval cycles, there will be no values, but in the cycle that includes the summary end time (one in six of the retrieval cycles), all 60 states would be returned with each state having a state time of 1 second for a total of 60 seconds of state time in a 10 second retrieval cycle.

ValueState Retrieval - Initial and Final Values

The values returned at the query start time are the result of applying the algorithm to the last cycle preceding the query range.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueState Retrieval - Handling NULL Values

NULLs are considered a state and are reported along with the other states.

RoundTrip Retrieval

RoundTrip retrieval is very similar to ValueState retrieval in that it performs calculations on state occurrences in the within a cycle period you specify. However, ValueState retrieval uses the time spent in a certain state for the calculation, and RoundTrip retrieval uses the time between consecutive leading edges of the same state for its calculations.

You can use the RoundTrip retrieval mode for increasing the efficiency of a process. For example, if a process produces one item per cycle, then you would want to minimize the time lapse between two consecutive cycles.

The RoundTrip mode returns a row for each state in any given cycle. RoundTrip retrieval only works with integer analog tags, discrete tags, and string tags. If real analog tags are specified in the query, then no rows are returned for these tags. RoundTrip retrieval is not applied to state summary or analog summary tags. NULL values are treated as any other distinct value and are used to analyze the round trip for disturbances.

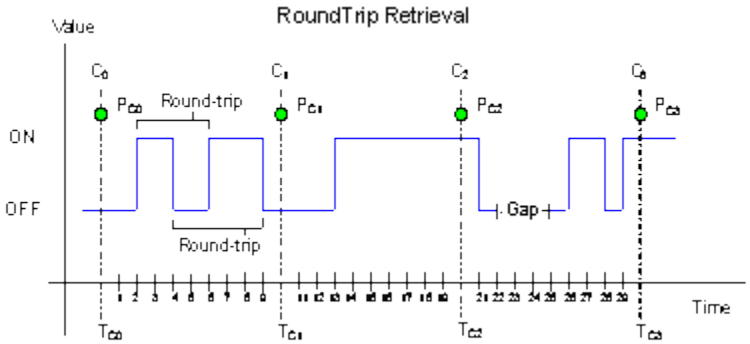
RoundTrip retrieval is supported for the History and StateWideHistory tables.

Any point on the boundary of the end cycle will be considered to the next cycle. The point on the boundary of the end query range is not counted in the calculation except that it is used to indicate that the previous state is a contained state.

If no roundtrip state is found within the cycle for a supported tag, a NULL StateTime value is returned. If there is no valid point prior to the phantom cycle, a NULL state is returned for the phantom cycle.

RoundTrip Retrieval - How It Works

The following illustration shows how RoundTrip retrieval returns values for a discrete tag.



This example has a start time of T_{C0} and an end time of T_{C3}. The resolution has been set in such a way that the historian returns data for three complete cycles starting at T_{C0}, T_{C1}, and T_{C2}, and an incomplete cycle starting at T_{C3}. Time is measured seconds.

A gap in the data occurs in the third cycle due to an I/O Server disconnect.

The state calculation is based on each cycle, and the values returned at the query start time are not regular initial values, but are the resulting values that occur after applying the algorithm to the last cycle preceding the query range. The returned points are P_{C0}, P_{C1}, P_{C2} and P_{C3}, shown in green at the top to indicate that there is no simple relationship between the calculated values and any of the actual points.

Assume the query is set so that the total contained time in the two states are returned. The timestamping is set

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

to use the cycle end time. The RoundTrip retrieval mode returns values for states that are completely contained within the cycle boundaries. The following is returned:

- For T_{C0}, the query returns two values (one for the "on" state and one for the "off" state), calculated as a result of the "phantom" cycle that preceeds the query start time. The value has a timestamp of the query start time.
- For T_{C1}, one row is returned for the "on" state, and one row is returned for the "off" state. The round-trip for the "on" state occurred one time for four seconds completely within the cycle boundary. The round-trip for the "off" state occurred one time during the cycle for five seconds.
- For T_{C2}, a round-trip did not occur for either state within the cycle boundaries. One NULL row is returned for this cycle.
- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The state was "on" for a single contained time of two seconds between the cycle boundaries. The state was "off" one time during the cycle for one second completely within the cycle boundary. An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect.
- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The state was "on" for a single contained time of three seconds between the cycle boundaries. One row is returned for the "off" state for a total contained time of seven seconds. (The first round-trip for the "off" state includes the I/O Server disconnect for a length of four seconds. The second round-trip has a length of three seconds.) An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect, and the returned value is NULL because there is no round-trip during the cycle for it. The I/O Server disconnect that "disrupted" the off state is treated as its own state, thereby changing what would have been a single "off" state instance of five seconds into two instances of the "off" state for one second each.

RoundTrip Retrieval - Supported Value Parameters

You can use various parameters to adjust the values that must be returned in the RoundTrip retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- State Calculation (wwStateCalc)
- Analog Value Filtering (wwFilter)

RoundTrip Retrieval - Query Examples

To use the RoundTrip retrieval mode, set the following parameter in your query:

```
wwRetrievalMode = ‘RoundTrip’
```

The following queries compare the results between ValueState retrieval and RoundTrip retrieval.

This first ValueState retrieval query returns the average amount of time that the 'Reactor1OutletValve' tag is in "on" state and the average amount of time it is in the "off" state for a single cycle. Any state changes that occur across the cycle boundaries are not included.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
SELECT DateTime, vValue, StateTime
   FROM History
      WHERE TagName IN ('Reactor1OutletValve')
      AND DateTime >= '2009-09-16 12:35:00'
      AND DateTime <= '2009-09-16 12:55:00'
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'AvgContained'
      AND wwCycleCount = 1
```

The results are:

DateTime	vValue	StateTime
2009-09-16 12:35:00.0000000	0	215878
2009-09-16 12:35:00.0000000	1	61729
2009-09-16 12:55:00.0000000	1	62827.5
2009-09-16 12:55:00.0000000	0	212856

The first two rows are for the "phantom" cycle leading up to the query start time and have a timestamp of the query start time.

The second two rows show the average amount of time for each state and have a timestamp of the query end time (the default).

Compare these results to same basic query that instead uses RoundTrip retrieval:

```
SELECT DateTime, vValue, StateTime
   FROM History
      WHERE TagName IN ('Reactor1OutletValve')
      AND DateTime >= '2009-09-16 12:35:00'
      AND DateTime <= '2009-09-16 12:55:00'
      AND wwRetrievalMode = 'RoundTrip'
      AND wwStateCalc = 'AvgContained'
      AND wwCycleCount = 1
```

The results are:

DateTime	vValue	StateTime
2009-09-16 12:35:00.0000000	1	277607
2009-09-16 12:35:00.0000000	0	278580
2009-09-16 12:55:00.0000000	0	275683.5
2009-09-16 12:55:00.0000000	1	273845

RoundTrip Retrieval - Initial and Final Values

The values returned at the query start time are the result of applying the algorithm to the last cycle preceding the query range.

RoundTrip Retrieval - Handling NULL Values

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Like in the ValueState retrieval mode, the NULL state is treated as a valid distinct value. This allows you to analyze round trips for disturbances.

Edge Detection for Events (wwEdgeDetection)

For AVEVA Historian, an event is the moment at which a detection criterion is met on historical tag values in the AVEVA Historian. At a basic level, anything that can be determined by looking at stored data can be used as an event.

When detecting events, it is useful to pinpoint rows in a result set where the satisfaction of criteria in a WHERE clause changed from true to false, or vice versa. For example, you may want to know when the level of a tank went above 5 feet. The WHERE clause in a query for this example might be TANKLEVEL > 5. As the tank level approaches 5 feet, the criterion does not return true. Only when the level crosses the line from not satisfying the criterion to satisfying it, does the event actually occur. This imaginary "line" where the change occurs is called the edge.

Over a period of time, there may be many instances where the criteria cross the "edge" from being satisfied to not satisfied, and vice-versa. The values on either side of this "edge" can be detected if you configure your event tag to include this information. There are four possible options for edge detection: none, leading, trailing, or both. You will get differing results based on which option you use:

Option	Results
None	Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
Leading	Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false).
Trailing	Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true).
Both	All rows satisfying both the leading and trailing conditions are returned.

Edge detection only applies to analog and discrete value detectors. Also, edge detection is handled slightly differently based on whether you are using analog tags or discrete tags.

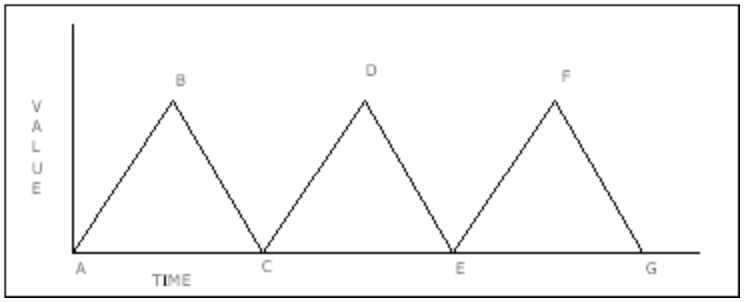
You can also use the ToDiscrete() query filter to determine when data values cross a particular threshold. For more information, see Converting Analog Values to Discrete Values (ToDiscrete).

For more information on event detection with the Classic Event subsystem, see Classic Event Subsystem in the *AVEVA Historian Administration Guide*.

Edge Detection for Analog Tags

For example, the behavior of the WHERE clause as it processes a result set can be illustrated as:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Slopes A-B, C-D and E-F are rising edges, while slopes B-C, D-E and F-G are falling edges. The slopes are affected by the WHERE clause, which is a combination of the wwEdgeDetection option and the comparison operator used against the value.

The following table describes the rows that would be returned for the various edge detection settings:

Operator	**=**	**<**	**>**	**<=**	**>=**
Leading	Falling and rising edges; first value that meets the criteria.	Falling edge only; first value to meet the criteria.*	Rising edge only; first value to meet the criteria.	Falling edge only; first value to meet the criteria. *	Rising edge only; first value to meet the criteria.
Trailing	Falling and rising edges; first value to fail the criteria after a value meets the criteria.	Rising edge only; equal to the first value to fail the criteria.	Falling edge only; first value to fail the criteria.*	Rising edge only; first value to fail the criteria.	Falling edge only; first value to fail the criteria.*

* If the falling edge is a vertical edge with no slope, the query will return the lowest value of that edge.

The following query selects all values of "SysTimeSec" that are greater than or equal to 50 from the History table between 10:00 and 10:02 a.m. on December 2, 2001. No edge detection is specified.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'None'
```

The results are:

DateTime	Value
2001-12-02 10:00:50.000	50

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-02 10:00:52.000	52
2001-12-02 10:00:54.000	54
2001-12-02 10:00:56.000	56
2001-12-02 10:00:58.000	58
2001-12-02 10:01:50.000	50
2001-12-02 10:01:52.000	52
2001-12-02 10:01:54.000	54
2001-12-02 10:01:56.000	56
2001-12-02 10:01:58.000	58

Leading Edge Detection for Analog Tags

If Leading is specified as the parameter in the edge detection time domain extension, the only rows in the result set are those that are the first to successfully meet the WHERE clause criteria (returned true) after a row did not successfully meet the WHERE clause criteria (returned false).

The following query selects the first values of "SysTimeSec" from the History table to meet the Value criterion between 10:00 and 10:02 a.m. on December 2, 2001.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'Leading'
```

The query will return only the two values that were greater than or equal to 50 for the time period specified:

DateTime	Value
2001-12-02 10:00:50.000	50
2001-12-02 10:01:50.000	50

Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags. Notice that even though the original query returned ten rows, the edge detection only returns the first row recorded after the event criteria returned true.

Trailing Edge Detection for Analog Tags

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If Trailing is specified as the parameter in the edge detection extension, the only rows in the result set are those that are the first to fail the criteria in the WHERE clause (returned false) after a row successfully met the WHERE clause criteria (returned true).

The following query selects the first values of "SysTimeSec" from the History table to fail the Value criterion between 10:00 and 10:02 a.m. on December 2, 2001.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'Trailing'
```

The query returns only the two values that were the first to fail the criteria in the WHERE clause for the time period specified:

DateTime	Value
2001-12-02 10:01:00.000	0
2001-12-02 10:02:00.000	0

Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags. Notice that even though the original query returned ten recorded rows for each value, the edge detection only returns the first row recorded after the event criteria returned false.

Both Leading and Trailing Edge Detection for Analog Tags

If Both is specified as the parameter in the edge detection extension, all rows satisfying both the leading and trailing conditions are returned.

The following query selects values of "SysTimeSec" from the History table that meet both the Leading and Trailing criteria between 10:00 and 10:02 a.m. on December 2, 2001.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'Both'
```

The results are:

DateTime	Value
2001-12-02 10:00:50.000	50

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-02 10:01:00.000	0
2001-12-02 10:01:50.000	50
2001-12-02 10:02:00.000	0

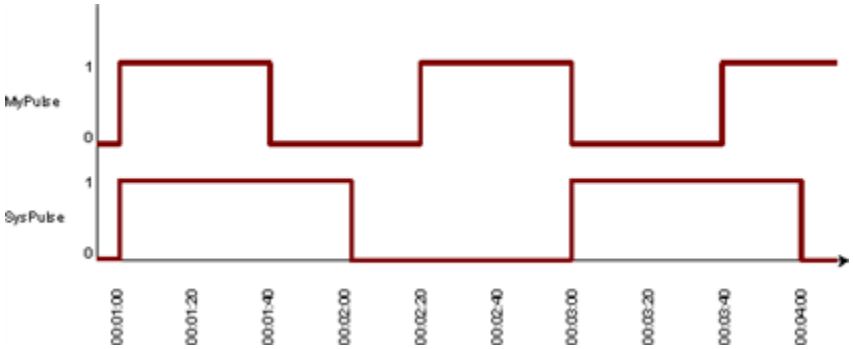
Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags. Notice that value of the first row in the original query is returned in the result set.

Edge Detection for Discrete Tags

Edge detection for discrete tags operates differently than for analog tags. For example, assume the following discrete tags are stored.

Tag	Description
SysPulse	Transitions between 1 and 0 every minute.
MyPulse	Transitions between 1 and 0 every 40 seconds.

A representation of the data stored in the system is as follows:



The following queries select values of "SysPulse" and "MyPulse" that have a value of 1 (On) from the History and WideHistory tables between 12:59 and 1:04 a.m. on December 8, 2001. No edge detection is specified.

This is a query of the History table:

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND wwRetrievalMode = 'Delta'
         AND Value = 1
         AND wwEdgeDetection = 'None'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1
2001-12-08 00:01:00.000	MyPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1
2001-12-08 00:03:40.000	MyPulse	1

The following is a query of the WideHistory table:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse, MyPulse FROM WideHistory
   WHERE DateTime > "2001-12-08 00:59:00"
      AND DateTime < "2001-12-08 01:05:00"
      AND SysPulse = 1
      AND MyPulse = 1
      AND wwRetrievalMode = "Delta"
      AND wwEdgeDetection = "None"
   ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1

Leading Edge Detection for Discrete Tags

If Leading is specified as the parameter in the edge detection time domain extension, the only rows in the result set are those that are the first to successfully meet the WHERE clause criteria (returned true) after a row did not successfully meet the WHERE clause criteria (returned false).

The following queries select values of "SysPulse" and "MyPulse" that have a value of 1 (On) from the History and WideHistory tables between 12:59 and 1:04 a.m. on December 8, 2001.

This example queries the History table, if the WHERE clause criteria specify to return only discrete values equal to 1 (On), then applying a leading edge detection does not change the result set.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND Value = 1
         AND wwEdgeDetection = 'Leading'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1
2001-12-08 00:01:00.000	MyPulse	1
2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:03:40.000	MyPulse	1

This example queries the WideHistory table, applying a leading edge detection requires that the condition only evaluate to true if both values are equal to 1 (On).

```
SELECT DateTime, SysPulse, MyPulse FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse,
MyPulse
   FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
         AND DateTime <= "2001-12-08 01:04:00"
         AND SysPulse = 1
         AND MyPulse = 1
         AND wwEdgeDetection = "Leading"
      ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1
2001-12-08 00:03:40.000	1	1

Compare these results with the same query using no edge detection, as shown in Edge Detection for Discrete Tags. If you look at the diagram, you might think that a row could be returned at 00:03:00, but because both tags change at exactly this instant, their values are not returned. In a normal process, it is unlikely that two tags would change at exactly at the same instant.

Trailing Edge Detection for Discrete Tags

If Trailing is specified as the parameter in the edge detection extension, the only rows in the result set are those that are the first to fail the criteria in the WHERE clause (returned false) after a row successfully met the WHERE clause criteria (returned true).

This example queries the History table. If the WHERE clause criteria specifies returning only discrete values equal to 1 (On), then applying a trailing edge detection is the same as reversing the WHERE clause (as if Value = 0 was applied).

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND Value = 1
         AND wwEdgeDetection = 'Trailing'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:40.000	MyPulse	1
2001-12-08 00:02:00.000	SysPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:03:00.000	MyPulse	1
2001-12-08 00:04:00.000	SysPulse	1

This example queries the WideHistory table. It applies a trailing edge detection returns the boundaries where the condition ceases to be true (one of the values is equal to 0).

```
SELECT DateTime, SysPulse, MyPulse FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse,
MyPulse
   FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
         AND DateTime <= "2001-12-08 01:04:00"
         AND SysPulse = 1
         AND MyPulse = 1
         AND wwEdgeDetection = "Trailing"
      ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:40.000	1	1
2001-12-08 00:04:00.000	1	1

Compare these results with the same query using no edge detection, as shown in Edge Detection for Discrete Tags. If you look at the diagram, you might think that a row could be returned at 00:03:00, but because both tags change at exactly this instant, their values are not returned. In a normal process, it is unlikely that two tags would change at exactly at the same instant.

Both Leading and Trailing Edge Detection for Discrete Tags

If Both is specified as the parameter in the edge detection extension, all rows satisfying both the leading and trailing conditions are returned.

The following queries select values of "SysPulse" and "MyPulse" that meet an edge detection of Both for a value criterion of 1 (On) from the History and WideHistory tables between 12:59 and 1:04 a.m. on December 8, 2001.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND Value = 1
         AND wwEdgeDetection = 'Both'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:01:00.000	MyPulse	1
2001-12-08 00:01:40.000	MyPulse	1
2001-12-08 00:02:00.000	SysPulse	1
2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1
2001-12-08 00:03:00.000	MyPulse	1
2001-12-08 00:03:40.000	MyPulse	1
2001-12-08 00:04:00.000	SysPulse	1

```
SELECT DateTime, SysPulse, MyPulse FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse,
MyPulse
   FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
         AND DateTime <= "2001-12-08 01:04:00"
         AND SysPulse = 1
         AND MyPulse = 1
         AND wwEdgeDetection = "Both"
      ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1
2001-12-08 00:01:40.000	1	1
2001-12-08 00:03:40.000	1	1
2001-12-08 00:04:00.000	1	1

Compare these results with the same query using no edge detection, as shown in the Edge Detection for Discrete Tags.

Predictive Filter

AVEVA Historian supports predictive retrieval. Beginning with AVEVA Historian 2014 R2 Patch 01, the historian can return predictive data based on a "simple linear regression" (SLR) algorithm. More capabilities will be added in future releases.

With AVEVA Historian, you can create a query based on data you have stored to predict additional values in a trend. Historian returns predictive data based on a "simple linear regression" (SLR) algorithm.

For example, based on your currently stored values, you could use the predictive retrieval feature to help predict if a certain production target will be met by the end of the shift. Or, if the Historian loses communication with the data source, you could use predictive retrieval to determine whether and when a tank is likely to become empty.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

You can predict:

- Values in between other values.
- Values that extend beyond stored values.

For example, suppose you already captured data for a tag with timestamps up to 3 p.m. on a certain day, but not for the rest of the shift, which ran until 5 p.m., because of a power cut. With predictive retrieval, you can view the interpolated results based between 3 p.m. and 5 p.m. These results are based on the data you received through 3 p.m.

The following is an example of a query that retrieves stored values and reports both those values and additional predictive data:

```
SELECT DateTime, Value, wwFilter
FROM History
   WHERE TagName = 'Tag1'
      AND DateTime >= '2014-01-01 0:00:00.000'
      and DateTime < '2014-01-01 1:00:00.000'
      and wwFilter = 'SLR()'
```

In this example, "SLR" stands for "simple linear regression," the algorithm used by AVEVA Historian to analyze currently stored values and predict other values within the detected trend.

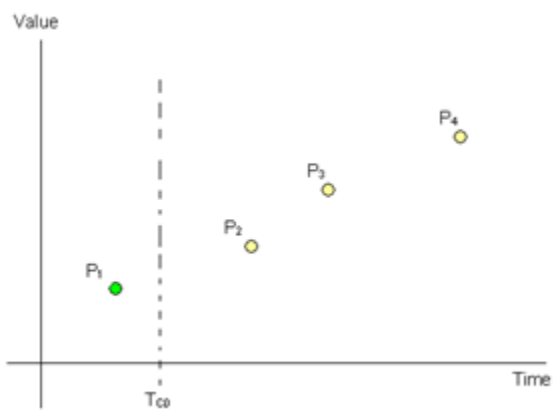
Bounding Value Retrieval

The bounding value retrieval mode returns either the start bound point or the end bound point for a requested point in time. For a start bound point, Historian retrieves the first value on or before the requested date/time. For an end bound point, Historian retrieves the first value after the requested date/time.

If no time is specified, Historian returns the bounding point at the current time.

Bounding Value Retrieval - How It Works

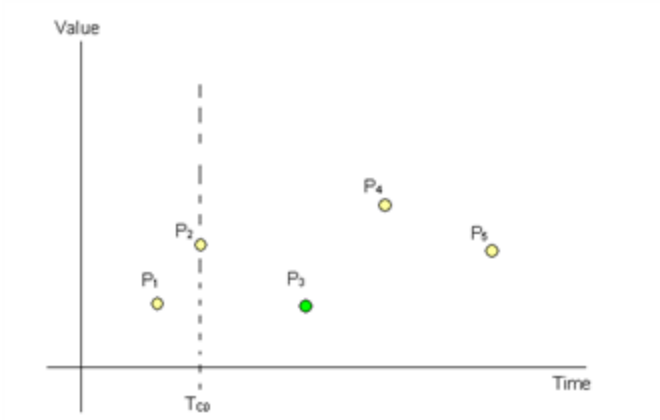
The following illustration shows how bounding value retrieval returns a start bound point:



In this case, Historian retrieves the first point on or before the datetime requested in the query. The line (T_{C0}) is the timestamp for which the start bound point is requested. P_1 is returned because that is the start or first point for the query date time.

Historian can also use bounding value retrieval to return an end bound point, as in the following illustration:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In this case, Historian returns first point after the datetime requested in the query. T_{C0} is the timestamp for which the end bound point is requested and P_3 is returned as the ending bound point because this is the first point after the query date time.

Bounding Value Retrieval - Query Examples

You can use the Bounding Value retrieval mode to return a start bound point or an end bound point for a specified date and time. If no time is specified, Historian returns the bounding point at the current time.

To return a start bound point, set the following parameter in your query.

```
wwRetrievalMode = 'StartBound'
```

To return an end bound point, set the following parameter in your query.

```
wwRetrievalMode = 'EndBound'
```

Example 1 - Retrieve start bound point

```
select DateTime,TagName,Value
where TagName 'Plant2.R31.BatchNum'
and wwRetrievalMode = 'StartBound'
and DateTime >= '2019-04-24 12:00:00'
```

The results are:

DateTime	TagName	Value
2019-04-24 11:53:08.5430000	Plant2.R31.BatchNum	912

Example 2 - Retrieve end bound point

```
select DateTime,TagName,Value
where TagName in 'Plant2.R31.BatchNum'
and wwRetrievalMode = 'EndBound'
and DateTime >= '2019-04-24 12:00:00'
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value
2019-04-24 14:11:13.3840000	Plant2.R31.BatchNum	926

Understanding Retrieval Options

In all retrieval modes, you can adjust which values the historian returns by specifying retrieval options. The retrieval options are implemented as special parameters that you set as part of the retrieval query. This section explains each of these options. For an overview of which options apply to which retrieval modes, see Which Options Apply to Which Retrieval Modes?.

Which Options Apply to Which Retrieval Modes?

The following table shows which retrieval options apply to which modes. If you specify an option in a mode where it isn't applicable, the option is ignored.

	wwCycleCount	wwResolution	wwTimeDeadband	wwValueDeadband	wwVersion	wwInterpolationType	wwTimeStampRule	wwQualityRule	wwStateCalc	wwFilter	wwValueSelector	Predictive Filter	wwExpression	Bounding
Cyclic	Y	Y			Y		Y	Y		Y	Y		Y	
Daily	Y				Y								Y	
Delta			Y	Y	Y			Y		Y	Y	Y	Y	Y
Full					Y			Y		Y	Y	Y	Y	Y
Interpolated	Y	Y			Y	Y	Y	Y		Y	Y			
Best Fit	Y	Y			Y	Y		Y		Y	Y	Y	Y	
Average	Y	Y			Y	Y	Y	Y		Y	Y			
Minimum	Y	Y			Y			Y		Y	Y			
Maximum	Y	Y			Y			Y		Y	Y			
Integ	Y	Y			Y	Y	Y	Y		Y	Y			

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ral														
Slope					Y			Y		Y	Y			
Counter	Y	Y			Y		Y	Y		Y				
Value State	Y	Y			Y		Y	Y	Y	Y				
RoundTrip	Y	Y			Y		Y	Y	Y	Y				
Bounding	Y	Y												

Using Retrieval Options in a Transact-SQL Statement

You can retrieve data in the AVEVA Historian extension tables using normal Transact-SQL code, as well as the specialized SQL time domain extensions provided by the AVEVA Historian. The AVEVA Historian extensions provide an easy way to query time-based data from the history tables. They also provide additional functionality not supported by Transact-SQL.

Note: The wwParameters extension is reserved for future use. The wwRowCount parameter is still supported, but is deprecated in favor of wwCycleCount.

The extensions are implemented as "virtual" columns in the extension tables. When you query an extension table, you can specify values for these column parameters to manipulate the data that will be returned. You will need to specify any real-time extension parameters each time that you execute the query.

For example, you could specify a value for the wwResolution column in the query so that a resolution is applied to the returned data set:

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND Value >= 50
         AND wwResolution = 10
         AND wwRetrievalMode = 'cyclic'
```

Because the extension tables provide additional functionality that is not possible in a normal SQL Server, certain limitations apply to the Transact-SQL supported by these tables. For more information, see Unsupported or Limited Syntax Options.

Although the Microsoft SQL Server may be configured to be case-sensitive, the values for the virtual columns in the extension tables are always case-insensitive.

Note: You cannot use the IN clause or OR clause to specify more than one condition for a time domain extension. For example, "`wwVersion IN ('original', 'latest')`" and "`wwRetrievalMode = 'Delta' OR wwVersion = 'latest'`" are not supported.

For general information on creating SQL queries, see your Microsoft SQL Server documentation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)

In retrieval modes that use cycles, the cycle count determines the number of cycles for which data is retrieved. The number of actual return values is not always identical with this cycle count. In "truly cyclic" modes (Cyclic, Interpolated, Average, and Integral), a single data point is returned for every cycle boundary. However, in other cycle-based modes (Best Fit, Minimum, Maximum, Counter, ValueState, and RoundTrip), multiple or no data points may be returned for a cycle, depending on the nature of the data.

The length of each cycle (the "resolution" of the returned values) is calculated as follows:

$$D_C = D_Q / (n - 1)$$

Where D_C is the length of the cycle, D_Q is the duration of the query, and n is the cycle count.

Instead of specifying a cycle count, you can specify the resolution. In that case, the cycle count is calculated based on the resolution and the query duration. For more information, see Resolution (Values Spaced Every X ms) (wwResolution).

This option is relevant in the following retrieval modes:

- Cyclic Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

The application of the cycle count also depends on whether you are querying a wide table. If you are querying the History table, the cycle count determines the number of rows returned per tag. For example, a query that applies a cycle count of 20 to two tags will return 40 rows of data (20 rows for each tag). If you are querying a WideHistory table, the cycle count specifies the total number of rows to be returned, regardless of how many tags were queried. For example, a query that applies a cycle count of 20 to two tags returns 20 rows of data.

Values chosen:

- If wwResolution and wwCycleCount are not specified, then a default of 100 cycles are chosen.
- If wwResolution and wwCycleCount are set to 0, then a default of 100000 cycles are chosen.
- If wwResolution and wwCycleCount are both set, then wwCycleCount is ignored.
- If wwCycleCount is specified and is less than 0, then a default of 100 cycles are chosen.
- For ValueState retrieval, if the start time of the cycle is excluded, no states are returned for the first cycle.
- For ValueState retrieval, if the end time of the cycle is excluded, no states are returned for the last cycle.

Cycle Count - Query Examples

See the following examples of queries that demonstrate the cycle count behavior if applied to a single tag or to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

multiple tags in the same query:

- Cycle Count - Query 1: Using a Single Tag
- Cycle Count - Query 2: Using Multiple Tags

Cycle Count - Query 1: Using a Single Tag

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime < '2001-12-09 11:36'
         AND wwRetrievalMode = 'Cyclic'
         AND wwCycleCount = 300
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.200	SysTimeSec	0
2001-12-09 11:35:00.400	SysTimeSec	0
2001-12-09 11:35:00.600	SysTimeSec	0
…		
2001-12-09 11:35:59.200	SysTimeSec	59
2001-12-09 11:35:59.400	SysTimeSec	59
2001-12-09 11:35:59.600	SysTimeSec	59
2001-12-09 11:35:59.800	SysTimeSec	59

Cycle Count - Query 2: Using Multiple Tags

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeMin','SysTimeSec')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime < '2001-12-09 11:36'
         AND wwRetrievalMode = 'Cyclic'
         AND wwCycleCount = 300
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.200	SysTimeMin	35
2001-12-09 11:35:00.200	SysTimeSec	0
2001-12-09 11:35:00.400	SysTimeMin	35
2001-12-09 11:35:00.400	SysTimeSec	0
2001-12-09 11:35:00.600	SysTimeMin	35
2001-12-09 11:35:00.600	SysTimeSec	0
...		
2001-12-09 11:35:59.200	SysTimeMin	35
2001-12-09 11:35:59.200	SysTimeSec	59
2001-12-09 11:35:59.400	SysTimeMin	35
2001-12-09 11:35:59.400	SysTimeSec	59
2001-12-09 11:35:59.600	SysTimeMin	35
2001-12-09 11:35:59.600	SysTimeSec	59
2001-12-09 11:35:59.800	SysTimeMin	35
2001-12-09 11:35:59.800	SysTimeSec	59

Notice that the values of the two tags are mixed together in the same column.

Resolution (Values Spaced Every X ms) (wwResolution)

In retrieval modes that use cycles, the resolution is the sampling interval for retrieving data, that is, the length of each cycle.

The number of cycles, therefore, depends on the time period and the resolution:

number of cycles = time period / resolution

The number of actual return values is not always identical with this cycle count. In "truly cyclic" modes (Cyclic, Interpolated, Average, and Integral), a single data point is returned for every cycle boundary. However, in other cycle-based modes (Best Fit, Minimum, Maximum, Counter, and ValueState), multiple or no data points may be returned for a cycle, depending on the nature of the data.

Note: The rowset is guaranteed to contain one row for each tag in the normalized query at every resolution interval, regardless of whether a physical row exists in history at that particular instance in time. The value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

contained in the row is the last known physical value in history, at that instant, for the relevant tag.

Instead of specifying a resolution, you can specify the cycle count directly. In that case, the resolution is calculated based on the cycle count and the query duration. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).

This option is relevant in the following retrieval modes:

- Cyclic Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

For delta retrieval, you can achieve similar results by using a time deadband. For more information, see Time Deadband (wwTimeDeadband).

Resolution - Query Example

The following query returns rows that are spaced at 2 sec (2000 msec) intervals over the requested time period. Data is retrieved cyclically.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeMin','SysTimeSec')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:02.000	SysTimeMin	35
2001-12-09 11:35:02.000	SysTimeSec	2
2001-12-09 11:35:04.000	SysTimeMin	35
2001-12-09 11:35:04.000	SysTimeSec	4

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:06.000	SysTimeMin	35
...		
2001-12-09 11:35:54.000	SysTimeMin	35
2001-12-09 11:35:54.000	SysTimeSec	54
2001-12-09 11:35:56.000	SysTimeMin	35
2001-12-09 11:35:56.000	SysTimeSec	56
2001-12-09 11:35:58.000	SysTimeMin	35
2001-12-09 11:35:58.000	SysTimeSec	58
2001-12-09 11:36:00.000	SysTimeMin	36
2001-12-09 11:36:00.000	SysTimeSec	0

About Phantom Cycles

The phantom cycle is the name given to the cycle that leads up to the query start time. It is used to calculate which initial value to return at the query start time for all retrieval modes. Some retrieval modes use the phantom cycle to simply find the last known value prior to the query start time, whereas other retrieval modes use the entire cycle to calculate aggregates. The different uses of the phantom cycle can be seen in the following table.

Simple use of phantom cycle	Cycles not defined, but similar simple use of time before query start time	Phantom cycle used to calculate aggregates
Cyclic	Delta	Min
Interpolated	Full	Max
Best Fit	Slope	Average
		Integral
		Counter
		ValueState
		RoundTrip

It's common to expect a single aggregate row returned for a particular time interval. You can accomplish this in several ways.

The following example is querying for hourly averages. It returns a single row time stamped at the query start time. If the query included the query end point by including an equal sign for it, the query would also have

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

returned an additional row at the query end time.

```
SELECT DateTime, Value, Quality, QualityDetail, OPCQuality
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime >= '2017-12-12 08:00:00'
         AND DateTime < '2017-12-12 09:00:00'
         AND wwRetrievalMode = 'Avg'
         AND wwResolution = 3600000
```

The results are:

DateTime	Value	Quality	QualityDetail	OPCQuality
2017-12-1208:00:00.0000000	29.5	0	192	192

What may be confusing in this example is the calculation of the average in the returned row for the phantom cycle leading up to the query start time. The query specifies a positive one hour time interval between the query start time and the query end time. You may therefore expect that the calculated and returned average should be for the specified interval.

However, the time difference between start and end time in the above query is not actually required because the resolution is provided explicitly (wwResolution = 36000000). If the query specified an end time equal to the specified start time and if it included the equal sign for the end time, the query would still return the same single row of data.

```
SELECT DateTime, Value, Quality, QualityDetail, OPCQuality
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime = '2017-10-16 08:00:00'
         AND DateTime <= '2017-10-16 09:00:00'
         AND wwRetrievalMode = 'Avg'
         AND wwCycleCount = 1
```

The results are:

DateTime	Value	Quality	QualityDetail	OPCQuality
2017-10-16 08:00:00.0000000	29.5	0	192	192

This second example also asks for hourly averages and it also returns only a single row of data stamped at the query start time. This query, however, must specify a time difference between the start and end time, because the resolution is not explicitly defined in the query.

As in the preceding query, the specified interval and cycle count of 1 may look like the returned row has been calculated for the specified interval, but the returned row is once again for the phantom cycle leading up to the start time.

For some queries, you may want to be certain to include values on a cycle boundary. For example, the following query is looking for a minimum value within a cycle. In this query, the beginning DateTime statement uses ">=" to ensure that the entire cycle is queried. Even if the minimum value happens to be at the beginning of the cycle, the following query will provide an accurate result:

```
   SELECT StartDateTime, *
      FROM History
         WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2016-03-31 15:41:10'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND DateTime < '2016-03-31 15:41:20'
        AND wwRetrievalMode = 'Min'
        AND Quality <> 133
        AND wwCycleCount = 1
```

The StartDateTime makes it easier to see which time interval was used to calculate the returned aggregate. This column returns the time stamp of the beginning of the cycle used for the aggregate calculation. The time stamp is always returned in accordance with the specified time zone and always has the same offset as the time stamp returned in the DateTime column, even when the two time stamps are on different sides of a DST change.

Assuming results are timestamped at the end of the cycle (as is done by default when wwTimeStampRule is set to END), the initial rows in the examples above would return a DateTime equal to '2009-10-16 08:00:00', and the StartDateTime column would return '2009-10-16 07:00:00' making it easy to interpret the result.

If instead the query were to ask for results time stamped at the beginning of the cycle with wwTimeStampRule set to START, the initial rows in the same examples would still return a DateTime equal to '2009-10-16 08:00:00', but the time stamp has now been shifted in accordance with the time stamp request. The result is therefore calculated for the specified time interval between 8 a.m. and 9 a.m. In this example, the new StartDateTime column would return the same time stamp as the DateTime column, '2009-10-16 08:00:00', again making it easier to interpret the result.

For retrieval modes for which cycles are defined, the StartDateTime column returns the cycle start time. These modes are:

- Cyclic Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

In the remaining retrieval modes, the StartDateTime column returns the same time stamp as the DateTime column.

For an additional example, see Querying Aggregate Data in Different Ways.

Value Deadband (wwValueDeadband)

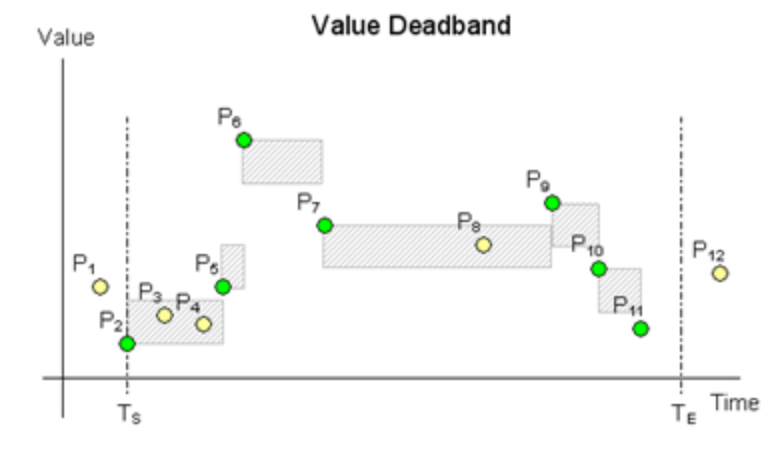
A value deadband for retrieval controls the value resolution of data returned in delta mode. Any data values that change less than the specified deadband are not returned. The deadband is a percentage of a tag's full scale in engineering units.

The deadband "base value" is reset each time that a value is returned, so that the last value returned acts as the basis for the deadband.

Changes in quality will force a value to be returned even if the value deadband has not been met.

The following illustration shows an example of applying a value deadband:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Data is retrieved for the time period starting with T_S and ending with T_E. All points in the graphic represent data values stored on the historian. The gray areas represent the value deadband, which starts anew with every returned value. Only the green points (P_2, P_5, P_6, P_7, P_9, P_{10}, P_{11}) are returned. The other points are not returned because they fall within a deadband.

Value Deadband - Query Examples

See the following examples of queries that use value deadband:

- Value Deadband - Query 1
- Value Deadband - Query 2

Note: Each of these examples returns data values for the analog tag 'SysTimeSec'. The minimum engineering unit for 'SysTimeSec' is 0, and the maximum engineering unit is 59.

Value Deadband - Query 1

This query specifies to return only data that changed by more than 10 percent of the tag's full engineering unit range. Using a value deadband of 10 percent equates to an absolute change of 5.9 for 'SysTimeSec.'

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwValueDeadband = 10
```

The results are:

DateTime	Value
2001-12-09 11:35:00.000	0
2001-12-09 11:35:06.000	6

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:12.000	12
2001-12-09 11:35:18.000	18
2001-12-09 11:35:24.000	24
2001-12-09 11:35:30.000	30
2001-12-09 11:35:36.000	36
2001-12-09 11:35:42.000	42
2001-12-09 11:35:48.000	48
2001-12-09 11:35:54.000	54
2001-12-09 11:36:00.000	0
2001-12-09 11:36:06.000	6
2001-12-09 11:36:12.000	12
2001-12-09 11:36:18.000	18
2001-12-09 11:36:24.000	24
2001-12-09 11:36:30.000	30
2001-12-09 11:36:36.000	36
2001-12-09 11:36:42.000	42
2001-12-09 11:36:48.000	48
2001-12-09 11:36:54.000	54
2001-12-09 11:37:00.000	0

Value Deadband - Query 2

This query specifies to only return data that changed by more than 5 percent of the tag's full engineering unit range. Using a value deadband of 5 percent equates to an absolute change of 2.95 for 'SysTimeSec.'

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwValueDeadband = 5
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	Value
2001-12-09 11:35:00.000	0
2001-12-09 11:35:03.000	3
2001-12-09 11:35:06.000	6
2001-12-09 11:35:09.000	9
2001-12-09 11:35:12.000	12
2001-12-09 11:35:15.000	15
2001-12-09 11:35:18.000	18
2001-12-09 11:35:21.000	21
2001-12-09 11:35:24.000	24
2001-12-09 11:35:27.000	27
2001-12-09 11:35:30.000	30
2001-12-09 11:35:33.000	33
2001-12-09 11:35:36.000	36
2001-12-09 11:35:39.000	39
2001-12-09 11:35:42.000	42
2001-12-09 11:35:45.000	45
2001-12-09 11:35:48.000	48
2001-12-09 11:35:51.000	51
2001-12-09 11:35:54.000	54
2001-12-09 11:35:57.000	57
2001-12-09 11:36:00.000	0
2001-12-09 11:36:03.000	3
2001-12-09 11:36:06.000	6
2001-12-09 11:36:09.000	9
2001-12-09 11:36:12.000	12
2001-12-09 11:36:15.000	15

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:36:18.000	18
2001-12-09 11:36:21.000	21
2001-12-09 11:36:24.000	24
2001-12-09 11:36:27.000	27
2001-12-09 11:36:30.000	30
2001-12-09 11:36:33.000	33
2001-12-09 11:36:36.000	36
2001-12-09 11:36:39.000	39
2001-12-09 11:36:42.000	42
2001-12-09 11:36:45.000	45
2001-12-09 11:36:48.000	48
2001-12-09 11:36:51.000	51
2001-12-09 11:36:54.000	54
2001-12-09 11:36:57.000	57
2001-12-09 11:37:00.000	0

Time Deadband (wwTimeDeadband)

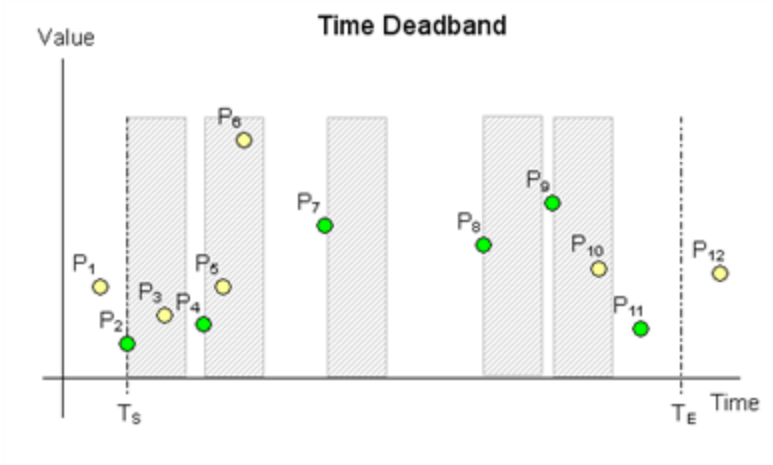
A time deadband for retrieval controls the time resolution of data returned in delta mode. Any value changes that occur within the time deadband are not returned.

Time deadbands can be applied to analog, discrete, and string tags.

The deadband "base value" is reset each time that a value is returned, so that the last value returned acts as the basis for the deadband.

The following illustration shows an example of applying a time deadband:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Data is retrieved for the time period starting with T_S and ending with T_E. All points in the graphic represent data values stored on the historian. The gray areas represent the time deadband, which starts anew with every returned value. Only the green points (P_2, P_4, P_7, P_8, P_9, P_{11}) are returned. The other points are not returned because they fall within a deadband.

Time Deadband - Query Examples

To apply a time deadband, set the wwTimeDeadband parameter in your query.

For examples, see the following:

- Time Deadband - Query 1
- Time Deadband - Query 2
- Time Deadband - Query 3

Note: All of these example queries return data values for the analog tag 'SysTimeSec'.

Time Deadband - Query 1

This query specifies to only return data that changed during a 5 second time deadband.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 5000
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:06.000	SysTimeSec	6

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:12.000	SysTimeSec	12
2001-12-09 11:35:18.000	SysTimeSec	18
2001-12-09 11:35:24.000	SysTimeSec	24
2001-12-09 11:35:30.000	SysTimeSec	30
2001-12-09 11:35:36.000	SysTimeSec	36
2001-12-09 11:35:42.000	SysTimeSec	42
2001-12-09 11:35:48.000	SysTimeSec	48
2001-12-09 11:35:54.000	SysTimeSec	54
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:06.000	SysTimeSec	6
2001-12-09 11:36:12.000	SysTimeSec	12
2001-12-09 11:36:18.000	SysTimeSec	18
2001-12-09 11:36:24.000	SysTimeSec	24
2001-12-09 11:36:30.000	SysTimeSec	30
2001-12-09 11:36:36.000	SysTimeSec	36
2001-12-09 11:36:42.000	SysTimeSec	42
2001-12-09 11:36:48.000	SysTimeSec	48
2001-12-09 11:36:54.000	SysTimeSec	54
2001-12-09 11:37:00.000	SysTimeSec	0

Time Deadband - Query 2

This query specifies to only return data that changed during a 4900 millisecond time deadband.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 4900
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:05.000	SysTimeSec	5
2001-12-09 11:35:10.000	SysTimeSec	10
2001-12-09 11:35:15.000	SysTimeSec	15
2001-12-09 11:35:20.000	SysTimeSec	20
2001-12-09 11:35:25.000	SysTimeSec	25
2001-12-09 11:35:30.000	SysTimeSec	30
2001-12-09 11:35:35.000	SysTimeSec	35
2001-12-09 11:35:40.000	SysTimeSec	40
2001-12-09 11:35:45.000	SysTimeSec	45
2001-12-09 11:35:50.000	SysTimeSec	50
2001-12-09 11:35:55.000	SysTimeSec	55
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:05.000	SysTimeSec	5
2001-12-09 11:36:10.000	SysTimeSec	10
2001-12-09 11:36:15.000	SysTimeSec	15
2001-12-09 11:36:20.000	SysTimeSec	20
2001-12-09 11:36:25.000	SysTimeSec	25
2001-12-09 11:36:30.000	SysTimeSec	30
2001-12-09 11:36:35.000	SysTimeSec	35
2001-12-09 11:36:40.000	SysTimeSec	40
2001-12-09 11:36:45.000	SysTimeSec	45
2001-12-09 11:36:50.000	SysTimeSec	50
2001-12-09 11:36:55.000	SysTimeSec	55
2001-12-09 11:37:00.000	SysTimeSec	0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time Deadband - Query 3

This query specifies to only return data that changed during a 2000 millisecond time deadband.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeSec','SysTimeMin')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 2000
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:03.000	SysTimeSec	3
2001-12-09 11:35:06.000	SysTimeSec	6
2001-12-09 11:35:09.000	SysTimeSec	9
2001-12-09 11:35:12.000	SysTimeSec	12
2001-12-09 11:35:15.000	SysTimeSec	15
2001-12-09 11:35:18.000	SysTimeSec	18
2001-12-09 11:35:21.000	SysTimeSec	21
2001-12-09 11:35:24.000	SysTimeSec	24
2001-12-09 11:35:27.000	SysTimeSec	27
2001-12-09 11:35:30.000	SysTimeSec	30
2001-12-09 11:35:33.000	SysTimeSec	33
2001-12-09 11:35:36.000	SysTimeSec	36
2001-12-09 11:35:39.000	SysTimeSec	39
2001-12-09 11:35:42.000	SysTimeSec	42
2001-12-09 11:35:45.000	SysTimeSec	45
2001-12-09 11:35:48.000	SysTimeSec	48
2001-12-09 11:35:51.000	SysTimeSec	51

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:54.000	SysTimeSec	54
2001-12-09 11:35:57.000	SysTimeSec	57
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:00.000	SysTimeMin	36

History Version (wwVersion)

The AVEVA Historian allows you to overwrite a stored tag value with later versions of the value. The original version of the value is still maintained, so that effectively, multiple versions of the tag value exist at the same point in time.

When retrieving data, you can specify whether to retrieve the originally stored version or the latest version that is available. To do this, set the history version option to "Original" for the original version or "Latest" for the latest available version. If you do not specify the version, the latest version is returned.

To distinguish between a latest value and an original value, the historian returns a special QualityDetail value of 202 for a latest point with good quality. Note, however, that replicated data may show a QualityDetail of 202 for points other than the latest point.

This option is relevant in all retrieval modes.

History Version - Query Example

This example illustrates using history version. First, consider this query:

```
SELECT TagName, DateTime, Value, wwVersion
   FROM History
      WHERE TagName IN ('SysTimeHour', 'SysTimeMin')
         AND DateTime >= '2001-12-20 0:00'
         AND DateTime <= '2001-12-20 0:05'
         AND wwRetrievalMode = 'Delta'
         AND wwVersion = 'Original'
```

The results are:

TagName	DateTime	Value	wwVersion
SysTimeMin	2001-12-20 00:00:00.000	0	ORIGINAL
SysTimeHour	2001-12-20 00:00:00.000	0	ORIGINAL
SysTimeMin	2001-12-20 00:01:00.000	1	ORIGINAL
SysTimeMin	2001-12-20 00:02:00.000	2	ORIGINAL
SysTimeMin	2001-12-20	3	ORIGINAL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	00:03:00.000		
SysTimeMin	2001-12-20 00:04:00.000	4	ORIGINAL
SysTimeMin	2001-12-20 00:05:00.000	5	ORIGINAL

When retrieving the latest version, the wwVersion parameter always returns with a value of LATEST for all values, even though many of the values may actually be the original values that came from the I/O Server. To distinguish between an actual latest value and an original value, a special QualityDetail of 202 is returned for a good, latest point. Note, however, that replicated data may show a QualityDetail of 202 for points other than the latest point.

For example:

```
SELECT DateTime, Value, Quality, QualityDetail, OPCQuality, wwVersion
   FROM History
      WHERE TagName IN ('PV')
         AND DateTime >= '2005-04-17 11:35:00'
         AND DateTime <= '2005-04-17 11:36:00'
         AND wwRetrievalMode = 'Delta'
         AND wwVersion = 'Latest'
```

The results are:

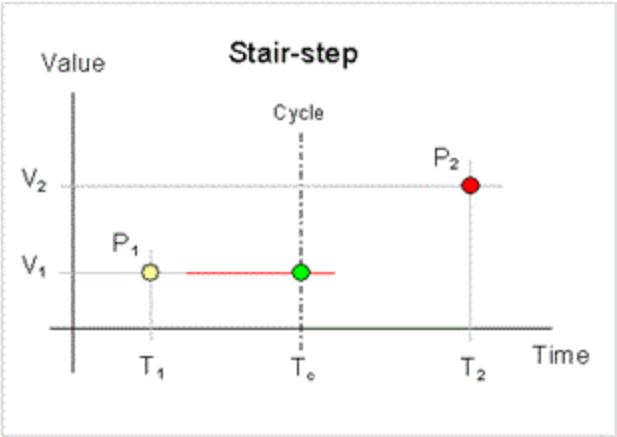
DateTime	Value	Quality	QualityDetail	OPCQuality	wwVersion
2005-04-17 11:35:00.000	12.5	0	192	192	LATEST
2005-04-17 11:35:15.000	17.3	0	192	192	LATEST
2005-04-17 11:35:30.000	34.0	0	202	192	LATEST
2005-04-17 11:35:45.000	43.1	0	192	192	LATEST
2005-04-17 11:36:00.000	51.2	0	192	192	LATEST

Interpolation Type (wwInterpolationType)

For various retrieval modes, you can control how analog tag values at cycle boundaries are calculated if there is no actual value stored at that point in time. The options are as follows:

- **Stairstep:** No interpolation occurs. The value at the cycle boundary is assumed to be the same value as the last stored value before the cycle boundary.The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned.

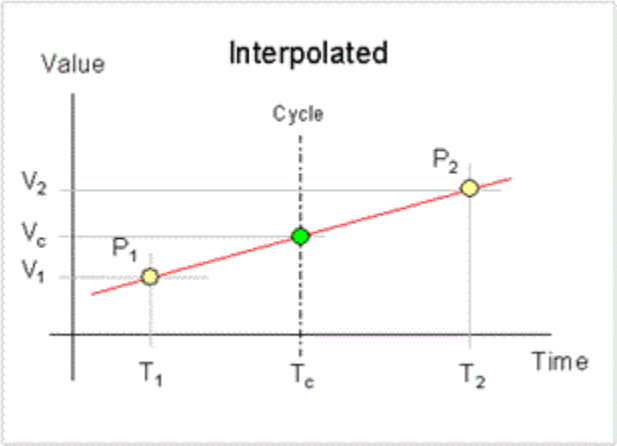
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- **Linear:** The historian calculates a new value at the cycle boundary by interpolating between the last stored value before the boundary and the first stored value after the boundary. If either of these values is NULL, it returns the last stored value before the boundary.

  Expressed as a formula, V_c is calculated as:

  $V_c = V_1 + ((V_2 - V_1) * ((T_c - T_1) / (T_2 - T_1)))$



The type of data that you want to retrieve usually determines the interpolation type to use. For example, if you have a thermocouple, the temperature change is linear, so it's best to use linear interpolation. If you have a tag that contains discrete measurements, such as a set point, then you probably want to use stair-stepped values. In general, it is recommended that you use linear interpolation as the general setting, and use stair-stepped values for the exceptions.

This option is relevant in the following retrieval modes:

- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Integral Retrieval

The quality of an interpolated point is determined by the wwQualityRule setting. For more information, see Quality Rule (wwQualityRule).

The interpolation type can be set on three levels:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- The AVEVA Historian system-wide setting. The system-wide setting must be either stair-step or interpolated. For more information, see "System Parameters" on page 36. This setting is configured using the AVEVA Historian Configuration Editor.
- The individual analog tag setting. You can configure an individual analog tag to use the system-wide setting or either stair-stepped values or linear interpolation. The individual tag setting will override the system-wide setting. This setting is configured using the AVEVA Historian Configuration Editor.
- The setting for the wwInterpolationType parameter in the query. This setting overrides any other setting for all tags in the query.

The wwInterpolationType parameter is dynamically used both for input for the query, when you need to override the individual tag settings, and for output for each individual row to show whether a particular row value was calculated using linear interpolation (returned as "LINEAR") or if it is a stair-stepped value (returned as "STAIRSTEP").

To force a query to always use linear interpolation whenever applicable, specify the following in the query:

```
AND wwInterpolationType = 'Linear'
```

To force a query to always return stair-stepped values, specify the following in the query:

```
AND wwInterpolationType = 'StairStep'
```

TimeStamp Rule (wwTimeStampRule)

For various cycle-based retrieval modes, you can control whether the returned values are timestamped at the beginning or at the end of each cycle.

To force a query to timestamp results at the start of a cycle, specify the following in the query:

```
AND wwTimeStampRule = 'Start'
```

To force a query to timestamp results at the end of a cycle, specify the following in the query:

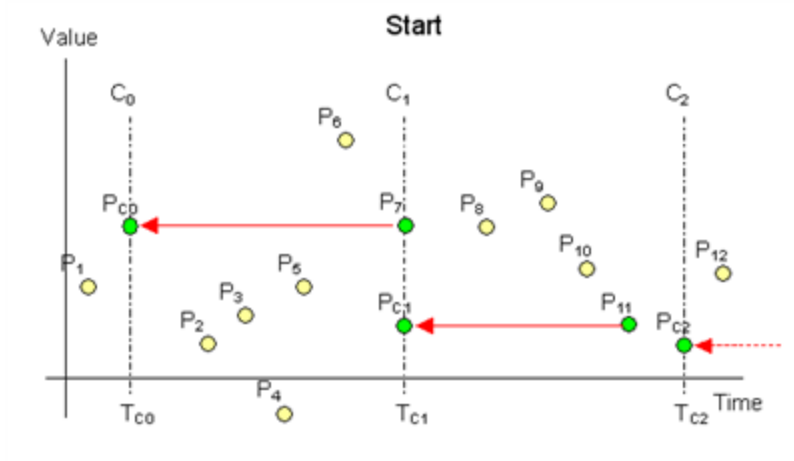
```
AND wwTimeStampRule = 'End'
```

If you include the wwTimeStampRule column in your SELECT statement, it will show which timestamp rule has been applied for the individual row, if applicable.

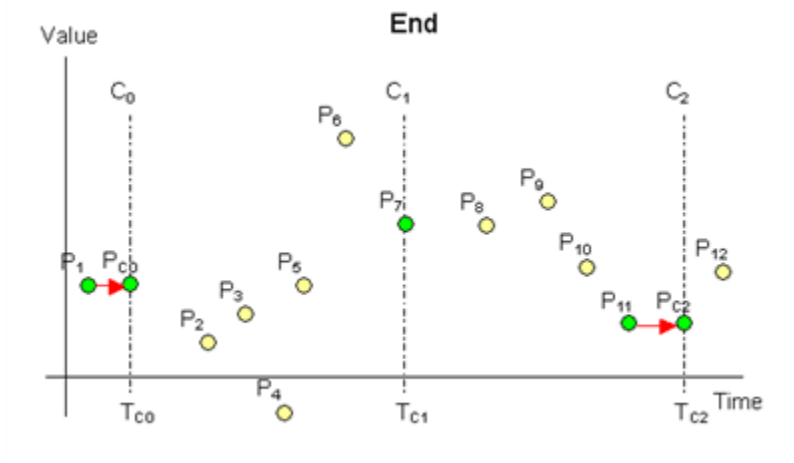
The options are as follows:

- **Start:** The value for a given cycle is stamped with the cycle start time. For example, in the following illustration of a cyclic query, the following values are returned at the cycle boundaries:
  - At T_{C0}: P_7, because it falls on the cycle boundary. In cyclic mode, if there is a value right on the cycle boundary, it is considered to belong to the cycle before the boundary. In this case, this is the cycle starting at T_{C0} and ending at T_{C1}, and because the Start timestamp rule is used, the value is timestamped at T_{C0}.
  - At T_{C1}: P_{11}, because it is the last value in the cycle starting at T_{C1} and ending at T_{C2}
  - At T_{C2}: The last value in the "phantom" cycle starting at T_{C2}

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- **End:** The value for a given cycle is stamped with the cycle end time. For example, in the following illustration of a cyclic query, the following values are returned at the cycle boundaries:
  - At T_{C0}: P_1, because it is the last value in the "phantom" cycle ending at T_{C0}. Because the End timestamp rule is used, the value is timestamped at T_{C0}.
  - At T_{C1}: P_7, because it falls on the cycle boundary. In cyclic mode, if there is a value right on the cycle boundary, it is considered to belong to the cycle before the boundary. In this case, this is the cycle starting at T_{C0} and ending at T_{C1}, and because the End timestamp rule is used, the value is timestamped at T_{C1}.
  - At T_{C2}: P_{11}, because it is the last value in the cycle ending at T_{C2}



- **Server default:** Either Start or End is used, depending on the system parameter setting on the AVEVA Historian.

This option is relevant in the following retrieval modes:

- Cyclic Retrieval (AVEVA Historian 9.0 and later)
- Interpolated Retrieval
- Average Retrieval
- Integral Retrieval
- Counter Retrieval

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- ValueState Retrieval
- RoundTrip Retrieval

Time Zone (wwTimeZone)

For AVEVA Historianversion 8.0 and later, all history data is stored in Coordinated Universal Time (UTC). The wwTimeZone extension allows you to specify the time zone to be used for the timestamps of the returned data values. The Retrieval subsystem will convert the timestamps to local time in the specified time zone.

The wwTimeZone extension may be assigned any of the values stored in the TimeZone column of the TimeZone table in the Runtime database.

The TimeZone table is repopulated at every system startup from Microsoft operating system registry entries, and will therefore reflect the time zones available from the server operating system, including any new or custom time zones which might be added by operating system service packs or installed software.

The Retrieval subsystem will automatically correct for daylight savings time in the requested time zone. When computing daylight savings and time zone parameters, the settings of the server operating system are used. The Retrieval subsystem does not provide any means for using client-side settings.

If wwTimeZone is not specified, the time zone for retrieval defaults to the time zone of the AVEVA Historian computer.

For example:

```
SELECT TagName, DateTime, Value, wwTimeZone
   FROM History
      WHERE TagName IN ('SysTimeHour', 'SysTimeMin')
         AND DateTime >= '2001-12-20 0:00'
         AND DateTime <= '2001-12-20 0:05'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeZone = 'W. Europe Standard Time'
```

The results are:

TagName	DateTime	Value	wwTimeZone
SysTimeMin	2001-12-20 00:00:00.000	0	W. Europe Standard Time
SysTimeHour	2001-12-20 00:00:00.000	15	W. Europe Standard Time
SysTimeMin	2001-12-20 00:01:00.000	1	W. Europe Standard Time
SysTimeMin	2001-12-20 00:02:00.000	2	W. Europe Standard Time
SysTimeMin	2001-12-20 00:03:00.000	3	W. Europe Standard Time
SysTimeMin	2001-12-20 00:04:00.000	4	W. Europe Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SysTimeMin	2001-12-20 00:05:00.000	5	W. Europe Standard Time

Quality Rule (wwQualityRule)

The quality rule can be used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC.

- A quality rule of GOOD means that data values with uncertain (64) OPC quality are not used in the retrieval calculations and are ignored. Values with bad QualityDetail indicate gaps in the data.
- A quality rule of EXTENDED means that data values with both good and uncertain OPC quality are used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data.
- A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values do not cause the overall calculations to return NULL.

You can apply a quality rule to all retrieval modes.

The OPTIMISTIC setting for the quality rule lets you retrieve information that is possibly incomplete but may nevertheless provide better results where the calculation cycle contains data gaps. This setting calculates using the last known good value prior to the gap (if possible). The logic for determining the quality of the points returned remains unchanged. The integral retrieval mode is an exception to this where the integral is scaled up to cover gaps. For more information, see Integral Retrieval.

The following figure shows a counter retrieval situation in which three of the four shown cycle boundaries are located in data gaps. Without using OPTIMISTIC, counter queries would return a NULL at all cycle boundaries because the mode needs valid good values at each end of the cycle calculate a precise difference.

filter_counter_query

If the query were to specify OPTIMISTIC, the counter mode will always return rows with numeric counter values and good quality. These rows may or may not be precise. The PercentGood column of the row returns the percentage of time in each cycle in which retrieval was able to find values stored with good quality, so if the PercentGood is anything less than 100, then the returned row may be incorrect. Quality is returned as uncertain if percent good is not 100 percent.

Now look at the counter values that are returned using OPTIMISTIC quality in the preceding illustration. The query skips the value to be returned at the first cycle boundary, because there is not enough information about the cycle prior to that boundary. At the second cycle boundary, the value 0 will be returned, because there was a gap in the data for the entire first cycle. In the second cycle, there are two points, P1 and P2. The query uses P2 as the end value of the cycle and infers a start value of the cycle from P1. At the third cycle boundary, T_{c2}, the query returns P2 – P1. Similarly, at the last cycle boundary, the query returns P4 – P3.

For the integral retrieval mode, the query does not summarize data for gaps because there is no way to know which value to use for the summarization. However, if the query specifies OPTIMISTIC quality, the query uses the last known good value for the summarization in the gap. As described for the counter retrieval example, the PercentGood column also expresses the quality of the calculated value in integral retrieval, so if the PercentGood is anything less than 100, then the returned row may be incorrect.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Quality Rule - Query Examples

To force a query to exclude points with doubtful OPC quality, specify the following in the query:

```
   AND wwQualityRule = 'Good'
```

To force a query to use points with both good and doubtful OPC quality, specify the following in the query:

```
   AND wwQualityRule = 'Extended'
```

For examples, see the following:

- Quality Rule - Query 1
- Quality Rule - Query 2
- Quality Rule - Query 3
- Quality Rule - Query 4
- Quality Rule - Query 5
- Quality Rule - Query 6
- Quality Rule - Query 7
- Quality Rule - Query 8

Quality Rule - Query 1

If you include the wwQualityRule column in a SELECT statement, it will show which quality rule was used for the individual row, if applicable.

You can combine OPC qualities in a query. For example, if you combine a mixture of good OPC qualities (such as 192 to 219), a good OPC quality (192) will be returned as a combined result.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
          WHERE TagName = 'I0R5'
          AND DateTime >= '2009-09-12 00:20'
          AND DateTime <= '2009-09-12 00:40'
          AND wwResolution = 10000
          AND wwRetrievalMode = 'Avg'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	193
I0R5	2009-09-12 00:14	3	195
I0R5	2009-09-12 00:22	0	196
I0R5	2009-09-12 00:25	1	199
I0R5	2009-09-12 00:27	0	200
I0R5	2009-09-12 00:29	2	207

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

I0R5	2009-09-12 00:33	3	215
I0R5	2009-09-12 00:36	0	216
I0R5	2009-09-12 00:39	1	219

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
I0R5	2009-09-12 00:20	2.6	192	192	AVERAGE
I0R5	2009-09-12 00:30	1.0	192	192	AVERAGE
I0R5	2009-09-12 00:40	1.6	192	192	AVERAGE

Quality Rule - Query 2

Similar to Quality Rule - Query 1, if you combine a mixture of doubtful OPC qualities, a doubtful OPC quality (64) will be returned as the combined OPC quality.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
         WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Integral'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	65
I0R5	2009-09-12 00:14	3	68
I0R5	2009-09-12 00:22	0	71
I0R5	2009-09-12 00:25	1	74
I0R5	2009-09-12 00:27	0	79
I0R5	2009-09-12 00:29	2	80
I0R5	2009-09-12 00:33	3	88
I0R5	2009-09-12 00:36	0	92

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

I0R5	2009-09-12 00:39	1	64

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
I0R5	00:20	26.0	64	64	INTEGRAL
I0R5	00:30	10.0	64	64	INTEGRAL
I0R5	00:40	16.0	64	64	INTEGRAL

Quality Rule - Query 3

When you combine the same OPC quality then that OPC quality will be returned. However, when there is no good point in a cycle for cyclic modes such as Integral, Average, Counter, or AnalogSummary, the returned NULL value will have an OPC quality of 0 and a Quality Detail of 65536, regardless of combined qualities.

```
SELECT TagName, StartDateTime, EndDateTime, OPCQuality, PercentGood, wwRetrievalMode, first
   FROM AnalogSummaryHistory
         WHERE TagName = 'F0R5'
         AND StartDateTime >= '2009-09-12 00:20'
         AND EndDateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Cyclic'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
F0R5	2009-09-12 00:07	1.6	78
F0R5	2009-09-12 00:14	3.1	78
F0R5	2009-09-12 00:22	0.2	78
F0R5	2009-09-12 00:25	0.8	78
F0R5	2009-09-12 00:27	0.4	78
F0R5	2009-09-12 00:29	2.2	78
F0R5	2009-09-12 00:33	3.3	78
F0R5	2009-09-12 00:36	0.3	78
F0R5	2009-09-12 00:39	1.2	78

The results are:

Tagname	StartDate	EndDate	OPCQuality	PercentGood	wwRetrieval	first

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Time	Time			Mode	
F0R5	2009-09-12 00:20	2009-09-12 00:30	78	100	CYCLIC	0.200
F0R5	2009-09-12 00:30	2009-09-12 00:40	78	100	CYCLIC	3.300

Quality Rule - Query 4

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
      WHERE TagName = 'F0R5'
      AND DateTime >= '2009-09-12 00:20'
      AND DateTime <= '2009-09-12 00:40'
      AND wwResolution = 10000
      AND wwRetrievalMode = 'Avg'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
F0R5	2009-09-12 00:07	1.6	15
F0R5	2009-09-12 00:14	3.1	15
F0R5	2009-09-12 00:22	0.2	15
F0R5	2009-09-12 00:25	0.8	15
F0R5	2009-09-12 00:27	0.4	15
F0R5	2009-09-12 00:29	2.2	15
F0R5	2009-09-12 00:33	3.3	15
F0R5	2009-09-12 00:36	0.3	15
F0R5	2009-09-12 00:39	1.2	15

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
F0R5	2009-09-12 00:20	NULL	65536	0	AVERAGE
F0R5	2009-09-12 00:30	NULL	65536	0	AVERAGE

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

F0R5	2009-09-12 00:40	NULL	65536	0	AVERAGE

Quality Rule - Query 5

When you combine a mixture of good, bad, and uncertain OPC qualities, a doubtful OPC quality (64) will be returned as a combined result.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
      WHERE TagName = 'F0R5'
      AND DateTime >= '2009-09-12 00:20'
      AND DateTime <= '2009-09-12 00:40'
      AND wwResolution = 10000
      AND wwRetrievalMode = 'Avg'
      AND wwQualityRule = 'Optimistic'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
F0R5	2009-09-12 00:07	1.6	15
F0R5	2009-09-12 00:14	3.1	69
F0R5	2009-09-12 00:22	0.2	78
F0R5	2009-09-12 00:25	0.8	200
F0R5	2009-09-12 00:27	0.4	15
F0R5	2009-09-12 00:29	2.2	92
F0R5	2009-09-12 00:33	3.3	88
F0R5	2009-09-12 00:36	0.3	199
F0R5	2009-09-12 00:39	1.2	196

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
F0R5	2009-09-12 00:20	2.012	64	64	AVERAGE
F0R5	2009-09-12 00:30	0.820	64	64	AVERAGE
F0R5	2009-09-12 00:40	1.751	64	64	AVERAGE

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Quality Rule - Query 6

For RoundTrip, StateSummary, and ValueState modes, the OPC qualities are only combined with the same state in a cycle. If the state only occurs once in a cycle, then the qualities of that state will be returned. The returned NULL state will always have an OPC quality of 0 and Quality Detail of 65536. The same qualities are returned for a state that has no roundtrip in RoundTrip mode.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, StateTime
   FROM History
      WHERE TagName = 'I001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'RoundTrip'
         AND wwStateCalc = 'MaxContained'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I001	2009-09-12 00:12	1	90
I001	2009-09-12 00:15	2	65
I001	2009-09-12 00:22	1	85
I001	2009-09-12 00:23	2	75
I001	2009-09-12 00:26	1	75
I001	2009-09-12 00:29	2	70

The results are:

Tagname	DateTime	Value	QualityDetail	OPC-Quality	StateTime
I001	2009-09-12 00:20	NULL	65536	0	NULL
I001	2009-09-12 00:20	1.0	90	90	NULL
I001	2009-09-12 00:20	2.0	65	65	NULL
I001	2009-09-12 00:20	1.0	64	64	4000
I001	2009-09-12 00:20	2.0	64	64	6000

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Quality Rule - Query 7

The returned Quality Detail is the same as OPC quality unless there is special flag for certain indication; for example, when there is indication for rollover in counter mode.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality
   FROM History
      WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Avg'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	218
I0R5	2009-09-12 00:14	3	218
I0R5	2009-09-12 00:22	0	218
I0R5	2009-09-12 00:25	1	218
I0R5	2009-09-12 00:27	0	218
I0R5	2009-09-12 00:29	2	218
I0R5	2009-09-12 00:33	3	218
I0R5	2009-09-12 00:36	0	218
I0R5	2009-09-12 00:39	1	218

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality
I0R5	2009-09-12 00:20	2.6	218	218
I0R5	2009-09-12 00:30	1.0	218	218
I0R5	2009-09-12 00:40	1.6	218	218

Quality Rule - Query 8

For Interpolated mode only, the returned row with Linear wwInterpolationType will have combined qualities.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode,
wwInterpolationType
   FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Interpolated'
         AND wwInterpolationType = 'Linear'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	193
I0R5	2009-09-12 00:14	3	195
I0R5	2009-09-12 00:22	0	196
I0R5	2009-09-12 00:25	1	199
I0R5	2009-09-12 00:27	0	200
I0R5	2009-09-12 00:29	2	207
I0R5	2009-09-12 00:33	3	215
I0R5	2009-09-12 00:36	0	216
I0R5	2009-09-12 00:39	1	219

The results are:

Tagname	DateTime	Value	QualityDetail	OPC Quality	wwRetrievalMode	wwInterpolationType
I0R5	2009-09-12 00:20	0.8	192	192	Interpolated	Linear
I0R5	2009-09-12 00:30	2.3	192	192	Interpolated	Linear
I0R5	2009-09-12 00:40	1.0	192	219	Interpolated	Linear

Note: Cyclic, Full, Delta, Maximum, Minimum, and BestFit do not have combined qualities; therefore, the rules are not applied to these modes..

State Calculation (wwStateCalc)

The state calculation setting applies to ValueState and RoundTrip retrieval.

For ValueState retrieval, you can choose the type of state calculation (aggregation) to be performed on the data:

- **Minimum:** The shortest amount of time that the tag has been in each unique state.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Maximum:** The longest amount of time that the tag has been in each unique state.
- **Average:** The average amount of time that the tag has been in each unique state.
- **Total:** The total amount of time that the tag has been in each unique state.
- **Percent:** The total percentage of time that the tag has been in each unique state.
- **MinContained**: The shortest amount of time each tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **MaxContained**: The longest amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **AvgContained**: The average amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **TotalContained**: The total amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **PercentContained**: The percentage of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.

All results except Percent are in milliseconds. Percent is a percentage typically between 0.0 and 100.0. The percentage can be higher than 100 in certain circumstances.

The nature of the data and how you set the cycle count determines whether you should use a "contained" version of the aggregation. The calculations apply to each unique value state that the tag was in during each retrieval cycle for the query. The total and percent calculations are always exact, but the minimum, maximum, and average calculations are subject to "arbitrary" cycle boundaries that do not coincide with the value changes. Therefore, non-intuitive results may be returned. This is most apparent for slowly-changing tags queried over long cycles.

For example, a string tag that assumes only two distinct values changing every 10 minutes is queried with a cycle time of two hours. Going into a cycle, the value (state) at the cycle boundary is recorded. If the value then changes a short while into the cycle, the state found at the cycle start time will most likely end up being the minimum value. Likewise, the state at the cycle end time is cut short at the cycle end time. The two cut-off occurrences in turn skew the average calculation.

For RoundTrip retrieval, you can only specify the following types of state calculations (aggregations) to be performed on the data. The calculations are for each unique state within each retrieval cycle for the query.

- **MinContained**. The shortest time span between consecutive leading edges of any state that occurs multiple times within the cycle, while disregarding state occurrences that are not fully contained inside of the cycle.
- **MaxContained**. The longest time span between consecutive leading edges of any state that occurs multiple times within the cycle, while disregarding state occurrences that are not fully contained inside of the cycle.
- **AvgContained**. The average time span between consecutive leading edges of any state that occurs multiple times within the cycle, while disregarding state occurrences that are not fully contained inside of the cycle. (This is the default.)
- **TotalContained**. The total time span between consecutive leading edges of any state that occurs multiple times within the cycle while disregarding state occurrences that are not fully contained inside of the cycle.
- **PercentContained**. The percentage of the cycle time spent in time span between consecutive leading edges for a state that occurs multiple times within the cycle while disregarding value occurrences that are not fully contained inside of the cycle.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Analog Value Filtering (wwFilter)

You can use the following analog filters for all retrieval modes:

- Statistical removal of outliers
- Analog to discrete conversion
- Zero around a base value

These filters are applied in a query to retrieve data from the History table, WideHistory table, or StateWideHistory table. These filter only apply to analog tags. All other types of tags, including analog summary tags, are not supported.

You need to specify a filter name in the virtual column wwFilter, with or without an override, to the set of parameters that are defined for the specified filter. The filters are specified as C-like functions: parentheses are always required, even when you choose not to override the default parameters by passing no parameters.

The default value for the wwFilter column is 'NoFilter'. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.

When you use the analog filters in a query that uses wwQualityRule, wwQualityRule is applied first and wwFilter is applied later. You can only use one filter per query.

Statistically Removing Outliers (SigmaLimit)

This analog filter removes outliers from a set of analog points based on the assumption that the distribution of point values in the set is a normal distribution.

The following illustration shows an example of outliers.

filter_outliers

You can filter outliers by specifying a filter called 'SigmaLimit()'. This filter has one parameter defined for specifying the value of n. This parameter is of type double. If the parameter is omitted, then a default parameter of 2.0 is used.

When this filter is specified in any retrieval mode, a time weighted mean, ì (mu), and time weighted standard deviation, ó (sigma), are found for each analog tag for the entire query range including phantom cycles if any, and points falling outside of the range [ì - nó, ì + nó] are removed from the point set before the points are processed further. In other words, the value will be filtered out if value > ì + nó or value < ì – nó.

Time weighted standard deviation is calculated as:

Math.Sqrt((integralOfSquares - 2 * timeWeightedAverage * integral + totalTime * timeWeightedAverage * timeWeightedAverage)/totalTime);

This is the single pass equivalent to the formula:

filter_equation

Ranges where the value is NULL are excluded from these calculations.

A cyclic query example using a 'SigmaLimit()' filter without specifying the n value would look like this:

```
SELECT DateTime, Value, wwFilter
   FROM History
      WHERE TagName = ('TankLevel')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'Cyclic'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND wwFilter = 'SigmaLimit()'
```

Not specifying the n-value as done here is the same as specifying 'SigmaLimit(2)'. The result set might look like this:

DateTime	Value	wwFilter
2008-01-15 15:00:00.000	34.56	SigmaLimit()
2008-01-15 16:00:00.000	78.90	SigmaLimit()
2008-01-15 17:00:00.000	12.34	SigmaLimit()

If the first value would be filtered out by the SigmaLimit filter, the value will be replaced with the time weighted mean.

Converting Analog Values to Discrete Values (ToDiscrete)

The analog to discrete conversion filter allows you to convert value streams for any analog tag in the query tag list into discrete value streams. The filter can be used with all the retrieval modes.

To convert analog values to discrete values, specify the ToDiscrete() filter in the wwFilter column. This filter has two parameters:

Parameter	Valid Values	Default Value
CutoffValue	any double value	0.0
Operator	>, >=, or <=	>

The following are supported syntaxes.

- ToDiscrete()
- ToDiscrete(2)
- ToDiscrete(2, >=)

The following are unsupported syntaxes.

- ToDiscrete(2,)
- ToDiscrete(, >=)
- ToDiscrete(>=)

The cutoff value holds the value that signifies the boundary between values that are to be interpreted as OFF and values that are to be interpreted as ON.

The operator parameter controls the value range relative to the cutoff value to convert to the ON value and vice versa.

NULLs encountered in the analog value stream are copied unchanged to the discrete value stream. The quality of each discrete point is copied from the analog point that causes its production. However, the quality detail for values modified with this filter will have the QualityDetail flag 0x2000 (value changed by filter) set. For example, consider the following ValueState query:

```
SELECT DateTime, vValue, StateTime, wwFilter
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'ValueState'
         AND wwStateCalc = 'MinContained'
         AND wwResolution = 7200000
         AND wwFilter = 'ToDiscrete(29, >)'
```

Here the operator is specified as >, so values greater than but not including 29 are internally converted to ON, whereas values from 0 to 29 are converted to OFF. This query could return the following rows:

DateTime	vValue	StateTime	wwFilter
2008-01-15 15:00:00.000	0	30000	ToDiscrete(29, >)
2008-01-15 15:00:00.000	1	30000	ToDiscrete(29, >)
2008-01-15 17:00:00.000	0	30000	ToDiscrete(29, >)
2008-01-15 17:00:00.000	1	30000	ToDiscrete(29, >)

The values returned in the StateTime column show that the shortest amount of time that SysTimeSec had values equivalent to either ON or OFF and remained in that state was 30 seconds. A similar RoundTrip query would look like this:

```
SELECT DateTime, vValue, StateTime, wwFilter
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'RoundTrip'
         AND wwStateCalc = 'MaxContained'
         AND wwResolution = 7200000
         AND wwFilter = 'ToDiscrete(29, <=)'
```

Here the operator is specified as <=, so the resulting conversion is exactly opposite to that performed in the previous query. Now values smaller than or equal to 29 are internally converted to ON, whereas values from 30 to 59 are converted to OFF. This query could return the following rows:

DateTime	vValue	StateTime	wwFilter
2008-01-15 15:00:00.000	0	60000	ToDiscrete(29, <=)
2008-01-15 15:00:00.000	1	60000	ToDiscrete(29, <=)
2008-01-15 17:00:00.000	0	60000	ToDiscrete(29, <=)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2008-01-15 17:00:00.000	1	60000	ToDiscrete(29, <=)

The values returned in the StateTime column now show that the longest amount of time found between roundtrips for both the OFF and the ON state within the 2-hour cycles was 60 seconds.

Using the ToDiscrete() filter is similar to using edge detection for event tags. Edge detection returns the actual value with a timestamp in history for when a value matched a certain criteria. The ToDiscrete() filter returns either a 1 or 0 to show that the criteria threshold was crossed. The ToDiscrete() filter is more flexible, however, in the following ways:

- You can use it with delta and full retrieval.
- You can combine it with "time-in-state" calculations to determine how long a value is above a certain threshold or the duration between threshold times.

Use the ToDiscrete() filter if you are mostly interested in when something occurred, and not necessarily the exact value of the event.

For more information on edge detection, see Edge Detection for Events (wwEdgeDetection).

"Zeroing" Around a Base Value (SnapTo)

This analog filter lets you force values in a well-defined range around one or more base values to "snap to" that base value. For example, you can use this filter when a tank is known to be empty, but the tag that stores the tank level returns a "noisy" value close to zero.

The filter can be used with all retrieval modes, but its main benefits are in the aggregate retrieval modes: average, integral, minimum, and maximum.

To zero values around the base value, specify the SnapTo() filter in the wwFilter column of the query.

The syntax for this filter is:

```
SnapTo([tolerance[,base_value_1[, base_value_2]…]])
```

This filter has two parameters:

Parameter	Valid Values	Default Value
Tolerance	any double value	0.01
BaseValue	zero, one, or up to 100 comma-separated double values	single base value of 0.0

The following are supported syntaxes.

- SnapTo() – Same as SnapTo(0.01, 0.0)
- SnapTo(3.7) – Same as SnapTo(3.7, 0.0)
- SnapTo(3,) – Syntax Error
- SnapTo(,0) – Syntax error
- SnapTo(,) – Syntax error
- SnapTo(3, 4, -5) – Tolerance=3, Base Values 4 and -5.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

When the Snap to filter is specified, point values falling inside any of the ranges [Base value – Tolerance, Base value + Tolerance] will be forced to the base value before the point goes into further retrieval processing. The result is undefined if the base value +/- tolerance exceeds the range of the double data type. The range is calculated using this expression:

```
If (x <= Base value + Tolerance AND x >= Base value – Tolerance)
x = Base value
```

where *x* is the value of the point then if ranges overlap, the first matching base value will be used.

A query example from the History table looks like this:

```
SELECT DateTime, Value, wwFilter
   FROM History
      WHERE TagName = ('TankLevel')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'Average'
         AND wwResolution = 3600000
         AND wwFilter = 'SnapTo(0.01, 0, 1000)'
```

The following rows might be returned:

DateTime	Value	wwFilter
2008-01-15 15:00:00.000	0	SnapTo(0.01, 0, 1000)
2008-01-15 16:00:00.000	875.66	SnapTo(0.01, 0, 1000)
2008-01-15 17:00:00.000	502.3	SnapTo(0.01, 0, 1000)

When a value is snapped, the QualityDetail bit flag 0x2000 is set.

If the filter syntax is not correct, a syntax error is returned and no rows are returned.

Selecting Values for Analog Summary Tags (wwValueSelector)

For an analog summary tag, multiple summarized values can be stored in the historian for a single summarization period. When you query analog summary data, a single value, time, and quality (VTQ) must first be extrapolated from the summarized values.

You set the value selector in the query to specify which summarized value to return. The possible values are as follows:

Value Selector Setting	Value Returned	Timestamp Returned
AUTO	The retrieval mode determines the value. See the following table for how AUTO applies to the value selection. This is the default value.	The retrieval mode determines the timestamp. See the following table for how AUTO applies to the value selection. This is the default value.
FIRST	The first value that occurs within the summary period.	The actual timestamp of the first value occurrence within the summary period.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value Selector Setting	Value Returned	Timestamp Returned
LAST	The last value that occurs within the summary period.	The actual timestamp of the last value occurrence within the summary period.
MIN or MINIMUM	The first minimum value that occurs within the summary period.	The actual timestamp of the first minimum value occurrence within the summary period.
MAX or MAXIMUM	The first maximum value that occurs within the summary period.	The actual timestamp of the first maximum value occurrence within the summary period.
AVG or AVERAGE	A time-weighted average calculated from values within the summary period.	The summary period start time.
INTEGRAL	An integral value calculated from values within the summary period.	The summary period start time.
STDDEV or STANDARDDEVIATION	A standard deviation calculated from values within the summary period.	The summary period start time.

The following table describes the value to be considered if the value selector is set to AUTO:

Retrieval Mode	Analog Summary Behavior
Cyclic	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
Delta	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
Full	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
Interpolated	The retrieval mode determines the appropriate value to return. See the following table for how AUTO applies to the value selection. This is the default value.
Best Fit	The first, last, min, and max points from analog summaries are all considered as analog input points. Best fit analysis is done with these points. If the analog summary percentage good is not 100%, the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval Mode	Analog Summary Behavior
	cycle is considered to have a NULL.
Average	The averages of analog summaries are calculated using the values from the Average column of the AnalogSummaryHistory table. Interpolation type is ignored for analog summary values, and STAIRSTEP interpolation is always used. PercentGood is calculated by considering the TimeGood of each analog summary. If cycle boundaries do not exactly match the summary periods of the stored analog summaries, the averages and time good are calculated by prorating the average and time good values for the portion of the time the summary period overlaps with the cycle. Quality will be set to 64 (uncertain) if cycle boundaries do not match summary periods. If the QualityDetail of any analog summary considered for a cycle is uncertain (64), the resulting quality is set to 64.
Minimum	The first minimum value within the summary period is used. The actual timestamp of the first minimum value occurrence within the summary period is used.
Maximum	The first maximum value within the summary period is used. The actual timestamp of the first maximum value occurrence within the summary period is used.
Integral	The integrals of analog summaries are calculated using the Integral column of the AnalogSummaryHistory table. Interpolation type is ignored for analog summary values, and STAIRSTEP interpolation is always used. PercentGood is calculated by considering the TimeGood of each analog summary. If cycle boundaries do not exactly match the summary periods of the stored analog summaries, the integrals and time good are calculated by prorating the integral and time good values for the portion of the time the summary period overlaps with the cycle. Quality is set to 64 (uncertain) if cycle boundaries do not match summary periods. If the QualityDetail of any analog summary considered for a cycle is uncertain (64), the resulting quality will

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval Mode	Analog Summary Behavior
	be set to 64.
Slope	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
ValueState	Cannot be used with analog summary data. No values are returned.
Counter	Cannot be used with analog summary data. No values are returned.
RoundTrip	Cannot be used with analog summary data. No values are returned.

For an analog summary tag, if any of the data within a summary period has an OPCQuality other than Good, the OPCQuality returned will be Uncertain. This is true even for summary values that are not calculated, such as first, last, minimum, maximum, and so on. For example, if the OPCQuality for a last value is actually Good, but there was a I/O Server disconnect during the summary calculation period, the OPCQuality for the last value is returned as Uncertain. A QualityDetail of 202 is used to distinguish between the original point and the latest point. (Note, however, that replicated data may show a QualityDetail of 202 for points other than the latest point.)

Expressions (wwExpression)

Expressions allow you to perform mathematical and logical operations on tag data.

This option is relevant in the following retrieval modes:

- Cyclic Retrieval
- Delta Retrieval
- Full Retrieval
- Best Fit Retrieval

Note: Using expressions requires the "advanced" license feature to be enabled. If the advanced license is not available when an expression is used, no data is retrieved and an error message is displayed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

What is an expression?

An expression is made up of a combination of the following elements:

1. Values
   - A typical value is made up of three components:
     - A value with a base type of Boolean, integer, or double
     - A date and time in UTC format
     - A 16-bit integer representing the OPC quality of the value
   - Literal values are also supported, which have no date and time associated with them, and are always assumed to be "good" quality.
2. Streams
   - A stream is a time-ordered collection of values. A stream has either a stair-step or linear interpolation model, and an associated engineering unit.
   - In the context of an expression, one tag is the initial stream, but the expression can produce additional streams derived from a tag. For example, the expression `TI101.PV + 4.0` produces a result stream, but the initial stream is the stream associated with `TI101.PV`.
3. Arithmetic and logical operators
   - Standard arithmetic operators such as addition, subtraction, multiplication and division are supported, along with logical operators for comparison, such as greater than (>) and less than (<). For a full list of supported operators, see Supported arithmetic operators.
4. Scalar functions
   - A variety of scalar functions are supported for manipulating data, such as calculating the absolute value, rounding values to the nearest integer, or unit of measure conversion. For a full list of supported scalar functions, see Supported scalar functions.
5. Time-series functions
   - A variety of time-series functions are supported for comparing data, such as calculating minimum/maximum values, or time in state. For a full list of supported functions, see Supported time-series functions.
   - Time intervals can be expressed in either of two ways:
     a. As a number of days. For example, a value of 1 represents a daily period, while a value of .125 represents a period of three hours.

        Note: Daily time periods support Daylight Saving Time. This means the "1 day" period is 24 hours, except on days when the Daylight Saving Time change is made; on those days, it will be either 23 or 25 hours.

     b. As a named period. The following named periods are supported: *minute*, *hour*, *day*, *week*, *month*, *fixedday*, *fixedweek*, *fixedmonth*. These periods are anchored to a rounded time period. For example, "hour" produces hourly intervals where the minutes, seconds and milliseconds are all 0 in local time, while "minute" produces intervals where the seconds and milliseconds are all 0 in local time.

        Note: The *day*, *week* and *month* named periods support Daylight Saving Time, while the *fixedday*, *fixedweek* and *fixedmonth* named periods do not. For example, the length of a *day* period can be 23, 24, or 25 hours, depending on whether a Daylight Saving Time transition takes place. The length of a *fixedday* period is always 24 hours, *fixedweek* is always 7 days times 24 hours, and *fixedmonth* is always the number of calendar days in the month times 24 hours.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Implicit unit conversion

The following table describes the relationships between engineering unit dimensions when they are multiplied or divided by each other.

Factor1	Factor2	Product
Data Rate	Time	Data
Density	Volume	Mass
Energy Density	Volume	Energy
Length	Area	Volume
Length	Length	Area
Mass Flow	Time	Mass
Pressure	Area	Force
Pressure Rate	Time	Pressure
Resistance	Current	Electric Potential
Speed	Time	Length
Temperature Rate	Time	Temperature
Unknown Rate	Time	Unknown
Volumetric Flow	Time	Volume
Radiant Exposure	Time	Irradiance
Rate	Time	Ratio
Rotation Speed	Time	Rotation
Electric Potential	Power	Current
Acceleration	Mass	Force
Force	Speed	Power
Mass	Specific Energy	Energy

Each row in the table corresponds to four equivalent dimension relationships:

- *Dimension 1 * Dimension 2 = Result Dimension*
- *Dimension 2 * Dimension 1 = Result Dimension*
- *Result Dimension / Dimension 1 = Dimension 2*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- *Result Dimension / Dimension 2 = Dimension 1.*

When working with expressions involving tags or streams with different engineering units, the engineering units of the results are determined as follows:

1. When adding or subtracting streams with units in the same dimension, all streams are first converted to the same unit as the first stream, and then the operations are performed. The results are in the same unit as the first stream. For example, using the expression `SysTimeHour + SysTimeMin`, both tags use units in the *Time* dimension. When the expression is evaluated, the units from the first tag (hours) are used, so `SysTimeMin` is first converted to hours, and the results are in hours.
2. When adding or subtracting streams with one or more streams having "undefined" units, the results are in the same unit as the first stream.
3. When multiplying or dividing a dimensionless stream or numeric constant with a stream that has an associated unit, the results are in that unit.
4. When multiplying or dividing streams where both units are related by one of the entries in the *DimensionMath* table, the result is in the associated result dimension's unit.
5. When multiplying streams where both units are in the same dimension, the scale factors of the two units are multiplied together. If the scale factor matches a unit in the result dimension, the results are in this unit. Otherwise, the original units are first converted to their dimension's base units, and the results are in the base units of the result dimension.
6. Any other combinations cannot be implicitly converted, so the results are in "undefined" units.

Note: AVEVA Historian 2023 updates the base units for some dimensions to enable implicit unit conversion. If you have additional Historian nodes configured for replication, the engineering unit catalog will be out of sync on tier 2 nodes until they are upgraded to the 2023 version.

Interpolation with Expressions

When retrieving data with expressions, the system determines the interpolation type for the expression based on the interpolation type of the individual tags in the expression.

If the interpolation type in the tag metadata is linear then linear interpolation is used, otherwise stair-step interpolation is used. For more information, see Interpolated Retrieval.

Special rules apply when using some expression functions:

1. An expression using the SIGN function uses stair-step interpolation.

   For example, Pump101.Flow is an analog tag representing the flow rate for a pump. While the tag uses linear interpolation, the expression `SIGN(Pump101.Flow)` uses stair-step interpolation.
2. An expression using the ROUND function uses stair-step interpolation.

   For example, Tank100.Level is an analog tag representing the level of fluid in a tank. While the tag uses linear interpolation, the expression `ROUND(Tank100.Level)` uses stair-step interpolation.

Supported arithmetic operators

AVEVA Historian supports the use of standard mathematical operators in expressions. The following operators are supported:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Symbol	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
%	Modulo
( )	Operator grouping

Arithmetic Operator Examples

The following examples show expressions that use arithmetic operators. The tag names used in these examples are:

- *Pump1.FlowRate* measures the flow rate of a pump in liters per second (L/s)
- *Pump2.FlowRate* measures the flow rate of a pump in liters per minute (L/min)
- *TankCapacity* measures the available capacity of a tank in kiloliters (kL)
- *TimeElapsed* measures elapsed time in seconds (s)

Example 1

The following expression returns the sum of flow rates for two pumps.

```
Pump1.FlowRate + Pump2.FlowRate
```

Because the tags use different units, and there is no explicit unit conversion specified, the units of *Pump2.FlowRate* are implicitly converted to liters/second to match the units of *Pump1.FlowRate* before the values are added together.

Example 2

The following expression returns a value per minute representing the total volume flowing through both pumps during that minute.

```
TOTAL(Pump1.FlowRate + Pump2.FlowRate, 1 minute)
```

Example 3

The following expression calculates a pump's efficiency as a percentage. This example uses water at 50 degrees Celsius, which has a density of 992.24 kg/m^3 and head of 3.5m, the gravitational constant of 9.81 m/s^2, and "Pump1.Power" represents the power used by the pump in Watts.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Pump1.FlowRate * 3.5 * 992.24 * 9.81 / 100.0 / Pump1.Power
```

Example 4

The following expression returns the travel time in seconds for material in a pipe. The pipe has a diameter of 0.15m, and a length of 100m.

```
3.1415 * 0.15 ^ 2.0 * 100 / UOM(Pump1.FlowRate, m3/sec)
```

Supported logical operators

AVEVA Historian supports the use of standard mathematical operators in expressions. The following operators are supported:

Symbol	Operation
=	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

Supported logical functions

In addition to logical operators, AVEVA Historian also supports the use of logical functions in expressions. The following logical functions are supported:

Function	Usage
AND(*arguments*)	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.
OR(*arguments*)	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.
NOT(*argument*)	Evaluates the argument to a Boolean *TRUE* or *FALSE* value, and returns the opposite.

AND

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **AND**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.

OR

Format	*Stream* = **OR**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.

NOT

Format	*Stream* = **NOT**(*expression*)
Arguments	Any expression.
Description	Evaluates the expression to a boolean *TRUE* or *FALSE* value, then returns the opposite.

Supported scalar functions

AVEVA Historian supports using scalar functions in expressions. These functions have a 1:1 relationship between the values in the input and output streams.

The following scalar functions are supported:

Function	Purpose
ABS	Calculates the absolute value of a stream.
CASTUOM	Changes a stream's unit of measure without changing the values.
IF	Produces a stream based on the results of a logical expression.
IFBAD	Overrides "bad" quality values in a stream.
IFLONGER	Evaluates whether or not a logical expression is true over a period greater than or equal to a specified time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	interval, and returns a TRUE or FALSE value or stream.
IFSHORTER	Evaluates whether or not a logical expression is true over a period shorter than a specified time interval, and returns a TRUE or FALSE value or stream.
IFSTALE	Replaces values from the input stream that are older than a specified interval with an alternative value.
LOG	Calculates the base 10 logarithm of a stream's values.
ROUND	Rounds a stream's values to two decimal places.
SIGN	Replaces a stream's values with 1 or -1 based on the current value's sign.
SQRT	Calculates the square root of a stream's values.
TIMESHIFT	Shifts the time stamps in a stream by a specified time interval.
TRUNCATE	Removes the fractional portion of a stream's values.
UOM	Converts a stream's values to a specified unit of measure.

ABS

The ABS() function produces a stream of results calculating the absolute value of a stream, converting negative values into positive values.

Syntax

Format	*Stream* = **ABS**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all negative values converted to positive values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ABS Examples

ABS Examples

Example 1

The following expression returns the difference between the flow rates of two pumps.

```
ABS( Pump101.Flow - Pump102.Flow )
```

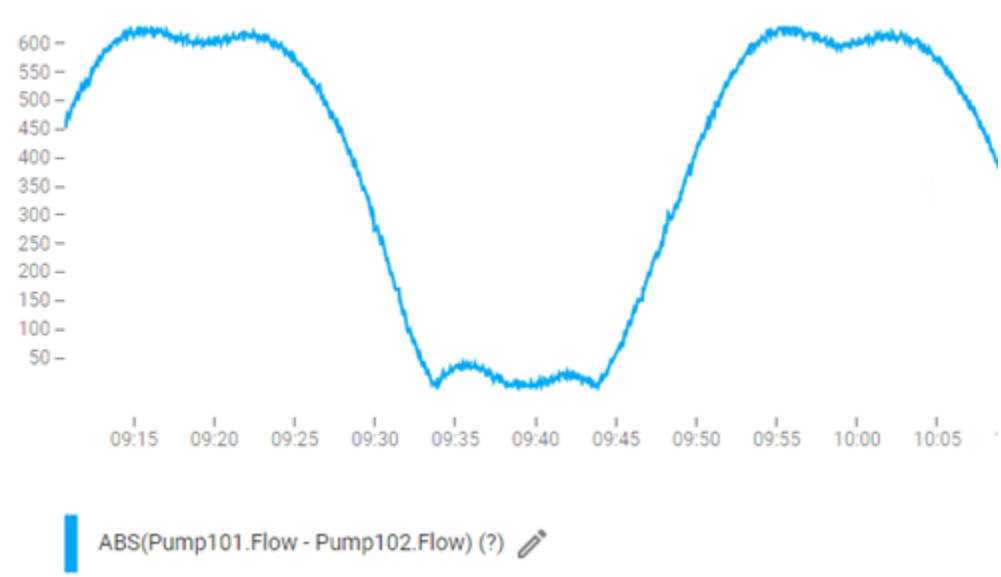
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( Pump101.Flow - Pump102.Flow )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:07:51.4320000	259.439523422241	192
2022-03-31 09:07:53.2590000	268.941925048828	192
2022-03-31 09:07:55.2590000	264.847595214844	192
...	...	...
2022-03-31 10:07:49.2590000	457.803894042969	192

Represented as a line chart over a 1 hour interval:



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression produces a oscillating stream of values from 0 to 30, and back to 0:

```
ABS( SysTimeSec - 30 )
```

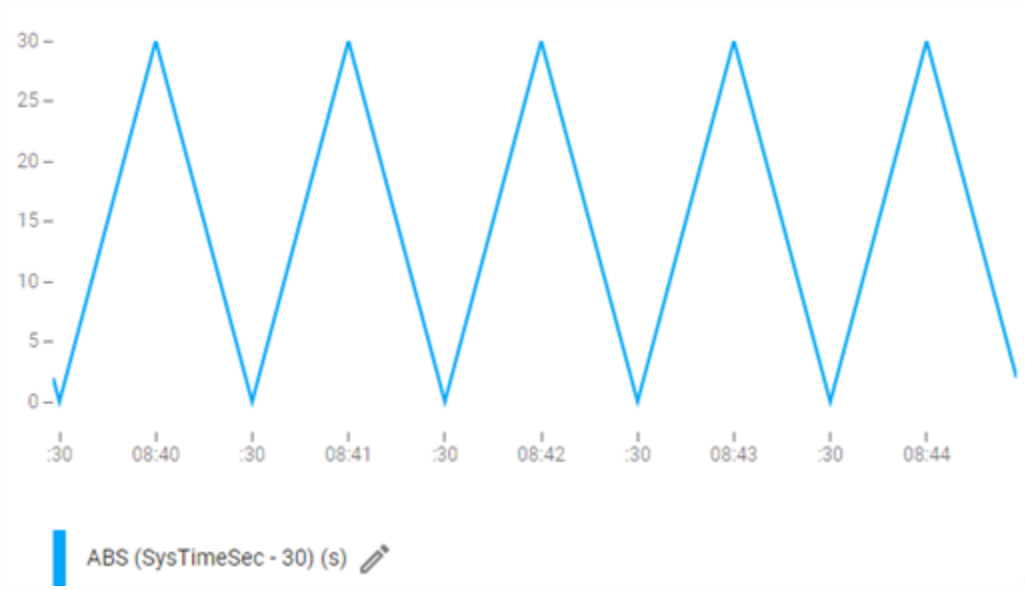
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( SysTimeSec - 30 )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:47:52.3500000	22	192
2022-03-31 09:47:53.0000000	23	192
2022-03-31 09:47:54.0000000	24	192
...	...	...
2022-03-31 10:47:51.0000000	21	192

Represented as a line chart over a 5 minute interval:



CASTUOM

The CASTUOM() function changes a stream's unit of measure to the specified unit of measure without any data conversion, preserving the original values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **CASTUOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the the provided stream to use the specified unit of measure while preserving the existing values.

IF

The IF() function produces a stream based on the results of a logical expression.

Syntax

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*, *FalseStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE *FalseStream* - A stream of values from which to select when the expression evaluates to FALSE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected for the output stream. Otherwise, the corresponding value from the *FalseStream* is selected.

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	for the output stream. Otherwise, a NULL or "bad quality" value is selected.

Format	*Stream* = **IF**(*LogicalExpression*)
Arguments	*LogicalExpression* - A logical expression to be evaluated
Description	Produces an output stream of Boolean values representing the results of *LogicalExpression*.

IF Examples

IF Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, but any negative values are replaced with 0.

```
IF( Pump101.Flow < 0, 0, Pump101.Flow )
```

Represented as a query:

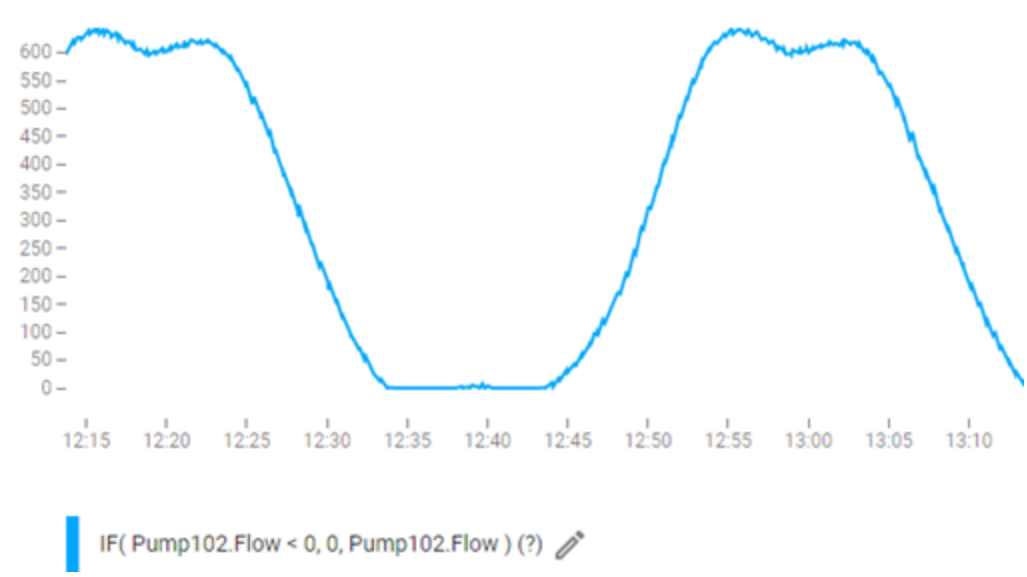
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'IF( Pump102.Flow < 0, 0, Pump102.Flow )'
```

Results over a 5 minute interval:

DateTime	Value	OPCQuality
2022-03-31 12:40:01.2590000	4.00229740142822	192
2022-03-31 12:40:03.2590000	0.365068227052689	192
2022-03-31 12:40:05.2590000	0	192
...	...	...
2022-03-31 12:44:59.2590000	27.5106182098389	192

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 2

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns 0.

```
IF( SysPerfCPUTotal > 90, 1, 0 )
```

Example 3

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns a NULL value.

```
IF( SysPerfCPUTotal > 90, 1 )
```

Example 4

The following expression returns a true value (1) when system CPU usage exceeds 90%, otherwise it returns a false value (0).

```
IF( SysPerfCPUTotal > 90 )
```

IFBAD

The IFBAD() function replaces values from the input stream that have a bad OPC Quality with an alternative value.

Syntax

Format	*Stream* = **IFBAD**(*Stream*, *AltValue*)
Arguments	*Stream* - A stream of values *AltValue* - A replacement value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	Replaces every bad value in the input stream with *AltValue*.

Format	*Stream* = **IFBAD**(*Stream*, *AltStream*)
Arguments	*Stream* - A stream of values *AltStream* - A stream of alternative values
Description	Replaces every bad value in the input stream with the value from *AltStream* at the corresponding time.

IFBAD Examples

IFBAD Examples

Example 1

The following expression replaces all bad values for the input stream Pump102.Flow with "500."

```
IFBAD(Pump102.Flow, 500)
```

Example 2

The following expression replaces all bad values in the input stream Pump102.Flow with values from the alternative stream Pump115.Flow at the corresponding time.

```
IFBAD(Pump102.Flow, Pump115.Flow)
```

LOG

The LOG() function produces a stream of results calculating the base 10 logarithm of a stream's values.

Syntax

Format	*Stream* = **LOG**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values converted to their base 10 logarithm.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LOG Examples

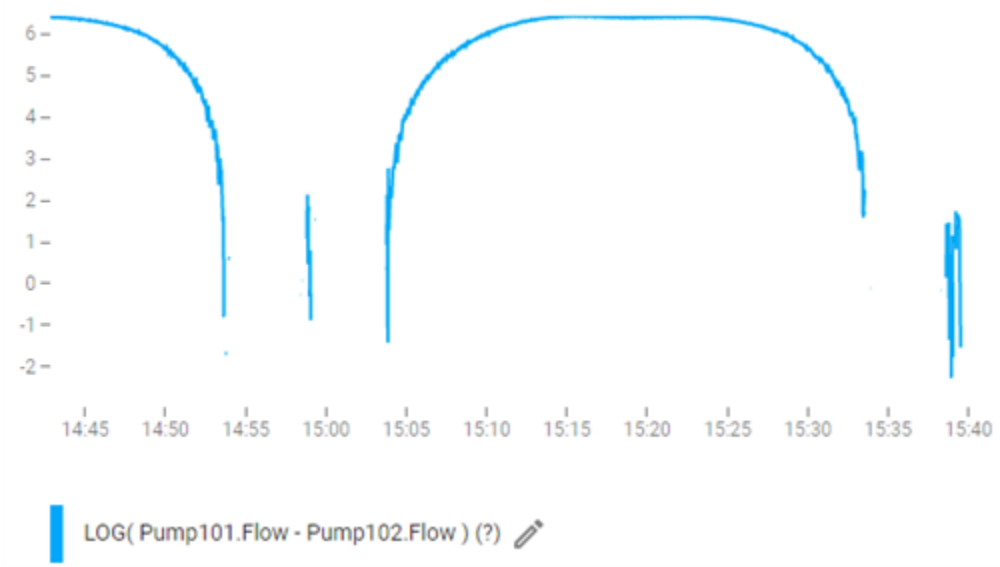
LOG Examples

Example 1

The following expression returns the base 10 logarithm of the difference in flow rates for two pumps.

```
LOG( Pump101.Flow - Pump102.Flow )
```

Represented as a line chart over a 1 hour interval:



ROUND

The ROUND() function produces a stream of results rounding a stream's values to the nearest integer.

Syntax

Format	*Stream* = **ROUND**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values rounded to the nearest integer.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ROUND Examples

ROUND Examples

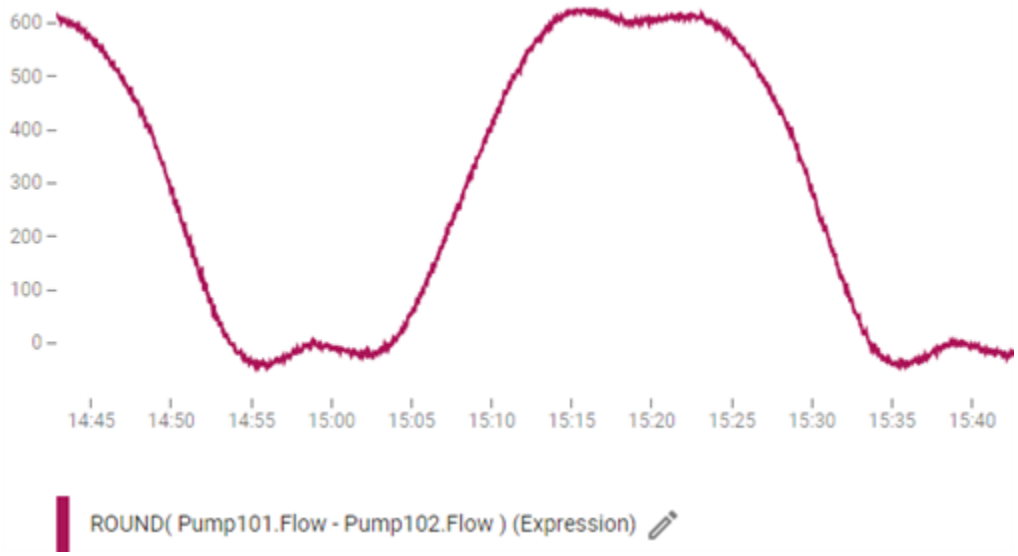
Example 1

The following expression returns the difference in flow rates for two pumps, rounded to the nearest integer.

```
ROUND( Pump101.Flow - Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag.

Represented as a line chart over a 1 hour interval:



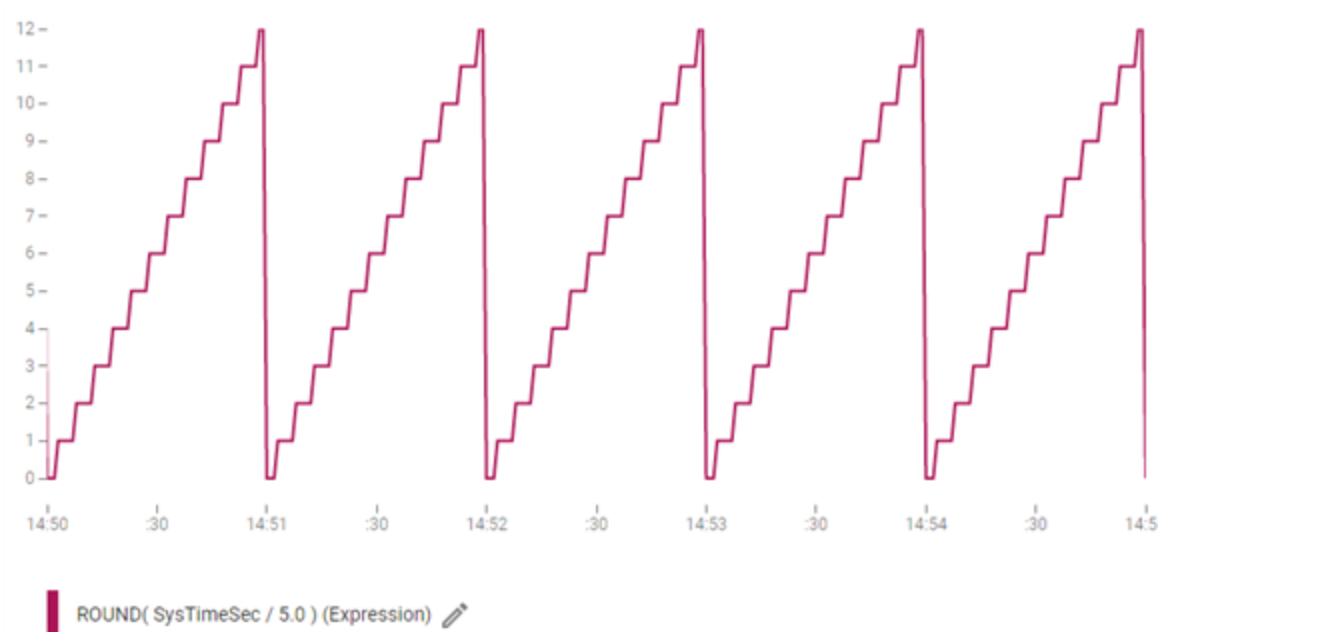
Example 2

The following expression returns the second value of the time, divided by 5 and rounded to the nearest integer.

```
ROUND( SysTimeSec / 5.0 )
```

Represented as a line chart over a 5 minute interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



SIGN

The SIGN() function produces a stream of results indicating the sign of a stream's values.

Syntax

Format	*Stream* = **SIGN**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values replaced based on their sign. Positive values are replaced with 1, negative values are replaced with -1, and zero values remain zero.

SIGN Examples

SIGN Examples

Example 1

The following expression returns the sign of the value of the difference in flow rates between two pumps.

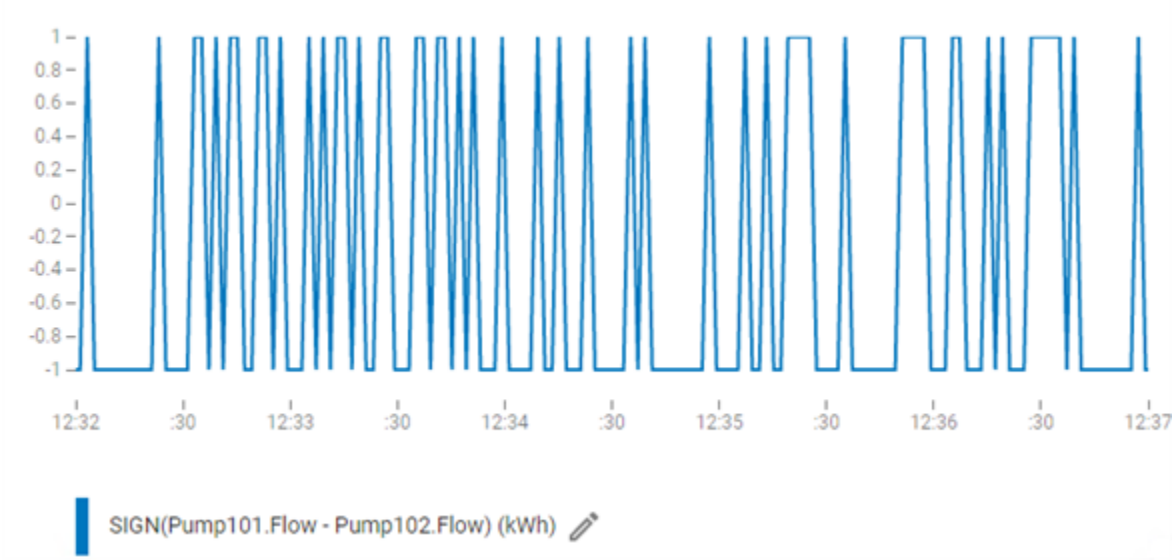
```
SIGN( Pump101.Flow - Pump102.Flow )
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For each data point in the results, the value will be:

- -1 when the second pump has a higher flow rate
- 0 when the flow rates are equal
- 1 when the first pump has a higher flow rate.

Represented as a line chart over a 5 minute interval:



SQRT

The SQRT() function produces a stream of results calculating the square root of a stream's values.

Syntax

Format	*Stream* = **SQRT**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, but with all values being replaced with their square roots.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQRT Examples

SQRT Examples

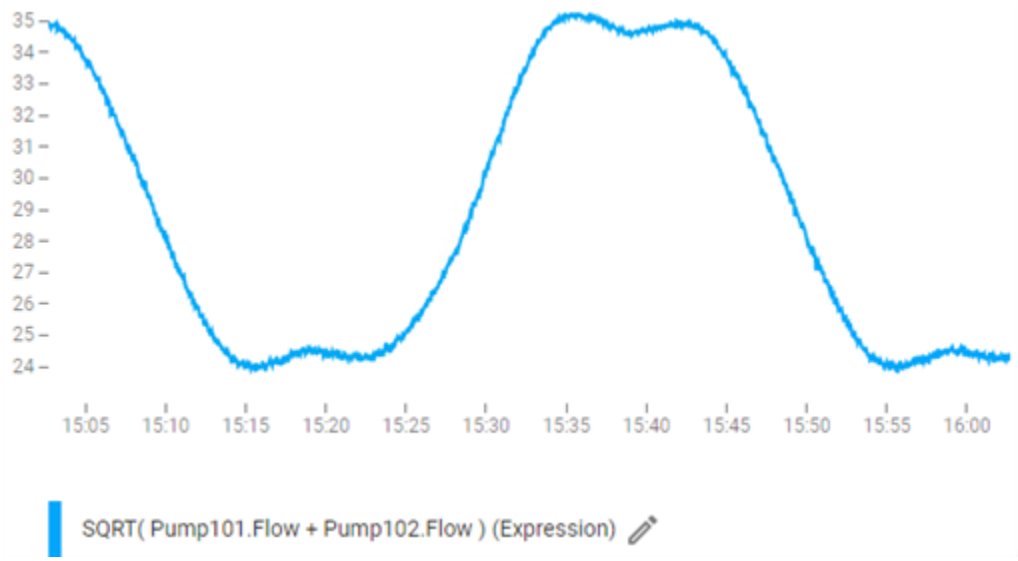
Example 1

The following expression returns the square root of the sum of flow rates for two pumps.

```
SQRT( Pump101.Flow + Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag, so the expression's results are in liters per second.

Represented as a line chart over a 1 hour interval:



TIMESHIFT

The TIMESHIFT() function shifts the time stamps in the input stream by adding or subtracting a specified time interval.

Syntax

Format	*Stream* = **TIMESHIFT**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A stream of values *TimeInterval* - The time period to add (or subtract if negative)
Description	The specified interval of time is added to (or

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	subtracted from) the time stamps in the input stream.

TIMESHIFT Examples

TIMESHIFT Examples

Example

The following expression shifts the time stamp of each flow rate value forward by one hour.

```
TIMESHIFT( Pump101.Flow, Hour )
```

Example 2

The following expression shifts the time stamp of the current time's minute value back by two minutes.

```
TIMESHIFT( SysTimeMin, -2 minute )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TIMESHIFT( SysTimeMin, -2 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:57:00.0000000	59	192
2022-03-31 15:58:00.0000000	0	192
2022-03-31 15:59:00.0000000	1	192
2022-03-31 16:00:00.0000000	2	192
...	...	...

TRUNCATE

The TRUNCATE() function produces a stream of results removing the fractional portion of a stream's values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **TRUNCATE**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with the fractional part of all values removed.

TRUNCATE Examples

TRUNCATE Examples

Example 1

The following expression returns the product of flow rates for two pumps, with the fractional portion of each value removed.

```
TRUNCATE( Pump101.Flow * Pump102.Flow )
```

Represented as a query:

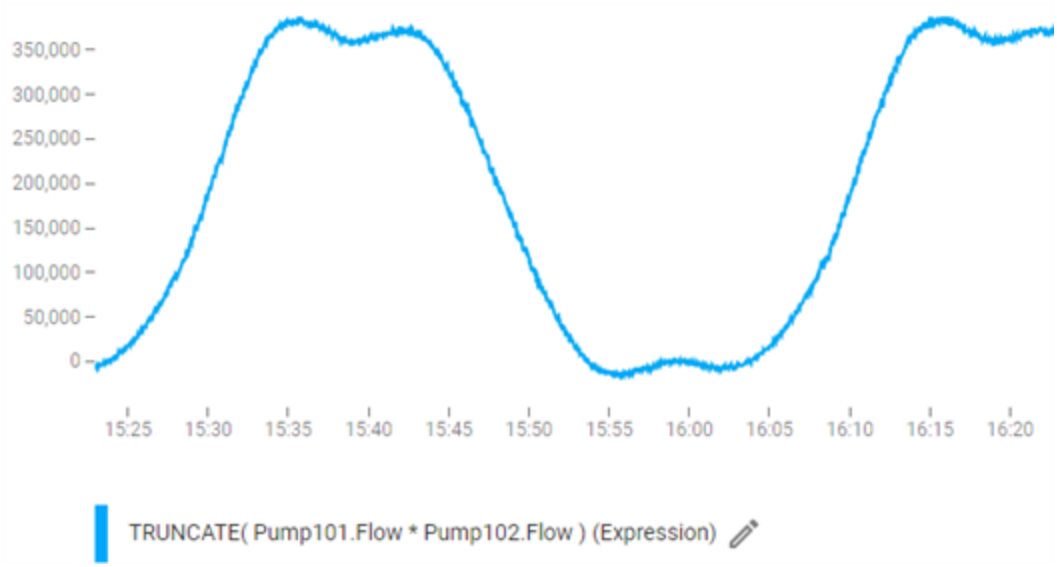
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( Pump101.Flow * Pump102.Flow )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:24:49.2590000	11030	192
2022-03-31 15:24:51.2590000	14974	192
2022-03-31 15:24:53.2590000	14542	192
2022-03-31 15:24:55.2590000	14887	192
...	...	...

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 2

The following expression divides the current time's second value by 5, removing the fractional portion.

```
TRUNCATE( SysTimeSec / 5.0 )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( SysTimeSec / 5.0 )'
AND wwRetrievalMode = 'Full'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:31:39.0000000	7	192
2022-03-31 15:31:40.0000000	8	192
2022-03-31 15:31:41.0000000	8	192
2022-03-31 15:31:42.0000000	8	192
...	...	...

UOM (Unit of Measure conversion)

The UOM() function converts the values in a stream to a specified unit of measure.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **UOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the values in the provided stream to use the specified unit of measure. If the specified unit does not exist, or is not correctly mapped to a catalog unit for conversion, the original unit is returned instead.

UOM Examples

UOM Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, with the units converted to gallons per second.

```
UOM( Pump101.Flow, gal/min )
```

Supported time-series functions

AVEVA Historian supports using time-series functions in expressions. These functions enable you to compare and manipulate data.

The following time-series functions are supported:

Function	Purpose
AVERAGE	Calculates the time-weighted average for a stream.
AVERAGES	Calculates the statistical average for a stream.
COUNT	Returns a count of the delta points for a stream.
COUNTALL	Returns a count of the data points for a stream.
DURATION	Calculates the duration of *TRUE* states in an

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	expression.
METER	Returns the delta change of a stream's value.
MIN/MAX	Selects the minimum/maximum value of a stream.
PREV/PREVGOOD	Returns the last value from the previous time period.
TOTAL	Calculates the time-weighted total for a stream.
TOTALS	Calculates the statistical total for a stream.

AVERAGE

The AVERAGE() function calculates the time-weighted average value of a stream or array of streams.

When using AVERAGE() in a query, if no resolution is specified, then the behavior is the same as average retrieval mode with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the average value is calculated for each cycle.

Syntax

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the time-weighted average value of a stream or array of streams.

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the time-weighted average of a stream or array of streams over the specified time interval, based on local time. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVERAGE Examples

AVERAGE Examples

Example 1

The following expression returns the average flow rate of a pump.

```
AVERAGE(Pump101.Flow)
```

Example 2

The following expression returns the average flow rate of a pump over 10 minute intervals.

```
AVERAGE(Pump101.Flow, 10 Minute)
```

Example 3

The following expression returns the average flow rate of all streams that include ".Flow" in their tag name at 10 minute intervals.

```
AVERAGE(SELECT("TagName",".Flow"), 10 Minute)
```

Example 4

The following expression returns the average minute value of the time over 5 minute intervals.

```
AVERAGE(SysTimeMin, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'AVERAGE( SysTimeMin, 5 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 12:00:00.0000000	2	192
2022-03-31 12:05:00.0000000	7	192
2022-03-31 12:10:00.0000000	12	192
2022-03-31 12:15:00.0000000	17	192
...	...	...

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVERAGES

The AVERAGES() function calculates the statistical average value of a stream or array of streams over a specified time interval.

Note: This is the preferred method for calculating averages from discrete samples such as those from a checkweigher. For most measurements, however, AVERAGE is preferred.

Syntax

Format	*Stream* = **AVERAGES**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the statistical average for a stream or array of streams over the specified time interval. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNT

The COUNT() function returns a count of the delta points for a stream. Note that if the quality detail of a point differs from the previous point, this counts as a delta point, even if the value doesn't change.

Syntax

Format	*Stream* = **COUNT**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of delta points during the query period. The query period is based on local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **COUNT**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the delta points are counted
Description	Returns a count of delta points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNT Examples

COUNT Examples

Example 1

The following expression returns the number of delta points in a freezer unit's temperature.

```
COUNT(Freezer15.Temperature)
```

Example 2

The following expression returns the number of delta points in a freezer unit's temperature over hour intervals.

```
COUNT(Freezer15.FlowTotal, hour)
```

COUNTALL

The COUNTALL() function returns a count of all data points for a stream.

Syntax

Format	*Stream* = **COUNTALL**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of all data points during the query period. The query period is based on local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **COUNTALL**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the points are counted
Description	Returns a count of all data points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNTALL Examples

COUNTALL Examples

Example 1

The following expression returns the number of data points reported by a pressure alert on a boiler.

```
COUNT(Boiler03.PressureAlert)
```

Example 2

The following expression returns the number of points reported by a pressure alert on a boiler over a one hour interval.

```
COUNT(Boiler03.PressureAlert, hour)
```

DURATION (Time in State)

The DURATION() function returns a stream representing the duration in seconds of each "true" state in an expression.

Syntax

Format	*Stream* = **DURATION**(*StreamLogicalExpression*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated
Description	The logical expression represented by

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	StreamLogicalExpression is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results.

Format	*Stream* = **DURATION**(*StreamLogicalExpression*, *TimeInterval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *TimeInterval* - The time interval over which to calculate durations
Description	The logical expression represented by *StreamLogicalExpression* is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results, with durations being split across the specified time period. For example, if StreamLogicalExpression evaluates to a *TRUE* value for a time period of 07:55-08:10, and the specified *Interval* is hourly, the output contains one "5 minute" and one "10 minute" result. Without specifying an *Interval*, the output would instead contain one "15 minute" result.

DURATION Examples

DURATION Examples

Example 1

The following expression returns the amount of time in seconds over the specified interval that the number of seconds in the time is 20 or greater.

```
DURATION(SysTimeSec >= 20)
```

Example 2

The following expression returns the amount of time per one hour interval that a tank's level is at or above 500 kL.

```
DURATION(Tank400.Level > 500, hour)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(Tank400.Level >= 500, hour)'
AND DateTime > '2022-04-03 00:00:00'
```

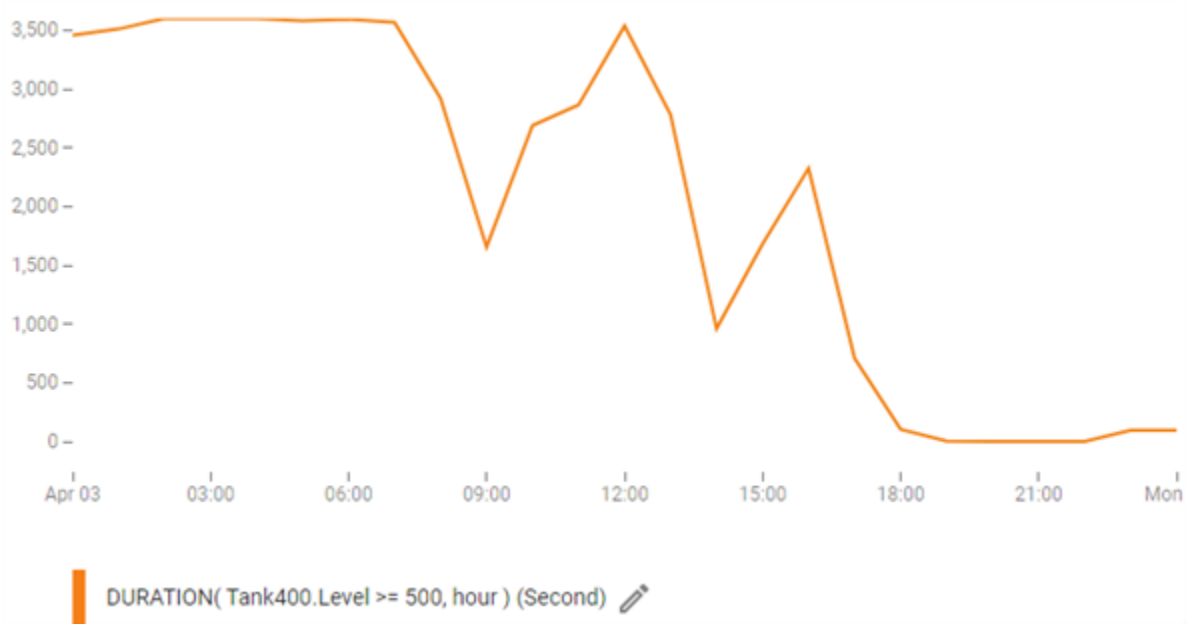
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
AND DateTime < '2022-04-04 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-04-03 01:00:00.0000000	3600	192
2022-04-03 02:00:00.0000000	3578	192
2022-04-03 03:00:00.0000000	3592	192
...	...	...
2022-04-03 23:00:00.0000000	94	192

Represented as a line chart over a 1 day interval:



Example 3

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater, over 5 minute intervals.

```
DURATION(SysTimeSec >= 20, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(SysTimeSec >= 20, 5 minute)'
```

Results:

DateTime	Value	OPCQuality
2022-04-01 08:45:00.0000000	200	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2022-04-01 08:50:00.0000000	200	192
2022-04-01 08:55:00.0000000	200	192
2022-04-01 09:00:00.0000000	200	192
...	...	...

Example 4

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater over 30 second intervals, and the results are converted to minutes.

```
UOM(DURATION(SysTimeSec >= 20, 0.5 minute), Minute)
```

IFLONGER

The IFLONGER() function evaluates whether or not a logical expression is true over a period greater than or equal to a specified time interval, and returns a TRUE or FALSE value or stream.

Syntax

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the boolean TRUE is returned. Otherwise the boolean FALSE is returned.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	when the expression evaluates to *TRUE* for the length of the specified interval or longer
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. If it is not *TRUE*, a "bad quality" value is selected.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*, *FalseStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer *FalseStream* - A stream of values from which to select when the expression does not evaluate to *TRUE* for at least the length of the specified interval
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. Otherwise the corresponding value from *FalseStream* is selected.

IFSTALE

The IFSTALE() function replaces values from the input stream that are older than a specified interval with an alternative value.

Syntax

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltValue*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltValue* - A replacement value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	Replaces every value in the input stream older than *Interval* with *AltValue*.

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltStream*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltStream* - A stream of alternative values
Description	Replaces every value in the input stream older than *Interval* with the value from *AltStream* at the corresponding time.

METER

The METER() function returns the delta change of the value of a stream or array of streams. This works both for tags without rollover, and tags that reset upon reaching a rollover value.

If an expression includes a single tag, the rollover value defined in the AVEVA Historian will be used. If an expression includes multiple tags, or performs an operation on a tag, a rollover value of 0 is assumed. For example, the following would use the Historian-defined rollover value:

- METER(Tag1, hour)

The following would assume a rollover value of 0:

- METER(Tag1 + Tag2, hour)
- METER(Tag1 + 5, hour)

Otherwise, the METER function works similarly to counter retrieval mode. For more details about rollover, delta change, and common use cases for this function, see Counter Retrieval.

Syntax

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the delta change in the stream or array of streams during the query period. The query period is based on local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which the delta change is calculated.
Description	Returns the delta change in the stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

METER Example

METER Example

Example 1

The following expression returns the changes in a pump's total flow.

```
METER(Pump101.FlowTotal)
```

Example 2

A common use of this function is for counters with rollover. The following expression returns the number of units to come out of a production line at one hour intervals.

```
METER(ProductionLine2.UnitCount, hour)
```

Example 3

The following expression returns the number of units to come out of all streams that include "UnitCount" in their tag name at one hour intervals.

```
METER(SELECT("TagName","UnitCount"), hour)
```

MIN/MAX

The MIN() and MAX() functions produce a stream of results selecting the minimum or maximum value of a stream or array of streams. The returned timestamps correspond to the minimum/maximum values retrieved, rather than the beginning of the time interval.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

When using MIN() or MAX() in a query, if no resolution is specified, the behavior is the same as minimum or maximum retrieval modes with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the minimum/maximum values are calculated for each cycle.

Syntax

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the minimum (or maximum) value of a stream or array of streams.

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*, *TimeInterval*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which minimum or maximum values are calculated.
Description	Returns the minimum (or maximum) value of a stream or array of streams over the specified time interval.

MIN/MAX Examples

MIN/MAX Examples

Example 1

The following expressions return the minimum (or maximum) flow rate of a pump.

```
MIN(Pump101.Flow)
MAX(Pump101.Flow)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'MIN(Pump101.Flow)'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Result:

DateTime	Value	OPCQuality
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 2

The following expressions return the minimum (or maximum) flow rate of a pump over 30 minute time intervals.

```
MIN(Pump101.Flow, 30 minute)
MAX(Pump101.Flow, 30 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'MIN(Pump101.Flow, 30 minute)'
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-03-30 00:13:45.2590000	592.065856933594	192
2022-03-30 00:43:07.2590000	592.34765625	192
2022-03-30 01:01:07.2590000	593.351257324219	192
...	...	...
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 3

The following expression returns the minimum flow rate of all streams measured in the volumetric flow dimension over 30 minute time intervals.

```
MIN(SELECT("Dimension","Volumetric Flow"), 30 Minute)
```

PREV/PREVGOOD

The PREV() and PREVGOOD() functions return the last value from the previous time period for the provided stream.

When using PREV() or PREVGOOD() in a query, if no resolution is specified then the query returns an error.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **PREV**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns the last value from the previous time period, where the period is specified by *TimeInterval*.

Format	*Stream* = **PREVGOOD**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns similar results to **PREV**(), but values with "bad" or "uncertain" quality are skipped. **PREVGOOD** will continue seeking backwards until a "good" value is found, going beyond the previous period if necessary. This means **PREVGOOD** always returns a value with "good" quality.

PREV/PREVGOOD Examples

PREV/PREVGOOD Examples

Example 1

The following expression

```
Line1.Total - PREV( Line1.Total, hour )
```

TOTAL

The TOTAL() function produces a stream of results calculating the time-weighted total for a stream or array of streams.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the time-weighted total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated.
Description	Calculates the time-weighted total for a stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

TOTAL Examples

TOTAL Examples

Example 1

The following expression returns the total volume flowing through a pump over 1 hour intervals.

```
TOTAL(Pump101.Flow, hour)
```

Example 2

The following expression returns the total volume of all streams measured in the volumetric flow dimension over

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

1 hour intervals.

```
TOTAL(SELECT("Dimension","Volumetric Flow"), hour)
```

Example 3

The following expression returns the total energy produced by a turbine over the query period.

```
TOTAL(Turbine306.Output)
```

Example 4

The following expression returns the total energy for all streams with a tag name that starts with "Turbine" and ends with ".Output" over the query period.

```
TOTAL(SELECT("TagName","Turbine*.Output"))
```

TOTALS

The TOTALS() function produces a stream of results calculating the statistical total for a stream or array of streams.

Syntax

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the statistical total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated
Description	Calculates the statistical total for a stream or array of streams over the specified time interval

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

Retrieval Styles for Trend

The Trend application allows you to use "retrieval styles" that automatically switch retrieval modes for trend tags based on the trend duration and/or tag type. For example, you could configure a style that uses delta retrieval for short time periods and best-fit retrieval for longer periods. This helps you to balance response speed and accuracy. Also, retrieval styles allow you to calculate moving averages and retrieve data from the Historian's summary tables.

Working with Retrieval Styles

Retrieval styles cover the following retrieval options:

- Retrieval mode
- Resolution for cycle-based modes (as time duration or number of pixels per cycle)
- Data retrieval source (history or summary tables)
- Moving average calculation
- State calculation for ValueState retrieval

If you want to use additional retrieval options, such as deadbands, row limits or a quality rule, you must specify them in the Trend application at the application and/or tag level. For more information, see Configuring Retrieval Options and Configuring Trend Options for a Tag.

You can use retrieval styles at the application and/or tag level. When you specify a retrieval style at the application level, that style is used for all tags that do not have a different style specified.

The Trend application comes with various predefined styles that you can use. For a description of each style, see Using the Standard Retrieval Styles. Alternatively, you can define your own retrieval styles to suit your needs. For more information, see Location and Structure of Retrieval Styles and Creating and Editing Retrieval Styles.

Location and Structure of Retrieval Styles

Retrieval styles are stored at the application level in the following XML file:

`C:\ProgramData\ArchestrA\ActiveFactory\Trend\RetrievalStyles.xml`

After you add or edit a style in this file, it is available to all users of the Trend application on the system. You can edit the file in any text or XML editor. Note the following requirements:

- You must save the file in UTF-8 encoding.
- Names and values are case-sensitive. If any name or value is misspelled, the Trend application may hang on startup.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Structure of the Retrieval Styles File

The retrieval styles file has the following structure:

```
XML header
Style collection
   Retrieval style 1
      Style names for different locales
      Duration range 1
         Retrieval type 1
         ...
         Retrieval type n
      Duration range 2
      ...
      Duration range n
   Retrieval style 2
   ...
   Retrieval style n
End of style collection
```

That is:

- The file contains exactly one style collection, represented by the `styleCollection` XML element. For more information, see styleCollection XML Element.
- The style collection contains one or more retrieval styles, represented by the `retrievalStyle` XML element. Each of these represents a style that is available for use in the Trend application. For more information, see retrievalStyle XML Element.
- Each retrieval style contains one or more duration ranges, represented by the `duration` XML element. A duration range defines the retrieval types that are to be used for trend queries whose duration is within a specified range. Duration ranges should be arranged in descending length. For more information, see duration XML Element.
- Each duration range contains one or more retrieval types, represented by the `retrieval` XML element. The retrieval type defines the retrieval options that are to be used for tags of a certain type. For more information, see retrieval XML Element.

The retrieval type that gets used for a given tag is determined as follows:

1. You select a retrieval style in the Trend application.
2. The Trend application searches that retrieval style for a duration range that covers the duration of your trend. This would be the first duration range with a time period that is shorter than the trend duration.
3. When it has found a suitable duration range, it searches that duration range for a retrieval type that suits the type of the tag.

A simple example file might look like this:

```
<?xml version="1.0" encoding="utf-8" ?>
   <styleCollection version="9.2" xmlns="urn:retrievalstyle-schema">
      <retrievalStyle server="InSQL" minVersion="8.0" maxVersion="9.0" enabled="true">
         <styleName locale="en">My style</styleName>
         <styleName locale="ja">My style in Japanese</styleName>
         <styleName locale="zh-CN">My style in Chinese</styleName>
         <styleName locale="de">My style in German</styleName>
         <styleName locale="fr">My style in French</styleName>
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      <duration minSpan="P0Y0M1DT0H0M0S">
         <retrieval tagType="Discrete" source="History" retrievalMode="Delta"
         stateCalc="*" resolution="0" pixels="0" movingAverageValues="0" />
         <retrieval tagType="All" source="History" retrievalMode="Cyclic" stateCalc="*"
         resolution="0" pixels="5" movingAverageValues="0" />
      </duration>
      <duration minSpan="P0Y0M0DT0H0M0S">
         <retrieval tagType="All" source="History" retrievalMode="Delta" stateCalc="*"
         resolution="0" pixels="0" movingAverageValues="0" />
      </duration>
   </retrievalStyle>
</styleCollection>
```

In this case, the file only defines one style named `My style`. When querying two days of data for a discrete tag using this style, delta retrieval is used (the first `retrieval` element in the first `duration` element). For an analog tag, cyclic retrieval with one cycle for every five pixels of the trend width is used instead (the second `retrieval` element in the first `duration` element). For queries that are shorter than one day, delta retrieval is used regardless of the tag type (the only `retrieval` element in the second `duration` element).

Creating and Editing Retrieval Styles

To create or edit retrieval styles, you edit the retrieval styles file in a text or XML editor. Read Location and Structure of Retrieval Styles first to get an overview of how this file is structured.

The following procedure provides you the basic steps to add a new style. For details on each XML element, refer to the corresponding subsection.

To create a new style:

1. Under the `styleCollection` element, add a `retrievalStyle` element.
2. Under the `retrievalStyle` element, add a `styleName` elements for each locale in which you want to use the style. For more information, see retrievalStyle XML Element.
3. Decide at which trend durations you want to switch retrieval options. Under your `retrievalStyle` element, add `duration` elements for each of these "switching points." For more information, see duration XML Element.
4. For each `duration` element, add `retrieval` elements as needed to define retrieval types. For more information, see retrieval XML Element.

Retrieval Style XML Elements

The following sections describe each of the XML elements in the retrieval styles file. For information on how they fit together, see Location and Structure of Retrieval Styles.

styleCollection XML Element

The `styleCollection` element represents a collection of retrieval styles. It is the container for multiple retrieval styles represented by `retrievalStyle` elements. It has two required attributes:

- `version`: Specifies the format version of the style collection. The only valid value is `9.2`.
- `xmlns`: Specifies the XML namespace to be used. Set this attribute to `urn:retrievalstyle-schema`.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The retrieval styles file can only contain single `styleCollection` element.

retrievalStyle XML Element

The `retrievalStyle` element represents single retrieval style. It is the container for two other elements:

- `styleName`: Specifies the name of the style for the locale specified by the `locale` attribute. This is the name by which you can access the style when the Trend application runs under the specified locale.

  You can specify the locale as a two-character ISO language code or a four-character combination of language code and country code. If you specify a name for a two-character locale, it is used for all sub-locales that do not have a separate name defined. For example, if you specify a name for the `de` locale, it is used for the `de-DE`, `de-AT` and `de-CH` locales unless you specify separate names for those locales.

  You must specify a `styleName` element for all styles that you want to use in a given locale. If a style does not have a name defined for a locale, the Trend application does not show it when running under that locale. The only exception is when you run the Trend application under a locale for which no style names are defined at all. In that case, the styles are shown with their names for the `en` locale.
- `duration`: Specifies a duration range. For more information, see duration XML Element.

It has three required attributes:

- `server`: Specifies the server type for which the style can be used. Always set this attribute to `InSQL`.
- `minVersion`: The minimum Historian version that the retrieval style can work with, either `8.0` or `9.0`. If the Trend application is connected to a Historian whose version is lower than the version specified here, the style is not used.

  Specify `9.0` if your style uses functionality that is not supported for AVEVA Historian 8.0.
- `enabled`: Specifies whether the style is active. To temporarily disable the style, set this attribute to `false`.

It has one optional attribute:

- `maxVersion`: The maximum Historian version against which the retrieval style can be used. This attribute is not currently used.

duration XML Element

The `duration` element represents a duration range. It contains the retrieval types that are used when the trend period is longer than the time period it specifies.

A retrieval style can contain any number of `duration` elements. However, you should arrange these elements in descending length. This is because the Trend application uses the first suitable duration range that it finds, that is, the first duration range with a time period shorter than the current trend period.

For example, assume you have three duration ranges defined in the following order:

- 1 day
- 4 hours
- 0 seconds

For a query with a duration of 2 days, the Trend application uses the retrieval types defined for the "1 day"

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

duration range because it is the first range whose time period is shorter than 2 days. Now assume the same duration ranges are ordered like this:

- 4 hours
- 1 day
- 0 seconds

In this case, the Trend application uses the retrieval types defined for the "4 hours" duration range because again, it is the first range whose time period is shorter than 2 days. The more suitable "1 day" duration range is ignored.

Note: You should always define a duration range with a time period of 0 seconds. This serves as a "catch-all" for trend periods that aren't covered by any other duration range.

The `duration` element has one required attribute:

- `minSpan`: Specifies the time period as a standard XML duration value, for example, `P0Y0M1DT0H0M0S`. The number to the left of `Y` represents the number of years, the number to the left of `M` represents the number of months, and so on (`D` = days, `H` = hours, `M` = minutes, `S` = seconds). `P` and `T` are separator characters.

It is the container for one other element:

- `retrieval`: Specifies a retrieval type. For more information, see retrieval XML Element.

retrieval XML Element

The `retrieval` element represents a retrieval type, that is, a set of retrieval options for a certain type of tag.

You can have multiple `retrieval` elements in a duration range. However, you should order them from the most specific to the least specific one. This is because the Trend application uses the first suitable retrieval type that it finds, that is, the first retrieval type with a matching tag type and available history source.

For example, assume that you have three retrieval types defined in the following order:

- Analog tags, Summary data
- Analog tags, History data
- All tags, History data

For an analog tag, the Trend application first tries to retrieve summary data according to the first retrieval type. If no summary data is available, it retrieves history data according to the second retrieval type. Now assume the retrieval types are ordered like this:

- Analog tags, History data
- Analog tags, Summary data
- All tags, History data

In this case, the Trend application never tries to retrieve summary data for an analog tag; it never considers the second retrieval type because it has already found a suitable retrieval type in the first one.

You should always define a retrieval type with a tag type of "All" and a history source of "History." This serves as a "catch-all" for tags that aren't covered by any other retrieval style.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The `retrieval` element has seven required attributes:

- `tagType`: Specifies the tag type for which the retrieval type should be used. Valid values are `All`, `Analog`, `Discrete`, and `String`.
- `source`: Specifies the history source from which to retrieve data. Valid values are `History` to retrieve data from history tables and `Summary` to retrieve data from summary tables. When using `Summary`, you must specify the summary frequency in the `resolution` attribute.
- `retrievalMode`: Specifies which retrieval mode to use. Valid values are `Cyclic`, `Delta`, `Full`, `Interpolated`, `BestFit`, `Average`, `Min`, `Max`, `Integral`, `Slope`, `Counter`,`ValueState`, and `RoundTrip`. Make sure that you specify a retrieval mode that is supported for the specified tag type. For example, Counter retrieval does not work with string tags. Therefore, if you try to retrieve data for a string tag in Counter mode, the Historian does not return any data.

  For information on each mode, see Understanding Retrieval Modes.
- `stateCalc`: Specifies which state calculation to use in ValueState and RoundTrip retrieval. Valid values are `Min`, `Max`, `Average`, `Total`, and `Percent`. For more information, see State Calculation (wwStateCalc). If you are not using ValueState retrieval, specify an asterisk (`*`).
- `resolution`: Specifies the retrieval resolution in milliseconds when retrieving history data in cycle-based retrieval modes, or the summary frequency in seconds when retrieving summary data. For more information, see Resolution (Values Spaced Every X ms) (wwResolution)

  Alternatively, you can set this attribute to 0 and specify a retrieval resolution using the `pixels` attribute.
- `pixels`: Specifies the retrieval resolution for cycle-based retrieval modes as the number of pixels per cycle. The number of cycles is the width of the trend chart divided by the value of this attribute. For example, if the chart is 500 pixels wide and the `pixels` attribute is set to 5, then 100 cycles are used.

  Alternatively, you can set this attribute to 0 and specify a retrieval resolution using the `resolution` attribute. If you specify non-zero values for both the `pixels` and the `resolution` attributes, `resolution` takes precedence.
- `movingAverageValues`: Specifies whether to calculate moving averages when retrieving history data. If set to 0, no moving averages are calculated. Otherwise, moving averages are calculated using the number of values specified in this attribute.

Using the Standard Retrieval Styles

The following table describes the standard retrieval styles available in the Historian Client Trend application.

Style name	Description
BestFit-5	Best Fit retrieval with one cycle per five pixels.
BestFit-10	Best Fit retrieval with one cycle per ten pixels.
BestFit-15	Best Fit retrieval with one cycle per 15 pixels.
Cyclic (ActiveFactory 9.1)	Cyclic retrieval with one cycle per two pixels.
Integral (ad hoc)	Integral retrieval for queries longer than 15 minutes. Resolution depends on query duration. Best-fit retrieval with one cycle per ten pixels for queries

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Style name	Description
	shorter than 15 minutes.
Averages (From Summaries or Ad Hoc)	Tries to retrieve analog averages from summary tables. If no summary data is available, uses Average retrieval for analog tags and ValueState retrieval for discrete tags. Resolution depends on query duration.
Averages (ad hoc)	Average retrieval for analog tags and ValueState retrieval for discrete tags. Resolution depends on query duration.
Summary (InSQL 8.0)	Tries to retrieve analog averages from summary tables for queries longer than 15 minutes. If no summary data is available, uses Cyclic retrieval with one cycle per pixel for queries longer than 8 hours and Delta retrieval for shorter queries.
Counter-20	Counter retrieval with one cycle per 20 pixels.
Counter on round periods	Counter retrieval with cycles at even time periods (one minute, one hour, etc. depending on the resolution).
Time In State (percent)	ValueState retrieval with percent calculation for queries longer than one minute. Resolution depends on query duration. Full retrieval for shorter queries.
Time In State (Avg)	ValueState retrieval with average calculation for queries longer than one minute. Resolution depends on query duration. Full retrieval for shorter queries.
RoundTrip (PercentContained)	RoundTrip retrieval with percentcontained calculation for queries longer than one minute. Resolution depends on query duration. Full retrieval for shorter queries.
RoundTrip (AvgContained)	RoundTrip retrieval with averagecontained calculation for queries longer than one minute. Resolution depends on query duration. Full retrieval for shorter queries.
MovingAverage (12-5 sec)	Moving averages for analog tags with 12 values and a resolution of five seconds. Delta retrieval for all other tags.
MovingAverage (30-1 sec)	Moving averages for analog tags with 30 values and a resolution of one second. Delta retrieval for all other

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Style name	Description
	tags.
MovingAverage (10-1 pixel)	Moving averages for analog tags with 10 values and a resolution of one cycle per pixel. Delta retrieval for all other tags.

Retrieval Styles, Application Settings, and Tag Settings

There are various ways to set retrieval options in the Trend application: using a retrieval style vs. using custom retrieval options, using the application-wide options vs. using tag-level options. Also, there are some differences depending on which Historian version you are using. The following guidelines help you understand which retrieval settings are actually used in a given situation.

1. Settings at the tag level override settings at the application level. For the aaHistClientTrend control, this means that the properties starting with `CurrentTag...` override the properties starting with `RetrievalOptions...`
2. Because a retrieval style definition can contain multiple sets of retrieval settings with different retrieval modes, some of the settings available for editing at the application or tag level may turn out to be irrelevant for the retrieval mode that actually gets used for a given query. However, because there is no way to know in advance which retrieval mode will be used, the settings are still available for editing. The same applies to properties in the aaHistClientTrend control.
3. At the application level, you can specify additional options for retrieving data from Historians with a version earlier than 9.0. For more information, see Configuring Other Options. These settings override any style that you may have selected at the application level. For example, if you have set these options to enable delta retrieval for periods below 15 minutes, but you have selected a style at the application level that specifies cyclic retrieval for all time periods, the Trend application enforces delta retrieval for all time periods below 15 minutes regardless of the settings in the style.

   However, if you select the style at the tag level, then the style settings override the application options. In the above example, cyclic retrieval would be used for all time periods regardless of the application settings specifying delta retrieval.
4. If there is a conflict between a setting specified in a style and a setting specified using one of the aaHistClientTrend control's properties (for example, retrieval resolution), the style setting overrides the property setting.

Historian Client Web User Guides

This section contains guides for Historian Client Web. There are separate guides for the modern look and classic look.

Historian Client Web Classic

This guide describes the features and functionality of Historian Client Web in the classic look.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Logging in to Historian Client Web

When you open Historian Client Web, you may be asked to log in. Log in with your SSO or Windows credentials. If your environment is using the connected experience, logging in with Windows integrated login will only give you access to demo mode. Demo mode has a 5 minute limit.

Chart your data

With AVEVA Historian Client Web, you can search for specific tags and see them in a single chart view.

From here, you can choose the chart type, add or remove tags from the chart, and fine-tune how the content is displayed. You can also save and share your content.

1. Add search criteria.
2. The Search Results are categorized by unit of measure on the left. Choose a category to work with.

   Note: Units of measure are case-sensitive. That means ml (milliliters) and ML (megaliters) are not identified as the same measurement.

3. Your data displays in the main area. The icon next to the chart name indicates whether the chart is shared. Select the icon to display the chart's owner.
4. The Gallery on the right lets you change the chart style and select the data tags you want to display.
5. Your data displays in the main area.
6. From the dropdown below the chart ("Showing the..."), choose a data retrieval mode and frequency (daily or hourly).
7. At the bottom of the screen, you can refine the timeframe for your data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. Select the share icon to save, share, download, or embed your data analysis.

Search for data or content

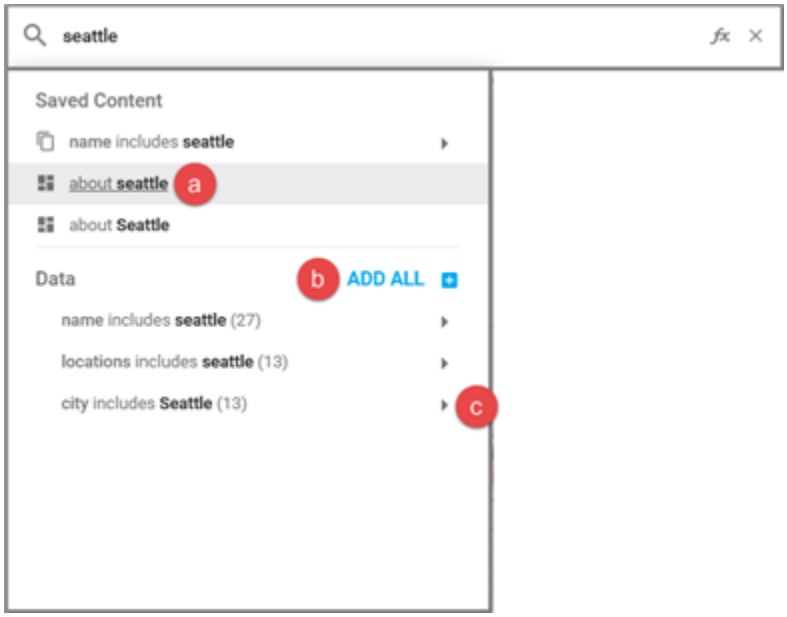
From AVEVA Historian Client Web, you can search for specific information about your operations and choose how to display it.

To search for data or content

1. In the search bar, enter the term(s) for the data you want to view.

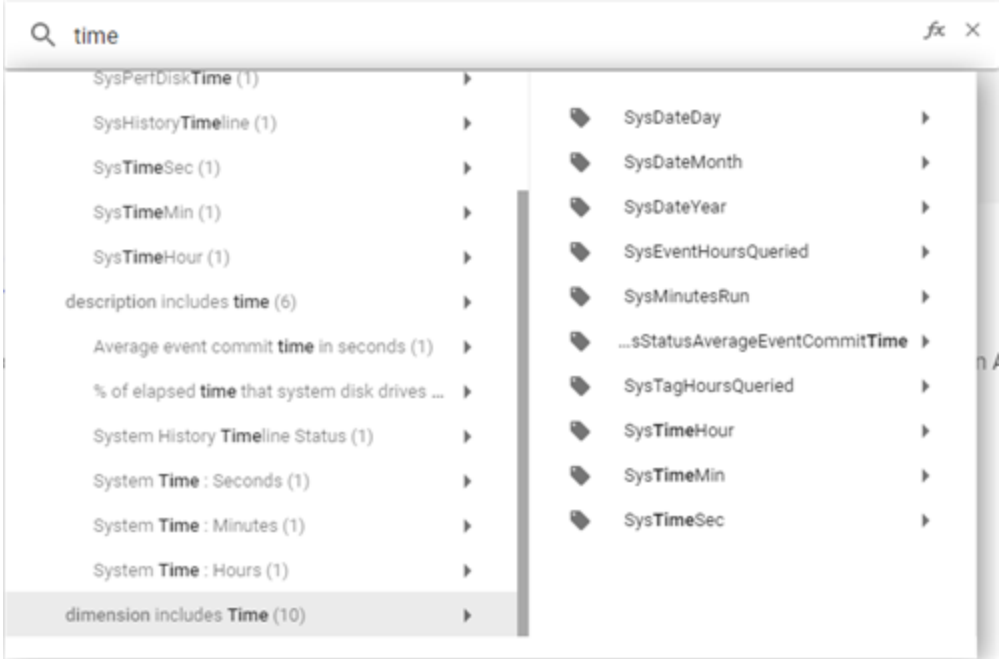
   For example, suppose you wanted data related to your Seattle operation. You could type "seattle" to see saved content and data related to this search criteria.

   The search results list shows Saved Content (including a list of matching saved content, saved dashboards, and keywords) and Data (including a list of tags with names or descriptions containing your search term).



   Search results also list tags with a matching extended property value. (Extended property values are additional searchable categories and terms added to tags by an administrator.) For example, you could type "time" to find a list of tags assigned to the Time dimension.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Note: A string made up of only special characters (such as "$$$$%%") is not searchable.

2. Select items from the dropdown list to include them in the operational data you want to view.

Select this...	To display this...
a. Underlined text	All data tags described by that line.
b. **Add All**	Everything in the category.
c. >	The source and tagname for a given result. Then select > next to a specific tag and select **Add Tag**.

3. Repeat steps 1 and 2 until you have all the operational data you want to view.

Your data displays in a single chart view.

Chart data with expressions

Expressions allow you to perform mathematical and logical operations on tag data.

Note: Using expressions requires the "advanced" license feature to be enabled. If the advanced license is not available when an expression is used, no data is retrieved and an error message is displayed.

What is an expression?

An expression is made up of a combination of the following elements:

1. Values
   - A typical value is made up of three components:
     - A value with a base type of Boolean, integer, or double
     - A date and time in UTC format

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

    - A 16-bit integer representing the OPC quality of the value
  - Literal values are also supported, which have no date and time associated with them, and are always assumed to be "good" quality.
2. Streams
  - A stream is a time-ordered collection of values. A stream has either a stair-step or linear interpolation model, and an associated engineering unit.
  - In the context of an expression, one tag is the initial stream, but the expression can produce additional streams derived from a tag. For example, the expression `TI101.PV + 4.0` produces a result stream, but the initial stream is the stream associated with `TI101.PV`.
3. Arithmetic and logical operators
  - Standard arithmetic operators such as addition, subtraction, multiplication and division are supported, along with logical operators for comparison, such as greater than (>) and less than (<). For a full list of supported operators, see Supported arithmetic operators.
4. Scalar functions
  - A variety of scalar functions are supported for manipulating data, such as calculating the absolute value, rounding values to the nearest integer, or unit of measure conversion. For a full list of supported scalar functions, see Supported scalar functions.
5. Time-series functions
  - A variety of time-series functions are supported for comparing data, such as calculating minimum/maximum values, or time in state. For a full list of supported functions, see Supported time-series functions.
  - Time intervals can be expressed in either of two ways:
    a. As a number of days. For example, a value of 1 represents a daily period, while a value of .125 represents a period of three hours.

    Note: Daily time periods support Daylight Saving Time. This means the "1 day" period is 24 hours, except on days when the Daylight Saving Time change is made; on those days, it will be either 23 or 25 hours.

    b. As a named period. The following named periods are supported: *minute*, *hour*, *day*, *week*, *month*, *fixedday*, *fixedweek*, *fixedmonth*. These periods are anchored to a rounded time period. For example, "hour" produces hourly intervals where the minutes, seconds and milliseconds are all 0 in local time, while "minute" produces intervals where the seconds and milliseconds are all 0 in local time.

    Note: The *day*, *week* and *month* named periods support Daylight Saving Time, while the *fixedday*, *fixedweek* and *fixedmonth* named periods do not. For example, the length of a *day* period can be 23, 24, or 25 hours, depending on whether a Daylight Saving Time transition takes place. The length of a *fixedday* period is always 24 hours, *fixedweek* is always 7 days times 24 hours, and *fixedmonth* is always the number of calendar days in the month times 24 hours.

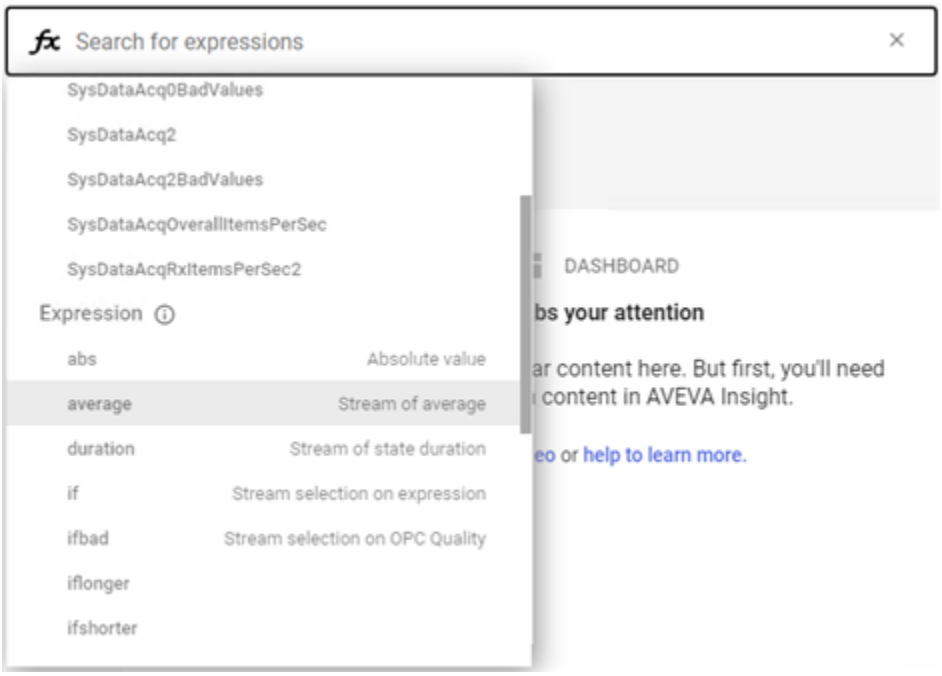
To enter an expression:

1. In the search bar, select the fx icon to switch to expression mode.

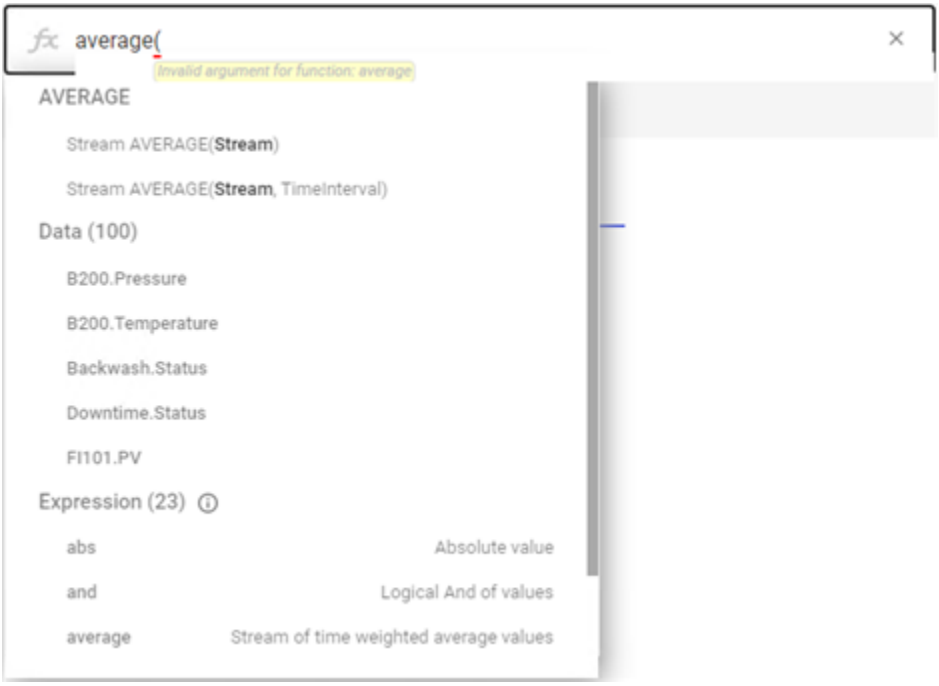
Note: If you do not see the fx icon, either your system does not have the advanced feature license activated, or your administrator has disabled expression mode.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. In this mode, the search results list displays suggestions for completing expressions, in addition to content and data matching your search term. Enter your expression by typing into the search bar. You can also select an item from the list of suggestions to insert it into your expression.



3. As you enter your expression into the search bar, AVEVA Historian Client Web recognizes the use of functions, and offers you suggestions for completing the function by showing you the various options available to you as you type. Other suggestions can include tag names, existing chart data, or function names, depending on the context of your expression as you build it.



4. When the expression is complete, press Enter. If the expression is valid, it is added to your chart.

Note: The case-sensitivity of tag names in expressions depends on the collation setting of the SQL Server,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

while function names are always case-insensitive. AVEVA Historian Client Web prevents you from adding the same expression to your chart multiple times. For example, if the SQL Server's collation is not case-sensitive, the two expressions `SysTimeMin + SysTimeHour` and `systimemin + systimehour` are considered identical, so the second one cannot be added after the first. If the server's collation is case-sensitive, both expressions can be added to your chart. As another example, the two expressions `AVERAGE(Pump1.FlowRate)` and `average(Pump1.FlowRate)` are considered identical because function names are case-insensitive.

5. To return to standard search mode, select ✕ in the search bar.

Edit an expression

You can edit an expression after you have added it to your chart.

To edit an expression:

1. In the legend area of your chart, locate the expression you wish to edit and select its **Edit Expression** icon . The Edit Expression dialog displays.

2. Modify the expression by editing the text in the **Expression** field.
3. A unique **Alias** for the expression is automatically created when you first add the expression to your chart. Update the text in the Alias field to change the expression's alias.
4. Select **SAVE** to update the expression with your changes, or **CANCEL** to keep the original values.

Implicit unit conversion

The following table describes the relationships between engineering unit dimensions when they are multiplied or divided by each other.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Factor1	Factor2	Product
Data Rate	Time	Data
Density	Volume	Mass
Energy Density	Volume	Energy
Length	Area	Volume
Length	Length	Area
Mass Flow	Time	Mass
Pressure	Area	Force
Pressure Rate	Time	Pressure
Resistance	Current	Electric Potential
Speed	Time	Length
Temperature Rate	Time	Temperature
Unknown Rate	Time	Unknown
Volumetric Flow	Time	Volume
Radiant Exposure	Time	Irradiance
Rate	Time	Ratio
Rotation Speed	Time	Rotation
Electric Potential	Power	Current
Acceleration	Mass	Force
Force	Speed	Power
Mass	Specific Energy	Energy

Each row in the table corresponds to four equivalent dimension relationships:

- *Dimension 1 * Dimension 2 = Result Dimension*
- *Dimension 2 * Dimension 1 = Result Dimension*
- *Result Dimension / Dimension 1 = Dimension 2*
- *Result Dimension / Dimension 2 = Dimension 1.*

When working with expressions involving tags or streams with different engineering units, the engineering units of the results are determined as follows:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

1. When adding or subtracting streams with units in the same dimension, all streams are first converted to the same unit as the first stream, and then the operations are performed. The results are in the same unit as the first stream. For example, using the expression `SysTimeHour + SysTimeMin`, both tags use units in the *Time* dimension. When the expression is evaluated, the units from the first tag (hours) are used, so `SysTimeMin` is first converted to hours, and the results are in hours.
2. When adding or subtracting streams with one or more streams having "undefined" units, the results are in the same unit as the first stream.
3. When multiplying or dividing a dimensionless stream or numeric constant with a stream that has an associated unit, the results are in that unit.
4. When multiplying or dividing streams where both units are related by one of the entries in the *DimensionMath* table, the result is in the associated result dimension's unit.
5. When multiplying streams where both units are in the same dimension, the scale factors of the two units are multiplied together. If the scale factor matches a unit in the result dimension, the results are in this unit. Otherwise, the original units are first converted to their dimension's base units, and the results are in the base units of the result dimension.
6. Any other combinations cannot be implicitly converted, so the results are in "undefined" units.

Note: AVEVA Historian 2023 updates the base units for some dimensions to enable implicit unit conversion. If you have additional Historian nodes configured for replication, the engineering unit catalog will be out of sync on tier 2 nodes until they are upgraded to the 2023 version.

Interpolation with Expressions

When plotting data with expressions, the system determines the interpolation type for the expression based on the interpolation type of the individual tags in the expression.

If the interpolation type in the tag metadata is linear then linear interpolation is used, otherwise stair-step interpolation is used. For more information, see *Interpolated Retrieval* in the *AVEVA Historian Retrieval Guide*.

Special rules apply when using some expression functions:

1. An expression using the SIGN function uses stair-step interpolation.

   For example, Pump101.Flow is an analog tag representing the flow rate for a pump. While the tag uses linear interpolation, the expression `SIGN(Pump101.Flow)` uses stair-step interpolation.
2. An expression using the ROUND function uses stair-step interpolation.

   For example, Tank100.Level is an analog tag representing the level of fluid in a tank. While the tag uses linear interpolation, the expression `ROUND(Tank100.Level)` uses stair-step interpolation.

Selecting an array of streams

Some functions allow you to specify an array of multiple streams. This is supported for the following functions:

- AVERAGE
- AVERAGES
- METER
- MIN
- MAX

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- TOTAL
- TOTALS

The array of streams is defined by a common property. The following properties are supported for defining an array:

- Location
- Dimension
- TagName

The general format for selecting an array of streams is SELECT ("*PropertyName*", "*PropertyValue*"). Supported wildcards vary depending on the property value:

- **Dimension:** Wildcards are not supported.
- **TagName:** The ***** wildcard is supported. This matches any number of characters.
- **Location:** The following MQTT wildcard characters are supported:
  - +: Replaces one topic level
  - #: Matches any number of topic levels either before or after, depending on where it's placed.

The specific syntax for each property is described below.

Syntax

Format	SELECT("Location", "*Location Name*")
Arguments	*Location Name* - A location or part of a location. This will match any locations that include the given argument. So "Sea" and "eattl" will both match "Seattle".
Description	Returns an array of streams that include the given argument in their location.

Format	SELECT("Dimension", "*Unit Dimension*")
Arguments	*Unit Dimension* - Any unit dimension that is in the Engineering Units Catalog.
Description	Returns an array of streams that match the given unit dimension.

Format	SELECT("TagName", "*Tag Name*")
Arguments	*Tag Name* - A tag name or part of a tag name. This will match any tag names that include the given argument.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	So "SysTime" and "sTime" will both match "SysTimeHour", "SysTimeMin", and "SysTimeSec".
Description	Returns an array of streams that include the given argument in their tag name.

Example 1

The following expression returns the statistical total for all streams that have a location that includes the string "Toronto" over 1 week intervals.

```
TOTALS(SELECT("Location","Toronto"), 1 week)
```

Example 2

The following expression returns the statistical average for all streams measured in a temperature dimension over 30 minute intervals.

```
AVERAGES(SELECT("Dimension","Temperature"), 30 Minute)
```

Example 3

The following expression returns the highest value for all streams with a tag name that includes the string "boiler" over 1 day intervals.

```
MAX(SELECT("TagName","boiler"), 1 day)
```

Supported arithmetic operators

AVEVA Historian Client Web supports the use of standard mathematical operators in expressions. The following operators are supported:

Symbol	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
%	Modulo
( )	Operator grouping

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Arithmetic Operator Examples

The following examples show expressions that use arithmetic operators. The tag names used in these examples are:

- *Pump1.FlowRate* measures the flow rate of a pump in liters per second (L/s)
- *Pump2.FlowRate* measures the flow rate of a pump in liters per minute (L/min)
- *TankCapacity* measures the available capacity of a tank in kiloliters (kL)
- *TimeElapsed* measures elapsed time in seconds (s)

Example 1

The following expression returns the sum of flow rates for two pumps.

```
Pump1.FlowRate + Pump2.FlowRate
```

Because the tags use different units, and there is no explicit unit conversion specified, the units of *Pump2.FlowRate* are implicitly converted to liters/second to match the units of *Pump1.FlowRate* before the values are added together.

Example 2

The following expression returns a value per minute representing the total volume flowing through both pumps during that minute.

```
TOTAL(Pump1.FlowRate + Pump2.FlowRate, 1 minute)
```

Example 3

The following expression calculates a pump's efficiency as a percentage. This example uses water at 50 degrees Celsius, which has a density of 992.24 kg/m^3 and head of 3.5m, the gravitational constant of 9.81 m/s^2, and "Pump1.Power" represents the power used by the pump in Watts.

```
Pump1.FlowRate * 3.5 * 992.24 * 9.81 / 100.0 / Pump1.Power
```

Example 4

The following expression returns the travel time in seconds for material in a pipe. The pipe has a diameter of 0.15m, and a length of 100m.

```
3.1415 * 0.15 ^ 2.0 * 100 / UOM(Pump1.FlowRate, m3/sec)
```

Supported logical operators

AVEVA Historian Client Web supports the use of standard mathematical operators in expressions. The following operators are supported:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Symbol	Operation
=	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

Supported logical functions

In addition to logical operators, AVEVA Historian Client Web also supports the use of logical functions in expressions. The following logical functions are supported:

Function	Usage
AND(*arguments*)	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.
OR(*arguments*)	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.
NOT(*argument*)	Evaluates the argument to a Boolean *TRUE* or *FALSE* value, and returns the opposite.

AND

Format	*Stream* = **AND**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.

OR

Format	*Stream* = **OR**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if any arguments evaluate to *TRUE*, otherwise

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	FALSE.

NOT

Format	*Stream* = **NOT**(*expression*)
Arguments	Any expression.
Description	Evaluates the expression to a boolean *TRUE* or *FALSE* value, then returns the opposite.

Supported scalar functions

AVEVA Historian Client Web supports using scalar functions in expressions. These functions have a 1:1 relationship between the values in the input and output streams.

The following scalar functions are supported:

Function	Purpose
ABS	Calculates the absolute value of a stream.
CASTUOM	Changes a stream's unit of measure without changing the values.
IF	Produces a stream based on the results of a logical expression.
IFBAD	Overrides "bad" quality values in a stream.
IFLONGER	Evaluates whether or not a logical expression is true over a period greater than or equal to a specified time interval, and returns a TRUE or FALSE value or stream.
IFSHORTER	Evaluates whether or not a logical expression is true over a period shorter than a specified time interval, and returns a TRUE or FALSE value or stream.
IFSTALE	Replaces values from the input stream that are older than a specified interval with an alternative value.
LOG	Calculates the base 10 logarithm of a stream's values.
ROUND	Rounds a stream's values to two decimal places.
SIGN	Replaces a stream's values with 1 or -1 based on the current value's sign.
SQRT	Calculates the square root of a stream's values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TIMESHIFT	Shifts the time stamps in a stream by a specified time interval.
TRUNCATE	Removes the fractional portion of a stream's values.
UOM	Converts a stream's values to a specified unit of measure.

ABS

The ABS() function produces a stream of results calculating the absolute value of a stream, converting negative values into positive values.

Syntax

Format	*Stream* = **ABS**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all negative values converted to positive values.

ABS Examples

ABS Examples

Example 1

The following expression returns the difference between the flow rates of two pumps.

```
ABS( Pump101.Flow - Pump102.Flow )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( Pump101.Flow - Pump102.Flow )'
```

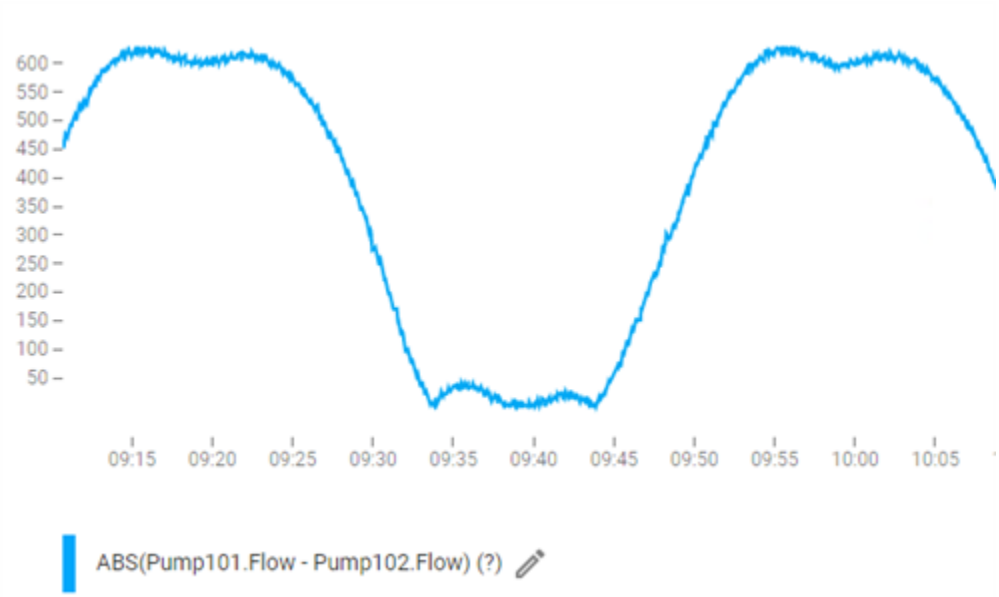
Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:07:51.4320000	259.439523422241	192
2022-03-31 09:07:53.2590000	268.941925048828	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2022-03-31 09:07:55.2590000	264.847595214844	192
...	...	...
2022-03-31 10:07:49.2590000	457.803894042969	192

Represented as a line chart over a 1 hour interval:



Example 2

The following expression produces a oscillating stream of values from 0 to 30, and back to 0:

```
ABS( SysTimeSec - 30 )
```

Represented as a query:

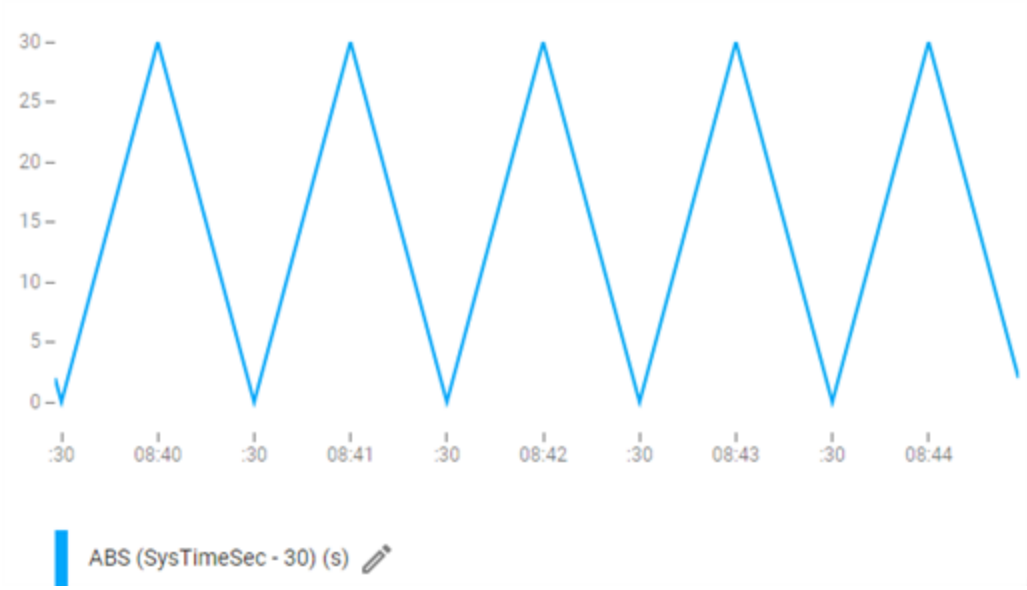
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( SysTimeSec - 30 )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:47:52.3500000	22	192
2022-03-31 09:47:53.0000000	23	192
2022-03-31 09:47:54.0000000	24	192
...	...	...
2022-03-31 10:47:51.0000000	21	192

Represented as a line chart over a 5 minute interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



CASTUOM

The CASTUOM() function changes a stream's unit of measure to the specified unit of measure without any data conversion, preserving the original values.

Syntax

Format	*Stream* = **CASTUOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the the provided stream to use the specified unit of measure while preserving the existing values.

IF

The IF() function produces a stream based on the results of a logical expression.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*, *FalseStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE *FalseStream* - A stream of values from which to select when the expression evaluates to FALSE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected for the output stream. Otherwise, the corresponding value from the *FalseStream* is selected.

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected for the output stream. Otherwise, a NULL or "bad quality" value is selected.

Format	*Stream* = **IF**(*LogicalExpression*)
Arguments	*LogicalExpression* - A logical expression to be evaluated
Description	Produces an output stream of Boolean values representing the results of *LogicalExpression*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IF Examples

IF Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, but any negative values are replaced with 0.

```
IF( Pump101.Flow < 0, 0, Pump101.Flow )
```

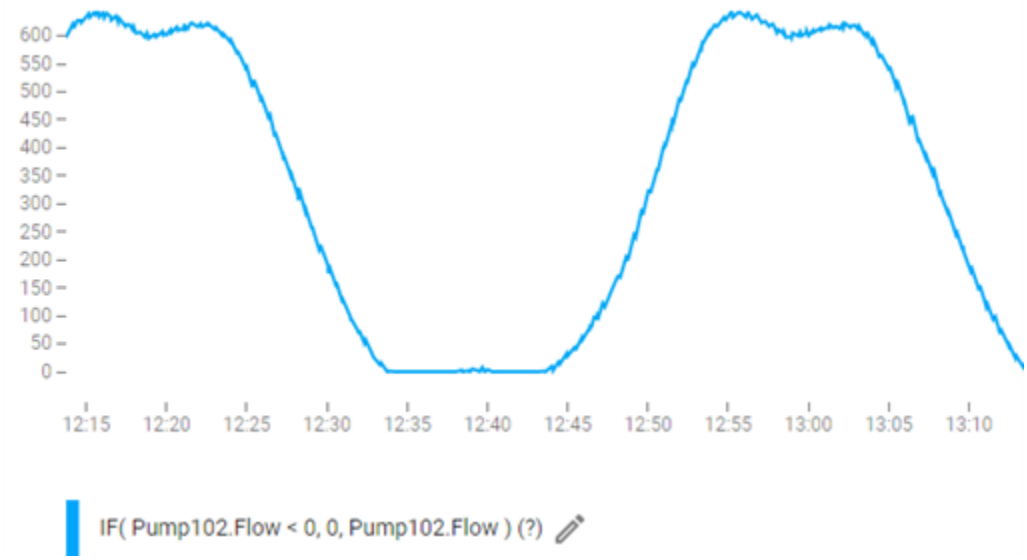
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'IF( Pump102.Flow < 0, 0, Pump102.Flow )'
```

Results over a 5 minute interval:

DateTime	Value	OPCQuality
2022-03-31 12:40:01.2590000	4.00229740142822	192
2022-03-31 12:40:03.2590000	0.365068227052689	192
2022-03-31 12:40:05.2590000	0	192
...	...	...
2022-03-31 12:44:59.2590000	27.5106182098389	192

Represented as a line chart over a 1 hour interval:



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns 0.

```
IF( SysPerfCPUTotal > 90, 1, 0 )
```

Example 3

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns a NULL value.

```
IF( SysPerfCPUTotal > 90, 1 )
```

Example 4

The following expression returns a true value (1) when system CPU usage exceeds 90%, otherwise it returns a false value (0).

```
IF( SysPerfCPUTotal > 90 )
```

IFBAD

The IFBAD() function replaces values from the input stream that have a bad OPC Quality with an alternative value.

Syntax

Format	*Stream* = **IFBAD**(*Stream*, *AltValue*)
Arguments	*Stream* - A stream of values *AltValue* - A replacement value
Description	Replaces every bad value in the input stream with *AltValue*.

Format	*Stream* = **IFBAD**(*Stream*, *AltStream*)
Arguments	*Stream* - A stream of values *AltStream* - A stream of alternative values
Description	Replaces every bad value in the input stream with the value from *AltStream* at the corresponding time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IFBAD Examples

IFBAD Examples

Example 1

The following expression replaces all bad values for the input stream Pump102.Flow with "500."

```
IFBAD(Pump102.Flow, 500)
```

Example 2

The following expression replaces all bad values in the input stream Pump102.Flow with values from the alternative stream Pump115.Flow at the corresponding time.

```
IFBAD(Pump102.Flow, Pump115.Flow)
```

IFLONGER

The IFLONGER() function evaluates whether or not a logical expression is true over a period greater than or equal to a specified time interval, and returns a TRUE or FALSE value or stream.

Syntax

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the boolean TRUE is returned. Otherwise the boolean FALSE is returned.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Interval - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. If it is not *TRUE*, a "bad quality" value is selected.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*, *FalseStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer *FalseStream* - A stream of values from which to select when the expression does not evaluate to *TRUE* for at least the length of the specified interval
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. Otherwise the corresponding value from *FalseStream* is selected.

IFLONGER Examples

IFLONGER Examples

Example 1

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period longer than or equal to 1 minute. If it does, it returns a value of TRUE. If it doesn't, it returns a value of FALSE.

```
IFLONGER(Pump1.FlowRate > 75.0, 1 minute)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression evaluates whether or not the stream SysTimeSec remains greater than or equal to 20 for a period longer than or equal to half a minute. If it does, it returns the value 5. If it doesn't, it returns a "bad quality" value.

```
IFLONGER(SysTimeSec >= 20, 0.5 minute, 5)
```

Example 3

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period longer than or equal to 1 minute. If it does, it returns the value from Pump1.Flowrate. If it doesn't, it returns a value of 75.

```
IFLONGER(Pump1.FlowRate > 75.0, 1 minute, Pump1.FlowRate, 75.0)
```

IFSHORTER

The IFSHORTER() function evaluates whether or not a logical expression is true over a period shorter than a specified time interval, and returns a TRUE or FALSE value or stream.

Format	*Stream* = **IFSHORTER**(*StreamLogicalExpression*, *Interval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for less time than the specified *Interval*, the boolean TRUE is returned. Otherwise the boolean FALSE is returned.

Format	*Stream* = **IFSHORTER**(*StreamLogicalExpression*, *Interval*, *TrueStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for less time than the length of the specified interval
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	TRUE for a period of time less than the specified *Interval*, the corresponding value from *TrueStream* is selected. If it is not *TRUE*, a "bad quality" value is selected.

Format	*Stream* = **IFSHORTER**(*StreamLogicalExpression*, *Interval*, *TrueStream*, *FalseStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for less time than the length of the specified interval *FalseStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for less time than the specified *Interval*, the corresponding value from *TrueStream* is selected. Otherwise the corresponding value from *FalseStream* is selected.

IFSHORTER Examples

IFSHORTER Examples

Example 1

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period shorter than 1 minute. If it does, it returns a value of TRUE. If it doesn't, it returns a value of FALSE.

```
IFSHORTER(Pump1.FlowRate > 75.0, 1 minute)
```

Example 2

The following expression evaluates whether or not the stream SysTimeSec remains greater than or equal to 20 for a period shorter than half a minute. If it does, it returns the value 5. If it doesn't, it returns a "bad quality" value.

```
IFSHORTER(SysTimeSec >= 20, 0.5 minute, 5)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 3

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period shorter than 1 minute. If it does, it returns the value from Pump1.Flowrate. If it doesn't, it returns a value of 75.

```
IFSHORTER(Pump1.FlowRate > 75.0, 1 minute, Pump1.FlowRate, 75.0)
```

IFSTALE

The IFSTALE() function replaces values from the input stream that are older than a specified interval with an alternative value.

Syntax

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltValue*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltValue* - A replacement value
Description	Replaces every value in the input stream older than *Interval* with *AltValue*.

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltStream*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltStream* - A stream of alternative values
Description	Replaces every value in the input stream older than *Interval* with the value from *AltStream* at the corresponding time.

IFSTALE Examples

IFSTALE Examples

Example 1

The following expression replaces every value in the input stream older than 40 minutes with the value -1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
IFSTALE(SysTimeHour, 40 minute, -1)
```

Example 2

The following expression replaces every value in the input stream H100.TotalFlow older than 40 minutes with the value from the alternative stream Pump1.FlowRate + Pump2.FlowRate at the corresponding time.

```
IFSTALE(H100.TotalFlow, 1 hour, Pump1.FlowRate + Pump2.FlowRate)
```

LOG

The LOG() function produces a stream of results calculating the base 10 logarithm of a stream's values.

Syntax

Format	*Stream* = **LOG**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values converted to their base 10 logarithm.

LOG Examples

LOG Examples

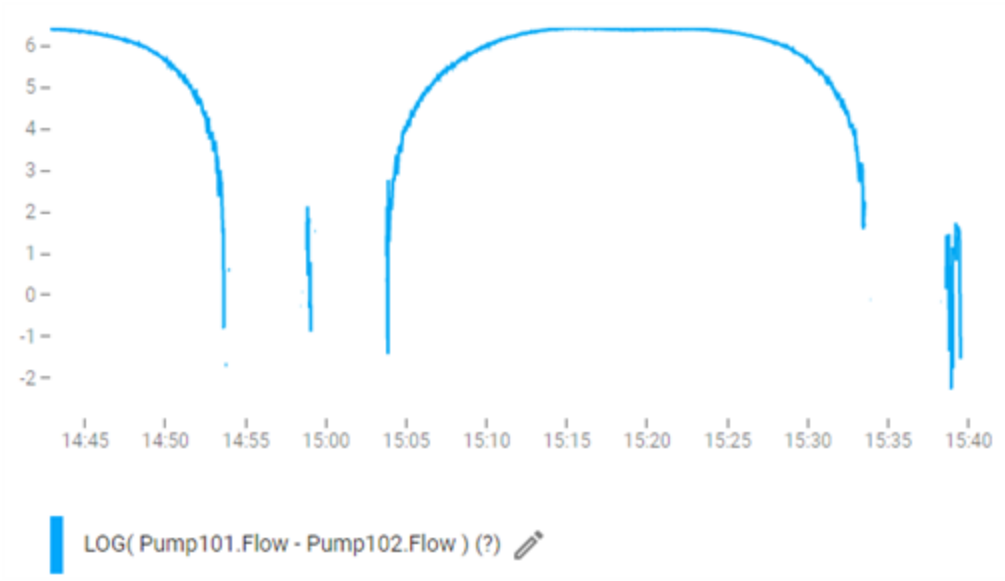
Example 1

The following expression returns the base 10 logarithm of the difference in flow rates for two pumps.

```
LOG( Pump101.Flow - Pump102.Flow )
```

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



ROUND

The ROUND() function produces a stream of results rounding a stream's values to the nearest integer.

Syntax

Format	*Stream* = **ROUND**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values rounded to the nearest integer.

ROUND Examples

ROUND Examples

Example 1

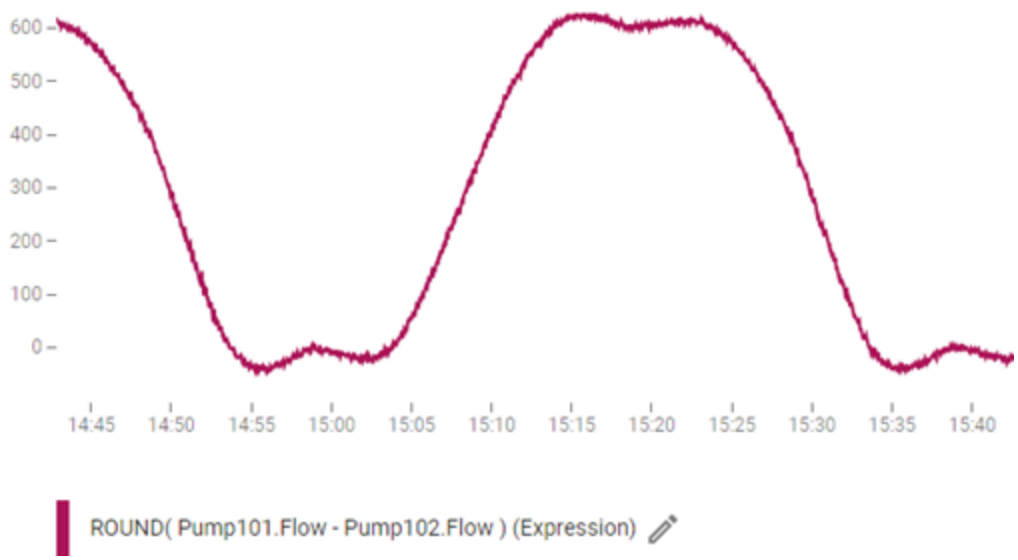
The following expression returns the difference in flow rates for two pumps, rounded to the nearest integer.

```
ROUND( Pump101.Flow - Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag.

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

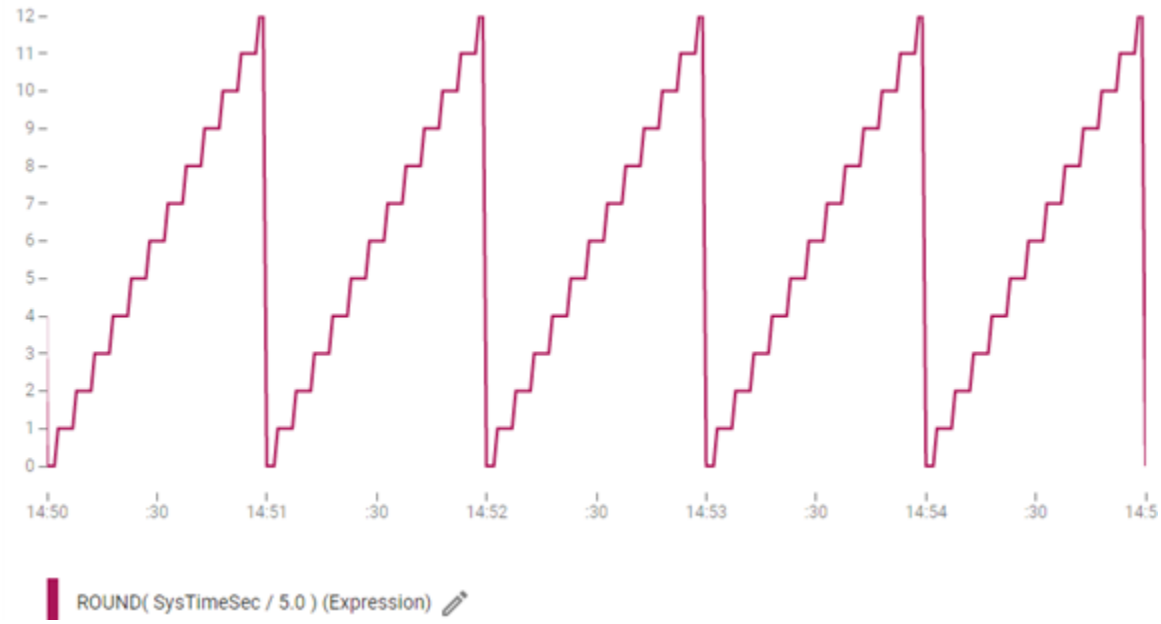


Example 2

The following expression returns the second value of the time, divided by 5 and rounded to the nearest integer.

```
ROUND( SysTimeSec / 5.0 )
```

Represented as a line chart over a 5 minute interval:



SIGN

The SIGN() function produces a stream of results indicating the sign of a stream's values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **SIGN**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values replaced based on their sign. Positive values are replaced with 1, negative values are replaced with -1, and zero values remain zero.

SIGN Examples

SIGN Examples

Example 1

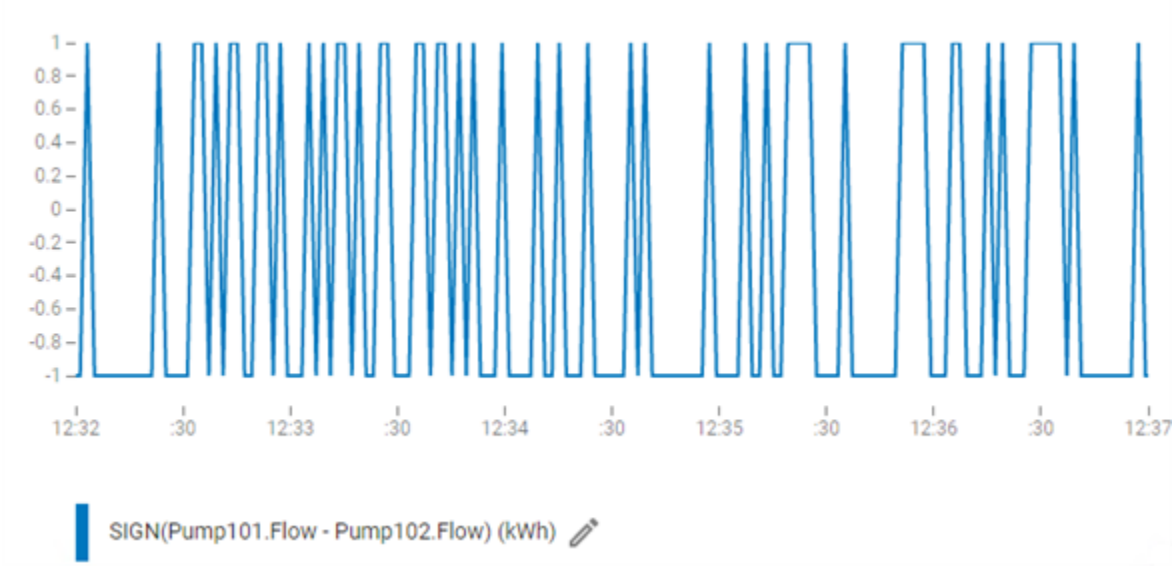
The following expression returns the sign of the value of the difference in flow rates between two pumps.

```
SIGN( Pump101.Flow - Pump102.Flow )
```

For each data point in the results, the value will be:

- -1 when the second pump has a higher flow rate
- 0 when the flow rates are equal
- 1 when the first pump has a higher flow rate.

Represented as a line chart over a 5 minute interval:



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQRT

The SQRT() function produces a stream of results calculating the square root of a stream's values.

Syntax

Format	*Stream* = **SQRT**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, but with all values being replaced with their square roots.

SQRT Examples

SQRT Examples

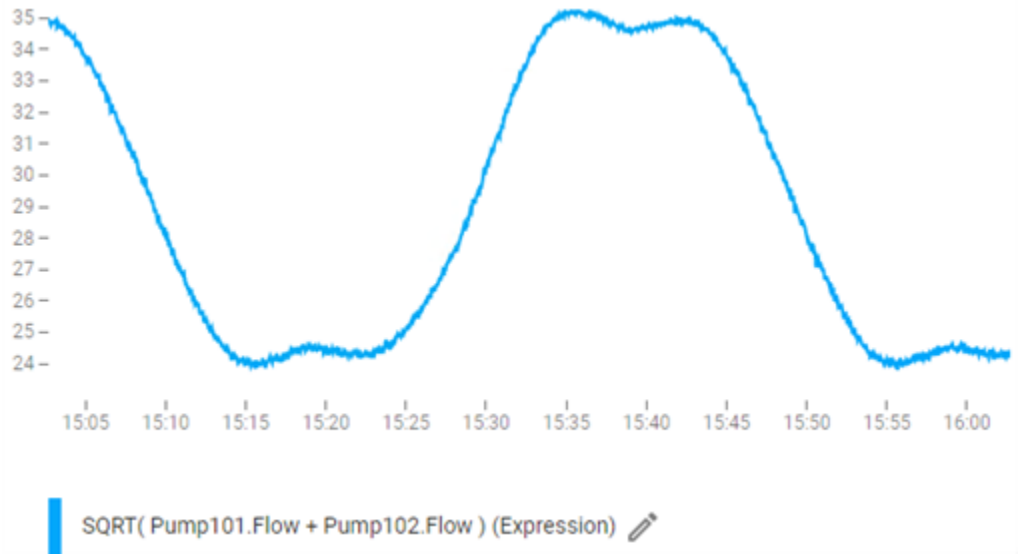
Example 1

The following expression returns the square root of the sum of flow rates for two pumps.

```
SQRT( Pump101.Flow + Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag, so the expression's results are in liters per second.

Represented as a line chart over a 1 hour interval:



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TIMESHIFT

The TIMESHIFT() function shifts the time stamps in the input stream by adding or subtracting a specified time interval.

Syntax

Format	*Stream* = **TIMESHIFT**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A stream of values *TimeInterval* - The time period to add (or subtract if negative)
Description	The specified interval of time is added to (or subtracted from) the time stamps in the input stream.

TIMESHIFT Examples

TIMESHIFT Examples

Example

The following expression shifts the time stamp of each flow rate value forward by one hour.

```
TIMESHIFT( Pump101.Flow, Hour )
```

Example 2

The following expression shifts the time stamp of the current time's minute value back by two minutes.

```
TIMESHIFT( SysTimeMin, -2 minute )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TIMESHIFT( SysTimeMin, -2 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:57:00.0000000	59	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2022-03-31 15:58:00.0000000	0	192
2022-03-31 15:59:00.0000000	1	192
2022-03-31 16:00:00.0000000	2	192
...	...	...

TRUNCATE

The TRUNCATE() function produces a stream of results removing the fractional portion of a stream's values.

Syntax

Format	*Stream* = **TRUNCATE**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with the fractional part of all values removed.

TRUNCATE Examples

TRUNCATE Examples

Example 1

The following expression returns the product of flow rates for two pumps, with the fractional portion of each value removed.

```
TRUNCATE( Pump101.Flow * Pump102.Flow )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( Pump101.Flow * Pump102.Flow )'
```

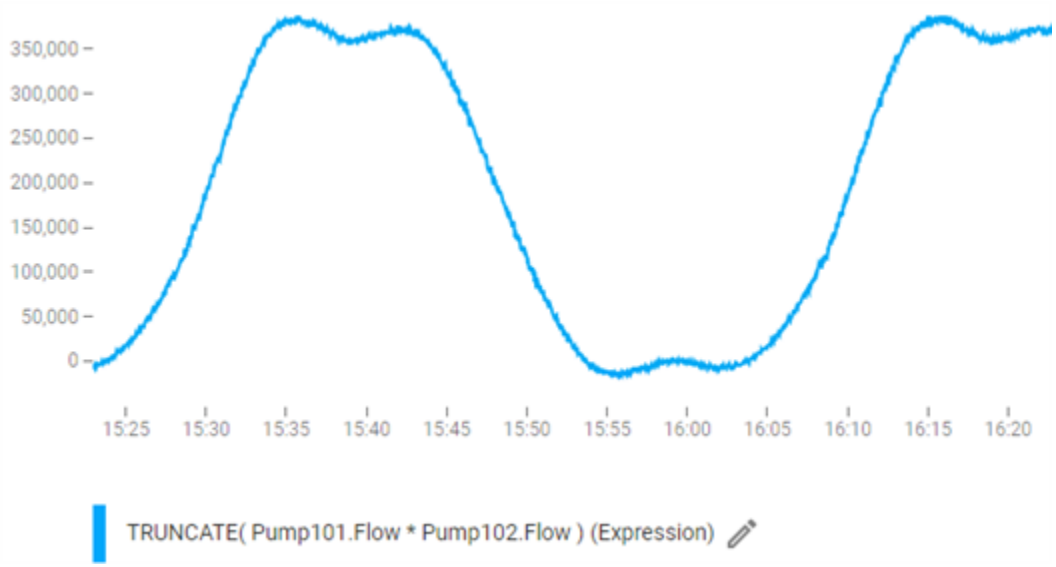
Results:

DateTime	Value	OPCQuality
2022-03-31 15:24:49.2590000	11030	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2022-03-31 15:24:51.2590000	14974	192
2022-03-31 15:24:53.2590000	14542	192
2022-03-31 15:24:55.2590000	14887	192
...	...	...

Represented as a line chart over a 1 hour interval:



Example 2

The following expression divides the current time's second value by 5, removing the fractional portion.

```
TRUNCATE( SysTimeSec / 5.0 )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( SysTimeSec / 5.0 )'
AND wwRetrievalMode = 'Full'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:31:39.0000000	7	192
2022-03-31 15:31:40.0000000	8	192
2022-03-31 15:31:41.0000000	8	192
2022-03-31 15:31:42.0000000	8	192
...	...	...

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

UOM (Unit of Measure conversion)

The UOM() function converts the values in a stream to a specified unit of measure.

Syntax

Format	*Stream* = **UOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the values in the provided stream to use the specified unit of measure. If the specified unit does not exist, or is not correctly mapped to a catalog unit for conversion, the original unit is returned instead.

UOM Examples

UOM Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, with the units converted to gallons per second.

```
UOM( Pump101.Flow, gal/min )
```

Supported time-series functions

AVEVA Historian Client Web supports using time-series functions in expressions. These functions enable you to compare and manipulate data.

The following time-series functions are supported:

Function	Purpose
AVERAGE	Calculates the time-weighted average for a stream.
AVERAGES	Calculates the statistical average for a stream.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

COUNT	Returns a count of the delta points for a stream.
COUNTALL	Returns a count of the data points for a stream.
DURATION	Calculates the duration of *TRUE* states in an expression.
METER	Returns the delta change of a stream's value.
MIN/MAX	Selects the minimum/maximum value of a stream.
PREV/PREVGOOD	Returns the last value from the previous time period.
TOTAL	Calculates the time-weighted total for a stream.
TOTALS	Calculates the statistical total for a stream.

AVERAGE

The AVERAGE() function calculates the time-weighted average value of a stream or array of streams.

When using AVERAGE() in a query, if no resolution is specified, then the behavior is the same as average retrieval mode with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the average value is calculated for each cycle.

Syntax

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the time-weighted average value of a stream or array of streams.

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the time-weighted average of a stream or array of streams over the specified time interval,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	based on local time. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

AVERAGE Examples

AVERAGE Examples

Example 1

The following expression returns the average flow rate of a pump.

```
AVERAGE(Pump101.Flow)
```

Example 2

The following expression returns the average flow rate of a pump over 10 minute intervals.

```
AVERAGE(Pump101.Flow, 10 Minute)
```

Example 3

The following expression returns the average flow rate of all streams that include ".Flow" in their tag name at 10 minute intervals.

```
AVERAGE(SELECT("TagName",".Flow"), 10 Minute)
```

Example 4

The following expression returns the average minute value of the time over 5 minute intervals.

```
AVERAGE(SysTimeMin, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'AVERAGE( SysTimeMin, 5 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 12:00:00.0000000	2	192
2022-03-31 12:05:00.0000000	7	192
2022-03-31 12:10:00.0000000	12	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2022-03-31 12:15:00.0000000	17	192
...	...	...

AVERAGES

The AVERAGES() function calculates the statistical average value of a stream or array of streams over a specified time interval.

Note: This is the preferred method for calculating averages from discrete samples such as those from a checkweigher. For most measurements, however, AVERAGE is preferred.

Syntax

Format	*Stream* = **AVERAGES**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the statistical average for a stream or array of streams over the specified time interval. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

AVERAGES Examples

AVERAGES Examples

Example 1

The following expression returns the average weight of items coming off a line at 10 minute intervals.

```
AVERAGES(Line3.ItemWeight, 10 Minute)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression returns the average weight of all streams that include "ItemWeight" in their tag name at 10 minute intervals.

```
AVERAGES(SELECT("TagName","ItemWeight"), 10 Minute)
```

COUNT

The COUNT() function returns a count of the delta points for a stream. Note that if the quality detail of a point differs from the previous point, this counts as a delta point, even if the value doesn't change.

Syntax

Format	*Stream* = **COUNT**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of delta points during the query period. The query period is based on local time.

Format	*Stream* = **COUNT**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the delta points are counted
Description	Returns a count of delta points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNT Examples

COUNT Examples

Example 1

The following expression returns the number of delta points in a freezer unit's temperature.

```
COUNT(Freezer15.Temperature)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression returns the number of delta points in a freezer unit's temperature over hour intervals.

```
COUNT(Freezer15.FlowTotal, hour)
```

COUNTALL

The COUNTALL() function returns a count of all data points for a stream.

Syntax

Format	*Stream* = **COUNTALL**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of all data points during the query period. The query period is based on local time.

Format	*Stream* = **COUNTALL**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the points are counted
Description	Returns a count of all data points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNTALL Examples

COUNTALL Examples

Example 1

The following expression returns the number of data points reported by a pressure alert on a boiler.

```
COUNT(Boiler03.PressureAlert)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression returns the number of points reported by a pressure alert on a boiler over a one hour interval.

```
COUNT(Boiler03.PressureAlert, hour)
```

DURATION (Time in State)

The DURATION() function returns a stream representing the duration in seconds of each "true" state in an expression.

Syntax

Format	*Stream* = **DURATION**(*StreamLogicalExpression*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated
Description	The logical expression represented by *StreamLogicalExpression* is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results.

Format	*Stream* = **DURATION**(*StreamLogicalExpression*, *TimeInterval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *TimeInterval* - The time interval over which to calculate durations
Description	The logical expression represented by *StreamLogicalExpression* is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results, with durations being split across the specified time period. For example, if StreamLogicalExpression evaluates to a *TRUE* value for a time period of 07:55-08:10, and the specified *Interval* is hourly, the output contains one "5 minute" and one "10 minute" result. Without specifying an *Interval*, the output would instead contain one "15 minute" result.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DURATION Examples

DURATION Examples

Example 1

The following expression returns the amount of time in seconds over the specified interval that the number of seconds in the time is 20 or greater.

```
DURATION(SysTimeSec >= 20)
```

Example 2

The following expression returns the amount of time per one hour interval that a tank's level is at or above 500 kL.

```
DURATION(Tank400.Level > 500, hour)
```

Represented as a query:

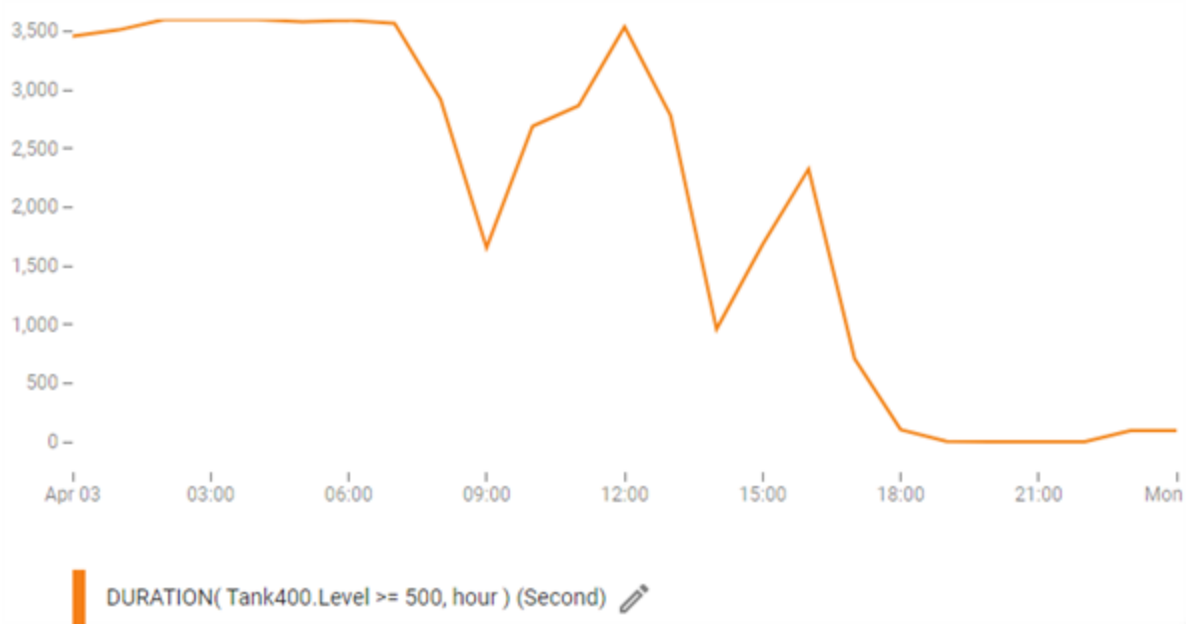
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(Tank400.Level >= 500, hour)'
AND DateTime > '2022-04-03 00:00:00'
AND DateTime < '2022-04-04 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-04-03 01:00:00.0000000	3600	192
2022-04-03 02:00:00.0000000	3578	192
2022-04-03 03:00:00.0000000	3592	192
...	...	...
2022-04-03 23:00:00.0000000	94	192

Represented as a line chart over a 1 day interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 3

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater, over 5 minute intervals.

```
DURATION(SysTimeSec >= 20, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(SysTimeSec >= 20, 5 minute)'
```

Results:

DateTime	Value	OPCQuality
2022-04-01 08:45:00.0000000	200	192
2022-04-01 08:50:00.0000000	200	192
2022-04-01 08:55:00.0000000	200	192
2022-04-01 09:00:00.0000000	200	192
...	...	...

Example 4

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater over 30 second intervals, and the results are converted to minutes.

```
UOM(DURATION(SysTimeSec >= 20, 0.5 minute), Minute)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

METER

The METER() function returns the delta change of the value of a stream or array of streams. This works both for tags without rollover, and tags that reset upon reaching a rollover value.

If an expression includes a single tag, the rollover value defined in the AVEVA Historian will be used. If an expression includes multiple tags, or performs an operation on a tag, a rollover value of 0 is assumed. For example, the following would use the Historian-defined rollover value:

- METER(Tag1, hour)

The following would assume a rollover value of 0:

- METER(Tag1 + Tag2, hour)
- METER(Tag1 + 5, hour)

Otherwise, the METER function works similarly to counter retrieval mode. For more details about rollover, delta change, and common use cases for this function, see Counter Retrieval.

Syntax

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the delta change in the stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which the delta change is calculated.
Description	Returns the delta change in the stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

METER Example

METER Example

Example 1

The following expression returns the changes in a pump's total flow.

```
METER(Pump101.FlowTotal)
```

Example 2

A common use of this function is for counters with rollover. The following expression returns the number of units to come out of a production line at one hour intervals.

```
METER(ProductionLine2.UnitCount, hour)
```

Example 3

The following expression returns the number of units to come out of all streams that include "UnitCount" in their tag name at one hour intervals.

```
METER(SELECT("TagName","UnitCount"), hour)
```

MIN/MAX

The MIN() and MAX() functions produce a stream of results selecting the minimum or maximum value of a stream or array of streams. The returned timestamps correspond to the minimum/maximum values retrieved, rather than the beginning of the time interval.

When using MIN() or MAX() in a query, if no resolution is specified, the behavior is the same as minimum or maximum retrieval modes with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the minimum/maximum values are calculated for each cycle.

Syntax

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the minimum (or maximum) value of a stream or array of streams.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*, *TimeInterval*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which minimum or maximum values are calculated.
Description	Returns the minimum (or maximum) value of a stream or array of streams over the specified time interval.

MIN/MAX Examples

MIN/MAX Examples

Example 1

The following expressions return the minimum (or maximum) flow rate of a pump.

```
MIN(Pump101.Flow)
MAX(Pump101.Flow)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'MIN(Pump101.Flow)'
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Result:

DateTime	Value	OPCQuality
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 2

The following expressions return the minimum (or maximum) flow rate of a pump over 30 minute time intervals.

```
MIN(Pump101.Flow, 30 minute)
MAX(Pump101.Flow, 30 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
WHERE wwExpression = 'MIN(Pump101.Flow, 30 minute)'
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-03-30 00:13:45.2590000	592.065856933594	192
2022-03-30 00:43:07.2590000	592.34765625	192
2022-03-30 01:01:07.2590000	593.351257324219	192
...	...	...
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 3

The following expression returns the minimum flow rate of all streams measured in the volumetric flow dimension over 30 minute time intervals.

```
MIN(SELECT("Dimension","Volumetric Flow"), 30 Minute)
```

PREV/PREVGOOD

The PREV() and PREVGOOD() functions return the last value from the previous time period for the provided stream.

When using PREV() or PREVGOOD() in a query, if no resolution is specified then the query returns an error.

Syntax

Format	*Stream* = **PREV**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns the last value from the previous time period, where the period is specified by *TimeInterval*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **PREVGOOD**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns similar results to **PREV**(), but values with "bad" or "uncertain" quality are skipped. **PREVGOOD** will continue seeking backwards until a "good" value is found, going beyond the previous period if necessary. This means **PREVGOOD** always returns a value with "good" quality.

PREV/PREVGOOD Examples

PREV/PREVGOOD Examples

Example 1

The following expression

```
Line1.Total - PREV( Line1.Total, hour )
```

TOTAL

The TOTAL() function produces a stream of results calculating the time-weighted total for a stream or array of streams.

Syntax

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the time-weighted total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams,*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	TimeInterval)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated.
Description	Calculates the time-weighted total for a stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

TOTAL Examples

TOTAL Examples

Example 1

The following expression returns the total volume flowing through a pump over 1 hour intervals.

```
TOTAL(Pump101.Flow, hour)
```

Example 2

The following expression returns the total volume of all streams measured in the volumetric flow dimension over 1 hour intervals.

```
TOTAL(SELECT("Dimension","Volumetric Flow"), hour)
```

Example 3

The following expression returns the total energy produced by a turbine over the query period.

```
TOTAL(Turbine306.Output)
```

Example 4

The following expression returns the total energy for all streams with a tag name that starts with "Turbine" and ends with ".Output" over the query period.

```
TOTAL(SELECT("TagName","Turbine*.Output"))
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TOTALS

The TOTALS() function produces a stream of results calculating the statistical total for a stream or array of streams.

Syntax

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the statistical total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated
Description	Calculates the statistical total for a stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TOTALS Examples

TOTALS Examples

Example 1

The following expression returns the calculated statistical total volume flowing through a pump over a 10 minute interval.

```
TOTALS(Pump102.Flow, 10 minute)
```

Example 2

The following expression returns the calculated statistical total volume for all streams with a tag name that includes "Flow" over a 10 minute interval.

```
TOTALS(SELECT("TagName","Flow"), 10 minute)
```

Example 3

The following expression returns the calculated statistical total energy produced by a turbine over the query period.

```
TOTALS(Turbine306.Output)
```

Example 4

The following expression returns the calculated statistical total energy for all streams with a tag name that starts with "Turbine" and ends with ".Output" over the query period.

```
TOTALS(SELECT("TagName","Turbine*.Output"))
```

Choose a tag category from Search Results

Search Results on the left side displays the tags that result from your data search. Your results are grouped into categories which are organized by unit of measure, such as degrees Celsius, percentage, and parts per million (PPM).

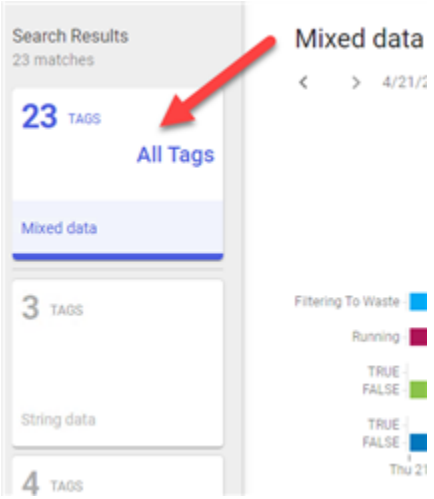
The All Tags category shows all tags, regardless of unit of measure.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To choose a tag category to work with

- From Search Results on the left, select the tag category you want to work with.

  Note: Units of measure are case-sensitive. That means ml (milliliters) and ML (megaliters) are not identified as the same measurement



About the All Tags category

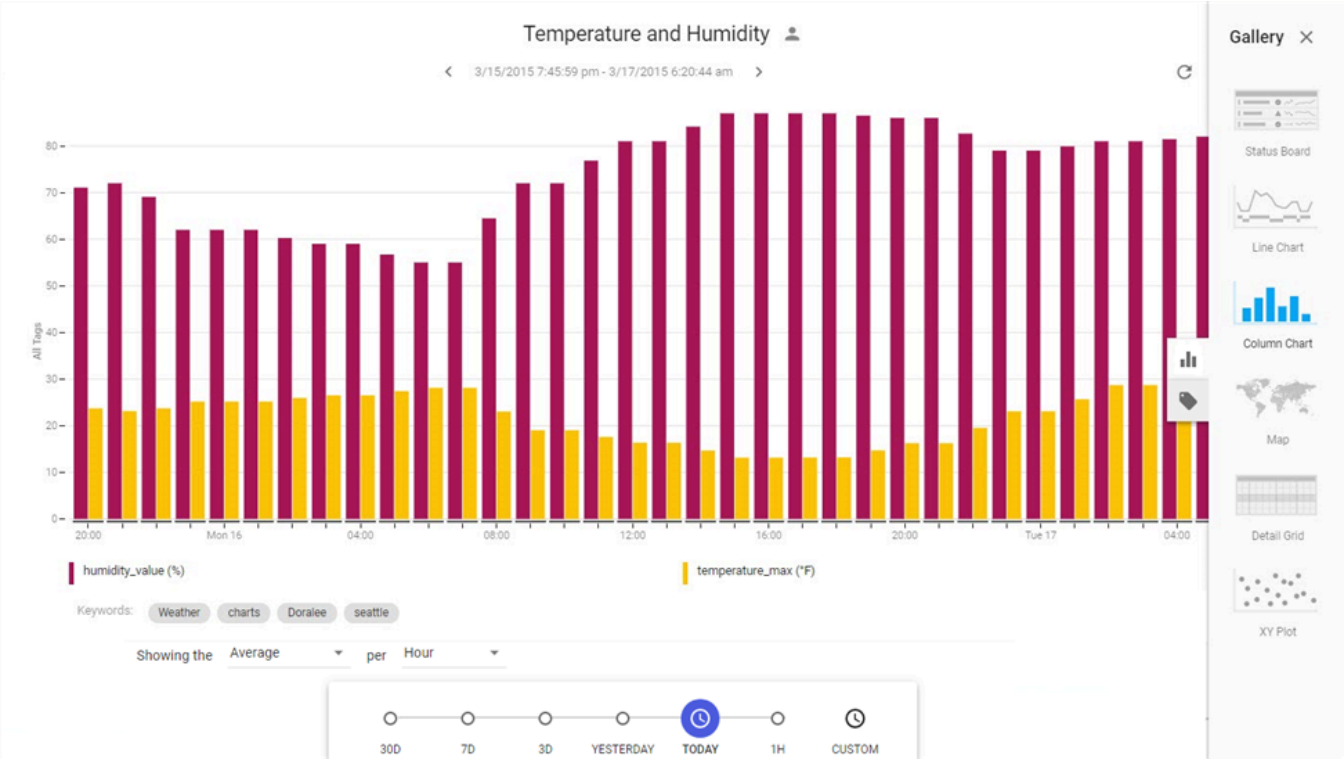
If you want to compare tags that are measured by different units of measure, you can select a special category called All Tags from the Search Results area. All Tags is a special category that combines all numeric tags with different units of measure.

Note: Other products, including AVEVA Historian, use the term "analog" instead of "numeric". If your data is sourced from an AVEVA Historian, your associated analog tags will display here.

This category includes every numeric tag that results from your search query.

For example, the chart below is created from the All Tags category. This chart shows various weather-related tags that are each measured with a different unit of measure. In this case, you can see cloudiness and humidity (as percentage), atmospheric pressure (as hPa units), wind speed (as mph), and temperature (as degrees Fahrenheit) all on one chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



For this chart:

- Each tag and its corresponding Y-axis are color coded.
- The tagname and corresponding color display at the bottom of the chart.
- The maximum value (rounded up) for each tag and corresponding color display at the top of the chart. The minimum value (rounded down) for each tag is displayed directly below the tag's maximum, near the X-axis.
- The X-axis (horizontal) shows a timeline while the Y-axis (vertical) shows tag values.
- Values for all tags display when you hover over the chart. Move the cursor back and forth to see values for specific points in time.

Choose tags to display in your chart

The Data panel on the right side lets you select the tags to display in your chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To select tags to display

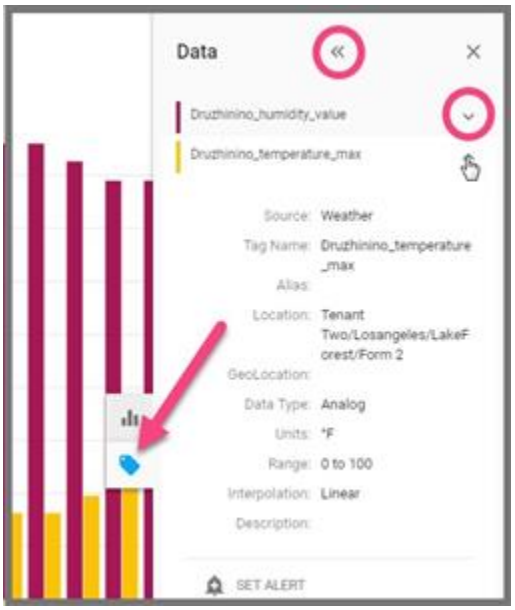
1. From the Search Results on the left side of the screen, select a tag category to work with.
2. On the right side of the screen, select the tag icon to show the Data panel.

   The Data panel lists the individual tags for the data tag category selected on the left side of the screen.
3. Select tags to include in your chart. When you select a tag, it is assigned a color in the Data panel and in the chart.

   You can select up to 30 tags to view and analyze in a single chart.

   For more information on navigating within the chart, see Navigate your charted data.

   Tips: To expand the Data panel, select the expand panel icon « at the top of the panel. To deselect all tags, select the deselect icon at the bottom of the panel.



Additionally, on the Data panel, you can select to display details for a tag.

View tag details

To view details for a tag

1. Select to view tags in the Gallery (right side).
2. Select next to the tag.

   The tag details display.

Discover alias and location for a tag

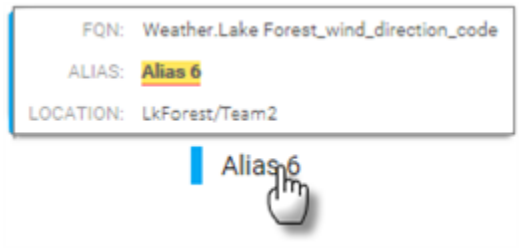
To view a tag's alias and corresponding location

- In the legend area of a chart, or in the Data panel, point to the tag you want to learn about.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A tooltip displays with these details about the tag:

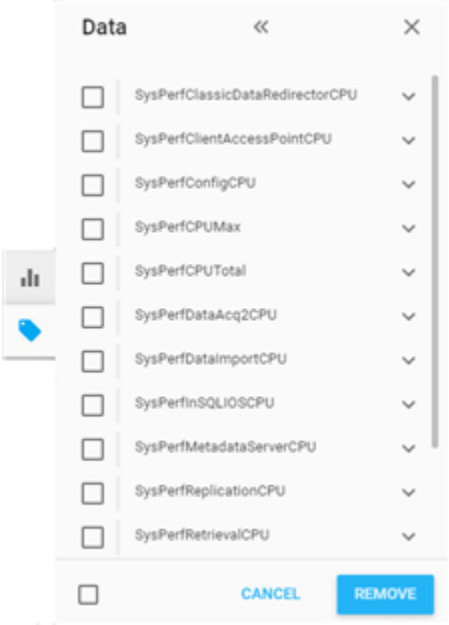
- The tag's data source name followed by the tag name (also known as its fully qualified name, or FNQ)
- Any associated alias
- Any associated location.



Remove a tag

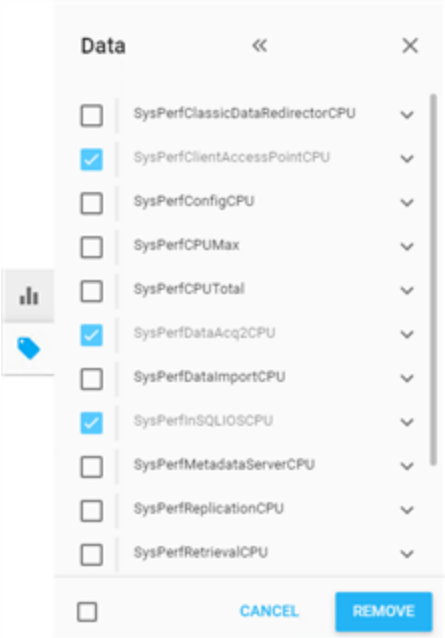
To remove a tag from the data set:

1. Select to view tags in the Gallery (right side).
2. Select **EDIT** from the bottom of the gallery. A checkbox displays next to each tag in the list.



3. Select the checkbox next to each tag that you want to remove. You can also use the checkbox at the bottom of the gallery to alternate between selecting all tags, or selecting none.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. Select **REMOVE** to remove all the selected tags from the list, or **CANCEL** to keep all the tags.

Choose a chart style

The Gallery on the right side lets you select style of chart to display.

To choose a chart for your tags

1. From the Search Results on the left side of the screen, select a tag category to work with.
2. On the right side of the screen, select the chart gallery icon .
3. From the list of charts in the Gallery, choose how you want the tags displayed.

   Note: The list of available charts depends on the category of tags you choose.

Available chart types

In the Gallery, you can choose from among various layout types, including:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description	Customization and other options	Available for tag types
Column Chart	Vertical bar chart of each selected tag. Values per time interval are calculated by ending time.	Select daily, hourly, or 15-minute intervals. If the chart is selected in the mixed tag group and non-analog (i.e., string or discrete) tags are available, aggregate the data by a non-analog tag instead of an interval by selecting the **Event** option and the tag. Select an individual bar for details and a link to a detailed Line Chart of subset of the data.	Numeric String/Discrete
Cumulative Line	Continuous line chart of totals.	Hover over chart for a legend showing individual data tag values for a particular time and date. Select to add a limit line or to toggle autoscale and legend.	Numeric
Detail Grid	Numeric minimum, maximum, average, and last values for selected tags.	Select daily, hourly, or 15-minute values.	Numeric String/Discrete
Diagnostic Grid	Descriptive text, numeric values, and quality details about the selected tags.		Numeric String/Discrete
Gantt Chart	Shows which state each discrete or string tag is in over time. Charts for discrete and string tag are stacked.	Hover over the X (horizontal) axis to select pan and zoom options. The Y (vertical) axis lists all values shown on the chart for each discrete or string tag. As you zoom in or out, this list may get smaller or larger.	String/Discrete
Line Chart	Continuous line of values for each selected analog	Hover over the X (horizontal) axis to select	Numeric Mixed

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	tag and gantt chart for each discrete tag.	pan and zoom options. Hover over chart for a legend showing individual tag values for a particular time and date. Select ⚙ to add a limit line or to toggle autoscale and legend. **Tip:** If the chart and/or legend become too crowded to read clearly, try using autoscale or reduce the number of tags included in the chart.	
Rolling Sum	Vertical bar chart of totals by selected time increment. **Note:** This chart type is sometimes called a cumulative sum chart.	Select ⚙ to add a limit line or toggle legend. Select an individual bar for details and a link to a detailed Line Chart of subset of the data. Select data retrieval mode (average, last, or total). Select daily, hourly, or 15-minute values.	Numeric
Status Board	Designed for a big picture view, including current value, trend, and a thumbnail chart of recent performance	Select the thumbnail chart for more details and a link to a detailed Line Chart for that tag.	Numeric String/Discrete Mixed
Summary	A table of values for discrete or string tags.		String/Discrete
Summary Chart	Horizontal bar chart of each selected numeric tag.		Numeric
Summary Grid	For a set of tags measured by the same engineering unit, the values for each by time increment. Values per time interval are calculated by ending time.	Select data retrieval mode (average, minimum, maximum, sum, or last). Select daily, hourly, or 15-minute values.	Numeric String/Discrete
Time In State Chart	For string and discrete	Select ⚙ to add a limit	String/Discrete

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	data, horizontal bar chart showing time in state for selected tags.	line or to toggle legend.	
XY Plot	Scatter plot of individual reported values for selected data tags as they compare to X-axis tag.	Select ⚙ to change the chart to lines with circles, or to toggle autoscale and legend. Select tag for X axis.	Numeric String/Discrete

Navigate your charted data

You can navigate charted data using these methods:

- Scroll through data values (right and left)
- Pan or zoom values in a line chart

Scroll through data values (right and left)

In single chart view, your data may span a timeframe that cannot be displayed on a single page. In that case, you can use the < and > indicators to scroll forward and backward in time to see your data values.

Pan, zoom, or expand details in a line chart

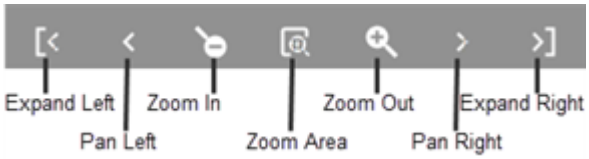
You can pan, zoom, and expand data details in a line chart when you use the Single Axis option.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Note: Panning and zooming is *not* available when you use the Multi Axis or Stacked Axis option on a line chart.

Pan, zoom, and expand controls

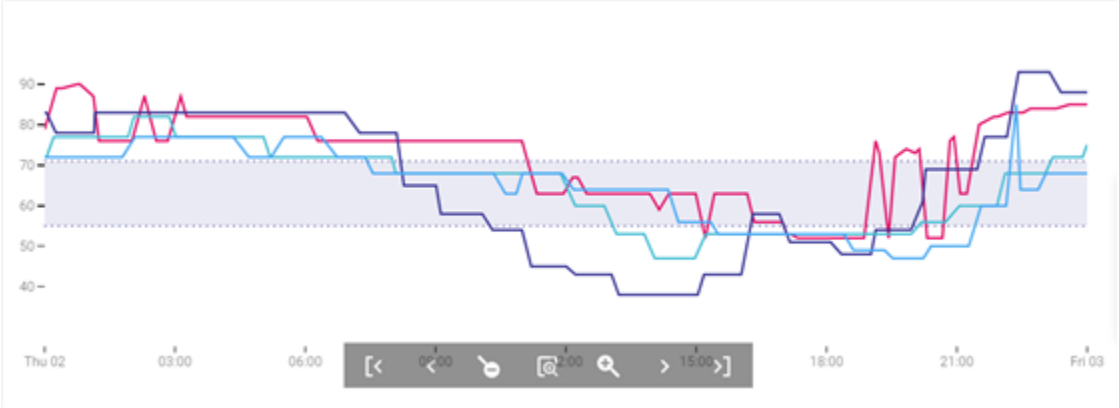
These controls are available for you to use:



- Expand Left -- Add more detail by expanding the reporting boundary on the left.
- Pan Left -- Keep the same level of detail, but shift focus to the left.
- Zoom In -- Add more detail by expanding the right and left reporting boundaries.
- Zoom Area -- Mark an area on the chart and zoom in to show more detail.
- Zoom Out -- Show less detail by narrowing the right and left reporting boundaries.
- Pan Right -- Keep the same level of detail, but shift focus to the right.
- Expand Right -- Add more detail by expanding the reporting boundary on the right.

To view the pan, zoom, and expand controls

- Hover over the X (horizontal) axis of your line chart.



See these topics for details on using pan, zoom, and expand functionality:

- Zoom in and out
- Pan left or right
- Expand detail left or right

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Zoom in or out (line chart)

You can use a mouse or touch screen to zoom in or out of chart data.

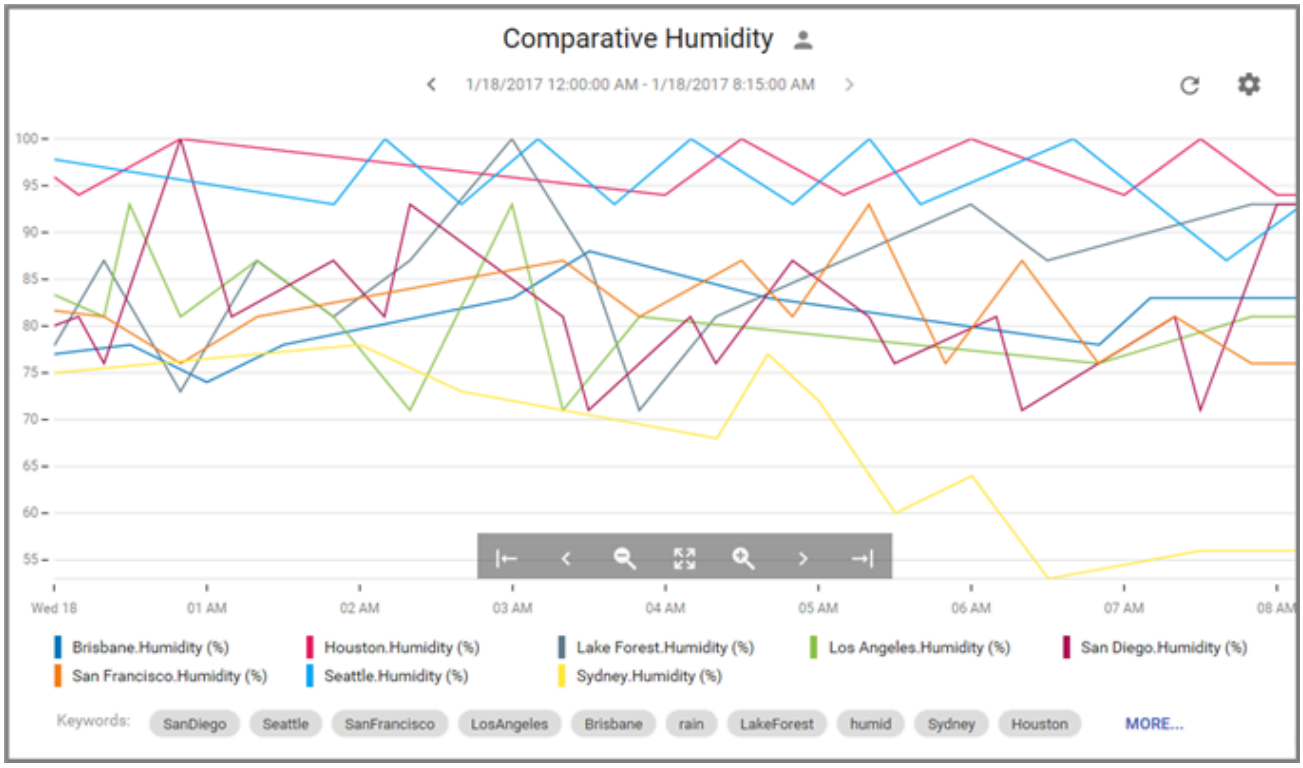
When you zoom in and out, you are affecting the timeframe (horizontal X axis) and not the values (vertical Y axis).

- When you zoom in, you see more detailed data for a smaller timeframe.
- When you zoom out, you see less detail for a longer timeframe.

Note: If you are using Insight on a touch screen with Microsoft Edge, be sure to adjust the touch screen settings.

To zoom in

1. Hover over the X (horizontal) axis of your line chart to see the pan and zoom controls.



2. Click to reduce the timeframe and increase the level of detail displayed.

Or, use one of these methods to zoom in:

- Double-click the chart. The X-axis (horizontal) location that you click will stay stationary and the time increments will expand to the right and left.
- Click the chart and move the mouse wheel forward.
- Double-tap the chart.
- Pinch and move fingers together.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To zoom out

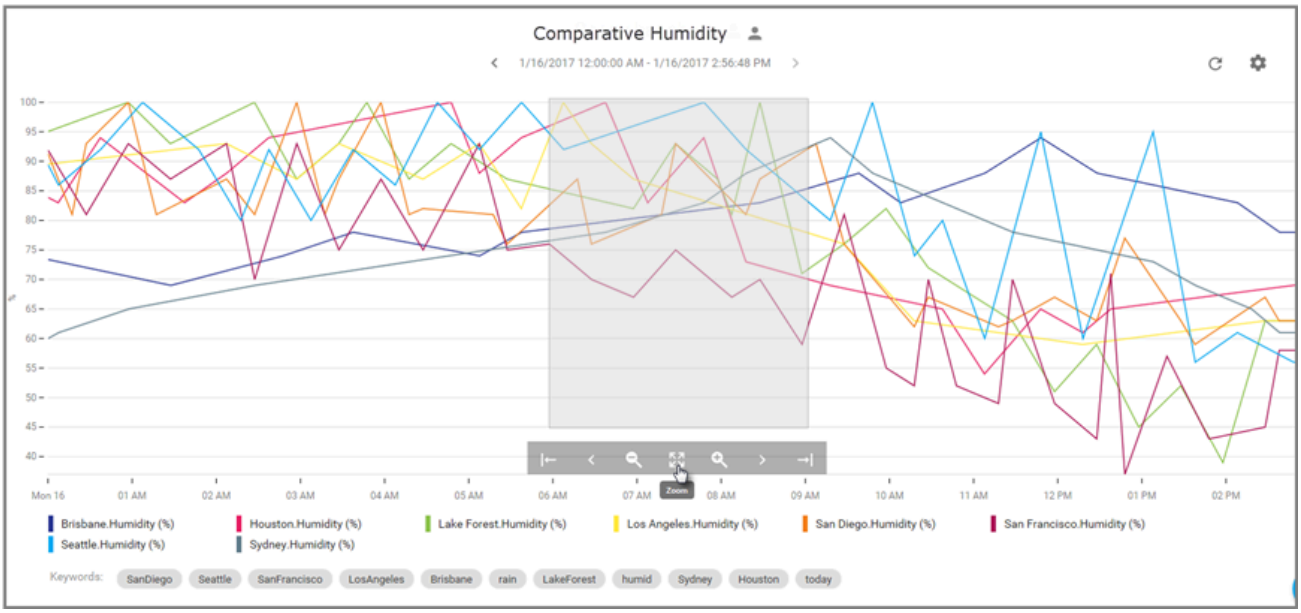
1. Hover over the X (horizontal) axis of your line chart to see the pan and zoom controls.
2. Click to increase the timeframe and reduce the level of detail displayed.

Or, use one of these methods to zoom out:

- Click the chart and move the mouse wheel backward.
- Pinch and move fingers apart.

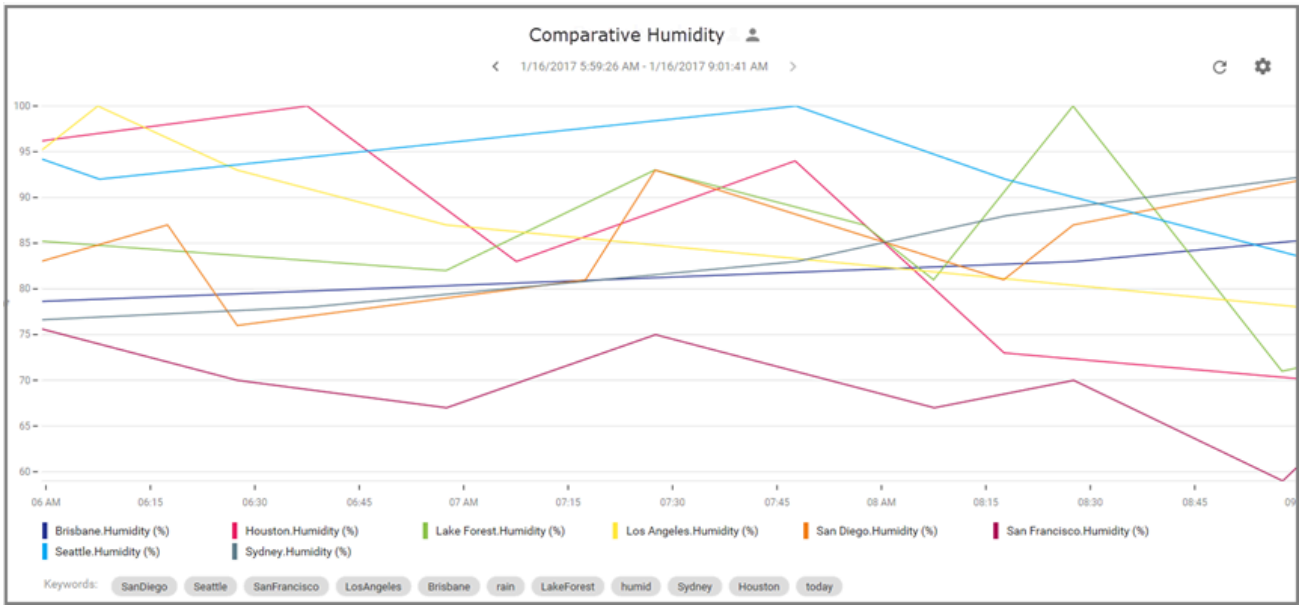
To zoom into a particular area

1. Hover over the X (horizontal) axis and select Zoom Area.
2. Drag the cursor to mark the area -- indicating a specific timeframe -- for which you'd like more detail.



The chart redraws to display a more detailed view for the timeframe you specified.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



About touch screen settings for Edge

By default, Microsoft Edge touch event setting is set to "Always off". Enable this setting to use Edge with Historian Client Web on a touch screen. This is particularly important when using the zoom feature in Historian Client Web.

To enable touch settings for Edge

1. In the Microsoft Edge address bar, type "about:flags".
2. Under **Touch**, change **Enable touch events** from to "Only on when a touchscreen is detected".
3. Restart the browser.

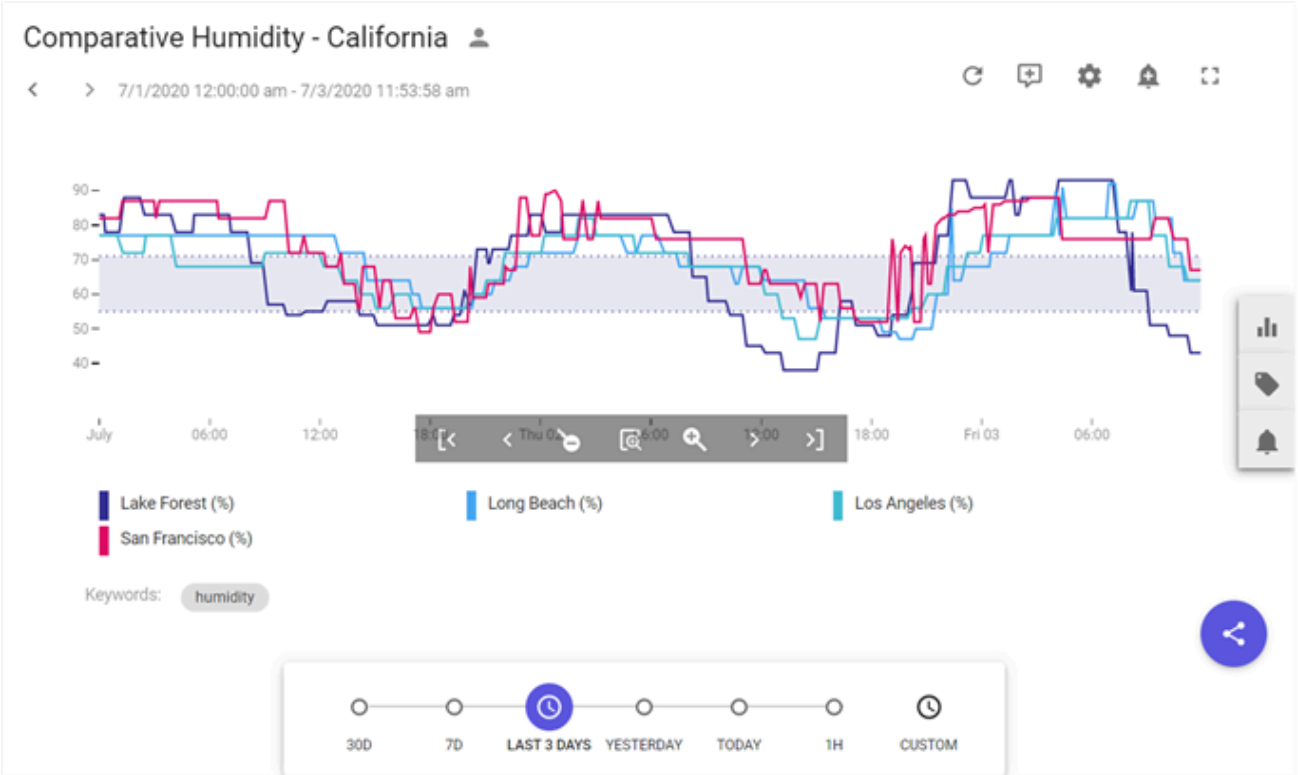
Pan left or right (line chart)

You can use a mouse or touch screen to pan to the left and right to explore chart data.

To pan to the left or right

1. Hover over the X (horizontal) axis of your line chart to see the pan and zoom controls.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Select to pan left. This shifts the time window to the left by half of the current duration.

   Or, select to pan right. This shifts the time window to the right by half of the current duration.

Or, use one of these methods to pan chart data:

- Select and drag to the left or right.
- Touch and swipe to the left or right.

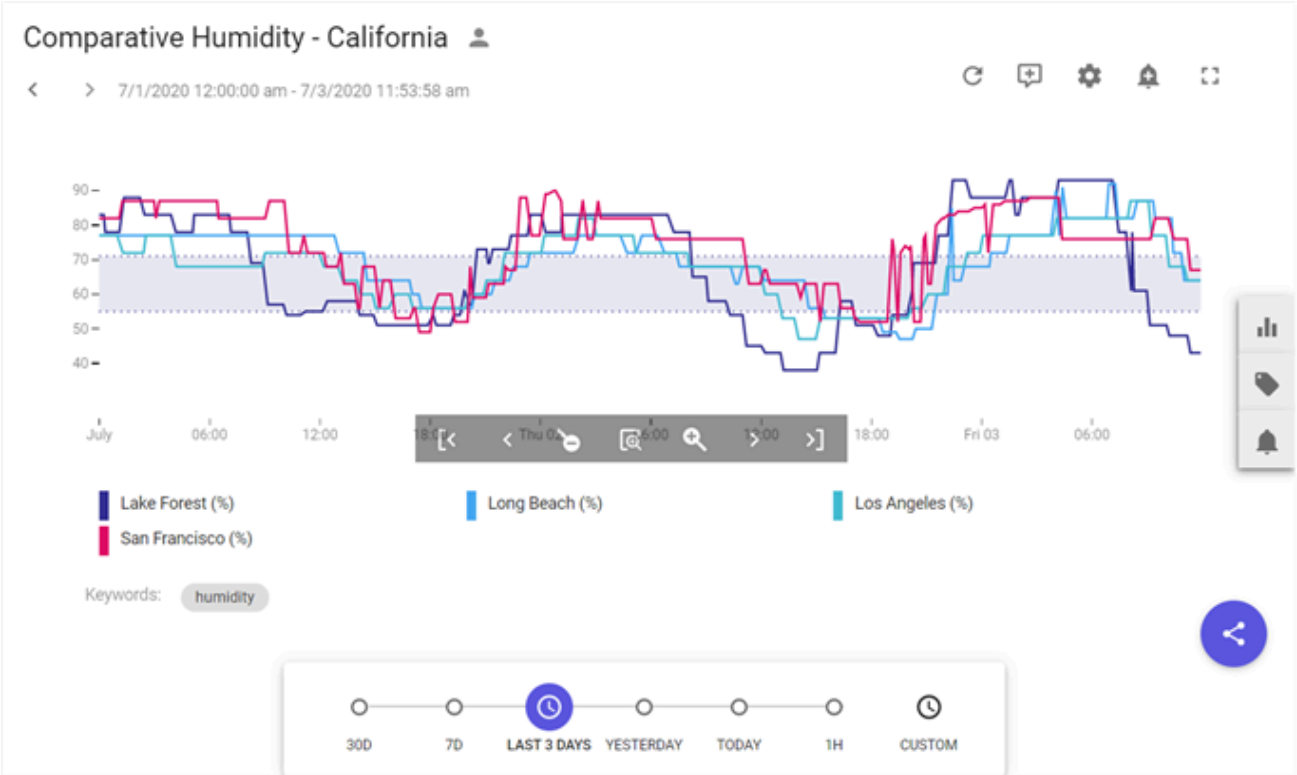
Expand detail left or right (line chart)

You can use a mouse or touch screen to expand detail to the left and right to explore chart data. When you expand details, the time is expanded by half of the current duration.

To expand chart details to the left or right

1. Hover over the X (horizontal) axis of your line chart to see the pan and zoom controls.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Select [<] to expand the chart to the left.

   Or, select [>] to expand the chart to the right.

Apply chart options

You can further constrain the data you want to see by selecting:

- Retrieval mode and reporting interval
- Timeframe you want reported

Beyond choosing your data and chart type, you can customize your chart these ways:

- Stack or merge charted tag values
- Display or hide a tag's values
- Add a limit line to a chart
- Adjust Y-axis autoscaling for line charts
- Annotate your chart with comments

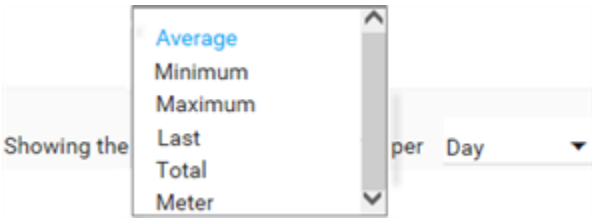
Change retrieval mode and interval for your chart

In AVEVA Historian Client Web, you can change the data values you view by using one of six retrieval modes: average, maximum, minimum, last, total, or meter. You can also choose to report daily or hourly values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To specify a retrieval mode and interval

1. Below your chart, from the "Showing the" dropdown list, select a retrieval mode.



2. In the second dropdown list, next to "per", select the interval: **Day** or **Hour**.

   The retrieval mode and interval that you select affect the data shown in your chart.

Understand retrieval mode

You can choose to see average, minimum, maximum, last, or sum data values.

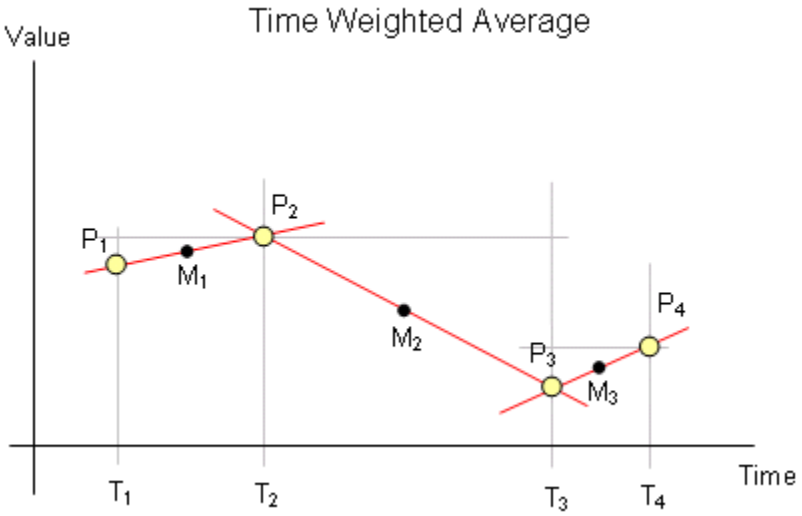
- **Average**

  When you choose to view the Average for your data, AVEVA Historian Client Web uses a time-weighted average algorithm to calculate each data value shown.

  With Average, values are multiplied by the time difference between the points to determine the time-weighted value. So, the longer a data tag had a particular value, the more weight that value holds in the overall average. The overall average is determined by adding all of the time-weighted values and then dividing that number by the total amount of time.

  For the following data values of a tag that uses linear interpolation, the time-weighted average is computed as:

  $(((P_1 + P_2) / 2) \times (T_2 - T_1)) + (((P_2 + P_3) / 2) \times (T_3 - T_2)) + (((P_3 + P_4) / 2) \times (T_4 - T_3)) / (T_4 - T_1) = \text{Average}$



  If the same data tag uses stair-step interpolation, the time-weighted average is:

  $((P_1 \times (T_2 - T_1)) + (P_2 \times (T_3 - T_2)) + (P_3 \times (T_4 - T_3))) / (T_4 - T_1) = \text{Average}$

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Maximum**
  When you choose to view the Maximum for your data, AVEVA Historian Client Web shows the maximum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. NULL is returned if the cycle contains one or more NULL values.

  The number of cycles is based on the specified resolution or cycle count. However, maximum retrieval is not a true cyclic mode. Apart from the initial value, all points returned are delta points.

  Maximum retrieval only works with analog data. For all other data, normal delta results are returned.
- **Minimum**
  When you choose to view the Minimum for your data, AVEVA Historian Client Web shows the minimum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. NULL is returned if the cycle contains one or more NULL values.

  The number of cycles is based on the specified resolution or cycle count. However, minimum retrieval is not a true cyclic mode. Apart from the initial value, all points returned are delta points.

  Minimum retrieval only works with analog data. For all other data, normal delta results are returned.
- **Last**
  When you choose to view the Last for your data, Insight shows the final data value recorded for the timeframe you specify.
- **Total**
  When you choose to view the Total for your data, AVEVA Historian Client Web uses a integral algorithm to calculate each data value shown.

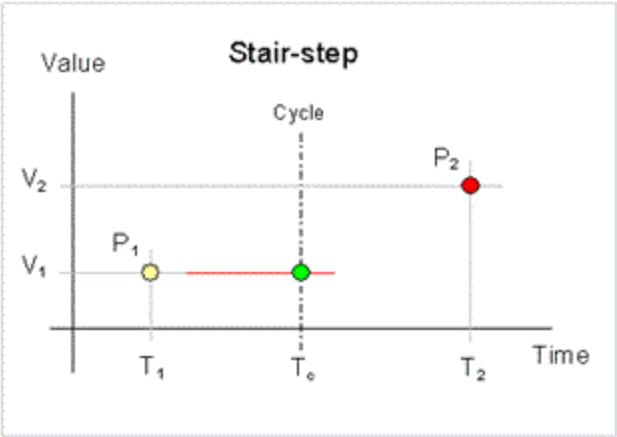
  The total equals the time-weighted average multiplied by the total amount of time in the given timeframe. For example, if the time-weighted average for a tag during a 1-minute cycle is 3.5 liters per second, the total (calculated by the integral retrieval mode) is 210 for that cycle (3.5 liters per second multiplied by 60 seconds).
- **Meter**
  When you choose to view the Meter for your data, AVEVA Historian Client Web gives you the total counted units for this tag from the beginning to the end of the time period.

Understand interpolation type

Analog tag values can be retrieved using stairstep or linear interpolation methods to help calculate a value if none is stored for a given point in time.

- **Stairstep:** The value at the cycle boundary is assumed to be the same value as the last stored value before the cycle boundary.The last known point is returned with the given cycle time. If no valid value can be found, NULL is returned.

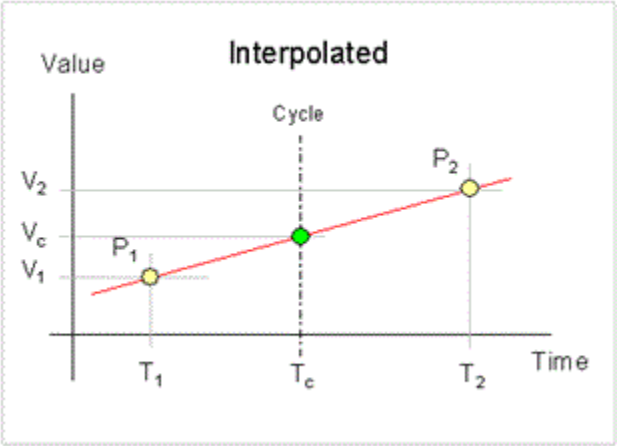
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- **Linear:** The historian calculates a new value at the cycle boundary by interpolating between the last stored value before the boundary and the first stored value after the boundary. If either of these values is NULL, it returns the last stored value before the boundary.

  Expressed as a formula, V_c is calculated as:

  $V_c = V_1 + ((V_2 - V_1) * ((T_c - T_1) / (T_2 - T_1)))$



Select a preset timeframe

To select a timeframe for your tag analysis

- Select an item on the timeline.

The timeframe you select affects the data shown in your chart. If you select a preset timeframe, such as "Today" or "Last 3 Days" the datestamp for the displayed data will be relative to the date you view it. For example, suppose you define and save content on January 4th that specifies "Yesterday" as the timeframe. When you view that data on January 4th, the data will be from January 3rd. If you open the saved content a week later, on January 11th, the data will be for January 10th.

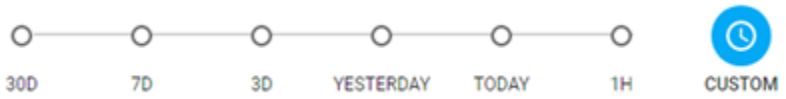
Alternately, you can define a customized timeframe that is not relative to the date you view it.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

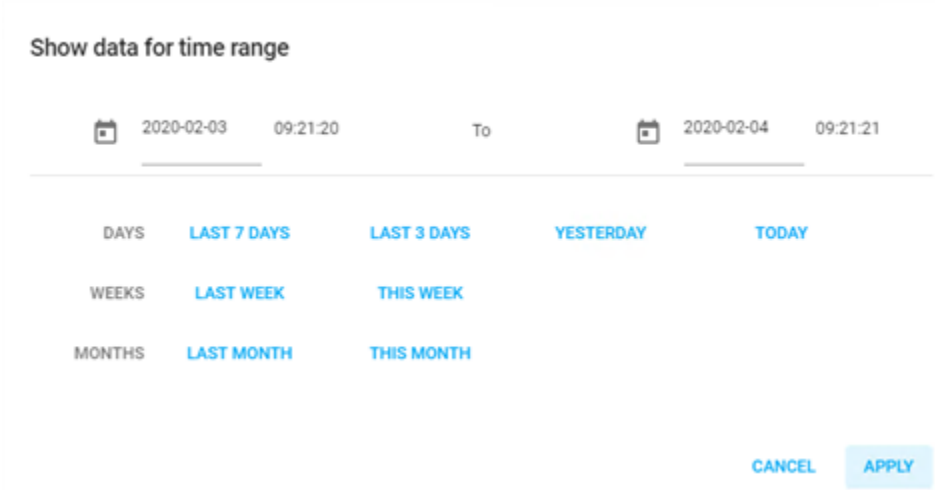
Define a customized timeframe

To select a timeframe for your chart

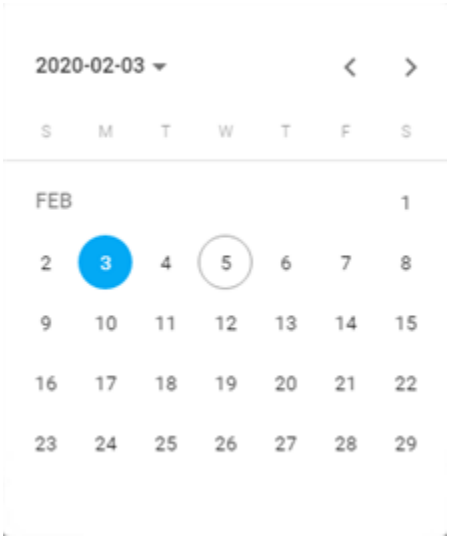
1. From the timeline, select **Custom**.



The Show Data for Time Range box displays:



2. Set specific start and end dates and times:
   a. Select to set a specific start date.



   b. Specify a start time using HH:MM:SS format.
   c. Do the same for an end date and time.

Note: Optionally, instead of specific dates and times, you can select a relative timeframe (for example, "Today", "Last Week", or "This Year").

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

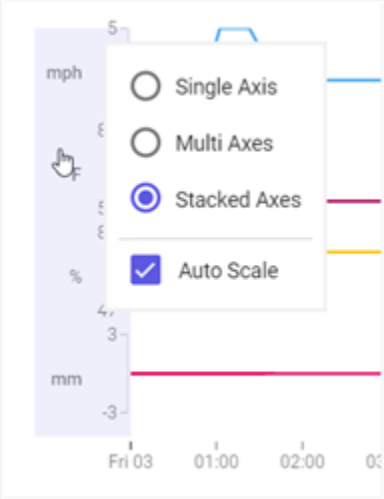
3. Select **Apply**.

Stack or merge charted tag values

You can change how values are recorded and applied on the Y axis (the vertical axis) for a chart.

To select stacked or merged chart options

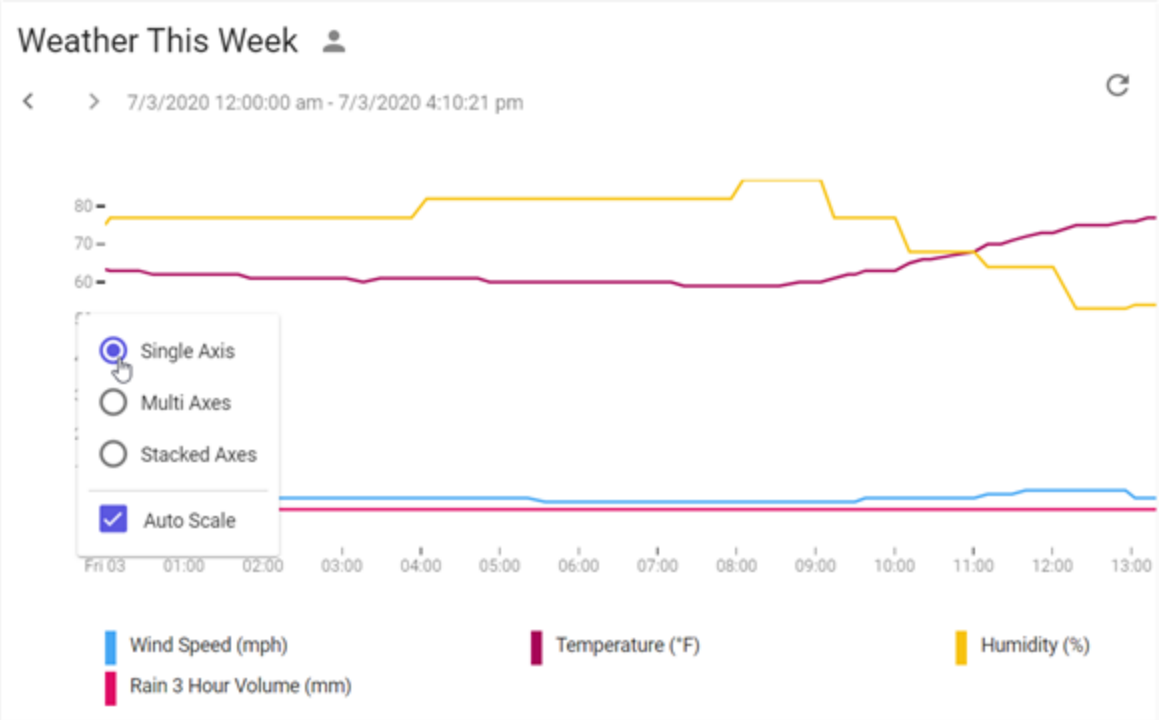
1. Click anywhere in the Y-axis area of your chart.
2. From the option menu that displays, select an option for the chart's Y axis:
   - **Single axis.** Shows a single Y-axis scale that is shared by all tags in the chart.
   - **Stacked axes.** Show individual scales on the Y axis for each tag and stack the single-tag charts vertically. This is the default option.
   - **Multi axes.** Does not show Y-axis values. Instead, the Y-axis is shared by all tags, but tags each use their own range of values. All tags are charted on the same chart.
3. Select or clear the **Auto Scale** checkbox to enable disable autoscaling.



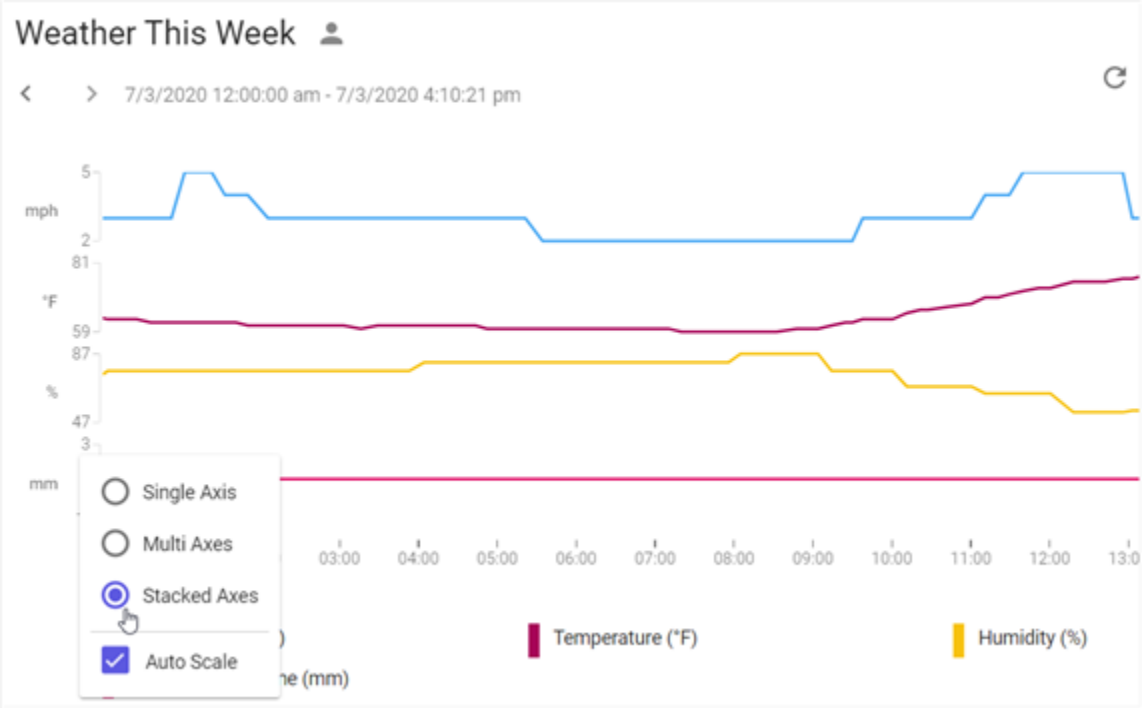
Examples

A **single-scale Y-axis chart** uses one numeric scale for all tags shown on the chart, regardless of the units of measure used for each tag. This example shows autoscaling enabled.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

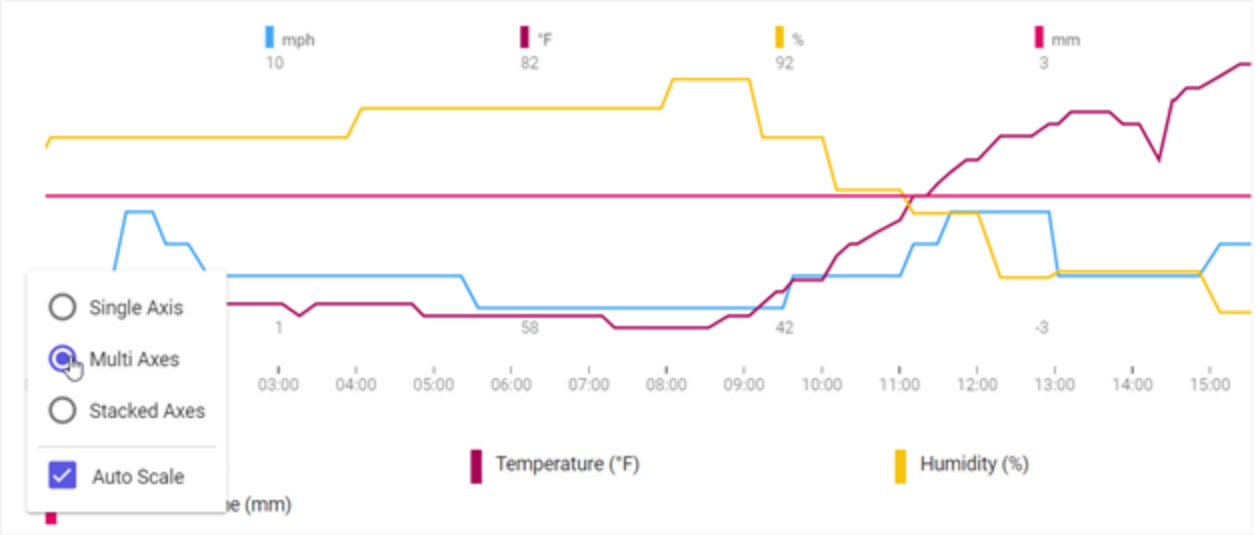


A **stacked Y-axis chart** is a collection of single-tag charts. It uses an individual Y-axis scale for each tag and stacks each of the single-tag charts vertically on the screen.



A **multi-scale Y axis chart** also uses individual Y-axis scales for each tag, but includes all tags on the same chart. Since the Y axis represents multiple scales, the upper value for each tag is displayed above of the chart.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Display or hide a tag

You can easily hide or redisplay a tag within a chart.

To hide a tag on your chart

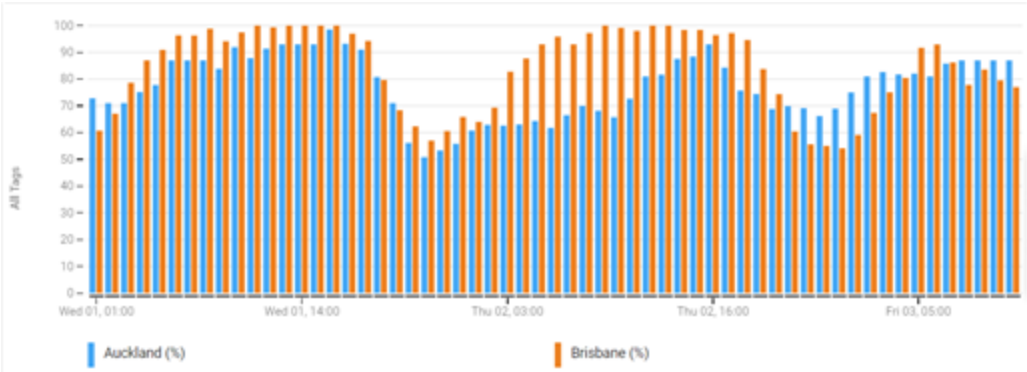
- In the legend below the chart, select the tag to hide it.

Notes:
- When all charted tags are from the same data source, the legend omits the data source name.
- Additionally, if all charted tags have names with a common beginning or ending, that common part may be trimmed in the legend to show only the unique portion of the tag names.
- If a tag has an assigned alias, the alias is used instead.

To redisplay a hidden tag

- Select the tag in the legend again.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If you want to remove a tag from a chart so that it is no longer listed in the legend or Data panel, see Remove tags.

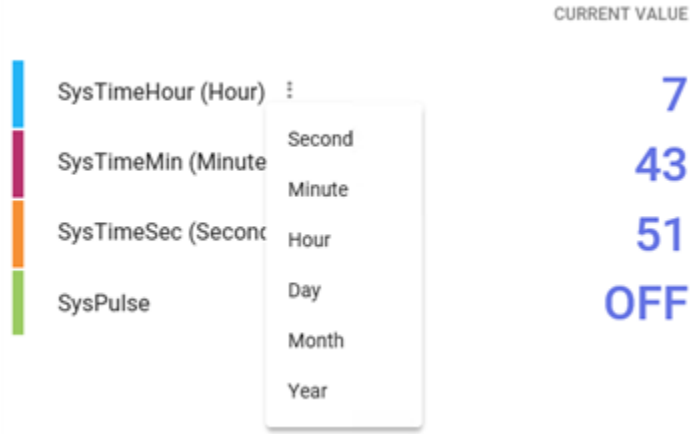
Engineering Unit Conversion

For analog (numeric) tags that are not expression tags, you can select the engineering units for each tag to display in the chart. (Unit of measurement conversion is not supported for expression tags.)

Note: The engineering unit conversion feature is licensed separately. If you do not have the applicable license, the button ⋮ for unit conversion is not displayed.

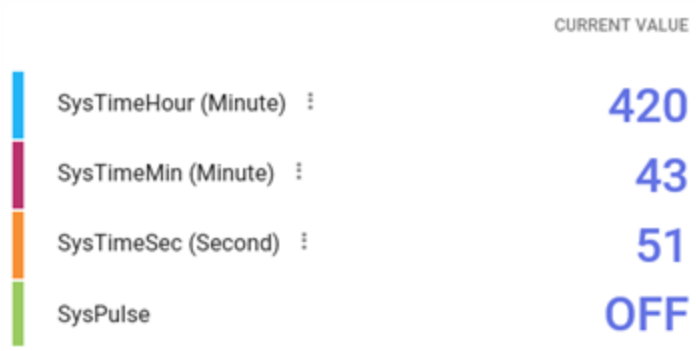
To convert engineering units for tags in your chart

1. In the legend area of the chart, a button ⋮ displays beside each analog tag. Clicking the button displays a popup list of all the related units from the engineering unit catalog for the selected tag.



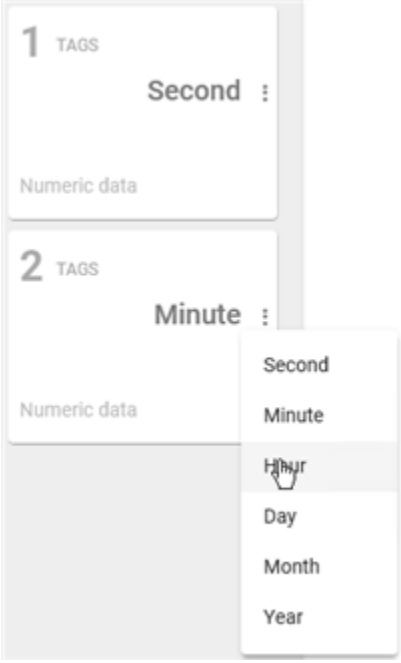
2. Select an engineering unit from the list. The tag data is converted to the selected unit and the chart is refreshed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

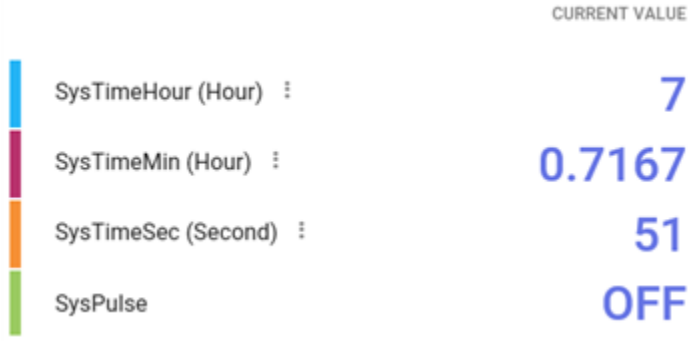


3. You can also change multiple tags at once using the search results on the left side of the page. Search results are grouped into categories which are organized by unit of measure.

   Click the ⋮ button next to the unit of measure on a search category tile.



4. Select a unit of measure from the list, and all tags from the group display in the new unit.



Add a limit line to a chart

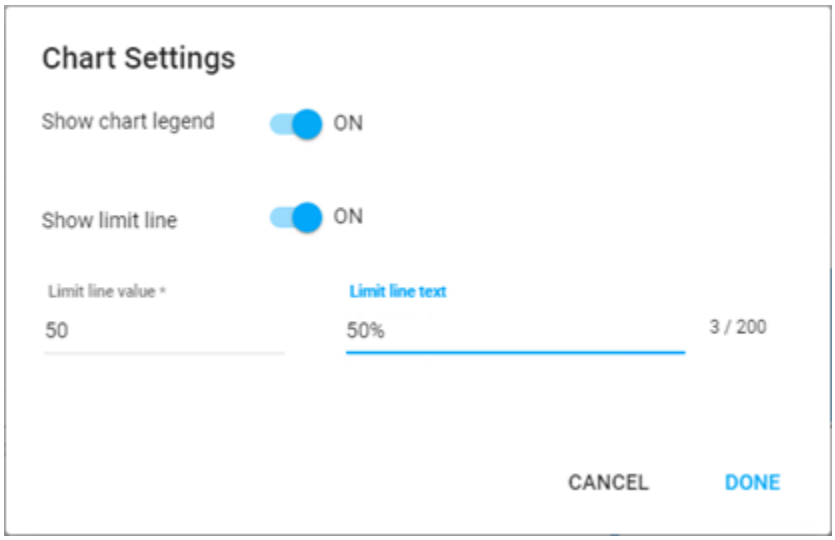
A limit line on a chart makes it easy to detect data values that fall above or below a significant value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

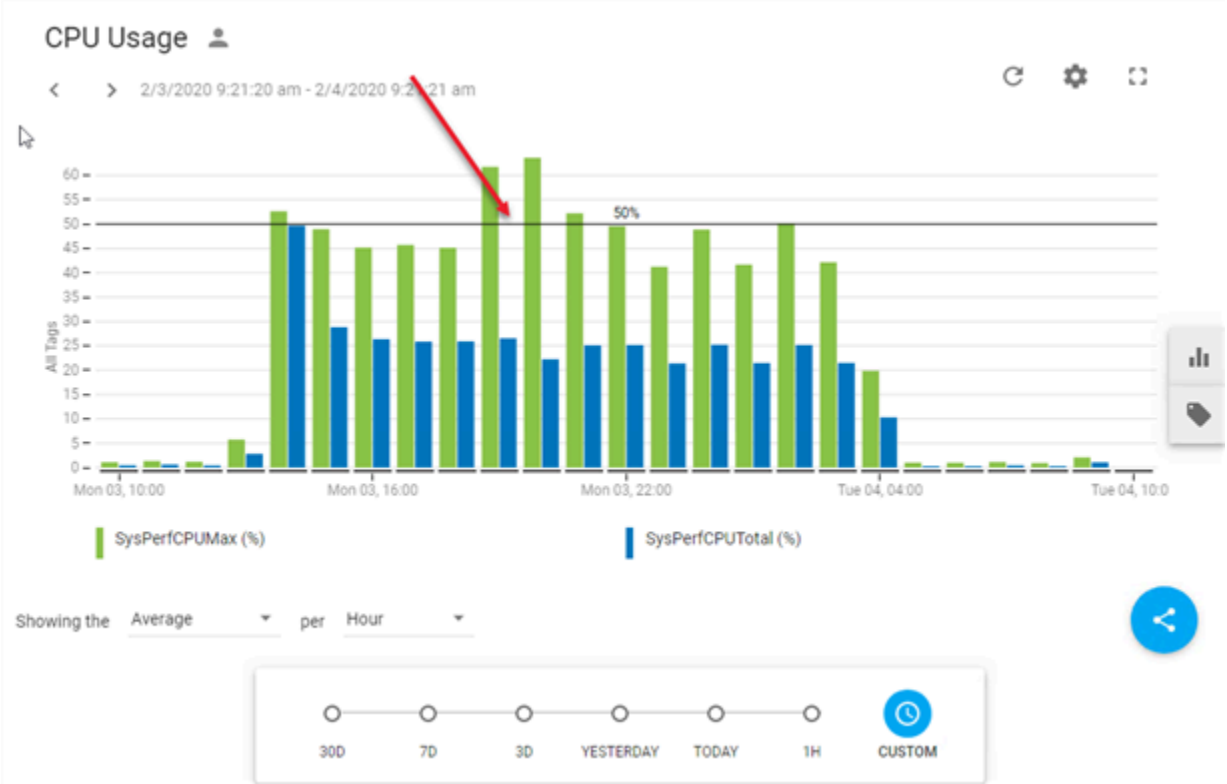
Note: Limit lines are available for only some chart types.

To add a limit line to your chart

1. Select your data and select a chart type.
2. Select ⚙.
3. Type the numeric limit and give the line a name.



4. Select **Done**. Your chart will now include a limit line.



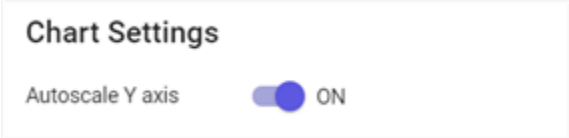
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adjust Y-axis autoscaling for line charts

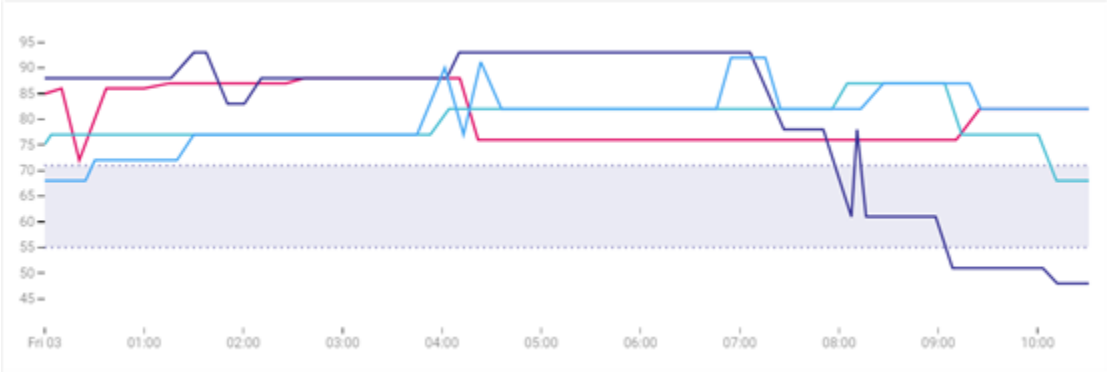
On a line chart, you can scale to actual data values or to full range of potential data values. For example, if a data tag can report values from 0 to 100, but actual results are only 20 to 80, you can enable the autoscale feature to show the range of actual values (20 to 80). If you disable autoscale, you'll see the full range of possible values (0 to 100).

To scale to actual data values

1. At the top of the results chart, select ⚙.
2. Set **Autoscale Y** to ON (slide to right). This sets the Y axis to show the range of only the actual data shown.



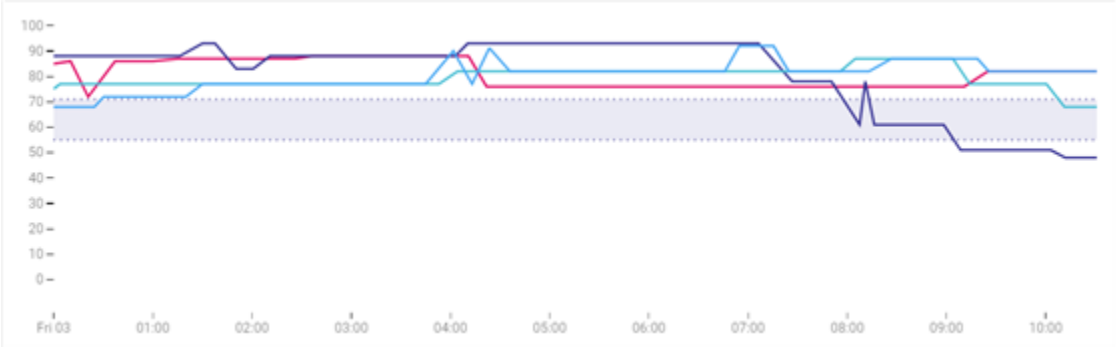
For example:



To show the full data range

1. At the top of the results chart, select ⚙.
2. Set **Autoscale Y** to OFF (slide to left). This sets the Y axis to show the full range of possible data values.

   For example:

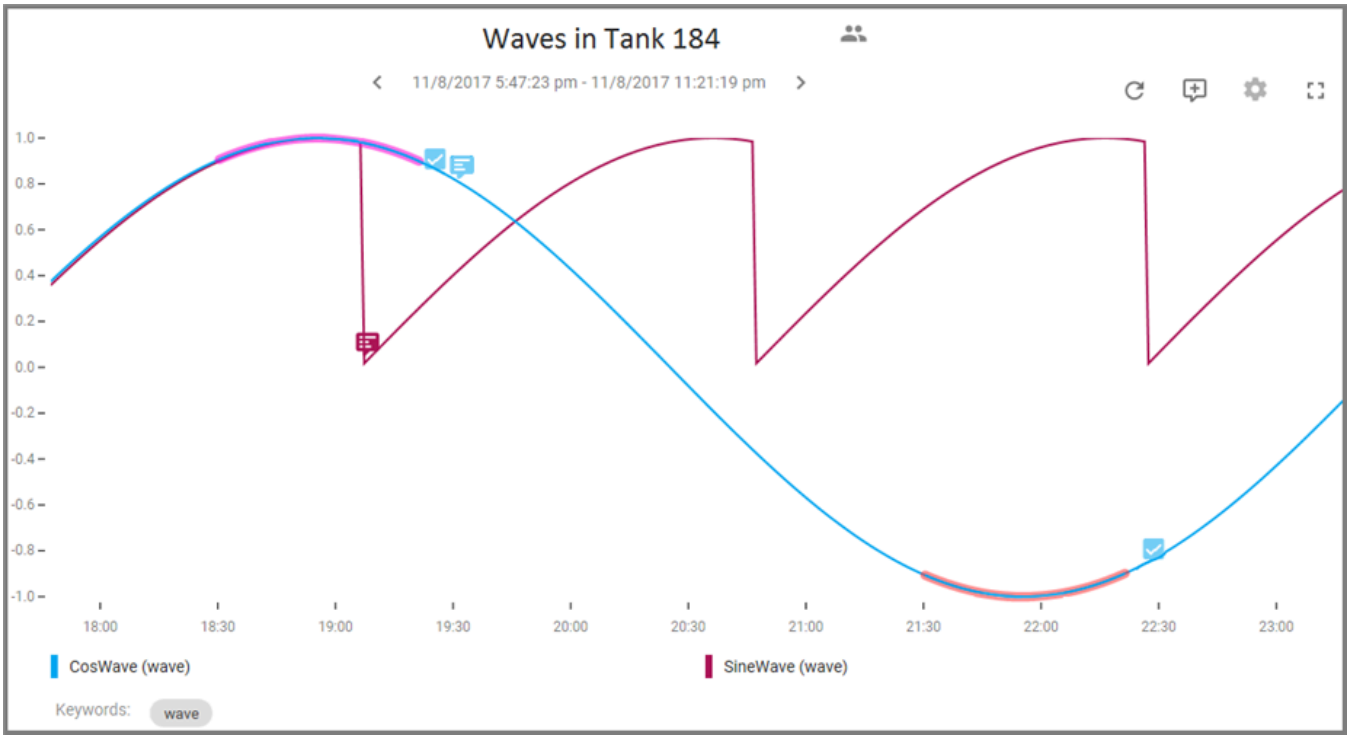


Track comments and events

In addition to process data, your saved content can contain comments and events.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Note: Comments and events are supported on line charts only.



Add comments to a chart

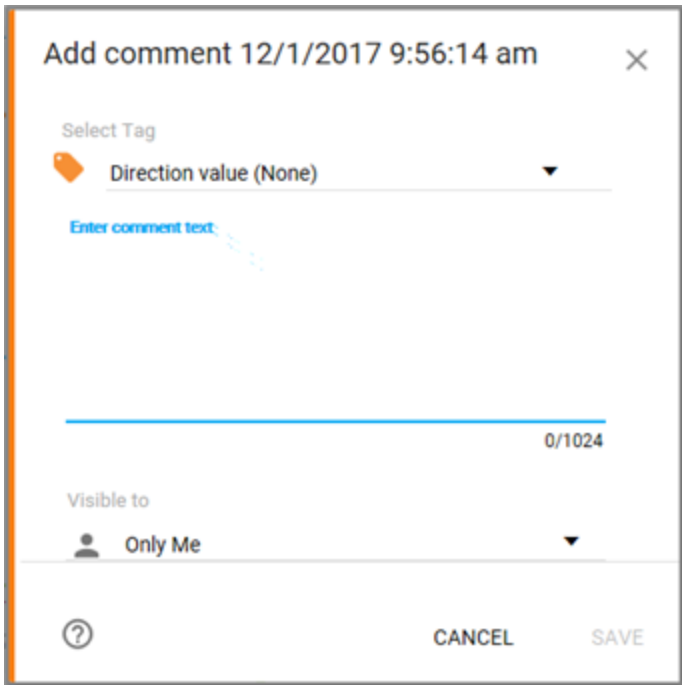
With comments, you can add clarification or more information to a specific point in time for any tag you've charted.

You can add comments to tags in any line chart.

To add a comment to your chart

1. In the upper-right corner, select .
2. On your chart, click a point in time to open the Add Comment dialog box.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

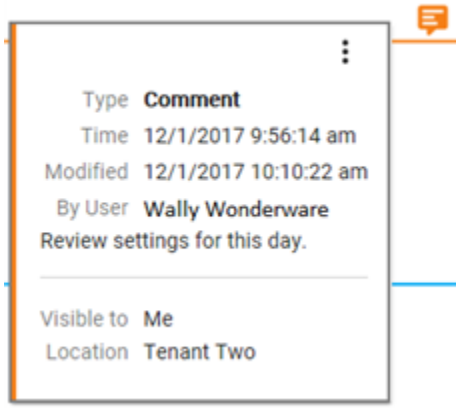


3. From the **Select Tags** dropdown, select the tag to which you want to associate your comment.
4. Type a comment in field provided.
5. From the **Visible to** dropdown, specify who can see this comment:
   - Only yourself.
   - Your team members. If you are part of multiple teams, specify which team.
6. Select **Save**.

Edit your comment

To revise a comment you've added to a chart

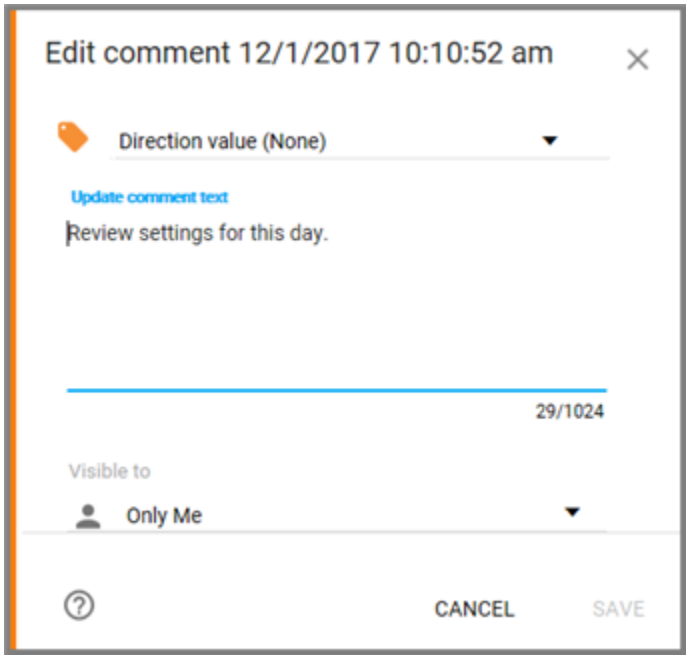
1. In the chart, locate and click for your comment.



2. In the upper-right corner of the Comment dialog, click ⋮.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

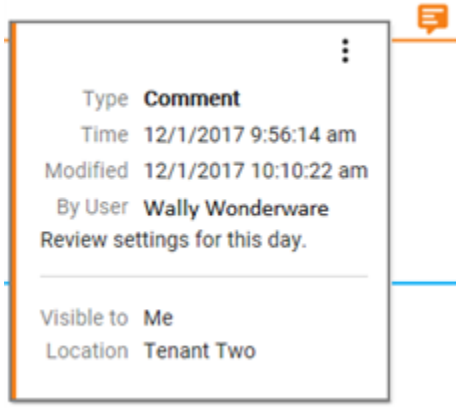
3. Click **Edit**.
4. Edit the comment as needed.
5. From the **Visible to** dropdown, change who can see this comment, if needed:
   - Only yourself.
   - Your team members. If you are part of multiple teams, specify which team.
6. Click **Save**.



Delete your comment

To delete a comment from a chart

1. In the chart, locate and click for your comment.



2. In the upper-right corner of the Comment dialog, click ⋮.
3. Click **Remove**.

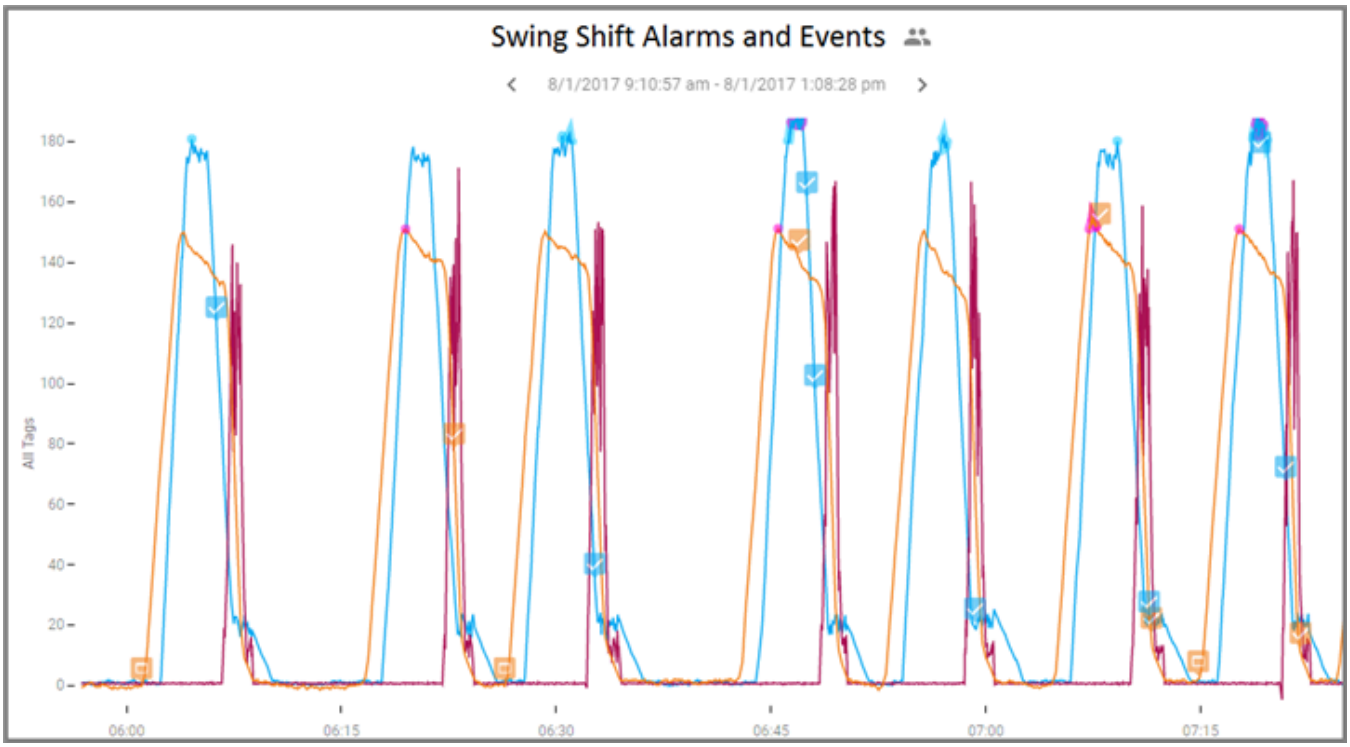
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

4. A warning dialog displays. Click **Remove** to confirm.

View events on a chart

AVEVA Historian Client Web supports AVEVA Application Server events captured through AVEVA Historian replication or the AVEVA Historian SDK. (For more information on Application Server events, see the AVEVA System Platform documentation.)

When you chart tags with associated events on a line chart, those events will be displayed as part of your line chart.



Note: Alarms and events are not supported for discrete or string tags in AVEVA Historian Client Web. (While you can include alarm and event information for discrete tags using AVEVA Historian Client Trend, it is not currently possible using AVEVA Historian Client Web.)

On a line chart, AVEVA Historian Client Web can display:

	Alarm state The alarm state is displayed as a highlighted line that begins with an Alarm.Set event and ends with an Alarm.Clear event. **Note:** If either the Alarm.Set or Alarm.Clear for an alarm occurs outside the displayed timeframe, the highlighted are will still display. However, if both Alarm.Set and Alarm.Clear for the same alarm occur outside of the displayed timeframe, the highlight will not be displayed.
	Alarm Acknowledge event

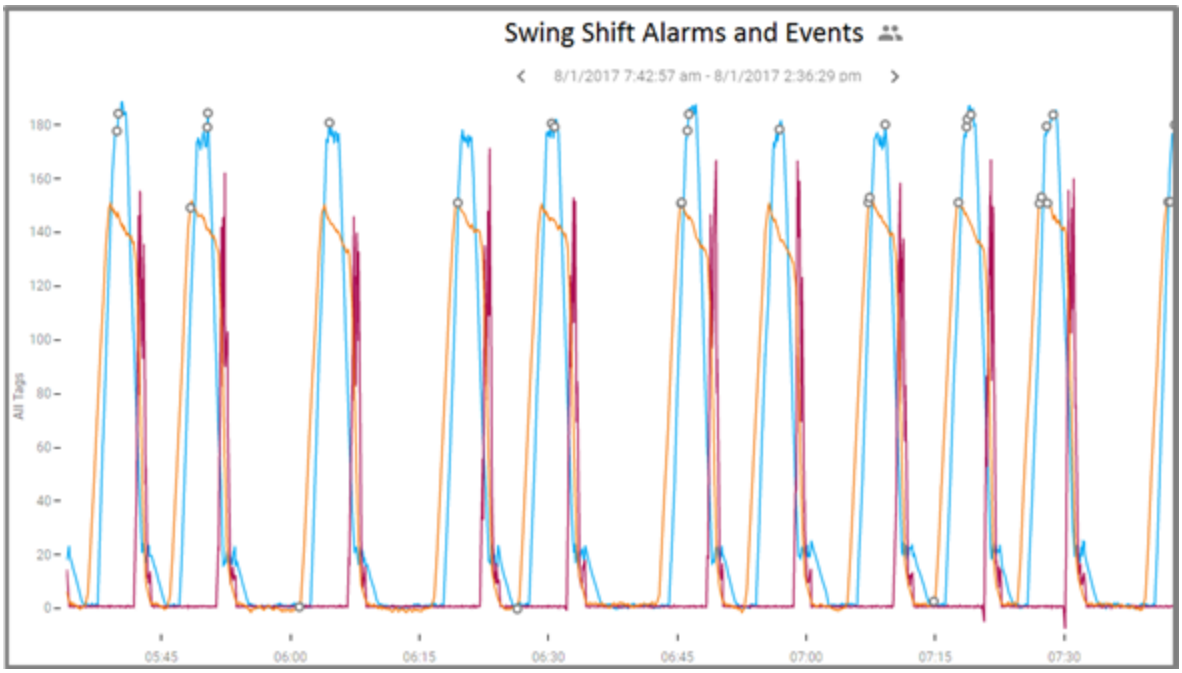
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	When an alarm is acknowledged, that point in time is marked by an Alarm Acknowledge icon (left).
	Application Write event When the application records an event, it is marked with an Application Write icon (shown left).
	User Write event When a user records an event, it is marked with a User Write icon (shown left).
	User Write Secured event When a User.Write.Secured event occurs, it is marked with a User Write Secured icon (shown left)
	User Write Verified event When a User.Write.Verified event occurs, it is marked with a User Write Verified icon (shown left)
	Multiple events To reduce clutter, if there are too many events to display neatly for the timeframe presented, AVEVA Historian Client Web uses the Multiple Events icon (left). This occurs if any tag displayed has more than 30 events for the timeframe shown. If you zoom in, you will see individual event and alarm icons. If you zoom out far enough, the Multiple Events icon disappears.

Note: Events can appear in the display area for a chart even if no tag data is available.

When the displayed timeframe includes one or more tags with more than 30 events, individual event icons are replaced with Multiple Events icons.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



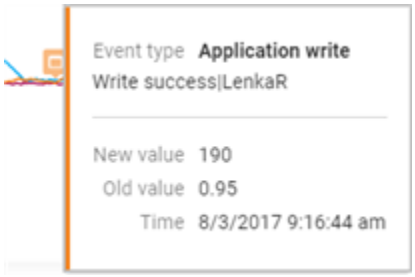
If more than 100 events per tag occur within the displayed timeframe, you'll see no icons Instead, you'll see a toast notification saying there are too many events to display. In this case, you can zoom in to see more events details.

To view event icons when none display

- To see individual event icons, zoom in.

To view details about a single event

- Click the event icon to see its details.

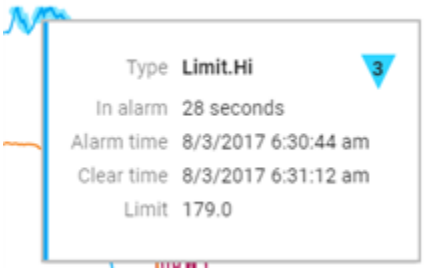


To view details about an alarm state

- Click anywhere on the highlighted area.

The details box includes a color-coded alarm state icon:

- 1 (Red) = Critical
- 2 (Yellow) = High
- 3 (Blue) = Medium
- 4 (Bright Pink) = Low

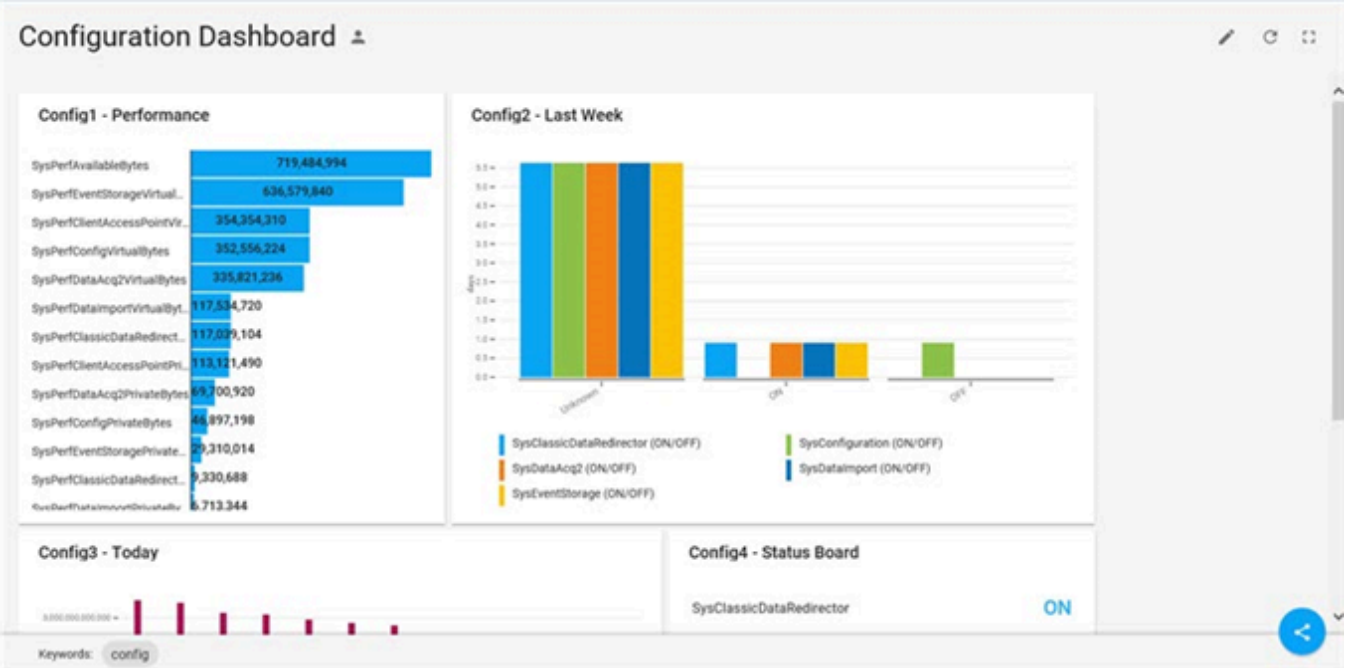


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Work with dashboards

Dashboards give you a "big picture" view of related information by displaying related charts together in one place.

In AVEVA Historian Client Web, you can create and share dashboards.



Tip: Select a chart to see a more detailed view. You can also show or contact the content owner.

For dashboards, data is refreshed at this rate:

Type of data	Refreshed this often
Less than 1hour	30 seconds
1 hour to 1 day	1 minute
1 to 5 days	5 minutes
5 days to 1 month	15 minutes
More than 1 month	1 hour

About keywords and dashboards

Keywords can be added to charts. Then you can use those keywords to identify which charts you want to group together in a single dashboard.

For example, if you saved several charts related to your Seattle operations and tagged them all with the keyword "seattle", you could easily search for that keyword and view all your saved content on one page.

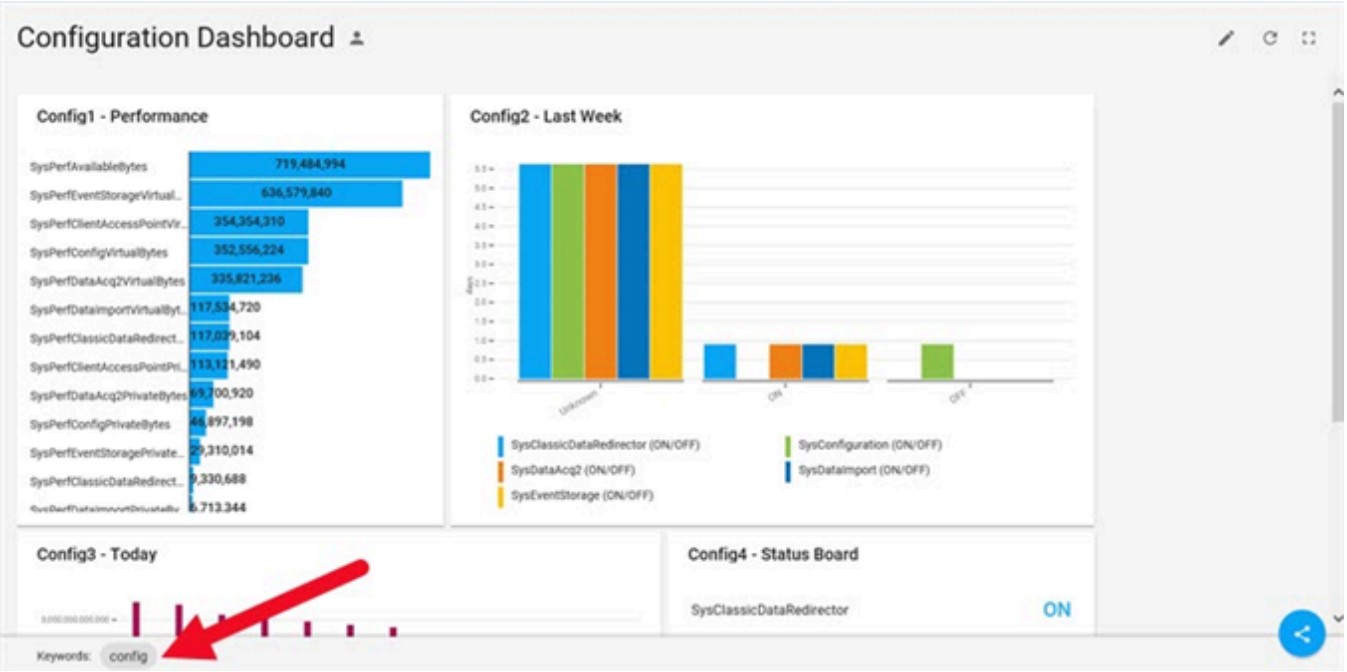
Use multiple keywords to make your dashboard more specific. For example, to create a dashboard of charts related specifically to power operations in Seattle, you would enter this in the Search field (assuming charts have

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

been created with related keywords):

```
power seattle
```

The resulting dashboard will show only those charts and grids that are marked with both those keywords:



If you save a dashboard, then add that dashboard's keyword(s) to other charts, those charts will not be automatically added to the saved dashboard. However, when you view the dashboard, a prompt will appear at the bottom of the page that there is additional content that matches the chart's keywords. Select **ADD CONTENT** to add the new charts to the dashboard.

If you remove a keyword from a chart that is part of a saved dashboard, it won't be removed automatically from the dashboard.

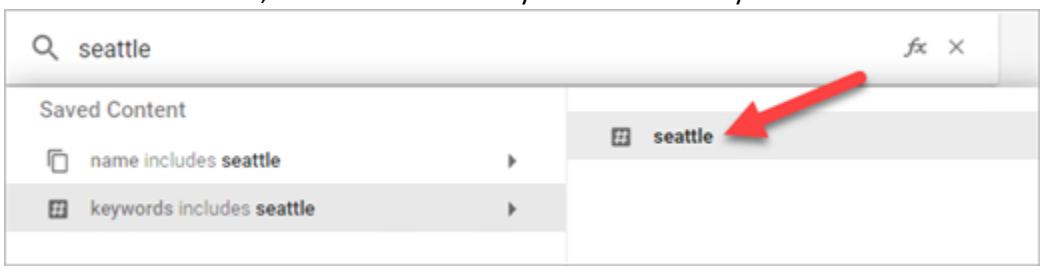
Once a dashboard has been created, you can add or remove any chart to it, regardless of keywords.

Create a dashboard

A dashboard is a group of related charts that you can view together on one screen.

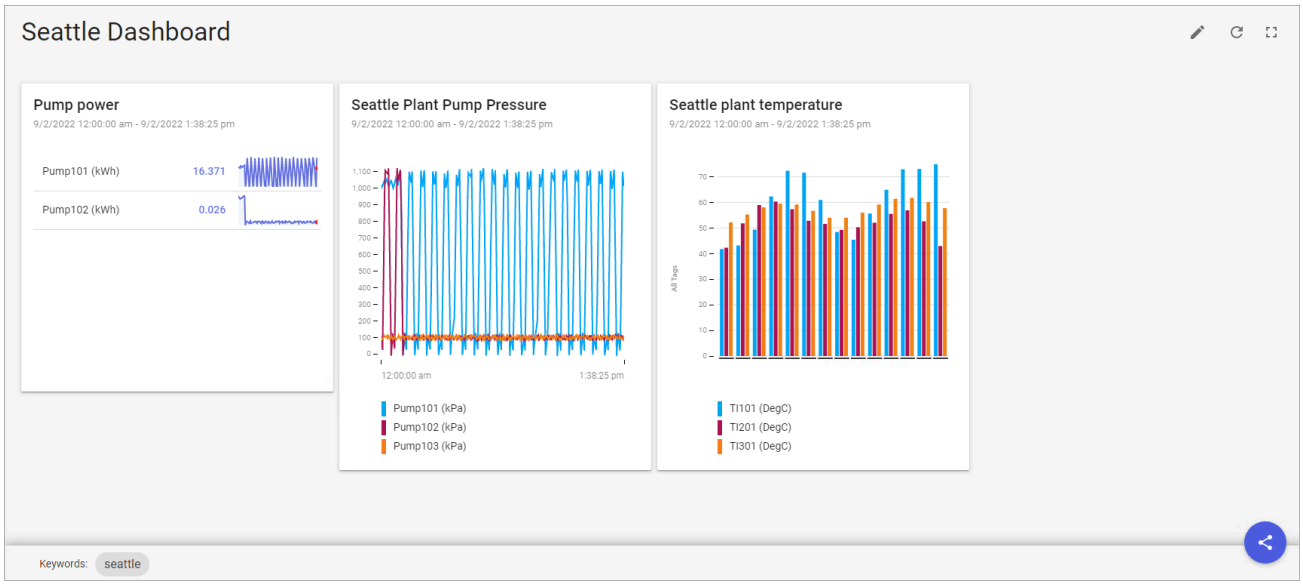
To create a dashboard

1. To create a dashboard, you must have at least one chart that has been saved with a keyword.
2. Search for the keyword or keywords you want to associate with this dashboard. If you are searching for multiple keywords, enter each keyword in the search bar, separated by a space.
3. Under **Saved Content**, select the desired keyword or set of keywords from the list.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The dashboard is created. It includes all charts that are associated with the selected keyword(s). The keyword(s) you selected appear at the bottom of the dashboard.



4. Select edit to add charts to the dashboard or remove charts from the dashboard.
5. Select Share in the lower-right corner to save or share the dashboard.

Note: When you share a dashboard, all of the associated content is also shared and marked with this icon:

If some of your content isn't shared yet, a prompt asks if you'd like to proceed by sharing both the dashboard and all of its content.

About keywords

You can use keywords to help identify and categorize your saved content. You can also use keywords to search for saved content and to create dashboards of related content.

Keyword facts

- You can add or edit keywords when you save a chart.
- A keyword is required in order to create a dashboard. After a dashboard has been created from a keyword(s), any charts may be added to it, including charts without keywords.
- You don’t have to register or create keywords anywhere except on the content itself.
- Keywords can contain any combination of letters, numbers, and special characters. No spaces are allowed.
- Searches for keywords are not case-specific; uppercase and lowercase letters will return the same result.
- Keywords can be reused and are not tied to a user.
- You can assign up to 20 keywords to content when you save it.

Add a chart to a dashboard

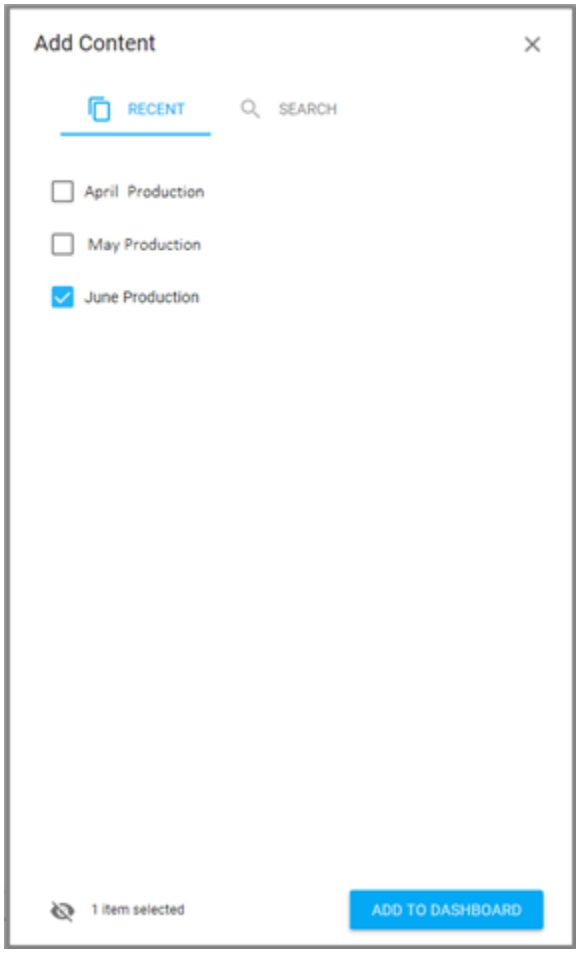
If you are the dashboard's owner, or an administrator, you can add a chart from your dashboard.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To add a chart from a dashboard

1. Select to edit the dashboard.
2. In the upper-right corner of the window, select to add content.

   a. Select each chart that you want to add.
   b. Select **Add to Dashboard**.
   c. Select to save your changes.



Remove a chart from a dashboard

If you are the dashboard's owner or an administrator, you can remove charts from a dashboard.

To remove a chart from a dashboard

1. Select to edit the dashboard.
2. Select In the upper-right corner of the of the dashboard you want to remove.
3. Select to save your changes.

Save a dashboard

To save a dashboard

1. Create a dashboard.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. From the lower-right corner, select .
3. Select Save .
4. Type a name for your dashboard.
5. Select **Save**.

Now your dashboard is available whenever you want to reuse it.

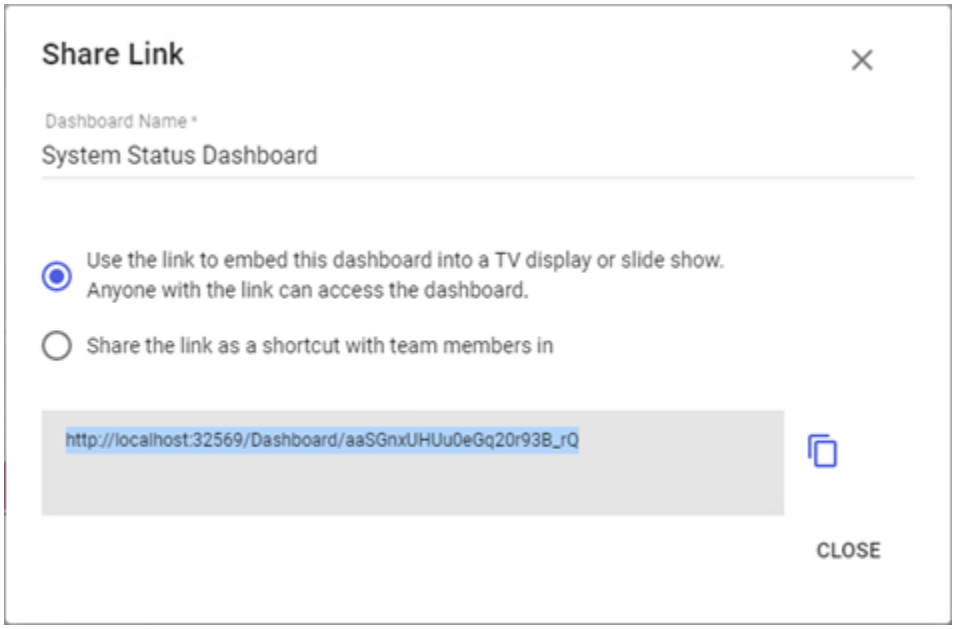
Share a link to a dashboard

AVEVA Historian Client Web allows administrators to copy and share a link to your saved dashboard. Anyone with this link can access the dashboard.

To share a link to your saved dashboard

1. In the lower-right of the window, select , then select Share Link .

   The Share Link dialog shows a link that allows public access to the dashboard. Anyone with this link can view, but not change, the dashboard.

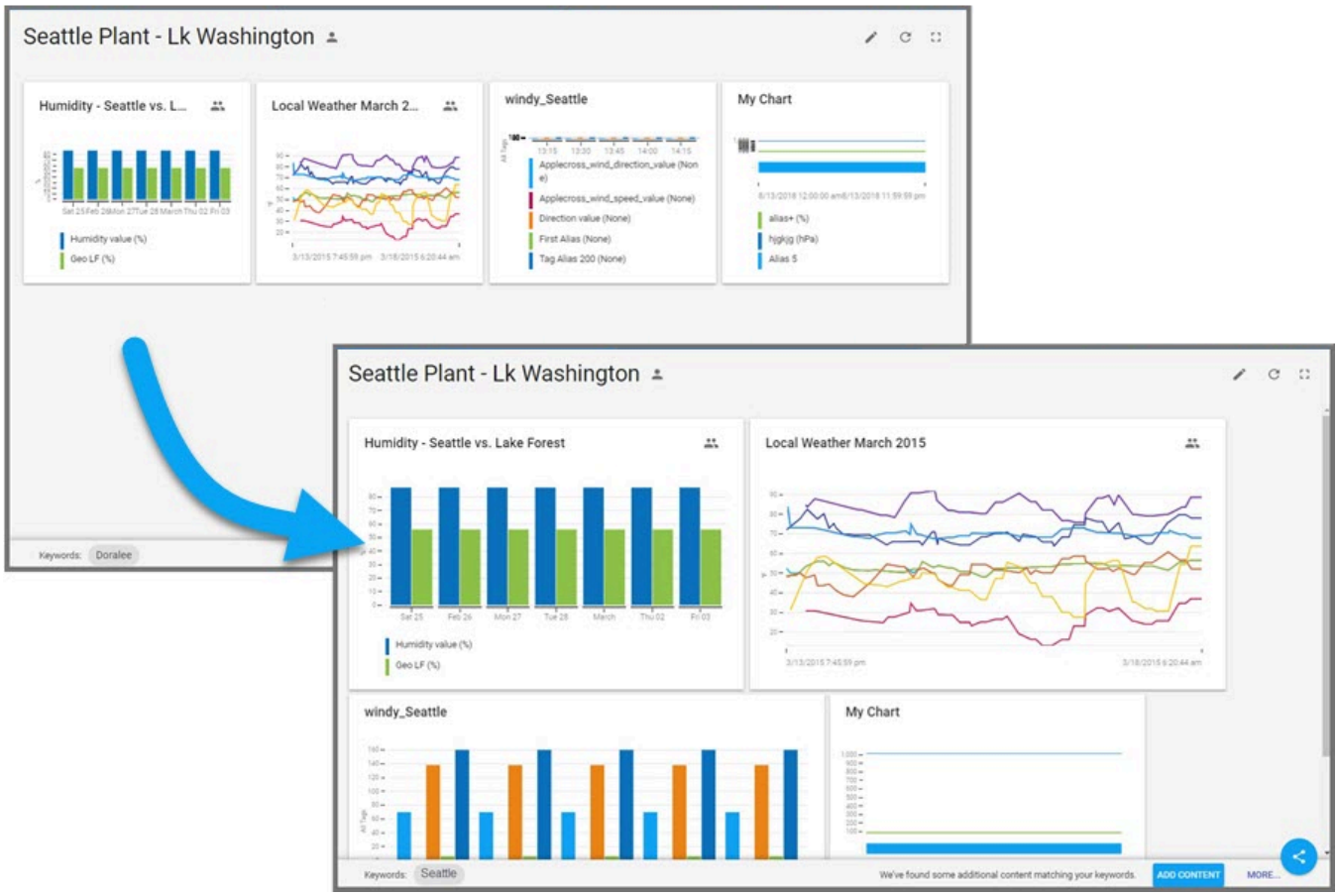


2. Select to copy the link to the clipboard. Then you can:
   - Paste it into any document and share with others.
   - Embed it in a web page or other document to share with others and/or display display publicly (for example, on a large TV screen in the lobby of your building).

Customize dashboard appearance

You can customize the size and location of charts in your dashboard.

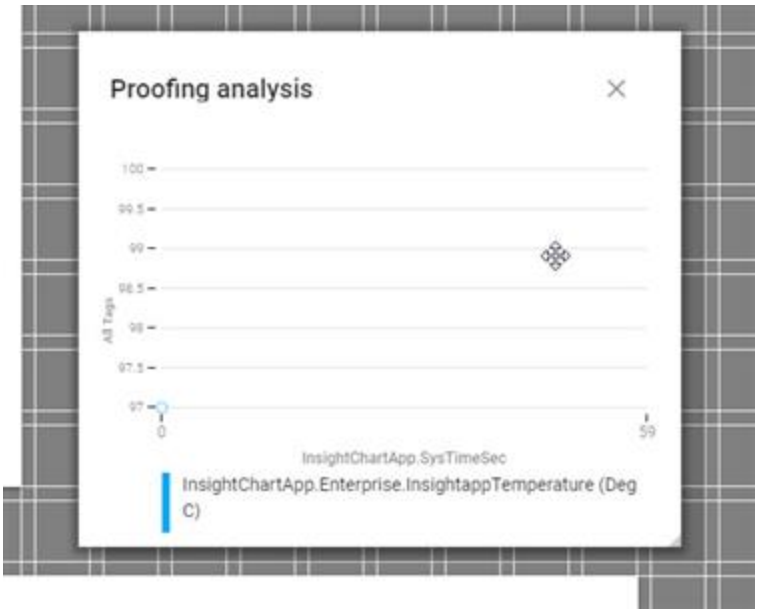
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



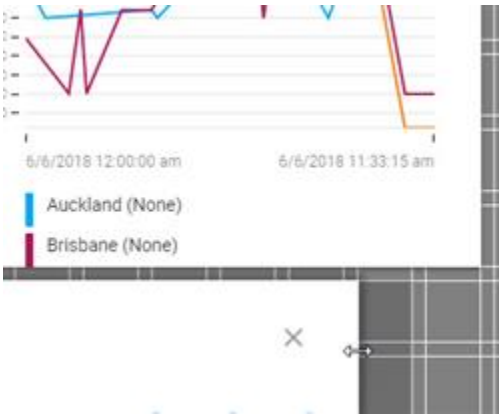
To customize your dashboard

1. Select .
2. Change how the charts appear in your dashboard. You can:
   - **Move a chart** -- Drag a chart (hold down the mouse button or long-press on a touch screen) and move it to a new location.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- **Resize a chart** -- Drag a chart's edge or corner and adjust its length and/or width.



Note: As you move or resize a chart, grid lines appear to help guide you.

3. Select ✓ to save your changes.

   If the dashboard was not previously saved, you will be prompted to save the dashboard.

Save and share content

Use the Share menu to save and share your content.

The content you save includes all data tags and related settings (such as the chart used).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

From the Share menu, you can:

- Save your content
- Share your content with others
- Embed your content elsewhere
- Download content to a CSV file

In AVEVA Historian Client Web, you can also view and reuse recent content.

Save your current content

1. Instead of repeating the steps involved in defining your content, you can save your content with a name and reuse it later.

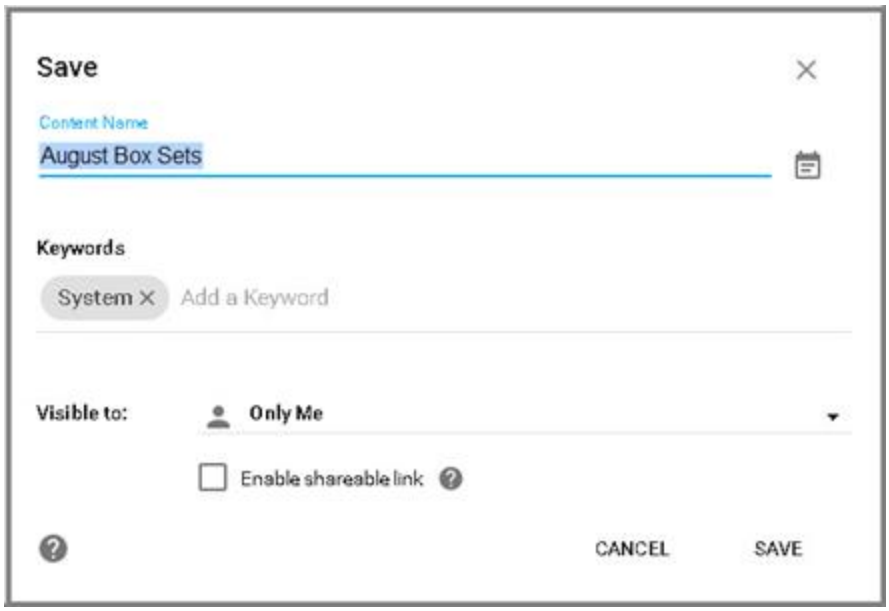
The content you save includes all data tags and related settings (such as the chart used).

Notice that the data values themselves are not saved because the timeframe is relative to the current date and time.

To save your current content

1. Select the data and chart you want to view.
2. Below your chart, select .
3. Select .
4. Type a name for your content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select and choose an option:
   - Choose **Fixed Dates** to keep the specific dates or times you've specified when you reopen this content. (For example, always show data for January 3, 2018.)
   - Choose **Latest** to keep the amount of time you specified, but show the latest data when you reopen the content. (For example, always show data for the current day, "Today".)
6. Optionally, type one or more keywords to associate with your content.

   Keywords let you categorize your content and help you search for it more quickly. You can also use keywords to group related content into a dashboard.
7. In **Visible to**, specify who can see this content.
8. If you want to share a link to this content, mark the **Enable shareable link** check box.
9. Select **Save**.

Now your content is available whenever you want to reuse it.

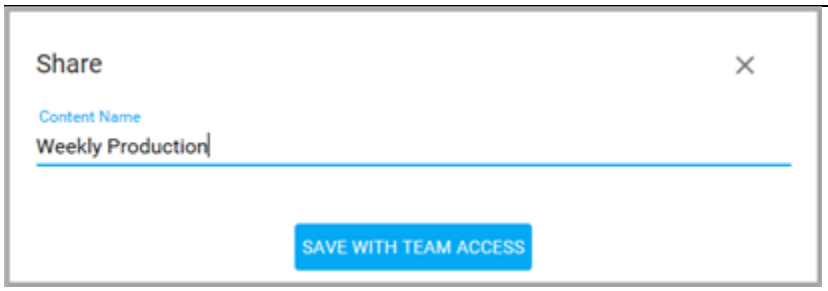
Share content link with others

To share a link to your saved content

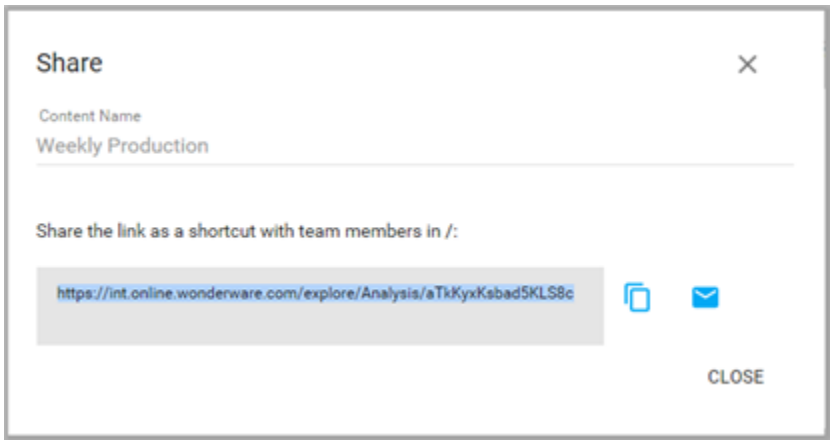
1. On the lower-right, screen, select , and then select Share.

   Note: If you have not saved content before sharing, AVEVA Historian Client Web will prompt you to save it with access for your team first:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Select the way you want to share this content:

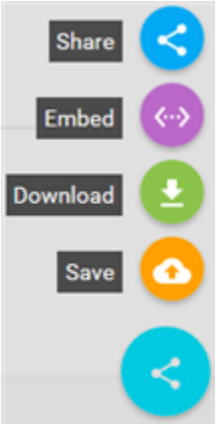


- Select to copy the link to the Clipboard and then paste it in a document for sharing.
- Select to email a link to this content to someone.

Embed content

You can embed content from AVEVA Historian Client Web into a web page or other HTML-based document.

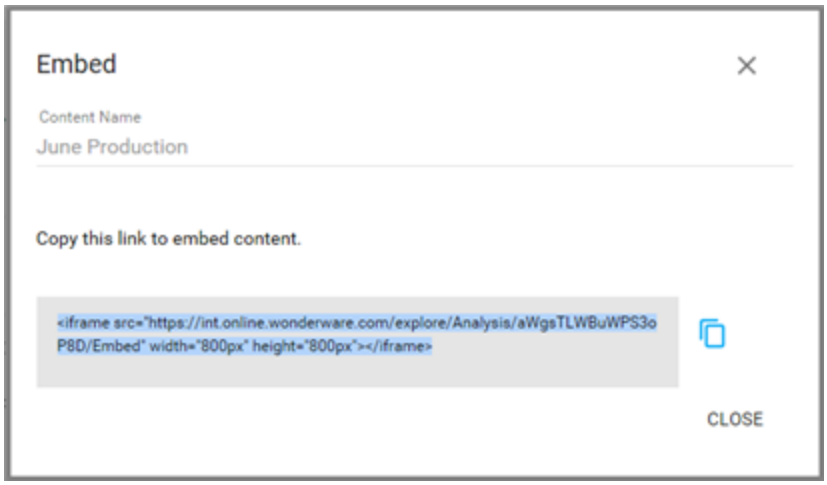
To embed your content elsewhere



1. The content should have the **Enable shareable** link option enabled. If it doesn't, you'll be prompted to enable it when you select Embed in the next step.
2. In the lower-right of the window, select , then select Embed.

   The Embed dialog shows an HTML snippet containing a link to the content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Select to copy this HTML element.

   Note: For iPad, press and hold, and then select the entire link. Then select **Copy** from the iPad options displayed.

4. Paste the copied iframe code into a web page or other HTML-based platform that supports iframes.

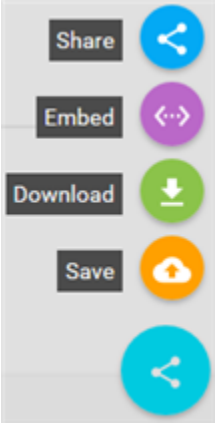
Download content to a file

You can download content to a CSV file for use in other applications, such as Microsoft Excel.

Note: This functionality is not available on iPad.

To download content to a CSV file

1. Below your chart, select .
2. Select Download.
3. Select CSV .
4. Follow the instructions at the bottom of the screen.



Note that the content included in the CSV file reflects what you see in the chart. Typically, there is more detailed data stored than what is shown in the chart.

Download content to an Excel spreadsheet (XLSX) file

If you have configured the Excel add-in, you can also download content to an XLSX file for use in Microsoft Excel. The resulting spreadsheet makes use of the Excel add-in's dynamic functions to retrieve data, depending on the chart type.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

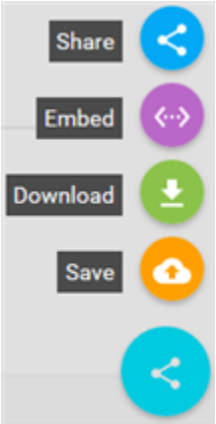
Chart Type	Excel Add-in Page
Status Board Line Chart XY Plot Gaant Chart	Detailed Values • For more information, see Retrieving Detailed Values and Using the HISTORIAN.DETAIL Function.
Column Chart Detail Grid	Periodic Numeric • For more information, see Retrieving Periodic Numeric Values and Using the HISTORIAN.PERIODIC Function.
Summary Chart Summary Grid Diagnostic Grid	Time In State (Discrete/String tags) • For more information, see Retrieving Time in State and Using the HISTORIAN.TIMEINSTATE Function. Periodic Numeric (Analog tags) • For more information, see Retrieving Periodic Numeric Values and Using the HISTORIAN.PERIODIC Function.
Time in State Chart	Time In State • For more information, see Retrieving Time in State and Using the HISTORIAN.TIMEINSTATE Function.
Rolling Sum	Not supported
Cumulative Line	Not supported

Note: If the Excel add-in is not configured, when you open a downloaded XLSX file the add-in function cannot retrieve data, and the formula indicates an error. After configuring the add-in, you can reload the spreadsheet and the data is retrieved correctly. See Registering and Installing the Excel Add-In for more information.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To download content to an XLSX file

1. Below your chart, select .
2. Select .
3. Select EXCEL .
4. Follow the instructions at the bottom of the screen.



Export Data to Excel

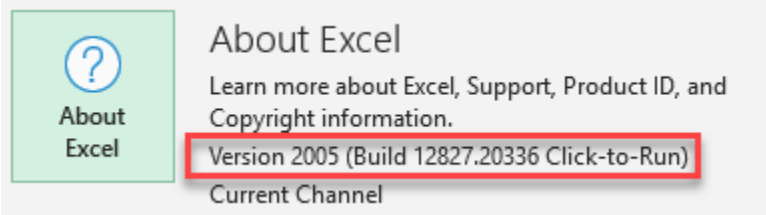
The version of Excel that you use must be version 2002 (Build 12527.20470) or later for complete functionality.

This add-in relies on the Microsoft Excel API version 1.11 for complete functionality. Partial functionality is available with the Microsoft Excel API version 1.9. Please refer to the following document from Microsoft for a list of supported Excel versions by API level: https://docs.microsoft.com/en-us/office/dev/add-ins/reference/requirement-sets/excel-api-requirement-sets.

Note: If you are using a standalone (non-subscription) version of Microsoft Office, you must use Office 2021 or later. Earlier standalone versions of Office do not support the required Excel API version. If you are using Office 365, update to version 2002 or later.

To determine your Excel version:

1. Open an existing workbook in Excel, or create a new one.
2. Select the **File** menu.
3. Select **Account**.
4. Locate the About Excel section on the screen to view the version information:



Minimum Supported Versions of Microsoft Excel

The following table summarizes the minimum supported version of Excel for specific add-in functionality:

Excel Version	Excel Build	Excel API Version	Add-in Functionality
1903	11425.20204	1.9	*Custom Functions* - The ability to use custom

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

			functions is enabled, but the output range must be refreshed manually to update the data.
2002	12527.20470	1.11	*Dynamic Array Formulas* - Custom functions can be used, and the output range is updated automatically when the function parameters change.
2008	13127.20408	1.12	*Date/Time Formatting* - Date/time format defaults to the format defined by the system's regional settings.

Configuring Internet Security Options for the Excel Add-In

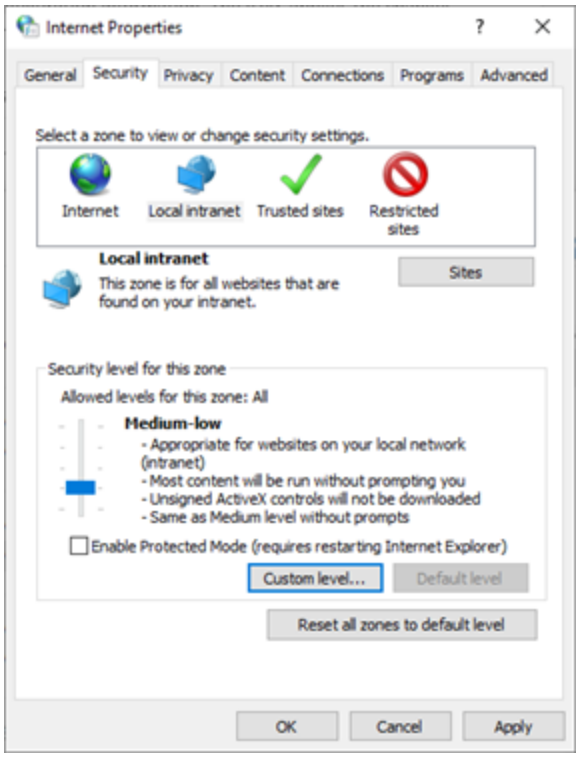
For the Excel add-in to work correctly, some internet security settings may need to be adjusted on your system.

To configure internet security settings for the Excel add-in:

1. Open the **Internet Options/Internet Properties** dialog. You can find this dialog in multiple ways:
   a. Launch Internet Explorer. From the **Tools** menu, select **Internet Options**.
   b. From the Windows Control Panel, launch **Internet Options**.
   c. From the Windows search bar, search for **Internet Options** and launch it from the search results.
2. The **Internet Options/Internet Properties** dialog displays. Select the **Security** tab, then select the **Local intranet** zone.

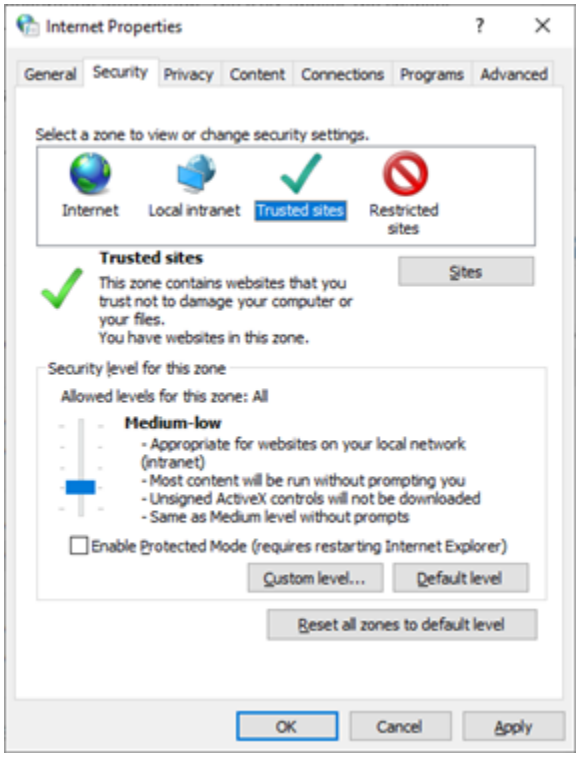
   Set the **Security level for this zone** slider to **Medium-low** for the Local intranet zone, and ensure **Enable Protected Mode** is not selected.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Select the **Trusted sites** zone.

   Set the **Security level for this zone** slider to **Medium-low** for the Trusted sites zone, and ensure **Enable Protected Mode** is not selected.



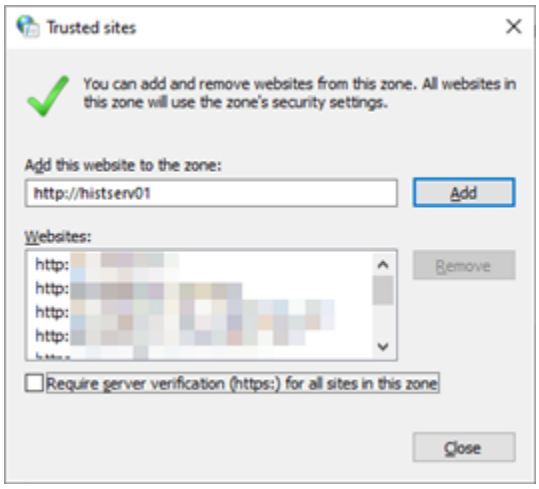
4. With the **Trusted sites** zone selected, click the **Sites** button. The **Trusted sites** dialog displays.

   In the **Add this website to the zone** field, enter your Historian server's name, formatted as

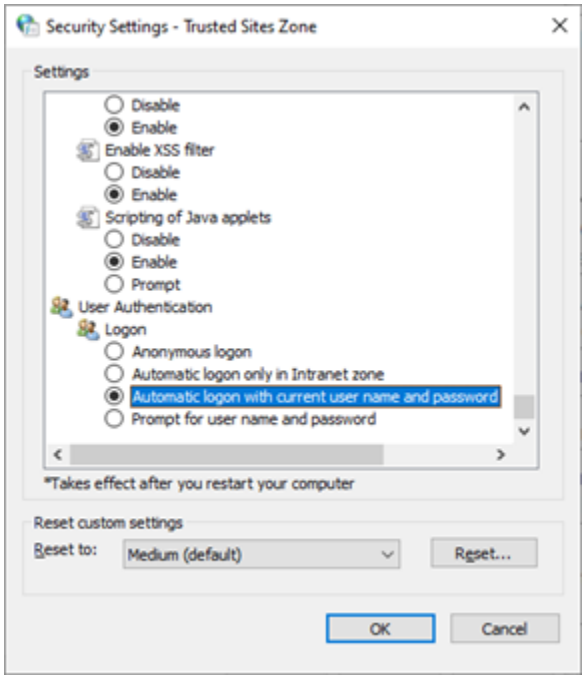
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

`http://your_server_name`, then click **Add**.

For example, if your Historian server is named `histserv01`, add `http://histserv01` to the list of trusted sites.



5. Ensure that **Require server verification (https:) for all sites in this zone** is not selected, then click **Close**.
6. With the **Trusted sites** zone selected, click the **Custom level...** button. The **Security Settings - Trusted Sites Zone** dialog displays.



Locate the **User Authentication** section, and set the **Logon** option to **Automatic login with current user name and password**, then click **OK.** Security configuration for the Excel add-in is complete.

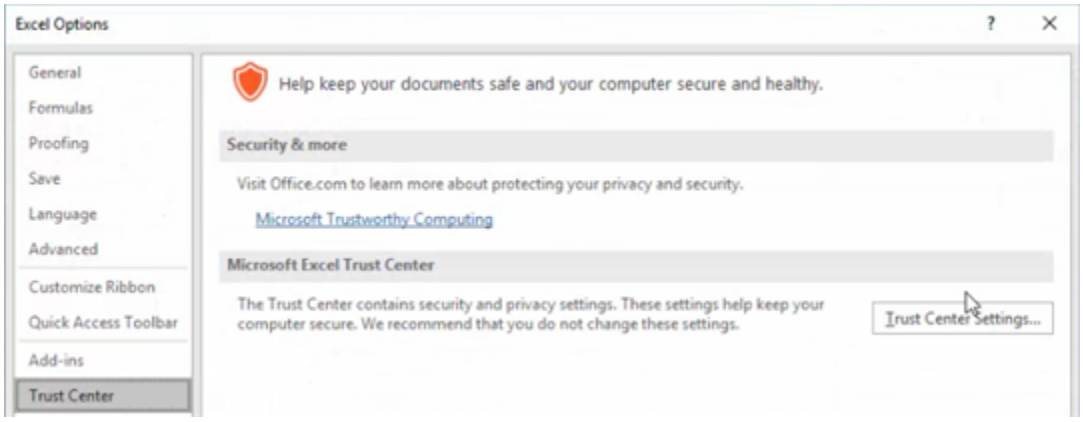
Registering and Installing the Excel Add-In

You must register and install the Excel add-in before you can use it with AVEVA Historian Client Web.

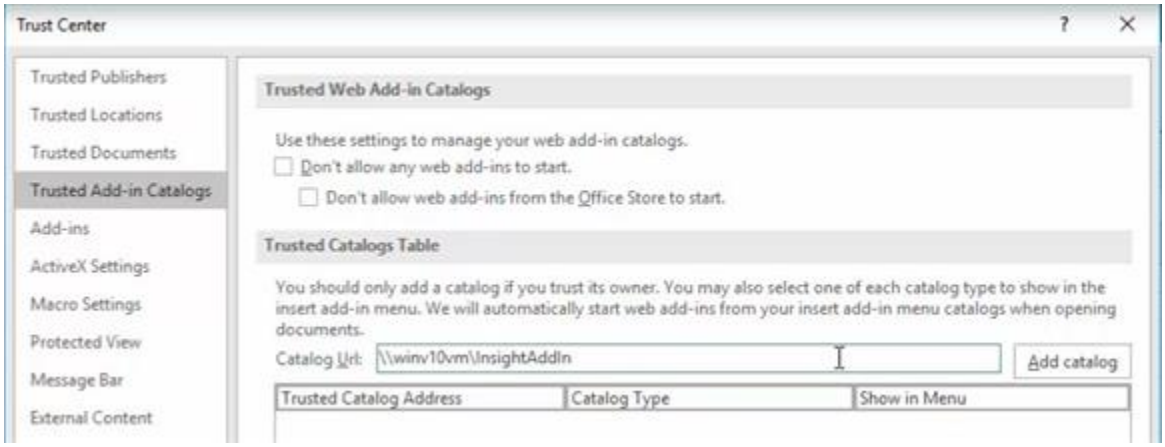
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To register and install the Excel add-in:

1. Open a blank workbook in Excel.
2. Select the **File** menu, then select **Options**. The **Excel Options** dialog displays.



3. Select **Trust Center**, and then click **Trust Center Settings**. The **Trust Center** dialog displays.

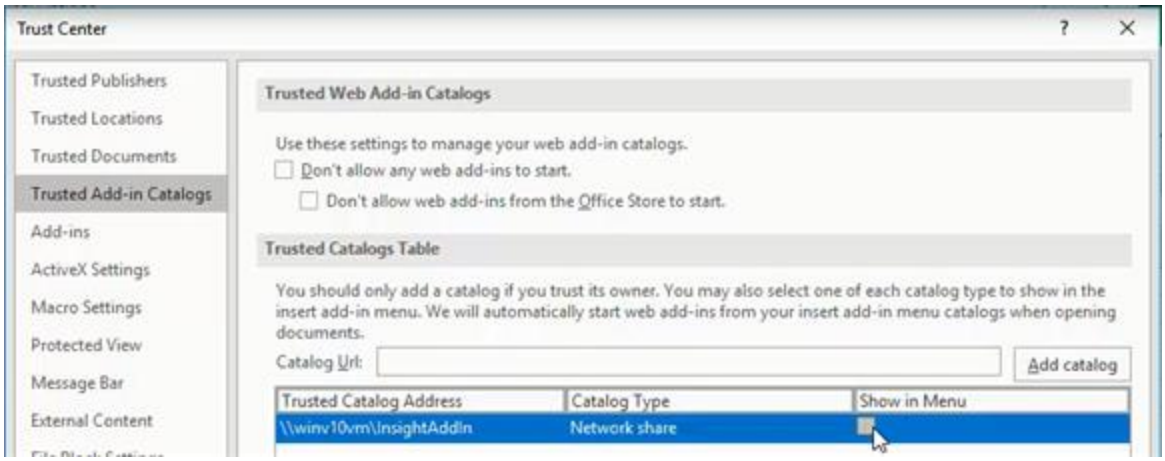


4. Select **Trusted Add-in Catalogs**.
5. In the **Catalog Url** field, enter the UNC path for the shared location created on the Historian server. The UNC path should use this format:

   `\\your_server_name\InsightAddIn`

6. Click **Add catalog**. A new line appears in the table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. Select the new line, then select the **Show in Menu** option.
8. Click **OK**, then restart Excel to apply the changes.

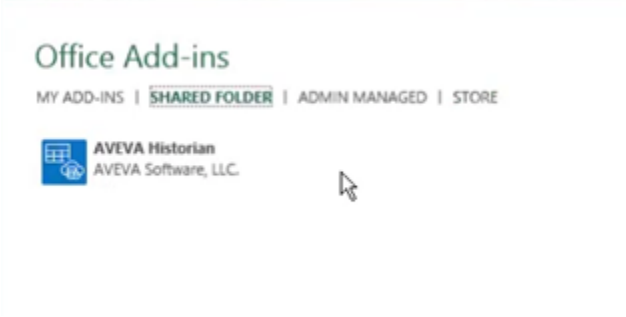
Note: If you experience difficulty launching the Excel Add-In, your system may be missing the required Microsoft Edge WebView2 runtime. To install the runtime, locate and run `MicrosoftEdgeWebView2RuntimeInstallerX64.exe`, which you can find in the following location:

```
\\your_server_name\InsightAddIn
```

Applying the Add-In to a Workbook

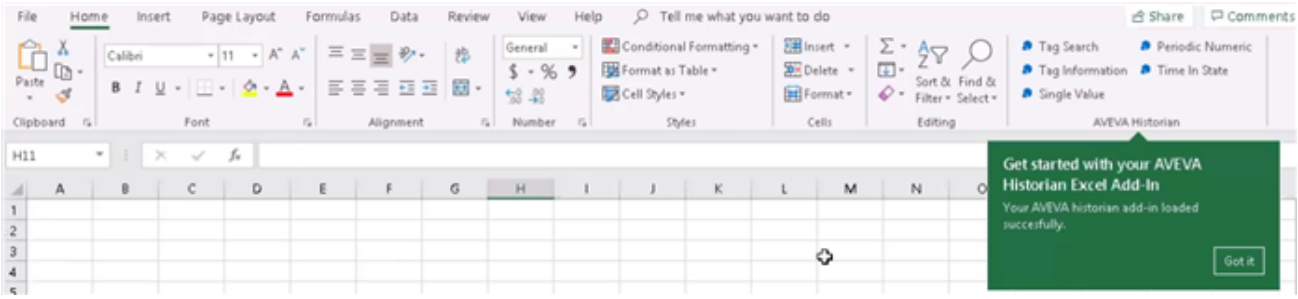
To apply the add-in to a workbook:

1. Open a workbook in Excel.
2. Select the **Insert** menu, then click **My Add-ins**. The Office Add-ins dialog displays.
3. Select **Shared Folder**, then select the AVEVA Historian add-in.

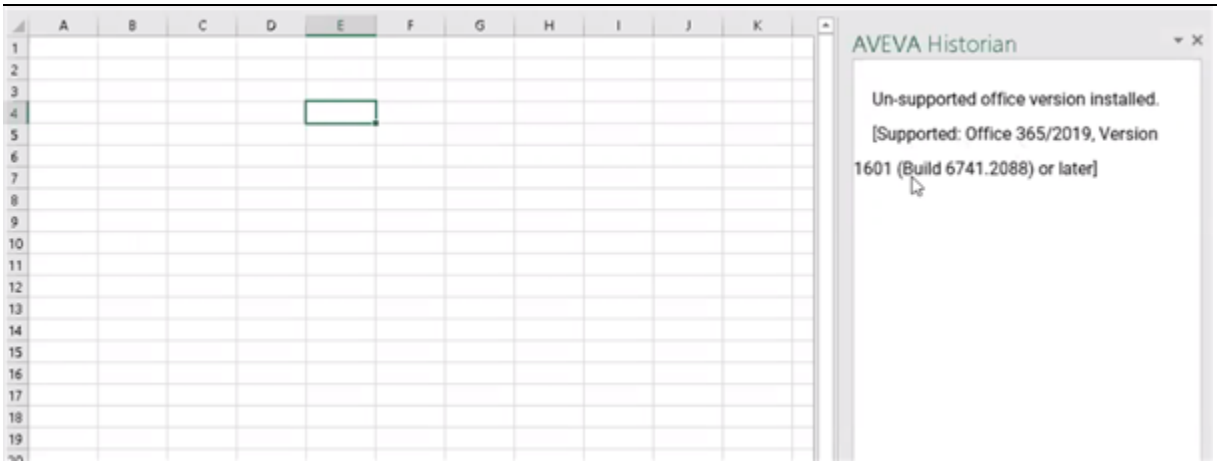


4. Select **OK**.
5. The AVEVA Historian add-in appears in the menu bar.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Notes: If your version of Excel is not supported by the add-in, an error message displays in the side panel.



If you attempt to access a remote node that is configured to require trusted connections, the add-in may fail to initialize, and you may receive a security warning. This occurs when the remote node is configured with a self-signed certificate that is not trusted by your system. Refer to *Using a Self-Signed Certificate* in the *AVEVA Historian Administration Guide* for more information.

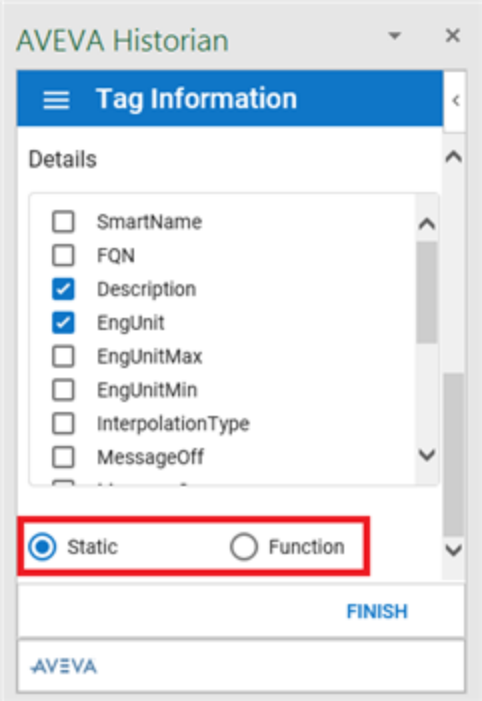
Using the Excel Add-In

The Excel add-in enables you to create reports using data retrieved from AVEVA Historian.

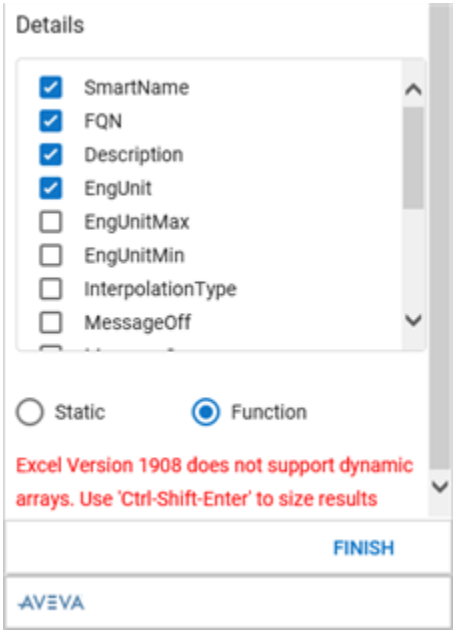
You can create static reports, in which the data is retrieved during report creation and remains fixed. You can also create dynamic reports using custom Excel functions, in which the data is retrieved from the server whenever the Excel workbook is opened, or the function parameters are changed.

While using the add-in, you can choose between these two options by selecting either the **Static** or **Function** option from the add-in dialog:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If the installed version of Excel does not support all the functionality of the add-in, a message displays in the add-in dialog. For example, if your version of Excel does not support dynamic array formulas, a message similar to the following displays:

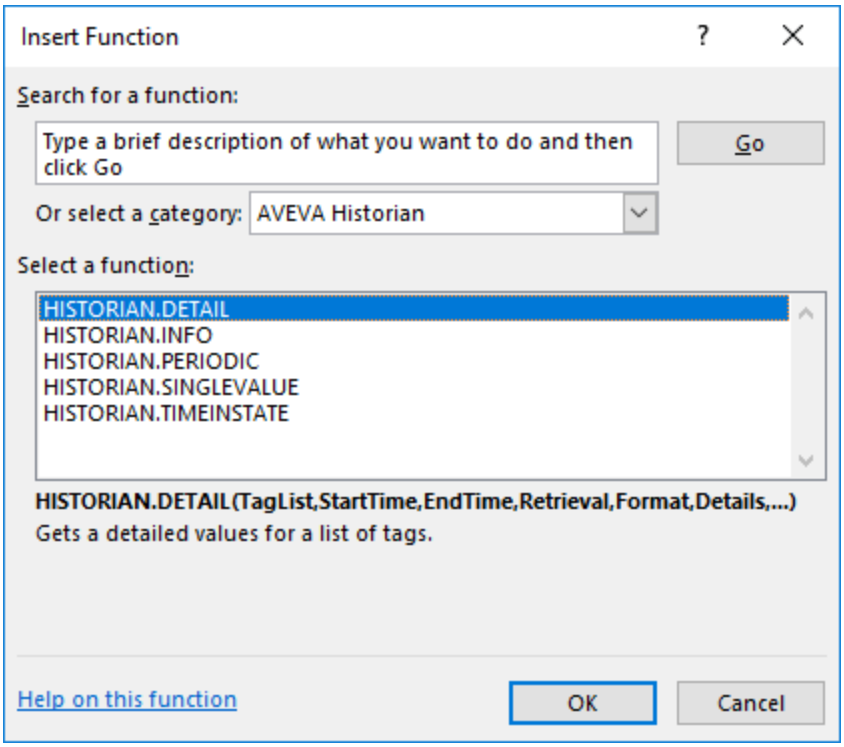


You can use custom functions without dynamic array formula support, but the output must be refreshed manually. To enable dynamic array formula support, which enables custom functions in reports to update automatically, upgrade your Microsoft Office installation to version 2002 or higher.

See Minimum Supported Versions of Microsoft Excel for more information about specific add-in functionality available by Excel version.

When the add-in is enabled, the custom functions are included in the AVEVA Historian category and are available using the HISTORIAN namespace.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



For more information about specific functions, see:

- Using the HISTORIAN.INFO Function
- Using the HISTORIAN.SINGLEVALUE Function
- Using the HISTORIAN.PERIODIC Function
- Using the HISTORIAN.TIMEINSTATE Function
- Using the HISTORIAN.DETAIL Function

About date/time formatting

The default formatting for date/time values depends on which version of the Excel API is supported by your installed version of Excel. For more information about specific features supported by different Excel API versions, see Minimum Supported Versions of Microsoft Excel.

- **Excel API Version 1.12 (or higher)**

  If your version of Excel supports the Excel API version 1.12 or higher, by default date/time values are formatted according to your system's regional settings.

- **Excel API Version 1.11 (or lower)**

  If your version of Excel supports the Excel API version 1.11 or lower, by default date/time values are formatted according to your system's language setting. AVEVA Historian contains a preset list of appropriate default date/time formats for all supported languages.

Customizing the date/time format

After you have retrieved data using the Single Value, Periodic Numeric, or Time In State task panes, you can

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

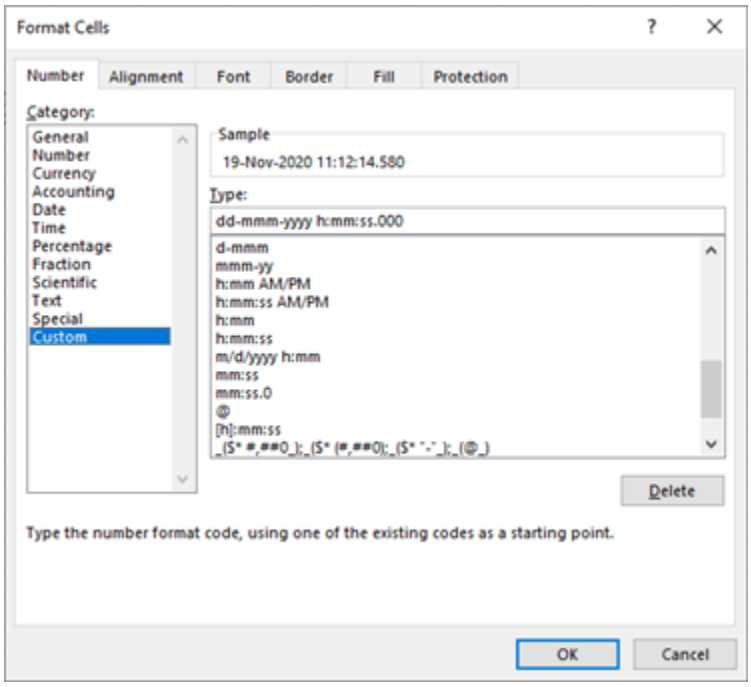
apply your own custom date/time format if the default format does not meet your requirements. Date/time values are formatted based on the following rules:

1. If a custom format is applied to the first value in a row or column of date/time values, the same custom format is applied to the rest of the values in the row or column.
2. If no custom format is applied to the first value, or the General format is applied, all date/time values in the row or column use the default format specified above.

To customize the date/time format:

1. Create a report using the Single Value, Periodic Numeric, or Time In State task panes, including one or more date/time fields in the output.
2. Depending whether the tag direction is down or across, the date/time values will be listed in rows or columns.

   Locate the first cell in a row or column containing a date/time value, and apply formatting using Excel's **Format Cells** dialog.



3. Click **FINISH** on the task pane to refresh your report. The custom date/time format you selected is applied to all date/time values in that row or column.
4. To restore the default date/time format, select the first date/time value in the row or column and apply the General format, then click **FINISH** to refresh the report. The date/time format for that row or column changes back to the default format.

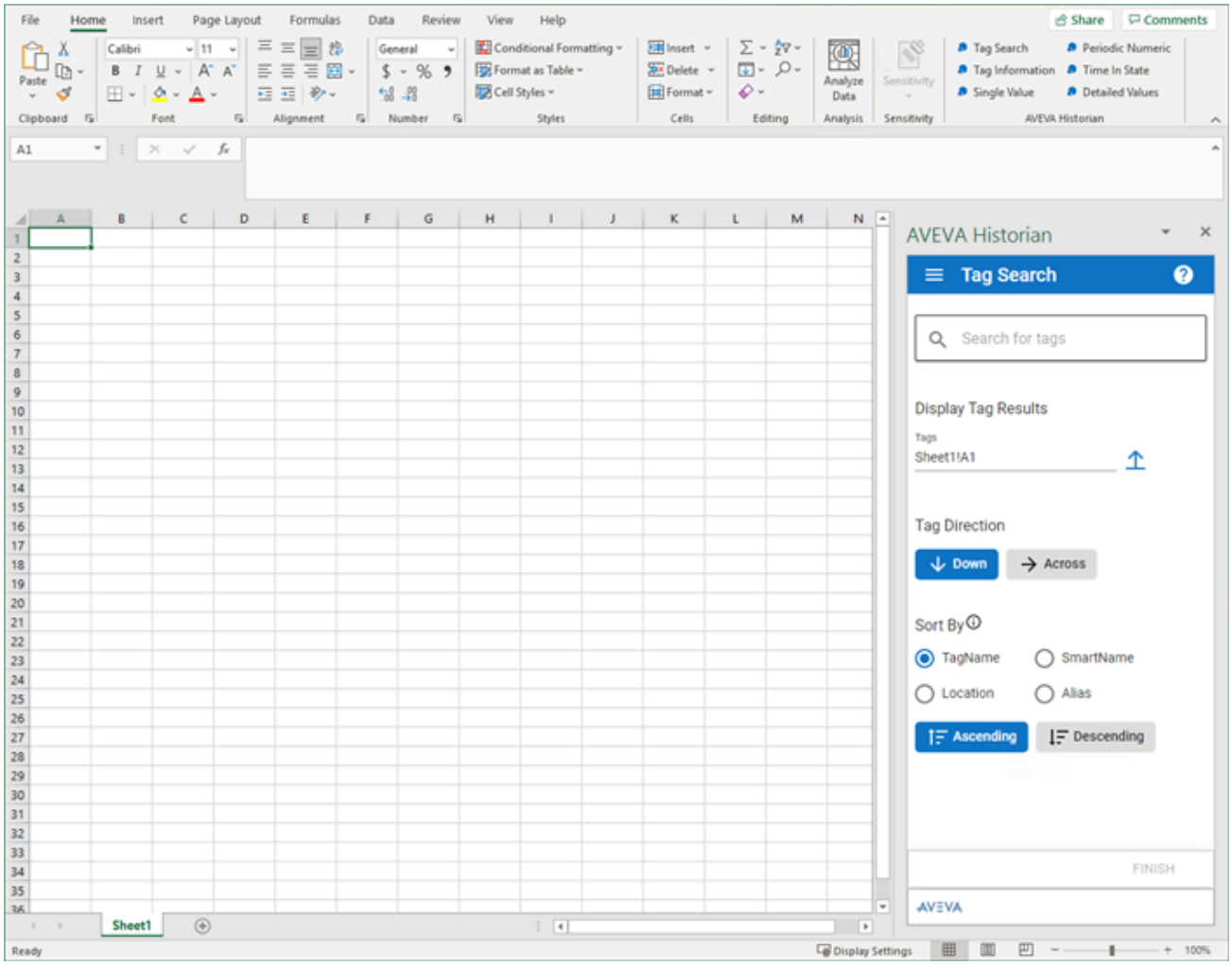
Searching for Tags

You can use the tag search function to customize which tags are included in your results.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

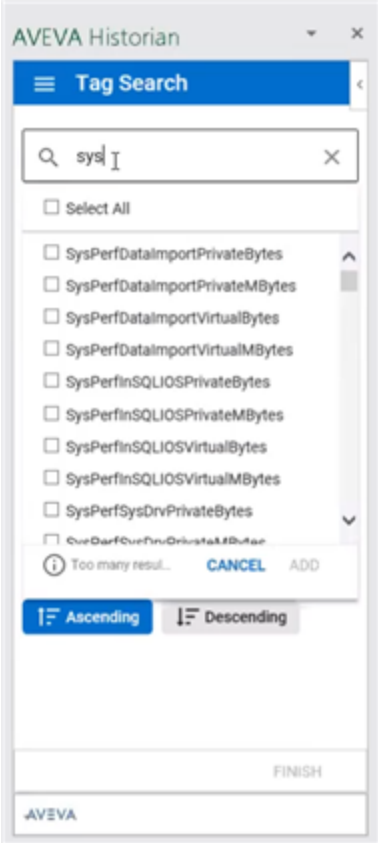
To search for tags:

1. Select **Tag Search** from the AVEVA Historian add-in menu. If the add-in is already displayed, you can also select **Tag Search** from the hamburger menu . The AVEVA Historian add-in panel displays.



2. Enter a search term in the search box. A list of tags filtered by your search term displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Notes: The maximum number of tags displayed in the list is 100. If your search terms return more results than this, a notice is displayed below the list indicating there are too many results to be displayed at once. Refine your search terms to match fewer tags.

While only the tag names are displayed in the list, the search terms apply to the *TagName*, *Description*, *Engineering Unit*, *Alias*, and *Location* columns.

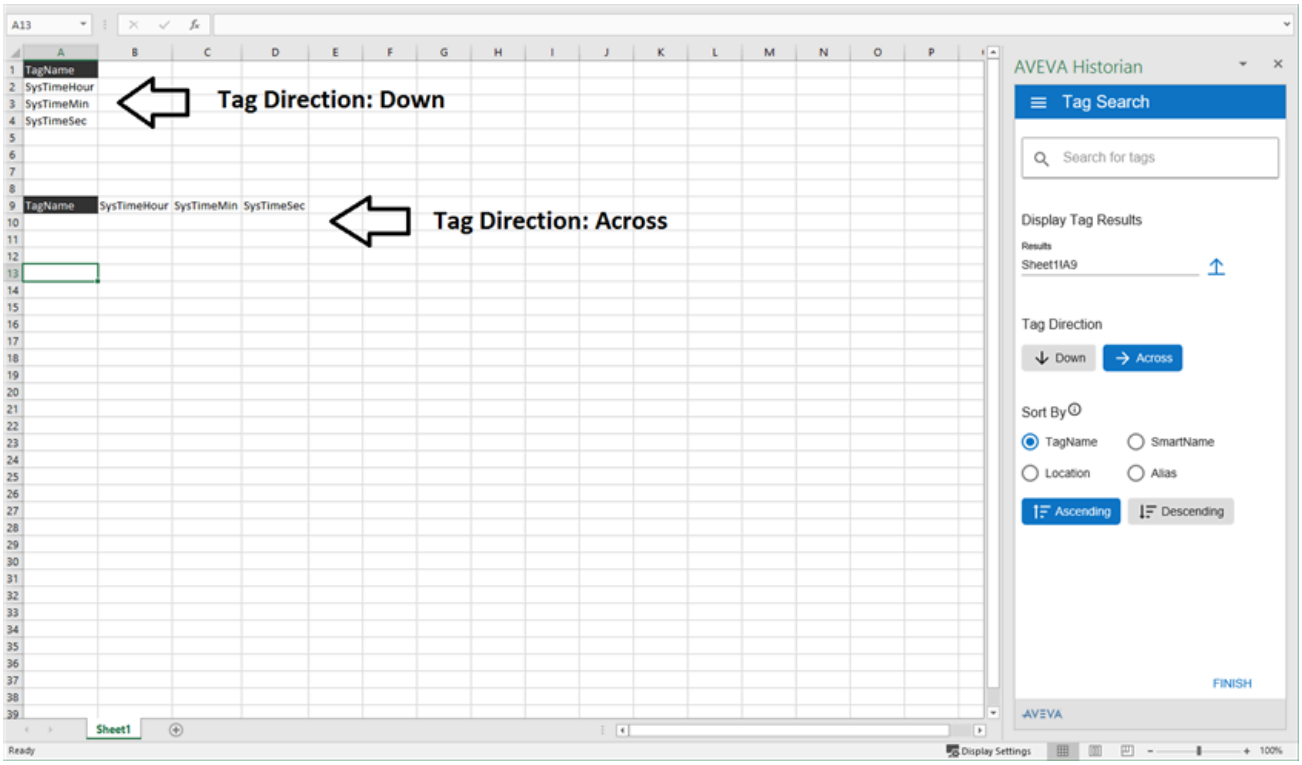
3. Select the individual tags you want to include in your report, or **Select All**, then click **ADD**.
4. To add the selected tags to your spreadsheet, select the cell in your spreadsheet where you want the tag names inserted, and then click . You can also manually enter the cell range into the **Tags** field.

Note: The results include one or more headings, so remember to allow extra space for headings when selecting your output range

The **Display Tag Results** field updates to indicate the selected cell.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. Select **Down** to list the selected tag names in a column, or **Across** to list them in a row.
6. Choose which tag property to **Sort By**, and select either **Ascending** or **Descending** order.
7. Click **Finish**. Your spreadsheet is updated with the list of selected tags.



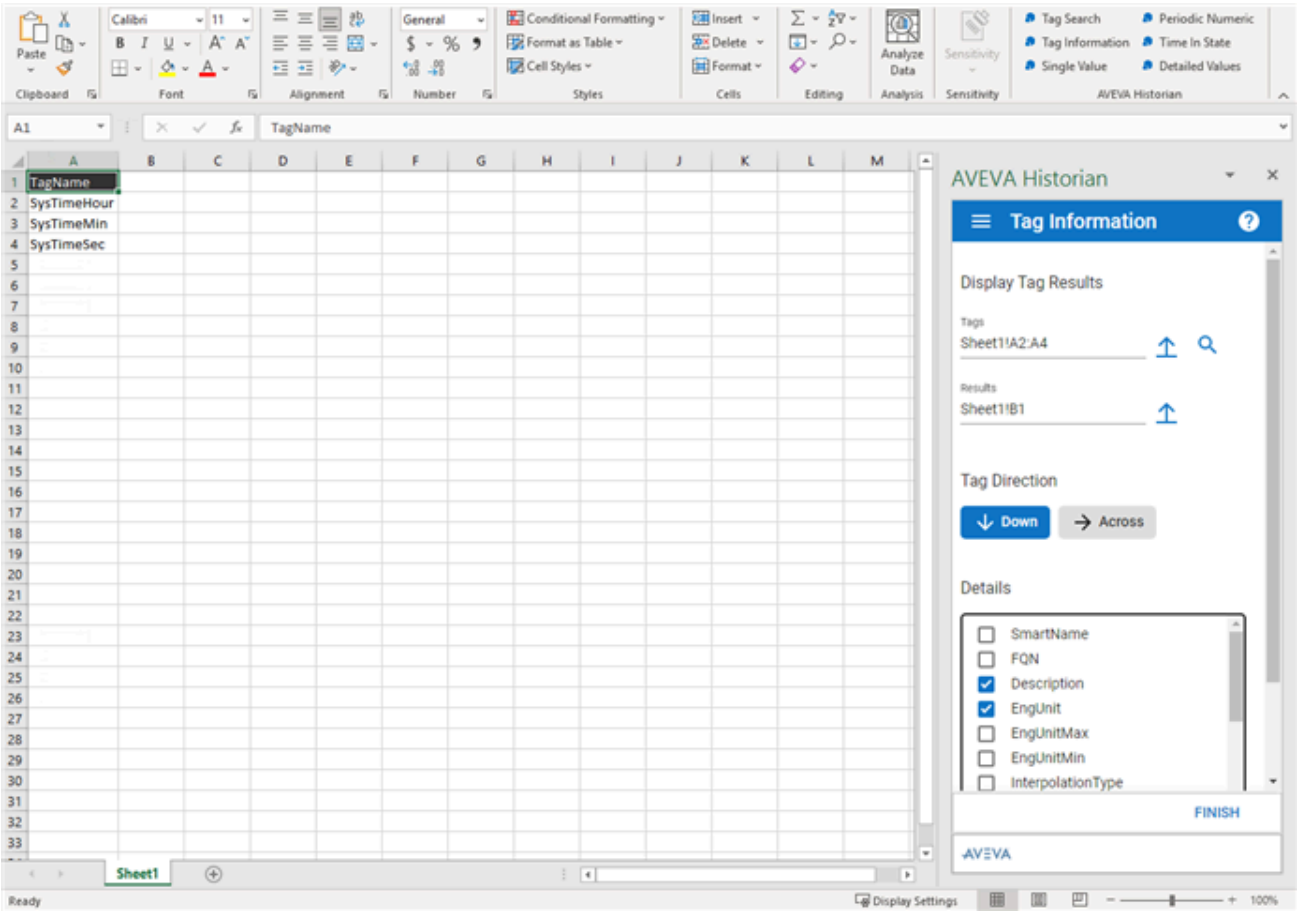
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieving Tag Information

Once you have a list of tags, you can retrieve information about those tags to use for the labels and headings in your report.

To retrieve tag information:

1. Select **Tag Information** from the AVEVA Historian add-in menu. If the add-in is already displayed, you can also select **Tag Information** from the hamburger menu . The AVEVA Historian add-in panel displays.



2. To choose the list of tags for which to retrieve information, select a range of cells containing a list of tag names, then click beside the **Tags** field.

   Note: If you used the **Tag Search** page to select your tags, the range is automatically pre-selected for you.

3. Select the cell where you want to insert the results, then click beside the **Results** field.

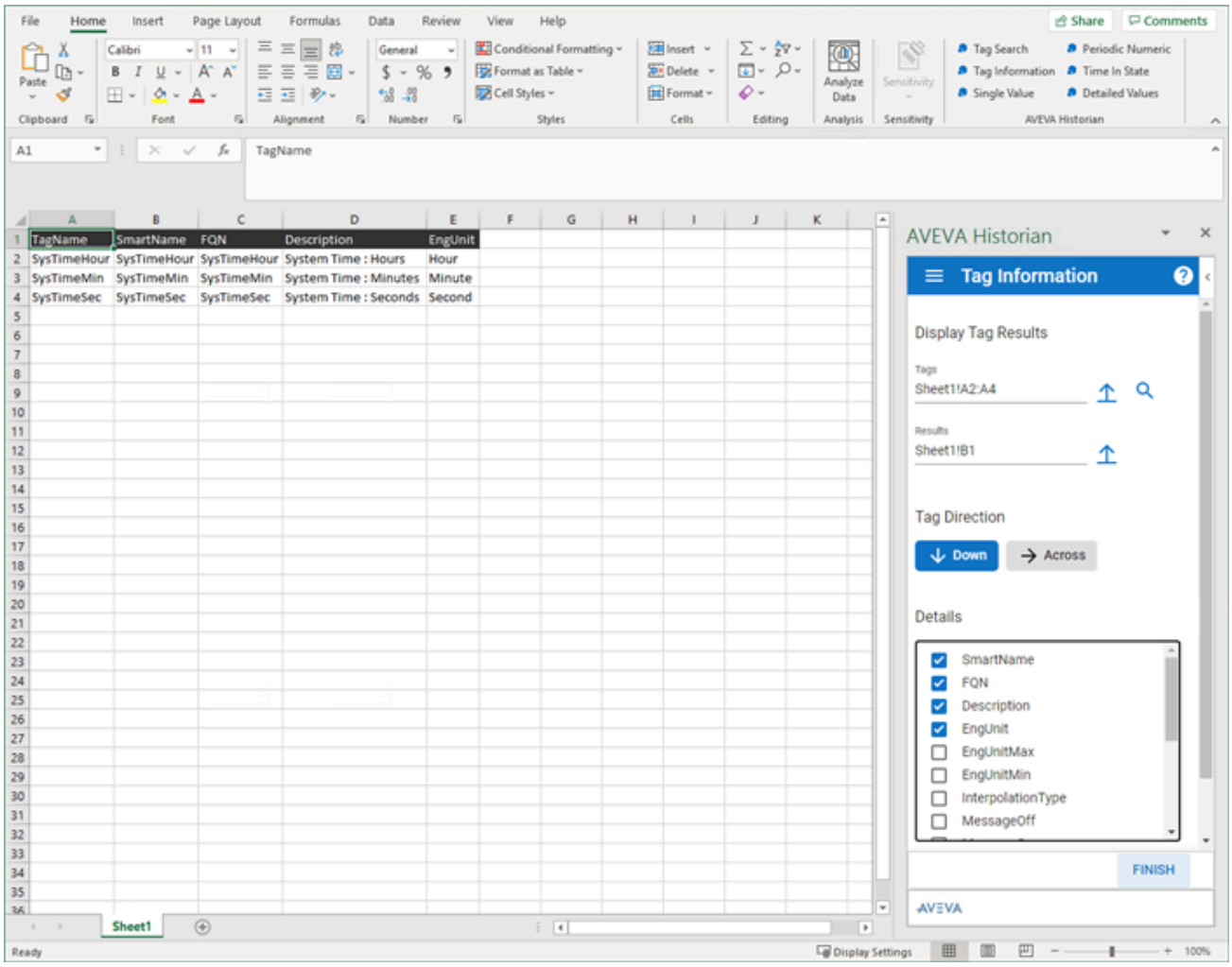
   Note: If you have already inserted data from another add-in page, the **Results** starting location defaults to the next cell adjacent to your existing data, using the same **Tag Direction**.

4. Select **Down** to list the results in rows by tag name, or **Across** to list the results in columns by tag name.
5. Select the tag properties for which you want to retrieve data from the **Details** list.

   Note: For an explanation of the SmartName property, see Using Smart Names for Tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

6. Select **Static** or **Function** to decide how to retrieve the requested data. If you select **Static**, the data will be retrieved and inserted directly into the report. If you select **Function**, a formula will be inserted instead, using the *HISTORIAN.INFO* function. If you edit the formula manually, the retrieved data will be refreshed immediately based on your changes. See Using the HISTORIAN.INFO Function for more information.
7. Click **FINISH**. The selected tag information is retrieved and copied to your spreadsheet.



Using the HISTORIAN.INFO Function

Using the HISTORIAN.INFO Function

The HISTORIAN.INFO function retrieves tag information for a specified list of tags.

Syntax

HISTORIAN.INFO(*TagList*, *Details*, *Down*)

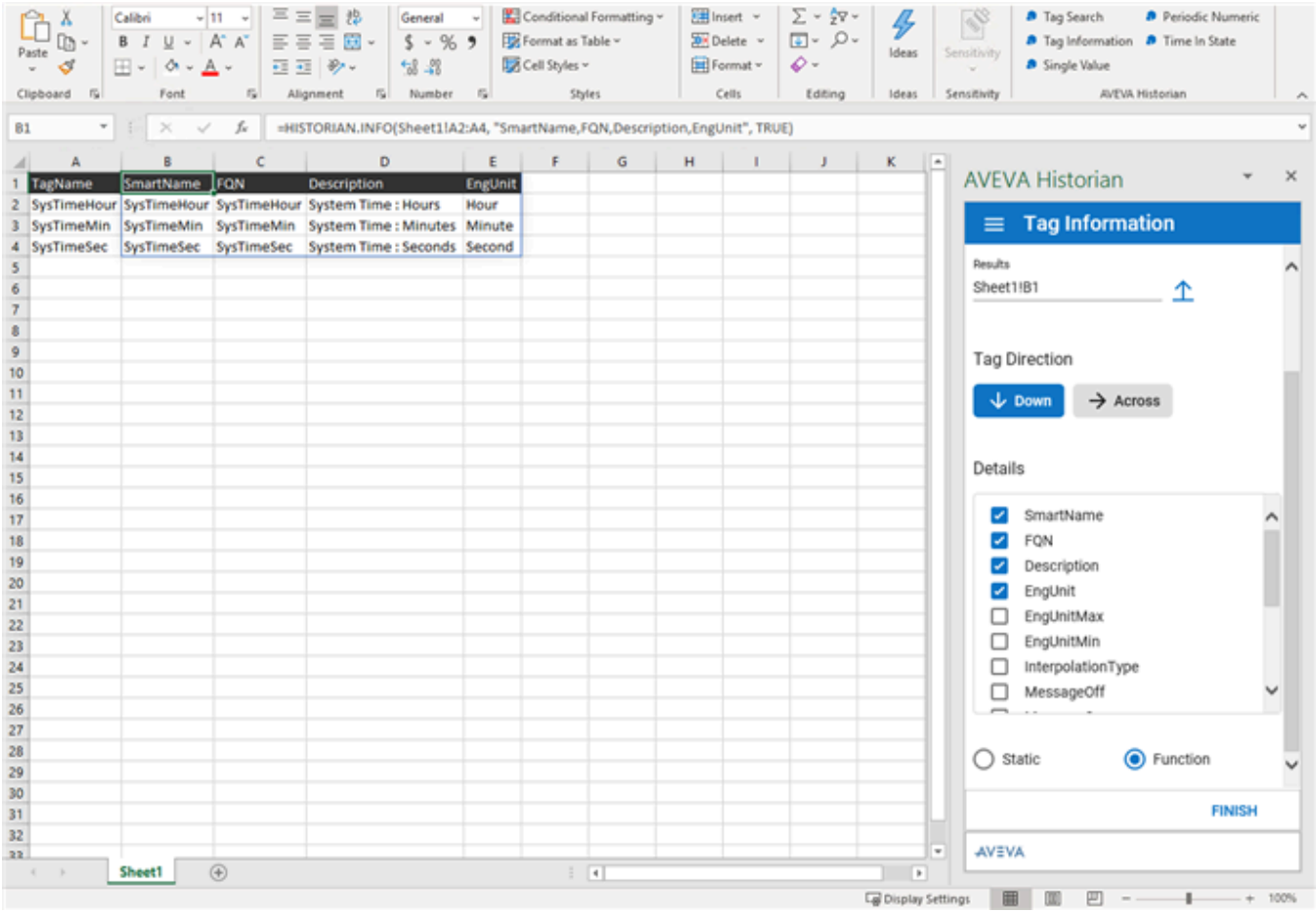
- **TagList** - *Required*. The range of cells containing the list of tag names you want to retrieve information about.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Details** - *Required*. A comma-separated list of properties to retrieve for the selected tags. The following properties are supported: `SmartName`, `FQN`, `Description`, `EngUnit`, `EngUnitMax`, `EngUnitMin`, `InterpolationType`, `MessageOff`, `MessageOn`, `SourceTag`, `TagName`, `HierarchicalName`, `Alias`, `Location`, and `Namespace`.
- **Down** - *Optional*. If `TRUE`, the tags are listed in rows. If `FALSE`, the tags are listed in columns. If no value is specified, the default value of `TRUE` is used.

Example

Recreating the simple report from the previous example, we select the **Function** option instead of the **Static** option. This results in a formula being inserted into the spreadsheet, instead of static data. There is a brief pause while communicating with the server, then the retrieved data is displayed.



The formula generated using this example is:

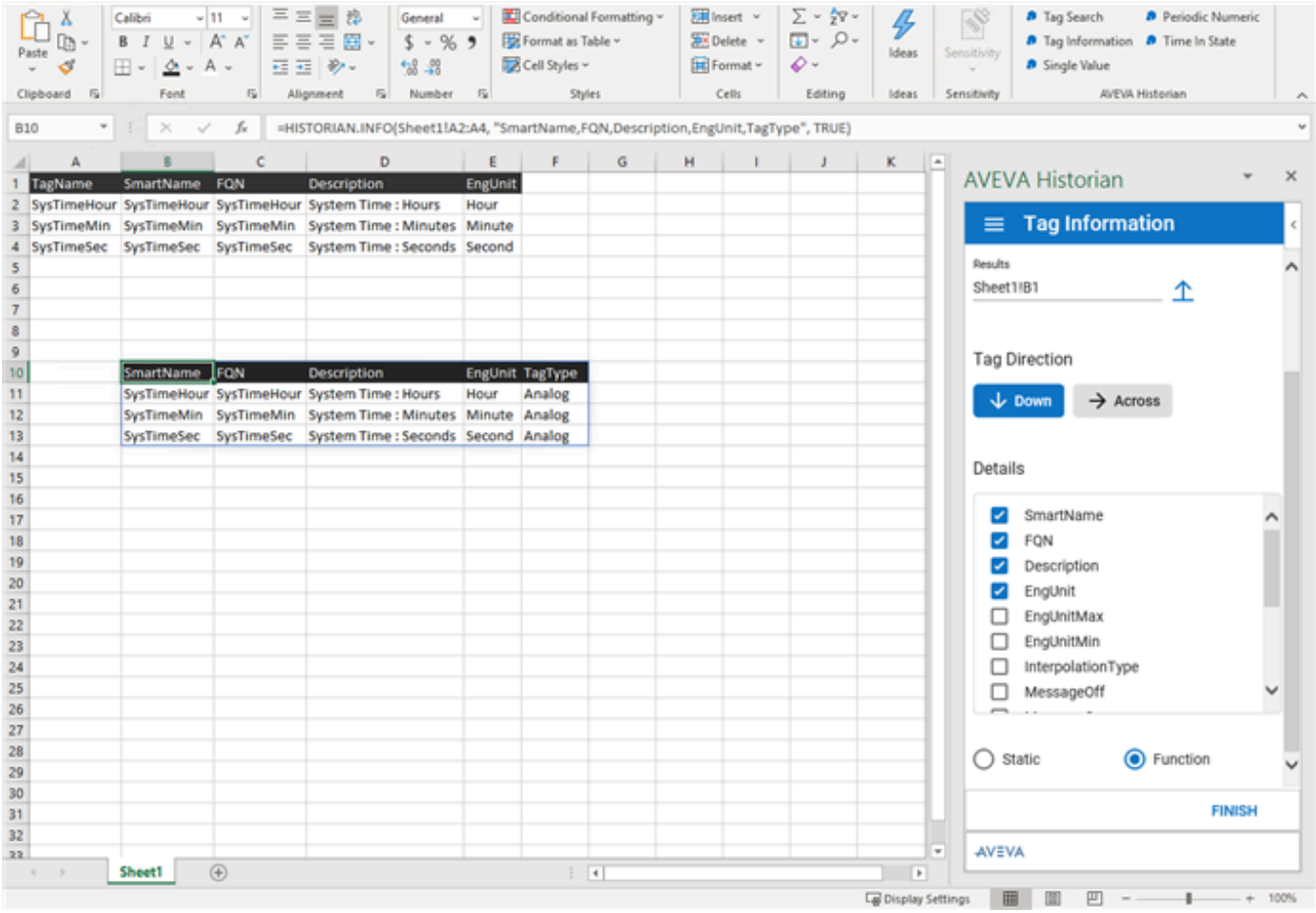
```
=HISTORIAN.INFO(Sheet1!A2:A4, "SmartName,FQN,Description,EngUnit", TRUE)
```

You can modify the formula, and the results refresh automatically to reflect your changes. You can also use the function directly, without using the add-in panel. For example, the following formula is entered into cell B10:

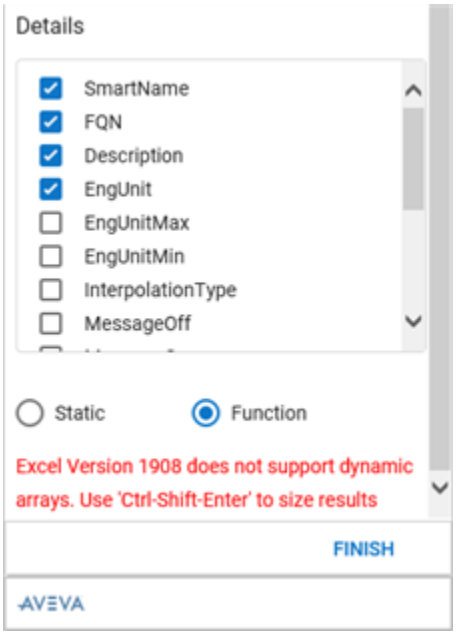
```
=HISTORIAN.INFO(Sheet1!A2:A4, "SmartName,FQN,Description,EngUnit,TagType", TRUE)
```

The resulting report is identical to the first, but with the additional *TagType* property added.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If your version of Excel does not support dynamic arrays, a message similar to the following displays:



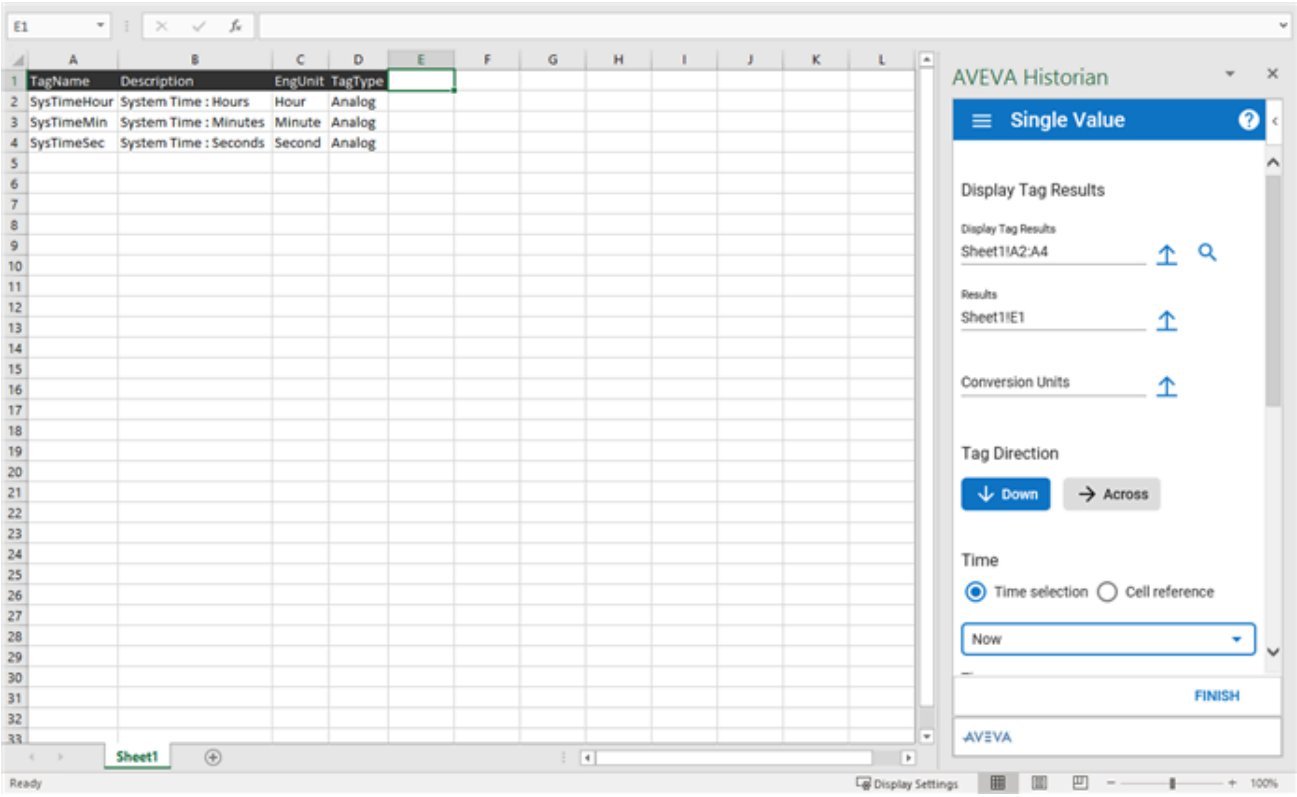
In this case, although you can still use the custom function, the size of the output range cannot be refreshed automatically. You must manually size the range by selecting it, and then press **Ctrl-Shift-Enter**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieving Single Values

To retrieve single values:

1. Select **Single Value** from the AVEVA Historian add-in menu. If the add-in is already displayed, you can also select **Single Value** from the hamburger menu . The AVEVA Historian add-in panel displays.



2. To choose the list of tags for which to retrieve information, select a range of cells containing a list of tag names, then click beside the **Tags** field.

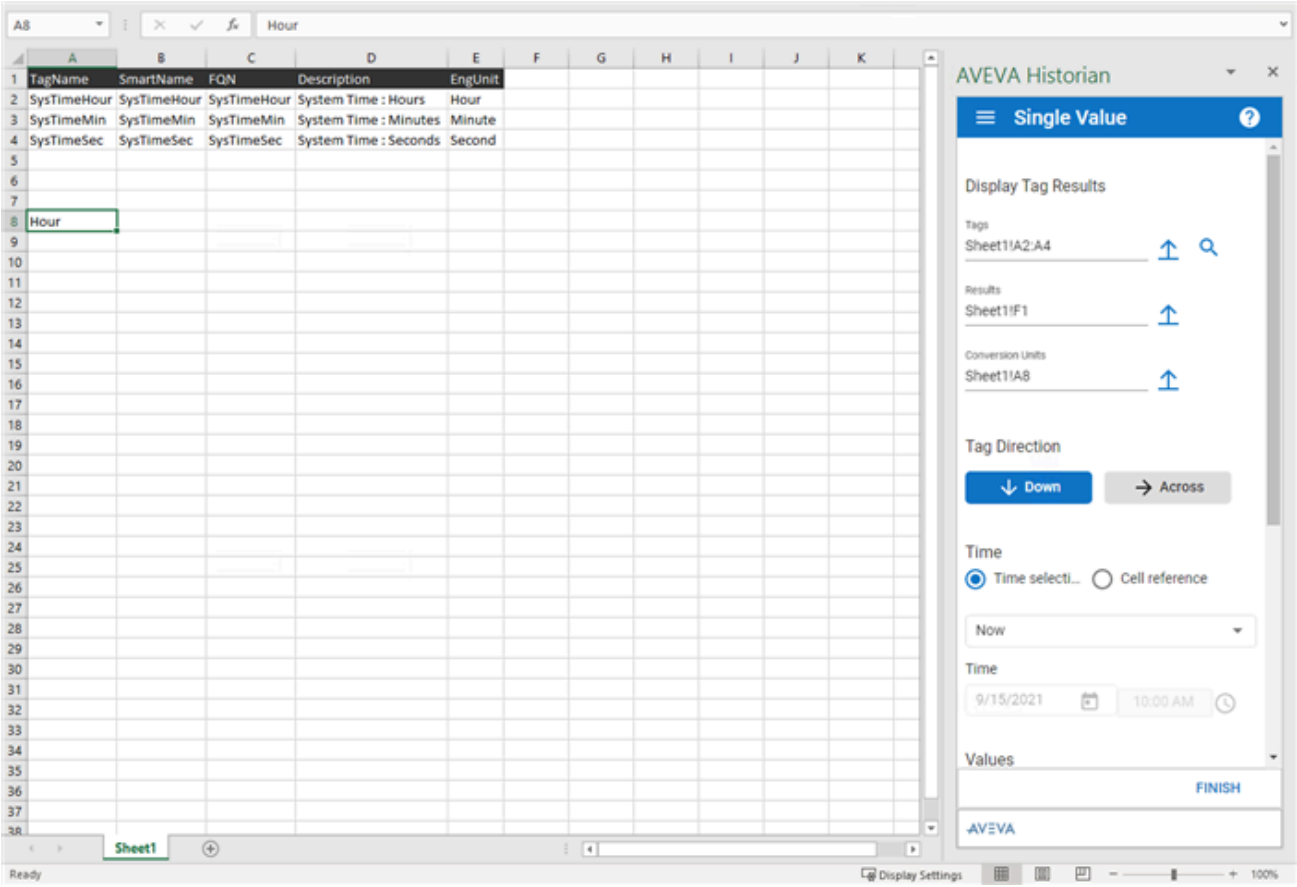
   Note: If you used the **Tag Search** page to select your tags, the range is automatically pre-selected for you.

3. Select the cell where you want to insert the results, then click beside the **Results** field. Normally, you will select a range adjacent to the tag list or tag information results to build out your report.

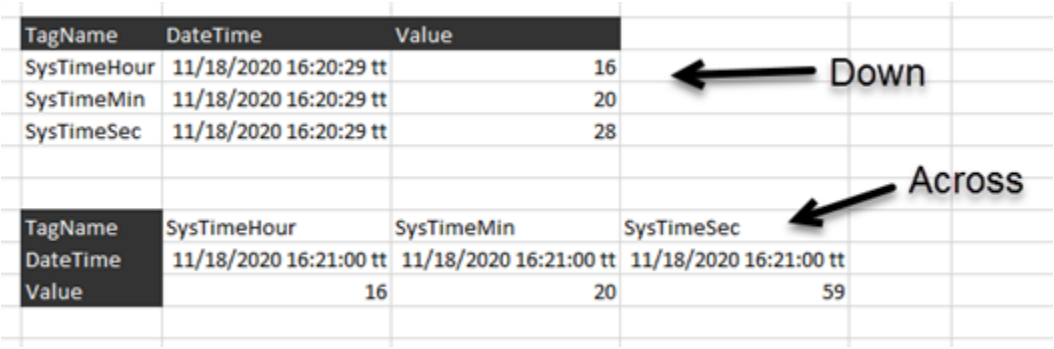
   Note: If you have already inserted data from another add-in page, the **Results** starting location defaults to the next cell adjacent to your existing data, using the same **Tag Direction**.

4. The **Conversion Units** field allows you to compare values more easily by converting the retrieved data to display in the same engineering unit. Select a cell or range containing the names of target engineering units, then click beside the Conversion Units field. When tag data is retrieved, it displays in the selected unit if an appropriate conversion is available in the Engineering Unit Catalog. Otherwise, it displays in the default engineering unit for the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select **Down** to list the results in rows by tag name, or **Across** to list the results in columns by tag name. The following example shows the results of retrieving a value for 3 tags in each format.



TagName	DateTime	Value
SysTimeHour	11/18/2020 16:20:29 tt	16
SysTimeMin	11/18/2020 16:20:29 tt	20
SysTimeSec	11/18/2020 16:20:29 tt	28

Down

Across

TagName	SysTimeHour	SysTimeMin	SysTimeSec
DateTime	11/18/2020 16:21:00 tt	11/18/2020 16:21:00 tt	11/18/2020 16:21:00 tt
Value	16	20	59

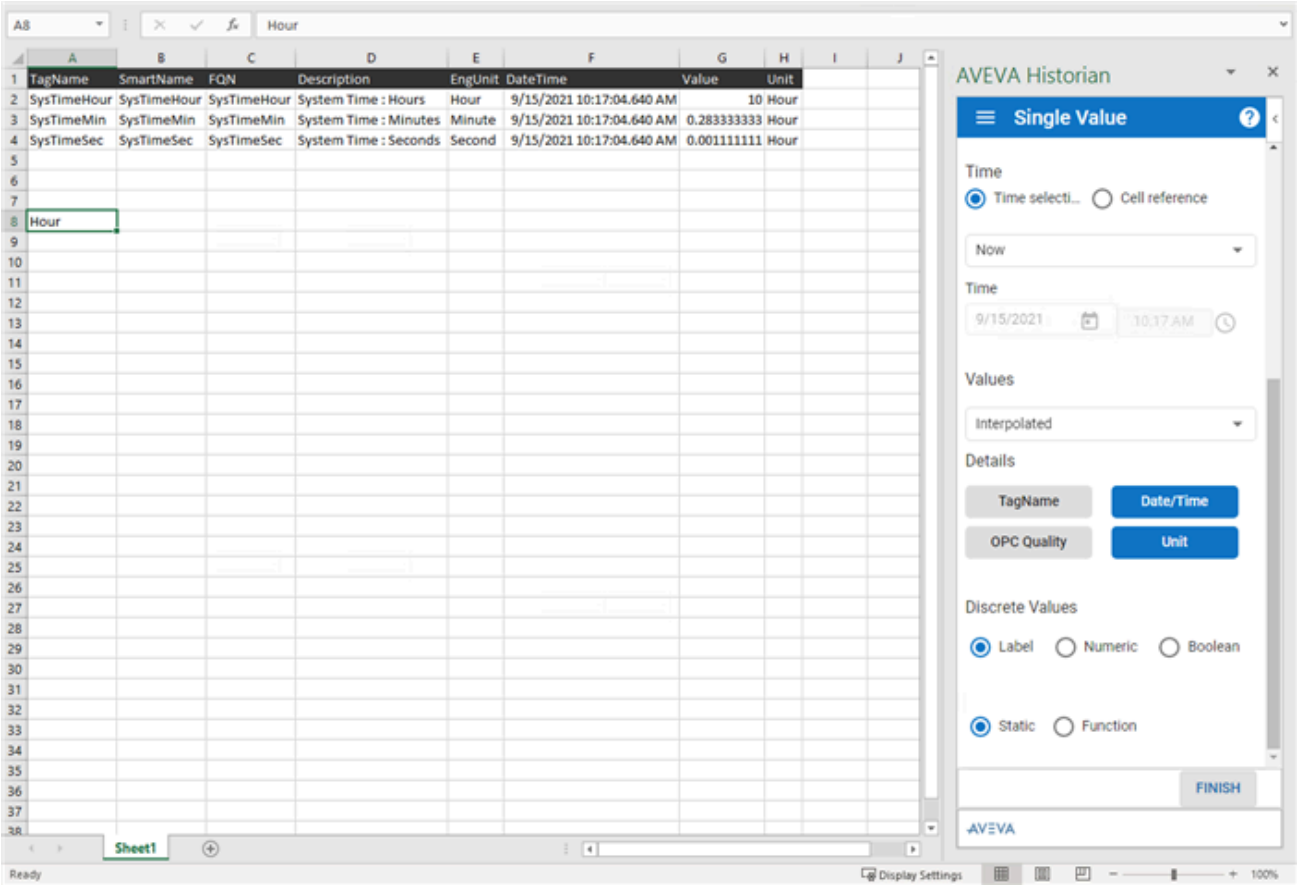
6. You can choose the **Time** for which to retrieve the tag values in two ways:
   a. Select **Time selection**, and choose a time value from the drop-down list.
   b. Select **Cell reference**, and choose a cell in your spreadsheet containing a datetime value.
7. Select which **Values** to retrieve from the following options:
   a. **Interpolated** - If there is no data point stored at the exact time requested, the add-in retrieves interpolated values. The interpolation is dependent on the server-side tag configuration for interpolation type.
   b. **Previous** - The add-in retrieves the first value on or before the requested date/time, and uses the requested date/time as the time stamp.
   c. **Next with time** - The add-in retrieves the first value after the requested date/time, and uses the stored

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

time stamp.

d. **Previous with time** - The add-in retrieves the first value on or before the requested date/time, and uses the stored time stamp.

8. Select which **Details** to retrieve along with the values for the selected tags. You can choose to include any combination of **TagName**, **Date/Time**, **OPC Quality**, and **Unit**.
9. Select whether to display **Discrete Values** in **Label** format, **Numeric** format, or **Boolean** format.
10. Select **Static** or **Function** to decide how to retrieve the requested data. If you select **Static**, the data is retrieved and inserted directly into the report. If you select **Function**, a formula is inserted instead, using the *HISTORIAN.SINGLEVALUE* function. If you edit the formula manually, the retrieved data is refreshed immediately based on your changes. See Using the HISTORIAN.SINGLEVALUE Function for more information.
11. Click **FINISH**. The selected tag information is retrieved and copied to your spreadsheet.



Using the HISTORIAN.SINGLEVALUE Function

Using the HISTORIAN.SINGLEVALUE Function

The HISTORIAN.SINGLEVALUE function retrieves single values for a specified list of tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

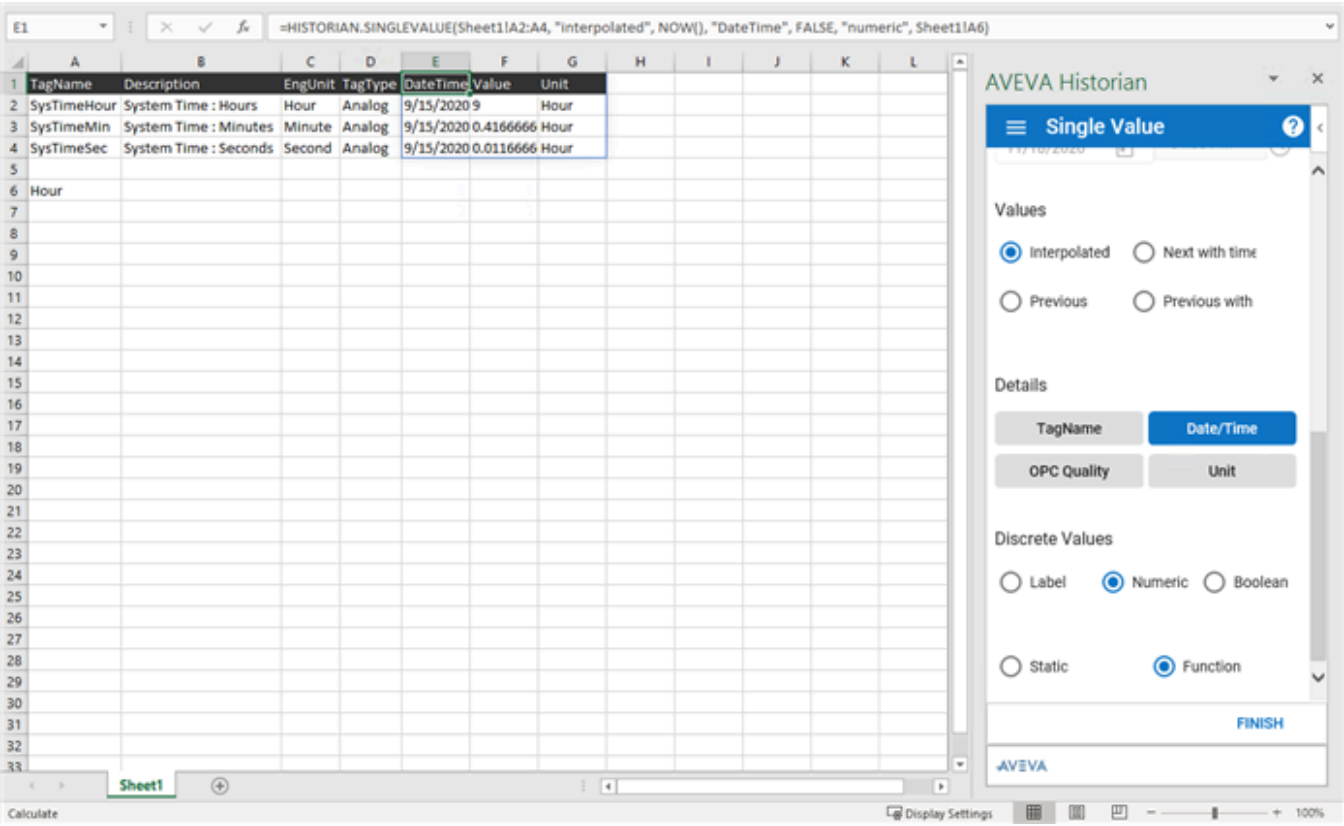
Syntax

HISTORIAN.SINGLEVALUE(*TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress*)

- **TagList** - *Required*. The range of cells containing the list of tag names you want to retrieve information about.
- **ValueType** - *Required*. The type of values to retrieve. The following types are supported: `Interpolated`, `EndBound` (next with time), `StartBound` (previous), or `previouswithtime`.
- **Time** - *Required*. The date and time for which to retrieve the tag values, expressed as a serial date number.
- **Details** - *Required*. A comma-separated list of additional properties to retrieve for the selected tags. The following properties are supported: `TagName`, `DateTime`, and `OPCQuality`.
- **Down** - *Optional*. If TRUE, the tags are listed in rows. If FALSE, the tags are listed in columns. If no value is specified, the default value of TRUE (rows) is used.
- **DiscreteFormat** - *Optional*. The format in which to display discrete values. The following formats are supported: `label`, `numeric`, or `boolean`. If no value is specified, the default value of `label` is used.
- **UOMAddress** - *Optional*. A cell, or range of cells, containing the names of engineering units. When tag data is retrieved, it is converted to display in this unit if an appropriate conversion is defined in the Engineering Units Catalog. Otherwise, it displays in the tag's default engineering unit.

Example

Recreating the simple report from the previous example, we select the **Function** option instead of the **Static** option. This results in a formula being inserted into the spreadsheet, instead of static data. There is a brief pause while communicating with the server, then the retrieved data is displayed.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

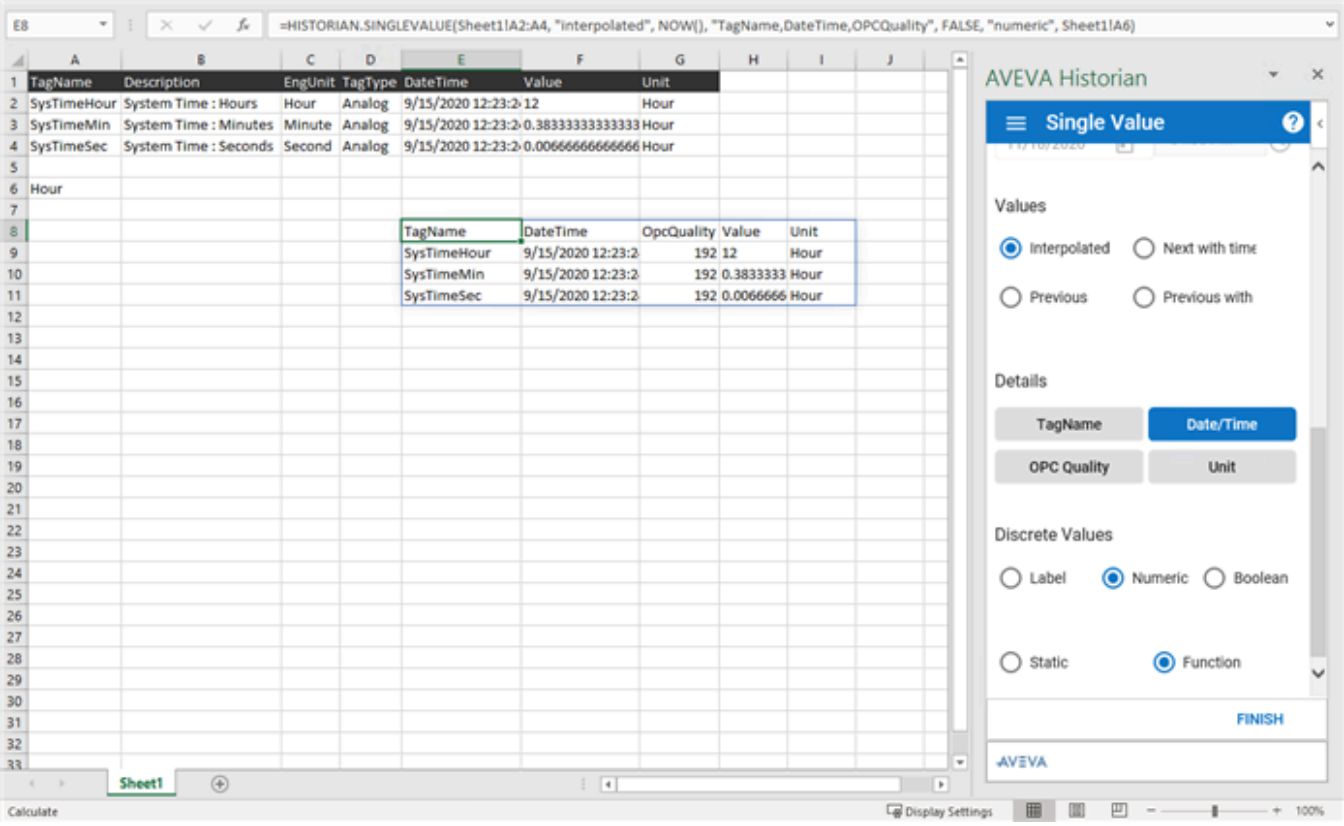
The formula generated using this example is:

```
=HISTORIAN.SINGLEVALUE(Sheet1!A2:A4, "interpolated", NOW(), "DateTime", FALSE, "numeric",
Sheet1!A6)
```

You can modify the formula, and the results refresh automatically to reflect your changes. You can also use the function directly, without using the add-in panel. For example, the following formula is entered into cell E8:

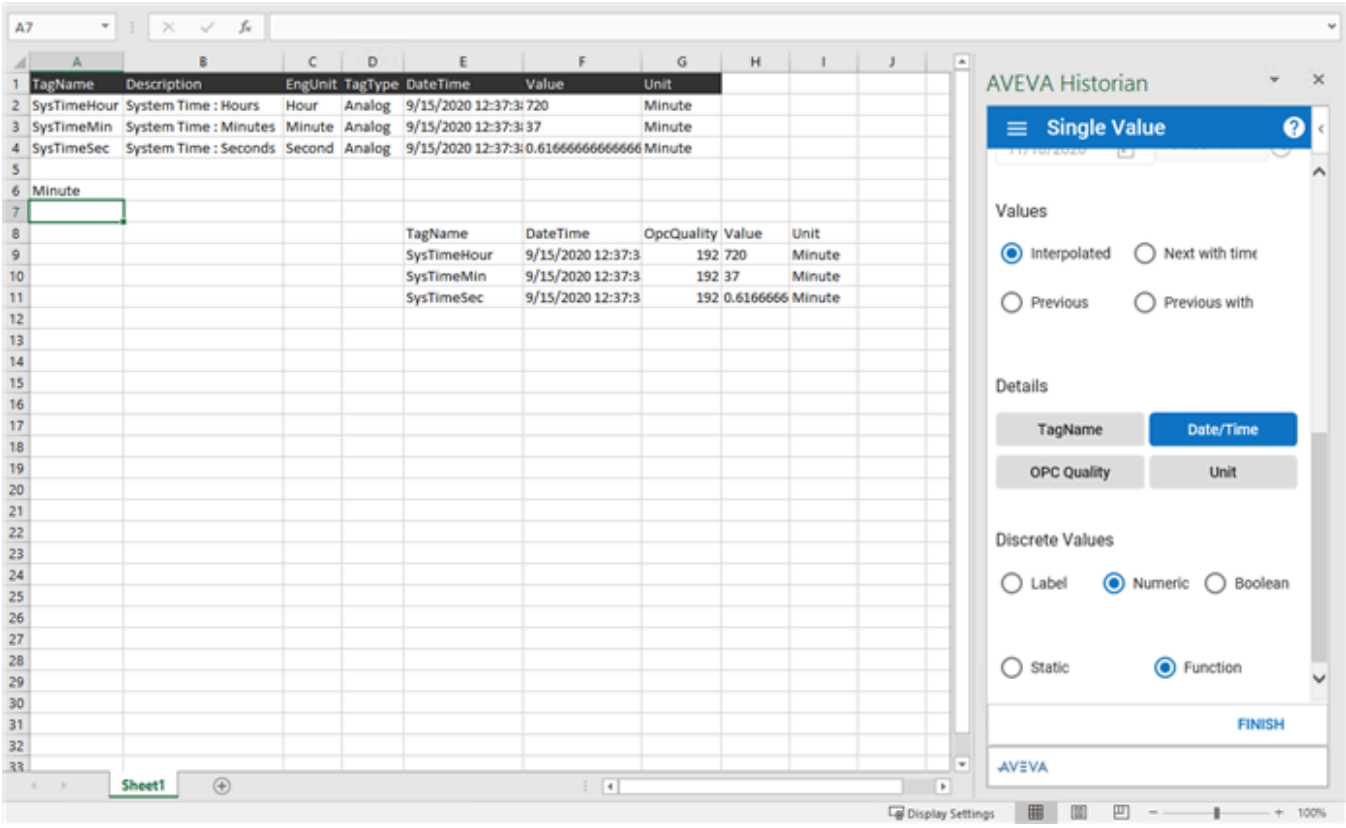
```
=HISTORIAN.SINGLEVALUE(Sheet1!A2:A4, "interpolated", NOW(), "TagName,DateTime,OPCQuality",
FALSE, "numeric", Sheet1!A6)
```

The resulting report is identical to the first, but with the additional *TagName* and *OPCQuality* properties added.

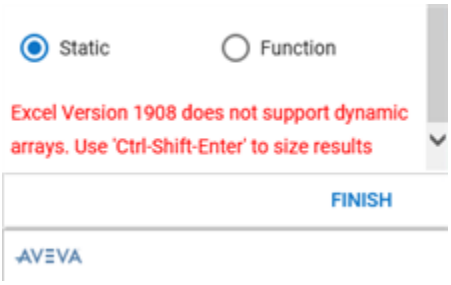


Changing the value of a cell referenced by a custom function refreshes all the custom functions in the worksheet, re-retrieving the data from the database. For example, changing the unit in cell A6 from *Hour* to *Minute* causes the two custom functions in the worksheet to be refreshed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If your version of Excel does not support dynamic arrays, a message similar to the following displays:



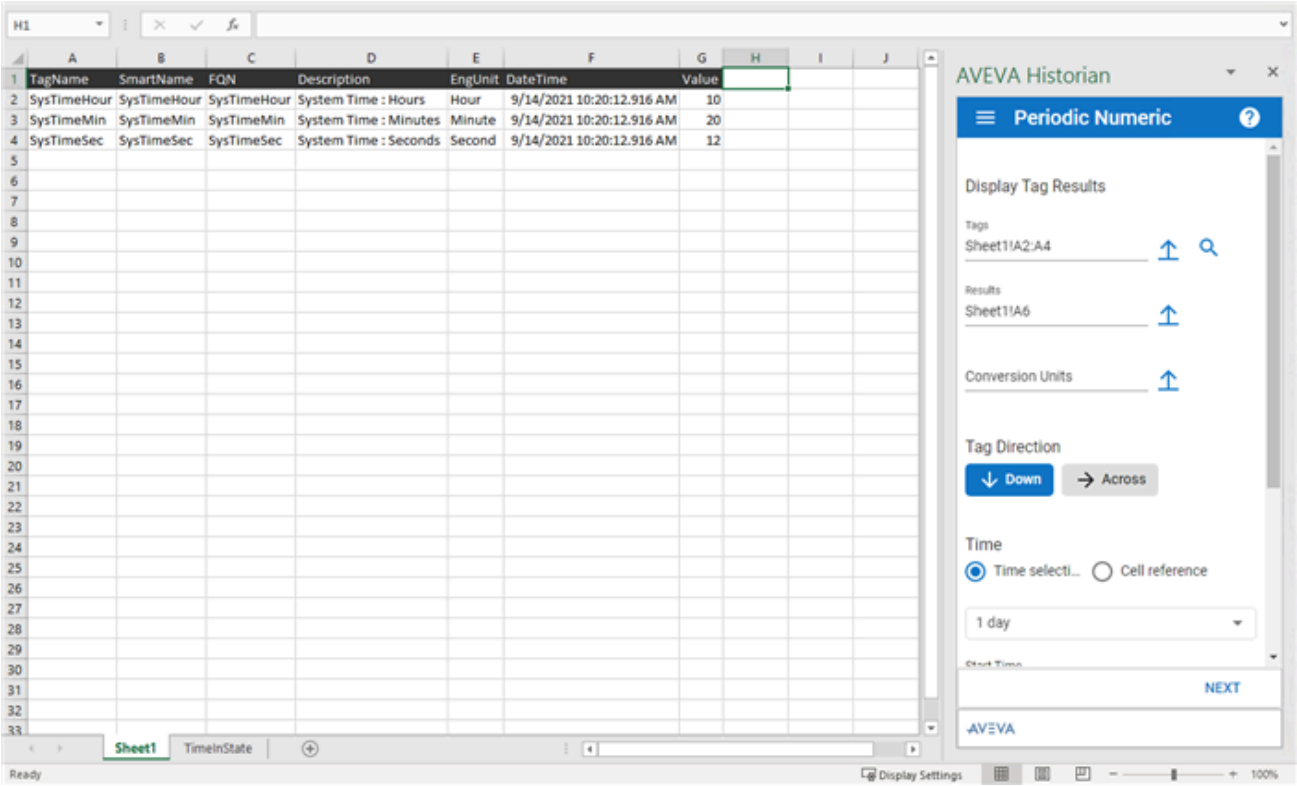
In this case, although you can still use the custom function, the size of the output range cannot be refreshed automatically. You must manually size the range by selecting it, and then press **Ctrl-Shift-Enter**.

Retrieving Periodic Numeric Values

To retrieve periodic numeric values:

1. Select **Periodic Numeric** from the AVEVA Historian add-in menu. If the add-in is already displayed, you can also select **Periodic Numeric** from the hamburger menu ☰. The AVEVA Historian add-in panel displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Note: Datetime results can be displayed up to millisecond resolution by applying custom cell formatting, such as `yy-mm-dd hh:mm:ss.000`.

2. To choose the list of tags for which to retrieve information, select a range of cells containing a list of tag names, then click beside the **Tags** field. You can also manually enter the range into the **Tags** field.

   Note: If you used the **Tag Search** page to select your tags, the range is automatically pre-selected for you.

3. Select the cell where you want the results inserted, then click beside the **Results** field. You can also manually enter the cell address in the **Results** field.

   Note: If you have already inserted data from another add-in page, the **Results** starting location defaults to a cell beneath your existing data, using the same **Tag Direction**.

4. The **Conversion Units** field allows you to compare values more easily by converting the retrieved data to display in the same engineering unit. Select a cell (or range) containing the names of target engineering units, then click beside the Conversion Units field. When tag data is retrieved, it displays in the selected unit if an appropriate conversion is available in the Engineering Unit Catalog. Otherwise, it displays in the tag's default engineering unit.
5. Select **Down** to list the results in rows by tag name, or **Across** to list the results in columns by tag name.
6. You can choose the **Time** to retrieve the tag values for in two ways:
   a. Select **Time selection**, and choose a time value from the drop-down list.
   b. Select **Cell reference**, and choose cells in your spreadsheet containing datetime values for the start and end time.
7. Select the time **Interval** over which to retrieve values.

   Note: When retrieving values for a time period of one day or longer, and with a time interval of one hour or

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

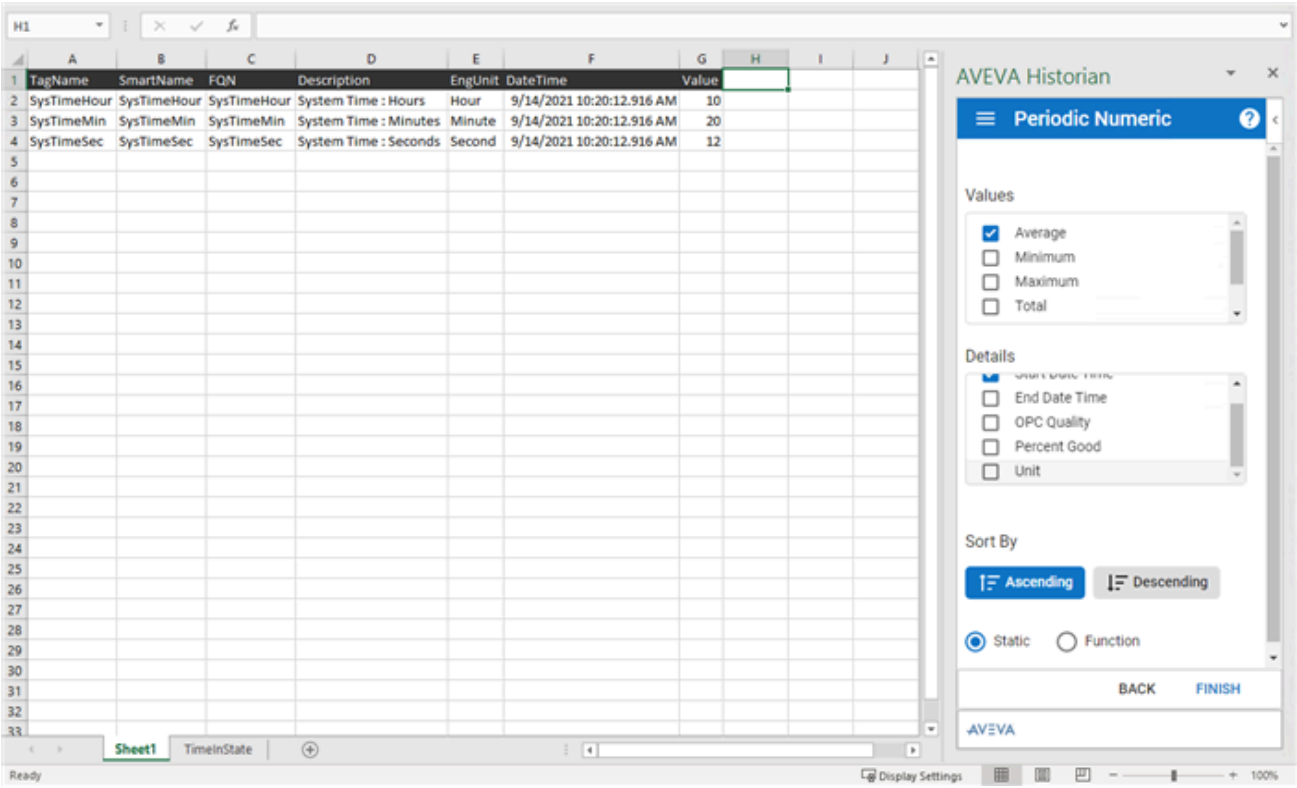
longer, the start time of the first cycle is rounded up to the nearest hour, and the summary is retrieved per the newly-aligned intervals.

For example, if you select a start time of November 21 at 6:30 AM, an end time of November 22 at 10:30 AM, and a time interval of 1 hour, the start time is rounded up to November 21 at 7:00 AM.

The retrieved intervals are:

November 21 7:00 AM to November 21 8:00 AM
November 21 8:00 AM to November 21 9:00 AM
November 21 9:00 AM to November 21 10:00 AM
...
November 22 10:00 AM to November 22 10:30 AM

8. Click **NEXT**. The next page of the add-in displays.

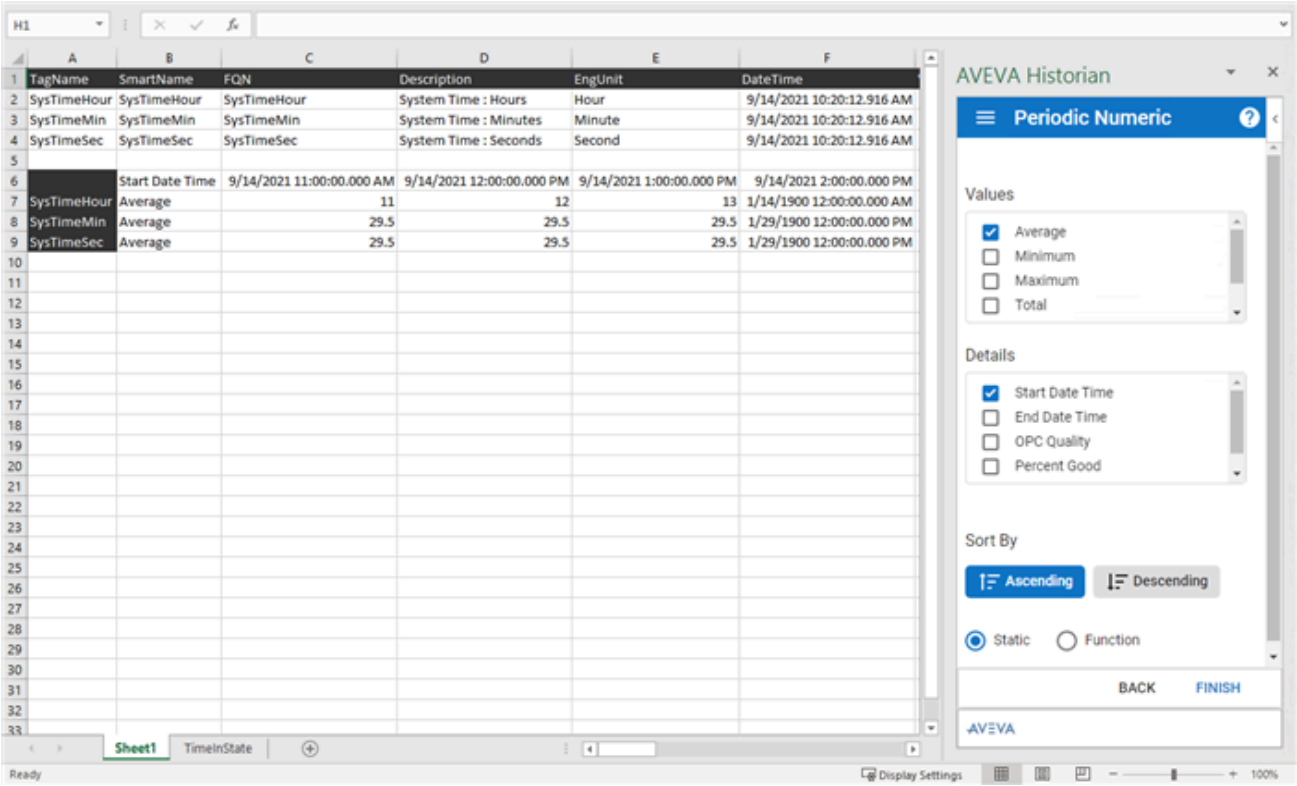


9. Select which **Values** to include for each tag. You can choose to include any combination of **Average**, **Minimum**, **Maximum**, **Total**, **Count**, and **Last**.
10. Select which **Details** to retrieve along with the values for the selected tags. You can choose to include any combination of **Start Date Time**, **End Date Time**, **OPC Quality**, **Percent Good**, and **Unit**.

Note: In most cases you only need to display one time value to identify the interval, usually the start time.

11. Select whether to sort the data in **Ascending** or **Descending** order by time.
12. Select **Static** or **Function** to decide how to retrieve the requested data. If you select **Static**, the data is retrieved and inserted directly into the report. If you select **Function**, a formula is inserted instead, using the *HISTORIAN.PERIODIC* function. If you edit the formula manually, the retrieved data will be refreshed immediately based on your changes. See Using the HISTORIAN.PERIODIC Function for more information.
13. Click **FINISH**. The selected tag information is retrieved and copied to your spreadsheet.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Using the HISTORIAN.PERIODIC Function

Using the HISTORIAN.PERIODIC Function

The HISTORIAN.PERIODIC function retrieves a periodic numeric summary for a specified list of tags.

Syntax

HISTORIAN.PERIODIC(*TagList, StartTime, EndTime, Interval, Details, Down, Ascending, UOMAddress, SliceBy*)

- **TagList** - *Required*. The range of cells containing the list of tag names you want to retrieve information about.
- **StartTime** - *Required*. The date and time to start the data summary, expressed as a serial date number.
- **EndTime** - *Required*. The date and time to end the data summary, expressed as a serial date number.
- **Interval** - *Required*. The time interval over which values will be retrieved, expressed as a number of days. For example, 1/24 represents an interval of approximately one hour, while 1/4 represents an interval of 6 hours.
- **Details** - *Required*. A comma-separated list of additional properties to retrieve for the selected tags. The following properties are supported: `Average`, `Minimum`, `Maximum`, `Total`, `Count`, `Last`, `StartDateTime`, `EndDateTime`, `OpcQuality`, `PercentGood` and `Unit`.
- **Down** - *Optional*. If `TRUE`, the tags are listed in rows. If `FALSE`, the tags are listed in columns. If no value is specified, the default value of `TRUE` (rows) is used.
- **Ascending** - *Optional*. If `TRUE`, data is sorted in ascending order by time. Otherwise, the data is sorted in

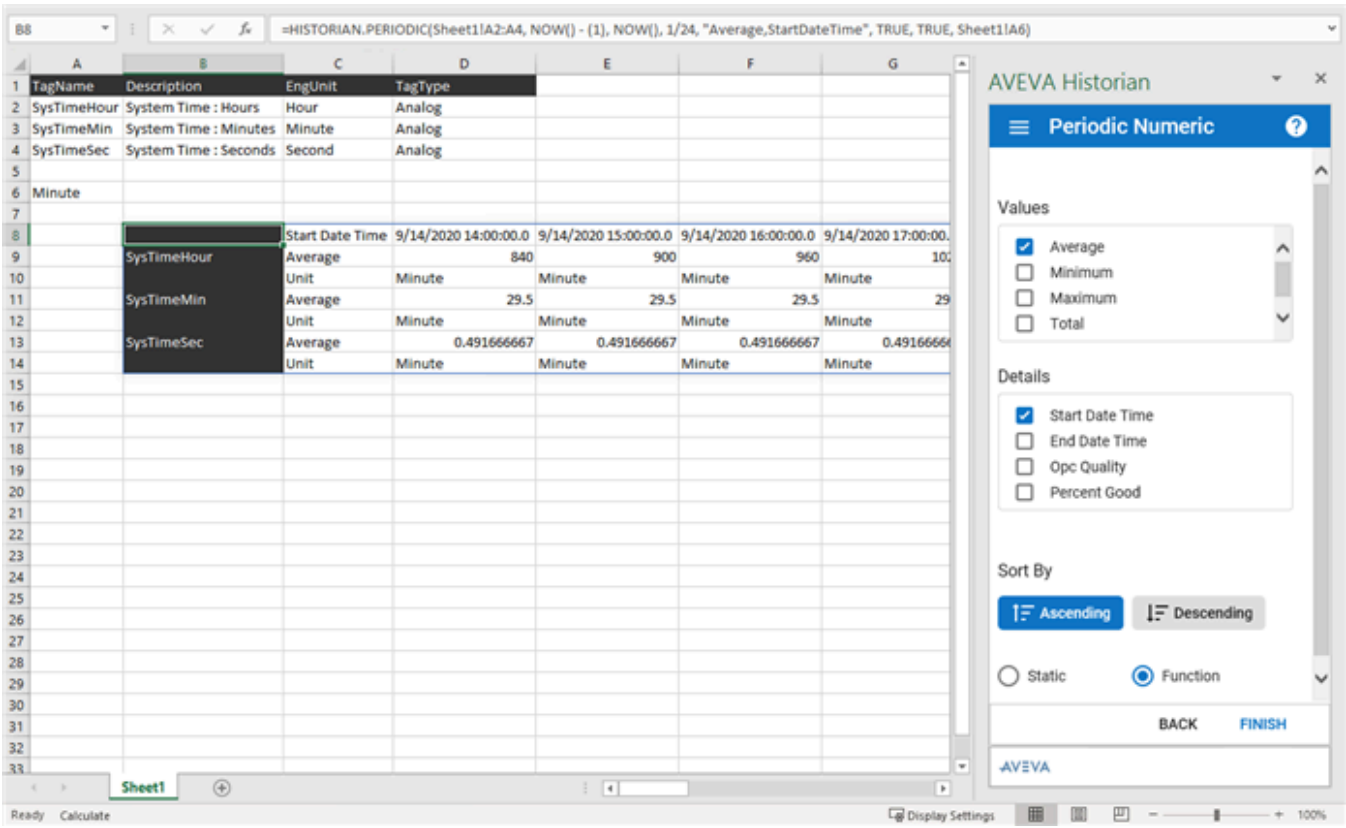
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

descending order by time. If no value is specified, the default value of TRUE (ascending) is used.

- **UOMAddress** - *Optional*. A cell reference containing the name of an engineering unit. When tag data is retrieved, it is converted to display in this unit if an appropriate conversion is defined in the Engineering Unit Catalog. Otherwise, it displays in the tag's default engineering unit.
- **SliceBy** - *Optional*. The range of cells containing a list of SliceBy tag names.

Example

Recreating the simple report from the previous example, we select the **Function** option instead of the **Static** option. This results in a formula being inserted into the spreadsheet, instead of static data. There is a brief pause while communicating with the server, then the retrieved data is displayed.



The formula generated using this example is:

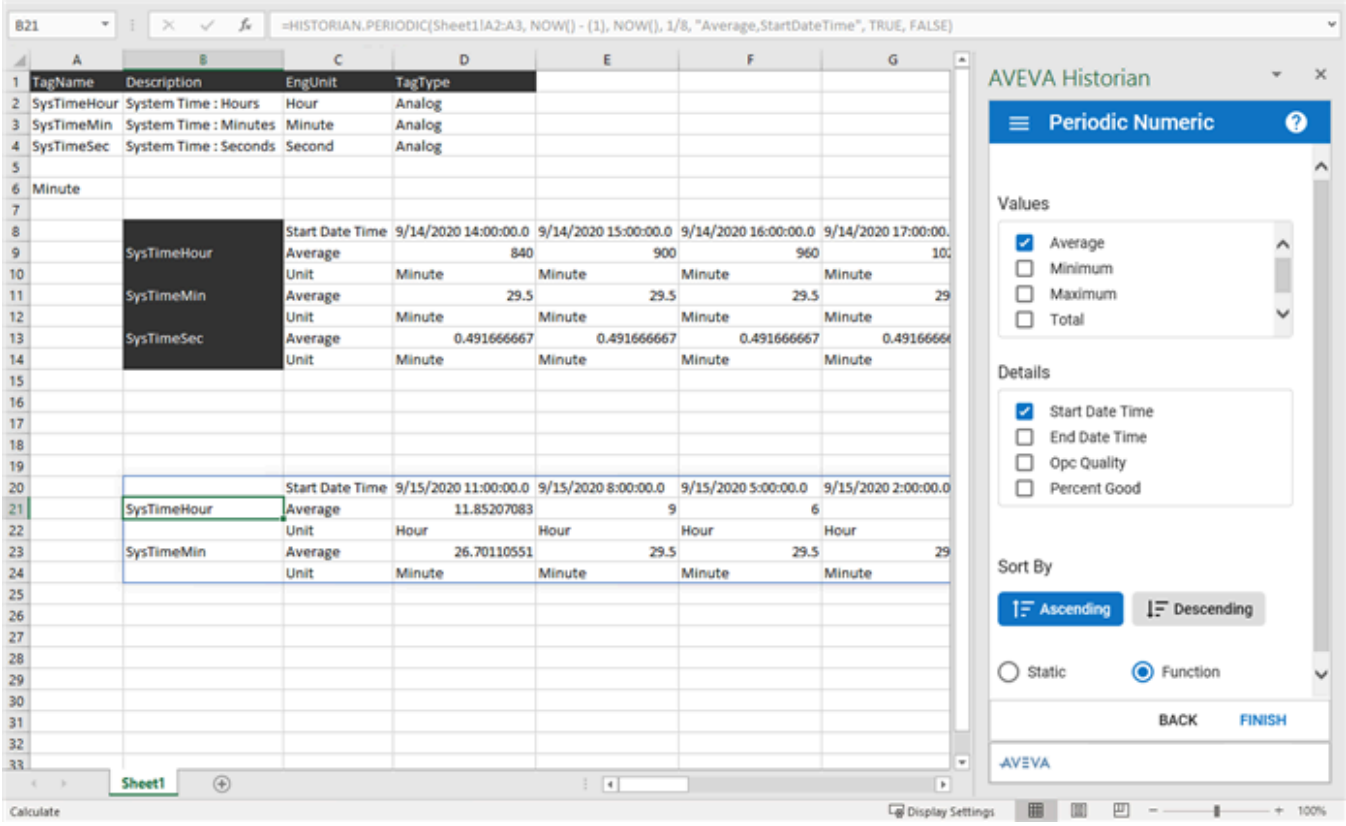
```
=HISTORIAN.PERIODIC(Sheet1!A2:A4, NOW() - (1), NOW(), 1/24, "Average,StartDateTime", TRUE, TRUE, Sheet1!A6)
```

You can modify the formula, and the results refresh automatically to reflect your changes. You can also use the function directly, without using the add-in panel. For example, the following formula is entered into cell B20:

```
=HISTORIAN.PERIODIC(Sheet1!A2:A3, NOW() - (1), NOW(), 1/8, "Average,StartDateTime", TRUE, FALSE)
```

The resulting report is similar to the first, but one of the tags is omitted because of the modified range, the time interval changes from hourly (1/24) to every 3 hours (1/8), the data is sorted in descending order, and, because there is no conversion unit specified, the data displays in the default unit for each tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If your version of Excel does not support dynamic arrays, a message similar to the following displays:

In this case, although you can still use the custom function, the size of the output range cannot be refreshed automatically. For example, if you run the function with a date range of 1 day (returning 24 rows or columns) and want to change the date range to 2 days (returning 48 rows or columns), the output range cannot be automatically resized to accommodate the additional results. You must manually select the new output range, edit the function parameters, and then press **Ctrl-Shift-Enter**.

Retrieving Time in State

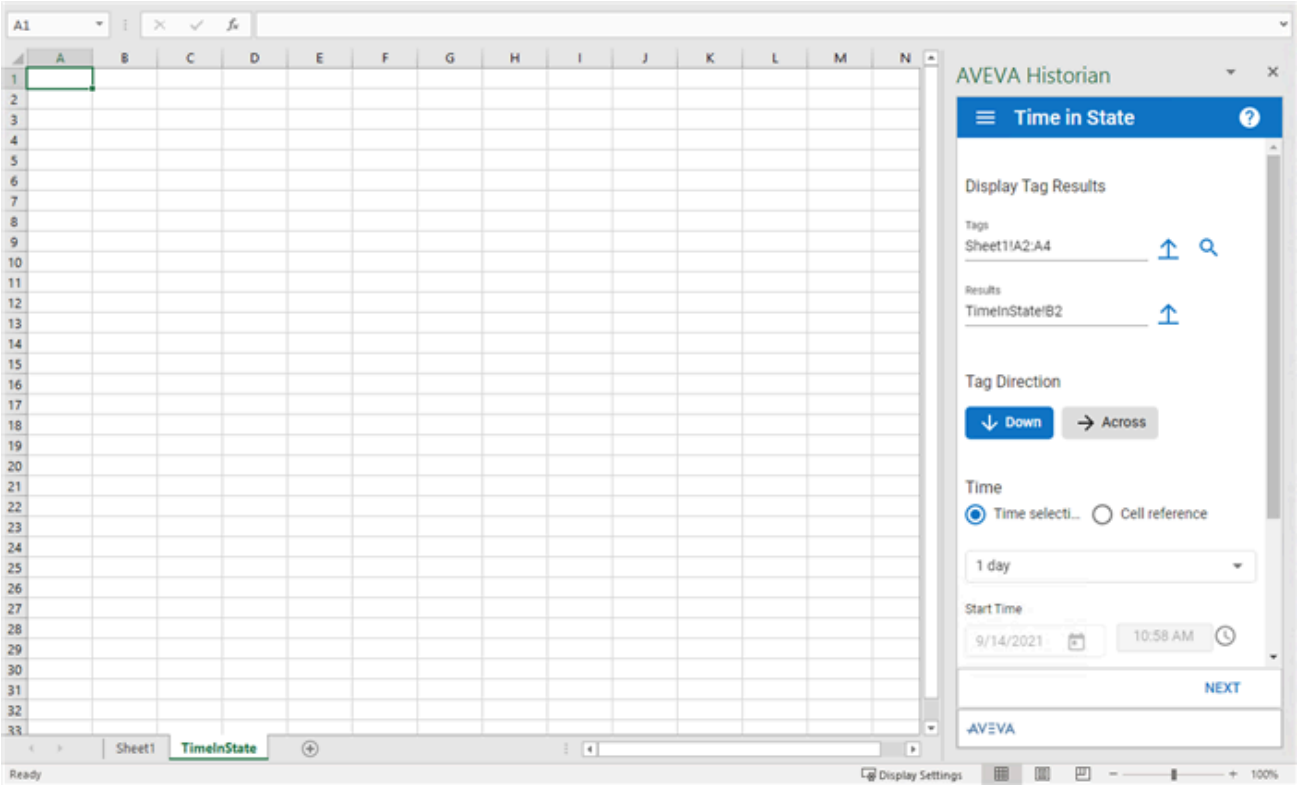
Note: The Time in State query only applies to Discrete or String tags. Analog tags are not supported.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To retrieve time in state values:

1. Select **Time In State** from the AVEVA Historian add-in menu. If the add-in is already displayed, you can also select **Time In State** from the hamburger menu . The AVEVA Historian add-in panel displays.

   Note: When you select **Time In State**, a **TimeInState** sheet is added to your workbook if it does not already exist, and results are placed in this sheet by default.



2. To choose the list of tags to retrieve information for, select a range of cells containing a list of tag names, then click beside the **Tags** field.

   Note: If you used the **Tag Search** page to select your tags, the range is automatically pre-selected for you.

3. Select the cell where you want to insert the results, then click beside the **Results** field.
4. Select **Down** to list the results in rows by tag name, or **Across** to list the results in columns by tag name.
5. You can choose the **Time** to retrieve the tag values for in two ways:
   a. Select **Time selection**, and choose a time value from the drop-down list.
   b. Select **Cell reference**, and choose cells in your spreadsheet containing datetime values for the start and end time.
6. Select the time **Interval** over which to retrieve values.

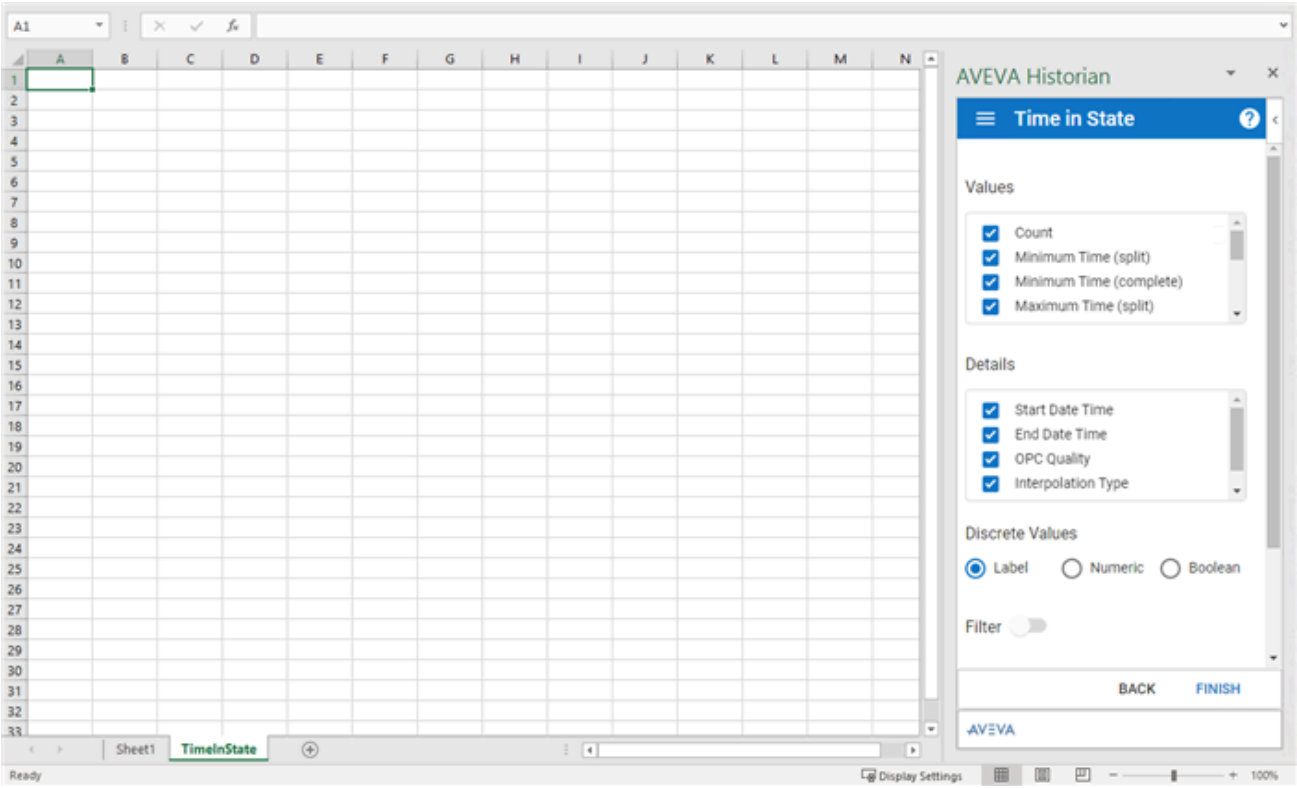
   Note: When retrieving values for a time period of one day or longer, and with a time interval of one hour or longer, the start time of the first cycle is rounded up to the nearest hour, and the summary is retrieved per the newly-aligned intervals.

   For example, if you select a start time of November 21 at 6:30 AM, an end time of November 22 at 10:30 AM, and a time interval of 1 hour, the start time is rounded up to November 21 at 7:00 AM.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The retrieved intervals are:
November 21 7:00 AM to November 21 8:00 AM November 21 8:00 AM to November 21 9:00 AM November 21 9:00 AM to November 21 10:00 AM ... November 22 10:00 AM to November 22 10:30 AM

7. Click **NEXT**. The next page of the add-in displays.



8. Select which **Values** to include for each tag. You can choose to include any combination of:
   - **Count -** The number of values occurring within the time interval.
   - **Minimum Time (split) -** The minimum time spent in this state for the tag over the time interval, including occurrences that only fall partially within the interval.
   - **Minimum Time (complete) -** The minimum time spent in state for the tag over the time interval, excluding occurrences that do not fall completely within the interval.
   - **Maximum Time (split) -** The maximum time spent in state for the tag over the time interval, including occurrences that only fall partially within the interval.
   - **Maximum Time (complete) -** The maximum time spent in state for the tag over the time interval, excluding occurrences that do not fall completely within the interval.
   - **Average Time (split) -** The average time spent in state for the tag over the time interval, including occurrences that only fall partially within the interval.
   - **Average Time (complete) -** The average time spent in state for the tag over the time interval, excluding occurrences that do not fall completely within the interval.
   - **Total Time (split) -** The total time spent in state for the tag over the time interval, including occurrences that only fall partially within the interval.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Total Time (complete) -** The total time spent in state for the tag over the time interval, excluding occurrences that do not fall completely within the interval.
- **Percent (split) -** The percentage of the total time spent in state for the tag over the time interval, including occurrences that only fall partially within the interval.
- **Percent (complete) -** The percentage of the total time spent in state for the tag over the time interval, excluding occurrences that do not fall completely within the interval.

9. Select whether to display **Discrete Values** in **Label** format, **Numeric** format, or **Boolean** format.

   Note: A discrete tag has two states, and these tags can have labels associated with the two states. For example, for a tag representing a valve's position, the **Label** values might be "Open" or "Closed", the **Numeric** values are 0 or 1, and the **Boolean** values are True or False.

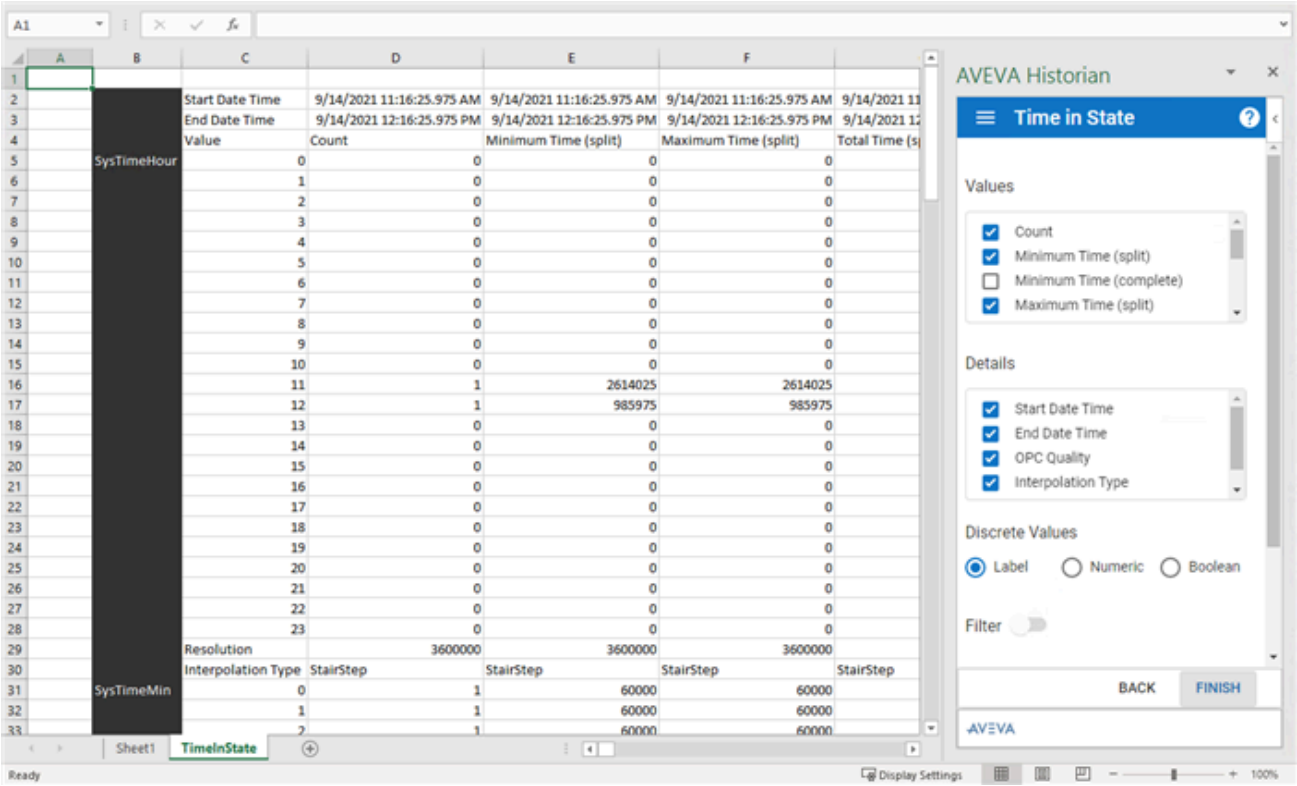
10. Select which **Details** to retrieve along with the values for the selected tags. You can choose to include any combination of **Start Date Time**, **End Date Time**, **OPC Quality**, **Interpolation Type**, and **Resolution**.
11. If there are discrete tags in the list, the **Filter** option is available.

    You can use this option to filter your results, based on the selected format for displaying discrete values.

    a. Turn on the **Filter** by activating the toggle switch

.
    b. A list of state filter options displays below the toggle switch when it is activated. The items in the list are dependent on the selected **Discrete Values** format:
       a. **Label** - All labels for the selected discrete tags are available to choose from.
       b. **Numeric** - The values 0 and 1 are available.
       c. **Boolean** - True and False values are available.
    c. Select the state values from the filter list that you want to include in your results. Any state values not selected are filtered from the results. For example, for a tag representing a valve's position, you may not want both *Open* and *Closed* states included in your report, since you can infer the time spent in one state from the time spent in the other.
12. Select whether to sort the data in **Ascending** or **Descending** order by time.
13. Select **Static** or **Function** to decide how to retrieve the requested data. If you select **Static**, the data will be retrieved and inserted directly into the report. If you select **Function**, a formula will be inserted instead, using the *HISTORIAN.TIMEINSTATE* function. If you edit the formula manually, the retrieved data will be refreshed immediately based on your changes. See Using the HISTORIAN.TIMEINSTATE Function for more information.
14. Click **FINISH**. The selected tag information is retrieved and copied to your spreadsheet.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Using the HISTORIAN.TIMEINSTATE Function

Using the HISTORIAN.TIMEINSTATE Function

The HISTORIAN.TIMEINSTATE function retrieves a time-in-state summary for a specified list of tags.

Syntax

HISTORIAN.TIMEINSTATE(*TagList, StartTime, EndTime, Interval, Details, Down, Ascending, DiscreteFormat, Filter*)

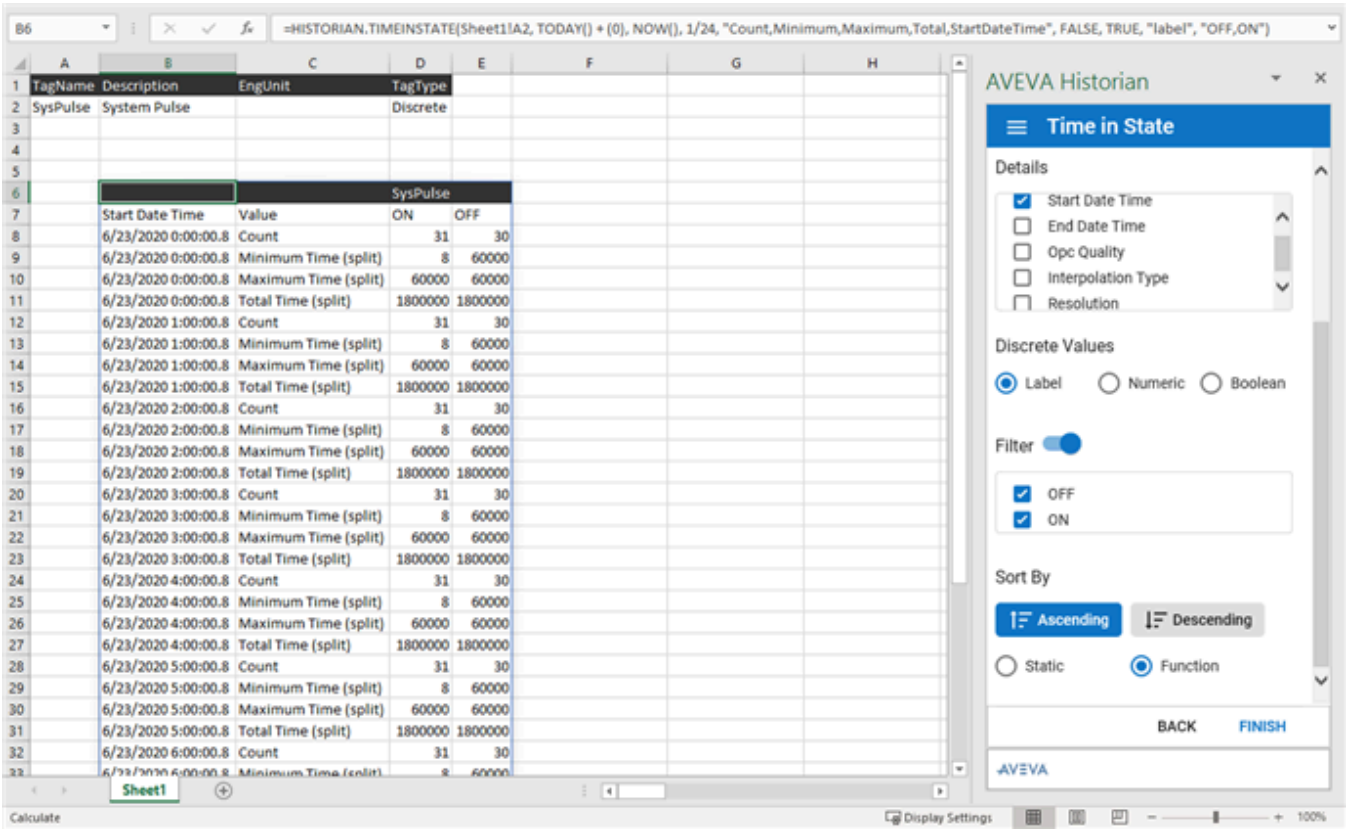
- **TagList** - *Required*. The range of cells containing the list of tag names you want to retrieve information about.
- **StartTime** - *Required*. The date and time to start the data summary, expressed as a serial date number.
- **EndTime** - *Required*. The date and time to end the data summary, expressed as a serial date number.
- **Interval** - *Required*. The time interval over which values will be retrieved, expressed as a number of days. For example, 1/24 represents an interval of one hour, while 1/4 represents an interval of 6 hours.
- **Details** - *Required*. A comma-separated list of properties to retrieve for the selected tags. The following properties are supported: `Count`, `Minimum`, `MinimumContained`, `Maximum`, `MaximumContained`, `Average`, `AverageContained`, `Total`, `TotalContained`, `Percent`, `PercentContained`, `StartDateTime`, `EndDateTime`, `OpcQuality`, `InterpolationType`, and `Resolution`.
- **Down** - *Optional*. If `TRUE`, the tags are listed in rows. If `FALSE`, the tags are listed in columns. If no value is specified, the default value of `TRUE` (rows) is used.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Ascending** - *Optional*. If TRUE, data is sorted in ascending order by time. Otherwise, the data is sorted in descending order by time. If no value is specified, the default value of `TRUE` (ascending) is used.
- **DiscreteFormat** - *Optional*. The format in which to display discrete values. Valid options are `label`, `numeric`, or `boolean`. If no value is specified, the default value of `label` is used.
- **Filter** - *Optional*. A comma-separated list of state values. If a list is provided, only values matching the provided list of states will be included in the output. If no value is provided, then the output is not filtered.

Example

Recreating the simple report from the previous example, we select the **Function** option instead of the **Static** option. This results in a formula being inserted into the spreadsheet, instead of static data. There is a brief pause while communicating with the server, then the retrieved data is displayed.



The formula generated using this example is:

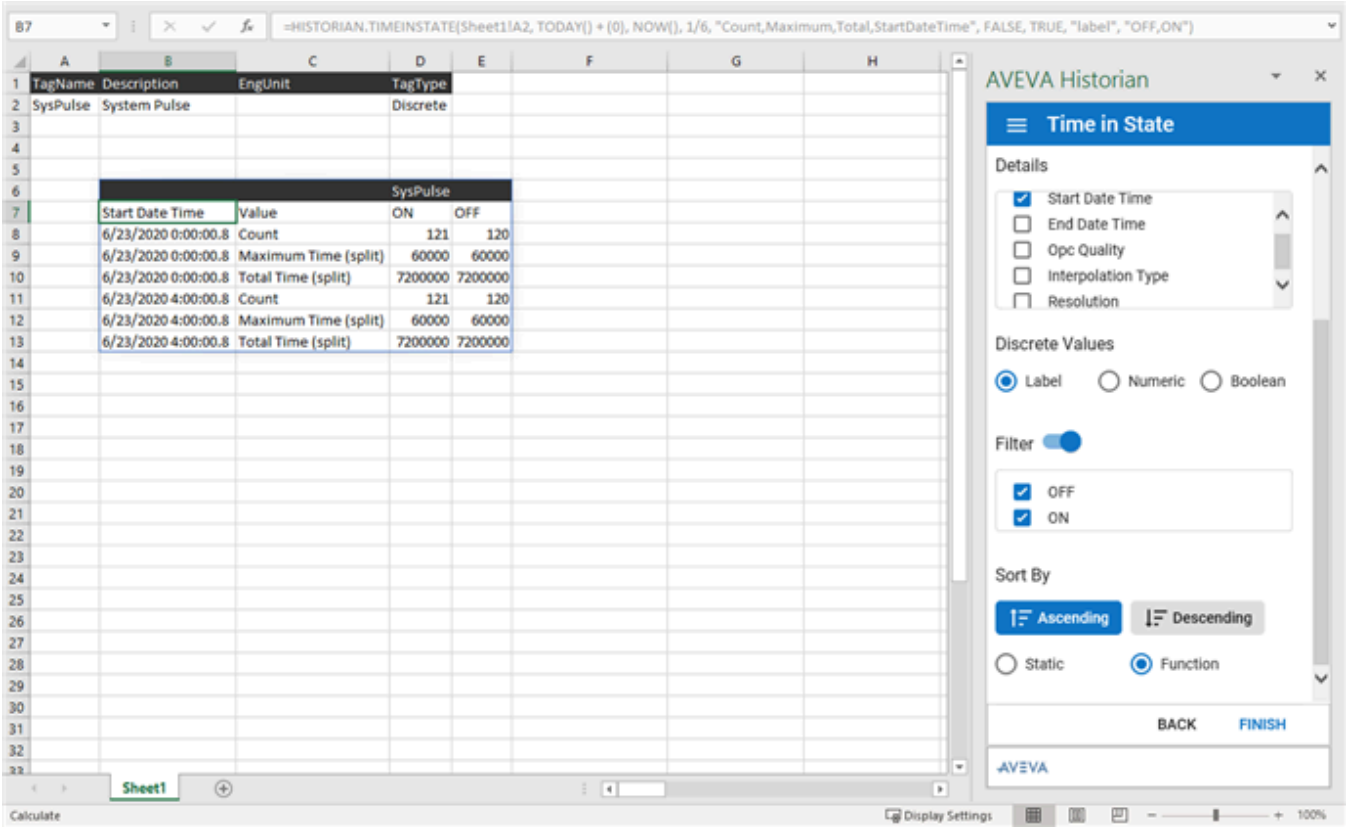
```
=HISTORIAN.TIMEINSTATE(Sheet1!A2, TODAY() + (0), NOW(), 1/24,
"Count,Minimum,Maximum,Total,StartDateTime", FALSE, TRUE, "label", "OFF,ON")
```

You can modify the formula, and the results refresh automatically to reflect your changes. You can also use the function directly, without using the add-in panel. For example, the formula in cell B6 is replaced with:

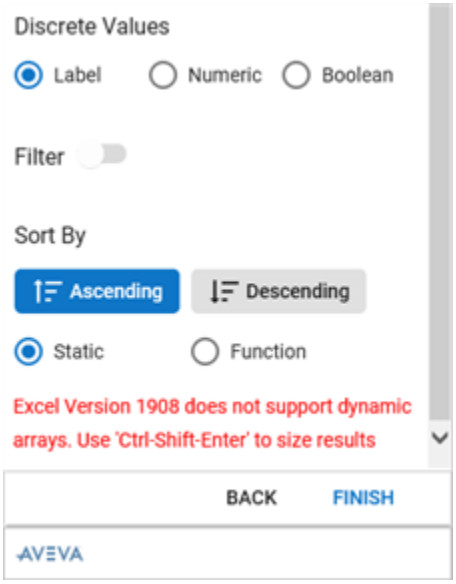
```
=HISTORIAN.TIMEINSTATE(Sheet1!A2, TODAY() + (0), NOW(), 1/6,
"Count,Maximum,Total,StartDateTime", FALSE, TRUE, "label", "OFF,ON")
```

The displayed report is regenerated, and now the time interval is every 4 hours (1/6), and the minimum time property is no longer displayed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If your version of Excel does not support dynamic arrays, a message similar to the following displays:



In this case, although you can still use the custom function, the size of the output range cannot be refreshed automatically. For example, if you run the function with a time interval of 1/6 (every 4 hours) and then want to change the interval to 1/12 (every 2 hours), the output range cannot be automatically resized to accommodate the additional results. You must manually select the new output range, edit the function parameters, and then press **Ctrl-Shift-Enter**.

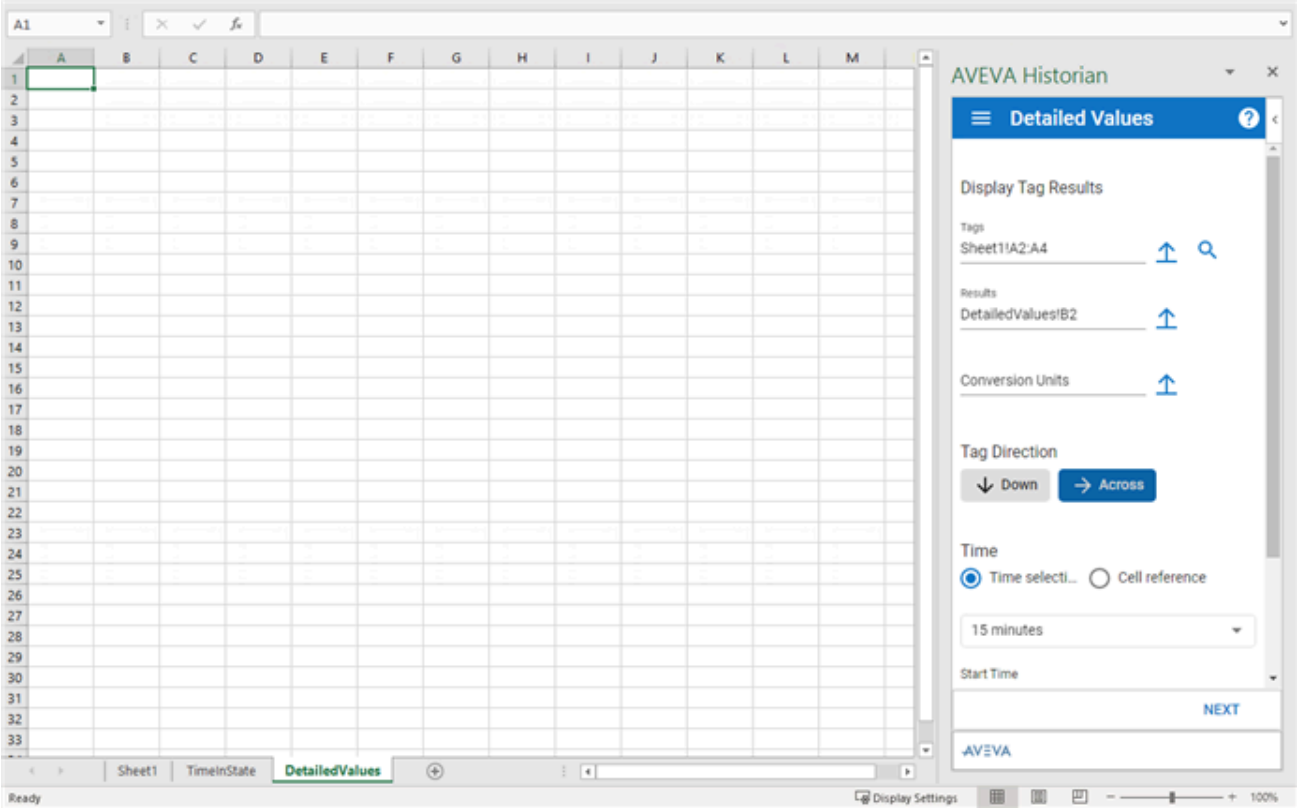
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieving Detailed Values

To retrieve detailed values:

1. Select **Detailed Values** from the AVEVA Historian add-in menu. If the add-in is already displayed, you can also select **Detailed Values** from the hamburger menu . The AVEVA Historian add-in panel displays.

   Note: When you select **Detailed Values**, a **DetailedValues** sheet is added to your workbook if it does not already exist, and results are placed in this sheet by default.



2. To choose the list of tags to retrieve information for, select a range of cells containing a list of tag names, then click beside the **Tags** field.

   Note: If you used the **Tag Search** page to select your tags, the range is automatically pre-selected for you.

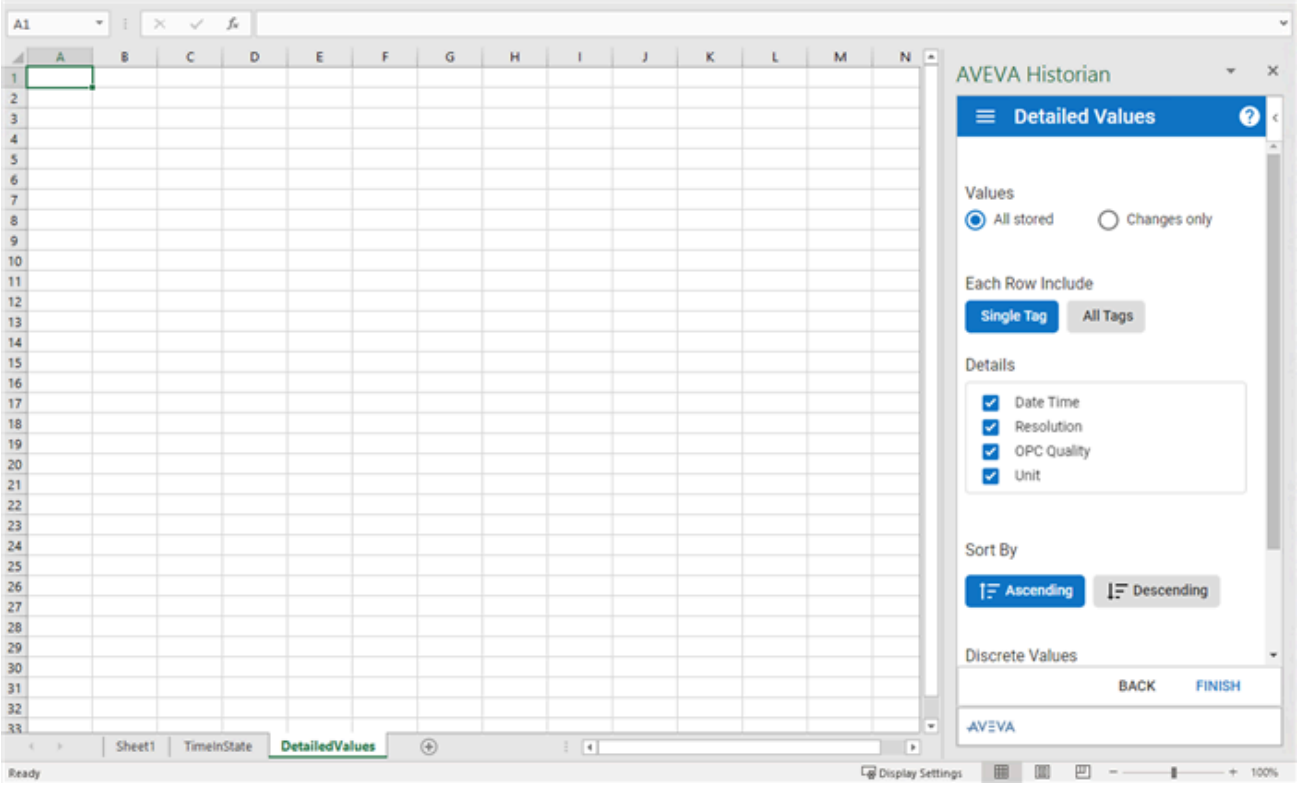
3. Select the cell where you want to insert the results, then click beside the **Results** field.
4. The **Conversion Units** field allows you to compare values more easily by converting the retrieved data to display in the same engineering unit. Select a cell or range containing the names of target engineering units, then click beside the Conversion Units field. When tag data is retrieved, it displays in the selected unit if an appropriate conversion is available in the Engineering Unit Catalog. Otherwise, it displays in the default engineering unit for the tag.
5. Choose the **Tag Direction**. Select **Across** to retrieve the data in columns, one value per row. Select **Down** to retrieve the data in rows, one value per column.
6. You can choose the **Time** for which to retrieve the tag values in two ways:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

a. Select **Time selection**, and choose a time value from the drop-down list.

b. Select **Cell reference**, and choose cells in your spreadsheet containing datetime values for the start and end time.

7. Click **NEXT**. The next page of the add-in displays.



8. Select which **Values** to include for each tag. Choosing **All stored** retrieves all stored values for each tag, while choosing **Changes only** retrieves only the values where there is a change from the previous reading.

9. Choose whether to view a single tag or all tags in each row or column of retrieved data.

a. If you select **Single Tag** and the Tag Direction is Across, each row represents a specific time stamp and the values are arranged in columns by tag name.

	SysTimeHour				SysTimeMin			
Date Time	Value	Resolution	OPC Quality	Unit	Value	Resolution	OPC Quality	Unit
12/15/2021 10:16:06.393 AM	10	233607	192	Hour	16	53607	192	Minute
12/15/2021 10:17:00.000 AM	10	233607	192	Hour	17	60000	192	Minute
12/15/2021 10:18:00.000 AM	10	233607	192	Hour	18	60000	192	Minute
12/15/2021 10:19:00.000 AM	10	233607	192	Hour	19	60000	192	Minute
12/15/2021 10:20:00.000 AM	10	0	192	Hour	20	60000	192	Minute
12/15/2021 10:21:00.000 AM	10	0	192	Hour	21	60000	192	Minute
12/15/2021 10:22:00.000 AM	10	0	192	Hour	22	60000	192	Minute
12/15/2021 10:23:00.000 AM	10	0	192	Hour	23	60000	192	Minute
12/15/2021 10:24:00.000 AM	10	0	192	Hour	24	60000	192	Minute

b. If you select **All Tags** and the Tag Direction is Across, each row represents a single value for a specific tag. If multiple tags contain values for the same time stamp, the values are listed in separate rows.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Date Time	TagName	Value	Unit	OPC Quality
12/15/2021 10:16:41.763 AM	SysTimeHour	10	Hour	192
12/15/2021 10:16:41.763 AM	SysTimeMin	16	Minute	192
12/15/2021 10:17:00.000 AM	SysTimeMin	17	Minute	192
12/15/2021 10:18:00.000 AM	SysTimeMin	18	Minute	192
12/15/2021 10:19:00.000 AM	SysTimeMin	19	Minute	192
12/15/2021 10:20:00.000 AM	SysTimeHour	10	Hour	192
12/15/2021 10:20:00.000 AM	SysTimeMin	20	Minute	192
12/15/2021 10:21:00.000 AM	SysTimeMin	21	Minute	192
12/15/2021 10:22:00.000 AM	SysTimeMin	22	Minute	192
12/15/2021 10:23:00.000 AM	SysTimeMin	23	Minute	192

c. If you select **Single Tag** and the Tag Direction is Down, each column represents a specific time stamp and the values are arranged in rows by tag name.

	Date Time	12/15/2021 10:17:08.636 AM	12/15/2021 10:18:00.000 AM	12/15/2021 10:19:00.000 AM	12/15/2021 1(
SysTimeHour	Value	10	10	10	
	Resolution	171364	171364	171364	
	OPC Quality	192	192	192	
	Unit	Hour	Hour	Hour	Hour
SysTimeMin	Value	17	18	19	
	Resolution	51364	60000	60000	
	OPC Quality	192	192	192	
	Unit	Minute	Minute	Minute	Minute

d. If you select **All Tags** and the Tag Direction is Down, each column represents a single value for a specific tag. If multiple tags contain values for the same time stamp, the values are listed in separate columns.

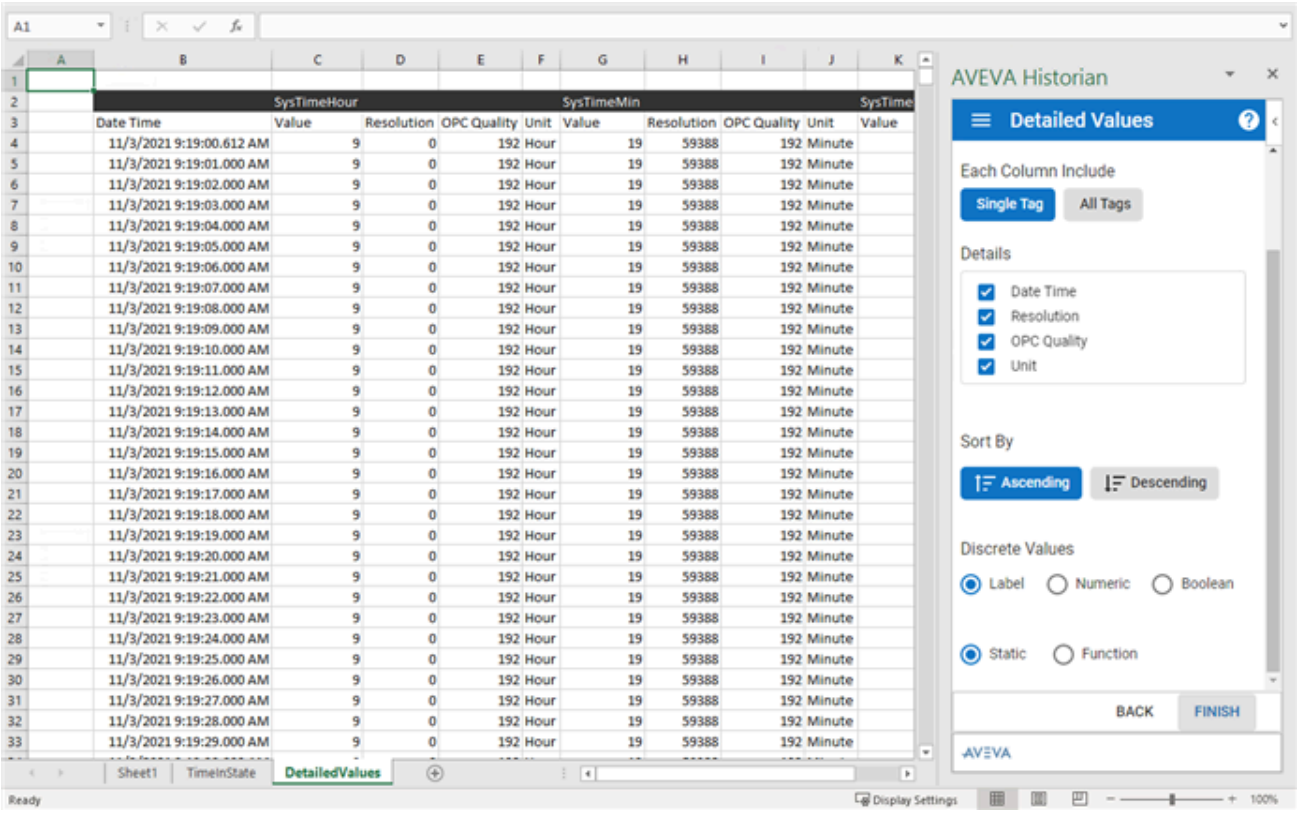
Date Time	12/15/2021 10:17:28.768 AM	12/15/2021 10:17:28.768 AM	12/15/2021 10:18:00.000 AM	12/15/2021 10:19:00.
TagName	SysTimeHour	SysTimeMin	SysTimeMin	SysTimeMin
Value	10	17	18	
Unit	Hour	Minute	Minute	Minute
OPC Quality	192	192	192	

10. Select which **Details** to retrieve along with the values for the selected tags. You can choose to include any combination of **Date Time**, **Resolution**, **OPC Quality**, and **Unit**.
11. Choose whether to sort by tag name in **Ascending** or **Descending** order.
12. Select whether to display **Discrete Values** in **Label** format, **Numeric** format, or **Boolean** format.

Note: A discrete tag has two states, and these tags can have labels associated with the two states. For example, for a tag representing a valve's position, the **Label** values might be "Open" or "Closed", the **Numeric** values are 0 or 1, and the **Boolean** values are True or False.

13. Select **Static** or **Function** to decide how to retrieve the requested data. If you select **Static**, the data will be retrieved and inserted directly into the report. If you select **Function**, a formula will be inserted instead, using the *HISTORIAN.DETAIL* function. If you edit the formula manually, the retrieved data will be refreshed immediately based on your changes. See Using the HISTORIAN.DETAIL Function for more information.
14. Click **FINISH**. The selected tag information is retrieved and copied to your spreadsheet.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Using the HISTORIAN.DETAIL Function

Using the HISTORIAN.DETAIL Function

The HISTORIAN.DETAIL function retrieves detailed values for a specified list of tags.

Syntax

HISTORIAN.DETAIL(*TagList, StartTime, EndTime, Retrieval, Format, Details, Down, Ascending, DiscreteFormat, UOMAddress*)

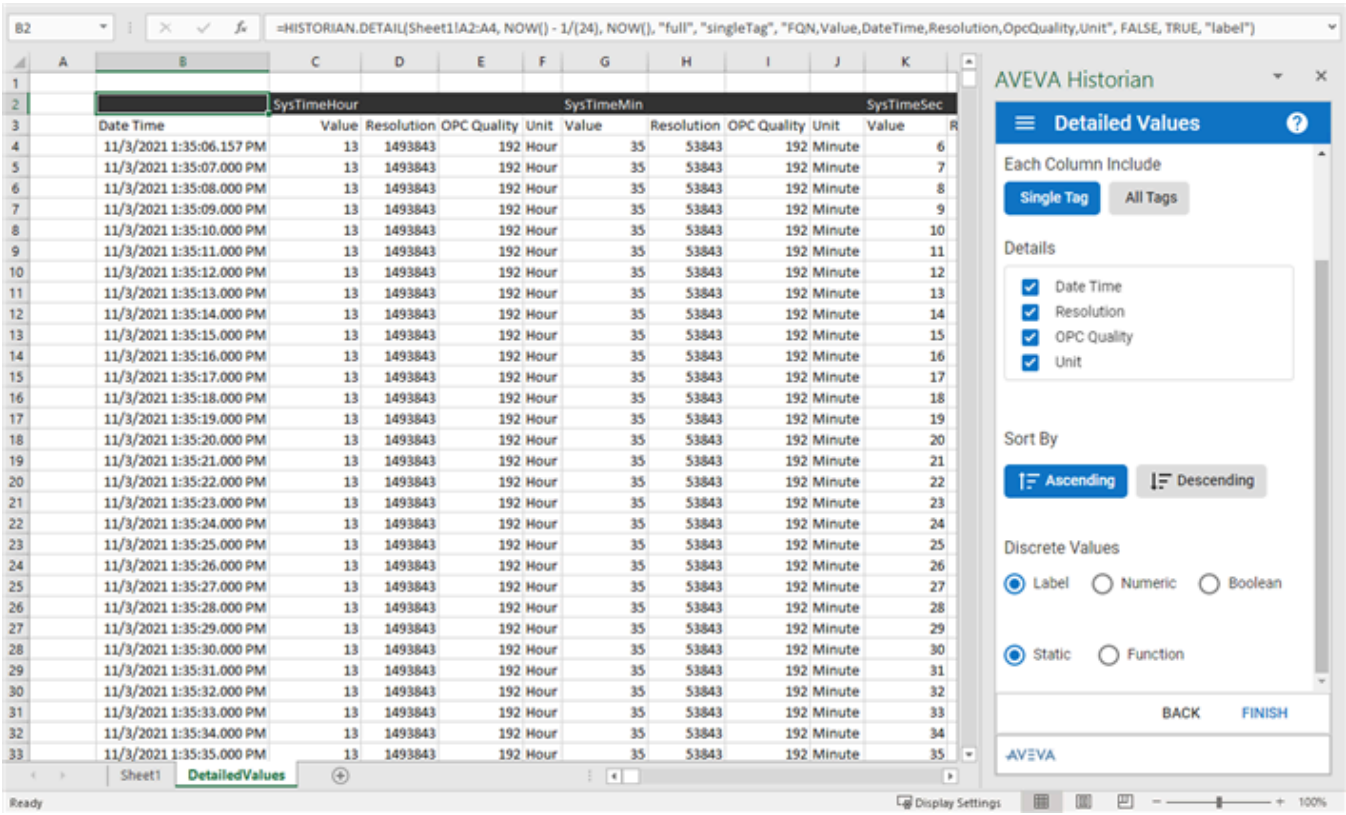
- **TagList** - *Required*. The range of cells containing the list of tag names you want to retrieve information about.
- **StartTime** - *Required*. The date and time to start the data summary, expressed as a serial date number.
- **EndTime** - *Required*. The date and time to end the data summary, expressed as a serial date number.
- **Retrieval** - *Required*. Indicates the retrieval mode. Supported modes are `full`, `delta`.
- **Format** - *Required*. Indicates the output format. Supported modes are `singleTag`, `allTags`.
- **Details** - *Required*. A comma-separated list of properties to retrieve for the selected tags. The following properties are supported: `DateTime`, `Resolution`, `OpcQuality`, `Unit`, and `Value`.
- **Down** - *Optional*. If `TRUE`, the tags are listed in rows. If `FALSE`, the tags are listed in columns. If no value is specified, the default value of `TRUE` (rows) is used.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Ascending** - *Optional*. If `TRUE`, data is sorted in ascending order by time. Otherwise, the data is sorted in descending order by time. If no value is specified, the default value of `TRUE` (ascending) is used.
- **DiscreteFormat** - *Optional*. The format in which to display discrete values. Valid options are `label`, `numeric`, or `boolean`. If no value is specified, the default value of `label` is used.
- **UOMAddress** - *Optional*. A cell, or range of cells, containing the names of engineering units. When tag data is retrieved, it is converted to display in this unit if an appropriate conversion is defined in the Engineering Units Catalog. Otherwise, it displays in the tag's default engineering unit.

Example

Recreating the simple report from the previous example, we select the **Function** option instead of the **Static** option. This results in a formula being inserted into the spreadsheet, instead of static data. There is a brief pause while communicating with the server, then the retrieved data is displayed.



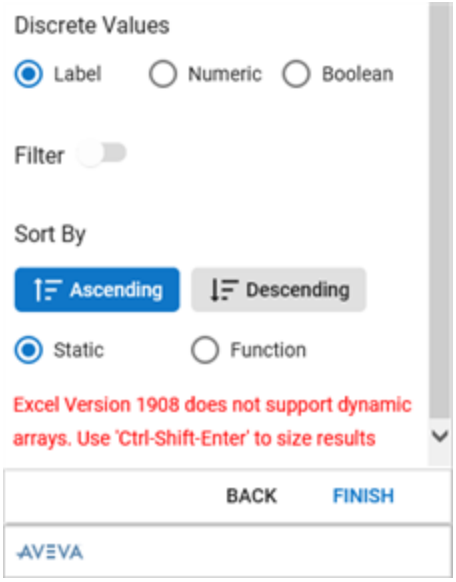
The formula generated using this example is:

```
=HISTORIAN.DETAIL(Sheet1!A2:A4, NOW() - 1/(24), NOW(), "full", "singleTag",
"FQN,Value,DateTime,Resolution,OpcQuality,Unit", FALSE, TRUE, "label")
```

You can modify the formula, and the results refresh automatically to reflect your changes. You can also use the function directly, without using the add-in panel.

If your version of Excel does not support dynamic arrays, a message similar to the following displays:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In this case, although you can still use the custom function, the size of the output range cannot be refreshed automatically. You must manually size the range by selecting it, and then press **Ctrl-Shift-Enter**.

Using Smart Names for Tags

Tag labels represent many different pieces of information, such as tag names, location, units, keywords, etc. This can make it difficult to differentiate fully-qualified tag names when multiple tags share some common values.

Smart Names can be used to make tag names more readable in your reports. Smart Names are generated for a group of fully-qualified tag names by applying the following rules:

- For each tag, start with the *Alias* if one is defined, otherwise start with the *TagName*
- Prefixes and suffixes are treated independently
- Prefixes and suffixes break only on whitespace and punctuation: - , . _ / \ |
- Prefixes and suffixes are case-insensitive
- Common prefixes and suffixes are truncated from all labels
- If removing common prefixes and suffixes results in an empty label, do not truncate
- If removing common prefixes and suffixes results in any duplicate labels, do not truncate
- The final format of the Smart Name is "*Trimmed Label* [*Trimmed Location*]".

  Note: The location portion of the Smart Name is optional, depending on whether location information is both available, and different, for the selected tags

Example 1:

This example demonstrates tags with common prefixes and suffixes.

Labels before trimming	Labels after trimming
a.b.c.d.product.level.LA	product

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

a.b.c.d.water.level.LA	water

Example 2:

This example demonstrates tags with common prefixes and suffixes that would result in an empty label if truncated.

Labels before trimming	Labels after trimming
a.b.c.d	a.b.c.d
a.b.e.c.d	a.b.e.c.d

Example 3:

This example demonstrates tags with common suffixes, with available location information.

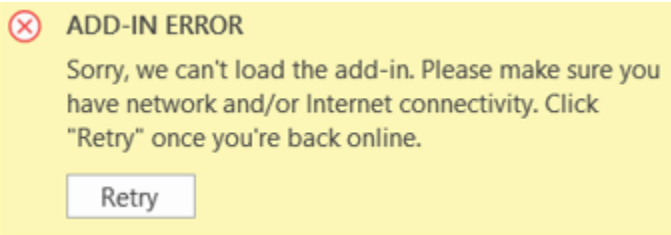
Labels before trimming	Location before trimming	Labels after trimming
product.level.LA	LA/Plant1	product
water.level.LA	LA/Plant1	water [Plant1]
water.level.LA	LA/Plant2	water [Plant2]

Troubleshooting Issues using the Excel Add-In

This section describes some common issues you may encounter while trying to install or use, and provides steps to help you resolve the issue.

Issue: Network connectivity error when loading the add-in

When you attempt to use the add-in, a network connectivity error message displays:



This can occur when Office needs to connect to the internet to refresh your subscription information, for example. To resolve this issue, ensure your system is connected to the internet and try to load the add-in again.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

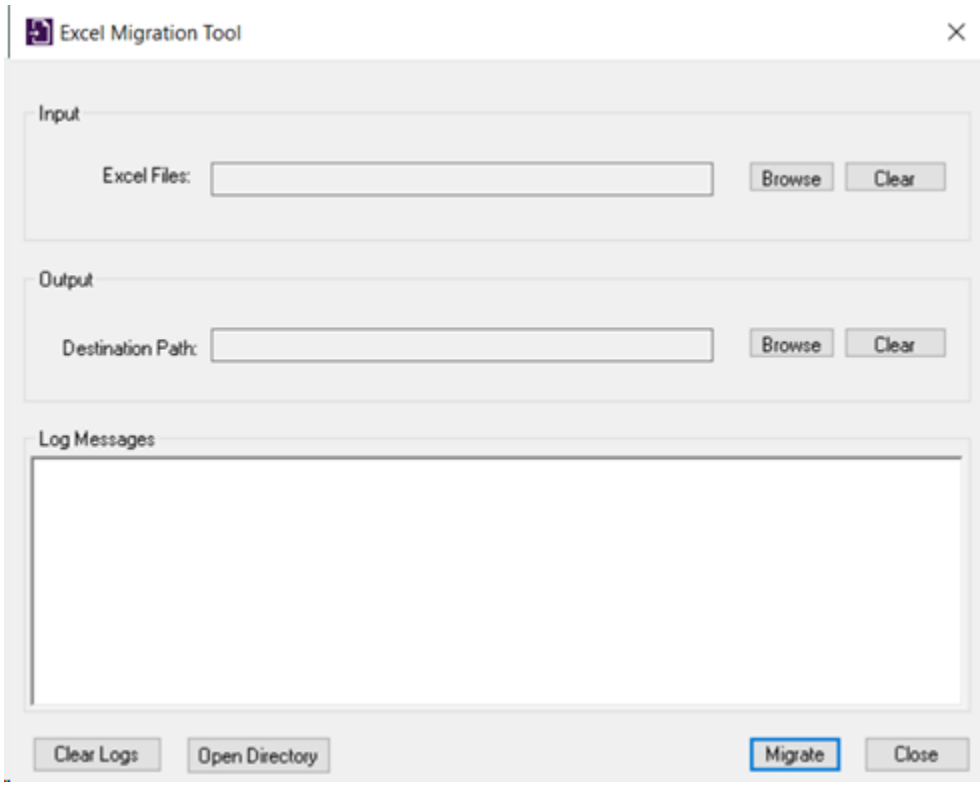
Migrate data from Historian Client Workbook to the Excel Add-in

The Excel Migration Tool updates Excel workbooks created in Historian Client Workbook to formulas compatible with the Historian Excel add-in.

This tool is supported to migrate files from Excel 2016 or later and works with Office365 Excel 32/64bit.

Use the Excel Migration tool

1. Open **<Historian installation folder>\Historian\x64\aahExcelMigration.exe**. The default installation path is **C:\Program Files (x86)\Wonderware\Historian\x64\aahExcelMigration.exe**.



2. In the Input area, select **Browse** and select the Excel files that you wish to migrate. (Excel files must be in the .xlsx format.)
3. In the Output area, the Destination Path field shows where the migrated files will be saved. Select **Browse** if you wish to change the path.
4. Select **Migrate**. The migration runs. Information about the status of the migration appears in the Log Messages field. Select **Clear Logs** if you wish to remove the log messages.
5. Select **Open Directory** to open the destination path that the Excel files have been migrated to.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Supported formulas for migration

Supported formulas for migration

Formulas	Historian Client Workbook Formulas	Excel Add-in Formula
Live Formulas	wwAnalogLive1, wwDiscreteLive1, wwStringLive1, wwlive1, wwAnalogLive, wwDiscreteLive, wwStringLive, wwReplicatedLive	Historian.SingleValue
Tag Details Formulas	wwAnalogTagDetails1, wwDiscreteTagDetails1, wwStringTagDetails1, wwReplicatedTagDetails	Historian.Info
History Formulas	wwHistory3, wwWideHistory3, wwStringHistory, wwDiscreteHistory, wwAnalogHistory	Historian.Detail
Aggregate Formulas	wwAggregate, WwwideAggregate. TagAnalysis (Analog Tag Analysis & Discrete Tag Analysis)	Historian.Periodic

Event tags are not supported, so they are omitted during migration.

Live formula migration mapping details

Live formula migration mapping details

wwlive1 Formula

```
wwLive1 =wwLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality,
DisplaySourceTag, DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplayOPCQuality	OPC Quailty--Details
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

wwAnalogLive1

```
wwAnalogLive1 =wwAnalogLive1(DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

wwDiscreteLive1:

```
wwDiscreteLive1 =wwDiscreteLive1( DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

wwStringLive1

```
wwStringLive1 =wwStringLive1( DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

wwReplicatedLive

```
wwReplicatedLive =wwReplicatedLive(DataSource, TagRange, DisplayTagName, DisplayDatetime,
DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag,
DisplaySourceServer, OptionRange)
```

Equivalent Formula in the Excel add-in:

```
HISTORIAN.SINGLEVALUE(TagList, ValueType, Time, Details, Down, DiscreteFormat, UOMAddress)
```

Historian Client Workbook	Excel Add-in
Data Source	NA
TagRange	TagList
DisplayTagName	TagName-Details
DisplayDatetime, DisplayMilliseconds	Time
DisplayQuality,	NA
ReplacePoorQuality,	NA
DisplaySourceTag, DisplaySourceServer, OptionRange	NA

Tag details formula migration mapping details

Tag details formula migration mapping details

wwAnalogTagDetails1

```
wwAnalogTagDetails1 =wwAnalogTagDetails1( DataSource, TagRange, Description, EngUnit,
EURange, RawRange, Storage, SourceTag , SourceServer, OptionRange)
```

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace.

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
Description As Boolean, _	Description

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
EngUnit As Boolean, _	EngUnit
EURange As Boolean, _	EngUnitMax,EngUnitMin
RawRange As Boolean, _	NA
Storage As Boolean, _	NA
Source Tag	Source Tag
Source Server	NA
Optional OptionRange	NA

wwDiscreteTagDetails1

`wwDiscreteTagDetails1 =wwDiscreteTagDetails1(DataSource, TagRange, Description, Messages, Storage, SourceTag, SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace.

Historian Client Workbook	Excel Add-in
DataSource As String	NA
TagRange As Range	TagList
Description As Boolean	Description
Messages As Boolean	MessageOff,MessageOn
Storage As Boolean	NA
Optional OptionRange	NA
Source Tag	Source Tag
Source Server	NA

wwStringTagDetails1

`wwStringTagDetails1 =wwStringTagDetails1( DataSource, TagRange, Description, MaxLength, SourceTag, SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client Workbook	Excel Add-in
DataSource As String	NA
TagRange As Range	TagList
Description As Boolean	Description
MaxLength As Boolean	NA
Source Server	NA
Source Tag	Source Tag
Optional OptionRange	NA

wwReplicatedTagDetails

`wwReplicatedTagDetails =wwReplicatedTagDetails(DataSource, TagRange, Description, Storage, SourceTag, SourceServer, OptionRange)`

Equivalent formula in the Excel add-in: `HISTORIAN.INFO(TagList, Details, Down)`

Details: smartName, FQN, Description, EngUnit, EngUnitMax, EngUnitMin, InterpolationType, MessageOff, MessageOn, SourceTag, TagName, HierarchicalName, Alias, Location, and Namespace

Historian Client Workbook	Excel Add-in
DataSource As String, _	NA
TagRange As Range, _	TagList
Description As Boolean, _	Description
Storage As Boolean, _	NA
SourceTag As Boolean, _	Source Tag
SourceServer As Boolean, _	NA
Optional OptionRange	NA

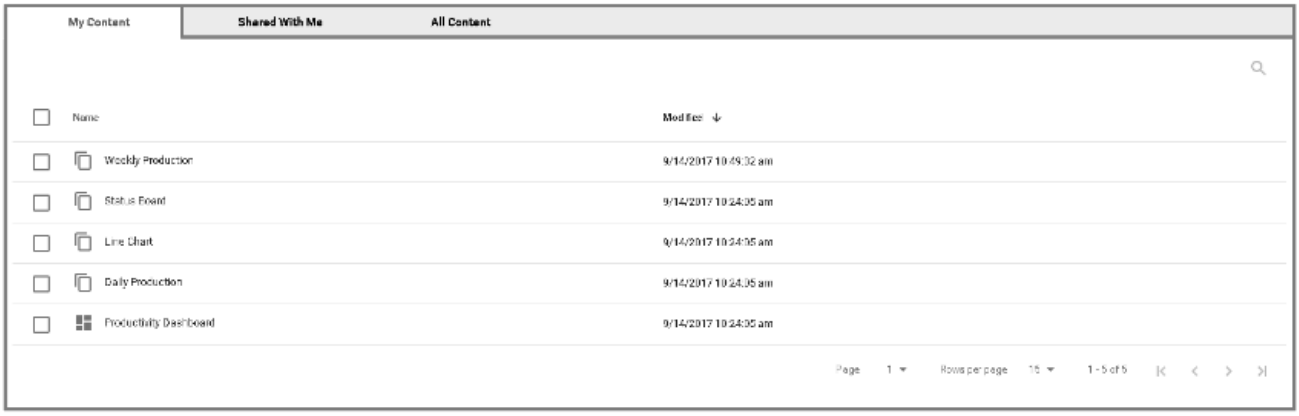
Manage content

You can work with all of the content you have saved and all of the content shared with you.

To view shared content

- Select ☰, and then select **Manage Content**.
  A list with three tabs displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



From here, you can:

- Select a tab to view content you've saved, content shared with you by others, and all content (both yours and shared with you).
- From the My Content tab, see which content you've shared:
  - means shared with the rest of your team.
  - means not shared, but saved for your private use.
- For any specific content, select to choose an action:
  - **Rename** to give the shared content a different name. (You must be the owner to do this.)
  - **Make a copy** to save a duplicate with a new name.
  - **Remove** to delete the item. (You must be the owner to do this.)

  Note: When you remove content, you also remove any unique keywords associated with that content.

Open saved content

Once you've saved content, you can reuse it later.

To open and reuse saved content

1. Select , then select **Manage Content**.
2. Select the name of the content you want to open.

   The content -- including all data tags and related settings (such as stacked chart mode) -- opens on the screen.

Copy saved content

To copy saved content

1. Select , and then select **Manage Content**.
2. Next to the the name of the content you want to use, select .
3. Select **Make a Copy**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

By default, the new name will be "Copy of" followed by the original name. You can select ⋮ and then select **Rename** to give it a new name.

Show or contact the content owner

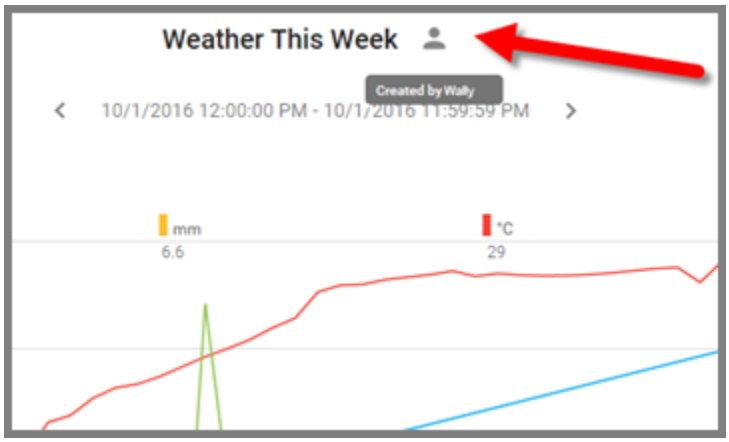
AVEVA Historian Client Web lets you know who owns a chart or dashboard, and whether it is shared.

To find out if the content is shared

- Find the icon next to the chart or dashboard title. It will look like one of these:
  means the content isn't shared.
  means the content is shared.

To find out who owns the content

- Next to the chart or dashboard title, hover over or select the icon to see the owner's name.



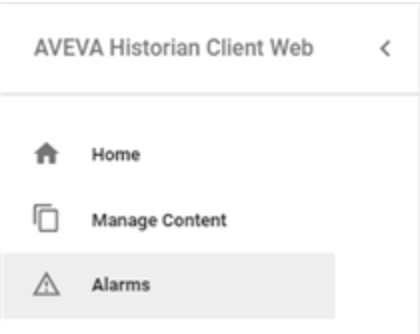
View alarms

The **Alarms** page allows you to view alarms for multiple asset levels, and filter alarms by a specified range of time. Alarms are configured in Historian.

Accessing the Alarm page

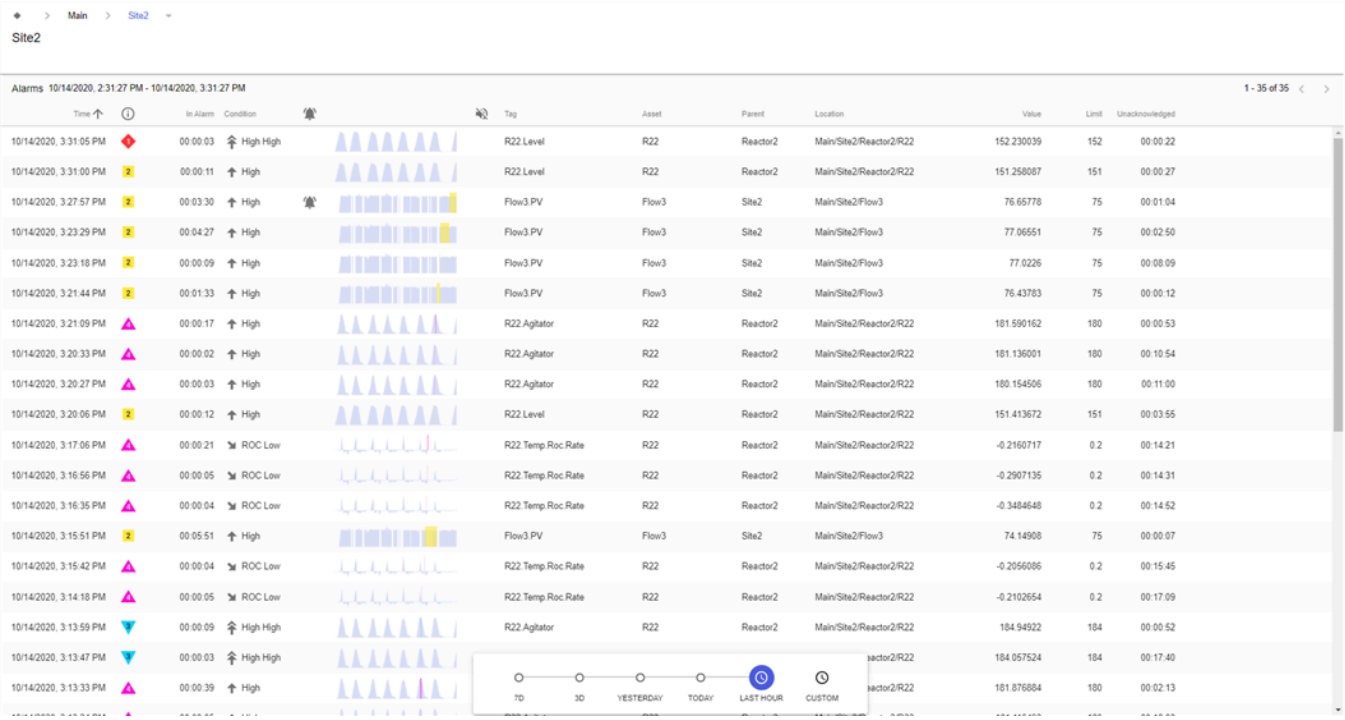
To access the **Alarm** page, select ☰, and then select **Alarms**.

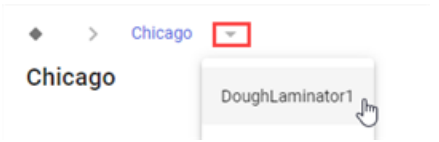
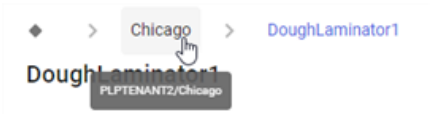
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Navigating the Alarm page

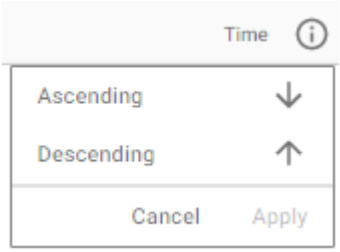
The alarm page contains a navigation bar showing the asset hierarchy, and a table which provides details on active alarms.



To filter alarms down to a specific child asset, use the down arrow on the navigation bar.	
You can filter through several layers of child assets. To return to a higher level, click the name of the higher level asset.	

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

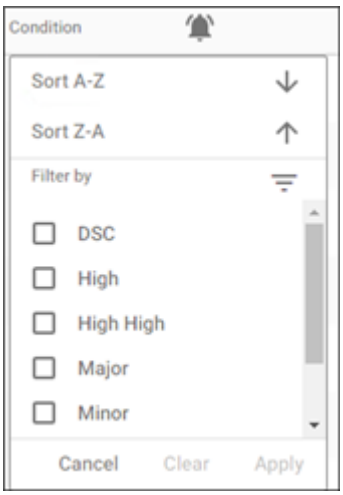
You can sort alarms by the Time, Alarm Severity, Condition, Tag, or Asset column. Select the heading of the desired column and select whether to sort in **Ascending** or **Descending** order.



You can filter alarms by the Alarm Severity, Condition, Tag, and Asset columns.

Select the values you want to filter by, then click **Apply**. The filter is applied, and an icon appears next to the column name.

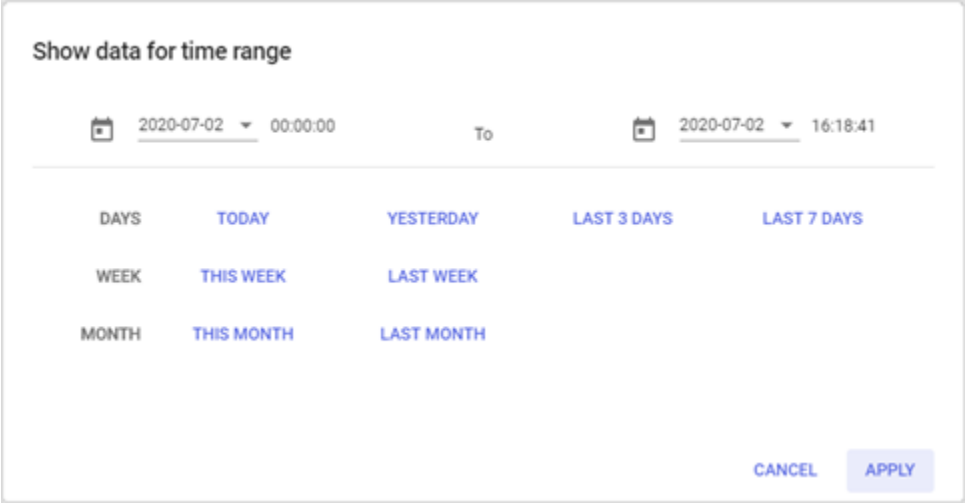
Select **Clear** to remove any filters applied to the column. Select **Cancel** to close the dialog without making changes.



To show alarm data for a specific time range

1. Locate the timeline at the bottom of the **Alarm** page.
2. Select from one of the predefined ranges (last 7 days, last 3 days, yesterday, today, and the last hour), or **Custom** to provide your own range.

   If selecting Custom, the **Show data for time range** window will appear.



3. Provide a start date & time and an end date & time, or select from one of the predefined options.
4. Click **Apply**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

API References

You can use these technical resources to customize an Insight solution:

- Historian Data REST API
  Allows you to upload and retrieve data from Insight.
- Insight Chart API
  Allows you to embed and customize the charts containing your Insight data in your own website or dashboard.

Historian Data REST API: Customize data retrieval

With the Historian Data REST API, you can make data retrieval requests.You can submit requests to the Historian Data REST API using a web browser or client-side applications such as:

- Microsoft Excel (2013, 2016, or Office 365)
- Business Intelligence (BI) systems, such as Tableau and Microsoft Power BI

Supported OData features

OData is an industry standard for querying and updating data from a variety of sources. The implementation of OData for the Historian Data REST API includes support for:

- JSON and atom formats.
- OData version 4 with version 2 of the Historian Data REST API.
- OData version 3 with version 1 of the Historian Data REST API (not recommended).
- Pagination. That is, if your request returns more than 5000 results, they will be returned in pages of up to 5000 records. Each page will include a link to retrieve the next page of records.
- A subset of the OData system query options.

For more information, see OData.org JSON Verbose Format specification.

Recommendation: For best results, when you want to view the data returned by the Historian Data REST API, use the JSONView extension for the Chrome browser.

Supported versions

AVEVA Historian Client Web supports versions 1 and 2 of the Historian Data REST API.

Version 2

This is the current and recommended version of the Historian Data REST API.

Version 2 of this REST AP is based on version 1 and includes further enhancements. Version 2 includes these differences from version 1:

- The `datetimeoffset` parameter is not supported as part of the `DateTime` syntax.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- While Raw and ProcessValue entities use `DateTime`, summary entities use `StartDateTime` and `EndDateTime`.
- For most version 2 queries, single quotes are not used for `DateTime`. For example:

  ```
  http://localhost:32569/Historian/v2/ProcessValues
  ?$filter=DateTime+gt+2017-07-13T00:00:00
  ```

  However, when querying events, single quotes are required for `DateTime`. For example:

  ```
  http://localhost:32569/Historian/v2/Events
  ?$filter=EventTime+gt+'2017-07-13T00:00:00'
  ```

  By contrast, version 1 queries do use single quotes for `DateTime`. For example:

  ```
  http://localhost:32569/Historian/v1/ProcessValues
  ?$filter=DateTime+gt+datetimeoffset'2017-07-13T00:00:00'
  ```
- TagProperty and Events entity types are now open type. That is, dynamic properties can be added to the response at runtime. This can be verified using `$metadata` endpoint URL:

  ```
  http://localhost:32569/Historian/v2/$metadata
  ```

  where there will be additional attribute, `OpenType="true"`, under `<EntityType>` section.
- The Tags resource returns all the properties (fixed and extended) for a tag. A tag's extended properties will be added to the response only if they exist. (The extended property name will not be listed for tags that do not have a given extended property.)
- The combined Summary resource is not supported in version 2. Use the individual AnalogSummary and StateSummary resources instead to retrieve the summary of a tag.
- Version 2 adds support for the OData `contains` function for applicable resources and properties.

Version 1

This is the original version of Historian Data REST API based on the OData v4 specification.

- Version 1 uses the `DateTime` format used in this example:

  ```
  http://localhost:32569/Historian/v1
  /AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'
  +and+StartDateTime+ge+datetimeoffset'2016-05-14T00:00:00.000-07:00'
  +and+EndDateTime+le+datetimeoffset'2016-05-16T00:00:00.000-07:00'
  ```

iHistory and Account Authentication

The iHistory web service requires users to be authenticated via a login process before they can retrieve data. The process for this differs between AVEVA Historian Client Web and AVEVA Insight:

- **AVEVA Historian Client Web**
  Uses Windows integrated security. A user must belong to the aaAdministrators, aaPowerUsers, or aaUsers Windows group. The iHistory web service uses Negotiate authentication, which is supported by most modern browsers and web service clients (such as Microsoft Excel).
- **AVEVA Insight (cloud-based)**
  Uses OpenID Connect and Basic authentication. Users must be invited to a specific account within AVEVA Insight and can then access all data published to that account. When using OpenID Connect, the iHistory web service uses "bearer token" authentication, per the OpenID Connect standard. As relatively new standard, some web service clients (for example, Microsoft Excel) do not have native support for it. However, some of

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

those same clients (for example, Microsoft Power BI) support Basic authentication.

When you query on-premises data via iHistory, you will be prompted for authentication to the Historian.

- If you have a valid Windows login on the historian, you can retrieve general information about the services and the kinds of data available.
- If you belong to the aaAdministrators, aaPowerUsers, or aaUsers Windows group, you can retrieve actual application data -- including events, process history, tag information, and so on.

Data retrieval

The Historian Data REST API allows you to retrieve data from your solution. Data retrieval with the Historian Data REST API is available through both Historian Client Web and Historian Client Web, so ensure that you refer to the correct syntax for your product from the examples below.

Forming retrieval requests

Use the following syntax for retrieval requests, depending on whether you are using HTTP or HTTPS:

```
http://<servername>:32569/Historian/<api_version>/<resource>?<query>
https://<servername>:32573/Historian/<api_version>/<resource>?<query>
```

Note: 32569 and 32573 are the default ports for HTTP and HTTPS. If your environment is configured to use different ports for either protocol, replace the ports used in this document with the appropriate ports for your environment.

The URL includes the following elements:

Syntax element	Explanation
<*servername*>:32569 or <*servername*>:32573	The servername and port number used by AVEVA Historian Client Web. If the server is the local computer, this could be "localhost:32569" for HTTP or "localhost:32573" for HTTPS.
Historian/<*api_version*>	The API version. For example, "v2".
<*resource*>	The Historian Data REST API retrieval resource. See the complete list.
?<*query*>	A query using OData parameters and operators

For example, this retrieval request gets analog summary data via an endpoint URL using basic authentication:

```
http://localhost:32569/Historian/v2/AnalogSummary
?$filter=FQN+eq+'Depot.Train09'+and+StartDateTime+ge+2017-06-09T09:00:00-07:00+and
+EndDateTime+ge+2017-06-09T10:00:00-07:00&Resolution=3600000
```

Supported OData parameters

These parameters are supported for system query via the Historian Data REST API. They are all case-sensitive.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameter	Description
$filter	Specifies an expression or function that must evaluate to true for a record to be returned in the collection. A $filter expression supports references to properties and literals. Literal values include: • Strings enclosed in single quotes • Numbers and Boolean values (true or false) Filtering for process value and summary data is case-sensitive. Filtering for other values, however, is case-insensitive. For example, if you filter property values based on a value of "true," values such as "TRUE," "True," and "true" could be returned. The case returned in the query results reflects the case of the stored value. Note that you can use $filter for FQN, but only for exact matches. Use the TagFilter parameter for partial FQN matches. $filter is typically used to specify a start and end date and time for the queried period. This is required for endpoints that return time-series data, but is recommended for most queries in order to reduce server load and make the number of returned results easier to work with.
$select	Specifies a subset of properties to return.
$skip	Specifies the number of records to skip from the beginning of the result set.
$skiptoken	Used to get the next record set that satisfies the query conditions. This parameter cannot be used in queries, but you will see it upon query execution.
$top	Specifies the maximum number of records to return. This subset is formed by selecting only the first *N* items of the set, where *N* is a positive integer specified by this query parameter.

Time-series extension parameters

The Historian Data REST API supports the following custom extension parameters in retrieval queries to better control your search results.

- **RetrievalMode** specifies how the resulting data is calculated for Raw and ProcessValues entities. Valid values are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

• Average	• Cyclic	• Integral	• Minimum
• BestFit	• Delta	• Interpolated	• Slope
• Counter	• Full	• Maximum	

- **Resolution** specifies the granularity of data returned for Raw, ProcessValues, and Summary entities.
- **TagFilter** is used to specify which tags to return data on. TagFilter must be used with $filter, which defines the range of results. TagFilter only works with the ProcessValues call. TagFilter supports the following operators:
  - startswith
  - endswith
  - contains

Supported operators

These logical operators are supported for queries:

Operator	Description
eq	Equal
ne	Not equal
gt	Greater than
ge	Greater than or equal
lt	Less than
le	Less than or equal
and	Logical and
or	Logical or
not	Logical negation

Note: Use "%20" to indicate a space. Use "%27" to indicate a single quote.
If you are using the JSONView viewer in the Chrome browser, you can use a plus sign (+) to indicate a space to make the URI string more readable.

If the expression includes multiple values for the criteria, you must specify each criteria separately using the "or" operator. For example:

```
...
((Priority+eq+100+or+Priority+eq+200+or+Priority+eq+500
+or+Priority+eq+700)+and+(filter …))
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax for combining expressions

- Use **and** to use multiple terms for the same parameter. For example:
  $filter=DateTime+ge+2023-08-31T00:00:00z+**and**+DateTime+le+2023-08-31T12:00:00z
- Use **&** to join separate parameters. For example:
  $filter=DateTime+ge+2023-08-31T00:00:00z**&**TagFilter=startswith(FQN, 'turbine')
- In a query, you can have only a single expression specifying a period of time combined with a single expression filtering specific tags. Combine expressions with the "and" operator. The expression itself can be complex, using any of the supported logical operators. Use parentheses ( ) to create precedence groups within an expression.

Retrieval resources

The Historian Data REST API exposes various resources through an endpoint URL that is specific to your Insight solution.

This API includes the following resources for retrieving data:

Process Data and Events Resources	Tag Property Resources	Resources for Version 1 Only
• ProcessValues • AnalogSummary • StateSummary • Events	• Tags • TagProperties • TagPropertyValues • TagGroups • TagSuggest • TagSearch • TagExtendedProperties	• Summary • Daily • Hourly • Minutely

Note: All property names are case-sensitive. Event storage preserves the case that you provide for any property value. For example, a property value of "TRUE" is different than "True" and "true."

ProcessValues

ProcessValues

Description	Retrieves a set of process value records (where each record includes value + time + quality, or VTQ) for the specified tags.
URL	`/ProcessValues`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Method	GET
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z • **EndDateTime=**[DateTimeOffset] The ending date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z
Optional Parameters	• **OPCQuality=**[Int32] The data quality as reported by the source. • **Value=**[Double] 0 or 1. • **Bounding=**[true] If the query specifies a start point, this parameter shows the first data point that occurs on or before the query start. If the query specifies an end point, this parameter shows the first data point that occurs on or after the query end. The actual date and time of the start/end point are shown, even if it's outside the bounds of the query. • **Text=**[String] For string tags, contain the value. For discrete tags, contains the message associated the value (0 or 1). • **TagFilter=**[String] Allows you to use a query string using OData filter query notation to filter results by tag attributes. Supported tag attributes include FQN, description. TagFilter can include up to 20 AND clauses, or up to 20 OR clauses. You cannot mix AND and OR in the same query. Operators should be lowercase, and searches are case-insensitive. *Valid Format:*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`Historian/v2/ProcessValues?TagFilter=startswith(Source,'MVDS') and TagType eq 'string'`
	Invalid Format:
	`Historian/v2/ProcessValues?TagFilter=startswith(Source,'MVDS') And TagType eq 'string'`
	• **Expression=**[String]
	For analog tags, you can perform UOM conversion on retrieved tag data by including a string expression in the following format:
	`UOM([tag FQN 1],[engineering unit name 1];[tag FQN 2],[engineering unit name 2];...)`
	You can provide one or more pairs of tag names and engineering unit names, and during retrieval the data will be converted to the specified units, if valid. If the unit conversion is not valid, the tag's base unit is used instead.
	Note: This feature requires special licensing. If a license is not available and the "Expression" parameter is used, no data is retrieved and an error message is displayed.

Success Response	`Code: 200` `Content: { fqn: plant12.pump6, datetime: 2016-09-03T18:44:09.352247Z}`

Error Response	`Code: 404 NOT FOUND`
	`Content: { error : "FQN doesn't exist" }`
	or
	`Code: 401 UNAUTHORIZED`
	`Content: { error : "You are unauthorized to make this request." }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 1**
	This query returns process values for a specific tag identified by its fully qualified name (*datasource.tagname*). Using a "$filter" clause, it specifies a tag named tank_level within the Baytown data source. The result is a list of values for the tank_level tag.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'`
	Scenario 2
	This query includes a start date time, end date time, and other query parameters.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'+and+DateTime+ge+2014-07-04T23:57:29Z+and+DateTime+le+2014-07-05T00:02:29Z&RetrievalMode=BestFit&Resolution=6500`

Sample Query	**Scenario 3**
	This query returns the current day of the month, and uses the Expression parameter to retrieve the value in hours.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'SysDateDay'&Expression=UOM([SysDateDay],[Hour])`
	Sample Output
	`{ "odata.metadata": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [ { "FQN": "SysDateDay", "DateTime": "2020-10-29T16:57:29.0128451Z", "OpcQuality": 192, "Value": 696.0, "Text": "696", "Unit": "Hour" } ] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 4**
	This query returns process values for a list of tags with fully qualified names (FQN) beginning with "sys", with values falling within a specific date range. If you don't know the fully qualified name of a specific tag, you can use the TagFilter parameter to get a list of possible matches.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=startswith(FQN,'sys')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [{ "FQN": "SysClassicDataRedirector", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "" }, { "FQN": "SysClientAccessPoint", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "" }, { "FQN": "SysConfigStatus", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units" }] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 5**
	This query returns process values for a list of tags whose FQN ends with "sec", with values falling within a specific date range.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=endswith(FQN,'sec')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [{ "FQN": "SysDataAcqOverallItemsPerSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units/s" }, { "FQN": "SysStatusRxEventsPerSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units/s" }] }`

Sample Query	**Scenario 6**
	This query returns process values for a list of tags whose FQN contains "timesec", with values falling within a specific date range.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=contains(FQN,'timesec')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [ { "FQN": "SysTimeSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "Second"`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`}`
	`]`
	`}`

Sample Query	**Scenario 7**
	This query returns process values for a list of tags whose description contains "timesec", with values falling within a specific date range.
	`https://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=Contains(Description,'timesec')`
	Sample Output
	`{`
	`"@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues",`
	`"value": [`
	`{`
	`"FQN": "SysTimeSec",`
	`"DateTime": "2019-01-15T03:57:29Z",`
	`"OpcQuality": 0,`
	`"Value": null,`
	`"Text": null,`
	`"Unit": "Second"`
	`}`
	`]`
	`}`
	Similar to the previous examples, you can search for tags with descriptions that start with, end with, or contain specific text.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AnalogSummary

AnalogSummary

Description	Retrieves analog statistics for the specified tags.
URL	`/AnalogSummary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z • **EndDateTime=**[DateTimeOffset] The ending date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z
Optional Parameters	• **RetrievalMode=**[string] Possible values are: Cyclic, Full. Default is Cyclic. • **Resolution=**[Int] In milliseconds. Any positive integer. • **SliceBy=**[Int|Discrete|String] Performs dynamic resolution/cycle computation by tags. Returns one Analog Summary value per tag per dynamic cycle with start and end date time. SliceBy can support up to 10 tags. For more information on using SliceBy, see Using SliceBy. • **SliceByValue=**[string] Specifies the filter criterion to get the summary values for SlicedBy, based on that filter value. • **OPCQuality=**[Int32]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Defines the OPC quality for the data. Normal OPC quality retrieval logic is applied if all the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned.
	• **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **First=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the first point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **FirstDateTime=**[DateTimeOffset] Time stamp associated with first value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle. • **Last=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the last point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **LastDateTime=**[DateTimeOffset] Time stamp associated with last value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle. • **Minimum=**[Double]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the minimum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **MinDateTime=**[DateTimeOffset] Time stamp associated with Min value. NULL if Min is NULL. • **Maximum=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the maximum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **MaxDateTime=**[DateTimeOffset] Time stamp associated with Max value. NULL if Max is NULL.
	• **Average=**[Double] Time weighted average value of retrieval cycle. This is calculated by using the individual summary averages. The calculation is "Sum(average * delta t) / Total time of average in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Average query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) • **StdDev=**[Double] Time weighted standard deviation value of the retrieval cycle. The value is calculated using time weighted sums (Integrals) and time weighted sums of squares (IntegralOfSquares) values, prorated for any partially contained storage cycles. For analog tags, similar StdDev values are produced for each cycle.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Integral=**[Double] Area under value curve of retrieval cycle. The calculation is "Sum(value * delta t) / Total time of integral in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Integral query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) For analog tags, similar Integral values are produced for each cycle. • **Count=**[Double] Number of values in a particular cycle. • **Expression=**[String] For analog tags, you can perform UOM conversion on retrieved tag data by including a string expression in the following format: `UOM([tag FQN 1],[engineering unit name 1];[tag FQN 2],[engineering unit name 2];...)` You can provide one or more pairs of tag names and engineering unit names, and during retrieval the data will be converted to the specified units, if valid. If the unit conversion is not valid, the tag's original units are used instead. **Note:** This feature requires special licensing. If a license is not available, the data is returned in the original units without conversion.
Sample Query	**Scenario 1** This query produces a list of tags that end with "level". In this case, the user doesn't know the fully qualified name (FQN) of a specific tag and wants a short list of possible matches. `https://online.wonderware.[DOMAINSUFFIX]/apis/historian/v2/AnalogSummary?TagFilter=endswith(FQN,'level').` See more TagFilter examples. **Scenario 2** This query produces a list of values with analog summary for the tank_level tag. Notice that this example uses a fully qualified name ("Baytown.tank_level"), which is a combination of a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	data source name ("Baytown") and a tagname ("tank_level"). `https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'` Because this query specifies no start or end time and no resolution, these defaults are used for the returned results: • EndTime defaults to DateTime.UtcNow • StartTime defaults to one hour before EndTime • Resolution defaults toTimespan • Count defaults to1 (number of returned rows)

	Scenario 3
	This query produces a list of tag values with analog summary data. The "$filter" clause narrows the results further by specifying these parameters:
	• Fully qualified name of the tag (FQN+eq+'Baytown.tank_level')
	• Start and end times
	• Result set returned at 1 hour intervals (Resolution=3600000). There are 3.6 million milliseconds in an hour.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'+and+StartDateTime+ge+2017-05-14T00:00:00.000Z+and+EndDateTime+le+2017-05-16T00:00:00.000Z&Resolution=3600000`
	Scenario 4
	This query specifies only StartTime and Resolution (600000 ms, or 10 minutes), but no EndTime.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level' and+StartDateTime+ge+2017-06-29T00:00:00Z&Resolution=600000`
	In this case, Insight assumes:
	• EndTime defaults to DateTime.UtcNow
	The number of rows returned (Count) depends on the StartTime and EndTime.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Scenario 5 This query specifies only EndTime, but no StartTime or Resolution. `https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'and+EndDateTime+le+2017-06-29T00:00:00Z'` In this case, Insight assumes: • StartTime defaults to one hour before EndTime • Resolution defaults to Timespan • Count defaults to 1 (number of returned rows)
	Scenario 6 This query specifies uses SliceBy. `https://online.wonderware.com/apis/Historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.R21.Level'+and+StartDateTime+ge+2019-03-05T00:00:00.000Z+and+EndDateTime+le+2019-03-05T12:00:00.000Z&SliceBy=Baytown.R21.Batch` In this case, Insight calculates an analog summary for the Baytown..R21.Level tag and batches the results per value for the Baytown.R21.Batch tag.

Using SliceBy

Using SliceBy

You can use a SliceBy parameter with an Analog Summary query to retrieve average values by batch within a specific time range.

For example, this is a query for an analog summary of the SysTimeMin tag, sliced by the SysTimeHour tag for a certain time range:

```
{
   "FQN":["SysTimeMin"],
   "StartDateTime":"2018-02-28T18:30:33.966Z",
   "EndDateTime":"2018-02-28T21:00:00Z",
   "SliceBy":["SysTimeHour"]
}
```

If the first or last batch is only partially complete, Historian will provide results as follows:

- If the first batch is only a partial set, the average is calculated for the amount of time represented by that batch. For example, suppose you have a tag that last changed at 7:50 a.m. and the query includes a start time of 8:00 a.m. The average includes values beginning at 7:50 a.m.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To include a partial first batch, use “>=” with your StartDateTime parameter. For example:

```
"StartDateTime">="2018-02-28T08:00:00.000Z"
```

To exclude a partial first batch, use “>” with StartDateTime. For example:

```
"StartDateTime">"2018-02-28T08:00:00.000Z"
```

- If the last batch is only a partial set, the wwResolution for the last batch will be NULL. For example, if the SliceBy query is for 8a.m. to noon, but the last batch is still in progress, the result will include “wwResolution=NULL”.

Batches can be organized using GroupBy. For example, you could use GroupBy to organize data by tag state.

StateSummary

StateSummary

Description	Retrieves state summary values for the specified tags.
URL	`/StateSummary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **RetrievalMode=**[string] Possible values are: Cyclic, Full. Default is Cyclic. • **Resolution=**[Int] In milliseconds. Any positive integer. • **OPCQuality=**[Int32]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **Text=**[string] Non-numeric state. • **Average=**[Double] Average time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. • **AverageContained=**[Double] Average time in this state among all occurrences of this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Minimum=**[Double] Minimum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **MinimumContained=**[Double] The minimum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Maximum=**[Double] Maximum time in this state among all occurrences

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **MaximumContained=**[Double] The maximum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Total=**[Double] Total time in this state during this retrieval cycle, including state occurrences that fall only partially within the period. • **TotalContained=**[Double] Total time in this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Percent=**[Double] Percent of the time during this retrieval cycle that the tag was in this state, including state occurrences that fall only partially within the period. • **PercentContained=**[Double] The precentage of the entire retrieval cycle time that the tag was in this state, excluding state occurrences that fall only partially within the period. This is a ratio between StateTimeTotalContained and StateTimeTotal expressed as a percentage in the range 0 to 100. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Count=**[Int64] The number of times the state occurred within the retrieval cycle, including states that only partially

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	occur in the cycle. • **CountContained=**[Int64] The number of times the state occurred fully contained within the retrieval cycle. States that only partially occur in the cycle are not counted. • **LastDateTime=**[DateTimeOffset] The time stamp for the last received value.

Sample Query	**Scenario 1**
	This query specifies no start or end time and no resolution.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03'`
	In this case, these defaults are used for the returned results:
	• EndTime defaults to DateTime.UtcNow
	• StartTime defaults to one hour before EndTime
	• Resolution defaults toTimespan
	• Count defaults to1 (number of returned rows)
	Scenario 2
	This query specifies only StartTime and Resolution (600000ms, or 10 minutes), but no EndTime.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03' and StartDateTime ge 2017-06-29T00:00:00Z&Resolution=600000`
	In this case, Insight assumes:
	• EndTime defaults to DateTime.UtcNow
	The number of rows returned (Count) depends on the StartTime and EndTime.
	Scenario 3
	This query specifies only EndTime, but no StartTime or Resolution.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03' and EndDateTime le 2017-06-29T00:00:00Z`
	In this case, Insight assumes:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• StartTime defaults to one hour before EndTime • Resolution defaults toTimespan • Count defaults to1 (number of returned rows)

Events

Events

Description	Retrieves information about events and alarms. **Note:** All property names are case-sensitive. Event storage preserves the case that you provide for any property value. For example, a property value of "TRUE" is different than "True" and "true."
URL	`/Events`
Method	`GET`
Optional Parameters	• **ID**=[GUID] Globally unique identifier for the event. • **EventTime**=[DateTime] UTC time stamp indicating when the event occurred. • **Type**=[String] Main categorization of the event. Examples of valid values: - Alarm.Acknowledge - Application.Write - Alarm.Clear - User.Write - Alarm.Set - User.Write.Secured - Alarm.Write - User.Write.Verified • **Priority**=[Int32] Value indicating the importance of the event. Values range from 1 to 999, with lower numbers indicating higher importance. • **Namespace** =[String] The namespace for this event tag. • **Severity**=[Int32] Categorization of the urgency of the event: 1 - Critical 2 - Major

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	3 - Minor 4 - Informational • **EventTimeUTCOffsetMins**=[Int32] For local time, the offset in minutes from UTC time. • **ReceivedTime**=[DateTime] UTC time stamp indicating when the event was received by the Historian server. • **IsAlarm**=[Bool] "true" or "false" indicating whether this event is an alarm. • **Comment**=[String] A comment providing more information about the event. **Note:** If an alarm comment from Application Server contains a backslash (for example, \n), then an extra backslash appears in the event result (for example, \\n) in the browser. • **InTouchType**=[String] InTouch Type value. Examples include: - ALM - RTN - ACK - SYS • **ValueString**=[String] The current value string. • **PreviousValueString**=[String] The previous value string.
Optional Alarm Parameters	• **Alarm_ID**=[String] ID of the original Alarm event. For "Alarm.Set" events, this will be the same as the event ID. • **Alarm_Class**=[String] InTouch alarm classification. Examples include "DSC" (discrete), "VALUE" (LoLo, Hi, etc.), "DEV" (deviation), and "ROC" (rate of change). • **Alarm_Type**=[String] InTouch alarm type. • **Alarm_InAlarm**=[Boolean] "true" or "false" indicating whether the Alarm is still in the active state. • **Alarm_Acknowledged**=[Boolean]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	"true" or "false" indicating whether the user has acknowledged this alarm. • **Alarm_Condition**=[String] The condition being alarmed. Examples include "Limit.Hi", "ROC.Lo", and "System" among others. • **Alarm_ValueString**=[String] Value logged for the variable related to the event. • **Alarm_LimitString**=[String] Limit being alarmed. • **Alarm_UnAckDuration**=[Int32] The duration, in milliseconds, for which the alarm went un-acknowledged. • **Alarm_Duration**=[Int32] The duration, in milliseconds, for which the alarm was active. • **Alarm_IsSilenced**=[Boolean] "true" or "false" indicating whether the alarm was silenced. • **Alarm_IsShelved**=[Boolean] "true" or "false" indicating whether the alarm was shelved. • **Alarm_ShelveStartTimeUTC**=[DateTime] Scheduled start of the shelve time if the alarm has been shelved. • **Alarm_ShelveEndTimeUTC**=[DateTime] Scheduled end of the shelve time if the alarm has been shelved. • **Alarm_ShelveReason**=[String] The reason the alarm was shelved. • **Alarm_ShelveUserLogin**=[String] The login ID for the person who shelved the alarm. • **Alarm_ShelveNode**=[String] The node of the shelved alarm.
Optional Provider Parameters	• **Provider_NodeName**=[String] Name of the node that generated the event. • **Provider_System**=[String] Software system that generated the event. Examples include "Application Server", "InTouch", and "InBatch". • **Provider_ApplicationName**=[String]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Application that generated the event. For Application Server, this will be the galaxy name. For InTouch, it will be the InTouch application name. • **Provider_SystemVersion**=[String] Software version (for example, 4566.1210.5811.1) for the component identified by Provider_ApplicationName. • **Provider_InstanceNames**=[String] Provider-specific string that uniquely identifies the instance an application on a given node.
Optional Source Parameters	• **Source_Name**=[String] The name of the data source for this event tag. • **Source_ProcessVariable**=[String] Process variable to which the event is related. • **Source_ProcessVariable_Units**=[String] Engineering units used for the process variable. • **Source_ConditionVariable**=[String] Condition variable related to the event. For example, if "TIC101" has a field attribute "PV" and this is a "Hi" alarm, this value will be "TIC101.PV.Hi". • **Source_Object**=[String] Non-hierarchical name for the object to which the event is related, for example, "TIC101". • **Source_HierarchicalObject**=[String] Hierarchical name for the source object. For example, "Reactor_001.TIC". • **Source_Area**=[String] Non-hierarchical Area name. For example, "Bottling_Zone". • **Source_HierarchicalArea**=[String] Hierarchical Area name. For example, "Plant_001.Building_002.Mixing". • **Source_Engine**=[String] Non-hierarchical engine name. For example, "AppEngine_001". • **Source_Platform**=[String] Non-hierarchical platform name. For example "WinPlatform_001".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Optional User Parameters	• **User_Account**=[String] This is the login name for the operator for the given application. • **User_Name**=[String] Name of the user. • **User_NodeName**=[String] Computer name from which a user action was executed. • **User_Email**=[String] The user's email address. • **User_Phone**=[String] The user's phone number. • **User_InstanceName**=[String] An instance name for the user. • **User_Agent**=[String] Application name that the user was running when the event was generated.
Optional Verifier Parameters	• **Verifier_Account**=[String] This is the login name for theverifier. • **Verifier_Name**=[String] Name of the verifier. • **Verifier_NodeName**=[String] Computer name from which a verifier action was executed.

Sample Queries	**Sample 1**
	This query lists all events that occur from 2017-07-13 onward.
	`https://localhost:32573/Historian/v2/Events ?$filter=EventTime+gt+'2017-07-13T00:00:00'`
	Sample 2
	This query lists all alarm clearing events.
	`http://localhost:32569/Historian/v2/Events ?$filter=type+eq+'Alarm.Clear'`
	Sample 3
	This query lists events from 2017-05-14 to 2017-05-15 at 9 am. It skips the first 1000 events, then lists the top 2000 events after that.
	`http://localhost:32569/Historian/v2/ Events/?$filter=EventTime gt 2017-05-14T09:00:00.000Z and EventTime lt 2017-05-15T09:00:00.000Z&$skip1000&$top=2000`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Output	{ @odata.context: "http://localhost:32569/Historian/v2/Events/$metadata#Events", value: [{ id: "ee9b042c-181e-4872-b4e8-cb9575c1f4f9", eventtime: "2017-07-13T00:00:01.047Z", type: "Alarm.Clear", receivedtime: "2017-07-13T00:00:01.2553666Z", source_name: "AaEvents", namespace: "AaEvents", alarm_acknowledged: false, alarm_class: "VALUE", alarm_condition: "Limit.LoLo", alarm_durationms: 5006, alarm_id: "f1e21e43-352e-d3db-7552-6e77f193d02d", alarm_inalarm: false, alarm_isshelved: false, alarm_issilenced: false, alarm_limitstring: "10.0", alarm_originationtime: "2017-07-12T23:59:56.041Z", alarm_state: "UNACK_RTN", alarm_tagtype: "S", alarm_type: "LoLo", comment: "Severity 1", eventtimeutcoffsetmins: -420, intouchtype: "LoLo", isalarm: true, priority: 1, provider_applicationname: "AlarmsandEvents", provider_nodename: "INSIGHTCHARTAPP", provider_system: "Application Server", provider_systemversion: "4966.1210.14578.2", severity: 1, source_area: "Plant_Area", source_conditionvariable: "Drum_Conveyor.HorizontalMovement.LoLo", source_hierarchicalarea: "Enterprise.Site.Plant.Plant_Area", source_hierarchicalobject: "Drum_Conveyor", source_object: "Drum_Conveyor", source_processvariable: "Drum_Conveyor.HorizontalMovement", valuestring: "30.0" },

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tags

Tags

Description	Using the GET verb, retrieves tag metadata. For version 2, this also includes tag extended properties. Using the POSTverb, manages multiple tags at once. Using the DELETE verb, deletes a specified tag.
URL	`/Tags`

Methods	`GET`
	`POST`
	`DELETE`

Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **Source=**[string] The data source. • **Description=**[string] The description of the tag. • **EngUnit=**[string] The engineering units used for the tag's recorded values. **EngUnitMax=**[Double] The maximum value of the tag, measured in engineering units. • **EngUnitMin=**[Double] The minimum value of the tag, measured in engineering units. The minimum value of the tag, measured in engineering units. • **InterpolationType=**[string] The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. • **IntegralDivisor=**[Double] The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only. • **RolloverValue=**[Double] The rollover value for the tag. • **MessageOff=**[string] The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. • **MessageOn=**[string] The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. • **TagName=**[string] The unique name of the tag within the AVEVA Historian system. • **TagType=**[string] The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the AVEVA Historian TagRef table. • **Raw** Identifies related the raw acquired value. • **Minutely**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Identifies related minutely summary. • **Hourly** Identifies related hourly summary. • **Daily** Identifies related daily summary.

Get tags

Sample Query	**Scenario 1**
	This query lists tag metadata for all tags in all data sources for your Insight solution.
	`http://localhost:32569/Historian/v2/Tags`
	Scenario 2
	This query lists metadata for tags in a particular data source. The data source in this example is named Baytown.
	`http://localhost:32569/Historian/v2/Tags ?$filter=Source+eq+'Baytown'`

Sample Output	{ @odata.context: "http://localhost:32569/ Historian/v2/$metadata#Tags", value: [{ FQN: "14th aug app.Auto", Source: "14th aug app", Description: "Automatic mode", EngUnit: "", EngUnitMax: 0, EngUnitMin: 0, InterpolationType: "None", MessageOff: "Manu", MessageOn: "Auto", TagName: "Auto", TagType: "Discrete", Alias: "nareshtag-discrete", Location: "/Application Tag 2", Raw@odata.navigationLink: "http://localhost:32569/Historian/v2/Tags ('14th%252baug%252bapp.Auto')/Raw", Minutely@odata.navigationLink: "http://localhost:32569/Historian/v2/Tags ('14th%252baug%252bapp.Auto')/Minutely", Hourly@odata.navigationLink: "http://localhost:32569/Historian/v2 /Tags('14th%252baug%252bapp.Auto')/Hourly", Daily@odata.navigationLink: "http://localhost:32569/Historian/v2/Tags ('14th%252baug%252bapp.Auto')/Daily"

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`}`
	`],`
	`@odata.nextLink: "http://localhost:32569/Historian/v2/Tags?$skiptoken=koen66nr4nGCReC02FqdxO4Y62C7zBVsv3uf7mO8XMQqxZveqw8+v2/bVCsFdevT"`

Delete a tag

Sample Query	This query deletes a tag named Weather.Brisbane.Cloudiness.
	`DELETE /Historian/v2/Tags('Weather.Brisbane.Cloudiness') HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Content-Type: application/json`
	`{`
	`"delete":`
	`{`
	`"FQN":["datasourcename.tagname"]`
	`}`
	`}`

Manage multiple tags

Sample Query	**Scenario 1: Bulk query**
	This query retrieves multiple tags.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchqaruntime.cloudapp.net:8080`
	`Authorization: Basic …….`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"query":`
	`{`
	`"FQN":['Weather.Brisbane.Cloudiness','Weather.Auckland.Cloudiness'],`
	`"select":"FQN,TagName"`
	`}`
	`}`
	Scenario 2: Bulk edit
	This makes updates to multiple tags. You can edit up to 100 tags at a time, with maximum of 50 properties per tag.
	Note: Historian Data REST API extended properties map to the AVEVA Historian SDK tag extended properties API.
	This makes updates to a tag's standard and extended

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	properties.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Authorization: Basic …`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"properties":`
	`[`
	`{`
	`"FQN": "18Jul_kc.test1",`
	`"PropertyName": "Alias",`
	`"Text": "SysTimeSec",`
	`"Type": "String",`
	`"Searchable":false`
	`}`
	Scenario 3: Bulk delete
	This deletes multiple tags.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"delete":`
	`{`
	`"FQN":["datasourcename.tagname1",`
	`"datasourcename.tagname2"]`
	`}`
	`}`

Sample Output	The output from a bulk edit includes results per tag. If any of a tag's properties is invalid, the tag fails and results in an error message.
	`{`
	`"value":`
	`[`
	`{`
	`"FQN": "System.Tag1",`
	`"$StatusCode": 200`
	`},`
	`{`
	`"FQN": "System.Tag2",`
	`"$StatusCode":500,`
	`"$ErrorMessage":"Alias property contains illegal characters"`
	`}`
	`]`
	`}`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagProperties

TagProperties

Description	Lists all properties used in the system. Tag properties include both base and extended properties of a tag. Examples of TagProperties are DataSourceName and Unit.
URL	`/TagProperties`
Method	`GET`
Required Parameters	• **Name=**[string] The unique name of the tag property.
Optional Parameters	• **Values** Valid values for this property.
Sample Query	This query lists all properties and extended properties for your Insight solution: `http://localhost:32569/Historian/v2/TagProperties`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagProperties",`
	`value: [`
	`{`
	`Name: "DataSourceName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "Namespace",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "TagName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "FullyQualifiedName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`...`

TagPropertyValues

TagPropertyValues

Description	Retrieves values for one or more tag properties. Tag properties include base and extended properties of Tag. Examples of TagProperties are DataSourceName and Unit.
URL	`/TagpropertyValues`
Method	`GET`
Required Parameters	• **Name=**[string] Unique name of the tag property value. • **Value=**[string]
Optional Parameters	• **Tags** Identifies related Tags.
Sample Query	This query lists the property name ("Source") and value (that is, the data source name) for all data sources: `http://localhost:32569/Historian/v2/TagPropertyValues`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagPropertyValues",`
	`value: [`
	`{`
	`Name: "Source",`
	`Value: "Meter",`
	`Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Meter')/Tags"`
	`},`
	`{`
	`Name: "Source",`
	`Value: "Weather",`
	`Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagPropertyValues`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`(Name='Source',Value='Weather')`
	`/Tags"`
	`}`
	`]`
	`}`

TagGroups

TagGroups

Description	Retrieves information about data sources. Each data source contains a group of tags.
URL	`/TagGroups`
Method	`GET`
Required Parameters	• **GroupID=**[string] Globally unique identifier for the tag group.
Optional Parameters	• **GroupName=**[string] The name of the data source. • **TypeID=**[string] The type of data source. • **ParentID=**[string] • **Scope=**[string] Used for tag-level security, this defines a location within the data source. • **Tags** Identifies related Tags. • **Groups** Identifies related TagGroups.
Sample Query	This query lists the data sources (labeled "TagGroups" in this API) for your Insight solution identified by solution ID. In this example, the solution ID used is "ik97r5". `http://localhost:32569/Historian/v2/TagGroups`
Sample Output	`{` `@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagPropertyValues",`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	value: [
	{
	Name: "Source",
	Value: "Meter",
	Tags@odata.navigationLink:
	"http://localhost:32569/Historian/v2/
	TagPropertyValues(Name='Source',Value='Meter
	')
	/Tags"
	},
	{
	GroupID: "1cd29ff2-f7f3-428b-
	a593-55c917136a39",
	GroupName: "Weather",
	Type: "1000000",
	ParentID: "0",
	Scope: "Public",
	Tags@odata.navigationLink:
	"http://localhost:32569/Historian/v2/
	TagGroups
	('1cd29ff2-f7f3-428b-
	a593-55c917136a39')/Tags",
	Groups@odata.navigationLink:
	"http://localhost:32569/Historian/v2/
	TagGroups
	('1cd29ff2-f7f3-428b-
	a593-55c917136a39')/Groups"
	}
	]
	}

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagSuggest

TagSuggest

Description	Retrieves content or tags based on search criteria from Insight. It returns the search results based on tag name, content name, keywords, description, and so on. It also returns the summary of the search result -- for example, total matching search results for the given search text. You can use the these query options: • q -- (query) Limits the search to only specified fields instead of searching all fields. • Type -- Limits the search to only a certain type of data (Tag or SavedContent).
URL	`/TagSuggest`
Method	`GET`
Required Parameters	• **Value=**[string] The suggested value.
Optional Parameters	• **FieldName=**[string] The related field name. • **Count=**[Double] The hit count for this query string. • **SearchRanking=**[Double] The search ranking for the tag. • **DisplayText=**[string] The associated display text. • **Search** • **Tags** Identifies related Tags. • **ExpandSuggest**
Sample Query	**Scenario 1** This query returns values from a search of all fields (default). `http://localhost:32569/Historian/v2/`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`TagSuggest`
	Scenario 2
	This query uses the "q" query option to limit the search to only the "location" field.
	`http://localhost:32569/Historian/v2/TagSuggest ?q=&searchfields=location`
	Scenario 3
	This query uses the "Type" option to limit the search to only a certain type of data (Tag or SavedContent).
	`http://localhost:32569/Historian/v2/TagSuggest?q=d&Type=Tag`

TagSearch

TagSearch

Description	Provides tagname results based on provided search parameters. The data returned by this entity is dependent on the historian's implementation of search functionality. Some historians will return empty result sets. It finds all the matching results for a given query string. It is basically used to filter the results based on the suggestion results. It searches for the matching record only on the fields provided in the key name as part of the query parameter. The results contain basic tag metadata information; such as FQN, tagname, source, and search ranking. You can use these query options: • q - Specifies the query string (for example, the value typed by the user into the search box). • kn - Specifies the key name (field name) to which the search will be applied. • kv - Specifies the key value which will be used for the search.
URL	`/TagSearch`
Method	`GET`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **Source=**[string] The data source. • **TagName=**[string] The name of the tag. • **DisplayText=**[string] The displayed text for the tag. • **SearchRanking=**[Double] The search ranking for this tag. • **Tag** Returns a URL to retrieve a list of Tag entity values for the tags that match.
Sample Query	This example uses the "q", "kn", and "kv" query options. `http://localhost:32569/Historian/v2/TagSearch ?q=r&kn=source&kv=atron`

TagExtendedProperties

TagExtendedProperties

Note: This resource has been deprecated. See sample queries under Tags for current information on retrieving properties associated with a tag.

Description	Retrieves both standard and extended properties for a tag.
URL	`/TagExtendedProperties`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **PropertyName=**[string] The name of the property.

Optional Parameters	• **Value=**[Double]
	The value of the property.
	• **Text=**[string]
	The associated text for the property.
	• **Type=**[string]
	Can be one of these:
	  • [Edm.]String
	  • [Edm.]Int16, Int32, Int64
	  • [Edm.]Double
	  • [Edm.]DateTimeOffset
	  • [Edm.]Guid
	  • [Edm.]Boolean
	  • [Edm.]Geography types
	  • [Edm.]Geometry types

Summary (v1 only)

Summary (v1 only)

Description	Retrieves summary data for a user-defined interval. The additional query option "Resolution" is used to specify the interval to be used for this entity. The earliest and latest requested values of StartDateTime and/or EndDateTime are used for determining the span of time covered by the summary intervals returned.
URL	`/Summary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog** An embedded collection that maps to AnalogSummary. • **Split** An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained** An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Daily (v1 only)

Daily (v1 only)

Description	Retrieves daily (24-hour resolution) summary values for the tags specified. The default for Daily is to report summary values for the last week.
URL	`/Daily`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **TimeZone=**[string] The timezone used for the timeframe. See "Accepted values for TimeZone" below. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog=**[analog statistics] An embedded collection that maps to AnalogSummary. • **Split=**[collection of state statistics] An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained=**[collection of state statistics] An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query. • **Minute** • **Hourly**

Accepted values for TimeZone

UTC offset in minutes	TimeZone value
0	UTC
0	Morocco Standard Time
0	GMT Standard Time
0	Greenwich Standard Time
60	W. Europe Standard Time
60	Central Europe Standard Time
60	Romance Standard Time
60	Central European Standard Time
60	W. Central Africa Standard Time
60	Namibia Standard Time
120	Jordan Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

120	GTB Standard Time
120	Middle East Standard Time
120	Egypt Standard Time
120	E. Europe Standard Time
120	Syria Standard Time
120	West Bank Standard Time
120	South Africa Standard Time
120	FLE Standard Time
120	Israel Standard Time
120	Kaliningrad Standard Time
120	Libya Standard Time
180	Arabic Standard Time
180	Turkey Standard Time
180	Arab Standard Time
180	Belarus Standard Time
180	Russian Standard Time
180	E. Africa Standard Time
210	Iran Standard Time
240	Arabian Standard Time
240	Astrakhan Standard Time
240	Azerbaijan Standard Time
240	Russia Time Zone 3
240	Mauritius Standard Time
240	Georgian Standard Time
240	Caucasus Standard Time
270	Afghanistan Standard Time
300	West Asia Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

300	Ekaterinburg Standard Time
300	Pakistan Standard Time
330	India Standard Time
330	Sri Lanka Standard Time
345	Nepal Standard Time
360	Central Asia Standard Time
360	Bangladesh Standard Time
360	Omsk Standard Time
390	Myanmar Standard Time
420	SE Asia Standard Time
420	Altai Standard Time
420	W. Mongolia Standard Time
420	North Asia Standard Time
420	N. Central Asia Standard Time
420	Tomsk Standard Time
480	China Standard Time
480	North Asia East Standard Time
480	Singapore Standard Time
480	W. Australia Standard Time
480	Taipei Standard Time
480	Ulaanbaatar Standard Time
510	North Korea Standard Time
525	Aus Central W. Standard Time
540	Transbaikal Standard Time
540	Tokyo Standard Time
540	Korea Standard Time
540	Yakutsk Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

570	Cen. Australia Standard Time
570	AUS Central Standard Time
600	E. Australia Standard Time
600	AUS Eastern Standard Time
600	West Pacific Standard Time
600	Tasmania Standard Time
600	Vladivostok Standard Time
630	Lord Howe Standard Time
660	Bougainville Standard Time
660	Russia Time Zone 10
660	Magadan Standard Time
660	Norfolk Standard Time
660	Sakhalin Standard Time
660	Central Pacific Standard Time
720	Russia Time Zone 11
720	New Zealand Standard Time
720	UTC+12
720	Fiji Standard Time
720	Kamchatka Standard Time
765	Chatham Islands Standard Time
780	Tonga Standard Time
780	Samoa Standard Time
840	Line Islands Standard Time
-60	Azores Standard Time
-60	Cape Verde Standard Time
-120	UTC-02
-120	Mid-Atlantic Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

-180	Tocantins Standard Time
-180	E. South America Standard Time
-180	SA Eastern Standard Time
-180	Argentina Standard Time
-180	Greenland Standard Time
-180	Montevideo Standard Time
-180	Saint Pierre Standard Time
-180	Bahia Standard Time
-210	Newfoundland Standard Time
-240	Paraguay Standard Time
-240	Atlantic Standard Time
-240	Venezuela Standard Time
-240	Central Brazilian Standard Time
-240	SA Western Standard Time
-240	Pacific SA Standard Time
-240	Turks And Caicos Standard Time
-300	SA Pacific Standard Time
-300	Eastern Standard Time (Mexico)
-300	Eastern Standard Time
-300	Haiti Standard Time
-300	Cuba Standard Time
-300	US Eastern Standard Time
-360	Central America Standard Time
-360	Central Standard Time
-360	Easter Island Standard Time
-360	Central Standard Time (Mexico)
-360	Canada Central Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

-420	US Mountain Standard Time
-420	Mountain Standard Time (Mexico)
-420	Mountain Standard Time
-480	Pacific Standard Time (Mexico)
-480	UTC-08
-480	Pacific Standard Time
-540	Alaskan Standard Time
-540	UTC-09
-570	Marquesas Standard Time
-600	Aleutian Standard Time
-600	Hawaiian Standard Time
-660	UTC-11
-720	Dateline Standard Time

Hourly (v1 only)

Hourly (v1 only)

Description	Retrieves hourly summary values for the tags specified. The default for Hourly is to report summary values for the last 24 hours.
URL	`/Hourly`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **TimeZone=**[string] The timezone used for the timeframe. See "Accepted values for TimeZone" below. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog=**[analog statistics] An embedded collection that maps to AnalogSummary. • **Split=**[collection of state statistics] An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained=**[collection of state statistics] An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query. • **Minute**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Minutely (v1 only)

Minutely (v1 only)

Description	Retrieves summary values with a 1-minute resolution for the tags specified. Default duration for minutely summaries is the last 1 hour. The default for Minutely is to report summary values for the last hour.
URL	`/Minutely`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **TimeZone=**[string] The timezone used for the timeframe. See "Accepted values for TimeZone" below. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog=**[analog statistics] An embedded collection that maps to AnalogSummary. • **Split=**[collection of state statistics] An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained=**[collection of state statistics] An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query.

SystemParameters (on-premises only)

SystemParameters (on-premises only)

Description	Retrieves names and values for system parameters. (On-premises AVEVA Historian Client Web only)
URL	`/SystemParameters`
Method	`GET`
Required Parameters	• **Name=**[string] The unique name for the system parameter.
Optional Parameters	• **Value=**[string] The value of this system parameter. • **Tags** Identifies related tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval examples

You can send retrieval requests to the Historian Data REST API from any modern web browser and from OData client-side tools. Examples of such tools include Microsoft Excel (2013, 2016, or Office 365), and Business Intelligence (BI) systems, such as Tableau and Microsoft Power BI.

Select below to see query examples using specific tools:

- Browser query examples
- Postman query examples
- Excel query examples
- PowerBI query example

Retrieve data via browser query

Retrieve data via browser query

You can simply type an OData HTTP request in the address bar of any modern browser to query Historian Data API.

- **List root entities**

  This example lists the entities you can explore and submit additional queries against.

  ```
  http://localhost:32569/Historian/v2
  ```

  For more information, see the Data Dictionary for Historian Data API.

- **List data sources (tag groups)**

  This shows the data sources (labeled "TagGroups" in this API) for your Insight solution. If you have accounts for more than one solution, this shows the data sources for your original Insight solution.

  ```
  http://localhost:32569/Historian/v2/TagGroups
  ```

- **List tag metadata for all tags**

  This shows metadata for all tags in all data sources for your original Insight solution.

  ```
  http://localhost:32569/Historian/v1/historian/v2/Tags
  ```

- **List tag metadata for a data source**

  This shows metadata for tags in a particular data source. The data source in this example is named Baytown.

  ```
  http://localhost:32569/Historian/v2/Tags?$filter=Source+eq+'Baytown'
  ```

- **List tag values for a data source**

  This example expands on the last one. It uses a "$filter" clause that specifies a particular tag named tank_level within the Baytown data source. The result is a list of values for the tank_level tag.

  ```
  http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'
  ```

- **List tag values with analog summary data for a data source**

  This example result is a list of values with analog summary for the tank_level tag.

  ```
  http://localhost:32569/Historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'
  ```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Notice that this example uses a fully qualified name ("Baytown.tank_level"), which is a combination of a data source name ("Baytown") and a tagname ("tank_level").

Retrieve data using Postman

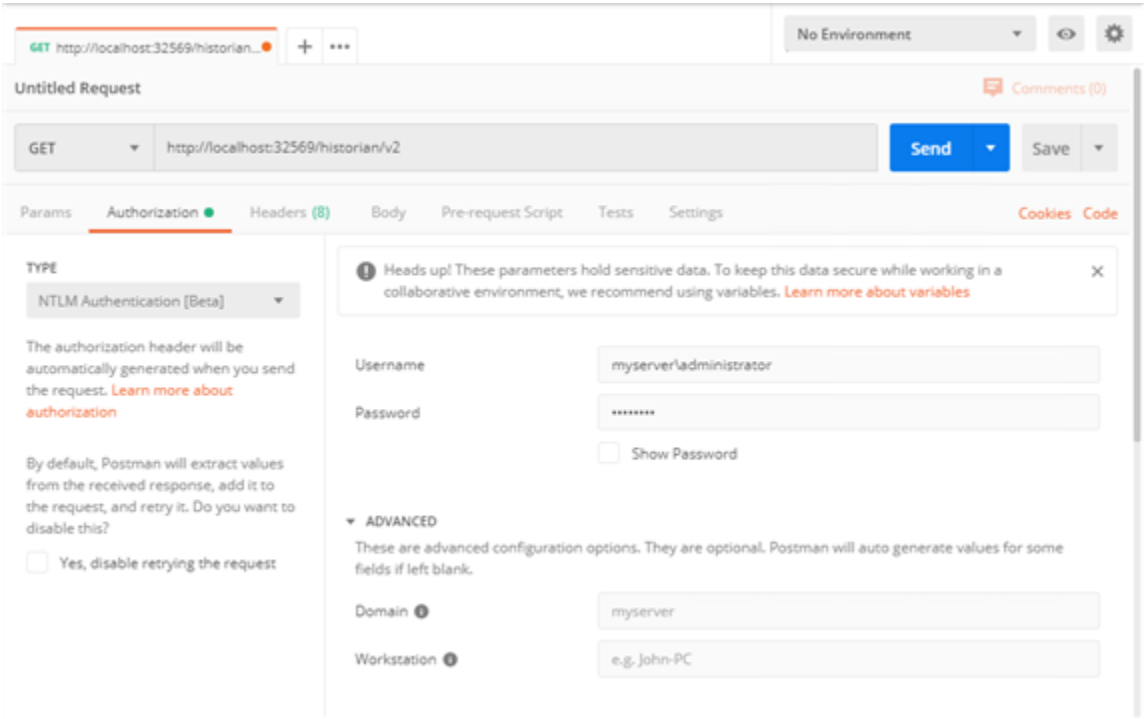
Retrieve data using Postman

A great way to explore and learn about the AVEVA Insight APIs is by using a simple client, such as Postman. Postman is a free developer tool allows you to query any API. You can download Postman from https://postman.com/.

To query the AVEVA Insight APIs using Postman

1. Next to the GET button, type this address:

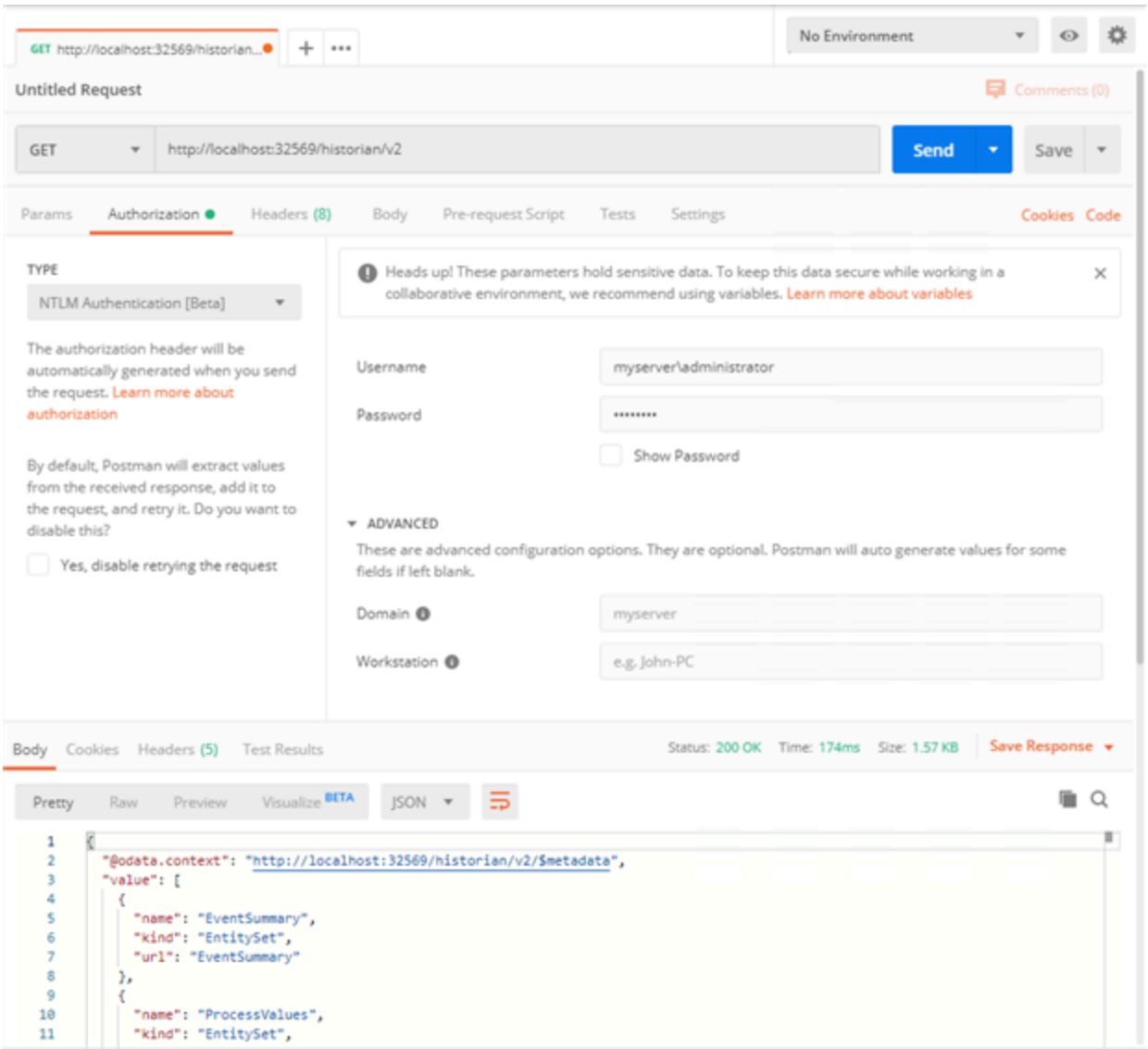
```
http://localhost:32569/historian/v2
```



2. From the **Type** dropdown list, select **NTLM Authentication [Beta]** to specify the authentication to use.
3. Enter the username and password for a user account with sufficient privileges to access the API.
4. In the upper-right of the screen, select **Send**.

   The result is a catalog of data entities from AVEVA Insight.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Excel query examples

Excel query examples

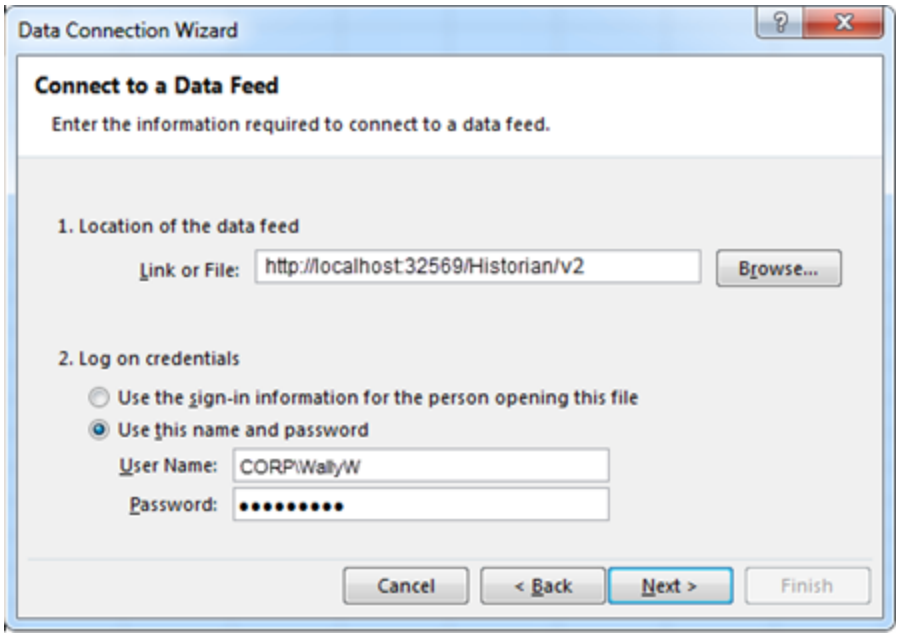
To manually import Insight data to Excel

1. Open Excel 2013 or later.
2. From **Data**,choose **Get External Data**, choose **From Other Sources**, and then choose **From OData Data Feed**.
3. In the **Link or File** box, specify a URL for your query, formatted as an atom feed (that is, end with "$format=atom'").

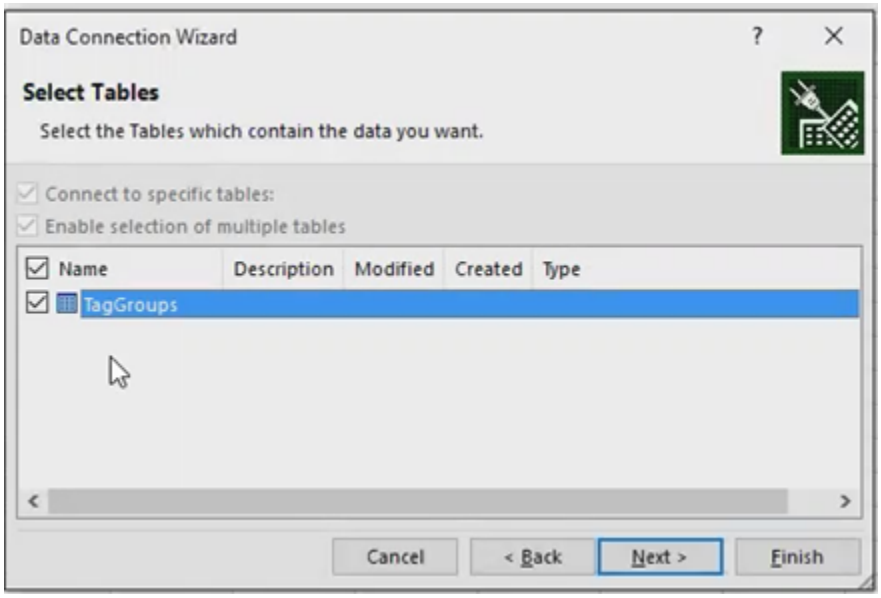
   Here is a query example:

   ```
   http://localhost:32569/Historian/v2/TagGroups?$format=atom
   ```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

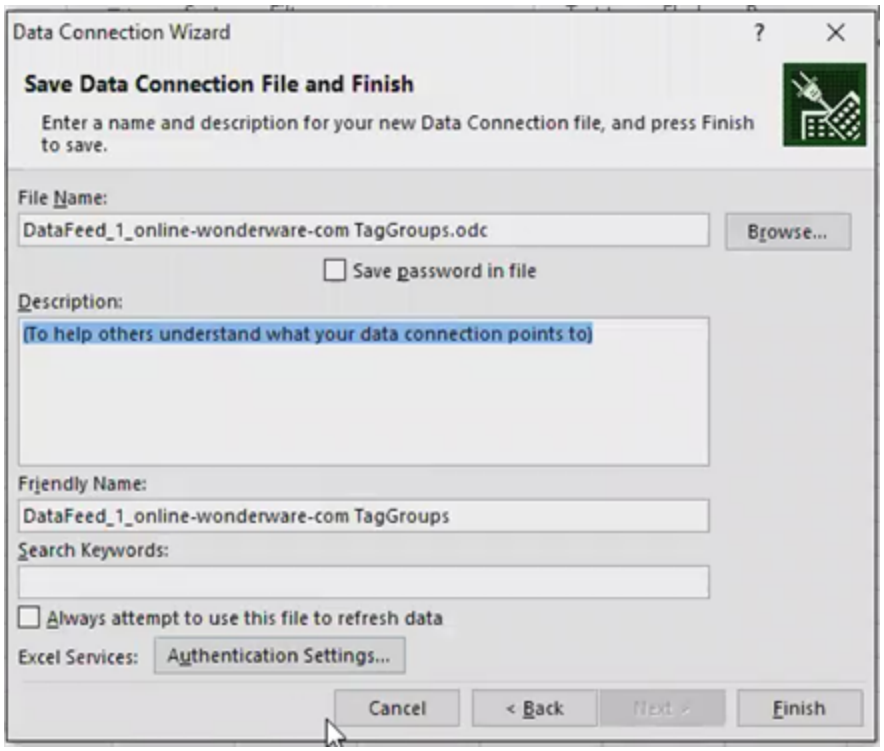


4. Click **Use this name and password**, and specify your Insight user name and password. Click **Next**.
5. In the **Select Tables** box, mark the element to include. Click **Next**.

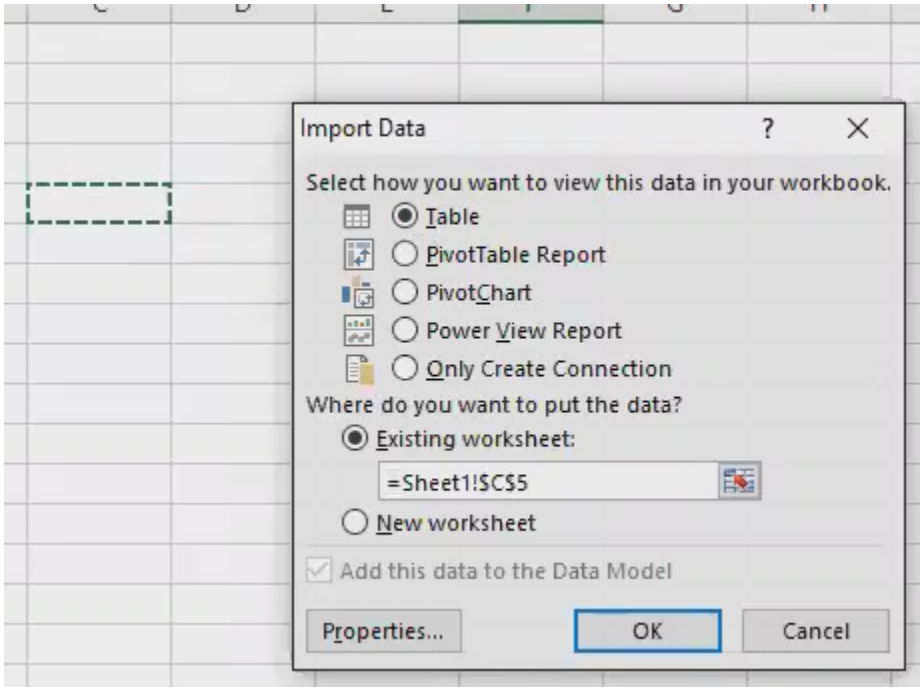


6. Review the information displayed on **Save Data Connection File and Finish** and then click **Finish**. Excel creates an ODC (OData Connection) file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. Specify how and where to place your data in the spreadsheet. Click **OK**.



Excel will place the data where you specified:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GroupID	GroupName	Type	ParentID	Scope
5d181b0f-ba93-4c81-83de-0aae9f946b5b	Wylie	1000000	0	Public
503f8aab-35c0-434a-a256-1a493ac1b539	TestNew	1000000	0	Public
42cc15b9-3bc9-417c-8957-97f22db2acfa	mwoTest	1000000	0	Public
2bb8b991-671b-4835-9207-5b27ff54a2f5	Frankfurt	1000000	0	Public
2b89caa2-e00c-497a-8a73-c897e1084da7	SAMC2016	1000000	0	Public
fb167546-fb19-4405-a17b-c4a5a31eed0f	Canterbury	1000000	0	Public
d8ec8390-c9ba-4192-850f-8790b09a3256	test	1000000	0	Public
8f707b8f-94a8-4a97-adfa-99f2c0518617	InTouch	1000000	0	Public
747fe221-5d98-4c3d-bae8-5ea268d40538	Baytown	1000000	0	Public

PowerBI query example

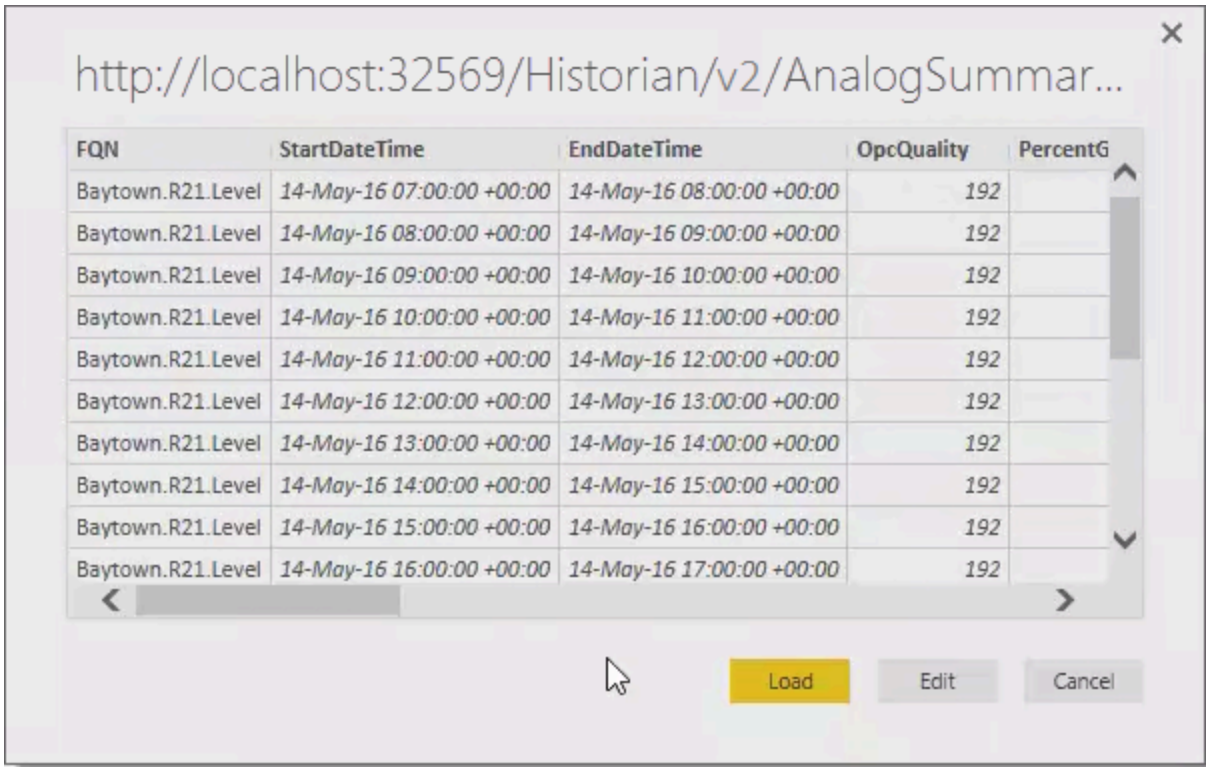
PowerBI query example

To import Insight data to PowerBI

1. Open PowerBI.
2. Click **Get Data**, and choose **OData Feed**.
3. Specify the query URL and click **OK**.

4. Specify **Basic Authentication**, and provide your Insight username and password.
5. Click **Connect**.
6. Power BI shows a preview of your data retrieved from Insight.

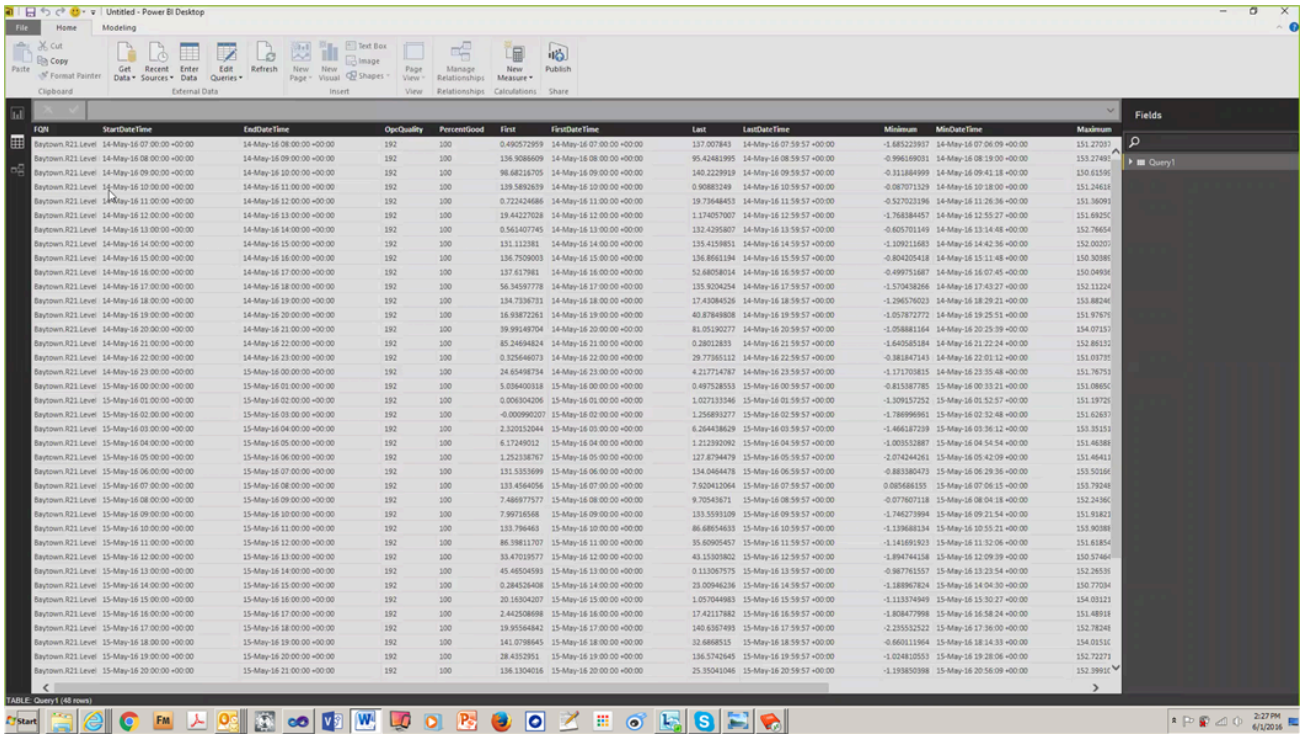
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



FQN	StartDateTime	EndDateTime	OpcQuality	PercentG
Baytown.R21.Level	14-May-16 07:00:00 +00:00	14-May-16 08:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 08:00:00 +00:00	14-May-16 09:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 09:00:00 +00:00	14-May-16 10:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 10:00:00 +00:00	14-May-16 11:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 11:00:00 +00:00	14-May-16 12:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 12:00:00 +00:00	14-May-16 13:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 13:00:00 +00:00	14-May-16 14:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 14:00:00 +00:00	14-May-16 15:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 15:00:00 +00:00	14-May-16 16:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 16:00:00 +00:00	14-May-16 17:00:00 +00:00	192	

7. Click **Load**.

8. In the upper-left corner, click to view the table of results.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Chart creation with URL-based query

You can create an Insight chart from a URL query. This is useful when you want to trigger the creation of an Insight chart from another application.

Note: You must be signed in to the Insight solution that corresponds to the data you reference before you can query it.

Parts of a URL-based query

The query must be in this format:

<base URL>?Tags=*<tag list><optional parameters, where parameters are joined by "&">*

Here is an example of a URL-based query specifying chart parameters:

```
http://myhistorian:32569/?tags=WeatherApp.Auckland.Cloudiness&chartType=LineChart&startT
ime=2017-29-04T08:45:00.000&endTime=2017-03-30T08:45:00.000
```

The URL-based query includes these parts:

Part of the query URL	In the example, it looks like this:
The base URL, including a specific identifier for your Insight solution	http://myhistorian:32569/
An indicator showing that parameters are to follow	?
Query parameters	tags=WeatherApp.Auckland.Cloudiness &chartType=LineChart &startTime=2017-03-29T08:45:00.000 &endTime=2017-03-30T08:45:00.000

A note about commas: For Chart URL, a comma is a special character. If you need to comma-separate multiple tags, you must do one of the following:
- Replace "," with "\,".
- Use a double-encoded tag. To generate a double-encoded tag, go to site such as https://meyerweb.com/eric/tools/dencoder/, click **Encode** twice, and use the resulting tag.

Supported query parameters

These query parameters are supported.While tag names are case-sensitive, parameters are not. (Parameters are case-insensitive.)

Parameter	**Description**
Tags	(required) Defines the tags to include in the chart. Use the case-sensitive, fully qualified name for each tag (that is, the data source name plus the tag name). Separate tag names with commas. For example: Tags=MyDataSource.Pump12.Temp\,MyDataSource.Pu

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameter	Description
	mp12.Speed
ChartType	Specifies the type of chart to use. Valid chart types are: ColumnChart, CumulativeLine, CumulativeSum DetailGrid, DiagnosticGrid, GanttChart LineChart, RollingSum, StatusBoard Summary, SummaryChart, SummaryGrid TimeInStateChart, XYPlot **Note:** CumulativeSum and RollingSum are the same thing. Both are available for backward compatibility. For example: ChartType=ColumnChart Default: StatusBoard
StartTime	Specifies the when the reporting time period begins, using this format: yyyy-MM-dd'T'HH:mm:ss.fff For example: StartTime=2017-03-27T08:47:40.763 StartTime is in local time (ISO format) for the client computer. Default: Today
EndTime	Specifies the when the reporting time period begins, using this format: yyyy-MM-dd'T'HH:mm:ss.fff For example: EndTime=2017-03-28T12:47:40.763 EndTime is in local time (ISO format) for the client computer. Default: Today
DurationHours	Specifies the duration to report, in hours. DurationHours is used only if provided. For example: DurationHours=2
sliceBy	Specifies the event aggregate in a column chart. This parameter accepts the fully qualified name of a non-analog tag. If sliceBy is provided in the URL along with resolution

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameter	Description
	or timeAggregate, those parameters are ignored and event aggregate is selected for the chart. If no tag name, or an invalid tag name, is provided for sliceBy, the sliceBy option is ignored. For example: sliceBy=IntouchAlarm.ConcValve1
resolution	Specifies a time aggregate for a column chart in milliseconds. Accepted values are 86400000 (DAY), 3600000 (Hour) and 900000 (15 Min). For example: resolution=900000
timeAggregate	Specifies a time aggregate for a column chart in daily or hourly values. Valid values are: Day, Hour,15minutes, or Event. Alternatively, you can provide 1,2,3 or 4 for Day,Hour,15minute, or Event, respectively If the sliceBy query parameter is used, then timeAggregate is automatically set to Event and any other timeAggregate option is ignored For example: timeAggregate=Day If resolution and timeAggregate are both provided, only resolution is considered.
valueAggregate	Specifies a value aggregate for a chart supporting value aggregate options. Valid values are minimum, maximum, integral, average, or last as long as the value is valid for the corresponding chart.

About StartTime, EndTime, and DurationHours

You do not need to specify StartTime, EndTime, and DurationHours.

- When all three are specified, Insight uses StartTime and EndTime, but ignores DurationHours.
- When either StartTime or EndTime is specified along with DurationHours, the two specified parameters are used for StartTime/EndTime calculations.
- When either StarTime or EndTime is specified without DurationHours, CurrentTime and the specified parameter are used for StartTime/EndTime calculations.
- When none of these three parameters are specified, the chart shows tag values for today ending with the current time.

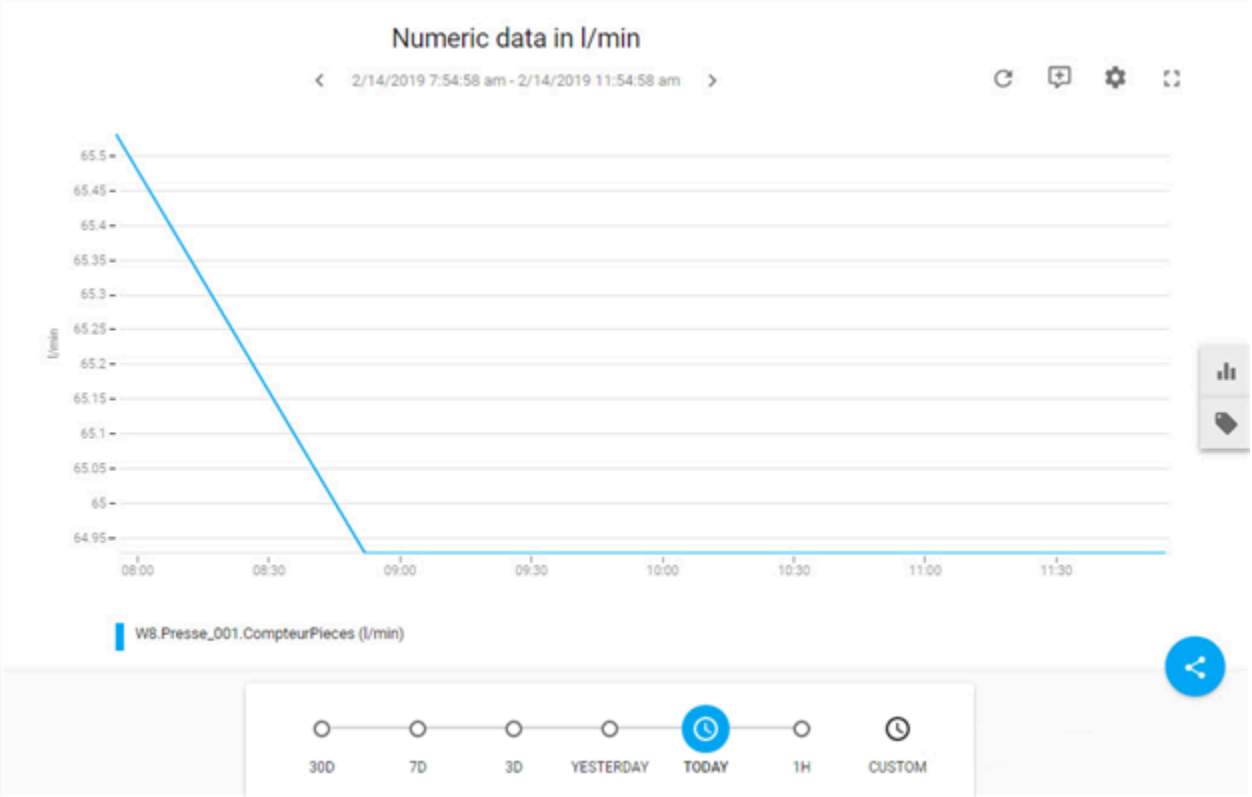
Example

For example, after logging into the corresponding Insight solution for the data you want charted, you could type this URL in a web browser:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
http://myhistorian:32569/?DurationHours=4&tags=Seattle.W8.Press_001.Pieces&chartType=LineChart
```

This chart would display:



Content Scripting REST API

The Content Scripting REST API allows you to programmatically define layout and data details to create Insight content.

For examples of code to create different types of charts, see the Examples section.

Endpoint URL

Content Scripting REST API uses this endpoint URL:

```
http://localhost:32569/Historian/v2/Contents?overwrite=true
```

The query parameter "overwrite" is used to overwrite the existing content with the same name. If this parameter is not provided and if the content with same name exist it will throw the duplicate content exist (409) error.

PostData Resource

Description	Describes the data and layout that make up a set of private or shared content.
URL	`http://localhost:32569/Historian/v2/Contents?overwrite=true`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Method	POST

Required Parameters	• **Name**
	The name of the content. If content with same name already exist, the API will fail and you'll see "duplicate content exists" as an HTTP status message.
	If you use "overwrite=true", the existing content with same name will be overwritten.
	• **CreatedUserFirstName**
	The first name of user associated with this content. This name displays as a tooltip or in the Content Management page. This name is not validated, so be sure to check spelling, etc. Example:
	`"CreatedUserFirstName": "Wally"`
	• **CreatedUserLastName**
	The last name of user associated with this content. This name displays as a tooltip or in the Content Management page. This name is not validated, so be sure to check spelling, etc. Example:
	`"CreatedUserLastName": "Wonderware"`
	• **UserID**
	The email ID of the user who created this personalized content. This user ID does not get validated, so be sure check for spelling, etc. yourself. Example:
	`UserID:"wally.wonderware@gmail.com"`
	• **ChartType**
	Specifies the chart type you want to use for this content.
	Examples:
	`"ChartType":"Line Chart"`
	`"ChartType":"Status Board"`
	`"ChartType":"Cumulative Line"`
	`"ChartType":"Summary Chart"`
	Note: If chart type provided is not one of the value listed,error 401 results.
	For complete code examples that create different types of charts, see the Examples section.
	If the layout is "All Tags" and is the active layout, these chart types are supported:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• Column Chart • Composite Chart • Detail Grid • Map Chart • Status Board (default) • XY Plot If the layout is created for analog tags, these chart types are supported: • Column Chart • Cumulative Line • Cumulative Sum • Detail Grid • Diagnostic Grid • Line Chart • Map Chart • Status Board (default) • Summary Chart • Summary Grid • XY Plot **Note:** "Cumulative Sum" is the ChartType value used for a rolling sum chart. If the layout is created for non-analog tags, these chart types are supported:\ • Diagnostic Grid • Gantt Chart • Map Chart • Status Board (default) • Summary • Summary Grid • Time-in-State Chart • **LayoutInfo** Information about the layout. At minimum, LayoutInfo must include at minimum TagDetails with some tags value.
Optional Parameters	• **TimePreset** The relative or absolute time duration for the selected chart. Supported values are: Relative time: • 1 = Last 30 days • 2 = Last 7 days • 3 = Last 3 days • 4 = Yesterday • 5 = Today (default)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• 6 = Last hour
	• 8 = This week
	• 9 = Last week
	• 10 = This month
	• 11 = Last month
	Absolute time:
	User can also provide absolute time in UTC format. Start time and end time should be separated by a comma, and start time should be the first. Example:
	`"TimePreset": "2018-01-19T22:23:43.836Z,2018-01-21T22:23:43.836Z"`
	Example of custom duration relative to current time:
	`"TimePreset": "OffsetInMilliSecs:67288974"`
	• **Keywords**
	• An array of keywords for this content. Example: "Keywords":["SecretFormula"]
	• **ShareMode**
	Makes the content public or private. The following values are supported:
	• 73 = Private content
	• 74 = Shared content (default)
	Example:
	`"ShareMode":"74"`
	• **Scope**
	The scope for the content to be created. Default value is "/". Examples:
	`"Scope":"/"`
	`"Scope":"/Montreal/"`

Sample Query	`{ "upsert": { "Name": "Column chart ContentWithScript", "UserID": "wally.wonderware@gmail.com", "CreatedUserFirstName": "Wally", "CreatedUserLastName": "Wonderware", "ChartType": "Column Chart", "LayoutInfo": { "TagDetails": [ { "TagName": "Weather.Seattle.Cloudiness", "IsSelected": "true"`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`},`
	`{`
	`"TagName": "Weather.Seattle.Wind Direction",`
	`"IsSelected": "true"`
	`},`
	`{`
	`"TagName": "Weather.Seattle.Wind Speed",`
	`"IsSelected": "true"`
	`}`
	`],`
	`"TimePreset":"5",`
	`"TimeAggregate":"2"`
	`},`
	`"Keywords":[]`
	`}`
	`}`

Notes	• The details returned by this API includes an ID that is used internally to identify the specific content in the UI. • If you display the content you post, and then change sharemode, timepreset, and/or timeaggregate values, you must refresh the displayed content page to see updates. You can refresh the entire page, or search for the content again via the search box.

LayoutInfo

Description	Contains the TagDetails and Layout properties.
Required Parameters	• **Layout** An array of layouts to display. The first layout must use the name "All Numeric". See Layout. • **TagDetails** An array of tags to display. See TagDetails.

Sample	`"LayoutInfo": {`
	`"TagDetails": [`
	`{`
	`"TagName":`
	`"alarmstests.W12R12-SQL16.SysPerfAvailableMB`
	`ytes",`
	`"IsSelected": "true"`
	`}`
	`],`
	`"Layout": [`
	`{`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	"Name": "MB",
	"IsActiveGroup": "true",
	"LayoutProperties": {
	"selectedOptions": {
	"yAxis": {
	"type": "stacked"
	},
	"scaleY": false
	}
	}
	}
	]
	}

Layout

Layout

Description	This object is used to configure the chart layout. Each Layout object represents the engineering unit group displayed on the left panel. Layout object is used to decide which layout will be selected/active when content is opened, should chart display in stack single or multi axis y mode and another configuration related to the chart. For example, the LayoutProperties object in following post data is used to display line chart in All Numeric group in stack mode, with limit line visible and Y-axis scaled to data available in it.

Required Parameters	• **Name**
	Names the tag group to be used this layout. These are possible values:
	  • **For analog tags listed in the Tags parameter:** The name of any tag group for a specific unit of measure.
	  • **For non-analog tags listed in the Tags parameter:** "" (An empty string), specifying the "String data" tag group will be used.
	  • **If isActiveGroup is set to true** "All Numeric".
	• **isActiveGroup**
	Indicate whether this layout is active. These are

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	possible values:
	`"IsActiveGroup":"true",`
	`"IsActiveGroup":"false",`
	`"IsActiveGroup":""`
	• **LayoutProperties**
	Contains a set of options for the layout. See LayoutProperties.
	• **TimeAggregate**
	For charts that support time aggregation (such as column charts), specifies the time aggregation. Supported values are:
	• 1 = Day
	• 2 = Hour
	• 3 = 15-minute
	• 4 = Event (only applicable for column charts using sliceBy)
	Example:
	`"TimeAggregate": "1"`

LayoutProperties

LayoutProperties

Description	Contains a set of layout options nested within the **selectedOptions** object.
Required Parameters	• **selectedOptions** An array of selectedOptions.

Sample	`"LayoutProperties": {`
	`"selectedOptions": {`
	`"xAxis": {`
	`"label": "DateTime",`
	`"ticks": -1,`
	`"visible": true`
	`},`
	`"yAxis": {`
	`"label": "All Tags",`
	`"ticks": -1,`
	`"visible": true,`
	`"type": "stacked"`
	`},`
	`"limitLine": {`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`"position": 10,`
	`"text": "Limit Line Text",`
	`"visible": false`
	`},`
	`"legend": {`
	`"visible": true`
	`},`
	`"scaleY": true,`
	`}`
	`}`

selectedOptions

selectedOptions

Description	Defines layout options.

Optional Parameters	• **xAxis**
	Configures the X axis of the specified chart (for example, line chart or composite chart). Parameters are:
	  • **label**
	  Defines the axis label text.
	  • **ticks**
	  Specifies the number of tick marks to display. Use "-1" to display the chart's default ticks.
	  • **visible**
	  Specifies whether to make the axis visible (true or false).
	  • **tag**
	  For XY plots, specifies the fully qualified name of a tag used for the X axis.
	  Example:
	  `"xAxis": {`
	  `"label": "Dallas.Wind Speed",`
	  `"tag": "Weatherapp.Dallas.Wind Speed",`
	  `"ticks": -1,`
	  `"visible": true`
	  `},`
	• **yAxis**
	Configures the Y axis of the specified chart (for example, line chart or composite chart). Parameters are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **label**
  Defines the axis label text.
- **ticks**
  Specifies the number of tick marks to display. Use "-1" to display the chart's default ticks.
- **visible**
  Specifies whether to make the axis visible (true or false).
- **type**
  Specifies the type of Y axis to use -- stacked, single, or multi.

  Example:

```
yAxis: {
"label": "All Tags",
"ticks": -1,
"visible": true,
"type": "stacked"
}
```

- **limitLine**
  Defines a limit line for the chart. Parameters are:
  - **position**
    Sets the position on the Y axis for the limit line.
  - **text**
    Defines the limit line text.
  - **visible**
    Specifies whether the limit line displays on the chart (true or false).

    Example:

```
"limitLine": {
"position": 10,
"text": "Limit Line Text",
"visible": false
}
```

- **legend**
  Specifies whether to include a legend for the chart. There is one parameter:
  - **visible**
    Indicate whether the legend displays (true or false).

    Example:

```
"legend": {
"visible": true
}
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **scaleY**
  For a line chart or composite chart, specifies whether the Y axis scales according to the data in the chart.

  Example:

  ```
  "scaleY": true
  ```

- **scaleX**
  For an XY plot, specifies whether the X axis scales according to the data in the chart.

  Example:

  ```
  "scaleX": true
  ```

- **lookBack**
  For integers in a cumulative sum chart or rolling sum chart, specifies the lookback period.

  Example:

  ```
  "lookBack": 7
  ```

- **units**
  In a cumulative sum chart or rolling sum chart, specifies the units used for the lookBack value -- days or hours.

  Example:

  ```
  "unit": "hours"
  ```

- **sortType**
  In a summary bar chart, indicates the sorting order for tags. Supported values are:

  - Largest to Smallest
  - Smallest to Largest
  - A-Z

    Example:

    ```
    "sortType":"Largest To Smallest"
    ```

- **displayLines**
  For XY plots, specifies whether plot points will be joined with a line.

  Example:

  ```
  "displayLines": true
  ```

- **aggregation**
  For a chart that supports value aggregation (such as a column chart), this sets the selected value aggregation for chart. Supported values are:

  - average
  - meter (analog tags only)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• integral (analog tags only)
	• last
	• maximum
	• minimum
	• summary (non-analog tags only)
	• total (non-analog tags only)
	
	Example:
	`"aggregation": "Maximum"`
	
	• **map**
	For map charts, this sets the selected value aggregation for chart. Parameters are:
	  • **zoomLevel**
	  Specifies the zoom level of the map. If not specified or invalid, the chart automatically zooms to fit all tags.
	  Example:
	  `"zoomLevel": 10,`
	  • **center**
	  Specifies the geolocation for the center of the map, specified with longitude and latitude.
	  Example:
	  `"center": {`
	  `"Longitude": -110.44413636978399,`
	  `"Latitude": 48.38785248165351`
	  `}`
	
	• **SliceByTag**
	For column charts, this sets an event aggregate for the specified non-analog tag. The value must be a fully qualified tag name.
	Example:
	"sliceByTag": "Weather.Auckland.Clouds",

Sample	`"LayoutProperties": {`
	`"selectedOptions": {`
	`"xAxis": {`
	`"label": "DateTime",`
	`"ticks": -1,`
	`"visible": true`
	`},`
	`"yAxis": {`
	`"label": "All Tags",`
	`"ticks": -1,`
	`"visible": true,`
	`"type": "stacked"`
	`},`
	`"limitLine": {`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	"position": 10,
	"text": "Limit Line Text",
	"visible": false
	},
	"legend": {
	"visible": true
	},
	"scaleY": true,
	"map": {
	"zoomLevel": 10,
	"center": {
	"Longitude": -110.44413636978399,
	"Latitude": 48.38785248165351
	}
	}
	}
	}

TagDetails

TagDetails

Description	Describes one tag in the layout.
Required Parameters	• **TagName** Provides the fully qualified name of the tag. • **IsSelected** Indicate whether this tag is selected. Tags are selected by default.

Sample	{
	"TagName":
	"_PLPWSP2017_AnE.PLPWSP2017.Smiley.HX",
	"IsSelected": "true",
	}

Examples

You can use any client tool to create a Content Scripting API request. Select the links below to examine some scripting examples.

Use Chrome Developer tool to capture content

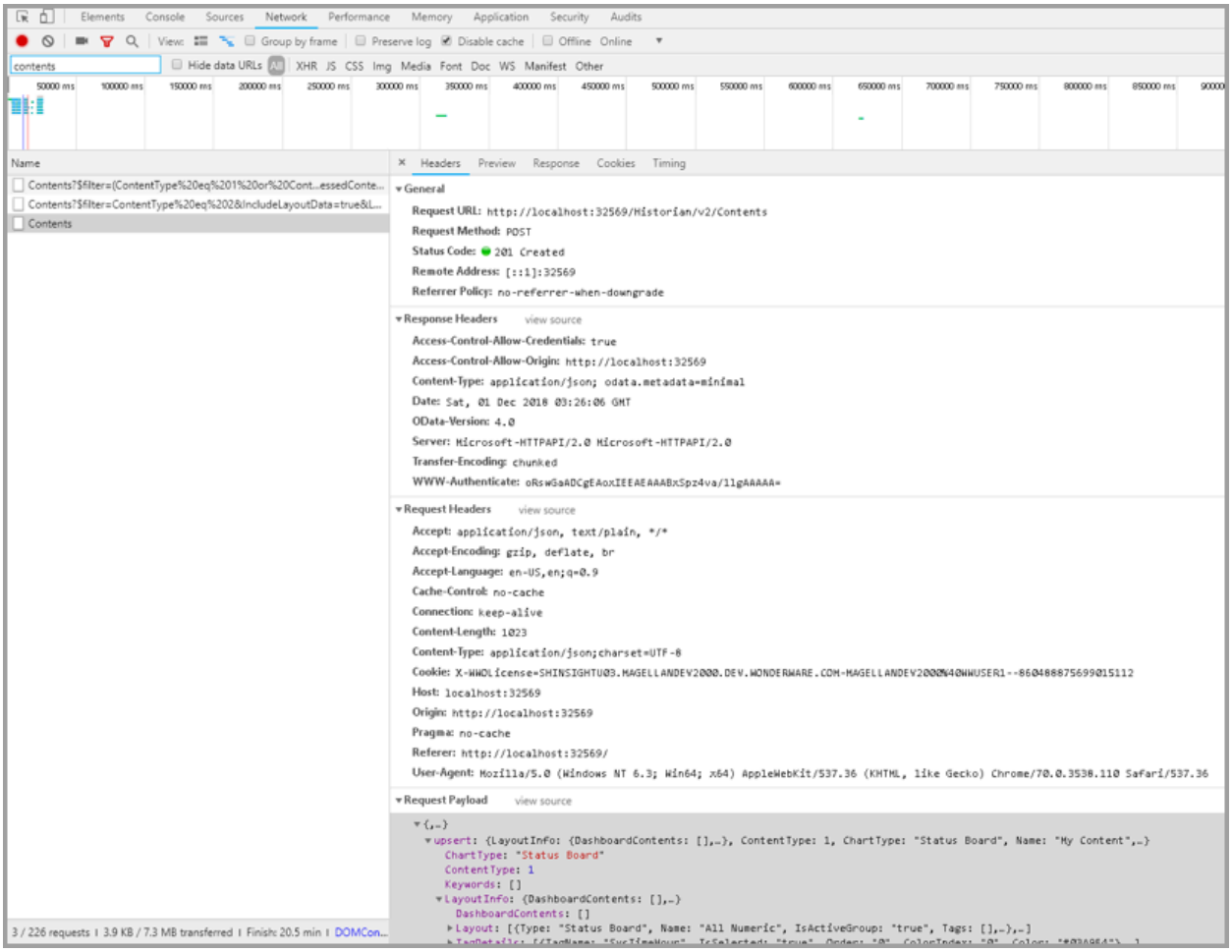
You can use the template from the Chrome Developer tool and a PowerShell script to create content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To capture content for a post request with Chrome Developer tool

1. Using a Chrome browser, create content in AVEVA Historian Client Web.
2. Press F12 to open the Chrome Developer tool.
3. In the Chrome Developer tool:
   a. Select the **Network** tab.
   b. Click ⊘ to clear the tool's displayed information.
4. In AVEVA Historian Client Web, save your content.
5. In the Developer tool:
   a. Click the "Content" line to see details.
   b. In the **General** section, note the Request URL:

   c. In the **Request Payload** section, copy all the content.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

6. Paste the request payload into a JSON formatter and make any edits.
   - Remove these items: URL property, CreationDate, LastModifiedDate, LastAccessedDate, and SharedDate, the numeric portion of the UserID (retain the email address), and TenantID.
   - Optionally, modify these items as needed: FavoriteName, UserID (email address only), and UserName.

     Thjs is an example of the resulting code:


```
{
"upsert": {
"LayoutInfo": {
"DashboardContents": [
],
"Layout": [
{
"Type": "Column Chart",
"Name": "All Numeric",
"IsActiveGroup": "true",
"Tags": [
],
"UIProperties":
"{\"version\":{\"major\":1,\"minor\":5,\"timestamp\":\"\"},\"selectedOptions\":
{\"limitLine\":{\"visible\":false,\"position\":10,\"text\":\"Limit Line
Text\"},\"legend\":
{\"visible\":true},\"xAxis\":{\"label\":\"DateTime\"},\"yAxis\":{\"label\":
\"All Tags\"},\"dates\":{\"from\":
\"2018-11-15T19:30:00.000Z\",\"to\":\"2018-11-15T20:04:33.099Z\"},\"aggregation\"
:
\"Maximum\",\"isChartOptionAvailable\":true,\"isTimeAggregateAvailable\":
true,\"isValueAggregateAvailable\":true,\"isCSVDownloadAllowed\":true,\"timeAggre
gate\":\"15MINUTES\"}}"
},
{
"Type": "",
"Name": "None",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
},
{
"Type": "",
"Name": "65F",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
}
],
"TagDetails": [
{
"TagName": "Weather.Auckland.Daily Cooling Degrees",
"IsSelected": "false",
"Order": "",
"ColorIndex": "undefined",
"Color": "#eeeeee"
},
```


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
{
"TagName": "Weather.Auckland.Weather ID",
"IsSelected": "true",
"Order": "0",
"ColorIndex": "1",
"Color": "#AD1457"
}
],
"TimeAggregate": 3,
"TimePreset": "OffsetInMilliSecs:1700224"
},
"ContentType": 1,
"ChartType": "Column Chart",
"Name": "Quarterly Report",
"Keywords": [
],
"Location": "/",
"SharedMode": 98
}
}
```

7. Paste the resulting code (shown below in blue) in Powershell as a POST request.

   For the **-Uri** parameter, use the Request URL from step 7 (shown in red below).

   For example:

```
$header = @{
"Content-Type"="application/json;charset=UTF-8"
};
$postdata='{
"upsert": {
"LayoutInfo": {
"DashboardContents": [
],
"Layout": [
{
"Type": "Status Board",
"Name": "All Numeric",
"IsActiveGroup": "true",
"Tags": [
],
"UIProperties":
"{\"version\":{\"major\":1,\"minor\":5,\"timestamp\":\"\"},\"selectedOptions\":{
\"dates\":{\"from\":\"2018-12-01T00:00:00.000Z\",\"to\":\"2018-12-01T03:25:52.10
8Z\"},\"isCSVDownloadAllowed\":true}}"
},
{
"Type": "",
"Name": "Minute",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
},
{
"Type": "",
"Name": "Hour",
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
    "IsActiveGroup": "false",
    "Tags": [
    ],
    "UIProperties": ""
    },
    {
    "Type": "",
    "Name": "Second",
    "IsActiveGroup": "false",
    "Tags": [
    ],
    "UIProperties": ""
    }
    ],
    "TagDetails": [
    {
    "TagName": "SysTimeHour",
    "IsSelected": "true",
    "Order": "0",
    "ColorIndex": "0",
    "Color": "#03A9F4"
    },
    {
    "TagName": "SysTimeMin",
    "IsSelected": "true",
    "Order": "1",
    "ColorIndex": "1",
    "Color": "#AD1457"
    },
    {
    "TagName": "SysTimeSec",
    "IsSelected": "true",
    "Order": "2",
    "ColorIndex": "2",
    "Color": "#F57F17"
    }
    ],
    "TimeAggregate": 2,
    "TimePreset": 5
    },
    "ContentType": 1,
    "ChartType": "Status Board",
    "Name": "My Content",
    "Keywords": [
    ],
    "Location": "",
    "SharedMode": 73
    }
    }'
    try{
    $response = Invoke-WebRequest -Uri http://localhost:32569/Historian/v2/Contents
    -Method Post -Body $PostData -Headers $header -ErrorVariable RespErr
    -UseDefaultCredentials
    $response;
    }
    catch [System.Net.WebException] {
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
    if($_.Exception.Response.StatusCode -eq "Conflict"){
    Write-Host "It's a duplicate Content"
    }else{
    Write-Host "The Request failed.";
    $respStream = $_.Exception.Response.GetResponseStream()
    $reader = New-Object System.IO.StreamReader($respStream)
    $ResponseContent = $reader.ReadToEnd();
    $ResponseContent;
    $RespErr;
    }
    }
```

Create content with minimum required properties

This example shows the minimum required properties for postData. It creates the default status board chart using the "All Tags" group and the specified tags.

```
{
"upsert": {
"Name": "ContentWithScript",
"ChartType": "Status Board",
"LayoutInfo": {
"TagDetails": [
{
"TagName": "Weatherapp.Richmond.Wind Speed"
}
]
}
}
}
```

Create content using all properties

This postData example uses all properties to create an XY plot.

```
{
"upsert":{
"Name": "Wind Speed",
"ChartType": "XY Plot",
"TimePreset": "OffsetInMilliSecs:16028059",
"TimeAggregate": "2",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"xAxis": {
"label": "Dallas.Wind Speed",
"tag": "Weatherapp.Dallas.Wind Speed",
"ticks": -1
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
},
"yAxis": {
"type": "single"
},
"limitLine": {
"position": 10,
"text": "Limit Line Text",
"visible": false
},
"legend": {
"visible": true
},
"scooter": {
"visible": true
},
"sortType": "SmallesttoLargest",
"lookBack": 7,
"units": "hours",
"scaleX": true,
"scaleY": true,
"displayLines": false,
"aggregation": "avg_xy"
}
}
}
],
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"wind"
]
}
}
```

Create content with PowerShell

This creates a status board using the "All Tags" tag group.

Note: The header requires an API key. For information on creating an API key, see Acquire an API key for data retrieval.

```
$header = @{
"Content-Type"="application/json;charset=UTF-8"
};
$PostData = '{
"upsert": {
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"Name": "ContentWithScript",
"ChartType": "Status Board",
"LayoutInfo": {
"TagDetails": [
{
"TagName": "Weatherapp.Richmond.Wind Speed"
}
]
}
}
}
'
try{
$response = Invoke-WebRequest -Uri http://localhost:32569/Historian/v2/
Contents?overwrite=true -Method Post -Body $PostData -Headers $header -ErrorVariable
RespErr
$response;
}
catch [System.Net.WebException] {
if($_.Exception.Response.StatusCode -eq "Conflict"){
Write-Host "This is a duplicate content"
}else{
Write-Host "The Request failed.";
$respStream = $_.Exception.Response.GetResponseStream()
$reader = New-Object System.IO.StreamReader($respStream)
$ResponseContent = $reader.ReadToEnd();
$ResponseContent;
}
}
```

Create a column chart

This example produces a column chart using the "MB" tag group and these parameters:

- Value aggregation is maximum
- Time aggregation is 15 minutes

```
{
"upsert": {
"Name": "Column Chart",
"TimePreset": "5",
"UserName": "Wally",
"UserID": "wally.wonderware@gmail.com",
"TimeAggregate": "3",
"LayoutInfo": {
"Layout": [
{
"Name": "MB",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"aggregation": "maximum"
}
}
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
}
],
"TagDetails": [
{
"TagName": "alarmstests.W12R12-SQL16.SysPerfAvailableMBytes",
"IsSelected": "true"
}
],
}
}
}
```

Create a column chart with sliceBy support

This example produces a column chart with the sliceBy parameter. Note that:

1. The layout selected is “All Numeric” and the associated IsActiveGroup property is set to true.
2. Analog and non-analog tags are added in the TagDetails.
3. The sliceByTag propetry under LayoutProperties in the active layout should be set to a non-analog tag. The tag is added through TagsDetails.
4. timeAggregate under LayoutProperties in the active layout should be set to EVENT.

```
{
"upsert": {
"LayoutInfo": {
"Layout": [
{
"Type": "Column Chart",
"Name": "All Numeric",
"IsActiveGroup": "true",
"Tags": [
],
"LayoutProperties": {
"selectedOptions": {
"limitLine": {
"visible": false,
"position": 10,
"text": "Limit Line Text"
},
"aggregation": "Integral",
"sliceByTag": "Weather.Auckland.Clouds",
"timeAggregate": "EVENT"
}
}
},
{
"Type": "",
"Name": "",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
},
{
"Type": "",
"Name": "None",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
}
],
"TagDetails": [
{
"TagName": "Weather.Auckland.Weather ID",
"IsSelected": "true"
},
{
"TagName": "Weather.Auckland.Clouds Text",
"IsSelected": "false"
}
],
"TimeAggregate": 4,
"TimePreset": 2
},
"UserID": "wally.wonderware@gmail.com",
"CreatedUserFirstName": "Wally",
"ContentType": 1,
"ChartType": "Column Chart",
"Name": "Test ContentAPI-Layout-Case",
"Keywords": [
"sliceby"
],
"Location": "/",
"SharedMode": 98
}
}
```


Create a composite chart

This example produces a composite chart using the All Numeric tag group.


```
{
"upsert": {
"Name": "ContentWithScript",
"UserID": "wally.wonderware@gmail.com",
"CreatedUserFirstName": "Wally",
"CreatedUserLastName": "Wonderware",
"ChartType": "Composite Chart",
"LayoutInfo": {
"TagDetails": [
{
"TagName": "alarmstests.W12R12-SQL16.SysPerfAvailableMBytes",
"IsSelected": "true"
}
]
},
```


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"Scope": "/"
}
}
```

Create a Gantt chart

This example produces a Gantt chart. The Name property of the layout is empty, so the "String data" tag group is used.

```
{
"upsert": {
"Name": "Gantt Chart ContentWithScript",
"ChartType": "Gantt Chart",
"LayoutInfo": [
{
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Direction Text",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Auckland.Wind Speed Text",
"IsSelected": "true"
}
]
}
]
}
}
```

Create a line chart

This example produces a line chart using the following parameters:

- The chart type is a line chart that uses the "MB" tag group.
- The Y-axis is stacked and not scaled.
- A keyword ("SecretRecipe") is assigned to this content.
- Time duration for this content is "Today" (where TimePreset is 5).

```
{
"upsert": {
"Name": "Line Chart ContentWithScript",
"ChartType": "Line Chart",
"LayoutInfo": {
"Layout": [
{
"Name": "MB",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"yAxis": {
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"type": "stacked"
},
"scaleY": false
}
}
}
],
"TagDetails": [
{
"TagName": "alarmstests.W12R12-SQL16.SysPerfAvailableMBytes",
"IsSelected": "true"
}
]
}
"Keywords": [
"SecretRecipe"
]
}
}
```

Create a rolling sum chart

This example creates a rolling sum chart using the "mph" tag group.

```
{
"upsert": {
"Name": "ContentWithScript",
"ChartType": "Cumulative Sum",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"lookBack": 7,
"units": "hours"
}
}
}
],
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"wind"
]
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
}
}
```

Create an XY plot

This example produces an XY Plot chart using the "mph" tag group. it specifies a tag for the X axis and sets scaling to "true" for both X and Y axes.

Note: The layout object corresponds to the engineering unit of X axis tag should be present in layout object.

```
{
"upsert": {
"Name": "XYPlot Chart ContentWithScript",
"ChartType": "XY Plot",
"TimePreset": "OffsetInMilliSecs:16028059",
"TimeAggregate": "2",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"xAxis": {
"label": "Dallas.Wind Speed",
"tag": "Weatherapp.Dallas.Wind Speed"
},
"scaleX": true,
"scaleY": true,
"displayLines": false,
"aggregation": "avg_xy"
}
}
}
],
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"windy"
]
}
}
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Errors

Duplicate content

If content name already exist and "Overwrite=true" is not used, the API call fails with http status 409 and generates this message:

```
Already content exists with name 'ContentWithScript'
```

Unsupported parameter

If PostData has any unsupported parameter, the API call fails with http status 400 and generates this message:

```
Content should have valid name
Invalid post data. ChartType property does not have valid value.
```

Historian Client Web Modern

This guide describes the features and functionality of Historian Client Web in the modern look.

Use Historian Client Web

See About the Home page.

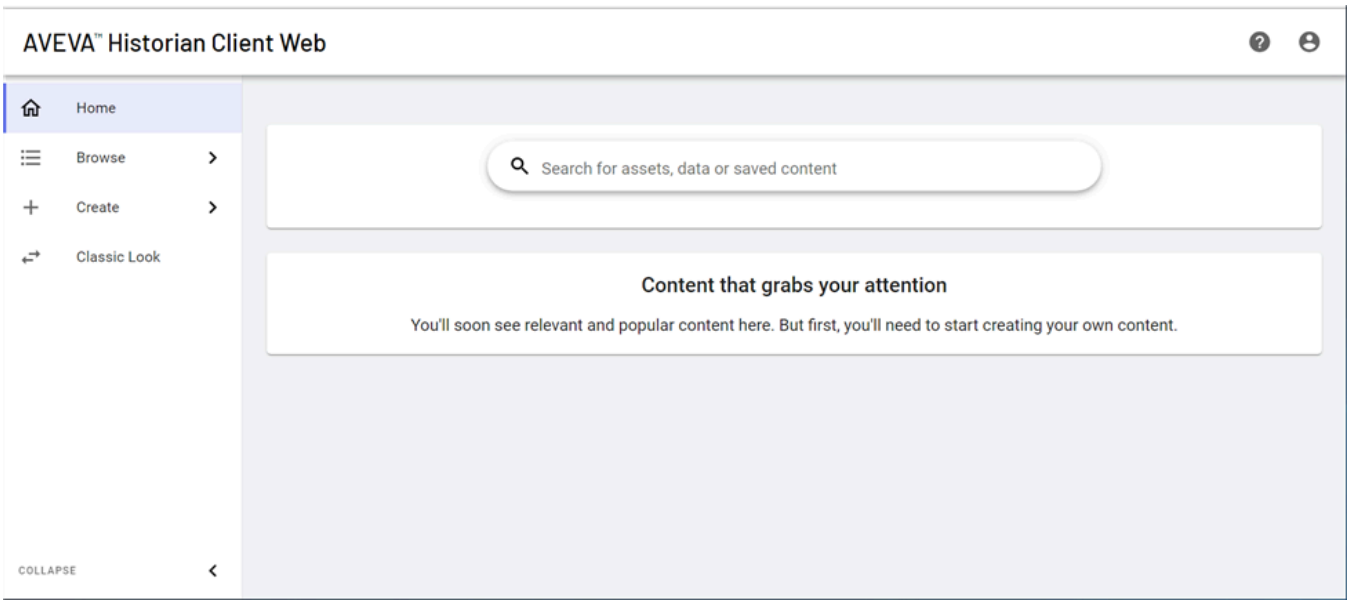
Logging in to Historian Client Web

When you open Historian Client Web, you may be asked to log in. Log in with your SSO or Windows credentials. If your environment is using the connected experience, logging in with Windows integrated login will only give you access to demo mode. Demo mode has a 5 minute limit.

About the Home page

AVEVA Historian Client Web gives you instant access to all of your organization's production and performance data any way you want it. You'll find your most recent content on the Home page.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Note: AVEVA Historian Client Web is supported with Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari on iPad with Retina display (in landscape mode).

Search for Data or Content

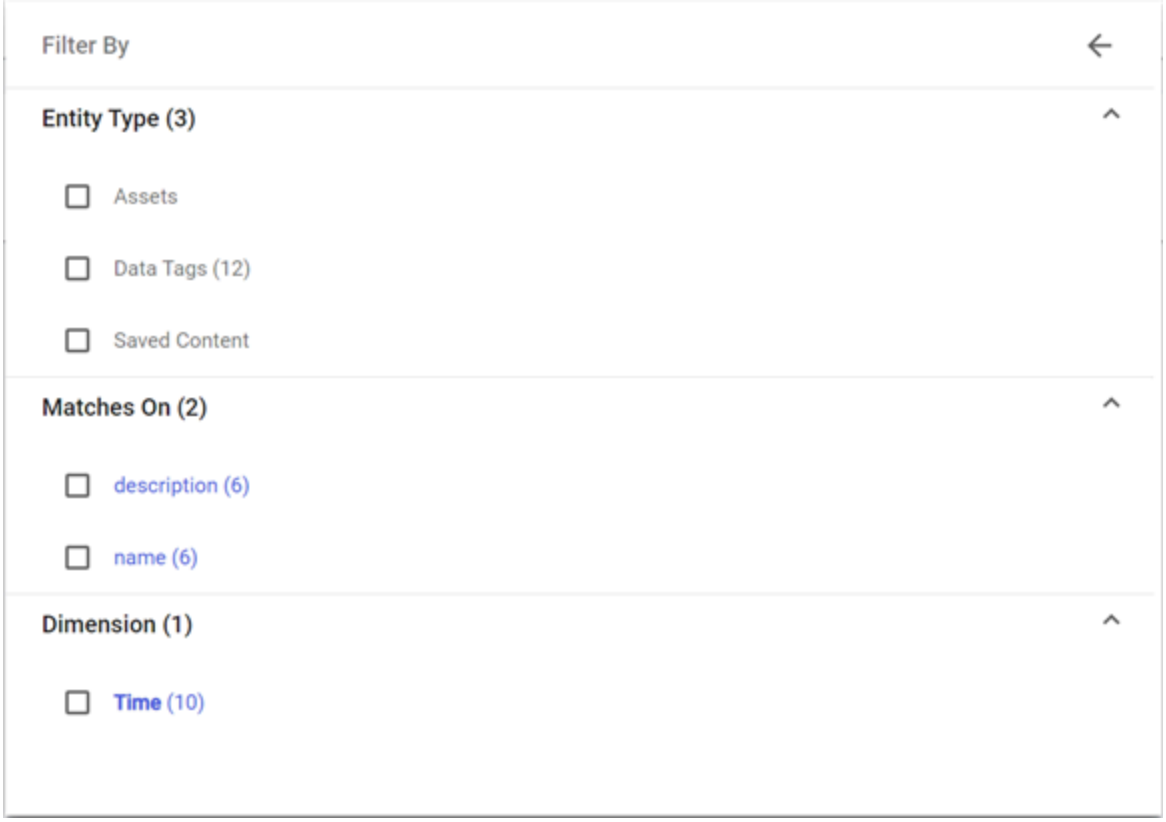
To open the search panel, place your cursor in the search box on the home page.

Filter data using specific criteria

Your query may produce a lot of results. You can refine your search results by applying filters found in the right side of the search panel.

The filter options are hidden by default. To open the filters, select . Select to return back to the previous search panel.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Note: You may find an inconsistency with the number of search results returned and the count listed in the the filter panel. This occurs when there is a large amount of results available. To avoid this issue, add more specific search terms so that you get a more accurate number of search results.

Search by asset type

You can search by asset type by selecting the **Assets** entity type, an asset from the search results, and then either directly select the asset name in the list or select **Display Asset Page**.

Selecting multiple assets and selecting **Display List** will open the **Asset List** page showing your selected items.

Create new content or dashboard

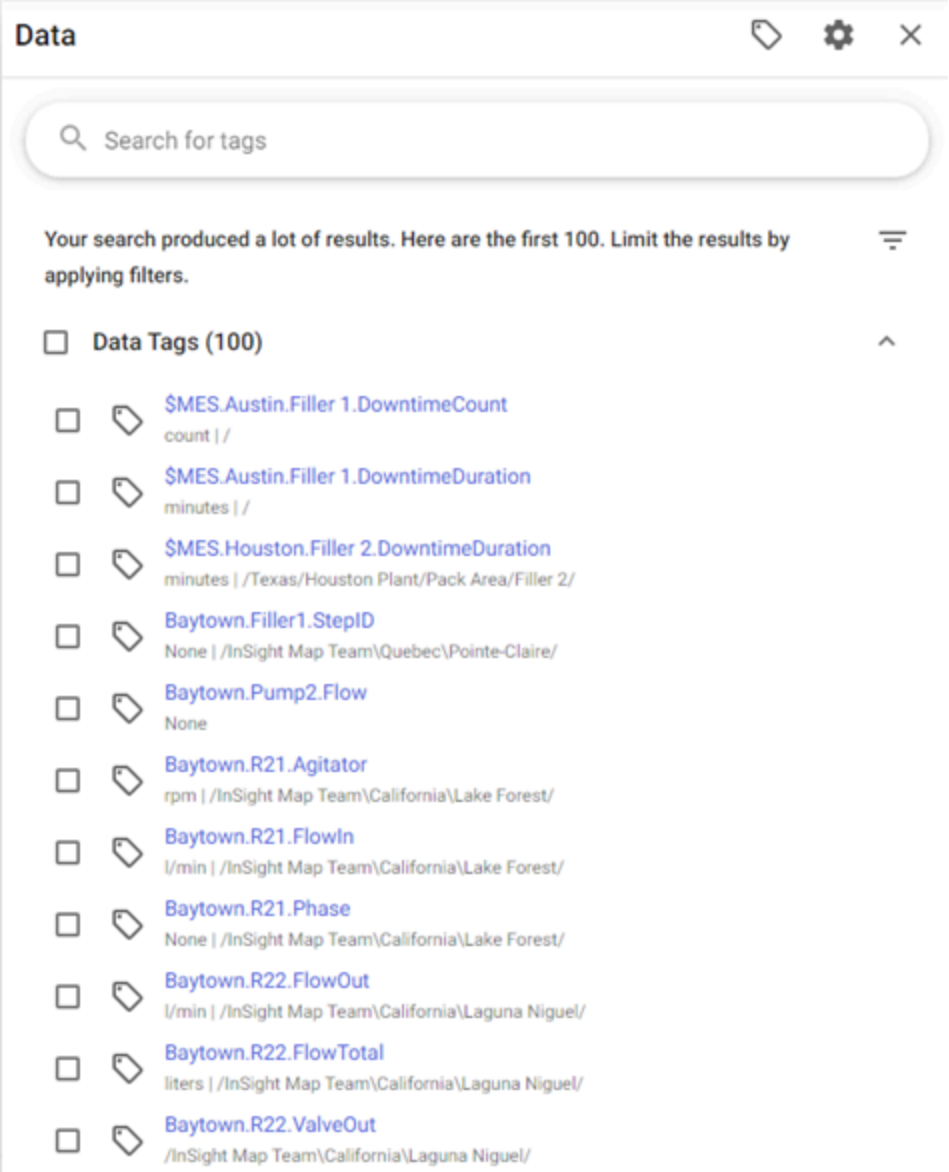
You can create new content or a new dashboard directly from your search results. To create new content, select at least one tag from your search results, then select **Create New Content**. To create a new dashboard, select at least two Saved Content items from your search results, then select **Create Dashboard**.

Search for tags from the Data panel

Search functionality is also available in the Data panel. When you create visual content, select **Data** to open the the Data panel, select **Add Tags**, and begin searching or applying filters.

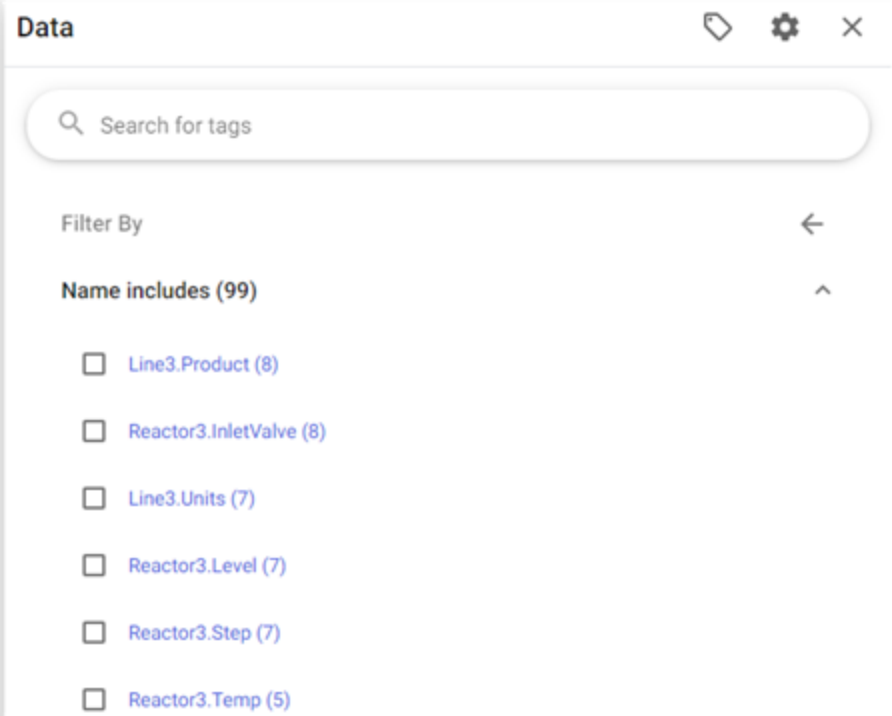
Note: Assets and **Saved Content** entity types cannot be added to visual content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



The filter options are hidden by default. To open the filters, select . Select to return back to the previous search panel.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Create and manage visuals

Visuals visually represent your data or, in certain cases, data from external sources.

Once created, visuals can be incorporated into dashboards for creating custom views for your data.

Create visuals

There are several ways to create visuals in Historian Client Web.

- Select **Create** from the navigation rail, and then select **Visual** to open the visual editor. Choose a visual type, and then select tags from the Data panel.
- Search for data from the search bar and select the tags to include in your visual. Select **Create New Content**. Use the Data panel to add or remove data tags. Use the Settings panel to choose a visual type and define properties.
- Select **Browse** from the navigation rail, and then select **Tags**. Select tags to include in your visual, and then select the **Create New Content** icon. Use the Data panel to add or remove data tags. Use the Settings panel to choose a visual type and define properties.

Select which data to include on visuals

Decide which data to include in your visuals by adding and removing tags.

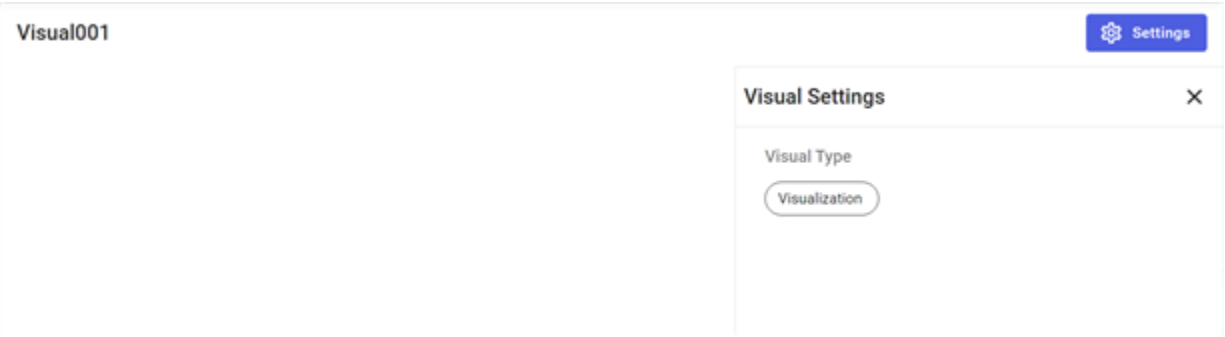
1. Open an existing visual or create a new one.
2. Select **Data** to open the visual editor**.**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. To hide a tag, select the color bar to the left of its name. Select it again to unhide it.
4. To remove tags:
   a. Select **Edit**.
   b. Check the box next to the tags you want to remove.
   c. Select **Remove**.

Choose a visual type

Select **Settings** to open the **Content Settings** panel. From here, select the **Visualization** visual type.

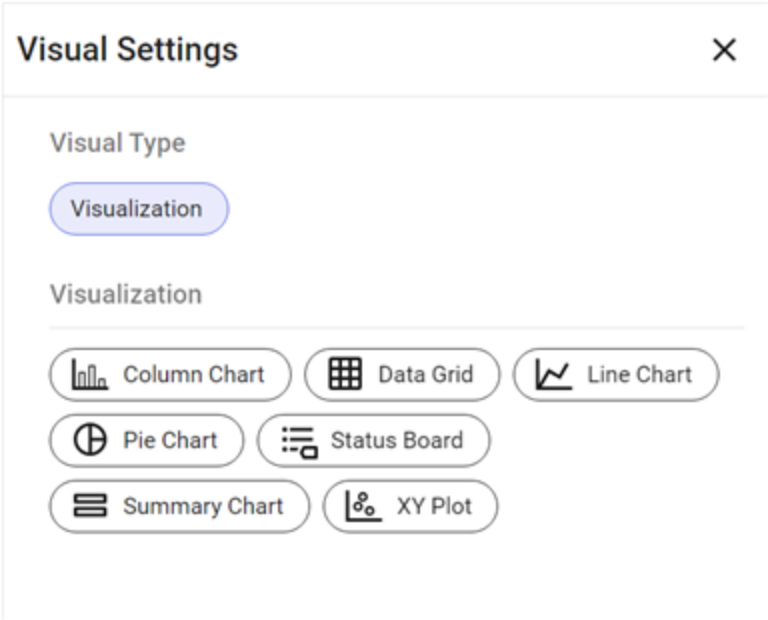


Visualization content types

Visualization content types

The **Visualization** visual content type provides several chart and graphic options. Each includes properties that allow you to customize how the data is presented.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Column Chart

Column Chart

The following properties are available for **Column Charts**:

- **Legend** - Enables the chart legend.
- **Statistic** - Sets the retrieval mode for this chart. Select from:
  - Average
  - Minimum
  - Maximum
  - Last
  - Total
  - Meter

    For detailed information on these retrieval modes, see Understand retrieval mode.
- **Column Type** - Sets a column type. Select from:
  - Hourly
  - Daily
  - Event
  - Custom Duration
    - Once set, specify a duration in the **Custom Direction** field. This field accepts a time aggregation value in *HH:MM* format.
  - Time In State

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

    - Supports string and discrete tags only.
- **Stacking** - Specifies a stacking type. Select from:
  - None
  - Raw
  - Percent

Data Grid

Data Grid

The Data Grid visualization option lets you view data in a table format. To access the Data Grid option, select the Visualization type when creating a new visual. Use the Settings pane to configure the grid. For example, set the retrieval mode using the Statistic option. For details about retrieval modes, see Understand retrieval mode. You can also specify the time aggregation.

Line Charts

Line Charts

Line charts support analog, digital/discrete, and string tags.

The following properties are available for **Line Charts**:

- **Legend** - Enables the chart legend. Only available with *Single* and *Multi* Y-Axis types.
- **Y-Axis** - Sets the Y-Axis type on a line chart. Select from:
  - *Single*
  - *Multi*
  - *Stacked*
- **Auto Scale** - When set, the scaling of the Y-Axis will be determined by the range of values from within the selected time period.

Pie Charts

Pie Charts

The Pie Chart visualization option lets you view data in a pie chart. To access the Pie Chart option, select the Visualization type when creating a new visual. Use the Settings pane to configure the chart. For example, set the retrieval mode using the Statistic option. For details about retrieval modes, see Understand retrieval mode.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Status Boards

Status Boards

The following properties are available for **Status Boards**:

- **Current Value -** Enables the Current Value column.
- **Trend -** Enables the Trend column.
- **Sparkline -** Enables the Sparkline column.

Summary Chart

Summary Chart

The following properties are available for Summary charts.

- **Statistic** - Sets the retrieval mode for this chart. Select from:
  - Average
  - Minimum
  - Maximum
  - Last
  - Total

    For detailed information on these retrieval modes, see Understand retrieval mode.
- **Sort By** - Sets the sorting method for tags included in the chart. Select from:
  - Largest to Smallest
  - Smallest to Largest
  - A - Z

XY Plot

XY Plot

The XY Plot content option is a scatter chart that lets you see correlations between a single tag (X-Axis) to group of tags (Y-Axis). For example, see the relationship between temperature and pressure.

When selecting tags, only analog tags are available and the last tag selected is used as the X-axis.

Although the chart doesn't show time, the data points are taken from the same point in time so that you can see correlations. When there is a stored value at a given point in time, the stored value is used. However, when there

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

isn't a stored value for a given point in time, the value is calculated using the linear or stairstep interpolation method.

To access the XY Plot option, select the Visualization content type when creating new content. Then customize the presentation of your data from the Content Settings pane. For example, change which tag to use as the X-axis, autoscale either axis, and show lines between data points.

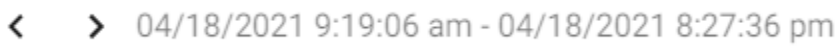
Interact with Charts

You can interact with charts while in the content editor view.

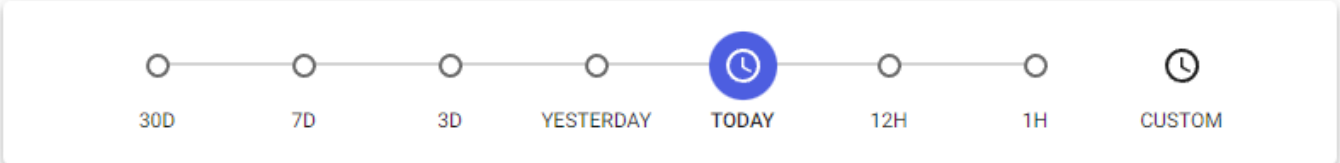
Some functionality, like zooming in and out, can also be used on a chart embedded in a Dashboard.

Adjust the range of time for your data set

To adjust the range of time in your data set, use the arrows to move forward and backward through available time ranges.



You can also use the timeline bar to select a predefined specific period of time, or select **Custom** to define your own.



Pan on line charts

To pan on a line or stacked line chart, select and then drag left or right on the chart. Any changes made to the displayed range by panning can be retained by selecting **Save**.

Zoom on Line Charts or Stacked Line Charts

To zoom in on a specific time range in a line or stacked line chart in the **Content Editor** view, select and then drag across a range of time to zoom in on. Release your mouse button to zoom in. To zoom back out, select **Reset Zoom**.

Note: Using the zoom function alters the time range displayed on the chart. If there are more than 10,000 visible tags when you zoom in, the line chart stops displaying string and discrete tags, and you'll receive a notification asking you to reduce your time range.

You can also use your mouse scroll wheel to zoom. Scroll up to zoom in, scroll down to zoom out. Select **Reset Zoom** to restore the original zoom level. Zooming with the mouse scroll wheel also works with charts embedded

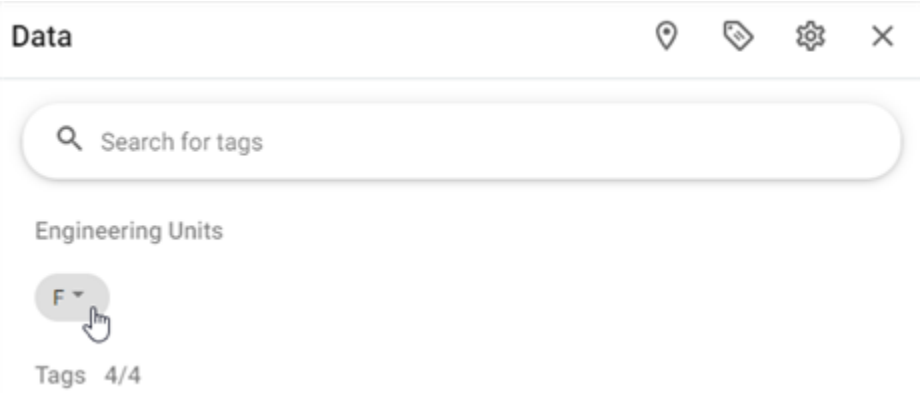
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

in a dashboard.

With both scroll methods, you can save the current zoom level by setting the desired zoom and selecting **Save**.

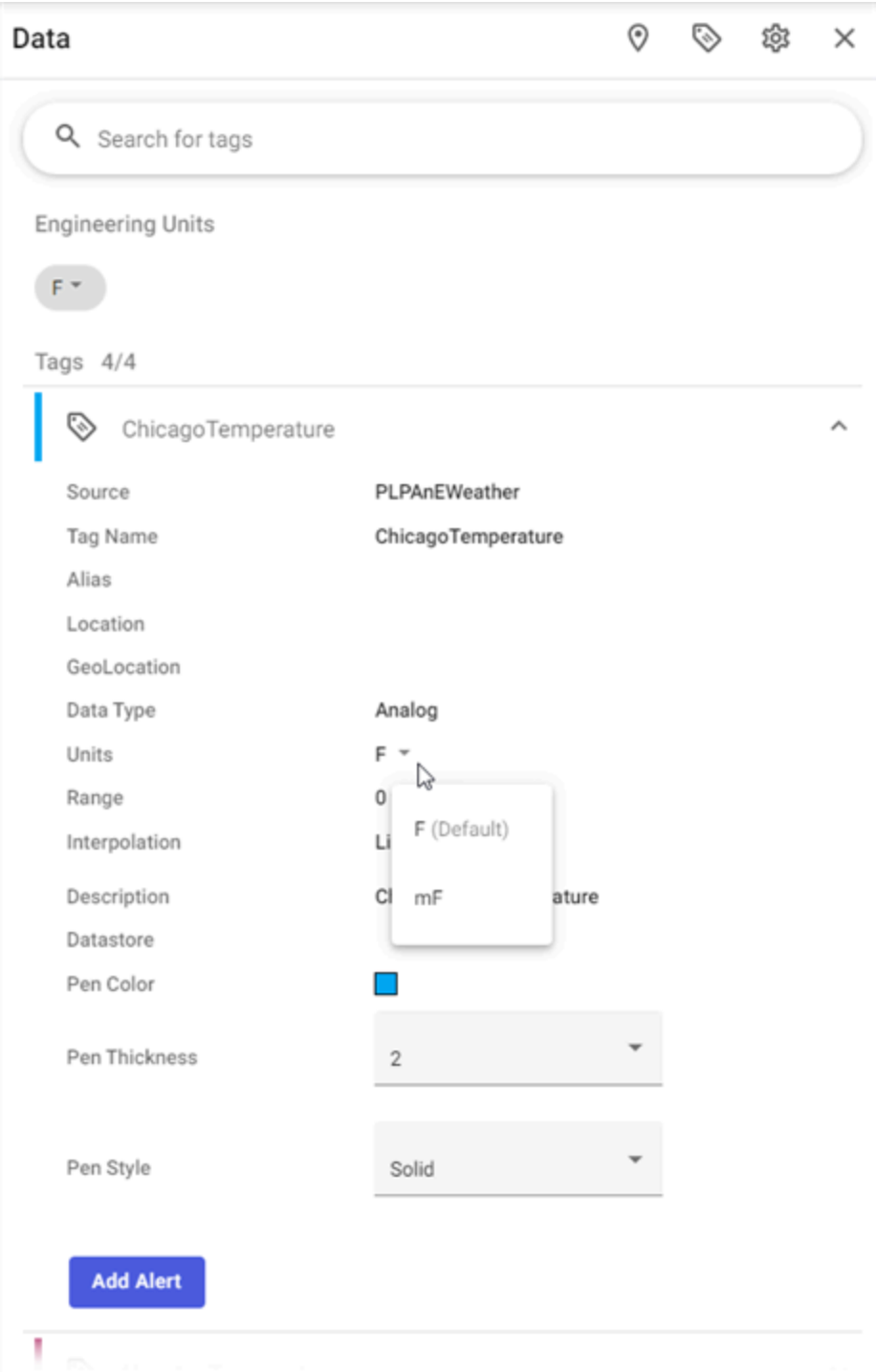
Change engineering units

If engineering units are defined for tags included in your data set, **Engineering Units** will appear below **Search** on the **Data** panel. Here, you can see all engineering units included with your current data set.



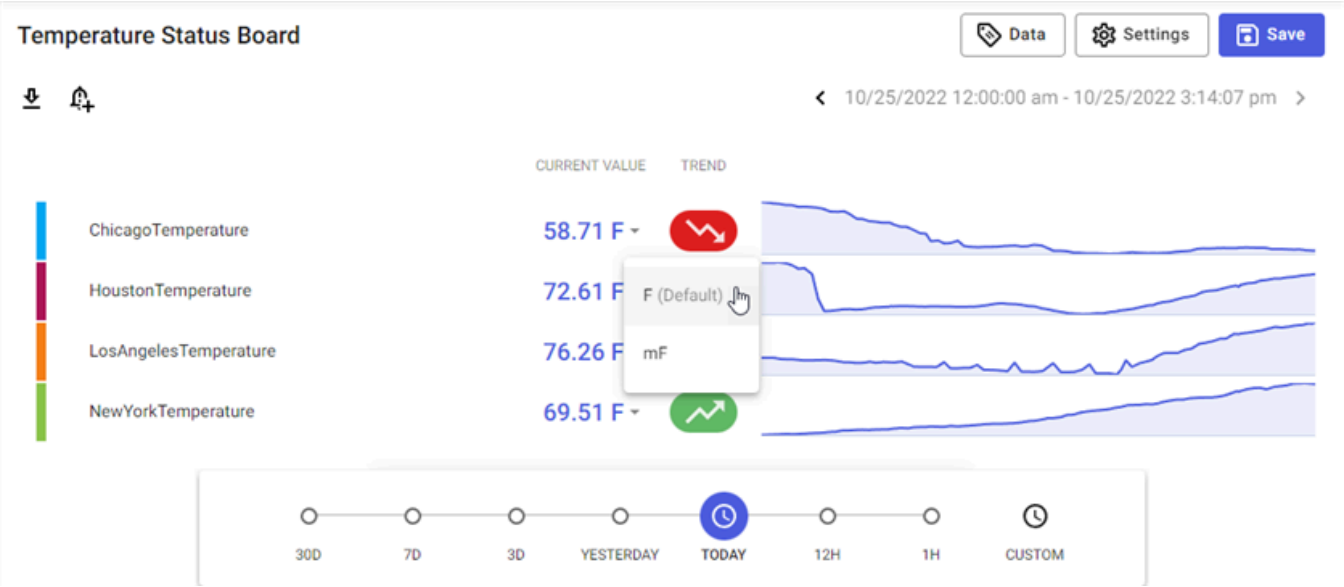
Select the unit you wish to convert, then select the unit you wish to convert to from the list. To convert the unit for an individual tag, expand the tag details and select the desired unit from the **Units** field.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In a status board, you can select engineering unit conversions under the **Current Value** column.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Use Dual Cursors on Line Charts

Using dual cursors on a line chart allows you to calculate the data between two frames. Historian Client Web will do calculations of the data based on the position of the cursors. It calculates the start/end cursors, then calculates the maximum, minimum, average, and delta value based on the position of the cursors.

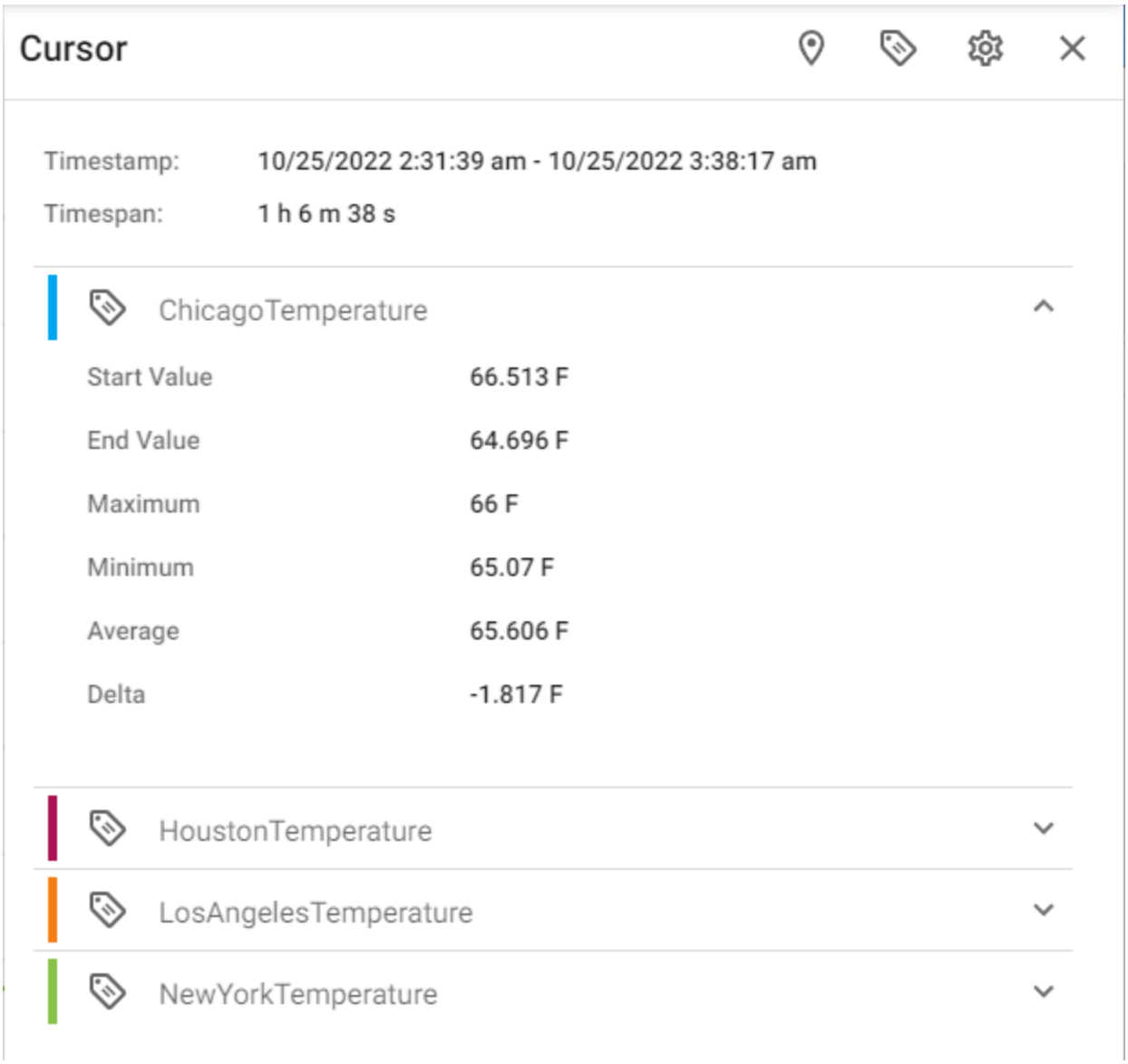
To begin, select the **Add Cursor** icon in the top left corner of the line chart. You can then select two spots on the line chart to place the cursors.

Note: You will not be able to view any data in the **Cursor** panel until two cursors have been placed on the chart.

Select **Cursor** from the top right of the screen to open the **Cursor** panel.

Expand a data tag to view the start/end values, maximum, minimum, average, and delta information.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



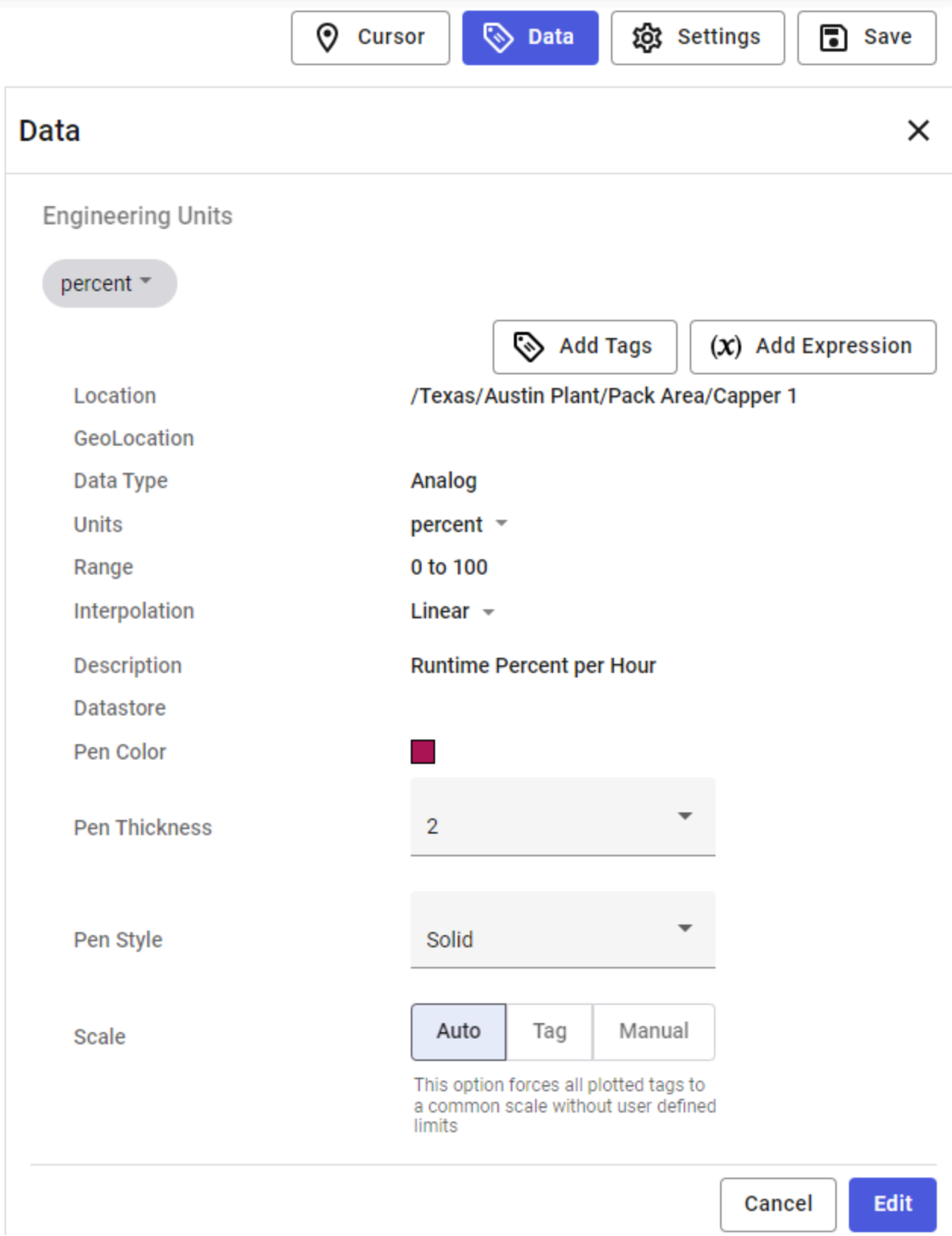
You are not able to save your dual cursor position. When you switch the **Y-Axis** type between **Single** and **Multi**, the cursors will remain in place. However, if you switch to another content page the cursor positions will not be saved.

Scale the Y-Axis

You can use the **Scale** fields to manually scale the y-axis to better represent the data on display. This option is only available for the **Line** chart and the y-axis type must be set to **Stacked** or **Multi**.

To set the **Scale**, open the **Data** panel, then select a tag. From here, toggle the **Scale** checkbox and then type in the range you'd like to see in the **Min** and **Max** fields.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

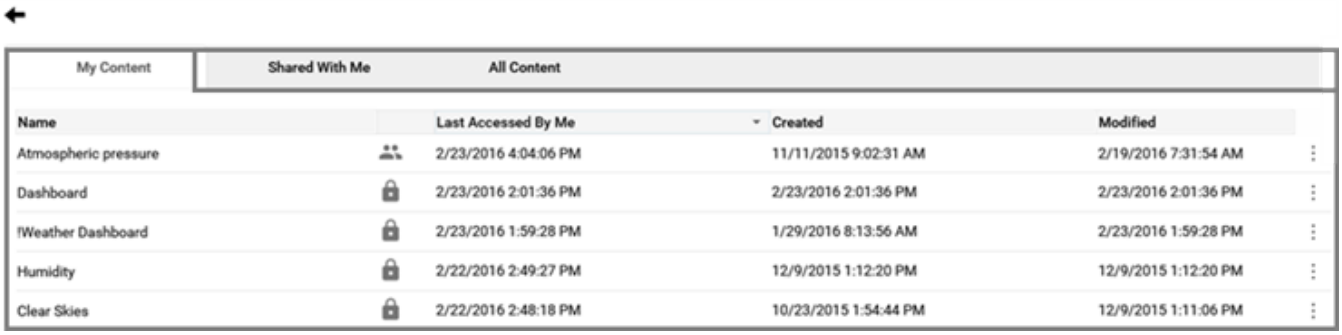


You can save your changes by closing the **Data** panel and then selecting **Save**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Manage existing content

To manage existing content, select **Content** from the navigation rail, then select **Manage Content**. Manage content



From here, you can:

- Select a tab to view content you've saved, analysis shared with you by others, and all content (both yours and shared with you).
- From the My Content tab, see which content you've shared:
  - means shared with the rest of your team.
  - means not shared, but saved for your private use.
- For any specific content, select to choose an action:
  - **Rename** to give the shared content a different name. (You must be the owner to do this.)
  - **Make a copy** to save a duplicate with a new name.
  - **Remove** to delete the item from the Recent list. (You must be the owner to do this.)

    Note: When you remove content, you also remove any unique keywords associated with that content.
- Select to return to the Home page.

Chart data with expressions

Expressions allow you to perform mathematical and logical operations on tag data.

What is an expression?

An expression is made up of a combination of the following elements:

1. Values
   - A value is made up of three components; a value with a base type of Boolean, integer, or double, along with a date and time in UTC format, and a 16-bit integer representing the OPC quality of the value.
   - Literal values are also supported, which have no date and time associated with them, and are always assumed to be "good" quality.
2. Streams
   - A stream is a time-ordered vector of values. A stream has either a stair-step or linear interpolation model, and an associated engineering unit.
   - In the context of an expression, one or more tags is the starting point for a stream, but the expression

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

can produce additional streams derived from a tag. For example, the expression `TI101.PV + 4.0` produces a result stream, but the starting point is the stream associated with `TI101.PV`.

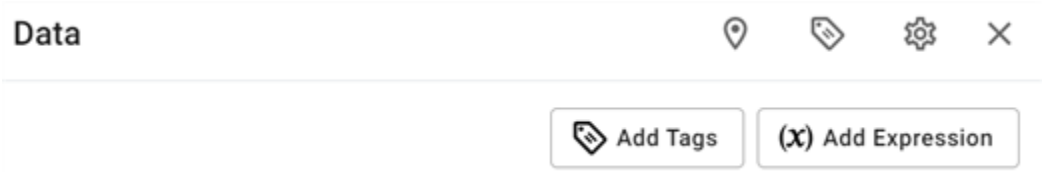
3. Arithmetic and logical operators
   - Standard arithmetic operators such as addition, subtraction, multiplication and division are supported, along with logical operators for comparison, such as greater than (>) and less than (<). For a full list of supported operators, see Supported arithmetic operators.
4. Scalar functions
   - A variety of scalar functions are supported for manipulating data, such as calculating the absolute value, rounding values to the nearest integer, or unit of measure conversion. For a full list of supported scalar functions, see Supported scalar functions.
5. Time-series functions
   - A variety of time-series functions are supported for comparing data, such as calculating minimum/maximum values, or time in state. For a full list of supported functions, see Supported time-series functions.
   - Time intervals can be expressed in either of two ways:
     a. As a number of days. For example, a value of 1 represents a daily period, while a value of 1/24 represents an hourly period. This is consistent with the way Microsoft Excel handles delta time values.

        Note: Daily time periods support Daylight Savings Time. This means the "1 day" period is 24 hours, except on days when the Daylight Savings Time change is made, when it will be either 23 or 25 hours.

     b. As a named period. The following named periods are supported: *minute*, *hour*, *day*, *week*, *month*, *productionday*, *productionweek*, *productionmonth*, *utcday*, *utcweek*, *utcmonth*, *shift*. These periods are anchored to a rounded time period. For example, "hour" produces hourly intervals where the minutes, seconds and milliseconds are all 0 in local time, while "minute" produces intervals where the seconds and milliseconds are all 0 in local time.

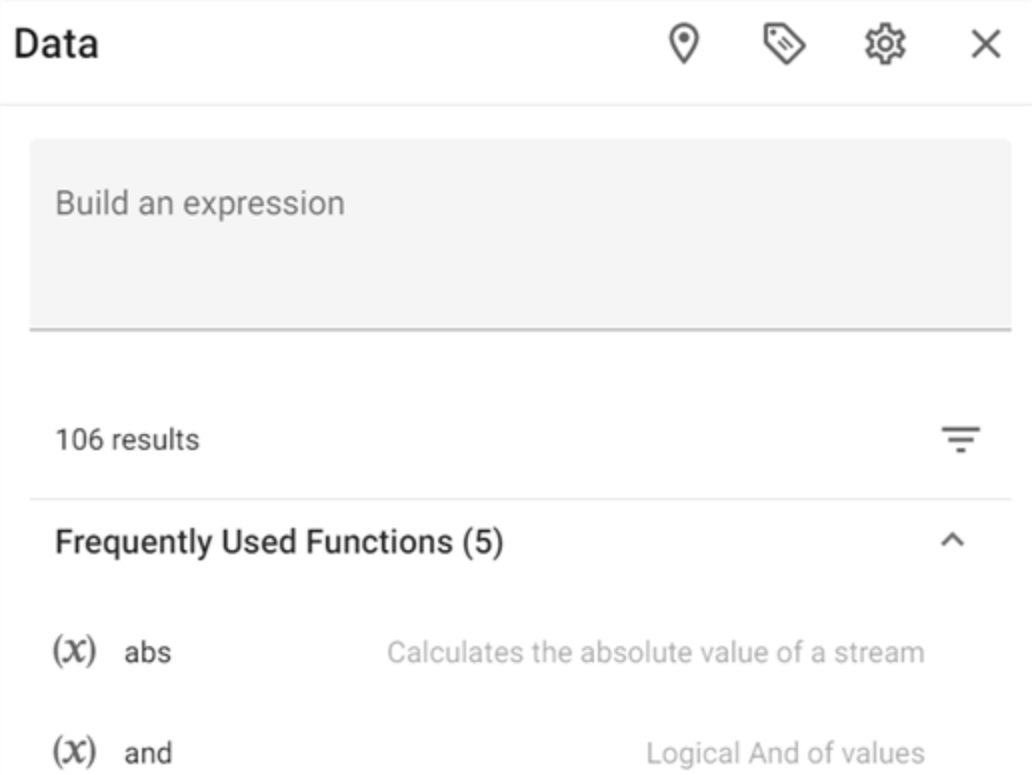
Add an expression

To add an expression, open the **Data** panel and select **Add Expression**.



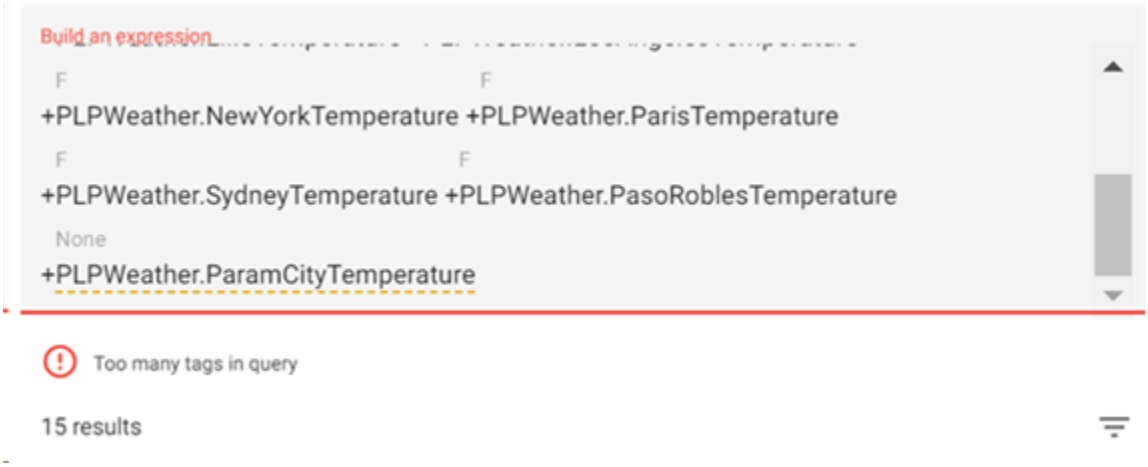
In this mode, the search results list displays suggestions for completing expressions, in addition to data matching your search term. Enter your expression by typing into the search bar. You can also select an item from the list of suggestions to insert it into your expression.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



As you enter your expression into the search bar, Historian Client Web recognizes the use of functions, and offers you suggestions for completing the function by showing you the various options available to you as you type. Other suggestions can include functions, local tags, and data, depending on the context of your expression as you build it. The input field will expand as your expression gets longer so that you are able to view everything without having to scroll. However, you will only be able to add 10 tags to your expression. If you add more than 10 tags, you will get an error, and you will be unable to save your expression.

Note: You can select the function in the search results and view the online help for that particular function.



Once you have completed the expression, select **Save expression**. If the expression is valid, it is added to your content. The **Save expression** will appear greyed out until the expression is valid.

Note: You can only add up to five expressions per chart.

To return to standard search mode, close the data panel and then re-open it.

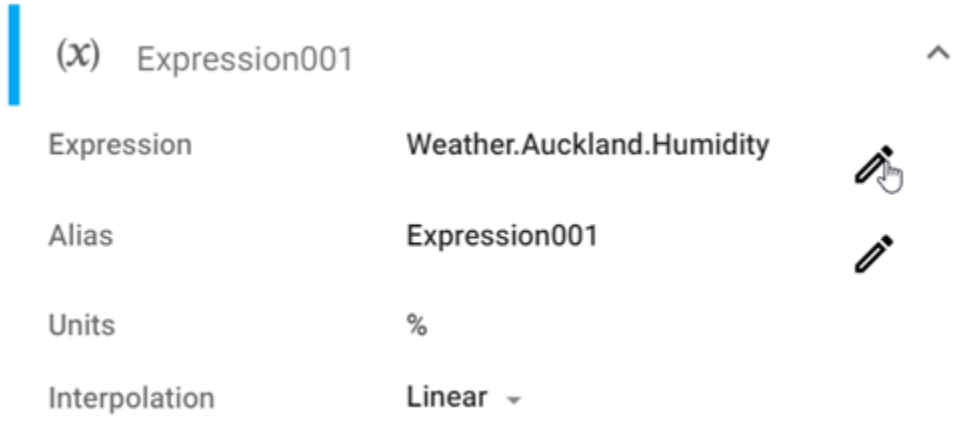
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Edit the expression fields

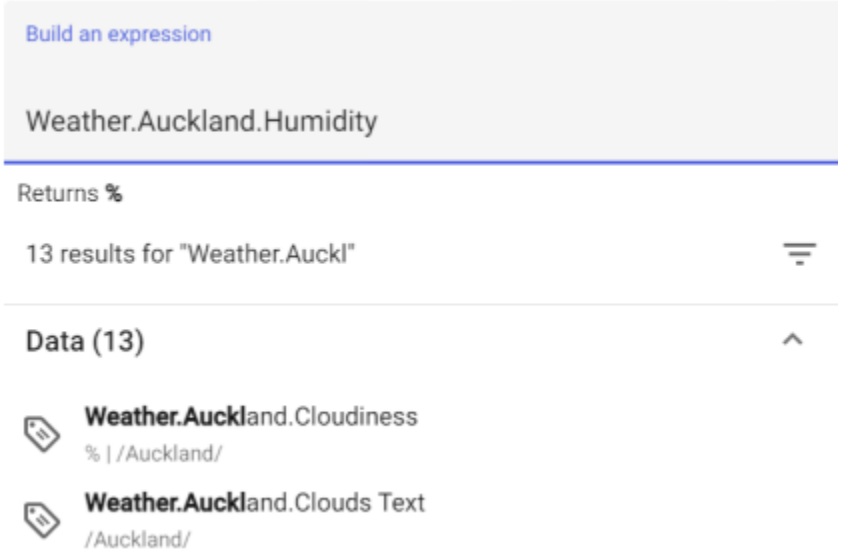
You can edit an expression after you have added it to your chart. To edit an expression, expand the item you want to edit and select ✎. You can edit either the **Expression** name or **Alias**.

Editing the Tag Name field

Select ✎ next to the tag name you want to edit.



Your expression will appear in the search field, where you can make any edits or updates to the expression. You can place your cursor at any point in the tag you want to replace and the search panel will update based on the cursor location.

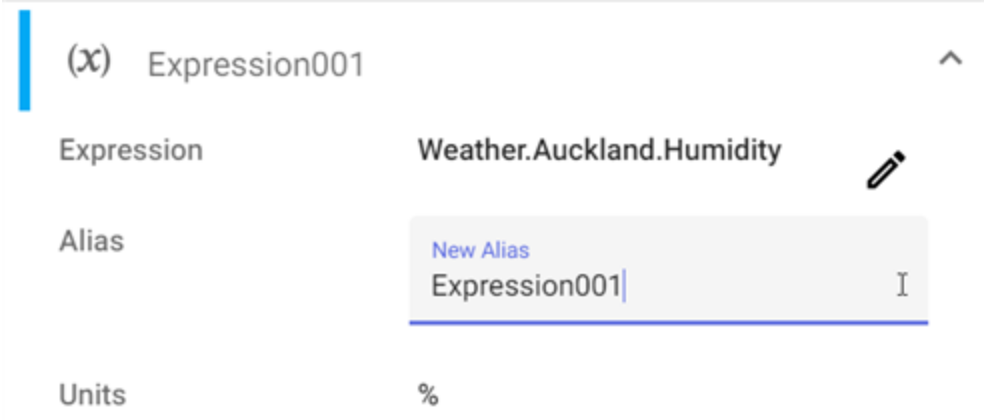


Press **Enter** or select **Save expression** when you are done editing. The field will be updated.

Editing the Alias field

Select ✎ next to the alias you want to edit.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



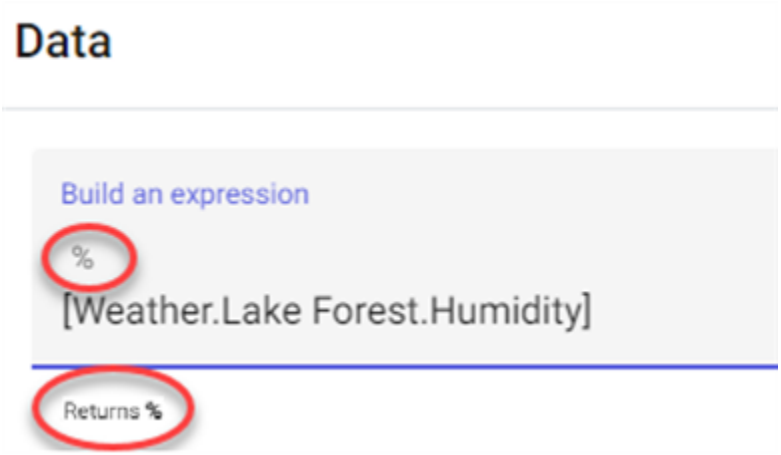
When creating a new alias, it's important to note that:

- You can't use the same name as another alias.
- You can't use special characters.
- Your alias can't be more than 512 characters.

Once you have updated the **Alias** field, press **Enter**.

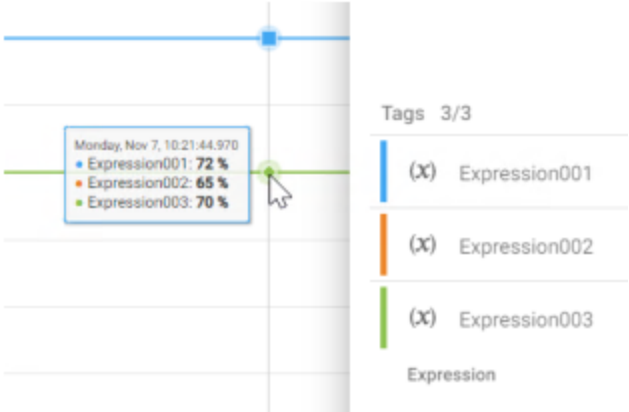
Viewing the Units

The **Units** field displays the engineering units used for your expression.



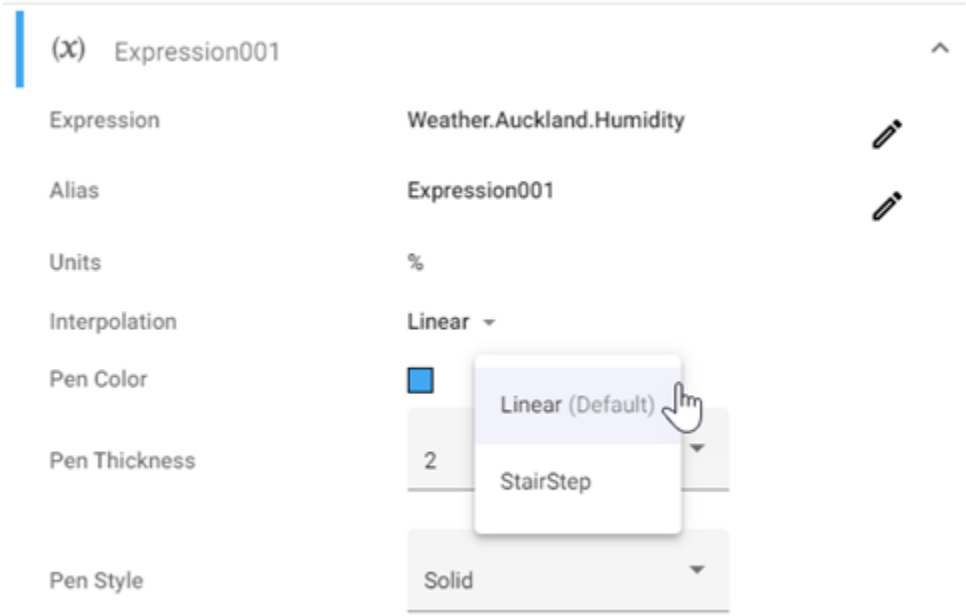
The tooltip in the chart and the **Cursor** panel will display the units as well.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Selecting the Interpolation type

You can override the interpolation settings defined with expression tags using the **Interpolation** field. This option is available for expression tags displayed on the Line Chart only.

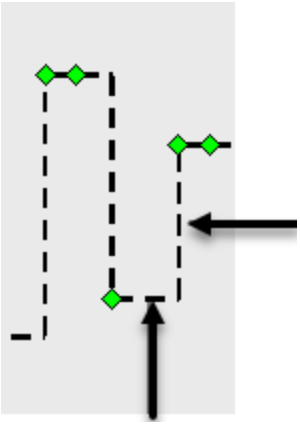


The following options are available:

Note: (Default) is used to indicate the default **Interpolation** setting of the data tag. This can be used to reset the data tag to the original state, if needed.

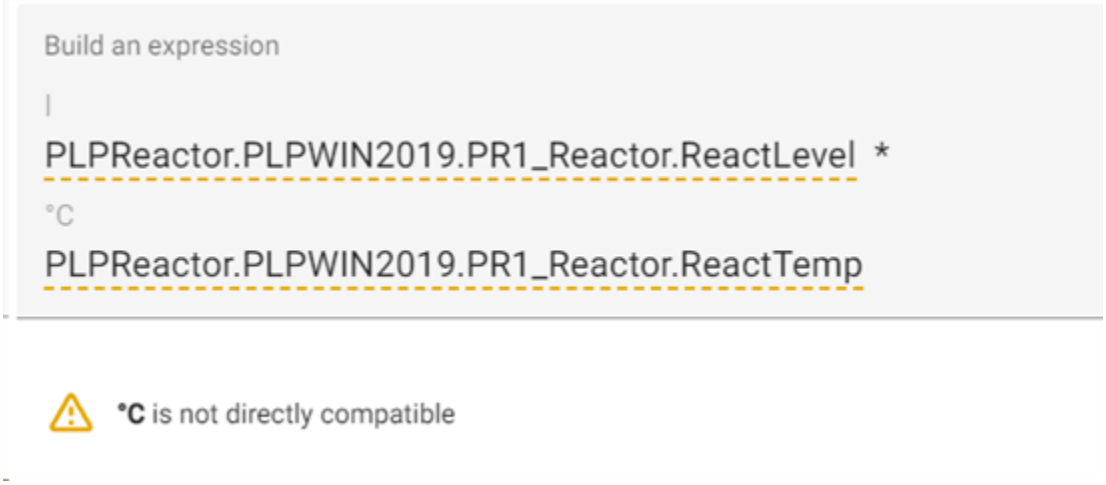
- **Linear**: Shows a smooth line. The value between two points is extrapolated in a linear line.
- **Step Last (StairStep)**: Shows a step line. The horizontal part of the right-angle is drawn from the previous value, and the vertical part indicates the next value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Expression error indicator

Help text will appear below the function expression search box to identify any errors that may occur when entering your expressions.



Error indicator examples:

- If you enter two arithmetic operators in a row (for example, ++) you will get an error stating "`Tag name or stream function expected`".
- If you only have one parameter (a function requires two parameters) you will get an error stating "`Invalid argument for function`".
- If you enter an invalid function (for example, `averages1`) you will get an error saying "`Undefined function`".
- If you use string tags you will get an error stating "`String tag type not allowed`".
- If you fail to provide a unit you will get an error stating "`Unit expected`".
- The second argument for the average function is a time interval, if that is not provided you will get an error stating "`Time interval expected`".
- If a parenthesis is missing you will get an error stating "`Parenthesis expected`".
- If two tags are added and separated by a space you will get an error stating "`Arithmetic or logical operator expected`".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- If an incorrect tag is added you will get an error stating `"Tag not found"`.

  Note: When there is more than one tag and if the unit of other tag is compatible with unit of another tag, it will be underlined in a green color. If the unit is not compatible, it will be underlined in an orange color.

Implicit unit conversion

The following table describes the relationships between engineering unit dimensions when they are multiplied or divided by each other.

Factor1	Factor2	Product
Data Rate	Time	Data
Density	Volume	Mass
Energy Density	Volume	Energy
Length	Area	Volume
Length	Length	Area
Mass Flow	Time	Mass
Pressure	Area	Force
Pressure Rate	Time	Pressure
Resistance	Current	Electric Potential
Speed	Time	Length
Temperature Rate	Time	Temperature
Unknown Rate	Time	Unknown
Volumetric Flow	Time	Volume
Radiant Exposure	Time	Irradiance
Rate	Time	Ratio
Rotation Speed	Time	Rotation
Electric Potential	Power	Current
Acceleration	Mass	Force
Force	Speed	Power
Mass	Specific Energy	Energy

Each row in the table corresponds to four equivalent dimension relationships:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- *Dimension 1 * Dimension 2 = Result Dimension*
- *Dimension 2 * Dimension 1 = Result Dimension*
- *Result Dimension / Dimension 1 = Dimension 2*
- *Result Dimension / Dimension 2 = Dimension 1.*

When working with expressions involving tags or streams with different engineering units, the engineering units of the results are determined as follows:

1. When adding or subtracting streams with units in the same dimension, all streams are first converted to the same unit as the first stream, and then the operations are performed. The results are in the same unit as the first stream. For example, using the expression `SysTimeHour + SysTimeMin`, both tags use units in the *Time* dimension. When the expression is evaluated, the units from the first tag (hours) are used, so `SysTimeMin` is first converted to hours, and the results are in hours.
2. When adding or subtracting streams with one or more streams having "undefined" units, the results are in the same unit as the first stream.
3. When multiplying or dividing a dimensionless stream or numeric constant with a stream that has an associated unit, the results are in that unit.
4. When multiplying or dividing streams where both units are related by one of the entries in the *DimensionMath* table, the result is in the associated result dimension's unit.
5. When multiplying streams where both units are in the same dimension, the scale factors of the two units are multiplied together. If the scale factor matches a unit in the result dimension, the results are in this unit. Otherwise, the original units are first converted to their dimension's base units, and the results are in the base units of the result dimension.
6. Any other combinations cannot be implicitly converted, so the results are in "undefined" units.

   Note: AVEVA Historian 2023 updates the base units for some dimensions to enable implicit unit conversion. If you have additional Historian nodes configured for replication, the engineering unit catalog will be out of sync on tier 2 nodes until they are upgraded to the 2023 version.

Interpolation with Expressions

When plotting data with expressions, the system determines the interpolation type for the expression based on the interpolation type of the individual tags in the expression.

If the interpolation type in the tag metadata is linear then linear interpolation is used, otherwise stair-step interpolation is used. For more information, see *Interpolated Retrieval* in the *AVEVA Historian Retrieval Guide*.

Special rules apply when using some expression functions:

1. An expression using the SIGN function uses stair-step interpolation.

   For example, Pump101.Flow is an analog tag representing the flow rate for a pump. While the tag uses linear interpolation, the expression `SIGN(Pump101.Flow)` uses stair-step interpolation.
2. An expression using the ROUND function uses stair-step interpolation.

   For example, Tank100.Level is an analog tag representing the level of fluid in a tank. While the tag uses linear interpolation, the expression `ROUND(Tank100.Level)` uses stair-step interpolation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Supported arithmetic operators

Historian Client Web supports the use of standard mathematical operators in expressions. The following operators are supported:

Symbol	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
%	Modulo
( )	Operator grouping

Examples

The following examples show expressions that use arithmetic operators. The tag names used in these examples are:

- *FlowRate.Pump1* measures the flow rate of a pump in liters per second (L/s)
- *FlowRate.Pump2* measures the flow rate of a pump in liters per minute (L/min)
- *TankCapacity* measures the available capacity of a tank in kiloliters (kL)
- *TimeElapsed* measures elapsed time in seconds (s)

Example 1

The following expression returns the sum of flow rates for two pumps, and assigns the alias *TotalFlow* to the expression.

```
Pump1.FlowRate + Pump2.FlowRate
```

Because the tags use different units, and there is no explicit unit conversion specified, the units of *Pump2.FlowRate* are implicitly converted to liters/second to match the units of *Pump1.FlowRate* before the values are added together.

Example 2

The following expression returns a value per minute representing the total volume flowing through both pumps during that minute.

```
TOTAL(Pump1.FlowRate + Pump2.FlowRate, 1 minute)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 3

The following expression calculates a pump's efficiency as a percentage. This example uses water at 50 degrees Celsius, which has a density of 992.24 kg/m^3 and head of 3.5m, the gravitational constant of 9.81 m/s^2, and "Pump1.Power" represents the power used by the pump in Watts.

```
Pump1.FlowRate * 3.5 * 992.24 * 9.81 / 100.0 / Pump1.Power
```

Example 4

The following expression returns the travel time in seconds for material in a pipe. The pipe has a diameter of 0.15m, and a length of 100m.

```
3.1415 * 0.15 ^ 2.0 * 100 / UOM(Pump1.FlowRate, m3/sec)
```

Supported logical operators

Historian Client Web supports the use of standard mathematical operators in expressions. The following operators are supported:

Symbol	Operation
=	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

Supported logical functions

In addition to logical operators, Historian Client Web to supports the use of logical functions in expressions. The following logical functions are supported:

Function	Usage
AND(*arguments*)	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.
OR(*arguments*)	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.
NOT(*argument*)	Evaluates the argument to a Boolean *TRUE* or *FALSE* value, and returns the opposite.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AND

Format	*Stream* = **AND**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.

OR

Format	*Stream* = **OR**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.

NOT

Format	*Stream* = **NOT**(*expression*)
Arguments	Any expression.
Description	Evaluates the expression to a boolean *TRUE* or *FALSE* value, then returns the opposite.

Supported scalar functions

AVEVA Historian Client Web supports using scalar functions in expressions. These functions have a 1:1 relationship between the values in the input and output streams.

The following scalar functions are supported:

Function	Purpose
ABS	Calculates the absolute value of a stream.
CASTUOM	Changes a stream's unit of measure without changing the values.
IF	Produces a stream based on the results of a logical expression.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IFBAD	Overrides "bad" quality values in a stream.
IFLONGER	Evaluates whether or not a logical expression is true over a period greater than or equal to a specified time interval, and returns a TRUE or FALSE value or stream.
IFSHORTER	Evaluates whether or not a logical expression is true over a period shorter than a specified time interval, and returns a TRUE or FALSE value or stream.
IFSTALE	Replaces values from the input stream that are older than a specified interval with an alternative value.
LOG	Calculates the base 10 logarithm of a stream's values.
ROUND	Rounds a stream's values to two decimal places.
SIGN	Replaces a stream's values with 1 or -1 based on the current value's sign.
SQRT	Calculates the square root of a stream's values.
TIMESHIFT	Shifts the time stamps in a stream by a specified time interval.
TRUNCATE	Removes the fractional portion of a stream's values.
UOM	Converts a stream's values to a specified unit of measure.

ABS

The ABS() function produces a stream of results calculating the absolute value of a stream, converting negative values into positive values.

Syntax

Format	*Stream* = **ABS**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all negative values converted to positive values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ABS Examples

ABS Examples

Example 1

The following expression returns the difference between the flow rates of two pumps.

```
ABS( Pump101.Flow - Pump102.Flow )
```

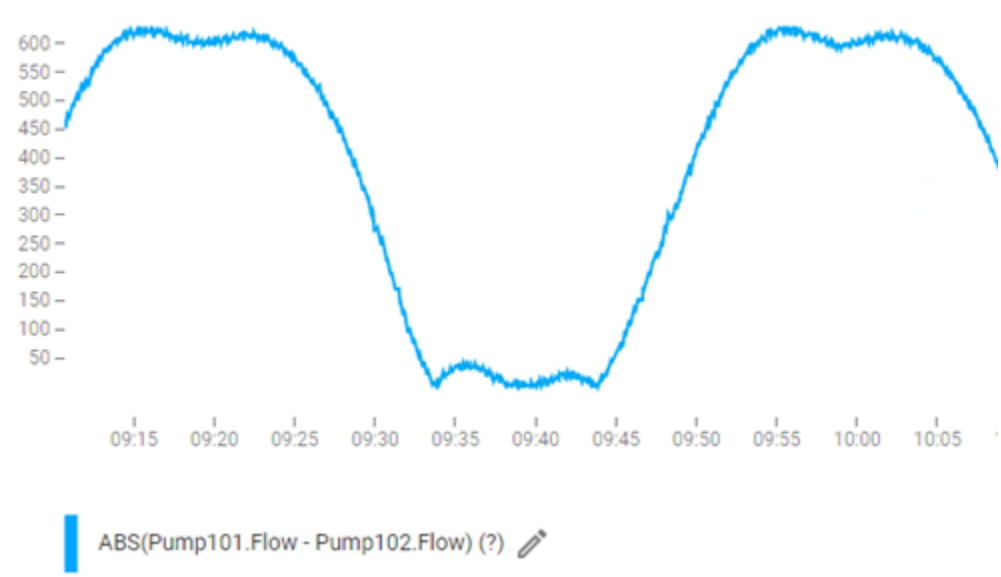
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( Pump101.Flow - Pump102.Flow )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:07:51.4320000	259.439523422241	192
2022-03-31 09:07:53.2590000	268.941925048828	192
2022-03-31 09:07:55.2590000	264.847595214844	192
...	...	...
2022-03-31 10:07:49.2590000	457.803894042969	192

Represented as a line chart over a 1 hour interval:



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression produces a oscillating stream of values from 0 to 30, and back to 0:

```
ABS( SysTimeSec - 30 )
```

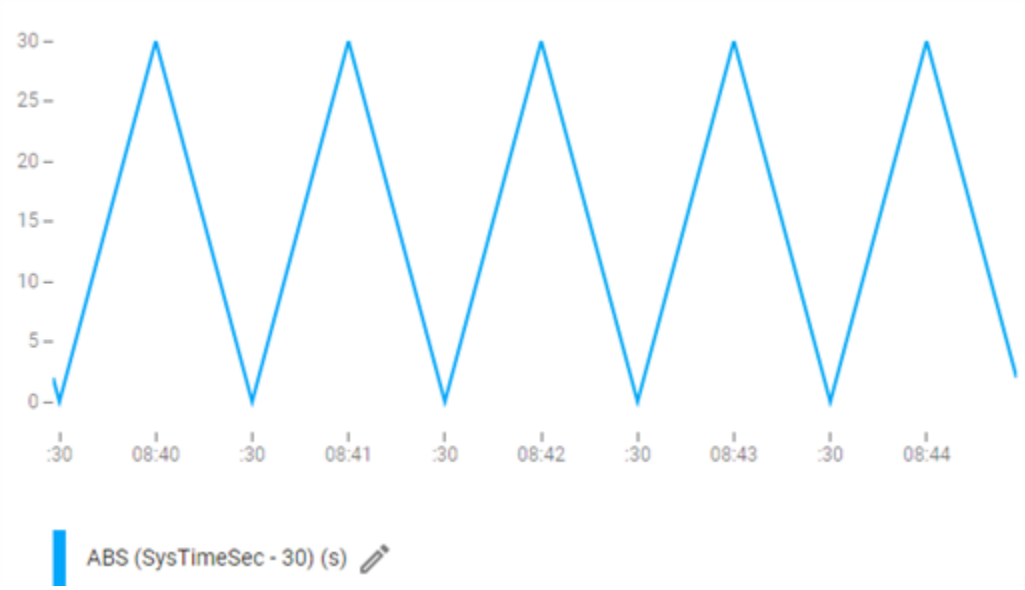
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( SysTimeSec - 30 )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:47:52.3500000	22	192
2022-03-31 09:47:53.0000000	23	192
2022-03-31 09:47:54.0000000	24	192
...	...	...
2022-03-31 10:47:51.0000000	21	192

Represented as a line chart over a 5 minute interval:



CASTUOM

The CASTUOM() function changes a stream's unit of measure to the specified unit of measure without any data conversion, preserving the original values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **CASTUOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the the provided stream to use the specified unit of measure while preserving the existing values.

IF

The IF() function produces a stream based on the results of a logical expression.

Syntax

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*, *FalseStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE *FalseStream* - A stream of values from which to select when the expression evaluates to FALSE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected for the output stream. Otherwise, the corresponding value from the *FalseStream* is selected.

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	for the output stream. Otherwise, a NULL or "bad quality" value is selected.

Format	*Stream* = **IF**(*LogicalExpression*)
Arguments	*LogicalExpression* - A logical expression to be evaluated
Description	Produces an output stream of Boolean values representing the results of *LogicalExpression*.

IF Examples

IF Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, but any negative values are replaced with 0.

```
IF( Pump101.Flow < 0, 0, Pump101.Flow )
```

Represented as a query:

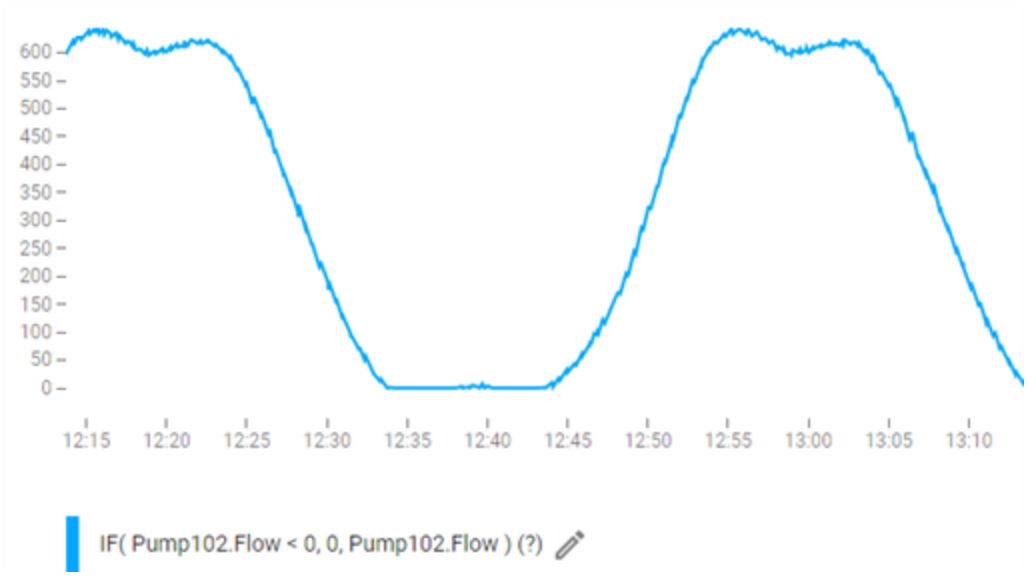
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'IF( Pump102.Flow < 0, 0, Pump102.Flow )'
```

Results over a 5 minute interval:

DateTime	Value	OPCQuality
2022-03-31 12:40:01.2590000	4.00229740142822	192
2022-03-31 12:40:03.2590000	0.365068227052689	192
2022-03-31 12:40:05.2590000	0	192
...	...	...
2022-03-31 12:44:59.2590000	27.5106182098389	192

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 2

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns 0.

```
IF( SysPerfCPUTotal > 90, 1, 0 )
```

Example 3

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns a NULL value.

```
IF( SysPerfCPUTotal > 90, 1 )
```

Example 4

The following expression returns a true value (1) when system CPU usage exceeds 90%, otherwise it returns a false value (0).

```
IF( SysPerfCPUTotal > 90 )
```

IFBAD

The IFBAD() function replaces values from the input stream that have a bad OPC Quality with an alternative value.

Syntax

Format	*Stream* = **IFBAD**(*Stream*, *AltValue*)
Arguments	*Stream* - A stream of values *AltValue* - A replacement value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	Replaces every bad value in the input stream with *AltValue*.

Format	*Stream* = **IFBAD**(*Stream*, *AltStream*)
Arguments	*Stream* - A stream of values *AltStream* - A stream of alternative values
Description	Replaces every bad value in the input stream with the value from *AltStream* at the corresponding time.

IFBAD Examples

IFBAD Examples

Example 1

The following expression replaces all bad values for the input stream Pump102.Flow with "500."

```
IFBAD(Pump102.Flow, 500)
```

Example 2

The following expression replaces all bad values in the input stream Pump102.Flow with values from the alternative stream Pump115.Flow at the corresponding time.

```
IFBAD(Pump102.Flow, Pump115.Flow)
```

IFLONGER

The IFLONGER() function evaluates whether or not a logical expression is true over a period greater than or equal to a specified time interval, and returns a TRUE or FALSE value or stream.

Syntax

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the boolean TRUE is returned. Otherwise the boolean FALSE is returned.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. If it is not *TRUE*, a "bad quality" value is selected.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*, *FalseStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer *FalseStream* - A stream of values from which to select when the expression does not evaluate to *TRUE* for at least the length of the specified interval
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. Otherwise the corresponding value from *FalseStream* is selected.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IFLONGER Examples

IFLONGER Examples

Example 1

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period longer than or equal to 1 minute. If it does, it returns a value of TRUE. If it doesn't, it returns a value of FALSE.

```
IFLONGER(Pump1.FlowRate > 75.0, 1 minute)
```

Example 2

The following expression evaluates whether or not the stream SysTimeSec remains greater than or equal to 20 for a period longer than or equal to half a minute. If it does, it returns the value 5. If it doesn't, it returns a "bad quality" value.

```
IFLONGER(SysTimeSec >= 20, 0.5 minute, 5)
```

Example 3

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period longer than or equal to 1 minute. If it does, it returns the value from Pump1.Flowrate. If it doesn't, it returns a value of 75.

```
IFLONGER(Pump1.FlowRate > 75.0, 1 minute, Pump1.FlowRate, 75.0)
```

IFSHORTER

The IFSHORTER() function evaluates whether or not a logical expression is true over a period shorter than a specified time interval, and returns a TRUE or FALSE value or stream.

Format	*Stream* = **IFSHORTER**(*StreamLogicalExpression*, *Interval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for less time than the specified *Interval*, the boolean TRUE is returned. Otherwise the boolean FALSE is returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **IFSHORTER**(*StreamLogicalExpression*, *Interval*, *TrueStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for less time than the length of the specified interval
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time less than the specified *Interval*, the corresponding value from *TrueStream* is selected. If it is not *TRUE*, a "bad quality" value is selected.

Format	*Stream* = **IFSHORTER**(*StreamLogicalExpression*, *Interval*, *TrueStream*, *FalseStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for less time than the length of the specified interval *FalseStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for less time than the specified *Interval*, the corresponding value from *TrueStream* is selected. Otherwise the corresponding value from *FalseStream* is selected.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IFSHORTER Examples

IFSHORTER Examples

Example 1

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period shorter than 1 minute. If it does, it returns a value of TRUE. If it doesn't, it returns a value of FALSE.

```
IFSHORTER(Pump1.FlowRate > 75.0, 1 minute)
```

Example 2

The following expression evaluates whether or not the stream SysTimeSec remains greater than or equal to 20 for a period shorter than half a minute. If it does, it returns the value 5. If it doesn't, it returns a "bad quality" value.

```
IFSHORTER(SysTimeSec >= 20, 0.5 minute, 5)
```

Example 3

The following expression evaluates whether or not the stream Pump1.Flowrate remains greater than 75 for a period shorter than 1 minute. If it does, it returns the value from Pump1.Flowrate. If it doesn't, it returns a value of 75.

```
IFSHORTER(Pump1.FlowRate > 75.0, 1 minute, Pump1.FlowRate, 75.0)
```

IFSTALE

The IFSTALE() function replaces values from the input stream that are older than a specified interval with an alternative value.

Syntax

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltValue*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltValue* - A replacement value
Description	Replaces every value in the input stream older than *Interval* with *AltValue*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltStream*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltStream* - A stream of alternative values
Description	Replaces every value in the input stream older than *Interval* with the value from *AltStream* at the corresponding time.

IFSTALE Examples

IFSTALE Examples

Example 1

The following expression replaces every value in the input stream older than 40 minutes with the value -1.

```
IFSTALE(SysTimeHour, 40 minute, -1)
```

Example 2

The following expression replaces every value in the input stream H100.TotalFlow older than 40 minutes with the value from the alternative stream Pump1.FlowRate + Pump2.FlowRate at the corresponding time.

```
IFSTALE(H100.TotalFlow, 1 hour, Pump1.FlowRate + Pump2.FlowRate)
```

LOG

The LOG() function produces a stream of results calculating the base 10 logarithm of a stream's values.

Syntax

Format	*Stream* = **LOG**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values converted to their base 10 logarithm.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LOG Examples

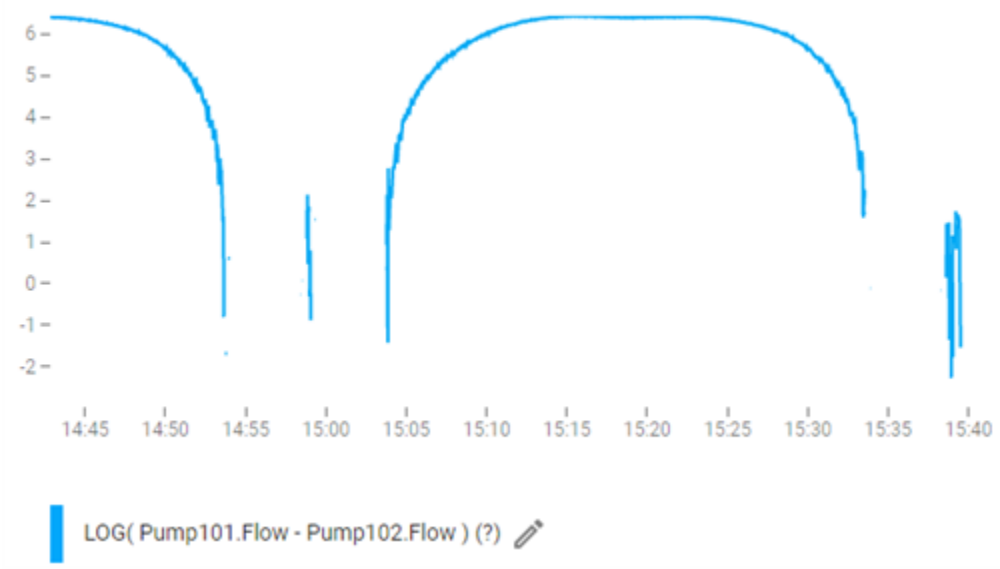
LOG Examples

Example 1

The following expression returns the base 10 logarithm of the difference in flow rates for two pumps.

```
LOG( Pump101.Flow - Pump102.Flow )
```

Represented as a line chart over a 1 hour interval:



ROUND

The ROUND() function produces a stream of results rounding a stream's values to the nearest integer.

Syntax

Format	*Stream* = **ROUND**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values rounded to the nearest integer.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ROUND Examples

ROUND Examples

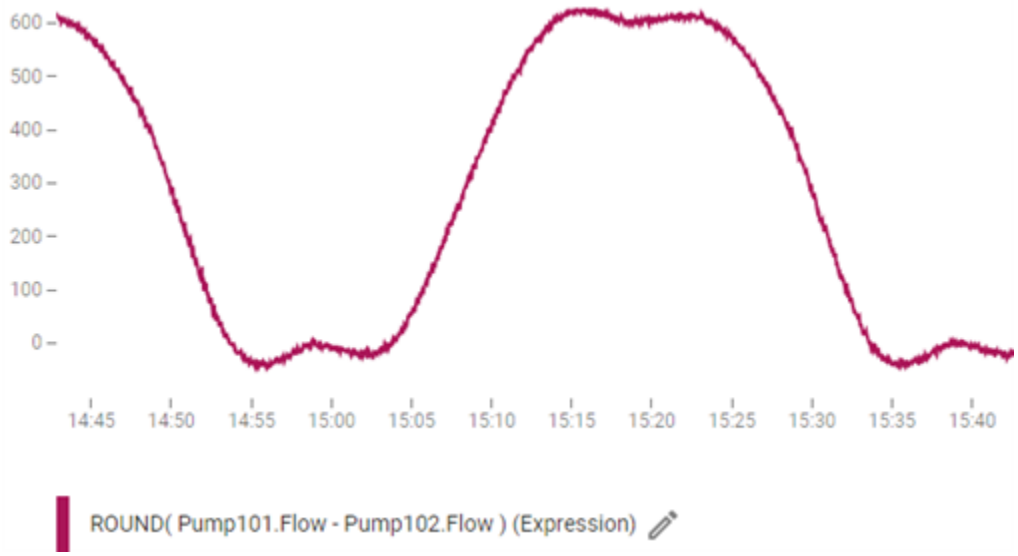
Example 1

The following expression returns the difference in flow rates for two pumps, rounded to the nearest integer.

```
ROUND( Pump101.Flow - Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag.

Represented as a line chart over a 1 hour interval:



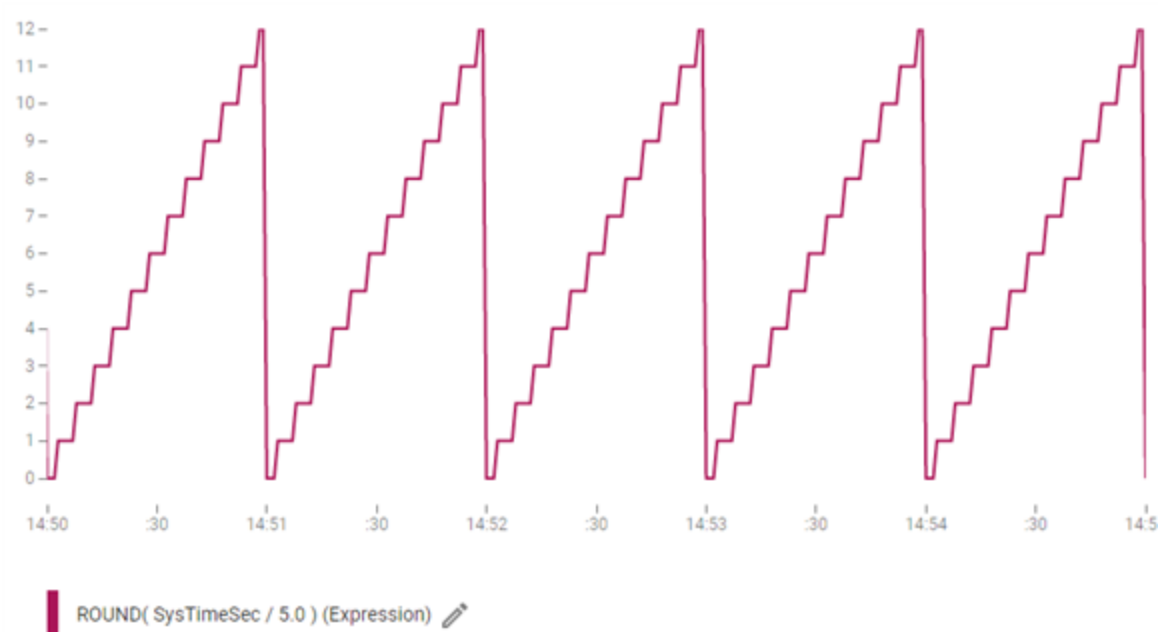
Example 2

The following expression returns the second value of the time, divided by 5 and rounded to the nearest integer.

```
ROUND( SysTimeSec / 5.0 )
```

Represented as a line chart over a 5 minute interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



SIGN

The SIGN() function produces a stream of results indicating the sign of a stream's values.

Syntax

Format	*Stream* = **SIGN**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values replaced based on their sign. Positive values are replaced with 1, negative values are replaced with -1, and zero values remain zero.

SIGN Examples

SIGN Examples

Example 1

The following expression returns the sign of the value of the difference in flow rates between two pumps.

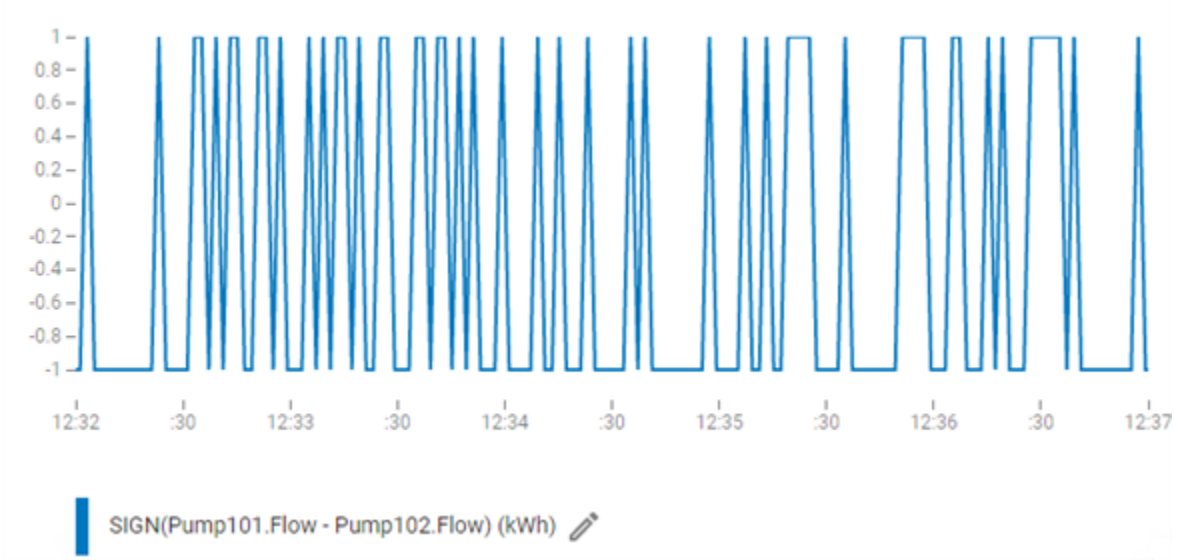
```
SIGN( Pump101.Flow - Pump102.Flow )
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For each data point in the results, the value will be:

- -1 when the second pump has a higher flow rate
- 0 when the flow rates are equal
- 1 when the first pump has a higher flow rate.

Represented as a line chart over a 5 minute interval:



SQRT

The SQRT() function produces a stream of results calculating the square root of a stream's values.

Syntax

Format	*Stream* = **SQRT**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, but with all values being replaced with their square roots.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQRT Examples

SQRT Examples

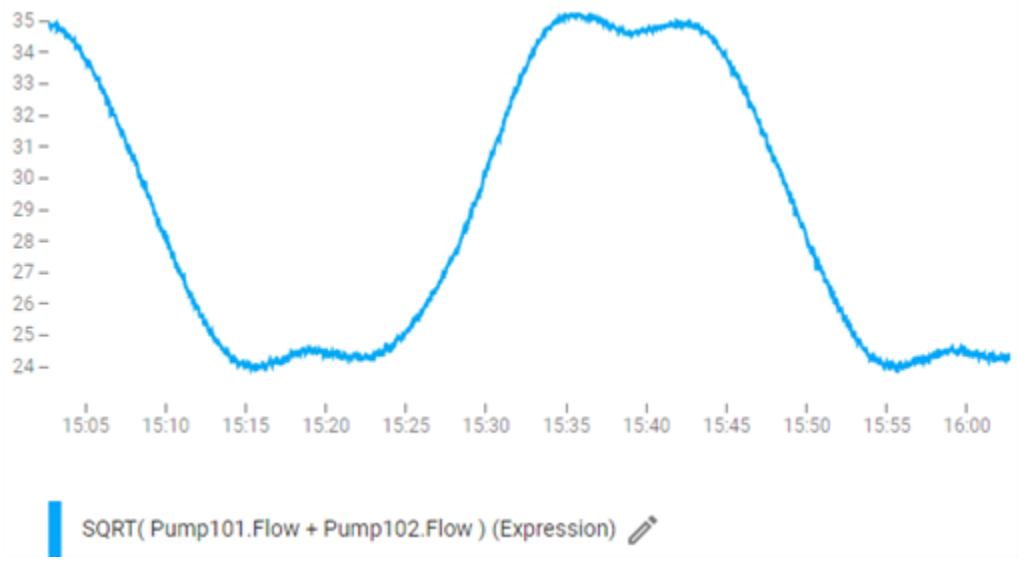
Example 1

The following expression returns the square root of the sum of flow rates for two pumps.

```
SQRT( Pump101.Flow + Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag, so the expression's results are in liters per second.

Represented as a line chart over a 1 hour interval:



TIMESHIFT

The TIMESHIFT() function shifts the time stamps in the input stream by adding or subtracting a specified time interval.

Syntax

Format	*Stream* = **TIMESHIFT**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A stream of values *TimeInterval* - The time period to add (or subtract if negative)
Description	The specified interval of time is added to (or

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	subtracted from) the time stamps in the input stream.

TIMESHIFT Examples

TIMESHIFT Examples

Example

The following expression shifts the time stamp of each flow rate value forward by one hour.

```
TIMESHIFT( Pump101.Flow, Hour )
```

Example 2

The following expression shifts the time stamp of the current time's minute value back by two minutes.

```
TIMESHIFT( SysTimeMin, -2 minute )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TIMESHIFT( SysTimeMin, -2 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:57:00.0000000	59	192
2022-03-31 15:58:00.0000000	0	192
2022-03-31 15:59:00.0000000	1	192
2022-03-31 16:00:00.0000000	2	192
...	...	...

TRUNCATE

The TRUNCATE() function produces a stream of results removing the fractional portion of a stream's values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **TRUNCATE**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with the fractional part of all values removed.

TRUNCATE Examples

TRUNCATE Examples

Example 1

The following expression returns the product of flow rates for two pumps, with the fractional portion of each value removed.

```
TRUNCATE( Pump101.Flow * Pump102.Flow )
```

Represented as a query:

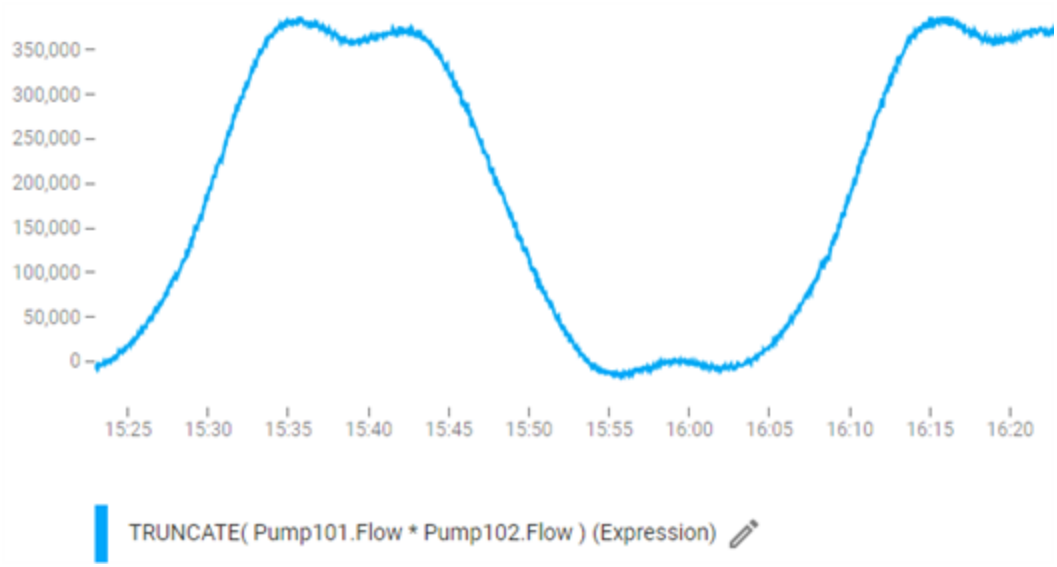
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( Pump101.Flow * Pump102.Flow )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:24:49.2590000	11030	192
2022-03-31 15:24:51.2590000	14974	192
2022-03-31 15:24:53.2590000	14542	192
2022-03-31 15:24:55.2590000	14887	192
...	...	...

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 2

The following expression divides the current time's second value by 5, removing the fractional portion.

```
TRUNCATE( SysTimeSec / 5.0 )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( SysTimeSec / 5.0 )'
AND wwRetrievalMode = 'Full'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:31:39.0000000	7	192
2022-03-31 15:31:40.0000000	8	192
2022-03-31 15:31:41.0000000	8	192
2022-03-31 15:31:42.0000000	8	192
...	...	...

UOM (Unit of Measure conversion)

The UOM() function converts the values in a stream to a specified unit of measure.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **UOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the values in the provided stream to use the specified unit of measure. If the specified unit does not exist, or is not correctly mapped to a catalog unit for conversion, the original unit is returned instead.

UOM Examples

UOM Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, with the units converted to gallons per second.

```
UOM( Pump101.Flow, gal/min )
```

Supported time-series functions

AVEVA Historian Client Web supports using time-series functions in expressions. These functions enable you to compare and manipulate data.

The following time-series functions are supported:

Function	Purpose
AVERAGE	Calculates the time-weighted average for a stream.
AVERAGES	Calculates the statistical average for a stream.
COUNT	Returns a count of the delta points for a stream.
COUNTALL	Returns a count of the data points for a stream.
DURATION	Calculates the duration of *TRUE* states in an

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	expression.
METER	Returns the delta change of a stream's value.
MIN/MAX	Selects the minimum/maximum value of a stream.
PREV/PREVGOOD	Returns the last value from the previous time period.
TOTAL	Calculates the time-weighted total for a stream.
TOTALS	Calculates the statistical total for a stream.

AVERAGE

The AVERAGE() function calculates the time-weighted average value of a stream or array of streams.

When using AVERAGE() in a query, if no resolution is specified, then the behavior is the same as average retrieval mode with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the average value is calculated for each cycle.

Syntax

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the time-weighted average value of a stream or array of streams.

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the time-weighted average of a stream or array of streams over the specified time interval, based on local time. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVERAGE Examples

AVERAGE Examples

Example 1

The following expression returns the average flow rate of a pump.

```
AVERAGE(Pump101.Flow)
```

Example 2

The following expression returns the average flow rate of a pump over 10 minute intervals.

```
AVERAGE(Pump101.Flow, 10 Minute)
```

Example 3

The following expression returns the average flow rate of all streams that include ".Flow" in their tag name at 10 minute intervals.

```
AVERAGE(SELECT("TagName",".Flow"), 10 Minute)
```

Example 4

The following expression returns the average minute value of the time over 5 minute intervals.

```
AVERAGE(SysTimeMin, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'AVERAGE( SysTimeMin, 5 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 12:00:00.0000000	2	192
2022-03-31 12:05:00.0000000	7	192
2022-03-31 12:10:00.0000000	12	192
2022-03-31 12:15:00.0000000	17	192
...	...	...

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVERAGES

The AVERAGES() function calculates the statistical average value of a stream or array of streams over a specified time interval.

Note: This is the preferred method for calculating averages from discrete samples such as those from a checkweigher. For most measurements, however, AVERAGE is preferred.

Syntax

Format	*Stream* = **AVERAGES**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the statistical average for a stream or array of streams over the specified time interval. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

AVERAGES Examples

AVERAGES Examples

Example 1

The following expression returns the average weight of items coming off a line at 10 minute intervals.

```
AVERAGES(Line3.ItemWeight, 10 Minute)
```

Example 2

The following expression returns the average weight of all streams that include "ItemWeight" in their tag name at 10 minute intervals.

```
AVERAGES(SELECT("TagName","ItemWeight"), 10 Minute)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

COUNT

The COUNT() function returns a count of the delta points for a stream. Note that if the quality detail of a point differs from the previous point, this counts as a delta point, even if the value doesn't change.

Syntax

Format	*Stream* = **COUNT**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of delta points during the query period. The query period is based on local time.

Format	*Stream* = **COUNT**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the delta points are counted
Description	Returns a count of delta points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNT Examples

COUNT Examples

Example 1

The following expression returns the number of delta points in a freezer unit's temperature.

```
COUNT(Freezer15.Temperature)
```

Example 2

The following expression returns the number of delta points in a freezer unit's temperature over hour intervals.

```
COUNT(Freezer15.FlowTotal, hour)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

COUNTALL

The COUNTALL() function returns a count of all data points for a stream.

Syntax

Format	*Stream* = **COUNTALL**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of all data points during the query period. The query period is based on local time.

Format	*Stream* = **COUNTALL**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the points are counted
Description	Returns a count of all data points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNTALL Examples

COUNTALL Examples

Example 1

The following expression returns the number of data points reported by a pressure alert on a boiler.

```
COUNT(Boiler03.PressureAlert)
```

Example 2

The following expression returns the number of points reported by a pressure alert on a boiler over a one hour interval.

```
COUNT(Boiler03.PressureAlert, hour)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DURATION (Time in State)

The DURATION() function returns a stream representing the duration in seconds of each "true" state in an expression.

Syntax

Format	*Stream* = **DURATION**(*StreamLogicalExpression*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated
Description	The logical expression represented by *StreamLogicalExpression* is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results.

Format	*Stream* = **DURATION**(*StreamLogicalExpression*, *TimeInterval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *TimeInterval* - The time interval over which to calculate durations
Description	The logical expression represented by *StreamLogicalExpression* is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results, with durations being split across the specified time period. For example, if StreamLogicalExpression evaluates to a *TRUE* value for a time period of 07:55-08:10, and the specified *Interval* is hourly, the output contains one "5 minute" and one "10 minute" result. Without specifying an *Interval*, the output would instead contain one "15 minute" result.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DURATION Examples

DURATION Examples

Example 1

The following expression returns the amount of time in seconds over the specified interval that the number of seconds in the time is 20 or greater.

```
DURATION(SysTimeSec >= 20)
```

Example 2

The following expression returns the amount of time per one hour interval that a tank's level is at or above 500 kL.

```
DURATION(Tank400.Level > 500, hour)
```

Represented as a query:

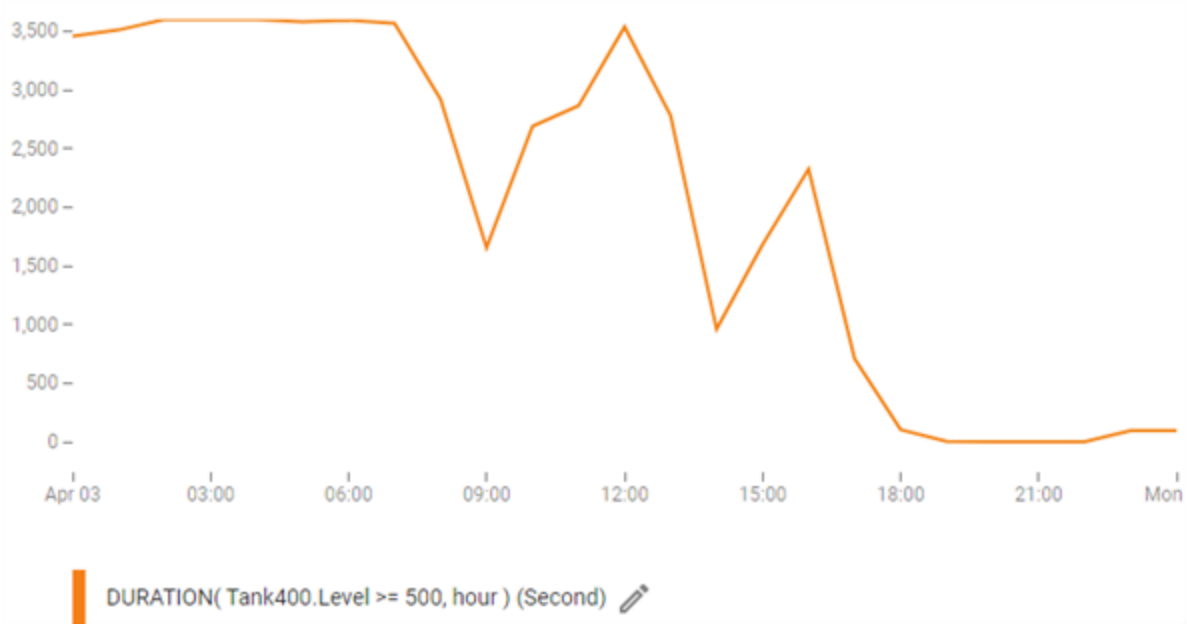
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(Tank400.Level >= 500, hour)'
AND DateTime > '2022-04-03 00:00:00'
AND DateTime < '2022-04-04 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-04-03 01:00:00.0000000	3600	192
2022-04-03 02:00:00.0000000	3578	192
2022-04-03 03:00:00.0000000	3592	192
...	...	...
2022-04-03 23:00:00.0000000	94	192

Represented as a line chart over a 1 day interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 3

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater, over 5 minute intervals.

```
DURATION(SysTimeSec >= 20, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(SysTimeSec >= 20, 5 minute)'
```

Results:

DateTime	Value	OPCQuality
2022-04-01 08:45:00.0000000	200	192
2022-04-01 08:50:00.0000000	200	192
2022-04-01 08:55:00.0000000	200	192
2022-04-01 09:00:00.0000000	200	192
...	...	...

Example 4

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater over 30 second intervals, and the results are converted to minutes.

```
UOM(DURATION(SysTimeSec >= 20, 0.5 minute), Minute)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

METER

The METER() function returns the delta change of the value of a stream or array of streams. This works both for tags without rollover, and tags that reset upon reaching a rollover value.

If an expression includes a single tag, the rollover value defined in the AVEVA Historian will be used. If an expression includes multiple tags, or performs an operation on a tag, a rollover value of 0 is assumed. For example, the following would use the Historian-defined rollover value:

- METER(Tag1, hour)

The following would assume a rollover value of 0:

- METER(Tag1 + Tag2, hour)
- METER(Tag1 + 5, hour)

Otherwise, the METER function works similarly to counter retrieval mode. For more details about rollover, delta change, and common use cases for this function, see Counter Retrieval.

Syntax

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the delta change in the stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which the delta change is calculated.
Description	Returns the delta change in the stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

METER Example

METER Example

Example 1

The following expression returns the changes in a pump's total flow.

```
METER(Pump101.FlowTotal)
```

Example 2

A common use of this function is for counters with rollover. The following expression returns the number of units to come out of a production line at one hour intervals.

```
METER(ProductionLine2.UnitCount, hour)
```

Example 3

The following expression returns the number of units to come out of all streams that include "UnitCount" in their tag name at one hour intervals.

```
METER(SELECT("TagName","UnitCount"), hour)
```

MIN/MAX

The MIN() and MAX() functions produce a stream of results selecting the minimum or maximum value of a stream or array of streams. The returned timestamps correspond to the minimum/maximum values retrieved, rather than the beginning of the time interval.

When using MIN() or MAX() in a query, if no resolution is specified, the behavior is the same as minimum or maximum retrieval modes with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the minimum/maximum values are calculated for each cycle.

Syntax

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the minimum (or maximum) value of a stream or array of streams.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*, *TimeInterval*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which minimum or maximum values are calculated.
Description	Returns the minimum (or maximum) value of a stream or array of streams over the specified time interval.

MIN/MAX Examples

MIN/MAX Examples

Example 1

The following expressions return the minimum (or maximum) flow rate of a pump.

```
MIN(Pump101.Flow)
MAX(Pump101.Flow)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'MIN(Pump101.Flow)'
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Result:

DateTime	Value	OPCQuality
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 2

The following expressions return the minimum (or maximum) flow rate of a pump over 30 minute time intervals.

```
MIN(Pump101.Flow, 30 minute)
MAX(Pump101.Flow, 30 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
WHERE wwExpression = 'MIN(Pump101.Flow, 30 minute)'
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-03-30 00:13:45.2590000	592.065856933594	192
2022-03-30 00:43:07.2590000	592.34765625	192
2022-03-30 01:01:07.2590000	593.351257324219	192
...	...	...
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 3

The following expression returns the minimum flow rate of all streams measured in the volumetric flow dimension over 30 minute time intervals.

```
MIN(SELECT("Dimension","Volumetric Flow"), 30 Minute)
```

PREV/PREVGOOD

The PREV() and PREVGOOD() functions return the last value from the previous time period for the provided stream.

When using PREV() or PREVGOOD() in a query, if no resolution is specified then the query returns an error.

Syntax

Format	*Stream* = **PREV**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns the last value from the previous time period, where the period is specified by *TimeInterval*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **PREVGOOD**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns similar results to **PREV**(), but values with "bad" or "uncertain" quality are skipped. **PREVGOOD** will continue seeking backwards until a "good" value is found, going beyond the previous period if necessary. This means **PREVGOOD** always returns a value with "good" quality.

PREV/PREVGOOD Examples

PREV/PREVGOOD Examples

Example 1

The following expression

```
Line1.Total - PREV( Line1.Total, hour )
```

TOTAL

The TOTAL() function produces a stream of results calculating the time-weighted total for a stream or array of streams.

Syntax

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the time-weighted total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams,*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	TimeInterval)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated.
Description	Calculates the time-weighted total for a stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

TOTAL Examples

TOTAL Examples

Example 1

The following expression returns the total volume flowing through a pump over 1 hour intervals.

```
TOTAL(Pump101.Flow, hour)
```

Example 2

The following expression returns the total volume of all streams measured in the volumetric flow dimension over 1 hour intervals.

```
TOTAL(SELECT("Dimension","Volumetric Flow"), hour)
```

Example 3

The following expression returns the total energy produced by a turbine over the query period.

```
TOTAL(Turbine306.Output)
```

Example 4

The following expression returns the total energy for all streams with a tag name that starts with "Turbine" and ends with ".Output" over the query period.

```
TOTAL(SELECT("TagName","Turbine*.Output"))
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TOTALS

The TOTALS() function produces a stream of results calculating the statistical total for a stream or array of streams.

Syntax

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the statistical total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated
Description	Calculates the statistical total for a stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TOTALS Examples

TOTALS Examples

Example 1

The following expression returns the calculated statistical total volume flowing through a pump over a 10 minute interval.

```
TOTALS(Pump102.Flow, 10 minute)
```

Example 2

The following expression returns the calculated statistical total volume for all streams with a tag name that includes "Flow" over a 10 minute interval.

```
TOTALS(SELECT("TagName","Flow"), 10 minute)
```

Example 3

The following expression returns the calculated statistical total energy produced by a turbine over the query period.

```
TOTALS(Turbine306.Output)
```

Example 4

The following expression returns the calculated statistical total energy for all streams with a tag name that starts with "Turbine" and ends with ".Output" over the query period.

```
TOTALS(SELECT("TagName","Turbine*.Output"))
```

Work with dashboards

Dashboards allow you to collect related pieces of content and save them to a reusable view. You can add charts, maps, and graphics to a Dashboard.

Once created, Dashboards can be shared with other members of your solution or kept private.

Create a dashboard

To get started with Dashboards, either:

- Select **Create** from the navigation rail, then select **Dashboard**.
- Select one or more saved content items from search results, then select **Create Dashboard**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Add content to a dashboard

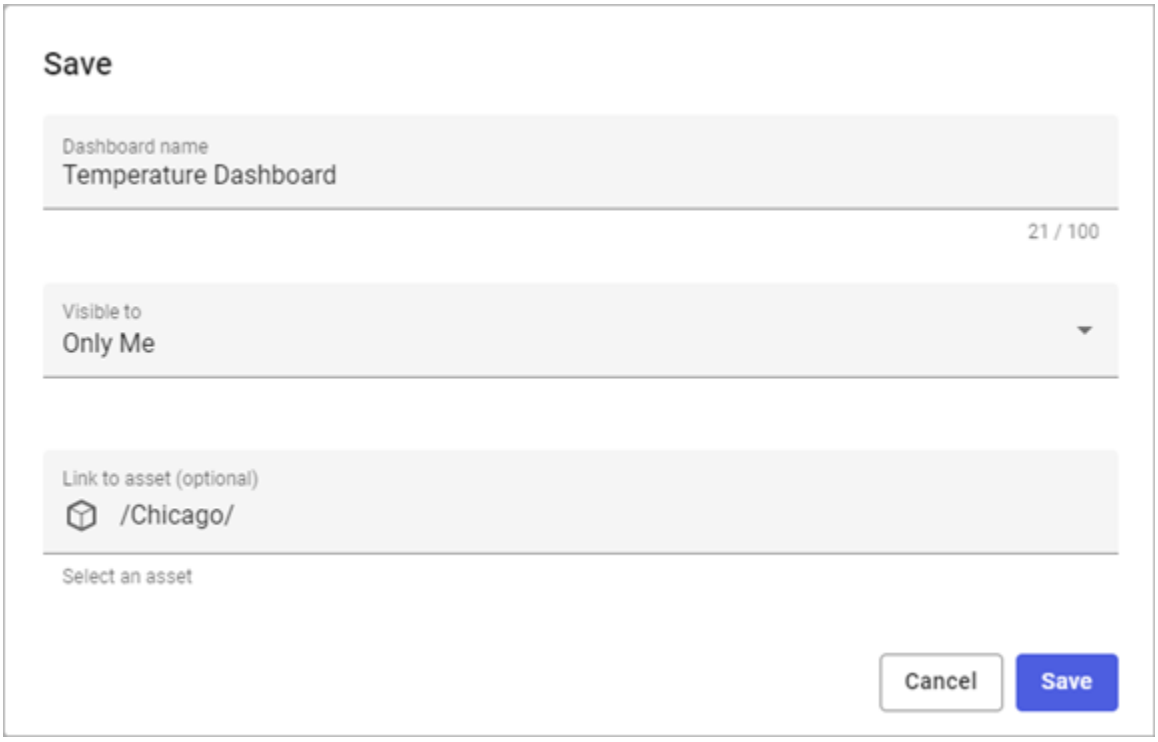
To add content to a dashboard, select **Add** to open the **Content Browser** pane. From here, search for specific content to add using the **Search** field, or select from the suggested content below it.

Content placed on the Dashboard can be rearranged as you see fit by selecting and dragging the content.

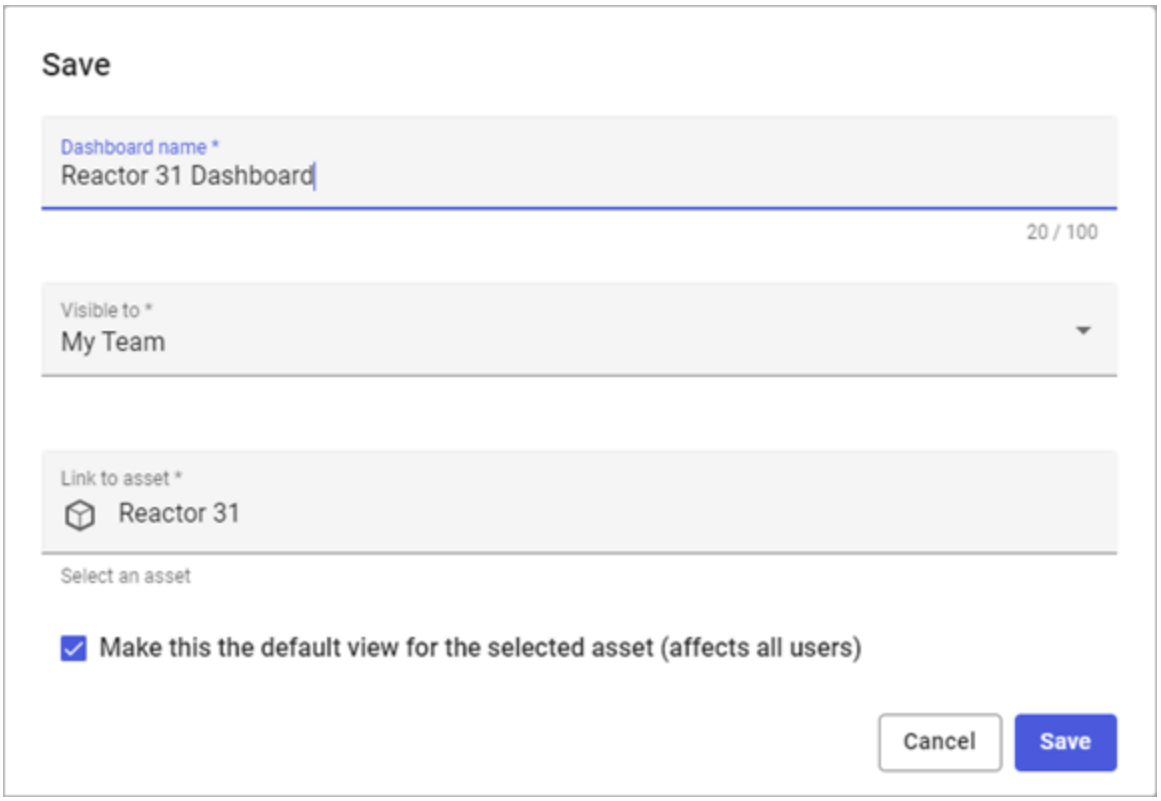
You can set a time range for all content on your dashboard by using the selector at the bottom of the dashboard.

Save and share your dashboard

To save your content, click **Save** to open the **Save** dialog. From here, you can enter a name for your content, specify sharing settings, link your content to a specific asset, and set the dashboard as the default view for the linked asset, depending on the options you select.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



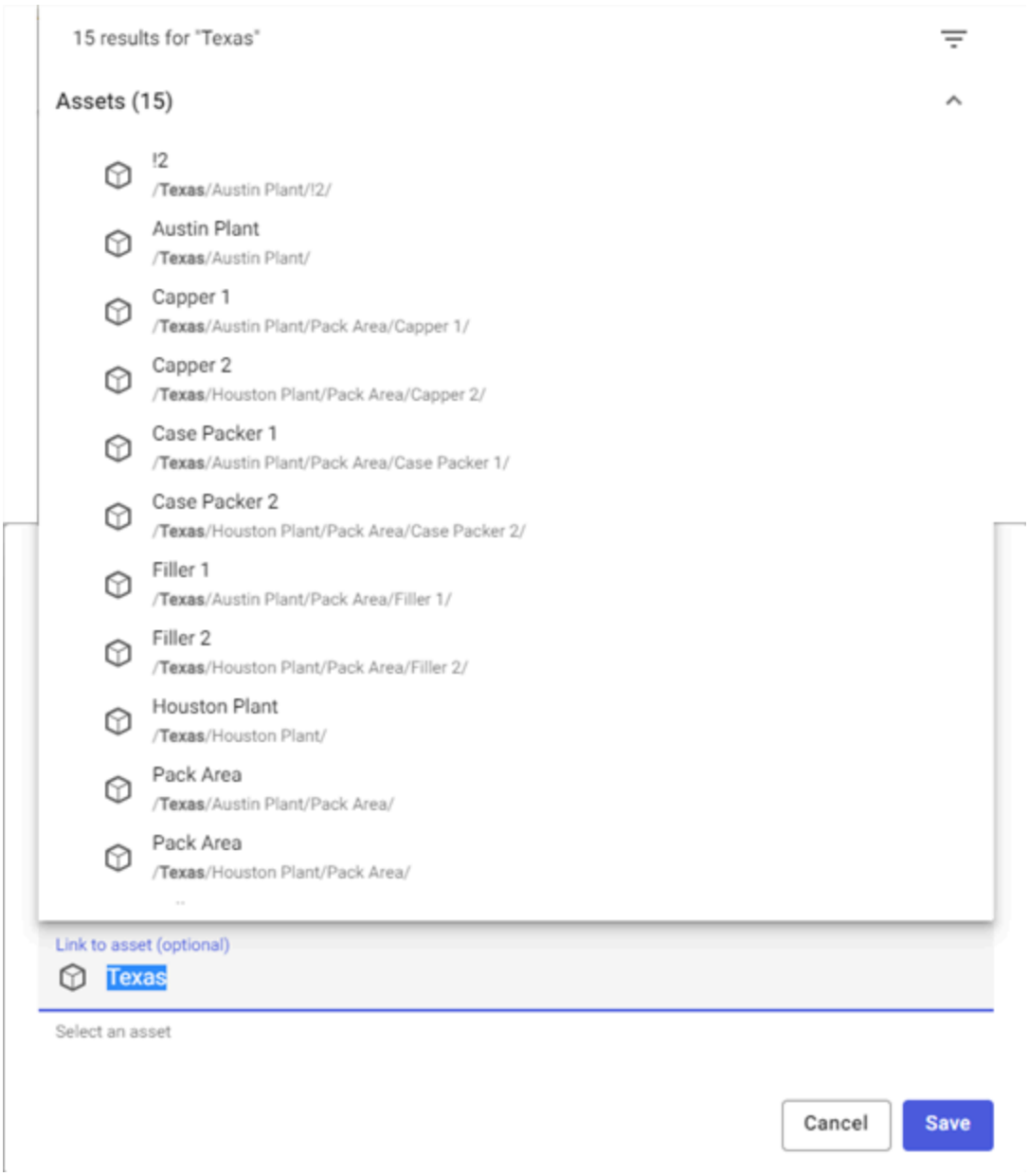
Adjust share settings for your dashboard

Under **Visible to**, you can select *Only Me* to make your dashboard private. Or, you can select *My Team* to share the dashboard with other users in your solution.

Link dashboard to a specific asset

Under **Link to asset**, you can link your dashboard to an asset. This will add a link for the dashboard to the **Associated Content** card of the asset page.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



For an asset to appear in the **Link to asset** list, it must be linked directly, or indirectly through its child assets, to all of the tags included in your content. If no assets meet this condition, the list selection defaults to the solution root.

For example, consider the following *DeviceA* and *DeviceB* assets and their tag assignments:

- */Houston/Houston Plant/Area1/DeviceA*

  This asset has the following tags:

  - *Tag1*
  - *Tag2*
- */Houston/Houston Plant/Area1/DeviceB*

  This asset has the following tags:

  - *Tag3*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- *Tag4*

If the tags included in your content are *Tag1* and *Tag2*, then the following assets will appear in the list:

/Houston

/Houston/Houston Plant

/Houston/Houston Plant/Area1

/Houston/Houston Plant/Area1/DeviceA

DeviceB won't appear in the list because neither of its tags are included in your content.

If the tags included in your content are all four of the device tags, then the following assets will appear in the list:

/Houston

/Houston/Houston Plant

/Houston/Houston Plant/Area1

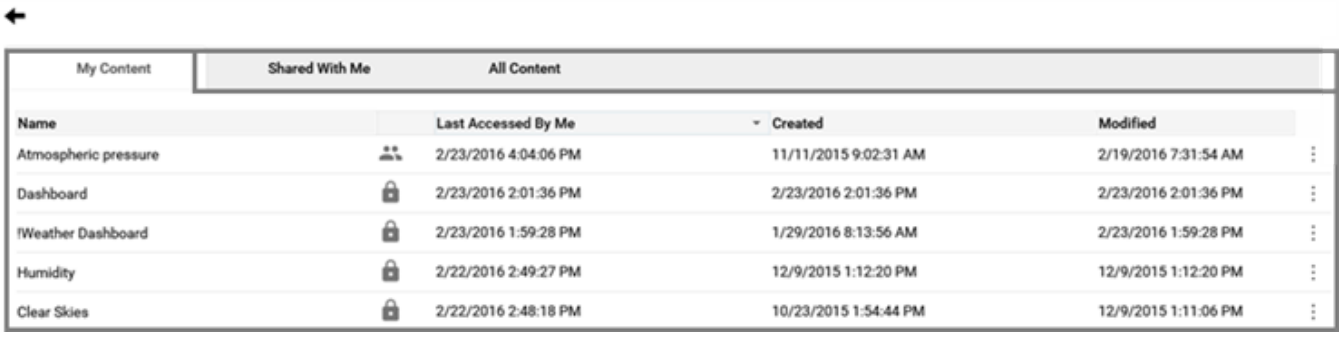
DeviceA and *DeviceB* won't appear in this list because neither of them are linked to the other device's tags, either directly or indirectly.

Set the dashboard as the default view for the asset

If you make the dashboard visible to your team and link it to an asset, you have the option to set this dashboard as the default view for the asset. If you select **Make this the default view for the selected asset (affects all users)**, this dashboard will display first when the asset is opened by any user.

Manage existing content

To manage existing content, select **Content** from the navigation rail, then select **Manage Content**. Manage content



From here, you can:

- Select a tab to view content you've saved, analysis shared with you by others, and all content (both yours and shared with you).
- From the My Content tab, see which content you've shared:
  - means shared with the rest of your team.
  - means not shared, but saved for your private use.
- For any specific content, select to choose an action:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Rename** to give the shared content a different name. (You must be the owner to do this.)
- **Make a copy** to save a duplicate with a new name.
- **Remove** to delete the item from the Recent list. (You must be the owner to do this.)

  Note: When you remove content, you also remove any unique keywords associated with that content.

- Select ← to return to the Home page.

View alarms

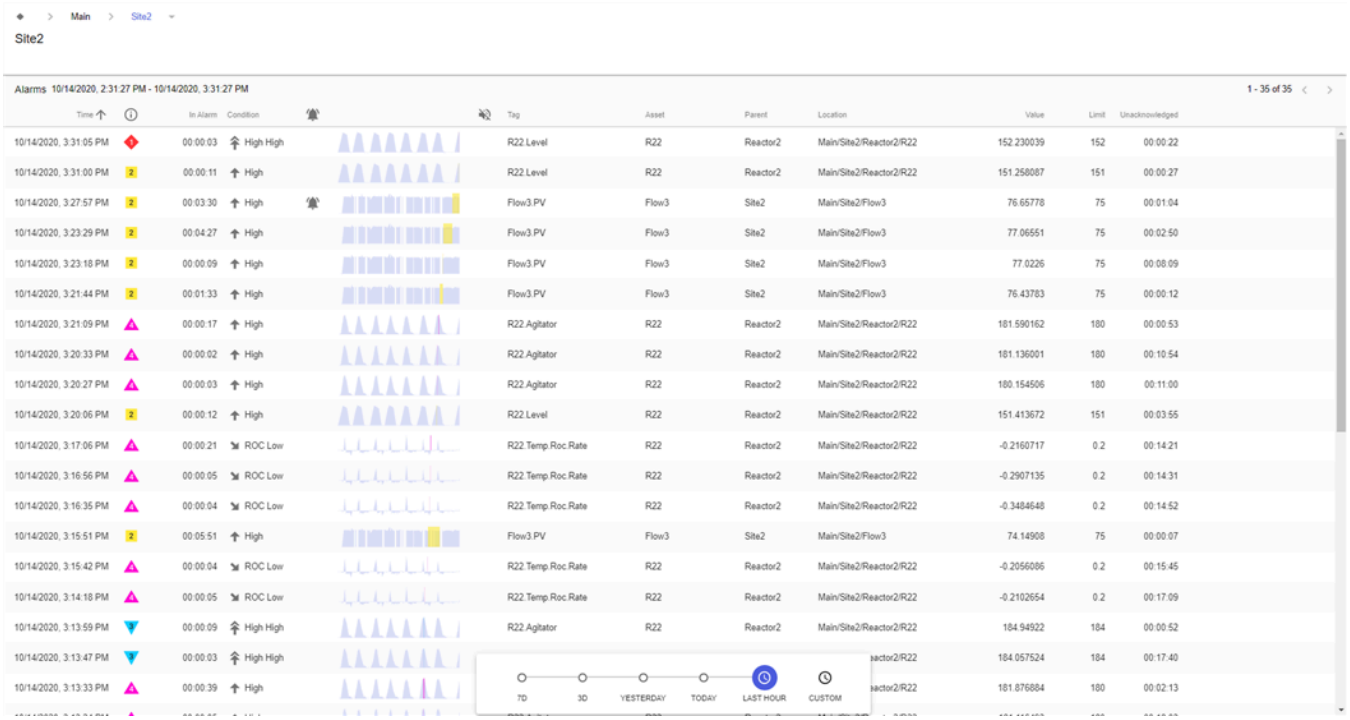
The **Alarms** page allows you to view alarms for multiple asset levels, and filter alarms by a specified range of time. Alarms are configured in Historian.

Accessing the Alarm page

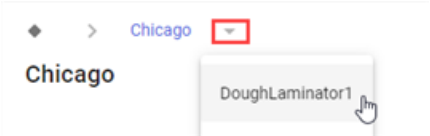
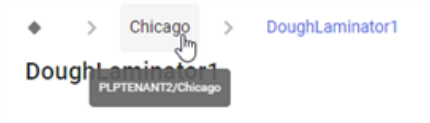
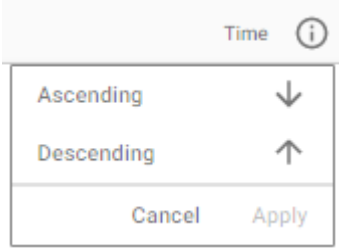
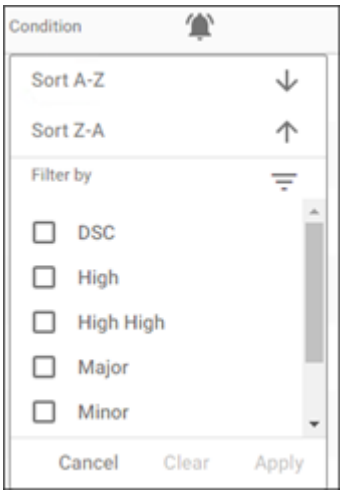
To access the **Alarm** page, select **Browse** from the navigation rail, and then select **Alarms**.

Navigating the Alarm page

The alarm page contains a navigation bar showing the asset hierarchy, and a table which provides details on active alarms.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

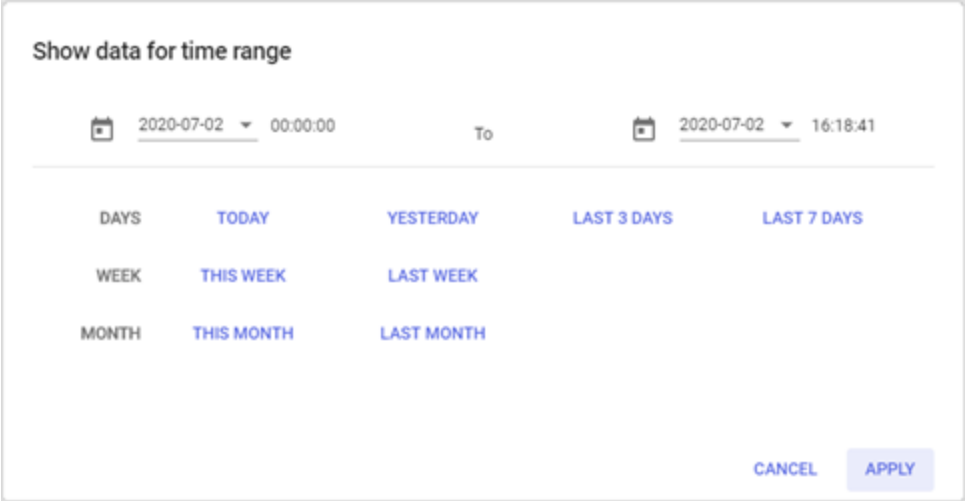
To filter alarms down to a specific child asset, use the down arrow on the navigation bar.	
You can filter through several layers of child assets. To return to a higher level, click the name of the higher level asset.	
You can sort alarms by the Time, Alarm Severity, Condition, Tag, or Asset column. Select the heading of the desired column and select whether to sort in **Ascending** or **Descending** order.	
You can filter alarms by the Alarm Severity, Condition, Tag, and Asset columns. Select the values you want to filter by, then click **Apply**. The filter is applied, and an icon appears next to the column name. Select **Clear** to remove any filters applied to the column. Select **Cancel** to close the dialog without making changes.	

To show alarm data for a specific time range

1. Locate the timeline at the bottom of the **Alarm** page.
2. Select from one of the predefined ranges (last 7 days, last 3 days, yesterday, today, and the last hour), or **Custom** to provide your own range.

   If selecting Custom, the **Show data for time range** window will appear.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Provide a start date & time and an end date & time, or select from one of the predefined options.
4. Click **Apply**.

Use Historian Client Web APIs

See Introduction to Historian Client Web APIs.

Introduction to Historian Client Web APIs

provides several APIs, tools, and other resources to Historian Client Web solution builders. As a solution builder, you can create and customize Historian Client Web solutions for the groups of Historian Client Web users that you work with.

Solution customization

You can use the tag dictionary to define tag extended properties for tags within a data source.

You can also use these solution customization tools and resources:

- Historian Data REST API
  Allows you to upload and retrieve data from your Historian Client Web solution.
- Content Scripting REST API
  Allows you to programmatically define layout and data details to create Historian Client Web content.

Historian Data REST API: Customize data retrieval

With the Historian Data REST API, you can make data retrieval requests.You can submit requests to the Historian Data REST API using a web browser or client-side applications such as:

- Microsoft Excel (2013, 2016, or Office 365)
- Business Intelligence (BI) systems, such as Tableau and Microsoft Power BI

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Supported OData features

OData is an industry standard for querying and updating data from a variety of sources. The implementation of OData for the Historian Data REST API includes support for:

- JSON and atom formats.
- OData version 4 with version 2 of the Historian Data REST API.
- OData version 3 with version 1 of the Historian Data REST API (not recommended).
- Pagination. That is, if your request returns more than 5000 results, they will be returned in pages of up to 5000 records. Each page will include a link to retrieve the next page of records.
- A subset of the OData system query options.

For more information, see OData.org JSON Verbose Format specification.

Recommendation: For best results, when you want to view the data returned by the Historian Data REST API, use the JSONView extension for the Chrome browser.

Supported versions

AVEVA Historian Client Web supports versions 1 and 2 of the Historian Data REST API.

Version 2

This is the current and recommended version of the Historian Data REST API.

Version 2 of this REST APIs based on version 1 and includes further enhancements. Version 2 includes these differences from version 1:

- The `datetimeoffset` parameter is not supported as part of the `DateTime` syntax.
- While Raw and ProcessValue entities use `DateTime`, summary entities use `StartDateTime` and `EndDateTime`.
- For most version 2 queries, single quotes are not used for `DateTime`. For example:

  ```
  http://localhost:32569/Historian/v2/ProcessValues
  ?$filter=DateTime+gt+2017-07-13T00:00:00
  ```

  However, when querying events, single quotes are required for `DateTime`. For example:

  ```
  http://localhost:32569/Historian/v2/Events
  ?$filter=EventTime+gt+'2017-07-13T00:00:00'
  ```

  By contrast, version 1 queries do use single quotes for `DateTime`. For example:

  ```
  http://localhost:32569/Historian/v1/ProcessValues
  ?$filter=DateTime+gt+datetimeoffset'2017-07-13T00:00:00'
  ```
- TagProperty and Events entity types are now open type. That is, dynamic properties can be added to the response at runtime. This can be verified using `$metadata` endpoint URL:

  ```
  http://localhost:32569/Historian/v2/$metadata
  ```

  where there will be additional attribute, `OpenType="true"`, under `<EntityType>` section.
- The Tags resource returns all the properties (fixed and extended) for a tag. A tag's extended properties will be added to the response only if they exist. (The extended property name will not be listed for tags that do

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

not have a given extended property.)

- The combined Summary resource is not supported in version 2. Use the individual AnalogSummary and StateSummary resources instead to retrieve the summary of a tag.
- Version 2 adds support for the OData `contains` function for applicable resources and properties.

Version 1

This is the original version of Historian Data REST API based on the OData v4 specification.

- Version 1 uses the `DateTime` format used in this example:

```
http://localhost:32569/Historian/v1
/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'
+and+StartDateTime+ge+datetimeoffset'2016-05-14T00:00:00.000-07:00'
+and+EndDateTime+le+datetimeoffset'2016-05-16T00:00:00.000-07:00'
```

Data Retrieval

The Historian Data REST API allows you to retrieve data from your solution. Data retrieval with the Historian Data REST API is available through both Historian Client Web and Historian Client Web, so ensure that you refer to the correct syntax for your product from the examples below.

Forming retrieval requests

Use the following syntax for retrieval requests, depending on whether you are using HTTP or HTTPS:

```
http://<servername>:32569/Historian/<api_version>/<resource>?<query>
https://<servername>:32573/Historian/<api_version>/<resource>?<query>
```

Note: 32569 and 32573 are the default ports for HTTP and HTTPS. If your environment is configured to use different ports for either protocol, replace the ports used in this document with the appropriate ports for your environment.

The URL includes the following elements:

Syntax element	Explanation
<*servername*>:32569 or <*servername*>:32573	The servername and port number used by AVEVA Historian Client Web. If the server is the local computer, this could be "localhost:32569" for HTTP or "localhost:32573" for HTTPS.
Historian/<*api_version*>	The API version. For example, "v2".
<*resource*>	The Historian Data REST API retrieval resource. See the complete list.
?<*query*>	A query using OData parameters and operators

For example, this retrieval request gets analog summary data via an endpoint URL using basic authentication:

```
http://localhost:32569/Historian/v2/AnalogSummary
?$filter=FQN+eq+'Depot.Train09'+and+StartDateTime+ge+2017-06-09T09:00:00-07:00+and
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
+EndDateTime+ge+2017-06-09T10:00:00-07:00&Resolution=3600000
```

Supported OData parameters

These parameters are supported for system query via the Historian Data REST API. They are all case-sensitive.

Parameter	Description
$filter	Specifies an expression or function that must evaluate to true for a record to be returned in the collection. A $filter expression supports references to properties and literals. Literal values include: • Strings enclosed in single quotes • Numbers and Boolean values (true or false) Filtering for process value and summary data is case-sensitive. Filtering for other values, however, is case-insensitive. For example, if you filter property values based on a value of "true," values such as "TRUE," "True," and "true" could be returned. The case returned in the query results reflects the case of the stored value. Note that you can use $filter for FQN, but only for exact matches. Use the TagFilter parameter for partial FQN matches. $filter is typically used to specify a start and end date and time for the queried period. This is required for endpoints that return time-series data, but is recommended for most queries in order to reduce server load and make the number of returned results easier to work with.
$select	Specifies a subset of properties to return.
$skip	Specifies the number of records to skip from the beginning of the result set.
$skiptoken	Used to get the next record set that satisfies the query conditions. This parameter cannot be used in queries, but you will see it upon query execution.
$top	Specifies the maximum number of records to return. This subset is formed by selecting only the first *N* items of the set, where *N* is a positive integer specified by this query parameter.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time-series extension parameters

The Historian Data REST API supports the following custom extension parameters in retrieval queries to better control your search results.

- **RetrievalMode** specifies how the resulting data is calculated for Raw and ProcessValues entities. Valid values are:

• Average	• Cyclic	• Integral	• Minimum
• BestFit	• Delta	• Interpolated	• Slope
• Counter	• Full	• Maximum	

- **Resolution** specifies the granularity of data returned for Raw, ProcessValues, and Summary entities.
- **TagFilter** is used to specify which tags to return data on. TagFilter must be used with $filter, which defines the range of results. TagFilter only works with the ProcessValues call. TagFilter supports the following operators:
  - startswith
  - endswith
  - contains

Supported operators

These logical operators are supported for queries:

Operator	Description
eq	Equal
ne	Not equal
gt	Greater than
ge	Greater than or equal
lt	Less than
le	Less than or equal
and	Logical and
or	Logical or
not	Logical negation

Note: Use "%20" to indicate a space. Use "%27" to indicate a single quote.
If you are using the JSONView viewer in the Chrome browser, you can use a plus sign (+) to indicate a space to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

make the URI string more readable.

If the expression includes multiple values for the criteria, you must specify each criteria separately using the "or" operator. For example:

```
...
((Priority+eq+100+or+Priority+eq+200+or+Priority+eq+500
+or+Priority+eq+700)+and+(filter …))
```

Syntax for combining expressions

- Use **and** to use multiple terms for the same parameter. For example:

  $filter=DateTime+ge+2023-08-31T00:00:00z+**and**+DateTime+le+2023-08-31T12:00:00z
- Use **&** to join separate parameters. For example:

  $filter=DateTime+ge+2023-08-31T00:00:00z**&**TagFilter=startswith(FQN, 'turbine')
- In a query, you can have only a single expression specifying a period of time combined with a single expression filtering specific tags. Combine expressions with the "and" operator. The expression itself can be complex, using any of the supported logical operators. Use parentheses ( ) to create precedence groups within an expression.

Retrieval resources

The Historian Data REST API exposes various resources through an endpoint URL that is specific to your Insight solution.

This API includes the following resources for retrieving data:

Process Data and Events Resources	Tag Property Resources	Resources for Version 1 Only
• ProcessValues • AnalogSummary • StateSummary • Events	• Tags • TagProperties • TagPropertyValues • TagGroups • TagSuggest • TagSearch • TagExtendedProperties	• Summary • Daily • Hourly • Minutely

Note: All property names are case-sensitive. Event storage preserves the case that you provide for any property value. For example, a property value of "TRUE" is different than "True" and "true."

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ProcessValues

ProcessValues

Description	Retrieves a set of process value records (where each record includes value + time + quality, or VTQ) for the specified tags.
URL	`/ProcessValues`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z • **EndDateTime=**[DateTimeOffset] The ending date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z
Optional Parameters	• **OPCQuality=**[Int32] The data quality as reported by the source. • **Value=**[Double] 0 or 1. • **Bounding=**[true] If the query specifies a start point, this parameter shows the first data point that occurs on or before the query start. If the query specifies an end point, this parameter shows the first data point that occurs on or after the query end. The actual date and time of the start/end point are shown, even if it's outside the bounds of the query. • **Text=**[String] For string tags, contain the value. For discrete tags, contains the message associated

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	the value (0 or 1).
	• **TagFilter=**[String]
	Allows you to use a query string using OData filter query notation to filter results by tag attributes. Supported tag attributes include FQN, description.
	TagFilter can include up to 20 AND clauses, or up to 20 OR clauses. You cannot mix AND and OR in the same query.
	Operators should be lowercase, and searches are case-insensitive.
	Valid Format:
	`Historian/v2/ProcessValues?TagFilter=startswith(Source,'MVDS') and TagType eq 'string'`
	Invalid Format:
	`Historian/v2/ProcessValues?TagFilter=startswith(Source,'MVDS') And TagType eq 'string'`
	• **Expression=**[String]
	For analog tags, you can perform UOM conversion on retrieved tag data by including a string expression in the following format:
	`UOM([tag FQN 1],[engineering unit name 1];[tag FQN 2],[engineering unit name 2];...)`
	You can provide one or more pairs of tag names and engineering unit names, and during retrieval the data will be converted to the specified units, if valid. If the unit conversion is not valid, the tag's base unit is used instead.
	Note: This feature requires special licensing. If a license is not available and the "Expression" parameter is used, no data is retrieved and an error message is displayed.

Success Response	`Code: 200` `Content: { fqn: plant12.pump6, datetime: 2016-09-03T18:44:09.352247Z}`

Error Response	`Code: 404 NOT FOUND`
	`Content: { error : "FQN doesn't exist" }`
	or
	`Code: 401 UNAUTHORIZED`
	`Content: { error : "You are unauthorized to make this request." }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 1**
	This query returns process values for a specific tag identified by its fully qualified name (*datasource.tagname*). Using a "$filter" clause, it specifies a tag named tank_level within the Baytown data source. The result is a list of values for the tank_level tag.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'`
	Scenario 2
	This query includes a start date time, end date time, and other query parameters.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'+and+DateTime+ge+2014-07-04T23:57:29Z+and+DateTime+le+2014-07-05T00:02:29Z&RetrievalMode=BestFit&Resolution=6500`

Sample Query	**Scenario 3**
	This query returns the current day of the month, and uses the Expression parameter to retrieve the value in hours.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'SysDateDay'&Expression=UOM([SysDateDay],[Hour])`
	Sample Output
	`{ "odata.metadata": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [ { "FQN": "SysDateDay", "DateTime": "2020-10-29T16:57:29.0128451Z", "OpcQuality": 192, "Value": 696.0, "Text": "696", "Unit": "Hour" } ] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 4**
	This query returns process values for a list of tags with fully qualified names (FQN) beginning with "sys", with values falling within a specific date range. If you don't know the fully qualified name of a specific tag, you can use the TagFilter parameter to get a list of possible matches.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=startswith(FQN,'sys')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [{ "FQN": "SysClassicDataRedirector", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "" }, { "FQN": "SysClientAccessPoint", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "" }, { "FQN": "SysConfigStatus", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units" }] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 5**
	This query returns process values for a list of tags whose FQN ends with "sec", with values falling within a specific date range.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=endswith(FQN,'sec')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [{ "FQN": "SysDataAcqOverallItemsPerSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units/s" }, { "FQN": "SysStatusRxEventsPerSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units/s" }] }`

Sample Query	**Scenario 6**
	This query returns process values for a list of tags whose FQN contains "timesec", with values falling within a specific date range.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=contains(FQN,'timesec')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [ { "FQN": "SysTimeSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "Second"`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`}`
	`]`
	`}`

Sample Query	**Scenario 7**
	This query returns process values for a list of tags whose description contains "timesec", with values falling within a specific date range.
	`https://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=Contains(Description,'timesec')`
	Sample Output
	`{`
	`"@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues",`
	`"value": [`
	`{`
	`"FQN": "SysTimeSec",`
	`"DateTime": "2019-01-15T03:57:29Z",`
	`"OpcQuality": 0,`
	`"Value": null,`
	`"Text": null,`
	`"Unit": "Second"`
	`}`
	`]`
	`}`
	Similar to the previous examples, you can search for tags with descriptions that start with, end with, or contain specific text.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using Startbound and Endbound Retrieval Modes

Using Startbound and Endbound Retrieval Modes

Startbound

The Startbound retrieval mode is used to obtain the last-known value of a given tag before the date and time specified in the request.

The return will include the last updated VTQ record occurring before the EndDateTime. In the following POST body example, an `EndDateTime` of `00:11` is provided with a `Startbound` retrieval mode. The return will be the record at the `00:08` timestamp `["00:08", "10"]` because there was no record at `00:10`.

```
{
   "FQN": ["MyTag1"],
   "RetrievalMode": "Startbound",
   "EndDateTime": "00:11"
}
```

Note the following when using the Startbound retrieval mode:

- If you only specify an FQN and Startbound retrieval mode without EndDateTime, then the time at execution of the request will be used as the EndDateTime.
- If you specify an EndDateTime that is exactly equivalent to the VTQ, that VTQ will be returned. From the above example, if the EndDateTime is set to `00:06`, the timestamp and value returned will be `["00:06", "7"]`.
- If the provided EndDateTime occurs before the first VTQ for the specified tag (i.e., "`00:01`"), an empty array will be returned.
- If specifying Startbound with a GET request, you will not be able to use EndDateTime. Instead, use `DateTime le` (less than or equal to), or `DateTime lt` (less than). This will result in the return of the last-updated value with a timestamp less than the DateTime specified.

Endbound

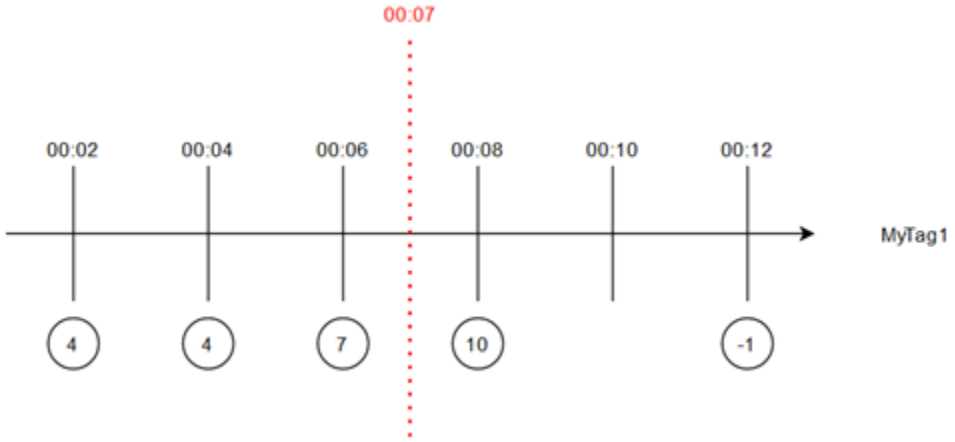
The Endbound retrieval mode is used to obtain the first-known value of a given tag after the date and time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

specified in the request.

The return will include the last updated VTQ record occurring after the StartDateTime. In the following example, a StartDateTime of `00:07` is provided with an EndBound retrieval mode. The return will be the record at the `00:08` timestamp `["00:08", "10"]`.

```
{
   "FQN": ["MyTag1"],
   "RetrievalMode": "Endbound",
   "StartDateTime": "00:07"
}
```



Note the following when using the Endbound retrieval mode:

- If you only specify an FQN and Endbound retrieval mode without StartDateTime, then the time at execution of the request will be used as the StartDateTime. This will result in an empty array.
- If you specify a StartDateTime that is exactly equivalent to the VTQ, that VTQ will be returned. From the above example, if the StartDateTime is set to `00:06`, the timestamp and value returned will be `["00:06", "7"]`.
- If the provided StartDateTime occurs before the first VTQ for the specified tag (i.e., "`00:01`"), the first value (i.e., "4") will be returned.
- If specifying Endbound with a GET request, you will not be able to use EndDateTime. Instead, use `DateTime ge` (greater than or equal to), or `DateTime gt` (greater than). This will result in the return of the last-updated value with a timestamp greater than the DateTime specified.

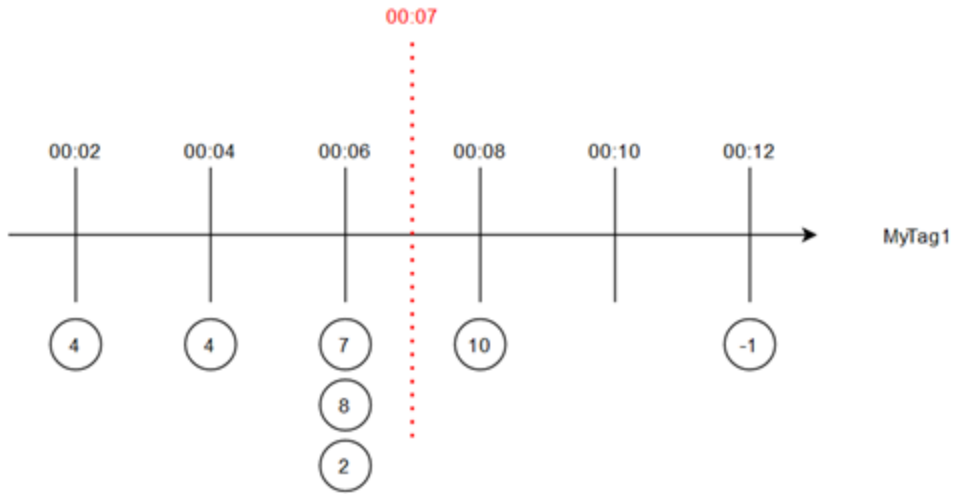
ProcessValues retrieval for versioned data sets

In specific use cases, Historian may store multiple records for a given tag recorded with the same timestamp. When this occurs, at least one record is stored as a "versioned-original."

In specific situations, Historian can sometimes record multiple records labeled as "versioned-original" with the same timestamp. In the example below, values of `7`, `8`, and `2` are recorded at `00:06`.

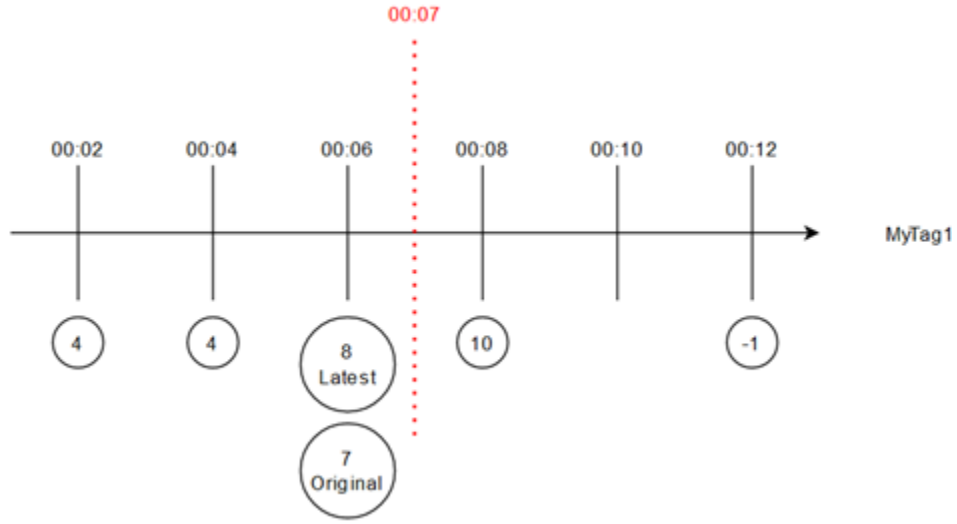
In this scenario, any of those values may be returned with a Startbound or Endbound request to ProcessValues.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In the event that two records exist at the same timestamp, where one is "versioned-original" and the other "versioned-latest," Historian Client Web will return the value associated with "versioned-latest" by default. From the example POST body below, the returned value would be 8.

```
{
   "FQN": ["MyTag1"],
   "RetrievalMode": "Startbound",
   "EndDateTime": "00:07",
   "Version": "Latest"
}
```



In order to toggle between the two records, you must specify `"Version": "Latest"` or `"Version": "Original"` in your query.

Filtering returns by quality

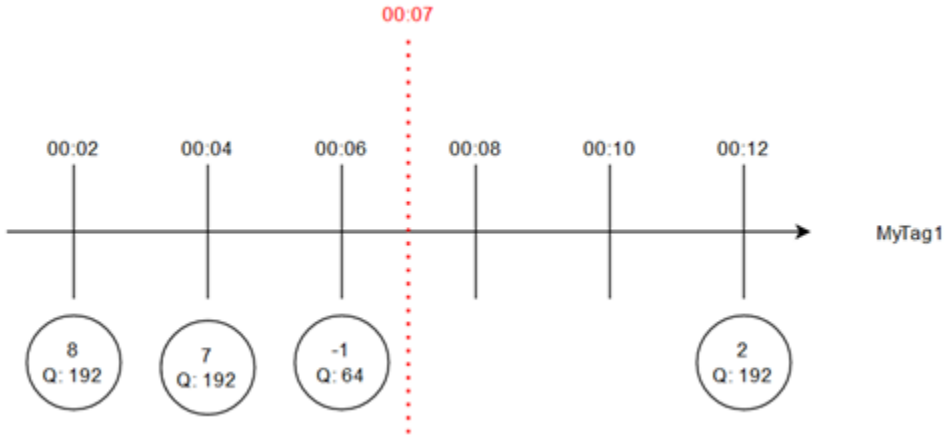
Records can be filtered for OPC quality so that "bad" records are not returned. To filter for quality, append your query with “`QualityRule”: “Optimistic”`, as shown below:

```
{
   "FQN": ["MyTag1"],
   "RetrievalMode": "Startbound",
```

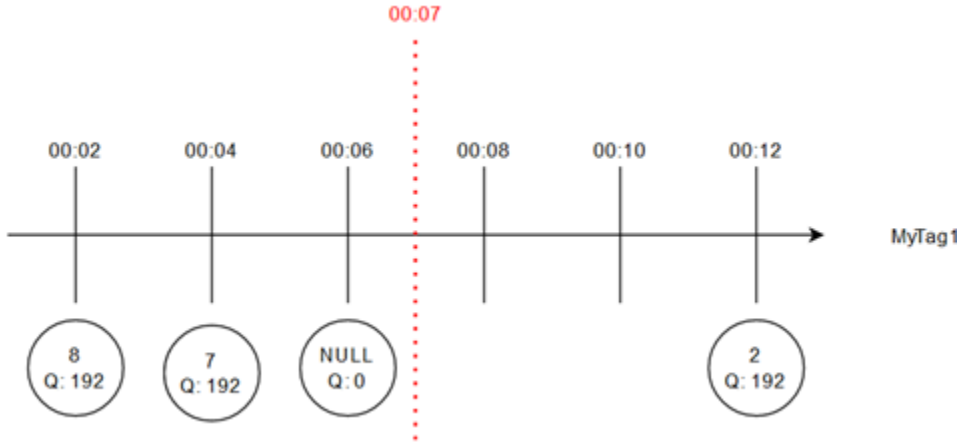
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
    "EndDateTime": "00:07",
    "QualityRule": "Optimistic"
}
```

“`QualityRule”: “Optimistic”` configures the query to return only records with an OPC Quality of “uncertain” (quality 64) or “good” (quality 192). In the example below, including “`QualityRule”: “Optimistic”` means that `-1` at `00:06` is returned.



In this second example below, the record at `00:06` with an OPC quality of `0` is not returned. Instead, the value of `7` at `00:04` is returned.



Returning multiple records with MaxRowCount

Returning multiple records with MaxRowCount

A ProcessValues request with Startbound or Endbound retrieval modes can return multiple records by including the parameter `MaxRowCount`,

The following GET request specifies a `MaxRowCount` of 3 and the `Startbound` retrieval mode:

```
https://online.wonderware.com/apis/Historian/v2/ProcessValues?$filter=FQN eq
'MountainWaterTreatment.CartridgeFilterOutlet.TemperatureC' and DateTime le
2021-09-03T18:44:09.352247Z&RetrievalMode=Startbound&MaxRowCount=3
```

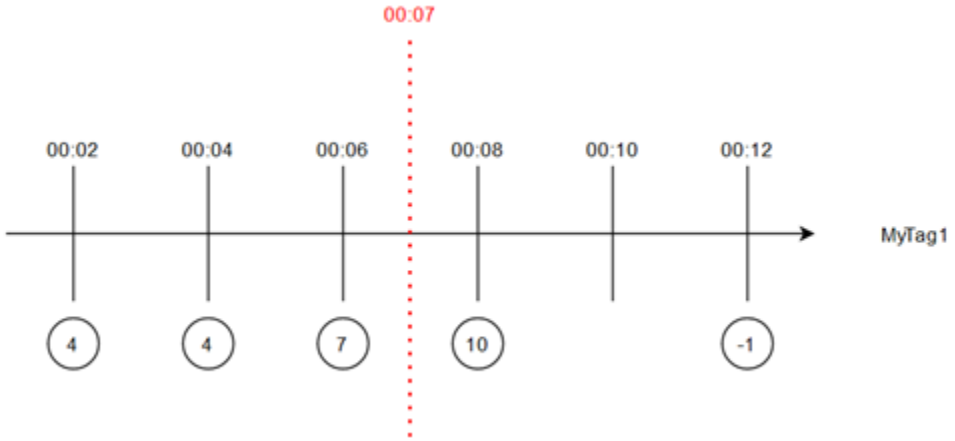
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

As shown in the example below, the above query will return the records for `["00:08", "10"]`, `["00:06", "7"]` and `["00:04", "4"]`.

The following GET request specifies a `MaxRowCount` of 2 and the `Endbound` retrieval mode:

```
https://online.wonderware.com/apis/Historian/v2/ProcessValues?$filter=FQN eq
'MountainWaterTreatment.CartridgeFilterOutlet.TemperatureC' and DateTime ge
2021-09-03T18:44:09.352247Z&RetrievalMode=Endbound&MaxRowCount=2
```

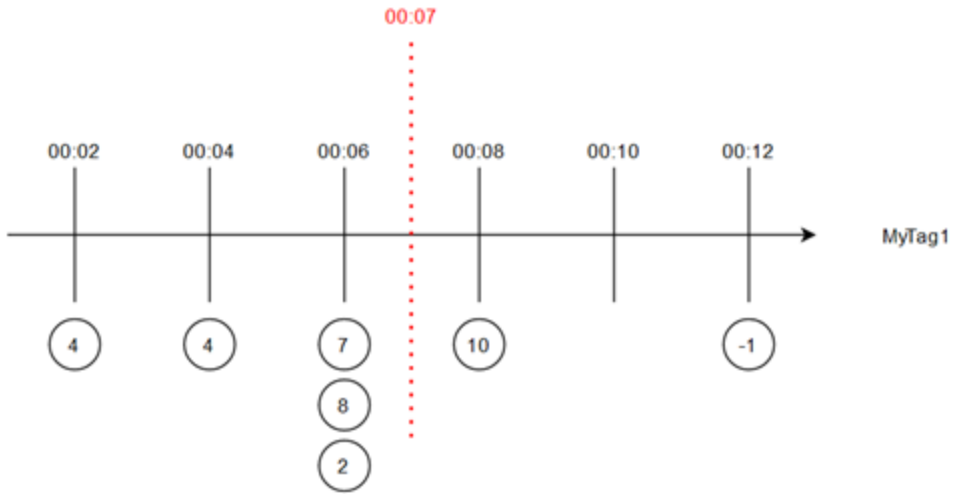
As shown in the example below, the above query will return the records for `["00:08", "10"]` and `["00:12", "-1"]`.



Notes on using MaxRowCount with Startbound and Endbound

- In scenarios involving versioned-original records, requests using MaxRowCount will include versioned-original records up to the amount of records specified with the parameter. For example:

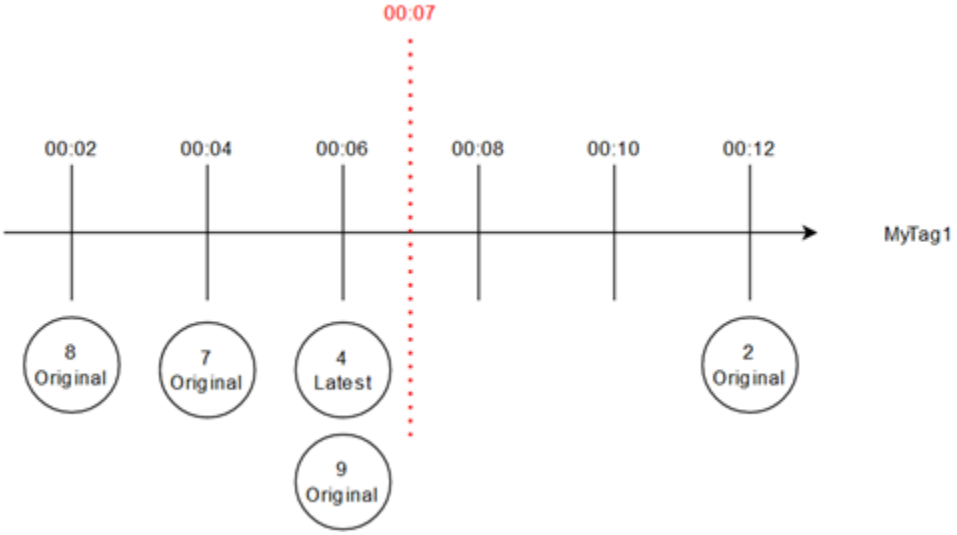
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In the above data set, specifying a MaxRowCount of 3 and a EndDateTime of 00:07 in a ProcessValues call using Startbound retrieval will return the values recorded at 00:06: 7, 8, and 2.

If MaxRowCount is set to 2, any two of those three values will be returned.

- In scenarios involving versioned-original and versioned-latest records, requests using MaxRowCount will prioritize versioned-latest records by default. In the following example, using Startbound with a MaxRowCount of 3, the values of 4, 7, and 8 will be returned:



The query can be adjusted to specify that only original records be returned. From the following example query, the values returned will be 9, 7, and 8.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AnalogSummary

AnalogSummary

Description	Retrieves analog statistics for the specified tags.
URL	`/AnalogSummary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z • **EndDateTime=**[DateTimeOffset] The ending date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z
Optional Parameters	• **RetrievalMode=**[string] Possible values are: Cyclic, Full. Default is Cyclic. • **Resolution=**[Int] In milliseconds. Any positive integer. • **SliceBy=**[Int|Discrete|String] Performs dynamic resolution/cycle computation by tags. Returns one Analog Summary value per tag per dynamic cycle with start and end date time. SliceBy can support up to 10 tags. For more information on using SliceBy, see Using SliceBy. • **SliceByValue=**[string] Specifies the filter criterion to get the summary values for SlicedBy, based on that filter value. • **OPCQuality=**[Int32]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Defines the OPC quality for the data. Normal OPC quality retrieval logic is applied if all the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned.
	• **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **First=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the first point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **FirstDateTime=**[DateTimeOffset] Time stamp associated with first value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle. • **Last=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the last point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **LastDateTime=**[DateTimeOffset] Time stamp associated with last value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle. • **Minimum=**[Double]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the minimum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **MinDateTime=**[DateTimeOffset] Time stamp associated with Min value. NULL if Min is NULL. • **Maximum=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the maximum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **MaxDateTime=**[DateTimeOffset] Time stamp associated with Max value. NULL if Max is NULL.
	• **Average=**[Double] Time weighted average value of retrieval cycle. This is calculated by using the individual summary averages. The calculation is "Sum(average * delta t) / Total time of average in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Average query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) • **StdDev=**[Double] Time weighted standard deviation value of the retrieval cycle. The value is calculated using time weighted sums (Integrals) and time weighted sums of squares (IntegralOfSquares) values, prorated for any partially contained storage cycles. For analog tags, similar StdDev values are produced for each cycle.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Integral=**[Double] Area under value curve of retrieval cycle. The calculation is "Sum(value * delta t) / Total time of integral in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Integral query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) For analog tags, similar Integral values are produced for each cycle. • **Count=**[Double] Number of values in a particular cycle. • **Expression=**[String] For analog tags, you can perform UOM conversion on retrieved tag data by including a string expression in the following format: `UOM([tag FQN 1],[engineering unit name 1];[tag FQN 2],[engineering unit name 2];...)` You can provide one or more pairs of tag names and engineering unit names, and during retrieval the data will be converted to the specified units, if valid. If the unit conversion is not valid, the tag's original units are used instead. **Note:** This feature requires special licensing. If a license is not available, the data is returned in the original units without conversion.
Sample Query	**Scenario 1** This query produces a list of tags that end with "level". In this case, the user doesn't know the fully qualified name (FQN) of a specific tag and wants a short list of possible matches. `https://online.wonderware.[DOMAINSUFFIX]/apis/historian/v2/AnalogSummary?TagFilter=endswith(FQN,'level').` See more TagFilter examples. **Scenario 2** This query produces a list of values with analog summary for the tank_level tag. Notice that this example uses a fully qualified name ("Baytown.tank_level"), which is a combination of a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	data source name ("Baytown") and a tagname ("tank_level"). `https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'` Because this query specifies no start or end time and no resolution, these defaults are used for the returned results: • EndTime defaults to DateTime.UtcNow • StartTime defaults to one hour before EndTime • Resolution defaults toTimespan • Count defaults to1 (number of returned rows)

	Scenario 3
	This query produces a list of tag values with analog summary data. The "$filter" clause narrows the results further by specifying these parameters:
	• Fully qualified name of the tag (FQN+eq+'Baytown.tank_level')
	• Start and end times
	• Result set returned at 1 hour intervals (Resolution=3600000). There are 3.6 million milliseconds in an hour.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'+and+StartDateTime+ge+2017-05-14T00:00:00.000Z+and+EndDateTime+le+2017-05-16T00:00:00.000Z&Resolution=3600000`
	Scenario 4
	This query specifies only StartTime and Resolution (600000 ms, or 10 minutes), but no EndTime.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level' and+StartDateTime+ge+2017-06-29T00:00:00Z&Resolution=600000`
	In this case, Insight assumes:
	• EndTime defaults to DateTime.UtcNow
	The number of rows returned (Count) depends on the StartTime and EndTime.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Scenario 5 This query specifies only EndTime, but no StartTime or Resolution. `https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'and+EndDateTime+le+2017-06-29T00:00:00Z'` In this case, Insight assumes: • StartTime defaults to one hour before EndTime • Resolution defaults to Timespan • Count defaults to 1 (number of returned rows)
	Scenario 6 This query specifies uses SliceBy. `https://online.wonderware.com/apis/Historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.R21.Level'+and+StartDateTime+ge+2019-03-05T00:00:00.000Z+and+EndDateTime+le+2019-03-05T12:00:00.000Z&SliceBy=Baytown.R21.Batch` In this case, Insight calculates an analog summary for the Baytown..R21.Level tag and batches the results per value for the Baytown.R21.Batch tag.

Using SliceBy

Using SliceBy

You can use a SliceBy parameter with an Analog Summary query to retrieve average values by batch within a specific time range.

For example, this is a query for an analog summary of the SysTimeMin tag, sliced by the SysTimeHour tag for a certain time range:

```
{
   "FQN":["SysTimeMin"],
   "StartDateTime":"2018-02-28T18:30:33.966Z",
   "EndDateTime":"2018-02-28T21:00:00Z",
   "SliceBy":["SysTimeHour"]
}
```

If the first or last batch is only partially complete, Historian will provide results as follows:

- If the first batch is only a partial set, the average is calculated for the amount of time represented by that batch. For example, suppose you have a tag that last changed at 7:50 a.m. and the query includes a start time of 8:00 a.m. The average includes values beginning at 7:50 a.m.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To include a partial first batch, use “>=” with your StartDateTime parameter. For example:

```
"StartDateTime">="2018-02-28T08:00:00.000Z"
```

To exclude a partial first batch, use “>” with StartDateTime. For example:

```
"StartDateTime">"2018-02-28T08:00:00.000Z"
```

- If the last batch is only a partial set, the wwResolution for the last batch will be NULL. For example, if the SliceBy query is for 8a.m. to noon, but the last batch is still in progress, the result will include “wwResolution=NULL”.

Batches can be organized using GroupBy. For example, you could use GroupBy to organize data by tag state.

EventSummary

EventSummary

Description	Retrieves summaries for events from the Historian Client Web solution.
URL	`/EventSummary`
Method	`GET`

Optional Parameters	• **GroupBy=**[String]
	The type of Historian Client Web data summarized. Accepted values include:
	  • TagName
	  • Area
	  • HierarchicalArea
	  • Object
	  • Alarm
	• **Value=**[String]
	The data to be summarized. For example, if using GroupBy to retrieve TagName, the value is the FQN of the tag.
	• **Count=**[Int32]
	The number of alarms generated based on the value, and grouped by group.
	• **TotalMs=**[Int64]
	Total alarm time of the value.
	• **TotalUnAckMs=**[Int64]
	Total unacknowledged alarm time of the value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 1**
	This query returns a summary of the data generated in the last hour.
	`https://online.wonderware.[DOMAINSUFFIX]/apis/historian/v2/EventSummary`
	Scenario 2
	This query returns a summary of the data generated in the last hour, filtered for records where GroupBy is "Area".
	`https://online.wonderware.[DOMAINSUFFIX]/apis/historian/v2/EventSummary?$filter=GroupBy eq 'Area'`
	Scenario 3
	This query returns the summary of the last day and specifies only one tag, `Reactor_001.ReactLevel`.
	`https://online.wonderware.[DOMAINSUFFIX]/apis/historian/v2/EventSummary?$filter=GroupBy eq 'Alarm'&EventFilter=EventTime ge 2018-04-16T01:30:56.136Z and EventTime le 2018-04-17T01:35:56.136Z and source_processvariable eq 'Reactor_001.ReactLevel'`

StateSummary

StateSummary

Description	Retrieves state summary values for the specified tags.
URL	`/StateSummary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **RetrievalMode=**[string] Possible values are: Cyclic, Full. Default is Cyclic. • **Resolution=**[Int] In milliseconds. Any positive integer. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **Text=**[string] Non-numeric state. • **Average=**[Double] Average time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. • **AverageContained=**[Double] Average time in this state among all occurrences of this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Minimum=**[Double] Minimum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	cycle if possible. • **MinimumContained=**[Double] The minimum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Maximum=**[Double] Maximum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **MaximumContained=**[Double] The maximum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Total=**[Double] Total time in this state during this retrieval cycle, including state occurrences that fall only partially within the period. • **TotalContained=**[Double] Total time in this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Percent=**[Double] Percent of the time during this retrieval cycle that the tag was in this state, including state occurrences that fall only partially within the period. • **PercentContained=**[Double] The precentage of the entire retrieval cycle time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	that the tag was in this state, excluding state occurrences that fall only partially within the period. This is a ratio between StateTimeTotalContained and StateTimeTotal expressed as a percentage in the range 0 to 100. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Count=**[Int64] The number of times the state occurred within the retrieval cycle, including states that only partially occur in the cycle. • **CountContained=**[Int64] The number of times the state occurred fully contained within the retrieval cycle. States that only partially occur in the cycle are not counted. • **LastDateTime=**[DateTimeOffset] The time stamp for the last received value.

Sample Query	**Scenario 1**
	This query specifies no start or end time and no resolution.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03'`
	In this case, these defaults are used for the returned results:
	• EndTime defaults to DateTime.UtcNow
	• StartTime defaults to one hour before EndTime
	• Resolution defaults toTimespan
	• Count defaults to1 (number of returned rows)
	Scenario 2
	This query specifies only StartTime and Resolution (600000ms, or 10 minutes), but no EndTime.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03' and StartDateTime ge 2017-06-29T00:00:00Z&Resolution=600000`
	In this case, Insight assumes:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• EndTime defaults to DateTime.UtcNow The number of rows returned (Count) depends on the StartTime and EndTime. **Scenario 3** This query specifies only EndTime, but no StartTime or Resolution. `http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03' and EndDateTime le 2017-06-29T00:00:00Z` In this case, Insight assumes: • StartTime defaults to one hour before EndTime • Resolution defaults toTimespan • Count defaults to1 (number of returned rows)

Events

Events

Description	Retrieves information about events and alarms. **Note:** All property names are case-sensitive. Event storage preserves the case that you provide for any property value. For example, a property value of "TRUE" is different than "True" and "true."
URL	`/Events`
Method	`GET`
Optional Parameters	• **ID**=[GUID] Globally unique identifier for the event. • **EventTime**=[DateTime] UTC time stamp indicating when the event occurred. • **Type**=[String] Main categorization of the event. Examples of valid values: - Alarm.Acknowledge - Application.Write - Alarm.Clear - User.Write

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	- Alarm.Set - User.Write.Secured - Alarm.Write - User.Write.Verified • **Priority**=[Int32] Value indicating the importance of the event. Values range from 1 to 999, with lower numbers indicating higher importance. • **Namespace** =[String] The namespace for this event tag. • **Severity**=[Int32] Categorization of the urgency of the event: 1 - Critical 2 - Major 3 - Minor 4 - Informational • **EventTimeUTCOffsetMins**=[Int32] For local time, the offset in minutes from UTC time. • **ReceivedTime**=[DateTime] UTC time stamp indicating when the event was received by the Historian server. • **IsAlarm**=[Bool] "true" or "false" indicating whether this event is an alarm. • **Comment**=[String] A comment providing more information about the event. **Note:** If an alarm comment from Application Server contains a backslash (for example, \n), then an extra backslash appears in the event result (for example, \\n) in the browser. • **InTouchType**=[String] InTouch Type value. Examples include: - ALM - RTN - ACK - SYS • **ValueString**=[String] The current value string. • **PreviousValueString**=[String] The previous value string.
Optional Alarm Parameters	• **Alarm_ID**=[String] ID of the original Alarm event. For "Alarm.Set"

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	events, this will be the same as the event ID. • **Alarm_Class**=[String] InTouch alarm classification. Examples include "DSC" (discrete), "VALUE" (LoLo, Hi, etc.), "DEV" (deviation), and "ROC" (rate of change). • **Alarm_Type**=[String] InTouch alarm type. • **Alarm_InAlarm**=[Boolean] "true" or "false" indicating whether the Alarm is still in the active state. • **Alarm_Acknowledged**=[Boolean] "true" or "false" indicating whether the user has acknowledged this alarm. • **Alarm_Condition**=[String] The condition being alarmed. Examples include "Limit.Hi", "ROC.Lo", and "System" among others. • **Alarm_ValueString**=[String] Value logged for the variable related to the event. • **Alarm_LimitString**=[String] Limit being alarmed. • **Alarm_UnAckDuration**=[Int32] The duration, in milliseconds, for which the alarm went un-acknowledged. • **Alarm_Duration**=[Int32] The duration, in milliseconds, for which the alarm was active. • **Alarm_IsSilenced**=[Boolean] "true" or "false" indicating whether the alarm was silenced. • **Alarm_IsShelved**=[Boolean] "true" or "false" indicating whether the alarm was shelved. • **Alarm_ShelveStartTimeUTC**=[DateTime] Scheduled start of the shelve time if the alarm has been shelved. • **Alarm_ShelveEndTimeUTC**=[DateTime] Scheduled end of the shelve time if the alarm has been shelved. • **Alarm_ShelveReason**=[String] The reason the alarm was shelved. • **Alarm_ShelveUserLogin**=[String] The login ID for the person who shelved the alarm.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Alarm_ShelveNode**=[String] The node of the shelved alarm.
Optional Provider Parameters	• **Provider_NodeName**=[String] Name of the node that generated the event. • **Provider_System**=[String] Software system that generated the event. Examples include "Application Server", "InTouch", and "InBatch". • **Provider_ApplicationName**=[String] Application that generated the event. For Application Server, this will be the galaxy name. For InTouch, it will be the InTouch application name. • **Provider_SystemVersion**=[String] Software version (for example, 4566.1210.5811.1) for the component identified by Provider_ApplicationName. • **Provider_InstanceNames**=[String] Provider-specific string that uniquely identifies the instance an application on a given node.
Optional Source Parameters	• **Source_Name**=[String] The name of the data source for this event tag. • **Source_ProcessVariable**=[String] Process variable to which the event is related. • **Source_ProcessVariable_Units**=[String] Engineering units used for the process variable. • **Source_ConditionVariable**=[String] Condition variable related to the event. For example, if "TIC101" has a field attribute "PV" and this is a "Hi" alarm, this value will be "TIC101.PV.Hi". • **Source_Object**=[String] Non-hierarchical name for the object to which the event is related, for example, "TIC101". • **Source_HierarchicalObject**=[String] Hierarchical name for the source object. For example, "Reactor_001.TIC". • **Source_Area**=[String] Non-hierarchical Area name. For example, "Bottling_Zone".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Source_HierarchicalArea**=[String] Hierarchical Area name. For example, "Plant_001.Building_002.Mixing". • **Source_Engine**=[String] Non-hierarchical engine name. For example, "AppEngine_001". • **Source_Platform**=[String] Non-hierarchical platform name. For example "WinPlatform_001".
Optional User Parameters	• **User_Account**=[String] This is the login name for the operator for the given application. • **User_Name**=[String] Name of the user. • **User_NodeName**=[String] Computer name from which a user action was executed. • **User_Email**=[String] The user's email address. • **User_Phone**=[String] The user's phone number. • **User_InstanceName**=[String] An instance name for the user. • **User_Agent**=[String] Application name that the user was running when the event was generated.
Optional Verifier Parameters	• **Verifier_Account**=[String] This is the login name for theverifier. • **Verifier_Name**=[String] Name of the verifier. • **Verifier_NodeName**=[String] Computer name from which a verifier action was executed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Queries	**Sample 1**
	This query lists all events that occur from 2017-07-13 onward.
	`https://localhost:32573/Historian/v2/Events ?$filter=EventTime+gt+'2017-07-13T00:00:00'`
	Sample 2
	This query lists all alarm clearing events.
	`http://localhost:32569/Historian/v2/Events ?$filter=type+eq+'Alarm.Clear'`
	Sample 3
	This query lists events from 2017-05-14 to 2017-05-15 at 9 am. It skips the first 1000 events, then lists the top 2000 events after that.
	`http://localhost:32569/Historian/v2/ Events/?$filter=EventTime gt 2017-05-14T09:00:00.000Z and EventTime lt 2017-05-15T09:00:00.000Z&$skip1000&$top=2000`

Sample Output	{ @odata.context: "http://localhost:32569/ Historian/v2/Events /$metadata#Events", value: [{ id: "ee9b042c-181e-4872-b4e8-cb9575c1f4f9", eventtime: "2017-07-13T00:00:01.047Z", type: "Alarm.Clear", receivedtime: "2017-07-13T00:00:01.2553666Z", source_name: "AaEvents", namespace: "AaEvents", alarm_acknowledged: false, alarm_class: "VALUE", alarm_condition: "Limit.LoLo", alarm_durationms: 5006, alarm_id: "f1e21e43-352e-d3db-7552-6e77f193d02d", alarm_inalarm: false, alarm_isshelved: false, alarm_issilenced: false, alarm_limitstring: "10.0", alarm_originationtime: "2017-07-12T23:59:56.041Z", alarm_state: "UNACK_RTN", alarm_tagtype: "S", alarm_type: "LoLo", comment: "Severity 1", eventtimeutcoffsetmins: -420, intouchtype: "LoLo", isalarm: true, priority: 1, provider_applicationname: "AlarmsandEvents", provider_nodename: "INSIGHTCHARTAPP",

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`provider_system: "Application Server",`
	`provider_systemversion: "4966.1210.14578.2",`
	`severity: 1,`
	`source_area: "Plant_Area",`
	`source_conditionvariable: "Drum_Conveyor.HorizontalMovement.LoLo",`
	`source_hierarchicalarea: "Enterprise.Site.Plant.Plant_Area",`
	`source_hierarchicalobject: "Drum_Conveyor",`
	`source_object: "Drum_Conveyor",`
	`source_processvariable: "Drum_Conveyor.HorizontalMovement",`
	`valuestring: "30.0"`
	`},`

Tags

Tags

Description	Using the GET verb, retrieves tag metadata. For version 2, this also includes tag extended properties. Using the POSTverb, manages multiple tags at once. Using the DELETE verb, deletes a specified tag.
URL	`/Tags`

Methods	`GET`
	`POST`
	`DELETE`

Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **Source=**[string] The data source. • **Description=**[string] The description of the tag. • **EngUnit=**[string] The engineering units used for the tag's recorded values. **EngUnitMax=**[Double] The maximum value of the tag, measured in engineering units.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **EngUnitMin=**[Double] The minimum value of the tag, measured in engineering units. The minimum value of the tag, measured in engineering units. • **InterpolationType=**[string] The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. • **IntegralDivisor=**[Double] The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only. • **RolloverValue=**[Double] The rollover value for the tag. • **MessageOff=**[string] The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. • **MessageOn=**[string] The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	TRUE state. • **TagName=**[string] The unique name of the tag within the AVEVA Historian system. • **TagType=**[string] The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the AVEVA Historian TagRef table. • **Raw** Identifies related the raw acquired value. • **Minutely** Identifies related minutely summary. • **Hourly** Identifies related hourly summary. • **Daily** Identifies related daily summary.

Get tags

Sample Query	**Scenario 1**
	This query lists tag metadata for all tags in all data sources for your Insight solution.
	`http://localhost:32569/Historian/v2/Tags`
	Scenario 2
	This query lists metadata for tags in a particular data source. The data source in this example is named Baytown.
	`http://localhost:32569/Historian/v2/Tags ?$filter=Source+eq+'Baytown'`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/ Historian/v2/$metadata#Tags",`
	`value: [`
	`{`
	`FQN: "14th aug app.Auto",`
	`Source: "14th aug app",`
	`Description: "Automatic mode",`
	`EngUnit: "",`
	`EngUnitMax: 0,`
	`EngUnitMin: 0,`
	`InterpolationType: "None",`
	`MessageOff: "Manu",`
	`MessageOn: "Auto",`
	`TagName: "Auto",`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`TagType: "Discrete",`
	`Alias: "nareshtag-discrete",`
	`Location: "/Application Tag 2",`
	`Raw@odata.navigationLink:`
	`"http://localhost:32569/Historian/v2/Tags`
	`('14th%252baug%252bapp.Auto')/Raw",`
	`Minutely@odata.navigationLink:`
	`"http://localhost:32569/Historian/v2/Tags`
	`('14th%252baug%252bapp.Auto')/Minutely",`
	`Hourly@odata.navigationLink:`
	`"http://localhost:32569/Historian/v2`
	`/Tags('14th%252baug%252bapp.Auto')/Hourly",`
	`Daily@odata.navigationLink:`
	`"http://localhost:32569/Historian/v2/Tags`
	`('14th%252baug%252bapp.Auto')/Daily"`
	`}`
	`],`
	`@odata.nextLink: "http://localhost:32569/`
	`Historian/v2/Tags`
	`?$skiptoken=koen66nr4nGCReC02FqdxO4Y62C7zBVs`
	`v3uf7mO8XMQqxZveqw8`
	`+v2/bVCsFdevT"`

Delete a tag

Sample Query	This query deletes a tag named Weather.Brisbane.Cloudiness.
	`DELETE /Historian/v2/`
	`Tags('Weather.Brisbane.Cloudiness') HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Content-Type: application/json`
	`{`
	`"delete":`
	`{`
	`"FQN":["datasourcename.tagname"]`
	`}`
	`}`

Manage multiple tags

Sample Query	**Scenario 1: Bulk query**
	This query retrieves multiple tags.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchqaruntime.cloudapp.net:8080`
	`Authorization: Basic …….`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"query":`
	`{`
	`"FQN":['Weather.Brisbane.Cloudiness','Weathe`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`r.Auckland.Cloudiness'],`
	`"select":"FQN,TagName"`
	`}`
	`}`
	
	Scenario 2: Bulk edit
	
	This makes updates to multiple tags. You can edit up to 100 tags at a time, with maximum of 50 properties per tag.
	
	Note: Historian Data REST API extended properties map to the AVEVA Historian SDK tag extended properties API.
	
	This makes updates to a tag's standard and extended properties.
	
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Authorization: Basic …`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"properties":`
	`[`
	`{`
	`"FQN": "18Jul_kc.test1",`
	`"PropertyName": "Alias",`
	`"Text": "SysTimeSec",`
	`"Type": "String",`
	`"Searchable":false`
	`}`
	
	Scenario 3: Bulk delete
	
	This deletes multiple tags.
	
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"delete":`
	`{`
	`"FQN":["datasourcename.tagname1",`
	`"datasourcename.tagname2"]`
	`}`
	`}`

Sample Output	The output from a bulk edit includes results per tag. If any of a tag's properties is invalid, the tag fails and results in an error message.
	
	`{`
	`"value":`
	`[`
	`{`
	`"FQN": "System.Tag1",`
	`"$StatusCode": 200`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`},`
	`{`
	`"FQN": "System.Tag2",`
	`"$StatusCode":500,`
	`"$ErrorMessage":"Alias property contains illegal characters"`
	`}`
	`]`
	`}`

TagProperties

TagProperties

Description	Lists all properties used in the system. Tag properties include both base and extended properties of a tag. Examples of TagProperties are DataSourceName and Unit.
URL	`/TagProperties`
Method	`GET`
Required Parameters	• **Name=**[string] The unique name of the tag property.
Optional Parameters	• **Values** Valid values for this property.
Sample Query	This query lists all properties and extended properties for your Insight solution: `http://localhost:32569/Historian/v2/TagProperties`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagProperties",`
	`value: [`
	`{`
	`Name: "DataSourceName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "Namespace",`
	`Type: "String",`
	`ReadOnly: true`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`},`
	`{`
	`Name: "TagName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "FullyQualifiedName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`...`

TagPropertyValues

TagPropertyValues

Description	Retrieves values for one or more tag properties. Tag properties include base and extended properties of Tag. Examples of TagProperties are DataSourceName and Unit.
URL	`/TagpropertyValues`
Method	`GET`
Required Parameters	• **Name=**[string] Unique name of the tag property value. • **Value=**[string]
Optional Parameters	• **Tags** Identifies related Tags.
Sample Query	This query lists the property name ("Source") and value (that is, the data source name) for all data sources: `http://localhost:32569/Historian/v2/TagPropertyValues`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagPropertyValues",`
	`value: [`
	`{`
	`Name: "Source",`
	`Value: "Meter",`
	`Tags@odata.navigationLink:`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	"http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Meter')/Tags"
	},
	{
	Name: "Source",
	Value: "Weather",
	Tags@odata.navigationLink:
	"http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Weather')/Tags"
	}
	]
	}

TagGroups

TagGroups

Description	Retrieves information about data sources. Each data source contains a group of tags.
URL	`/TagGroups`
Method	`GET`
Required Parameters	• **GroupID=**[string] Globally unique identifier for the tag group.
Optional Parameters	• **GroupName=**[string] The name of the data source. • **TypeID=**[string] The type of data source. • **ParentID=**[string] • **Scope=**[string] Used for tag-level security, this defines a location within the data source. • **Tags** Identifies related Tags. • **Groups** Identifies related TagGroups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	This query lists the data sources (labeled "TagGroups" in this API) for your Insight solution identified by solution ID. In this example, the solution ID used is "ik97r5". `http://localhost:32569/Historian/v2/TagGroups`
Sample Output	`{ @odata.context: "http://localhost:32569/Historian/v2/$metadata#TagPropertyValues", value: [{ Name: "Source", Value: "Meter", Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Meter')/Tags" }, { GroupID: "1cd29ff2-f7f3-428b-a593-55c917136a39", GroupName: "Weather", Type: "1000000", ParentID: "0", Scope: "Public", Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagGroups('1cd29ff2-f7f3-428b-a593-55c917136a39')/Tags", Groups@odata.navigationLink: "http://localhost:32569/Historian/v2/TagGroups('1cd29ff2-f7f3-428b-a593-55c917136a39')/Groups" }] }`

TagSuggest

TagSuggest

Description	Retrieves content or tags based on search criteria from Insight. It returns the search results based on tag name, content name, keywords, description, and so on. It also returns the summary of the search result --

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	for example, total matching search results for the given search text. You can use the these query options: • q -- (query) Limits the search to only specified fields instead of searching all fields. • Type -- Limits the search to only a certain type of data (Tag or SavedContent).
URL	`/TagSuggest`
Method	`GET`
Required Parameters	• **Value=**[string] The suggested value.
Optional Parameters	• **FieldName=**[string] The related field name. • **Count=**[Double] The hit count for this query string. • **SearchRanking=**[Double] The search ranking for the tag. • **DisplayText=**[string] The associated display text. • **Search** • **Tags** Identifies related Tags. • **ExpandSuggest**

Sample Query	**Scenario 1**
	This query returns values from a search of all fields (default).
	`http://localhost:32569/Historian/v2/TagSuggest`
	Scenario 2
	This query uses the "q" query option to limit the search to only the "location" field.
	`http://localhost:32569/Historian/v2/TagSuggest ?q=&searchfields=location`
	Scenario 3
	This query uses the "Type" option to limit the search

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	to only a certain type of data (Tag or SavedContent). `http://localhost:32569/Historian/v2/TagSuggest?q=d&Type=Tag`

TagSearch

TagSearch

Description	Provides tagname results based on provided search parameters. The data returned by this entity is dependent on the historian's implementation of search functionality. Some historians will return empty result sets. It finds all the matching results for a given query string. It is basically used to filter the results based on the suggestion results. It searches for the matching record only on the fields provided in the key name as part of the query parameter. The results contain basic tag metadata information; such as FQN, tagname, source, and search ranking. You can use these query options: • q - Specifies the query string (for example, the value typed by the user into the search box). • kn - Specifies the key name (field name) to which the search will be applied. • kv - Specifies the key value which will be used for the search.
URL	`/TagSearch`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **Source=**[string] The data source. • **TagName=**[string]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The name of the tag. • **DisplayText=**[string] The displayed text for the tag. • **SearchRanking=**[Double] The search ranking for this tag. • **Tag** Returns a URL to retrieve a list of Tag entity values for the tags that match.
Sample Query	This example uses the "q", "kn", and "kv" query options. `http://localhost:32569/Historian/v2/TagSearch ?q=r&kn=source&kv=atron`

TagExtendedProperties

TagExtendedProperties

Note: This resource has been deprecated. See sample queries under Tags for current information on retrieving properties associated with a tag.

Description	Retrieves both standard and extended properties for a tag.
URL	`/TagExtendedProperties`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **PropertyName=**[string] The name of the property.
Optional Parameters	• **Value=**[Double] The value of the property. • **Text=**[string] The associated text for the property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Type=**[string] Can be one of these: • [Edm.]String • [Edm.]Int16, Int32, Int64 • [Edm.]Double • [Edm.]DateTimeOffset • [Edm.]Guid • [Edm.]Boolean • [Edm.]Geography types • [Edm.]Geometry types

Comments

Comments

Description	• Retrieve the comment for ProcessValue on a given tag.
URL	`/Comments`
Method	`GET`
Required Parameters	• **Type** The filter option for the comment resource must specify 'ProcessValue'. • **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **DateTime=**[DateTimeOffset] (v2 and later) Specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z. By default, the end time is set to the current time (now) and the start time is set to one hour before.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	This example retrieves comments for a specific time range for a tag: `https://online.wonderware.[DOMAINSUFFIX]/apis/Historian/v2/Comments?$filter= Type eq ‘ProcessValue’ and FQN eq ‘Crane5.TestTag5_1’ and DateTime ge 2014-06-17T09:51:49.7687436Z and DateTime lt 2017-02-18T09:56:49.7687436Z`
Sample Output	`{ "@odata.context": "https://online.wonderware.[DOMAINSUFFIX]/Historian/v2/$metadata#Annotations", "value": [{ "Id": "8a5a98a9-77a5-469d-b223-2a7838c37bb5", "CreatedUserName": "Arun", "SharedMode": "Team", "SharedTime": "2017-07-17T02:40:19.4058692Z", "Location": "\Lake Forest", "DateCreated": "2017-07-17T02:40:19.4058692Z", "Text": "SampleAnnotation 1", "DateModified": "2017-07-17T02:40:19.4058692Z", "Type": "ProcessValue", "FQN": "aaaa", "DateTime": "2017-07-16T19:40:19.403869-07:00" }, { "Id": "aa09e2cc-be07-4180-9591-009987efc53f", "CreatedUserName": "Arun", "SharedMode": "Team", "SharedTime": "2017-07-17T03:40:19.4068693Z", "Location": "\", "DateCreated": "2017-07-17T03:40:19.4068693Z", "Text": "SampleAnnotation 2", "DateModified": "2017-07-17T03:40:19.4068693Z", "Type": "ProcessValue", "FQN": "aaaa", "ProcessDateTime": "2017-07-16T19:40:19.403869-07:00" }] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval examples

You can send retrieval requests to the Historian Data REST API from any modern web browser and from OData client-side tools. Examples of such tools include Microsoft Excel (2013, 2016, or Office 365), and Business Intelligence (BI) systems, such as Tableau and Microsoft Power BI.

Select below to see query examples using specific tools:

- Browser query examples
- Postman query examples
- Excel query examples
- PowerBI query example

Retrieve data via browser query

Retrieve data via browser query

You can simply type an OData HTTP request in the address bar of a modern browser to query Historian Data API.

- **List root entities**

  This example lists the entities you can explore and submit additional queries against.

  ```
  https://<servername>:32569/Historian/v2
  ```

  For more information, see Resources.

- **List data sources (tag groups)**

  This shows the data sources (labeled "TagGroups" in this API) for your Historian Client Web solution. If you have accounts for more than one solution, this shows the data sources for your original Historian Client Web solution.

  ```
  https://<servername>:32569/Historian/v2/TagGroups
  ```

- **List tag metadata for all tags**

  This shows metadata for all tags in all data sources for your original Historian Client Web solution.

  ```
  https://<servername>:32569/Historian/v2/Tags
  ```

- **List tag metadata for a data source**

  This shows metadata for tags in a particular data source. The data source in this example is named Baytown.

  ```
  https://<servername>:32569/Historian/v2/Tags?$filter=Source+eq+'Baytown'
  ```

- **List tag values for a data source**

  This example expands on the last one. It uses a "$filter" clause that specifies a particular tag named tank_level within the Baytown data source. The result is a list of values for the tank_level tag.

  ```
  https://<servername>:32569/Historian/v2/
  ProcessValues?$filter=FQN+eq+'Baytown.tank_level'
  ```

- **List tag values with analog summary data for a data source**

  This example result is a list of values with analog summary for the tank_level tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
https://<servername>:32569/Historian/v2/
AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'
```

Notice that this example uses a fully qualified name ("Baytown.tank_level"), which is a combination of a data source name ("Baytown") and a tagname ("tank_level").

Retrieve data using Postman

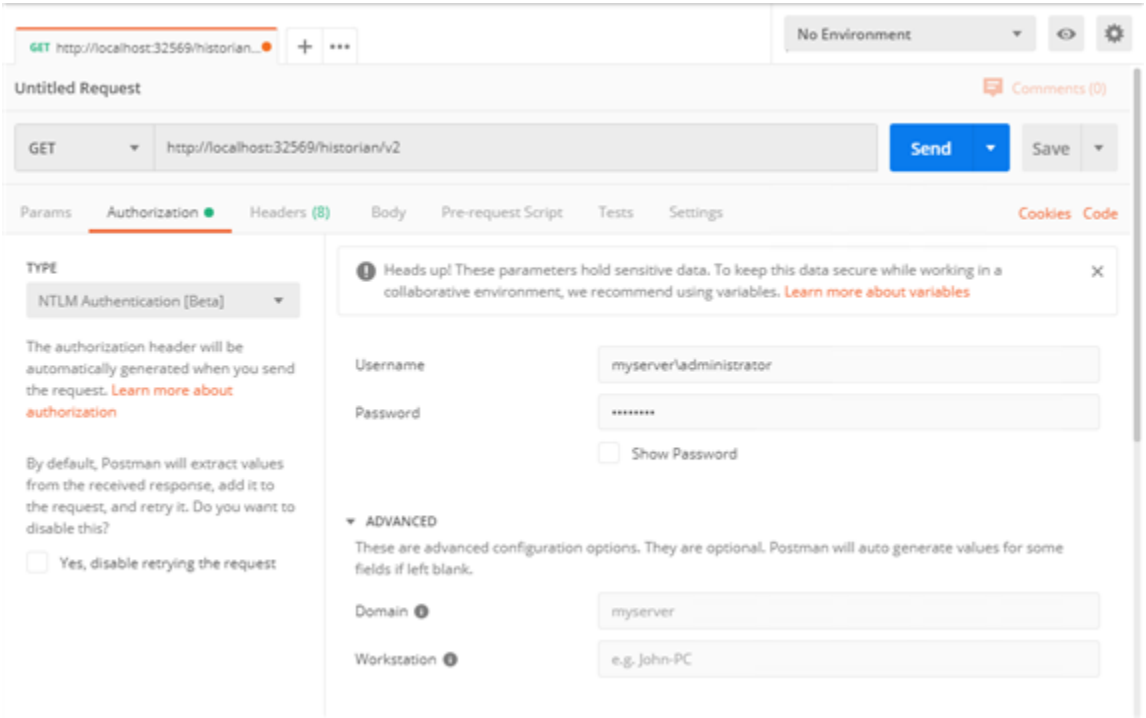
Retrieve data using Postman

A great way to explore and learn about the Historian Client Web APIs is by using a simple client, such as Postman. Postman is a free developer tool allows you to query any API. You can download Postman from https://postman.com/.

To query the Historian Client Web APIs using Postman

1. Next to the GET button, type this address:

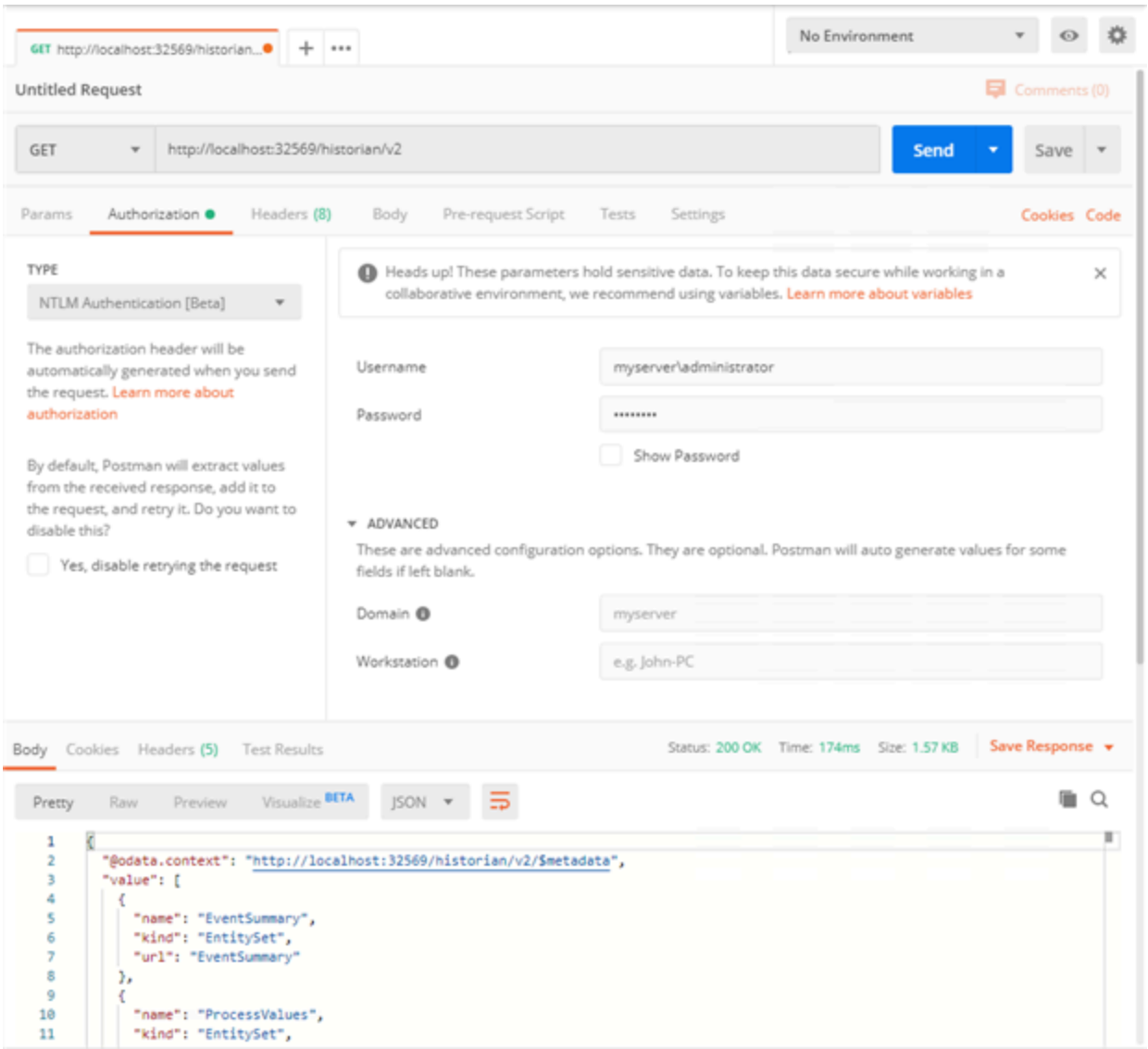
```
http://localhost:32569/historian/v2
```



2. From the **Type** dropdown list, select **NTLM Authentication [Beta]** to specify the authentication to use.
3. Enter the username and password for a user account with sufficient privileges to access the API.
4. In the upper-right of the screen, select **Send**.

   The result is a catalog of data entities from Historian Client Web.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Retrieve data using Excel

Retrieve data using Excel

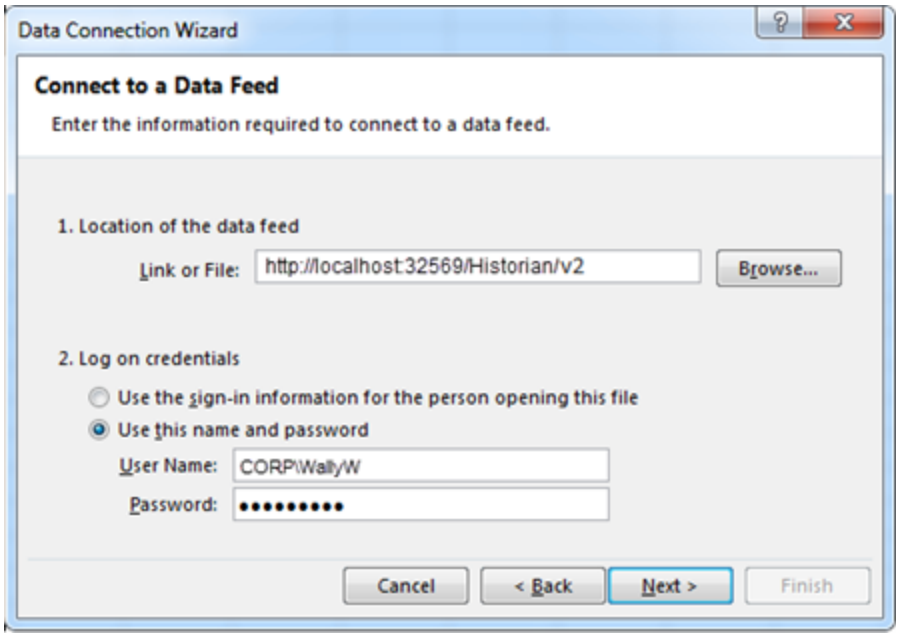
To manually import Historian Client Web data to Excel

1. Open Excel 2013 or later.
2. From **Data**,choose **Get External Data**, choose **From Other Sources**, and then choose **From OData Data Feed**.
3. In the **Link or File** box, specify a URL for your query, formatted as an atom feed (that is, end with "$format=atom’").

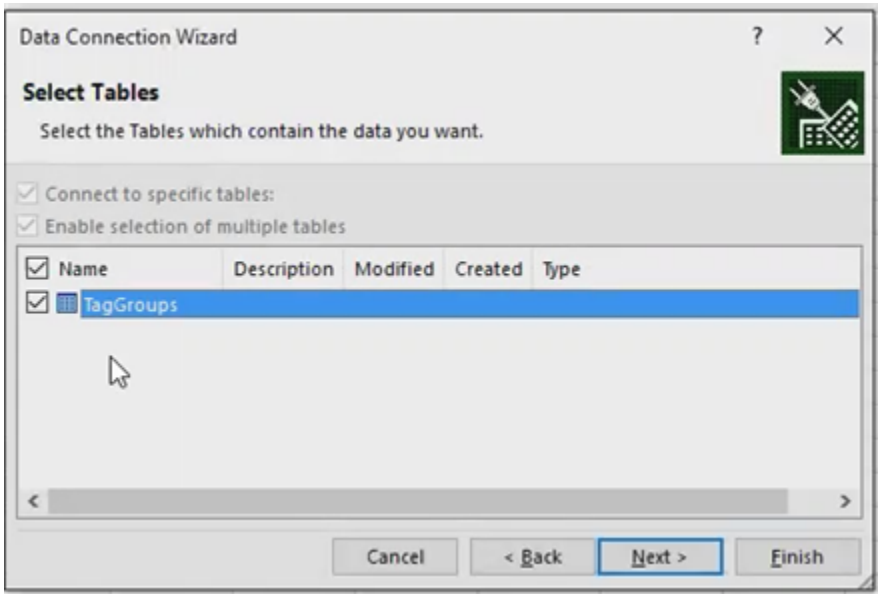
   Here is a query example:

   ```
   http://localhost:32569/Historian/v2/TagGroups?$format=atom
   ```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

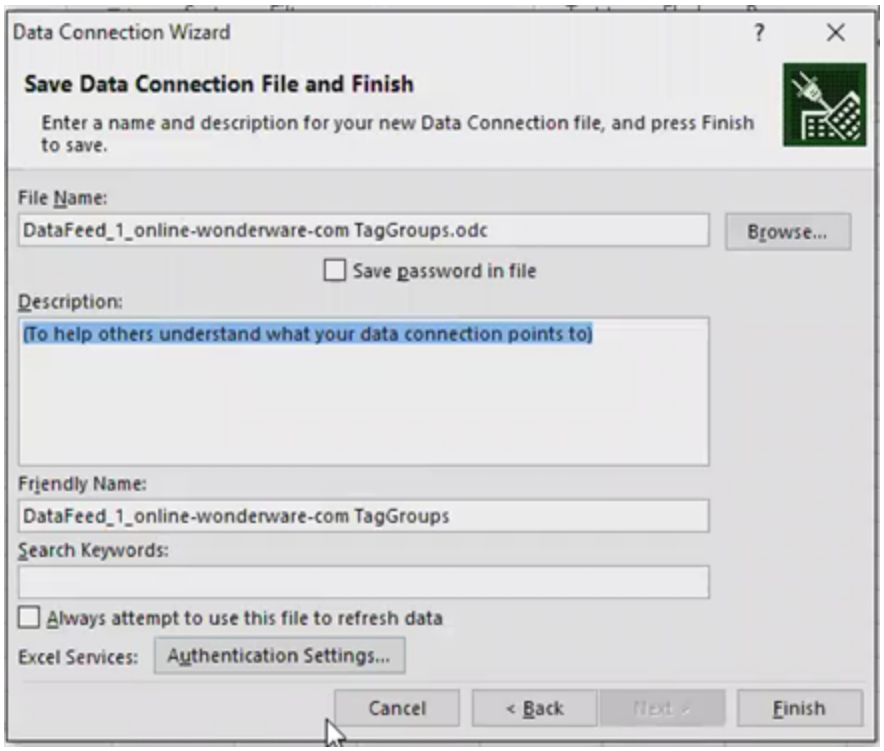


4. Click **Use this name and password**, and specify your Insight user name and password. Click **Next**.
5. In the **Select Tables** box, mark the element to include. Click **Next**.

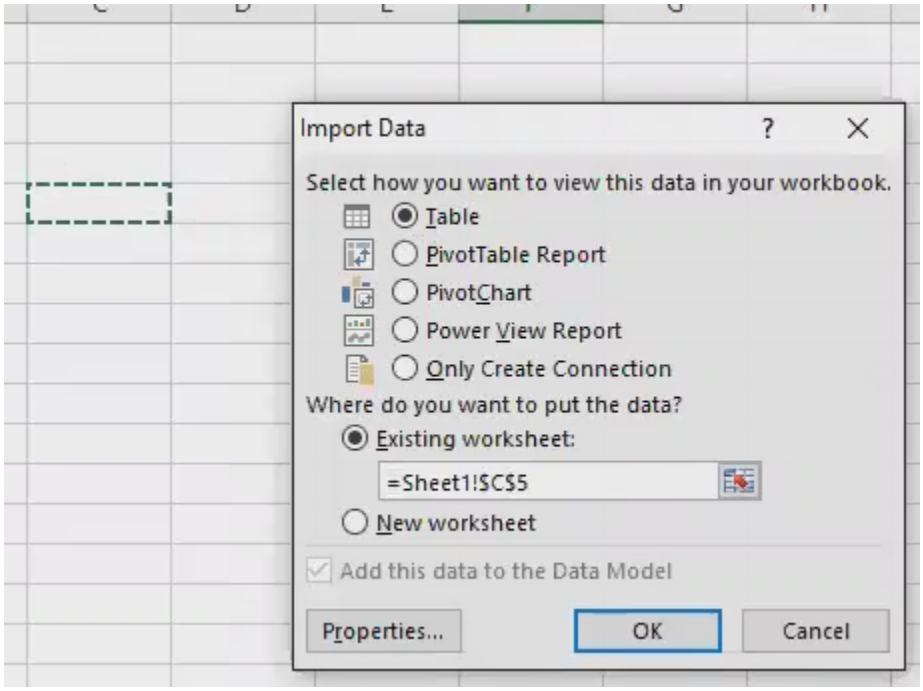


6. Review the information displayed on **Save Data Connection File and Finish** and then click **Finish**. Excel creates an ODC (OData Connection) file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. Specify how and where to place your data in the spreadsheet. Click **OK**.



Excel will place the data where you specified:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GroupID	GroupName	Type	ParentID	Scope
5d181b0f-ba93-4c81-83de-0aae9f946b5b	Wylie	1000000	0	Public
503f8aab-35c0-434a-a256-1a493ac1b539	TestNew	1000000	0	Public
42cc15b9-3bc9-417c-8957-97f22db2acfa	mwoTest	1000000	0	Public
2bb8b991-671b-4835-9207-5b27ff54a2f5	Frankfurt	1000000	0	Public
2b89caa2-e00c-497a-8a73-c897e1084da7	SAMC2016	1000000	0	Public
fb167546-fb19-4405-a17b-c4a5a31eed0f	Canterbury	1000000	0	Public
d8ec8390-c9ba-4192-850f-8790b09a3256	test	1000000	0	Public
8f707b8f-94a8-4a97-adfa-99f2c0518617	InTouch	1000000	0	Public
747fe221-5d98-4c3d-bae8-5ea268d40538	Baytown	1000000	0	Public

Retrieve data using PowerBI

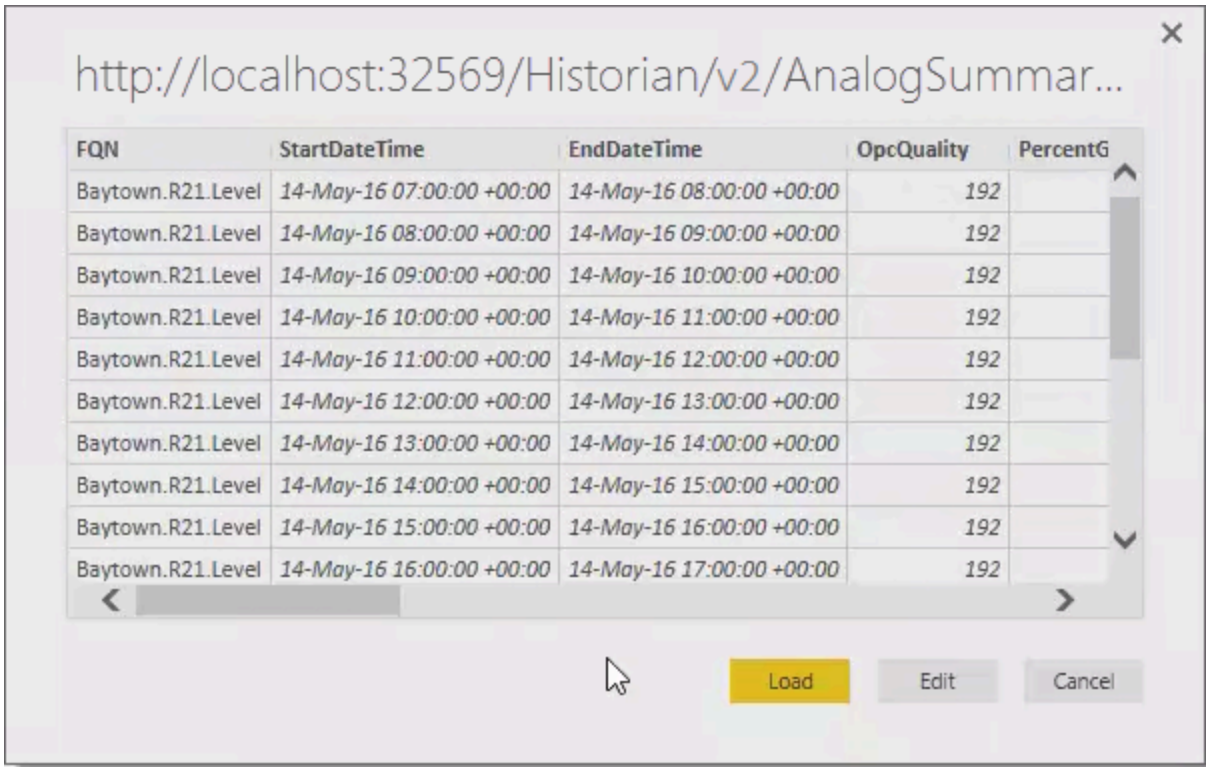
Retrieve data using PowerBI

To import Historian Client Web data to PowerBI

1. Open PowerBI.
2. Click **Get Data**, and choose **OData Feed**.
3. Specify the query URL and click **OK**.

4. Specify **Basic Authentication**, and provide your Insight username and password.
5. Click **Connect**.
6. Power BI shows a preview of your data retrieved from Historian Client Web.

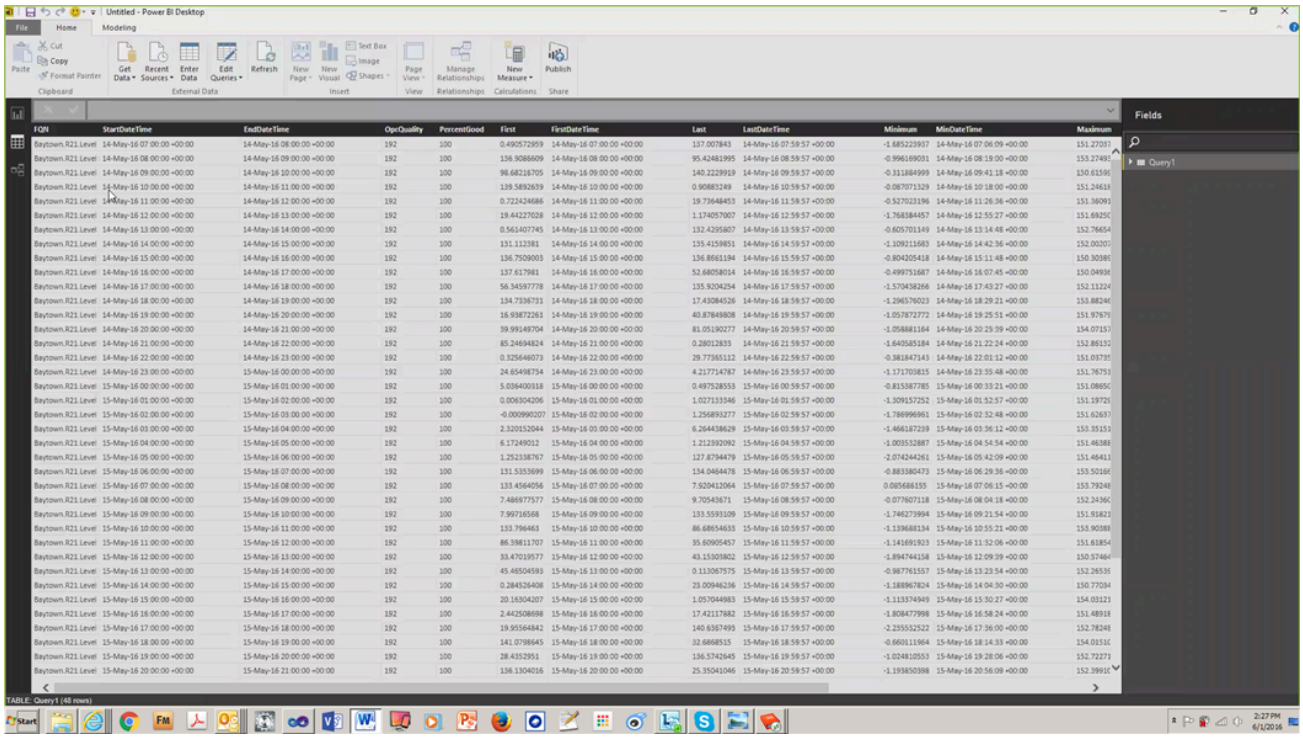
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



FQN	StartDateTime	EndDateTime	OpcQuality	PercentG
Baytown.R21.Level	14-May-16 07:00:00 +00:00	14-May-16 08:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 08:00:00 +00:00	14-May-16 09:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 09:00:00 +00:00	14-May-16 10:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 10:00:00 +00:00	14-May-16 11:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 11:00:00 +00:00	14-May-16 12:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 12:00:00 +00:00	14-May-16 13:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 13:00:00 +00:00	14-May-16 14:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 14:00:00 +00:00	14-May-16 15:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 15:00:00 +00:00	14-May-16 16:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 16:00:00 +00:00	14-May-16 17:00:00 +00:00	192	

7. Click **Load**.

8. In the upper-left corner, click to view the table of results.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval errors

The OData error codes listed in the following table may be returned by an operation on any of the storage services.

Error code	HTTP status code	User message
ConditionNotMet	Not Modified (304)	The condition specified in the conditional header(s) was not met for a read operation.
MissingRequiredQueryParameter	Bad Request (400)	A required query parameter was not specified for this request.
UnsupportedQueryParameter	Bad Request (400)	One of the query parameters specified in the request URI is not supported.
InvalidQueryParameterValue	Bad Request (400)	An invalid value was specified for one of the query parameters in the request URI.
OutOfRangeQueryParameterValue	Bad Request (400)	A query parameter specified in the request URI is outside the permissible range.
RequestUrlFailedToParse	Bad Request (400)	The url in the request could not be parsed.
InvalidUri	Bad Request (400)	The requested URI does not represent any resource on the server.
InvalidHttpVerb	Bad Request (400)	The HTTP verb specified was not recognized by the server.
OutOfRangeInput	Bad Request (400)	One of the request inputs is out of range.
InvalidAuthenticationInfo	Bad Request (400)	The authentication information was not provided in the correct format. Verify the value of Authorizationheader.
InvalidInput	Bad Request (400)	One of the request inputs is not valid.
Unauthorized	Unauthorized (401)	Unauthorized: Access is denied due to invalid credentials.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Error code	HTTP status code	User message
ResourceNotFound	Not Found (404)	The specified resource does not exist.
InternalError	Internal Server Error (500)	The server encountered an internal error. Please retry the request.
OperationTimedOut	Internal Server Error (500)	The operation could not be completed within the permitted time.
ServerBusy	Service Unavailable (503)	The server is currently unable to receive requests. Please retry your request.

Use a URL-based query to create a chart

You can create a chart from a URL query. This is useful when you want to trigger the creation of a chart from another application.

Note: The syntax and parameters for creating a chart with a URL-based query differ between the modern look and classic look. To create a URL-based query for the Historian Client Web classic look, refer to the Historian Client Web classic look instructions.

Parts of a URL-based query

The query must be in this format:

<base URL><path>?Tags=*<tag list><optional parameters, where parameters are joined by "&">*

For example:

```
http://myhistorian:32569/ClientWeb/
content?tags=WeatherApp.Auckland.Cloudiness&chartType=LineChart&startTime=2017-29-04T08:
45:00.000&endTime=2017-03-30T08:45:00.000
```

The URL-based query includes these parts:

Part of the query URL	In the example, it looks like this:
The base URL, including a specific identifier for your Historian solution	http://myhistorian:32569/
Path	ClientWeb/content
An indicator showing that parameters are to follow	?
Query parameters	tags=WeatherApp.Auckland.Cloudiness

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	&chartType=LineChart &startTime=2017-03-29T08:45:00.000 &endTime=2017-03-30T08:45:00.000

Supported query parameters

These query parameters are supported.While tag names are case-sensitive, parameters are not. (Parameters are case-insensitive.)

Parameter	Description
Tags	(required) Defines the tags to include in the chart. Use the case-sensitive, fully qualified name for each tag (that is, the data source name plus the tag name). Separate tag names with commas. For example: Tags=MyDataSource.Pump12.Temp\,MyDataSource.Pump12.Speed
ChartType	Specifies the type of chart to use. Valid chart types are ColumnChart, LineChart, and SummaryChart. For example: ChartType=ColumnChart Default: StatusBoard
StartTime	Specifies the when the reporting time period begins, using this format: yyyy-MM-dd'T'HH:mm:ss.fff For example: StartTime=2017-03-27T08:47:40.763 Including milliseconds is optional. By default, StartTime is in local time (ISO format) for the client computer. Add a **Z** at the end of the time to use UTC instead. For example: StartTime=2017-03-27T08:47:40.763Z Default: Today
EndTime	Specifies the when the reporting time period ends. It must be used in conjunction with StartTime or DurationHours, and cannot be a time in the future. Use the following format: yyyy-MM-dd'T'HH:mm:ss.fff For example: EndTime=2017-03-28T12:47:40.763 Including milliseconds is optional. By default, EndTime is in local time (ISO format) for the client computer. Add a **Z** at the end of the time to use UTC instead. For example:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameter	Description
	EndTime=2017-03-28T12:47:40.763Z Default: Today
DurationHours	Specifies the duration to report, in real number of hours. Use decimals to represent times that are less than an hour, for example 0.25 for 15 minutes. DurationHours is used only if provided. For example: DurationHours=2
sliceBy	Specifies the event aggregate in a column chart. This parameter accepts the fully qualified name of a string tag. If sliceBy is provided in the URL along with resolution or timeAggregate, those parameters are ignored and event aggregate is selected for the chart. If no tag name, or an invalid tag name, is provided for sliceBy, the sliceBy option is ignored. For example: sliceBy=IntouchAlarm.ConcValve1
resolution	Specifies a time aggregate for a column chart in milliseconds. Accepted values are 86400000 (DAY), 3600000 (Hour) and 900000 (15 Min). For example: resolution=900000
timeAggregate	Specifies a time aggregate for a column chart in daily or hourly values. Valid values are: Day, Hour, or Event. If the sliceBy query parameter is used, then timeAggregate is automatically set to Event and any other timeAggregate option is ignored For example: timeAggregate=Day If resolution and timeAggregate are both provided, only resolution is considered.
valueAggregate	Specifies a value aggregate for a chart supporting value aggregate options. Valid values are minimum, maximum, integral, average, or last as long as the value is valid for the corresponding chart.
#hidenavigation	Hides the navigation options. This is helpful to provide an uncluttered view of your content, and to remove the option to navigate elsewhere.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About StartTime, EndTime, and DurationHours

You do not need to specify StartTime, EndTime, and DurationHours.

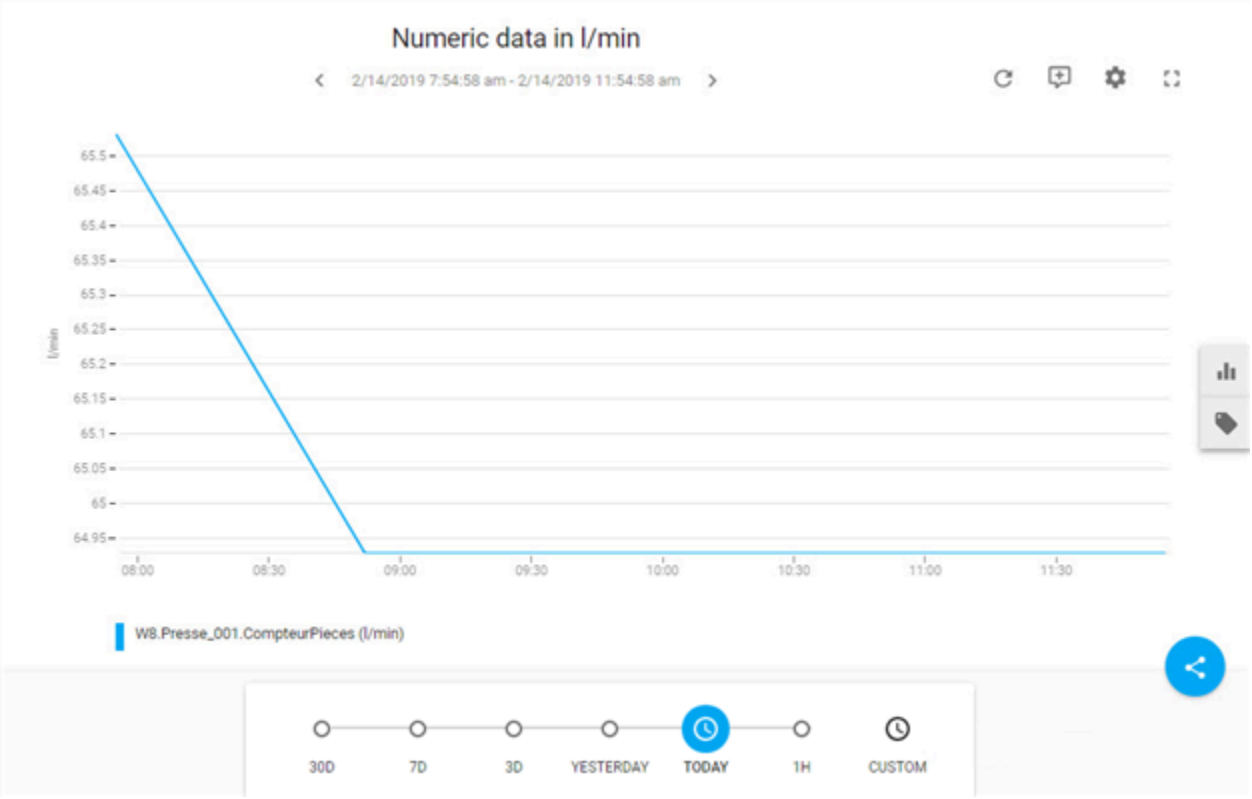
- When all three are specified, Historian Client Web uses StartTime and EndTime, but ignores DurationHours.
- When either StartTime or EndTime is specified along with DurationHours, the two specified parameters are used for StartTime/EndTime calculations.
- When StartTime is specified without DurationHours, CurrentTime and StartTime are used for StartTime/EndTime calculations.
- When none of these three parameters are specified, the chart shows tag values for today ending with the current time.

Example

For example, after logging into Historian Client Web, this URL is entered in a web browser:

```
http://myhistorian:32569/ClientWeb/
content?DurationHours=4&tags=Seattle.W8.Press_001.Pieces&chartType=LineChart
```

This chart displays:



Content Scripting REST API

IMPORTANT: The Content Scripting REST API is in preview and is subject to change with each release.

The Content Scripting REST API allows you to programmatically define layout and data details to create Historian Client Web content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For examples of code to create different types of charts, see the Examples section.

Endpoint URL

Content Scripting REST API uses this endpoint URL:

```
http://localhost:32569/Historian/v2/Contents?overwrite=true
```

The query parameter "overwrite" is used to overwrite existing content with the same name. If this parameter is not provided and content with the same name exists, a duplicate content exists (409) error occurs.

PostData Resource

Description	Describes the data and layout that make up a set of private or shared content.
URL	`http://localhost:32569/Historian/v2/Contents?overwrite=true`
Method	`POST`

Required Parameters	• **Name**
	The name of the content. If content with same name already exist, the API will fail and you'll see "duplicate content exists" as an HTTP status message.
	If you use "overwrite=true", the existing content with same name will be overwritten.
	• **CreatedUserFirstName**
	The first name of user associated with this content. This name displays as a tooltip or in the Content Management page. This name is not validated, so be sure to check spelling, etc. Example:
	`"CreatedUserFirstName": "Wally"`
	• **CreatedUserLastName**
	The last name of user associated with this content. This name displays as a tooltip or in the Content Management page. This name is not validated, so be sure to check spelling, etc. Example:
	`"CreatedUserLastName": "Wonderware"`
	• **UserID**
	The email ID of the user who created this personalized content. This user ID does not get validated, so be sure check for spelling, etc. yourself. Example:
	`UserID:"wally.wonderware@gmail.com"`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **ChartType**
  Specifies the chart type you want to use for this content.

  Examples:

```
"ChartType":"Line Chart"
"ChartType":"Status Board"
"ChartType":"Cumulative Line"
"ChartType":"Summary Chart"
```

  Note: If chart type provided is not one of the value listed,error 401 results.
  For complete code examples that create different types of charts, see the Examples section.

  If the layout is "All Tags" and is the active layout, these chart types are supported:

  - Column Chart
  - Composite Chart
  - Detail Grid
  - Map Chart
  - Status Board (default)
  - XY Plot

  If the layout is created for analog tags, these chart types are supported:

  - Column Chart
  - Cumulative Line
  - Cumulative Sum
  - Detail Grid
  - Diagnostic Grid
  - Line Chart
  - Map Chart
  - Status Board (default)
  - Summary Chart
  - Summary Grid
  - XY Plot

  Note: "Cumulative Sum" is the ChartType value used for a rolling sum chart.

  If the layout is created for non-analog tags, these chart types are supported:\

  - Diagnostic Grid
  - Gantt Chart
  - Map Chart
  - Status Board (default)
  - Summary
  - Summary Grid
  - Time-in-State Chart

- **LayoutInfo**
  Information about the layout. At minimum,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	LayoutInfo must include at minimum TagDetails with some tags value.

Optional Parameters	• **TimePreset**
	The relative or absolute time duration for the selected chart. Supported values are:
	Relative time:
	• 1 = Last 30 days
	• 2 = Last 7 days
	• 3 = Last 3 days
	• 4 = Yesterday
	• 5 = Today (default)
	• 6 = Last hour
	• 8 = This week
	• 9 = Last week
	• 10 = This month
	• 11 = Last month
	Absolute time:
	User can also provide absolute time in UTC format. Start time and end time should be separated by a comma, and start time should be the first. Example:
	`"TimePreset": "2018-01-19T22:23:43.836Z,2018-01-21T22:23:43.836Z"`
	Example of custom duration relative to current time:
	`"TimePreset": "OffsetInMilliSecs:67288974"`
	• **Keywords**
	• An array of keywords for this content. Example: "Keywords":["SecretFormula"]
	• **ShareMode**
	Makes the content public or private. The following values are supported:
	• 73 = Private content
	• 74 = Shared content (default)
	Example:
	`"ShareMode":"74"`
	• **Scope**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The scope for the content to be created. Default value is "/". Examples:
	`"Scope":"/"`
	`"Scope":"/Montreal/"`

Sample Query	`{ "upsert": { "Name": "Column chart ContentWithScript", "UserID": "wally.wonderware@gmail.com", "CreatedUserFirstName": "Wally", "CreatedUserLastName": "Wonderware", "ChartType": "Column Chart", "LayoutInfo": { "TagDetails": [{ "TagName": "Weather.Seattle.Cloudiness", "IsSelected": "true" }, { "TagName": "Weather.Seattle.Wind Direction", "IsSelected": "true" }, { "TagName": "Weather.Seattle.Wind Speed", "IsSelected": "true" }], "TimePreset":"5", "TimeAggregate":"2" }, "Keywords":[] } }`
Notes	• The details returned by this API includes an ID that is used internally to identify the specific content in the UI. • If you display the content you post, and then change sharemode, timepreset, and/or timeaggregate values, you must refresh the displayed content page to see updates. You can refresh the entire page, or search for the content again via the search box.

LayoutInfo

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	Contains the TagDetails and Layout properties.
Required Parameters	• **Layout** An array of layouts to display. The first layout must use the name "All Numeric". See Layout. • **TagDetails** An array of tags to display. See TagDetails.

Sample	"LayoutInfo": {
	"TagDetails": [
	{
	"TagName":
	"alarmstests.W12R12-SQL16.SysPerfAvailableMB
	ytes",
	"IsSelected": "true"
	}
	],
	"Layout": [
	{
	"Name": "MB",
	"IsActiveGroup": "true",
	"LayoutProperties": {
	"selectedOptions": {
	"yAxis": {
	"type": "stacked"
	},
	"scaleY": false
	}
	}
	}
	]
	}

Layout

Layout

Description	This object is used to configure the chart layout. Each Layout object represents the engineering unit group displayed on the left panel. Layout object is used to decide which layout will be selected/active when content is opened, should chart display in stack single or multi axis y mode and another configuration related to the chart. For example, the LayoutProperties object in following post data is used to display line chart in All Numeric group in stack mode, with limit line visible and Y-axis

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	scaled to data available in it.

Required Parameters	• **Name**
	Names the tag group to be used this layout. These are possible values:
	  • **For analog tags listed in the Tags parameter:** The name of any tag group for a specific unit of measure.
	  • **For non-analog tags listed in the Tags parameter:** "" (An empty string), specifying the "String data" tag group will be used.
	  • **If isActiveGroup is set to true** "All Numeric".
	• **isActiveGroup**
	Indicate whether this layout is active. These are possible values:
	`"IsActiveGroup":"true",`
	`"IsActiveGroup":"false",`
	`"IsActiveGroup":""`
	• **LayoutProperties**
	Contains a set of options for the layout. See LayoutProperties.
	• **TimeAggregate**
	For charts that support time aggregation (such as column charts), specifies the time aggregation. Supported values are:
	  • 1 = Day
	  • 2 = Hour
	  • 3 = 15-minute
	  • 4 = Event (only applicable for column charts using sliceBy)
	  Example:
	  `"TimeAggregate": "1"`

LayoutProperties

LayoutProperties

Description	Contains a set of layout options nested within the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	selectedOptions object.
Required Parameters	• **selectedOptions** An array of selectedOptions.

Sample	`"LayoutProperties": {`
	`"selectedOptions": {`
	`"xAxis": {`
	`"label": "DateTime",`
	`"ticks": -1,`
	`"visible": true`
	`},`
	`"yAxis": {`
	`"label": "All Tags",`
	`"ticks": -1,`
	`"visible": true,`
	`"type": "stacked"`
	`},`
	`"limitLine": {`
	`"position": 10,`
	`"text": "Limit Line Text",`
	`"visible": false`
	`},`
	`"legend": {`
	`"visible": true`
	`},`
	`"scaleY": true,`
	`}`
	`}`

selectedOptions

selectedOptions

Description	Defines layout options.

Optional Parameters	• **xAxis**
	Configures the X axis of the specified chart (for example, line chart or composite chart). Parameters are:
	  • **label**
	  Defines the axis label text.
	  • **ticks**
	  Specifies the number of tick marks to display. Use "-1" to display the chart's default ticks.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **visible**
  Specifies whether to make the axis visible (true or false).
- **tag**
  For XY plots, specifies the fully qualified name of a tag used for the X axis.

  Example:

```
"xAxis": {
"label": "Dallas.Wind Speed",
"tag": "Weatherapp.Dallas.Wind
Speed",
"ticks": -1,
"visible": true
},
```

- **yAxis**
  Configures the Y axis of the specified chart (for example, line chart or composite chart). Parameters are:
  - **label**
    Defines the axis label text.
  - **ticks**
    Specifies the number of tick marks to display. Use "-1" to display the chart's default ticks.
  - **visible**
    Specifies whether to make the axis visible (true or false).
  - **type**
    Specifies the type of Y axis to use -- stacked, single, or multi.

    Example:

```
      yAxis: {
      "label": "All Tags",
      "ticks": -1,
      "visible": true,
      "type": "stacked"
      }
```

- **limitLine**
  Defines a limit line for the chart. Parameters are:
  - **position**
    Sets the position on the Y axis for the limit line.
  - **text**
    Defines the limit line text.
  - **visible**
    Specifies whether the limit line displays on the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

chart (true or false).

Example:

```
"limitLine": {
"position": 10,
"text": "Limit Line Text",
"visible": false
}
```

- **legend**
  Specifies whether to include a legend for the chart. There is one parameter:
  - **visible**
    Indicate whether the legend displays (true or false).

    Example:

    ```
    "legend": {
    "visible": true
    }
    ```
- **scaleY**
  For a line chart or composite chart, specifies whether the Y axis scales according to the data in the chart.

  Example:

  ```
  "scaleY": true
  ```
- **scaleX**
  For an XY plot, specifies whether the X axis scales according to the data in the chart.

  Example:

  ```
  "scaleX": true
  ```
- **lookBack**
  For integers in a cumulative sum chart or rolling sum chart, specifies the lookback period.

  Example:

  ```
  "lookBack": 7
  ```
- **units**
  In a cumulative sum chart or rolling sum chart, specifies the units used for the lookBack value -- days or hours.

  Example:

  ```
  "unit": "hours"
  ```
- **sortType**
  In a summary bar chart, indicates the sorting order for tags. Supported values are:
  - Largest to Smallest

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Smallest to Largest
- A-Z

  Example:

```
"sortType":"Largest To Smallest"
```

- **displayLines**
  For XY plots, specifies whether plot points will be joined with a line.

  Example:

```
"displayLines": true
```

- **aggregation**
  For a chart that supports value aggregation (such as a column chart), this sets the selected value aggregation for chart. Supported values are:

  - average
  - integral (analog tags only)
  - last
  - maximum
  - meter (analog tags only)
  - minimum
  - summary (non-analog tags only)
  - total (non-analog tags only)

  Example:

```
"aggregation": "Maximum"
```

- **map**
  For map charts, this sets the selected value aggregation for chart. Parameters are:

  - **zoomLevel**
    Specifies the zoom level of the map. If not specified or invalid, the chart automatically zooms to fit all tags.

    Example:

```
"zoomLevel": 10,
```

  - **center**
    Specifies the geolocation for the center of the map, specified with longitude and latitude.

    Example:

```
"center": {
"Longitude":
-110.44413636978399,
"Latitude": 48.38785248165351
}
```

- **SliceByTag**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	For column charts, this sets an event aggregate for the specified non-analog tag. The value must be a fully qualified tag name. Example: "sliceByTag": "Weather.Auckland.Clouds",

Sample	"LayoutProperties": {
	"selectedOptions": {
	"xAxis": {
	"label": "DateTime",
	"ticks": -1,
	"visible": true
	},
	"yAxis": {
	"label": "All Tags",
	"ticks": -1,
	"visible": true,
	"type": "stacked"
	},
	"limitLine": {
	"position": 10,
	"text": "Limit Line Text",
	"visible": false
	},
	"legend": {
	"visible": true
	},
	"scaleY": true,
	"map": {
	"zoomLevel": 10,
	"center": {
	"Longitude": -110.44413636978399,
	"Latitude": 48.38785248165351
	}
	}
	}
	}

TagDetails

TagDetails

Description	Describes one tag in the layout.
Required Parameters	• **TagName** Provides the fully qualified name of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **IsSelected** Indicate whether this tag is selected. Tags are selected by default.

Sample	`{`
	`"TagName": "_PLPWSP2017_AnE.PLPWSP2017.Smiley.HX",`
	`"IsSelected": "true",`
	`}`

Examples

You can use any client tool to create a Content Scripting API request. Select the links below to examine some scripting examples.

Use Chrome Developer tool to capture content

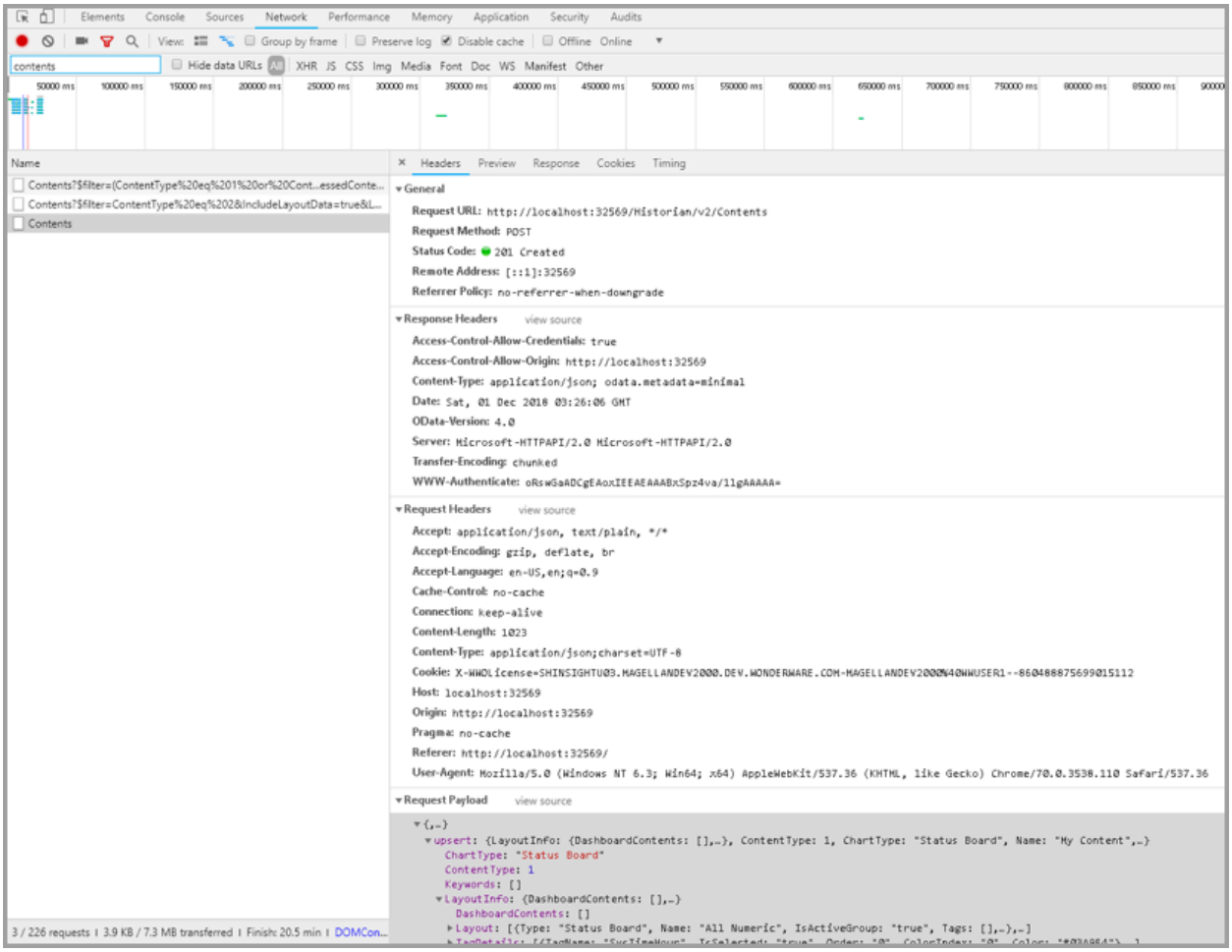
You can use the template from the Chrome Developer tool and a PowerShell script to create content.

To capture content for a post request with Chrome Developer tool

1. Using a Chrome browser, create content in AVEVA Historian Client Web.
2. Press F12 to open the Chrome Developer tool.
3. In the Chrome Developer tool:
   a. Select the **Network** tab.
   b. Click ⊘ to clear the tool's displayed information.
4. In AVEVA Historian Client Web, save your content.
5. In the Developer tool:
   a. Click the "Content" line to see details.
   b. In the **General** section, note the Request URL:

   c. In the **Request Payload** section, copy all the content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



6. Paste the request payload into a JSON formatter and make any edits.
   - Remove these items: URL property, CreationDate, LastModifiedDate, LastAccessedDate, and SharedDate, the numeric portion of the UserID (retain the email address), and TenantID.
   - Optionally, modify these items as needed: FavoriteName, UserID (email address only), and UserName.

   Thjs is an example of the resulting code:

```
{
"upsert": {
"LayoutInfo": {
"DashboardContents": [
],
"Layout": [
{
"Type": "Column Chart",
"Name": "All Numeric",
"IsActiveGroup": "true",
"Tags": [
],
"UIProperties":
"{\"version\":{\"major\":1,\"minor\":5,\"timestamp\":\"\"},\"selectedOptions\":
{\"limitLine\":{\"visible\":false,\"position\":10,\"text\":\"Limit Line
Text\"},\"legend\":
{\"visible\":true},\"xAxis\":{\"label\":\"DateTime\"},\"yAxis\":{\"label\":
\"All Tags\"},\"dates\":{\"from\":
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
\"2018-11-15T19:30:00.000Z\",\"to\":\"2018-11-15T20:04:33.099Z\"},\"aggregation\"
:
\"Maximum\",\"isChartOptionAvailable\":true,\"isTimeAggregateAvailable\":
true,\"isValueAggregateAvailable\":true,\"isCSVDownloadAllowed\":true,\"timeAggre
gate\":\"15MINUTES\"}}"
},
{
"Type": "",
"Name": "None",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
},
{
"Type": "",
"Name": "65F",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
}
],
"TagDetails": [
{
"TagName": "Weather.Auckland.Daily Cooling Degrees",
"IsSelected": "false",
"Order": "",
"ColorIndex": "undefined",
"Color": "#eeeeee"
},
{
"TagName": "Weather.Auckland.Weather ID",
"IsSelected": "true",
"Order": "0",
"ColorIndex": "1",
"Color": "#AD1457"
}
],
"TimeAggregate": 3,
"TimePreset": "OffsetInMilliSecs:1700224"
},
"ContentType": 1,
"ChartType": "Column Chart",
"Name": "Quarterly Report",
"Keywords": [
],
"Location": "/",
"SharedMode": 98
}
}
```


7. Paste the resulting code (shown below in `blue`) in Powershell as a POST request.

   For the **-Uri** parameter, use the Request URL from step 7 (shown in red below).

   For example:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
$header = @{
"Content-Type"="application/json;charset=UTF-8"
};
$postdata='{
"upsert": {
"LayoutInfo": {
"DashboardContents": [
],
"Layout": [
{
"Type": "Status Board",
"Name": "All Numeric",
"IsActiveGroup": "true",
"Tags": [
],
"UIProperties":
"{\"version\":{\"major\":1,\"minor\":5,\"timestamp\":\"\"},\"selectedOptions\":{
\"dates\":{\"from\":\"2018-12-01T00:00:00.000Z\",\"to\":\"2018-12-01T03:25:52.10
8Z\"},\"isCSVDownloadAllowed\":true}}"
},
{
"Type": "",
"Name": "Minute",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
},
{
"Type": "",
"Name": "Hour",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
},
{
"Type": "",
"Name": "Second",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
}
],
"TagDetails": [
{
"TagName": "SysTimeHour",
"IsSelected": "true",
"Order": "0",
"ColorIndex": "0",
"Color": "#03A9F4"
},
{
"TagName": "SysTimeMin",
"IsSelected": "true",
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"Order": "1",
"ColorIndex": "1",
"Color": "#AD1457"
},
{
"TagName": "SysTimeSec",
"IsSelected": "true",
"Order": "2",
"ColorIndex": "2",
"Color": "#F57F17"
}
],
"TimeAggregate": 2,
"TimePreset": 5
},
"ContentType": 1,
"ChartType": "Status Board",
"Name": "My Content",
"Keywords": [
],
"Location": "",
"SharedMode": 73
}
}'
try{
$response = Invoke-WebRequest -Uri http://localhost:32569/Historian/v2/Contents
-Method Post -Body $PostData -Headers $header -ErrorVariable RespErr
-UseDefaultCredentials
$response;
}
catch [System.Net.WebException] {
if($_.Exception.Response.StatusCode -eq "Conflict"){
Write-Host "It's a duplicate Content"
}else{
Write-Host "The Request failed.";
$respStream = $_.Exception.Response.GetResponseStream()
$reader = New-Object System.IO.StreamReader($respStream)
$ResponseContent = $reader.ReadToEnd();
$ResponseContent;
$RespErr;
}
}
```

Create content with minimum required properties

This example shows the minimum required properties for postData. It creates the default status board chart using the "All Tags" group and the specified tags.

```
{
"upsert": {
"Name": "ContentWithScript",
"ChartType": "Status Board",
"LayoutInfo": {
"TagDetails": [
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
{
"TagName": "Weatherapp.Richmond.Wind Speed"
}
]
}
}
}
```

Create content using all properties

This postData example uses all properties to create an XY plot.

```
{
"upsert":{
"Name": "Wind Speed",
"ChartType": "XY Plot",
"TimePreset": "OffsetInMilliSecs:16028059",
"TimeAggregate": "2",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"xAxis": {
"label": "Dallas.Wind Speed",
"tag": "Weatherapp.Dallas.Wind Speed",
"ticks": -1
},
"yAxis": {
"type": "single"
},
"limitLine": {
"position": 10,
"text": "Limit Line Text",
"visible": false
},
"legend": {
"visible": true
},
"scooter": {
"visible": true
},
"sortType": "SmallesttoLargest",
"lookBack": 7,
"units": "hours",
"scaleX": true,
"scaleY": true,
"displayLines": false,
"aggregation": "avg_xy"
}
}
}
],
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"wind"
]
}
}
```

Create content with PowerShell

This creates a status board using the "All Tags" tag group.

Note: The header requires an API key. For information on creating an API key, see Acquire an API key for data retrieval.

```
$header = @{
"Content-Type"="application/json;charset=UTF-8"
};
$PostData = '{
"upsert": {
"Name": "ContentWithScript",
"ChartType": "Status Board",
"LayoutInfo": {
"TagDetails": [
{
"TagName": "Weatherapp.Richmond.Wind Speed"
}
]
}
}
}
'
try{
$response = Invoke-WebRequest -Uri http://localhost:32569/Historian/v2/
Contents?overwrite=true -Method Post -Body $PostData -Headers $header -ErrorVariable
RespErr
$response;
}
catch [System.Net.WebException] {
if($_.Exception.Response.StatusCode -eq "Conflict"){
Write-Host "This is a duplicate content"
}else{
Write-Host "The Request failed.";
$respStream = $_.Exception.Response.GetResponseStream()
$reader = New-Object System.IO.StreamReader($respStream)
$ResponseContent = $reader.ReadToEnd();
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
$ResponseContent;
}
}
```

Create a column chart

This example produces a column chart using the "MB" tag group and these parameters:

- Value aggregation is maximum
- Time aggregation is 15 minutes

```
{
"upsert": {
"Name": "Column Chart",
"TimePreset": "5",
"UserName": "Wally",
"UserID": "wally.wonderware@gmail.com",
"TimeAggregate": "3",
"LayoutInfo": {
"Layout": [
{
"Name": "MB",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"aggregation": "maximum"
}
}
}
],
"TagDetails": [
{
"TagName": "alarmstests.W12R12-SQL16.SysPerfAvailableMBytes",
"IsSelected": "true"
}
],
}
}
}
```

Create a column chart with sliceBy support

This example produces a column chart with the sliceBy parameter. Note that:

1. The layout selected is “All Numeric” and the associated IsActiveGroup property is set to true.
2. Analog and non-analog tags are added in the TagDetails.
3. The sliceByTag propetry under LayoutProperties in the active layout should be set to a non-analog tag. The tag is added through TagsDetails.
4. timeAggregate under LayoutProperties in the active layout should be set to EVENT.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
{
"upsert": {
"LayoutInfo": {
"Layout": [
{
"Type": "Column Chart",
"Name": "All Numeric",
"IsActiveGroup": "true",
"Tags": [
],
"LayoutProperties": {
"selectedOptions": {
"limitLine": {
"visible": false,
"position": 10,
"text": "Limit Line Text"
},
"aggregation": "Integral",
"sliceByTag": "Weather.Auckland.Clouds",
"timeAggregate": "EVENT"
}
}
},
{
"Type": "",
"Name": "",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
},
{
"Type": "",
"Name": "None",
"IsActiveGroup": "false",
"Tags": [
],
"UIProperties": ""
}
],
"TagDetails": [
{
"TagName": "Weather.Auckland.Weather ID",
"IsSelected": "true"
},
{
"TagName": "Weather.Auckland.Clouds Text",
"IsSelected": "false"
}
],
"TimeAggregate": 4,
"TimePreset": 2
},
"UserID": "wally.wonderware@gmail.com",
"CreatedUserFirstName": "Wally",
"ContentType": 1,
```


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"ChartType": "Column Chart",
"Name": "Test ContentAPI-Layout-Case",
"Keywords": [
"sliceby"
],
"Location": "/",
"SharedMode": 98
}
}
```

Create a composite chart

This example produces a composite chart using the All Numeric tag group.

```
{
"upsert": {
"Name": "ContentWithScript",
"UserID": "wally.wonderware@gmail.com",
"CreatedUserFirstName": "Wally",
"CreatedUserLastName": "Wonderware",
"ChartType": "Composite Chart",
"LayoutInfo": {
"TagDetails": [
{
"TagName": "alarmstests.W12R12-SQL16.SysPerfAvailableMBytes",
"IsSelected": "true"
}
]
},
"Scope": "/"
}
}
```

Create a Gantt chart

This example produces a Gantt chart. The Name property of the layout is empty, so the "String data" tag group is used.

```
{
"upsert": {
"Name": "Gantt Chart ContentWithScript",
"ChartType": "Gantt Chart",
"LayoutInfo": [
{
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Direction Text",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Auckland.Wind Speed Text",
"IsSelected": "true"
}
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
]
}
]
}
}
```

Create a line chart

This example produces a line chart using the following parameters:

- The chart type is a line chart that uses the "MB" tag group.
- The Y-axis is stacked and not scaled.
- A keyword ("SecretRecipe") is assigned to this content.
- Time duration for this content is "Today" (where TimePreset is 5).

```
{
"upsert": {
"Name": "Line Chart ContentWithScript",
"ChartType": "Line Chart",
"LayoutInfo": {
"Layout": [
{
"Name": "MB",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"yAxis": {
"type": "stacked"
},
"scaleY": false
}
}
}
],
"TagDetails": [
{
"TagName": "alarmstests.W12R12-SQL16.SysPerfAvailableMBytes",
"IsSelected": "true"
}
]
}
"Keywords": [
"SecretRecipe"
]
}
}
```

Create a map chart

This example produces a map chart using the following parameters:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- The chart type is a map chart that uses the "mph" tag group.
- The map is zoomed to level 10 and centered at specified latitude and longitude coordinates..
- A keyword ("Windy") is assigned to this content.

```
{
"upsert": {
"Name": "Map Chart ContentWithScript",
"UserID": "wally.wonderware@gmail.com",
"CreatedUserFirstName": "Wally",
"CreatedUserLastName": "Wonderware",
"ChartType": "Map Chart",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"map": {
"zoomLevel": 10,
"center": {
"Longitude": -110.44413636978399,
"Latitude": 48.38785248165351
}
}
}
}
}
}
],
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"Windy"
],
"Scope": "/"
}
}
```

Create a rolling sum chart

This example creates a rolling sum chart using the "mph" tag group.

```
{
"upsert": {
"Name": "ContentWithScript",
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"ChartType": "Cumulative Sum",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"lookBack": 7,
"units": "hours"
}
}
}
],
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"wind"
]
}
}
```

Create an XY plot

This example produces an XY Plot chart using the "mph" tag group. it specifies a tag for the X axis and sets scaling to "true" for both X and Y axes.

Note: The layout object corresponds to the engineering unit of X axis tag should be present in layout object.

```
{
"upsert": {
"Name": "XYPlot Chart ContentWithScript",
"ChartType": "XY Plot",
"TimePreset": "OffsetInMilliSecs:16028059",
"TimeAggregate": "2",
"LayoutInfo": {
"Layout": [
{
"Name": "mph",
"IsActiveGroup": "true",
"LayoutProperties": {
"selectedOptions": {
"xAxis": {
"label": "Dallas.Wind Speed",
"tag": "Weatherapp.Dallas.Wind Speed"
},
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"scaleX": true,
"scaleY": true,
"displayLines": false,
"aggregation": "avg_xy"
}
}
}
],
"TagDetails": [
{
"TagName": "Weatherapp.Auckland.Wind Speed",
"IsSelected": "true"
},
{
"TagName": "Weatherapp.Dallas.Wind Speed",
"IsSelected": "true"
}
]
},
"Keywords": [
"windy"
]
}
}
```


Errors

Duplicate content

If content name already exist and "Overwrite=true" is not used, the API call fails with http status 409 and generates this message:

```
Already content exists with name 'ContentWithScript'
```

Unsupported parameter

If PostData has any unsupported parameter, the API call fails with http status 400 and generates this message:

```
Content should have valid name
Invalid post data. ChartType property does not have valid value.
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Server Administrator Guides

This section includes guides for administrators of Historian server.

Historian Administrator Guide

Welcome

This guide provides information about administering and maintaining installed AVEVA Historian servers. This guide describes the tools to administer the historian, as well as how to configure the system to start storing plant data. This guide also describes administrative tasks such as changing the default security, configuring system-wide parameters, and monitoring the system.

The AVEVA Historian software is tightly integrated with Microsoft products. A working knowledge of both Microsoft SQL Server and Microsoft Windows operating systems is required. You should be familiar with administering Microsoft SQL Server and understand how to use the administrative tools provided with Microsoft Windows operating systems.

For more information on Microsoft SQL Server or the Microsoft Windows operating system, see your Microsoft documentation.

AVEVA Historian Documentation Set

The AVEVA Historian documentation set includes the following guides:

- *AVEVA System Platform Installation Guide*
  This guide provides information on installing the AVEVA Historian, including hardware and software requirements and migration instructions.
- *AVEVA Historian Concepts Guide*
  This guide provides an overview of the entire AVEVA Historian system and its key components.
- *AVEVA Historian Scenarios Guide*
  This guide discusses how to use AVEVA Historian to address some common customer scenarios.
- *AVEVA Historian Administration Guide*
  This guide describes how to administer and maintain an installed AVEVA Historian, such as configuring data acquisition and storage, managing security, and monitoring the system.
- *AVEVA Historian Retrieval Guide*
  This guide describes the retrieval modes and options that you can use to retrieve your data.
- *AVEVA Historian Database Reference*
  This guide provides documentation for all of the AVEVA Historian database entities, such as tables, views, and stored procedures.
- *AVEVA Historian Glossary*
  This guide provides definitions for terms used throughout the documentation set.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

In addition, the *AVEVA License Manager Guide* describes the AVEVA License Manager and how to use it to install, maintain, and delete licenses and license servers on local and remote computers.

Getting Started

About AVEVA Historian Licensing

You must have a license to run AVEVA Historian. Your license allows for a certain number of tags on your server. For example, you may have a 5,000-tag license. As you add tags, the License Server activates a portion of the license, in 50-tag increments. For example, if you initially add 120 tags to AVEVA Historian, License Server activates three 50-tag increments.

If you have used the full number of tags allowed by your license, you can add an incremental license (for example, to add another 25,000 tags) to your existing license.

If your AVEVA Historian server is an Enterprise Server that you are also using as a replication (tier-2) server for another historian, all of the replicated tags stored on this server count against this server's license.

Your AVEVA Historian license does not limit the number of these types of tags on your server:

- Locally replicated tags (that is, the original tag is on the same server)
- Auto-summary tags

If your license expires, or you don't yet have a license, you can still:

- Continue acquiring and storing tags indefinitely
- Retrieve values for up to 32 tags for the last 7 days

  For more information about licensing AVEVA Historian and other AVEVA products, see the *AVEVA Licensing Guide*.

Viewing License Information

To view license information

1. In the Operations Control Management Console, expand a server group, then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **View License Information**. The **License Information** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

This dialog box shows the type of license and, if applicable, the expiration date of the license for your AVEVA Historian.

Depending on the license you have, your system is allocated a certain number these resources:

- **Tag Count** -- Shows the total number of tags you can configure for data retrieval in AVEVA Historian. The **Used** column shows how many tags you are currently using.
- **History Duration (Days)** -- Shows the maximum number of days for which historical data can be retrieved.
- **Replication** -- Indicates whether data replication is allowed for this system.

Note: The license status and license tag count (allocated and used) also display in the system status window. See Viewing the Current System Status.

Refreshing the License Information

The Configuration Manager provides updated license information according to the license refresh rate (once an hour for server features). However, you can manually force the Configuration Manager to read the license file and refresh the license information. For example, if you recently added incremental tags to your license, you can manually refresh the license information to show that update.

To refresh the license information

1. In the Operations Control Management Console, expand a server group, then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Refresh License Information**. A confirmation dialog box appears.
3. Click **OK**.

Registering AVEVA Historian Servers

When you install AVEVA Historian, it registers the local machine as a historian server. If you want to administer remote historian servers, you must first register them within the console. When you register a server, you are giving the Operations Control Management Console a logical name and login IDs to connect to both:

- The AVEVA Historian Configuration Manager.
- The Microsoft SQL Server database.

You can register and administer multiple historians from within a single instance of the console. When registering a server, a list of your previously registered servers is available for selection.

To be able to administer the historian (for example, start and stop the server), you must provide a Windows

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

security login that has administrative rights on the AVEVA Historian computer. You also must be logged in as a Historian administrator, with the aaAdministrators database role enabled. If you are using the console remotely for the AVEVA Historian, you do not need to be an administrator on the computer on which you are using the console on.

If you do not supply the login when you register the server, you are prompted to supply it when you attempt to execute an administrative command. If the login you supply does not have administrative permissions, the Management Console is set to read-only mode.

The SQL Server login you use must have the "aaAdministrators" database role to make changes to the historian system configuration, as it is stored in the Runtime database. By default, Windows accounts that are members of the local Windows "aaAdministrator" group are assigned this role. If you do not log in with the SQL Server administrative permissions, functionality is restricted. You must have aaPowerUsers capability enabled to make tag-level changes.

All registration information associated with a particular server name is stored in the Windows registry on the computer running the Operations Control Management Console, not in the console file (.MSC). In addition, all registration information is stored according to the current user. This has the following implications:

- If you register the same historian in multiple console files (.MSC), and you then edit the status or configuration for the historian in one .MSC file, the status and configuration is reflected in the other .MSC files in which that historian appears.
- If you copy a saved .MSC file from one computer to another, the registration properties for a particular historian are not copied with the .MSC file.
- The same historian can have different registration properties for each user who logs on to the Operations Control Management Console computer, even though all users may be using the same .MSC file.

Important: In 2022, Microsoft is releasing a phased update to address a security issue with DCOM on Windows. After the third phase of this update is applied, administering remote historian servers will no longer be possible using the Operations Control Management Console. Instead, you can administer remote Historian servers by first connecting with the remote desktop software of your choice, and then using the Operations Control Management Console on the remote server.

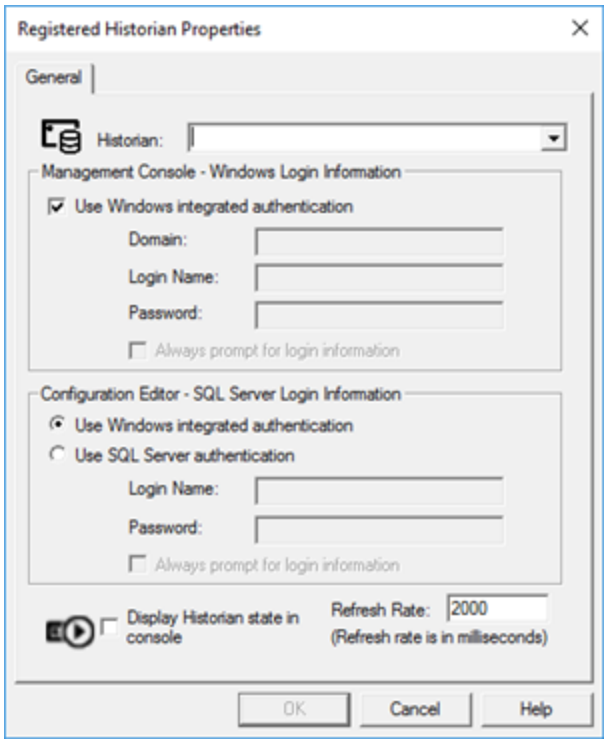
For more up-to-date information about the vulnerability, and a timetable for its phased release, see https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-26414.

Registering an AVEVA Historian

To register an AVEVA Historian:

1. In the Operations Control Management Console tree, right-click the server (or group) and then click **New Historian Registration**. The **Registered Historian Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. In the **Historian** box, either type the name of a new server to register or select a previously registered server from the list. If you select a previously registered server, all options saved for that server appear in the dialog box. If you edit these options and click OK, the new settings are saved.
3. In the **Management Console - Windows Login Information** area, choose a method for the Management Console to connect to the Configuration Manager. The Configuration Manager runs as a Windows service on the historian computer.
   a. Select **Use Windows integrated authentication** to connect to the Configuration Manager using the Windows user account logged into the system.
   b. To use a specific Windows user account, clear **Use Windows integrated authentication**, then enter specific user credentials:

      Domain

      Name of the Windows domain in which the login is validated. A domain is a group of computers that share a central database for security authentication.

      Login Name

      Valid login name for Windows.

      Password

      Valid login password for Windows.
   c. If you select **Always prompt for login information**, stored login information is not used and, instead, a login prompt appears each time access is required.
4. In the **Configuration Editor - SQL Server Login Information** area, configure the login that the Configuration Editor uses to authenticate to the associated Microsoft SQL Server.

   Note: Use the correct case for login IDs and passwords if your database is case-sensitive.

   a. Select **Use Windows integrated authentication** to use the Windows user account logged into the system. The logged-in user should be a valid user on the AVEVA Historian computer, and the user

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

account needs to be assigned to the proper Runtime database roles.

b. To use a valid SQL Server login, click **Use SQL Server authentication**. The following options become available:

Login Name

Valid login ID for the SQL Server.

Password

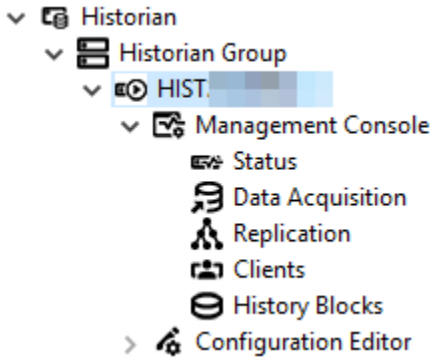
Valid login password for the SQL Server.

Always prompt for login information

If selected, stored login information is not used and, instead, a login prompt appears each time access is required.

Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.

5. If you select **Display Historian state in console**, the Historian's icon in the Operations Control Management Console indicates its running status:



Icon	Status
	Server is running
	Server is shutting down
	Server is not running
	Server is starting up

If **Display Historian state in console** is not selected, the Historian uses the default icon .

6. In the **Refresh Rate** box, type the rate at which the status, client connections, and data acquisition information are refreshed in the details pane. You can specify a value of 0 or between 500 ms and 86,400,000 ms. If you set this rate to 0, the server status is checked one time when the console opens. After that, you need to manually refresh the details pane.
7. Click **OK**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Editing Registration Properties

To edit registration properties for a historian

1. In the Operations Control Management Console tree, right-click the server, and then click **Edit Historian Registration Properties**. The **Registered Historian Properties** dialog box appears.

   For information on the options in this dialog box, see Registering an AVEVA Historian.
2. Edit the properties and then click **OK**.

Deleting a Registered Historian

Deleting a registered historian server simply removes it from the Operations Control Management Console list. All registration options are stored along with the server name in case you want to register it again later.

To delete a registered historian

1. In the Operations Control Management Console tree, right-click the server and then click **Delete**. You are prompted to confirm the deletion.
2. Click **Yes**.

Moving a Registered Server to a Different Group

Moving a server to a different group within the console requires deleting it and then registering it again under the target group.

To move a registered server to a different server group

1. In the console tree, delete the server you want to move. For more information, see Deleting a Registered Historian.
2. Right-click the group to which you want to move the server, and then click **New Historian Registration**. The **Registered Historian Properties** dialog box appears.
3. In the **Historian** box, select the server you just deleted from the list.
4. Click **OK**.

About Administrative Tools

This section describes key administrative tools that you will use with AVEVA Historian:

- Operations Control Management Console, which includes the Management Console and Configuration Editor
- SQL Server Management Studio

AVEVA Historian also includes some data import tools. For more information, see AVEVA Historian Database Export/Import Utility and Historian Data Importer.

In addition, see Microsoft documentation for administrative tools included with the Windows operating system and SQL Server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About the Operations Control Management Console

With the Operations Control Management Console, you can:

- Start and stop the AVEVA Historian software
- Monitor the system
- Make configuration changes

The Operations Control Management Console is a saved Microsoft Management Console (MMC) file, which has an .msc extension. Microsoft Management Console is a container application that can host one or more third-party snap-in applications.

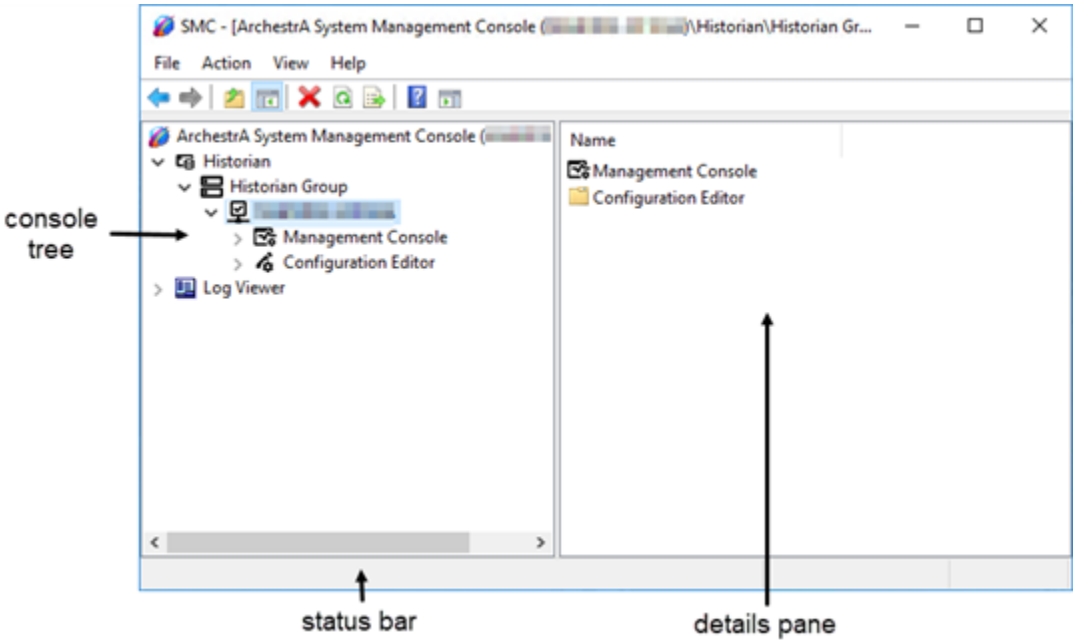
The snap-in for the historian includes a main console tree to add one or more servers that can be administered. The console tree functions like Windows Explorer or the folder view in Internet Explorer. The snap-in also includes areas for monitoring and controlling each historian in the console tree, as well as for configuring each server.

The Operations Control Management Console can be installed on a different computer than the historians you want to administer. You can perform all monitoring and administrative tasks from a single computer anywhere on your network.

Some of the general functionality of the Operations Control Management Console is provided by the MMC container. See the Microsoft Management Console documentation for for more detailed information on using the MMC.

Note: The Operations Control Management Console is different from regular consoles in that you can alter the position of the first column. Also, when you shut down the console and restart it, any changes to the column layout are not persistent.

The Operations Control Management Console window consists of two main areas: the console tree and the details pane.



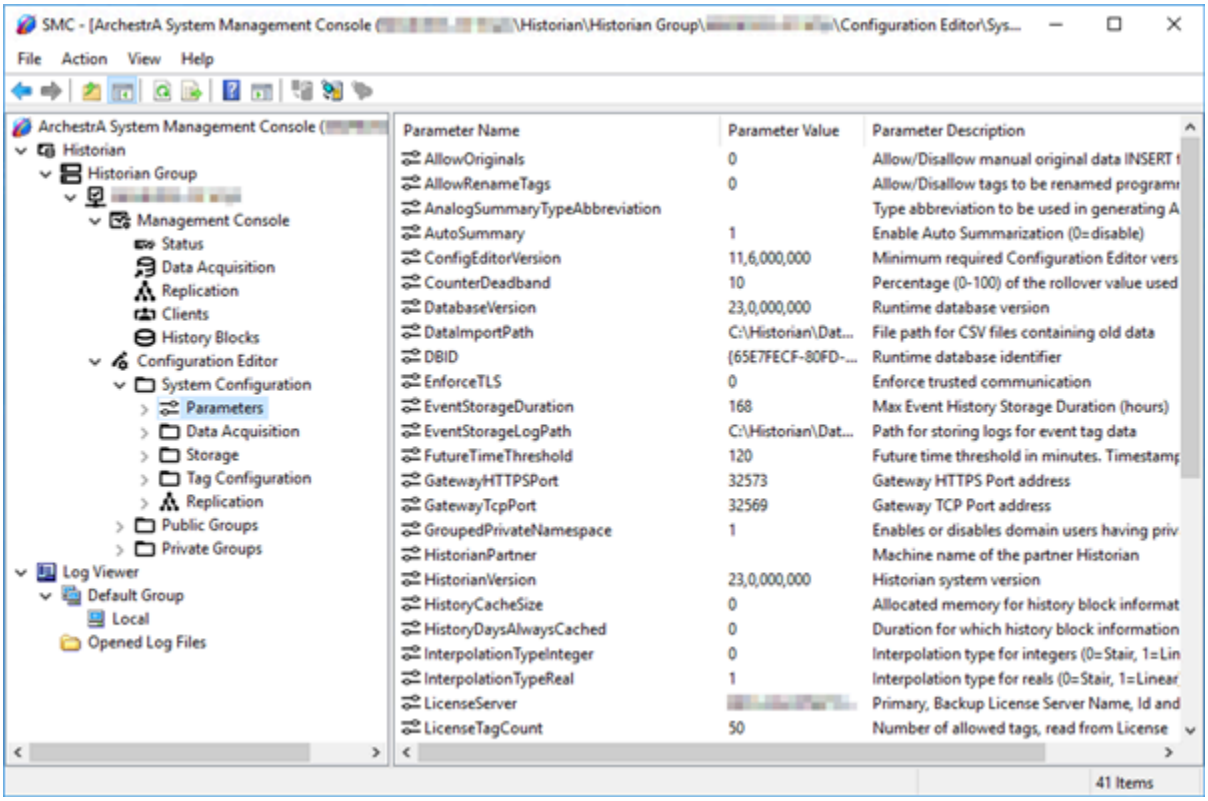
The console tree (also called the scope pane) contains all items available within the console. For the AVEVA Historian software, this includes the registered servers, the Management Console, and the Configuration Editor.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Additional ArchestrA consoles, such as the Log Viewer, may appear in the Operations Control Management Console.

If the Operations Control Management Console is installed on the same computer as the historian, the server is automatically registered and appears under the default **Historian Group** item in the console tree. However, if the Operations Control Management Console is installed on a remote computer, you must register a historian. For more information, see Registering AVEVA Historian Servers.

The details pane (also called the results pane) shows the relevant data pertaining to the item currently selected in the console tree.



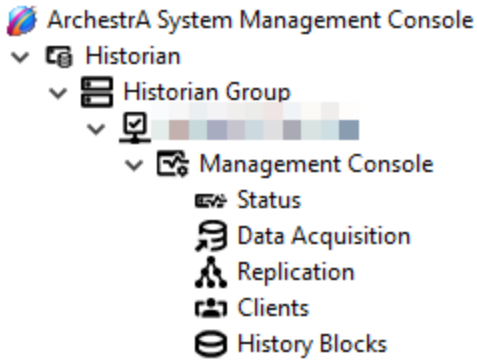
If you double-click an item in the details pane, a Properties dialog box appears, if applicable.

For some of the tree items, you can export all associated information shown in the details pane to a text file. Exported items include History Blocks and anything under the Configuration Editor item. You can save a particular sub-range of rows by first highlighting them with the mouse. To export, right-click the parent item in the console tree pane and then click Export List. You can open the file using any text editor and then print the data.

About the Management Console

You can use the Management Console portion of the main console tree to start and stop the AVEVA Historian, as well as perform some system-level tasks, such as monitoring the status of the server and resetting error counts.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

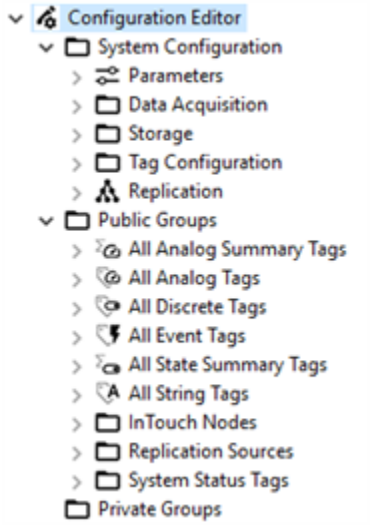


If you have multiple historian servers registered in the console, make sure that you select the server you want to manage before you right-click in the tree to select a short-cut menu command.

Note: Before you can use the Operations Control Management Console to administer an AVEVA Historian, the historian server must be registered within the application. You can add and register any server that you can connect to on the network. Also, if you are administering many servers, you can organize them into groups in the console tree.

About the Configuration Editor

Use the Configuration Editor portion of the console tree to configure the AVEVA Historian.



For example, the Configuration Editor allows you to:

- Import a tag data dictionary from an InTouch application. For more information, see Importing an InTouch Data Dictionary.
- Add, edit, and delete tags. For more information, see Defining Tags.
- Configure data acquisition, such as I/O Servers, topics, and tags. For more information, see Configuring Data Acquisition.
- Configure paths to storage locations. For more information, see Managing Data Storage.
- Administer system-wide properties, such as modification tracking. For more information, see Viewing or Changing System-Wide Properties.
- Configure replication servers, groups, and tags. For more information, see Managing and Configuring

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Replication.

- Create groups in the public and private folders. For more information, see Creating Server Groups.

Note: You can also manage classic event definitions from here. For more information, see Configuring Classic Events.

Configuration Editor Toolbar Buttons

Toolbar buttons specific to the Configuration Editor are:

Button	Description
	Add a new item under the currently selected item in the console tree.
	Open a **Properties** dialog box for the currently selected item in the details pane.
	Delete the currently selected item.
	Start the wizard to add an analog tag.
	Start the wizard to add a discrete tag.
	Start the wizard to add an event tag.
	Start the wizard to add a string tag.
	Commit database changes to the system. For more information, see Dynamic Configuration.
	Start the Tag Importer wizard. For more information, see Importing an InTouch Data Dictionary.
	Open a dialog box to search for database modifications. For more information, see Tracking Modifications.
	Open the **Tag Finder** dialog box to search for tags to add to a tag grouping in the console tree. For more information, see Using the Tag Finder.

MMC toolbars are not moveable or redockable.

Determining the Configuration Editor Version

The version of the Configuration Editor that you are using must be the same version as the AVEVA Historian that

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

you want to manage.

To determine the Configuration Editor version

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Click **Configuration Editor**.
3. On the **Help** menu, click **About AVEVA Historian Configuration Editor**. The **About AVEVA Historian Configuration Editor** dialog box appears, showing the current version.
4. Click **OK**.

Operations Control Management Console Menu Commands

The following commands appear on both the **Action** menu and on the shortcut menu that can be accessed by right-clicking an item in the console tree. The appearance of certain menu commands depends on which item is selected from the tree. Also, the same menu command may appear for multiple tree items.

This table does not describe standard menu commands such as **Copy** and **Delete**.

Command	Description
New Historian Group	Add a new server group to the console tree.
New Historian Registration	Register an AVEVA Historian for use with the console.
Edit Historian Registration Properties	Change the registration properties for the selected historian.
Start Historian/Stop Historian	Start or stop the historian.
Start Module/Stop Module	Start or stop individual modules that are a part of the historian.
Server Startup Options	Configure optional modules to start when the historian starts.
Reset Error Counts	Reset the number of errors and warnings back to 0.
View License Information	View licensing details for the historian.
Refresh License Information	Update the licensing information.
Reinitialize Topic/All Topics	Disconnect and then reconnect to one or more topics.
Load Messages	Change the language in which system messages appear. This command only appears if you have existing error logs from versions prior to the IndustrialSQL Server version 9.0 historian.
Track Modifications	Show a list of modifications to the system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Command	Description
Commit Pending Changes	Commit changes to the system configuration.
Import Tags	Import tag definitions from an InTouch application into the historian.
New IDAS	Add a new IDAS.
New I/O Server Type	Add a new I/O Server type.
New I/O Server	Add a new I/O Server.
New Topic	Add a new topic.
New Analog Tag	Add a new analog tag.
New Discrete Tag	Add a new discrete tag.
New String Tag	Adding a new string tag.
New Event Tag	Add a new event tag.
New Message	Add a new message.
New Engineering Unit	Add a new engineering unit.
New Tag	Add a new tag for the type you select in the console tree.
New Group	Add a new tag grouping in the Public Groups or Private Groups area of the console tree.
Add Tags to Group	Access the **Tag Finder** dialog box to search for tags to add to a tag group.
Filter	Apply a filter to the list of tags in the details pane.
Add Replication Schedule	Add a new replication schedule.
Create Replication Groups	Add new replication groups.
New Replication Server	Add a new replication server.
Add Single Tag	Add a single tag for replication.
Add Multiple Tag	Add multiple tags for replication.

Microsoft SQL Server Management Studio

As an administrator, you will probably spend the majority of your time interacting with Microsoft SQL Server

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

through the Microsoft SQL Server Management Studio, in which you can manage the Microsoft SQL Server areas of AVEVA Historian Server.

Using Microsoft SQL Server Management Studio, you can perform the following database functions (including the Runtime database):

- Register servers
- Manage backups
- Manage databases
- Manage devices
- Manage logins and permissions
- Manage replication
- Manage objects, such as tables, views, stored procedures, triggers, indexes, rules, defaults, and user-defined data types
- Schedule tasks
- Drag-and-drop objects from one server to another or within a server
- Generate SQL scripts

To start Microsoft SQL Server Management Studio, on the Start menu of the Windows Taskbar, point to the **Microsoft SQL Server** program group, and then click **SQL Server Management Studio**.

Registering a Server in Microsoft SQL Server Management Studio

The first time the Microsoft SQL Server Management Studio starts, the Local server node is added by default. To manage a remote server using Microsoft SQL Server Management Studio, you must first register it. When you register a server, you are giving the Microsoft SQL Server Management Studio a logical name and user account to log on to the Microsoft SQL Server database.

The instructions for registering a server vary depending on which version of Microsoft SQL Server you are using. See your Microsoft documentation for information on how to start the registration wizard.

Navigating in Microsoft SQL Server Management Studio

You must be connected to a server to manage it in Microsoft SQL Server Management Studio.

After you connect to the server, you can view the entire console tree associated with the server, including all of the devices and databases. You can expand folders just as you do in Windows Explorer.

About AVEVA Historian Client Web

AVEVA Historian Client Web is a browser client included as part of AVEVA Historian. It is the on-premises version of AVEVA Insight. This product gives you instant access to all of your organization's production and performance data any way you want it.

Historian Client Web provides an easy-to-use graphical interface for analyzing data, creating charts, and compiling dashboards of related information. Once you save your content, you can share it with other team members or reuse it in other documents.

Note: AVEVA Historian Client Web supports Google Chrome, Microsoft Internet Explorer 11, Microsoft Edge on

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Surface with Windows 10, Mozilla Firefox, and Safari on iPad with Retina display (in landscape mode).

Starting AVEVA Historian Insight

To open and use AVEVA Historian Client Web in Microsoft Edge or Chrome:

1. Open Microsoft Edge or Chrome, and then type the following URL:
   `http://<servername>:32569`
   where *servername* is the name of the AVEVA Historian server.
2. Select ☰ and then select **Help** for more information.

To open and use AVEVA Historian Client Web in Firefox

1. Open Firefox and, in the address bar, type:
   `about:config`
2. If prompted, agree to the caution statement from Firefox3.x or later.
3. After the configuration page loads, in the filter box type:
   `network.negotiate-auth`
4. Edit the value for "network.negotiate-auth.trusted-uris" by double-clicking the row and typing the following:
   `<servername>`
   Examples: "historiansvr01" or "localhost"

Starting and Stopping AVEVA Historian

About the Startup Process

When AVEVA Historian is started, these things happen:

- Start the associated Microsoft SQL Server database, if not already running.
- Verify start information stored in the SQL Server and the registry.
- Start each historian process.
- Create a new history block on disk to store data.
- Start communication with the data sources (IDASs).
- Begin storing data.

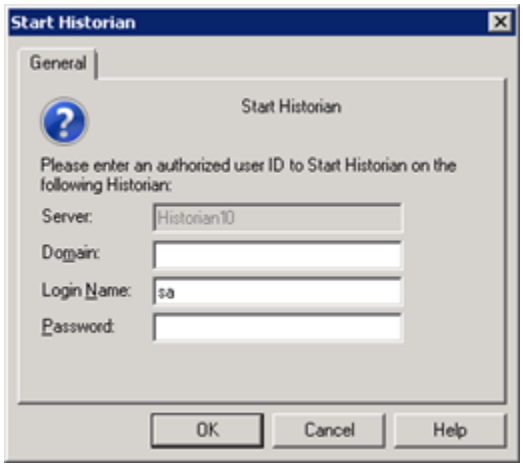
Starting the AVEVA Historian

To start the historian

1. Start the Operations Control Management Console. (From the Windows **Start** menu, point to **Programs**, **AVEVA**, and then click the **Operations Control Management Console** icon .)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. In the Operations Control Management Console tree, expand a server group and then expand a server.
3. Right-click **Management Console** and then click **Start Historian**. If the login credentials were not already saved, the standard login screen displays.



4. Enter the domain and the login name and password. The domain can be "." to identify the local computer.
5. Click **OK**.

Note: You cannot start the system if there is insufficient space in the circular storage location (less than 50 percent of the minimum threshold).

For the historian to start, TCP/IP must be enabled for the SQL Server. The following error message appears if TCP/IP is not enabled: "Fatal initialization error - unable to start. (Failed to open connection to configuration database)." By default, SQL Server Express and Developer Edition do not have TCP/IP enabled.

About Connecting to SQL Server

The Configuration Editor requires a valid Windows or SQL Server login account to connect to the AVEVA Historian. You can specify this account when you register a server. If you did not configure the account upon registration, or if you selected to display a logon prompt, you need to provide a login account as soon as you click the Configuration Editor item in the console tree.

For more information on SQL Server logins, see SQL Server Security.

When a connection is established, the Configuration Editor must evaluate the user permissions. If SQL Server authentication is used, the user is a member of the aaAdministrators or aaPowerUsers group, and full permissions are available. In all other cases, read-only permissions are applied.

Manually Starting SQL Server

The Operations Control Management Console, used to start the AVEVA Historian, also starts the Microsoft SQL Server.

Alternatively, you can manually start the Microsoft SQL Server with the SQL Server Configuration Manager. This application is loaded as part of the Microsoft SQL Server installation.

To start SQL Server with Configuration Manager

1. On the **Start** menu, point to the Microsoft SQL Server program group, then open the **Configuration Tools** folder, and click **SQL Server Configuration Manager**. The **SQL Server Configuration Manager** window

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

appears.

2. In the **Server** list, click the default server name for Microsoft SQL Server. The services appear in the details pane.
3. Right-click the service you want to start and select **Start** to start the service. The status of the service appears at the bottom of the window.

Ports used by AVEVA Historian

These ports may be used by AVEVA Historian:

Port	Description
135	**DCOM port** Used by remote OCMC.
135-139, 445	**TCP/UDP ports for classic remote IDAS** Used to communicate with the Remote IDAS 2014 R2 and earlier machine
1433	**Default SQL Server port** This port is configurable.
5413	**SuiteLink port** if the sending node is also receiving SuiteLink data, SuiteLink needs port 5413 to be open.
8080	**InTouch Access Anywhere server port** The host computer's firewall is configured to permit inbound and outbound network traffic on port 8080.
32565	**Default Historian TCP port** This port is configurable and is used for: • Data replication • Communication with remote IDAS version 2023 R2 and later.
32568	**Default Classic Historian TCP port** • This port was used for data replication and remote IDAS communication with Historian versions 2023 and earlier, and remains for compatibility with previous versions.
32569	**Default HTTP Historian Client Web/REST TCP port** This port is configurable and is used for:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• Interaction with OData interface • Data queries via InSight or the Historian REST API to the Historian Server • Event retrieval It is the IP port used if you are using HTTP and a gateway.
32573	**Default HTTPS Historian Client Web/REST TCP port** This port is configurable and is used for: • Interaction with OData interface • Data queries via InSight or the Historian REST API to the Historian Server • Event retrieval It is the IP port used if you are using HTTPS and a gateway.
55555	**License Server port**

Stopping the AVEVA Historian

When you stop the historian, it stops storing new data. However, it still functions as a data provider to clients. Services such as HCAP, retrieval, indexing, and the OLE DB provider are left running. To completely stop these services, shut down and disable the historian. For more information, see Shutting Down the Entire AVEVA Historian.

To stop the historian

1. In the Operations Control Management Console tree, expand a server group and then expand a server.

   Right-click **Management Console** and then click **Stop Historian**. If the login credentials were not already saved, the standard login screen appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Enter the domain (if necessary) and the login name and password. You can optionally select to not stop IDASs configured for store-and-forward.
3. Click **OK**.

Starting and Stopping Modules

Some of the components that make up the AVEVA Historian can be stopped and started individually without affecting data acquisition, storage, and retrieval. These modules include the event subsystem, Historian I/O server, and Historian I/O server (classic).

To start a module

1. In the console tree, expand a server group and then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Start Module**. The **Select Modules to Start** dialog box appears.

   The **Server** field shows the name of the AVEVA Historian server to which the options apply.
3. In the **Modules** window, click to select the optional modules to start or stop. (Only those modules that are currently stopped appear in the **Modules** window.)

   For more information about the Classic Event subsystem, see Classic Event Subsystem.

   For more information about the Historian I/O server and Historian I/O server (classic), see AVEVA Historian Processes.

   For more information about system modules related to the client access point, see Data Acquisition Components.

   To select all of the modules, click **Select All**. To cancel the selection of all of the modules, click **Select None**.
4. Click **OK**.

To stop a module

1. In the console tree, expand a server group and then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Stop Module**. The **Select Modules to Stop** dialog box appears.

   The **Server** field shows the name of the AVEVA Historian server to which the options apply.
3. In the **Modules** window, click to select the optional modules to start or stop. (Only those modules that are currently started appear in the **Modules** list.)

   For more information about the Classic Event subsystem, see Classic Event Subsystem.

   For more information about the Historian I/O server and Historian I/O server (classic), see AVEVA Historian Processes.

   For more information about system modules related to the client access point, see Data Acquisition Components.

   To select all of the modules, click **Select All**. To cancel the selection of all of the modules, click **Select None**.
4. Click **OK**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Closing the Operations Control Management Console

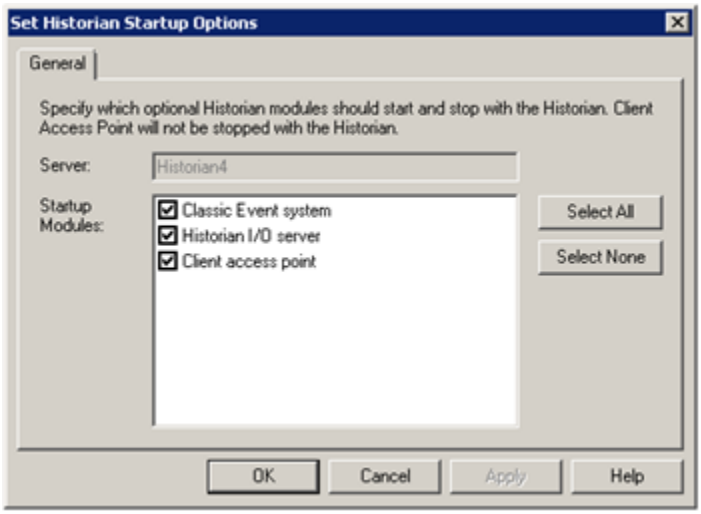
If you made any changes to the Operations Control Management Console, such as making server registration changes or adding tag groups, you are prompted to save those changes when you close the console.

Configuring General Startup Options

You can configure the AVEVA Historian to automatically start when the computer starts. In addition, you can configure the optional modules to automatically start when the main historian subsystems start. These modules can be stopped and started individually without affecting data acquisition, storage, and retrieval.

To configure general startup options

1. In the console tree, expand a server group and then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Server Startup Options**. The standard login screen appears.
3. Enter the domain, ID, and password in the dialog box and click **OK**. The **Set Historian Startup Options** dialog box appears.



   The **Server** box shows the node name of the computer hosting AVEVA Historian to which the options apply.
4. To automatically start one or more optional modules when the AVEVA Historian starts, click to select the modules in the **Startup Modules** window.

   To select all modules, click **Select All**. To cancel the selection of all modules, click **Select None**.

   For more information about the Classic Event subsystem, see Classic Event Subsystem.

   For more information about the AVEVA Historian I/O Server (aahIOSvrSVC), see AVEVA Historian Processes.

   For more information about system modules related to the client access point, see Data Acquisition Components.
5. Click **OK**.

Shutting Down the Entire AVEVA Historian

During a normal start and stop of the AVEVA Historian, the Configuration Manager service, the retrieval service,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

and the OLE DB provider are not shut down and they continue to run. A complete shutdown stops the entire system, including these services. Also, the Configuration Manager service is disabled so that it cannot be restarted.

Note: The Configuration Manager service is different than the Configuration Editor portion of the Operations Control Management Console management tool.

To shut down the entire system

1. In the console tree, expand a server group and then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Shutdown (and disable) Historian**.
3. Enter the domain, ID, and password in the dialog box and click **OK**.
4. You are prompted to verify the shutdown.
5. Click **OK**.

When the shutdown is complete, “Disconnected” appears for the system status.

WARNING! If the `shutdown -s -t 0` command is used to force a shutdown on the historian server or if the server is powered off by unplugging the power cord, data loss will occur at the Shutdown/PowerOff time point.

Item	Value
System time	7/28/2014 7:14:21 PM
Time of last start	7/28/2014 4:53:37 PM
Elapsed time since last start	2 hrs 20 mins
Time of last stop	7/28/2014 4:53:17 PM
Time of last reconfiguration	7/28/2014 6:21:05 PM
Configuration status	Normal
System status	Running
License status	Valid
Total number of tags in database	192

To start the system again, you first need to start the Configuration Manager service and then restart the historian.

To start the entire system

1. In the console tree, expand a server group and then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Enable (allow to run) Historian**. A confirmation dialog box appears.
3. Click **OK**.
4. Right-click **Management Console** and then click **Start Historian**. The standard login screen appears.
5. Enter the domain (if necessary) and the login name and password.
6. Click **OK**.

Creating Server Groups

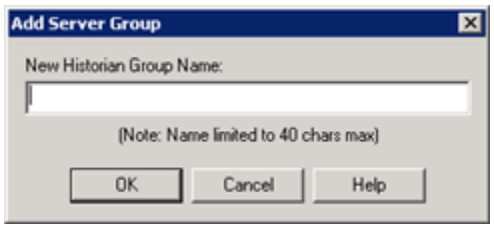
In the Operations Control Management Console, you can organize multiple AVEVA Historian instances into groups. A default group, called "Historian Group," is created for you. You can add servers to this existing group, delete or rename the group, and add other groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding a Server Group

To add a server group

1. In the Operations Control Management Console tree, right-click **Historian** and then click **New Historian Group**. The **Add Server Group** dialog box appears.



2. In the **New Historian Group Name** box, type the name of the group. The group name must contain no more than 40 characters.
3. Click **OK**.

Renaming a Server Group

To rename a server group

1. In the Operations Control Management Console tree, right-click the server group and then click **Rename**.
2. In the box that appears, type a new name for the server group. The group name must contain no more than 40 characters.

Deleting a Server Group

WARNING! When you delete a server group, you delete all server registrations within that group.

To delete a server group

- In the Operations Control Management Console tree, right-click the server group and then click **Delete**.

About System-Level Consistency

AVEVA Historian includes these features for system-level consistency:

- Time handling
- System parameters
- System message
- System processes
- System driver
- System tags

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time Handling

Timestamps for all data are stored in Coordinated Universal Time (UTC). The current UTC time is derived from the current local time and the time zone setting in the operating system of the computer on which the AVEVA Historian is running. During data retrieval, timestamps are returned in local time, by default. You can convert the timestamps so that they are shown in local time by using a special query parameter.

You should use the international date/time format for all timestamps used in queries. The format is:

YYYYMMDD HH:*MM*:*SS.000*

where:

- *YYYY* = year
- *MM* = month
- *DD* = day
- *HH* = hour
- *MM* = minutes
- *SS* = seconds
- *000* = milliseconds

The format for timestamps returned from queries is controlled by the default language settings of the SQL Server login. Make sure that you configure the default language setting for SQL Server logins correctly, especially in environments with mixed languages/regional settings.

If you have multiple historians and/or are using remote data sources, it is very important that you synchronize the time between the computers. For more information, see Time Synchronization for Data Acquisition.

Make sure that you have selected the operating system setting to automatically adjust time for daylight savings, if the time zone in which the computer is located observes daylight savings time.

System Parameters

A system parameter is a parameter that controls some aspect of the overall AVEVA Historian behavior. The following table describes the default system parameters:

Name	Description
AllowOriginals	Used to allow the insertion of original data for I/O Server tags. You must set this parameter to 1 before importing .lgh original data. For more information, see Importing, Inserting, or Updating History Data.
AnalogSummaryTypeAbbreviation	Abbreviation used when generating analog summary tagnames. For more information, see Specifying Naming Schemes for Replication.
AutoSummary	Used for auto-summarization of tag values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description
ConfigEditorVersion	(Not editable.) The minimum version number of the Configuration Editor that can edit the Runtime database. Used internally by the system.
CounterDeadband	Percentage (0-100) of the rollover value used to distinguish resets and reversals from rollovers.
DatabaseVersion	(Not editable.) Current version number of the Runtime database.
DataImportPath	Path to the CSV file for an import of external data. For more information, see Importing Data from CSV Files.
DBID	(Not editable.) Identifier for the Runtime database.
EnforceTLS	Indicates whether trusted communication is enforced. 1 = trusted communication is enforced, 0 = trusted communication is not enforced.
EventStorageDuration	Maximum duration, in hours, that event records are stored in the EventHistory table.
EventStorageLogPath	Path for storing logs for event tag data.
Future Time Threshold	Specifies a time threshold in minutes, after which streamed data values will be re-timestamped. If 0, then no future time threshold is applied.
GatewayHTTPSPort	Specifies the TCP port used for encrypted gateway communication.
GatewayTcpPort	Specifies the TCP port used for unencrypted gateway communication.
GroupedPrivateNamespace	Specifies whether domain user accounts can have private namespaces can be created for domain user accounts. 1 = Domain users are used; 0 = Domain users are not used.
HistorianPartner	The computer name of a partner historian. You can use either the host name, fully qualified name, or an IP address. Leading backslashes are optional. The name/IP used must be one that can be correctly resolved by all of the AppEngine and Historian Client computers that will connect to the partner. For more information, see Using a Redundant Historian.
HistorianVersion	(Not editable.) Current version number and build number of the AVEVA Historian. The value for this

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description
	parameter is automatically supplied when the system starts.
HistoryCacheSize	Allocation of system memory, in MB, of data collected by the Classic Storage subsystem from AVEVA Historian version 11.5 or earlier. The default is 0. For more information, see Memory Management for Retrieval of Classic Storage Data.
HistoryDaysAlwaysCached	The duration, in days, for which history block information is always loaded in memory. The default is 0. Applicable only for tag information of data collected by the Classic Storage subsystem from AVEVA Historian version 11.5 or earlier.
InterpolationTypeInteger	The type of interpolation for data values of type integer. 0=Stair-step; 1=Linear. The default is 0. For more information on interpolation, see Interpolation Type (wwInterpolationType) in the *AVEVA Historian Retrieval Guide*.
InterpolationTypeReal	The type of interpolation for data values of type real. 0=Stair-step; 1=Linear. The default is 1.
LicenseServer	Provides the name and port for the License Servers from which the license is acquired.
LicenseTagCount	(Not editable.) The number of licensed tags allocated to the system, in 500-tag increments.
MaxCyclicStorageTimeout	Controls the amount of time, in seconds, that AVEVA Historian waits for new values of cyclically stored tags before storing the previous one and making it available for retrieval. This parameter is used to balance three things: • Source latency -- the difference in the source-supplied timestamp and when the value is received by the historian • Data rate -- the rate at which delta values are reported by the source • Storage latency -- the amount of time that the Storage subsystem must wait before it ends a cycle and assigns a value to it Setting this parameter to a value other than 0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description
	ensures that the storage subsystem is not waiting indefinitely to confirm a final value for the cycle.
ModLogTrackingStatus	Turns modification tracking on or off. The value you specify will determine what modifications are tracked. For more information, see Turning Modification Tracking On/Off.
OldDataSynchronizationDelay	Time delay, in seconds, between when changes for "old" data (inserts, updates, and store-and-forward data) must be sent from the tier-1 historian to the tier-2 historian.
QualityRule	Indicates whether the system should use values having a quality of Good and Uncertain, or having only a quality of Good. 0 = Good and Uncertain; 1 = Good only. The default is 0. For more information on the quality rule, see Quality Rule (wwQualityRule) in the *AVEVA Historian Retrieval Guide*.
RealTimeWindow	The maximum delay, in seconds, for which data is considered real-time data applying the swinging door deadband. The delay is relative to the current time. Valid values are between 30 and 300 milliseconds. The default is 60. For more information, see About the Real-Time Data Window.
ReplicationConcurrentOperations	Limits the total number of retrieval client objects performing calculations in a retrieval based calculations for a time cycle.
ReplicationDefaultPrefix	The default prefix for replication tags on the tier-2 historian. If you change ReplicationDefaultPrefix system parameter, all replication tags that use the old prefix are not updated to use the newer prefix. For more information, see Specifying Naming Schemes for Replication.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description
ReplicationTcpPort	The TCP port number the Historian Client Access Port (HCAP) service listens on for any incoming-connection requests from an HCAL-enabled client. It must match the port number the HCAL-enabled client is using for communication with the historian. When modifying this system parameter on an historian node, you must also modify the port number in the Windows Firewall exception list for the historian replication service or another HCAL-enabled client to the same value. This port number must be unique on the historian node; that is, no other applications on the historian node should be listening on this port number.
SimpleReplicationNamingScheme	The default naming scheme used for configuring simple replication tags. For more information, see see Specifying Naming Schemes for Replication.
StateSummaryTypeAbbreviation	Abbreviation used when generating state summary tagnames. For more information, see see Specifying Naming Schemes for Replication.
SuiteLinkTimeSyncInterval	Frequency, in minutes, that IDASs will attempt to synchronize the timestamping mechanism for associated I/O Servers. If this parameter is set to 0, no time synchronization will occur. For more information, see Time Synchronization for Data Acquisition.
SummaryCalculationTimeout	The maximum expected delay, in minutes, for calculating summary data for replicated tags. Setting this parameter too high will delay associated summary calculations unnecessarily. Setting it too low will cause the system to prematurely calculate summaries and then later require additional processing to correct those calculations.
SummaryReplicationNamingScheme	The default naming scheme used for configuring summary replication tags. For more information, see see Specifying Naming Schemes for Replication .
SummaryStorageDuration	Maximum duration, in hours, that summary records will be stored in the legacy SummaryHistory table.
SysPerfTags	Used to turn on performance monitoring tags for the AVEVA Historian system. 0 = Off; 1 = On. The default is 1. For more information, see Performance Monitoring Tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Name	Description
TimeStampRule	Used to determine which timestamp within a retrieval cycle to use for a data value. 0 = Use the timestamp at the start of the cycle; 1 = Use the timestamp at the end of the cycle. The default is 1. For more information, see TimeStamp Rule (wwTimeStampRule) in the *AVEVA Historian Retrieval Guide*.
TimeSyncIODrivers	If enabled, the AVEVA Historian will send time synchronization commands to all associated remote IDASs. For more information, see Time Synchronization for Data Acquisition.

System Messages

System messages include error messages and informational messages about the state of the AVEVA Historian as a whole or for any of the internal subsystems and individual processes.

System messages are logged to these places:

- ArchestrA Logger
- Windows Event Log
  You can view this log with the Windows Event Viewer. Not all messages are logged to the Windows event log. In general, only user actions and exceptional events are written to this log. The messages are logged with the "Historian" or the name of the AVEVA Historian service as the source.

System messages are divided into the following categories:

Category	Description
FATAL	The process cannot continue. An error of this severity results in a system shutdown.
CRITICAL	These types of errors will cause malfunctions in the data storage or retrieval systems, such as data loss or corruption.
ERROR	General errors. For example, address validation errors during system startup. These errors may result in an orderly shutdown of the system, but will not preclude system operation in most cases.
WARNING	Messages that simply notify the operator of parameter settings or events that have occurred. For example, failure to link a dynamically-linked procedure entry point for a non-obligatory function will be logged as a warning.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Category	Description
INFO	Messages relating to startup progress or the commencement of active data storage.
DEBUG	Debugging messages, which will not typically appear in released versions of the system.

AVEVA Historian messages are logged to the Log Viewer as follows:

- Critical, fatal, and error messages are logged as "Error" messages. The appropriate indicator, either "CRITICAL," FATAL," or "ERROR," will be prefixed to message.
- Warnings will be logged as "Warning" message, with no prefix.
- Informational messages will be logged as "Info" messages, with no prefix.
- Debug messages will be logged as "Trace" messages, with no prefix.

For information on monitoring the system, see Monitoring the System.

AVEVA Historian Processes

The following table describes the AVEVA Historian processes:

Service/Process Name	Executable Name	Description
AVEVA Historian Client Access Point (aahClientAccessPoint)	aahClientAccessPoint.exe	Manages communication from HCAL clients to the historian . For more information, see Configuring Data Acquisition.
AVEVA Historian Configuration (InSQLConfiguration)	aahCfgSvc.exe	Handles all configuration requests, as well as hosts the interfaces for manual data input and retrieval. For more information, see About the Configuration Subsystem.
AVEVA Historian Event Storage Process	aahEventStorage.exe	Manages the storage of alarms and events to history blocks.
AVEVA Historian Classic Event System (InSQLEventSystem)	aahEventSvc.exe	Searches through history data and determines if specific events have occurred. For more information, see Classic Event Subsystem.
AVEVA Historian Indexing (InSQLIndexing)	aahIndexSvc.exe	Manages the internal indexing of history data that was stored by AVEVA Historian prior to 2014 R2.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Service/Process Name	Executable Name	Description
AVEVA Historian IOServer	aahIDataService.exe	Provides realtime data values from the historian to network clients using an iData-compliant protocol. For more information, see the *AVEVA Historian Retrieval Guide*.
AVEVA Historian IOServer (classic) (InSQLIOServer)	aahIOSvrSvc.exe	Provides realtime data values from the historian to network clients using the SuiteLink protocol. For more information, see the *AVEVA Historian Retrieval Guide*.
Data Import Subsystem (InSQLManualStorage)	aahManStSvc.exe	Accepts incoming data from CSV files and store-and-forward history blocks from Classic IDASs (shipped with AVEVA Historian prior to 2017). Stores the data in history blocks.
AVEVA Historian OData/REST web service	aahOWINHostLocal.exe	Retrieves alarm and event data from history block storage.
Classic Data Redirector Subsystem (InSQLStorage)	aahStoreSvc.exe	Accepts incoming real-time data from Classic IDASs and redirects it to AVEVA Historian storage.
AVEVA Historian Storage Process	aahStorage.exe	Storage process. Accepts incoming plant data and stores it to history blocks. Not a system service. For more information, see About Data Storage.
AVEVA Historian System Driver (InSQLSystemDriver)	aahDrvSvc.exe	Generates data values for various system monitoring tags. For more information, see About System Driver and System Tags.
AVEVA Historian Metadata Server	aahMetadataServer.exe	Maintains the tag metadata cache for tags stored by the storage process. Not a system service.
AVEVA Historian Search	HistorianSearch-x64.exe	Search process. Used to store and retrieve tags, saved content and keywords. **Note:** The search process has a fixed minimum overhead of 500 MB of memory usage. If a large

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Service/Process Name	Executable Name	Description
		number of tags are involved in a search, it is not unusual for the process to consume up to 1 GB of memory.

For more information on Windows services, see your Microsoft documentation.

About System Driver and System Tags

The system driver is an internal process that monitors key variables within an operating AVEVA Historian and outputs the values by means of a set of system tags. The system driver runs as a Windows service and starts automatically when the historian is started.

The system tags are automatically created when you install the historian. Also, additional system tags are created for each IDAS and replication server you configure.

The current value for an analog system tag is sent to the Storage subsystem according to a specified rate, in milliseconds. All date/time tags report the local time for the historian.

Legacy tags from upgraded systems may be retained.

Error Count Tags

The following analog tags have a storage rate of 1 minute (60000 ms). All error counts are since the AVEVA Historian is restarted or since or the last error count reset.

TagName	Description
SysCritErrCnt	Number of critical errors
SysErrErrCnt	Number of non-fatal errors
SysFatalErrCnt	Number of fatal errors
SysWarnErrCnt	Number of warnings

Date Tags

The following analog tags have a storage rate of 5 minutes (300000 ms).

TagName	Description
SysDateDay	Day of the month
SysDateMonth	Month of the year
SysDateYear	Four-digit year

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time Tags

All of the following tags are analog tags. Each value change is stored (delta storage).

TagName	Description
SysTimeHour	Hour of the day
SysTimeMin	Minute of the hour
SysTimeSec	Second of the minute

Storage Space Tags

The following analog tags have a storage rate of 5 minutes (300000 milliseconds). Space remaining is measured in MB.

TagName	Description
SysSpaceAlt	Space left in the alternate storage path
SysSpaceBuffer	Space left in the buffer storage path
SysSpaceMain	Space left in the circular storage path
SysSpacePerm	Space left in the permanent storage path

I/O Statistics Tags

The following analog tags can be used to monitor key I/O information.

TagName	Description
SysDataAcq*N*BadValues*	Number of data values with bad quality received. This tag has a storage rate of 5 seconds. The maximum is 1,000,000.
SysDataAcq*N*OutsideRealtime*	The number of values per second that were discarded because they arrived outside of the real-time data window. This tag has a storage rate of 5 seconds. The maximum is 1,000,000. This tag has been deprecated and will only be available in systems migrated from AVEVA Historian 2014 and earlier.
SysDataAcqOverallItemsPerSec	The number of items received from all data sources, including HCAP. This tag has a storage rate of 10 seconds. The maximum is 100,000.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagName	Description
SysDataAcqRxItemPerSecN*	Tag value update received per second. This tag has a storage rate of 10 seconds.
SysDataAcqRxTotalItemsN*	Total number of tag updates received since last startup for this IDAS. This tag has a storage rate of 10 seconds.
SysPerfDataAcqNBadValues*	Number of data values with bad quality received. This tag has a storage rate of 5 seconds. The maximum is 1,000,000.
SysStatusAverageEventCommitSize	Number of events written to the A2ALMDB database per minute.
SysStatusAverageEventCommitTime	Average time, in seconds, it takes to write events to the A2ALMDB database.
SysStatusEventCommitPending	Number of events that have not yet been written to the A2ALMDB database.
SysStatusRxEventsPerSec	Number of events received per second, calculated every 10 seconds.
SysStatusRxItemsPerSec	Tag value update received per second for the system driver. This tag has a storage rate of 10 seconds.
SysStatusRxTotalDuplicateEvents	Total number of duplicate events received through different channels since startup (and discarded as duplicates).
SysStatusRxTotalEvents	Total number of events received since startup.
SysStatusRxTotalItems	Total number of tag updates received since last startup for the system driver. This tag has a storage rate of 10 seconds.
SysStatusTopicsRxData	Total number of topics receiving data. Each active IDAS "topic" and each active HCAL connection are counted. Note that process and event history, even from the same source, count as separate connections.

*This status tag will exist for each defined IDAS. The identifying number (N) in the is the IODriverKey from the IODriver table. The number 0 designates MDAS and only applies to the SysDataAcq*N*BadValues and SysDataAcq*N*OutsideRealtime tags.

System Monitoring Tags

Unless otherwise noted, for the following discrete tags, 0 = Bad; 1 = Good.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag	Description
SysClassicManual Storage	Status of the data import service (aahManStSvc.exe).
SysClassicStorage	Status of the classic data redirector service (aahStoreSvc.exe).
SysClientAccessPoint	Status of the Client Access Point service (aahClientAccessPoint.exe).
SysConfiguration	Status of the configuration service (aahCfgSvc.exe). This parameter is set to 1 as long as a dynamic configuration is required or in progress.
SysDataAcqN*	Status of the IDAS service (aahIDASSvc.exe).
SysEventStorage	Status of the event storage service (aahEventStorage.exe).
SysEventSystem	Status of the classic event system service (aahEventSvc.exe).
SysIndexing	Status of the indexing service (aahIndexSvc.exe).
SysInSQLIOS	Status of the AVEVA Historian I/O Server (aahIOSvrSvc.exe).
SysMetadataServer	Status of the metadata server process (aahMetadataServer.exe)
SysOLEDB	Status of the OLE DB provider (loaded by SQL Server).
SysPulse	Discrete "pulse" tag that changes every minute.
SysReplication	Status of Replication service (aahReplSvc.exe).
SysRetrieval	Status of the retrieval service (aahRetSvc.exe).
SysStatusSFDataPending	Discrete tag indicating if one or more HCAL clients have store-and-forward data that needs to be sent to the historian. NULL = Unknown; 0 = No store-and-forward data; 1 =At least one HCAL client has data.
SysStorage	Status of the storage process (aahStorage.exe).
SysSystemDriver	Status of the system driver (aahDrvSvc.exe).
SysStatusMode	Analog tag indicating the operational state of the historian. If the value is NULL, the historian is stopped. 0 = Read-only mode. 1 = Read/write mode.

*This status tag will exist for each defined IDAS. The identifying number (N) appended to the end of the tag is the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IODriverKey from the IODriver table.

Miscellaneous (Other) Tags

The following table describes miscellaneous tags.

TagName	Description
SysConfigStatus	Number of database items affected by a dynamic configuration (that is, the number of entries in the ConfigStatusPending table when the commit is performed). This value is cumulative and not reset until the system is completely restarted.
SysHistoryCacheFaults	The number of history blocks loaded from disk per minute. The maximum value is 1,000. The storage rate for this analog tag is 60 seconds. For more information on the history cache, see Memory Management for Retrieval of Classic Storage Data.
SysHistoryCacheUsed	Number of bytes used for history block information. The maximum value is 3,000,000,000. The storage rate for this analog tag is 30 seconds.
SysHistoryClients	The number of clients that are connected to the Indexing service. The maximum value is 200. The storage rate for this analog tag is 30 seconds.
SysMinutesRun	Minutes since the last startup. The storage rate is 60 seconds for this analog tag.
SysRateDeadbandForcedValues	The total number of values that were forced to be stored as a result of using a swinging door storage deadband. This number reflects all forced values for all tags since the system was started.
SysString	String tag whose value changes every hour.
SysTagHoursQueried	A floating point value updated every minute that indicates the total number of "tag retrieval hours" queried by all client applications during that minute. For example, if a single trend queries four tags for a 15-minute period, that is "1.0 tag retrieval hours". All tags, including replication sync queue tags and non-existent tags, are counted. Unlicensed tags are not counted.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Classic Event Subsystem Tags

The following table describes the Classic Event subsystem tags.

TagName	Description
SysEventCritActionQSize	Size of the critical action queue.
SysEventDelayedActionQSize	Number of entries in the delayed action queue.
SysEventNormActionQSize	Size of the normal action queue.
SysEventSystem	A discrete tag that indicates the status of the event system service (aahEventSvc.exe). 0 = Bad; 1 = Good.
SysStatusEvent	Snapshot event tag whose value changes every hour.

Replication Subsystem Tags

The Replication Service collects the following custom performance counters about its own operation, where N is a primary key of the tier-2 historian in the Runtime database of the tier-1 historian. These values are stored cyclically every 10 seconds.

TagName	Description
SysReplicationSummaryCalcQueueItemsTotal	Current number of summary calculations stored in the summary calculation queue of all tier-2 historians.
SysReplicationSummaryClientsTotal	Current number of concurrent retrieval clients performing summary calculations on the tier-1 historian for all tier-2 historians.
SysReplicationSyncQueueItems*N*	Current number of items stored in the synchronization queue on the tier-2 historian of key *N*.
SysReplicationSyncQueueItemsTotal	Current number of items stored in the synchronization queue on the tier-1 for all tier-2 historians.
SysReplicationSyncQueueValuesPerSec*N*	Average synchronization queue values per second sent to the tier-2 historian of key *N*.
SysReplicationSyncQueueValuesPerSecTotal	Average values processed by the replication synchronization queue processor for all tier-2 historians.
SysReplicationTotalTags*N*	Total number of tags being replicated to the tier-2 historian of key *N*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagName	Description
SysReplicationTotalValues*N*	Total number of values sent to the tier-2 historian of key *N* since the startup of the replication service.
SysReplicationTotalValuesTotal	Total number of values sent to all tier-2 historians since the startup of the replication service.
SysReplicationValuesPerSec*N*	Average values per second sent to the tier-2 historian of key *N*.
SysReplicationValuesPerSecTotal	Average values per second sent to all tier-2 historians.

Performance Monitoring Tags

You use performance monitoring tags to monitor CPU loading and other performance parameters for various AVEVA Historian processes. (All of these values map to equivalent counters that are used in the Microsoft Performance Logs and Alerts application.)

The following tags allow you to monitor the percentage CPU load for all processors:

System Tag	Description
SysPerfAvailableBytes	Amount of free memory (RAM). If the amount of available memory is over 4,294,967,296, then the tag shows the remainder of the amount of memory divided by 4,294,967,296.
SysPerfAvailableMBytes	Amount of free memory (RAM). Use this tag to monitor systems that have a larger amount of memory. The value for this tag is the amount of available memory divided by 1 million.
SysPerfCPUMax	The highest CPU load of any single core, expressed as a percentage (0-100). For example, on a quad core system where the current loads for each core are 25%, 40%, 60% and 10%, this tag will be "60".
SysPerfCPUTotal	The overall processor load as a percentage of all cores (0-100).
SysPerfDiskTime	Percentage of elapsed time that the disk drive was busy servicing read or write requests.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

System Tag	Description
SysPerfMemoryPages	Rate at which pages are read from or written to disk to resolve hard page faults.

The remaining system tags are used to monitor performance for each historian service or process and for the Microsoft SQL Server service. For more information on services, see AVEVA Historian Processes.

There are six system performance tags per each service or process. These tags adhere to the following naming convention:

- SysPerf<*service*>CPU
- SysPerf<*service*>HandleCount
- SysPerf<*service*>PageFaults
- SysPerf<*service*>PrivateBytes
- SysPerf<*service*>PrivateMBytes
- SysPerf<*service*>ThreadCount
- SysPerf<*service*>VirtualBytes
- SysPerf<*service*>VirtualMBytes

where <*service*> can be any of the following:

- ClassicManualStorage
- ClassicStorage
- ClientAccessPoint
- Config
- DataAcq
- EventStorage
- EventSys
- Indexing
- InSQLIOS
- MetadataServer
- Replication
- Retrieval
- SQLServer
- Storage
- SysDrv

These tags have a cyclic storage rate of 5 seconds.

Note: The six performance tags will exist for each defined IDAS. The identifying number (N) appended to the end of the "DataAcq" portion of the tagname is the IODriverKey from the IODriver table. For example, 'SysPerfDataAcq1CPU'.

The following table describes the suffixes assigned to the names of system performance tags:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Suffix	Description
CPU	Current percentage load on the service, expressed as a percentage of total CPU load. For example, on a quad core system, if the service is using 20% of one core, 40% of another core, and 0% of the other two cores, this tag will be 15%.
HandleCount	Total number of handles currently open by each thread in the service. A handle is a identifier for a particular resource in the system, such as a registry key or file.
PageFaults	Rate, per second, at which page faults occur in the threads executing the service. A page fault will occur if a thread refers to a virtual memory page that is not in its working set in main memory. Thus, the page will not be fetched from disk if it is on the standby list (and already in main memory) or if it is being used by another process.
PrivateBytes	Current number of bytes allocated by the service that cannot be shared with any other processes. If the amount is over 4,294,967,296, then the tag shows the remainder of the amount divided by 4,294,967,296.
PrivateMBytes	Current number of Mbytes allocated by the service that cannot be shared with any other processes.
ThreadCount	Current number of active threads in the service. A thread executes instructions, which are the basic units of execution in a processor.
VirtualBytes	Current size, in bytes, of the virtual address space that is being used by the service. If the amount is over 4,294,967,296, then the tag shows the remainder of the amount divided by 4,294,967,296.
VirtualMBytes	Current size, in Mbytes, of the virtual address space that is being used by the service.

Important: You need to ensure that the memory that SQL Server reserves for the AVEVA Historian is adequate for the expected load. Based on your particular environment, you may need to adjust the SQL Server MemToLeave allocation. For more information on MemToLeave, see the Microsoft documentation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Defining Tags

About Tags

A tag is a variable in AVEVA Historian that represents a parameter or plant data point. For a tag, real-time or historical data is stored by the AVEVA Historian Storage subsystem, and then retrieved, or read back, by the Data Retrieval subsystem.

Each tag in the system is identified by a unique name. You can configure the following types of tags:

Analog	Event
Discrete	Analog summary
String	State summary

Note: Analog summary and state summary tags are discussed in Managing and Configuring Replication.

Configuration information for each type of tag is stored in the historian, as well as the history for tags over time. Event tags do not store values, but rather definitions for events to be detected by the system and the subsequent actions to be triggered.

Using the Configuration Editor, you can view or edit information for existing tag definitions, create definitions for new tags, or delete existing tags.

Note: If you already have tags defined for an InTouch application, you can import the definitions using the Tag Importer. For more information, see Importing an InTouch Data Dictionary.

Tag Naming Conventions

Tag names may contain :

- Letters
  Any letter as defined by the Unicode Standard. The Unicode definition of letters includes Latin characters from a through z and from A through Z, in addition to letter characters from other languages.
- Digits
  Any numerical character.
- Special characters
  Any graphics character.

  These special characters may NOT be used in tag names:
  - Characters whose ASCII table code is 0 through 32 -(non-graphic characters)
  - + - * / \ = ( ) ` ~ ! ^ & @ [ ] { } | : ; ’ , < > ? " space

It is highly recommended that you adhere to the rules for SQL Server identifiers as well.

For "conventional" tag names, the first character may be a:

- Letter
- Digit
- Dollar sign ($) or pound sign (#)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Subsequent characters may be:

- A digit, but then the tagname must contain at least one letter
- Any of the supported special characters.

Due to storage formatting requirements, you cannot use either a quotation mark (") or one or more single quotation marks (') at the beginning or at the end of a tag name.

Tag names that do not comply with these rules are regarded as "unconventional."

In a SQL query against a wide table, unconventional tag names must be delimited with brackets ([ ]), because the tag name is used as a column name.

Tag Properties (Tag Metadata)

Every AVEVA Historian tag is associated with one or more tag metadata instances. A tag metadata instance is a set of properties identified by a unique TagId. The TagId is a 16-byte globally unique identifier (GUID). Tag metadata properties include tag name, description, tag type, storage type, creation time, and so on.

Tag metadata properties describe what the tag is, where the data for that tag is coming from, how the timestamped data values(VTQs) of that tag should be stored, and how they should be retrieved and displayed by client applications. The following table describes the most important tag metadata properties:

Property	Description
TagName	A Unicode string (UTF-16) of up to 256 characters.
TagId	A 16-byte GUID.
DateCreated	The UTC timestamp when the tag metadata instance was created.
CreatedBy	The name of the user or application created the tag metadata instance.
TagType	The type of tag: analog, discrete, string, event, or summary. For more information, see Types of Tags in the *AVEVA Historian Concepts Guide*.
AcquisitionType	The method by which the tag's values are acquired.
StorageType	The method of storing the tag's values.

Depending on values of these properties, additional properties provide more specific details. For example if a tag is an analog tag—that is, it represents a variable measuring a continuous physical quantity such as the temperature of a boiler—the tag metadata property RawType specifies what kind of numeric type is used, either float or integer. If it is an integer, the IntegerSize property specifies the number of bits in that integer, and so on.

Note: The ability to set up an alternate file storage location for tag metadata is possible to ensure it is available should the primary location become corrupt or not accessible.

For the full list of tag metadata properties, see the tag-related tables in the Tables chapter in the *AVEVA Historian*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Database Reference.

Tag Configuration Versioning

Each time there is a change to a tag property -- description, engineering unit, minimum/maximum range, etc. -- a new tag metadata instance may be created with a unique TagId. This allows AVEVA Historian to preserve the tag configuration history when different types of data is stored for the same tag over time.

Note: Because each new tag metadata instance consumes system resources, it's best to limit these types of updates. AVEVA Historian does not impose a strict limit on the number of metadata instances per tag, but recommends keeping that number under 200 to avoid performance degradation and instability.

Several different tag metadata instances can share the same tag name, resulting in several versions of the same tag. The most recently created tag metadata instance is known as the current version of that tag.

Here is an example:

You create a tag named MyTag. It stores 16-bit integer values from some device. A new tag metadata instance with a TagId -- 883DDAE3-E3F5-441C-A5FD-38AD97DEC070 -- gets created and then some data values get stored.

Several months later, the device is upgraded to generate 32-bit integer values. So, you reconfigure MyTag to store 32-bit integers. During that tag reconfiguration, a new tag metadata instance with another TagId -- FFA0E74C-12FD-49A6-8EBA-B30AFAEF55DA -- is created. When new 32-bit values are stored, they are associated with that new TagId.

Now although the current version of the MyTag references it as a 32-bit integer, the older 16-bit values are still accessible because they are associated with the older tag metadata instance preserved in the history.

About Floating-Point Values

Like most software, AVEVA Historian uses floating-point arithmetic. Single-precision floating-point values in AVEVA Historian are generally accurate to six decimal places and double-precision values are accurate to 15 decimal places.

Sometimes casting a single-precision floating point value as double-precision, or the reverse, means that the revision does not match the original value.

Internally, AVEVA Historian uses double precision for all calculations. This can lead to slight differences in the way results are displayed.

Standard computer representations (using IEEE 754) are stored as binary and don’t precisely match human-readable representations. Because of this, rounding for a value may not be obvious to users. For example, "230.4" may be rounded to a binary number that displays as "230.39999389648437".

For more details, refer to this Wikipedia article on the IEEE 754 standard for floating-point numbers in computers: https://en.wikipedia.org/wiki/IEEE_754-1985.

Viewing and Configuring Tags

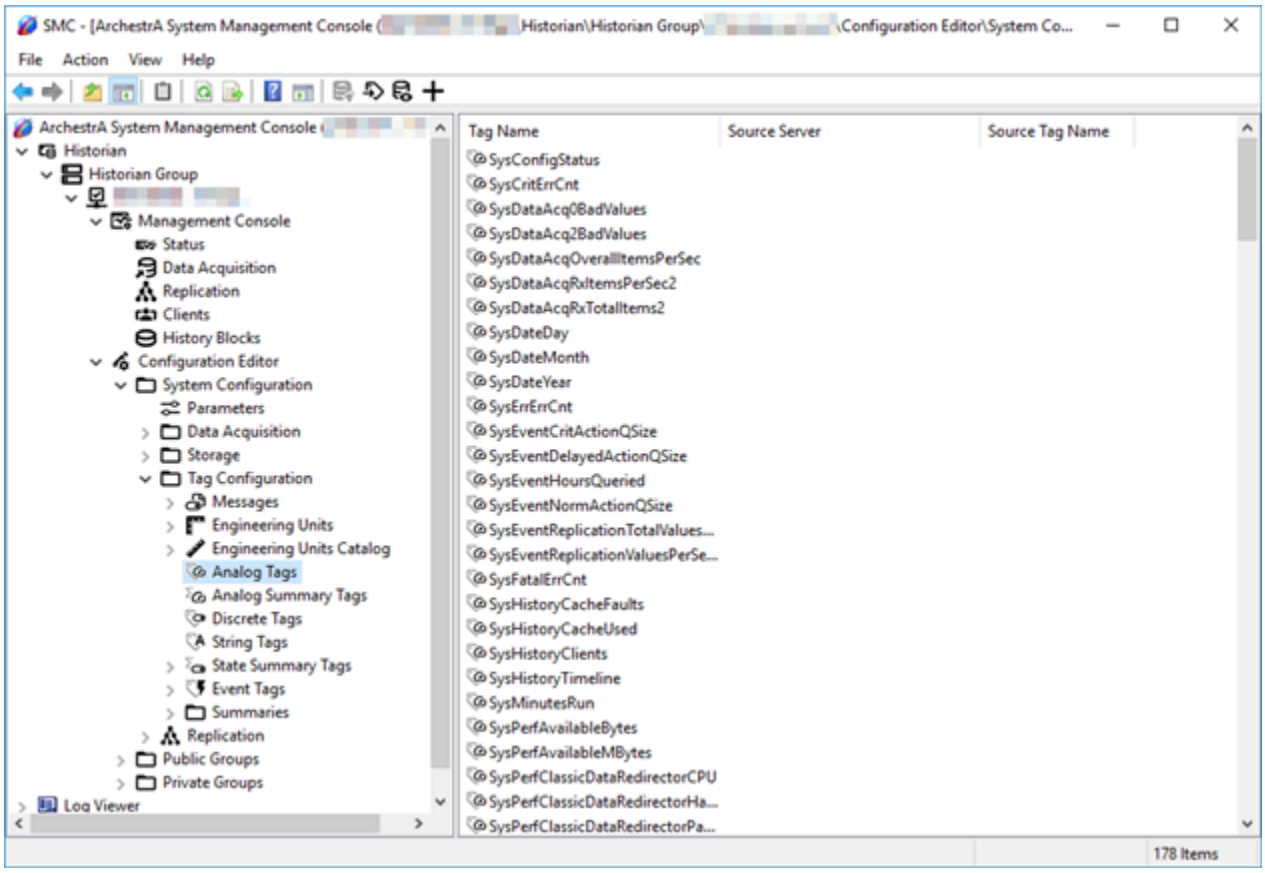
You can view, add, and configure tags using the Operations Control Management Console.

To view tag information

1. Open the Operations Control Management Console.
2. In the console tree area, expand a server group and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.



4. Select one of these tag categories to view the tags in that category:

• Analog Tags	• Discrete Tags	• State Summary Tags
• Analog Summary Tags	• String Tags	• Event Tags

To see details about a particular tag

- In the right pane, double-click the tag name.

Configuring Analog Tags

You can configure general information, acquisition details, storage details, limit information, and summary setup information for a selected analog tag, as well as add new analog tags to the system.

Adding an Analog Tag

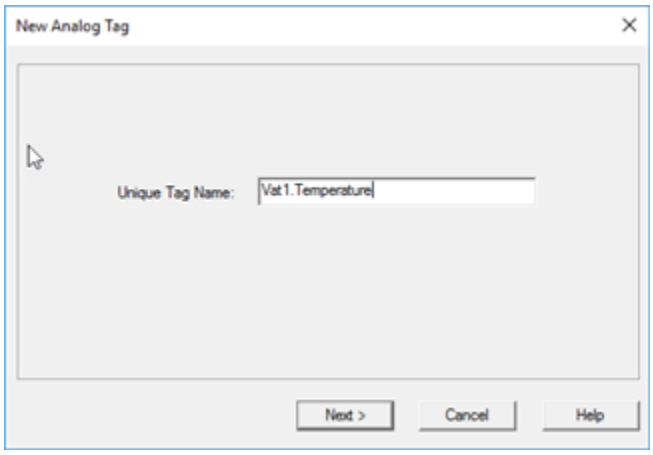
Be sure that you do not exceed your licensed tag count by adding another tag.

To add an analog tag

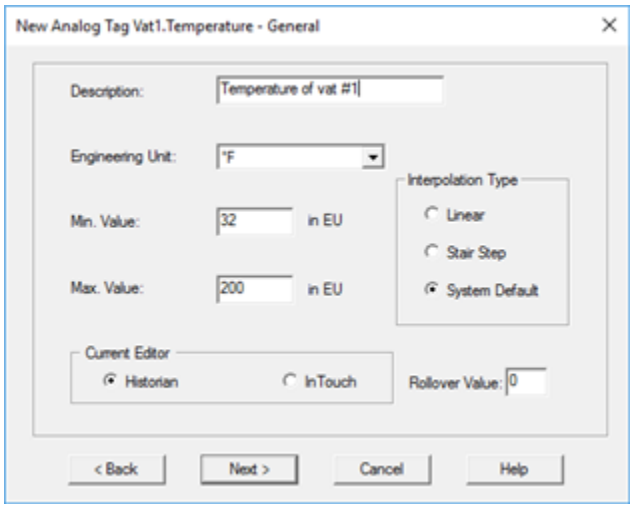
1. In the Operations Control Management Console, expand a server group and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Analog Tags**, and then click **New Tag**. The **New Analog Tag** wizard displays.



4. Enter a unique name for the analog tag. For information on allowable tag names, see Tag Naming Conventions.
5. Click **Next**. The General information dialog of the wizard displays.



6. In the **Description** box, type a description of the tag.
7. In the **Engineering Unit** list, select the unit of measure. Examples are mph, grams, and pounds.

   For information on adding an engineering unit to the system, see Configuring Engineering Units.
8. In the **Min Value** box, type the minimum value of the tag, measured in engineering units.
9. In the **Max Value** box, type the maximum value of the tag, measured in engineering units.

   In the **Current Editor** group, specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a reimport is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for reimporting; however, all changes made to the tag using the Configuration Editor are lost during the reimport. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

changes are not preserved.

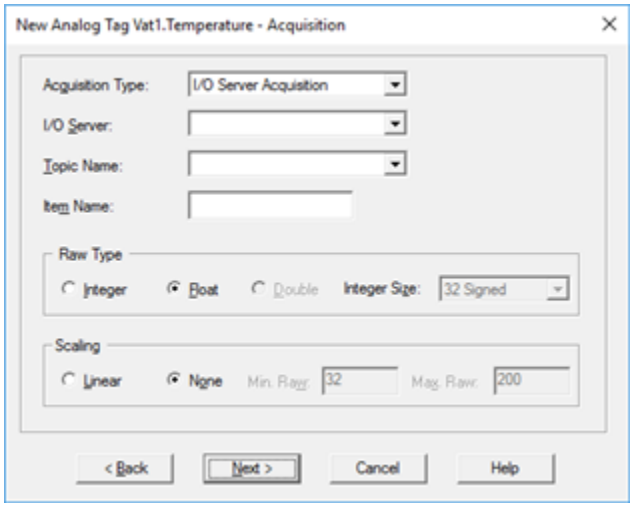
10. In the **Interpolation Type** group, type the analog value to use as the last point of the retrieval cycle.

    For more information, see Interpolated Retrieval in the AVEVA Historian *Retrieval Guide*.

    - **Linear**
      The system will calculate a new value at the given cycle time using linear interpolation.
    - **Stair Step**
      The last known point is returned with the given cycle time.
    - **System Default**
      The settings of both the InterpolationTypeReal and InterpolationTypeInteger system parameters are used.

      In the **Rollover Value** box, type the rollover value if this tag is a counter-type tag. (A typical example is for a flowmeter measuring flow or an integer counter, such as those used on a packing line.) The rollover value is the first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.

11. For more information, see Counter Retrieval in the AVEVA Historian *Retrieval Guide*.
12. Click **Next**. The Acquisition information dialog of the wizard displays.



13. In the **Acquisition Type** list, select the method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item.
14. In the **I/O Server** list, select the application name of the I/O Server. This name is usually the same as the executable file name. The list includes all I/O Servers defined in the system.
15. In the **Topic Name** list, select the name of the topic. The list includes all topics defined for the selected I/O Server.
16. In the **Item Name** box, type the address string of the tag.
17. If you are editing a discrete or string tag, click **OK**. Otherwise, continue with the next step.
18. In the **Raw Type** group, select the numeric type that matches the raw value as it is acquired.
    - **Integer**
      Integer value. If you select this option, a list appears in which you can select the integer size, in bits, and whether it is signed or unsigned.
    - **Float**

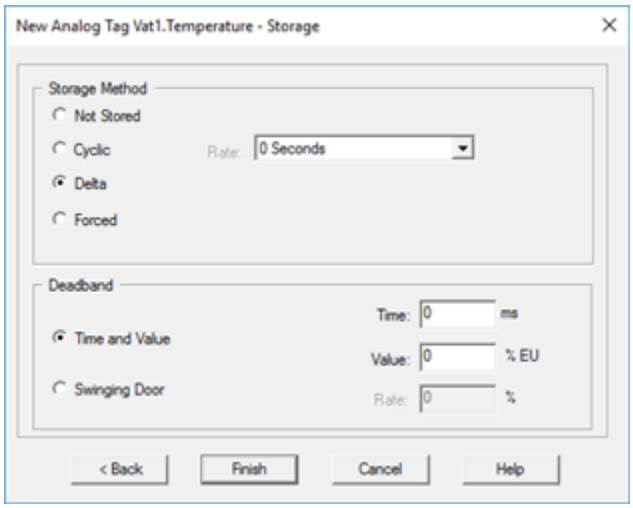
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IEEE single-precision floating (decimal) point value, which supports approximately 7 decimal places. All floating point calculations are performed with 64-bit resolution, but the result is stored as a 32-bit number. Note that IDAS/SuiteLink can only send single-precision values.

- **Double**
  IEEE double-precision floating (decimal) point value, which supports approximately 13 decimal places. The data is stored with 64-bit resolution. Note that if the source can only send single-precision values, storing as a double with a higher resolution consumes space with no added benefit.

Note: For Float and Double types, some values may vary slightly from those shown in the source. See About Floating-Point Values for more information.

19. In the **Scaling** group, select the type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. The following options are required for linear scaling.
    - **Min Raw**
      The minimum value of the raw acquired value.
    - **Max Raw**
      The maximum value of the raw acquired value.
20. Click **Next**. The Storage information dialog of the wizard displays.



21. In the **Storage Method** area, select the way in which values for the tag will be stored.
    - **Rate**
      The rate at which the tag is stored if the storage type is cyclic.
22. In the **Deadband** area, configure details for how the tag value is stored. The availability of options in this group depends on which storage method you selected.
    - **Time and Value**
      A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. In the **Time** box, type the time to use for this deadband.

      A value deadband is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. In the **Value** box, type the value to use for this deadband.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

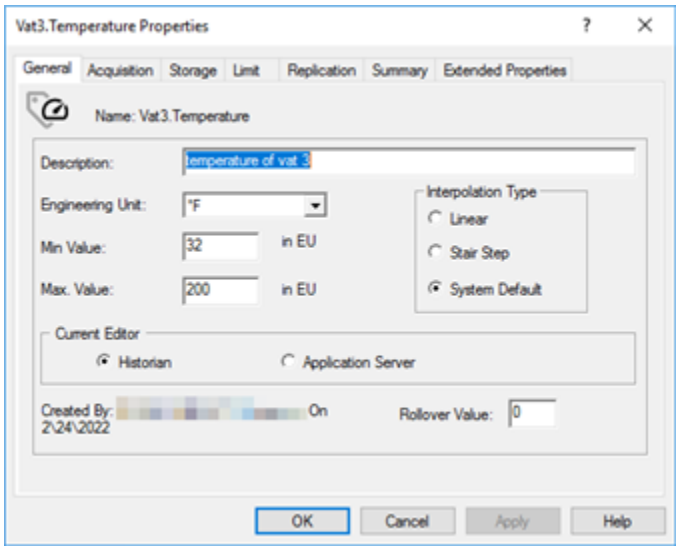
- **Swinging Door**
  A swinging door deadband is the percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. In the **Rate** box, type the rate to use for this deadband.

23. When you are done defining the new analog tag, click **Finish**.

Editing General Information for an Analog Tag

To edit general information for an analog tag

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Analog Tags**.
4. In the details pane, double-click the analog tag you want to edit. The **Properties** dialog displays.



5. Select the **General** tab.
6. In the **Description** box, type a description of the tag.
7. From the **Engineering Unit** list, select the unit of measure. Examples are mph, grams, and pounds.

   For information on adding an engineering unit to the system, see Configuring Engineering Units.

   Note: If the tag is initially configured using AVEVA Application Server, changes made to the engineering unit are not preserved. Instead, you must change the engineering unit at the source, and then redeploy the engine.

8. In the **Min Value** box, type the minimum value of the tag, measured in engineering units.
9. In the **Max Value** box, type the maximum value of the tag, measured in engineering units.

   In the **Current Editor** group, specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a reimport is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for reimporting; however, all changes made to the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tag using the Configuration Editor are lost during the reimport. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved.

10. In the **Interpolation Type** group, type the analog value to use as the last point of the retrieval cycle.

    For more information, see Interpolated Retrieval in the *AVEVA Historian Retrieval Guide*.

    - **Linear**
      The system will calculate a new value at the given cycle time using linear interpolation.
    - **Stair Step**
      The last known point is returned with the given cycle time.
    - **System Default**
      The settings of both the InterpolationTypeReal and InterpolationTypeInteger system parameters are used.

      In the **Rollover Value** box, type the rollover value if this tag is a counter-type tag. (A typical example is for a flowmeter measuring flow or an integer counter, such as those used on a packing line.) The rollover value is the first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.
11. For more information, see Counter Retrieval in the *AVEVA Historian Retrieval Guide*.
12. Click **OK**.

Editing Acquisition Information for a Tag

The **Acquisition** tab contains basically the same configuration information for analog, discrete, and string tags. However, the tabs for string and discrete tags do not include the **Raw Type** group or **Scaling** group.

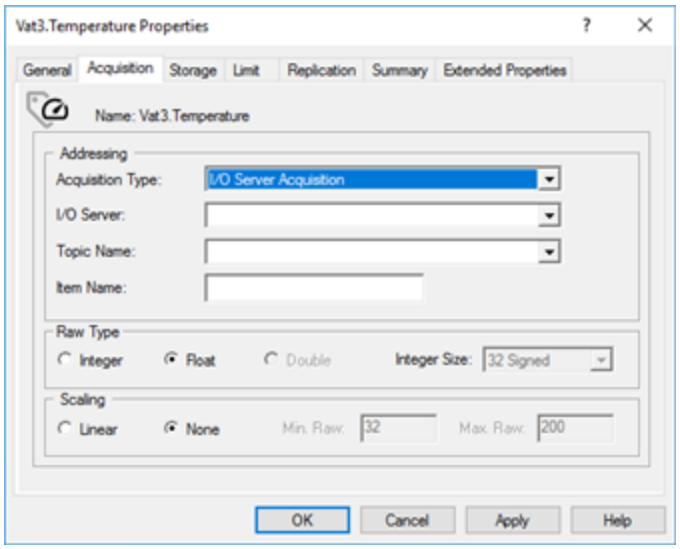
For information on data acquisition, see Configuring Data Acquisition.

If you change the configuration, then the changes are applied only to data with timestamps that are equal to or greater than the timestamp of the configuration change.

To edit acquisition information for a tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select the type of tag for which you want to edit the acquisition properties.
4. In the details pane, double-click the tag to edit. The **Properties** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select the **Acquisition** tab.
6. In the **Acquisition Type** list, select the method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item.
7. In the **I/O Server** list, select the application name of the I/O Server. This name is usually the same as the executable file name. The list includes all I/O Servers defined in the system.
8. In the **Topic Name** list, select the name of the topic. The list includes all topics defined for the selected I/O Server.
9. In the **Item Name** box, type the address string of the tag.
10. If you are editing a discrete or string tag, click **OK**. Otherwise, continue with the next step.
11. In the **Raw Type** group, select the numeric type that matches the raw value as it is acquired.
    - **Integer**
      Integer value. If you select this option, a list appears in which you can select the integer size, in bits, and whether it is signed or unsigned.
    - **Float**
      IEEE single-precision floating (decimal) point value, which supports approximately 7 decimal places. All floating point calculations are performed with 64-bit resolution, but the result is stored as a 32-bit number. Note that IDAS/SuiteLink can only send single-precision values.
    - **Double**
      IEEE double-precision floating (decimal) point value, which supports approximately 13 decimal places. The data is stored with 64-bit resolution. Note that if the source can only send single-precision values, storing as a double with a higher resolution consumes space with no added benefit.

    Note: For Float and Double types, some values may vary slightly from those shown in the source. See About Floating-Point Values for more information.
12. In the **Scaling** group, select the type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. The following options are required for linear scaling.
    - **Min Raw**
      The minimum value of the raw acquired value.
    - **Max Raw**
      The maximum value of the raw acquired value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

13. Click **OK**.

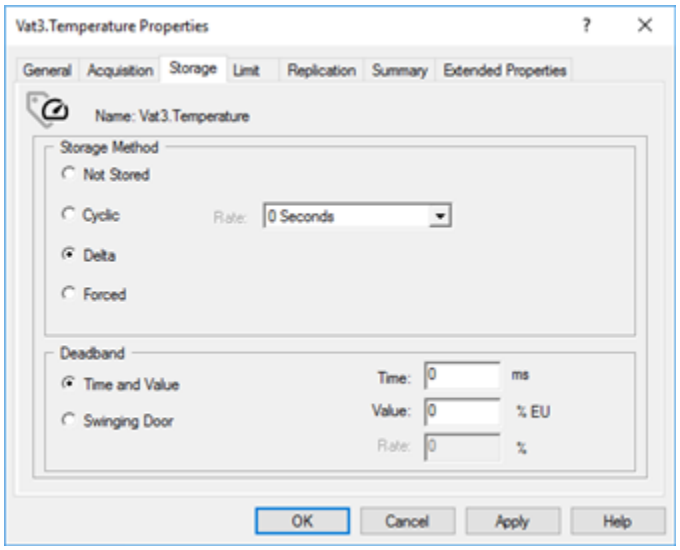
Editing Storage Information for an Analog Tag

For more information on storage, see Managing Data Storage.

If you change the configuration, then the changes are applied only to data with timestamps that are equal to or greater than the timestamp of the configuration change.

To edit storage information for an analog tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Analog Tags**.
4. In the details pane, double-click the analog tag to edit. The **Properties** dialog displays.



5. Select the **Storage** tab.
6. In the **Storage Method** area, select the way in which values for the tag will be stored.
   - **Rate**
     The rate at which the tag is stored if the storage type is cyclic.
7. In the **Deadband** area, configure details for how the tag value is stored. The availability of options in this group depends on which storage method you selected.
   - **Time and Value**
     A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. In the **Time** box, type the time to use for this deadband.

     A value deadband is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. In the **Value** box, type the value to use for this deadband.
   - **Swinging Door**
     A swinging door deadband is the percentage of deviation in the full-scale value range for an analog tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

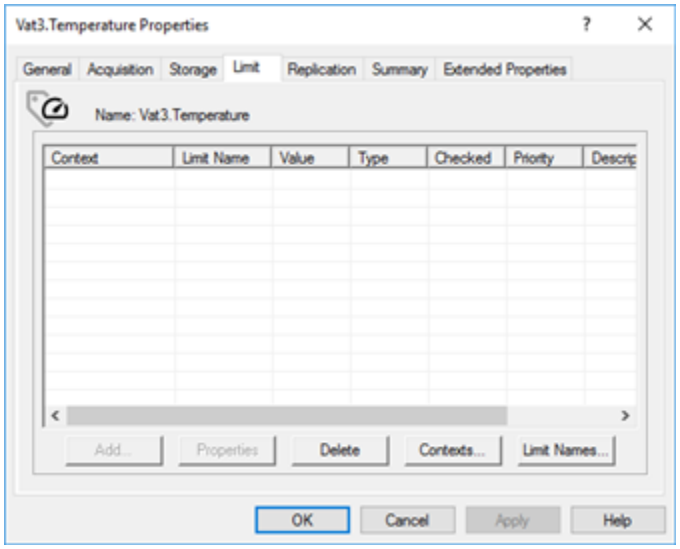
The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. In the **Rate** box, type the rate to use for this deadband.

8. Click **OK**.

Editing Limit Information for an Analog Tag

To edit limit information for an analog tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Analog Tags**.
4. In the details pane, double-click the analog tag to edit. The **Properties** dialog displays.



5. Select the **Limit** tab. The following information about limits for the tag appears:
   - **Context**
     The description of the context.
   - **Limit Name**
     The name for the limit.
   - **Value**
     The value that is used as a specific limit for a tag. In theory, a tag can have an infinite number of limits defined.
   - **Type**
     The type of limit; that is, whether it is a rising (up) or falling (down) limit.
   - **Checked**
     Used to specify whether a tag imported from InTouch is configured for automatic limit checking. Only checked limits are imported.
   - **Priority**
     The priority for the limit. Priorities can range from 1 to over 2 billion, with 1 being the highest priority.
   - **Description**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The description of the limit.

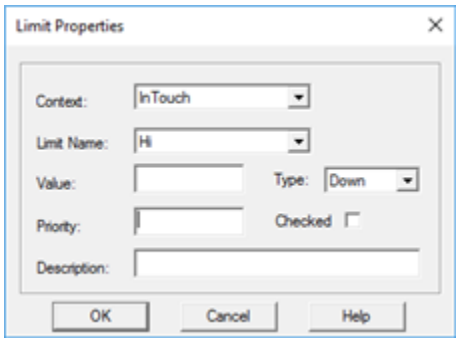
6. To add a limit, click **Add**. The **Limit Properties** dialog box appears. For more information, see Configuring Limits
7. To view properties for a limit, click **Properties**. The **Limit Properties** dialog box appears. For more information, see Configuring Limits.
8. To delete a limit, select the limit in the window and then click **Delete**.
9. To view or add context definitions, click **Contexts**. For more information, see Configuring Context Definitions.
10. To view or add a limit names, click **Limit Names**. For more information, see Configuring Limit Names.
11. Click **OK**.

Configuring Limits

Before you add a new limit, you must first add a limit name and a context. For more information, see Configuring Limit Names and Configuring Context Definitions.

To add a limit or view properties for a limit

1. On the **Limit** tab of the analog tag **Properties** dialog box, click **Add** or **Properties**. The **Limit Properties** dialog box appears.



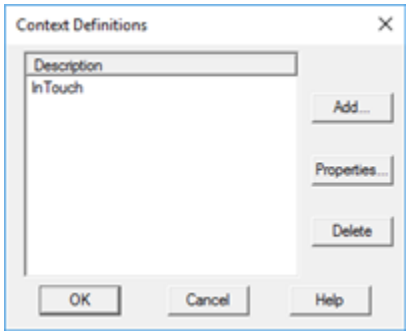
2. In the **Context** list, select the description of the context.
3. In the **Limit Name** list, select the name for the limit.
4. In the **Value** box, type the value that is used as a specific limit for a tag. In theory, a tag can have an infinite number of limits defined.
5. In the Type list, select the type of limit; that is, whether it is a rising (up) or falling (down) limit.
6. In the **Priority** box, type the priority for the limit. Priorities can range from 1 to over 2 billion, with 1 being the highest priority.
7. Select the **Checked** check box to enable automatic limit checking.
8. In the **Description** box, type the description of the limit.
9. Click **OK**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring Context Definitions

To add or view context definitions

1. On the **Limit** tab of the analog tag **Properties** dialog box, click **Contexts**. The **Context Definitions** dialog box appears.

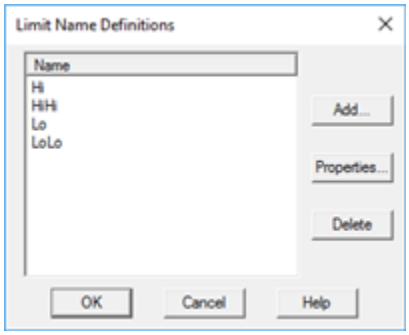


   All defined contexts are listed in the window.
2. To add a context, click **Add** and then type the name of the new context in the dialog box that appears. Click **OK**.
3. To change the context name, select a context in the window, and then click **Properties**. Type the new name in the dialog box that appears. Click **OK**.
4. To delete a context, select the context in the window and then click **Delete**.
5. Click **OK**.

Configuring Limit Names

To add or view limit names

1. On the **Limit** tab of the analog tag **Properties** dialog box, click **Limit Names**. The **Limit Name Definitions** dialog box appears.



   All defined limits are listed.
2. To add a limit name, click **Add** and then type the name of the new limit. Click **OK**.
3. To change a limit name, select a limit, click **Properties,** and then type in a new name. Click **OK**.
4. To delete a limit, select a limit and then click **Delete**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

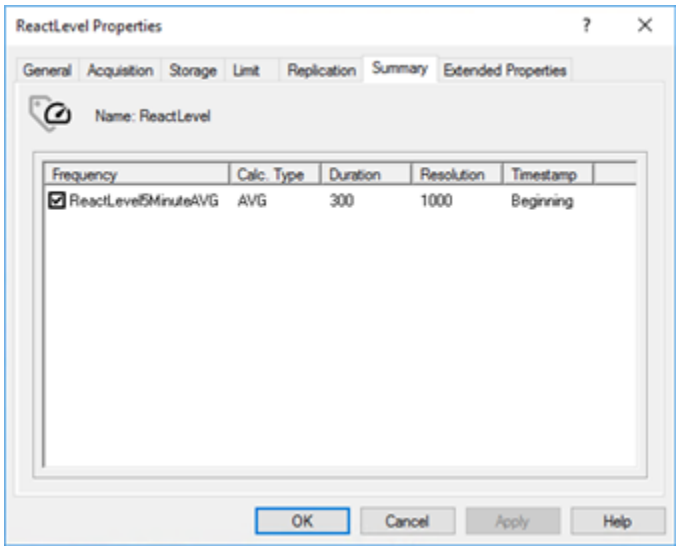
5. Click **OK**.

Editing Summary Information for an Analog Tag

Summaries are aggregation operations that can be set up to be automatically performed for analog tags.

To edit summary information for a tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Analog Tags**.
4. In the details pane, double-click the analog tag you want to edit. The **Properties** dialog displays.



5. Select the **Summary** tab.

   A check mark appears in the **Frequency** column of the summary operation in which the selected analog tag is included.
6. To remove the selected analog tag from an operation, clear the checkbox in the **Frequency** column.
7. To add the analog tag to any summary operation, select the check box in the **Frequency** column for the desired operation.
8. Click **OK**.

Editing Extended Properties for an Analog Tag

Extended properties can be used to add application-specific metadata to tags. When you search for tags in AVEVA Historian Client Web or AVEVA Insight, you can locate tags based on these property values. Likewise, with SQL queries you can select tags based on these property values without needing to know the tag name.

The following extended properties are predefined by the system, and can be added to any analog tags:

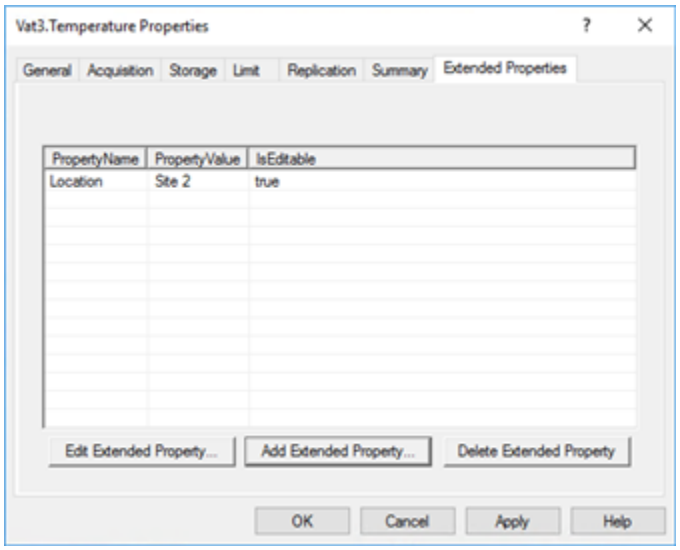
- **Alias** - Used as the tag's preferred label in charts.
- **Dimension** - Groups analog tags together by related engineering units.
- **Location** - Places the tag within the asset model of AVEVA Insight.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

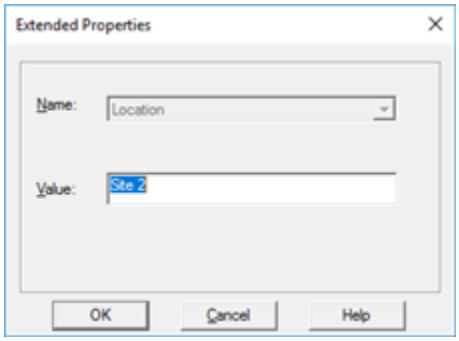
You can define your own custom extended properties by using the database import/export feature. See Adding New Tag Extended Property Values for more information.

To edit extended property information for an analog tag:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Analog Tags**.
4. In the details pane, double-click the analog tag you want to edit. The **Properties** dialog displays.

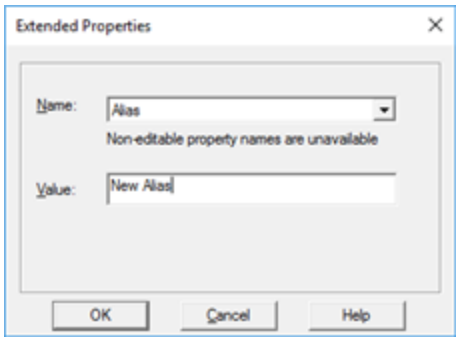


5. Select the **Extended Properties** tab.
6. To edit an existing property value, select the property in the list and click **Edit Extended Property...**. The **Extended Properties** dialog displays.



7. Enter a new **Value** for the property, then click **OK**.
8. To add a property to a tag, click **Add Extended Property...**. The **Extended Properties** dialog displays.

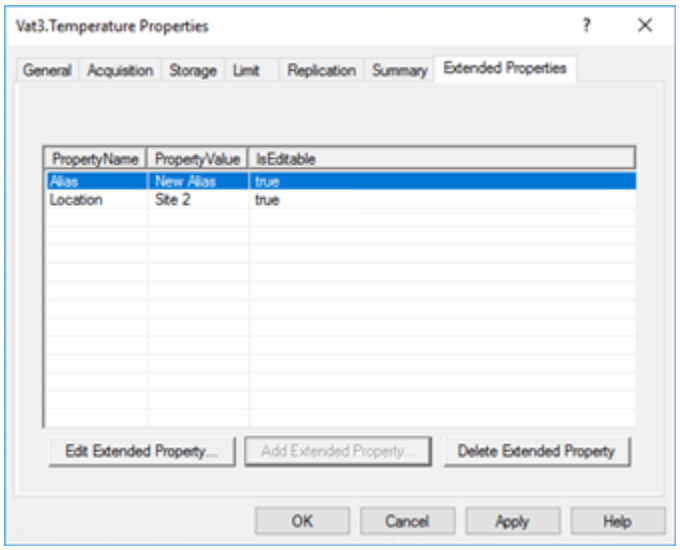
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



9. Select the **Name** of the extended property from the drop-down list. The list contains all editable extended properties that are not already present on the tag.

   Note: You can add new extended properties by exporting a configuration text file, adding new TagExtendedPropertyName entries, and importing the modified file. For more information, see Importing or Exporting Tag Information and Editing the Configuration Text File.

10. Enter a **Value** for the property, then click **OK**. The new property is added to the list.



11. To remove a property from a tag, select the property and click **Delete Extended Property**. The property is removed from the list.
12. Click **OK** or **Apply** to save your changes.

Configuring the Engineering Units Catalog

Engineering units can be application-specific, so AVEVA Historian must know how to relate these units to canonical definitions of engineering units. The Engineering Units Catalog is used to define the relationships between different units of measure, enabling support for converting between related units of measure.

Using the Engineering Units Catalog, you can define dimensions, unit systems, and how to convert between units within the same dimension. Defining these entries and correctly specifying unit conversions can take some care, so AVEVA Historian has preconfigured the catalog with the entries we expect most applications will need.

You can hide units for unit systems and dimensions that aren't used at your site, making it easier to configure the association between engineering units and their canonical units in the catalog.

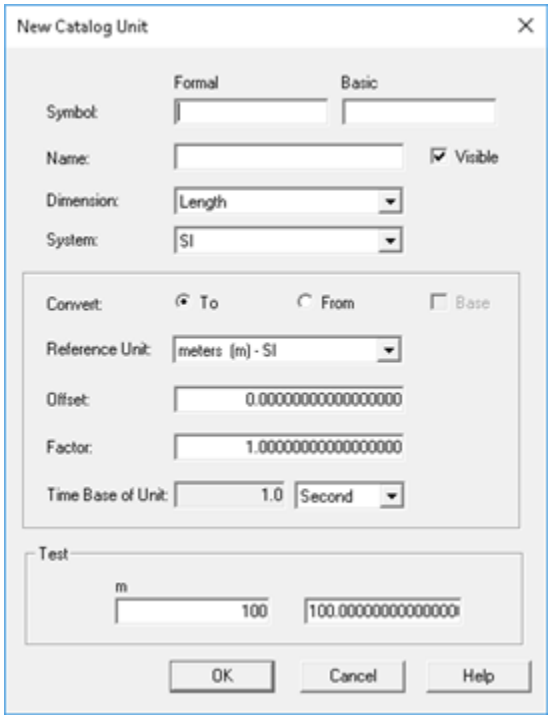
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding a Catalog Unit

Although you can add custom units to the catalog, before doing so you should make sure the required unit is not already defined. Dimensions can have multiple valid names, so you should check under alternate dimension names before creating a new unit. For example, *length* and *distance* are different names for the same dimension, as are *voltage* and *electric potential*.

To add an engineering catalog unit:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Engineering Units Catalog**, and then click **New Engineering Unit Catalog...**. The **New Catalog Unit** dialog displays.



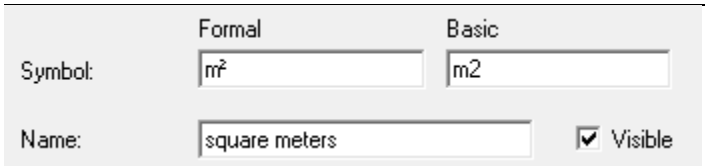
4. For the unit's **Symbol**, enter **Formal** and **Basic** values.

   If the **Basic** value is blank, it is automatically populated with the **Formal** value.

   The **Basic** value is displayed as the unit's symbol by default.

   Notes: Unit symbols are case-sensitive and unique.

   In many cases, the formal and basic symbols are identical. In some cases, such as when extended characters are part of the symbol, it is appropriate to use the formal field for those values.



5. Enter a **Name** to describe the unit.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

6. Selecting **Visible** allows this unit to be used as a canonical unit when creating an engineering unit. See Adding an Engineering Unit for more information.
7. Select the **Dimension** and unit **System** this unit is associated with. See Managing Dimensions, Unit Systems, and Catalog Unit Visibility for more information.
8. If the selected dimension does not already have a base unit defined for the selected unit system, select **Base** to define this new unit as the base unit. If the dimension already has a base unit defined, this box is disabled.
9. Select **OK** to save the unit.

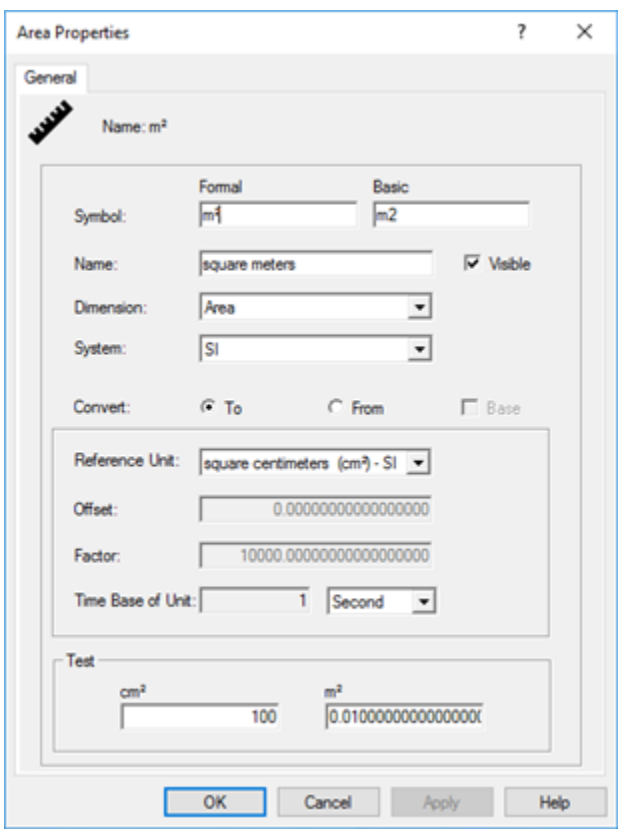
   See Editing a Catalog Unit for more information.

Editing a Catalog Unit

To edit an engineering catalog unit

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Engineering Units Catalog**.
4. In the details pane, double-click the engineering unit catalog you want to edit. The **Properties** dialog displays.

   Note: Modifying a system-defined catalog is not recommended. If you select a system-defined catalog, a warning message displays. Click **Yes** to continue to the **Properties** dialog.



5. For the unit's **Symbol**, enter **Formal** and **Basic** values.

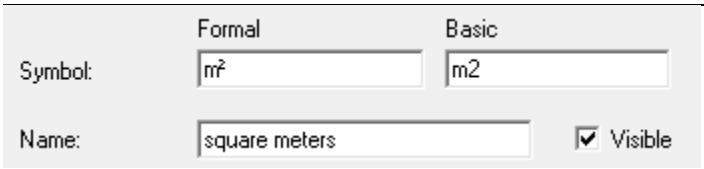
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If the **Basic** value is blank, it is automatically populated with the **Formal** value.

The **Basic** value is displayed as the unit's symbol by default.

Note: Unit symbols are case-sensitive and unique.

In many cases, the formal and basic symbols are identical. In some cases, such as when extended characters are part of the symbol, it is appropriate to use the formal field for those values.



6. Enter a **Name** to describe the unit.
7. Selecting **Visible** allows this unit to be used as a canonical unit when creating an engineering unit. See Adding an Engineering Unit for more information.
8. Select the **Dimension** and unit **System** this unit is associated with. See Managing Dimensions, Unit Systems, and Catalog Unit Visibility for more information.
9. If the selected dimension does not already have a base unit defined for the selected unit system, select **Base** to define this new unit as the base unit. If the dimension already has a base unit defined, this box is disabled.
10. The remaining fields are used to define the relationship between this unit and the other units in this dimension.

    Select a **Reference Unit** from the list. The list contains all visible units for the selected dimension, in the following format:

    Unit Name (*Formal Symbol*) - *Unit System*

    The fields in the **Test** section are labeled with the formal symbol for this unit, and the selected **Reference Unit**. Selecting **To** converts from this unit to the selected reference unit, while selecting **From** converts from the selected reference unit to this unit. Changing the value in one of the test fields changes the value in the other field based on this formula:

    new value = (*original value* / *factor*) - *offset*

    Change the values of **Offset** and **Factor** to define how to convert the new unit to the reference unit. You can use the **Test** fields to verify the results. See Example: Adding a Catalog Unit for more information.
11. Enter the **Time Base of Unit**. When converting rates to accumulated quantities, for example "liters/minute" to "liters", or "kilometers/hour" to "kilometers", Historian must know the time basis of the rate measurement to calculate the quantity. For units that do not measure rates, you can leave this set to the default value of "seconds". For units that do measure rates (e.g. "per minute", "per hour", etc.) be sure to select an appropriate value. The most commonly used values are listed in the drop-down list, but you can set a custom value if the provided options do not meet your needs. For example, if the rate is measured in units/week, select **Custom** and enter 604800, as 1 week = 604800 seconds.
12. Select **OK** or **Apply** to save your changes.

Note: The values of the **Reference Unit**, **To**, **From**, and **Test** fields are only used for displaying the unit conversion calculations, and are not persisted to the database.

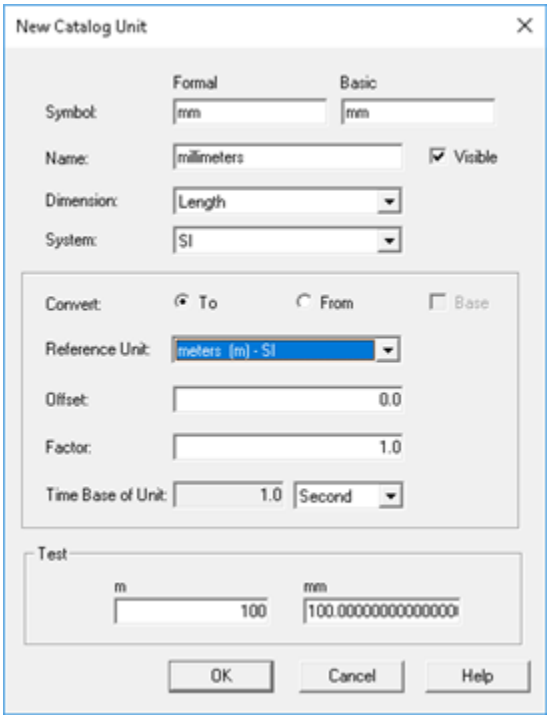
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example: Adding a Catalog Unit

This example demonstrates adding millimeters as a catalog unit.

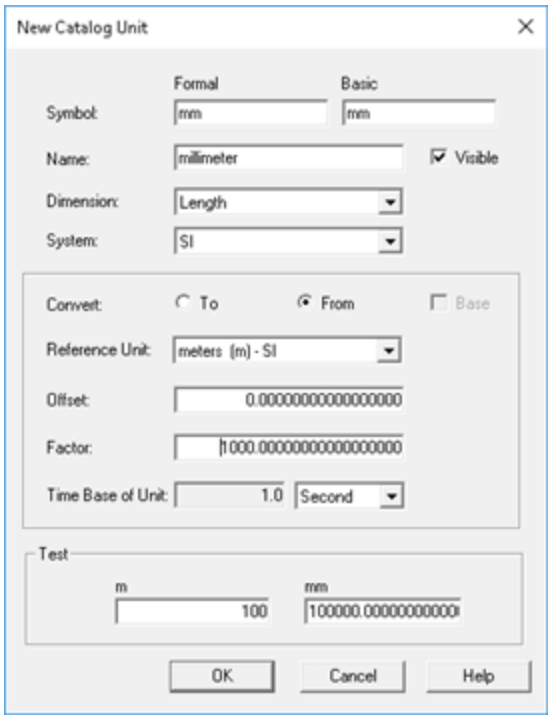
To add a catalog unit:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Engineering Units Catalog**, and then click **New Engineering Unit Catalog...**. The **New Catalog Unit** dialog displays.
4. Enter the following values:
   - **Name**: millimeters
   - **Symbol** (**Formal** and **Basic**): mm
   - **Dimension**: Length
   - **System**: SI
   - **Reference Unit**: meters (m) - SI

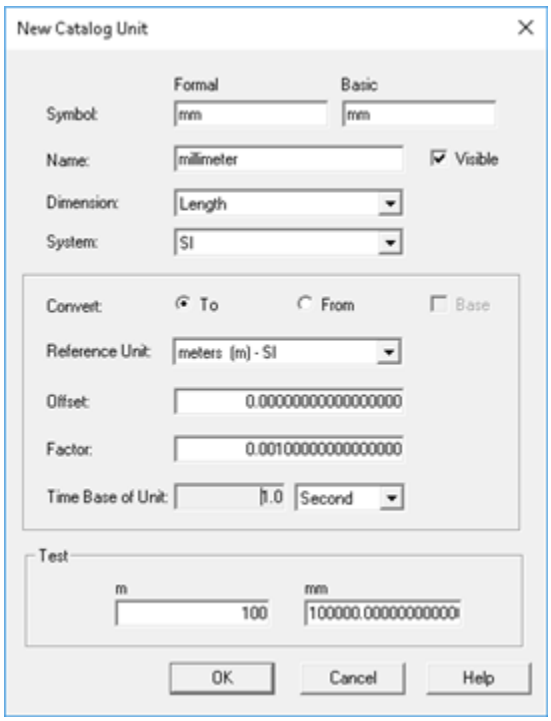


5. A meter is equal to 1000 millimeters. To define this, select **From** as the convert option, and enter 1000 for the **Factor**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

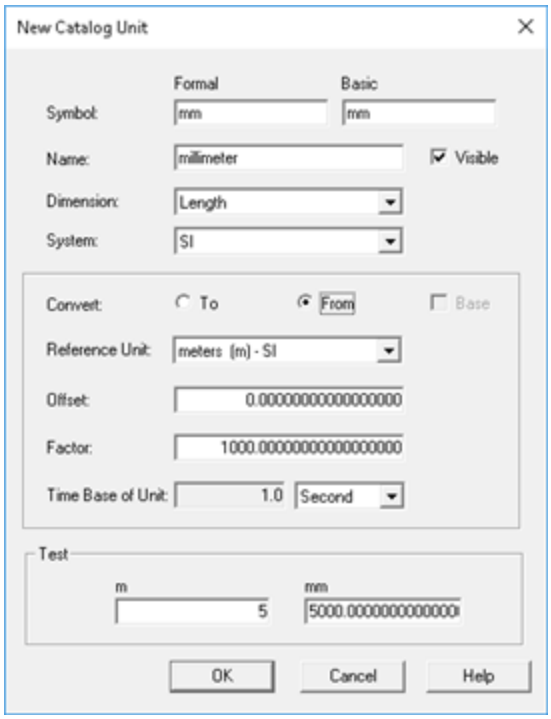


You can also define the conversion in the opposite direction. In this case, a millimeter is equal to 0.001 meters. To define this, select **To** as the convert option, and enter 0.001 for the **Factor**.



6. You can now use the **Test** fields to verify the conversion factor and offset are set correctly by changing the value of the **m** or **mm** fields, and seeing how the other is affected. For example, entering 5 in the **m** field calculates the number of **mm** as 5000.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. When you are satisfied the conversion values are correct, select **OK** to save the unit.

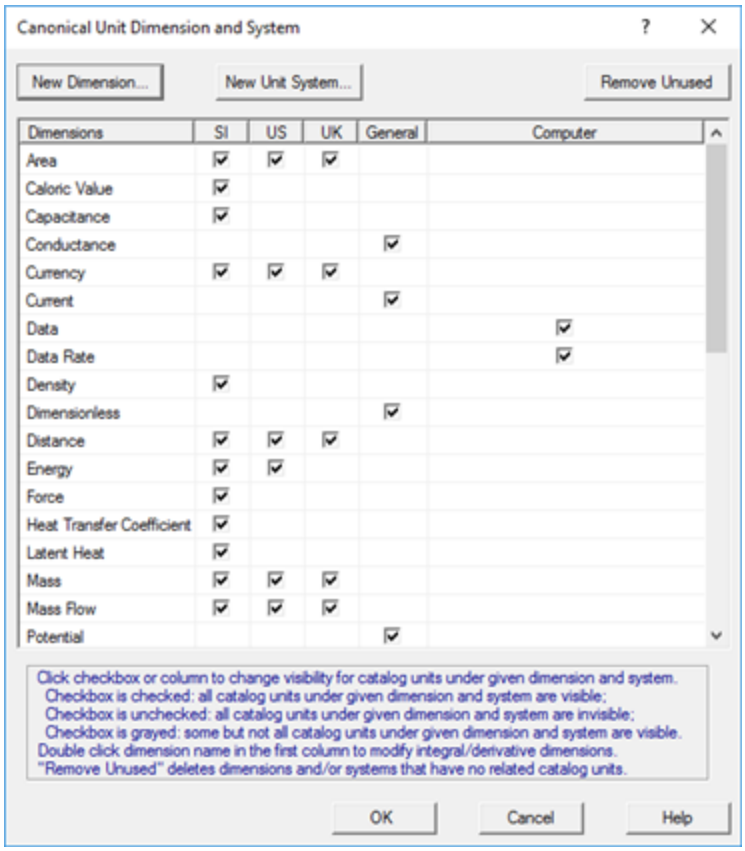
   Note: Before the new catalog unit can be used for unit conversion, a corresponding engineering unit must be added and mapped to the catalog unit. See Configuring Engineering Units for more information.

Managing Dimensions, Unit Systems, and Catalog Unit Visibility

To manage dimensions, unit systems, and the visibility of catalog units:

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Engineering Units Catalog**, and then select **Unit Dimensions and Systems...**. The **Canonical Unit Dimension and System** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Dimensions are listed in the left column, and the remaining column headers indicate the unit systems. The presence of a checkbox at the intersection of a dimension and unit system indicates that there is at least one catalog unit defined for that dimension/unit system combination, and the state of the checkbox indicates the visibility of the defined catalog units.

☑	A checked box indicates all units are visible.
☑	A checked box with a grey background indicates at least one unit is visible.
☐	A cleared box indicates no units are visible.

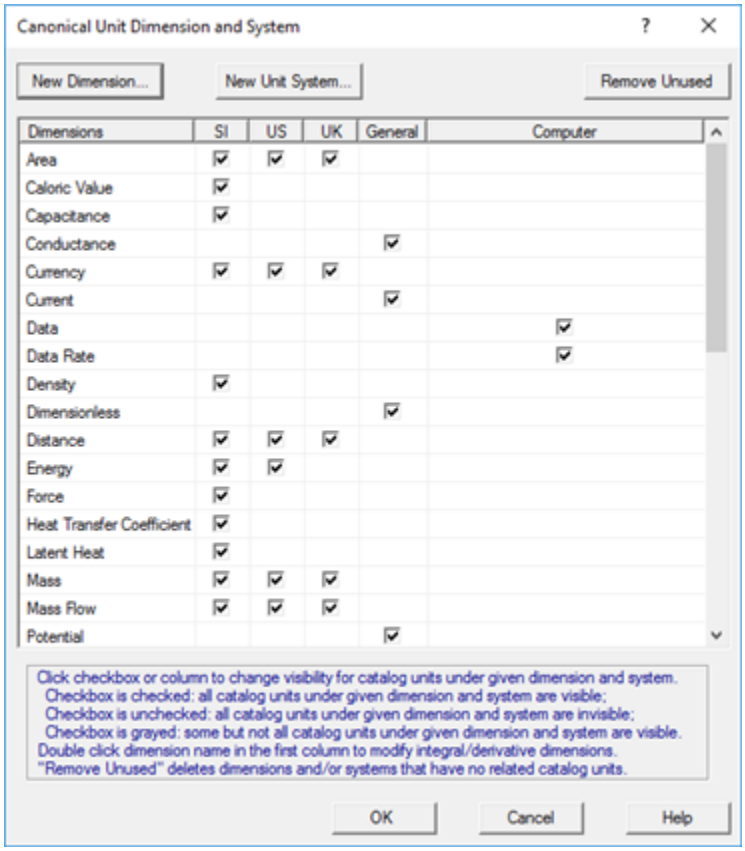
4. Select **New Dimension...** to add a new dimension. See Adding or Editing a Dimension for more information.
5. Select **New Unit System...** to add a new unit system. See Adding a Unit System for more information.
6. Select **Remove Unused** to delete any dimensions and unit systems for which there are no catalog units defined.
7. Double-click the name of a dimension in the **Dimensions** column to edit an existing dimension.
8. Select individual checkboxes to toggle visibility for dimension/unit system combinations. You can also click a column header to toggle every checkbox in that column on or off.
9. Select **OK** to save any visibility changes, or **Cancel** to close the dialog without saving.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

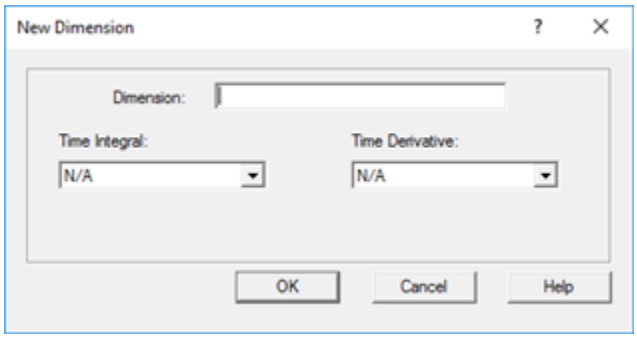
Adding or Editing a Dimension

To add or edit a dimension:

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Engineering Units Catalog**, and then select **Unit Dimensions and Systems...**. The **Canonical Unit Dimension and System** dialog displays.



4. Select **New Dimension...** to create a new dimension, or double-click the name of a dimension in the **Dimensions** column to edit it. The dimension properties dialog displays.

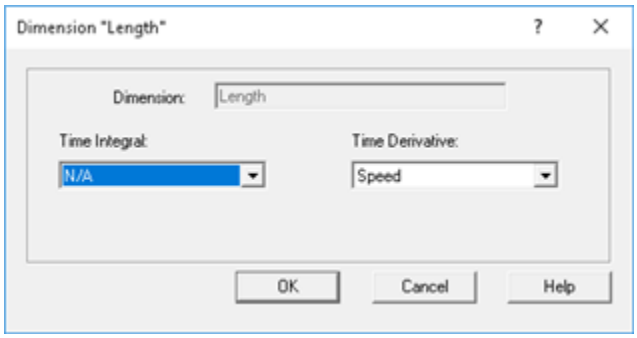


5. If you are creating a new dimension, enter a name in the **Dimension** field. If you are editing an existing dimension, you cannot change the name and this field is read-only.

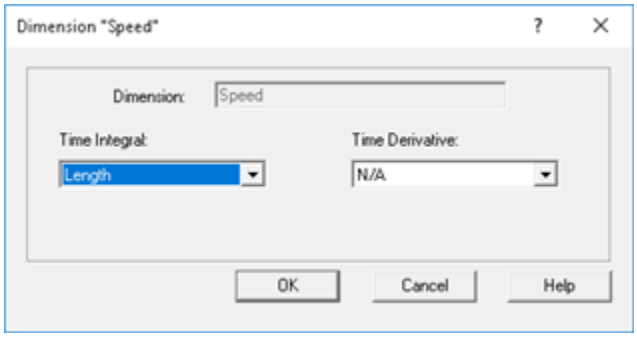
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

6. If the dimension has a corresponding integral or derivative dimension, select the **Time Integral** dimension and/or **Time Derivative** dimension from the list. Otherwise, choose N/A. The time integral dimension is appropriate for converting a dimension that measures a rate, while the time derivative is appropriate for converting a dimension that measures a quantity.

   For example, the "Length" dimension's time derivative is speed.



   Likewise, the "Speed" dimension's time integral is length.



   Note: Time integrals and time derivatives can be used in "integral" or "rate of change" queries to request data in alternate units. For example, a "meters/second" tag can be used to query the distance traveled over a period of time.

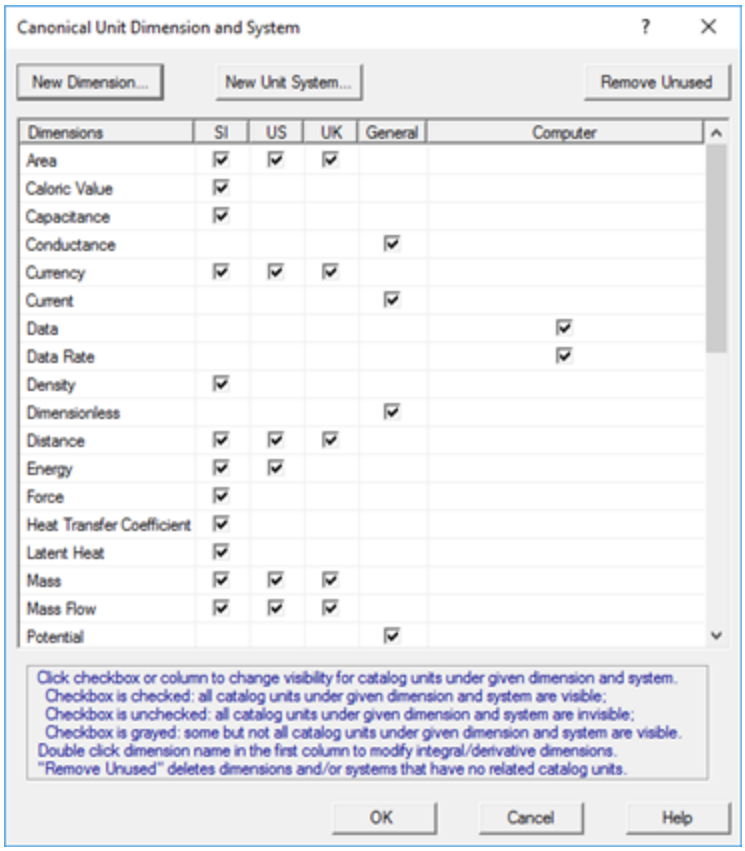
7. Select **OK** when you are done editing.

Adding a Unit System

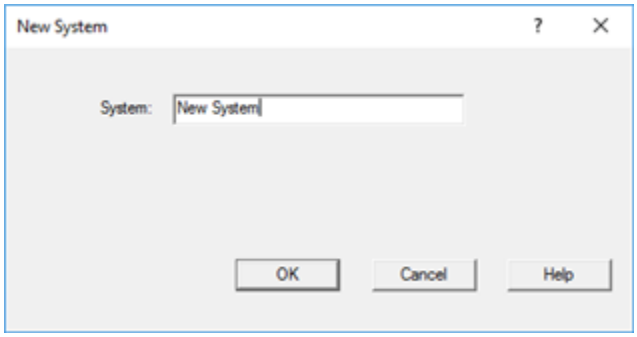
To add a new unit system:

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Engineering Units Catalog**, and then select **Unit Dimensions and Systems...**. The **Canonical Unit Dimension and System** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. Select **New Unit System...**. The **New System** dialog displays.

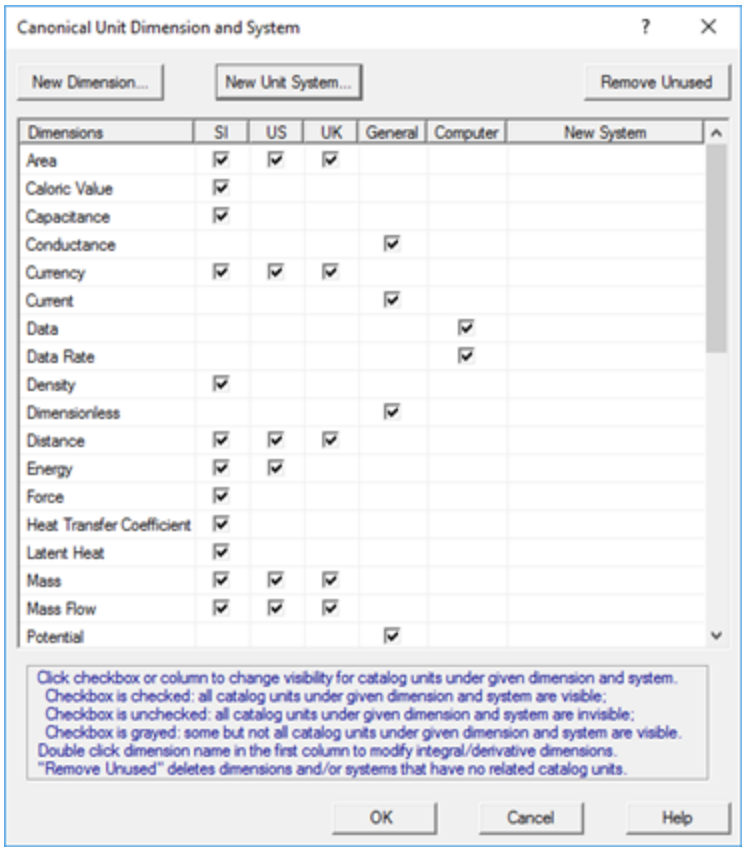


5. In the **System** field, enter a name for the new unit system.

   Select **OK** to create the new system.

6. The new unit system displays as the final column in the **Canonical Unit Dimension and System** dialog.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Configuring Engineering Units

An engineering unit is the unit of measure for an analog tag. Examples of engineering unit types are CPS, pounds, or degrees.

Note: Engineering units are case-sensitive. That means that AVEVA Historian differentiates between ml (milliliters) and ML (megaliters), for example.

A tag's engineering unit displays in charts or reports where the tag is used. The Historian can only convert data between units defined in the engineering units catalog. To use your own familiar labels, you can create engineering units and map them to canonical units in the engineering units catalog. See Configuring the Engineering Units Catalog for more information.

Viewing Defined Engineering Units

Engineering units are configured for a single server. Any defined engineering unit can be used when adding or editing an analog tag.

To view defined engineering units

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Expand **Engineering Units** to view all currently-defined engineering units.

   Note: Engineering units are case-sensitive. That means AVEVA Historian differentiates between ml

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

(milliliters) and ML (megaliters), for example.

4. When you select an engineering unit in the tree, a list of tags that make use of that engineering unit appears in the details pane.

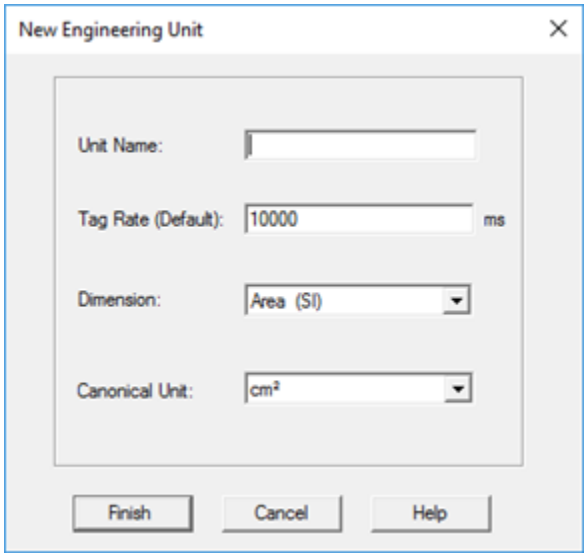
Adding an Engineering Unit

After you add an engineering unit, you can configure both existing and new analog tags to use the new engineering unit.

Note: Engineering units are case-sensitive. That means AVEVA Historian differentiates between ml (milliliters) and ML (megaliters), for example.

To add an engineering unit

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Engineering Units**, and then click **New Engineering Unit**. The **New Engineering Unit** wizard displays.



4. Enter the **Unit Name.**
5. In the **Tag Rate** box, enter the default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag.
6. Select the appropriate **Dimension** for the engineering unit. The values displayed in this list are a combination of the available dimensions from the database, and the unit system in parentheses.
7. Select a **Canonical Unit** for the engineering unit. This list contains all visible units from the Engineering Units Catalog corresponding with the selected dimension. See Configuring the Engineering Units Catalog for more information.

   Notes: Some units with the same name or symbol may have different definitions in different unit systems, so take care to select the correct unit. For example, the U.S. gallon and imperial gallon have different volumes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

While it is possible to map multiple engineering units to the same canonical unit, to avoid confusion it is not recommended. For example, you could choose to create three engineering units named *liters*, *litres*, and *L*, and map them all to *L* in the engineering units catalog.

8. Click **Finish**.

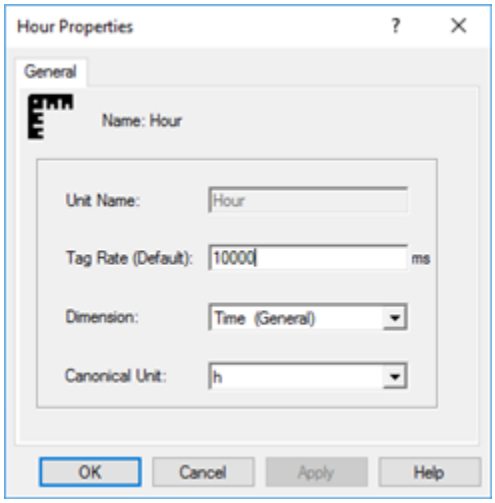
Editing an Engineering Unit

If you make changes to an engineering unit, the changes are applied to all tags in the system that are currently using that engineering unit.

Note: Engineering units are case-sensitive. That means AVEVA Historian differentiates between ml (milliliters) and ML (megaliters), for example.

To edit an engineering unit

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Engineering Units**.
4. In the details pane, double-click the engineering unit to edit. The **Properties** dialog box displays.



5. In the **Tag Rate** box, enter the default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag.
6. Select a **Canonical Unit** for the engineering unit. This list contains all visible units from the Engineering Units Catalog corresponding with the selected dimension. See Configuring the Engineering Units Catalog for more information.
7. Click **OK**.

Configuring Discrete Tags

You can configure general information and acquisition details for a selected discrete tag, as well as add new discrete tags to the system.

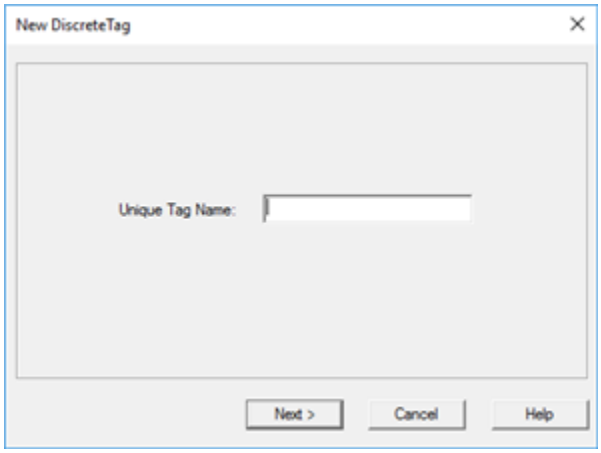
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding a Discrete Tag

Be sure that you do not exceed your licensed tag count by adding another tag.

To add a discrete tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Discrete Tags**, and then click **New Tag**. The **New Discrete Tag** wizard displays.



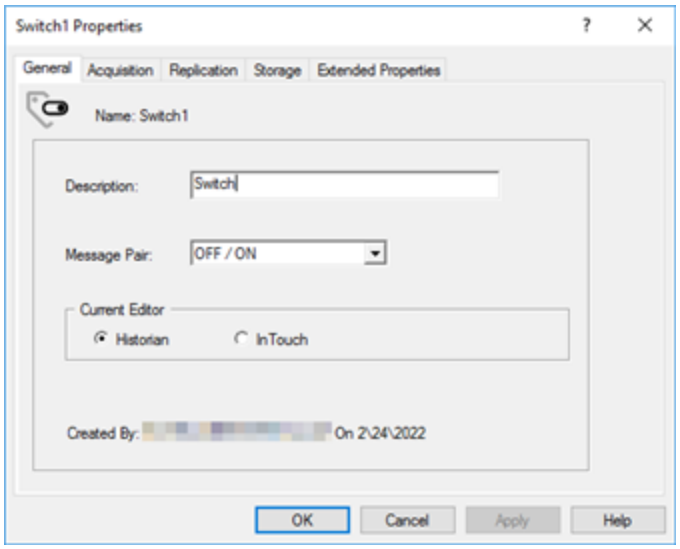
4. Type a unique name for the discrete tag and click **Next**. For information on allowable tag names, see Tag Naming Conventions.
5. You are then prompted to define general, acquisition, and storage information for the tag.
   - For more information on configuring general properties, see Editing General Information for a Discrete Tag.
   - For more information on configuring acquisition, see Editing Acquisition Information for a Tag.
   - For more information on configuring storage, see Editing Storage Information for a Discrete Tag.
6. When you are finished defining the new discrete tag, click **Finish**.

Editing General Information for a Discrete Tag

To edit general information for a discrete tag:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Discrete Tags**.
4. In the details pane, double-click the discrete tag to edit. The **Properties** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select the **General** tab.
6. In the **Description** box, type a description of the tag.
7. In the **Message Pair** list, select the message pair to associate with the FALSE/TRUE states of the discrete tag.

   For information on adding a message pair to the system, see Configuring Message Pairs.
8. In the **Current Editor** group, specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a reimport is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for reimporting; however, all changes made to the tag using the Configuration Editor are lost during the reimport. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved.
9. Click **OK**.

Editing Storage Information for a Discrete Tag

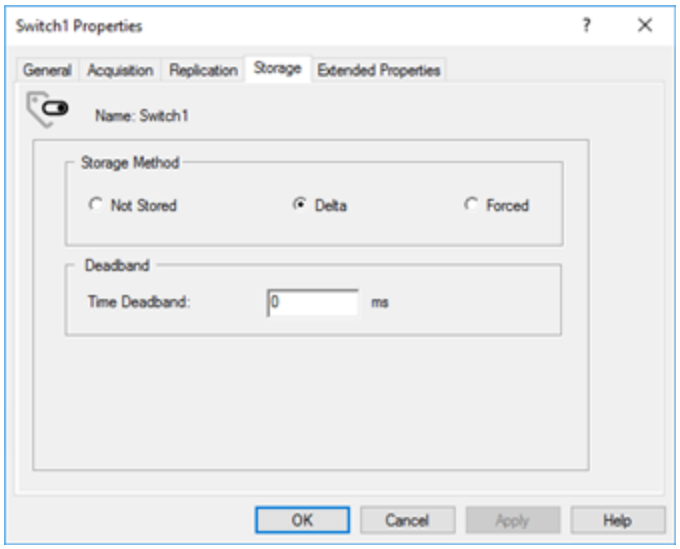
If you change a tag's configuration, the changes are applied only to data with timestamps that are equal to or greater than the timestamp of the configuration change.

For related information about data storage, see Managing Data Storage.

To edit storage information for a discrete tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Discrete Tags**.
4. In the details pane, double-click the discrete tag to edit. The **Properties** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select the **Storage** tab.
6. In the **Storage Method** area, select the way in which values for the tag are stored.
7. In the **Deadband** group, configure details for how the tag value are stored. The availability of this group depends on which storage method you select.
   - **Time Deadband**
     A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.
8. Click **OK**.

Editing Extended Properties for a Discrete Tag

Extended properties can be used to add application-specific metadata to tags. When you search for tags in AVEVA Historian Client Web or AVEVA Insight, you can locate tags based on these property values. Likewise, with SQL queries you can select tags based on these property values without needing to know the tag name.

The following extended properties are predefined by the system, and can be added to any discrete tags:

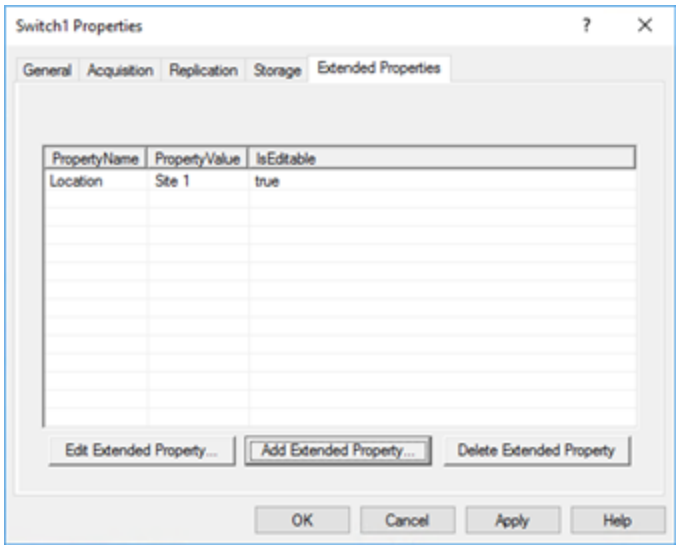
- **Alias** - Used as the tag's preferred label in charts.
- **Location** - Places the tag within the asset model of AVEVA Insight.

You can define your own custom extended properties by using the database import/export feature. See Adding New Tag Extended Property Values for more information.

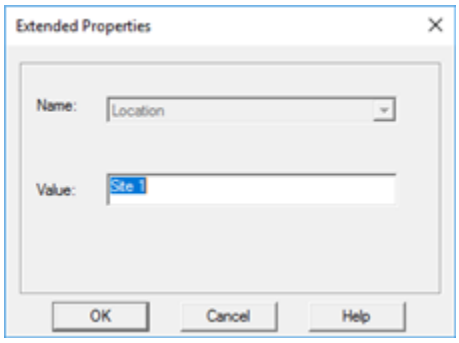
To edit extended property information for a discrete tag:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Discrete Tags**.
4. In the details pane, double-click the discrete tag you want to edit. The **Properties** dialog displays.

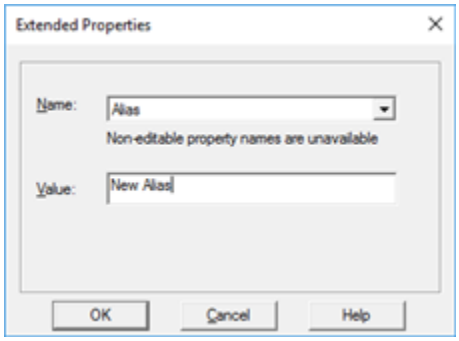
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select the **Extended Properties** tab.
6. To edit an existing property value, select the property in the list and click **Edit Extended Property...**. The **Extended Properties** dialog displays.



7. Enter a new **Value** for the property, then click **OK**.
8. To add a property to a tag, click **Add Extended Property...**. The **Extended Properties** dialog displays.

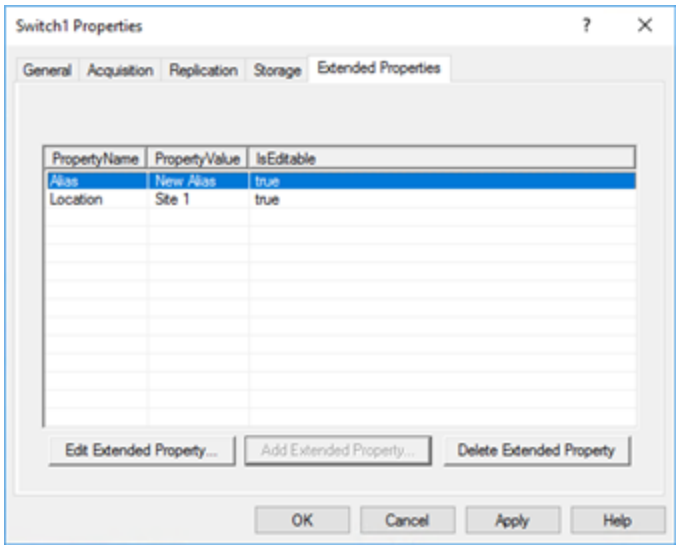


9. Select the **Name** of the extended property from the drop-down list. The list contains all editable extended properties that are not already present on the tag.

   Note: You can add new extended properties by exporting a configuration text file, adding new TagExtendedPropertyName entries, and importing the modified file. For more information, see Importing or Exporting Tag Information and Editing the Configuration Text File.

10. Enter a **Value** for the property, then click **OK**. The new property is added to the list.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



11. To remove a property from a tag, select the property and click **Delete Extended Property**. The property is removed from the list.
12. Click **OK** or **Apply** to save your changes.

Configuring Message Pairs

A message pair consists of the messages associated with the FALSE or TRUE state of the discrete tag. A discrete tag set to 0 is in the FALSE, or OFF, state. A discrete tag set to 1 is in the TRUE, or ON, state.

Viewing the Current Message Pairs for a Server

Message pairs are configured for a single server. Any defined message pair can be used when defining or editing a discrete tag.

To view all message pairs

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Messages** to view all currently-defined message pairs in the details pane.

   If you expand **Messages** and then select a message in the console tree, a list of tags that make use of that message appears in the details pane.

Editing a Message Pair

If you make changes to a message pair, that change is applied to all tags in the system that are currently using that message pair.

To edit a message pair

1. In the Operations Control Management Console, expand a server group and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **Messages**.
4. In the details pane, double-click the message to edit. The **Properties** dialog box appears.

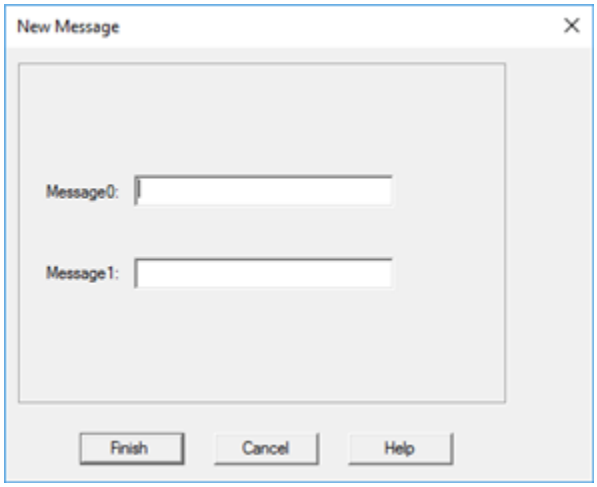
5. In the **Message0** box, type the message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state.
6. In the **Message1** box, type the message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state.
7. Click **OK**.

Adding a Message Pair

If you add a message pair, you can configure both existing and new discrete tags to use the new message pair.

To add a message pair

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Messages**, and then click **New Message**. The **New Message** wizard appears.



4. In the **Message0** box, type the message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. In the **Message1** box, type the message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state.
6. Click **Finish**.

Configuring String Tags

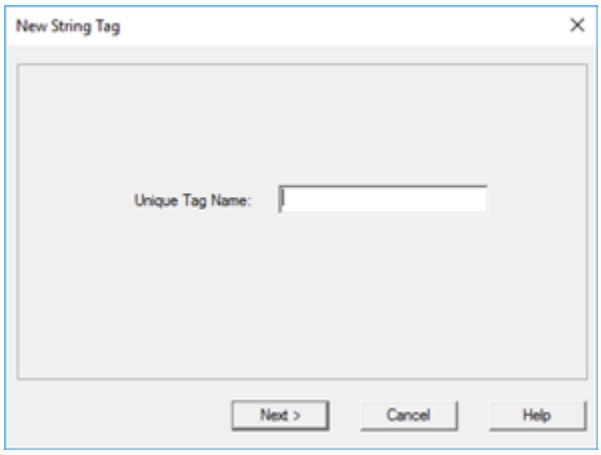
You can configure general information and acquisition details for a selected string tag, as well as add new string tags to the system.

Adding a String Tag

Be sure that you do not exceed your licensed tag count by adding another tag.

To add a string tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **String Tags**, and then click **New Tag**. The **New String Tag** wizard displays.



4. Type a unique name for the string tag and click **Next**. For information on allowable tag names, see Tag Naming Conventions.
5. You are then prompted to define general, acquisition, and storage information for the tag.
   - For more information on configuring general properties, see Editing General Information for a String Tag.
   - For more information on configuring acquisition, see Editing Acquisition Information for a Tag.
   - For more information on configuring storage, see Editing Storage Information for a String Tag.
6. Click **Finish**.

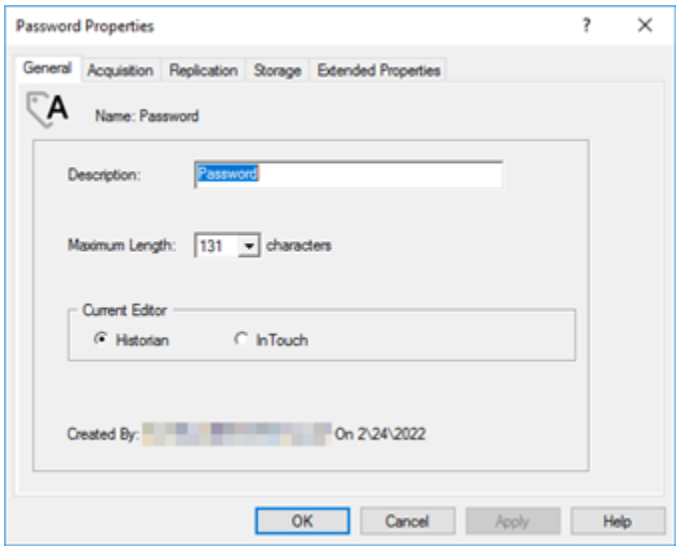
Editing General Information for a String Tag

To edit general information for a string tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **String Tags**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

4. In the details pane, double-click the string tag to edit. The **Properties** dialog displays.



5. In the **Description** box, type a description of the tag.
6. In the **Maximum Length** list, select the maximum number of characters for the string.

   Note: If the maximum length specified is 131 or more characters, the string is considered variable length. If you create a new string tag, it's best to set its maximum length to 131 characters and not use other values, which are provided only for backward compatibility. The variable-length string tags values are internally limited by 512 characters.

7. In the **Current Editor** group, specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a reimport is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for reimporting; however, all changes made to the tag using the Configuration Editor are lost during the reimport. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved.
8. Click **OK**.

Editing Storage Information for a String Tag

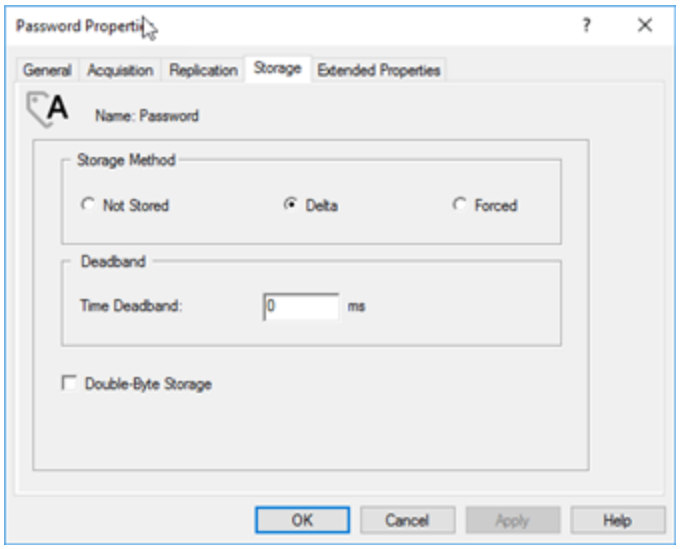
For more information on storage, see Managing Data Storage.

If you change the configuration, then the changes are applied only to data with timestamps that are equal to or greater than the timestamp of the configuration change.

To edit storage information for a string tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **String Tags**.
4. In the details pane, double-click the string tag to edit. The **Properties** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In the **Storage Method** group, select the way in which values for the tag are stored.
6. In the **Deadband** group, configure details for how the tag value is stored. The availability of this group depends on which storage method you select.

   Time Deadband

   A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

   Select the **Double-Byte Storage** check box to store the string as a double-byte UTF-16 Unicode string using 2 bytes per character. If you create a new string tag, it is recommended to select **Double-Byte Storage** for better compatibility with the Historian SDK.
7. Click **OK**.

Editing Extended Properties for a String Tag

Extended properties can be used to add application-specific metadata to tags. When you search for tags in AVEVA Historian Client Web or AVEVA Insight, you can locate tags based on these property values. Likewise, with SQL queries you can select tags based on these property values without needing to know the tag name.

The following extended properties are predefined by the system, and can be added to any analog tags:

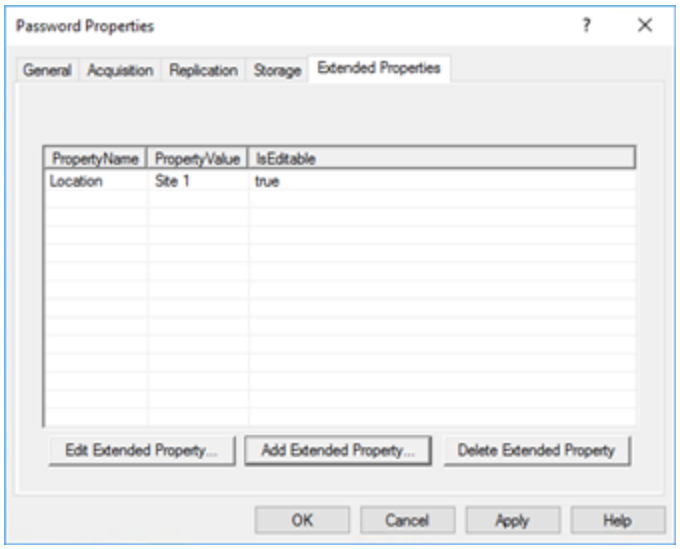
- **Alias** - Used as the tag's preferred label in charts.
- **Location** - Places the tag within the asset model of AVEVA Insight.

You can define your own custom extended properties by using the database import/export feature. See Adding New Tag Extended Property Values for more information.

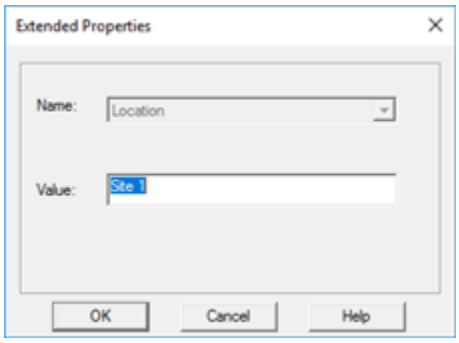
To edit extended property information for a string tag:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select **String Tags**.
4. In the details pane, double-click the discrete tag you want to edit. The **Properties** dialog displays.

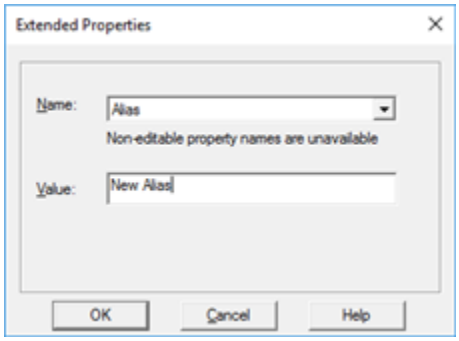
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select the **Extended Properties** tab.
6. To edit an existing property value, select the property in the list and click **Edit Extended Property...**. The **Extended Properties** dialog displays.



7. Enter a new **Value** for the property, then click **OK**.
8. To add a property to a tag, click **Add Extended Property...**. The **Extended Properties** dialog displays.

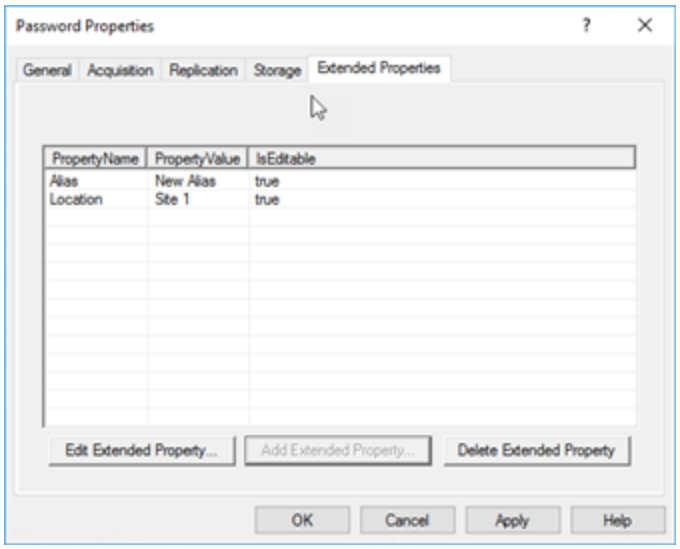


9. Select the **Name** of the extended property from the drop-down list. The list contains all editable extended properties that are not already present on the tag.

   Note: You can add new extended properties by exporting a configuration text file, adding new TagExtendedPropertyName entries, and importing the modified file. For more information, see Importing or Exporting Tag Information and Editing the Configuration Text File.

10. Enter a **Value** for the property, then click **OK**. The new property is added to the list.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



11. To remove a property from a tag, select the property and click **Delete Extended Property**. The property is removed from the list.
12. Click **OK** or **Apply** to save your changes.

Configuring Event Tags

Event tags are a special tag type that allow you to track when a condition or other specific event happens in relation to one or more specified tags. For example, you could create an event tag to indicate when a temperature control (represented by a certain analog tag) registers 100 degrees or more. Or, you could create an event tag to track each time a particular discrete tag registered "Jam" as the reason why a component stopped running.

Be sure that you do not exceed your licensed tag count when adding event tags.

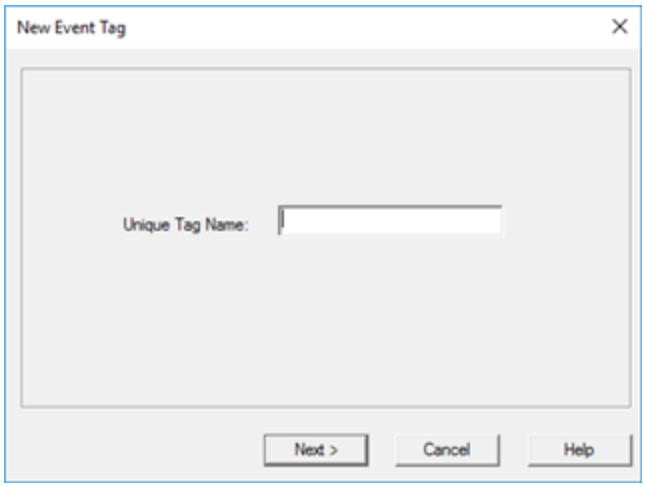
Note: For information about managing event tags that you created event tags through the Classic Event subsystem, see Configuring Classic Events.

Adding an Event Tag

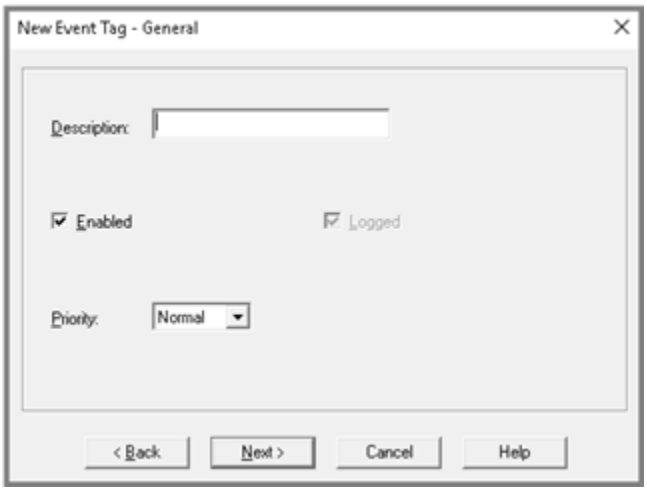
To add an event tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Right-click **Events Tags**, and then click **New Tag**.
4. The **New Event Tag** wizard appears. Type a unique name for the event, and click Next.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

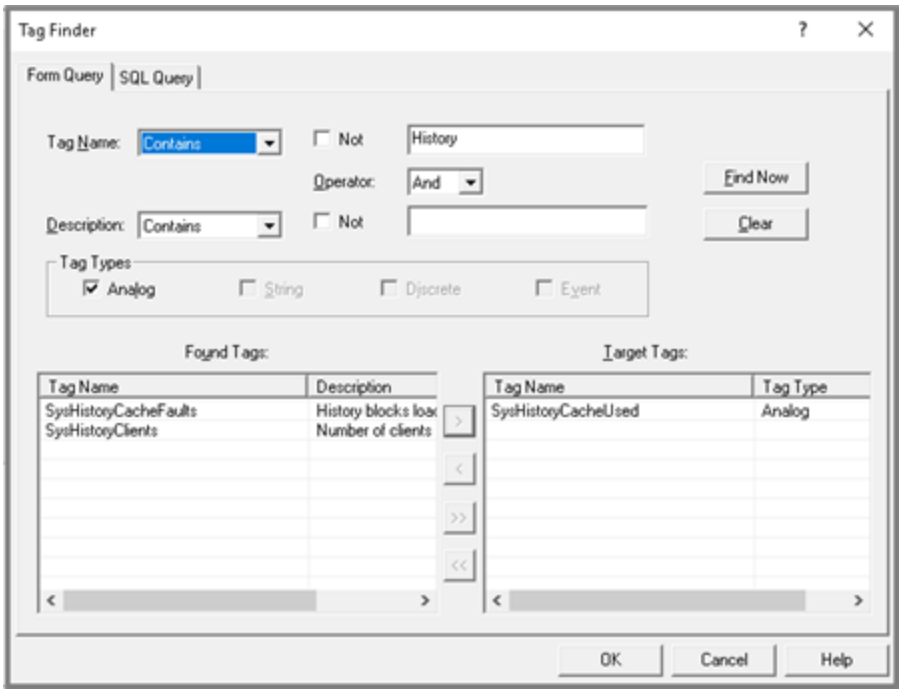


5. Type a description and click Next.

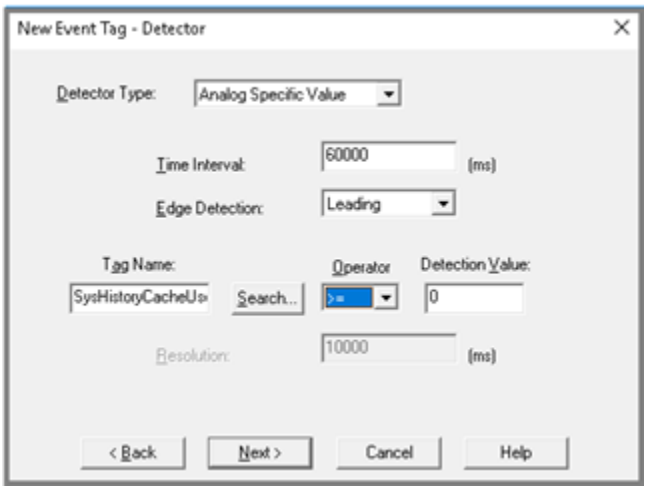


6. Use the Tag Finder to locate the tag or tags associated with this event.
   a. Type search criteria in the **TagName** and **Description** fields and click Find Now.
   b. From the **Found Tags** box, mark the tag(s) you want to associate with this event, and click the **>** button to move them to the **Target Tags** box. Click **OK**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

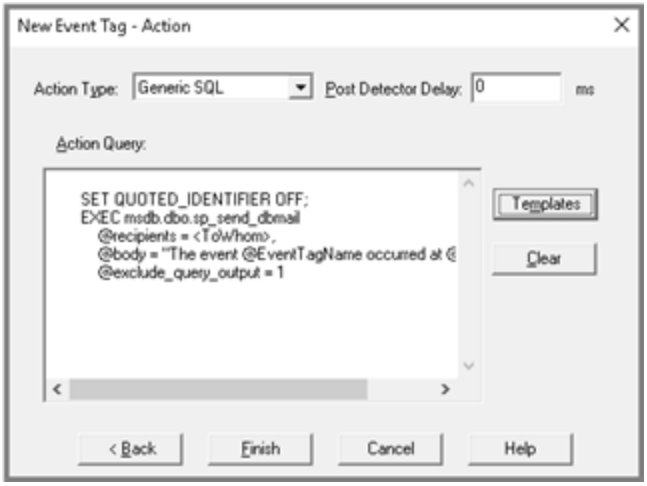


7. From the Detection Type dropdown, select a detector type for this event. Apply a time interval, edge detection, and other criteria as appropriate. Click **Next**.



8. From the **Action Type** dropdown, select the action that Historian will take as a result of this event. Click **Finish**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



9. When you are done defining the new analog tag, click **Finish**.

Copying Tag Definitions

You can use an existing tag definition as the basis for additional tag definitions.

To copy a tag definition

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select the appropriate tag type folder (for example, **Analog Tags**) so that the list of available tags appears in the details pane.
4. Perform any of the following:
   - Right-click the tag to copy in the details pane and click **Copy**. Then, right-click the tag type folder and click **Paste**.
   - Select the tag to copy and then drag it onto the tag type folder.

     The new tag wizard appears with the definition options of the copied tag set as defaults. The name of the new tag is the name of the copied tag, appended with a number. For example, "MyTag2."
5. Use the wizard to change any of the options for the new tag.
   - To change an analog tag, see Adding an Analog Tag.
   - To change a discrete tag, see Adding a Discrete Tag.
   - To change a string tag, see Adding a String Tag.
   - To change an event tag, see Adding an Event Tag.

Deleting a Tag

To delete a tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select the tag in the details pane and perform any of the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Click the **Delete** button on the toolbar.
- On the **Action** menu, click **Delete**.
- Right-click the tag and then click **Delete**.

Organizing Tags into Groups

In the Operations Control Management Console, tags are organized into two main groups:

- **Public Groups folder**
  The Public Groups folder contains all objects that are visible to all clients. If you have administrative permissions, you can create, rename, and delete groups in the Public Groups folder.

  You cannot change the following default groups:

• All Analog Tags	• All Event Tags
• All Discrete Tags	• InTouch Nodes
• All String Tags	• System Status Tags

- **Private Groups folder**
  The Private Groups folder contains all objects that are visible to the database user that is currently logged on. In SQL Server, database user accounts and Windows user accounts are mapped to login IDs. Users can create, rename, and delete groups in their Private Groups folder. Also, by default, every domain user can have their own private namespace and data annotation entries. You can turn off this functionality using the GroupedPrivateNamespace system parameter.

The console tree shortcut menu contains commands for adding groups in the hierarchy and adding tags to them. Open the shortcut menu by right-clicking on the item in the console tree. Add a group just as you add a new folder in the Windows Explorer. For example, create the "BoilerTags" group under the existing "Private Groups" group.

When you add tags to a new group, the original reference still appears in the default system group under **Tag Configuration** in the console tree. Any tag can belong to any number of groups, and any group can contain any number of tags.

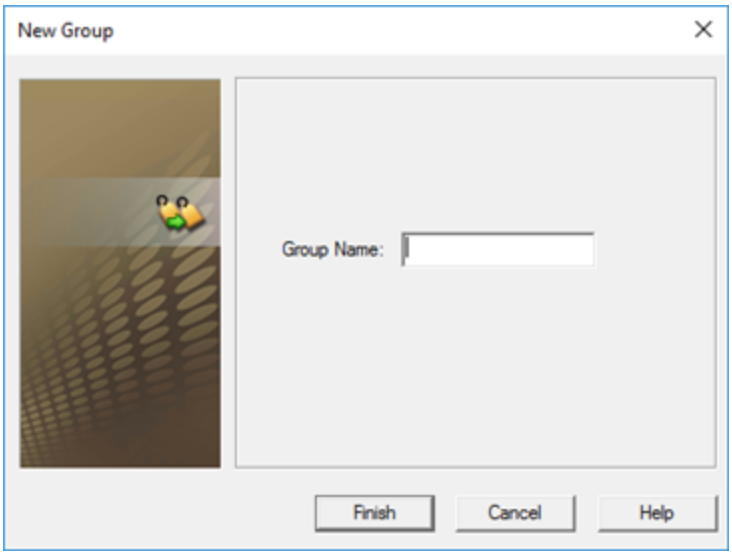
Adding a Group

To add a group

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, and then expand **Public Groups**.
3. Select the folder under which you want to create a group.
4. Perform any of the following:
   - On the **Action** menu, click **New Group**.
   - Right-click and then click **New Group**.
   - Click the **Add** button + on the toolbar.

     The **New Group** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In the **Group Name** box, type a name for the new group. The group name can be up to 255 characters and must be unique.
6. Click **Finish**.

Renaming a Group

You can rename any group that you have created in the console tree, except for public folders or tag references.

To rename a group

1. Select the group in the console tree.
2. Press F2 on your keyboard.

3. Type a new name for the folder and press **Enter**.

Adding a Tag to a Group

If you are a member of the aaUsers group, you can only add tags to a private group.

To add a tag to a group

1. Select the group to which you want to add a tag.
2. Do any of the following:
   - Drag the tag from the details pane into the folder.
   - Right-click and then click **Add Tags to Group**. The **Tag Finder** dialog box appears, in which you can search for and select tags to add.
   - Click the **Tag Finder** button on the toolbar to open the Tag Finder.

For more information on the Tag Finder, see Using the Tag Finder.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Deleting a Group or Tag Reference

To delete a tag from the AVEVA Historian, you must select the tag in the appropriate folder under **Tag Configuration**. After a tag is deleted, all references to it anywhere in the public or private folders are also deleted. For more information, see Deleting a Tag.

When you delete a private group, the group folder and all references to tags are deleted. The tags themselves are not deleted, and the original reference still appears in the default system group. You cannot delete public folders or the tag references contained in them.

To delete a group or tag reference

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Expand **Configuration Editor**, and then expand **Public Groups** or **Private Groups**.
3. Select the group in the console tree.
4. Delete the item by doing one of the following:
   - On the **Action** menu, click **Delete**.
   - Right-click the group or tag, and then click **Delete**.
   - Click the **Delete** toolbar button .

Filtering Tags in the OCMC Details Pane

In the Operations Control Management Console, you can select certain items in the console tree, a list of associated tags appears in the details pane. For example, if you click the **All Analog Tags** item in the **Public Groups** folder, a list of all analog tags in the system appears. If you click a specific message pair in the **Messages** folder, a list of tags that use that message pair appears.

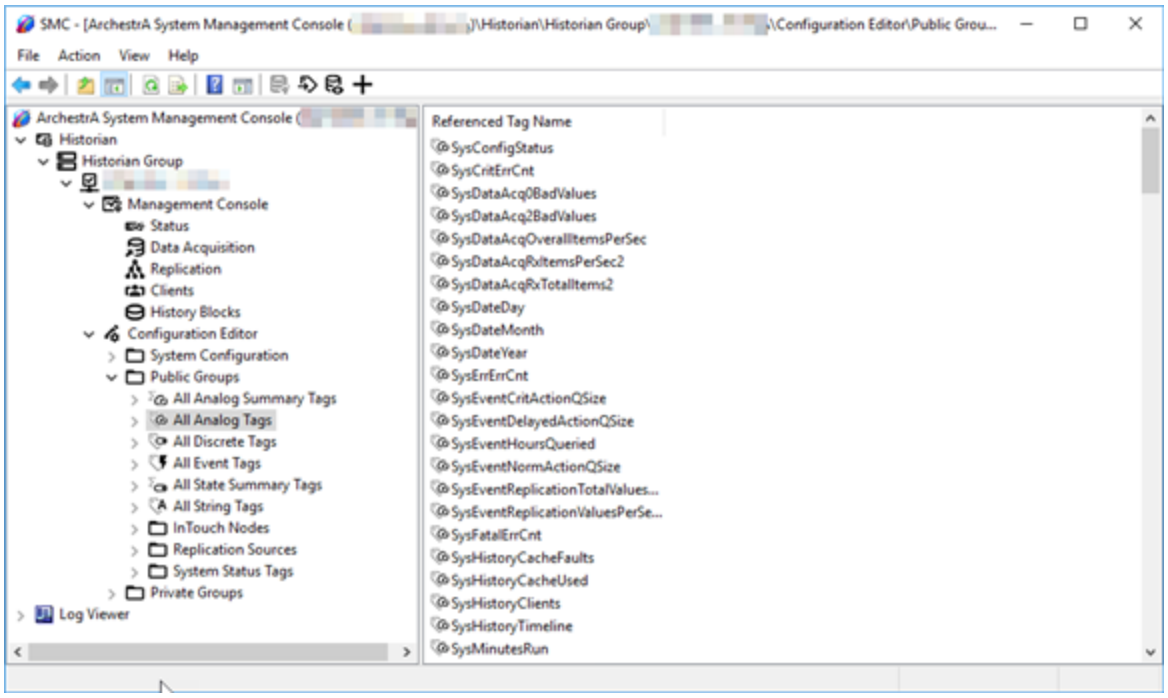
You can apply simple filtering to any list of tags that appears in the details pane. Also, you can configure a different filter for each tag list. A particular filter will remain associated with a list until you remove the filter.

Applying a Filter

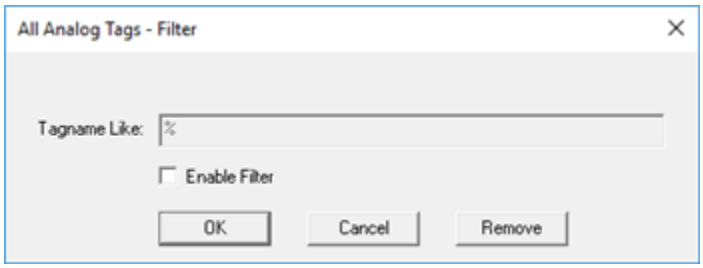
To apply a filter

1. In Operations Control Management Console, expand the console tree.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Right-click in the tag list in the details pane and select **Filter**. The **<item name> - Filter** dialog box appears.



3. Select **Enable Filter**.
4. In the **Tagname Like** box, type the filter string. The maximum length of the filter string is 100 alphanumeric characters.

   When filtering tags, use the percentage sign (%) as a wildcard character for any portion of the filter. For example, "SysDate%" or "SysPerf%ThreadCount."

   Case-sensitivity for the filter depends on the SQL Server settings. If the SQL Server is case-sensitive, then the filter is case-sensitive; if the SQL Server is case-insensitive, then the filter is case-insensitive. Also, the string you use is not validated.
5. Click **OK**.

The details pane refreshes automatically. The word "Filtered" appears in the bottom right corner of the details pane along with the number of tags that matches the filter string.

The filter remains in effect until you disable it or remove it, even if you close the Operations Control Management Console without saving your changes to the .msc file.

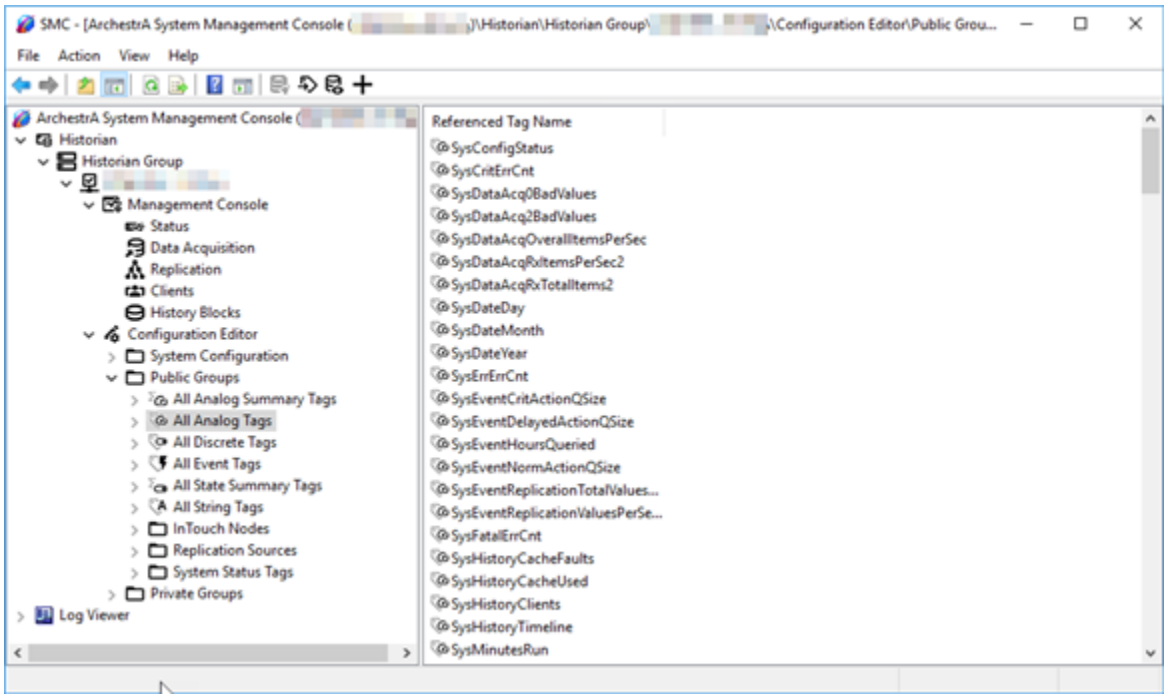
Disabling or Removing a Filter

Disabling the filter allows you to turn the filtering off without completely removing the last filter string you applied.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To disable the current filter

1. In Operations Control Management Console, expand the console tree.



2. Right-click in the tag list in the details pane and click **Filter**. The **<item name> - Filter** dialog box appears.
3. Click to clear the **Enable Filter** check box.
4. Click **OK**. Notice that the filter string remains in the **Tagname Like** box.

To remove a filter

1. Right-click in the tag list in the details pane and click **Filter**. The **<item name> - Filter** dialog box appears.

2. Click **Remove**.

Importing and Exporting Tag Configurations

For the AVEVA Historian, configuration information includes all of the definitions for entities within the system, such as tags, I/O Servers, and summary operations.

You can import information from an InTouch data dictionary (Tagname.x) using the Tag Importer wizard available from within the Operations Control Management Console application.

Also, you can export all of the configuration information from a historian to a text file. This allows for making

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

bulk additions or modifications using third-party tools, such as Microsoft Excel. The modified text file can be imported into the same or a different historian.

Importing an InTouch Data Dictionary

You can import an InTouch tag name database, also called the data dictionary, and use it to configure most of the AVEVA Historian. Importing a tag name database eliminates the need to manually configure I/O Server and tag definitions for both the InTouch HMI software and the historian. Information in the tag name database is automatically mapped to the appropriate tables within the historian Runtime database as part of the import process.

You can import tag name databases from multiple InTouch nodes, but you can only import one application from each node. Tag name databases from InTouch HMI software 6.0 or later can be imported.

To perform an import, you must have administrative permissions for the historian Runtime and Holding databases, (for example, be a member of the sysadmin or wwAdministrators roles or be in the Runtime and Holding db_owner roles).

However, to perform a delta import, you must be importing from an InTouch 7.1 or later tagname database. For more information, see Reimporting.

After you configure your system by importing I/O Server and tag definitions and then commit the changes, the historian acquires data for these tags. After the history data is being stored in the Runtime database, it can be manipulated using any SQL method for retrieving data supported by the historian. Functions specific to the historian, such as setting the resolution for a query, can be applied to the data, and the data can be retrieved from client applications, including InTouch HMI software.

InTouch user-defined tags are supported in the public namespace. For more information on user-defined tags, see the *InTouch HMI Data Management Guide*.

Before You Import

Using the tag importer functionality from within the Operations Control Management Console, you can import topic and other configuration data from one or more InTouch nodes. There are several factors that contribute to effectively importing tags from multiple InTouch nodes into the AVEVA Historian database.

- Determining import order
- Duplicate tags or addresses
- Import limitations for topic names
- Editing machine names
- Reimporting
- Importing information for DDE I/O Servers
- Holding database

The AllowOriginals system parameter needs to be set to 1 before importing .lgh original data.

Be sure to close InTouch WindowMaker before importing the Tagname.x file.

Note: Copying or moving an InTouch application in edited mode from one node to another node causes the application to be locked.

Although you can import from multiple InTouch nodes, only one application from each InTouch node may be

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

imported. That is, you cannot import an InTouch node with the same computer name as one you have already imported. However, you can import a Tagname.x from a repository and then edit the node name location. If you delete the application from the historian, you can import a different application from the same InTouch node.

When you delete an application from the historian, all tag information, annotations, snapshots, and summaries are deleted. Although the stored history data is not deleted from the history blocks, it is no longer accessible. If you perform a reimport, the existing history data already in the historian is accessible again.

Determining Import Order

If you are importing from more than one InTouch node, the following scenarios can exist:

- You have more than one InTouch node retrieving values from the same I/O tag, which is receiving values from the same point on a factory device.
- You have an InTouch node retrieving values from an I/O tag on another InTouch node, which is receiving its data value from a point on a factory device.
- All of your InTouch nodes are retrieving values from I/O tags on a dedicated InTouch node, which is receiving its data values from a point on a factory device. In this case, the dedicated InTouch node is set up to function as a "tag server."

When you import tagname dictionaries from multiple InTouch nodes, avoid importing duplicate I/O tags. To maximize the efficiency of importing data, first import the InTouch node that is functioning as the tag server or that contains the highest number of tags that have direct access to the data points from the factory floor devices.

As you import multiple nodes, always import a node with more tags having direct access, before importing a node having fewer tags with direct access.

Duplicate Tags or Tag Addresses

If the Tag Importer encounters a tag that already exists in the AVEVA Historian database, it can optionally add a string to the beginning or end of the tag name to make it unique within the historian. During the import, you can select:

- Whether or not you want a string to be added.
- The string to add.
- The placement of the string, either at the beginning or the end of the tag name.
- Whether to add a string to all tag names from that node, regardless if they are duplicates or not, or to only add a string when a duplicate tag is detected.

If, by adding a string, you create a duplicate tag during the importing process, the Tag Importer does not import that tag. For example, you choose to prefix all tags from a particular node with the letter "B," and you are importing a tag named "TestTag." However, a tag named "BTestTag" already exists in the historian. The Tag Importer does not import potential duplicate tags. To solve the problem of potential duplicate tags, change the names of these tags in InTouch HMI software to avoid the duplication and then try importing a second time.

If the Tag Importer encounters a duplicate address (consisting of the application, topic, and item) for one or

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

more tags, the information is still imported. However, if you are using DDE and have duplicate addresses, only the first tag (per tagname order) actually receives data. This is a DDE limitation.

Import Limitations for Topic Names

If the InTouch application you are importing contains topic names that are longer than 50 characters, the application is not imported.

Editing Computer Names

During an import, you are prompted to verify the InTouch computer name and the path to the InTouch application.

Normally, you should not change these default values. However, you need to change the InTouch computer name if you are importing multiple applications from the same InTouch computer. For example, perhaps you have a dedicated "application repository" computer on which you develop all of your InTouch applications. You then publish these applications to the actual production computers on which they will run. In this case, you can import all of the InTouch applications from the repository computer, but during the import process you need to change the InTouch computer name to the corresponding production computer name. The import wizard checks for duplicate InTouch nodes, but not until after you have the opportunity to rename the InTouch computer.

You cannot change the InTouch computer name and the path to the InTouch application during a reimport. However, you can manually edit this information in the InTouchNode table using SQL Server Query Analyzer (or any other query tool) before the reimport. When the reimport is performed, the new information will be used.

Reimporting

"Reimporting" means that all of the tag name dictionary information is imported, regardless of whether the information changed. No special configuration is required for reimporting an InTouch data dictionary. You may only perform a reimport procedure for the same InTouch node. The steps for performing a reimport procedure are basically the same as an initial import, with a few differences. For instructions on importing, see Importing or Reimporting a Dictionary.

Note: Back up the Runtime database before you reimport.

If you are reimporting and you do not choose to reimport all topics, only the tags for the topics you select are updated in the AVEVA Historian. All other topics remain unchanged.

You can also just reimport those tags that changed for a particular InTouch node since the last import. This is called "delta reimport," as opposed to the full reimport procedure . A delta reimport is usually faster than a full reimport procedure because only those tags that have changed since the last import are updated in the historian.

However, the delta reimport procedure does not provide the flexibility of the full reimport procedure. You cannot import a subset of the changed tags, nor can you edit the cyclic storage rate. For these capabilities, perform a full reimport procedure . For example, if you initially imported Topic A but not Topic B, a full reimport procedure is required to add Topic B to the historian database.

Any existing uniqueness settings and cyclic storage parameters (specified during the original import) will be

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

retained for the delta reimport procedure.

Note: Delta reimport is only supported for InTouch HMI software 7.1 and later.

Importing Information for DDE I/O Servers

If you are importing configuration information for remote I/O Servers that use DDE, you must first configure shares for these I/O Servers and then import the tagname database into AVEVA Historian. You do not need to configure shares for local DDE Servers.

When you import I/O Server configuration information from InTouch HMI software, the default protocol for all I/O Servers is set to SuiteLink. If you imported a DDE I/O Server(s), use the Operations Control Management Console to change the default protocol back to DDE for the server(s).

Holding Database

The Holding database temporarily stores topic and configuration data that has been imported from an InTouch node. When you import data from InTouch HMI software, the data is first mapped to table structures in the Holding database. Then, the data is moved into the Runtime database.

Importing or Reimporting a Dictionary

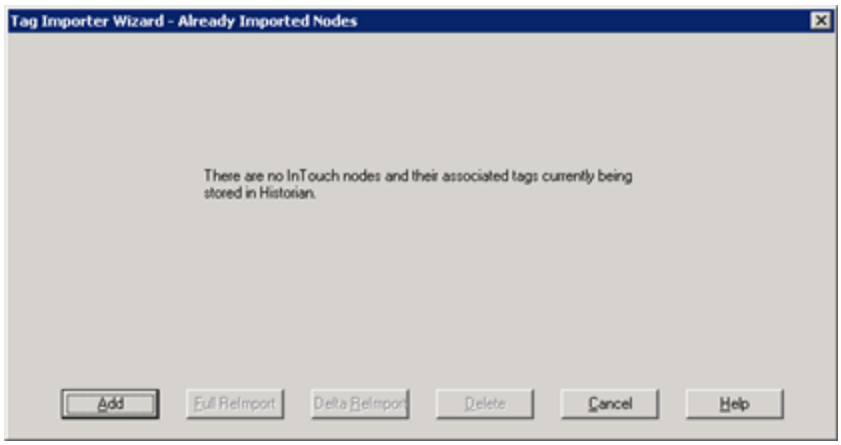
The Tag Importer allows you to select an InTouch tagname database (Tagname.x) and import all information from an InTouch application into the AVEVA Historian Runtime database.

To import from a Tagname.x file

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Click **Configuration Editor**.
3. Start the Tag Importer wizard by doing any of the following:
   - Right-click **Configuration Editor**, and then click **Import Tags**.
   - On the **Action** menu, click **Import Tags**.
   - Select the **Import Tags** button on the toolbar.
4. The **Welcome** dialog box appears. Click **Next** to start the import.

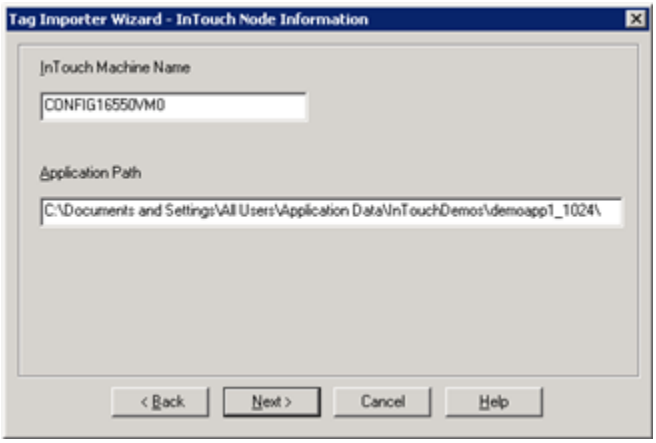
   The **Imported InTouch Nodes** dialog box appears. If there are no imported nodes currently in the AVEVA Historian, only the **Add**, **Cancel**, and **Help** buttons are available in the dialog box, and the dialog box only contains informational text.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Do any of the following:
   - To import an InTouch node and its associated tags, click **Add**. In the **Select Tagname.x** dialog box that appears, browse for the Tagname.x file (InTouch HMI software version 6.0 or later) that you want to import and then click **Open**. If you are importing for the first time, you are prompted to verify the import.
   - To reimport a node, select that node and click **Full ReImport**. For more information, see Reimporting.
   - To reimport only those tags that changed for a node, select the desired node and click **Delta ReImport.**
   - To delete a node and all its associated tags, select the desired node and click **Delete**. A warning box appears so that you can confirm the deletion.

     After a script runs, the **InTouch Node Information** dialog box appears.

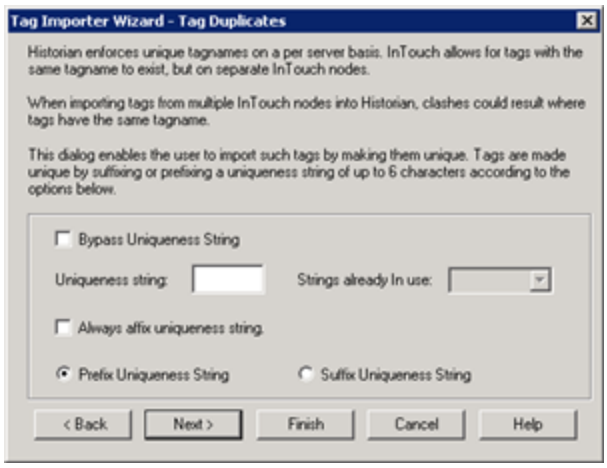


6. Verify the InTouch machine name and the path to the InTouch application.

   Options in this dialog box are unavailable during a reimport.

   The current InTouch machine name and the path to the current InTouch application are shown as defaults. For more information on changing the default machine name, Editing Computer Names.
7. Click **Next**. The **Tag Duplicates** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



8. Configure how the Tag Importer handles duplicate tags. Options in this dialog box are unavailable if you are reimporting.

   Bypass Uniqueness String

   Select to not append a uniqueness string to any duplicate tagnames. The Tag Importer does not import these duplicate tagnames.

   Uniqueness String

   The characters to add to the tagname to make it unique, if the Tag Importer determines that it is a duplicate. You can use up to 6 characters for the uniqueness string. You cannot leave the uniqueness string blank, and you cannot use a string that you used before. For information on allowable tagnames, see Tag Naming Conventions .

   Strings already in use

   The strings that are already appended to tagnames in the system.

   Always affix uniqueness string

   Used to append the uniqueness string to all imported tagnames from the selected node, regardless of whether they are duplicates. However, if affixing a string creates a duplicate tagname, the Tag Importer will not import that tag.

   Prefix Uniqueness String

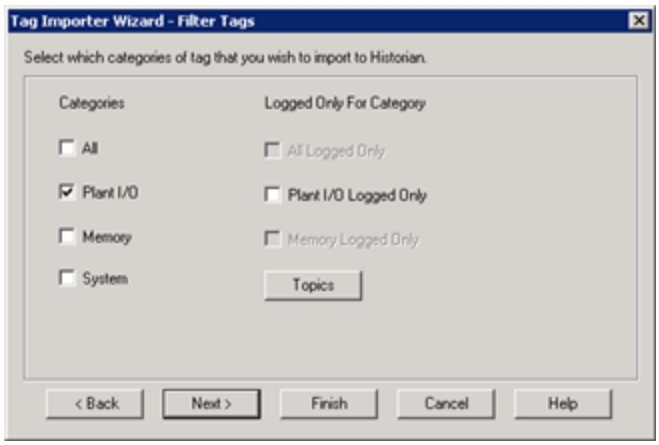
   Select to append the string to the beginning of the tagname.

   Suffix Uniqueness String

   Select to append the string to the end of the tagname.

9. Click **Next**. The **Filter Tags** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



10. Check the category or categories for the tags that you want to import:

    All

    Import all tag information from the data dictionary. If you select this option, all tag information from the other categories is automatically included.

    Plant I/O

    Import only tags receiving data from I/O Servers, including I/O discrete, I/O integer, I/O real, and I/O message tags.

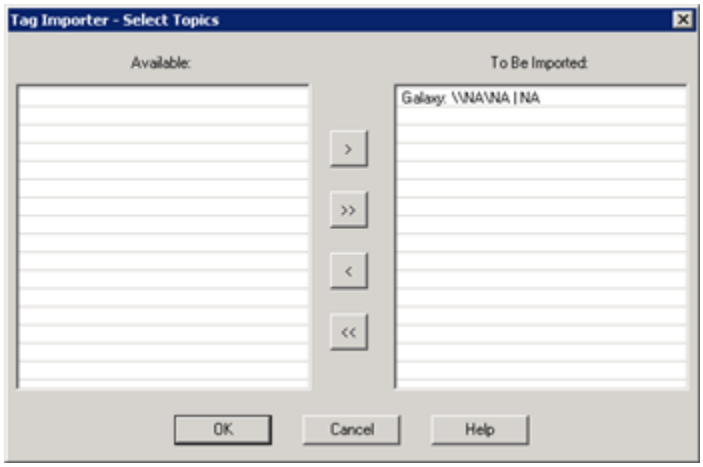
    Memory

    Import only InTouch memory tags, including memory discrete, memory integer, memory real, and memory message.

    System

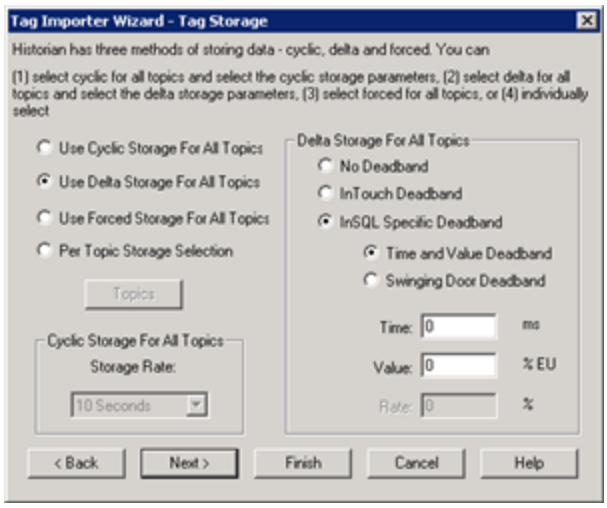
    Import only InTouch system tags ($<name>).

11. In the **Logged Only For Category** group, select whether to include only those tags that were configured in the InTouch HMI software to be logged, or all the tags for that category. These options are available if you selected **All**, **Plant I/O**, or **Memory** tags.
12. If you selected **All** tags or **Plant I/O** tags, you can individually specify which plant I/O topics you want to import. To do this, click **Topics**. The **Select Topics** dialog box appears.



13. Using the right and left arrow buttons, move the topics that you want to import into the **To Be Imported** window. Click **OK**.
14. Click **Next**. The **Tag Storage** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



For more information on storage, see Managing Data Storage.

15. To use cyclic storage, select **Use Cyclic Storage For All Topics**. **The Cyclic Storage For All Topics** area becomes available. In the **Storage Rate** list, select the desired rate. The cyclic storage rate is the time interval between consecutive stored values.
16. To use delta storage, select **Use Delta Storage For All Topics**. **The Delta Storage For All Topics** area becomes available. Select from the following options:

    No Deadband

    All changed values for a tag are stored.

    InTouch Deadband

    The time interval for storing a tag, as it was defined in the InTouch HMI software.

    AVEVA Historian Specific Deadband

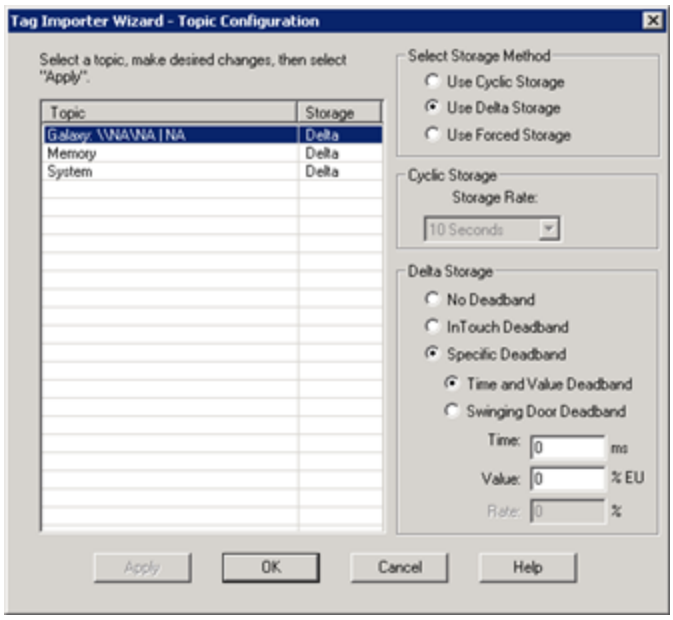
    Allows you to specify a time and value deadband or a swinging door deadband.

    A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

    A value deadband is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.

    A swinging door deadband is the percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.
17. To use forced storage, select **Use Forced Storage For All Topics**.
18. To assign the storage paradigm on a per-topic basis, select **Per Topic Storage Selection** and then click the **Topics** button. The **Topic Configuration** dialog box appears.

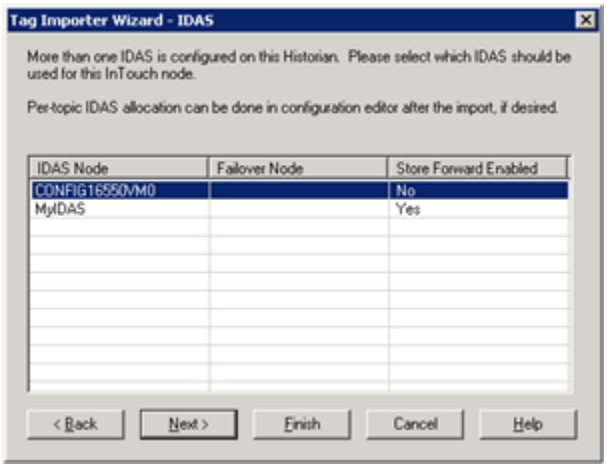
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



19. Configure the storage method for a topic. These options are similar to that of the **Tag Storage** dialog box. Click **Apply** to apply the new storage method.

    Note: If you do not click **Apply**, the storage rule options revert back to their previous settings.

20. Click **OK** to return to the **Tag Storage** dialog box.
21. Click **OK**. If you have more than one IDAS on the computer from which you are importing, the **IDAS** dialog box appears.

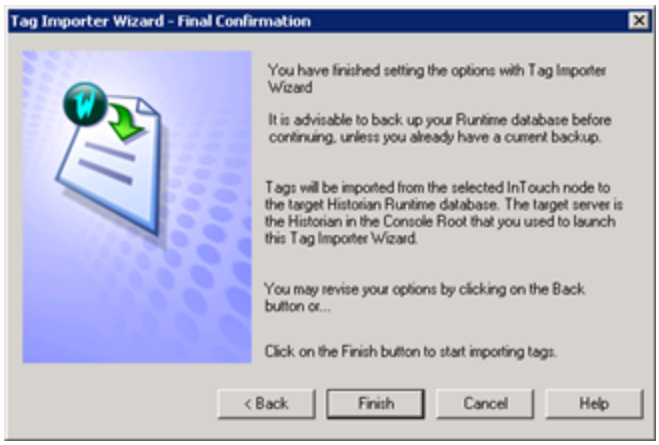


22. Select the IDAS that supplies the data values for the InTouch node.

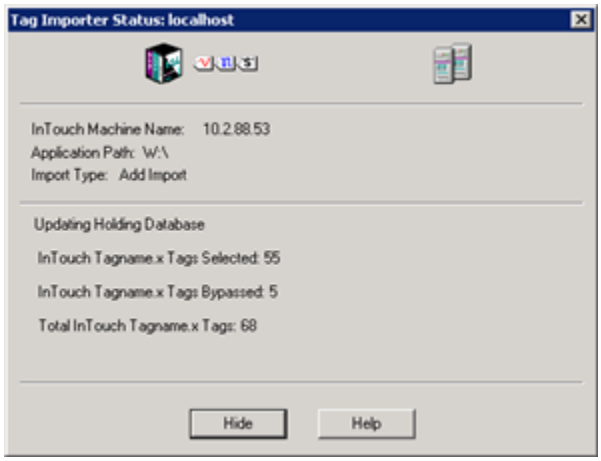
    For information on IDASs, including failover and store-and-forward options, see About IDASs.

23. Click **Finish** to start the import options. The **Final Confirmation** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



24. Click **Finish**. The **Tag Importer Status** dialog box appears.



25. If you click **Hide**, the dialog box closes, and the import process continues. The dialog box reappears when the import is complete.

    Note: For a reimport, only the tags for the topics you select are updated in the historian. Topics from the previous import remain unchanged.

26. When the import is complete, click **OK**.
27. Commit the changes to the system.

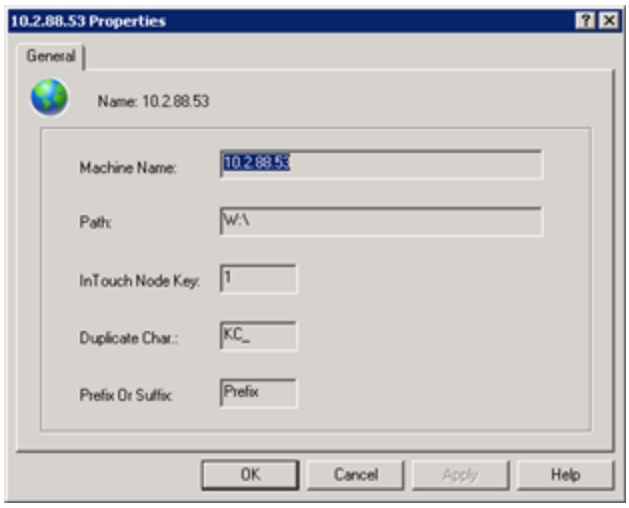
Viewing Properties for an Imported InTouch Node

In the Operations Control Management Console, you can view details for all imported InTouch nodes, as well as a list of tags associated with each node.

To view properties for an imported InTouch node

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Storage**.
3. Expand **Imported Nodes**.
4. Right-click an InTouch node, and then click **Properties**. The **InTouch Node Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. The read-only properties are as follows:

   Machine Name

   The name of the computer on which the InTouch application resides.

   Path

   The UNC path to the InTouch Tagname.X file.

   InTouch Node Key

   The unique numerical identifier for the named InTouch node.

   Duplicate Char

   The string that was added to a tag name as a prefix or suffix to make it unique.

   Prefix or Suffix

   Used to indicate whether unique tags were created by prefixing or suffixing the unique string for the node.
6. Click **OK**.

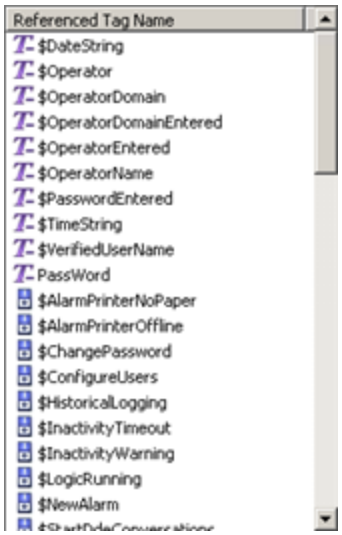
Viewing Tags Associated with an InTouch Node

In the **Public Groups** folder of the Operations Control Management Console tree, you can view a list of tags that have been imported from an InTouch node. You can also view the list of tags under the **System Configuration** folder, but with more details.

To view a list of tags in the public group

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **Public Groups**, and then expand **InTouch Nodes**.
3. Select the InTouch node for which you want to view a list of tags.
4. The tag list appears in the details pane.

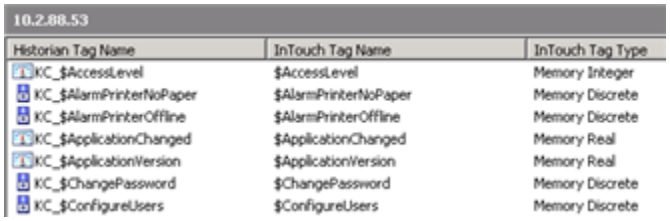
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. You can right-click any tag to access the **Properties** dialog box for that tag.

To view a list of tags under the system configuration

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Storage**.
3. Expand **Imported Nodes**.
4. Select the InTouch node for which you want to view a list of tags.
5. The tag list appears in the details pane.



10.2.88.53

Historian Tag Name	InTouch Tag Name	InTouch Tag Type
KC_$AccessLevel	$AccessLevel	Memory Integer
KC_$AlarmPrinterNoPaper	$AlarmPrinterNoPaper	Memory Discrete
KC_$AlarmPrinterOffline	$AlarmPrinterOffline	Memory Discrete
KC_$ApplicationChanged	$ApplicationChanged	Memory Real
KC_$ApplicationVersion	$ApplicationVersion	Memory Real
KC_$ChangePassword	$ChangePassword	Memory Discrete
KC_$ConfigureUsers	$ConfigureUsers	Memory Discrete

The columns shown are:

AVEVA Historian Tag Name

The unique name of the tag within the AVEVA Historian system.

InTouch Tag Name

The original tag name in an InTouch application. The tag name may be different than the AVEVA Historian tag name if a new name was generated to ensure uniqueness.

InTouch Tag Type

The type of tag in an InTouch application. For more information about InTouch tag types, see your InTouch documentation.

Importing or Exporting Tag Information

The Historian Database Export/Import Utility (aahDBDump.exe) is a standalone utility to export or import AVEVA Historian configuration information with a text file. Exporting and importing data are two independent operations.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The Historian Database Export/Import Utility is useful when you want to make bulk modifications to the configuration, instead of using the Configuration Editor to edit a single database entity at a time. You would simply export the text file, make the modifications, and then import the changes back to the historian. The utility is also useful for transferring configuration information from one historian to another.

You can also use the Export/Import Utility to export the Tag History table. This allows you to view different versions – and corresponding configurations – of the same tag.

You can export/import the configuration information for one or more of the following entities. For some of the entities, the utility supports additional filtering.

- Analog tags
- Discrete tags
- String tags
- Event tags
- IDASs
- I/O Servers
- Topics
- System parameters
- Storage locations
- Engineering units
- Messages
- Summary operations
- Snapshot tags
- Summary tags
- Replication servers
- Replication schedules
- Replication groups
- Replication tag entities
- Tag History
- Tag Extended Properties
- Saved content from Historian Client Web

The Historian Database Export/Import Utility does not export:

- InTouch node information. If you import tag definitions using the Tag Importer, and then export the database configuration, the node information is not included. If, after exporting, you rebuild the AVEVA Historian database, or want to import the database configuration into a different AVEVA Historian, you must first reimport the tag definitions for the InTouch application before you import the database configuration.
- Tags that have their current editor set to be AVEVA Application Server.

Starting with AVEVA Historian 2012 R2, the Tag table includes the AIHistory and ChannelStatus columns, which were not available in previous versions. If you import a configuration file that was exported using AVEVA Historian 2012 or earlier, the following values are used for these column settings:

- AIHistory = 1
- ChannelStatus = 1

The Historian Database Export/Import Utility requires a client connection to the SQL Server used by the historian to perform the export and import tasks. However, the historian does not need to be running.

The Historian Database Export/Import Utility does not maintain spaces in tag names during an export.

Encoding Formats for Configuration Exports

When you export configuration information, you can specify either Unicode or ASCII as the preferred encoding format for the text file. Select Unicode if you are exporting any information that uses double-byte characters (for

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

example, Japanese tag names).

When importing a text file, the Historian Database Export/Import Utility automatically detects the encoding of the file and converts all text to Unicode, if needed, before populating the AVEVA Historian database.

Configuration Exporter Error Log

The Historian Database Export/Import Utility keeps track of errors that occur during an import, on a line by line basis. When an error is encountered, you are prompted to stop the import or continue and process as much of the import as possible.

The utility logs progress and any errors encountered during the import to an error log file, named aahDBDumpLog.Txt. The error log file resides in the same folder as the utility executable and can be viewed with any program capable of reading text files, such as Notepad or Microsoft Excel. The error log file contains the:

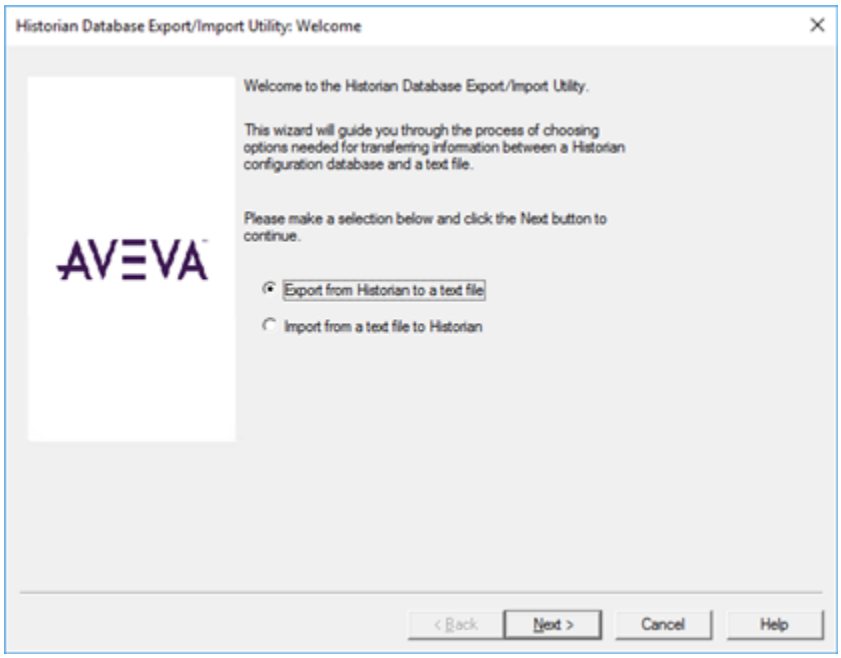
- Date and time.
- Name of the input file.
- Line numbers of the input file where errors occurred.
- SQL Server error messages reported when processing that line.

Each subsequent export or import operation appends to the log file. You should periodically delete older records to prevent the log file from becoming too large.

Exporting a Configuration

To export configuration information

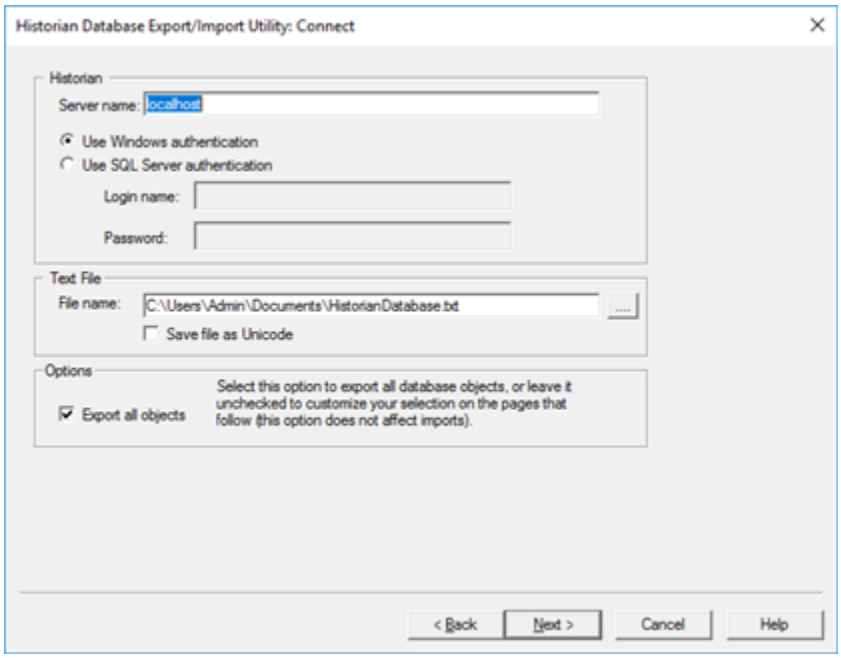
1. From the Windows **Start** menu, expand **AVEVA Historian**, and then click **Configuration Export and Import**. The **Historian Database Export/Import Utility Wizard** displays.



2. Select **Export from Historian to a text file**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Click **Next**. The **Connect** dialog displays.



4. In the **Server name** box, type the name of the AVEVA Historian for which you want to export configuration information.
5. Provide a login for the historian.

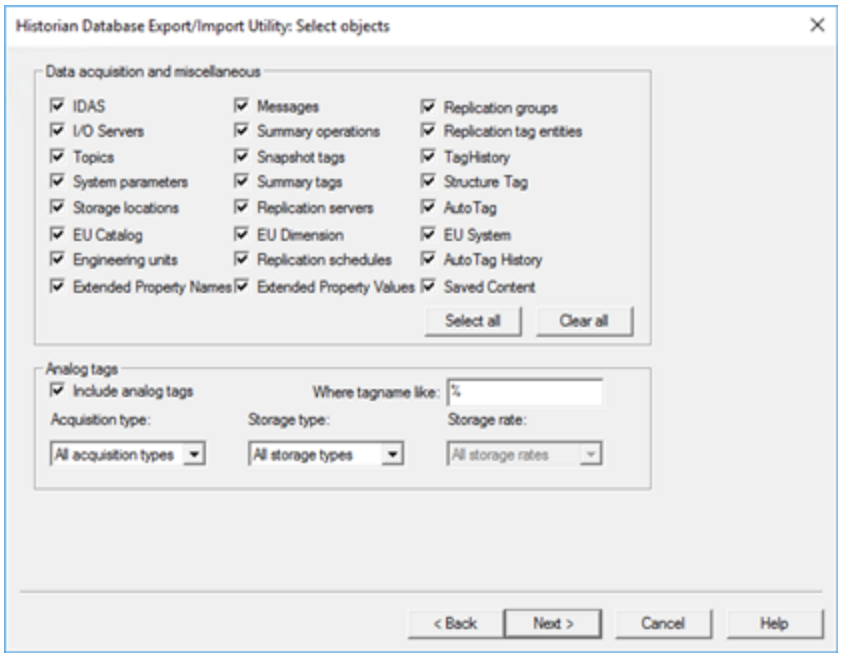
   Select **Use Windows authentication** to use your Windows logon account to connect to the AVEVA Historian.

   Select **Use SQL Server authentication** to use a SQL Server login. Enter a valid SQL Server username and password.

   Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.

6. In the **File name** box, type the path for the text file to export, or select ... and use the file browser to select the file.
7. If you are exporting and want to encode the data as Unicode, select the **Save file as Unicode** check box. For more information, see Encoding Formats for Configuration Exports.
8. To export configuration information for all database entities (for example, tags, engineering units, summary operations, and so on), select **Export all objects**. Skip to Step 17.
9. Click **Next**. (If you are exporting a file, and the file already exists at the location, you will be prompted to overwrite it.) The **Select Objects** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



10. In the **Data acquisition and miscellaneous** group, select one or more groups of definitions to export.

    IDAS

    An AVEVA Historian Data Acquisition Service (IDAS) is a software application that accepts data values coming from one or more I/O Servers and forwards them to a historian. For more information, see About IDASs.

    I/O servers

    An I/O Server is an application that provides data to a client over a network.

    Topics

    A topic is an application-specific subgroup of data elements. For more information, see I/O Server Addressing.

    System parameters

    A system parameter is a numeric or string value used for system configuration. System parameters are stored in the System Parameter table in the AVEVA Historian database. For more information, see SystemParameter in the *Historian Database Reference*.

    Storage locations

    The storage location is the directory in which historical data files are stored. Storage locations are stored in the StorageLocation table in the AVEVA Historian database. For more information, see StorageLocation in the *Historian Database Reference*.

    Engineering units

    An engineering unit is the unit of measure for an analog tag. For example, RPMs, milliseconds, degrees.

    Extended Property Names

    The names of the extended properties associated with a tag. For example, Alias, Location.

    Messages

    Messages are the string values associated with the TRUE (ON) or FALSE (OFF) states of a discrete value.

    Snapshot tags

    Tags that are defined to have value snapshots saved by the system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Summary operations

Aggregation calculations that are used to create summary values. See About Summary Replication.

Summary tags

Tag summaries.

Replication servers

A list of the replication servers that are configured for this instance of the historian. For information on adding and maintaining replication groups, see Managing and Configuring Replication.

Replication schedules

A list of the replication schedules that are configured for this instance of the historian.

Extended Property Values

The current stored values of the extended properties associated with a tag.

Replication groups

A list of the replication groups that are configured for this instance of the historian.

Replication tag entities

A list of the replication tags that are configured for this instance of the historian. This is the replication configuration for individual tags (simple or summary).

TagHistory

A list of different versions and corresponding configurations, including the unique Tag ID for each version, of the same tag.

Structure Tag

Tags from the StructureType and StructureAttribute tables.

AutoTag

Tags generated by Historian's auto-summary functionality.

AutoTag History

Tag history.

Saved Content

User-created dashboards and charts from Historian Client Web.

Note: If the DestinationTagId is empty, a new ID will be generated. If you copy a row to create a new tag entity, either leave the column empty or specify a unique ID.

---

11. To export analog tag definitions, select **Include Analog Tags**. System tags are not included.
12. In the **Where tagname like** box, type a string value in order to filter the tags by name. To include all tagnames, leave this option blank or use the wildcard symbol (%). The exporter recognizes all SQL Server wildcard characters. For example, to select all analog tags that have names starting with "MyTag", type "MyTag%".
13. In the **Acquisition type** list, select the filter for the source of the tag values.

    All acquisition types

    No filter set. The export file includes all tag definitions (that is, I/O tags, MDAS or HCAL tags, and tags for which values are not acquired).

    IOServer only

    Select to only include tag definitions that specify an I/O Server as the data source.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Manual only

Select to only include tag definitions that specify MDAS, HCAL, or manual data acquisition as the data source. For example, values from MDAS, HCAL, or Transact-SQL statements.

For more information on acquisition, see Configuring Data Acquisition.

14. In the **Storage type** list, select the filter for the storage type. The storage type determines how often the value for an analog tag is stored. An analog value can be stored either by time interval (cyclic) or when the value changes (delta).

    All storage types

    Specifies no filter. Cyclic, delta, and unstored tags are selected for export.

    Cyclic only

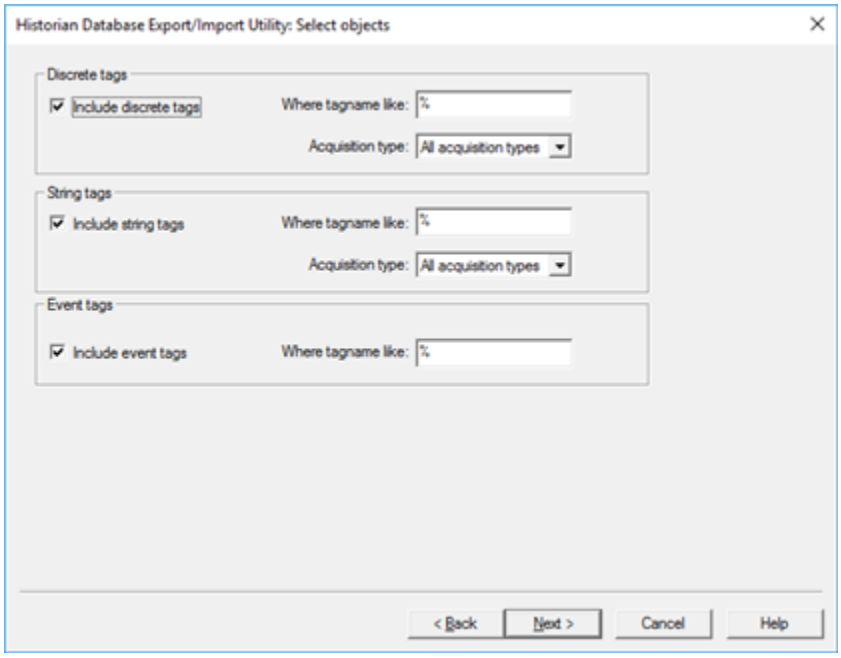
    Only include tag definitions that specify cyclic storage. If you select this option, you can set an additional filter on the storage rate in the **Storage rate** list. Otherwise, click **All storage rates**.

    Delta only

    Only include tag definitions that specify delta storage.

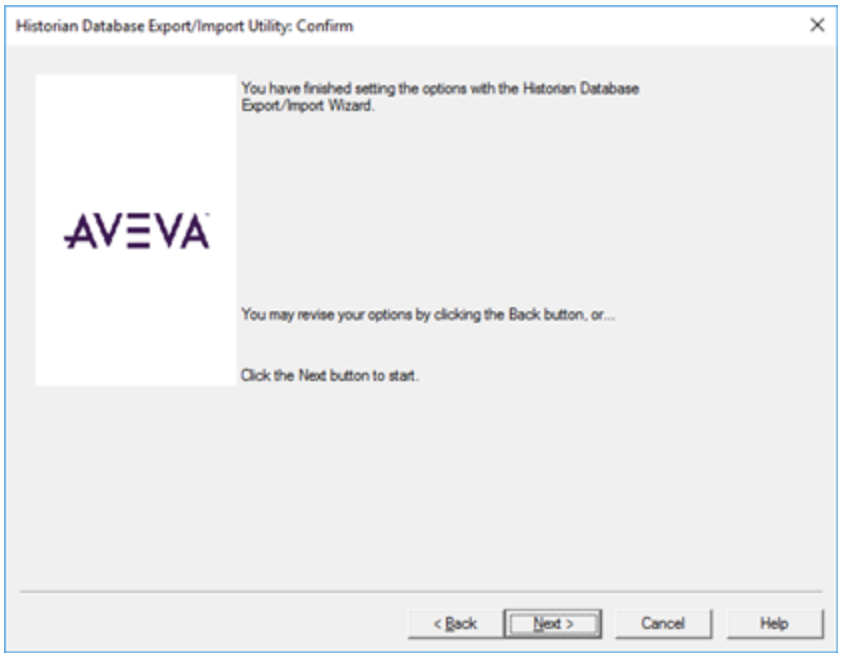
    For more information on storage types and rates, see Storage modes in the *AVEVA Historian Concepts Guide*.

15. Click **Next**. The **Select Objects** dialog displays.

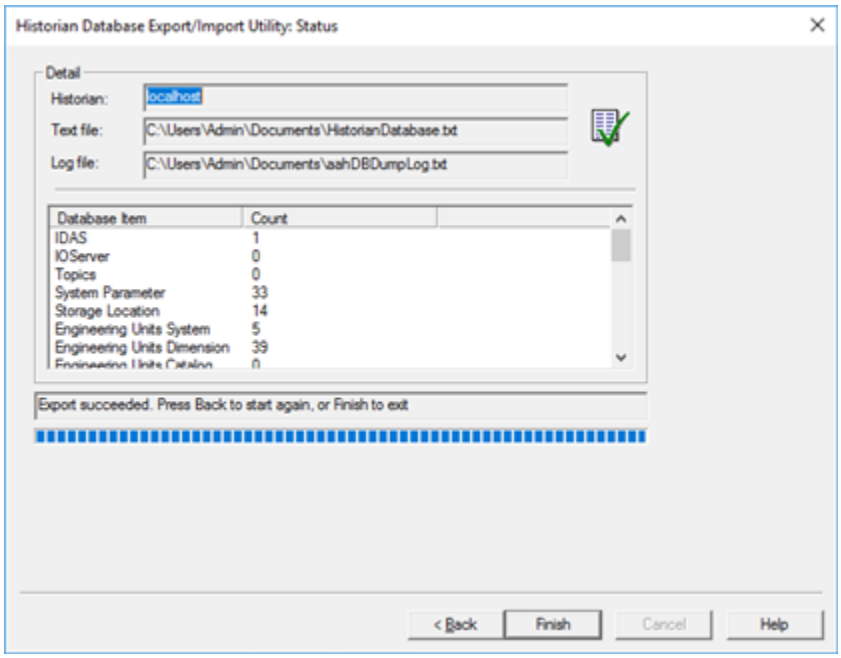


16. Configure the filters for discrete, string, and event tag definitions. System tags are not included. These options are the same as for analog tags.
17. Click **Next**. The **Confirm** dialog displays.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



18. Click **Next** to start the export. The **Status** dialog box appears, showing the results of the export. The number of objects exported is reported.



19. Click **Finish** to exit the wizard.

Importing a Configuration

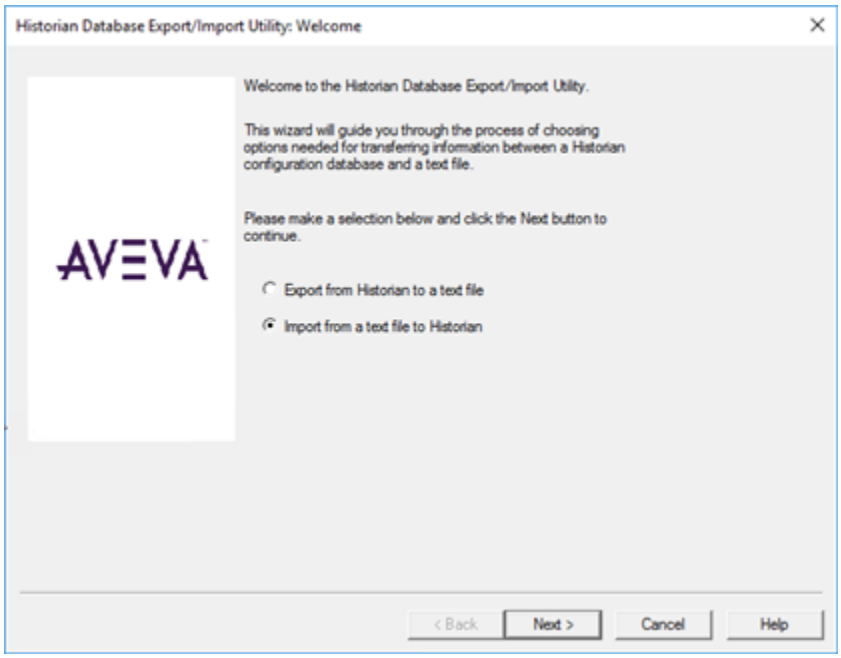
Important: The Historian Database Export/Import Utility offers considerable flexibility for modifying the contents of the Runtime database. However, after an import is complete, there is no rollback or "undo" capability. It is highly recommended that you make a backup of the Runtime database before performing an import.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

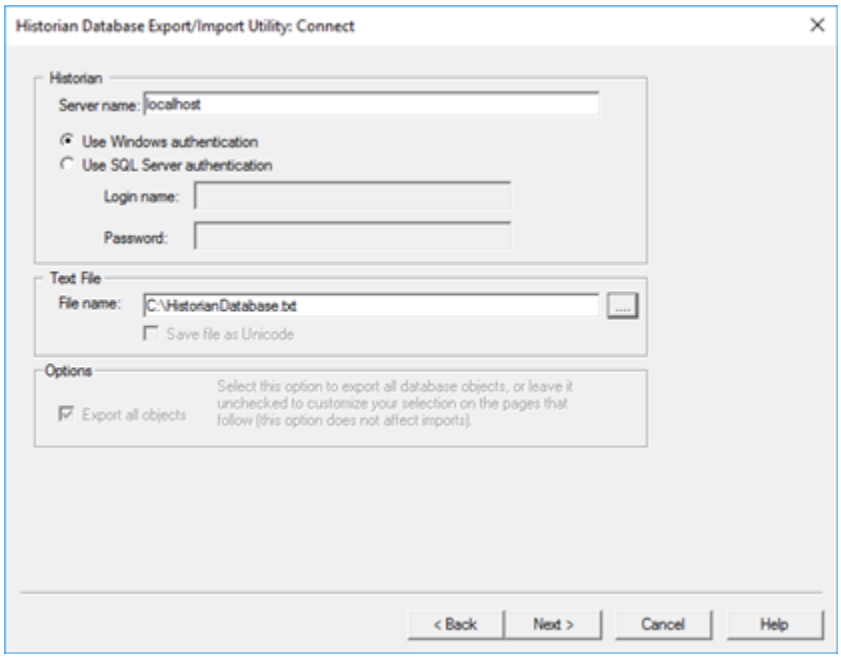
To import configuration information

Note: If importing a database file from Historian SP2023 R2 or earlier with saved content or dashboards, an error will occur. Update the Historian from which the file is being exported to SP2023 R2 P01 or later.

1. From the Windows **Start** menu, expand **AVEVA Historian**, and then click **Configuration Export and Import**. The **Historian Database Export/Import Utility Wizard** displays.



2. Select **Import from a text file to Historian**.
3. Click **Next**. The **Connect** dialog displays.

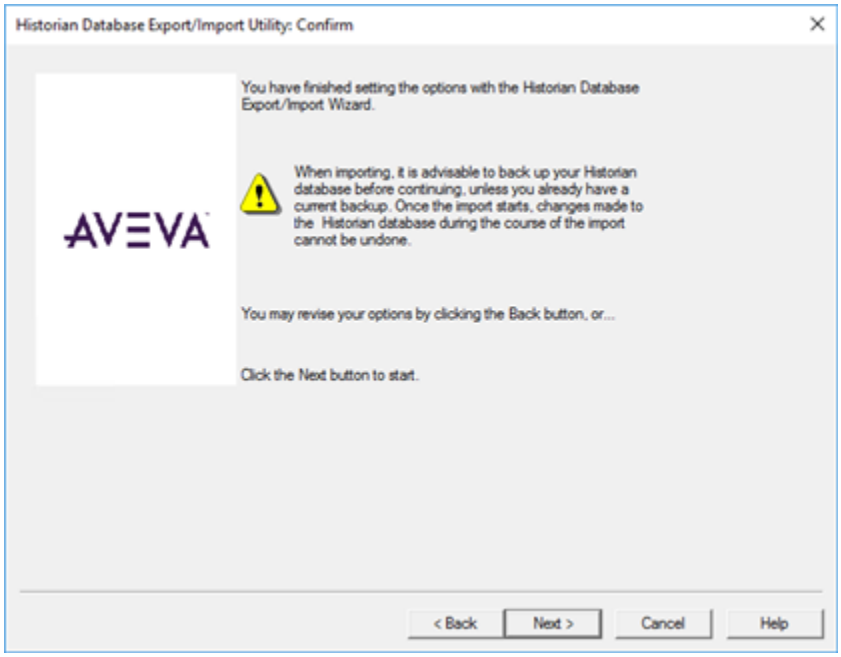


4. In the **Server name** box, type the node name of the computer hosting AVEVA Historian for which you want to import configuration information.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. Provide a login for the historian.
   - Select **Use Windows authentication** to use your Windows logon account to connect to the AVEVA Historian.
   - Select **Use SQL Server authentication** to use a SQL Server login. Enter a valid SQL Server username and password.

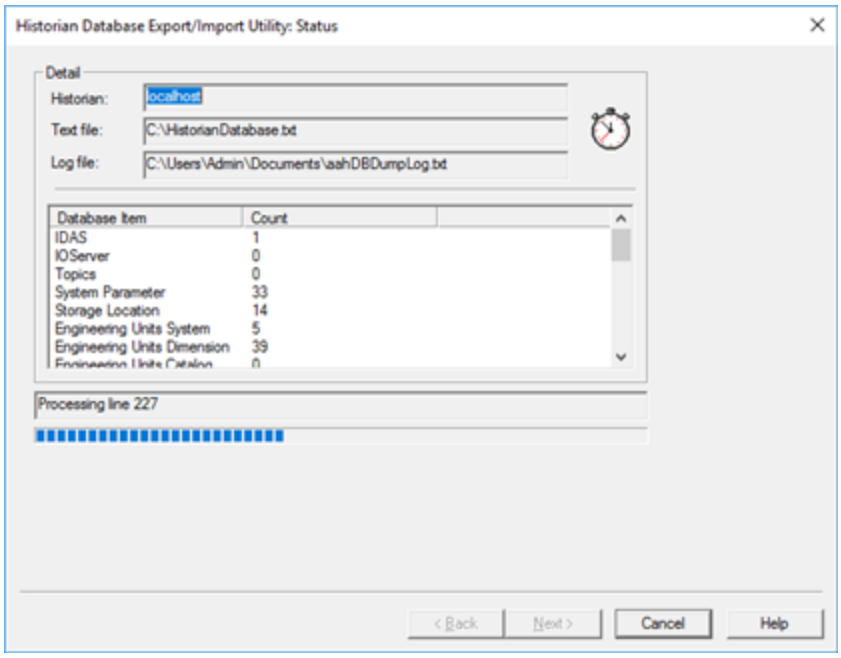
   Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.
6. In the **File name** box, type the path for the text file to import, or click the ellipsis button to browse to the location.
7. Click **Next**. The **Confirm** dialog box appears.



8. Click **Next** to start the import.

   Notes: If you are importing a text file that includes one or more delete mode indicators, the utility prompts you to verify each entity to delete, unless you select to turn off subsequent delete warnings. If you import a text file that includes IDAS configuration information, existing remote IDAS connections may fail. See Troubleshooting IDAS Connections for more information.
9. The **Status** dialog box displays, showing the results of the import. The number of objects imported is reported.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



10. Click **Finish** to exit the wizard.

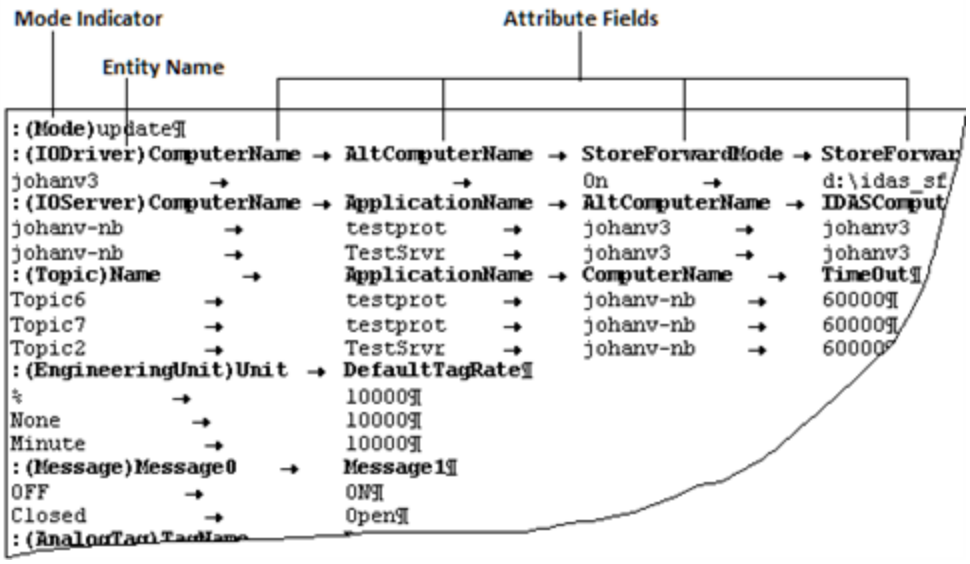
Editing the Configuration Text File

By editing a configuration text file, you can insert new objects into the database, modify existing objects, and delete existing objects. For example, you can add 10 new engineering units simply by adding 10 lines under the Engineering Unit entity line and then importing the configuration file into the AVEVA Historian. You can also ignore any changes to existing objects by skipping portions of the text file when importing.

Important: The order in which entities appear in the text file is important to ensure successful importing of the file. For example, if an analog tag is defined in the file, and the tag requires a new engineering unit, the new engineering unit should appear in the text file before the analog tag. The Historian Database Export/Import Utility scans the file once from top to bottom and makes no attempt at resolving ordering conflicts. As a general rule, the following order of entities in the text file should be maintained: IDAS, IOServer, Topics, System Parameter, Storage Location, EngineeringUnits, Messages, AnalogTags, DiscreteTags, StringTags, EventTags, SnapshotTags, SummaryOperations, SummaryTags, ReplicationSchedules, Replication Servers, Replication Groups, Replication Tag Entities, TagExtendedPropertyNames, and TagExtendedPropertyValues.

The following is an example of a configuration text file. All values must be separated by a tab stop.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Mode Indicators

The mode indicator determines whether the data is inserted, updated, deleted, or ignored. Valid values for the mode indicator are:

Value	Description
update:	If the line being imported corresponds to an existing entity in the database, the entity is updated with the contents of the line in the file. If the entity does not exist in the database, it is inserted.
insert:	If the line being imported corresponds to an existing entity in the database, that entity is left unmodified in the database. Only non-existing database entities are added when this value is specified for the mode indicator.
delete:	If the line being imported corresponds to an existing entity in the database, that entity is deleted; otherwise, nothing is done.
ignore:	The line in the text file is essentially ignored. This is useful for skipping portions of the text file when importing.

The very first line in the text file must be a valid mode indicator; otherwise, the importer reports an error and stops importing the file. Mode indicators can appear anywhere in the file and remain effective until the next mode indicator or the end of the file is encountered.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Line Entries

The text file contains header lines to indicate the type of database object referenced by the actual data lines following the header line. The header line consists of an entity name (shown within parentheses) followed by a series of attribute identifiers. The entity name loosely corresponds to a table (or tables) in the database, while the attribute identifiers resemble the column names in the tables. Note, however, that there is no strict correspondence between database tables and header lines. For example, a line in the text file related to an analog tag contains all the fields necessary to populate the Tag, AnalogTag, and other tables.

In the text file, a header line is followed by one or more lines containing actual data for the entity shown in the header line. Any particular entity can be repeated in the text file as many times as needed. For example, you can have a set of new analog tags inserted at the beginning of the file, and another set of analog tags deleted later in the file.

When you add lines to the end of the export file, make sure that the last line in the file is terminated by a carriage return/line feed. You can do this by pressing the Enter key on your keyboard at end of the line.

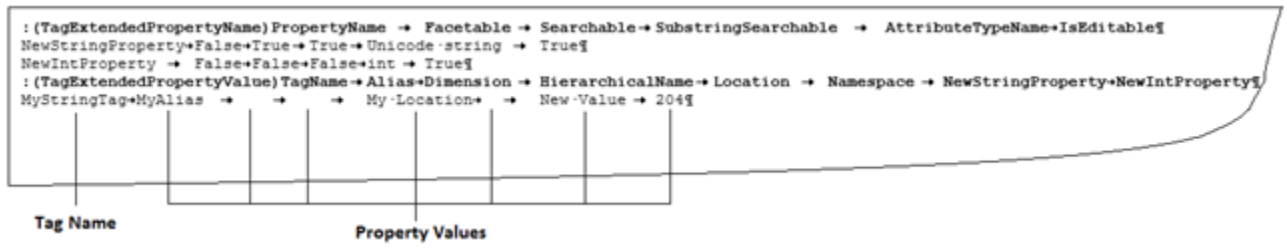
The value of the DefaultTagRate for an engineering unit must be one of the valid cyclic rates for analog tags. For more information, see Understanding Retrieval Options in the *AVEVA Historian Retrieval Guide*.

Note: The name "$local" appears in the export file, instead of the real computer name, for any object that has a computer name that refers to the local computer. When an import is performed, "$local" is translated into the name of the computer that is the target of the import.

Adding New Tag Extended Property Values

Line entries for tag extended property values use a different format, where each line contains all the extended property values for a specific tag.

The following example illustrates adding new tag extended properties, and setting extended property values for a single tag.



Two new extended properties, *NewStringProperty* and *NewIntProperty*, are defined by adding lines to the **TagExtendedPropertyName** section.

The **TagExtendedPropertyValue** header line begins with the *TagName*, followed by the system-defined extended properties *Alias*, *Dimension*, *HierarchicalName*, *Location*, and *Namespace*. The remainder of the line contains the property names defined in the **TagExtendedPropertyName** section, in this case *NewStringProperty* and *NewIntProperty*.

The remainder of the **TagExtendedPropertyValue** section consists of one line per tag, with each line containing all the extended property values for the specified tag, tab-delimited. In this example, the tag *MyStringTag* is assigned extended property values for *Alias*, *Location*, *NewStringProperty*, and *NewIntProperty*. The remaining properties are left blank.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Limitations

The maximum number of extended properties that can be defined is 10, including the 5 system-defined properties, *Alias*, *Dimension*, *HierarchicalName*, *Location*, and *Namespace*. This means that you can define up to 5 custom extended properties.

When you are using an SDK application, if more than 10 extended properties are defined, tag extended properties are not stored, and database warning messages are logged similar to the following example:

```
Main Metadata Server COM Exception caught, error = 38 (Database error)
```

When this occurs, the SDK application will continue retrying (and failing) until the application is closed.

Configuring Data Acquisition

About the Data Acquisition Subsystem

The purpose of the Data Acquisition subsystem is to accept and process incoming data that originates from data sources. One data source is an AVEVA-compatible I/O Server. An I/O Server is an application that provides factory data to a client by a specific protocol. An IDAS (Industrial Data Acquisition Service) is a component of the AVEVA Historian that accepts data values from an I/O Server and forwards them to the storage subsystem, which stores the data to disk.

You can also batch import existing history data into the system by means of a CSV file or by using the Historian Data Importer. For more information, see Importing, Inserting, or Updating History Data.

For information about monitoring data acquisition, see Monitoring Data Acquisition.

Data Acquisition Components

This table describes the components of the Data Acquisition subsystem. Many of the components run as Windows Services.

Component	Description
I/O Server (DAServer)	AVEVA-compatible software application that reads values from PLCs and other factory devices and forwards the real-time data to AVEVA applications.
Query Tools	Any database query tool capable of issuing Transact-SQL INSERT or UPDATE statements; for example, Microsoft SQL Server Query Analyzer.
Data Import Folder	Defined file folder to batch import tag values to the historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Component	Description
Historian Data Importer	Utility to import data from one or more CSV files or InTouch history files (.lgh). For more information, see Importing History Data.
Historian Client Access Layer (HCAL)	Process that can send non-I/O Server data to the historian to be historized. HCAL is used by Application Server 2012 R2 or later and custom client applications built with Historian SDK 2012 R2 or later.
Historian Client Access Point (HCAP)	Process that can accept non-I/O Server data and send it to the historian to be historized. HCAP is used by Application Server 2012 R2 or later and custom client applications built with Historian SDK 2012 R2 or later.
System Driver Service	Internal process that monitors the entire historian system and reports the status with a set of system tags. The system driver also sends data values to the Storage subsystem for the current date and time, as well as for predefined "heartbeat" tags, such as a discrete system pulse. For more information, see About System Driver and System Tags.

I/O Server Addressing

All AVEVA-compatible I/O Servers use DDE addressing, which includes the following distinct parts:

- **Computer name**
  This is the node name of the computer running I/O Server software.
- **Application name**
  This is the name of the application supplying data. The application name can include the name of the computer on which the application is running.
- **Topic name**
  A topic is an application-specific subgroup of data elements.
- **Item name**
  An item is a data value placeholder.

The format for the addressing is as follows:

*<computername>**<applicationname>**<topicname>*!*<itemname>*

The following table provides some examples of DDE addressing.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Address Information	I/O Server	InTouch	Microsoft Excel
application name	\\Computer1\Modbus	\\Computer1\VIEW	\\Computer1\Excel
topic name	ModMachine5	Tag name	Spreadsheet1
item name	Status	ReactLevel	A1 (cell name)

For the AVEVA Historian to acquire data from an I/O Server, the I/O Server addressing information must be added to the overall system configuration. You can use the Operations Control Management Console to manually add I/O Server definitions to the system, or you can import I/O Server definitions from existing InTouch applications.

For more information about manually adding I/O Server definitions, see Configuring I/O Servers.

For more information on importing I/O Server definitions from InTouch HMI software, see Importing and Exporting Tag Configurations.

I/O Server Redundancy

You can edit an I/O Server definition to include a "failover" I/O Server. This alternate I/O Server can be installed on the same computer as the primary IDAS or on another computer. If the network connection between the primary I/O Server and the IDAS fails, the IDAS automatically switches to the alternate I/O Server, provided that the alternate I/O Server is running. The switch may take some short period of time, and some data points may be lost during the transition.

Redirecting I/O Servers to InTouch HMI Software

When you redirect an I/O Server to InTouch HMI software, you are specifying to acquire tag values from a particular InTouch node that is using an I/O Server, instead of acquiring them directly from the I/O Server. This feature is useful when you need to reduce the loading on the I/O Server, or if the InTouch node is more accessible on the network.

When you redirect the I/O Server, the computer name and I/O Server type will reflect the InTouch node as the I/O Server from which data is acquired. For example, suppose you were using the a Modicon Modbus I/O Server on computer "I23238." The application name for the I/O Server address appeared as \\I23238\modbus. If you redirect this I/O Server to the InTouch node "InTouchNode1," then the address will be modified to reflect \\InTouchNode1\view.

Time Synchronization for Data Acquisition

All I/O Servers that support the SuiteLink protocol add a timestamp and quality stamp to plant data as the data is acquired.

It is important to understand how synchronization is handled between the timestamps for I/O Server, the computer clock for the IDAS(s), and the computer clock for the AVEVA Historian(s).

Note: This historian-controlled time synchronization method is not recommended on slow networks. If you are running AVEVA Historian on a slow network, please use a tool suited for your network configuration to synchronize your clocks.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

How time synchronization works

1. If you have multiple historians on your network, you should synchronize all computer clocks to a single master time server using standard Windows functionality.
2. Periodically, a historian automatically synchronizes the computer clock of any remote IDASs to its own computer clock. The IDAS synchronization is enabled by means of the *TimeSyncIODrivers* system parameter.
3. Every hour, an IDAS automatically synchronizes the *timestamping mechanism* of any associated I/O Servers with its own computer clock. This does not actually change the system clocks of any I/O Server computers. Instead, the difference in the system clocks on the two computers (I/O server and Historian) are determined, and a bias is calculated that is then applied to all values from that I/O server computer. For example, if the historian clock is seven seconds ahead of the I/O Server computer's clock, SuiteLink adds seven seconds to every timestamp from the I/O Server. If a topic is disconnected/reconnected due to a topic time-out or other communications failure, the I/O Server timestamping is not updated until the time synchronization interval has passed. You can change the frequency of the synchronization using the *SuiteLinkTimeSyncInterval* system parameter.

   The SuiteLink protocol also does some time adjustments to keep timestamps consistent across nodes. SuiteLink bases this adjustment on the time difference detected at startup and each hour. For example, NodeA and NodeB have a time difference of 17 seconds. The I/O Server is on NodeA, and the IDAS is on NodeB (either local to the historian or a remote IDAS for a historian on another NodeC). When the I/O Server on NodeA timestamps a value at 12:00:00.000, it is transmitted to NodeB with an adjusted timestamp of 12:00:17.000. If the historian is configured to timestamp at the source, this value is stored with a timestamp of 12:00:17.000. If, instead, the historian is configured to timestamp at the server, and there is a two-second communications latency, then the value is stored with a timestamp of 12:00:19.000.

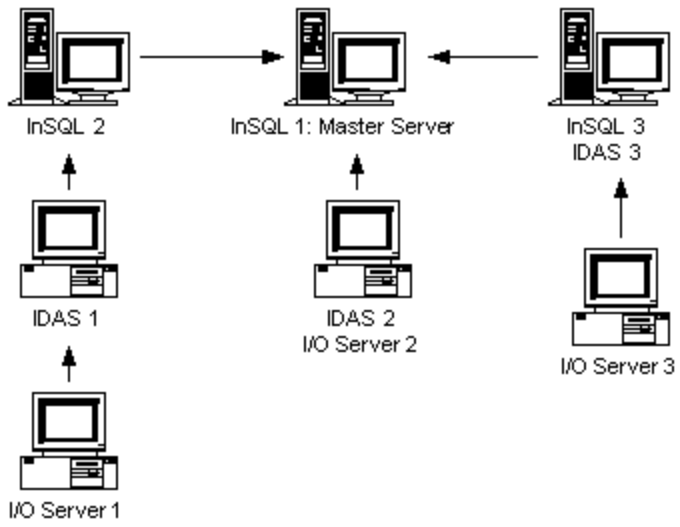
   For normal operations on systems with synchronized clocks, there is no adjustment made by SuiteLink and everything operates as expected. However, when either the systems are out of sync, or even were out of sync when SuiteLink communications between the nodes started, the timestamps will be adjusted. Because of the way SuiteLink adjusts timestamps, it is easy to produce misleading results if system tests involve adjusting system clocks on the systems, because SuiteLink does not immediately update its time skew.

   Note: Time synchronization does not apply to I/O Servers that use DDE because these servers do not perform timestamping. The time of the IDAS computer is always used for data coming from DDE from I/O Servers.

For more information on setting system parameters, see Editing System Parameters.

The following diagram shows an example of how computers can be synchronized to a single time:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

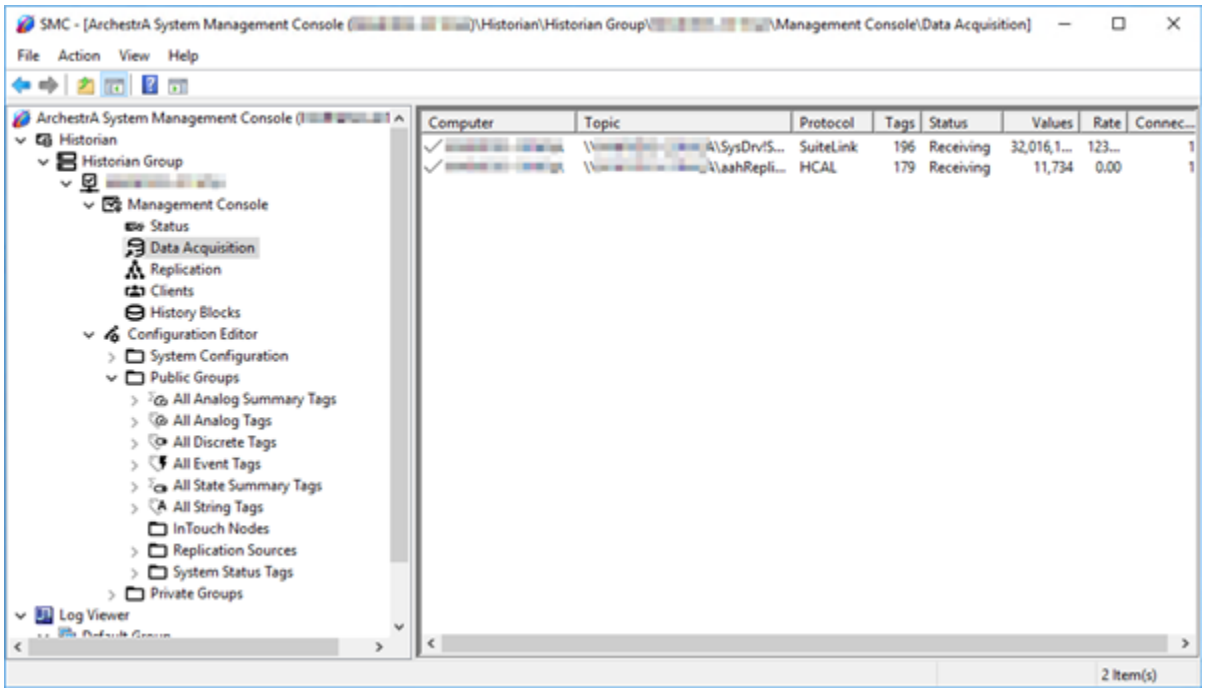


For an MDAS-enabled or HCAL-enabled client application, you can use the **net time** command (for the Windows operating system) to synchronize the client computer's clock to your master historian.

Viewing Data Acquisition Information

To view data acquisition information

1. Open the Operations Control Management Console.
2. In the console tree area, expand a server group and then expand a server.
3. Expand **Configuration Editor**, expand **Management Console**, and then click **Data Acquisition**.



Note that if you have configured an IDAS, you will see both SuiteLink and HCAL connections used by the IDAS.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Connections using the HCAL (Classic) protocol

Connections use **the HCAL (Classic)** protocol for the following reasons:

- The connecting system uses older software.
- The connecting system is on current software, but the connection uses a legacy port. Update the connection to point to the port for current HCAL protocol (32565 by default).

Configuring IDASs

Each AVEVA Historian server must have at least one IDAS (Industrial Data Acquisition Service) configured.

You can use the Operations Control Management Console to configure IDASs.

About IDASs

An IDAS (Industrial Data Acquisition Service) accepts data from one or more I/O Servers or other data source and sends it to AVEVA Historian for storage. If the connection to the historian is not available, IDAS caches the data locally and forwards it later when the server is back online.

Note: IDAS configuration information is stored in the _IODriver table in the Runtime database.

When you add an I/O Server definition to the historian, a topic object is created in the associated IDAS. A separate topic object exists for each unique combination of I/O Server computer, application, and topic. Each topic object maintains its own state: idle, connecting, connected, disconnecting, disconnected, overloaded, or receiving. Also, each topic object is assigned a data time-out value based on your assessment of how often data changes for that particular topic.

An IDAS can accept data from one or more I/O Servers but sends data only to a single historian.

An IDAS can run on the same physical computer as the historian, or on a remote computer. Use the `Ping` command to check the availability of the remote IDAS or historian computers.

IDAS seamlessly handles data values, irrespective of their time. For each data point acquired by IDAS, the timestamp, value, and quality are historized in accordance with the storage rules for the tag to which the data value belongs.

For information on configuring an IDAS, see Configuring Data Acquisition.

IDAS Configuration

During normal operation, when the historian is started, it configures an IDAS by sending it information about the tags (including their data sources) for which the IDAS is to acquire data. When the historian Storage subsystem is ready to accept data, IDAS automatically connects to its data sources, starts acquiring data, and sends the data to the historian Storage subsystem for historization.

The primary purpose for IDAS configuration files is to minimize network traffic and provide information for IDASs configured for autonomous startup. For more information on autonomous startup, see IDAS Autonomous Startup.

The IDAS saves configuration information to a file on the local hard drive in the following folder of the IDAS computer: `ProgramData\ArchestrA\Historian\IDAS\Configurations`.

The IDAS configuration file is named as follows:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

`idascfg_SERVERNAME_IDASKEY.dat`

where:

- *SERVERNAME* is the NetBIOS name of the historian computer
- *IDASKEY* is the value of the IODriverKey column for the IDAS in the Runtime database

You can change the IDAS configuration from the Operations Control Management Console. The historian dynamically reconfigures itself. If the IDAS is on a remote computer, the historian sends the updated configuration information to the IDAS. The IDAS reconfigures itself and updates the local configuration file. The IDAS continuously acquires and sends data during the reconfiguration process. The historian saves its copy of the updated IDAS configuration file in the following folder of the historian computer: `ProgramData\ArchestrA\Historian\Configuration\IDAS Configurations`.

After a successfully configuring IDAS, a copy of the IDAS configuration file is stored on the historian computer. The IDAS configuration file stored on the IDAS computer is identical.

Important: IDAS configuration files have a proprietary binary format. Do not modify these files.

If there is more than one autonomous configuration file on the IDAS computer (for example, if you deleted an IDAS on a node while it was disconnected and then added one again), only the newest file is used. A warning is logged on the IDAS computer. For more information on autonomous startup, see IDAS Autonomous Startup.

IDAS Data Processing

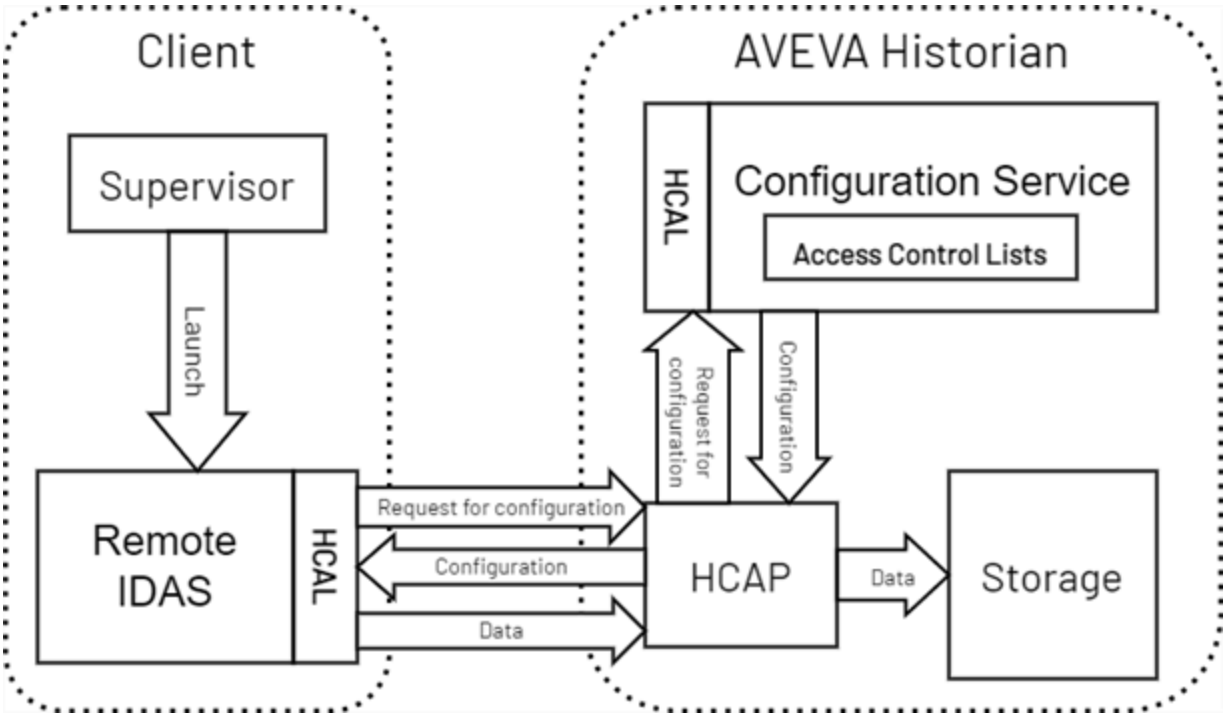
After receiving data from an I/O server, IDAS converts data values into storage data types, depending on the type of the tag associated with the value. For example, if the data value is associated with a floating point analog tag, the incoming value is converted to a floating point value before transmission to the Storage subsystem. However, no timestamp conversion is applied because both the I/O Servers and the AVEVA Historian Storage subsystem base time on Universal Time Coordinated (UTC).

In Historian 2017 and later versions, IDAS uses the HCAL infrastructure for data delivery to the Historian server. This means that the storage rules such as DELTA or CYCLIC are applied on the client side to reduce the network traffic.

IDAS Security and Firewalls

- Remote IDAS uses two-way communication:The remote IDAS requests and receives configuration information from the Historian server.
- The remote IDAS sends data collected from device interfaces to the Historian server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



AVEVA Historian provides two ways to authorize access:

- **Integrated security.** IDAS computers in the same domain as the historian can be configured with integrated security. Using this model, all users and computers that access historian data are assigned membership to one of the following user groups:
  - Administrators (aaAdministrators)
  - Power Users (aaPowerUsers)
  - Replication Users (aaReplicationUsers)
  - Users (aaUsers)
- **Workgroup security.** IDAS computers outside of the historian's domain can use username and password as security. This username and password must match a local user on the server node, and must be provided in the format `<server node hostname>/<user name>`.

  When the IDAS is configured with this type of security, an authentication token and is defined and forwarded to the remote IDAS computer. Each time the remote computer accesses the historian, it will use the token and the historian will use it to authenticate the remote computer before allowing access.

The remote IDAS must be able to communicate with the Historian server's HCAL TCP port (by default, port 32568 for Historian versions 2023 and earlier, or port 32565 for Historian versions 2023 R2 and later).

For remote IDAS versions 2020 R2 and earlier, the Historian must be able to communicate with the remote IDAS using its HCAL TCP port 32568.

For a Classic remote IDAS (from an AVEVA Historian version before 2017), requirements are different. A legacy remote IDAS supports only Windows integrated security. It requires consistent accounts on the Historian server and the remote IDAS:

- On the remote IDAS, this is configured using the ArchestrA Network User utility.
- On the Historian server, this is configured by setting the identity of the aahConfigSvc service from the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Windows Services Console. The Historian server must also be able to communicate with the Remote IDAS machine using TCP/UDP ports 135 through 139 and 445.

For more information on IDAS file sharing requirements, see IDAS Store-and-Forward Capability.

IDAS Error Logging

An IDAS logs all errors to the ArchestrA Logger Service. If the IDAS is installed on a remote computer, the ArchestrA Logger Service will also be installed on the remote computer. During normal operation, remote IDAS errors are logged to both the local logger and the logger on the AVEVA Historian computer.

If the network connection between the remote IDAS and the historian fails, no error messages are sent to the logger on the historian computer. Therefore, you should periodically use the Operations Control Management Console to check the log on the remote IDAS computer to ensure that no problems occurred. After the network connection is restored, error messages are not forwarded to historian computer.

IDAS Store-and-Forward Capability

IDAS includes "store-and-forward" capability, which protects against a temporary loss of data in the event that a remote IDAS cannot communicate with the AVEVA Historian.

Note: The store-and-forward option is not available if you have specified a failover IDAS.

If the remote IDAS cannot communicate with the historian, all data currently being processed can be stored (cached) locally on the computer running IDAS. This hard drive location is called the store-and-forward path and is configurable using the Operations Control Management Console.

Note: Be sure to specify a valid **Store Forward Path** path. If the path is not valid and accessible, the store-and-forward functionality will fail.

If the IDAS is unable to send the data to the historian, data is written to this path until the minimum threshold for the cache is reached, at which point no more data is stored. An error message is logged. Remote store-and-forward paths are not supported.

The following actions occur after the historian becomes available again:

- The historian verifies that the IDAS configuration information did not change while the IDAS was disconnected. The historian attempts to restore data transmission from the IDAS. The IDAS stops local data caching and resumes sending data acquired from its data sources to the historian.
- If historian detects a difference between its version of the IDAS configuration, and the IDAS version, it dynamically reconfigures the IDAS to synchronize configuration information. The IDAS applies the changes and updates its local IDAS configuration file. Then, the historian requests restoring data transmission from the IDAS.
- When the IDAS detects availability of the running historian, it sends the store-and-forward data to the historian at the same time it is sending real-time data.

After data from the store-and-forward cache is sent to the historian, the cache is deleted from the IDAS computer.

Enabling IDAS store-and-forward mode increases system resources used by the IDAS service because the Store-and-Forward subsystem must be initialized and then maintained in standby mode, ready to accept data.

If the historian computer has sufficient system resources, it is recommended to configure the local IDAS for store/forward as well. The local IDAS to continue store-and-forward data collection even if the other Historian

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

subsystems are stopped.

IDAS Redundancy

For each IDAS that you define for the system, you can specify a "failover" IDAS. If the AVEVA Historian stops receiving data from the primary IDAS, it automatically switches to the failover IDAS. The switch may take some short period of time, and some data may be lost during the transition.

Notes: Beginning with the AVEVA Historian 2023 release, IDAS redundancy is no longer supported. Any legacy IDAS already configured for failover can continue using the feature, but after connecting the IDAS to a Historian 2023 server, or upgrading it to version 2023, the failover configuration can no longer be modified.

You cannot specify a failover IDAS for an IDAS that has store-and-forward functionality enabled. These two features are mutually exclusive. Applications that require both failover and store-and-forward functionality should use a redundant Application Server with RedundantDIObjects.

IDAS Autonomous Startup

Normally, the AVEVA Historian Configuration Service starts an IDAS. However, a remote IDAS that is enabled for store-and-forward can be configured to start independently of the historian. Autonomous startup is useful when the historian is unavailable due to a network failure or when the historian is not running when the remote IDAS computer starts. Using autonomous startup, the IDAS starts caching store-and-forward data without waiting for a command from the historian.

For an IDAS to autonomously start, it must be configured to acquire data from at least one data source. During the configuration process, the IDAS must be connected to the historian to ensure that the configuration file is created on the local IDAS computer. An autonomous startup requires an existing local IDAS configuration file on the IDAS computer, so that it has all of the information it needs to begin acquiring data. For more information, see IDAS Configuration.

When an IDAS starts, it attempts to load the configuration information from the local configuration file. If it is able to do so, the IDAS uses that information to connect to its data sources and start acquiring data. As soon as the internal IDAS data buffers are full, the IDAS switches to store-and-forward mode and stores data to the local hard drive.

If there is more than one autonomous configuration file on the IDAS computer (for example, if you deleted an IDAS on a node while it was disconnected and then added one again), only the newest file is used. A warning is logged on the IDAS computer.

If the local configuration information cannot be loaded, the IDAS remains in an idle state until contacted by the historian. If the IDAS is not contacted by the historian within the default start time-out of 60 seconds, the IDAS shuts down. Note that the IDAS startup time-out is different than the time-out used by the IDAS during autonomous startup. Information on changing the default IDAS startup time-out is provided in a TechNote, which is available from technical support.

When the historian becomes available, data transmission from the IDAS will be restored. For more information, see IDAS Store-and-Forward Capability.

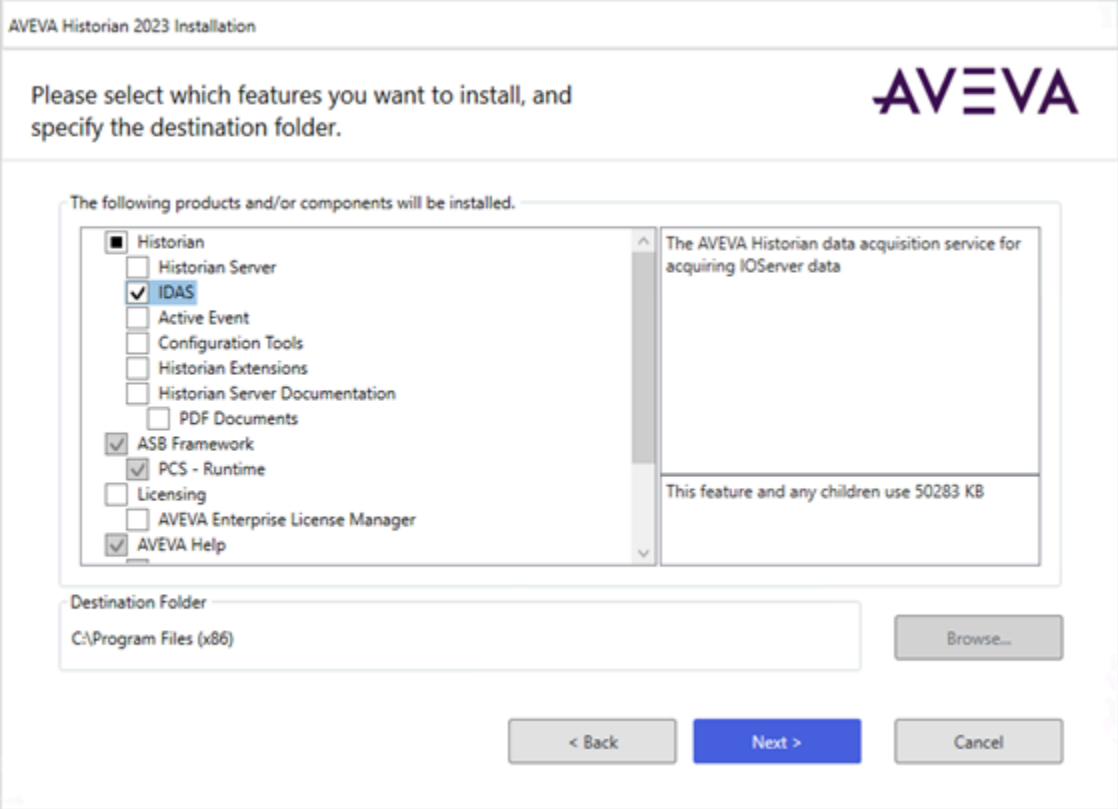
Configuring IDAS on a Remote Node

Configuring IDAS on a remote node requires the IDAS software to be installed on the node, without the Historian Server software.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Note: When the IDAS software is installed on the same node as a Historian Server, the IDAS is automatically configured to work with the Historian on the same node, and the IDAS configurator is not available.

You can install the IDAS software independently by running the AVEVA Historian installation, and selecting the IDAS component:



After the installation is complete, click **Configure** to launch the configurator and configure the IDAS node:

To configure IDAS on a remote node

1. Launch the configurator and select the **IDAS** node.
2. Enter the **Server Name** and **Port** for the Historian node this IDAS communicates with. For Historian versions 2023 R2 and later, the default value is 32565. For Historian versions 2023 and earlier, the default value is 32568.
3. Enter the **User Name** and **Password** of a Windows user account with access to the Historian node.
4. Select the authentication method:
   - **Authenticate using Single Sign-on:** Authenticate with the AVEVA Identity Manager. The IDAS system must be connected to the System Management Server for this option to be enabled. It must be the same System Management Server used by Historian, or SSO authentication will fail.
   - **Authenticate using User Name and Password:** Authenticate by entering a user name and password.
5. Select whether or not to accept untrusted certificates:
   - **Require a trusted certificate:** Require a trusted certificate for communication between the IDAS and Historian.

     The server's certificate must be trusted on the IDAS system. If both nodes are connected to the same System Management Server, certificate trust is managed by the System Management Server. If the IDAS system and Historian are not connected to the same System Management Server, when you test the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

connection, the configurator either prompts you to trust the server's certificate, or logs an error message if it cannot retrieve the certificate information.

- **Accept an untrusted certificate (NOT RECOMMENDED):** Allow an untrusted connection.

Note: Requiring a trusted certificate is strongly recommended. Untrusted connections should only be used in test environments.

6. Select **Test Connection** to verify the connection information. Any errors will be noted in the **Configuration Messages** section.
7. Once the connection is successful, click **Configure** to complete the IDAS configuration.

Troubleshooting IDAS Connections

This section will help you resolve some common issues you may encounter with IDAS connections.

Issue: The remote IDAS stops communicating with the Historian after IDAS configuration information is imported on the Historian node

After you import configuration information on a Historian node (see Importing a Configuration) existing remote IDAS nodes may be unable to connect to the Historian. This is caused by the existing connection token for the remote IDAS becoming invalid when the IDAS configuration is updated on the Historian node.

To resolve this issue, perform the following steps:

1. On the IDAS node, stop the **AVEVA Insight Supervisor** service (aahSupervisor.exe).
2. On the IDAS node, delete the existing ***.dat** files from **C:\ProgramData\ArchestrA\Historian\IDAS\Configurations**.
3. On the Historian node, launch the Operations Control Management Console (OCMC). Locate the IDAS node in the Configuration Editor section, right click the IDAS node, then select **Properties**.

   The **IDAS Properties** dialog displays.
4. Select the **Advanced** tab.
5. Click **Generate new connection token**, then click **OK** to close the dialog and apply the change.
6. Right-click the IDAS node and select **Commit Pending Changes...** if it is enabled.
7. On the IDAS node, start the **AVEVA Insight Supervisor** service.

   The IDAS restarts, retrieves the configuration information from the server, and connects to the Historian.

Issue: The remote IDAS fails to connect to the Historian and logs a RemoteCertificateNameMismatch error

When the remote IDAS attempts to connect to the Historian using a trusted connection, the connection may fail and log a message indicating a *RemoteCertificateNameMismatch* error has occurred. If the host name indicated in the error message matches the host name of the Historian node, then the remote IDAS node may be unable to resolve the Historian's host name via DNS.

To resolve this issue, ensure the remote IDAS node can correctly resolve the Historian node's host name. You can temporarily work around this issue by adding an entry to the hosts file on the remote IDAS node.

1. On the IDAS node, locate the **hosts** file in **C:\Windows\System32\drivers\etc** and open it in a text editor.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Add a line to the hosts file consisting of the Historian node's IP address, followed by the host name. For example, if your Historian node's IP address is 192.168.1.75, and the host name is HISTSRV01, add the following line to the hosts file on the IDAS node:

   ```
   192.168.1.75 HISTSRV01
   ```

3. Save the changes to the hosts file.

   The IDAS should now connect successfully to the Historian.

Important: After the DNS issue is fixed and the Historian node's host name can be correctly resolved via DNS, remove this entry from the hosts file to prevent future connection errors.

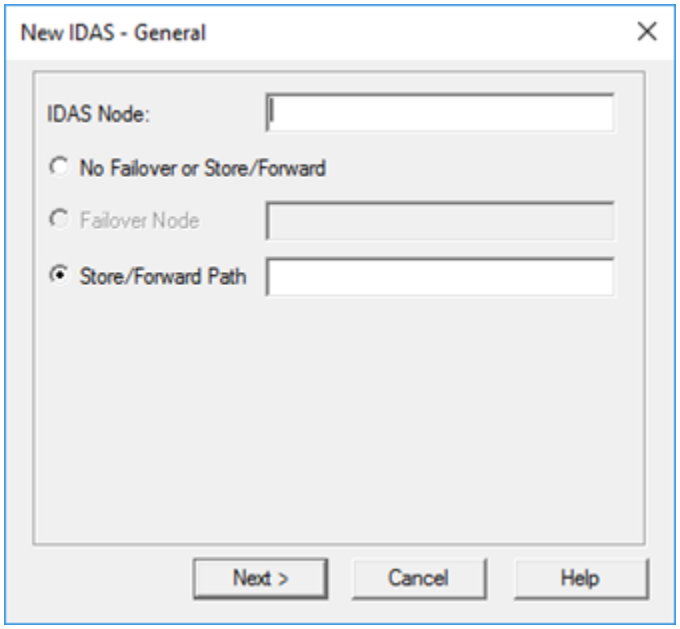
Adding an IDAS

If you are adding a remote IDAS, install and configure the IDAS software on the remote computer before setting up the IDAS configuration in the Operations Control Management Console. See Configuring IDAS on a Remote Node for more information.

During the installation, you are prompted to specify the network account that will be used by a remote IDAS and the historian for communication. This account must belong to the Windows Administrators group on both computers.

To add an IDAS

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**.
3. Right-click **Data Acquisition** and then click **New IDAS**.



4. Enter the name of the computer on which the IDAS runs in the **IDAS Node** field. The IDAS software must be installed on the target computer.

   Note: If the target IDAS node is also a Historian node, it does not work as a remote IDAS. Choose a node that has the IDAS software installed, but without the Historian Server software installed.

5. Specify the failover or store-and-forward option:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- If you don't want to use failover or store-and-forward, select **No Failover or Store/Forward**.
- To specify a backup IDAS, select **Failover Node**.

  Enter the name of the computer on which the redundant IDAS runs. You must use the fully qualified name of the computer, or the IP address. Verify that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online.

  Note: Failover is only supported with remote IDAS versions prior to 2023.

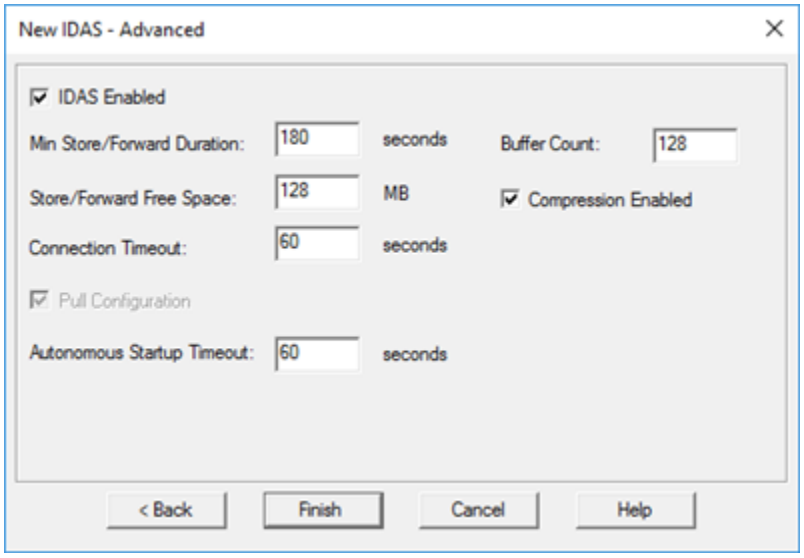
- To enable store-and-forward, select **Store/Forward Path**.

  Enter an absolute path (for example C:\IDASStoreForward). The Store/Forward Path is used to cache metadata about historized tags. Additionally, when communications to the Historian server are interrupted, it's used to historize the data. Always use the same store/forward path for the same IDAS; changing the path might leave behind unprocessed data that has not yet been sent to the Historian.

  If the IDAS is on the same physical computer as the Historian, the InSQLConfiguration virtual service account must have full file system permissions on this folder. If this is a remote IDAS, the aahSupervisor virtual service account must have full file system permissions on this folder.

  If the **Store/Forward Path** is incorrect, or the permissions are not set correctly, the store-and-forward functionality will fail.

6. Click **Next**.



7. Enter the advanced information for the new IDAS.

   For more information on these options, see Editing Advanced Information for an IDAS.

8. Click **Finish**.

The right pane of the Operations Control Management Console displays both the SuiteLink and HCAL connections for your new IDAS.

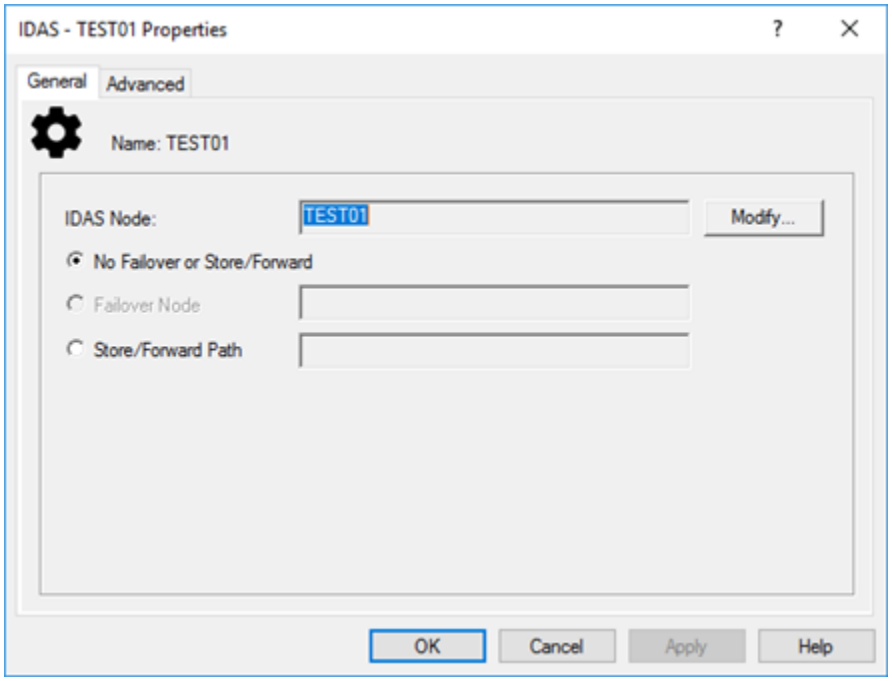
Editing General Information for an IDAS

To edit general information

1. In the Operations Control Management Console tree, expand a server group, and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Right-click the name of the IDAS to edit, and then click **Properties**. The **Properties** dialog box appears.
4. Click the **General** tab.



5. To change the name of the computer on which the IDAS runs, click **Modify** and then type the new name in the **IDAS Node** box. If you are creating a new IDAS definition or modifying an existing one, make sure that the IDAS software is installed on the target computer.

   Note: If the target IDAS node is also a Historian node, it does not work as a remote IDAS. Choose a node that has the IDAS software installed, but without the Historian Server software installed.

6. Specify the failover or store-and-forward option:
   - If you don't want to use failover or store-and-forward, select **No Failover or Store/Forward**.
   - To specify a backup IDAS, select **Failover Node**.

     Enter the name of the computer on which the redundant IDAS runs. You must use the fully qualified name of the computer, or the IP address. Verify that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online.

     Note: Failover is only supported with remote IDAS versions prior to 2023.

   - To enable store-and-forward, select **Store/Forward Path**.

     Enter an absolute path (for example C:\IDASStoreForward). The Store/Forward Path is used to cache metadata about historized tags. Additionally, when communications to the Historian server are interrupted, it's used to historize the data. Always use the same store/forward path for the same IDAS; changing the path might leave behind unprocessed data that has not yet been sent to the Historian.

     If the IDAS is on the same physical computer as the Historian, the InSQLConfiguration virtual service account must have full file system permissions on this folder. If this is a remote IDAS, the aahSupervisor virtual service account must have full file system permissions on this folder.

     If the **Store/Forward Path** is incorrect, or the permissions are not set correctly, the store-and-forward functionality will fail.

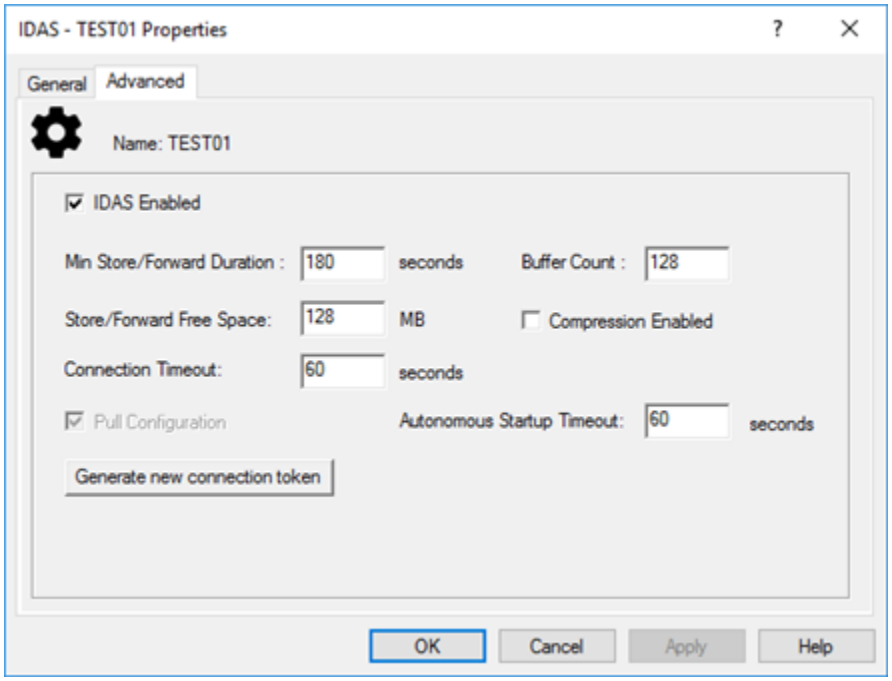
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

7. Click **OK**.

Editing Advanced Information for an IDAS

To edit advanced information

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Right-click the name of the IDAS to edit, and then click **Properties**. The **Properties** dialog box appears.
4. Click the **Advanced** tab.



5. Mark the **IDAS Enabled** check box to allow the system to use the IDAS.
6. Configure store-and-forward options:
   - In **Min Store/Forward Duration**, specify the minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
   - In **Buffer Count**, specify the number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates.
   - In **Store/Forward Free Space**, specify the minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer
   - Select **Compression Enabled** to allow data compression.
   - In **Connection Timeout**, specify amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks.

     For more information on store-and-forward, see IDAS Store-and-Forward Capability.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

7. Configure the IDAS for autonomous startup:
   - **Pull Configuration** is enabled by default when a new IDAS is configured. When enabled, the IDAS retrieves its configuration information from the Historian server automatically.
   - If the IDAS is configured for store/forward, the IDAS will start itself autonomously even if the historian is unavailable. In **Autonomous Startup Timeout**, specify the amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network.

     For more information on autonomous startup, see IDAS Autonomous Startup.
8. To revoke an IDAS connection and generate a new token, click the **Generate new connection token button**. This adds a new encrypted connection token in the database. This token is used to allow the remote IDAS to connect to Historian.
9. Click **OK**.

Setting a remote IDAS to "Classic"

When the Historian Server is upgraded from a prior version, all existing remote IDASes are set to "Classic". (That is, Classic = 1 in the _IODriver table).

If a remote IDAS is added after installation, it is created as a new IDAS. But if it is used for data exchange with a legacy system, it should be explicitly set to "Classic".

To set a remote IDAS to "Classic"

1. In SQL Server Management Studio, type this query:

```
Select * from _IODriver
```

2. This lists each IDAS on the computer. Notice that Classic IDAS shows "0" in the Classic column.
3. To change a Classic IDAS to a New IDAS, type this query (where the _IODriverKey is the unique key for the IDAS you want to change):

```
Update _IODriver
SET
      Classic = 1
      WHERE _IODriverKey = 3
```

If the legacy remote IDAS is later upgraded to Historian 2017, you can run a similar Transact-SQL script, but with "`SET Classic = 0`" to indicate that the IDAS is no longer part of a legacy system.

Deleting an IDAS

An IDAS cannot be deleted if topics and/or I/O Servers are still associated with it. Also, at least one IDAS must exist. It is recommended that you delete a remote IDAS while it is connected to the historian. This ensures that the temporary configuration files on the remote computer are deleted.

Configuring I/O Server Types

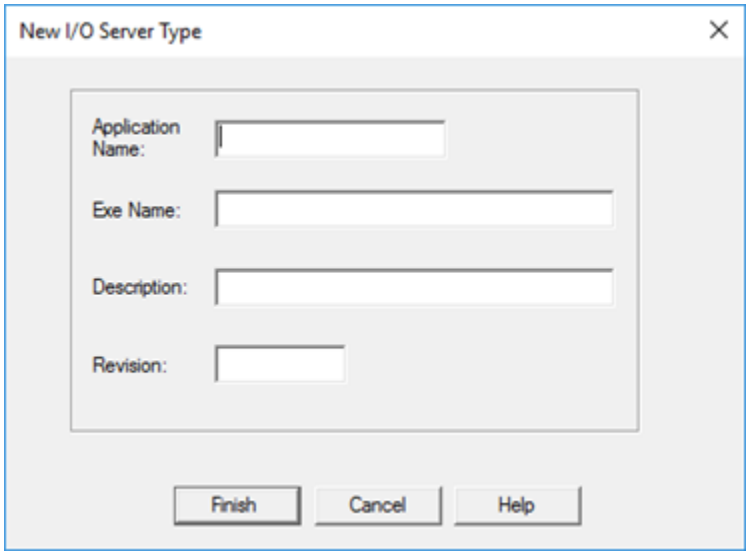
The Operations Control Management Console lists every supported AVEVA I/O Server type that is available at the time that AVEVA Historian is shipped. You can add new I/O Server types at any time. Before you add an I/O

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Server, make sure that its associated type is available in the system for selection.

Adding an I/O Server Type

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Right-click **I/OServer Types** and then click **New I/O Server Type**.



4. Configure the options.

   Application Name

   The application name of the I/O Server. This name is usually the same as the executable file name.

   Exe Name

   The name of the I/O Server's executable file.
5. In the **Description** box, type the description of the I/O Server type.
6. In the **Revision** box, type the revision number for the I/O Server.

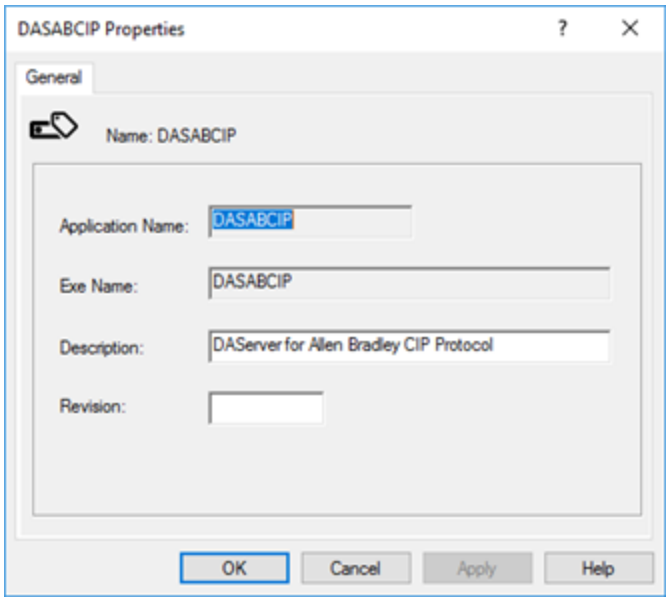
   Note: The values for the **Description** and **Revision** options are not used by the AVEVA Historian.
7. Click **Finish**.

Editing I/O Server Type Properties

To edit properties for an I/O Server type

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Click **I/O Server Types**. A list of types appears in the details pane.
4. Right-click the I/O Server type and then click **Properties**. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. You can only edit the description, revision letter, and platform for an I/O Server type. For information on these options, see Adding an I/O Server Type.
6. Click **OK**.

Deleting an I/O Server Type

To delete an I/O Server type

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. In the **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Click **I/O Server Types**. A list of types appears in the details pane.
4. Right-click the I/O Server type and then click **Delete**.

Configuring I/O Servers

I/O Servers and their associated topics can be imported from InTouch HMI software or added manually using the Operations Control Management Console.

In the Operations Control Management Console tree, selecting the Data Acquisition item shows a list of the configured I/O Servers in the details pane. Using the Operations Control Management Console, you can view, edit, and delete existing I/O Servers and their associated topics. You can also add new I/O Servers and topics. You cannot create an I/O Server tag unless an I/O Server and an associated topic are available.

If you edit I/O Server information and then reimport a tagname database using the Tag Importer wizard, the changes you made to the I/O Server will not be preserved.

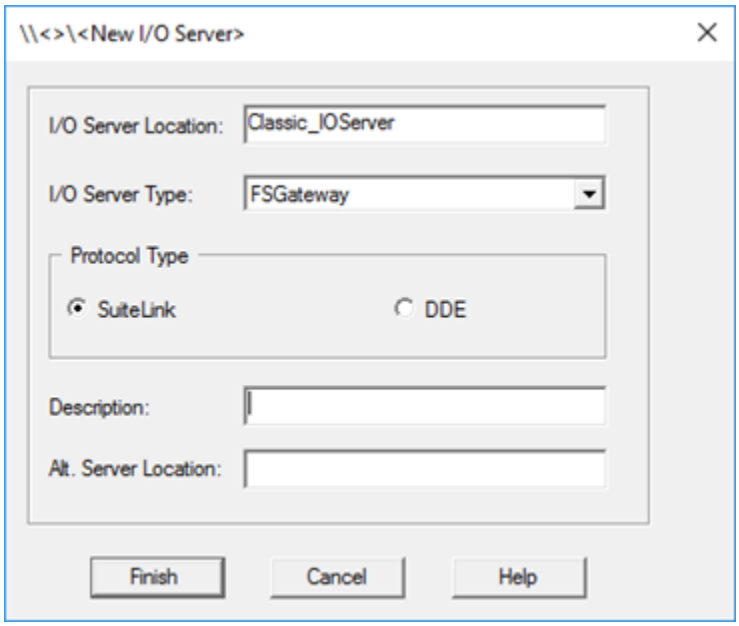
Adding an I/O Server

To add an I/O Server

1. In the Operations Control Management Console tree, expand a server group and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Right-click the IDAS to which you want to add the I/O Server, and then click **New I/O Server**.



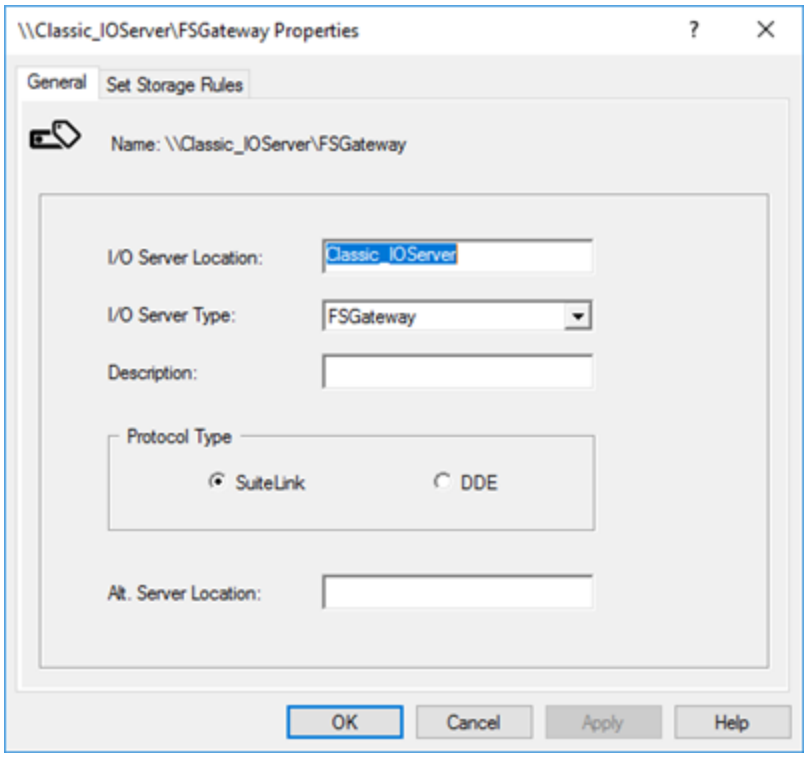
4. Enter the configuration information for the new I/O Server.
   For more information on these options, see Editing General Information for an I/O Server.
5. Click **Finish**.

Editing General Information for an I/O Server

To edit general information

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. In the **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Expand the IDAS associated with the I/O Server.
4. Right-click the name of the I/O Server to edit, and then click **Properties**. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In the **I/O Server Location** box, type the name of the computer on which the I/O Server runs.
6. In the **I/O Server Type** list, select the application name of the I/O Server. This name is usually the same as the executable file name.
7. In the **Description** box, type a description of the I/O Server.
8. In the **Protocol Type** group, select the protocol that the I/O Server uses to send data to the AVEVA Historian. For more information, see Supported Protocols in the *AVEVA Historian Concepts Guide*.

   Note: DDE is not supported if the historian is running on the Windows Server 2003, Windows Server 2008, or Windows Vista operating system.
9. In the **Alt. Server Location** box, type the name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made.
10. Click **OK**.

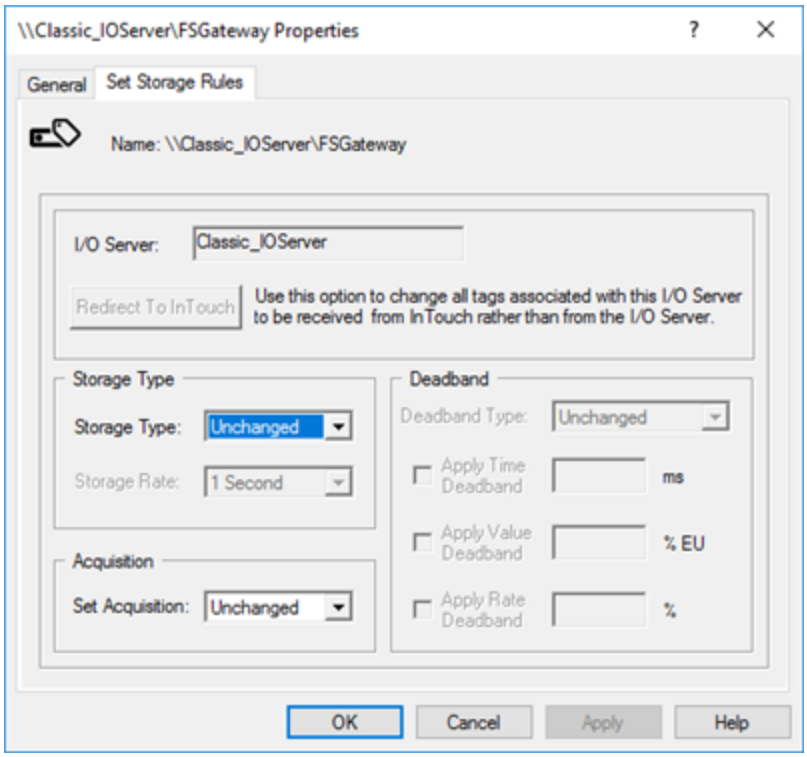
Editing Storage Rule Information for an I/O Server

When you set storage rules for a particular I/O Server, the rules apply to all tag values originating from that I/O Server.

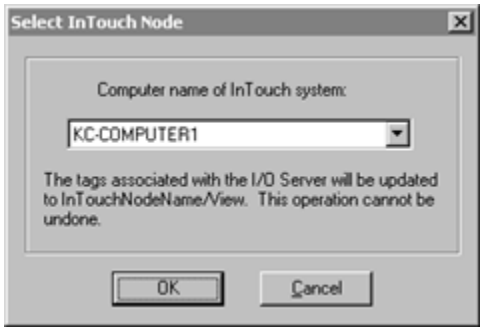
To edit storage rules

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. In the **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Expand the IDAS associated with the I/O Server.
4. Right-click the name of the I/O Server to edit, and then click **Properties**. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Click the **Set Storage Rules** tab.
6. To redirect the I/O Server to **InTouch** HMI software, click **Redirect to InTouch**. This button is only available if at least one I/O Server type is "VIEW." The **Select InTouch Node** dialog box appears.



In the **Computer name of InTouch system** list, select the name of the InTouch node from which you want to acquire tag values. If more than one InTouch nodes are imported, be sure to select the InTouch node that receives data from the I/O Server you are redirecting.

For more information, see Redirecting I/O Servers to InTouch HMI Software.

Important: After you redirect an I/O Server, you cannot undo.

Click **OK** to redirect the I/O Server.

7. In the **Storage Type** group, configure the storage rule for all the tags associated with the I/O Server:

Unchanged

No storage rule is applied at the I/O Server level.

Delta

Tag values are stored only if they have changed.

Cyclic

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag values are stored according to a fixed rate, which you can select from the **Storage Rate** list.

None

Tag values from this I/O Server are not stored into history.

Forced

All values received from this I/O Server are stored.

8. In the **Deadband** group, configure the deadband. Options in this group are only available if delta storage is selected in the **Storage Type** group. For the deadband type you select, configure the appropriate options.

   Unchanged

   No storage rule is applied at the I/O Server level.

   Time and/or Value

   A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

   A value deadband is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.

   Swinging Door

   A swinging door deadband is the percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.

9. In the **Set Acquisition** box, select whether or not to turn data acquisition from the I/O Server either on or off. The Unchanged option allows you to leave current acquisition settings unchanged, which is useful if you have a mix of acquired and not acquired tags on the I/O Server and do not want to go through all of them.

10. Click **OK**.

Deleting an I/O Server

If you delete an I/O Server and then reimport the tagname database that contained the I/O Server definition using the Tag Importer wizard, the I/O Server is added again. An I/O Server cannot be deleted if there are still topics associated with it.

Configuring Topics

A topic is a logical block of data from an I/O Server. Both the DDE and SuiteLink protocols use topics to locate information coming from I/O Servers.

Adding a Topic

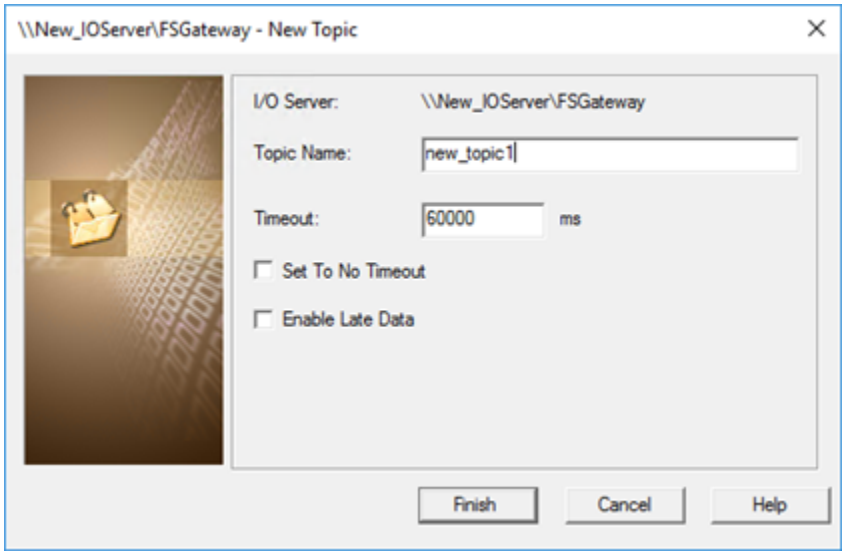
When you add a new topic for an I/O Server, a new row is added to the Topic table in the Runtime database.

Topic names must be unique for the I/O Server, not for the global system. You can have two topics with identical names, as long as they are on different I/O Servers.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To add a topic

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Expand the IDAS that contains the I/O Server.
4. Right-click the I/O Server, and then click **New Topic**. The **New Topic** wizard appears.



Note: If you are configuring a topic for a classic IDAS (that is, an IDAS that existed before installing Historian 2017), you'll also see fields for setting the idle duration and processing interval.

5. Enter the configuration information for the new topic. For more information on these options, see Editing General Information for a Topic and Editing Storage Rules for a Topic.

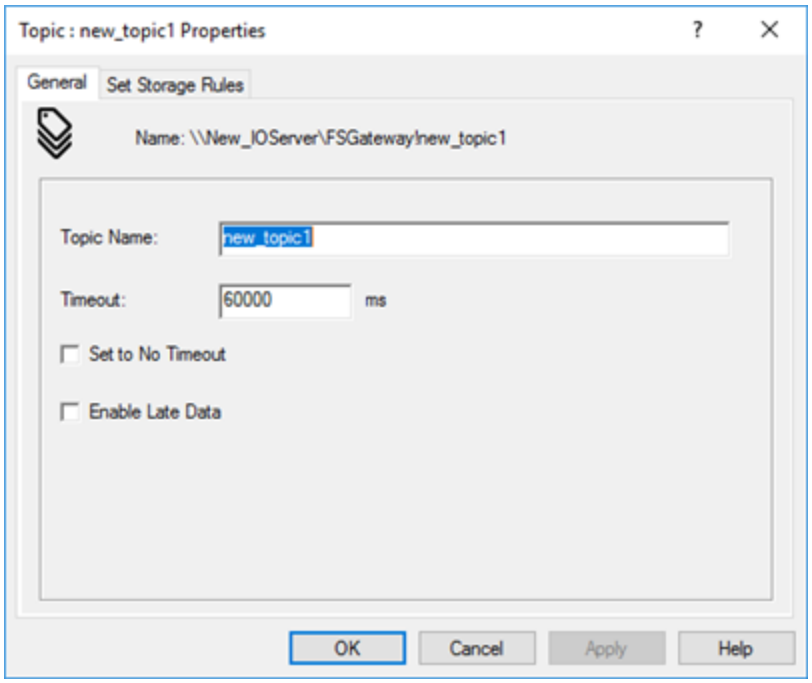
   You can set storage properties for a topic after you add it to the system.
6. Click **Finish**.

Editing General Information for a Topic

To edit general topic information

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Expand the IDAS and then the I/O Server that contains the topic to edit.
4. Right-click the topic, and then click **Properties**. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In the **Topic Name** list, type the name of the topic.
6. In the **Time Out** box, enter the time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic. The default is 60000 milliseconds.

   Note: You can also manually force a reconnect for one or all of the topics in the system. For more information, see Reinitializing I/O Topics.

7. To disable the time out, mark the **Set to No Time Out** check box.

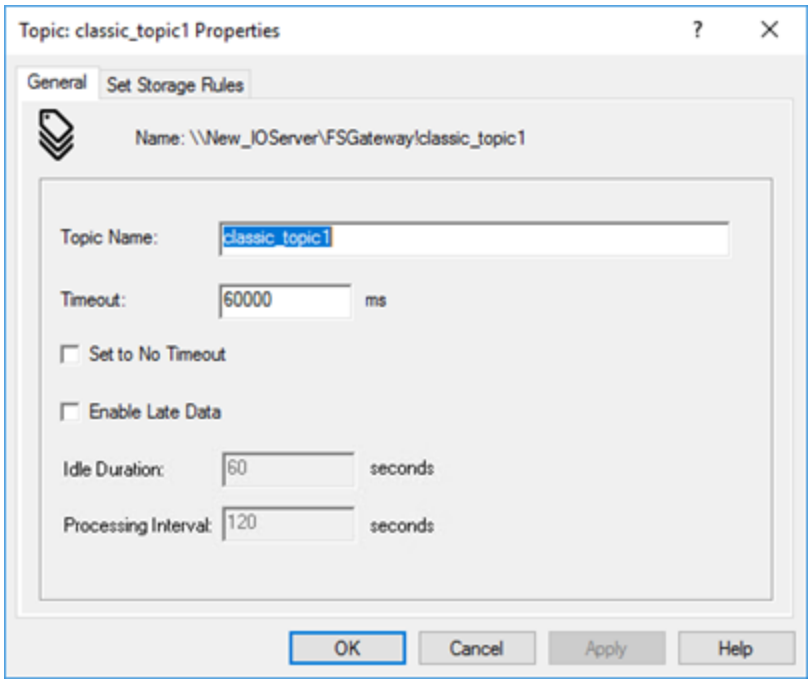
   You might want to disable the time out if the topic has data values that are not changing at all or changing very slowly. If you have a slow-changing tag for which a time out is occurring frequently, you will see periods of NULL data in history, as a result of the historian disconnecting and reconnecting. Disabling the time out prevents the historian from disconnecting, so that valid data is always being logged.

   The topic timeout is automatically set to 0 and disabled if you enable late data for the topic (configurable on the **Set Storage Rules** tab).

   To allow acquisition of "late" data, mark the **Enable Late Data** check box.

8. If you are configuring a topic for a classic IDAS (that is, an IDAS that existed before installing Historian 2017), configure these options:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- For **Idle Duration**, specify the amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds.
- For **Processing Interval**, specify the amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration.

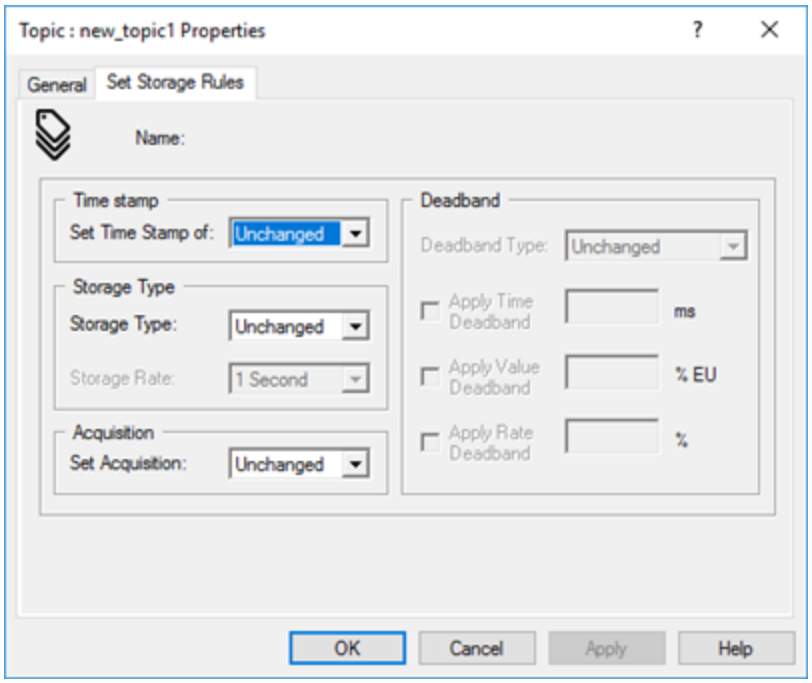
9. Click **OK**.

Editing Storage Rules for a Topic

To edit storage rules for a topic

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Data Acquisition**.
3. Expand the IDAS and then the I/O Server that contains the topic to edit.
4. Right-click the topic to edit, and then click **Properties**. The **Properties** dialog box appears.
5. Click the **Set Storage Rules** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



6. In the **Set Time Stamp of** list, select the whether the timestamp of the data source or the historian server (specifically, HCAL) should be used. Choosing the server option is useful if the source-supplied timestamp is unreliable. Note that the historian handles incoming data that has a timestamps in the future.

   Note: If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage.

7. In the **Storage Type** group, configure the storage rule for all the tags associated with the topic:
   - **Unchanged** -- Use this if no storage rule is applied at the topic level.
   - **Delta** -- Use this if tag values are stored only if they have changed.
   - **Cyclic** -- Use this if tag values are stored according to a fixed rate, which you can select from the **Storage Rate** list.
   - **None** -- Use this if tag values from this topic are stored are not stored into history.
   - **Forced** -- Use this if all values received from this topic are stored.
8. In the **Deadband** group, configure the deadband. Options in this group are only available if delta storage is selected in the **Storage Type** group. For the deadband type you select, configure the appropriate options.
   - **Unchanged** -- Use this if no storage rule is applied at the topic level.
   - **Time and/or Value** -- A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

     A value deadband is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
   - **Swinging Door** -- A swinging door deadband is the percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.

9. If you specify a Delta storage type, mark the corresponding check box and specify the parameters for the deadband type you selected:
   - **Apply Time Deadband** -- Specify time in milliseconds.
   - **Apply Value Deadband** -- Specify value as a percentage of the engineering unit for that tag.
   - **Apply Rate Deadband** -- Specify rate as a percentage.
10. In the **Set Acquisition** box, specify whether to turn data acquisition from the topic on or off. The Unchanged option allows you to leave current acquisition settings unchanged, which is useful if you have a mix of acquired and not acquired tags on the topic and do not want to go through all of them.
11. Click **OK**.

Deleting a Topic

If you delete a topic and then reimport the tagname database that contained the I/O Server definition using the Operations Control Management Console, the topic definition is added again to the database. A topic cannot be deleted if tags are still associated with it.

Reinitializing I/O Topics

You can manually reinitialize I/O conversations for topics using the Operations Control Management Console. When you reinitialize a topic, the existing I/O conversation is closed and the entire process for setting up the I/O conversation restarts. All I/O points associated with that topic are affected when the reinitialization occurs. You can either reinitialize all of the topics or a single topic.

Note: You can also enable an automatic topic time out, in which the AVEVA Historian issues a disconnect and reconnect for a topic that has not provided data within a specified time span. For more information, see Editing General Information for a Topic.

To reinitialize all topics

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Management Console**.
3. Right-click **Data Acquisition**, point to **All Tasks**, and then click **Reinitialize All Topics**. The **Reinitialize All Topics** dialog box appears.
4. Click **OK**.

To reinitialize a single topic

1. Click **Data Acquisition**.
2. In the details pane, right-click the topic you want to reinitialize, and then click **Reinitialize Topic**. The **Reinitialize One Topic** dialog box appears.
3. Click **OK**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Managing Data Storage

About Data Storage

AVEVA Historian uses these structures to store data:

- **SQL Server database file (.mdf)**
  Configuration information and classic event data is stored in a SQL Server database file (.mdf). When you install the historian, this database file is created for you and is named Runtime. The Runtime database file is named according to this convention: RuntimeDat_<*version_number*>_<*original_server_name*>.Mdf. The transaction log is named: RuntimeLog_<*version_number*>_<*original_server_name*>.Ldf. For general information on database files, see your Microsoft SQL Server documentation.

  Note: The Holding database is used internally by the historian if you import a tag database from an InTouch application. The file names for the Holding database are HoldingDat_<*version_number*>_<*original_server_name*>.Mdf and HoldingLog_<*version_number*>_<*original_server_name*>.Ldf.

- **History blocks**
  Processing data (including alarms and events), replication data (if configured), and auto-summary data in history blocks. A history block is a folder containing data files of a proprietary format and, possibly, subfolders. Every history block is bound to a fixed time interval specified at its creation. History block time intervals within the same storage partition do not overlap.

  If no data is acquired, or if a block is deleted, for a certain time period, there may be gaps in the history blocks. These are also known as *block gaps*.

- **Storage Partitions**
  A Historian server can be configured to run multiple storage instances at the same time, where each instance is associated with a storage partition (also called a storage shard). A storage partition is a set of folders with history blocks not overlapping in time. Currently two types of storage partition are supported - the Main storage partition for primary data, and Auto Summary storage partition for calculated summaries.

For backward compatibility, AVEVA Historian also supports these data storage structures:

- **A2ALMDB database**
  Since the release of Historian 2017, alarms and events are stored in historian data blocks by default. Available since AVEVA Historian 2014 R2 release, you have the option of changing this default (when configuring the historian) to use the legacy A2ALMDB (SQL) database.

About Data Storage Subsystem Processes

The Data Storage subsystem consists of one or more processes named aahStorage.exe. The Main and Auto-Summary storage instances are child processes. Therefore, in the Windows Task Manager you can see several instances of the aahStorage.exe process that are differentiated by their command lines.

The Storage subsystem is available in 32 and 64-bit versions. If you install AVEVA Historian on a 64-bit operating system, the Storage subsystem always runs as native 64-bit application. However, if an AVEVA Application Server engine was deployed in a 32-bit mode, the store-and-forward storage engine on that computer will also be 32-bit if there was no prior historian installation on the computer. The Storage subsystem also includes the Metadata Server process (aahMetadataServer.exe) responsible for caching Storage tag metadata persisted in the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Runtime database.

The Classic Data Redirector service (aahStoreSvc.exe) is responsible for redirecting to storage any streamed data coming from classic IDASs (version 2014 R2 or earlier) and the system driver. The aahManStSvc.exe service handles data that is imported from CSV files and store-and-forward data from IDASs from version 2014 R2 or earlier.

Integration with Microsoft SQL Server

While AVEVA Historian distinguishes itself in how it uniquely manages time-based data, it relies on SQL Server for certain foundational functionality.

Static Data Management

The SQL Server Runtime database easily manages relatively static data, like configuration data which does not change at a real-time rate. Over the life of a site, tags are added and deleted, descriptions are changed, and engineering ranges are altered. The Runtime database stores this type of information.

Runtime Database Structure

The Runtime database is the SQL Server online database for the entire AVEVA Historian. The Runtime database is shipped with a set of standard database entities, such as tables, views, and stored procedures to store configuration data for a typical factory. You can use the Configuration Editor within the Operations Control Management Console to easily add configuration data to the Runtime database that reflects your factory environment.

OLE DB Interface

Microsoft SQL Server Object Linking and Embedding for Databases (OLE DB) is used to access the time-based data that the historian stores outside of the SQL Server database. You can query the Microsoft SQL Server for both configuration information in the Runtime database and historical data on disk, and the integration appears seamless.

Historian also leverages SQL Server features such as database security, replication, and backups.

About Delta Storage Mode

AVEVA Historian supports four storage modes:

- **No storage** - No data values are stored.
- **Forced storage** - All collected data values are stored.
- **Cyclic storage** - Only data values that occur at a specified time interval are stored.
- **Delta storage** - Only changed data values are stored.

Using delta storage mode, the historian stores data based on a change in a value. Delta storage writes a historical record only if the current value changes from the previous value. Delta storage is also called "storage by exception." Delta storage is typically used to store discrete values, string values, and analog values that remain

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

constant for long periods of time. For example, you don't want to store the value of the discrete tag "PUMPON" every ten seconds if the pump usually stays on for months at a time. The value is stored along with a timestamp of when the change occurred, to an accuracy of 1 ms.

The following types of deadbands can be applied for delta storage:

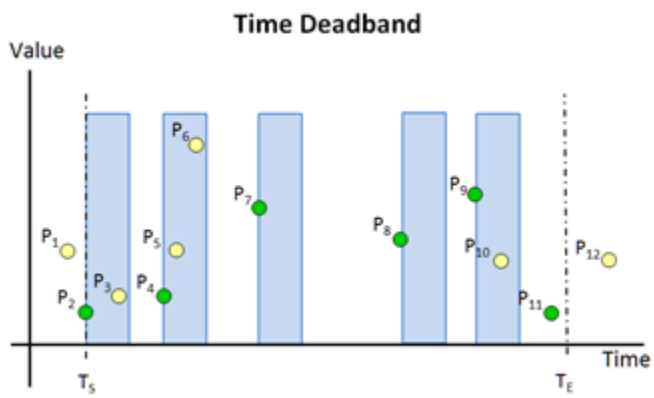
- Time deadband
- Value deadband
- Rate of change (swinging door) deadband

Time and Value Deadbands for Delta Storage

To further decrease the resolution of tag values stored in delta mode, use a time deadband or a value deadband.

- A *time deadband* is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored.

  This illustration shows an example of applying a time deadband:



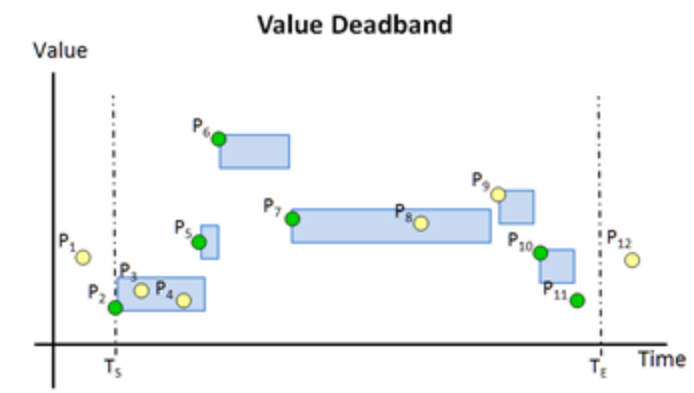
  Data is stored for the time period starting with TS and ending with TE. All points in the graphic represent data values for a given tag over time. The grey areas represent the time deadband, which starts anew with every returned value. Only the green points (P2, P4, P7, P8, P9, P11) are stored. The other points are not stored because they fall within a deadband.

  The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

- A *value deadband* is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored.

  This illustration shows an example of applying a value deadband:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



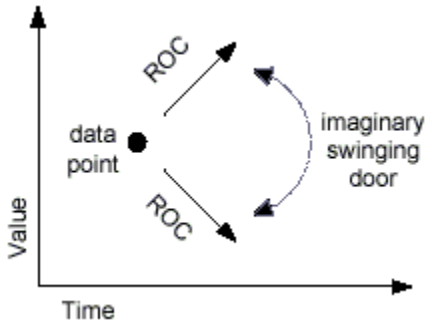
Data is stored for the time period starting with TS and ending with TE. All points in the graphic represent data values for a given tag over time. The grey areas represent the value deadband, which starts anew with every returned value. Only the green points (P2, P5, P6, P7, P10, P11) are stored. The other points are not stored because they fall within a deadband. P9 is not stored because P8 was discarded and it is within the percentage deviation.

The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.

Swinging Door Deadband for Delta Storage

A swinging door deadband is the percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.

The swinging door deadband is essentially a rate of change deadband, based on changes in the slope of the incoming data values. For example, specifying a swinging door deadband value of 10 percent means that points will be stored if the percentage change in slope of the consecutive data values exceeds 10 percent. The percentage of allowable "swing" in the data values gives this type of deadband its name.



Benefits of the Swinging Door Deadband

One benefit of using a swinging door deadband is that it reduces the disk space required to store data. However, because the storage system already provides a good compression ratio, the amount of disk space that is saved by applying this type of deadband for slow-changing tags (changing less than twice in a 15-minute interval) is negligible. For example, a tag that changes 12 times per hour will use 2K bytes of disk space in a 24-hour period.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

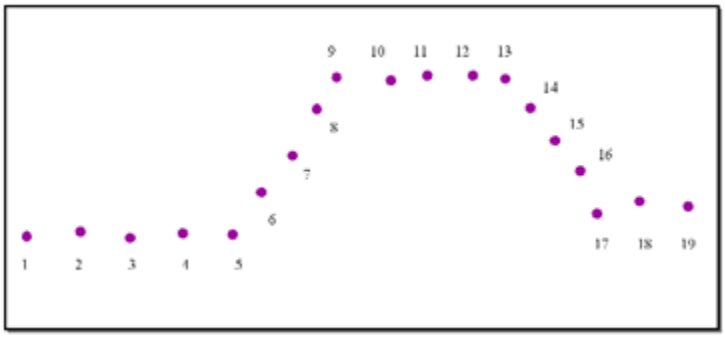
Even if only every fifth point is stored, the savings is only 1.5K bytes per day.

Another benefit of the swinging door deadband is that it captures the data value before the rate change, which is something that a value deadband does not do. If you trend data, the peaks and valleys of the trend curve are more defined to provide a more accurate picture of what is happening in your plant.

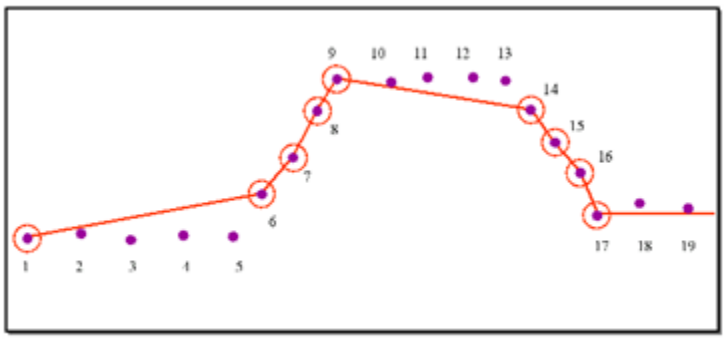
Generally, using a swinging door (rate) deadband provides better representation of the value change curve with the same or less number of values stored than regular value or time deadbands for delta storage.

The following graphics compare the trend curves of the same raw data, but with different deadbands applied.

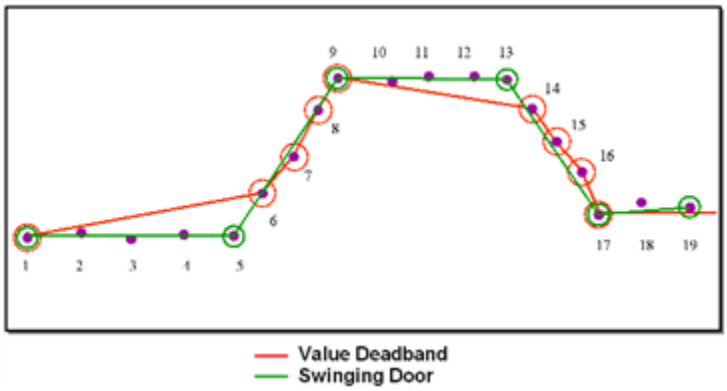
The following graph shows the trend of the actual raw data values:



The following graph shows the trend of the data values with a value deadband applied. Notice how only the first data value that deviates by the deadband from the previous value will be stored, and not any of the values between the starting value and the first deviating value.



The following graph shows the data values that will be stored for both a value deadband and a swinging door deadband. Notice how the swinging door deadband captures data before the deadband change, allowing for a more complete view of the data.



A swinging door deadband is most useful for tags that have a steady increase and decrease in slope, such as a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tank level or tank temperature that rises and falls. A swinging door deadband may not be appropriate for "noisy" signals, in which the value of the tag constantly fluctuates around a certain point for long periods of time. Also, the reduction in storage requirements offered by the swinging door deadband may not have much of an impact if you have an application with a small tag count (for example, 500 tags). In this case, it may not be necessary to use a deadband at all.

A swinging door deadband is applicable for analog tags that receive data from the following sources:

- Real-time data values from I/O Servers, MDAS, or HCAL
- Store-and-forward data from a remote IDAS
- Late data from an I/O Server topic that was configured for late data
- A "fast load" CSV import
- Real-time inserts of data using a Transact-SQL statement

A swinging door deadband is not applicable for manual inserts of data through a CSV import of a Transact-SQL statement.

To best visualize the tag that uses swinging door storage, plot a trend using the Historian Client Trend application and set the plot type from to "line" (rather than "step-line").

Additional Options that Affect the Swinging Door Deadband

The swinging door deadband (the rate deadband) can optionally be combined with a value deadband and/or a deadband override period. This combination will affect which values are actually stored.

If the data is store-and-forward data from a remote IDAS, the behavior of the swinging door algorithm also depends on the value of the real-time window in the AVEVA Historian, as specified by the RealTimeWindow system parameter.

- **Value deadband**
  When combined with rate deadband (with or without a deadband override period), the value deadband is always applied first, followed by the other deadbands. For the value deadband, the system checks the difference in value between the received point from the value of the last stored point. Only when this difference exceeds the value deadband does the system consider the point for rate evaluation.
- **Deadband "override" period**
  If the elapsed time since the last stored point exceeds the deadband override period, the last received point before the time at which the deadband override period expired is stored, regardless of value and rate deadband.
- **Real-time window (Classic Storage subsystem)**
  The real-time window setting for IDAS store-and-forward data (RealTimeWindow system parameter) allows for the expansion of the time window for which the storage system considers data to be "real-time." The real-time window is important for swinging door deadbanding because it determines the maximum length of time that a point will be "held" by the storage system without storing it, while waiting for the next point. For more information, see About the Real-Time Data Window.

Real-time window and deadband override periods are two independent modifiers that force the storage of received points that may have otherwise been discarded due to the setting of either the rate deadband or the value deadband.

The real-time window specification is more likely to select points for storage when the time period between

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

points received from the source is less than the real-time window, but the slope of the incoming data values is such that the rate deadband excludes the points from being stored.

The deadband override period is more likely to select points for storage if the rate at which points are received from the data source is slow (slower than the real-time window) and the rate deadband excludes the points from being stored.

For an illustration of how these factors work together to determine the actual values to be stored, see Swinging Door Deadband Examples.

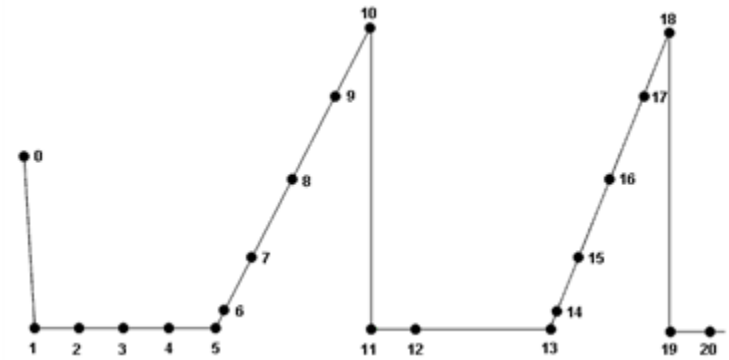
Whatever the combination of rate deadband, value deadband, and deadband override period specified, only points actually received from the data source are stored on disk. That is, points to be stored on disk are never "manufactured" by the swinging door algorithm. This is particularly relevant in understanding the behavior implied by specifying the real-time window and the deadband override period.

Swinging Door Deadband Examples

The effects of the different swinging door options can be illustrated using these three examples:

- Swinging Door Deadband: Rate Only
- Swinging Door Deadband: Rate and Value
- Swinging Door Deadband: Rate, Value, and Deadband Override Period

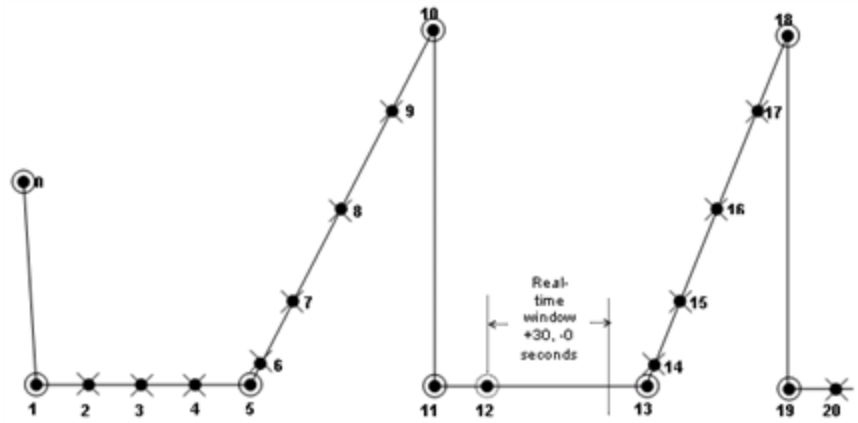
All of the examples are based on the following raw data. The numbered points represent actual values received from a data source.



Swinging Door Deadband: Rate Only

The following diagram depicts an ideal case, where the incoming signal is noise-free and with a proper rate deadband specification only (no value deadband or deadband override period).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Assume point 0 has been stored on disk. The system waits for point 2 to arrive before making its next storage decision. When point 2 is received, the storage engine calculates the change in slope as follows:

Slope0_1 is considered the base slope, and Slope1_2 is considered the current slope.

Slope0_1 = (Value1 - Value0) / (Time1 - Time0)

Slope1_2 = (Value2 - Value1) / (Time2 - Time1)

Slope_Change_Percent = 100* | (Slope1_2 - Slope0_1) / Slope0_1 |

If

Slope_Change_Percent > Rate_Deadband_Specified

In other words, if the percentage change in slope is greater than the specified rate deadband, the storage engine goes ahead and stores point 1 on disk. Next, it receives point 3. The base slope for point 2 will be the slope between points 1 and 2. The current slope will be the slope between points 2 and 3 only if point 1 was stored. If point 1 was not stored, then the base slope for point 2 will be the slope between points 0 and 1, and the current slope will be the slope between points 2 and 3.

The base slope for an evaluation point is not changed unless the previous point is stored; otherwise, the base slope will be the last known current slope that caused a point to be stored on disk.

Assuming point 1 is stored, because the slope between points 2 and 3 is about the same as the slope between points 1 and 2, the rate deadband criterion is not satisfied, and point 2 is discarded. When point 4 is received, the slope change calculation results in point 3 being discarded, and so on until point 6 arrives. Now the rate deadband criterion is satisfied (slope change between points 5 and 6 and points 1 and 2 is greater than the rate deadband specified), and point 5 is stored on disk.

The arrival of point 7, likewise, discards point 6 even though the actual slope between point 6 and point 7 may be quite high, and may even be higher than the rate deadband specified, it is not sufficiently different from the slope between points 5 and 6 to qualify point 6 to be stored. Following this logic through until point 12 is received results in the storage on disk of points 10 and 11, discarding all the other points in between.

Point 13 illustrates the effect of the real-time window setting. Under normal circumstances, point 12 would not qualify to be stored. If, however, the elapsed time between receiving point 12 and point 13 exceeds the time window in which the storage engine is able to store point 12 as a real-time point, point 12 is stored anyway, and the value of the SysRateDeadbandForcedValues system tag is incremented. In other words, if, while the system waits for point 13 to arrive, the timestamp of point 12 becomes so old that it reaches the limit for the real-time window, point 12 is stored regardless of whether it is outside the deadband.

The SysRateDeadbandForcedValues system tag counts the number of "extra" points stored as a result of an insufficient real-time window for swinging door storage.

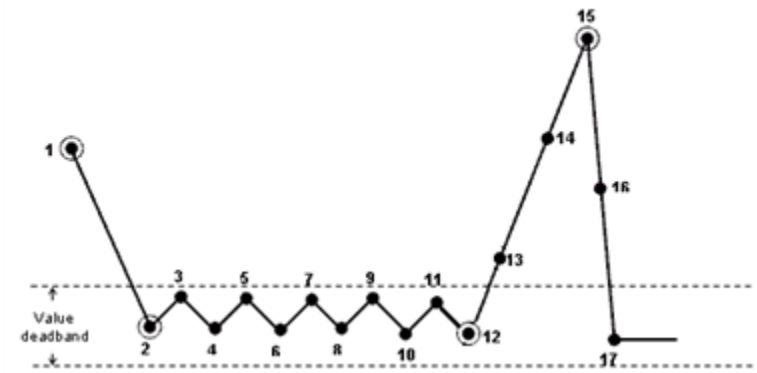
When point 14 arrives, the base slope for evaluating point 13 is between points 11 and 12, and not between

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

points 12 and 13, because point 12 was stored due to the real-time window expiration. A point stored due to the real-time window does not re-establish the base slope; only points stored due to exceeding the rate change causes the base slope to be re-established. Then "normal" rate change evaluation resumes, resulting in point 13 being stored, and so on.

Swinging Door Deadband: Rate and Value

In the following diagram, a signal with some "noise" is shown. The effect of applying both a rate and value deadband to swinging door storage is illustrated. The value deadband is indicated by two horizontal dashed lines.



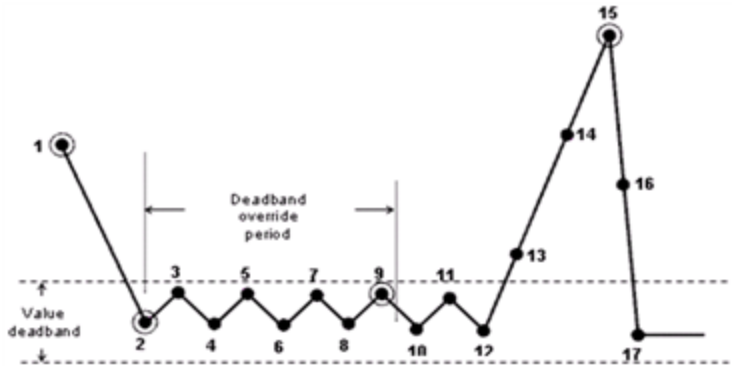
Assume that point 1 has been stored to disk. Point 3 passes the value deadband check, allowing points 2 and 3 to be evaluated for rate change. Assuming that the point exceeds the rate change requirement, then point 2 is stored. Until point 13 is received, all intermediate points are discarded by the value deadband filter. In this example, it is assumed that the change in slope between points 2 through 3 and points 12 through 13 is greater than the rate deadband, so point 12 is stored on disk. When point 14 is received, the normal operation begins.

If a rate deadband is applied without a value deadband, all of the "noisy" points (3 through 11) would have been stored, because the slope of the signal changes radically between successive points. The value deadband removes the noise, but also introduces some amount of distortion in the resultant signal.

Swinging Door Deadband: Rate, Value, and Deadband Override Period

The following graphic illustrates the effect of a rate deadband combined with a value deadband and a deadband override period.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Assume point 1 is stored to disk. Point 3 makes it through the value deadband check, allowing points 2 and 3 to be evaluated for rate change. Assuming the point exceeds the rate change requirement, then point 2 is stored.

Adding a value deadband alone could result in distortion of the stored data.

For example, suppose that the rate deadband is specified such that point 12 does not get stored. That is, the change in slope between points 2 through 3 and points 12 through 13 is not greater than the rate deadband. In that case, the data representation (points 1, 2, and 15) is grossly distorted because the value deadband is discarding key points.

To allow for better representation, a deadband override period may optionally be specified. If the elapsed time between the last stored point and the currently received point is more than the specified deadband, then the point immediately prior to the currently received point is stored. In this example, the elapsed time between point 2 and point 10 is more than the deadband, so point 9 is stored. The data actually stored to disk (points 1, 2, 9, and 15) is a better approximation of the original data.

It is important to note that after point 9 is stored, subsequent rate calculations use the slope between points 2 and 3 as the baseline for subsequent storage decisions because point 2 was last point that was stored normally by storage.

The deadband override period can have any value and is not related to the real-time window value.

Managing the AVEVA Historian Runtime Database

The Runtime database is a SQL Server database and can be managed with SQL Server Management Studio. The Runtime database stores relatively static information about how the AVEVA Historian is configured, such as tag definitions and I/O Server definitions.

Although the Runtime database does not store historical plant data, it stores other types of system-generated data that impacts the size of the database file. For example, information is added to the database file if you:

- Turned on modification tracking, because a record is stored for each modification made. For more information, see Tracking Modifications.
- Defined any events in the classic event system. Each detected event is logged in the Runtime database. If you configured any summary actions, the summarized values are stored in the Runtime database. Also, if you set up event snapshot actions, the values for the snapshots are logged in the database. For more information, see Classic Event Subsystem.

Be sure that you have enough disk space to accommodate a growing Runtime database file. By default, the Runtime database is configured to expand automatically when required.

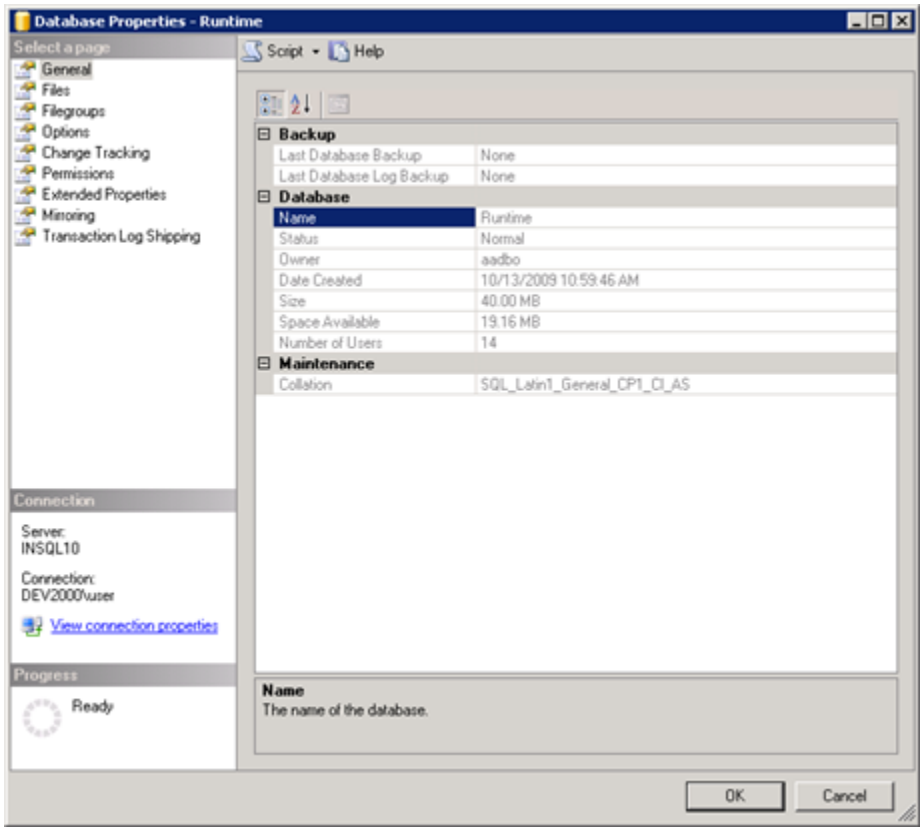
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Changing the Properties for the Runtime Database

You can view or change the properties for the Runtime database, such as the paths to the database files and transactions logs and the permissions on the database.

To view/change the Runtime database properties

1. In Microsoft SQL Server Management Studio, expand the AVEVA Historian, and then expand **Databases**.
2. Right-click the Runtime database and then click **Properties**. The **Database Properties** dialog box appears.



3. To view directory in which the database file and transaction log resides, as well as view the current size of the file and log, in the **Select a page** pane, click **Files**.

   Note: To see the database file in the Windows Explorer, look in the \DATA directory of the main Microsoft SQL Server directory.

4. Using the options in the **Runtime Properties** dialog box, you can recalculate the space available in the database or the transaction log. You can also set database options and grant and revoke statement permissions for database users and groups.

   Important: Do not modify the default permissions for the default historian logins and users, as this negatively affects the system.

   For more information on managing databases, see your Microsoft SQL Server Management Studio documentation.

5. Click **OK**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Managing the Runtime Database

Managing a database involves procedures such as performing backups or exporting data.

Note: You should not edit any of the pre-configured tables, stored procedures, or views that are shipped with the AVEVA Historian.

To manage the Runtime database

1. In Microsoft SQL Server Management Studio, expand the AVEVA Historian, and then expand **Databases**.
2. Right-click the Runtime database, point to **Tasks**, and then select the menu command for the task you want to perform.

For more information on managing databases, see your Microsoft SQL Server Management Studio documentation.

Backing Up the Runtime Database

We recommend that you back up all AVEVA Historian and SQL databases:

- Before you make any changes to the database, in case you want to return to the original configuration.
- On a regular schedule, to minimize data loss in the event of a disk failure. The best way to perform database backups is to set up automatic backups using SQL Server Management Studio. You should back up your database at least once a week.

When you perform a database backup, all system tables, user-defined objects, and data are copied to a separate file located on a backup device. Backup devices include disk files, floppy diskettes, zip disks, and tape drives. Backups can be easily managed using the SQL Server Management Studio.

You should not back up the current history block unless you are using a VSS-aware backup utility.

The master and msdb databases should be on the same backup schedule as the Runtime database.

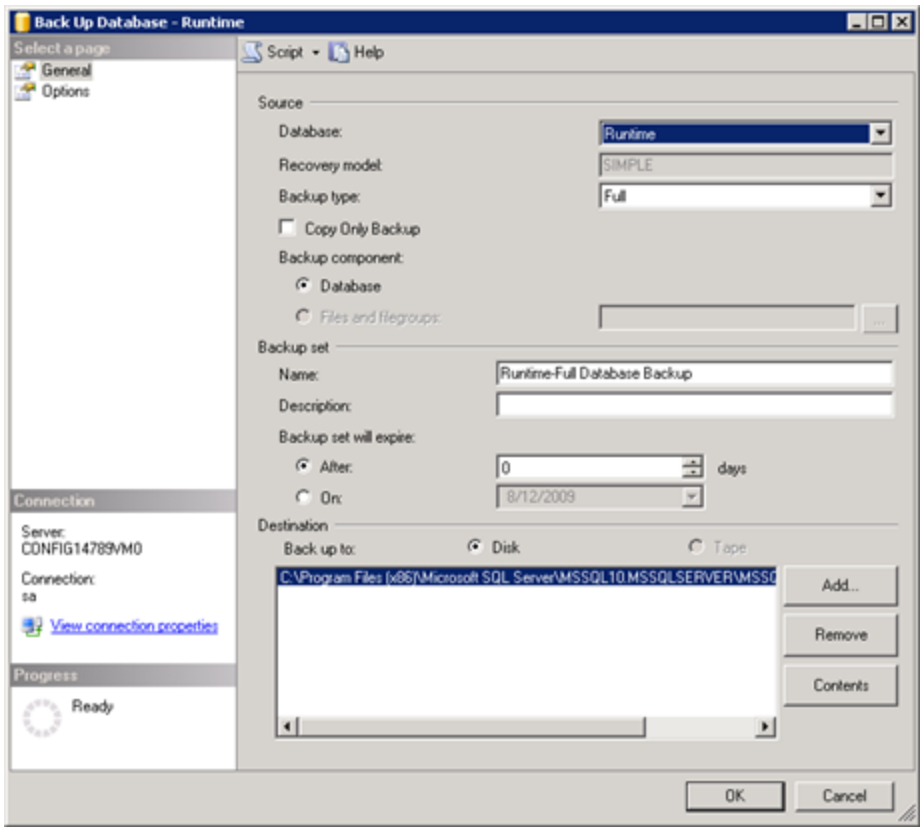
Backing Up the Database

Note: Any transactions that are in progress when the backup is performed are rolled back if that backup is later restored.

To backup the database

1. In Microsoft SQL Server Management Studio, expand the AVEVA Historian, and then expand **Databases**.
2. Right-click the **Runtime** database, point to **Tasks**, and then click **Back Up**. The **SQL Server Backup** dialog box appears.

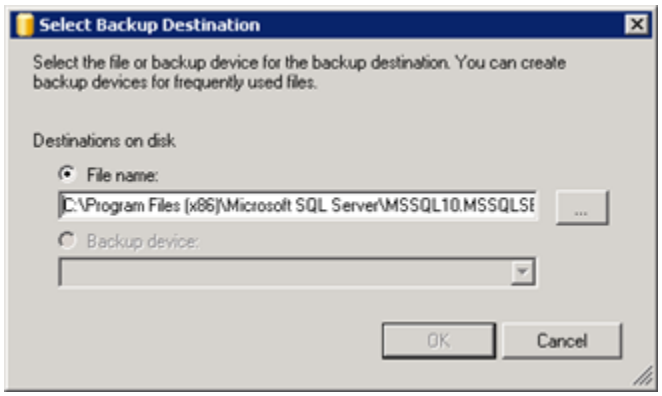
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Click **General**.
4. In the **Database** box, select **Runtime**.
5. To use an existing backup device or file for the backup, select the destination in the **Destination** window and then click **OK** to begin the backup.

   Note: For details on a particular backup destination, select the destination in the list and then click Contents.

6. If you do not have a backup destination defined, click **Add** to add a new destination. The **Select Backup Destination** dialog box appears.



7. Select to back up to either a file or device.

   File name

   Type or browse to a path for the location of the backup file. Be sure that you have enough free disk space to store the backup.

   Backup device

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Select an existing backup device or select **<New Backup Device>**. The **Backup Device Properties** dialog box appears. In the **File name** box, type a name for the device. As you type the name, the path for the backup will be modified. Verify that the path for the backup is correct. When you are done, click **OK** to create the backup device.

8. Click **OK** to close the **Select Backup Destination** dialog box.
9. The newly-created backup device now appears in the **Destination** window of the **SQL Server Backup** dialog box. Select the new backup device.
10. Click **OK** to perform the backup.

You can configure various options for database backups, such as an expiration date for a backup. You can also schedule automatic backups.

For a complete description of database backup and restoration using SQL Server Management Studio, including scheduling recommendations and transaction log backup, see your SQL Server Management Studio documentation.

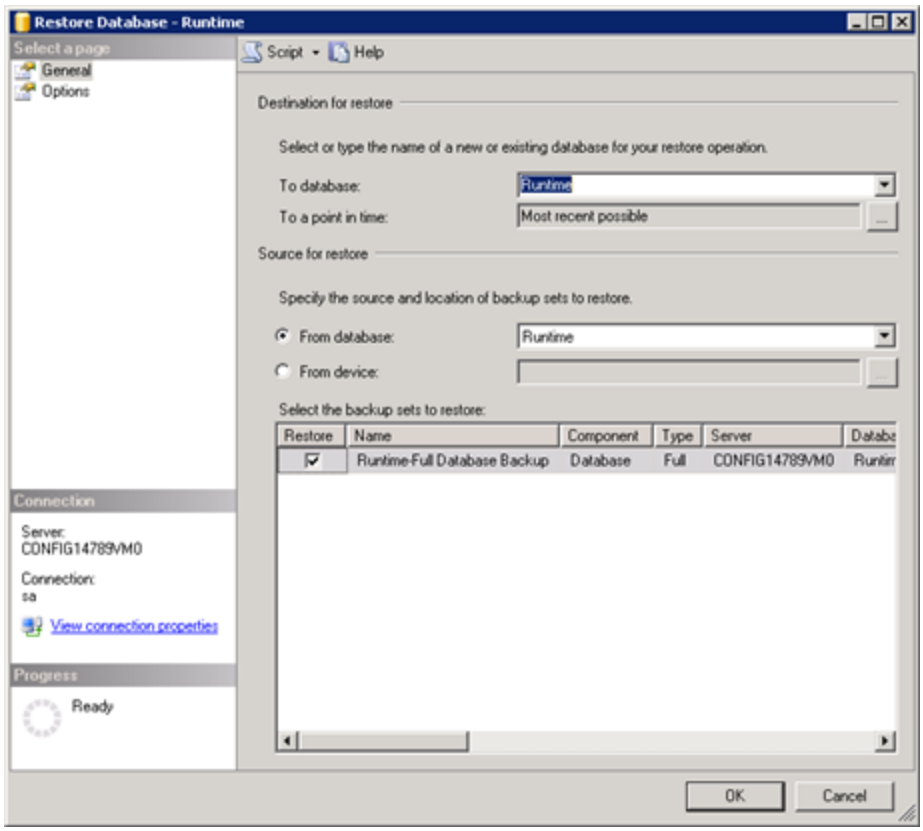
Restoring the Database

When you restore a database from backup, any information saved to the database since the backup was performed is overwritten with the restored information. All changes to the database since the backup are lost. Also, any transactions in progress when the backup was performed are rolled back.

To restore the database

1. In Microsoft SQL Server Management Studio, expand the AVEVA Historian, and then expand **Databases**.
2. Right-click the **Runtime** database, point to **Tasks**, and then click **Restore**. The **Restore Database** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Click **General**.
4. In the **Restore as database** list, select the **Runtime** database.
5. Select **Database** from the **Restore** options.
6. In the **First backup to restore list**, select the desired backup.
7. Click **OK**. The information is restored.

You can configure various options for database restoration. For more information on restoring from a backup using SQL Server Management Studio, see your SQL Server Management Studio documentation.

Managing a Runtime Database Object

A database object is a component of a database: table, index, trigger, view, key, constraint, default, rule, user-defined data type, or stored procedure. Anything that has a name and consumes space in a database is an object.

Note: Do not edit any of the pre-configured tables, stored procedures, or views that are shipped with AVEVA Historian.

To manage database objects

1. In Microsoft SQL Server Management Studio, expand the AVEVA Historian, and then expand **Databases**.
2. Expand the **Runtime** database.
3. All of the object groups in the database appear under the **Runtime** folder.
4. To manage any database object, simply right-click the object to open the **Properties** dialog box for that object.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. Click **OK**.

For more information on managing database objects, see your Microsoft SQL Server documentation.

Space Management for Event and Summary History

If you configured AVEVA Historian to detect events using the Classic Event subsystem, each detected event is logged into the EventHistory table of the Runtime database. If you configured summary actions, the aggregated values are stored in the SummaryHistory table. The duration for which event and summary history are kept are specified by the EventStorageDuration and SummaryStorageDuration system parameters, respectively.

Duration defaults are as follows:

History	Duration
Event	7 days (168 hours)
Summary	14 days (336 hours)

For information on changing the value of a system parameter, see Editing System Parameters.

For more information on the Classic Event subsystem, see Classic Event Subsystem.

Managing Partitions and History Blocks

Historical tag values from your facilities are stored to disk in directories containing special sets of files. These directories are called history blocks. Sets of history blocks are stored in a particular partition on the disk.

Normal tags are stored in one partition, and a second partition contains auto-summary data. See About the Auto-Summary Partition for more information.

You can use the Operations Control Management Console to view and manage partitions and history blocks.

For more about history blocks, see History Blocks in the *AVEVA Historian Concepts Guide*.

Storage Partition Locations

Every storage partition consists of up to four storage locations:

- Circular (mandatory)
- Alternate (optional)
- Buffer (optional, used for backward compatibility only)
- Permanent (optional, used for backward compatibility only)

The paths to the circular, buffer, and permanent storage partitions are initially defined during installation. The alternate storage partition can be defined later using the Operations Control Management Console.

Certain restrictions apply when specifying a path to the storage partition. The circular storage partition must be a local drive on the server, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage partitions can be anywhere on the network, it is strongly recommended to have the alternate storage partition configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

internal hard drive. If you use a network location, the InSQLConfiguration virtual service account must have full access to the location. You can grant that access by using the credentials of the computer account in the format <domain_name>\<computer_name>$. The partition locations must be specified using UNC notation. Mapped drives are not supported.

When planning your storage strategy, be sure to allow enough disk space for storing your plant data for the required length of time.

Note: If the Historian server runs out of disk space, an emergency shutdown is performed. After freeing disk space or specifying an alternate storage location, restarting the Historian may not correctly resume data acquisition. To resolve this, perform a complete shutdown and restart of the Historian. See Shutting Down the Entire AVEVA Historian for detailed instructions.

Circular Storage

Circular storage is used for the main historical data storage. The Storage subsystem creates history blocks of the configured default duration when the data falling into the corresponding time interval needs to be written to disk.

The circular storage location is used to write data in a "circular buffer" fashion. When the free disk space on the disk containing the circular storage location drops below a minimum threshold or when the data is of a specified age, the history block in that storage location is moved to the alternate storage location, and new history blocks get created when necessary. You can also limit the size of the circular storage location. When the contents of the circular storage location reach or exceed this limit, the oldest data will be moved to the alternate storage location. If no alternate storage location is configured, the data is deleted instead of being moved. For more information, see Automatic Deletion of History Blocks.

It is the responsibility of the system administrator to monitor disk space and back up history blocks to a safe location on a periodic basis.

Alternate Storage

When the free disk space in the circular storage location goes below the defined threshold, the circular directory exceeds the specified maximum size, or the blocks reach a certain age, the Storage subsystem will start moving the oldest history blocks to the alternate location, if configured.

History blocks in the alternate storage location are managed in the same way as the blocks in the circular storage location. However, blocks will not be deleted based on age until the sum of the specified ages for both the circular and alternate storage has passed. For example, if circular is set to 60 days, and alternate is set to 90 days, a block is deleted after 150 days.

If the alternate storage location reaches its deletion threshold limit, or configured maximum size, or age, the oldest history blocks are deleted.

A physical drive is strongly recommended, and cannot be the same drive used for circular storage. This storage location is optional.

Permanent Storage

Permanent storage locations are used to store critical data (for example, reactor trips) which must be excluded from the "circular buffer" management, so the Storage subsystem will never try to delete or move to the alternate location the permanent history blocks. This, however, may break the continuity of the history timeline in certain scenarios, and should be used with care, especially when data revision operations can be performed

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

across time intervals overlapping with history blocks stored in the permanent storage location. For that reason, this storage location is supported only for backward compatibility, and it is recommended to use a larger alternate location instead to ensure that important blocks are never deleted..

Data in a permanent storage location can be accessed and viewed along with the data stored in the circular storage location.

Buffer Storage

Buffer partitions are used for temporary purposes, such as retrieval from a data archive. This storage partition can reside on the same hard disk as the circular storage location or on a different disk. Data stored in the buffer storage partition can be accessed and viewed along with the data stored in the circular storage partition. Data is never deleted from this partition by the Storage subsystem.

About the Auto-Summary Partition

AVEVA Historian maintains a separate auto-summary partition for storing automatically generated summaries used for faster retrieval over long time intervals. There, Historian generates a local replication entity and a one-hour summary tag for every analog tag in the system. As values arrive for an analog tag, Historian automatically computes and records a summary.

Auto-summary values are stored in their own history blocks within the auto-summary partition. With auto-summarization, Historian can quickly and efficiently retrieve large-volume data for a long duration, even months or years.

Note: The auto-summary feature was available beginning with AVEVA Historian 2017. From the time you installed or upgraded to AVEVA Historian 2017, the system has been creating auto-summary values for your analog tags. To backfill values for time before that installation or upgrade, you can use the Replication Backfill Manager.

About Block Gaps

AVEVA Historian is designed to store data in a contiguous series of history blocks. But there are two cases that may result in block gaps -- that is, gaps between the history blocks:

- If no data was acquired for the timeframe corresponding to a particular history block, that block will not be created.
- If a history block is deleted for whatever reason, a block gap will result.

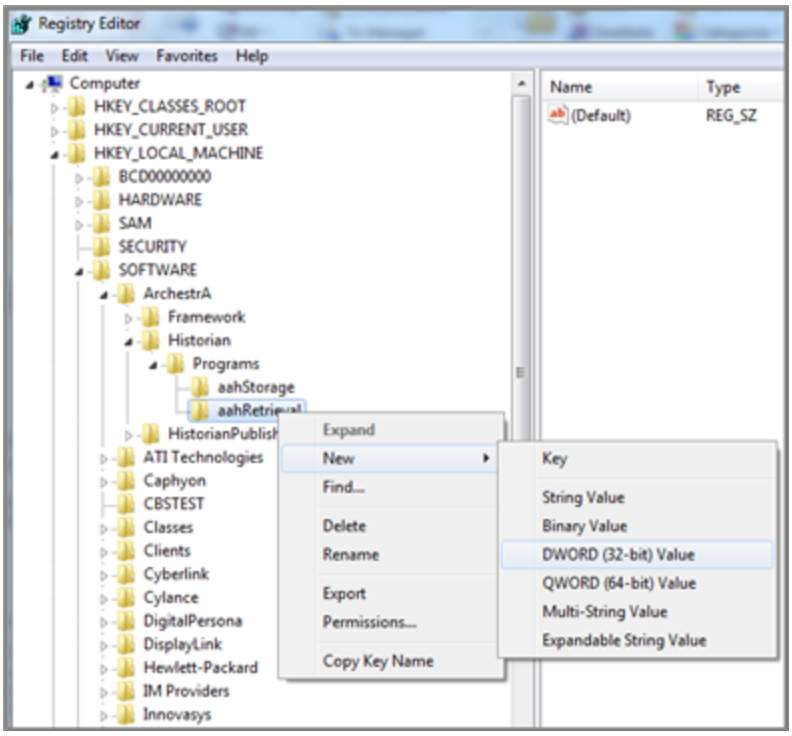
By default, gaps in data are charted (for example, by Trend and InSight) as straight lines. If you want data gaps to display as empty, you can change the Windows Registry settings.

To change Windows Registry settings to show block gaps

1. From the **Start** menu, run *RegEdit*.
2. Navigate to:

   HKEY_LOCAL_MACHINE -> SOFTWARE -> ArchestrA -> Historian -> Programs -> aahRetrieval
3. Create a DWORD Value called *UseBlockGaps* and set it to "1".
   a. Right-click **aahRetrieval**, select **New**, and then select **DWORD**.

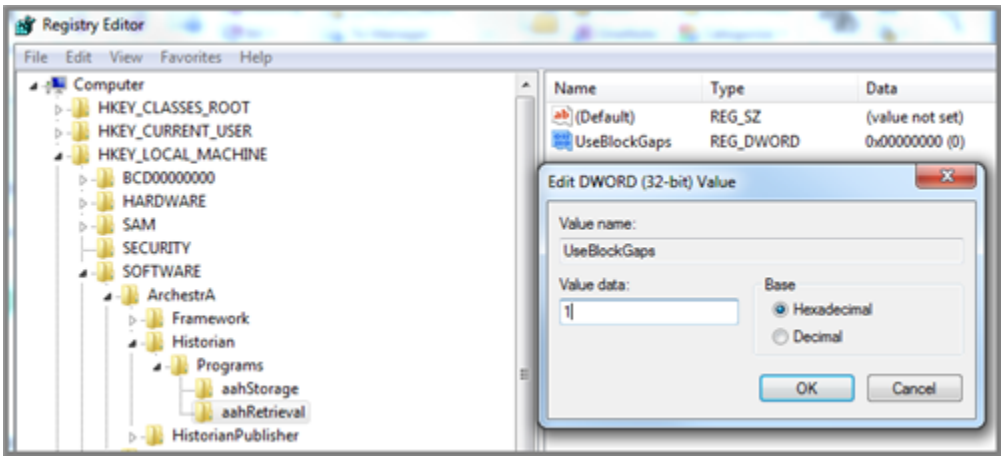
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



b. Type "UseBlockGaps" as the name of the new item.

c. Right-click *UseBlockGaps* and select **Modify**.

d. In **Value data**, type "1".



4. Exit RegEdit.

Note: If data is later inserted (for example, using a data update process) in a gap, the history block(s) will be created or recreated. In that case, the Block Gap option is set to OFF, and any tags that were not updated will show as flatlined data.

Viewing Storage and Auto-Summary Partitions

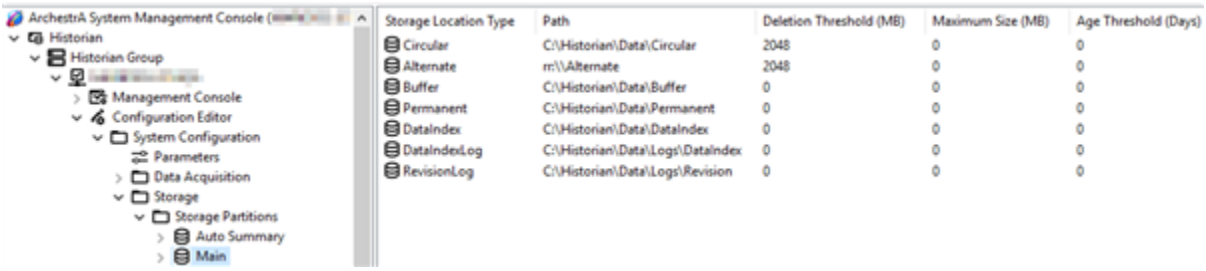
To view the storage partitions for your system

1. In the Operations Control Management Console, expand a server group and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Storage**.
3. Select **Storage Partitions**.
4. Select **Auto Summary** to see the auto-summary partition.

   Or, select **Main** to see all of the regular data partitions.

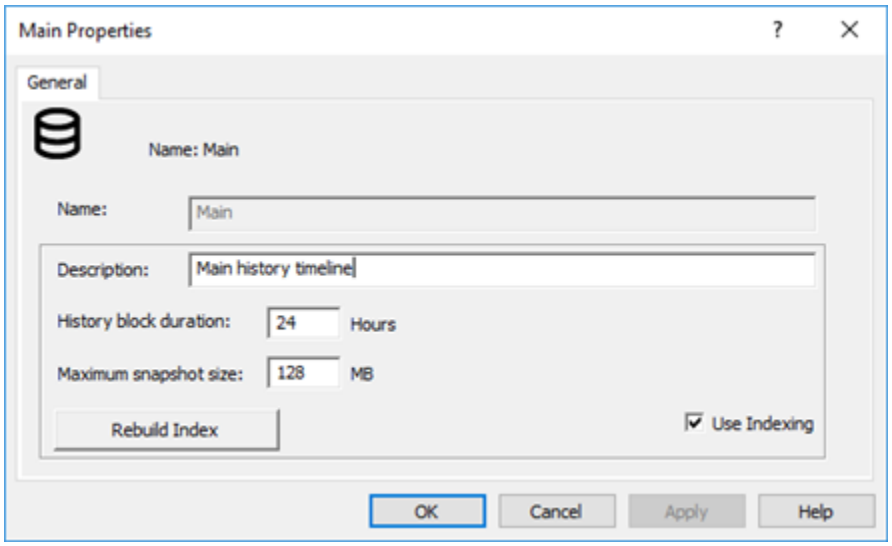


For more information about the storage location types included in each partition, see Storage Partition Locations.

Editing Storage Partition Properties

To edit storage partition properties

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Storage**.
3. Select **Storage Partitions**.
4. Right-click either **Auto Summary** or **Main**, and then select **Properties**. The **Properties** dialog box appears.



5. Update the **Description** of the storage partition if desired.
6. Enter the **History block duration**. Each history block stores all data for this duration. For more information, see History Block Notation and Creation.

   Note: Changing the history block duration does not take effect until after any previously-started blocks are complete.

7. Enter the **Maximum snapshot size**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. Indexing is enabled by default to improve retrieval performance. Select **Use Indexing** to enable indexing, or clear it to disable indexing for this storage partition.
9. Select **Rebuild Index** to manually rebuild the index.

   You must be in the aaAdministrators group to rebuild the index. If you aren't in the aaAdministrators group, you'll be prompted to provide credentials for a user with the appropriate permissions.
10. Select **OK** to save your changes.

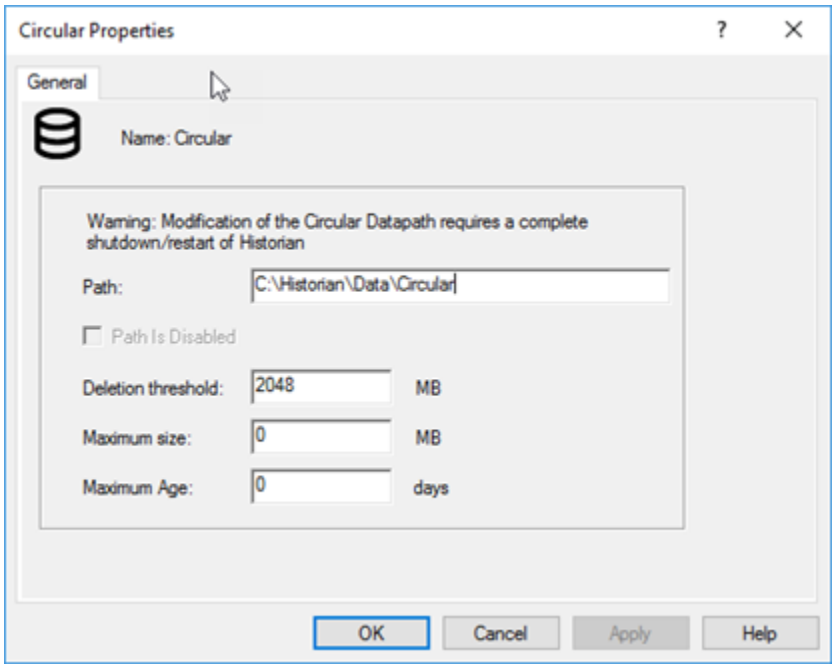
Editing Storage Location Properties

Storage locations and the history blocks they contain can be designated as circular, permanent, buffer, or alternate. Paths to these storage locations are specified when the historian is installed.

Important: Changing a storage location's path require reinitializing the system (that is, a complete shutdown and restart of the historian). Also, if a change is made to the default data paths, these directories must be manually created. The Operations Control Management Console validates the path you specify.

To edit storage location properties

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Storage**.
3. Select **Storage Partitions**, and then click either **Auto Summary** or **Main**. All defined storage locations appear in the details pane.
4. Right-click a storage location, and then click **Properties**. The **Properties** dialog box appears.



5. In the **Path** box, type the path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian Server or configured to be on a different internal hard drive. If you use a network location, the InSQLConfiguration virtual service account

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

must have full access to the location. You can grant that access by using the credentials of the computer account in the format <domain_name>\<computer_name>$. The locations must be specified using UNC notation. Mapped drives are not supported. If the path you specify does not currently exist, it is created.

Note: The paths to the storage areas are relative to the computer on which the historian is running. If you are running Operations Control Management Console on a separate network computer than the historian, the paths may not be same.

6. To disable the use of this path, select **Path is Disabled**. This option is not available for the circular storage location.
7. In the **Deletion Threshold** box, type the minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block (before any compression) by 1.5 to determine the minimum threshold.
8. In the **Maximum Size** box, type the limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used.
9. In the **Maximum Age** box, type the age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied.

   Note: The Deletion Threshold, Maximum Size, and **Maximum Age** options are unavailable for the permanent and buffer storage areas.
10. Click **OK**.

Viewing History Blocks

You can view details, such as the start and end time, for history blocks.

To view history block information

1. In the Operations Control Management Console tree, expand a server group and then expand a server.
2. Expand **Management Console** and then select **History Blocks**. The history block information appears in the details pane.

Start Time	End Time	Location	Duration
1/24/2022 12:00:00 AM	1/25/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs
1/23/2022 12:00:00 AM	1/24/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs
1/22/2022 12:00:00 AM	1/23/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs
1/21/2022 12:00:00 AM	1/22/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs
1/20/2022 12:00:00 AM	1/21/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs
1/19/2022 12:00:00 AM	1/20/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs
1/18/2022 12:00:00 AM	1/19/2022 12:00:00 AM	C:\Historian\Data\Circular\A2...	24 hrs 0 mins 0 secs

Column descriptions are as follows:

Start Time

The starting timestamp for the history block.

End Time

The ending timestamp for the history block.

Location

The path to the storage location. The circular storage location must be a local drive on the server, and the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the historian server or configured to be on a different internal hard drive. If you use a network location, the historian computer's Configuration service account should have full access to that network path. The locations must be specified using UNC notation. Mapped drives are not supported.

Duration

The time span for the history block.

TimeZone

The time zone of the history block.

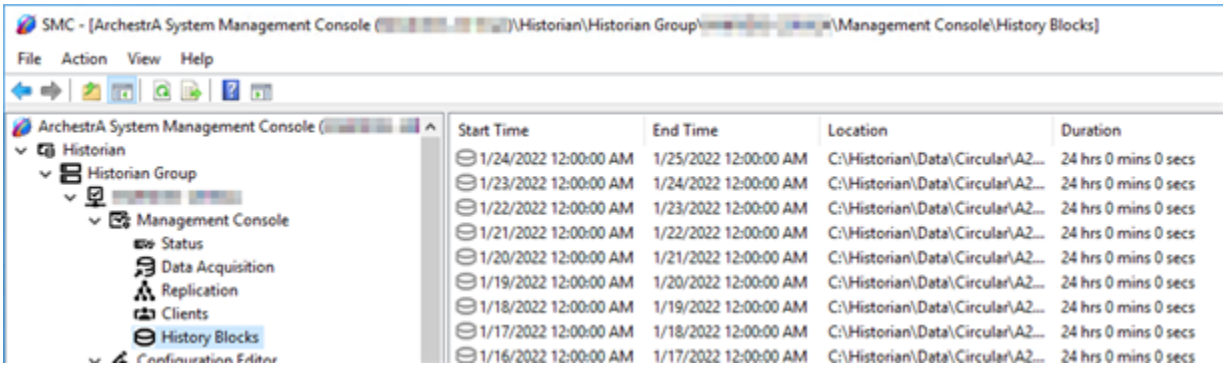
UTC Bias

The time offset from Coordinated Universal Time.

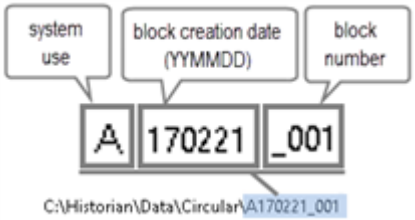
The data shown in the details pane is not automatically refreshed. To refresh the list from the history block information held by the Configuration Manager, right-click **History Blocks** in the console tree and then click **Refresh**. In most cases, this type of refresh is all that is needed.

History Block Notation and Creation

Each history block is contained in a single subdirectory in the circular storage directory.



The subdirectory name includes the date stamp of the AVEVA Historian computer at the time the block was created.



For example, this is a typical history block name. It has three parts:

- "A" is used by the system.
- "170221" matches the timestamp of the data it contains.
- "_001" is the numerical suffix that identifies this history block as the first block created that day. The block number increments if there are multiple blocks created on the same day.

A new history block is created when corresponding data is received that day. After that, new history blocks are

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

automatically created with a time duration specified for that storage partition. .

History blocks can be created for data with timestamps in the future or the past.

Changing the history block time span does not take effect until after any previously-started blocks complete, even if a previously-started block is holding data with timestamps in the future. For example, if the time span is changed from "daily" to "hourly", the first hourly block will be for 12:00 AM to 1:00 AM on the following day.

Automatic Deletion of History Blocks

History blocks in the circular and alternate storage locations may be automatically deleted to make room for new history blocks. Whether or not the blocks are deleted is determined by the minimum threshold and the maximum size and/or age specified for the storage location.

The history block management system will check for available space in the circular and alternate locations if it detects any changes made by other subsystems or the user in controlled directories.

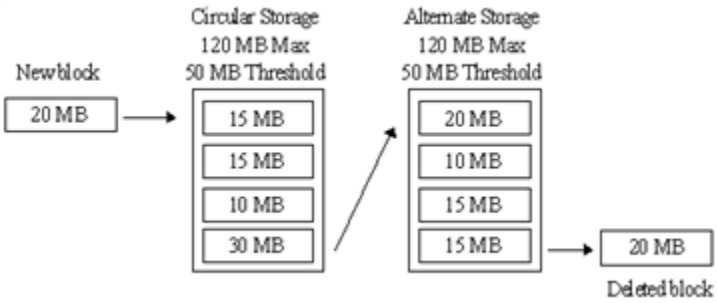
The history block management system computes the sum of the sizes of all history blocks (including the current one) in the circular storage location and determines if there is enough space on the drive to hold all of the blocks.

If the space available on the storage location drive is below a certain threshold, the Storage subsystem will delete enough of the oldest history blocks to bring the available disk drive space back to a positive value and then move the new history block in.

If an alternate storage location exists, the older block(s) will be moved there instead of being deleted. The alternate storage location functions exactly like the circular storage location. However, when the blocks exceed the set limits (minimum threshold, maximum size, or maximum age), the oldest blocks will be deleted from disk.

To avoid this loss of data, it is important that the system administrator regularly monitors the disk availability and periodically backs up old blocks to long term storage before they are deleted.

For example, a history block is stored in the circular storage location. The maximum size of the disk drive for circular storage is 120 MB. In addition to the circular storage location, an alternate location with a maximum disk drive size of 120 MB is defined. For both locations, the minimum threshold value is 50 MB. Essentially, this means that there is 70 MB of actual storage space.



Note: The sizes in this example are purposely small; the disk drives for storage locations should be much larger.

You should typically set the minimum threshold to a value that is 3 times larger than the size of the biggest history block. This will provide the history block management system enough time to copy oldest history block from the circular location to the alternate, and then delete block from the circular location.

If you monitor the disk drive space available in the circular or alternate storage location over time, the value will fluctuate between the threshold value and the maximum size of the location, with sharp increases when blocks are moved out. While the system is moving a block(s) out, the space available will dip just below the threshold value before the increase.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If the maximum threshold is reached before the age of the block reaches the specified limit, the block is moved or deleted. A block will be moved or deleted within one history block duration of it reaching the age limit. If, for any reason, the system is unable to move a block that is past the age limit, the block will not be deleted until the size or space limit is reached.

Backing Up History Blocks

It is highly recommended that you back up the history blocks to long-term storage media to avoid data loss due to media failure. You can perform backups using the Windows Backup utility or other backup tool. See the Microsoft Windows documentation for details.

About VSS-Aware Backups

The storage and event storage services coordinate with the Microsoft Windows Volume Shadow Copy Service (VSS) to enable consistent backups of history blocks while the Historian continues running. This means you do not need to shut down the Historian before performing a backup of history blocks.

Notes: You must shut down the Historian before restoring history blocks from backup.

Circular and alternate storage locations can be backed up independently of each other.

Backing Up Data Stored on Network Shares

If you are backing up data from a network share, VSS-aware backups are not available for this location from the Historian server.

For example, if your alternate storage location is on a network share instead of a local drive, you must manage backups of the alternate location from the server hosting the network share. Because this location is independent of the Historian server, the consistency of these backups cannot be guaranteed. See Storage Partition Locations for recommended best practices.

Adding Replication or Auto-Summary Values for a Defined Timeframe

You can use the Replication Backfill Manager to backfill the following types of historic data:

- **Auto-summary:** Beginning with AVEVA Historian 2017 (version 17.0.18000), Historian automatically creates a one-hour summary tag for every analog tag in the system. If you have historical data originally collected by Historian 2014 R2 or earlier, by default no auto-summary data will be calculated for the history blocks prior to upgrading to Historian 2017 or later. For example, if you ran Historian 2014 R2 or earlier from January 2010 until April 2020, then you migrated to Historian 2020, you'll only have auto-summary data beginning in April 2020. If you want to create auto-summary data for the prior 10 years of history blocks, use the backfill utility.
- **Replication:** If you have historical data originally collected prior to configuring replication, by default no older data will be replicated. For example, if you collected data in Historian from January 2020 and then configured replication to another Historian in April 2021 and to AVEVA Insight in November 2022, you'll only have replicated data in the other Historian beginning in April 2021 and in AVEVA Insight starting in November 2022. Use the backfill utility to replicate older data.

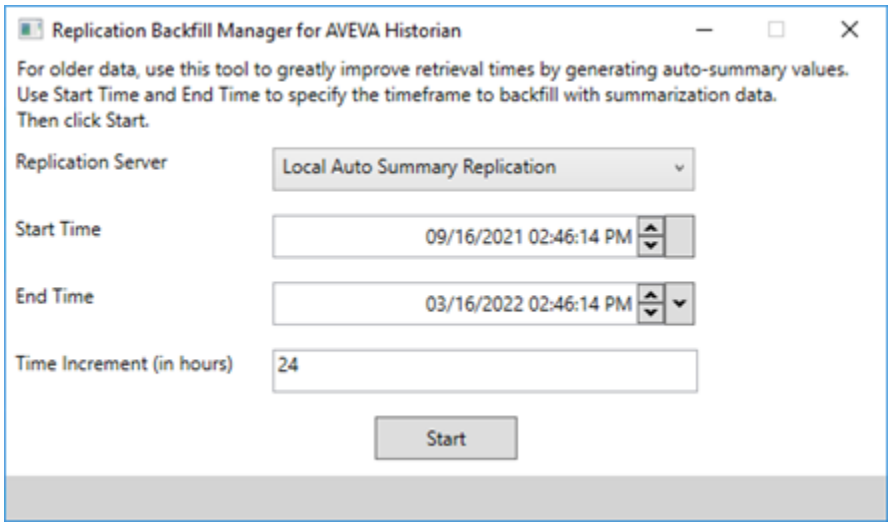
By default, the Replication Backfill Manager (aahBackfillUI.exe) is located in this folder:

```
C:\Program Files (x86)\Wonderware\Historian\x64
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To generate replication or auto-summary data

1. Open the Replication Backfill Manager (aahBackfillUI.exe).



2. Select a **Replication Server** from the list. You can backfill data to any replication server defined in the Operations Control Management Console.
3. Choose the **Start Time** and **End Time** of the period for which you want data to be generated. The default is to generate data for the previous 6 months.
4. Specify the **Time Increment (in hours)** for how much time each summary tag will summarize. The default is 24 hours per summary tag.
5. Click **Start** to begin the process.

There is no need to keep the utility open once it has initiated the backfill process. But if you reopen the utility, you can see the replication progress.

The progress bar indicates the progress of the synchronization queue requests sent to for auto-summary processing. It does not indicate that all the data has been processed and summarized in the auto-summary blocks. The processing will complete silently outside of the utility interface.

Adding History Blocks from Prior Versions to the System

You can add existing history blocks from prior versions to the system.

To add a history block

1. Shutdown and disable AVEVA Historian.
2. While being sure to avoid time intersections, copy the old history blocks to the appropriate subdirectory for the current AVEVA Historian system.
3. For old tags created by Classic Storage, be sure to set AITag=1.
4. Enable and start Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Importing, Inserting, or Updating History Data

Ways to Acquire History Data

You can import, insert, and update history data in various ways.

- **Import InTouch history data with Historian Data Importer**
  If you have existing InTouch history data, you can easily import it into the AVEVA Historian extension tables using the Historian Data Importer utility. See Importing Data from an InTouch History File.
- **Import a CSV file with Historian Data Importer**
  You can create a CSV file for the data and then use the following methods to import it. See Importing Data from CSV Files.
  - Use the Historian Data Importer to select and import the CSV file.
  - Use the Historian Data Importer to create a "watch" folder that you drop CSV files into. For more information, see Encoding Formats for Configuration Exports.
  - Drop CSV files into the predefined \DataImport watch folder.
- **Insert or update history data with Transact-SQL statements**
  With INSERT and UPDATE statements, you can insert or update history data in the AVEVA Historian extension tables. See Inserting or Updating Data with Transact-SQL Statements.
- **Rename tags with Tag Rename utility**
  With this tool, you can rename tags. See Renaming Tags.

You can track modifications to history data. For more information, see Modification Tracking for Historical Data Changes.

Guidelines for Importing, Inserting, and Updating History Data

Use the following guidelines to help you decide the best way to import or insert data into history. Each method has its strengths for certain applications, and often you will need to balance the need for speed against some limitations.

For a standard CSV import (as opposed to a "fast load" CSV import), the CSV file format and the format of the data contained within the file is very flexible. However, this flexibility requires the system to perform a large amount of processing on the data before it can be imported. Thus, there is an inverse relationship between amount of data to process and import speed. The time required to process a file is at least exponentially related to the number of values contained in the file.

Additional factors for a normal import are:

- If multiple files are to be processed at the same time, the total size of the CSV file is limited to less than 4 MB.
- The CSV file cannot contain more than 100,000 data values.
- The number of tags represented in the file cannot exceed 1024.
- Single files of up to 6 MB will be processed, provided that it does not exceed the file data and tag limits.

Performing a non-real-time insert with a Transact-SQL statement also requires a large amount of data processing.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The fastest way to insert/import data into the system is to use one of the methods that employs the real-time storage service to get the data into the history blocks. These include real-time inserts by Transact-SQL statements and "fast load" CSV imports. Real-time inserts can occur at a fairly high speed, so use this method when possible. Performing a "fast load" CSV import is also a high-speed option. To do a "fast load" import, however, the data must be in time-series order.

In general, choose to use the fast load import if:

- It is not is not feasible to perform a normal CSV import.
- You need to import very large CSV files.
- You want storage rules applied to the data you are importing. A normal CSV import does not apply storage rules; everything is stored as a delta.

Also, do not import fast load data for a tag if there is existing stored data for that tag in the same time range.

Importing History Data

You can use the Historian Data Importer (aahImport.exe) to import history data from these types of files:

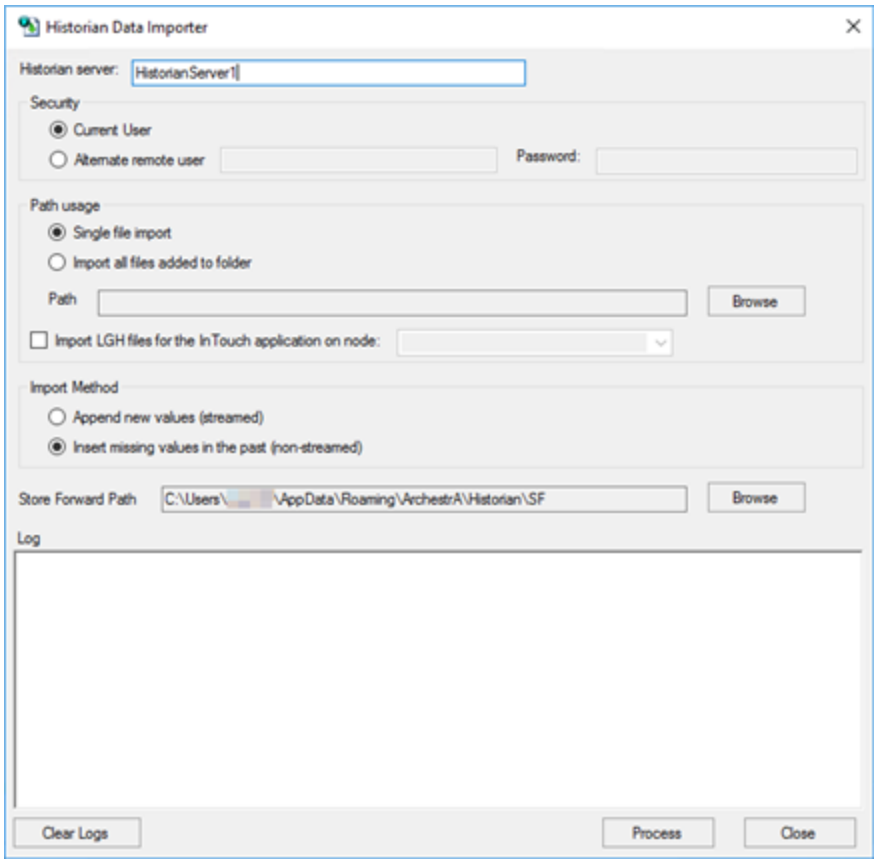
- **InTouch history (LGH) file**
  Before you perform the import, review the requirements in Importing Data from an InTouch History File.
- **CSV file**
  You must format the CSV file according to the "FastLoad" format. For more information, see Importing Data from CSV Files.

You can run the Historian Data Importer from a command prompt. For more information, see Running the Historian Data Importer from a Command Prompt.

To import history data

1. On the Windows **Start** menu, point to **Programs,** point to **AVEVA**, point to AVEVA Historian, and then click **Data Import**. The **Historian Data Importer** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. In the **Historian server** box, type the name of the historian into which you want to import history data.
3. In the **Security** area, provide a logon account that is an administrator for the AVEVA Historian.
   - Click **Current User** to use the account of the currently logged on Windows user to connect to the AVEVA Historian.
   - Click **Alternate remote user** to use a different Windows account to connect to the AVEVA Historian.
4. In the **Path usage** area, specify the location of the files to import.
   a. To import a single file, click **Single file import** and then click **Browse** to select the file.
   b. To import multiple files, click **Import all files added to folder** and then click **Browse** to select the folder that contains all of the files.

      The folder you select becomes a "watched folder." Any files you place in the folder at any time are automatically processed until you change the folder configuration or change to a single file import.
   c. If you are importing .lgh files, select the **Import LGH files for the InTouch application on node** check box and then select the name of the InTouch node (computer) from which you want to import data.

      Note: If the desired node is not listed, you need to first import the tagname database. For more information, see Importing an InTouch Data Dictionary.
5. In the **Import Method** area, specify how to integrate the data into storage.

   Append new values (streamed)

   The data in the import file must be ordered according to timestamp. If you are importing multiple files, you must start with the oldest first. That is, if there are two files containing data for the same tags, but from different time periods, the file with the oldest data must be imported first, followed by the newer data. Other data collection mechanisms (for example, SQL INSERTs, data sent from Application Server, or from a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian SDK application) may interfere with streamed imports if they ever supply values newer than the file data.

When it is processed on the historian server, streamed data is exposed through the Historian IO server, Historian IO (classic) server, the Live extension table, the History table, and other extension tables.

Insert missing values in the past (non-streamed)

The data in the import file must be ordered according to timestamp. However, if you are importing multiple files, you can import the files in any order.

Non-streamed data will not be reflected in the Historian IO server, Historian IO (classic) server, or in the Live extension table, but can be queried from the History table and other extension tables.

For more information on the different categories of data, see Data categories in the *AVEVA Historian Concepts Guide.*

6. For the **Store Forward Path** box, click **Browse** to select a folder in which processed data collects if a disconnect to the historian occurs after the data transfer to storage starts. After the connection is restored, the data transfer resumes.
7. Click **Process**.

   The results of the import are shown in the **Log** window. Errors are logged to the ArchestrA Logger. If the utility can't process a file, the file moves into a support folder that is automatically created.

   Note: The history importer does **not** report whether the data in the resultant CSV files is successfully imported into the AVEVA Historian extension tables.
8. Click **Close**.

Importing Data from an InTouch History File

The Historian Data Importer (aahImport.exe) is a stand-alone utility that allows you to import existing InTouch history data into the AVEVA Historian history blocks. InTouch history data is stored in one or more .lgh files located in the InTouch application folder.

Before you start the import, be sure that:

- You can access the LGH files (that is, files having an .lgh filename extension). The InTouch HMI software is not required to be installed on the same computer as the InTouch History Importer, and InTouch is not required to be running. The importer can import history data generated with InTouch HMI software version 7.0 or later.
- The tag definitions already exist in the AVEVA Historian database. The easiest way to make sure that you have all of the definitions is to use the AVEVA Historian Tag Importer to import the contents of the InTouch tag database. For more information, see Importing an InTouch Data Dictionary.
- You can log on to the AVEVA Historian. The InTouch History Importer requires a logon account to the historian to retrieve a list of all currently imported InTouch nodes.
- The full path to the InTouch LGH files (including the name of the actual LGH file) is not longer than 64 characters. This limitation is inherent in the underlying InTouch infrastructure used to access LGH files. If the path is longer than 64 characters, the importer produces an error message and prevents the import from continuing. If necessary, use a mapped drive or a drive substitution to shorten the name of the path.
- The InTouch application is not currently storing data to the LGH file that you plan to import.
- You change the value of the *AllowOriginals* system parameter to 1. This allows you to insert original data for I/O Servers. For more information on editing system parameters, see Editing System Parameters.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- The data you want to import does not interfere with data in the current history block for the same tags. For example, suppose you import tag definitions from an InTouch application and are currently storing the tag values received from the I/O Server in the historian. If you attempt to import existing InTouch data for these same tags, and the timestamps of the data to be imported fall within the current history block, the import may produce unexpected results. Wait until the next history block is created before attempting to import existing InTouch data.
- If you have a large amount of data to import, process the data in batches, starting from the most recent data to the oldest data. For example, you want to import one year's worth of InTouch history. Divide up the data so that one month of data is included in a batch. When you import the most recent batch, the utility automatically starts with the most recent block and then proceeds backwards. Then, import the second-most recent batch, and so on.

The fast load import mechanism used by the InTouch History Importer is intended for importing data into history for periods where no data for the tags currently exists. For delta stored tags, importing into a region where data already exists results in the addition of the new points. For cyclically stored values, however, the new points are imported on top of the existing cyclic values. For more information on fast load imports, see About Fast Load CSV File Imports.

Importing Data from CSV Files

You can import data into the history blocks, as long as the data is formatted according to a specific comma-separated values (CSV) format. The basic steps for importing data are:

1. Configure the data import folder, which is where you will put your formatted CSV files.

   For more information, see Predefined CSV File Import Folders.
2. Add tag definitions to the AVEVA Historian database for all data values you plan to import. Importing data for a tag that is not defined results in an error. If you are importing legacy InTouch data, you can use the Tag Importer to import a tagname database, which contains the tag definitions. Otherwise, you must manually add the tag definitions.

   For information on importing an InTouch tagname database, see Importing an InTouch Data Dictionary.

   For information on adding tag definitions manually, see Defining Tags.
3. Determine the type of import, either normal or "fast load."

   For more information, see About Normal CSV File Imports and About Fast Load CSV File Imports.
4. Determine if you want to insert original data for I/O Server tags. By default, the system does not insert original data for I/O Server tags by a CSV file. However, you can change this setting by changing the value of the AllowOriginals system parameter to 1.

   For more information on editing system parameters, see Editing System Parameters.
5. Format the CSV file according to the import type.

   For more information, see Formatting the CSV File for a Normal Import or Formatting the CSV File for a Fast Load Import.
6. Place the file into the appropriate data import folder, where it is automatically processed by the system.

   For more information, see Copying a CSV File into an Import Folder.

For a comparison of the various import methods, see Guidelines for Importing, Inserting, and Updating History Data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Predefined CSV File Import Folders

By default, import folders are created in the main AVEVA Historian data folder when the product is installed. For example, if you specified D:\Historian\DATA\Circular as the circular data folder, the CSV data import folder is D:\Historian\DATA\DataImport,.

Important: If you leave the data import path at the default location (on the drive hosting the circular data folder), placing large CSV files in a data import folder may prompt the AVEVA Historian disk management subsystem to immediately start moving or deleting history blocks to maintain the configured amount of required free space on the disk. It is highly recommended to change the data import folder to a different drive than your circular storage.

The different import folders are described in the following table:

Import Folder	Description
\DataImport	Used for normal CSV import files.
\FastLoad	Used for "fast load" CSV import files. Files in this folder are processed one at a time, in the order that they appear in Windows Explorer as you are viewing the folder contents.
\Manual	Used by MDAS for tags. If the amount of data spans multiple 64 KB chunks, files are collected in the \Support subdirectory until all of the data is received. The files are then copied to the \Manual folder for inclusion into history.

To manually change the predefined import folder

1. Use Windows Explorer to create the new folder.

   Note: Be sure that you maintain the \Manual\Support subfolder and optional \FastLoad subfolder. You cannot change the name of the \FastLoad or the \Support folder to another name.

2. Edit the DataImportPath system parameter to specify the new data import folder.

   For more information on editing system parameters, see Editing System Parameters.

3. Restart the AVEVA Historian.

About Normal CSV File Imports

Use the normal import mechanism if you primarily want to modify a small amount of existing data stored in the AVEVA Historian or store a small amount of new values. The insert of an entire CSV file results in a single new version of the data. If an inserted data point falls exactly on an existing timestamp, the data value is added to history. The existing data is maintained in history.

For guidelines on using this import method versus other import methods, see Guidelines for Importing, Inserting, and Updating History Data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About Fast Load CSV File Imports

Using the "fast load" CSV import mechanism, you can import original data very quickly, using essentially the same CSV file format as for a normal import, with some modifications.

A "fast load" import is much faster than a normal CSV import. For example, a CSV file that is 4 MB imports approximately 100 times faster. For larger files, the speed improvement gets substantially better. Also, there are no restrictions on the size of the file to import, or the number of tags or data values in the file. However, the data that is contained in the CSV file for a fast load import **must** be formatted in time sequential order. It is this ordering that allows the system to process a fast load CSV file more quickly than a normal CSV file.

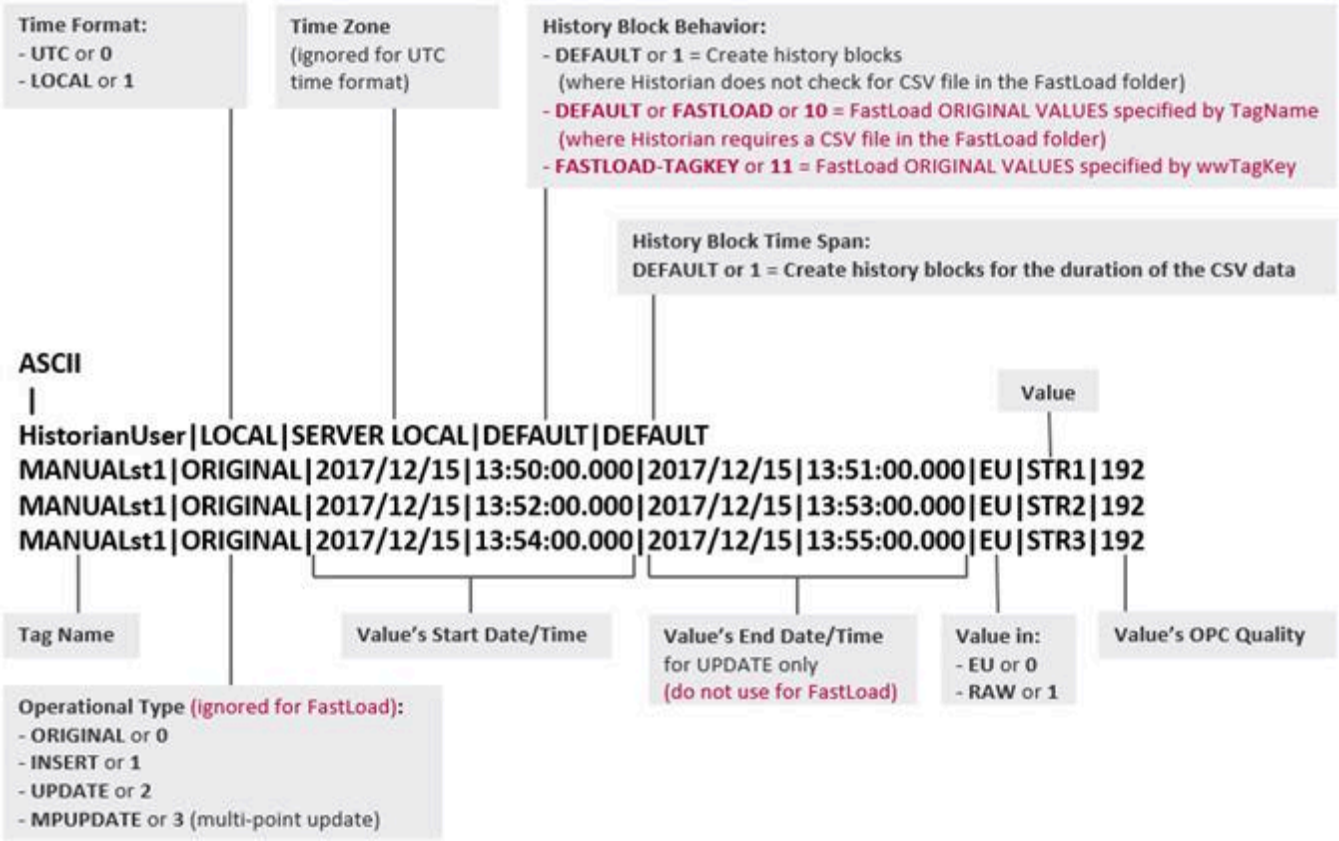
Put a formatted CSV file into a special \FastLoad import folder.

A fast load import can only be applied where there is no existing stored data.

For guidelines on using this import method versus other import methods, see Guidelines for Importing, Inserting, and Updating History Data.

General File Format for a CSV Import

The CSV file format for imports is as follows.



Formatting the CSV File for a Normal Import

To import external data into the history blocks, you must format your data according to the CSV file format as outlined in the following table. For a general illustration of the format, see General File Format for a CSV Import.

You can name the CSV file anything you want. For the format, note that:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Only one operation type per line is allowed.
- Multiple records per line of the same operation type is allowed.
- A multipoint update is a sequence of updates where the beginning of an update period is the end of the previous update. A multipoint update is faster than a simple sequence of inserts because a single version is used for all values. Use a multipoint update to mask underlying data with a new set of values for the specified time period.
  - Fields 3 and 4 of the values are used in single point update only and must be excluded from the record for a multipoint update. A single point update refers to the situation when an update value is assigned to a time period specified by the start date/time and end date/time. A multipoint update can replace a single or multiple previous points. It represents, like a single point update, a span of time that starts with the current row date/time and ending at the next row date/time. The value specified in each record is held as the latest value until the next record. The last record is ignored in a multipoint update.
  - The last record (time wise) will indicate the end of the previous update period. The value will be ignored.

If two multipoint update CSV files for the same tag are simultaneously copied to the \DataImport directory, the update spans across the total time for the two files. A query returning latest data hides (masks) the original version of the data from the end of the first file to the start of the second file.

For example, if the update in one file ranges from 00:00:00 to 00:05:00, and the other ranges from 00:10:00 to 00:15:00, the result is an update starting at 00:00:00 and ending at 00:15:00 ("latest"); the original data from 00:05:00 to 00:10:00 is masked as "original" data. No data is lost. To view either data from a query, use the wwVersion column to specify either "original" or "latest." By default, the latest data is shown. To prevent the masking of the original data, process the CSV files one at a time.

It is recommended not to use both inserts and original inserts for the same tag in the same file or files processed together.

When configuring the scaling setting (field 6), keep in mind that the data conversion to engineering units (a setting of 0) is performed before the value is stored. The reverse of the scaling formula for the tag is used to convert the data before storage. During retrieval, the scaling formula is applied so that the original inserted values are returned. For integer type tags, if the value after the conversion is not an integer value, it is rounded off. The rounding off can change the value to be exactly the same as the previous value, and thus the rounded off value is not stored to disk if delta storage is used. If the tag is a real type tag, the rounding off does not occur, and all values are stored.

The value to insert can be a NULL. For more information, see Handling of NULL Values in CSV Files.

For a fast load CSV import, the end time of the current block in the block80.inf file is considered to be the current time, not the current system time stamp. The end time in the .inf file is updated by the storage subsystem every 20 seconds.

Example CSV Files for a Normal Import

The following is an example of an insert of data values for a single tag, "ReactTemp." The pipe (|) is used as a delimiter.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
ASCII
|
PatrickP|1|Pacific Daylight Time|1|1
ReactTemp|1|2001/05/19|16:00:00.500|1|256.0|192
ReactTemp|1|2001/05/19|16:00:02.500|1|261.0|192
ReactTemp|1|2001/05/19|16:00:03.500|1|266.0|192
ReactTemp|1|2001/05/19|16:00:04.500|1|271.0|192
ReactTemp|1|2001/05/19|16:00:05.500|1|276.0|192
ReactTemp|1|2001/05/19|16:00:06.500|1|281.0|192
ReactTemp|1|2001/05/19|16:00:08.500|1|286.0|192
ReactTemp|1|2001/05/19|16:00:09.500|1|291.0|192
ReactTemp|1|2001/05/19|16:00:10.500|1|296.0|192
ReactTemp|1|2001/05/19|16:00:11.500|1|2101.0|192
ReactTemp|1|2001/05/19|16:00:12.500|1|2106.0|192
ReactTemp|1|2001/05/19|16:00:14.500|1|2111.0|192
ReactTemp|1|2001/05/19|16:00:15.500|1|2116.0|192
ReactTemp|1|2001/05/19|16:00:16.500|1|2121.0|192
```

The following is an example of an update of data values for a single tag, "Man1." A comma (,) is used as a delimiter.

```
ASCII
,
KristenC,1,Pacific Daylight Time,1,1
MAN1,2,2001/04/24,16:00:00.000,2001/04/24,16:05:00.000,1,1111,192
MAN1,2,2001/04/24,16:00:10.000,2001/04/24,16:15:00.000,1,2222,192
```

The following is an example of a multipoint update of data values for a single tag. A comma (,) is used as a delimiter. The last value is ignored.

```
ASCII
,
BenjaminY,1,Pacific Standard Time,1,1
MANUAL32SI3,3,2002/01/28,00:05:00.500,1,81000,192
MANUAL32SI3,3,2002/01/28,00:05:03.500,1,81010,192
MANUAL32SI3,3,2002/01/28,00:05:06.500,1,81020,192
MANUAL32SI3,3,2002/01/28,00:05:09.500,1,81030,192
MANUAL32SI3,3,2002/01/28,00:05:12.500,1,81040,192
MANUAL32SI3,3,2002/01/28,00:05:15.500,1,81050,192
MANUAL32SI3,3,2002/01/28,00:05:18.500,1,81060,192
MANUAL32SI3,3,2002/01/28,00:05:21.500,1,81070,192
MANUAL32SI3,3,2002/01/28,00:05:24.500,1,81080,192
MANUAL32SI3,3,2002/01/28,00:05:27.500,1,81090,192
MANUAL32SI3,3,2002/01/28,00:05:30.500,1,81100,192
MANUAL32SI3,3,2002/01/28,00:05:33.500,1,81110,192
MANUAL32SI3,3,2002/01/28,00:05:36.500,1,81120,192
MANUAL32SI3,3,2002/01/28,00:05:39.500,1,81130,192
MANUAL32SI3,3,2002/01/28,00:05:42.500,1,81140,192
MANUAL32SI3,3,2002/01/28,00:05:45.500,1,81150,192
MANUAL32SI3,3,2002/01/28,00:05:48.500,1,81160,192
MANUAL32SI3,3,2002/01/28,00:05:51.500,1,81170,192
MANUAL32SI3,3,2002/01/28,00:05:54.500,1,81180,192
MANUAL32SI3,3,2002/01/28,00:05:57.500,1,81190,192
MANUAL32SI3,3,2002/01/28,00:06:00.500,1,81200,192
```

Formatting the CSV File for a Fast Load Import

Important: The data points **must** be sorted in time sequential order for a successful "fast load" import.

The format for the fast load CSV file is essentially the same as the normal format, with a few exceptions.

For a general illustration of the CSV format, see General File Format for a CSV Import. For a detailed description of the normal format, see Formatting the CSV File for a Normal Import.

The fast load format exceptions are:

- All data in the file is treated as original data. The Operation Type field in the file header is ignored.
- The actual data values in the file **must** be in time sequential order, starting at the top of the file. This is the most important requirement. Values that have out-of-sequence timestamps are ignored. If a data value in the file has a timestamp that is earlier than the timestamp in the previous line in the file, the data value is discarded, regardless of whether it belongs to the same tag or a different tag.
- The file should contain only one data value per line.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example CSV Files for a Fast Load Import

The following is an example of an insert of original data values for a single tag, "Manual_01." The pipe (|) is used as a delimiter.

```
ASCII
|
RolandoM|1|Server Local|10|0
Manual_01|0|2004/12/08|04:00:17.000|0|22|192
Manual_01|0|2004/12/08|04:01:17.000|0|23|192
Manual_01|0|2004/12/08|04:02:17.000|0|24|192
Manual_01|0|2004/12/08|04:03:17.000|0|25|192
Manual_01|0|2004/12/08|04:04:17.000|0|26|192
Manual_01|0|2004/12/08|04:05:17.000|0|27|192
Manual_01|0|2004/12/08|04:06:17.000|0|28|192
Manual_01|0|2004/12/08|04:07:17.000|0|29|192
Manual_01|0|2004/12/08|04:08:17.000|0|30|192
Manual_01|0|2004/12/08|04:09:17.000|0|31|192
```

The following example shows an insert of original data values for a single tag, identified by a wwTagKey of 777. A comma (,) is used as a delimiter. The file is saved as UNICODE, where every character is represented by two bytes.

```
UNICODE
,
MikeA,1,Server Local,11,2
777,0,2004/12/09,12:05:24.000,0,100,192
777,0,2004/12/09,12:48:36.000,0,101,192
777,0,2004/12/09,13:31:48.000,0,102,192
777,0,2004/12/09,14:15:00.000,0,103,192
777,0,2004/12/09,14:58:12.000,0,104,192
777,0,2004/12/09,15:41:24.000,0,105,192
777,0,2004/12/09,16:24:36.000,0,106,192
777,0,2004/12/09,17:07:48.000,0,107,192
777,0,2004/12/09,17:51:00.000,0,108,192
```

Handling of NULL Values in CSV Files

The value to insert can be a NULL.

If the OPC Quality in the CSV file is between 0 to 63, then:

- The NULL value is stored.
- The Quality Detail is set to 249 (not a number).
- The OPC Quality is what was specified in the CSV file.

If the OPC Quality in the CSV file is greater than 63, then:

- The value that was specified in the CSV file is stored.
- The Quality Detail is set to 192 (unless the value specified is NULL in the CSV file, in which case the Quality Detail is set to 249).
- The OPC Quality is what was specified in the CSV file (unless the value specified is NULL in the CSV file).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If the value is not NULL, but the OPC Quality is less than 63, then:

- A NULL value is stored.
- The Quality Detail is set to 249 (not a number).
- The OPC Quality is what was specified in the CSV file.

Copying a CSV File into an Import Folder

After you copy one or more CSV files to an import folder, the AVEVA Historian attempts to read the file(s) only one time. If the read is successful, the data is automatically converted and merged in with the appropriate history block according to the date range provided in the CSV file. The CSV file is then deleted from the directory. If an error occurs during the import, the CSV file is moved to the \Support folder. A message is also posted in the error log.

At any given time, the manual storage service processes either fast load CSV files or normal CSV files; the two types are not processed concurrently.

Do not attempt a CSV file import if the manual storage process is initializing, as indicated by the icon in the status display of the Management Console.

Be sure that the access for the history blocks is read/write (if you copy them from a CD or DVD, for example, they are read-only).

Running the Historian Data Importer from a Command Prompt

You can run the Historian Data Importer in console mode from a command prompt to process CSV files.

The following table describes the command line arguments when running the utility in console mode. All the arguments are case-insensitive. The utility returns a 0 or a non-zero integral value, where 0 indicates success and a non-zero value indicates an error code.

Argument	Description	Optional?	Example
-? or -help	Shows the help.	Yes	aahimport.exe -? aahimport.exe -help
-f	Name of a single CSV file to process.	No	aahimport.exe -f "C:\CSVFiles\201312021201.csv"
-h	Name of the historian node on which the utility is running.	No	aahimport.exe -h HistNode
-e	This is the file encoding type to use. Valid values are: ASCII, UNICODE, or UTF-8.	Yes	aahimport.exe -e ASCII

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description	Optional?	Example
-fs	Specifies that data should be processed as streamed values.	Yes	aahimport.exe -f "C:\CSVFiles\201312021201.csv" -fs
-u and -p	Administrative security credentials for logging on to the Historian computer.	Yes	aahimport.exe -u "username" -p "password"
-fw	Specifies for the utility to "watch" the folder and process any valid CSV file that is placed there.	Yes	aahimport.exe -fw "C:\CSVFiles" -h HistNode

Inserting or Updating Data with Transact-SQL Statements

Using the Transact-SQL INSERT and UPDATE statements, you can insert or update data in the AnalogHistory, DiscreteHistory, StringHistory, and History tables (Value and OPCQuality only).

The AVEVA Historian uses the same security defined for SQL Server for inserting and updating data. However, as with a CSV file, you cannot delete any data value from storage.

If you are attempting to insert or update values whose time period spans across missing history blocks, the necessary history block(s) are recreated for the duration of the data.

For guidelines on using this import method versus other import methods, see Guidelines for Importing, Inserting, and Updating History Data.

INSERT ... VALUES Syntax

An INSERT statement with a VALUES clause is supported only if you use the four-part syntax.

Syntax

```
INSERT [INTO] {table_name | view_name} (column_list)
   VALUES ({DateTime: constant | variable},
      {TagName: constant | variable},
      {Value: constant | variable}
      [, {OPCQuality: constant | variable}]
      [, {wwTimeZone: constant | variable}]
      [, {wwVersion: constant | variable}] )
```

Using variables in the VALUES clause is permitted only with four-part naming. For more information, see "Using the Four-Part Naming Convention" in Chapter 6, "Data Retrieval Subsystem," in the AVEVA Historian Concepts Guide.

Arguments

table_name

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The name of the extension table into which you want to insert the data. Valid values are: AnalogHistory, DiscreteHistory, StringHistory or History.

view_name

The corresponding view name for an extension table. Valid values are: v_AnalogHistory, v_DiscreteHistory, v_StringHistory or v_History.

column_list

Mandatory columns are *DateTime*, *TagName* and *Value*. *OPCQuality, wwTimeZone,* and *wwVersion* are optional columns. If the *OPCQuality* column is omitted in an INSERT … VALUES statement, an *OPCQuality* value of 192 (Good) is inserted automatically. If the wwTimeZone column is omitted, the time zone of the server is assumed. The wwVersion column defaults to 'original' for non-I/O Server tags and for I/O Server tags.

Due to a restriction on the vValue (variant) column in Microsoft SQL Server, any string data inserted or updated must be done to the StringHistory table, not the History table.

The *column_list* parameter, which is optional in a regular SQL INSERT … VALUES statement, is mandatory for the AVEVA Historian INSERT … VALUES syntax.

Examples

The following examples show valid INSERT … VALUES statements using the "four-part" query syntax.

For more information on four-part queries, see "Query Syntax for the AVEVA Historian OLE DB Provider" in Chapter 6, "Data Retrieval Subsystem," in the AVEVA Historian Concepts Guide.

```
INSERT INSQL.Runtime.dbo.AnalogHistory (DateTime, TagName, Value, OPCQuality)
   VALUES ('1999-11-11 16:05:10', 'NonIOTag1', 56, 192)
INSERT INTO INSQL.Runtime.dbo.StringHistory (DateTime, TagName, Value, wwTimeZone,
wwVersion)
   VALUES ('1999-11-11 16:05:10', 'IOstring1', 'Batch 10', 'Eastern Standard Time',
   'latest')
```

You can also use the view name in place of the four-part name. For example, in the following queries, *v_History* and *v_AnalogHistory* are used instead of the four-part name *INSQL.Runtime.dbo.History* and *INSQL.Runtime.dbo.AnalogHistory*, respectively.

```
INSERT v_History (TagName, OPCQuality, Value, DateTime)
   VALUES ('NonIOtag1', 192, 56, '1999-11-11 16:05:10')
INSERT INTO v_History (TagName, DateTime, Value, OPCQuality)
   SELECT 'ManualReactTemp', DateTime, 32 + Value * 9 / 5, 192 FROM v_AnalogHistory
      WHERE TagName = 'ReactTemp'
      AND DateTime >= dateadd(mi, -50, getdate())
      AND DateTime < dateadd(mi, -10, getdate())
      AND wwRetrievalMode = 'Delta'
```

You can use SQL variables in a four-part query. For example.

```
DECLARE @Value float
DECLARE @DateTime DateTime
SET @Value = 1.2345
SET @DateTime = DateAdd(Minute, -10, GetDate())
INSERT v_History (DateTime, TagName, Value, OPCQuality)
   VALUES (@DateTime, 'NonIOTag1', @Value, 192)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using the wwVersion Parameter for INSERTs

You can use the wwVersion parameter to specify different handling for data that you are inserting. You can use the following values for wwVersion:

- REALTIME
- LATEST
- ORIGINAL

You can insert any quality detail values, but it is recommended that you use values of either 0 (BAD), 64 (Uncertain), or 192 (GOOD).

Inserting Real-time Original Data

Real-time data insertion by means of an INSERT statement is supported for all non-I/O Server tags. Data that is acquired in this manner is handled just like real-time data coming from an I/O Server tag. You insert real-time data into history by specifying REALTIME for the value of the wwVersion parameter in a Transact-SQL INSERT statement. All data inserted as real-time data is assumed to be "original" data. "Original" data is the first data value that is stored for a particular timestamp.

Real-time data is always inserted into the current history block and incorporated into the primary (real-time) data stream. This eliminates the performance overhead that is associated with inserting and/or versioning non-real-time data, thus making the inserts very efficient.

For example, you can create a client application that uses Transact-SQL INSERT statements to insert real-time data into the system.

The following restrictions apply to a real-time insert.

- You cannot insert real-time data for an I/O Server tag.
- You must have SQL Server permissions to perform an insert.

In the following example, a value is inserted for 'MyAnalogTag1' directly into the primary data stream. The timestamp for the value is determined by the result of the getdate() function:

```
INSERT INSQL.Runtime.dbo.AnalogHistory (DateTime, TagName, Value, OPCQuality, wwVersion)
   VALUES(getdate(), 'MyAnalogTag1', 10, 192, 'REALTIME')
```

You can also allow the system to timestamp the value by specifying a NULL value for the DateTime column. The timestamp is the current time of the historian. For example:

```
INSERT INSQL.Runtime.dbo.AnalogHistory (DateTime, TagName, Value, OPCQuality, wwVersion)
   VALUES(null, 'MyAnalogTag1', 10, 192, 'REALTIME')
```

Note that this is a special case that is supported by the REALTIME parameter; under normal circumstances a NULL value for the DateTime column produces an error.

Inserting Original Non-Streamed Data

You insert original data into history by specifying ORIGINAL for the value of the wwVersion parameter in a Transact-SQL INSERT statement.

You can insert original data for both I/O Server tags and non-I/O Server tags. However, to insert original data for I/O Server tags, the *AllowOriginals* system parameter must be set to 1. For more information, see Editing System

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters.

If original data is already stored in the history blocks with the same timestamps as the data you are inserting, the system retains the first set of original values and adds the second set of original values as well, resulting in multiple original values for the same timestamps. If you specify to retrieve original values, there is no way to determine the order in which the values were inserted. In a case such as this, it is better to insert revision data, if the added performance overhead is not a problem.

The following query inserts an original value for 'NonIOTag1' into history:

```
INSERT INSQL.Runtime.dbo.AnalogHistory (DateTime, TagName, Value, OPCQuality, wwVersion)
   VALUES('2002-11-11 16:05:10', 'NonIOTag1', 10, 192, 'ORIGINAL')
```

Inserting Latest Revision Data

The AVEVA Historian supports inserting the first (original) and last (latest) version of a tag value.

You insert revision data into history by specifying LATEST for the value of the wwVersion parameter in a Transact-SQL INSERT statement. This is essentially the same as performing an update, but without some of the limitations.

You can insert revision data for both I/O Server tags and non-I/O Server tags.

The following query inserts revision data for 'NonIOTag1' into history:

```
INSERT INSQL.Runtime.dbo.AnalogHistory (DateTime, TagName, Value, OPCQuality, wwVersion)
   VALUES('2002-11-11 16:05:10', 'NonIOTag1', 15, 192, 'LATEST')
```

UPDATE Syntax

The AVEVA Historian implements UPDATE only by the OPENQUERY function, not by a four-part syntax. The reason for this is the method of implementation of UPDATE in Microsoft SQL Server. If you attempt to update values using the four-part query syntax, you will receive an error.

Also, a limitation of using the OPENQUERY function is that SQL variables cannot be used in the UPDATE statement.

Updating data in history always results in a new history version, and can be performed multiple times; however, only the original or the latest version of the data is available upon retrieval.

Syntax

The syntax of the UPDATE statement in the OPENQUERY portion is:

```
SELECT * FROM OpenQuery(INSQL, 'UPDATE { table_name }
   SET
      column_name = constant [,...n]
   WHERE
      <search_condition>')
```

Arguments

table_name

The name of the extension table into which you want to update the data. Valid values are: AnalogHistory, DiscreteHistory, StringHistory or History.

column_name

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Valid values are: Value, OPCQuality. (Update of the vValue column of the History table is not supported.)

Remarks

For the `<search_condition>`, DateTime and TagName search criteria are mandatory. The DateTime criterion must refer to a time range; an update at a single time point ('DateTime=') is not supported.

Important: When updating data using the OLE DB provider, the greater than operator (>) and the less than operator (<) are always interpreted as >= and <=, respectively. For more information, see Example 2 in this section.

```
   DateTime >[=] earlier_datetime_value AND DateTime <[=] later_datetime_value
```

Similarly, TagName may refer to one or more tags:

```
   TagName = ...
```

-or-

```
   TagName [NOT] LIKE ...
```

-or-

```
   TagName IN ( ... )
```

`'TagName NOT IN (…)'` is not supported. This is similar to the capabilities of OpenQuery SELECT; 'NOT IN' syntax is also not supported here.

As with INSERT … VALUES, wwTimeZone is optional. If not specified, the time zone defaults to the time zone of the AVEVA Historian.

Important: Other types of search conditions (for example, using a condition on Value) are not supported.

Example 1

The supported UPDATE syntax is shown in the following example:

```
SELECT * FROM OPENQUERY(INSQL, 'UPDATE History SET Value = 10, OPCQuality = 192
   WHERE TagName LIKE "Line1V%"
      AND DateTime >= "1999-11-11 16:05:10"
      AND DateTime <= "1999-11-11 16:05:40" ')
```

This query sets the *Value* to 10 and the *OPCQuality* to 192 for all data values for the specified tags, when the specified DateTime criteria are met.

Example 2

For the following query, the data that is updated will include the timestamps of 2002-10-03 14:59:59 and 2002-10-03 16:00:00, respectively. Existing points at these timestamps will therefore be affected by the update.

```
SELECT * FROM OPENQUERY(INSQL, 'UPDATE History SET Value = 1, OPCQuality = 192
   WHERE TagName = "Manual_AD32SI1"
      AND DateTime > "2002-10-03 14:59:59"
      AND DateTime < "2002-10-03 16:00:00" ')
```

Renaming Tags

The Historian Tag Rename Utility lets you rename tags in AVEVA Historian while preserving the historical data associated with those tags. At rename, the old tag's data history is linked to the new tag name that you specify.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Note: For classic history blocks, use the earlier version of the Tag Rename utility. In this case, be sure to run the earlier version before you run the new Historian Tag Rename Utility.

Historian tags, along with other tags, are stored in the Runtime database in the Tag, AnalogTag, DiscreteTag, and StringTag tables. You can view these tags in the Operations Control Management Console. These tags cannot be renamed from the Historian Configuration Editor.

To view Historian tags in the Runtime database

- In the Operations Control Management Console, expand Historian, expand Historian Group, expand Configuration Editor, expand System Configuration, and then expand Tag Configuration.

The Historian also stores tag meta data, including information about tag name and tag version, in history blocks.

About Historian Tag Ownership

Although all tags are created and stored with their associated data in a historian server, each tag has an implicit "owner". That owner is the entity responsible for creation and re-creation of that tag and for sending data associated with it. Tag ownership may change over time, but at any point in time it can be clearly identified. A tag can have one of three possible owners:

- AVEVA Application Server. When an application server is deployed, a tag is created and its name is automatically generated from the object hierarchy. If, for some reason, that historian tag completely disappeared from the historian server, the AVEVA Application server could recreate that tag automatically at the next deployment.
- AVEVA Historian SDK application, if it adds a tag before sending data. As in the case with the AVEVA Application Server, that application can re-create the tag automatically if necessary.
- AVEVA Historian, if that tag is an IDAS tag or one created manually through the Configuration Editor or a stored procedure.

AVEVA Historian can become a tag owner for already-collected data if the associated AVEVA Application Server deployment or SDK application no longer exists and is no longer able to re-create or send data.

If a tag must be renamed, the first step is renaming it in the owner and then forcing the owner to re-create a tag with the new name. For AVEVA Application Server, it means making changes in the object hierarchy and redeploying. For an SDK application, it means renaming the tag in the application, restarting the application to reconnect to the historian, and then reconfiguring the current most recent version of the tag.

If, however, the Application Server no longer exists or the application is no longer used, the tag ownership is already transferred to the historian and that first step is unnecessary.

After the tag owner re-creates the tag with a new name, the new incoming data will be stored under that new tag name. However, previously collected data is still associated with the old name. To make that old data visible, you can use the Tag Rename Utility to update previous tag metadata records with the new tag name without changing the tagids of those records.

Preparing to Rename Application Server Tags

Before renaming Application Server tags with the Tag Rename Utility, be sure to do the following:

1. Complete the store-and-forward process to empty the App Engine store-and-forward folders, synchronizing

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the tag information.

2. Use the System Platform IDE to undeploy any Galaxies that are already deployed.
3. Rename the tags using the System Platform IDE.

   For more information on steps 1-3, see the Application Server documentation.
4. Stop, shutdown, and disable the Historian.

   For more information, see Starting and Stopping AVEVA Historian.
5. Use the Tag Rename Utility to rename the tags in Historian to match the new names given in step 3.

   For more information, see Renaming Tags Using the Tag Rename Utility.
6. Restart the Historian.
7. Redeploy the galaxies using the System Platform IDE.

Renaming Tags Using the Tag Rename Utility

Important: After connecting to the database, the Tag Rename Utility will check for the Runtime database version and will proceed only if you are using version 11.5 or higher.

To rename tags:

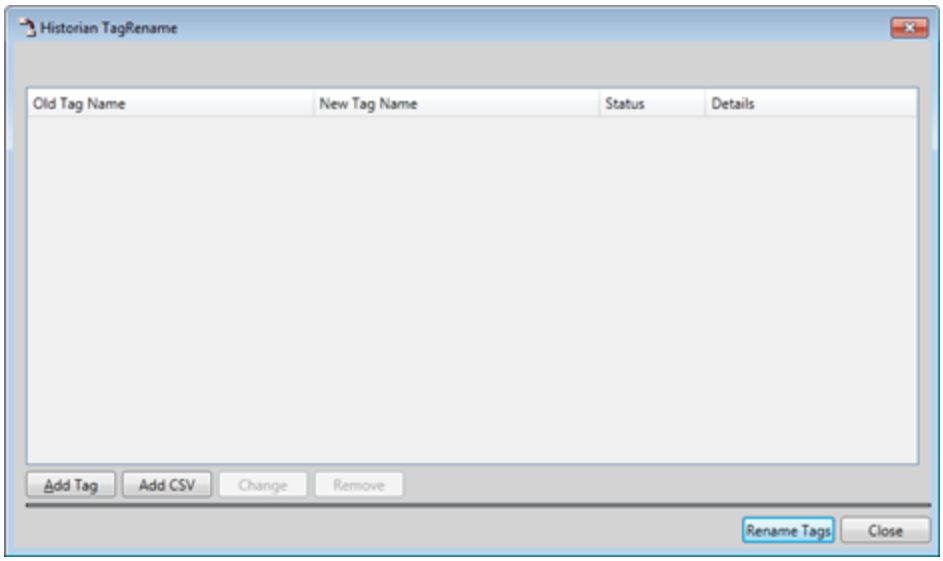
1. Open the Tag Rename Utility.
2. Provide authentication information in either of these ways:
   - Use Windows authentication. With this method, you are not required to provide a username and password. You will be connected with the credentials that you have already used to log onto the computer.
   - Use SQL Server authentication. With this method, you must provide the username and password for the SQL Server.

Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.

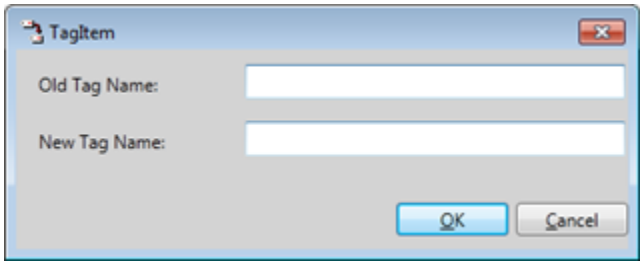
Note: For either authentication method to work, you must have administrator privileges on the SQL Server Runtime database.

3. Click **Connect** to connect to the Historian Runtime database.
4. Click **Add Tag** to specify the tag you want to rename.

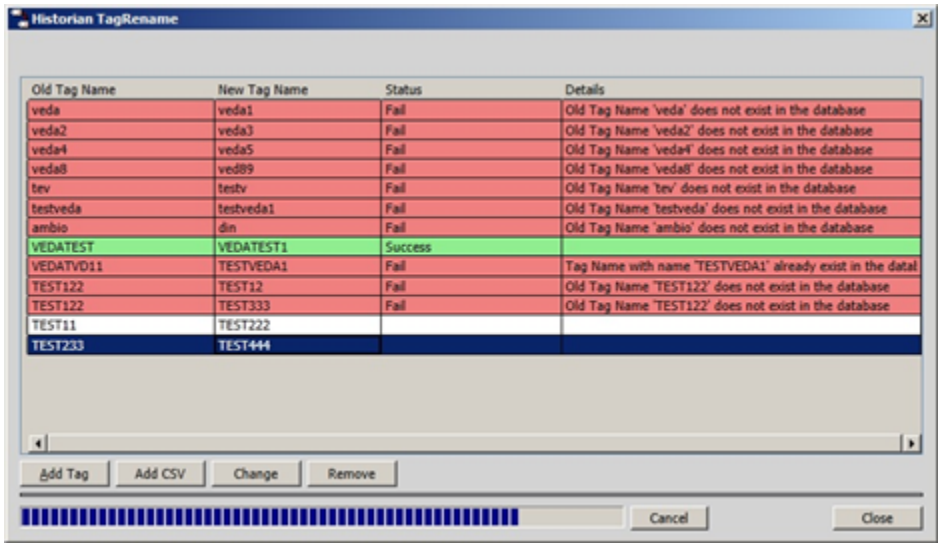
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In **Old Tag Name**, type the current tag name.



6. In **New Tag Name**, type a new name for the tag. The tag names may contain letters, numbers, and the special characters "_" and "$", but no spaces. Click **OK**.
7. Repeat steps 4 through 6 for each tag you want to rename.
8. Click **Rename Tags**. A screen like this displays your tags:



You may see both red and green outcomes as a result of renaming the tags.

- Green indicates that the Tag Rename Utility has successfully renamed the tag in the Runtime database.
- Red indicates that the rename was not successful. This typically happens because the connection failed,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the old tag did not exist, or the new tag name was a duplicate. The **Details** column shows the reasons for any failure.

Note: You can also rename several tags at a time by clicking **Add CSV**, and then specifying a .CSV file that contains an old tag name and a new tag name consecutively in each row.

9. After using the Tag Rename Utility, enable and start the Historian.

   For more information, see Starting and Stopping AVEVA Historian.

   Your tags will be updated to the new tag name and contain the history associated with the old tag.

10. If applicable, open the System Platform IDE and deploy the Galaxies.

    For more information, see the Application Server documentation.

Updating Replicated Data

Updating replicated data is very similar to updating non-replicated data with a few exceptions.

Always update data at the tier-1 historian. The replication process then propagates the information from the tier-1 historian to the tier-2 historian. In other words, do not attempt to update Tier-2 data. Remember that there is some latency with propagating updated information, typically from a few seconds to a few minutes. However, bandwidth limitations and the quantity of data to be replicated may also cause delays in propagation. After propagation, all replicated tag values and their OPC qualities are identical on both tiers.

If any data modification is performed on the tier-2 historian, including deletion of history blocks and if the data for the same tag is then modified on the tier-1 historian and an overlapping time interval, then the data modification on the tier-2 historian may be overwritten during replication. The replication process also creates new "patch" history blocks on the tier-2 historian when necessary.

If some history blocks are deleted on the tier-1 historian manually or due to scheduled disk management, the tier-2 values remain unchanged.

For more information on tag replication and replication servers, see Managing and Configuring Replication.

Managing and Configuring Replication

About Replication

With AVEVA Historian, you can replicate tag information from one historian to another. This creates a "tiered" relationship between historians. That is, the tier-1 historian send its replicated data to a tier-2 historian.

Historian also supports multi-tier replication. Data originating at tier 1 can be replicated to tier 2, then again to tier 3, and so on.

AVEVA Historian can replicate process data as well as alarms and events.

AVEVA Historian supports two types of replication:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Simple replication**
  You can replicate tag data directly using simple replication, where the tag information is replicated directly to the next-tier historian (for example, from tier 1 to tier 2). For simple replication, every value for a tag is copied.
- **Summary replication**
  You can also set up summary tags that receive a summarized version of the tag data.

Replication Schedules

Each real-time summary has a specified schedule, by which the summary is calculated and sent for storage to the next-tier historian with the timestamp of the cycle.

There are two types of replication schedules:

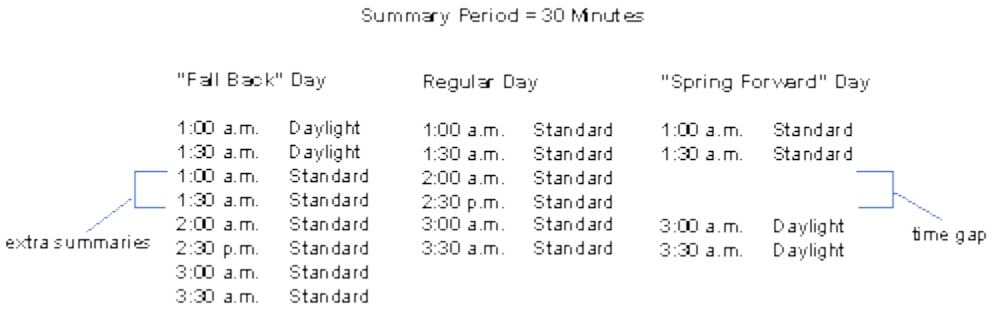
- **Periodic replication schedules**
  You can configure a summary to replicate based on an cycle such as 1 minute, 5 minutes, 1 hour, 1 day, and so on. The cycle boundaries are calculated starting at midnight, lower-tier (originating) server local time, and continue in fixed time increments. The time between cycles is constant within a day even through a daylight savings change. Note that the last cycle in a day may be shorter to force replication at midnight. The calculation cycle starts at midnight. For example, a 13-minute cycle is stored at 12:00 a.m., 12:13, 12:26, ... 11:27 p.m., 11:40, 11:53, and then again at 12:00 a.m.
- **Custom replication schedules**
  Custom schedules force replication cycles to occur at fixed times of the day in lower-tier (originating) server local time. You choose the fixed times of day.

Replication Schedules and Daylight Savings Time

Daylight Savings Time affects replication schedules that are triggered according to a time period, such as every hour, every thirty minutes, and so on. Replication schedules that are triggered at a fixed time that you specify are not affected.

In the following examples, the time change occurs at 2:00 a.m.

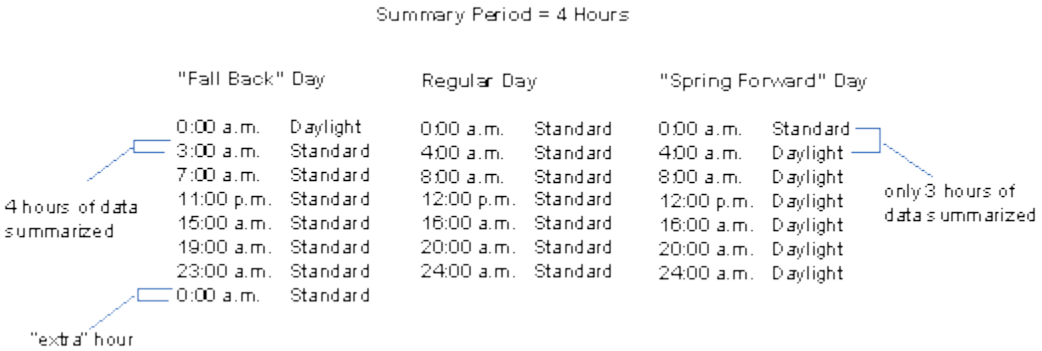
In this example, the summary period is configured to be every 30 minutes. On the "fall back" day, will be two extra summaries performed during the repeated hour for that day. For the "spring forward" day, there will be two summaries missing because of the skipped hour. The next replication occurs at the next scheduled time. In this case, it would be 3:00 a.m.



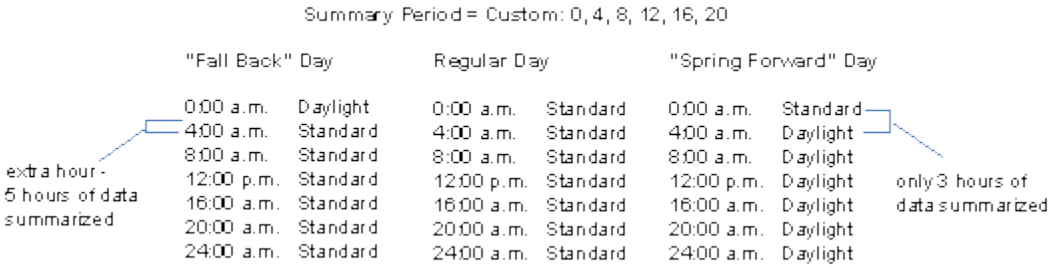
In the next example, the summary period is configured for every four hours. The scheduled summaries do not occur exactly on or within the boundaries of the time change hour. In this case, on the "fall back" day, the summary subsequent to the time change hour includes four hours of data for the "fall back" day. An extra

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

summary for an hour's worth of data is performed at the end of the "fall back" day. On the "spring forward" day, the summary period that contains the skipped hour includes one less hour of data.



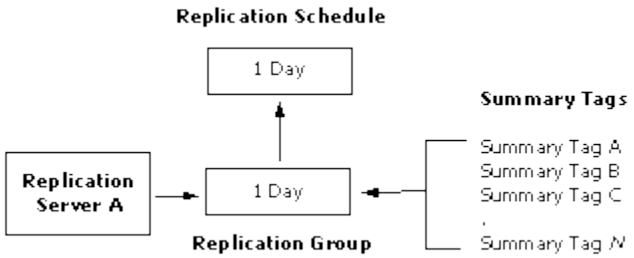
For a custom summary period, the summaries always occur at the fixed times of day that you specify in local time. However, the summary includes and extra hour of data for the "fall back" day (because of the overlap hour) and for the "spring forward" day (because of the skipped hour).



If a Daylight Savings Time change causes a scheduled time to be ambiguous, such as 1:30 a.m. on a "fall back" day when the clock jumps from 1:59 a.m. Daylight Savings Time to 1:00 a.m. standard time and the time could be interpreted as 1:30 a.m. Daylight Savings Time or 1:30 a.m. Standard Time, the replication will occur at the latter of the two occurrences. In this case it would be 1:30 a.m. Standard Time.

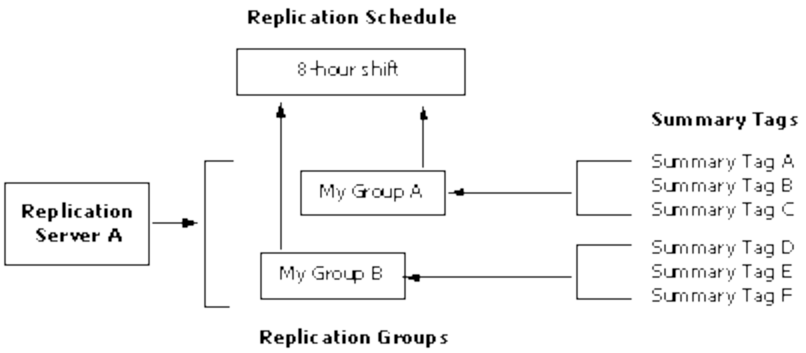
Replication Groups

A replication group abstracts a tag from a schedule. You can assign multiple summary tags to a single replication group.

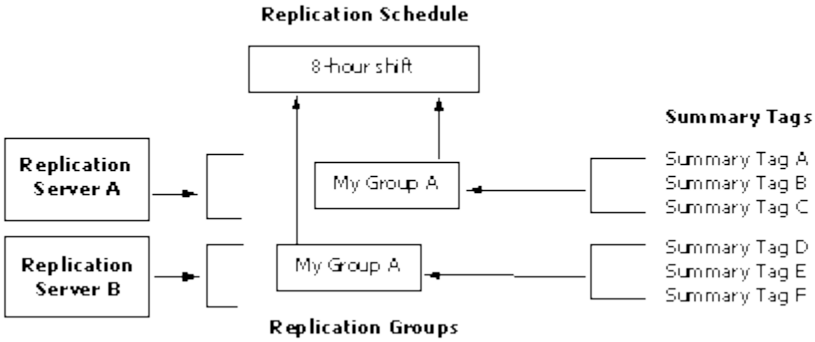


Multiple groups can be assigned to a single schedule. This simplifies maintenance. If you need to edit the schedule (for example, change the time of day for a shift end), you only need to edit the replication schedule, not the individual groups or summary tag configurations.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



A replication group must be unique for a type of summary tag, either analog or state. You can, however, have the same group name for analog summary tags as you do for state summary tags. You can also have the same replication group defined in multiple servers.



How Replication is Handled for Different Types of Data

An accurate map of data between replication tiers is maintained over time. This mapping includes both tag configuration and data synchronization.

Replication is unidirectional -- it goes from one tier to the next tier. For example, from the tier-1 historian to the tier-2 historian, but not from tier 2 to tier 1. If the data on a next-tier historian is changed in any way, the system does not try to map the change back to the lower-tier historian.

For example, suppose a tag from historian A is replicated in real-time to historian B. The tag on historian B has exactly the same data and OPC quality values as the tag on historian A. The replication system performs the following actions:

- When a new original value fitting the real-time window gets stored on historian A, it gets transmitted and stored on historian B, as well as the original value.
- If you perform an insert or update operation for some old values of the historian A, the same change is reflected on historian B.
- If some store-and-forward data gets merged into history on historian A, the same data gets transmitted to historian B and gets merged into history of historian B.

Replication is implemented in two ways: streaming replication and queued replication. The replication system uses a combination of streamed replication and queued replication as required.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Streaming Replication

When values of originating tier-1 tags are received from an IDAS or HCAL (AVEVA Application Server) and arrive at the tier-1 historian as a time-ordered data stream directly, the historian not only stores the data, but also forwards it to the Replication subsystem if replication is configured for those tags.

Then the Replication subsystem immediately streams that data to the tier-2 historian for simple replication, or performs summary calculations and streams the resulting summaries. Likewise, if there are more tiers beyond tier 2, the Replication subsystem streams the data beyond tier 2 to the next-level tiers.

This happens equally efficiently for tag values of timestamps close to the current system time and for late data tags.

If any next-tier historian becomes unavailable, the Replication subsystem continues to stream replicated data into the local store-and-forward path. When the connection is restored, all replicated data is sent as compressed snapshots to the next-tier historian and incorporated into history.

Streaming replication is the fastest and most efficient way of data replication, but there are some scenarios where it cannot be used. In that case, another method called queued replication is applied.

Queued Replication

AVEVA Historian uses queued replication in certain cases when streaming replication is not appropriate, such as when:

- **The data stream is interrupted.**
  For example, the remote IDAS configured for store-and-forward cannot communicate with the tier-1 historian for a long time. When the connection is restored, the store-and-forward data finally arrives at the tier-1 historian. But by that time, it may already be streaming newer data.
- **Lower-tier historian gets new data.**
  For example, an insert, update, or CSV file import operation is performed for lower-tier tag values. This means the summaries should be recalculated for that time period, and then re-replicated to the next-tier historian(s).
- **The lower-tier historian is stopped or restarted.**
  For example, if the lower-tier historian is restarted and there are some summaries spanning across the startup/shutdown time, they must be recalculated and re-replicated to the next-tier historian(s).

When such cases occur, the Replication subsystem receives notifications from the manual data storage service. The notifications contain information about what kind of lower-tier tag data (original or latest) has changed for a particular time interval. The Replication subsystem places that notification record into the replication queue stored in the Runtime database of the lower-tier historian. Later, when the connection to the next-tier historian is restored, the Replication subsystem processes that queue by querying the lower-tier data and replicating it to the next-tier historian(s). When the queue item is successfully processed, it is removed from the replication queue.

Although the Replication subsystem optimizes the queue by combining adjacent records, queued replication is slower and requires more system resources as compared to streamed replication, because it involves querying lower-tier data already stored on disk.

Queued replication does not support data values with timestamps before the year 1970.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag Configuration Synchronization between Tiered Historians

If a summary tag is deleted on the lower-tier historian, its corresponding tag on each next-tier historian remains intact to allow for retrieval of data already collected. A lower-tier tag cannot be deleted from the lower-tier historian if it is being replicated. You must first delete the tag replication and then delete the lower-tier tag.

A next-tier tag can be deleted from the next-tier historian, but it should be deleted only after the corresponding replication has been deleted from the lower-tier historian in that replication relationship. Otherwise, it will be recreated.

Replication Run-time Operations

Communication between tiers of replication historians may be interrupted because of a network outage or some other reason. When such interruptions occur, the replicated tags can still be configured and the data collected. Then, when communication is restored, the replication configuration and data are sent to the next-tier historian.

System and data integrity is not guaranteed if a disorderly shutdown occurs, such as a power outage.

Replication for tags will stop if:

- You delete the source tag configuration on the tier-1 historian.
- You configure the tier-1 tag so that its data values are not stored.
- An integer analog tag is being replicated as a state summary, and you change the source tag to be a real analog tag.

Replication Latency

Replication latency is the time it takes for the system to make a value available for retrieval on the next-tier historian from the moment it was stored or calculated on the previous tier.

Replication latency depends primarily on whether the streaming or queued replication method is being applied in each particular case and the available system resources to execute that method in each particular case.

Streaming replication tends to have a shorter latency period than queued replication as it deals with smaller amounts of data and is processed at a higher priority.

Replication Delay for "Old" Data

Replication delay identifies how frequently "old" data -- which includes inserts, updates, and store-and-forward data -- is sent from the one tier to the next-tier historian. The replication delay applies only to queued replication.

You specify the delay using the OldDataSynchronizationDelay system parameter. For more information, see System Parameters.

This delay represents your intent, while the replication latency identifies the real time difference. If the latency period becomes longer than the replication delay, the system will not be able to maintain the expected replication.

If you set the OldDataSynchronizationDelay system parameter to 0 (zero), all changes detected in the lower-tier are immediately sent to the next-tier historian(s), which may be very inefficient for certain applications.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Continuous Operation

If a next-tier historian that is configured for store-and-forward operations becomes unavailable, AVEVA Historian ensures these continuous operations:

- You can still add, modify and delete replication and summary objects in the local configuration of the lower-tier historian.
- You can store data locally for next-tier tags created before the next-tier historian became unavailable or while it is still unavailable.

Once the next-tier historian is available, the Replication Service does the following:

- Compares the latest replication and summary objects with the next-tier tags currently existing on the next-tier historian
- Performs a dynamic reconfiguration to ensure all data is synchronized.
- Sends reconfiguration history that was stored locally to the next-tier historian so it can be merged with other history information.

Once done, It will appear as though the disconnection between the tiers never took place.

Overflow Protection

If a next-tier historian cannot handle the incoming replication data, the Replication Service detects the situation and switches into store-and-forward mode. The data is then accumulated locally until the limit is reached. If that happens, all data to be sent to the next-tier historian is discarded and an error message is logged.

Security for Data Replication

Connections from a lower-tier historian to a next-tier historian must be authenticated before any replication task can be performed on the next-tier historian.

A local Windows user group called aaReplicationUsers is created on the next-tier historian during the next-tier historian installation. The Network Account (previously called the ArchestrA User) is automatically added to this group. Only members of the aaReplicationUsers group are allowed to perform replication tasks. These include adding, modifying, and sending values for replication tags. This group is not allowed to perform other non-replication tasks, such as adding or modifying a tier-1 tag.

When you configure a replication server on one historian, you must specify a valid Windows user account on the next-tier historian for the replication service to use.

For example, suppose you are configuring a replication server on a tier-1 historian. The tier-2 security account does not have to be a valid account on the tier-1 historian or even be in the same security domain as the tier-1 historian. If no replication user credentials are configured in at the tier-1 historian, the Network Account credential is passed to the tier-2 historian for authentication.

Adding a Replication Server

A replication server is a next-tier historian that receives and stores replicated data. The "next" tier depends on where the data is being sent from. A tier-1 historian replicates to its next tier, tier 2. The tier-2 historian

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

replicates to its next tier, tier 3, and so on.

A replication server is configured on the lower-tier historian; that is, the one sending the replication data:

- The tier-2 replication server configuration is defined on the tier-1 historian.
- Tier 3 is configured on tier 2, and so on.

When you define the configuration for a replication server, you set up replication schedules, groups, and tags for that next-tier historian.

Important: Replication is not supported between a case-sensitive historian and a case-insensitive next-tier historian.

If the next-tier historian doesn't yet exist or can't be reached over the network, the information is held until the lower-tier historian can connect with an instance of AVEVA Historian on the next-tier computer. If the lower-tier historian cannot communicate with the next-tier historian for any reason, data accumulates in the designated store-and-forward path.

By default, AVEVA Historian creates a local replication server (named Local Replication) as part of the installation process. You can create replication tags that use this local replication server or another replication server.

When you create a new replication server, the system automatically generates replication groups for analog summary and state summary replication types using default replication schedules. For more information, see Adding a Replication Group.

A few default replication schedules are also created for your new replication server, and you can use these schedules to create replication groups.

The system also creates a list of system tags for each replication server. For more information on the default system tags, see Replication Subsystem Tags in the *AVEVA Historian Concepts Guide*.

AVEVA Historian supports replication to another AVEVA Historian, AVEVA Insight, AVEVA Data Hub, and AVEVA PI Server.

- Adding AVEVA Historian as a Replication Server
- Adding AVEVA Insight as a Replication Server
- Adding AVEVA Data Hub as a Replication Server
- Adding AVEVA PI Server as a Replication Server

Adding AVEVA Historian as a Replication Server

If you want to connect from AVEVA Historian to another AVEVA Historian as a next-tier server, follow these steps.

To add AVEVA Historian as a replication server

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Replication**.
3. Right-click **Replication Servers** and select **New Replication Server**. The **New Replication Server** dialog box appears.
4. Select AVEVA Historian as the **Replication Environment**.
5. Configure the basic options for the replication server as follows:
   - **Node Name/IP Address**
     Type the node name or IP address of the computer where the next-tier historian resides. (This node

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

name or IP address does not need to be active or accessible when you set up the information for the new replication server.)

- **Description**
  Type a description of the server. This description appears in the Operations Control Management Console and in Historian Client reports.
- **Store & Forward path**
  Type the store-and-forward path for data for this next-tier replication server. This must be an absolute path on the current computer. Remote paths are not supported for store-and-forward addresses.
- **Replicate Alarms and Events**
  This is enabled by default. This replicates alarms and events to the replication server.

  Note that this is a simple replication of alarms and events. Summary replication does not apply.
- **Connection Info**
  Type the username and password for the replication server. To specify a Windows domain, prefix the username with the domain name and a backslash. For example: <domain name>\<username>.
- **TCP Port**
  Type the TCP port to use for the new replication server. The TCP port is used by the next-tier historian to receive data from the replication service on the lower-tier historian. This can be an integer between 1 and 65535. The default is 32565. This port number must match the *ReplicationTcpPort* system parameter value that is specified on the next-tier historian. Be sure that you open this port in Windows Firewall. The port must not conflict with any other ports used by other applications on the next-tier historian.
- Select whether or not to accept untrusted certificates:
  - **Require a trusted certificate:** Require a trusted certificate for communication between the IDAS and Historian.

    The server's certificate must be trusted on the IDAS system. If both nodes are connected to the same System Management Server, certificate trust is managed by the System Management Server. If the IDAS system and Historian are not connected to the same System Management Server, when you test the connection, the configurator either prompts you to trust the server's certificate, or logs an error message if it cannot retrieve the certificate information.
  - **Accept an untrusted certificate (NOT RECOMMENDED):** Allow an untrusted connection.

    Note: Requiring a trusted certificate is strongly recommended. Untrusted connections should only be used in test environments.

6. To test the connection to the new replication server computer, click **Test Connection**.

   The test succeeds only if:

   - The node name is valid.
   - Valid credentials for an account that has permission on the next-tier node to send replication data are entered.
   - If **Use trusted connection** is selected, the certificate from the next-tier historian must be trusted for successful communication between nodes. If both nodes are connected to the same System Management Server, certificate trust is managed for you by the System Management Server. If the next-tier historian is not connected to the same System Management Server, the configurator prompts you to trust the next-tier historian's certificate.

     If the test fails, see the ArchestrA Logger entries on both the lower-tier and next-tier historians for more information.
7. Click **Next**. The **New Replication Server - Advanced** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

You can use the default summary and simple replication tag naming schemes, or you can create your own.

8. In the **Summary Replication Tag Naming Scheme** and **Simple Replication Tag Naming Scheme** areas, select the replication tag naming scheme to use. Specify a custom naming scheme by selecting **Custom** and clicking the ellipsis button to the right of the box. The **Naming Scheme** dialog box appears. For information about configuring the naming scheme, see Specifying Naming Schemes for Replication.
9. Configure the remaining advanced settings as follows:
   - **Min SF Duration**

     Enter the minimum duration in seconds for the replication service to function in store-and-forward mode. The replication service functions in store-and-forward mode for this length of time even if the condition that caused the replication service to function in store-and-forward mode no longer exists. The duration can be an integer from 0 to 3600. Pick a value that provides a smooth transition for store-and-forward operation and prevents the system from repeatedly going in and out of store-and-forward mode.
   - **Buffer Count**

     Enter the number of 64KB buffers to allocate for the new replication server. This can be an integer from 128 to 2048. You may need to increase the buffer count to accommodate high data rates.
   - **SF free space**

     Minimum amount of free storage space required for store-and-forward to operate.
   - **Compression Enabled**

     Select this option to enable compression for the packets sent to the replication server. For guidelines on using compression, see the performance information in the *AVEVA System Platform Installation Guide*.
   - **Bandwidth**
     - Select **Unlimited** to allow unlimited bandwidth to be used by the HCAL to communicate with the historian.
     - Otherwise, leave it unselected and enter a value in the **Kbps** box to specify a custom bandwidth. This can be an integer from 10 to 1000000.

       Setting a limit on the bandwidth throttles the amount of data that is sent during operations such as a store-and-forward, an update, or a CSV export. You commonly use this feature when you have a WAN with a low bandwidth. For example, suppose your WAN has a 128kbps bandwidth between the lower-tier and next-tier historians. If you have real-time streaming data that requires 64kbps for normal operation, but the network is down for 8 hours and information is saved for store-and-forward, it will take 8 hours for the data to be uploaded. If you leave this set to **Unlimited**, it will work fine.

       But if you have 256kbps bandwidth that you must share with other applications, set this to 128kbps to throttle the store-and-forward data and anything that isn't streaming. This prevents the historian's operations from choking the other applications for bandwidth. The most common symptom of a problem is that when you come out of store-and-forward and start sending data, the remote desktop is very sluggish and unresponsive. The bandwidth limit also applies to streaming data. However, streaming data will not be throttled, and instead HCAL will be put into store-and-forward mode. For example, if the streaming bandwidth is always 96kbps, and you set the limit at 128kbps, there will be 32kbps remaining for store-and-forward to use. However, if you had a sudden spike and streaming jumped up to 156kbps, it would exceed the limit and force HCAL into store-and-forward mode.
   - **High Latency Network**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Select this option if the connection to the replication server takes place over a high-latency network (ping response times over 500 milliseconds). This option allows for longer connection timeouts. If you use this option, we recommend that you also enable compression.
- Leave it unselected if you are not expecting connection timeouts, as this setting can negatively affect performance on a low-latency network.

10. Click **Finish**. The new replication server appears in the replication server list.

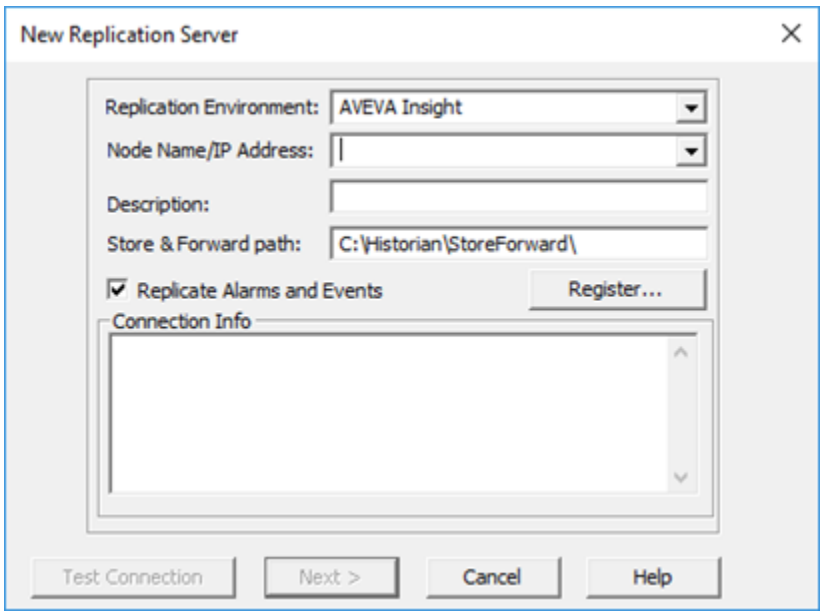
Adding AVEVA Insight as a Replication Server

If you want to connect from AVEVA Historian to AVEVA Insight as a next-tier server, follow these steps.

Note: You must have an account with AVEVA Insight to complete these steps. If you do not have an AVEVA Insight account, go to the AVEVA Insight site (https://insight.connect.aveva.com) and click **Register**.

To add AVEVA Insight as a replication server

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Replication**.
3. Right-click **Replication Servers** and select **New Replication Server**. The **New Replication Server** dialog box appears.



4. Select AVEVA Insight as the **Replication Environment**.
5. Specify a **Node Name** and **Description** for the next-tier server. The description you provide appears in the Operations Control Management Console and in Historian Client reports.

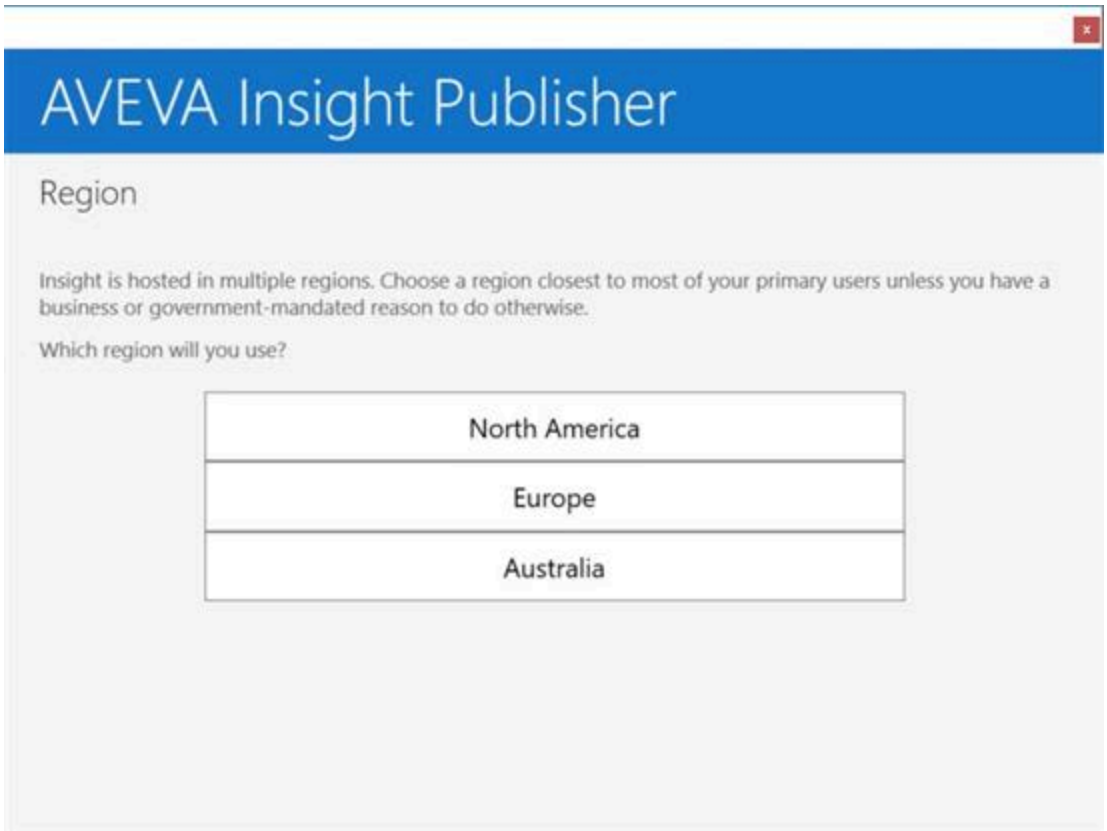
   Note: Although the field is called **Node Name/IP Address**, for AVEVA Insight this value is only used as a name, not as the connection target. The connection details are determined when you click the **Register...** button.

6. Click **Register**.

   This launches AVEVA Insight Publisher.

7. Follow the on-screen instructions to sign into AVEVA Insight, register your Insight data source, and publish your historian replication tags to Insight.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



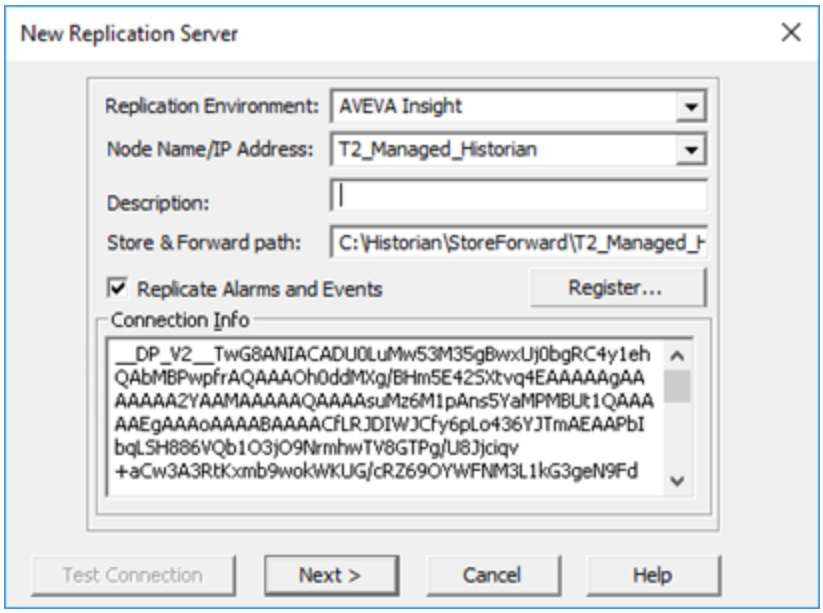
Note: For more information about publishing data to AVEVA Insight, see AVEVA Insight help (https://insight.connect.aveva.com/help).

8. Once the registration process is complete, you'll see encrypted information in the **Connection Info** box.

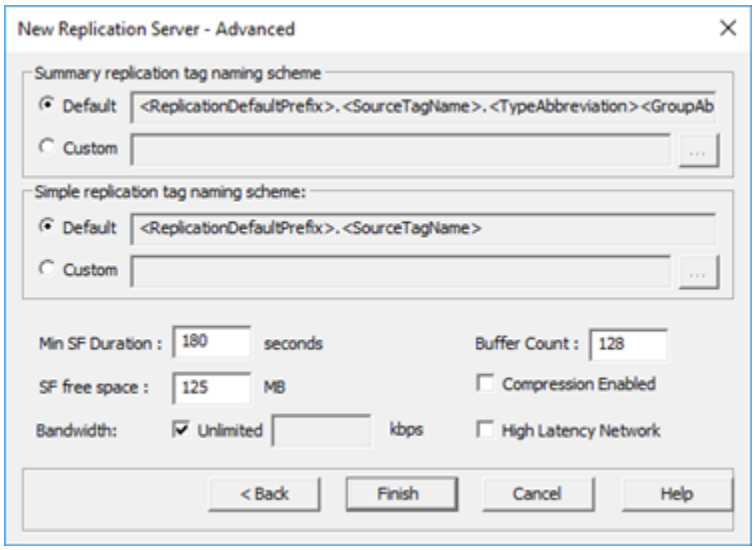
Important: Do not tamper with this encrypted string in any way. This information is valid only on the computer that created it. If you backup or restore Runtime or use the DB Config Import/Export utility to move it to a different computer, you must re-register using the same data source name and then select the **Replace** option.

This is an example of the encrypted information you'll see:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



9. Click **Next**. The **New Replication Server - Advanced** dialog box appears.



You can use the default summary and simple replication tag naming schemes, or you can create your own.

10. In the **Summary Replication Tag Naming Scheme** and **Simple Replication Tag Naming Scheme** areas, select the replication tag naming scheme to use. Specify a custom naming scheme by selecting **Custom** and clicking the ellipsis button to the right of the box. The **Naming Scheme** dialog box appears. For information about configuring the naming scheme, see Specifying Naming Schemes for Replication.
11. Configure the remaining advanced settings as follows:
    - **Min SF Duration**

      Enter the minimum duration in seconds for the replication service to function in store-and-forward mode. The replication service functions in store-and-forward mode for this length of time even if the condition that caused the replication service to function in store-and-forward mode no longer exists. The duration can be an integer from 0 to 3600. Pick a value that provides a smooth transition for store-and-forward operation and prevents the system from repeatedly going in and out of store-and-forward mode.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Buffer Count**

  Enter the number of 64KB buffers to allocate for the new replication server. This can be an integer from 128 to 2048. You may need to increase the buffer count to accommodate high data rates.
- **SF free space**
- **Compression Enabled**

  Select this option to enable compression for the packets sent to the replication server. For guidelines on using compression, see the performance information in the *AVEVA System Platform Installation Guide*.
- **Bandwidth**
  - Select **Unlimited** to allow unlimited bandwidth to be used by the HCAL to communicate with the historian.
  - Otherwise, leave it unselected and enter a value in the **Kbps** box to specify a custom bandwidth. This can be an integer from 10 to 1000000.

    Setting a limit on the bandwidth throttles the amount of data that is sent during operations such as a store-and-forward, an update, or a CSV export. You commonly use this feature when you have a WAN with a low bandwidth. For example, suppose your WAN has a 128kbps bandwidth between the lower-tier and next-tier historians. If you have real-time streaming data that requires 64kbps for normal operation, but the network is down for 8 hours and information is saved for store-and-forward, it will take 8 hours for the data to be uploaded. If you leave this set to **Unlimited**, it will work fine.

    But if you have 256kbps bandwidth that you must share with other applications, set this to 128kbps to throttle the store-and-forward data and anything that isn't streaming. This prevents the historian's operations from choking the other applications for bandwidth. The most common symptom of a problem is that when you come out of store-and-forward and start sending data, the remote desktop is very sluggish and unresponsive. The bandwidth limit also applies to streaming data. However, streaming data will not be throttled, and instead HCAL will be put into store-and-forward mode. For example, if the streaming bandwidth is always 96kbps, and you set the limit at 128kbps, there will be 32kbps remaining for store-and-forward to use. However, if you had a sudden spike and streaming jumped up to 156kbps, it would exceed the limit and force HCAL into store-and-forward mode.
- **High Latency Network**
  - Select this option if the connection to the replication server takes place over a high-latency network (ping response times over 500 milliseconds). This option allows for longer connection timeouts. If you use this option, we recommend that you also enable compression.
  - Leave it unselected if you are not expecting connection timeouts, as this setting can negatively affect performance on a low-latency network.

12. Click **Finish**. The new replication server appears in the replication server list.

Adding AVEVA Data Hub as a Replication Server

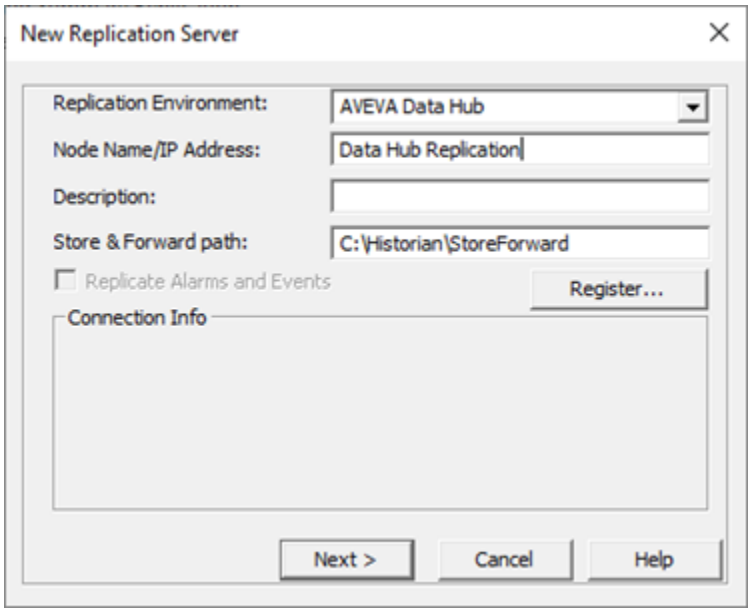
If you want to connect from AVEVA Historian to AVEVA Data Hub as a next-tier server, follow these steps.

To add AVEVA Data Hub as a replication server

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Replication**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

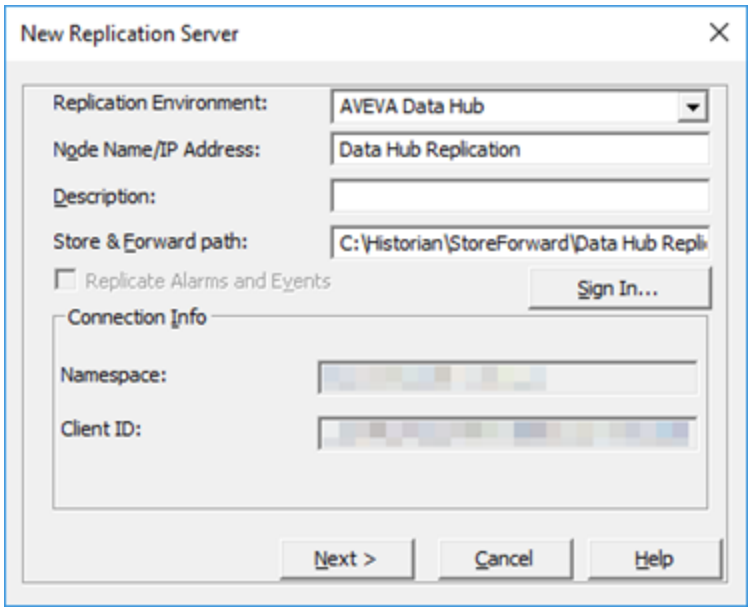
3. Right-click **Replication Servers** and select **New Replication Server**. The **New Replication Server** dialog box appears.



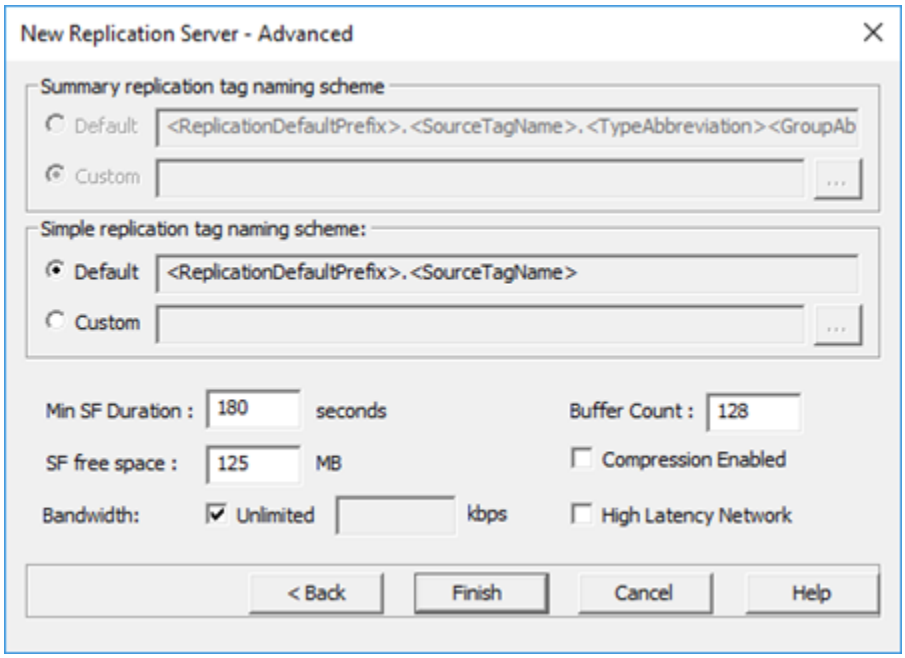
4. Select **AVEVA Data Hub** as the **Replication Environment**.
5. Configure the basic options for the replication server as follows:
   - **Node Name/IP Address**
     Specify a Node Name for the next-tier server.

     Note: Although the field is called **Node Name/IP Address**, for AVEVA Data Hub this value is only used as a name, not as the connection target. The connection details are determined when you click the **Register...** button.
   - **Description**
     Enter a description of the server. This description appears in the Operations Control Management Console and in Historian Client reports.
   - **Store & Forward path**
     Enter the store-and-forward path for data for this next-tier replication server. This must be an absolute path on the current computer. Remote paths are not supported for store-and-forward addresses.
   - **Replicate Alarms and Events**
     This option is not currently supported for replication to AVEVA Data Hub.
6. Click **Register...** to sign in to AVEVA Connect and select a tenant with access to AVEVA Data Hub.
7. After a successful login, a list of available namespaces displays. AVEVA Data Hub is hosted in multiple regions. Select a namespace in the appropriate region for your needs, then click **Register**. The **New Replication Server** dialog box redisplays. Your connection with AVEVA Data Hub is registered, and a **Client ID** is generated and added to the Connection Info section.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



8. Click **Next**. The **New Replication Server - Advanced** dialog box appears.



Summary replication is not supported for replication to AVEVA Data Hub.

9. In the **Simple Replication Tag Naming Scheme** area, select the replication tag naming scheme to use. Specify a custom naming scheme by selecting **Custom** and clicking the ellipsis button to the right of the box. The **Naming Scheme** dialog box appears. For information about configuring the naming scheme, see Specifying Naming Schemes for Replication.
10. Configure the remaining advanced settings as follows:
    - **Min SF Duration**

      Enter the minimum duration in seconds for the replication service to function in store-and-forward mode. The replication service functions in store-and-forward mode for this length of time even if the condition that caused the replication service to function in store-and-forward mode no longer exists. The duration can be an integer from 0 to 3600. Pick a value that provides a smooth transition for store-and-

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

forward operation and prevents the system from repeatedly going in and out of store-and-forward mode.

- **Buffer Count**

  Enter the number of 64KB buffers to allocate for the new replication server. This can be an integer from 128 to 2048. You may need to increase the buffer count to accommodate high data rates.
- **SF free space**
- **Compression Enabled, Bandwidth, High Latency Network**

  These options are not used for replication to AVEVA Data Hub.

11. Click **Finish**. The new replication server appears in the replication server list.

Adding AVEVA PI Server as a Replication Server

If you want to connect from AVEVA Historian to AVEVA PI Server as a next-tier server, follow these steps:

1. Prepare your PI Server for Receiving Replication Data.
2. Configure Replication to AVEVA PI Server.

If you are already using the PI Connector for AVEVA Historian and you want to migrate to using AVEVA Historian replication instead, to avoid data gaps in the PI system you should leave the PI Connector running while configuring replication to AVEVA PI Server.

Follow these steps to migrate from using the PI Connector to using AVEVA Historian replication to PI Server:

1. Ensure the PI Connector for AVEVA Historian is running.
2. Prepare your PI Server for Receiving Replication Data.
3. Configure Replication to AVEVA PI Server.
4. Configure all required tags for replication. See Configuring Tags to Be Replicated for more details.

   Note: The **Destination Tag Name** for each replicated tag must match the existing tag name used by the PI Connector.
5. Verify that all tags are replicating correctly.
6. Shut down the PI Connector for AVEVA Historian.

Prepare your PI Server for Receiving Replication Data

Before you can add your AVEVA PI Server as a replication server, you need to prepare your PI Server to receive replication data.

To prepare your PI Server for receiving replication data

The PI server must have the PI Web API installed with Open Message Format (OMF) enabled. (OMF is enabled by default.)

Replication to PI supports two different types of authentication: Basic and Kerberos.

The authentication type must be enabled in the PI Web API. Then the Historian replication server settings select which authentication type to use for replication.

The PI Web API configuration can be accessed through PI System Explorer. For more information, see Configuration at runtime in the PI Web API documentation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring the PI System for the user/computer accessing it

The user or computer accessing the PI system needs access to both of the following:

- The PI Asset Framework server
- The PI Data Archive server

Configure authentication as follows:

- **Basic authentication:** Give the user that will be entered into the Replication Server configuration interface access to the PI System.
- **Kerberos authentication:** By default, Replication runs as a virtual service account (NT SERVICE\InSQLConfiguration). If you cannot give the virtual service account access (because your IT policy prohibits giving a machine account access, for example), use the Windows MMC snap-in for Services to configure the AVEVA Historian Configuration service to run as a Windows user account. That account must be given appropriate permissions in the aaReplicationUsers and aaAdministrators user groups.

  Note: The replication server configuration UI still requires a value in the username and password fields, even though those values won't be used for Kerberos authentication. Enter any values.

Configuring the PI Asset Framework (PI AF) Server

To create a user mapping for the PI Asset Framework server:

1. Open **PI System Explorer**.
2. Select **File > Server Properties** to launch the PI AF Server Properties dialog.
3. Select the **Identities** tab.
4. Choose an identity with sufficient permissions and double click it. The Security Identity Properties dialog opens.

   For more details on configuring permissions in the PI Asset Framework Server, see Security configuration in PI AF in the Asset Framework and PI System Explorer documentation.
5. Select the **Mappings** tab.
6. Click the **Add** button and select the user or computer.

   Note: When adding a computer for Kerberos, the object type must be **Computers**.

Configuring the PI Data Archive Server

To create a user mapping for the PI Data Archive server:

1. Open **PI System Management Tools**.
2. Under Security go to Mappings & Trusts, right-click and select **New Mapping…** to open the Add New Mapping dialog.
3. For **Windows Account**, select the user (or computer) required
4. For **PI Identity**, select an identity with sufficient permissions to create tags and write data to the PI Data Archive.

   For more details on configuring permissions in the PI Data Archive Server, see Data Archive security in the Data Archive Administration documentation.

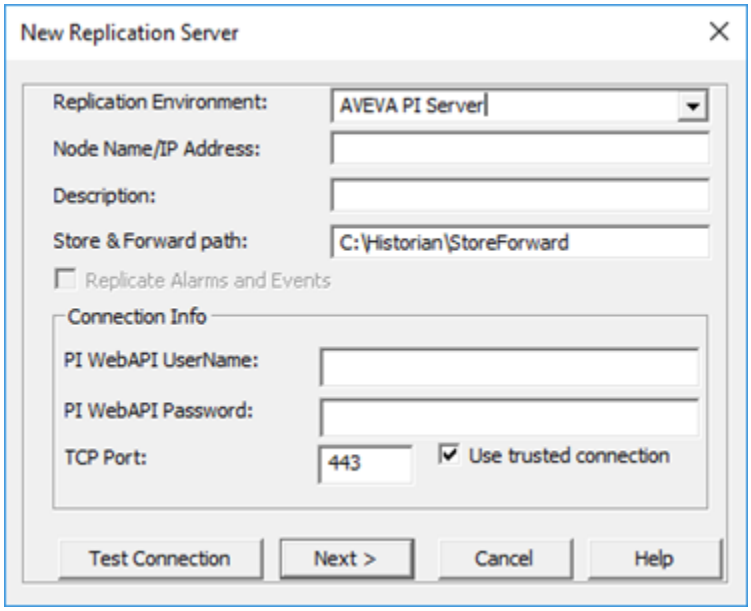
Note: When adding a computer for Kerberos, the object type must be **Computers**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configure Replication to AVEVA PI Server

To add AVEVA PI Server as a replication server

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Replication**.
3. Right-click **Replication Servers** and select **New Replication Server**. The **New Replication Server** dialog box appears.



4. Select **AVEVA PI Server** as the **Replication Environment**.
5. Configure the basic options for the replication server as follows:
   - **Node Name/IP Address**

     Enter the node name or IP address of the PI Web API node.
   - **Description**

     Enter a description of the server. This description appears in the Operations Control Management Console and in Historian Client reports.
   - **Store & Forward path**

     Enter the store-and-forward path for data for this next-tier replication server. This must be an absolute path on the current computer. Remote paths are not supported for store-and-forward addresses.
   - **Replicate Alarms and Events**

     This option is not currently supported for replication to AVEVA PI Server.
   - **Connection Info**

     Enter the username and password for the PI Web API User.
   - **TCP Port**

     Enter the listen port that the PI Web API is configured with.
   - **Use trusted connection**

     If this option is enabled, the endpoint certificate is validated when connecting to the PI server. If the

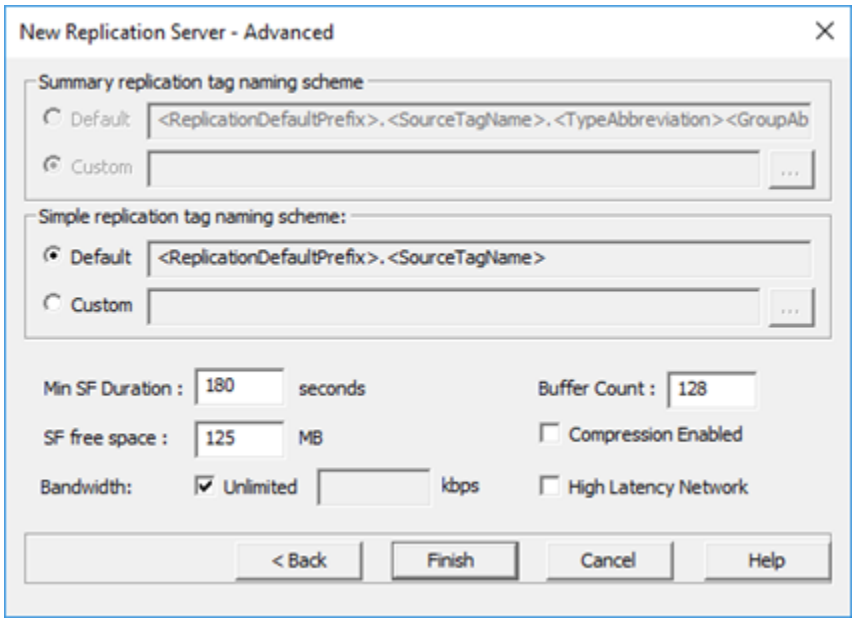
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

certificate is invalid, the connection is refused.

Important: This option should be enabled for improved security. If you are using a self-signed certificate for testing, you can disable this option, but make sure it is enabled for any production systems.

When this option is enabled the certificate used by the PI server must be trusted on the Historian server. Obtain a copy of the certificate from the PI server and see Trusting a Certificate for further directions.

6. Click **Test Connection** to verify the connection details.
7. When the test connection is successful, click **Next**. The **New Replication Server - Advanced** dialog box appears.



Summary replication is not supported for replication to AVEVA PI Server.

8. In the **Simple Replication Tag Naming Scheme** area, select the replication tag naming scheme to use. Specify a custom naming scheme by selecting **Custom** and clicking the ellipsis button to the right of the box. The **Naming Scheme** dialog box appears. For information about configuring the naming scheme, see Specifying Naming Schemes for Replication.
9. Configure the remaining advanced settings as follows:
   - **Min SF Duration**

     Enter the minimum duration in seconds for the replication service to function in store-and-forward mode. The replication service functions in store-and-forward mode for this length of time even if the condition that caused the replication service to function in store-and-forward mode no longer exists. The duration can be an integer from 0 to 3600. Pick a value that provides a smooth transition for store-and-forward operation and prevents the system from repeatedly going in and out of store-and-forward mode.
   - **Buffer Count**

     Enter the number of 64KB buffers to allocate for the new replication server. This can be an integer from 128 to 2048. You may need to increase the buffer count to accommodate high data rates.
   - **SF free space**
   - **Compression Enabled, Bandwidth, High Latency Network**

     These options are not used for replication to AVEVA PI Server.

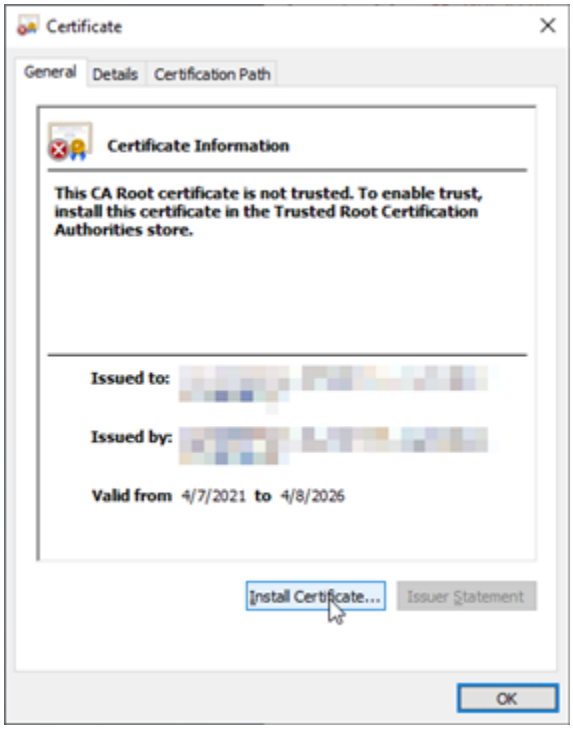
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

10. Click **Finish**. The new replication server appears in the replication server list.

Trusting a Certificate

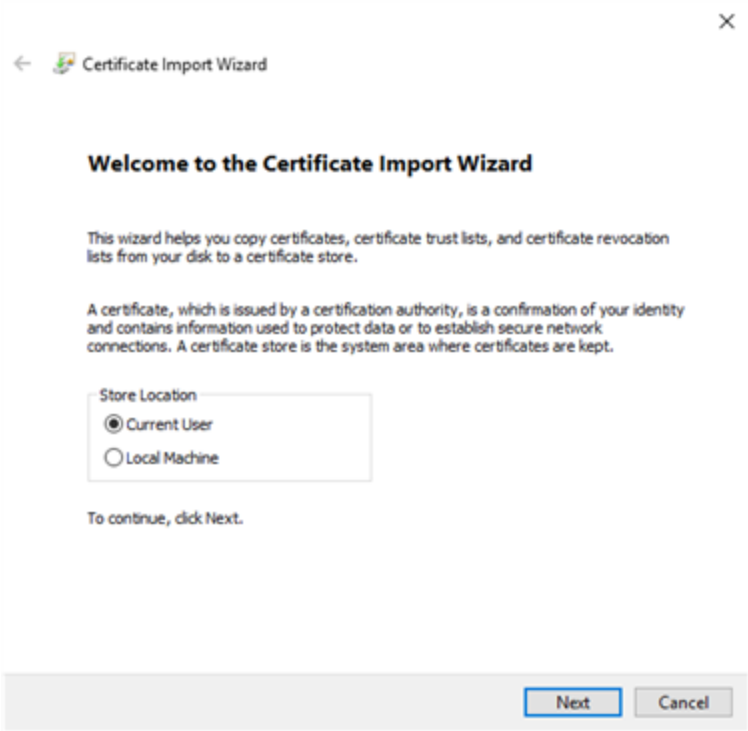
To install a certificate into the trusted root certificate store:

1. Locate and open the certificate file in Windows Explorer. The Certificate dialog displays:



2. Select **Install Certificate...**. The Certificate Import Wizard displays:

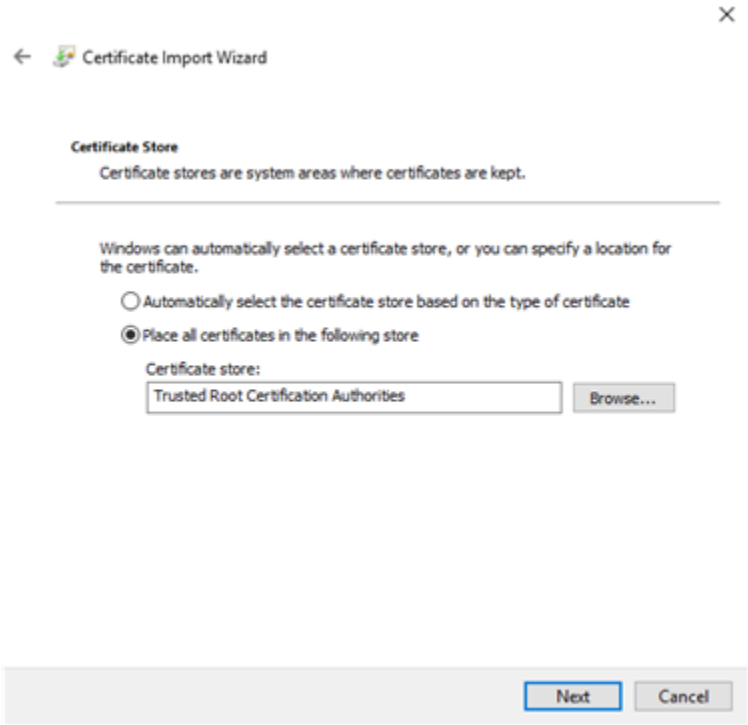
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Select **Current User** to install the certificate for only the current user, or **Local Machine** to install the certificate for all users on this system.

   Note: The **Local Machine** option requires administrative access to the system. If you do not have administrative access, select **Current User**.

   Click **Next**. The **Certificate Store** dialog displays:

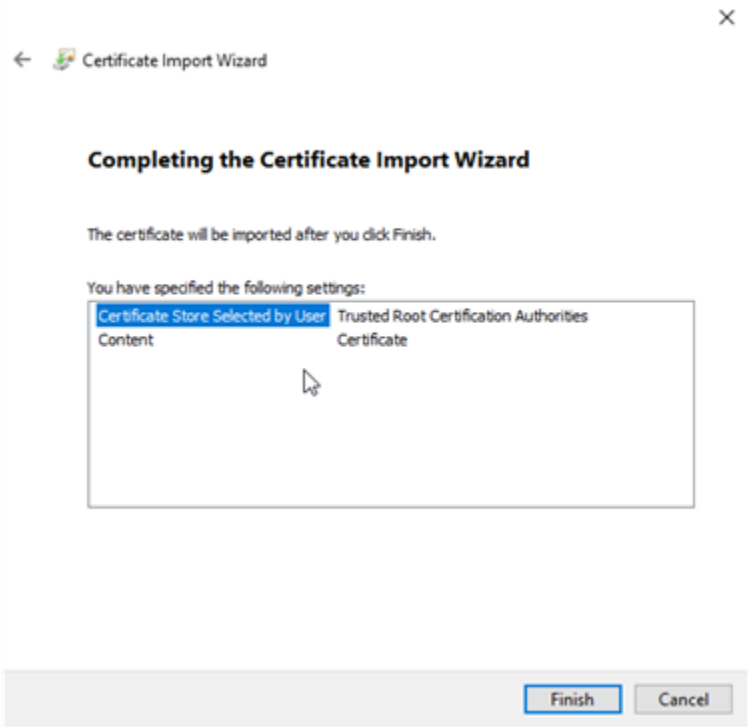


4. Select **Place all certificates in the following store**. Click **Browse...** and select **Trusted Root Certification**

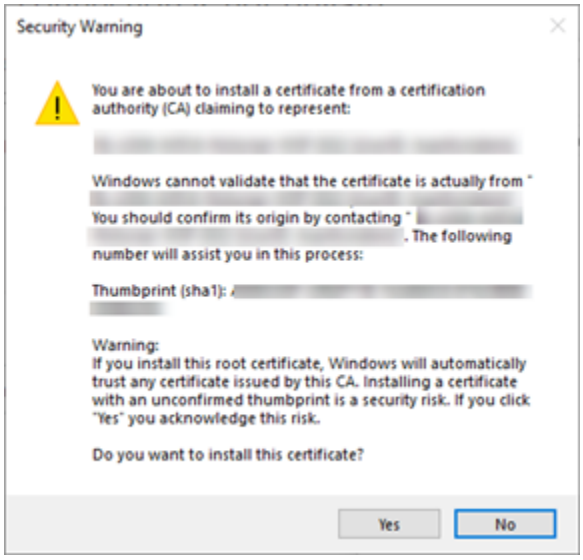
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Authorities as the **Certificate store**.

5. Click **Next.** The **Completing the Certificate Import Wizard** dialog displays:



6. Click **Finish** to complete the Certificate Import Wizard. A security warning displays:



Click **Yes** to acknowledge the warning. The certificate is now trusted on your machine.

Data Buffer Configuration

The System Platform adapter used for replicating data to AVEVA PI Server is configured to buffer a maximum of 1024 MB (1 GB) of data to disk in store/forward mode. If you replicate large enough data volumes and/or experience long enough communication outages between the Historian and PI servers, you may need to increase this limit to avoid losing replication data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To change the maximum data buffer size:

1. Locate the **Store & Forward path** for your configured replication server. You can find this by checking the properties of the replication server in the management console. For example, `C:\Historian\StoreForward\PiRepl01`.
2. Locate the `System_Buffering.json` file, which is in the `SystemPlatformAdapter\Configuration` sub-folder. For example, `C:\Historian\StoreForward\PiRepl01\SystemPlatformAdapter\Configuration\System_Buffering.json`.
3. Edit the `System_Buffering.json` file using the text editor of your choice. Change the value of the **MaxBufferSizeMB** setting to the maximum number of megabytes of storage to use for buffering data.

   Valid values are any integer from 1 to 2147483647. The default value is 1024.
4. Save your changes to `System_Buffering.json`, then restart the Historian to apply the changes.

Special Considerations for Tag Names

There are some limitations to be aware of when replicating to AVEVA PI Server.

Tag Name Length Limitations

The PI Server limits tag names to a maximum length of 200 characters. If the source tag name from the Historian is longer than 200 characters:

- The destination tag name on the PI Server is truncated at 200 characters
- The attribute name in the PI Asset Framework (AF) hierarchy is truncated at 200 characters, then the tag's ID (a GUID) is appended to preserve uniqueness.

Character Limitations

Certain characters are not supported for tag names by the PI Server. The following rules apply to tag names in AVEVA PI Server:

- The first character of a tag name must be alphanumeric, an underscore (_), or a percent sign (%)
- Double-byte characters are not supported
- Control characters, such as line feeds and tabs, are not supported
- The following characters are also not supported: * ? ; { } [ ] | \ ` ' "

If a tag's engineering unit contains an unsupported character, the tag's unit is set to *None* when it is replicated to the PI server.

If the *ReplicationDefaultPrefix* contains an unsupported character, the default value of the server's name is used.

If an unsupported character is detected in the source tag name, the AF attribute name uses the destination tag name instead. If the destination tag name also contains an unsupported character, the Operations Control Management Console detects the unsupported character and displays an error. The tag cannot be replicated until a valid destination tag name is set.

The *Location* tag extended property is used to build the AF hierarchy. If the *Location* property contains an unsupported character, it cannot be used and the tag is placed at the root of the hierarchy instead.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For more information about supported characters, refer to the following topics in the AVEVA PI Server documentation:

- https://docs.osisoft.com/bundle/pi-server/page/general-attributes.html
- https://docs.osisoft.com/bundle/pi-server/page/valid-characters-in-pi-af-object-names.html.

Specifying Naming Schemes for Replication

A replication naming scheme is a collection, or expression, of identifiers that fully describe a replication tag. Replication tag naming schemes identify the information being replicated quickly and easily and also prevent name collisions between tags that inadvertently have identical names. You are encouraged to plan your naming conventions and system requirements before implementing a naming scheme.

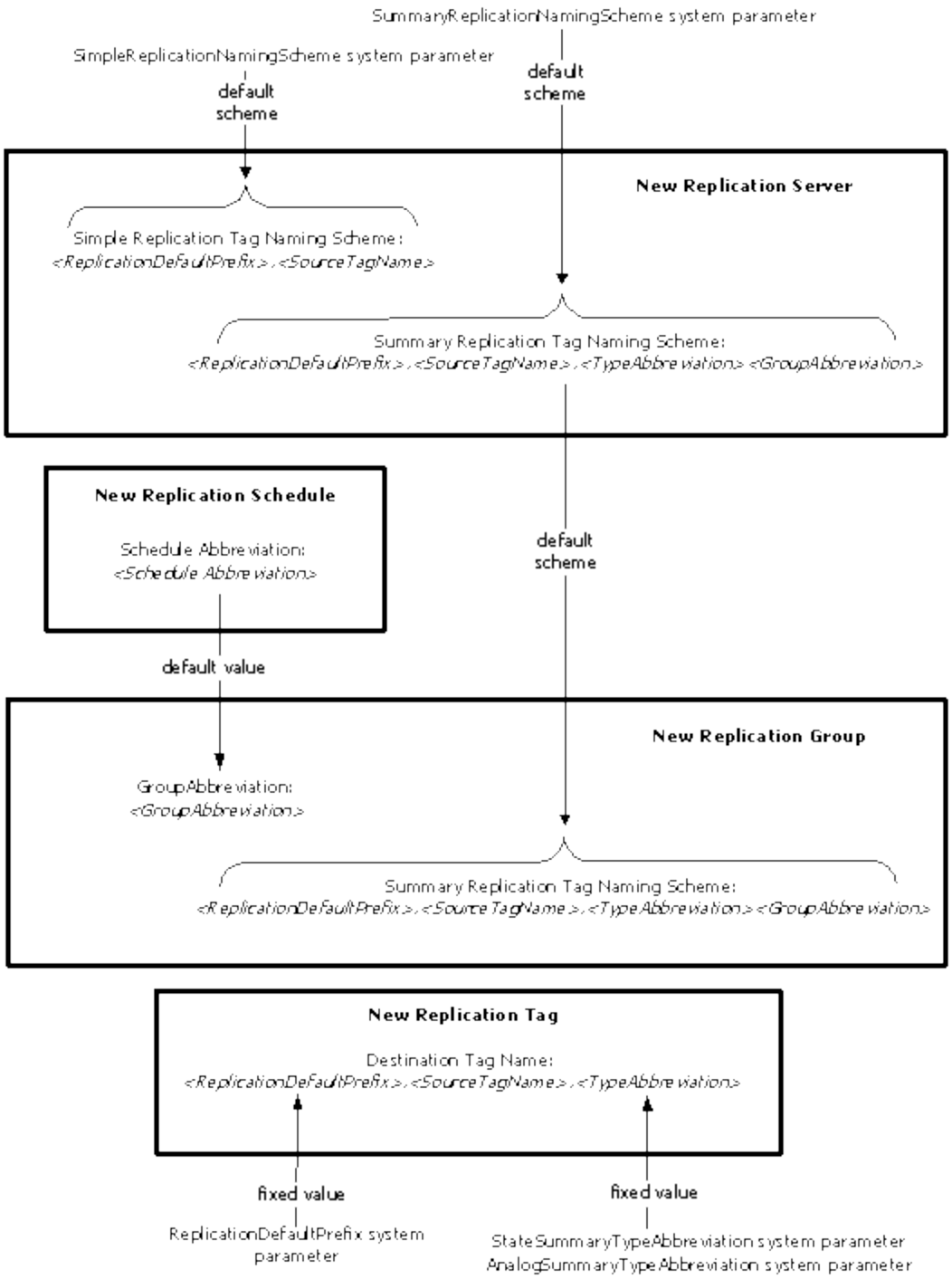
Note: Name collisions can occur when you have two or more tags (usually from different historians) that have the same name. The data stored from one tag is mixed with data from the other tag, contaminating the data on the replication server.

The naming scheme placeholders are described in the following table:

Value	Description
<ReplicationDefaultPrefix>	A prefix based on the replication server name.
<SourceTagName>	The name of the source tag on the lower-tier historian that is being replicated to the next-tier historian.
<TypeAbbreviation>	The type of summary tag: blank for analog summary or S for state summary. The type abbreviation is most useful for preventing collisions between analog tags. For an integer source tag, you can create both analog and state summary tags, which can result in tag naming collisions if you use only the other naming parameters.
<GroupAbbreviation>	The abbreviation for the replication group.

This diagram illustrates the naming scheme relationships:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



When you define a new replication server:

- The default scheme for the simple replication tag naming scheme is taken from the SimpleReplicationNamingScheme system parameter.
- The default scheme for the summary replication tag naming scheme is taken from the SummaryReplicationNamingScheme system parameter.

When you define a new replication schedule, you define a schedule abbreviation.

When you define a new replication group:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

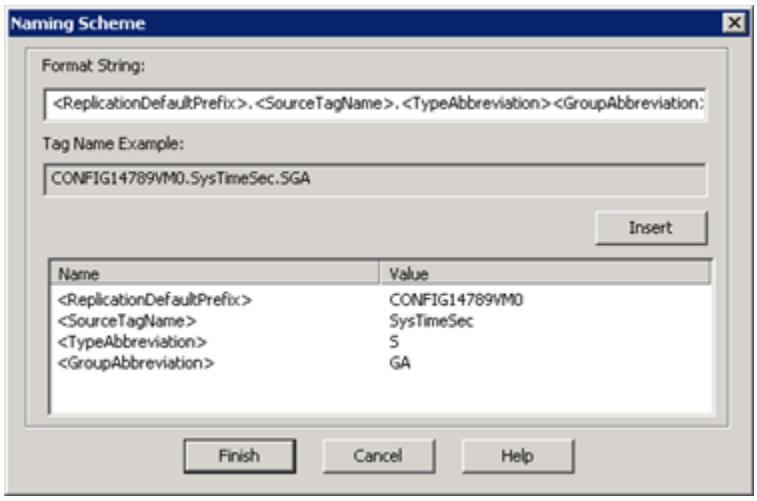
- The default value for the group abbreviation is taken from the schedule abbreviation for the associated replication schedule.
- The default scheme for the summary replication tag naming scheme is copied from the summary replication tag naming scheme of the associated replication server.

When you define a new replication tag:

- The value for the <ReplicationDefaultPrefix> is taken from the current setting of the ReplicationDefaultPrefix system parameter.
- The value for the <TypeAbbreviation> is taken from the StateSummaryReplicationTypeAbbreviation or AnalogSummaryReplicationTypeAbbreviation system parameter, depending on the type of replication (the "folder" within the server). The <TypeAbbreviation> is applicable only for summary replicated tags.

To configure the naming scheme

1. Specify a custom naming scheme in a dialog box by clicking **Custom**, and then clicking the button to the right of the box. The **Naming Scheme** dialog box appears.



2. Enter the naming scheme to use for tagging the replication tags. The parameters you can use and an example of what the string looks like appear in the **Tag Name Example** box.

   Note: Simple replication tags do not have a type abbreviation and are not assigned to a group. You can use the <TypeAbbreviation> and <GroupAbbreviation> parameters in a simple replication tag naming scheme, but they always have empty values.

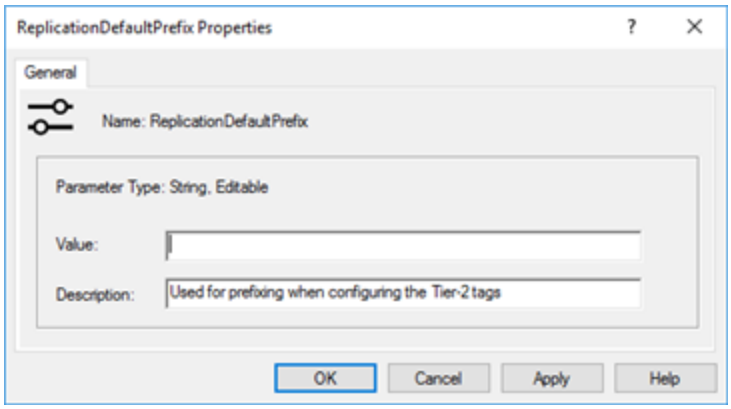
3. Click **Finish**.

Multiple lower-tier historians can replicate alarm and events to the same next-tier historian. Replication records on the next-tier historian include a source name to indicate the originating lower-tier historian. By default, that is the server name itself, but it can be modified.

To change source name for a lower-tier historian

1. In the OCMC, click **Configuration Editor**, click **System Configuration**, and then click **Parameters**.
2. In the list on the right, right-click **ReplicationDefaultPrefix** and then click **Properties**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



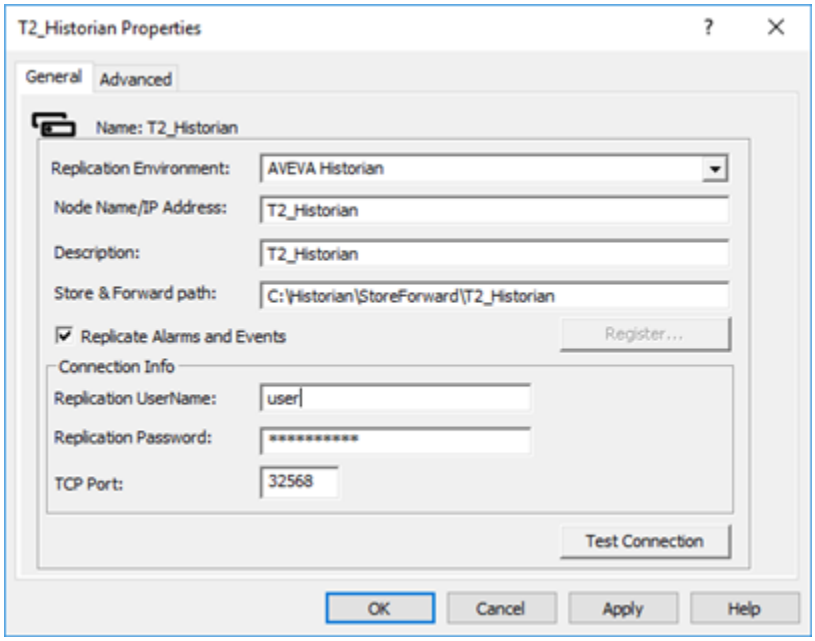
3. In **Value**, type a new name. Click **OK**.

Editing Replication Server Properties

You can edit the properties for a replication server. The general procedure and options are the same as for adding a replication server.

To edit properties for a replication server

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers.**
3. Right-click the replication server to be edited and then click **Properties**. The **Properties** dialog box appears.



4. Edit the general properties. The options are the same as for adding a new replication server. For more information, see Adding a Replication Server.

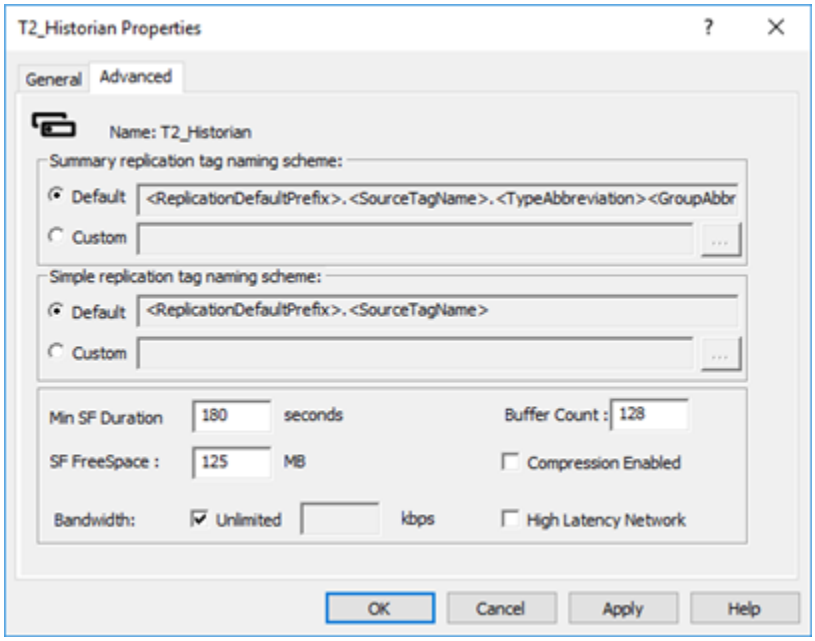
Important: Non-replicated data contains the replication server name as part of the store-and-forward identification information. Changing the node name/IP address can create orphaned data with no connection to the replication server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Also, if you have non-replicated data in the old store-and-forward path, changing the path can create orphaned data with no connection to the replication server.

If you change the store-and-forward path, you must commit the change to the database before the change will go into effect.

5. Click the **Advanced** tab.



6. Edit the advanced properties. The options are the same as for adding a new replication server. For more information, see Adding a Replication Server.

   When you change a naming scheme, the change is reflected for all subsequent tag entries, but the change does not affect any existing replication data already on the server.

7. When you are done, click **OK**.

Deleting a Replication Server

You can delete a replication server using the Operations Control Management Console. Deleting the server also automatically deletes the replication groups associated with the server.

WARNING! Be very certain that you are deleting the right replication server and that you have backed up data and tag information appropriately.

If you have configured a tag for replication to a server, you must first delete the tag replication before you can delete the server.

To delete a replication server

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers.**
3. Select the server in the details pane and perform any of the following:
   - Click the **Delete** button on the toolbar.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- On the **Action** menu, click **Delete**.
- Right-click the tag and then click **Delete**.

Configuring Tags to Be Replicated

Simple replication is when the destination tag on the next-tier historian has the same type and storage rules as the source tag on the lower-tier historian, and that the destination tag will also have the same values/data.

The rules for other kinds of tags also apply to replicated tags:

- Each tag is identified by a unique name.
- Configuration information for each type of tag is stored in the historian, as well as the history for tags over time.
- You can use the Configuration Editor to view and edit information for existing tag definitions, create definitions for new tags, or delete existing tags.

For more information about tags and tag naming conventions, see About Tags in the *AVEVA Historian Concepts Guide*.

You can create simple replication tags one at a time or in multiples.

If you delete a replication tag, the associated tag entities are deleted. Replication continues until you manually commit the change to the database or stop and restart the system. If you modify a replication tag, the changes are automatically committed.

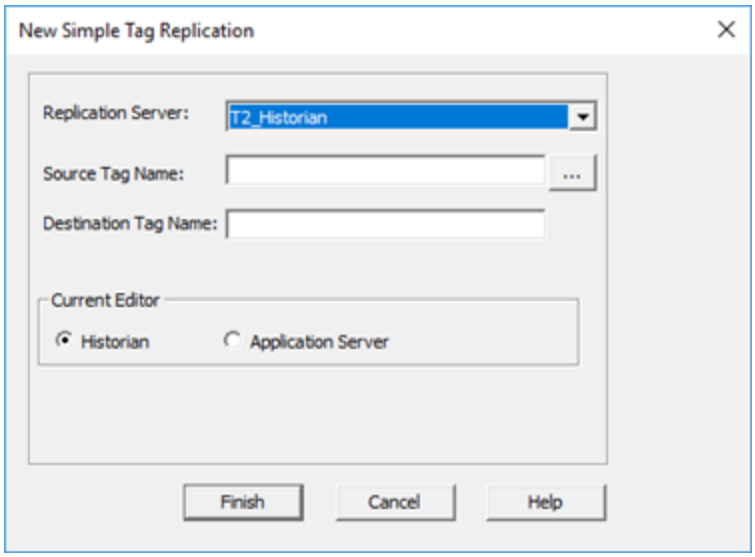
Adding a Single Tag for Simple Replication

You can configure a tag for simple replication. This is particularly useful if you have added or modified a tag, or are now using a tag as part of summary calculations, and just need to add a simple replication tag for that one tag.

To add a single simple replication tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, expand **Replication Servers**, then expand the replication server.
3. Right-click the **Simple Replication** folder and select **Add Single Tag**. The **New Simple Tag Replication** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. In the Replication Server box, select the replication server to create the tag on.
5. In the **Source Tag Name** box, type the name of the tag that provides the source data for the summary tag. (For more information on finding a tag, see Finding Source Tags.)
6. In the **Destination Tag Name** box, type the name of the simple replication tag.

Note: The **Current Editor** option is reserved for future use.

7. Click **Finish**. The simple replicated tag appears in the folder.

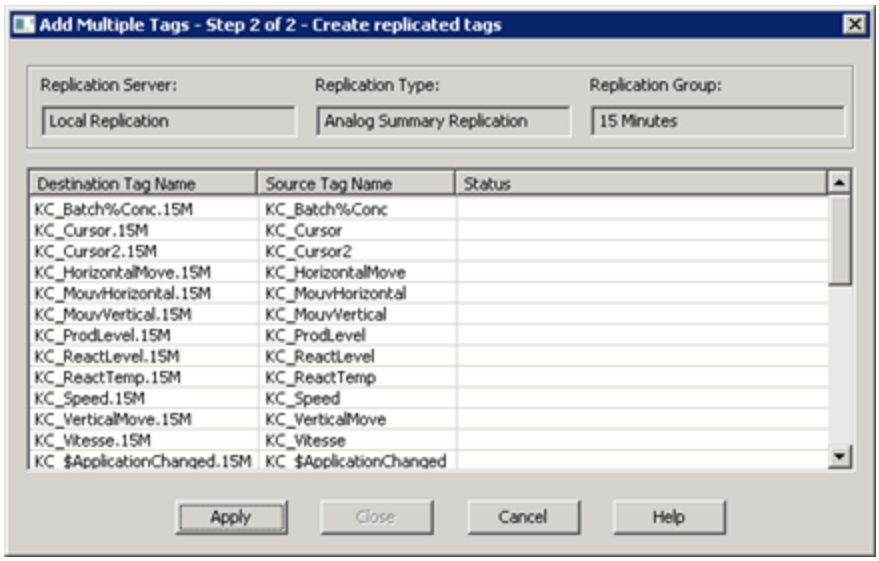
Adding Multiple Tags for Simple Replication

When you are setting up a new replication server or incorporating a new set of tags from an application, it is easier to configure tags for simple replication all at once rather than singly.

To add multiple simple replication tags

1. Start the Operations Control Management Console. You'll find it in the Start menu under AVEVA Utilities or AVEVA, depending on the version.
2. In the the left pane, expand a server group and then expand a server.
3. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, expand **Replication Servers**, then expand the replication server.
4. Right-click the **Simple Replication** folder and select **Add Multiple Tags**. The **Add Multiple Tags - Step 1 of 2** dialog box appears.
5. In the **Tag Name** box, select an option from the drop-down list. Check **Not** to negate the search option in the **Tag Name** box. Type the search string in the box to the right.
6. If you also want to search using a description, select an operator in the Operator box and then make entries in the **Description** boxes.
7. Click **Find Now** to start the search. Results matching the criteria appear in the **Found Tags** box.
8. Move tags from the **Found Tags** box to the **Target Tags** box by double-clicking them or highlighting the tags and using buttons to the right of the **Found Tags** box.
9. Click **OK**. The **Add Multiple Tags - Step 2 of 2** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



10. You can modify the destination tagname by editing it directly in the **Destination Tag Name** column.
11. Click **Apply**. The system adds the tags. A status indicating the outcome of operation appears for each tag in the Status field.

If you prefer, you can search for tags using a **SQL query**. Click the **SQL Query** tab, then use the same procedure as for adding a single summary tag using the **SQL Query** tab. For more information, see Adding a Summary Tag.

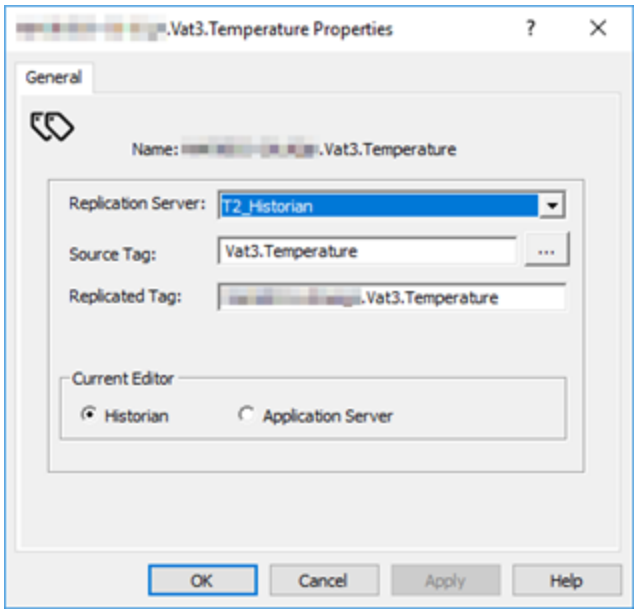
Editing Simple Replication Tag Properties

You can edit the properties for a simple replication tag. The general procedure is similar to the procedure for adding a simple replication tag.

To edit properties for a simple replication tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server, and then select the **Simple Replication** folder.
4. Double-click the tag you want to edit properties for. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In the **Replication Server** list, click the replication server.
6. In the **Source Tag** box, type the name of the tag that provides the source data for the tag. For more information on finding a tag, see Finding Source Tags.
7. In the **Replicated Tag** box, type the name of the replicated tag. The naming scheme shown for the replicated tag appears depends on the system parameter or as specified for the replication server.

   Note: The **Current Editor** option is reserved for future use.

8. Click **OK**. The tag's properties are updated.

Deleting a Simple Replication Tag

You can delete a simple replication tag through the Operations Control Management Console.

If you delete a replication tag from the lower-tier historian, data will no longer be replicated its corresponding next-tier historian. However, the replicated data for the deleted tag is not deleted from the next-tier historian for the simple replicated tag. Also, the replication tag is not deleted from the next-tier historian.

WARNING! Be very certain that you are deleting the right tag.

To delete a simple replication tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, expand **Replication Servers**, then expand the replication server, expand the **Simple Replication** folder.
3. Select the tag in the details pane and perform any of the following:
   - Click the **Delete** button on the toolbar.
   - On the **Action** menu, click **Delete**.
   - Right-click the tag and then click **Delete**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding a Replication Schedule

A replication schedule defines the interval or specific times for summary periods. Replication intervals are defined by a number of minutes or hours.

The AVEVA Historian sets up several standard replication schedules: 1, 5, 15, and 30 minutes, 1 hour, and 1 day. You can also add custom replication schedules to fit your own time requirements.

Replication is triggered at the schedule interval calculated from the beginning of the day (local time). For example, if you specify a 5-hour interval: replication will have summary values at 0:00, 5:00, 10:00, 15:00, 20:00 and again at 1:00. This is only the case for periodic schedules, not custom ones.

If a partial interval has elapsed at midnight, it will automatically trigger whether or not the interval has expired. All replication intervals cannot be longer than one day (24 hours).

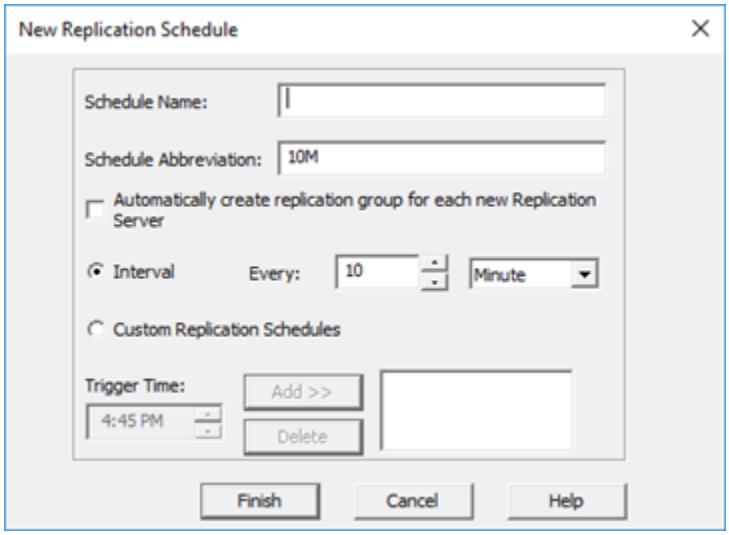
Custom replication schedules are a list of start times (local time) to trigger the summary. These will only trigger at the specified times and will not necessarily trigger at the end of the day.

You can also define a schedule abbreviation that is used when new replication groups are defined using the particular schedule. The schedule abbreviation is automatically generated as the interval is modified, but you can override the default. For example, with a tag named LevelTag1 configured for 7 minute replication using the schedule abbreviation 7m and being replicated from LocalServer1 to ReplServer2, the abbreviation is included in the destination tag name: LocalServer1.LevelTag1.7m.

The schedules you set up are expressed in local time and are impacted by Daylight Savings Time changes. For more information, see Replication Schedules and Daylight Savings Time .

To add a replication schedule

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Replication**.
3. Right-click **Replication Schedule** and select **New Replication Schedule**. The **New Replication Schedule** dialog box appears.



4. In the **Schedule Name** box, type the name of the schedule (up to 256 characters).
5. In the **Schedule Abbreviation** box, type the schedule abbreviation (up to 32 characters). This is used as part of the default naming scheme.
6. Check **Automatically create replication group for each new Replication Server** to have the Historian add this

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

schedule group to the default list of schedule groups whenever you create a replication server. (To add it to an existing server, you have to manually add the group.)

7. Select **Interval** and specify a time to set a regular time interval for replication. Alternatively, you can select **Custom Replication Schedules** and then enter specific trigger times for the replication schedule. A custom replication schedule can have up to 100 trigger times.
8. Click **Finish**. The new replication schedule appears in the replication schedule list.

Editing Replication Schedule Properties

You can edit the properties for a replication schedule. The general procedure is similar to the procedure for adding a replication schedule.

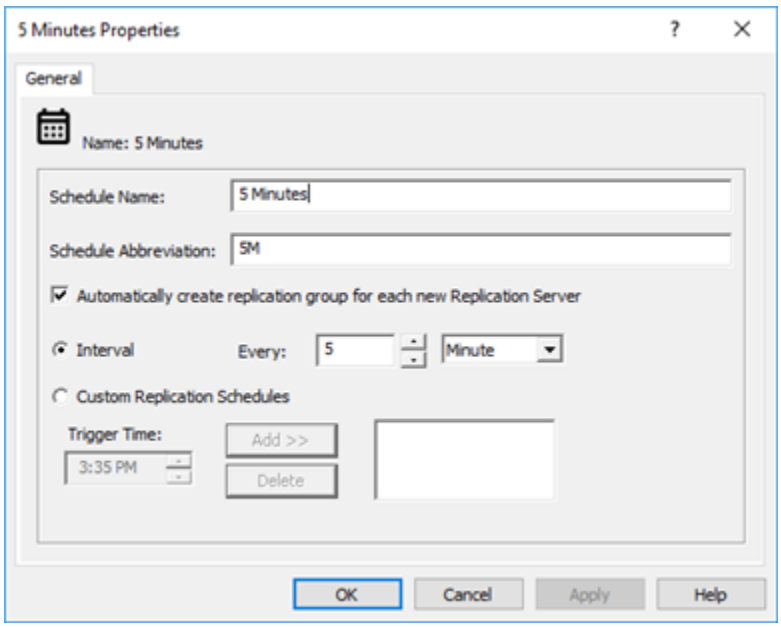
Replication schedules are referenced by replication groups rather than copied, so changes to a replication schedule affect all replication groups that reference that schedule. However, changes to the tag naming rules (such as the schedule abbreviation) do not affect any tags already configured for replication.

You can modify replication schedules when there are existing replication entities referencing the schedules or groups. When you change the schedule for a summary tag, an extra replication summary is sent at the time of the configuration change to transition to the new schedule.

If you change an interval replication schedule to a custom replication schedule, there is loss of data on the next-tier historian for the period from the last interval schedule to the first custom replication scheduled.

To edit properties for a replication schedule

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Schedule.**
3. Right-click the replication schedule to be modified and then click **Properties**. The **Properties** dialog box appears.



4. Edit the general properties. The options are the same as for adding a new replication schedule. For more information, see Adding a Replication Schedule.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. Click **OK**.

Deleting a Replication Schedule

You can delete a replication schedule through the Operations Control Management Console.

WARNING! Be very certain that you are deleting the right replication schedule.

If you have configured a replication schedule for use in a replication group, you must first remove the replication schedule from the group before you can delete the schedule.

To delete a replication schedule

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Schedules.**
3. Select the schedule in the details pane and perform any of the following:
   - Click the **Delete** button on the toolbar.
   - On the **Action** menu, click **Delete**.
   - Right-click the tag and then click **Delete**.

Adding a Replication Group

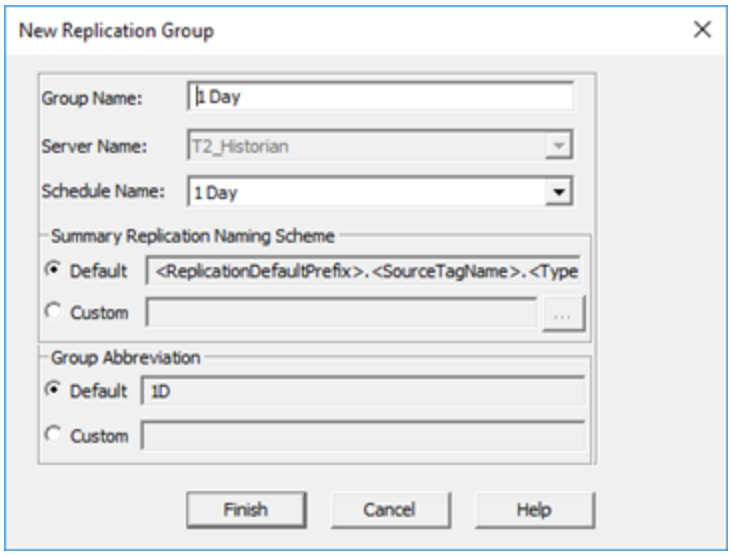
When you create a new replication group, you can add it to existing replication group configurations.

You can have an unlimited number of groups.

To add a replication group

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server you want to add the group to, and then right-click the folder you want to create the replication group in. Select **New Replication Group**. The **New Replication Group** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. In the Group Name box, type the name for the new group (up to 255 characters).
5. In the **Schedule Name** list, select an existing schedule to assign to the group.
6. In the **Summary Replication Tag Naming Scheme** area, select the replication tag naming scheme to use. Specify a custom naming scheme by selecting **Custom** and clicking the ellipsis button to the right of the box. The **Naming Scheme** dialog box appears. For information about configuring the naming scheme, see Specifying Naming Schemes for Replication.
7. In the **Group Abbreviation** area, select the default group abbreviation or type a custom group abbreviation.
8. Click **Finish**. The new replication group appears in the replication server's group list.

Editing Replication Group Properties

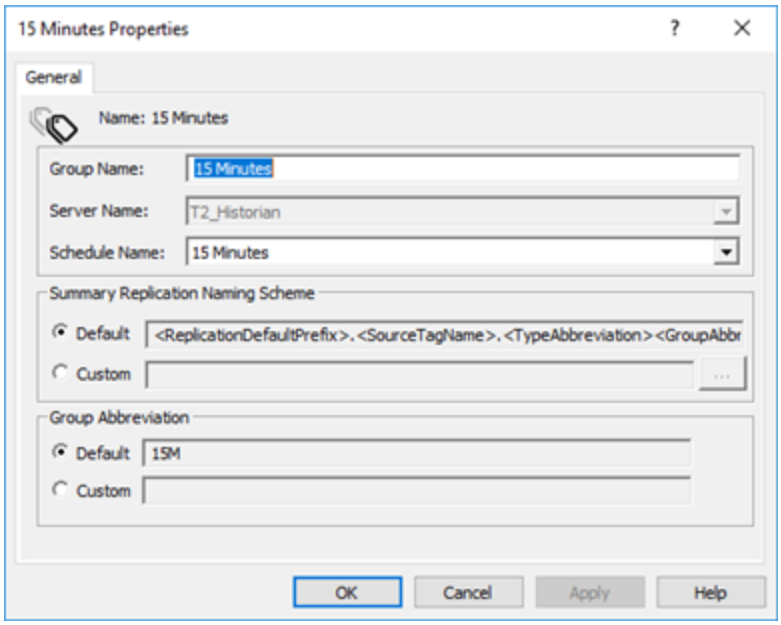
You can edit the properties for a replication group. The general procedure is similar to the procedure for adding a replication group.

You can modify replication groups when there are existing replication entities referencing the schedules or groups. When you change the schedule for a summary tag, an extra replication summary is sent at the time of the configuration change to transition to the new schedule.

To edit properties for a replication group

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server, and then expand the folder the replication group is in.
4. Right-click the replication group to be modified and then click **Properties**. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Edit the general properties. The options are the same as for adding a new replication group. For more information, see Adding a Replication Group.
6. Click **OK**. The replication group's properties are updated.

Deleting a Replication Group

You can delete a replication group through the Operations Control Management Console.

If you have configured replication tags in a replication group, you must first remove the replication tags from the group before you can delete the replication group.

If the replication group contains replication entities, you cannot delete the replication group.

WARNING! Be very certain that you are deleting the right replication group and that you have backed up data and tag information appropriately.

To delete a replication group

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server you want to delete the group from.
4. Select the group in the details pane and perform any of the following:
   - Click the **Delete** button on the toolbar.
   - On the **Action** menu, click **Delete**.
   - Right-click the tag and then click **Delete**.

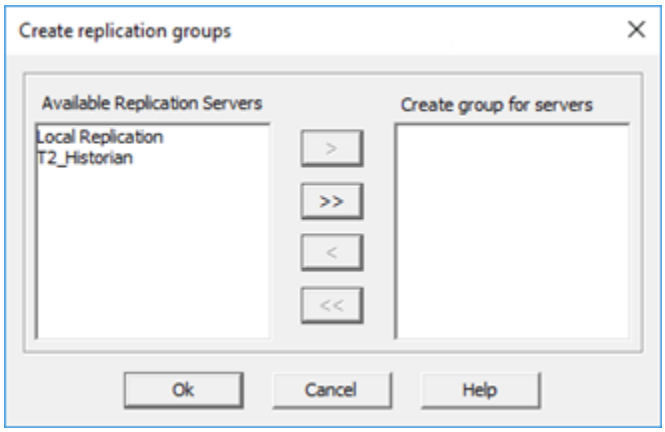
Creating a Replication Group for Multiple Servers

You can quickly create the same replication group for multiple servers. The replication group is based on a replication schedule that you select.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To create a replication group for multiple servers

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Replication**.
3. Select **Replication Schedules**.
4. In the details pane, right-click the desired schedule and then click **Create replication groups**. The **Create replication groups** dialog box appears.



5. Use the arrow buttons to move one or more replication servers from the **Available Replication Servers** window to the **Create group for servers** window.
6. Click **OK**.

About Summary Replication

Summary replication takes information from a source tag on a lower-tier historian and summarizes the tag data to a next-tier historian in one of two types of summary tag:

- Analog summary tag
- State summary tag

If a lower-tier historian is unable to perform a scheduled summary calculation for any reason, it adds a record about the event into a replication queue. When there are enough system resources available, or there is a specific event from another subsystem, the lower-tier historian can perform the summary calculations and clear the queue.

You can add analog or state summary tags one at a time or in multiples.

State summary replication is not supported for scaled tags.

About Analog Summary Replication

Analog summary replication produces summary statistics for analog tags. The statistics relate only to the recorded interval. Statistics available are:

- Time-weighted average
- Standard deviation

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Integral
- First value in a period with timestamp
- Last value in a period with timestamp
- Minimum value in a period with timestamp
- Maximum value in a period with timestamp
- Start time of summary period
- End time of summary period
- OPC Quality
- Percentage of values with Good quality
- Value

When you retrieve the data, you specify which calculation you want to return. For more information, see Querying the AnalogSummaryHistory View in the *AVEVA Historian Retrieval Guide*.

The functionality provided by analog summary replication is similar to using the minimum, maximum, average, and integral retrieval modes. For a comparison example, see Querying Aggregate Data in Different Ways in the *AVEVA Historian Retrieval Guide.*

When you use the AVEVA Historian SDK to retrieve analog summary tag data, the values returned through the SDK for analog summary tags from history correspond to the "Last" values in the AnalogSummaryHistory table when using defaults. Use the corresponding retrieval mode to get the minimum, maximum, average, slope, and integral values.

About State Summary Replication

State summary replication summarizes the states of a tag value. State summary replication can be applied to analog (integer only), discrete, and string tags.

You use this for analyzing process variables with a limited number of states, such as a machine's state of running/starting/stopping/off. State summary replication provides the following, for each distinct state:

- Total time
- Percent of the cycle
- Shortest time
- Longest time
- Average time
- OPC Quality
- Value

A state summary results in a series of values, each representing a different state, for the same tag and time period.

When you retrieve the data, you specify which calculation you want to return. For more information, see Querying the StateSummaryHistory View in the *AVEVA Historian Retrieval Guide*.

The functionality provided by analog summary replication is similar to using the ValueState and RoundTrip retrieval modes.

You can define state summary replication for a large number of states, but state data is dropped if the number of

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

states occurring in the same reporting period exceeds the maximum number of states allowed. You configure the maximum states when you create the state summary tag. The default number of maximum states is 10. The Replication subsystem will calculate summaries for the first 10 distinct states, in the order in which they occur in the data (not in numeric or alphabetic order). Be aware that the higher the number of maximum states, the more system resources are used to keep track of the time-in-state for each distinct state.

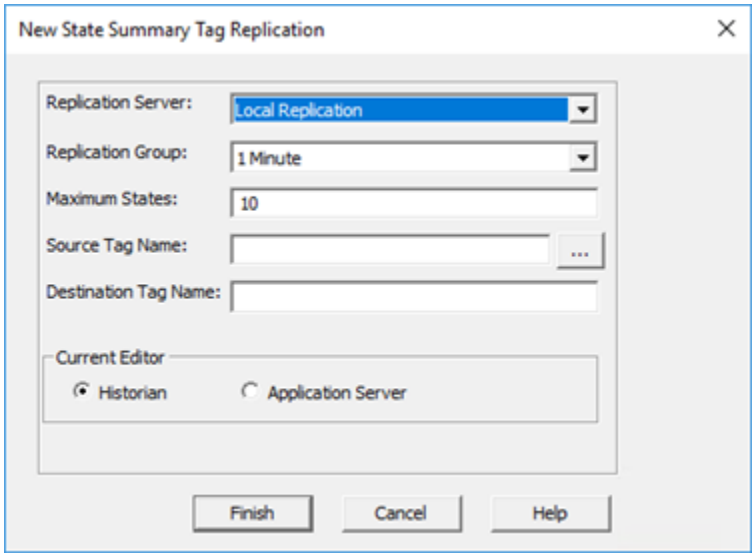
Adding a Summary Tag

You can add a single analog or state summary tag. This is particularly useful if you have added or modified a tag and simply need to add a summary tag for that one tag.

WARNING! If you are replicating from multiple source tags to the same summary tag on a next-tier replication server, the next-tier logger does not log a message of the naming conflict. As a result, it is possible to have multiple source tags overwriting each other in the same summary tag. Make sure you have a naming convention for your destination tags that avoids potential name collisions.

To add a single summary tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server, then expand the **Analog Summary Replication** or **State Summary Replication** folder.
4. Right-click the replication group and select **Add Single Tag**. The **New Analog Summary Tag Replication** dialog box or **New State Summary Tag Replication** dialog box appears.



5. In the **Replication Server** box, select the replication server to create the tag on.
6. In the **Replication Group** box, select the replication group to assign the tag to.
7. If you are defining a state summary tag, in the **Maximum States** box, type the maximum number of states allowed in the same reporting period. The default number of maximum states is 10. The replication subsystem will calculate summaries for the first 10 distinct states, in the order in which they occur in the data (not in numeric or alphabetic order). The higher the number of maximum states, the more system resources are used to keep track of the time-in-state for each distinct state.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. In the **Source Tag Name** box, type the name of the lower-tier tag that provides the source data for the summary tag. For more information on finding a tag, see Finding Source Tags.
9. In the **Destination Tag Name** box, type the name of the new next-tier summary tag. By default, the destination tag name appears as configured for the SummaryReplicationNamingScheme system parameter or as configured for the replication server.

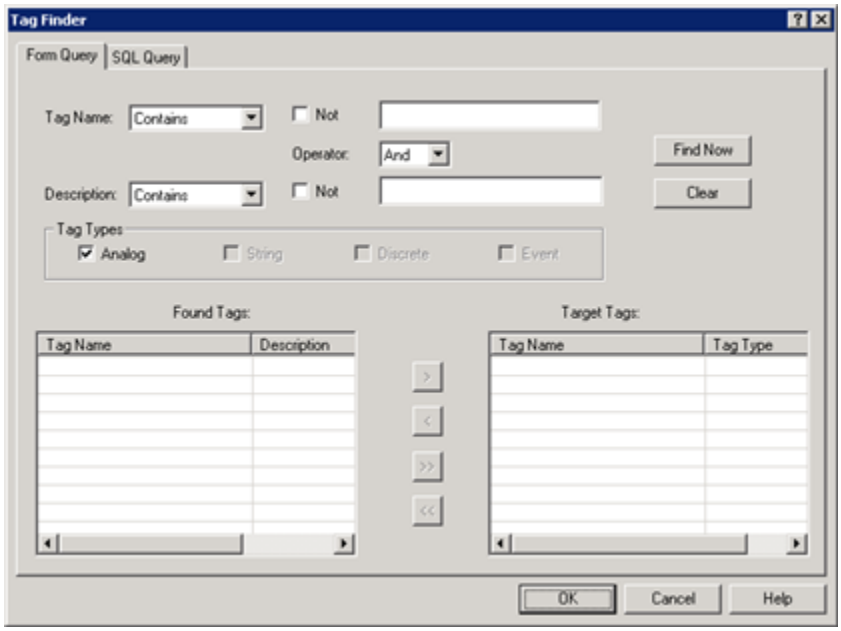
   Note: The **Current Editor** option is reserved for future use.
10. Click **Finish**. The new summary tag appears in the selected replication group.

Finding Source Tags

If you aren't sure of the exact tag name or need to verify which tag to use, you can search for tags as part of the process of adding summary tags.

To find a tag to use as the source tag

1. Click the button to the right of the **Source Tag Name** box. The **Tag Finder** dialog box appears.



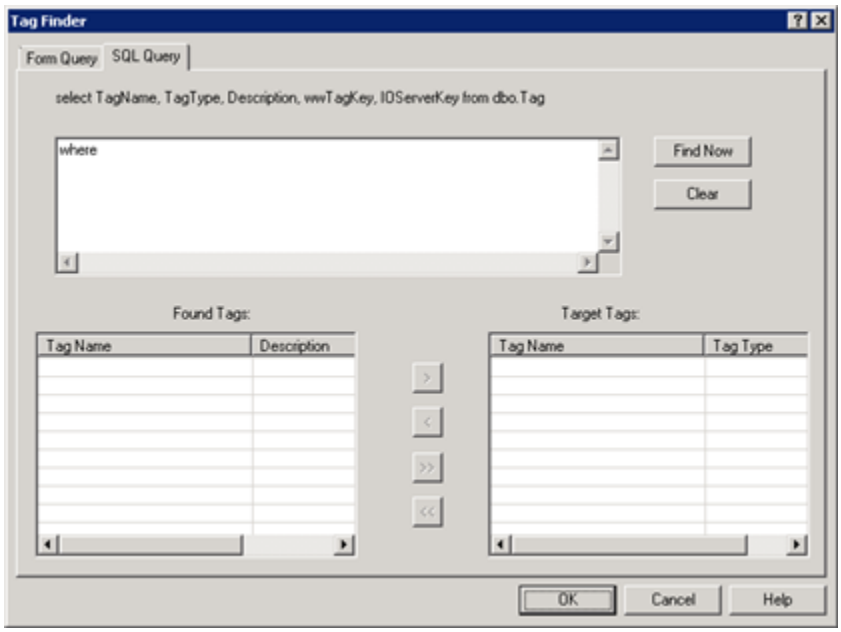
2. In the **Tag Name** box, select an option from the drop-down list. Check **Not** to negate the search option in the **Tag Name** box. Type the search string in the box to the right.
3. If you also want to search using a description, select an operator in the Operator box and then make entries in the **Description** boxes.
4. Click **Find Now** to start the search. Results matching the criteria appear in the **Found Tags** box.
5. Double-click a tag in the **Found Tags** box to move it to the **Target Tags** box. You can also use the > and < buttons to move tags between the boxes.
6. Click **OK**. The selected tag appears in the **Source Tag Name** box on the **Add Analog Summary Tag Replication** dialog box or the **Add State Summary Tag Replication** dialog box.

If you prefer, you can search for tags using a SQL query.

1. Click the button to the right of the **Source Tag Name** box. The **Tag Finder** dialog box appears.

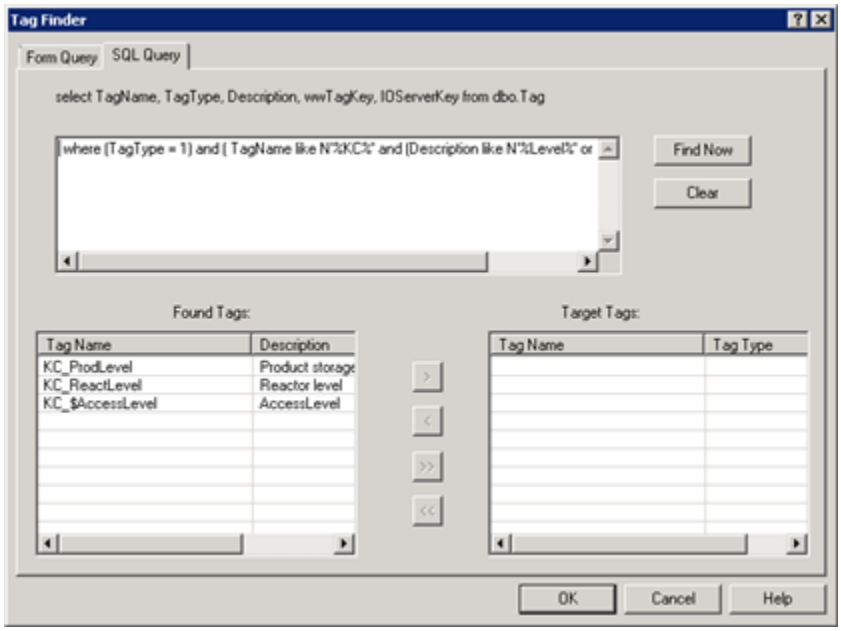
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Click the **SQL Query** tab.



3. Type a SQL query in the box and then click **Find Now**. The qualifying tags appear in the **Found Tags** box.

   If you want an example of how this works for your system, use the preceding procedure to select tags based on a string and/or description, then look at the **SQL Query** tab. The search criteria appear in a SQL query. For example, the following screen shows the results of searching for tags containing the string "KC" and a description containing the string "level."



4. Select a tag and click **OK** as described in the previous procedure. The selected tag appears in the **Source Tag Name** box on the **New Analog Summary Tag Replication** dialog box or the **New State Summary Tag Replication** dialog box.

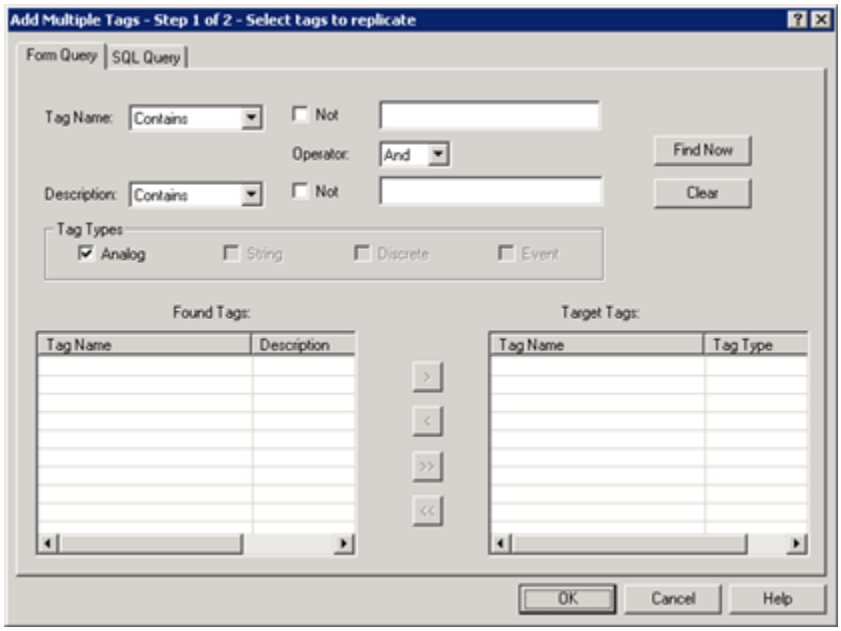
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding Multiple Summary Tags

When you are setting up a new replication server or incorporating a new set of tags from an application, it is easier to add summary tags all at once rather than singly.

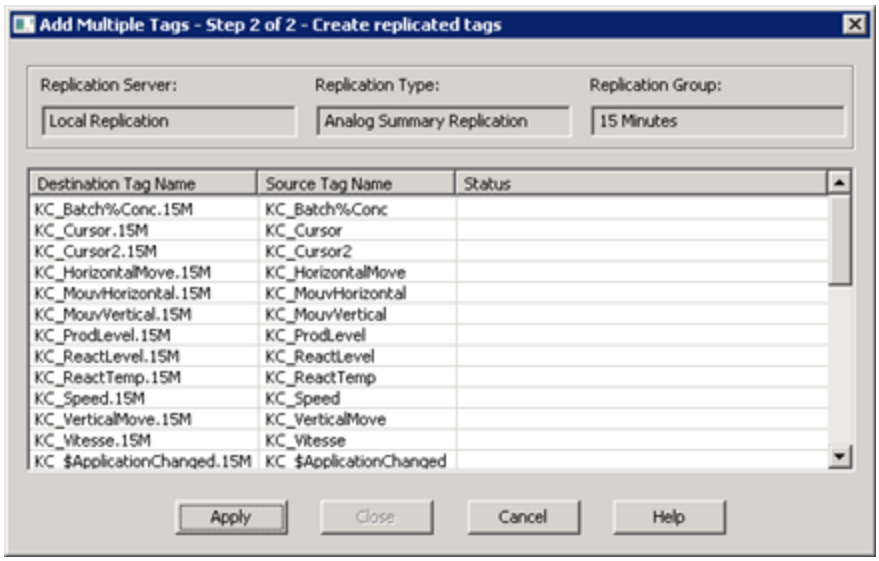
To add multiple summary tags

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server, then expand the **Analog Summary Replication** or **State Summary Replication** folder.
4. Right-click the replication group you want to add the tag to and select **Add Multiple Tags**. The **Add Multiple Tags - Step 1 of 2 - Select Tags to Replicate** dialog box appears.



5. In the **Tag Name** box, select an option from the drop-down list. Check **Not** to negate the search option in the **Tag Name** box. Type the search string in the box to the right.
6. If you also want to search using a description, select an operator in the Operator box and then make entries in the **Description** boxes.
7. Click **Find Now** to start the search. Results matching the criteria appear in the **Found Tags** box.
8. Move tags from the **Found Tags** box to the **Target Tags** box by double-clicking them or highlighting the tags and using the buttons to the right of the **Found Tags** box.
9. Click **OK**. The **Add Multiple Tags - Step 2 of 2 - Create replicated tags** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



10. Check the tag names. You can edit the destination tag names in this dialog box by selecting the specified tag and clicking the name to modify in the **Destination Tag Name** column. By default, the destination tag specified by the system parameter appears in this dialog box. The destination tagname can be modified by editing it directly in the **Destination Tag Name** column.
11. Click **Apply**. The system adds the tags to the specified replication group. A status indicating the outcome of operation appears for each tag in the **Status** column.

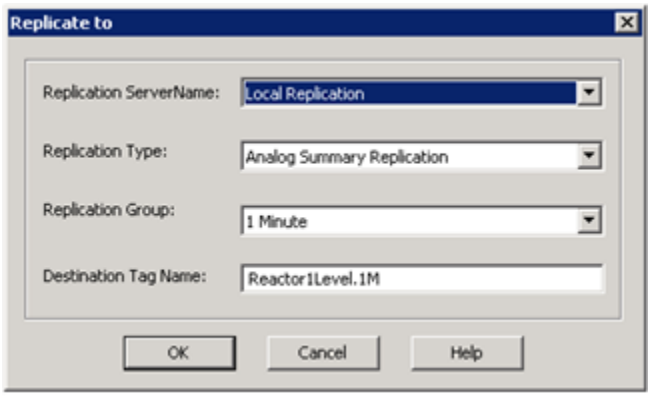
If you prefer, you can search for tags using a **SQL query**. Click the **SQL Query** tab, then use the same procedure as for adding a single summary tag using the **SQL Query** tab. For more information, see Adding a Summary Tag.

Creating a Summary Tag Quickly Using Default Settings

You can select an existing tag on a lower-tier server and then quickly configure a summary tag.

To create a replication tag

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select the appropriate folder. For example, **Analog Tags**.
4. In the details pane, right-click the source tag for which you want to create an associated summary tag and then click **Replicate To**. The **Replicate To** dialog box appears.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

5. In the **Replication Server Name** list, click the replication server.
6. In the **Replication Type** list, click the type of replication, either **Analog Summary Replication** or **State Summary Replication**.
7. In the **Replication Group** list, click the replication group.
8. In the **Destination Tag Name** box, type the name of the summary tag. By default the destination name appears as configured for the replication server.
9. Click **OK**. The **Operation Result** dialog box appears, showing the details of the source tag, destination tag, and the status of the operation.
10. Click **Close**.

Editing Summary Tag Properties

You can edit the properties for a summary tag. The general procedure is similar to the procedure for adding a summary tag.

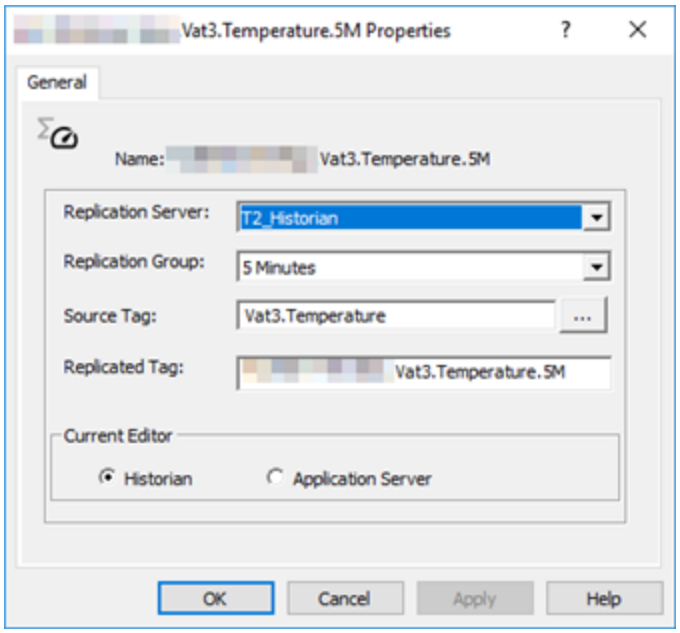
In case of a change that would result in a tag name collision with tags defined on the same lower-tier historian going to the same next-tier historian, the change is discarded and an error message appears. Collisions with replication tags configured from other lower-tier historians are overwritten without an error message from the next-tier historian. Be careful not to create or modify a replicated tag on a lower-tier historian to have the same tagname that already exists on a next-tier historian. The system does not prevent you from having a replicated tag on a next-tier historian receiving data from multiple lower-tier historians. However, when you retrieve data for that replicated tag on the next-tier historian using the tag name, an incorrect blend of data from the two (or more) data sources is returned. Collisions of non-replication tags on the next-tier historian fail at connection time and are logged to the ArchestrA Logger.

When you change the replication server node name or replication group, the replication tag is removed from the old group and added to the new group. Replication tags that are changed to a new replication server or group do not have their changes logged to the new server or group until changes are committed.

To edit properties for a summary tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server, expand the **Analog Summary Replication** folder or **State Summary Replication** folder, and then expand the replication group containing the summary tag.
4. Double-click the summary tag you want to edit properties for. The **Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. In the **Replication Server** list, click the replication server to edit.
6. In the **Replication Group** list, click the replication group.
7. In the **Source Tag** box, type the name of the tag that provides the source data for the summary tag. For more information on finding a tag, see Finding Source Tags.
8. In the **Replicated Tag** box, type the name of the summary tag.

   Note: The **Current Editor** option is reserved for future use.
9. Click **OK**.

Deleting Replication for a Summary Tag

You can delete replication for a summary tag through the Operations Control Management Console.

The data of the summary tag at the next-tier historian is not deleted when you delete a summary tag at the lower-tier historian. You can query for the summary tag on the next=tier historian for data as long as the summary tag exists on the next-tier historian.

WARNING! Be very certain that you are deleting the right summary tag.

To delete replication for a summary tag

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, expand **Replication**, then expand **Replication Servers**.
3. Expand the replication server, expand the **Analog Summary Replication** folder or the **State Summary Replication** folder, and then expand the replication group containing the summary tag.
4. Select the tag in the details pane and perform any of the following:
   - Click the **Delete** button on the toolbar.
   - On the **Action** menu, click **Delete**.
   - Right-click the tag and then click **Delete**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding Replication or Auto-Summary Values for a Defined Timeframe

You can use the Replication Backfill Manager to backfill the following types of historic data:

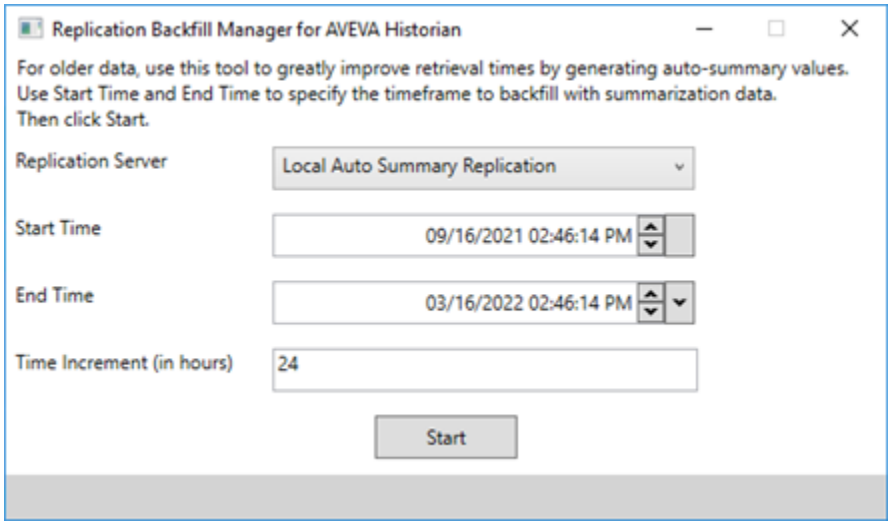
- **Auto-summary:** Beginning with AVEVA Historian 2017 (version 17.0.18000), Historian automatically creates a one-hour summary tag for every analog tag in the system. If you have historical data originally collected by Historian 2014 R2 or earlier, by default no auto-summary data will be calculated for the history blocks prior to upgrading to Historian 2017 or later. For example, if you ran Historian 2014 R2 or earlier from January 2010 until April 2020, then you migrated to Historian 2020, you'll only have auto-summary data beginning in April 2020. If you want to create auto-summary data for the prior 10 years of history blocks, use the backfill utility.
- **Replication:** If you have historical data originally collected prior to configuring replication, by default no older data will be replicated. For example, if you collected data in Historian from January 2020 and then configured replication to another Historian in April 2021 and to AVEVA Insight in November 2022, you'll only have replicated data in the other Historian beginning in April 2021 and in AVEVA Insight starting in November 2022. Use the backfill utility to replicate older data.

By default, the Replication Backfill Manager (aahBackfillUI.exe) is located in this folder:

```
C:\Program Files (x86)\Wonderware\Historian\x64
```

To generate replication or auto-summary data

1. Open the Replication Backfill Manager (aahBackfillUI.exe).



2. Select a **Replication Server** from the list. You can backfill data to any replication server defined in the Operations Control Management Console.
3. Choose the **Start Time** and **End Time** of the period for which you want data to be generated. The default is to generate data for the previous 6 months.
4. Specify the **Time Increment (in hours)** for how much time each summary tag will summarize. The default is 24 hours per summary tag.
5. Click **Start** to begin the process.

There is no need to keep the utility open once it has initiated the backfill process. But if you reopen the utility, you can see the replication progress.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

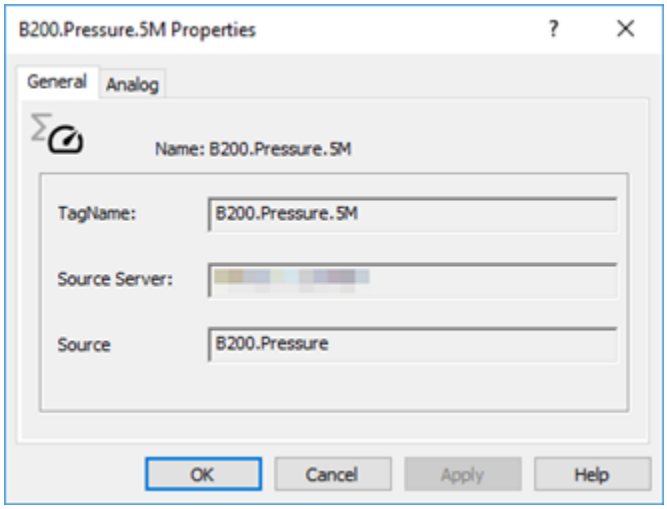
The progress bar indicates the progress of the synchronization queue requests sent to for auto-summary processing. It does not indicate that all the data has been processed and summarized in the auto-summary blocks. The processing will complete silently outside of the utility interface.

Viewing Source Details for a Summary Tag

You can view the source details for an analog or state summary tag, such as the source tag, source historian, and the engineering unit and raw value information of the source tag from a next-tier historian.

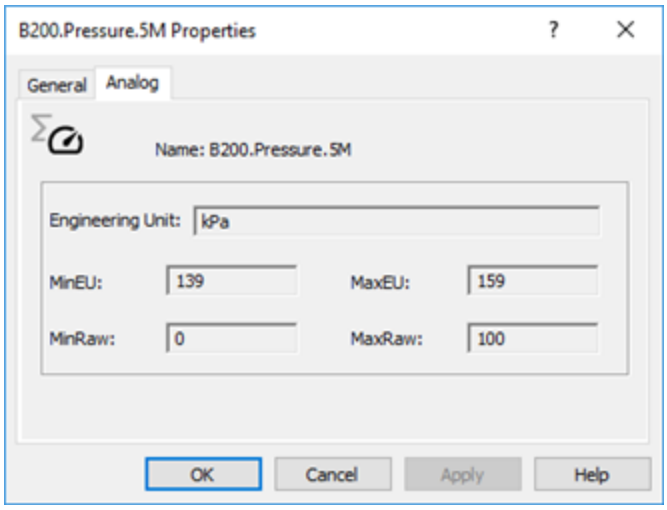
To view details for a summary tag from a next-tier historian

1. In the Operations Control Management Console on the next-tier historian, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Click either the **Analog Summary Replication** folder or the **State Summary Replication** folder.
4. In the details pane, right-click the summary tag for which you want to view details. The **Properties** dialog box appears.



5. Click the **General** tab. The following properties are shown:
   - The **TagName** box shows the name of the summary tag.
   - The **Source Server** box shows the name of the lower-tier historian from which the source data originates.
   - The **Source** box shows the name of the tag that provides the source data for the summary tag.
6. If you have selected an analog summary tag, click the **Analog** tab and view the properties.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- The **Engineering Unit** box shows the unit of measure. Examples are mph, grams, and pounds.
- The **MinEU** box shows the minimum value of the tag, measured in engineering units.
- The **MaxEU** box shows the maximum value of the tag, measured in engineering units.
- The **MinRaw** box shows the minimum value of the raw acquired value.
- The **MaxRaw** box shows the maximum value of the raw acquired value.

7. Click **OK**.

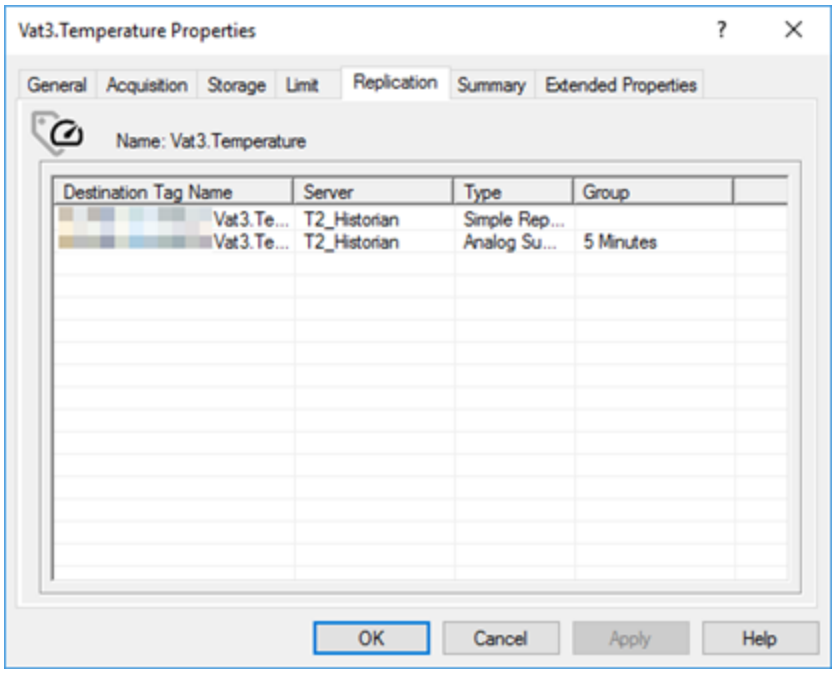
Viewing the List of Associated Replicated Tags for a Tag

You can view the list of all of the replicated tags that are configured/based on a particular source tag from a lower-tier historian.

To view the list of replicated tags for a source tag

1. In the Operations Control Management Console on the lower-tier historian, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select the type of tag for which you want to view the summary tags.
4. In the details pane, double-click the tag. The **Properties** dialog box appears.
5. Click the **Replication** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



A list is shown of all of the simple replicated, analog summary, and state summary tags that use this tag as a source.

6. Click **OK**.

Managing Security

About Security

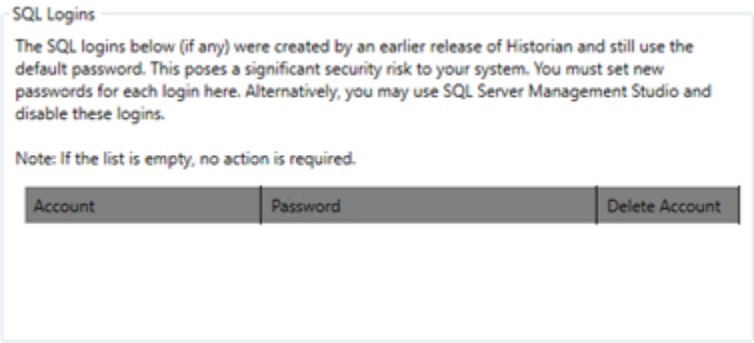
The AVEVA Historian uses two security mechanisms:

- Windows operating system security
- Microsoft SQL Server security

Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.

Note: During configuration, if you are upgrading from an earlier release, Historian requires that you change the passwords for any legacy default SQL logins, such as wwUser. When this is required, it is clearly indicated during configuration of security options.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



For clients to access the historian, they must pass through both of these security levels.

The historian Management Console (within the Operations Control Management Console) adds an additional layer of authorization to restrict access to functions affecting the state of the historian to only authorized users. For example, you can grant a Windows user account access to start and stop historian services by assigning it the Historian Power Users role, or by adding it to the aaPowerUsers Windows user group. For more information, see Adding Users and Assigning Roles.

Note: Some historian components require Windows and SQL Server logins.

For more information on configuring user rights assignments for local security policies, see the Microsoft documentation.

Security for AVEVA Historian is managed using the following tools:

- **Microsoft SQL Server Management Studio.**
  Use this application to manage access to the SQL Server and databases.
- **Windows Local Users & Groups MMC snap-in.**
  Use this to manage permissions on the historian and for the OData/REST web service interface. You can also use it as an alternative to configuring permissions within the database when using Windows authentication.

  For more information, see Managing Logins.
- **ArchestrA Change Network Account utility.**
  Use this utility to modify the Windows login for the historian services on remote servers. For example, if you are configuring an AppEngine to send data to a remote historian, this utility is used to choose the Windows user account with which to connect to the server.

Windows Operating System Security

To log on to AVEVA Historian as a client, the first thing a user must be able to do is to log on to the operating system on their computer. For the Windows operating system, a valid user account, consisting of a login ID (username) and password, is required to log on to the computer.

SQL Server also requires authentication in order for clients to connect to it. You can use either Windows authentication or SQL Server authentication. For more information, see SQL Server Security.

Default Windows User Account for AVEVA Historian Services

All of the modules in AVEVA Historian, except for the Management Console and the Configuration Editor, run as Windows services. During installation, these services are configured to run with either the built-in Network Service account, or with virtual service accounts whose passwords are managed automatically by the system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQL Server Security

Because the AVEVA Historian works closely with Microsoft SQL Server, it uses and takes advantage of the security features that Microsoft SQL Server has to offer. The purpose of security for a SQL Server is to control who can access the server, access specific databases within a server, and perform certain actions within a database.

A database user must pass through two stages of security for the historian:

- Authentication, which validates the user's identity to the server itself.
- Database authorization, which controls the database(s) that user can access, as well as the types of actions that the user can perform on objects within the database.

User authentication and database authorization are managed from Microsoft SQL Server Management Studio.

To access information in the AVEVA Historian databases, users need to be granted access to the databases. The historian is shipped with preconfigured database roles and user accounts to serve as a starting point for your security model. Database roles, SQL Server login IDs, and user accounts are managed using the Microsoft SQL Server Management Studio.

Authentication

Microsoft SQL Server authenticates users with individual login account and password combinations. After the user's identity is authenticated, if authentication is successful, the user is allowed to connect to a SQL Server instance. There are two types of authentication:

- **Windows authentication**
  Uers must connect to the SQL Server using a Windows user account (a Windows login ID and password that is provided during the Windows login session).
- **SQL Server authentication**
  Users must connect to the SQL Server using SQL Server login account (a SQL Server login ID and password).

SQL Server can operate in one of two security modes, which control the type of accounts that must be used for access to the server:

- **Windows authentication mode**
  The SQL Server only uses Windows authentication.
- **Mixed mode**
  The SQL Server uses both Windows authentication and SQL Server authentication. When connecting, users can choose to log in with a Windows user account or a SQL Server login.

  Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.

For more information about authentication, see your Microsoft SQL Server documentation.

Default Windows Security Groups

The following Windows security groups are created by default on the AVEVA Historian computer. Use these groups to assign different levels of database permissions to users.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- aaAdministrators
- aaPowerUsers
- aaUsers
- aaReplicationUsers

Each group is automatically configured to be a member of the SQL Server database role with the same name. For example, the aaAdministrators Windows security group is a member of the default aaAdministrators SQL Server database role. If you add Windows users to the aaAdministrators security group, they will automatically be given permissions of the aaAdministrators SQL Server database role.

AVEVA Historian Default Logins

When the AVEVA Historian is installed, default SQL Server logins are created that you can use for logging on to the historian from client applications. These default logins provide "out of the box" functionality in that you do not have to create logins to start using the system. The following table describes the pre-configured logins:

Login Name	Description
aaAdmin	A user who can access and modify all data and create objects. Cannot drop the database or truncate tables.
aaPower	A user with full read access and the ability to create objects and modify the contents of the non-core tables.
aaUser	A read-only user who can access all data, but cannot modify data or consume database resources.
aadbo	Database owner. Full permissions.

The default database for each of these logins is the historian Runtime database. This default security model is provided as a starting point for system security and is suitable for many types of installations.

These logins are valid if the Microsoft SQL Server is set to mixed mode security. If only Windows authentication is used, you must configure the access rights for each user.

Important: Never use blank passwords for logins.

The following logins are provided for backward compatibility only. They will be deprecated in a future release. Do not use these logins.

Login Name	Description
wwUser	Same as aaUser.
wwPower	Same as aaPower.
wwAdmin	Same as aaAdmin.
wwdbo	Same as aadbo.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Database Authorization

After a user successfully connects to the Microsoft SQL Server, the user needs authority to access databases on the server. This is accomplished by user accounts for each database. A database user consists of a user name and a login ID. Each database user must be mapped to an existing login ID.

User names are stored in the sysusers table in each database. When a user tries to access a database, the Microsoft SQL Server looks for an entry in the sysusers table and then tries to find a match in the syslogins table in the master database. If the Microsoft SQL Server cannot resolve the username, database access is denied.

The types of actions the user can perform in the database are based on authority information defined in the user account. The authority to perform a certain action is called a permission. There are two types of permissions: object permissions and statement permissions.

Permission	Description
Object	Regulates the actions that a user can perform on certain database objects that already exist in the database. Database objects include things such as tables, indexes, views, defaults, triggers, rules, and procedures. Object permissions are granted and revoked by the owner (creator) of the object.
Statement	Controls who can issue particular Transact-SQL statements. Database statements include commands such as SELECT, INSERT, or DELETE. Statement permissions, also called command permissions, can only be granted and revoked by the system administrator or the database owner.

Users can be grouped into roles, which is a single unit against which you can apply permissions. Permissions granted to, denied to, or revoked from a role also apply to any members of the role.

AVEVA Historian Default Users and Roles

AVEVA Historian is shipped with a number of preconfigured user accounts and roles.

Note: During installation, Historian requires that you change the passwords for any default login accounts, such as wwUser. Use of default passwords (which are often published in various documents) is highly discouraged.

The following table describes the default SQL Server usernames, the login IDs and database roles to which they belong, and the actions that they are allowed to perform in the Runtime database. You can add additional users and roles using SQL Server Enterprise Manager.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Login ID	Username in Database	Member of Role	Permissions
aaUser	aaUser	aaUsers	SELECT on all tables INSERT, UPDATE, DELETE on PrivateNameSpace and Annotation
aaPower	aaPower	aaPowerUsers	CREATE Table CREATE View CREATE Stored procedure CREATE Default CREATE Rule SELECT on all tables INSERT, UPDATE, DELETE on grouping tables
aaAdmin	aaAdmin	aaAdministrators	CREATE Table CREATE View CREATE Stored procedure CREATE Default CREATE Rule DUMP Database DUMP Transaction SELECT, INSERT, UPDATE, DELETE on all tables
aadbo	dbo	db_owner	Full database owner capabilities

The following users and roles are provided for backward compatibility only. They will be deprecated in a future release. Do not use these users and roles.

Login ID	Username in Database	Member of Role	Permissions
wwUser	wwUser	wwUsers	Same as for aaUser.
wwPower	wwPower	wwPowerUsers	Same as for aaPower.
wwAdmin	wwAdmin	wwAdministrators	Same as for aaAdmin.
wwdbo	wwdbo	db_owner	Same as for aadbo.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Each default role contains the corresponding SQL Server user account, as well as the corresponding default Windows security group. For more information on the default Windows security groups, see Default Windows Security Groups.

Default SQL Server Login for AVEVA Historian Services

Some components of the AVEVA Historian require a SQL Server login ID to access the master, Runtime, and Holding databases. By default, the historian uses the Network Account (previously called the ArchestrA User) to log on to the Microsoft SQL Server, using Windows authentication.

For Microsoft SQL Server, if the Windows user account is an administrative account on the local computer, it will map the account to the sysadmin fixed server role. (This user will be granted the same permissions as the **sa** SQL Server user account.) Because the Network Account is always a local administrative account, it will always have administrative permissions (sysadmin) within the SQL Server.

Management Console Security

The AVEVA Historian Management Console (which is part of the overall Operations Control Management Console) runs in the context of the logged on Windows user account. To protect against unauthorized access to the AVEVA Historian, you must specify a separate Windows user account that the Management Console will use to connect to the historian. You can specify this account when you set up the server registration properties. For more information on registration, see About Administrative Tools.

If the account specified is not a member of the local administrators group on the computer hosting the historian, the Management Console has "read-only" access. That is, you may view all the information shown in the Management Console, but you cannot perform any control actions on the historian, such as starting or stopping the system.

Important: To prevent possible unauthorized access, the password for the Management Console login account must NOT be blank.

Verifying the Authentication Mode for a SQL Server

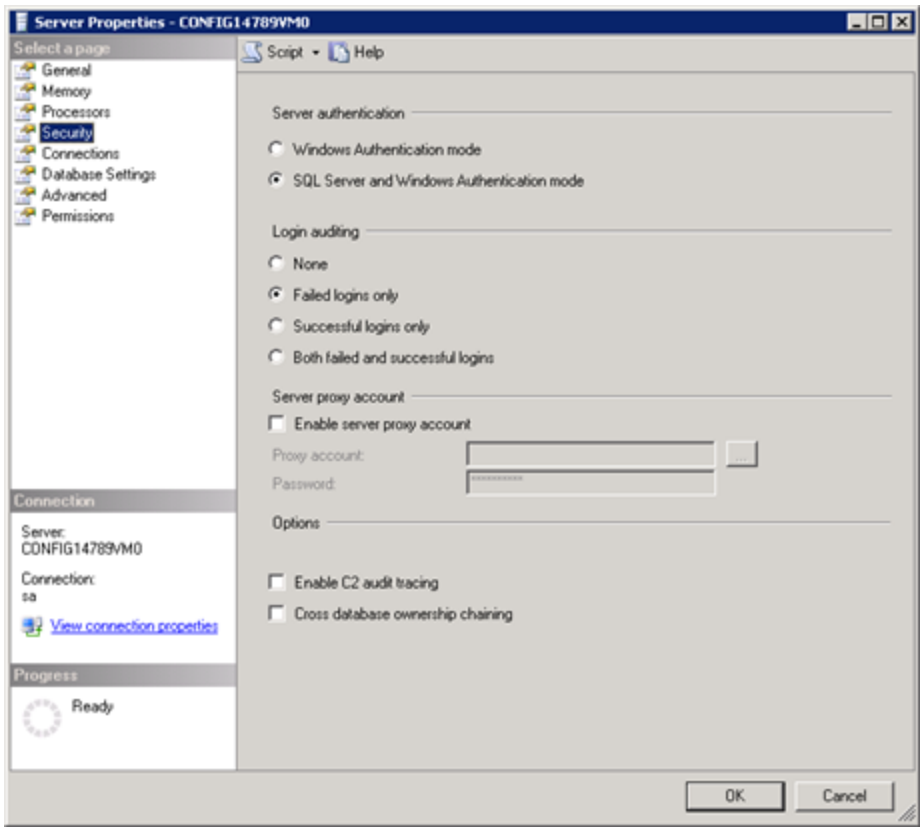
The AVEVA Historian is compatible with either Windows authentication mode or mixed mode (Windows authentication and SQL Server authentication).

Important: SQL Server authentication is provided for backward compatibility only. For improved security, we recommend that you use Windows authentication.

To verify the authentication mode

1. Start SQL Server Management Studio.
2. In the console tree, right-click the SQL Server.
3. In the shortcut menu that appears, click **Properties**. The **SQL Server Properties** dialog box appears.
4. Click the **Security** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Verify the authentication mode.

   The **SQL Server and Windows** option corresponds to mixed mode authentication. If you change the authentication mode, you must stop and restart the SQL Server. Also, modification tracking in the historian, if enabled, will not occur until you restart the historian.

   The AVEVA Historian services log on to SQL Server using the Network Account (previously called the ArchestrA User), which is a Windows account. For information, see Default Windows User Account for AVEVA Historian Services.

6. Click **OK**.

Managing Logins

A login account must be added to the Microsoft SQL Server before a user can access a SQL Server. By default, only members of the sysadmin and securityadmin server roles can add SQL Server logins. Logins are managed using SQL Server Management Studio.

Note: Creating individual login accounts for each user is not required if Windows authentication mode is used in SQL Server. You can map Windows user accounts and groups to SQL Server logins using the SQL Server Management Studio. For more information, see Adding a User to a Role.

A member of the sysadmin server role can add logins and configure certain login options, such as a username (login ID), password, a default database, and a default language. If the user is not assigned a username in the default database, the user's login name is used.

In addition to the default Microsoft SQL Server logins, four more default logins are created during AVEVA Historian installation: aaAdmin, aadbo, aaPower, and aaUser.

If a large number of users will be connecting to the database with the same set of permissions, creating a single

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

database role to grant access for all of these users will reduce the work involved in account management. The individual users can then be added to the database role. For more information, see your Microsoft documentation.

Four Windows security groups are created when you install the historian:

- aaAdministrators
- aaPowerUsers
- aaReplicationUsers
- aaUsers

These groups are mapped to SQL Server database roles of the same name. You can assign different levels of capability to users by adding the users to the Windows groups.

For more information about default AVEVA Historian logins and Windows security groups, see About Security.

If you are a member of the sysadmin server role, you can add, modify, and remove logins, as well as administer database roles. For detailed information on managing logins, see your Microsoft documentation.

Viewing Login Properties

To view properties for a login:

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Security** and then click **Logins**. The default logins appear in the details pane.
3. Double-click the login for which you want to see the properties. The **SQL Server Login Properties** dialog box appears.

For information on configuring login properties, see your Microsoft documentation.

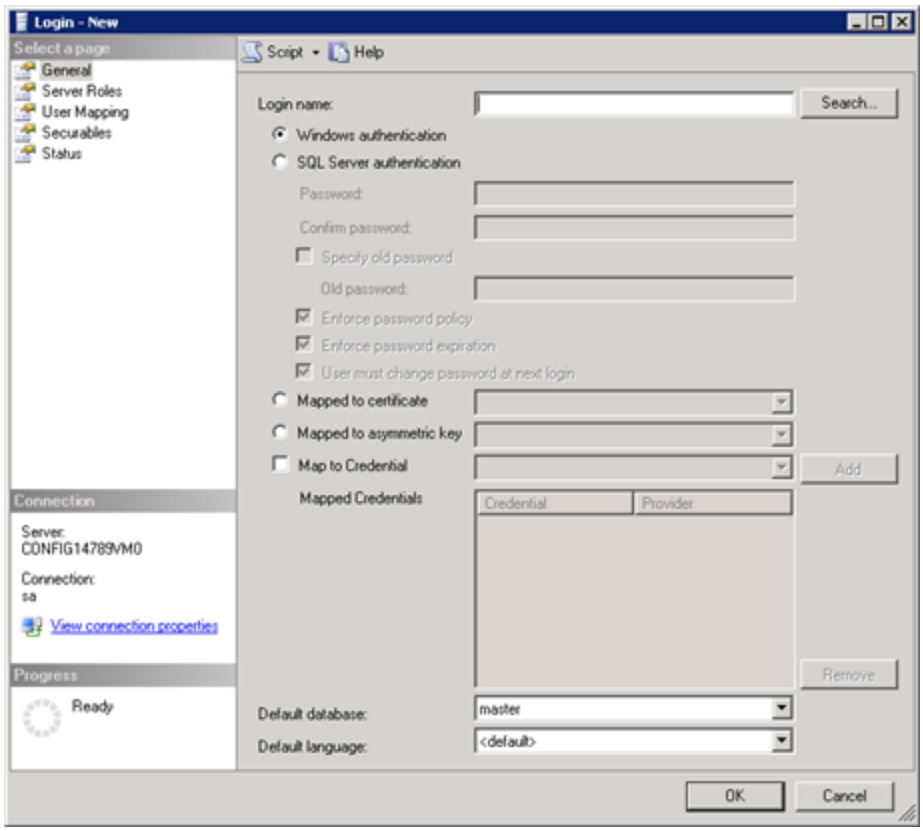
Adding a Login

You can add a login that uses either Windows authentication (recommended) or SQL Server authentication.

To add a login

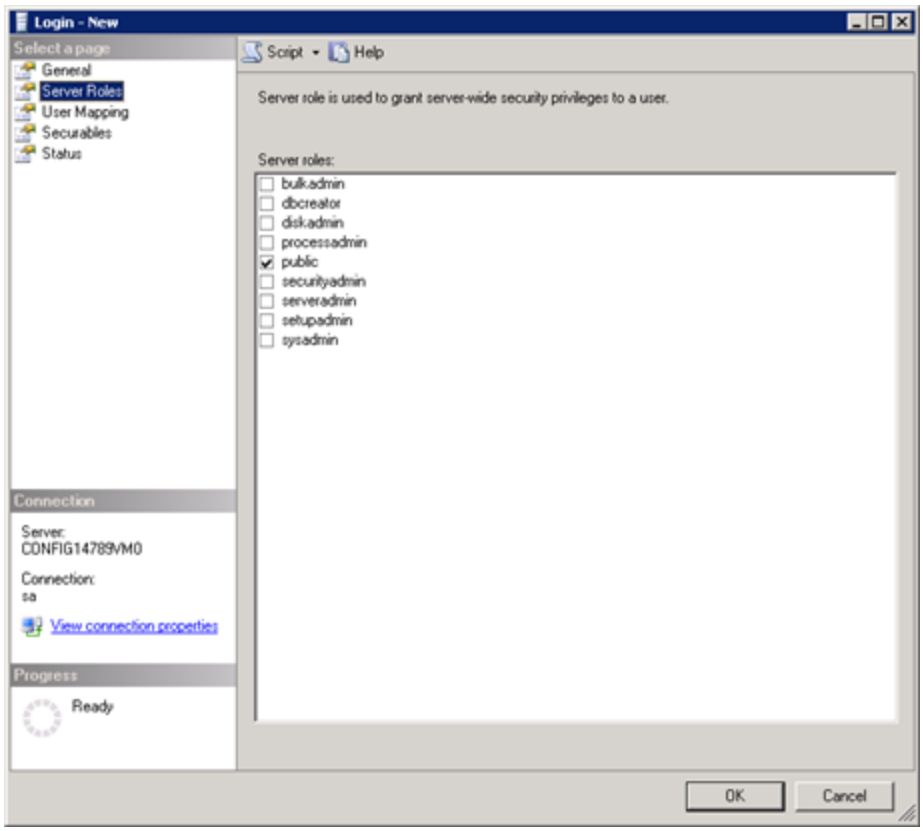
1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Security** and then right-click the **Logins** folder.
3. In the shortcut menu that appears, click **New Login**. The **SQL Server Login Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



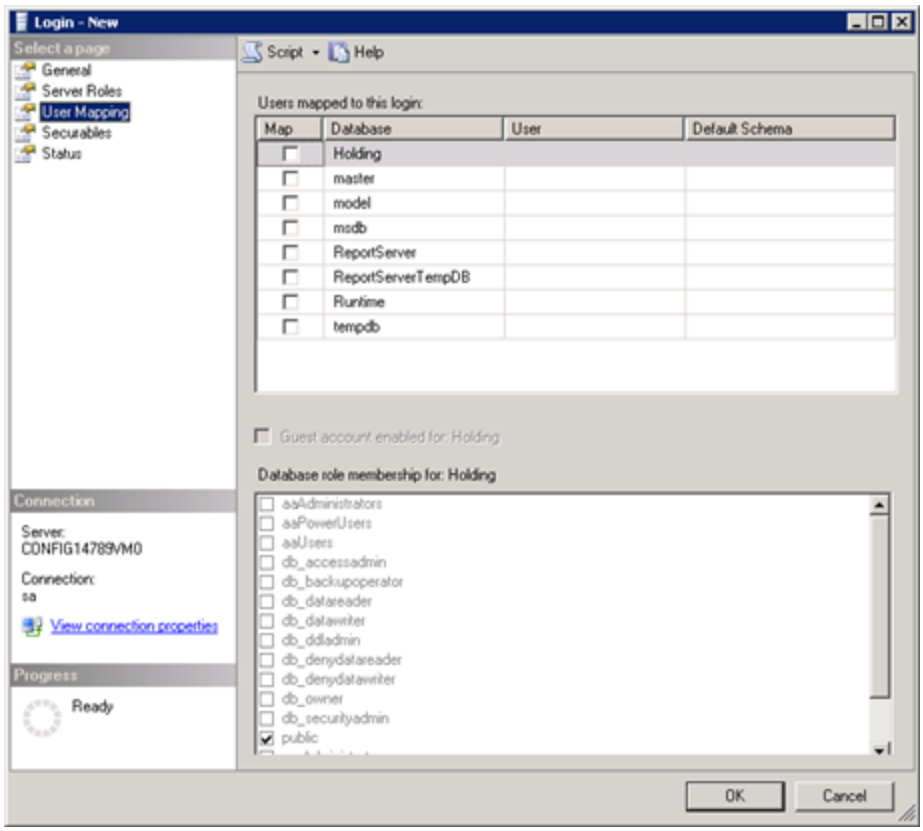
4. Do the following:
   a. In the **Name** box, type the name of the new login. If you are using Windows authentication, click **Search** and browse the network for a Windows user account.
   b. In the **Authentication** group, configure the new login to use Windows authentication or SQL Server authentication. If you use SQL Server authentication, you must enter a password for the login.
   c. In the **Database** list, select the database that the login will use by default.
   d. Select a language from the **Language** list, or leave as <Default> to use United States English.
5. Click **Server Roles**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



6. To assign the new login to an existing server role(s), select the appropriate check box in the list. This will probably not be necessary unless you are defining a power user who will require specific administrative capabilities on the server.
7. Click the **User Mapping** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



8. The **User** column contains the username to map to the login ID. The default username is the same as the login name.
9. Select the databases that can be accessed by the new login. AVEVA Historian users generally require access to the Runtime and Holding databases. They only need access to the master database if they are to be granted administrative (or higher) privileges.

   When you select a database, available database roles for that database appear in the **Database Roles** list.

   By default, all new logins are a member of the **Public** database role. You can select an additional or different role for the login for a particular database.
10. When you have configured the login, click **OK**.
11. If you created a login with a SQL Server password, you are prompted to confirm the new password. Confirm the password and then click **OK**.

Local Times and System Times

The Historian Console is date/time format sensitive, but the historian server is not. However, SQL Server handles converting string values to dates and vice versa in queries and as a result, is sensitive to the date/time format. For example, SQL Server bases its interpretation of "12/30/09" as DMY, MDY, or YMD based on the "default date order" associated with the "Default Language" for the SQL Server login used by the database connection.

Note: You cannot directly set the default date order for a login. You can only set the default language, which has an associated default date order.

If you are using SQL authentication for your logins or you are using Windows authentication and you have an explicit SQL login, changing the date format is straightforward using SQL Management Studio.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If you are using Windows authentication with a Windows group login such as BUILTIN/Administrators, it is not always apparent which group applies to a particular Windows account. If all the logins can use the same date order/language, change them all to that one.

If you need differing date formats, set the default language for new logins with the langid value in the syslanguages table (sys.syslanguages in Transact-SQL). The initial default language is based on the language version of the SQL Server installation. You can use the SET LANGUAGE or SET DATEORDER statements or sp_addlogin or sp_defaultlanguage in Transact-SQL to override the default language for a particular login for a session.

Managing Users and Roles using the Configurator

To make managing a large number of database users easier, each username can be assigned to a Microsoft SQL Server role. All members of a role inherit the permissions that are assigned to that role. For example, if the user "MaryH" is added to the "aaPowerUsers" role, that user is automatically granted the permissions for that role. If a role name is not specified, the user is added only to the public role, which includes all users. There are two types of roles: server roles and database roles.

Using the AVEVA Historian Security configurator, you can manage users and role assignments all at once from a simple interface.

You can also use SQL Server Management Studio to individually assign Windows users and user groups, as well as Microsoft SQL Server users and roles, to roles. For more information, see Managing Users and Roles using SQL Server Management Studio.

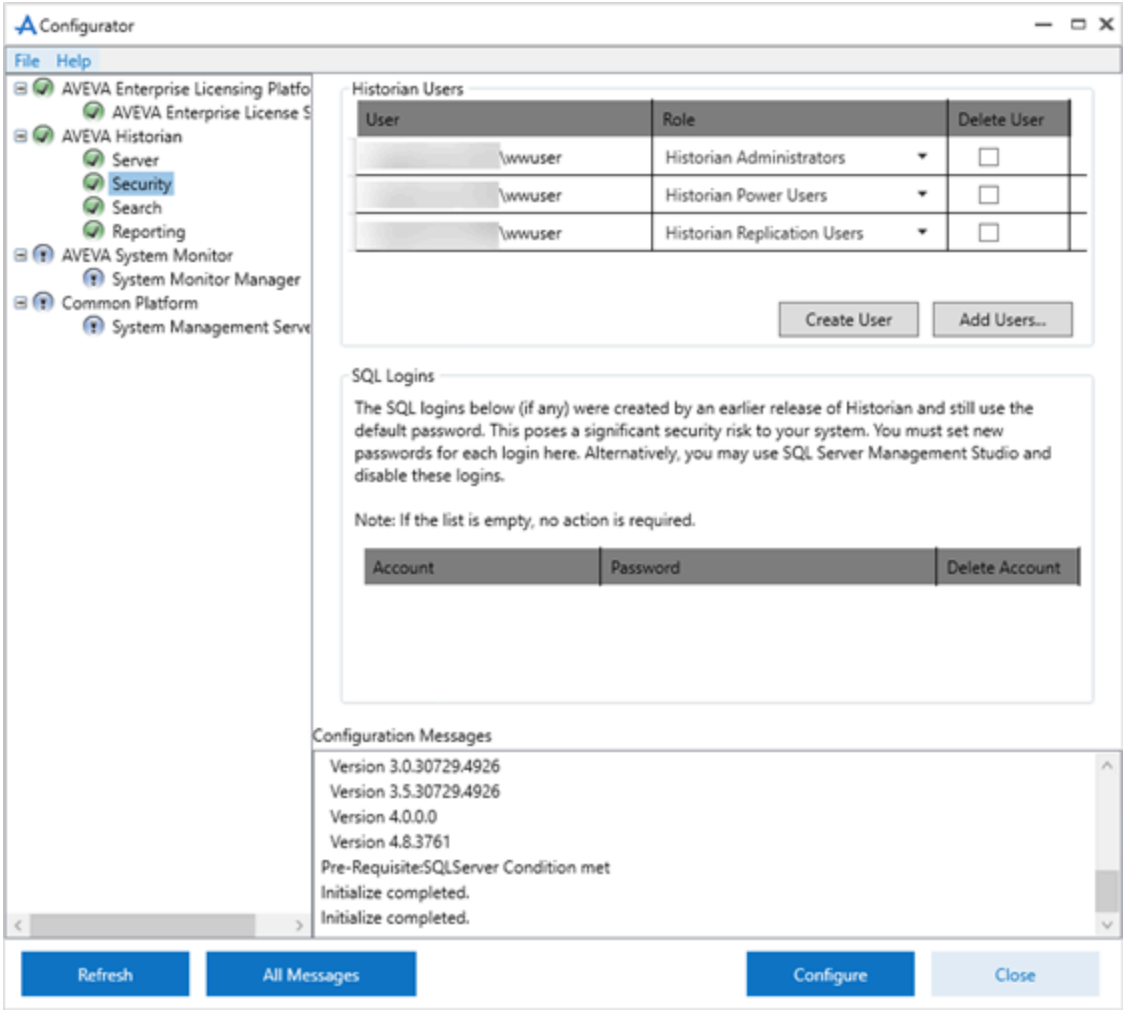
For more information about default AVEVA Historian users and roles, see About Security.

Viewing All Users and Role Assignments

To view all users and role assignments:

1. Launch the configurator, and select the AVEVA Historian Security node.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



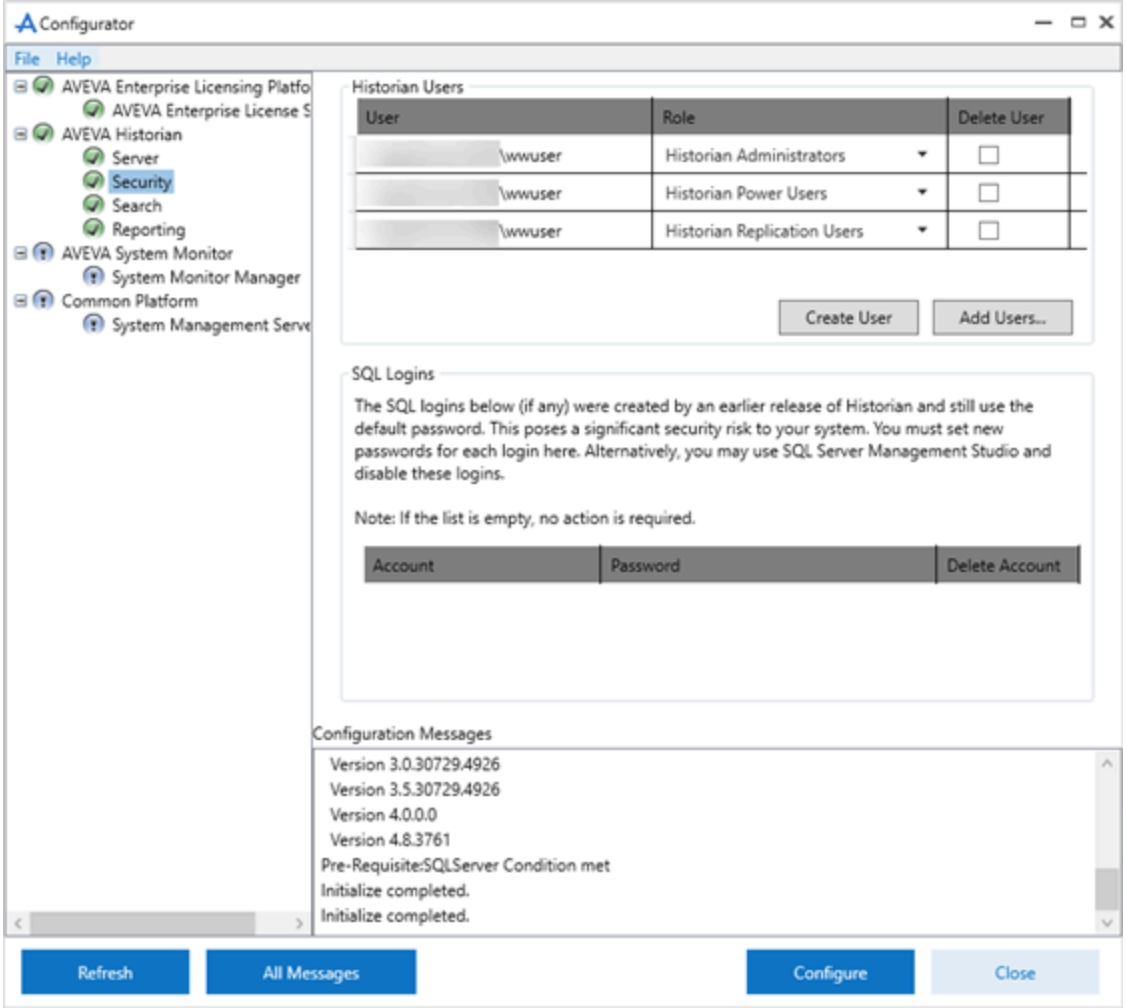
2. The **Historian Users** section lists all active user accounts assigned to Historian roles, with one entry for each role the user is assigned to.

Adding Users and Assigning Roles

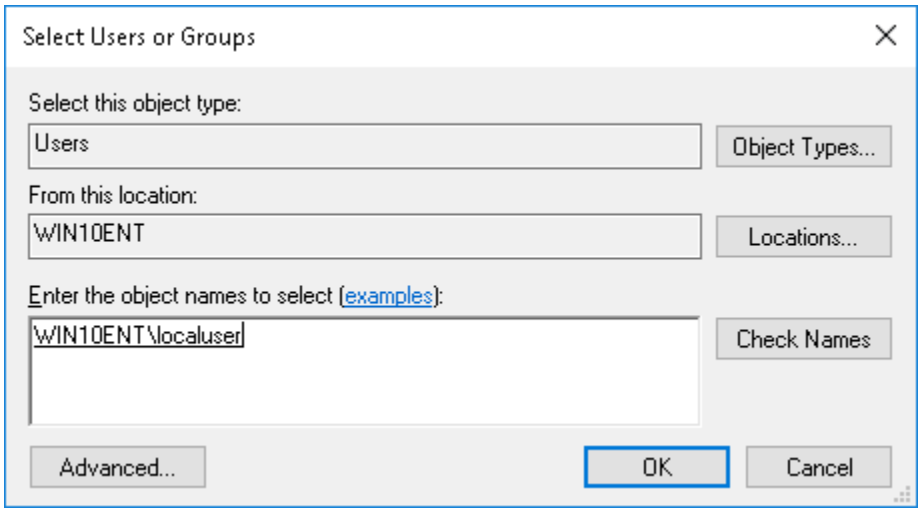
To add a user and assign a role:

1. Launch the configurator, and select the AVEVA Historian Security node.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. You can assign roles to Windows domain and local user accounts, and the configurator sets up the SQL Server user and role mappings accordingly. To assign a role to an existing domain or local user account, select **Add Users...**. The standard Windows user selection dialog displays.

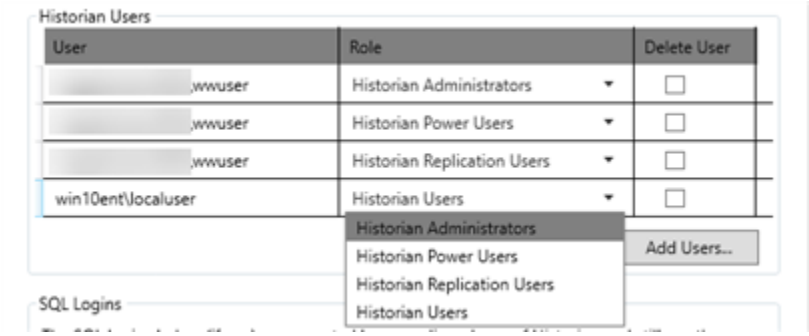


Select a user account with this dialog, and select **OK**.

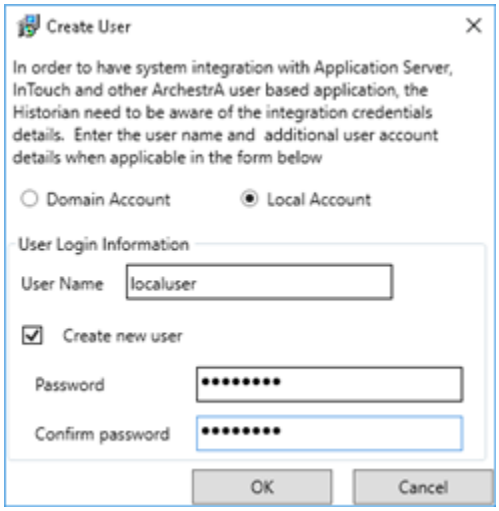
3. The user account is added to the Historian Users section, and assigned the Historian Users role by default. To

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

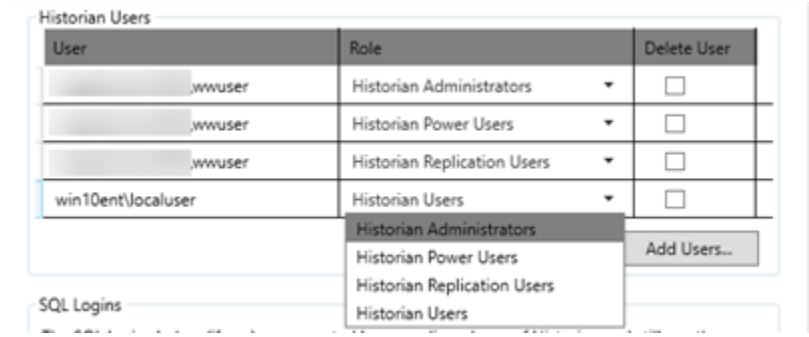
assign a different role, click ▾ beside the role name, and select another role from the list. To assign multiple roles to a user account, repeat steps 2 and 3 for each different role.



4. To create a new local Windows user account and assign it a role, select **Create User**. The create user dialog displays.



5. Select **Local Account**, enter a new **User Name**, select **Create new user**, and enter the new user's **Password**. Select **OK** when you are finished.
6. The user account is created with the password you specified. It is added to the **Historian Users** section, and assigned the **Historian Users** role by default. To assign a different role, click ▾ beside the role name, and select another role from the list.



7. Repeat the above steps until you have finished adding users and role assignments, and then select **Configure**. All of your changes are applied.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Managing Users and Roles using SQL Server Management Studio

To make managing a large number of database users easier, each username can be assigned to a Microsoft SQL Server role. All members of a role inherit the permissions that are assigned to that role. For example, if the user "MaryH" is added to the "aaPowerUsers" role, that user is automatically granted the permissions for that role. If a role name is not specified, the user is added only to the public role, which includes all users. There are two types of roles: server roles and database roles.

Using SQL Server Management Studio, you can assign Windows users and user groups, as well as Microsoft SQL Server users and roles, to roles.

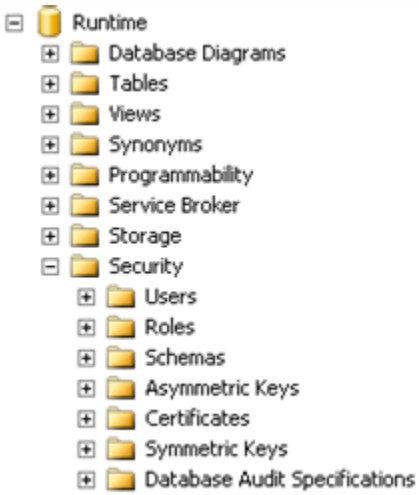
You can also use the AVEVA Historian Security configurator to manage users and role assignments all at once from a simple interface. For more information, see Managing Users and Roles using the Configurator.

For more information about default AVEVA Historian users and roles, see About Security.

Viewing All Users and Roles for a Database

To view all users and roles:

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Databases**, expand the database for which you want to view all users and roles; for example, the Runtime database, then expand **Security**.



3. To view all users, click **Users**. All users appear in the details pane.
4. To view all roles, click **Roles**. All roles appear in the details pane.

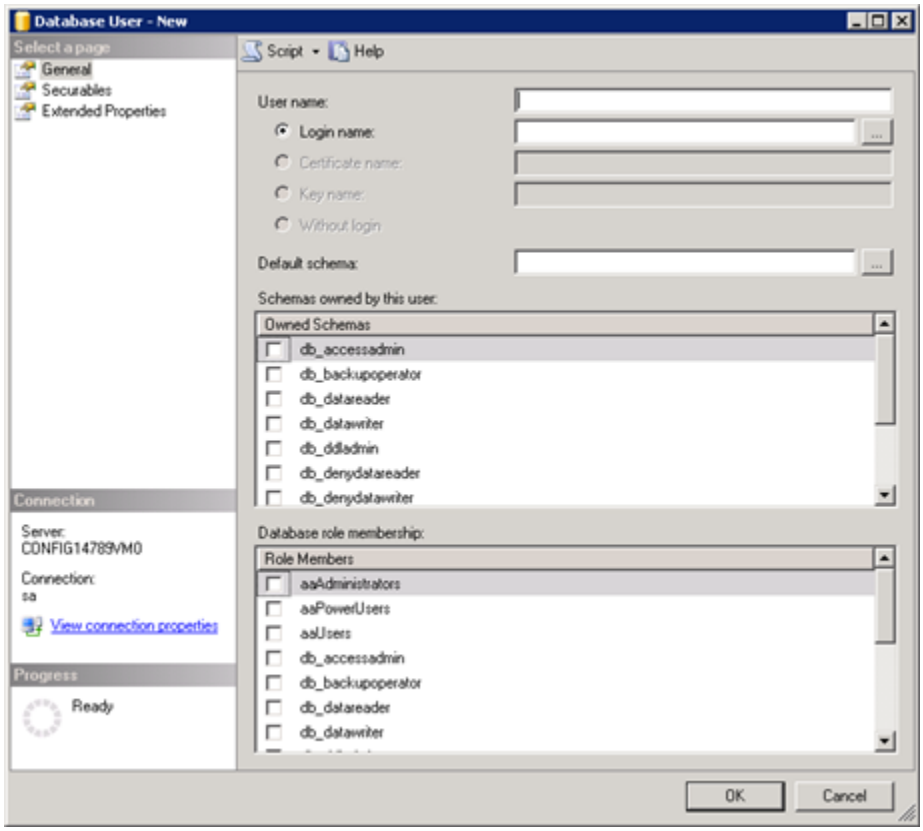
Adding a New Database User

To add a database user

1. In SQL Server Management Studio, expand the server group, and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Databases**, expand the database to which you want to add the new user, then expand **Security**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Right-click **Users** and then click **New Database User**. The **Database User Properties** dialog box appears.



4. In the **User name** box, type the new user name.
5. In the **Login name** list, select the login ID to associate with the user name. You can also select **<new>** and then type a new login ID to be added to the system at the same time as the database user.
6. In the **Default schema** list, select the default schema for the user. This schema owns all the objects this user creates, unless a different schema is specified.
7. In the **Owned Schemas** window, check the owned schemas for this user. (Schemas can only be owned by a single user.)
8. In the **Database role membership** window, select the database role to make the user a member of.

   Note: All users are also included in the public role. Membership in this role is permanent and cannot be altered.

9. Click **OK**.

Adding a User to a Role

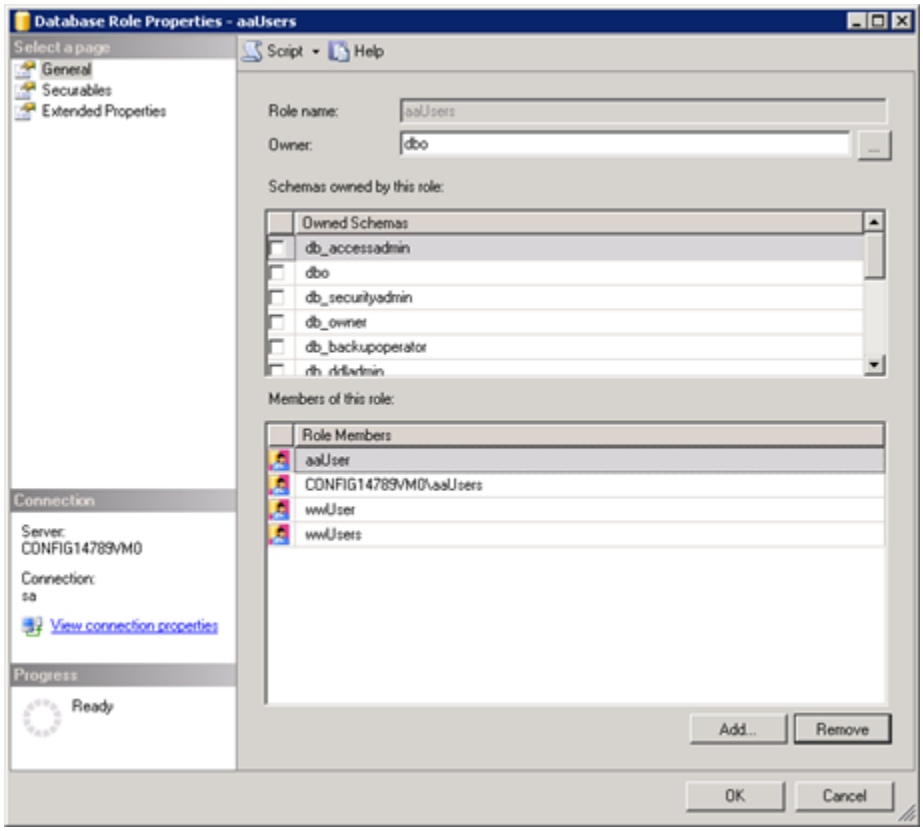
To add a user to a role, you must have system administrator permissions.

To add a user to a role:

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server instance associated with the AVEVA Historian.
2. Expand **Databases** and then expand the database to which you want to add the user to a role.
3. Click **Roles**. In the details pane, right-click the role to which you want to add a user and then click **Properties**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The **Database Role Properties** dialog box appears.



4. Click **Add**. In the dialog box that appears, select the user from the list and then click **OK**.
5. Click **OK**.

Managing Permissions

Permissions are the allowed actions that a user can perform in a designated SQL Server database. You can give object or statement permissions to any user or database role. Users inherit the permissions of any roles to which they belong.

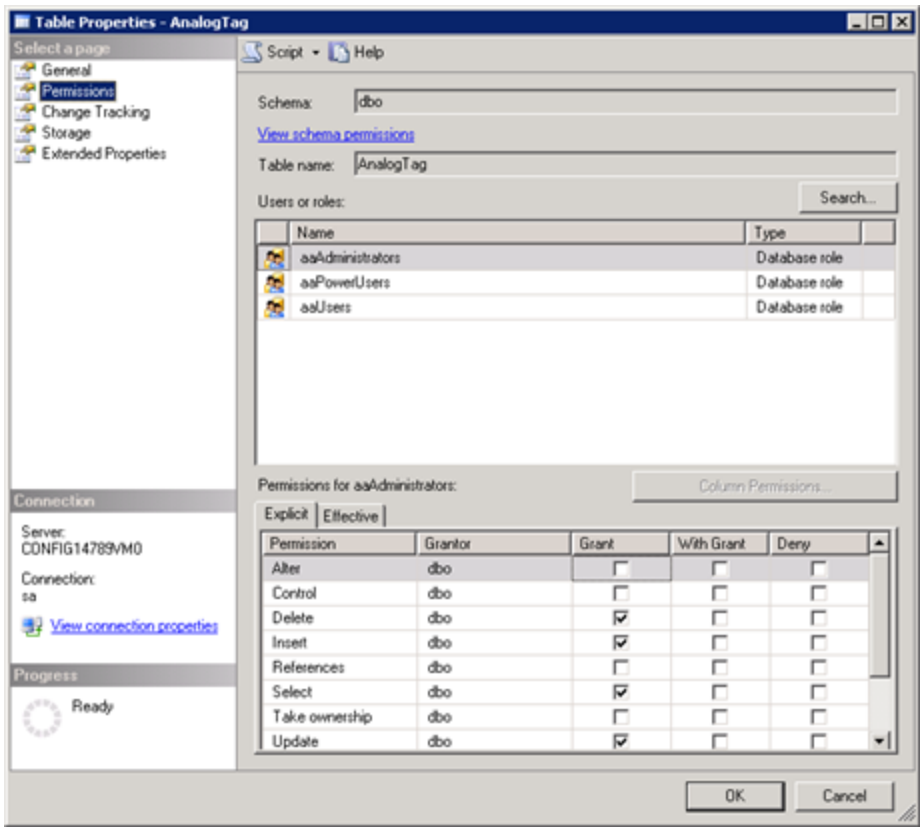
Setting Object Permissions

Object permissions control the actions that a user can perform on database objects, such as tables, indexes, views, defaults, triggers, rules, and procedures. You must be the owner (creator) of an object to grant and revoke permissions. You can grant object permissions by user and role, and by object.

To grant object permissions by object

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Databases** and then expand the database to which you want to add the object permission. For example, expand the Runtime database.
3. Right-click the object and then click **Properties**. The **Properties** dialog box appears.
4. Click **Permissions**. The **Permissions** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

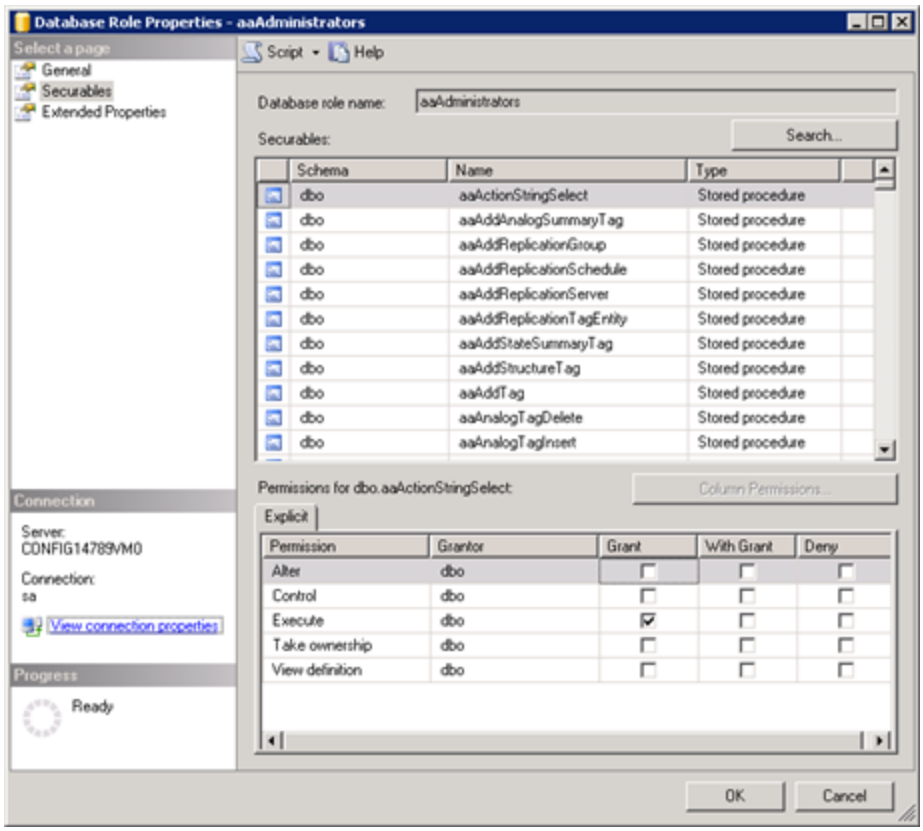


5. For each user or role, select permissions to grant for the object.
6. Click **OK**.

To grant object permissions by user or database role

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Databases** and then expand the database to which you want to add the object permission. For example, expand the Runtime database.
3. Click **Security**.
4. Click either **Users** or **Roles**.
5. If you are selecting a role, click **Database Roles** or **Application Roles**.
6. Right-click the user or role and then click **Properties**. The **Properties** dialog box for the user or role appears.
7. Click **Securables**. The **Securables** page appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



8. Select the object for which you want to grant permissions, and then select the permissions to grant listed in the **Explicit** tab.
9. Click **OK**.

Setting Statement Permissions

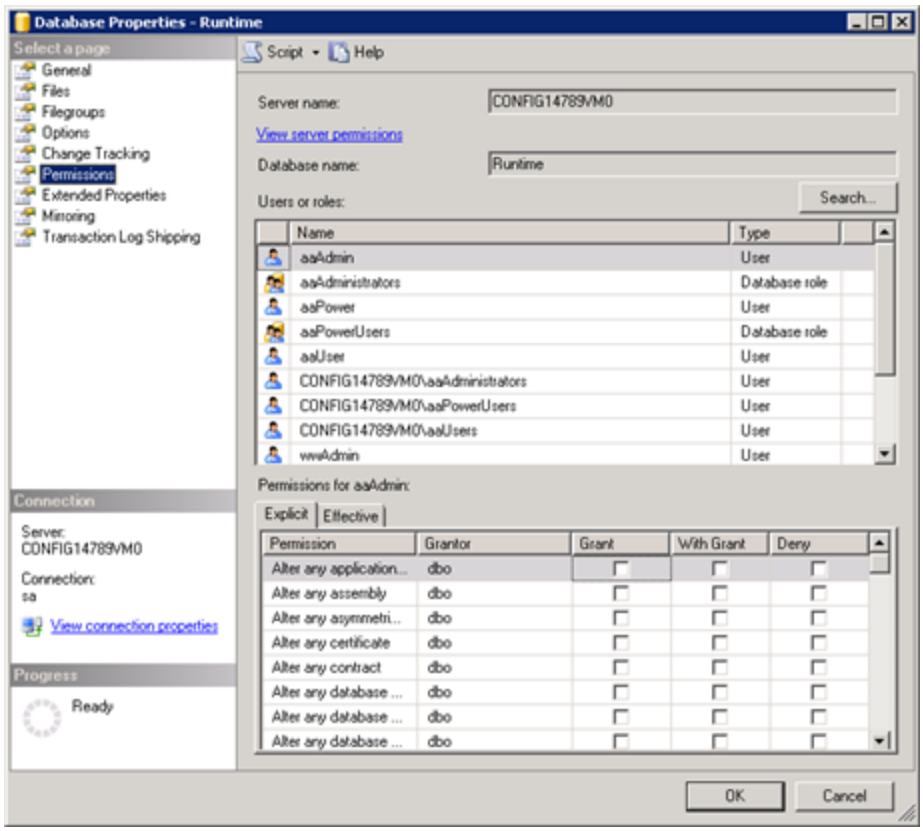
Statement permissions control who can issue particular Transact-SQL statements, such as SELECT, INSERT, or DELETE. You must be a member of the sysadmin or db_owner roles to grant and revoke statement permissions.

Note: CREATE DATABASE permissions can only be set from the master database.

To set statement permissions

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Databases** and then right-click the database for which you want to set statement permissions. For example, right-click the Runtime database.
3. Click **Properties**. The **Database Properties** dialog box appears.
4. Click **Permissions**. The **Permissions** page appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Select the user or role to which you want to grant permissions, and then select the permissions to grant.
6. Click **OK**.

Managing Passwords

Note: During installation, Historian requires that you change the passwords for any default login accounts, such as wwUser. Use of default passwords (which are often published in various documents) is highly discouraged.

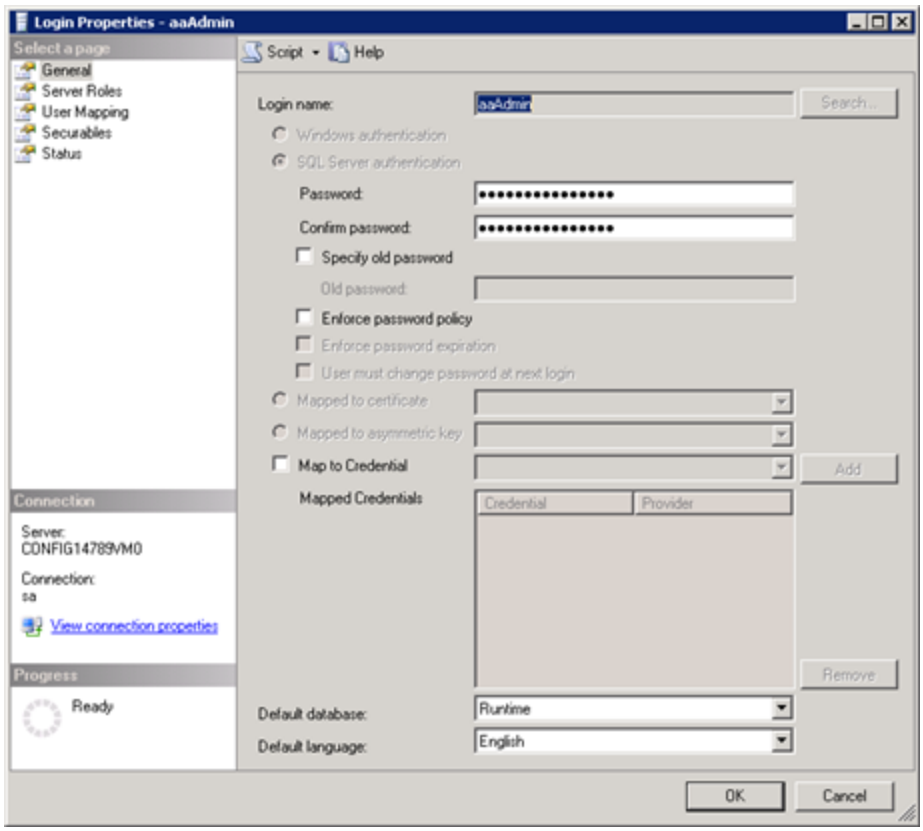
If you are a member of the sysadmin role, you can change the password for any login. If you are not a member of the sysadmin role, you can modify only your own password.

Important: If you are using mixed mode authentication, it is very important to have a password for the system administrator (sa) for the Microsoft SQL Server. If any user does not have a password, AVEVA reserves the right to refuse Technical Support services.

To change a password

1. In SQL Server Management Studio, expand the server group and then expand the SQL Server associated with the AVEVA Historian.
2. Expand **Security** and then click **Logins**.
3. In the details pane, right-click the user for which you want to change the password and then click **Properties**. The **Login Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. In the **Password** box, type the new password and then confirm it.
5. Click **OK**.

Adding a User to a Windows Operating System Group

When the AVEVA Historian is installed, default Windows security groups are created on the server computer and are automatically configured to be members of the database roles with the same names.

You must be an administrator to add a user to a group.

If Windows authentication mode is used with SQL Server, you can simply add Windows users to these groups. For example, you can add an Active Directory group to the aaUsers group and all members will be able to query the Historian *Runtime* database; you don't also need to add them to the login/role within SQL Server.

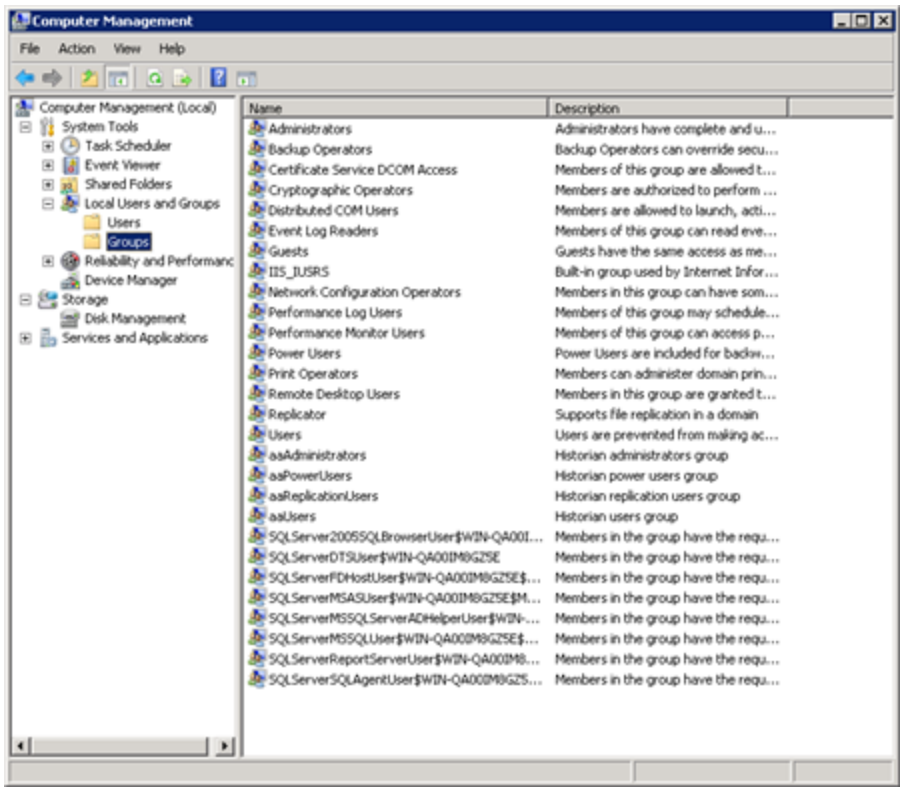
Note: You can also use the configurator to manage this. For more information, see Managing Users and Roles using the Configurator.

Access to the oData/REST interface is accomplished using Windows groups, not through SQL Server.

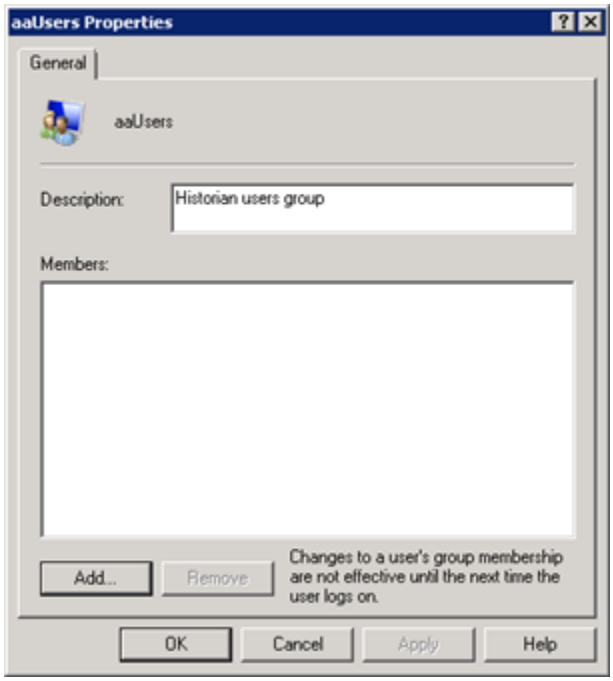
To add a user to a group

1. On the Windows **Start** menu, point to **Programs**, point to **Administrative Tools**, and then click **Computer Management**. The **Computer Management** console appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

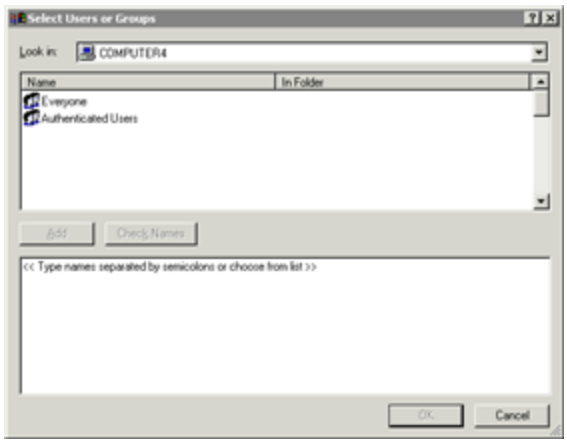


2. Expand **System Tools**, expand **Local Users and Groups**, and then click **Groups**.
3. In the details pane, right-click the name of the historian group to which you want to add a user.
4. In the shortcut menu that appears, click **Add to Group**. The **<Group Name> Properties** dialog box appears.



5. Click **Add**. The **Select Users or Groups** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



6. Select the users or other groups to add to the historian group.
7. Click **Add**.
8. Click **OK**. The **<Group Name> Properties** dialog box appears, showing the new users or groups in the **Members** window.
9. Click **OK**.

Viewing or Changing System-Wide Properties

Some administrative tasks apply to the entire AVEVA Historian system, such as configuring system parameters or committing configuration changes. You can also view a system report that includes information such as tag counts and data acquisition details.

About the Configuration Subsystem

Configuration data is information about elements that make up the AVEVA Historian, such as tag definitions, I/O Server definitions, and storage locations for historical data files. Configuration data is relatively static and does not change frequently during normal plant operation. The Configuration subsystem stores and manages configuration data.

Setting up the required databases and included entities (such as tables, stored procedures, and views) to support a typical factory environment would take countless hours. However, when you install the historian, all of these entities are defined for you, allowing you to quickly start using AVEVA Historian.

Configuration data is stored in SQL Server tables in the Runtime database. If you are already using InTouch HMI software, you can easily import much of this information from existing InTouch applications, thus preserving your engineering investment. If you are using Application Server, much of the AVEVA Historian configuration is handled automatically by Application Server. You can also use the Operations Control Management Console to manually add definitions and configure the system. You can make bulk modifications to your historian configuration or migrate the configuration from one historian to another using the AVEVA Historian Database Export/Import Utility.

You can reconfigure the system at any time with no interruption in the acquisition, storage, and retrieval of unaffected tags. Configuration data can be stored with a complete revision history.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuration Subsystem Components

The components of the Configuration subsystem are:

Component	Description
Runtime database	SQL Server database that stores all configuration information.
Configuration and management tools	Consists of the Operations Control Management Console client application, the AVEVA Historian Database Export/Import Utility, and the configuration tools shipped with Microsoft SQL Server. For more information, see About Administrative Tools.
Configuration Service (aahCfgSvc.exe)	Internal process that handles all status and configuration information throughout the system. This process runs as a Windows service.

About the Runtime and Holding Databases

A relational database management system (RDBMS) such as Microsoft SQL Server can contain many databases. A database is a collection of objects such as:

- Tables
- Stored procedures
- Views
- User-defined data types
- Users and groups

AVEVA Historian includes two preconfigured databases: the *Runtime* and *Holding* databases.

The historian embeds a full-featured Microsoft SQL Server. The historian supports all system tables associated with SQL Server. For more information on the Microsoft SQL Server tables, see your Microsoft documentation.

Note: When installed on a case-sensitive Microsoft SQL Server, the Runtime and Holding databases are case-sensitive. Be sure that you use the correct case when performing queries.

Runtime Database

The Runtime database is the online database against which the AVEVA Historian runs. The tables within the Runtime database store all configuration information, such as:

- System configuration
- Tag definitions
- InTouch integration information
- System namespaces and grouping information

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Classic Event subsystem configuration information
- User-entered annotations

Runtime database tables are usually used as references or lookup tables by the historian and client applications. Any changes to the historian are reflected in these configuration tables. The configuration tables exist as normal SQL Server tables and data within them can be modified by using the Microsoft Transact-SQL query language. For more information on Transact-SQL, see your Microsoft documentation.

The Runtime database also stores some types of history data:

- Modification tracking data
- Classic Event subsystem data

Tables that store modification tracking and event data are also normal SQL Server tables.

Finally, the Runtime database is used to **logically** store historized tag values. Although the tag values are stored in the history block files on disk, the values appear to be saved to tables in the Runtime database. For more information on history blocks, see History blocks and partitions. For more information on retrieving historized tag values, see About Data Retrieval.

Note: You cannot change the name of the Runtime database.

Holding Database

The Holding database temporarily stores topic and configuration data imported into AVEVA Historian from an InTouch node. When you import configuration data from an InTouch application, the data is first mapped to table structures in the Holding database. Then, the data is moved into the Runtime database.

Important: Do not modify any entities in the Holding database.

For more information about importing configuration information from an InTouch application, see Importing and Exporting Tag Configurations.

About the Configuration Service

The Configuration Service is an internal process that accepts configuration changes and updates the Runtime database. Thus, the Configuration Service is the only component that interacts with the configuration store.

The Configuration Service runs as a Windows service and accepts and distributes configuration information to and from different parts of the system by a set of interfaces. The Configuration Service also serves as a gateway for all information pertaining to the status of the different components of AVEVA Historian.

Dynamic Configuration

AVEVA Historian supports dynamic configuration. That means you can reconfiguration of tags and other objects in the historian database while the system is running. The historian automatically detects and applies the modifications to its internal run-time state without requiring the system to be restarted. In addition, clients do not experience interruptions due to configuration changes.

The dynamic configuration feature in the historian caters for all possible database modifications that affect the run-time operation of the system. The Configuration subsystem is designed to ensure that no loss of data occurs for tags that are not affected by the modifications being applied. However, tags that require a change in data

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

acquisition configuration will obviously lose data during the reconfiguration.

In most cases, the system continues to run uninterrupted. In the following cases, a restart of the system is required:

- When you change the main historization path in the system, a parameter that is rarely modified after installation.
- When you modify the DataImportPath system parameter.

For a description of the effect of various types of modifications made while the system is running, see Effects of Configuration Changes on the System.

Dynamic configuration is usually a two-step process:

1. Add, modify, or delete one or more objects in the database, using the Operations Control Management Console, Transact-SQL statements, or the database modification tool of your choice.

   As soon as you make a change, the Runtime database is updated to reflect the change. For example, when you add an analog tag using the wizard within the Configuration Editor, the database is updated as soon as you click **Finish**.
2. After making all of the modifications, you must commit the changes, which triggers the dynamic configuration process in the server. Modifications made to the system are done in a transactional fashion.

   The database modifications are not reflected in the running system until you commit the changes. You are committing changes to the system, not to the database.

You can commit changes to the configuration of the system as often as you want. You can also commit changes in batches or individually. There is no limit on the number of changes that may be committed to the system. Configuration changes typically take effect within 10 seconds under maximum data throughput conditions.

For information on cases in which a commit is prevented, see Cases in Which Configuration Changes Are Not Committed.

Effects of Configuration Changes on the System

Different types of dynamic changes to the database affect the system in different ways.

A summary of typical changes and their effect on the system after a commit is as follows.

- **Modifying system parameters**
  A modification to system parameters usually takes effect immediately. If you change the HistoryCacheSize parameter and commit the change, the cache is not immediately flushed to bring the cache size to less than or equal to the new value.
- **Modifying storage locations**
  Modifying the circular storage location requires a shutdown and restart of AVEVA Historian. Changes to the other storage locations take effect immediately.
- **Adding, deleting, and modifying tags**
  Modifying data acquisition characteristics of a tag could result in a brief period of data loss, for that tag. As a guideline, any change to the source of data for the tag (for example, modifying the item name, topic name, or I/O server name of the tag) will result in a short gap in data for the tag, while the system disconnects from the old data source and connects to the new data source.
- **Adding, deleting, and modifying IDASs**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Adding a new IDAS to the system results in a new set of system tags being added (the status and performance system tags associated with that IDAS).

Modifying an IDAS may result in data loss for the tags serviced by that IDAS (for example, moving an IDAS to another computer causes a disconnect from the data sources).

- **Adding, deleting, and modifying I/O Servers and topics**
  Modifying I/O Server or topic characteristics may result in data loss for their tags, if the modification implies a disconnect from the data source.

Cases in Which Configuration Changes Are Not Committed

If the system is not running, or storage is stopped, any commit is ignored and the contents of the ConfigStatusPending table are cleaned up. The exceptions are changes to the following fields in the SystemParameter table:

- HistoryCacheSize
- HistoryDaysAlwaysCached
- AutoStart

If the system is running, a commit is not allowed if a previous dynamic configuration is still in progress. A message appears, indicating that the commit is not allowed.

Viewing Properties for System Parameters

To open the **Properties** dialog box for an item in the details pane, do any of the following:

- Double-click the item.
- Right-click the item and click **Properties**.
- Select the item and click the **Properties** button .
- Select the item and click **Properties** on the **Action** menu.

Editing System Parameters

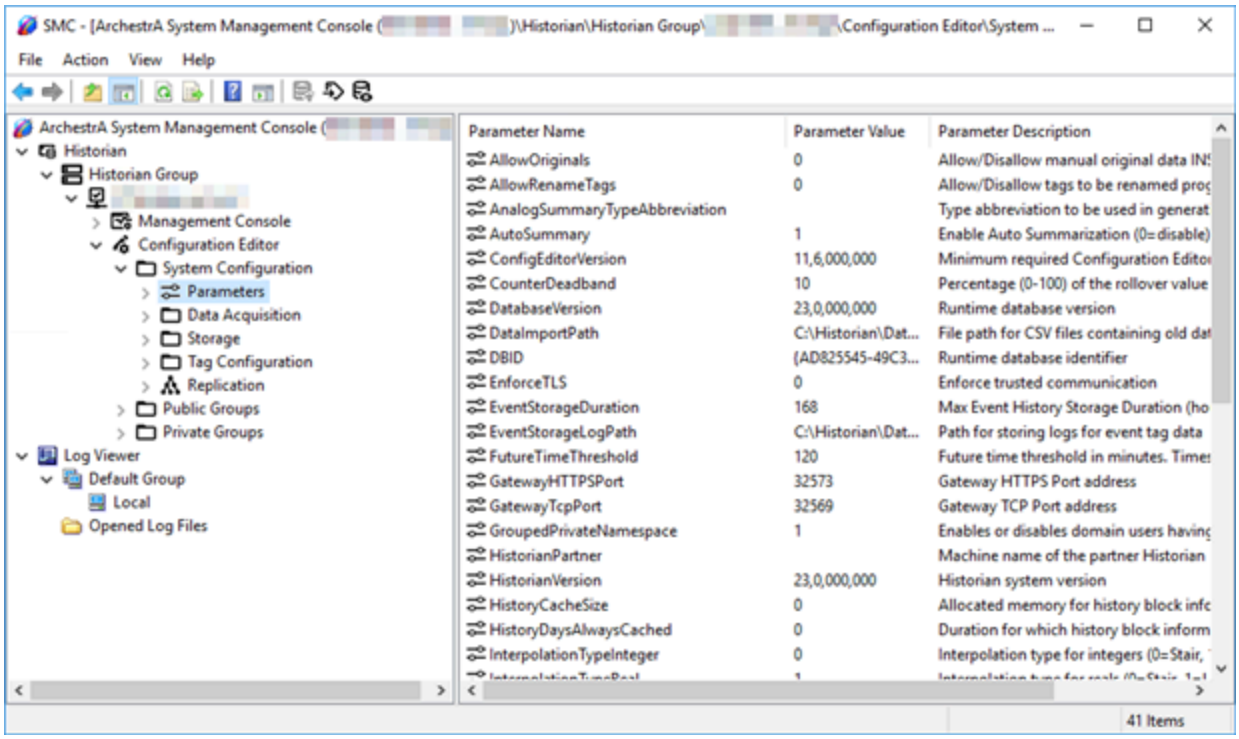
Note: Not all system parameters are editable.

For a list of system parameters, see System Parameters.

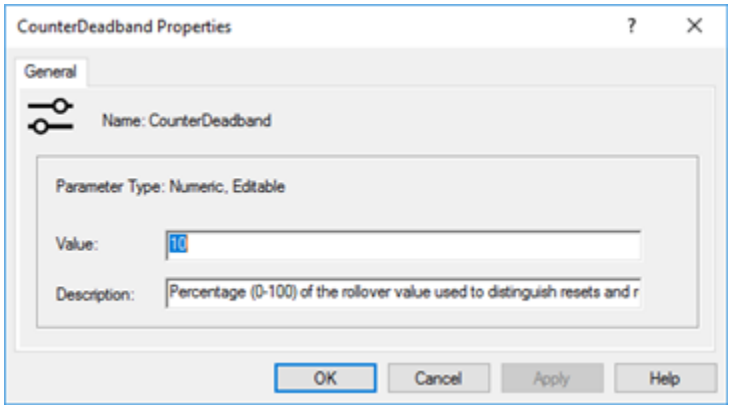
To edit a system parameter:

1. In the Operations Control Management Console, expand a server group, expand the server, and then expand Configuration Editor.
2. Expand **System Configuration** and then click **Parameters**. A list of all of the system parameters appears in the details pane.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Double-click the system parameter you want to edit. The **Properties** dialog box for that parameter displays.



4. In the **Value** box, type a new value of the system parameter.
5. (Optional) In the **Description** box, type a new description of the system parameter.
6. Click **OK**.

Adding a System Parameter

You can create your own named system parameters for the AVEVA Historian by adding rows to the SystemParameter table using a SQL script.

Committing Configuration Changes

After you make a change to the database (for example, add a tag), you must commit the change to the AVEVA Historian system. All database changes are immediately implemented. However, database modifications are not

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

applied to the system until you commit them. You are committing the changes to the system, not the database. The system reconfigures itself with no interruption for unaffected objects in the database.

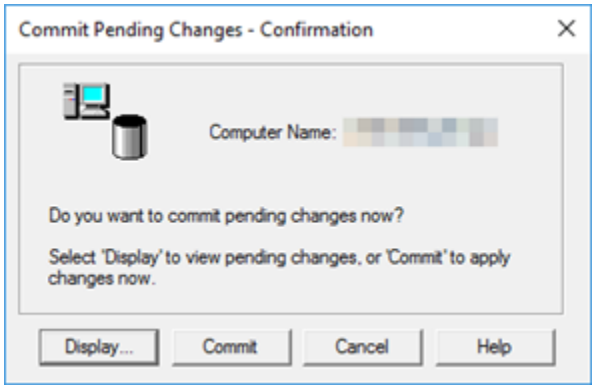
Changes cannot be committed:

- During the first five minutes after starting the historian.
- During the creation of a new data block because of a prior change.

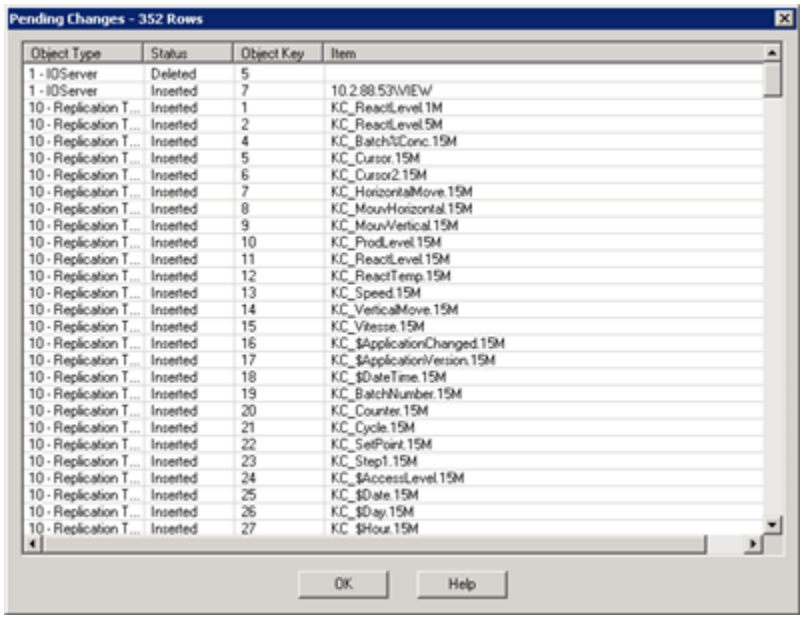
For more information, see Dynamic Configuration in the *AVEVA Historian Concepts Guide*.

To commit configuration changes to the system:

1. In the Operations Control Management Console, expand a server group and then expand a server.
2. Right-click **Configuration Editor** (or any sub-items in the console tree) and then click **Commit Pending Changes**. The **Commit Pending Changes - Confirmation** dialog box appears.



3. To view a list of the pending changes, click **Display**.



Object Type	Status	Object Key	Item
1 - IOServer	Deleted	5	
1 - IOServer	Inserted	7	10.2.88.53\VIEW
10 - Replication T...	Inserted	1	KC_ReactLevel.1M
10 - Replication T...	Inserted	2	KC_ReactLevel.5M
10 - Replication T...	Inserted	4	KC_Batch%Conc.15M
10 - Replication T...	Inserted	5	KC_Cursor.15M
10 - Replication T...	Inserted	6	KC_Cursor2.15M
10 - Replication T...	Inserted	7	KC_HorizontalMove.15M
10 - Replication T...	Inserted	8	KC_MouvHorizontal.15M
10 - Replication T...	Inserted	9	KC_MouvVertical.15M
10 - Replication T...	Inserted	10	KC_ProdLevel.15M
10 - Replication T...	Inserted	11	KC_ReactLevel.15M
10 - Replication T...	Inserted	12	KC_ReactTemp.15M
10 - Replication T...	Inserted	13	KC_Speed.15M
10 - Replication T...	Inserted	14	KC_VerticalMove.15M
10 - Replication T...	Inserted	15	KC_Vitesse.15M
10 - Replication T...	Inserted	16	KC_$ApplicationChanged.15M
10 - Replication T...	Inserted	17	KC_$ApplicationVersion.15M
10 - Replication T...	Inserted	18	KC_$DateTime.15M
10 - Replication T...	Inserted	19	KC_BatchNumber.15M
10 - Replication T...	Inserted	20	KC_Counter.15M
10 - Replication T...	Inserted	21	KC_Cycle.15M
10 - Replication T...	Inserted	22	KC_SetPoint.15M
10 - Replication T...	Inserted	23	KC_Step1.15M
10 - Replication T...	Inserted	24	KC_$AccessLevel.15M
10 - Replication T...	Inserted	25	KC_$Date.15M
10 - Replication T...	Inserted	26	KC_$Day.15M
10 - Replication T...	Inserted	27	KC_$Hour.15M

Column descriptions are as follows:

Object Type

Used to indicate the type of object to which the modifications apply.

Status

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Used to indicate the type of modification.

Object Key

The unique identifier of the modified object. If the modified object is a system parameter, the value will be 0. For all other object types, the value is from one of the following tables and columns: IODriver.IODriverKey; IOServer.IOServerKey; Topic.TopicKey; Tag.wwTagKey; StorageLocation.StorageType; SnapshotDetail.StorageSize.

Item

The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.

4. To commit the outstanding changes, click **Commit**.
5. An information box appears, showing the status of the reconfiguration.
6. Click **OK**.

Tracking Modifications

AVEVA Historian tracks modifications (inserts, updates, and deletions) to columns in the Runtime database. If your plant tracks changes for compliance with regulatory agencies, you can configure the historian to use modification tracking.

Modification tracking is system-wide; it is controlled by the ModLogTrackingStatus system parameter. You cannot turn modification tracking on or off at a table level. Enabling modification tracking decreases the historian's performance when making changes to the system. This is due to the extra work and space required to track the changes. However, there is no performance degradation during run-time operation.

Information in the modification tracking tables are stored in the data files of the Microsoft SQL Server database. If modification tracking is turned on, the amount of data that is stored in these files is greatly increased.

All of the objects for which modifications can be tracked are stored in the HistorianSysObjects table.

You can track changes to configuration data. For example, additions or changes to tag, I/O Server, and storage location definitions. For more information, see About Modification Tracking for Configuration Changes.

Note: Changes to history data are tracked through an internal stored procedure called aaInternalHistoryModTrack. It captures the data modification with History view.

The types of changes that will be tracked is controlled by the ModLogTrackingStatus system parameter. You can track inserts, updates, and deletes, as well as various combinations. For more information, see Turning Modification Tracking On/Off.

About Modification Tracking for Configuration Changes

For configuration data, when a modification is made to a table in the database, a record for the modification is inserted into the ModLogTable table. One row will be inserted for each separate type of modification, either an insert, update, or delete.

The actual value changes are recorded in the ModLogColumn table. Each column that is modified will result in a row inserted into the ModLogColumn table. The entry in the ModLogColumn table includes both the column value before the change and the new column value.

For example, if you added (inserted) a single analog tag to the system, the following changes would be reflected in the modification tracking tables:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Two rows would be added to the ModLogTable table, one to track the change to the Tag table and one to track the change to the AnalogTag table.
- One row for each of the columns in both of the Tag and AnalogTag tables will be added to the ModLogColumn table.

As another example, if you updated for a single analog tag the StorageType column in the Tag table and the ValueDeadband and RateDeadband columns in the AnalogTag table, the following changes would be reflected in the modification tracking tables:

- Two rows would be added to the ModLogTable table, one to track the change to the Tag table and one to track the change to the AnalogTag table.
- Three rows would be added to the ModLogColumn table to record the changes to the StorageType, ValueDeadband, and RateDeadband columns.
- Important things to note:

For a tier-2 historian, modification tracking for a replicated tag appears as being made by the system account that the configuration service is running under, which is typically NT AUTHORITY\SYSTEM. To find out who modified a tag, examine the ModLogTable of the tier-1 historian.

If modification tracking is enabled and tags are configured in offline mode and modified, only the last known modifications are recorded in the modification tables.

About Modification Tracking for Historical Data Changes

Modifications to history data can be performed by either executing Transact-SQL statements or by using the CSV import functionality. In the case of Transact-SQL statements, the AVEVA Historian OLE DB provider provides the change information to the modification tracking tables by means of a stored procedure. This stored procedure is also used by the Storage subsystem to communicate changes that are the result of a CSV import.

Although the history data that is changed is physically stored on disk in the history blocks, for the purposes of modification tracking, the data is considered to reside in the History_OLEDB extension table. For more information on extension tables, see Extension Tables for History Data.

When a modification is made to history data, a record for the modification is inserted into the ModLogTable table. One row will be inserted for each separate type of modification, either an insert or an update, for each tag.

The ModLogColumn table is used to store details for the column modification in the **History_OLEDB table**. The modified column will always be the vValue column. The total count of consecutive value changes attempted per tag is stored in the NewValue column of the ModLogColumn table.

The OldValue column contains the value stored in the column before the modification was made, if the modification was to a configuration table. For modifications to history data using SQL INSERT and UPDATE statements, this column contains the timestamp of the earliest data affected by the INSERT or UPDATE operation. If multiple changes are made to the same data, then only the most recent change will be contained in this column. This column is not used for modifications made to history data using a CSV file.

For example, if you insert 20 data values into history for the ReactTemp analog tag using a CSV import, the following changes would be reflected in the modification tracking tables:

- One row would be added to the ModLogTable table, to track the change to the **History_OLEDB table.** The UserName column will contain the name of the user as contained in the CSV file header.
- One row would be added to the ModLogColumn table to record that the value change occurred. A value of

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

20 will be stored in the NewValue column to indicate that 20 values were inserted.

Turning Modification Tracking On/Off

Use the ModLogTrackingStatus system parameter to configure modification tracking. The following table describes the allowable values:

Value	Type of Modification(s) Tracked
1	inserts
2	updates
3	inserts + updates
4	deletions
5	inserts + deletions
6	updates + deletions
7	inserts + updates + deletions

For information on editing system parameters, see Editing System Parameters.

For more information on modification tracking, see Tracking Modifications.

Viewing Database Modifications

For more information on modification tracking, see Tracking Modifications.

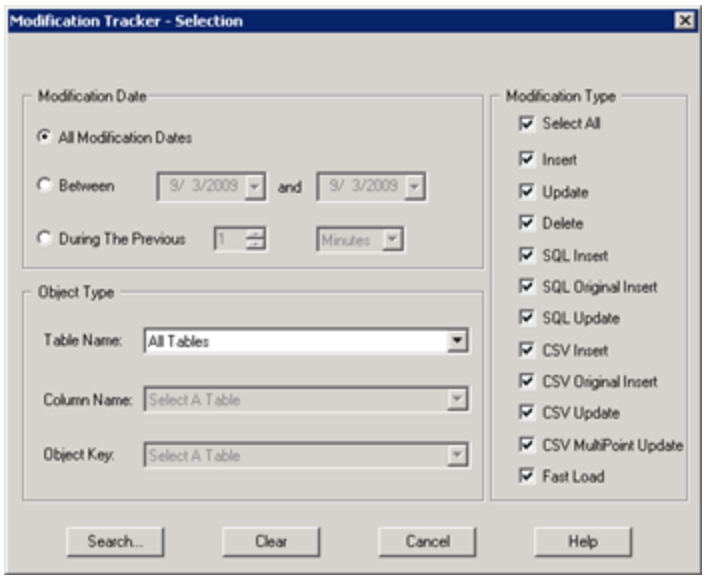
You can search for all database modifications or apply filtering to return modifications for only those tables and columns you specify.

Note: To view database modifications, you must enable modification tracking. For more information, see Turning Modification Tracking On/Off.

To view a current list of modifications:

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Right-click **Configuration Editor** (or any sub-items in the console tree) and then click **Track Modifications**. The **Modification Tracker - Selection** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. In the **Modification Date** area, configure the time span for the search.

   All Modification Dates

   Returns all changes to table columns made since modification tracking was first enabled.

   Between

   Returns all modifications between the start date and end date that you specify. Click the date arrow to access a calendar in which you can pick the date.

   During the Previous

   Returns all modifications for a recent time period. Durations can be in minutes, hours, days, weeks, or months.

4. In the **Modification Type** area, select the types of modifications to search for.
5. In the **Object Type** area, set the type of modifications to search for.

   Table Name

   Returns modifications for all tables in the database or for a specified table. Only tables that currently have modifications appear in the list.

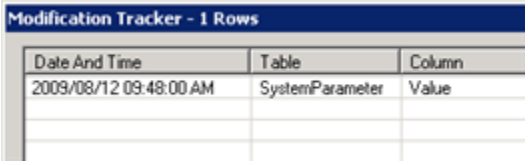
   Column Name

   Returns modifications for a specified column in the selected table. This option is only available if you select to filter on a single table.

   Object Key

   The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.

6. To reset the dialog box options back to the defaults, click **Clear**.
7. Click **Search** to search for database modifications according to the filter options you select. A list of all matching modifications appears.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column descriptions are as follows:

Date and Time

The timestamp of when the modification occurred.

Table

The name of the modified object.

Column

The name of the modified column.

Modification Type

The type of modification.

Row Key

The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.

New Value

The new value stored in the column, if the modification was to a configuration table. For modifications to history data, this column contains the total count of consecutive value updates attempted.

Old Value

The value stored in the column before the modification was made, if the modification was to a configuration table. For modifications to history data using SQL INSERT and UPDATE statements, this column contains the timestamp of the earliest data affected by the INSERT or UPDATE operation. If multiple changes are made to the same data, then only the most recent change will be contained in this column. This column is not used for modifications made to history data using a CSV file.

User

The name of the database user that made the modification. The value of this column reflects the Windows authentication user name (for example, DOMAIN\user_login_name) or the SQL Server authentication user name (for example, dbo), depending on how the user is logged into the SQL Server when the modification is made. In the case of a CSV file import, this column contains the user name as it appears in the CSV file.

8. To sort on a column, click the column name at the top of the window.
9. Click **Cancel** to close the dialog box.

Viewing the Runtime Database Report

The Runtime database report includes information such as:

- System parameters
- Total number of licensed tags
- Number of cyclically-stored tags for each storage rate
- Number of analog tags for each storage type (delta, cyclic, or "forced")
- Acquisition subsystem details
- Event tag definitions
- Summary information

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To view the database report:

1. In the Operations Control Management Console, expand a server group, and then expand a server.
2. Click **Configuration Editor**. The report appears in the details pane.

3. Click a major heading in the report to view a list of objects for that category.
4. To select, copy, or print the information, right-click in the window and then click the appropriate command from the shortcut menu.

Using a Redundant Historian

You can configure the AVEVA Historian to have a "partner" AVEVA Historian that can be used as a hot backup if the primary historian is not available. This is called a "redundant historian" setup.

If AVEVA Application Server is configured to send data to the AVEVA Historian, the AppEngine automatically sends data to both the primary historian and the specified partner. If one of the historians goes offline, the AppEngine stores the historized data until the historian comes back online. After the connection is restored, the AppEngine forwards the data to the historian that was offline.

The Historian Client automatically detects and selects the online Historian from the redundant pair.

The historians in a redundant setup are not intended to be a synchronized pair, where both the historian configuration and data are fully and automatically synchronized. It is up to you to make sure that the two historians are symmetrical and synchronized. The following recommendations are examples of actions you should take to keep the pair synchronized, or else incoming data is not stored as previously described.

- If you make configuration changes for one historian, be sure to perform the same actions on the partner.
- If you import a CSV file on one historian, you will need to repeat the import on the partner.
- If you add or update data to one Historian using SQL or the Historian SDK, you will need to repeat the action on the partner.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To specify a historian partner:

1. In the Operations Control Management Console, expand **Historian Group**, expand the server, and then expand **Configuration Editor**.
2. Expand **System Configuration** and then click **Parameters**. A list of all system parameters displays in the details pane.
3. Double-click the **Historian Partner** system parameter. The **System Parameter Properties** dialog box appears.
4. In the **Value** box, type the computer name of the partner historian. You can use either the host name, fully qualified name, or an IP address. Leading backslashes are optional.

   Note: In network environments where AppEngine and Historian Client computers on different subnets must access the partner, be careful to use a name or IP address that can be correctly resolved from all of those network locations and not just between the historian servers themselves.
5. Click **OK**.

Changing the Default Network Protocol

AVEVA Historian client/server connections are set in the same way as Microsoft SQL Server connections. No additional configuration is required to run client applications against the AVEVA Historian if you are using the default named pipes protocol.

However, you can change the client configuration parameters by using the SQL Server Client Network Utility. The historian supports clients using Net-Libraries for named pipes, IPX/SPX, TCP/IP sockets, and any other protocol supported by Microsoft SQL Server.

For more information changing the network protocol used by clients, see the documentation for the Microsoft SQL Server Client Network Utility.

Configuring a Custom TCP Port

AVEVA Historian uses default TCP port 32565 for communication.

- The "receiving" Historian (for example, the historian server for data collection or the tier-2 server for replication) needs to allow inbound connections on the configured port.
- No inbound connections need to be allowed into the "sending" application for this communications channel, but it may be needed by other components (for example, if the "sending" node is also receiving SuiteLink data, SuiteLink requires port 5413 to be opened).
- These requirements apply to the Windows Firewall and any hardware firewall sitting between the systems.

You can configure a custom port for data historization from AVEVA Application Server and for replication.

WARNING! If the historian is running and receiving data, clients using the port, such as AppEngines or replication, will be disconnected and go into store-and-forward mode. Make sure all clients have store-and-forward configured or data loss will occur.

To configure a custom port for Application Server:

1. Change the ReplicationTcpPort system parameter on the historian. Do the following:
   a. Start the Operations Control Management Console (OCMC).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

   b. Make sure that the historian is not shut down and disabled. If it is, right-click **Management Console**, point to **All Tasks**, and then click **Enable (allow to run) Historian**.
   c. Change the ReplicationTcpPort system parameter default value to the custom port number. For more information on editing system parameters, see Editing System Parameters.

      A message appears stating that you must manually update the firewall settings.
   d. Commit the configuration change. For more information on committing changes, see Committing Configuration Changes.

      The configuration service detects the change and restarts the Historian Client Access Point (HCAP) to allow it to use the custom port.
   e. To confirm the change, open the ArchestrA Logger to see a message from aahClientAccessPoint such as "Client access point (Opening Historian listening port nnnnnn)..." where nnnnnn is the custom port number you just configured.
2. Update the firewall settings on the Historian computer and update router/switch settings to allow communication through the custom port.
3. Change the TCP port on the Application Server. Do the following:
   a. Undeploy the Galaxy.
   b. Open the object editor for either the WinPlatform or the AppEngine.
   c. In the **History** area of the configuration options, expand **Advanced settings** and change the **TCP port** option value to the custom port number.
   d. Save the configuration.
   e. Repeat steps b through d for all AppEngines.
   f. Redeploy the Galaxy.

   Application Server now sends data to the historian through the custom port.

To configure a custom port for replication:

1. Change the ReplicationTcpPort system parameter to the custom port value on the tier-2 historian.
2. Update the firewall settings on the tier-2 historian computer and update router/switch settings to allow communication through the custom port.
3. Change the replication server TCP port setting to the custom port value on the tier-1 historian. For more information on editing replication server properties, see Editing Replication Server Properties.
4. Commit the configuration change. For more information on committing changes, see Committing Configuration Changes.

The tier-1 historian now replicates data to the tier-2 historian through the custom port.

Historian Client Web Customization

The following section outlines some additional tasks you can perform to customize your AVEVA Historian Client Web installation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

White Labeling

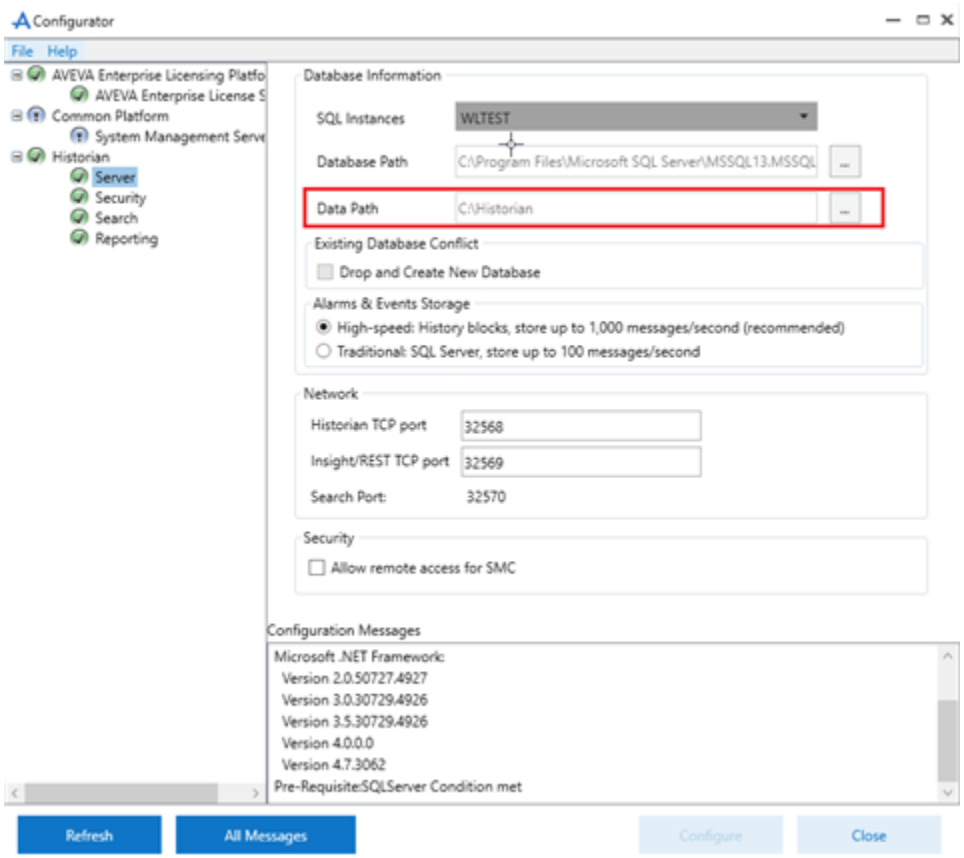
You can customize AVEVA Historian to change the look and feel of certain interface elements on the main page to match your company standards.

Configuring Customizable White Label Settings

UI customization is controlled by the presence of a "BrandingInfo.json" file. The file's location is the `Configuration` subfolder, located in the folder identified by the **Data Path** setting in the Historian Server configurator.

To configure customized white label settings:

1. Launch the configurator, and select the Historian Server node. Note the location specified by the **Data Path** setting.



2. Locate the folder named `Configuration` inside the folder identified by the **Data Path** setting. For example, if the Data Path is `C:\Historian`, then use the folder `C:\Historian\Configuration`.
3. Create a text file in the `Configuration` folder, named `BrandingInfo.json`. Copy the following text into the file to use as a template:

```
{
"HeaderBackgroundColor": "#0F76C7",
"HeaderForegroundColor": "#F4F4F4",
"SiteTitle": "AVEVA Historian Client Web",
"CompanyLogo": "/Images/AVEVA_Blue.svg",
"CompanyLogo_Link": "https://www.aveva.com/",
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
"InsightLogo": "/Images/wwo-logo-home.png",
"Copyright": "Copyright © 2020 AVEVA Group plc and its subsidiaries.<br>All rights
reserved."
}
```

4. Update the provided sample values with your own values as required:
   - HeaderBackgroundColor - This is the background color of the page header
   - HeaderForegroundColor - This is the text color of the page header
   - SiteTitle - This is the site title displayed in the page header
   - CompanyLogo - This is the path to the company logo
   - CompanyLogo_Link - This is the URL that is loaded when the company logo is clicked
   - InsightLogo - This is the path to the Insight logo displayed in the middle of the home/main page
   - Copyright - This is the copyright information that is displayed.

   Notes: A color value can be specified either as a name (ie. `blue`, `red`), or as a hexadecimal color code (ie. `#0012FF`, `#E83311`).

   The image files used for logos can be in either SVG or PNG file formats. Other file formats are incompatible.

   The paths specified for logo files are relative to the hosting directory path for Insight. The default path is `C:\Program Files (x86)\Wonderware\HistorianInsight\Server\wwwroot`. For example, if the value for **CompanyLogo** is set to `/Images/logo.png`, then the image file should be located in `C:\Program Files (x86)\Wonderware\HistorianInsight\Server\wwwroot\Images\logo.png`.
5. Save the file, and restart the **AVEVA Historian Web Client** service.
6. After the service has restarted, refresh the AVEVA Historian Web Client home page to see your custom values applied.

CORS Whitelisting

CORS origin configuration is a form of whitelisting mechanism for the back-end API to determine the origin of the web applications that are allowed to request resources from a different domain. This must be configured with the identity of the front-end web application so that the front-end application is allowed to access the API, and no other applications are allowed access.

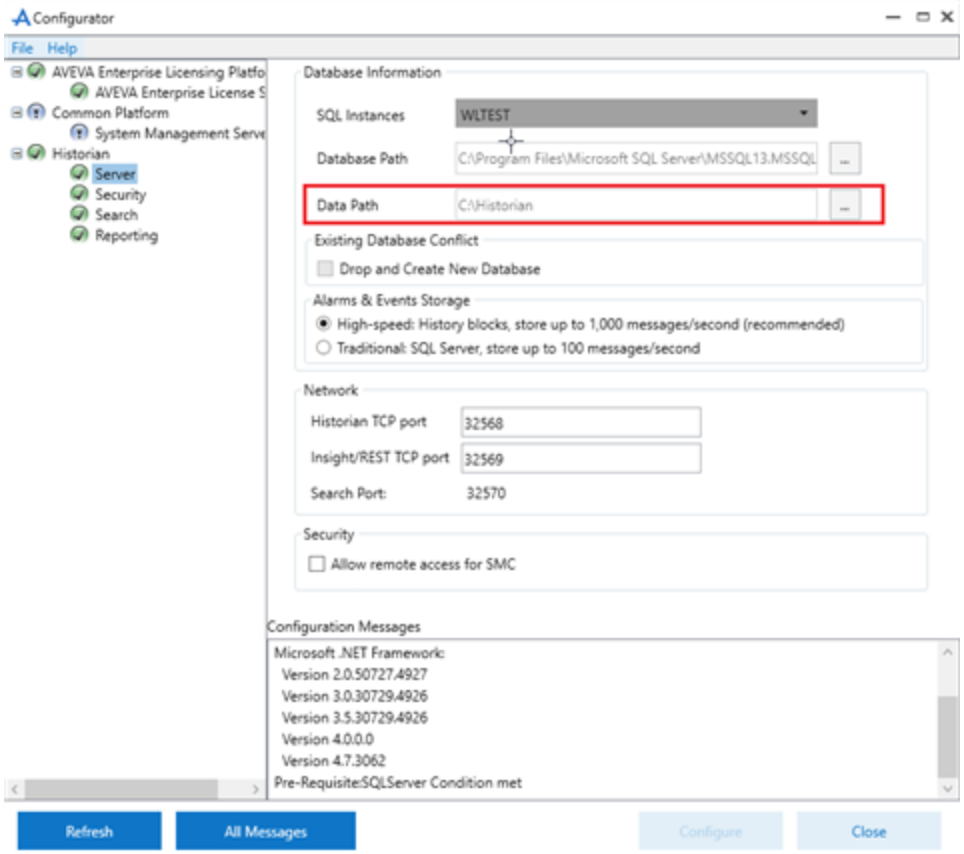
Configuring the CORS Whitelist

The CORS whitelist is controlled by the presence of a "CorsSetting.json" file. The file's location is the `Configuration` subfolder, located in the folder identified by the **Data Path** setting in the Historian Server configurator.

To configure CORS whitelist settings:

1. Launch the configurator, and select the Historian Server node. Note the location specified by the **Data Path** setting.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Locate the folder named `Configuration` inside the folder identified by the **Data Path** setting. For example, if the Data Path is `C:\Historian`, then use the folder `C:\Historian\Configuration`.
3. Create a text file in the `Configuration` folder, named `CorsSetting.json`. Copy the following text into the file to use as a template:

```
{
"Origin": ""
}
```

4. Within the second set of quotation marks, enter a comma-separated list of all the CORS origins granted access to the API, including scheme and port.

   For example, the following sample would grant API access to requests from server1.company.com on port 8080, and server.company2.com on port 80:

```
{
"Origin": "http://server1.company.com:8080,http://server.company2.com:80"
}
```

5. Save the file, then restart the **AVEVA Historian Web Client** and **AVEVA Historian Gateway** services.

Using HTTPS Instead of HTTP for Historian Client, Historian Client Web, and REST APIs

Typically, customers using Historian Client Web or the REST API can connect to a Historian server from a Historian Client or other client application using an unencrypted (HTTP) connection. (Even without an encrypted connection, the user credentials exchanged during login are still encrypted.) You can also use an encrypted connection (HTTPS) for the REST API, and this requires configuring an X.509 certificate for TLS (transport layer security).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

About TLS, HTTPS, and X.509 Certificates

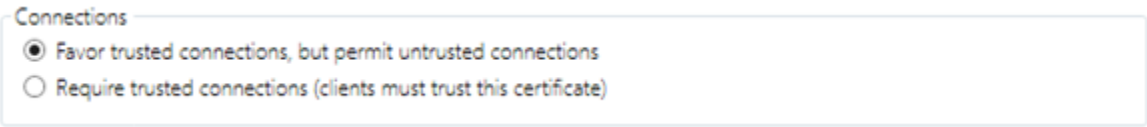
TLS allows for encrypted authentication credentials to be passed between a server and client. A certificate containing a private key is passed between the client and server to verify identification and allow access.

Using HTTPS ensures that communication between the client and server is encrypted, helping to prevent third parties from stealing or tampering with your data.

To configure the HTTPS connection to the Historian, you need an X.509 certificate. The certificate can be from a trusted authority or a self-signed certificate. During the installation and configuration of the Historian, you can import a certificate from a trusted authority if you have one, otherwise the configurator can create a self-signed certificate for you.

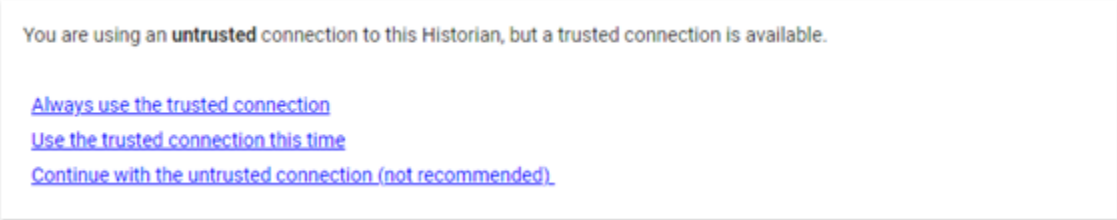
About Configuring Security

When you configure the Historian server, you choose one of two options to control what happens when a user connects using the unencrypted (HTTP) connection:



1. **Favor trusted connections, but permit untrusted connections**

   When this option is selected, users are informed there is a trusted connection available, and they can decide how to proceed using one of three options:

   - **Always use the trusted connection**

     If the user clicks this link, their browser will be permanently redirected to the HTTPS connection. Any future attempts to use the HTTP connection with the same browser are automatically redirected to the HTTPS connection without a prompt.
   - **Use the trusted connection this time**

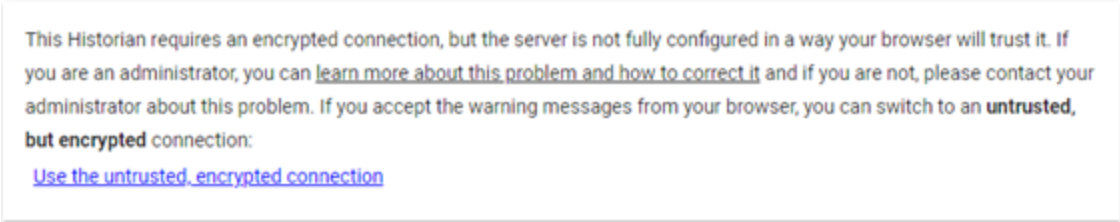
     Clicking this link redirects the browser to the HTTPS connection, but only for this session. The next time a connection is made in a new browser session, the user is prompted to choose again.
   - **Continue with the untrusted connection (not recommended)**

     If the user clicks this link, the browser continues using the HTTP connection, but only for this session. The next time a connection is made in a new browser session, the user is prompted to choose again.
2. **Require trusted connections (clients must trust this certificate)**

   When this option is selected, if you are using a certificate from a trusted authority, users are redirected to the HTTPS connection.

   If you are using an untrusted certificate, such as a self-signed certificate, the following informational message is displayed:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



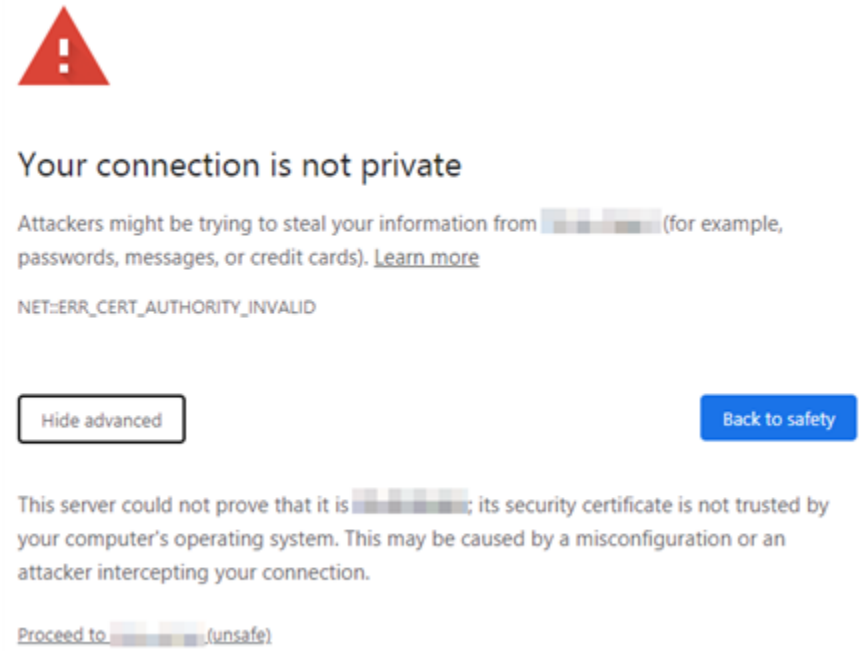
Users can click **Use the untrusted, encrypted connection** to use the HTTPS connection.

Warning: It is important to understand the risks associated with using an untrusted self-signed certificate. The browser warnings encountered while using a self-signed certificate could also indicate that the server has been compromised or hijacked by a third party. To avoid the risk of conditioning users to ignore important security warnings, follow the steps in the next section to enable remote clients to trust the self-signed certificate.

Using a Self-Signed Certificate

If you choose to use a self-signed certificate with the Historian, you are responsible for configuring all clients to trust that certificate. Clients that haven't trusted the certificate see a security warning in their browser.

For example, if you configure your Historian using a self-signed certificate, users connecting with the Google Chrome browser see a warning message similar to the following:



Enabling Trust for a Self-Signed Certificate

A self-signed certificate needs to be "trusted" for the certificate to work without warnings when you access AVEVA Historian Client Web in your browser. Trusting the certificate involves two steps:

1. Acquire a copy of the certificate.
2. Install the certificate into the trusted root certificate store.

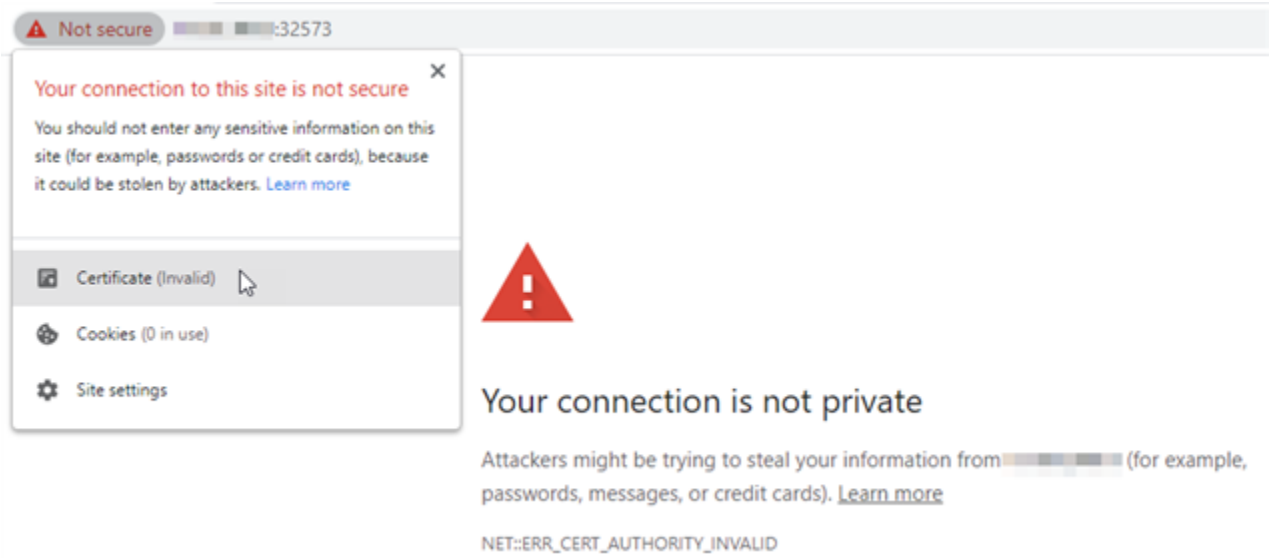
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Acquiring a Copy of the Self-Signed Certificate

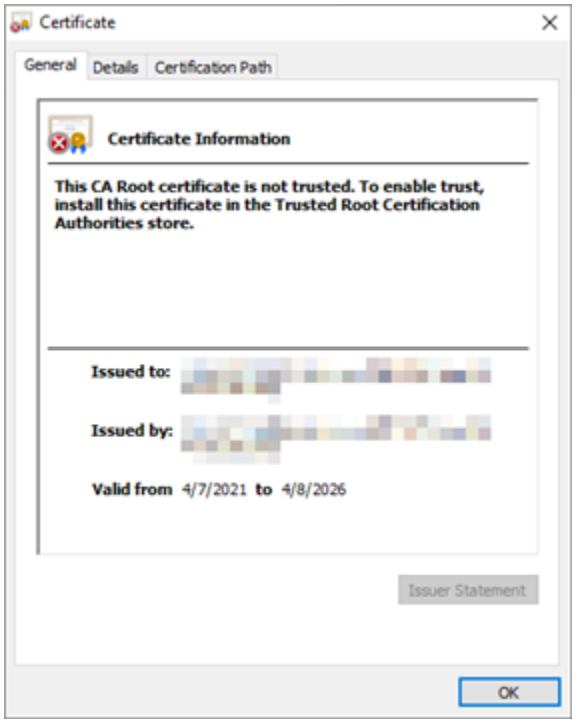
Before you can trust a self-signed certificate, you need a copy of the certificate on your system. If you already have a copy of the certificate, proceed to Trusting a Self-Signed Certificate.

To obtain a copy of the self-signed certificate:

1. In your browser, browse to the AVEVA Historian Client Web URL.
2. In the address bar, click on the warning message indicating your connection is not secure.

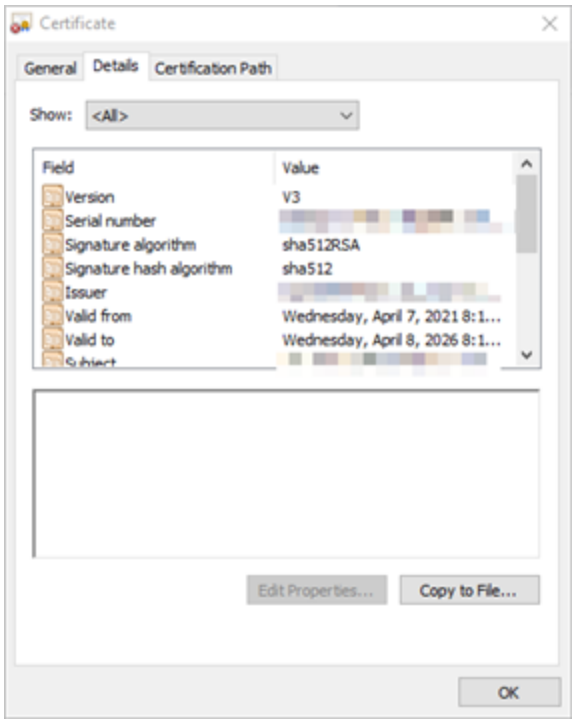


3. Click **Certificate (Invalid)**. The **Certificate** details dialog displays:

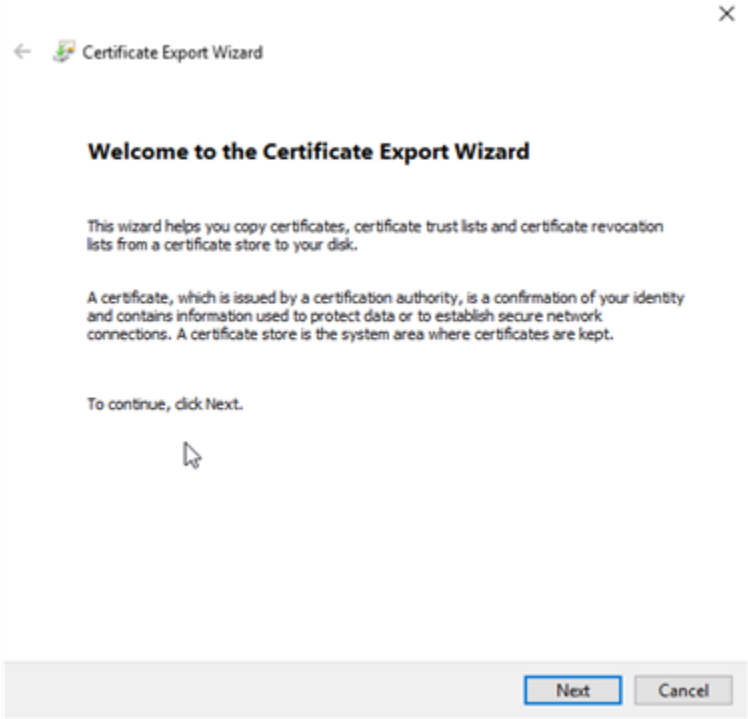


4. To trust the certificate, first you must save a copy. Select the **Details** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



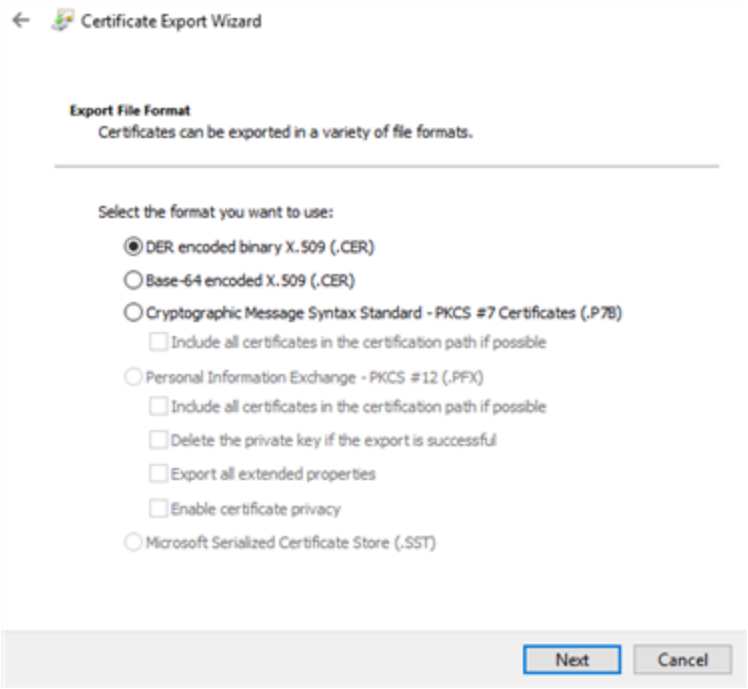
5. Click **Copy to File...**. The **Certificate Export Wizard** displays:



Click **Next**.

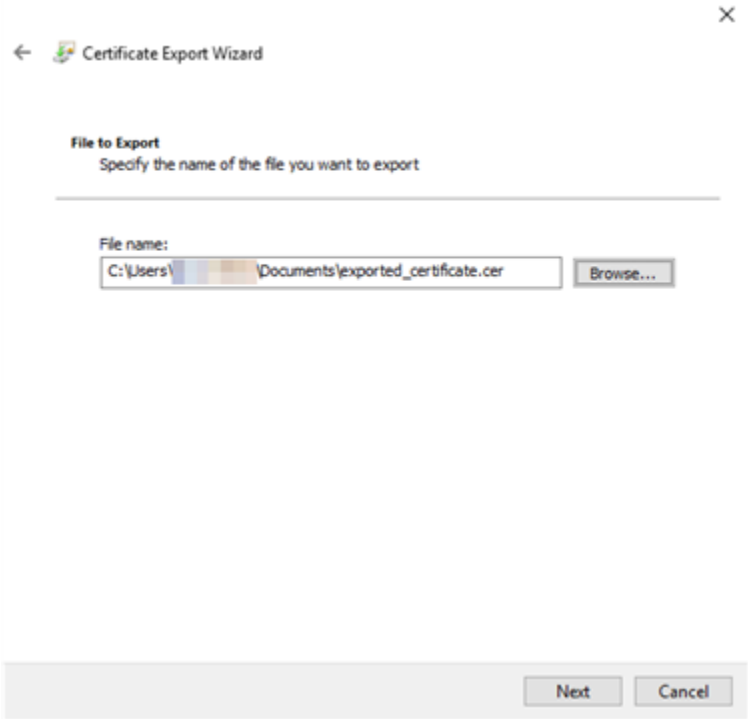
6. Select **DER encoded binary X.509 (.CER)** as the export file format:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Click **Next**.

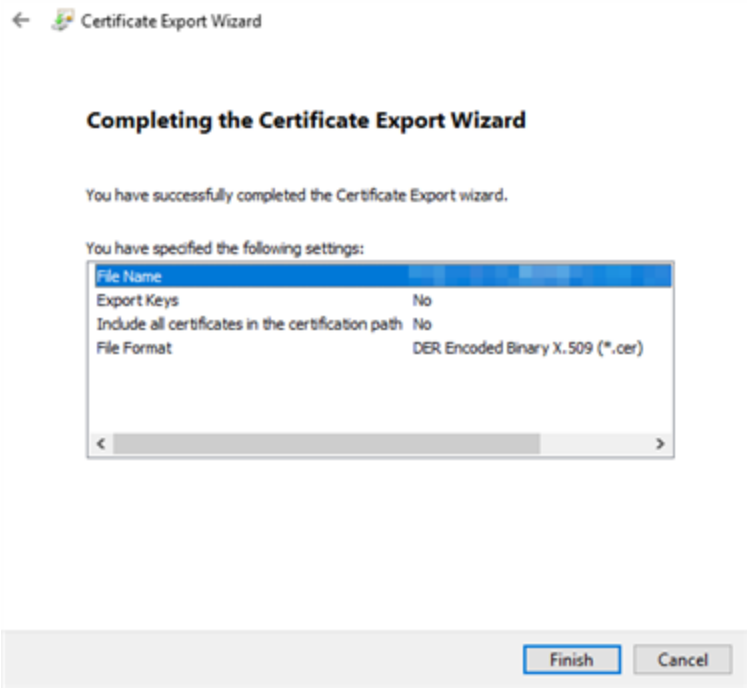
7. Click **Browse...** and choose a location to save the exported certificate.



Click **Next**.

8. Click **Finish** to export the certificate to the selected file:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



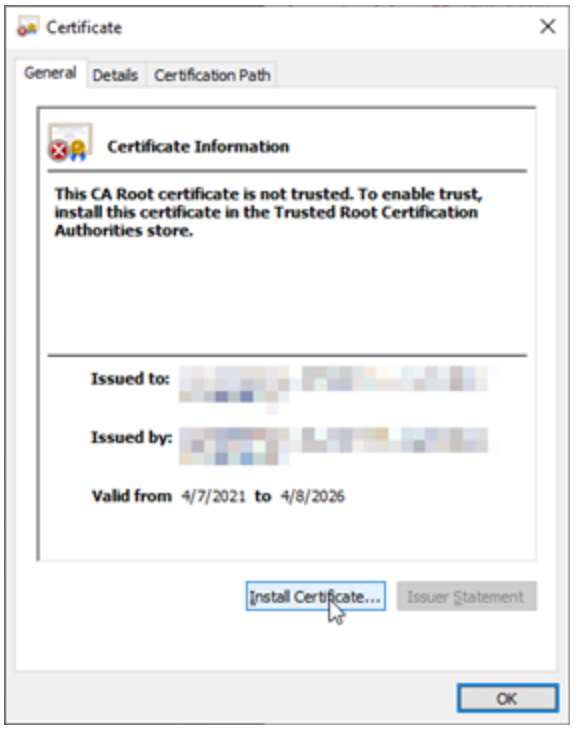
Trusting a Self-Signed Certificate

If the AVEVA Historian is configured with a self-signed certificate for TLS encryption, the certificate needs to be trusted on all client machines to avoid warning messages while using AVEVA Historian Client Web. To accomplish this, install the certificate into the trusted root certificate store on each client machine.

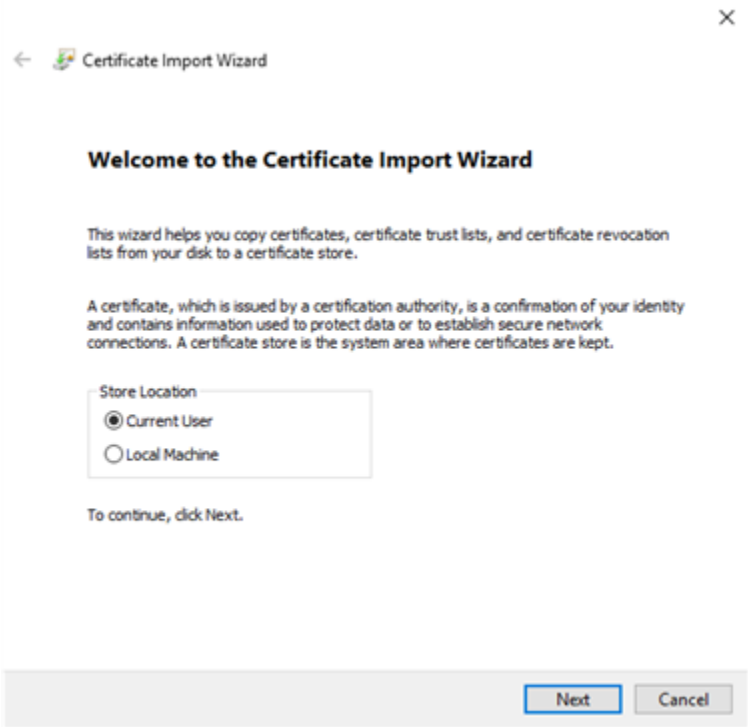
To install a self-signed certificate into the trusted root certificate store:

1. Locate and open the certificate file in Windows Explorer. The Certificate dialog displays:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. Select **Install Certificate...**. The Certificate Import Wizard displays:

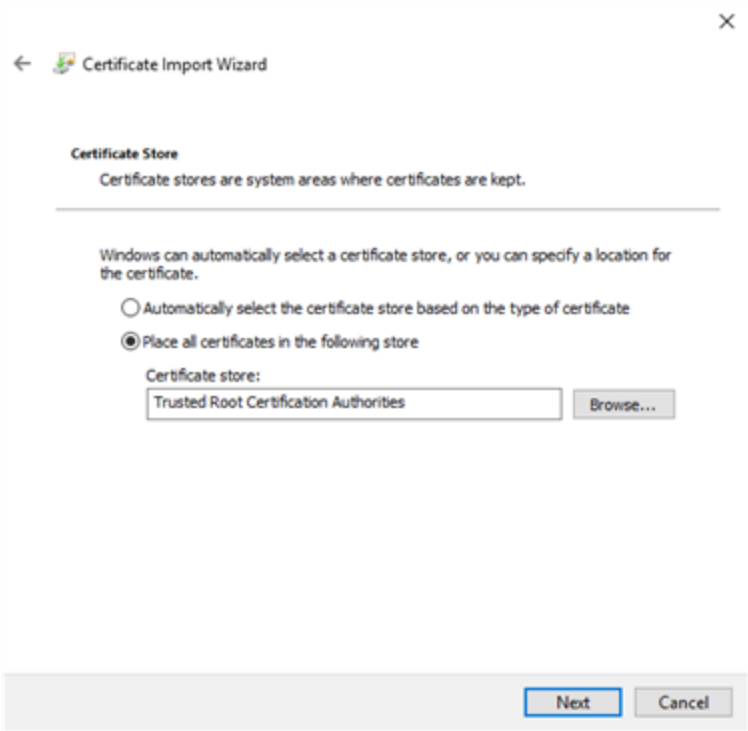


3. Select **Current User** to install the certificate for only the current user, or **Local Machine** to install the certificate for all users on this system.

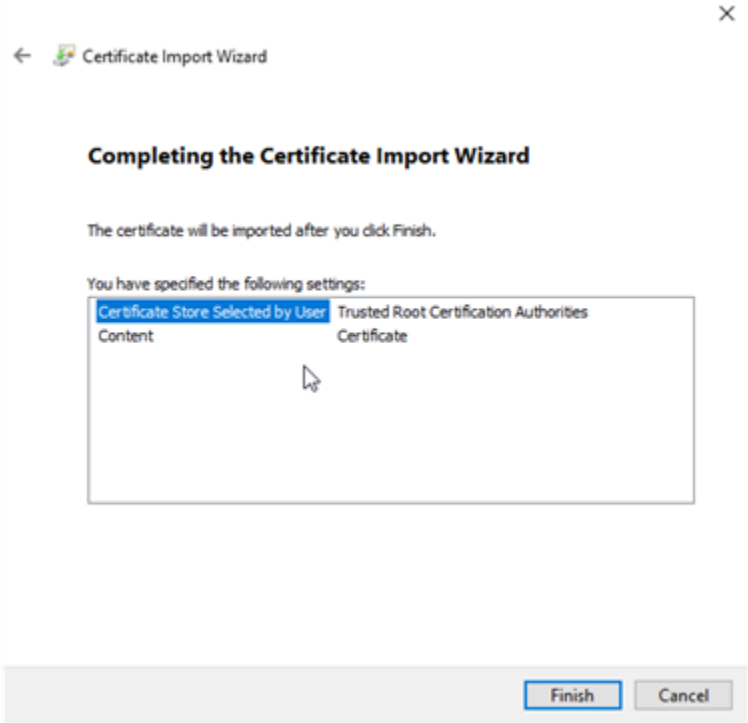
   Note: The **Local Machine** option requires administrative access to the system. If you do not have administrative access, select **Current User**.

   Click **Next**. The **Certificate Store** dialog displays:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

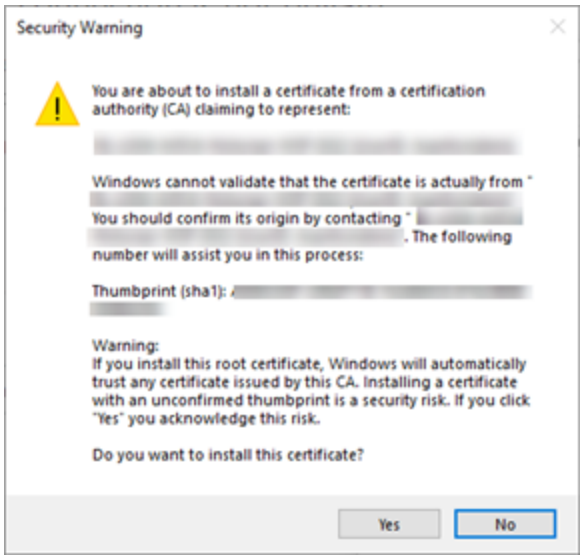


4. Select **Place all certificates in the following store**. Click **Browse...** and select **Trusted Root Certification Authorities** as the **Certificate store**.
5. Click **Next.** The **Completing the Certificate Import Wizard** dialog displays:



6. Click **Finish** to complete the Certificate Import Wizard. A security warning displays:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Click **Yes** to acknowledge the warning. The certificate is now trusted on your machine.

Monitoring the System

Performance of the AVEVA Historian can be considered in two conceptual contexts; as processes running within the Windows operating system, and as software modules acquiring and storing data.

Monitoring the General Status of AVEVA Historian

To monitor the status:

1. In the console tree, expand a server group and then expand a server.
2. Expand **Management Console** and then click **Status**. The overall status information appears in the details pane.
   - A snapshot of the current system status. For more information, see Viewing the Current System Status.
   - The status of different components of the system. For more information, see Viewing the Status of System Modules.
   - A log of status messages. For more information, see Viewing System Status Messages.

   Note: The information in the details pane is refreshed according to the rate specified in the registration properties for the server. For more information, see Registering AVEVA Historian Servers.

Viewing the Current System Status

The system status window of the details pane shows the current values for key system tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Item	Value
System time	2/20/2017 7:40:53 AM
Time of last start	2/16/2017 12:45:47 PM
Elapsed time since last start	3 dys 18 hrs 55 mins
Time of last stop	2/16/2017 12:45:03 PM
Time of last reconfiguration	2/16/2017 12:45:03 PM
Configuration status	Normal
System status	Running
License status	Valid
Total number of tags in database	281
Number of licensed tags in database	63
License tag count	500
Total number of data values receiv...	9,660,159
Overall data rate (per sec.)	27.40
Fatal errors	0
Critical errors	0
Errors	0
Warnings	2,247
Time of last error reset	2/16/2017 12:45:03 PM
Space available on circular path	52.6 GB
Space available on alternative path	Undefined or invalid path
Space available on buffer path	52.6 GB
Space available on permanent path	52.6 GB
System version	12,0,17000,000

Many of the items in the display are self-explanatory. All timestamps reflect the time of the AVEVA Historian computer, which may be different than the Operations Control Management Console running on a remote computer. However, all timestamps are formatted according to the Windows regional settings for the local computer. Descriptions for some of the items are as follows:

Time of last reconfiguration

The time that the last reconfiguration of the system was committed. For more information, see Dynamic Configuration.

System status

The current status of the system.The icon for the corresponding server in the console tree shows the current state.

Icon	State
	Server Connecting
	Server Connected
	Starting
	Running
	Server Stopping
	Stopped
	Disconnected

License status

The status of license validation. For more information on licensing, see the *AVEVA System Platform Installation Guide*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Total number of tags in database

The total number of all tags in the database.

Number of licensed tags in database

The total number of tags for which the historian will retrieve data. If the historian is unlicensed, the tag count shows the number of system tags.

Licensed tag count

The total number of tags you can configure for data retrieval in AVEVA Historian.

Total number of data values received

The number of tag values received since the Operations Control Management Console was started up. This value is continuously updated as long as the system is running.

Overall data rate

Average rate (per second) at which data values are acquired by the system.

Fatal errors, critical errors, errors and warnings

The number of errors detected since the AVEVA Historian was restarted or since an error reset was performed. For more information on errors, see System Messages.

Time of last error reset

The time that the error count was reset back to 0. For more information, see Resetting Error Counts.

Space available on XXX path

The total amount of space for historical data storage in the storage location. For more information about storage locations, see Storage Partition Locations.

System version

The current version of the AVEVA Historian.

Resetting Error Counts

Error counts are automatically set to zero at system startup and shutdown. You can also set the error counts back to zero at any time.

To reset error counts:

1. In the console tree, expand a server group and then expand a server.
2. Right-click **Management Console**, point to **All Tasks**, and then click **Reset Error Counts**. The **Reset AVEVA Historian Error Counts** confirmation box appears.
3. Click **OK**. The number of errors shown in the system status window resets to 0.

Viewing the Status of System Modules

The module status window of the details pane indicates whether or not the module is started.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Module	Status
Storage	Started
Classic data redirector	Started
Data import	Started
Replication	Started
Classic event system	Started
Retrieval	Started
Indexing	Started
OLE-DB provider	Started
Historian I/O server	Started
Client access point	Started
Metadata server	Started
Event storage	Started
System driver	Started
Data acquisition on	Started
Data acquisition on \\TEST01	Stopped

See the following table to find out more about each of these modules.

Module	For more information, see
Storage, Indexing, Metadata server, Event storage	Managing Data Storage
Classic data redirector	Classic Storage Subsystem
Replication	Managing and Configuring Replication
Classic Event system	Classic Events Subsystem
Retrieval, OLE DB provider, Historian I/O Server	*AVEVA Historian Retrieval Guide*
System driver	About System Driver and System Tags
Data acquisition, Client access point, Data import	Data Acquisition Subsystem

Viewing System Status Messages

Status messages are shown in the bottom window of the details pane. These messages are also written to the ArchestrA Log Viewer (not all messages written to the Log Viewer are shown here). For more information on the Log Viewer, see Monitoring System Messages.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time	Message
2/24/2022 4:10:48.561 PM	Started local IDAS
2/24/2022 4:10:48.324 PM	Configuring real-time data acquisition
2/24/2022 4:10:48.120 PM	Registry information acquired
2/24/2022 4:10:48.120 PM	Configuring system driver
2/24/2022 4:10:48.105 PM	Configuration information acquired
2/24/2022 4:10:46.556 PM	Reading configuration information from database
2/24/2022 4:10:46.525 PM	Reading configuration information

Viewing Status Information

Using the Management Console, you can monitor five main areas of the system: general system status, data acquisition, client connections, history blocks, and legacy error messages. These items appear in the console tree.

- If you click **Status**, the details pane shows the overall status for the AVEVA Historian, such as whether the server is running, the number of system errors, and the time since the last startup.
- If you click **Data Acquisition**, the details pane shows each data source (IOServer\topic or other client) that is supplying the historian with data.
- If you click **Replication**, the details pane shows a list of servers to which this Historian is replicating data. On the server actually receiving the data, there will be a corresponding entry under its **Data Acquisition** node.

  For more information on tag replication, see Managing and Configuring Replication.
- If you click **Clients**, the details pane shows the status of all clients that are currently connected to the historian.
- If you click **History Blocks**, the details pane shows a list of all of the history blocks stored on the historian computer.

  For more information on administering history blocks, see Managing Partitions and History Blocks.

  For more information on monitoring the general status, data acquisition, client connections, and the system message log, see Monitoring the System.

If you have multiple historian servers registered in the console, make sure that you select the server you want to manage before you right-click in the tree to select a short-cut menu command.

Monitoring Data Acquisition

You can monitor the status of data acquisition from all configured data sources. You can monitor how individual data sources are performing compared to past history or to another data source.

For more information on data acquisition, see Configuring Data Acquisition.

To view data acquisition status

1. In the console tree, expand a server group and then expand a server.
2. Expand **Management Console** and then click **Data Acquisition**. Data acquisition information appears in the details pane.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Computer	Topic	Protocol	Tags	Status	Values	Rate	Connections
[redacted]	\\[redacted]\SysDrv!System	SuiteLink	215	Receiving	132,292	125.12	1
TEST01	\\New_IOServer\FSGateway!new_topic1	SuiteLink	0	Idle	0	0.00	0
[redacted]	\\[redacted]\aahReplication(23.0.000)!...	HCAL	197	Receiving	5,383	0.00	1

Tier-1 historian sources appear in this pane when you view the data acquisition for a tier-2 historian.

Note: The information in the details pane is refreshed according to the rate specified in the registration properties for the server. For more information, see Registering AVEVA Historian Servers.

Column descriptions are as follows:

Computer

The name of the computer on which the data source runs.

Topic

The name of the topic.

Protocol

The protocol used by the AVEVA Historian to communicate with the data source.

Tags

The total number of tags associated with the data source.

Status

The status of data acquisition from the data source.

Values

The total number of tag values received from the data source.

Rate

Average number of data values received from the topic per second.

Connections

Number of connections to the I/O Server for the topic. This number is incremented.

Monitoring Replications

You can monitor the status of replications, including how well data is being replicated on the servers.

For more information on replication, see the Managing and Configuring Replication.

To view replication status

1. In the console tree, expand a server group and then expand a server.
2. Expand **Management Console** and then click **Replication**. Replication information appears in the details pane.

Replication Server	Status	Total Tags	Values / Sec	Sync Queue Values ...	Total Values
Local Replication	Replicating	1	0.00	0.00	0
T2_Historian	In StoreForward	1	0.00	0.00	0
T2_Historian	In StoreForward	1	0.00	0.00	0

Note: The information in the details pane is refreshed according to the rate specified in the registration properties for the server. For more information, see Registering AVEVA Historian Servers.

Column descriptions are as follows:

Replication Server

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The name of the replication server.

Status

The status of replication to the replication server.

Total Tags

The total number of tags being replicated to the replication server.

Values/Sec

The average number of data values replicated per second.

Sync Queue Values/Sec

The average number of synchronization queue values replicated per second.

Total Values

The total number of values being replicated.

Monitoring Client Connections

To view the client connection status

1. In the console tree, expand a server group and then expand a server.
2. Expand **Management Console** and then click **Clients**. The client connection information will appear in the details pane.

ID	Application	Computer	User	Connected At	Duration
8	aahReplication(23....		NT SERVICE\InSQLConfig...	5/16/2022 8:00:34 AM	7 hrs 21 mins 40 ...
5	aahIDAS(23.0.000)		NT SERVICE\InSQLConfig...	5/16/2022 8:00:16 AM	7 hrs 21 mins 58 ...
2	aahReplication(23....		NT SERVICE\InSQLConfig...	5/16/2022 7:59:35 AM	7 hrs 22 mins 39 ...
7	aahReplication(23....		NT SERVICE\InSQLConfig...	5/16/2022 8:00:31 AM	7 hrs 21 mins 43 ...
4	sqlservr(23.0.000)		InSQLConfiguration	5/16/2022 7:59:45 AM	7 hrs 22 mins 29 ...
1	aahReplication(23....		NT SERVICE\InSQLConfig...	5/16/2022 7:59:33 AM	7 hrs 22 mins 41 ...
9	aahReplication(23....		NT SERVICE\InSQLConfig...	5/16/2022 8:00:34 AM	7 hrs 21 mins 40 ...

Note: The information in the details pane is refreshed according to the rate specified in the registration properties for the server. For more information, see Registering AVEVA Historian Servers.

Column descriptions are as follows:

ID

Unique number that the AVEVA Historian assigns to the client.

Application

The executable name of the application that is accessing the historian.

Computer

The name of the computer on which the application in running.

User

The Windows login name under which the client application is running.

Connected At

The start time of the connection.

Duration

The length of time that the client has been connected.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For the **Connected At** and **Duration** columns, the timestamp reflects the time of the historian, shown using the Windows regional settings for the local computer.

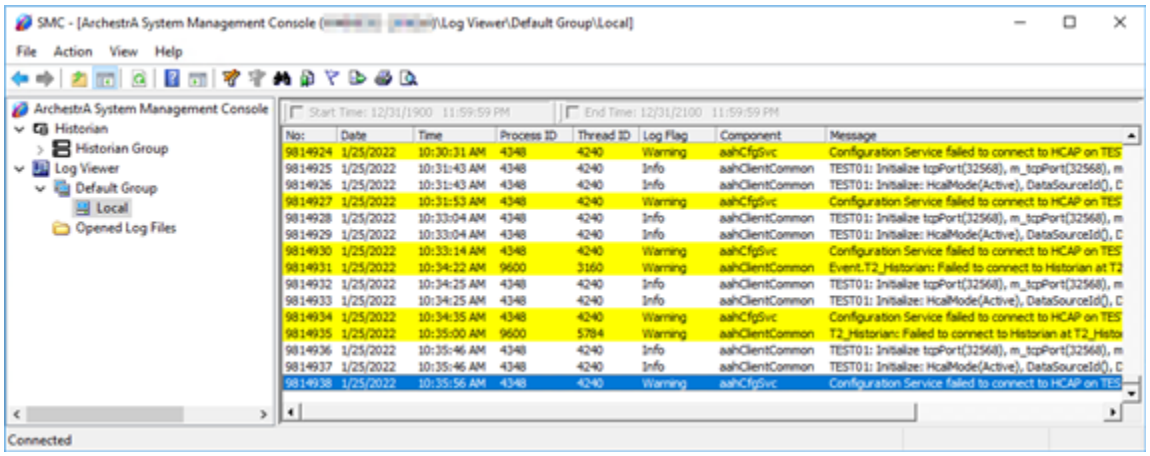
Monitoring System Messages

System messages provide information about the state of the AVEVA Historian as it starts up, runs, or shuts down. For more information about system messages, see System Messages .

From within the Operations Control Management Console, you can view the system messages generated by the AVEVA Historian using the Log Viewer.

To view system messages

1. In the console tree, expand **Log Viewer** and then expand **Default Group**.
2. Click **Local**. All of the messages appear in the details pane.



For more information on the Log Viewer, see the Log Viewer documentation.

Viewing Errors in the Windows Event Viewer

The Event Viewer is a Windows administrative tool for managing error log files. It allows the logs on any workstation or server to be viewed from any other workstation or server that is connected by a network. The Event Viewer can be used to check the logs for the following type of messages:

- Error messages from the operating system.
- Messages confirming that scheduled Windows events occurred correctly.
- Error messages from AVEVA Historian.

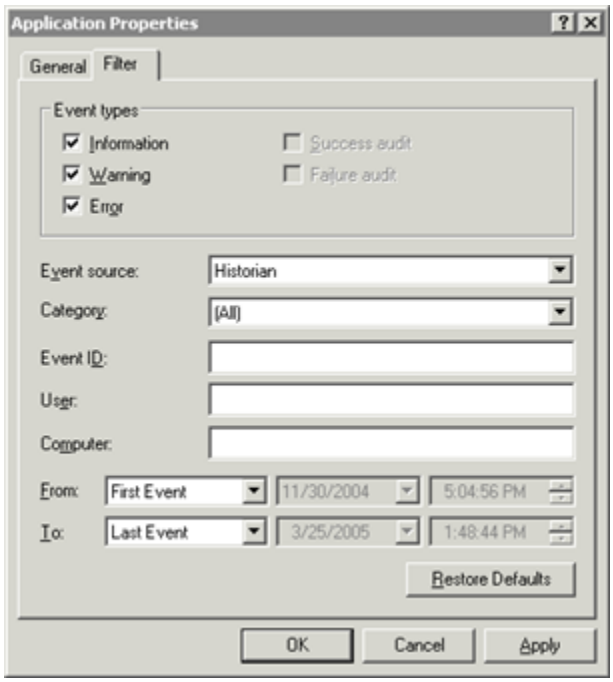
There are hundreds of messages that can appear in the logs, depending on how your system is configured and how healthy it is. It is important to know what the messages mean and what action is required.

To view errors in the Event Viewer

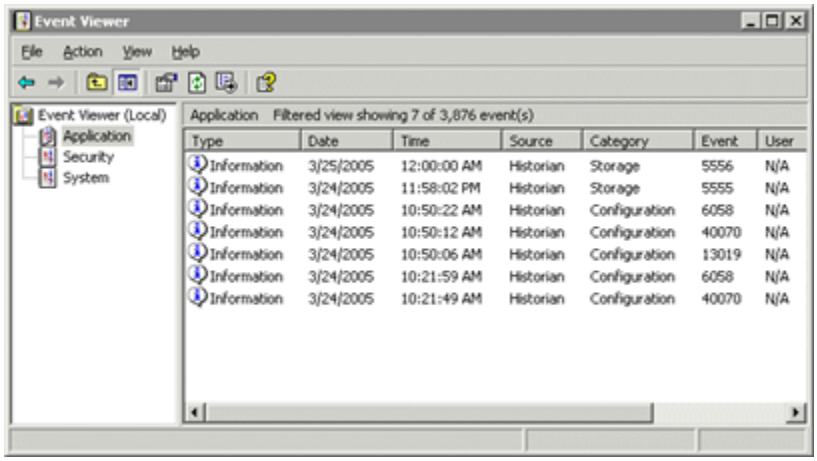
1. Start up the Event Viewer.
2. In the console tree, click **Application**. Messages from all applications appear in the details pane.
3. On the **View** menu, click **Filter**. The **Application Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

4. Click the **Filter** tab.



5. In the **Event source** list, click **Historian**.
6. Click **OK**. The details pane shows only AVEVA Historian errors.



7. To view the message text, double-click the message in the details pane.

Monitoring System Tags from within InTouch HMI Software

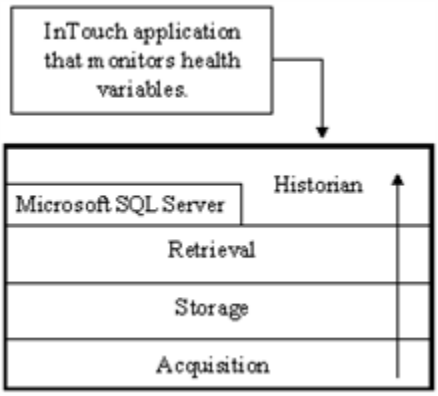
The overall health of an AVEVA Historian is monitored continuously by a dedicated system driver. Critical system variables (throughput rates, errors, remaining disk space, and so on) and timing counters are acquired by the system driver and stored in the same manner as plant tags. This driver allows remote monitoring of the current and historical state of the historian, and alerts users to problems in the system.

For more information on the system tags, see About System Driver and System Tags.

For example, you can write an InTouch application that monitors the historian system health tags. Monitoring the overall health by means of system tags is done from the "top" of the system, ensuring that each layer of the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

system is working properly, from the acquisition layer up through the historian:



Using Windows Performance Logs and Alerts

You can use Microsoft Performance Logs and Alerts console to monitor system variables that pertain to your computer's operating system and hardware. Performance Logs and Alerts allows you to view different types of counters that have been incorporated into the Windows operating system.

In Performance Logs and Alerts, "counters" are associated with objects and with instances of objects. Objects include memory, processes, servers, system, and so on. Instances of objects identify, for example, specific processes. Counters include such measurements as percentage of processor time, private bytes, available memory, and so on. The available counters depend on the object and the instances selected.

You can select one or more process instances so that the Performance Logs and Alerts provides measurements of counters for, for example, all the AVEVA Historian processes running within the Windows operating system.

Using counters within Performance Logs and Alerts can provide valuable information to assist in system tuning and to identify bottlenecks in a sluggish system. Using the Pool Non-paged Bytes counter of the memory object, for instance, can identify memory leaks that contribute to a poorly responsive system.

For information on using Performance Logs and Alerts, see the documentation for your Windows operating system.

Browsing the ArchestrA Model View Using Historian Clients

You can configure WinPlatforms and AppEngines in Application Server so that the ArchestrA model view for objects and attributes hosted by these objects can be replicated to the AVEVA Historian. Galaxies and objects in ArchestrA are represented in the historian as groups in the public namespace.

You can then browse the model view representation using any historian client application that shows the historian public groups, such as the Historian Client Trend.

Model View Representation in the Historian Namespace

The ArchestrA model view namespace is represented in the AVEVA Historian database as a public group namespace. Each Galaxy and object in the model view is represented as a namespace group in the database.

The top-level group reflects the name of the Galaxy. The top-level Galaxy group contains a group for every child Area and object, so that the ArchestrA object hierarchy is accurately reflected in the group/sub-group structure.

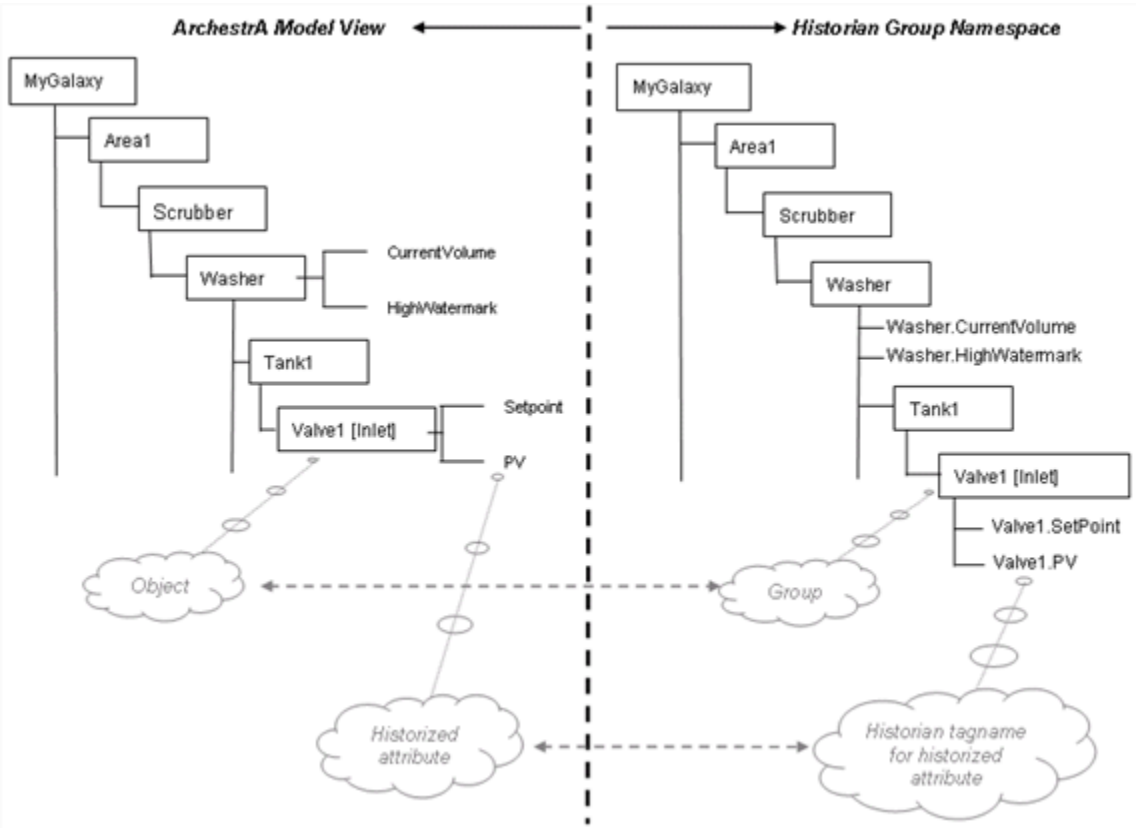
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Only one Galaxy can be represented in a single historian.

Each group under the Galaxy group is named according to the object in the model view that it represents. Each group can contain:

- Additional child groups.
- Historian tagnames for the historized attributes of the object that the group represents.
- Groups that represent objects without any historized attributes, if the objects contain child objects with historized attributes or if they contain special types of objects, such as traceability objects.

The following illustrates the mapping between a sample ArchestrA model view namespace and the corresponding group namespace in the historian:



Model View Replication to the Historian

Model view replication is the process of sending the model view hierarchy information to the AVEVA Historian and incorporating it into the public group namespace. Replication is initiated by ArchestrA.

Replication is a two-step process:

1. The historian is checked to see if there is a need to update the public group namespace with information from the model view.

   Replication only occurs when the public group namespace is determined to be out of sync with the model view, in terms of objects with historized attributes and/or special types of objects, such as traceability objects.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. The required model view changes are transmitted to the historian and incorporated into the public namespace.

   During replication, only the object/attribute information that needs to change in the historian is transmitted and processed, with the exception of Area objects. For example, if one object is added to a Galaxy of 1000 objects, the new object will be the only entity transmitted to the historian and incorporated into the public group namespace. Because the amount of data for Area objects is small, all Area information is transmitted and processed, without negatively impacting performance.

ArchestrA objects and attributes that can be replicated include:

All historized attributes.

- All objects that contain historized attributes.
- All objects that contain other objects with historized attributes. This allows for representation of the complete hierarchy from the Galaxy level down to lowest-level object that has historized attributes, even if objects at intermediate levels do not have any historized attributes.
- Some special types of objects that do not typically have historized attributes, such as traceability objects. Also, their parent objects are replicated, as needed, to fill out the entire hierarchy.

Attributes that are not historized do not appear in the historian namespace.

Replication occurs when:

- Objects with historized attributes and/or traceability objects are deployed.
- Objects with historized attributes and/or traceability objects are redeployed.
- The historian starts up, and there was a relevant change to the model view while the historian was offline. There may be a delay in the replication.

If you undeploy or delete an object, the changes will not be replicated until you perform a redeploy.

If replication fails to complete (for example, due to a network failure), ArchestrA will try to send the model information again during the next scan cycle, until the replication succeeds. No error message is logged to the ArchestrA Logger if replication fails; however, you can log a custom message using the "ModelViewSync" custom log flag.

Replication Configuration using the IDE

Use the System Platform IDE to enable or disable the automatic replication of the ArchestrA model view to the AVEVA Historian computer.

Replication is configured at the platform and engine levels.

Configuring Replication for a WinPlatform

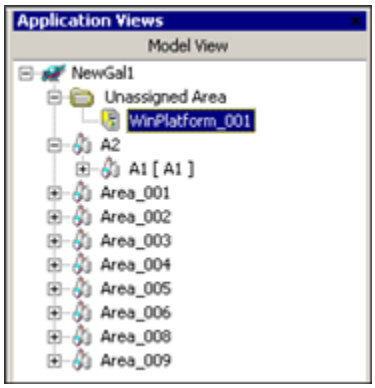
Configuring replication at the platform level simply enables any WinPlatform attributes, marked for historization, to be associated with the WinPlatform in the historian namespace.

To configure model view replication for a WinPlatform

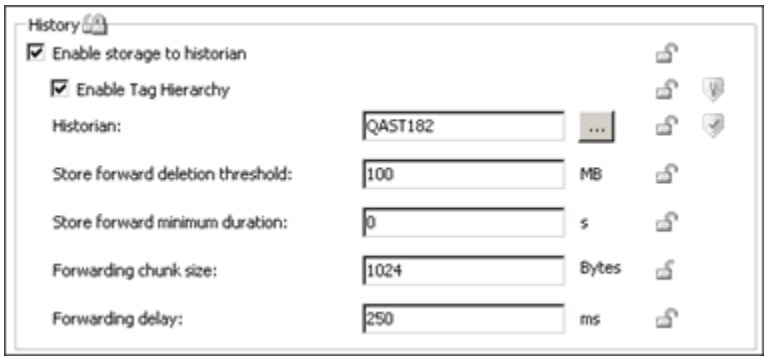
1. Start the IDE.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. In the model view, browse to the WinPlatform that you want to configure.



3. Open the object editor for the selected WinPlatform.
4. Click the **Engine** tab.



5. Select the **Enable storage to historian** check box, if not already checked.
6. Select the **Enable Tag Hierarchy** check box.
7. In the **Historian** box, specify the name of the AVEVA Historian computer.
8. Configure other history settings as required.
9. Close the editor, saving your changes.
10. Close the IDE.

Configuring Replication for an AppEngine

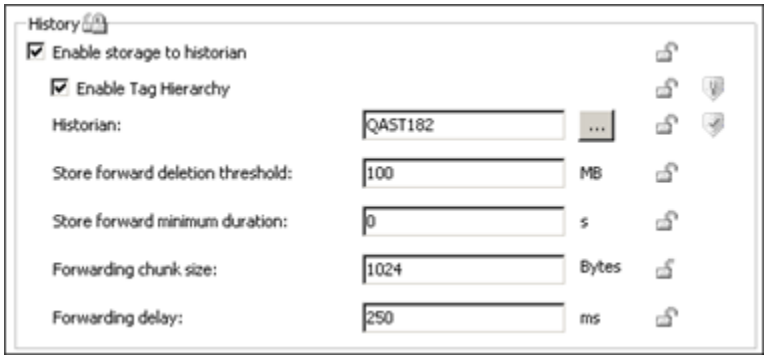
Configuring replication at the engine level enables the replication of AppEngine attributes marked for historization, as well as the replication of all qualifying objects hosted by the AppEngine, and their attributes.

To configure model view replication for an AppEngine

1. Start the IDE.
2. In the model view, browse to the AppEngine that you want to configure.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Open the object editor for the selected AppEngine.
4. Click the **General** tab.



5. Select the **Enable storage to historian** check box, if not already checked.
6. Select the **Enable Tag Hierarchy** check box.
7. In the **Historian** box, specify the name of the AVEVA Historian computer.
8. Configure other history settings as required.
9. Close the editor, saving your changes.
10. Close the IDE.

Enabling Replication at Runtime

You can enable or disable replication at run time without having to undeploy and redeploy the engine of any affected objects. To do this, set the Engine.Historian.EnableTagHierarchy attribute to True. This attribute is available for both WinPlatform and AppEngine objects.

Viewing Historized Attributes in the AVEVA Historian Configuration Editor

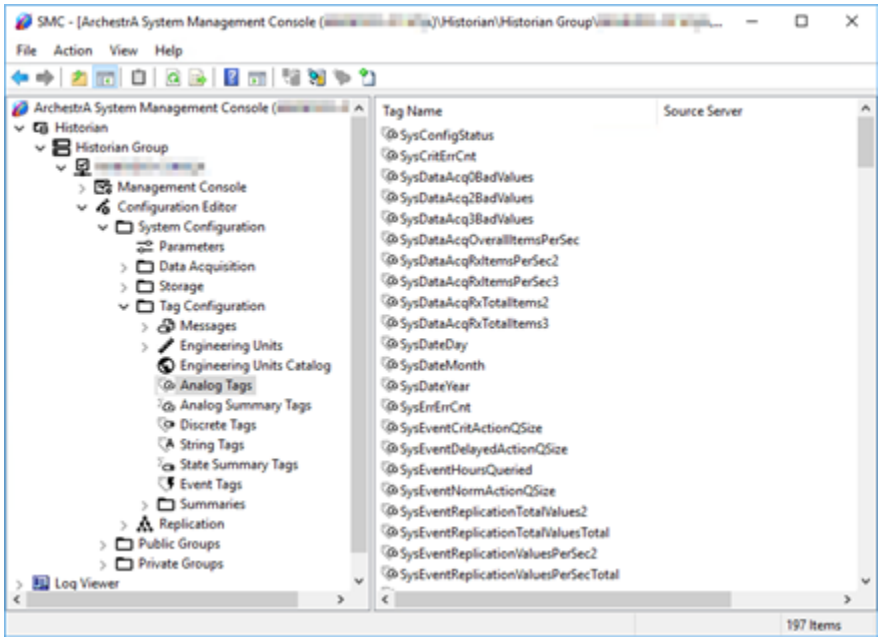
Note: You cannot view the model hierarchy in the Operations Control Management Console.

To view historized attributes

1. In the Operations Control Management Console, expand a server group and then expand a server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Select any of the tag type groups, such as **Analog Tags**.
4. In the details pane shows all tags of that type, including historized attributes from AVEVA Application Server.



Browsing the Model Hierarchy in a Historian Client

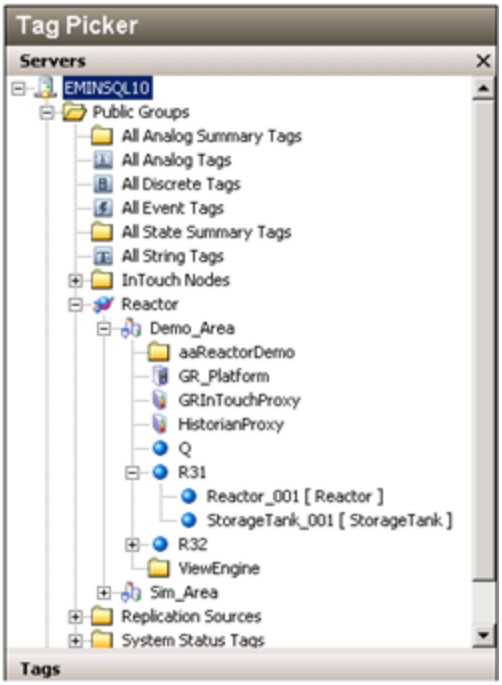
You can browse the ArchestrA model view hierarchy in any AVEVA Historian client that incorporates the **Public Groups** folder in the navigation tree, such as the Historian Trend client.

It is recommended that you not modify the model view hierarchy replication either directly from the database or by using an application such as the Historian Trend client.

To browse the model hierarchy

1. Start a client, such as the Historian Trend client.
2. Connect to the historian.
3. In the navigation tree, expand **Public Groups**.
4. Expand the group that reflects the name of the Galaxy you want to browse.

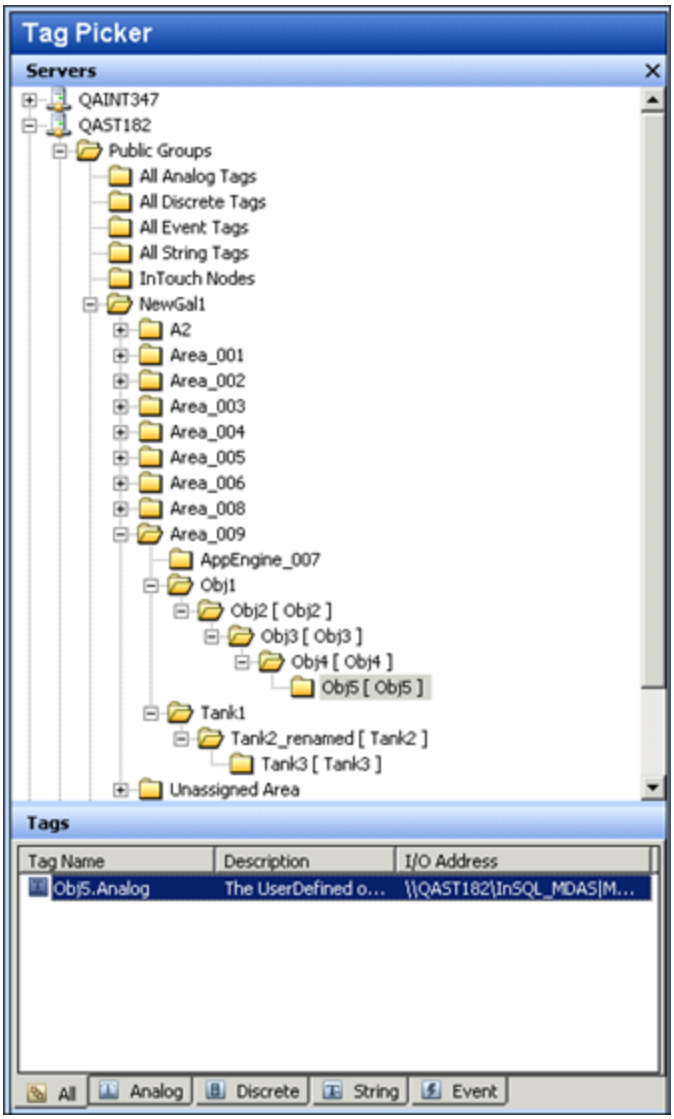
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



5. Navigate through the ArchestrA model view hierarchy and select a group representing an object with historized attributes.
6. Select attributes for which you want to view history data in the client.

For example, when you select a group in the Trend Tag Picker, the **Tags** pane shows a list of all the historian tagnames representing the historized attributes of the selected group (object).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If only the objects at the bottom of the model view hierarchy are deployed, the names for the objects higher in the hierarchy are not available to clients. To enable replication of the hierarchy in other applications, ArchestrA generates generic names for the undeployed objects. Client applications display these generic names instead of the actual names that appear in the IDE.

Legacy Features

If you install AVEVA Historian from an earlier version, you may still use some legacy features when

- Data is collected by previous historian versions
- Events are stored in A2ALMDB
- Classic Event subsystem is used for notifications
- Classic Storage subsystem was used for classic IDASes.

  Note that the Classic Storage subsystem is now replaced by the Classic Data Redirector process (aahStoreSvc.exe), which performs the same functionality.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

This table compares legacy features with upgraded features in Historian 2017:

Legacy feature	Upgraded feature in Historian 2017
Classic storage subsystem	The current storage subsystem. See Managing Data Storage.
Classic events subsystem (now Classic Data Redirector process)	Events are managed through the Alarms and Events subsystem.
A2ALMDB	New events are written to special history blocks for events. For more information, see Managing Partitions and History Blocks.

Classic Storage Subsystem

Note: The Classic Storage subsystem is now replaced by the Classic Data Redirector process (aahStoreSvc.exe), which performs the same functionality.

Starting with AVEVA Historian 2014 R2, classic storage as a subsystem no longer exists. However, all historical data that was collected by the Classic Storage subsystem from previous releases remains fully accessible in a seamless manner.

Any tags previously configured for classic storage will automatically use storage after you install AVEVA Historian 2014 R2 or later.

Starting with AVEVA Historian 2014 R2, data from the following sources are accepted by a classic storage "redirector" service (aahStoreSvc.exe) and sent to the current Storage subsystem.

- A remote IDAS that has not been upgraded to Historian 2017
- System driver

Memory Management for Retrieval of Classic Storage Data

The AVEVA Historian Indexing Service (aahIndexSvc.exe) is used to retrieve data that was stored by the Classic Storage subsystem.

For large systems, it is possible that loading the tag information from all of the history blocks will require more memory than the 2 GB limit that is imposed by the Windows operating system for a single process. The actual limit may be even be less than 2 GB, if the amount of installed RAM is insufficient.

The total amount of tag information for the history blocks depends not only on the total number of tags, but also on the number of tag versions, which are created during modifications to old data. Therefore, it is recommended that you monitor the memory consumption for all systems, large and small, if you are regularly performing data inserts, updates, or CSV file imports.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To avoid excessive memory consumption by the AVEVA Historian Indexing Service, tune and monitor the service for your system using the following system parameters and system tags.

- **HistoryCacheSize and HistoryDaysAlwaysCached system parameters.**
  You can limit the maximum amount of memory the Indexing Service can use for tag information by adjusting the value of the *HistoryCacheSize* system parameter. When this parameter is set 0 (default), the Indexing Service selects a default cache value automatically by taking into account the amount of installed physical memory and the maximum available address space for the process. In some rare cases when some specific performance tuning is needed, you may want to set the HistoryCacheSize parameter manually. In this case, the Indexing Service uses the specified value, but still may automatically change the effective HistoryCacheSize if the specified value is too low or too high.

  Regardless of whether the effective HistoryCacheSize was selected automatically (default) or specified by you, the Indexing Service manages the cache using a "least-recently used" algorithm. In this algorithm, when there is a request to access a history block that is not currently cached, the Indexing Service unloads the tag information from the least-recently used history block and then loads the tag information from the requested block.

  All of these operations are performed automatically in the background, but you may notice a slowness data retrieval if the data is retrieved from a block that is not currently loaded into memory. Keep in mind that the smaller the amount of memory that you allocate for the cache, the potentially longer it may take to service data requests.

  To guarantee the maximum retrieval performance for the newest history blocks (for example, if you a running a trend application for the last week), you can "lock" a certain number of the most recent history blocks in the cache. To do this, set the number of days to be locked in the cache by changing the *HistoryDaysAlwaysCached* system parameter.
- **SysHistoryCacheFaults and SysHistoryCacheUsed system tags.**
  To determine if you need to clamp the memory used by the Indexing Service, use the Windows Task Manager application or the Performance console to see how much memory is used by the aahIndexSvc.exe process. Also, you can monitor the SysHistoryCacheFaults and SysHistoryCacheUsed system tags. A high number of cache faults may be indicating that the cache size is insufficient. The SysHistoryCacheUsed system tag shows the number of bytes currently used for keeping the tag information. This tag may be helpful to see how much memory is consumed by the tag information, even if the memory management is not enabled.

At any time, you can observe the current status of the history blocks in the AVEVA Historian Management Console. When the tag information from a history block is not loaded into memory, the history block icon is dimmed. You can manually refresh the console window to see changes in the status for the history blocks.

About the Real-Time Data Window

The real-time data window is applicable only for store-and-forward data from remote IDASes that are not upgraded to Historian 2017.

The real-time "window" is the maximum delay, relative to current time of the server, in which data is considered real-time by storage. The real-time window can range from -30 seconds to +999 milliseconds of the current server time.

- For real-time data, the timestamp for the data value must fall within the time window.
- For late data, the timestamp for the data value can fall either inside or outside of the time window, depending on whether the late date setting is enabled for the topic. Note that late data can be processed

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

very slowly if the timestamp falls outside of the real-time window.

- For non-streamed data, the data can have any timestamp.

The following rules apply when storing data with timestamps relative to the real-time window:

- If the late data setting for a tag topic is not enabled and the received data is more than 30 seconds late, the value is discarded, and a warning is logged. If the received data value is within 30 seconds of the server time, it is stored as received.
- If the late data setting for a tag topic is enabled and the received data value is within the real-time window, then the value is stored by the real-time storage service with no changes. If the received data value is outside of the real-time window, then the value is passed to the alternate storage services and stored without changes.
- If the late data setting for a tag topic is enabled, the received data value is stored in delta mode, even if the tag is configured for cyclic storage and the received data value is within the real-time window.

For more information on the late data setting for a topic, see IDAS Late Data Handling.

You can adjust the real-time window for "late" data topics by configuring the RealTimeWindow system parameter. The real-time window for regular real-time data (not "late") is fixed as 30 seconds. For more information, see Editing System Parameters.

Adjusting the real-time window also has implications if you are using delta storage with a swinging door deadband. For more information, see Swinging Door Deadband for Delta Storage.

If the system does not have enough memory to adequately process real-time data, the window is adjusted internally. An appropriate message is logged. The value of the RealTimeWindow system parameter, however, remains unchanged.

Important: The real-time window is not intended to accommodate time synchronization problems between an IDAS and the historian. It is imperative that you properly synchronize the IDAS and historian. If the IDAS is sending data in a steady stream outside of the real-time window, it is likely there is a time synchronization problem. For more information, see Time Synchronization for Data Acquisition.

If a data value is discarded because it did not fit the requirements of the real-time window, the historian logs a warning message. Warning messages are logged at one intervals during the period when data is being discarded.

Determining If the Real-Time Window Is Configured Appropriately for a Swinging Door Deadband

To determine if the real-time window is configured correctly for a swinging door deadband, look at the number of data values that are forced to be stored while the system waits for the next valid data point to make the filtering calculation.

The SysRateDeadbandForcedValues system tag counts the number of "extra" points forced to be stored as a result of an insufficient real-time window for swinging door storage. Also, you can determine the number of points forced to be stored for an individual tag by executing a query that uses the full retrieval mode and specifies a quality detail of 2240, which indicates that these points were stored because of an insufficient real-time window.

If you find a large number of forced storage points, you can either reconfigure the tag to use delta storage or increase the real-time window.

Note: The first two points received for a tag configured for swinging door storage are always stored.

Also, use caution when setting the real-time window to accommodate a swinging door deadband.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- If your system has a large tag count or high data throughput, increasing the real-time window will increase the memory requirements for storage, because the storage system will have to process more data as real-time data, which is more resource-intensive than the storage of late data.
- If you increase the real-time window and you apply a swinging door deadband to a slow-changing tag, the amount of storage space required increases because the tag value is forced to be stored more often than if you used delta storage with no deadband.

Classic Event Subsystem

Plant events range from startups and shutdowns, through trips and shift changes, to batch events and operator actions.

You can use the AVEVA Historian Classic Event subsystem to detect events and associate actions when they are detected. At a basic level, anything that can be determined by examining stored data can be used as an event. The Classic Event subsystem can be configured to periodically check to see if an event occurred. This is called event detection. A subsequent action can then be triggered after an event is detected. However, there is no guarantee of immediacy for actions; in fact, other mechanisms can preempt actions under certain circumstances.

For the historian, event storage encapsulates more than just the fact that something happened. An event is the set of attributes describing the moment a detection criterion is met on historical tag values in the historian. Attributes of an event include the date and time that the event occurred in history and the date and time that it was detected. Records of detected events can be logged to the database regardless of whether or not any configured actions are subsequently initiated. In other words, sometimes it may be desirable to simply log the fact that an event occurred without initiating an action. The opposite may be true, as well.

In short, the Classic Event subsystem performs the following basic functions:

- Detects when events occur by comparing sets of criteria against historical data in the database.
- Optionally logs event records to a dedicated SQL server table (EventHistory).
- Optionally triggers a configured action each time an event is occurs.

For information about configuring events within the Classic Events Subsystem, see the Configuring Classic Events.

The Classic Event subsystem does not support Daylight Savings Time changes. The Replication subsystem, however, does handle Daylight Savings Time changes, and you can use replication to generate data summaries according to a schedule. For more information, see Managing and Configuring Replication.

Classic Event Subsystem Components

The following table describes the components of the Classic Event subsystem.

Component	Description
Configuration Editor	Part of the Operations Control Management Console. Used to specify event definitions and possible actions.
Runtime database	Stores event definition information and all data generated by the Event subsystem, such as records of event detections, data summaries, and data snapshots.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Component	Description
Event System Service (aahEventSvc.exe)	Internal process that coordinates event detection and action functions. This process runs as a Windows service. Using the Operations Control Management Console, you can configure the event service to automatically start and stop at the same time as the AVEVA Historian. The event service is responsible for: Reading event definition information from the Runtime database. Creating event detectors and actions, including allocating the necessary processing threads and establishing database connections. Initiating the event detection cycle.
SQL variables	Available for use in event queries.

You use the Operations Control Management Console to configure the Classic Event subsystem.

Uses for the Classic Event Subsystem

Generally, you should use the AVEVA Historian Classic Event subsystem to monitor non-critical system conditions that occur only occasionally. For example, possible event detections that you can set up include:

- Detect all occurrences in history when the value of a discrete tag is set to 0
- Detect if the system clock is set to a specified date and/or time
- Determine the state of information in the database by a SQL statement

You can use event actions to perform tasks such as the following:

- Send e-mail messages to remind managers about weekly maintenance checks
- Summarize plant data to create a statistical analysis over defined periods of time
- Take "snapshots" of system data
- Modify storage conditions (such as time and value deadbands)
- Generally perform any database-related task

The Classic Event subsystem is not designed to transfer data to and from the database continually and should not be used in this manner. The only exception is for summary actions; the Classic Event subsystem can continually process data aggregates so that they are available for reporting purposes.

The Classic Event subsystem should not be used as an alarm system. An alarm system such as provided with InTouch HMI software can be used to alert operators to specific satisfied conditions. The InTouch alarm system is intended as a notification system to inform operators of process and system conditions promptly upon their occurrence. The InTouch alarm system supports displaying, logging, and printing capabilities for process alarms and system events. (Alarms represent warnings of process conditions, while events represent normal system status messages.) For more information on the InTouch alarm system, see your InTouch documentation.

In contrast, the Classic Event subsystem is intended to initiate actions based upon historical event detection. An

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

alarm system presupposes an immediate message response is propagated for all configured alarms at the time the respective conditions are met. In this sense, the historian Classic Event subsystem is not an alarm system. The Classic Event subsystem queues up detected events and processes them accordingly based upon preconfigured priorities.

Classic Event Subsystem Features and Benefits

You can obtain a number of distinct operational benefits from properly using the features of the Classic Event subsystem. A list of key benefits is as follows:

- Unlike real-time alarming, the Classic Event subsystem determines events from stored historical data and is not dependent on real-time detection. No events are missed unless the machine is severely overloaded for a long period of time.
- The Classic Event subsystem is SQL-based, thus providing a means of managing database-related tasks within the system. You can use custom SQL queries as detectors, as well as create custom SQL-based actions.
- A number of preconfigured detectors and actions are available.
- Detections may be made by external sources. (A COM mechanism is available for invoking the detector in the Classic Event subsystem.)
- Time-based detection (based on the system clock time) allows you to schedule certain tasks, such as data aggregations (summaries).
- The Classic Event subsystem is designed to manage overload situations. If the system is currently busy due to some other processing for a period of time, the Classic Event subsystem will "catch up" at a later time during off-peak periods. If the overall AVEVA Historian is continuously overloaded, the Classic Event subsystem degrades in functionality gracefully.
- You can select which actions have priority and can assign certain actions (preferably only a few) never to be compromised, even under overload conditions.
- System tags are available to monitor Classic Event subsystem conditions.

Classic Event Subsystem Performance Factors

The overall performance of the AVEVA Historian Classic Event subsystem is subject to factors related to data storage and query processing time. Too often, systems are commissioned with specifications that estimate average or "typical" expected loading. Instead, you should size the system so that it can accommodate the peak load that you expect during the projected system life cycle. Some performance factors you should consider are:

- **Sufficient hardware.**
  Your selection of hardware is important to guarantee peak performance for the range of behaviors required for a given operating environment. For example, you should make sure that you have enough disk space to store the records of detected events and the results of any actions (summaries, value snapshots, and so on).
- **Processor availability.**
  The Classic Event subsystem is subject to processor availability as much as any other software sharing a common platform. At any given moment, multiple processes contend for processor time.
- **Nature of the database queries executed by the Classic Event subsystem.**
  For example, because Classic Event subsystem actions typically operate on normal SQL Server tables, they are subject to performance limitations of the Microsoft SQL Server. Also, query activity tends to be very CPU-intensive and is extremely sensitive to other concurrent activities being performed on the same server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Time intervals for SQL-based detectors.**
  For more information, see Time Intervals for SQL-Based Detectors.

Performance can vary greatly for the same event task, depending upon the computer configuration, user interaction, and other unpredictable activity common in a plant situation with shared database and server resources. It is often very difficult to determine precisely what combinations of hardware and software parameters will work optimally for your required operating environment. Therefore, you should test your Classic Event subsystem configuration before running it in a production environment to make sure that the system will not become overloaded during peak use.

Event Tags

An event tag is a name for an event definition in the system. For example, if you want to detect an event when a tank temperature reaches 100 degrees, you can define an event tag and name it "TankAt100." Event tags differ from the other tag types in the AVEVA Historian (analog, discrete, and string). Analog, discrete, and string tag types are the definitions of variables to be stored. In contrast, an event tag is a named reference for the definition of the specific event you want to detect, including an optional action to perform when the event is detected. An event tag provides a way to reference all event definition information in the system.

Event tags are created and maintained using the Operations Control Management Console. When you define an event tag, you must specify:

- A name, description, and other general configuration information.
- The event criteria, which describes the conditions that must exist for the event and how often the Classic Event subsystem checks to see if an event occurred.
- Whether or not to log the event detection.
- Whether or not to enable or disable event detection.
- An optional action that is triggered when an event is detected.

Event Detectors

Each event tag must have an associated event detector. An event detector is a mechanism for determining when the set of event criteria for an event tag has been satisfied. When you configure an event detector, you must first configure its type and then configure the parameters associated with that detector type. You can choose from the following types of event detectors:

- SQL-Based Detectors
- Schedule Detectors
- External Detectors

The generic SQL, analog specific value, and discrete specific value detectors are SQL-based detectors. The schedule detector is a time-based detector. The external detector is used when triggering an event by the ActiveEvent ActiveX control.

For all detectors, the Classic Event subsystem will initially base the query for data in history at the time the Classic Event subsystem starts. Subsequently, the Classic Event subsystem will base the query on the last successful detection; that is, the time of the most recent detection becomes the starting time for the next detection.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Analog specific value, discrete specific value, and generic SQL detectors operate on data stored in the database. The detection criteria for each of these detectors is a SQL statement that is executed against the AVEVA Historian. Generic SQL detectors can query against both the historian and Microsoft SQL Server.

Generic SQL Detectors

A generic SQL detector detects an event based on criteria that are specified in a SQL statement. You can use pre-configured SQL templates that are stored in the database as the basis for your script, or you can create your own script from scratch.

To use a pre-configured SQL template, simply select it from a list of available templates when defining the event tag.

If you create a new script, you will need to add it to the SQLTemplates table in the Runtime database in order for it to appear in the list of pre-configured templates. You should test your SQL queries in SQL Server Query Analyzer before using them in a generic SQL event detector.

Specific Value Detectors

Two specific value detectors are available:

- Analog specific value detector
- Discrete specific value detector

These detectors can be used to detect if a historical tag value matches the state defined by the detector criteria. For the criteria, historical values are compared to a target value that you specify. If a value matches the criteria, then an event is logged into the EventHistory table, and any associated actions will be triggered. For example, an analog specific value detector could be configured to detect if the value of 'MyAnalogTag' was ever greater than 1500. Likewise, a discrete value detector could be configured to detect if the value of 'MyDiscreteTag' was ever equal to 0.

For a specific value detectors, you can apply either edge detection or a resolution to the returned data. The resolution is used only when the edge detection is set to NONE (in which case the retrieval mode is cyclic). For more information, see these topics from the *AVEVA Historian Retrieval Guide*:

- Resolution (Values Spaced Every X ms) (wwResolution)
- Edge Detection for Events (wwEdgeDetection)

Time Intervals for SQL-Based Detectors

For SQL-based detectors, you must specify a time interval that indicates how often the detector will execute. The time interval is very important in that it affects both the response rate of any event actions and the overall performance of the system.

The detection of an event may occur significantly later than the actual time that the event occurred, depending

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

on the value you specify for the time interval. The time between when an event actually occurred in history and when it was detected is called latency.

For example, you configure a detector to detect a particular event based on a time interval of 10,000 ms (10 seconds). This means that every 10 seconds, the event detector will check to see if the event occurred. If the event occurs 2,000 ms (2 sec) after the last check, the event detector will not detect that the event occurred until the full 10 seconds has elapsed. Thus, if you want a greater possibility of detecting an event sooner, you should set the time interval to a lower value.

Also, the time interval affects when an associated action will occur, because there could be some actions that are queued to a time equal to or greater than the interval.

The following are recommendations for assigning time intervals:

- When configuring multiple event detectors, distribute them evenly across multiple time intervals; don't assign them all to the same interval.

  All configured detectors are first divided into groups, based on their assigned time interval. The detectors are then sequentially ordered for processing in the time interval group. The more detectors assigned to a particular time interval, the longer it will take the system to finally process the last one in the group. While this should not have a negative impact on actual detection of events, it may add to increased latency.
- Avoid assigning a faster time interval than is really necessary.

  The time interval for detectors should not be confused with a rate required by a real-time system that needs to sample and catch the changes. For the Classic Event subsystem, a slower time interval simply means that more rows are returned for each scan of the history data; no events are lost unless then detection window is exceeded (for more information, see "Detector overloads" on page 370). For example, you create an event tag with a detector time interval of 1 minute, and you expect an event to occur every 5 seconds. This means that the system would detect 12 events at each time interval. In most cases, this is an acceptable rate of detection. Also, assigning short time intervals will result in higher CPU loading and may lead to degraded performance.

For detailed information on how detectors are executed, see Classic Event Subsystem Resource Management.

The EventHistory table can be used to determine if too many event tags have the same time interval. If the latency between when the event actually occurs (stored in the DateTime column) and when it was detected (stored in the DetectDateTime column) is constantly growing and/or multiple event occurrences are being detected during the same detector time interval, you need to move some of the event detectors to a different time interval.

Schedule Detectors

The schedule detector is a time-based detector. A schedule detector detects whether the system clock is equal to or greater than a specific date and/or time. For example, you could log an event every week on Monday at 2:00 p.m.

Schedule detectors are different from other detectors in that they are real-time detectors. The value of the system clock is checked every second. Schedule detectors are very fast and can be used without great concern about efficiency. Thus, a schedule detector provides the only real-time event processing. However, there is no guarantee of when the action will occur.

All of the schedule detectors that you set up are handled by a dedicated scheduling thread. This allows for a separation between the processing load needed to execute schedule detectors and the processing load needed to perform all of the other event work. The scheduling thread will maintain a list of detection times in a time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

queue. If you add a schedule detector, the thread will register the detection time in the queue and then re-sort the list of all detection times from the earliest to the latest.

The time of the system clock is then compared with the time of the first item in the schedule queue. If the system clock time is equal to or greater than the time of the first item, the detection algorithm for the first item will be invoked and the detection will be performed.

The Classic Event subsystem does not account for Daylight Savings Time changes. If you set up a schedule detector that runs periodically with a specified start time, you will need to change the start time to reflect the time change. Another solution would be to use the time-weighted average retrieval mode instead of the Classic Event subsystem to generate averages, because the retrieval mode handles the Daylight Savings Time changes. However, if the period for the average is hourly, then it is recommended that you use the Classic Event subsystem, as the amount of data will not generally not be a factor in the speed of calculating the average.

External Detectors

For an external detector, event detection is triggered from an external source by the ActiveEvent ActiveX control that is provided as part of the AVEVA Historian. For example, an InTouch or Visual Basic script can invoke the necessary ActiveEvent methods to trigger an event. This ActiveX control must be installed on the computer from which you want to trigger the external event.

For more information, see Configuring an External Detector.

Event Actions

An event may or may not be associated with an event action. An event action is triggered after the event detector determines that the event has occurred. The Classic Event subsystem is not intended to run external processes. There is only a very limited ability to run external program files or to call methods from COM interfaces within the given system or network.

Actions are not required; there are times when you may want to simply store when events happened. In this case, you would select "None" for the action type when defining the event tag.

Generic SQL Actions

A generic SQL action executes an action that is outlined in a SQL statement. For example, a SQL action can update the database (for example, turning off storage for tags) or copy data to a separate table or database.

You can use pre-configured SQL templates that are stored in the database as the basis for your script, or you can create your own script entirely from scratch. You cannot submit multiple queries against the AVEVA Historian in a single event action and you cannot use GO statements. Also, if you are querying against history data, the SQL statement is subject to the syntax supported by the AVEVA Historian OLE DB provider. You should test your SQL queries in SQL Server Query Analyzer before using them in a generic SQL event action.

Snapshot Actions

A snapshot action logs into dedicated SQL Server tables the data values for selected analog, discrete, or string tags that have the same timestamp as the detected event. Quality is also logged. Value snapshots are stored in tables according to the tag type, either AnalogSnapshot, DiscreteSnapshot, or StringSnapshot.

A snapshot action requires an expensive SQL join between the extension tables and the snapshot tag table. The process of performing the join and logging the retrieved results to the snapshot tables can be very slow. This is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

because most of the tables used for event snapshots are normal SQL Server tables, subject to the data processing limitations of Microsoft SQL Server. Thus, the higher the number of snapshots that are being taken by the event system, the higher the transaction load on the Microsoft SQL Server.

Important: The Classic Event subsystem is not a data acquisition system. DO NOT attempt to use snapshot actions to move data stored in the extension tables to normal SQL Server tables. This type of misapplication is guaranteed to result in exceedingly poor throughput and storage rates.

When trying to determine how many snapshots can be made by the system, you should execute the intended snapshot queries to the server using a batch file, leaving the Classic Event subsystem out of the exercise. By executing repeated snapshot queries at the server as fast as the computer will allow, you can better determine how many snapshots can be performed on a system over a given time period. Using this result and applying a safety factor may provide a good guideline for assessing how much your system can safely handle. Keep in mind that discrete snapshots are many times slower than analog snapshots.

E-mail Actions

An e-mail action sends a pre-configured Microsoft Exchange e-mail message. Although e-mail actions are useful for sending non-critical messages triggered by an event detection, these types of actions are not to be used for alarm-type functionality. For e-mail notifications of alarm situations, use an alarm system such as the SCADAlarm alarm notification software.

Deadband Actions

Important: Deadband actions are no longer supported. Any configured deadband actions are ignored.

A deadband action changes the time and/or value storage deadband for one or more tags that are using delta storage. (Value deadbands only apply to analog tags.) Deadband change actions are useful for increasing data storage based on an event occurring. For example, an event detector has detected that a boiler has tripped, you might want to start saving the values of certain tags at a higher rate to help you determine the cause of the trip.

Summary Actions

A summary action is a set of aggregation calculations to be performed on a set of tags between a start time and an end time with a defined resolution. When you configure a summary action, you must define the type of aggregation you want to perform (called a summary operation) and the analog tags that you want to be summarized. The Classic Event subsystem performs average, minimum, maximum and sum calculations on the basis of a specific event being detected.

Note: Summary actions using the Classic Event subsystem are retained for backward compatibility. We recommend that you use the more robust and flexible Replication subsystem to perform data summaries. For more information, see Managing and Configuring Replication.

Data summaries are useful for:

- Extremely long-term data storage. Because summarized data takes up less space than full resolution data, even a moderately sized system can store daily summary information for many years.
- Production reporting. For many reporting purposes, aggregate data is more important than raw data. For example, the total mass produced in a day is often more relevant than the actual rate of production during the day.
- Integration with business systems. The full resolution, high-performance AVEVA Historian history and real-

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

time data tables are best accessed with tools that can take advantage of the AVEVA Historian time domain extensions. However, not all client tools support these SQL extensions. The summary tables reduce the volumes of data to manageable quantities that can be used by any normal SQL client application.

A summary action is usually triggered by a schedule detector. However, you can perform a summary as a result of any event detection.

Tag values with bad quality are not filtered out before the aggregation is performed. To perform an aggregation with only good quality, for example, use a generic SQL action that executes an aggregation calculation query on the History table where the value of the Quality column equals 0.

The results of all summaries are stored in the SummaryData table in the Runtime database.

Important: Use caution when setting up summary actions. Using a high resolution for your summary queries can have a negative impact on the overall performance of the system.

Average, minimum, and maximum values can also be determined by using the time-weighted average, minimum, and maximum retrieval modes, respectively. For more information on these retrieval modes, see Understanding Retrieval Modes in the *AVEVA Historian Retrieval Guide*. Keep the following in mind when deciding to use either the event summaries or the retrieval modes:

- For the time-weighted average retrieval mode, the amount of time between the data values is a factor in the average, whereas the event summary action is a straight statistical average. For more information, see Average Retrieval.
- Performing an average at retrieval eliminates problems that occur during Daylight Savings Time adjustments for schedule-based summaries. For more information, see Schedule Detectors.

For a comparison of all the different types of summaries that the AVEVA Historian supports, see Querying Aggregate Data in Different Ways in the *AVEVA Historian Retrieval Guide.*

Event Action Priorities

The Classic Event subsystem contains three different queues for event actions:

- **Critical queue**
  A critical queue contains any actions for event tags that have been assigned a critical priority. Actions for events that are given a critical priority will be processed first. It is extremely important that the critical queue is used with caution. Only singularly important actions with short processing times should be assigned as critical. You should never assign snapshot or summary actions as critical. There is no overload protection for processing critical actions; if the system becomes overloaded, actions may not execute in a timely fashion or may not execute at all.
- **Normal queue**
  This type of queue contains any actions for event tags that have been assigned a normal priority. All non-critical events are labeled with a "normal" priority and will be processed after the critical events.
- **Delayed queue**
  This type of queue contains any actions for event tags that have been assigned a post-detector delay. The post detector delay is the minimum amount of time that must elapse after an event was detected before the associated action can be executed.

Classic Event Subsystem Resource Management

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

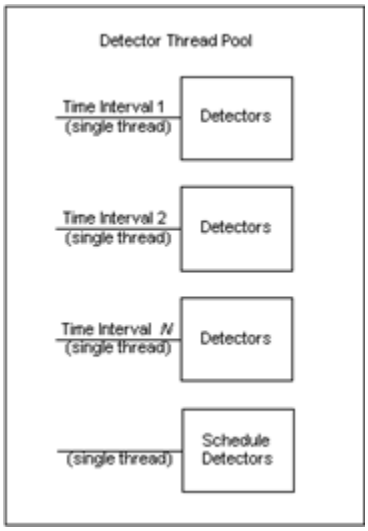
The Event System Service (aahEventSvc.exe) manages all of the system resources required to detect events and process actions. System resources are allocated for detectors and actions by means of threads. A thread is an operating system component that independently performs a particular function within a larger process. Within the overall process of the Classic Event subsystem, event detectors and actions are assigned different threads, so that they can execute independently of each other and thus perform more efficiently.

The Classic Event subsystem uses two thread groups, or "pools." One thread pool is for detectors and the other one is for actions. The Event Service automatically creates both of these thread pools if there is at least one event tag defined.

Other aspects of resource management include the number of database connections required by event system components, and how the system handles event overloads and query failures.

Detector Thread Pooling

The detector thread pool is made up of one or more threads allocated for SQL-based detectors and a single thread for schedule detectors. Each thread maintains a connection to the database. The detector thread pool is illustrated in the following diagram:



A SQL-based detector is assigned to a thread based on the time interval that you specify when you define the event tag. Each time interval requires its own thread. For example, you define three event detectors and assign them time intervals of 10, 15, and 20 seconds, respectively. Each event detector will be running in its own thread, for a total of three threads.

As another example, you define three event detectors, assigning the first two a 10 second interval, and the third a 15 second interval. The first two will be running under the same thread, while the third will be running under its own thread, for a total of two threads.

For multiple detectors that are assigned to the same time interval, the SQL detection statement for each event tag will be executed in sequential order. That is, the first SQL statement must return results before the next statement can be executed. After each detection has taken place (results are returned), the detection is logged into the EventHistory table and any associated action is queued into the action thread pool.

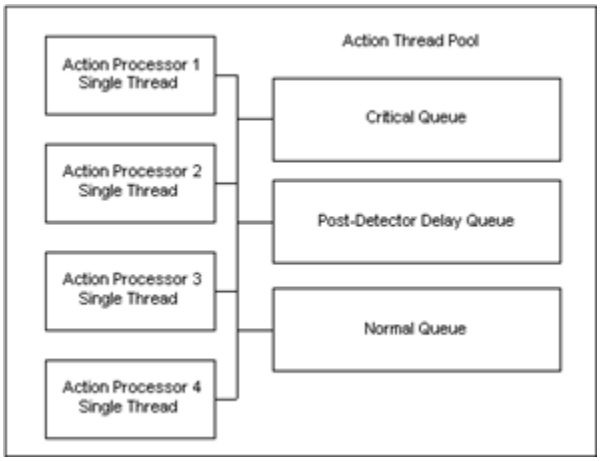
All schedule detectors are assigned to a single thread.

The efficiency of the detector thread pool depends on how you have spread the load when assigning time intervals to different event tags. Detections generally do not cause overloading on the system: the actions (especially snapshots and summaries) are where most processing and resource loading occurs.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Action Thread Pooling

The action thread pool is essentially a pool of four threads that execute actions from three different action queues. Each thread in the pool maintains a database connection.



The three action queues are:

- Critical queue
- Normal queue
- Post-detector delay queue

For detailed information about each of these queues, see Event Action Priorities.

As a processor thread completes its previous task, a new action will be fetched from one of the queues. If there are any actions in the critical queue, these will be processed first. Actions in the critical queue are executed in the order in which they were added to the queue; that is, the oldest action sitting in the queue will be processed first.

If the critical queue is empty, actions will be fetched from the post-detector delay queue. Actions in the post-detector delay queue are ordered by time. Actions assigned the shortest post-detector delay will be executed first.

If both the critical and post-detector delay queues are empty, actions will be fetched from the normal queue. Like critical actions, normal actions are processed in the order in which they were added to the queue.

Classic Event Subsystem Database Connections

The following table contains the number of SQL Server database connections required by the different components of the Classic Event subsystem.

Component	Number of Connections Used
Event Service	1
SQL-based detectors	1 per each time interval used
Schedule detectors	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Component	Number of Connections Used
Action threads	4

Handling of Event Overloads and Failed Queries

The Classic Event subsystem handles SQL-based detector and action queries that fail, as well as to degrade gracefully if detector and action overload conditions occur.

- **Event query failures**
  If the query for a SQL-based detector fails, the query will automatically be executed again. The detection window start time will remain the same until the next detection is made.

  For a failed SQL-based action query, the query will be submitted three times. The system will establish a new connection to the database each time the query executes. If the action query is a snapshot query, the snapshot tables will first be "cleaned up" as part of the re-query process.
- **Detector overloads**
  A detector overload occurs when the system cannot process all of the detectors in a timely manner. Detector overload is handled by means of the detection window. This window is defined by the difference between the current system time and the time of the last detection. If the window grows larger than one hour, some detections will be missed. This condition will be reported in the error log.
- **Action overloads**
  An action overload occurs when the system cannot process all of the actions in a timely manner. Only actions assigned a normal priority have overload protection. An action will not be loaded into the normal queue by a detector if the earliest action currently sitting in the queue has been there for an hour. (Basically, it is assumed that the system has become so overloaded that it has not had the resources to process a single action in the past hour.) This prevents an accumulation of actions in the normal queue when the system is unable to process them. The system will be allowed time to recover, and actions will not start to be queued again until the time difference between earliest and latest action in the queue is less than 45 minutes (75 percent of the time limit). In short, when the system becomes too overloaded, actions are not queued. This condition is reported in the error log, but not for every single action missed. The first one missed is reported, and thereafter, every hundredth missed action will be logged.

  There is no overload protection for critical actions, because these types of actions should only be configured for a very small number of critical events. There is also no overload protection for actions that have been assigned a post-detector delay.

For more information on action priorities, see Event Action Priorities. For more information on how actions are queued, see Action Thread Pooling.

Classic Event Subsystem Variables

The Classic Event subsystem uses a set of variables to facilitate event detections and actions. The purpose of these variables is to provide ease of query creation by a user (or a configuration editor). These variables are replaced with the associated values by the event components immediately before actual query execution. The query actually being received by the AVEVA Historian never contains the variables.

The variables and their associated values are as follows:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Variable	Description/Associated Value
@EventTime	Date/time of the detected event of the current detector.
@EventTagName	Tagname associated with the detected event.
@StartTime	Start date/time for the detector query.
@EndTime	End date/time for the detector query.

The @StartTime and @EndTime variables can be used only in detector strings. The @EventTime and @EventTagName variables can be used only in action strings.

All of the variables are case-sensitive.

Typically, a detection query executed by a detector component is similar to the following example:

```
SELECT DateTime
   FROM History
      WHERE Tagname = 'BoilerPressure' AND Value > 75
      AND DateTime > '@StartTime'
      AND DateTime < '@EndTime'
```

@StartTime and @EndTime are simply placeholders for the detector component to coordinate event detection over a moving time range.

The following action query show how event variables can be used:

```
SELECT * INTO TEMPTABLE
   FROM History
      WHERE DateTime = '@EventTime'
      AND TagName IN
      (SELECT TagName FROM SnapshotTag
      WHERE EventTagName = '@EventTagName'
      AND TagType = 1)
```

Note: These variables only function in the internal context of the Classic Event subsystem and do not apply to queries from client tools such as SQL Server Query Analyzer.

Classic Event Subsystem Tags

The following table describes the Classic Event subsystem tags.

TagName	Description
SysEventCritActionQSize	Size of the critical action queue.
SysEventDelayedActionQSize	Number of entries in the delayed action queue.
SysEventNormActionQSize	Size of the normal action queue.
SysEventSystem	A discrete tag that indicates the status of the event system service (aahEventSvc.exe). 0 = Bad; 1 = Good.
SysStatusEvent	Snapshot event tag whose value changes every hour.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring Classic Events

You can use the Classic Event subsystem to set up the detection of events and associate actions with those events. At a basic level, anything that can be determined by looking at historical or system data can be used as an event. The Classic Event subsystem stores data to SQL Server.

Important: The Classic Event subsystem is not a real-time system; rather, it operates on historical data. For real-time alarming, use an application such as the InTouch HMI.

When setting up an event, you must provide the following information:

- The criteria for the event. For example, the value of an analog tag being equal to 1500 could be an event. Also, the system clock on the AVEVA Historian computer reaching 9:30 a.m. on Monday morning could be an event.
- How often you want the Classic Event subsystem to check if an event occurred. This is called event detection.
- Whether or not you want information about the event detection saved to the database.
- Whether or not you want to execute an action as a result of a successful event detection and the type of action. For example, send an e-mail message.

The configuration information for the detection and action for a particular event must be given a unique name, which is stored as an event tag.

The Historian also supports the storage of alarms and events generated through external sources such as Application Server to history blocks.

Accessing Event Information

To access event information

1. In the console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Click **Event Tags**. All configured event tags appear in the details pane.

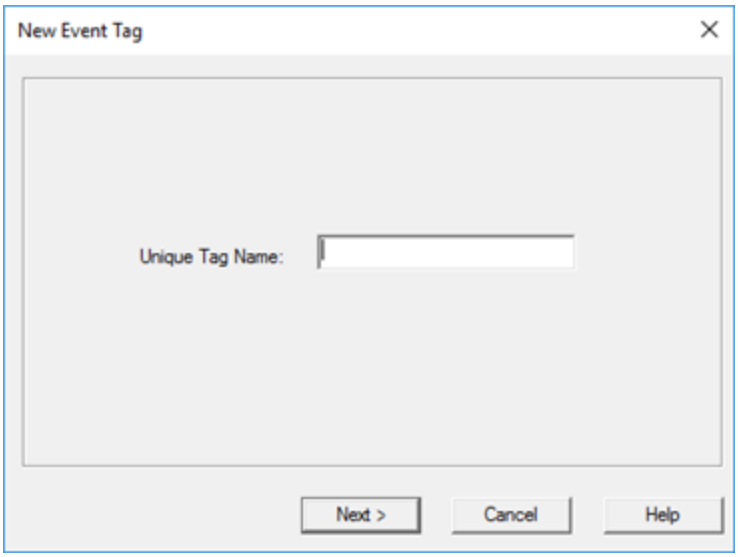
Adding an Event Tag

To add an event tag

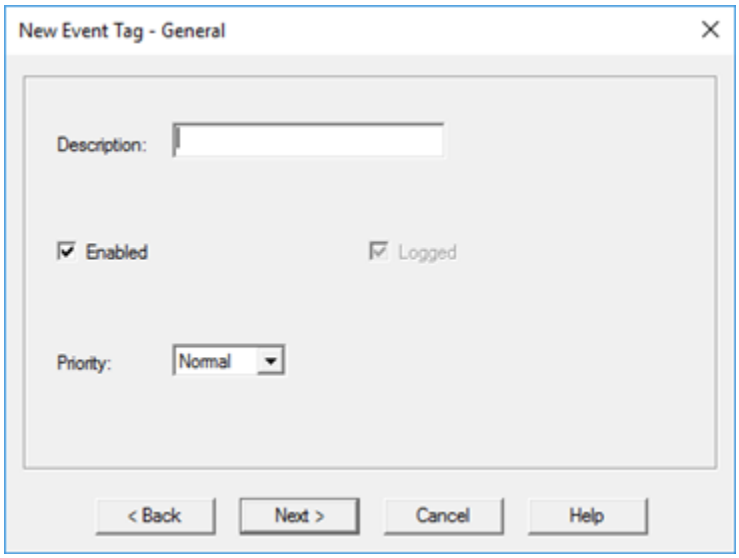
1. In the console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Click **Event Tags**.
4. Start the Event Tag wizard by doing any of the following:
   - Click the **Add** button + on the toolbar.
   - On the **Action** menu, click **New Tag**.
   - Right-click **Event Tags**, and then click **New Tag**.

     The **New Event Tag** wizard appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

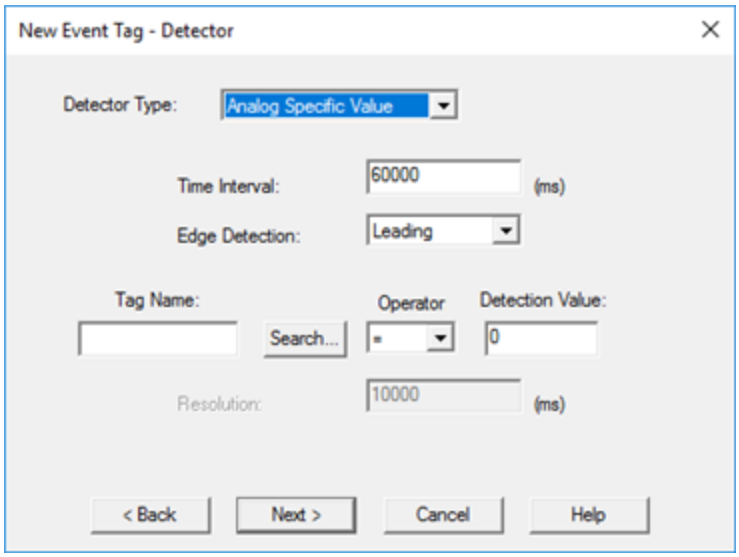


5. In the **Unique Tag Name** box, type a unique name for the event tag. For information on allowable tagnames, see Tag Naming Conventions.
6. Click **Next**. You are prompted to define general information for the event tag.



7. Configure the general options for the event tag. For more information, see Editing General Information for an Event Tag.
8. Click **Next**. You are prompted to configure the detector for the event tag.

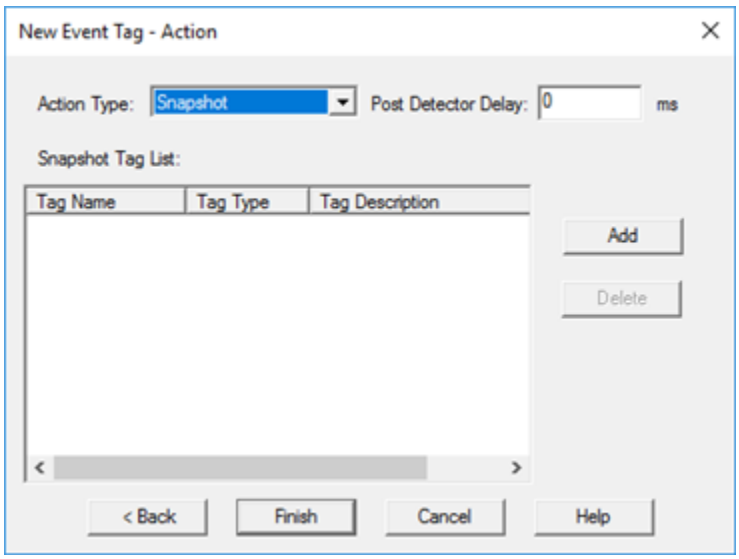
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



9. Configure the detector for the event tag. Detectors are external, generic SQL, analog specific value, discrete specific value, and schedule. The lower portion of the dialog box changes based on the detector type that you select.

For more information about...	See...
Configuring an external detector	Configuring an External Detector
Configuring an analog or discrete specific value detector	Configuring a Specific Value Detector
Configuring a schedule detector	Configuring a Schedule Detector
Configuring a generic SQL detector	Configuring a Generic SQL Detector

10. Click **Next**.
11. You are prompted to configure the action for the event tag.



12. Configure the action for the event tag. Actions are none, generic SQL, snapshot, e-mail, deadband, and

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

summary. The lower portion of the dialog box changes based on the action type that you select.

For more information about...	See...
Configuring a generic SQL action	Configuring a Generic SQL Action
Configuring a snapshot action	Configuring a Snapshot Action
Configuring an e-mail action	Configuring an E-mail Action
Configuring a deadband action	Configuring a Deadband Action
Configuring a summary action	Configuring a Summary Action

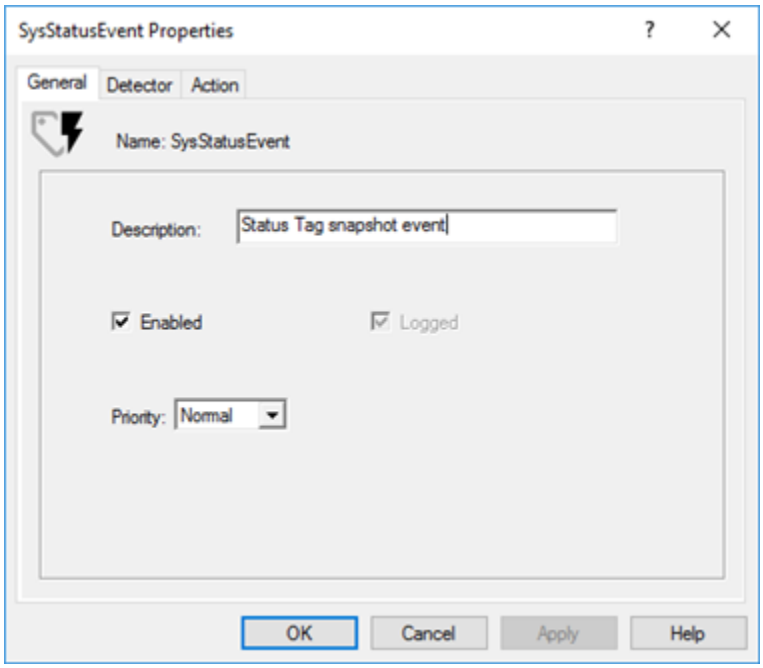
13. Click **Finish**.

Editing General Information for an Event Tag

General information for an event tag includes information about the tag definition. Event detectors and actions are defined separately and then associated with an event tag.

To edit general information for an event tag

1. In the console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Click **Event Tags**.
4. In the details pane, double-click the event tag to edit. The **Properties** dialog box appears.
5. Click the **General** property tab.



6. In the **Description** box, type a description of the tag.
7. Click **Enabled** to allow the detector and action for this event tag to run.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. Click **Logged** to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured.
9. In the **Priority** list, select a priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. For more information, see Event Action Priorities.
10. Click **OK**.

Configuring Detectors

You can set up the following types of detectors:

- Analog specific value
- Discrete specific value
- Schedule
- Generic SQL
- External

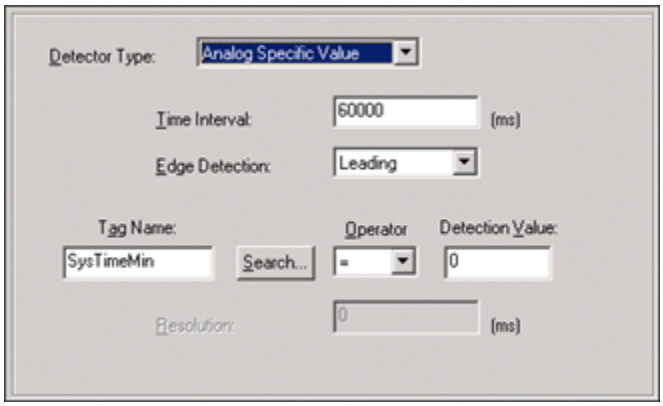
Note: If you change an event tag from using any SQL based detector to a time based detector, or vice versa, stop and restart the Event system. Or, delete the existing event tag and recreate it using the desired detector.

Configuring a Specific Value Detector

The configuration is basically the same for analog and discrete specific value detectors, with only a few small differences.

To configure a specific value detector

1. In the **Detector Type** list, select **Analog Specific Value** or **Discrete Specific Value**.



2. In the **Time Interval** box, type the interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).

   Be careful when assigning time intervals to event tags. For more information, see Time Intervals for SQL-Based Detectors .

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. In the **Edge Detection** list, select the "edge" for the event detection.

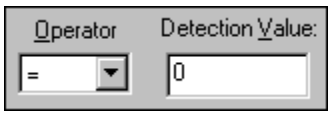
   A leading edge detection returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false). A trailing edge detection returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true). For an edge detection of "both," all rows satisfying both the leading and trailing conditions are returned.

   For more information, see Edge Detection for Events (wwEdgeDetection) .

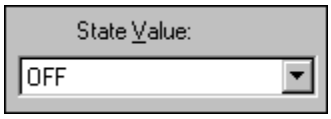
4. In the **Tag Name** box, type the name of the tag to which the event criteria will be applied. To search the database for a tag, click **Search**. The **Tag Finder** dialog box appears, in which you can query the database for tags. For more information, see Using the Tag Finder.

5. Set the value criteria for the tag.

   If you are configuring an analog specific value detector, in the **Operator** box, select an operator for the criteria. Then, in the **Detection Value** box, type a value against which the stored values for the tag are compared to determine if the event occurred.

   If you are configuring a discrete specific value detector, in the **State Valu**e list, select the target state of the discrete tag that causes the event to occur.

6. If you selected **None** in **Edge Detection** list, you can specify a resolution for the data. In the **Resolution** box, type a sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.

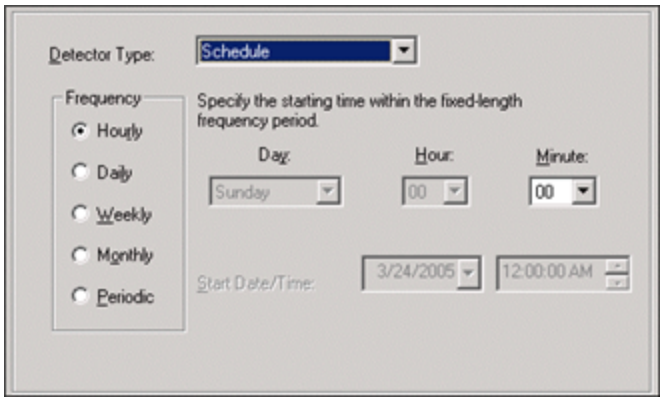
Configuring a Schedule Detector

This feature is included only for backward compatibility. You should instead use summary tags and replication. For more information on configuring replication, see Managing and Configuring Replication.

To configure a schedule detector

1. In the **Detector Type** list, select **Schedule**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. In the **Frequency** area, select how often you want the event to occur.

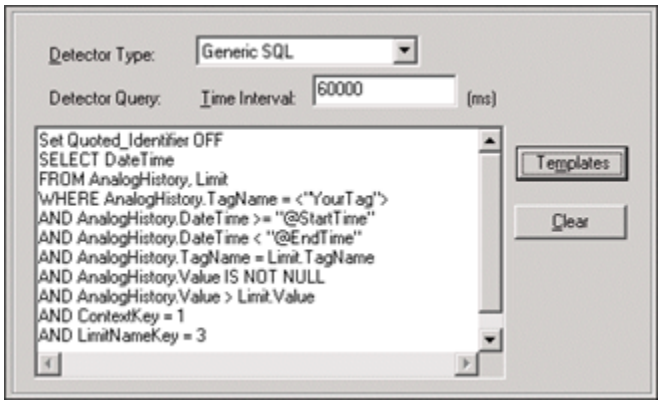
   When you select a frequency, different options to the right of the **Frequency** group become available.
3. Configure the time specific for the selected frequency.

Configuring a Generic SQL Detector

The AVEVA Historian does not validate SQL query syntax. First, test the SQL query using a tool such as Microsoft SQL Server Query Analyzer.

To configure a generic SQL detector

1. In the **Detector Type** list, select **Generic SQL**.



2. In the **Time Interval** box, type the interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).

   Be careful assigning time intervals to event tags. For more information, see Time Intervals for SQL-Based Detectors.
3. In the **Detector Query** window, enter the ad-hoc query that detects the event. To open a list of SQL templates to use for your query, click **Templates**.
4. To clear the window, click **Clear**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring an External Detector

An external detector is triggered using the AVEVA Historian ActiveEvent control. The detector is a COM component and has an external interface. An InTouch or Visual Basic script can trigger a historian event by using the ActiveEvent methods, which are similar to functions. Using the InvokeEventEx() method causes an external event to be detected within the Event subsystem.

After you select "External" as your detector type, you need to configure the security attributes for the ActiveEvent control and write the script that invokes the event. For more information, see Using ActiveEvent.

Configuring Actions

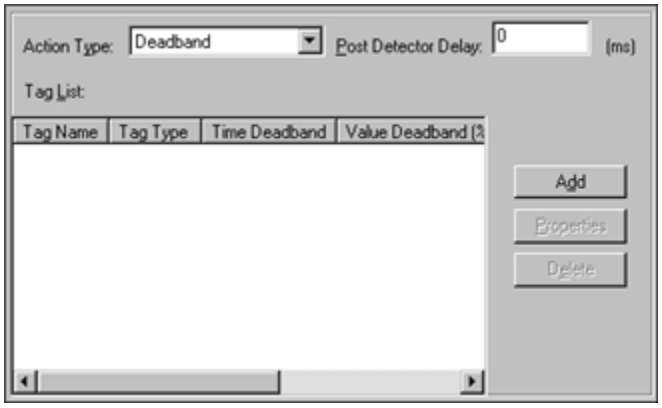
You can set up the following types of actions: deadband, snapshot, generic SQL, e-mail, and summary.

Configuring a Deadband Action

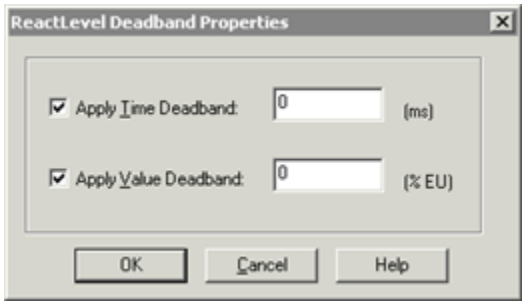
Important: Deadband actions are no longer supported. Any configured deadband actions are ignored.

To configure a deadband action

1. In the **Action Type** list, select **Deadband**.



2. To add one or more tags for which to set a new deadband, click **Add**. The **Tag Finder** dialog box appears, in which you can query the database for tags. For more information, see Using the Tag Finder.Select a tag in the **Tag List** list, and then click **Properties**. The **Deadband Properties** dialog box appears.



3. Configure the appropriate deadbands for the tag.

   Apply Time Deadband

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

Apply Value Deadband

The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. The value deadband applies only to analog tags.

4. Click **OK**. The changes appear in the **Tag List** window.
5. To delete a tag from the **Tag List** window, select the tag and then click **Delete**.
6. (Optional) In the **Post Detector Delay** box, type the amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.

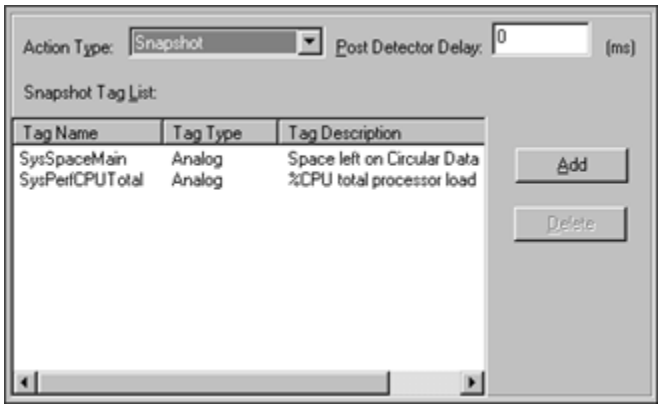
Note: If the tag list contains a tag that is deleted from the Runtime database, then the word "Deleted" appears as the tag type for the tag.

Configuring a Snapshot Action

A snapshot action records the values of a selected mix of analog, discrete, and string tags at the time that the event occurred.

To configure a snapshot action

1. In the **Action Type** list, select **Snapshot**.



All tags included in the snapshot are listed in the **Snapshot Tag List** list. Snapshots can include analog, discrete, and string tags.

2. To add one or more tags, click **Add**. The **Tag Finder** dialog box appears, in which you can query the database for tags. For more information, see Using the Tag Finder.
3. To delete a tag, select the tag in the **Snapshot Tag List** list and then click **Delete**.
4. (Optional) In the **Post Detector Delay** box, type the amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.

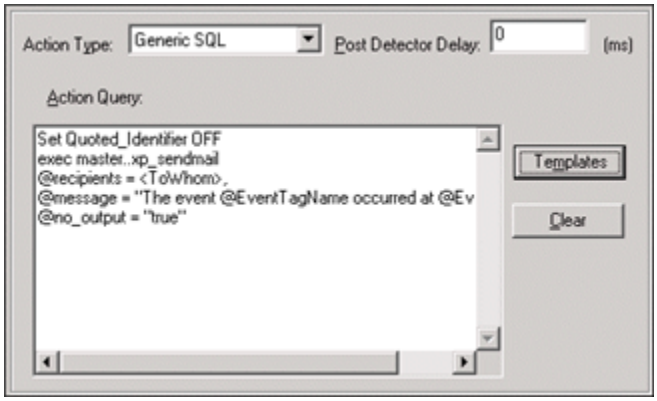
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring a Generic SQL Action

The AVEVA Historian does not validate the SQL query syntax. First test the SQL query using a tool such as Microsoft SQL Server Query Analyzer.

To configure a generic SQL action

1. In the **Action Type** list, select **Generic SQL**.



2. In the Action Query window, enter an ad-hoc query that detects the event. To access a list of SQL templates to use for your query, click **Templates**.

   For information on using event system variables in your query, see Classic Event Subsystem.
3. To clear the window, click **Clear**.
4. (Optional) In the **Post Detector Delay** box, type the amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.

Generic SQL Action Template for Executing a Command

Generic SQL Action Template for Executing a Command

To configure a generic SQL statement that executes a command, select the "Invoke an External Application" option in the list of generic SQL action templates:

```
master..xp_cmdshell '<Your Command>', no_output
```

In the syntax, replace <Your Command> with the desired command. Be sure to enclose the command in single quotes. For example:

```
master..xp_cmdshell 'dir *.exe', no_output
```

The **xp_cmdshell** extended stored procedure does not start a Windows application. You can only execute simple DOS commands, batch files or executables (.EXEs) that do not display a user-interface window.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

You must have the correct permissions set for the **xp_cmdshell** extended stored procedure for it to run. For more information, see your Microsoft SQL Server documentation.

Generic SQL Action Templates for E-mail

Generic SQL Action Templates for E-mail

Note: To use the `sp_send_dbmail`, you must first configure a default e-mail profile or specify an explicit profile. Refer to the SQL Server documentation for specific steps on how to configure each one. If an e-mail message is not sent as expected, check the SQL Server Log for possible errors.

Users can use the following queries to get information about whether the e-mail profile is configured.

- To view configured profiles, use this query:

```
SELECT * FROM msdb.dbo.sysmail_profile
```

- To view configured e-mail accounts, use this query:

```
SELECT * FROM msdb.dbo.sysmail_account
```

- To view the link between the profile and the account, use this query:

```
SELECT * FROM msdb.dbo.sysmail_profileaccount
```

- To view the information about the profile (Is Default/Public), use this query:

```
SELECT * FROM msdb.dbo.sysmail_principalprofile
```

To configure a generic SQL statement that sends an e-mail message, select one of the "Send an E-mail message…" options (one with a query and one without) in the list of generic SQL action templates. For example:

```
master..sp_send_dbmail
@recipients = <ToWhom>,
@body = 'The event @EventTagName occured at @EventTime',
@query = <"Your query">,
@exclude_query_output = <exclude_query_output>
```

In the second line of the syntax, replace <ToWhom> with the e-mail display name or complete e-mail address of the intended recipient or recipients. Assign the e-mail message to the @message variable. Be sure to enclose the recipient(s) and the message in single quotes. For example:

```
master..sp_send_dbmail
@recipients = 'John Doe',
@body = 'Check this out',
@query = 'SELECT TagName, DateTime FROM EventHistory
   WHERE TagName = "SysStatusEvent"
      AND DateTime = (SELECT Max (DateTime)
   FROM EventHistory
   WHERE TagName = "SysStatusEvent")',
@exclude_query_output = <exclude_query_output>
```

You must have the Microsoft SQL Server and the SQL Mail component set up properly in order for the **sp_send_dbmail** extended stored procedure to work correctly.

For more information, see Configuring an E-mail Action.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuring an E-mail Action

An e-mail action sends a pre-configured e-mail message when an event occurs. For the event system to support e-mail actions, you must properly configure all of the following:

- SQL Server login
- Microsoft Outlook Mail Client
- SQL Mail functionality for Microsoft SQL Server
- E-mail action for the event system

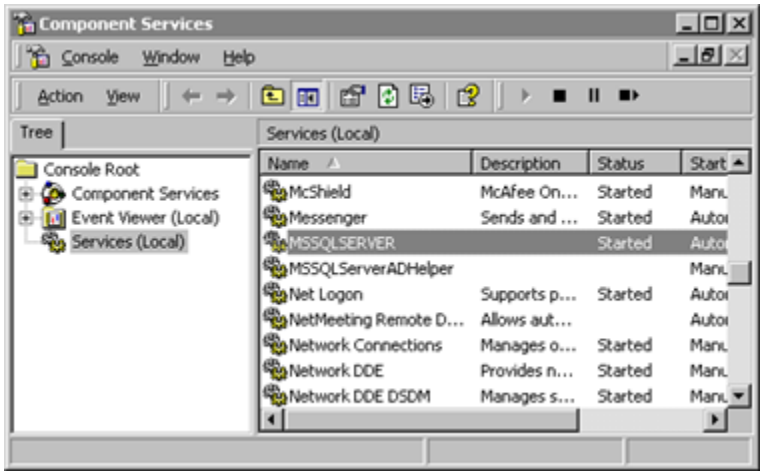
Note: The exact steps may vary depend on what version of the Windows operating system you are using.

Setting Up Microsoft SQL Server

Setting Up Microsoft SQL Server

For Microsoft SQL Mail to work correctly, Microsoft SQL Server must be configured to log on with a user account that has a valid MAPI mail profile. Perform these steps on the computer running the Microsoft SQL Server Service that will process the e-mail event.

1. On the Windows **Start** menu, point to **Programs**, point to **Administrative Tools**, and then click **Component Services**. The Component Services console appears.



2. In the console tree, click **Services**.
3. In the results pane, right-click on **MSSQLServer** and then click **Properties**. The **Properties** dialog box appears.
4. Click the **General** tab.
5. In the Startup Type list, click **Automatic**.
6. Click the **Log On** tab.
7. In the **Log On As** area, click **This account**. Enter the user account (domain or local) or click **Browse** to browse for a valid account. The user account you select needs to have a valid MAPI mail profile set up.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

8. For information on determining the MAPI mail profile, see Determining the Microsoft Outlook Mail Profile.
9. In the **Password** and **Confirm Password** boxes, enter the password for the user account.
10. Click **OK**.
11. Right-click the MSSQL Server service in the results pane, and then click **Stop** to stop the service.
12. After the MSSQL service stops, right-click the MSSQL Server service in the results pane, and then click **Start** to restart the service.
13. Close the Component Services console.

The next step in setting up an e-mail action is to determine the Microsoft Outlook profile. For more information, see Determining the Microsoft Outlook Mail Profile.

Determining the Microsoft Outlook Mail Profile

Determining the Microsoft Outlook Mail Profile

To properly set up the Microsoft SQL Server e-mail configuration, you need to know the name of the MAPI mail profile for the MSSQL Server service logon account. You must determine the MAPI profile name on the computer running MSSQL Server.

To determine the MAPI profile name

1. In Control Panel, double-click Mail.
2. Click **Show Profiles**.
3. Determine the MAPI mail profile.

   The mail profile is commonly set to "MS Exchange Settings" so that the use the user's profile on an Exchange Server is used.
4. Click **OK**.

   For more information on configuring user accounts for the Exchange Client, see your Microsoft documentation.

The next step in setting up an e-mail action is to configuring SQL mail in SQL Server Management Studio. For more information, see Setting Up SQL Mail in SQL Management Studio.

Setting Up SQL Mail in SQL Management Studio

Setting Up SQL Mail in SQL Management Studio

SQL mail is set up using the SQL Server Management Studio. Make sure that the Microsoft SQL Server is running. For more information on configuring mail profiles, see your Microsoft SQL Server documentation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

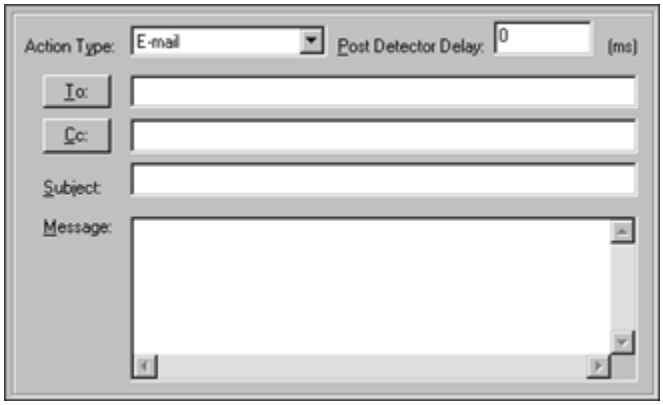
Configuring the E-mail Action

Configuring the E-mail Action

Note: Only Microsoft Outlook addresses can be used. Internet addresses are not directly supported.

To configure an e-mail action

1. In the **Action Type** list, select **E-mail**.



2. In the **To** line, enter the Outlook e-mail address of one or more persons to whom you want to send an e-mail message when an event occurs. You can also send a copy of the e-mail to one or more persons in the **Cc** line.

   You can access the address book of the e-mail account by clicking the **To** or **Cc** button.
3. In the **Subject** line, enter a synopsis of the e-mail. If you do not provide text for the subject line, "SQL Server Message" is used by default.
4. Enter the e-mail text in the **Message** window.
5. (Optional) In the **Post Detector Delay** box, type the amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.

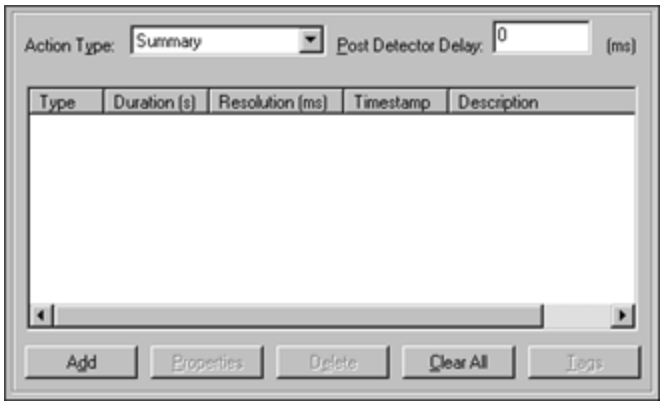
Configuring a Summary Action

If you only need scheduled data summaries, you should instead use summary tags and replication. However, if you want to trigger summaries based on events in history, you must use the event subsystem. For more information on configuring replication, see Managing and Configuring Replication.

To configure a summary action

1. In the **Action Type** list, select **Summary**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. To add a new summary operation, click **Add** and define the operation. For more information, see Adding a Summary Operation.
3. To assign analog or discrete tags to a summary operation, select the summary operation in the list and then click **Tags**. For more information on adding a summary tag, see Assigning a Tag to a Summary Operation.
4. To delete a summary action, select the summary operation in the window and then click **Delete**.
5. To clear the window of all summary operations, click **Clear All**.
6. To modify a summary action, select the summary operation in the window and then click **Properties**. For more information on the dialog box that appears, see Adding a Summary Operation.

   If you modify a summary operation, you may see inconsistencies between old summary data and new summary data. You can not save the modified summary operation if its criteria is identical to an existing summary operation associated with the current event tag.
7. (Optional) In the **Post Detector Delay** box, type the amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.

Adding a Summary Operation

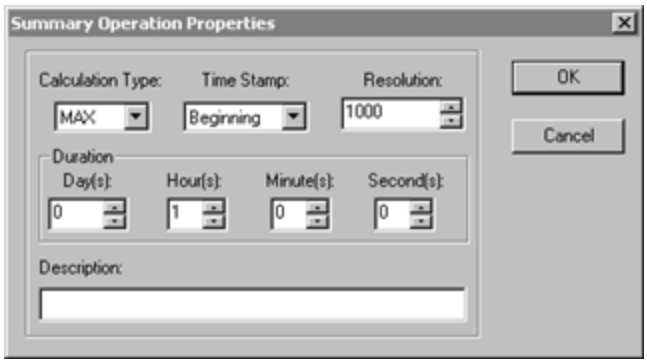
Adding a Summary Operation

You can add multiple summary operations for a single summary action, as long as no two summary operations have the exact same configuration.

To add a summary operation

1. In the summary action options, click **Add**. The **Summary Operation Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. In the **Calculation Type** list, select the type of calculation to be performed: SUM, MAX, MIN, or AVG.
3. In the **Time Stamp** list, select the timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends.
4. In the **Resolution** box, enter sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.

   In general, the higher the resolution, the more accurate the result, because you are including more values into your aggregation. However, the calculation takes longer and consume more server resources. Avoid very fine resolutions for summary actions associated with schedule detectors that cover long periods of time, such as weekly.

   Resolution is also very useful when calculating SUMS. For example, setting the resolution to 60,000 milliseconds for a flow in gallons per minute automatically produces a result that is the total volume.
5. In the **Duration** group, select the period for which the calculation must be performed.

   For example, if you are associating a summary action with a duration of 1 hour with a detector that is scheduled for 3:00 a.m. every Monday, then the system performs the aggregation on values stored between 2:00 a.m. and 3:00 a.m. on Mondays.
6. In the **Description** box, type a description of the summary operation.
7. Click **OK**.

The new summary operation now appears in the summary action grid.

Assigning a Tag to a Summary Operation

Assigning a Tag to a Summary Operation

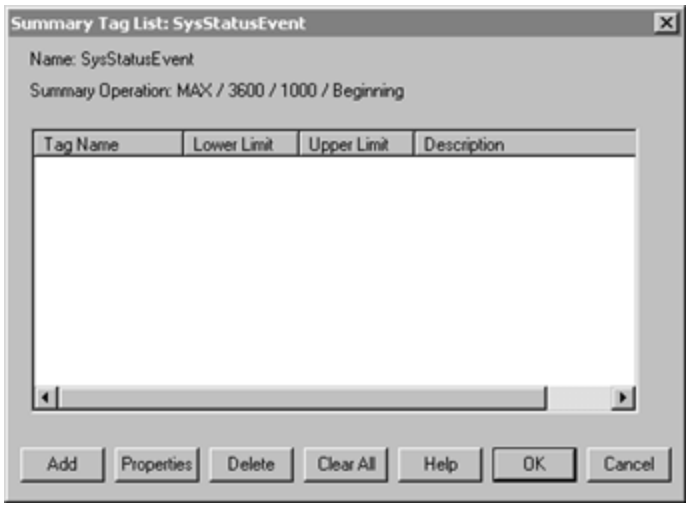
You can add more than one tag to a single summary.

You cannot add string tags to a summary operation.

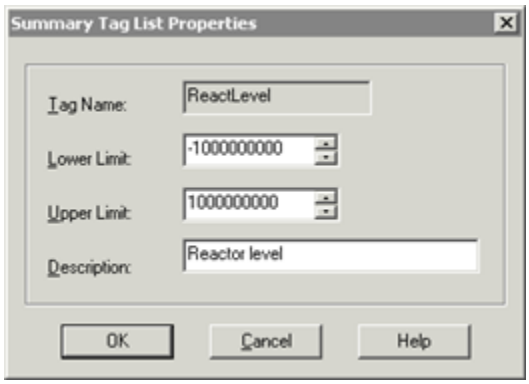
To assign a tag to a summary operation

1. In the summary action options, select the summary operation in the list and then click **Tags**. The **Summary Tag List** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. To search for a tag in the database, click **Add**. The **Tag Finder** dialog box appears, in which you can query the database for tags. For more information, see Using the Tag Finder.
3. After you add a tag, select the tag in the list and then click **Properties**. The **Summary Tag List Properties** dialog box appears.



4. In the **Lower Limit** and **Upper Limit** boxes, set the validity range for the summary tag. Setting a validity range allows you to control the lower or higher limits at which the calculation is performed.

   Upper Limit

   The upper limit of validity for the tag's value. Values higher than this limit are not used in the calculation. By default, this value is set to 1000000000.

   Lower Limit

   The lower limit of validity for the tag's value. Values lower than this limit are not used in the calculation. By default, this value is set to -1000000000.

   For example, if the lower validity limit is 1000, the calculation algorithm ignores all returned data with a value lower than 1000 when performing the aggregation.
5. In the **Description** box, type a description of the summarized tag. This normally describes the result of the operation, although this description can be the same as that of the tag on which the operation is performed.
6. Click **OK**. The new summary operation tag will appears in the **Summary Tag List** dialog box.
7. To delete a summary tag from the list, select the tag and then click **Delete**.
8. To delete all of the summary tags, click **Clear All**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

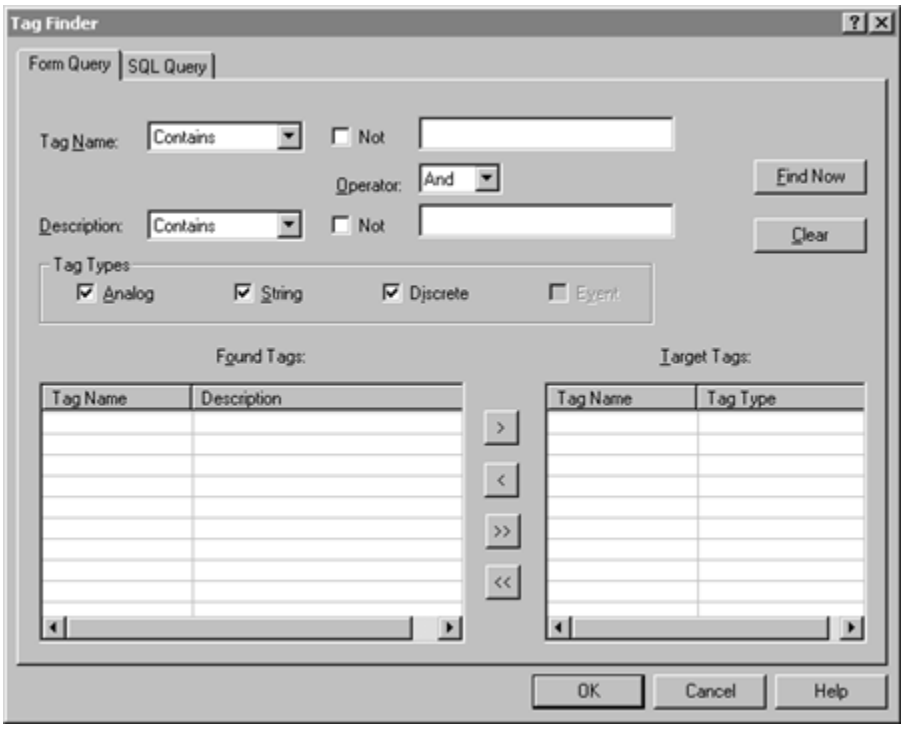
Using the Tag Finder

You can search the database for tags using the **Tag Finder** dialog box. This dialog box can be accessed, for example, by clicking the **Search** or **Add** button in a dialog box.

Using the Tag Finder, you can quickly search the database for tags that match a particular search pattern for either a tagname or a tag's description. You can either search for tags by using the point-and-click interface or by typing in your own SQL statement. After the Tag Finder returns a set of tags that match the query, you can select the ones you want to include.

Using the Form Query Tab

Use the Form Query tab of the Tag Finder dialog box to select the criteria to search the database.



To form and execute the query

1. In the **Tag Name** list, choose the phrase for the search criteria for the tagname. For example, "Ends with."
2. Enter the tagname search parameters for the query. For example, "level". When searching for tags, you only need to specify wildcard characters to exclude a middle portion of the search word. For example, "le%el".
3. To exclude the parameter for a search, click **Not**.
4. To add search parameters for a tag's description, select a logical operator from the **Operator** list.
5. In the **Description** list, choose the phrase for the search criteria for the tag description. This field is optional.
6. Enter the tag description search parameters for the query. This field is optional. When searching for tags, you only need wildcard characters to exclude a middle portion of the description.
7. To exclude the parameter for a search, click **Not**.
8. In the **Tag Types** area, select a tag group to search.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

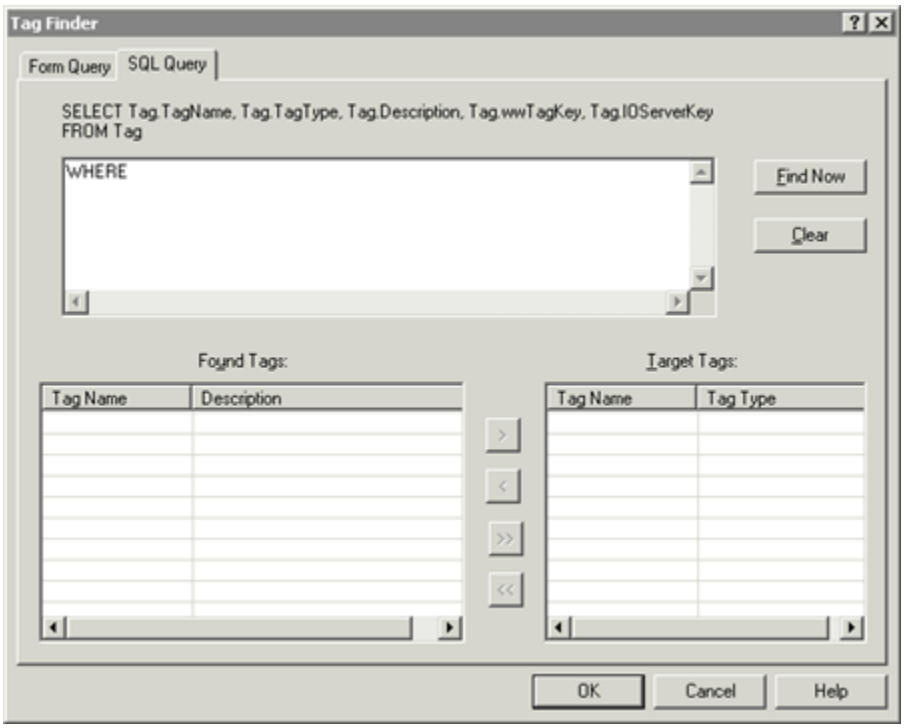
9. After you set the query parameters, click **Find Now** to run the query.

   The results of a tag search appears in the **Found Tags** window of the **Tag Finder** dialog box.
10. To add a tag, select the tag in the **Found Tags** window and then use the arrow button to move the selected tag into the **Target Tags** window.
11. Click **OK**.

To view the syntax used to query the database, click the **SQL Query** tab.

Using the SQL Query Tab

Use the **SQL Query** tab of the **Tag Finder** dialog box to enter and run your own SQL queries against the database.



To form and execute the query

1. In the query window, type in the WHERE clause parameters for the SQL query.

   Note: You cannot change the SELECT statement. The required tables and columns for the query result are already entered for you.
2. After you enter the query parameters, click **Find Now** to run the query.

   The results of a tag search appears in the **Found Tags** area of the **Tag Finder** dialog box.
3. To add a tag, select the tag in the **Found Tags** window and then use the arrow button to move the selected tag into the **Target Tags** window.
4. Click **OK**.

Retrieving Logged Event Data

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

When an event is detected, the event system logs the following into the EventHistory table: 1) the name of the event tag to which the criteria is associated; 2) the date/time stamp of the event occurrence; 3) the time the event is detected, and 4) the detection criteria information.

The detection criteria information, shown in the Edge column, is as follows:

Value	Description
0	Trailing Edge Detection (SQL Detectors)
1	Leading Edge Detection (SQL Detectors)
2	Detection on Both Edges (SQL Detectors)
3	No Edge Detection (SQL Detectors)
4	Schedule Detection
5	External Detection

If a snapshot action was configured for the event, the snapshot data is logged between the SnapshotTag table and the snapshot table for the tag type (for example, the AnalogSnapshot table). If a summary action is configured for the event, the aggregated data is stored in the SummaryHistory and SummaryData tables.

To view the event history, perform a query on EventHistory table. For example, an event tag, "EventTag1" detects when the value of "ReactLevel" was equal to 2000. The query to retrieve the event history on January 1, 2001, between 12:36 p.m. and 12:41 p.m. is:

```
SELECT * FROM EventHistory
   WHERE TagName = 'EventTag1'
      AND DateTime >= 'Jan 1 2001 12:36 PM'
      AND DateTime <= 'Jan 1 2001 12:41 PM'
```

To view action snapshot information for an event tag (no wildcards allowed), use the v_EventSnapshot view and specify the name of the event tag as the event in the WHERE clause. For example:

```
SELECT * FROM v_EventSnapshot
   WHERE Event = 'EventTag1'
```

This query returns the name of the event tag, the time of the event occurrence, and the detection time, as well as the name, value, and quality for each tag in the snapshot. One row is returned for each tag value.

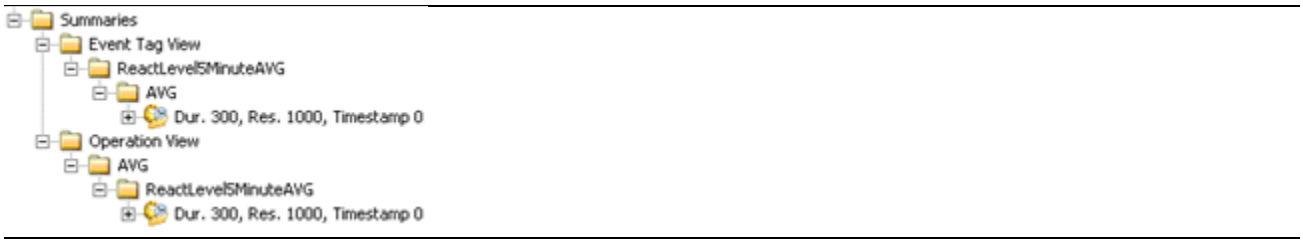
Viewing Summary Information

If you configure summary actions in the event system, you can view information pertaining to them in the console tree. You can also view the summary history.

To view summary information

1. In the console tree, expand a server group and then expand a server.
2. Expand **Configuration Editor**, expand **System Configuration**, and then expand **Tag Configuration**.
3. Expand **Summaries**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



4. To view all summaries sorted according to name of the event tag, expand **Event Tag View**.
5. To view all summaries grouped by summary operation (AVG, MIN, MAX, SUM), expand **Operation View**.

Viewing Summary Tag Properties

If you select the summary operation details item ("Dur. xxxx, Res. xxxx, Timestamp x") in the console tree, the summary tag properties appear in the details pane. The columns for the properties are as follows:

Tag Name

The name of the tag to be summarized.

Description

The description of the summarized tag. This normally describes the result of the operation, although this description can be the same as that of the tag on which the operation is performed.

Upper Limit

The upper limit of validity for the tag's value. Values higher than this limit are not used in the calculation. By default, this value is set to 1000000000.

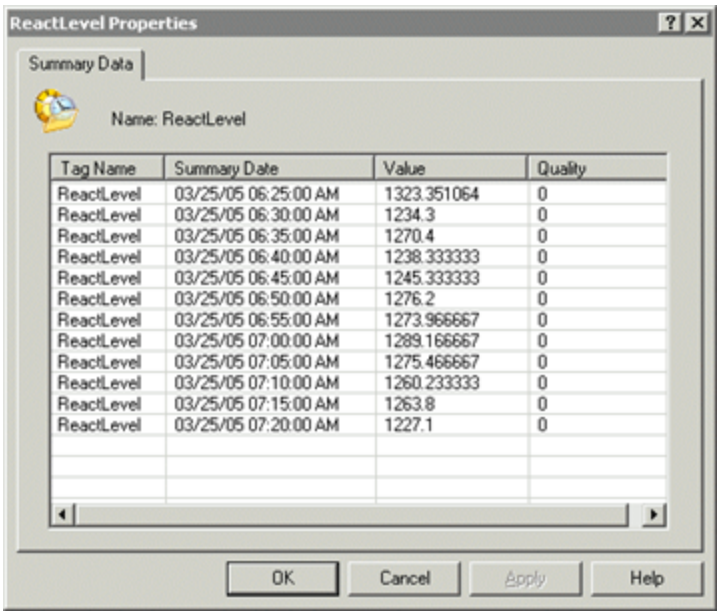
Lower Limit

The lower limit of validity for the tag's value. Values lower than this limit are not used in the calculation. By default, this value is set to -1000000000.

Viewing Data for a Summary Tag

To view the summary data for a event tag, click the summary operation details item ("Dur. xxxx, Res. xxxx, Timestamp x") in the console tree. The summary tag properties appear in the details pane. Double-click a summary tag in the pane.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Tag Name	Summary Date	Value	Quality
ReactLevel	03/25/05 06:25:00 AM	1323.351064	0
ReactLevel	03/25/05 06:30:00 AM	1234.3	0
ReactLevel	03/25/05 06:35:00 AM	1270.4	0
ReactLevel	03/25/05 06:40:00 AM	1238.333333	0
ReactLevel	03/25/05 06:45:00 AM	1245.333333	0
ReactLevel	03/25/05 06:50:00 AM	1276.2	0
ReactLevel	03/25/05 06:55:00 AM	1273.966667	0
ReactLevel	03/25/05 07:00:00 AM	1289.166667	0
ReactLevel	03/25/05 07:05:00 AM	1275.466667	0
ReactLevel	03/25/05 07:10:00 AM	1260.233333	0
ReactLevel	03/25/05 07:15:00 AM	1263.8	0
ReactLevel	03/25/05 07:20:00 AM	1227.1	0

The columns are as follows:

TagName

The unique name of the tag within the AVEVA Historian system.

Summary Date

The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.

Value

The value of the summary.

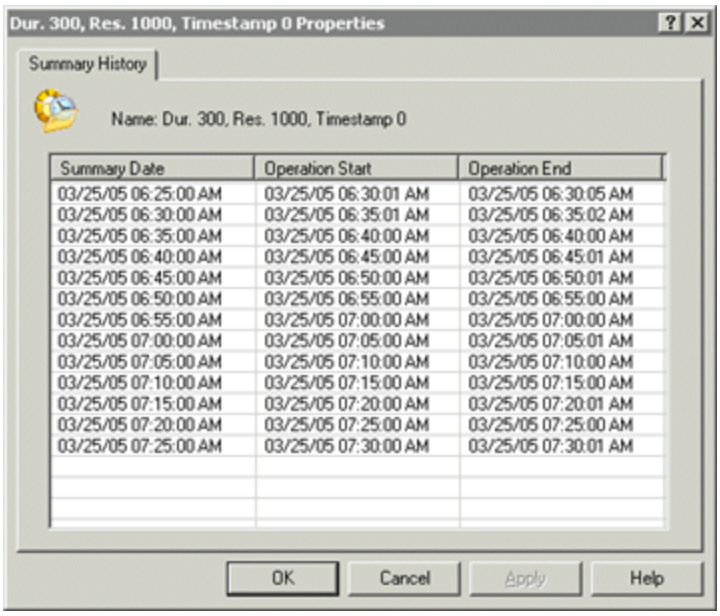
Quality

The basic data quality indicator associated with the data value.

Viewing History for a Summary Operation

To view the summary history for a particular operation, right-click the summary operation details item ("Dur. xxxx, Res. xxxx, Timestamp x") in the console tree and click **Properties**.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Summary Date	Operation Start	Operation End
03/25/05 06:25:00 AM	03/25/05 06:30:01 AM	03/25/05 06:30:05 AM
03/25/05 06:30:00 AM	03/25/05 06:35:01 AM	03/25/05 06:35:02 AM
03/25/05 06:35:00 AM	03/25/05 06:40:00 AM	03/25/05 06:40:00 AM
03/25/05 06:40:00 AM	03/25/05 06:45:00 AM	03/25/05 06:45:01 AM
03/25/05 06:45:00 AM	03/25/05 06:50:00 AM	03/25/05 06:50:01 AM
03/25/05 06:50:00 AM	03/25/05 06:55:00 AM	03/25/05 06:55:00 AM
03/25/05 06:55:00 AM	03/25/05 07:00:00 AM	03/25/05 07:00:00 AM
03/25/05 07:00:00 AM	03/25/05 07:05:00 AM	03/25/05 07:05:01 AM
03/25/05 07:05:00 AM	03/25/05 07:10:00 AM	03/25/05 07:10:00 AM
03/25/05 07:10:00 AM	03/25/05 07:15:00 AM	03/25/05 07:15:00 AM
03/25/05 07:15:00 AM	03/25/05 07:20:00 AM	03/25/05 07:20:01 AM
03/25/05 07:20:00 AM	03/25/05 07:25:00 AM	03/25/05 07:25:00 AM
03/25/05 07:25:00 AM	03/25/05 07:30:00 AM	03/25/05 07:30:01 AM

The Summary History columns are as follows:

Summary Date

The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.

Operation Start

The timestamp when the calculation started for the operation.

Operation End

The timestamp when the calculation completed for the operation.

Using ActiveEvent

ActiveEvent is an ActiveX control that notifies the Event subsystem when an event occurs in another application, such as InTouch HMI software. ActiveEvent is script-based. You can use it in any application that uses a COM-enabled scripting language to detect an event for that application. COM-enabled scripting languages include InTouch scripting and Visual Basic.

After you install the ActiveEvent control on an InTouch computer using the AVEVA Historian installation program, ActiveEvent does not automatically appear in the list of available ActiveX objects for use within WindowMaker. You need to run the Wizard/ActiveX installation from within WindowMaker, as well. For more information on performing a Wizard/ActiveX installation, see your InTouch documentation.

To enable external event detection for the historian, you must:

1. Create an event tag in the historian to store the event occurrence information. Make sure that the detection type is set to External.

   You can define the event tag so that the event is associated with an action that is triggered from the historian, such as executing a SQL script, sending an e-mail message, or recording the values of a set of tags at the time the event occurred.

   For more information, see Adding an Event Tag.
2. Install the ActiveEvent control so that it can be used in the ActiveX container application (for example, in

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

InTouch HMI software).

For more information on installing the ActiveEvent control, see the *AVEVA System Platform Installation Guide*.

3. Configure the DCOM security attributes for the external detector to be used with ActiveEvent. Security attributes must be set up on the AVEVA Historian computer.

   For information, see Configuring Security Attributes for ActiveEvent.

4. Write a script that notifies the historian event system of the external event.

   For more information, see ActiveEvent Methods.

Important: You cannot use ActiveEvent in an InTouch version 7.0 SP2 application.

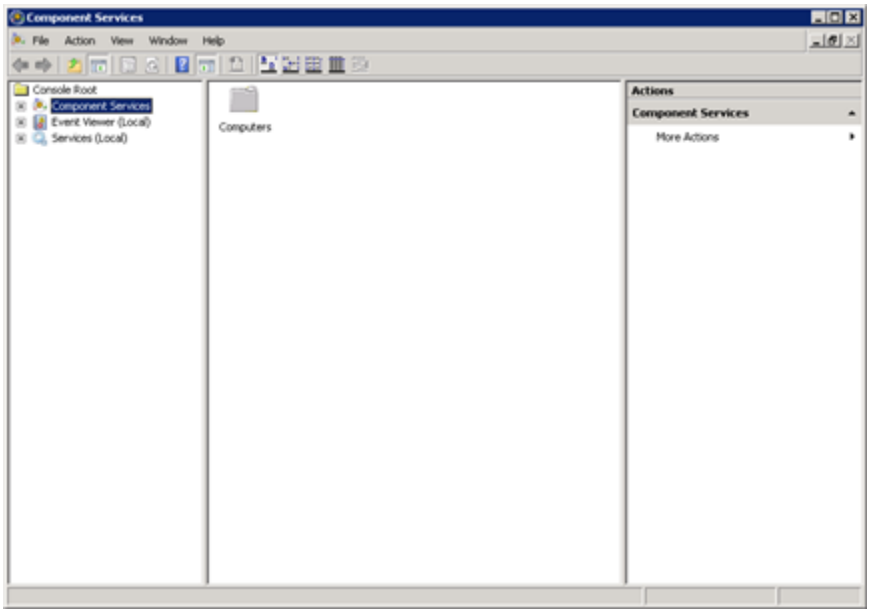
Synchronize the system time for the ActiveEvent computer with the system time for the historian. If the ActiveEvent computer time is ahead, the event system may generate NULL values for snapshot data.

Configuring Security Attributes for ActiveEvent

For ActiveEvent to work, security attributes (for example, permission to launch) must be correctly configured. Configure the security attributes on the AVEVA Historian computer using the DCOMCnfg.Exe program.

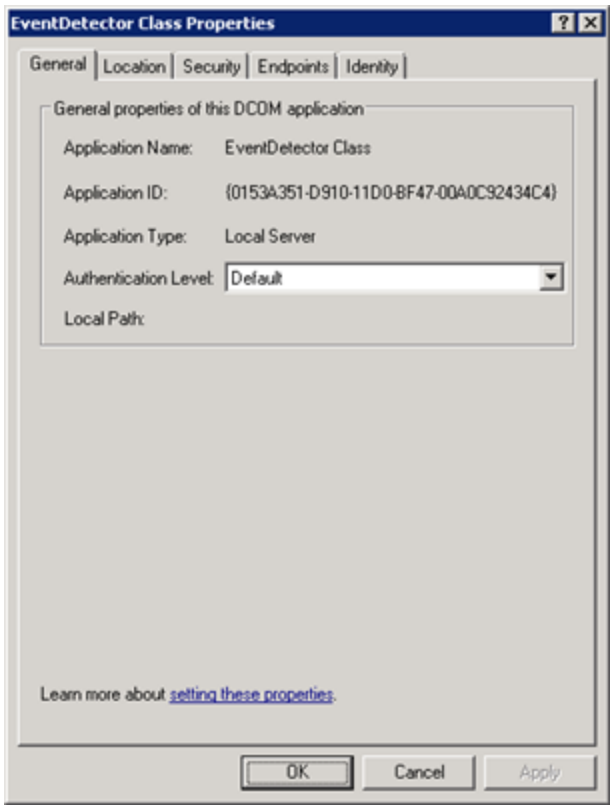
To configure security attributes for ActiveEvent:

1. On the Windows **Start** menu, click **Run**. The **Run** dialog box appears.
2. In the **Open** box, type DCOMCnfg.Exe.
3. Click **OK**. The **Distributed COM Configuration Properties** dialog box appears.

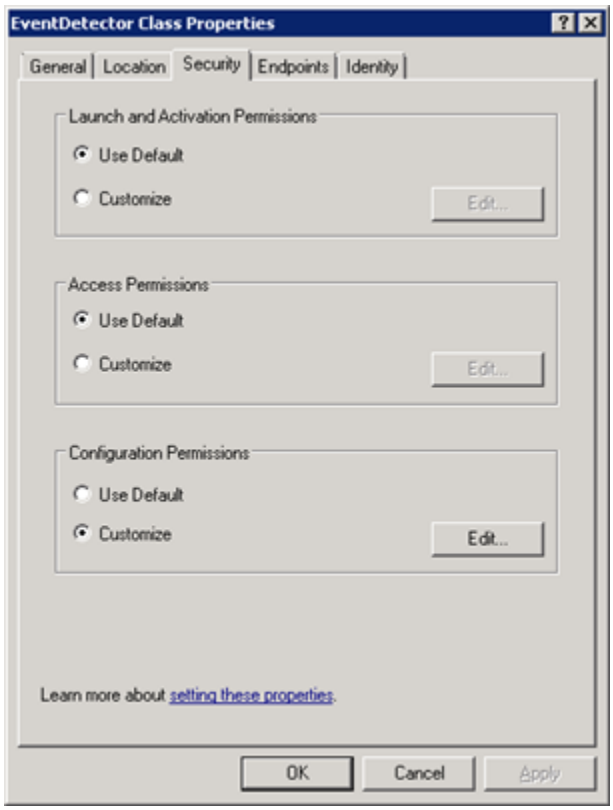


4. Click **Computers**, then **My Computer**, and then **DCOM Config**, and then select **EventDetector Class**.
5. Click **Properties**. The **EventDetector Class Properties** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

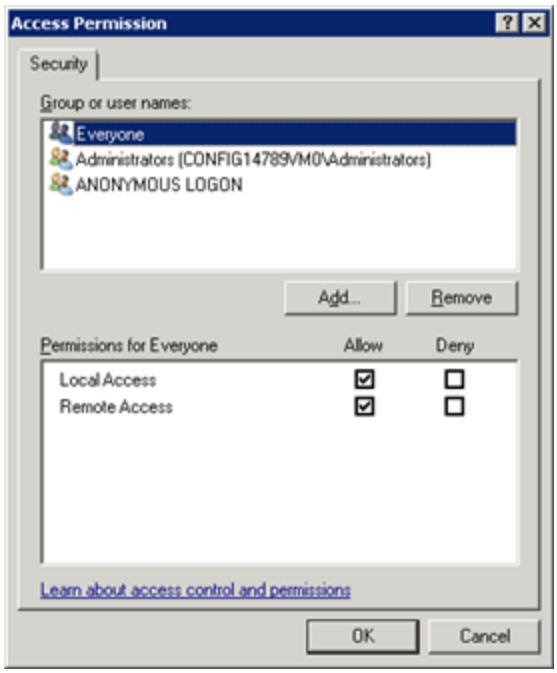


6. Click the **Security** tab.

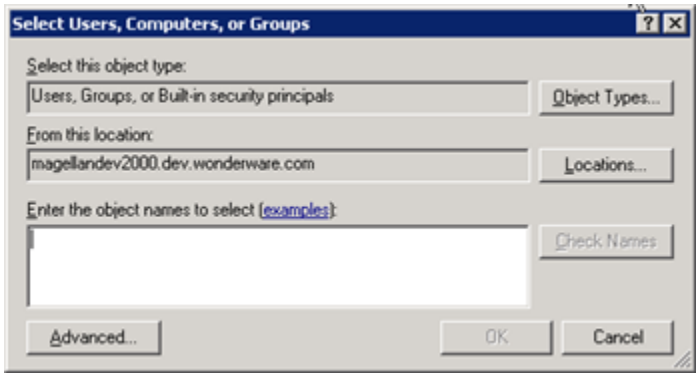


7. Select **Access permissions - Customize** and then click **Edit**. The **Access Permission** dialog box appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



8. Click **Add**. The **Select Users, Computers, and Groups** dialog box appears.



9. In the **Names** list, type **Everyone**.
10. Click **OK**. The **Access Permission** dialog box now shows the **Everyone** group with allowed access.
11. Click **OK** to return to the **EventDetector Class Properties** dialog box.

Repeat the general procedure to add **Everyone** to the launch permissions.

1. Select **Launch and Activation permissions - Customize** and then click **Edit**.
2. Repeat steps 8 through 11 to add the **Everyone** group to the list of users who have launch permissions.
3. Click **OK** to close the **EventDetector Class Properties** dialog box.
4. Click **OK** to close the DCOMCnfg.Exe program.

ActiveEvent Methods

Use ActiveEvent's methods in scripts to connect to an AVEVA Historian and trigger an event. The ActiveEvent control aids the remote triggering of events on the historian by first initializing with the historian computer name

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

and event tag, and then calling the InvokeEventEx() method.

ActiveEvent can be scripted using any scripting language that supports COM. For example, an InTouch script can trigger an AVEVA Historian event if you use this control in an InTouch application. You can also trigger an event from a Visual Basic script.

Note: ActiveEvent does not work in asynchronous mode in an InTouch application.

The following example InTouch script connects to a server named "WWHistorianServer1," adds the event tag called "ExternalEvent," and logs an "ExternalEvent" event tag event.

```
{ Connect ActiveEvent to your AVEVA Historian--only needs to be done once.}
intResult = #WWHistEvent1.InitializeEx( "WWHistorianServer1");
{Was initialization successful or are we already initialized? }
IF intResult == 0 OR intResult == 4 THEN
   intResult = #WWHistEvent1.AddEventTag("ExternalEvent");
   IF intResult == 0 THEN
      intResult = #WWHistEvent1.InvokeEventEx("ExternalEvent");
      IF intResult == 0 THEN
         sDisplayResult = "Logged event";
      ELSE
         sDisplayResult = "Failed to log event";
      ENDIF;
   ENDIF;
ENDIF;
```

AddEventTag()

AddEventTag()

Adds an event tag to the active event tag list

Method

```
AddEventTag(string EventTag)
```

Parameter

EventTag

Name of the event tag with which the ActiveEvent event detector is associated. ActiveEvent is used with an external type event detector.

Returns Value

0 = Success.

2 = Unable to execute method because ActiveEvent is not initialized.

7 = Remote function call failed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

InitializeEx()

InitializeEx()

Creates a connection to the AVEVA Historian.

Method

```
InitializeEx(string ComputerName)
```

Parameter

ComputerName

Name of the computer on which the historian is running. If you are not connecting to the historian over a network, use a blank string ("") for the computer name.

Note: You cannot use an AVEVA Historian alias for this parameter.

Returns Value

0 = Success.

1 = Unknown failure.

3 = Unable to initialize ActiveEvent.

4 = ActiveEvent is already initialized.

7 = Remote function call failed.

8 = Unable to determine local computer name.

Remarks

After you initialize the historian, use the IsConnected property to determine if the connection was successful. Also, you only need to initialize with the server one time. You can invoke an unlimited number of events after initialization has occurred.

If you are using the InTouch HMI software, initialization does not occur unless the ActiveEvent ActiveX control is part of an open window. This limits the use of the InvokeEventEx method within InTouch Application Scripts, Condition Scripts, or Data Change Scripts.

When you close an InTouch window, all ActiveX controls are automatically uninstantiated.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

InvokeEventAtTimeEx()

InvokeEventAtTimeEx()

Triggers the event at a specified date/time.

Method

```
InvokeEventAtTimeEx(string TagName, string EventDateTime)
```

Remarks

You can invoke an unlimited number of events after you initialize with an AVEVA Historian.

Parameter

TagName

Name of the event tag with which the ActiveEvent event detector is associated. ActiveEvent is used with an external type event detector.

EventDateTime

Date/time that you want the event triggered. This date is in local time for the historian. The event date and time must be formatted as:

```
YYYY-MM-DD hh:mi:ss.mmm
```

Returns Value

0 = Success.

1 = Unknown failure.

2 = Unable to execute method because ActiveEvent is not initialized.

5 = Unable to perform date/time conversion due to invalid format.

6 = Date/time cannot be a future date.

7 = Remote function call failed.

InvokeEventEx()

InvokeEventEx()

Triggers the event at the time this method is called.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Method

```
InvokeEventEx(string EventTag)
```

Remarks

You can invoke an unlimited number of events after you initialize with an AVEVA Historian.

Parameter

EventTag

Name of the event tag with which the ActiveEvent event detector is associated. ActiveEvent is used with an external type event detector.

Returns Value

0 = Success.

1 = Unknown failure.

2 = Unable to execute method because ActiveEvent is not initialized.

7 = Remote function call failed.

IsConnected

IsConnected

Determines whether a connection to the AVEVA Historian exists.

Method

```
IsConnected
```

Returns Value

0 = Not connected

1 = Connected to the historian

RemoveEventTag()

RemoveEventTag()

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Removes an event tag from the active event tag list.

Method

```
RemoveEventTag(string EventTag)
```

Parameter

EventTag

Name of the event tag to remove from the list of external events for the ActiveEvent control.

Returns Value

0 = Success.

2 = Unable to execute method because ActiveEvent is not initialized.

7 = Remote DCOM call failed.

Scripting Example: Triggering Events within an InTouch Application

To trigger an event within an InTouch application, include these methods in an InTouch script, similar to the following:

```
#WWHistEvent1.InitializeEx("Historian01"); {Initialized the server}
#WWHistEvent1.AddEventTag("ASVTag");
#WWHistEvent1.AddEventTag("SysStatusEvent"); {Added event tag}
#WWHistEvent1.InvokeEventEx("ASVTag");
#WWHistEvent1.InvokeEventEx("SysStatusEvent"); {Invoked event}
```

where:

- WWHistorianEvent1 is the name of the instantiation of the ActiveEvent ActiveX control
- Historian01 is the computer name for the AVEVA Historian (not an alias)
- ASVTag is the name of the event tag that is associated with this external detector

To add more tags to be detected, use the **AddEventTag()** method and use **InvokeEventEx()** specifying the tagname. A single ActiveEvent control handles many tags.

Scripting Example: Triggering Multiple Events within Visual Basic

In this Visual Basic script, the initialization occurs with AVEVA Historian running on a specified computer, and more than one event is invoked:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Private Sub Command1_Click()
   Dim ComputerName As String
   ComputerName = "Computer1"
   Dim TagName As String
   Dim Connected As Long
   TagName = "Event1"
   WWHistEvent1.InitializeEx ComputerName
   WWHistEvent1.AddEventTag TagName
   Connected = WWHistEvent1.IsConnected
   If Connected = 1 Then
      WWHistEvent1.InvokeEventEx TagName
      WWHistEvent1.InvokeEventEx TagName
      WWHistEvent1.InvokeEventEx TagName
      MsgBox ("Sent off three events")
   Else
      MsgBox ("Failed To Connect")
   End If
End Sub
```

History Block Storage for Alarms and Events

Historian can store alarm and events history in history blocks.

For information about setting up history blocks for alarms and events, see the *AVEVA System Platform Installation Guide*.

You can migrate alarms and events from the A2ALMDB database to history blocks. For more information, see Migrating Data from the A2ALMDB Database to History Blocks.

Note: Alarms and events from AVEVA Application Server versions prior to 2014 R2 were stored in the A2ALMDB database. Events previously configured using the Classic Event subsystem were stored in SQL Server tables in the Runtime database.

A2ALMDB Database

The *A2ALMDB* SQL Server database stores alarms and events generated through external sources such as Application Server.

Notes: Starting with AVEVA Historian 2014 R2, alarms and events from Application Server can be stored to history blocks instead of the *A2ALMDB* database. Also, the Classic Event subsystem stores data in the *Runtime* database, not the *A2ALMDB* database. For more information on the Classic Event subsystem, see Configuring Classic Events.

Earlier versions of AVEVA System Platform used WWALMDB database rather than A2ALMDB for alarms and events. If you are currently using WWALMDB and want to to use A2ALMDB instead, you may need to change your alarm historization settings from within the System Platform IDE and change your alarm queries to use A2ALMDB.

Managing the *A2ALMDB* database is necessary to ensure that the size of the database is maintained within its normal operating parameters. If the database is left unchecked and allowed to grow unbounded, the risk of losing valuable data increases.

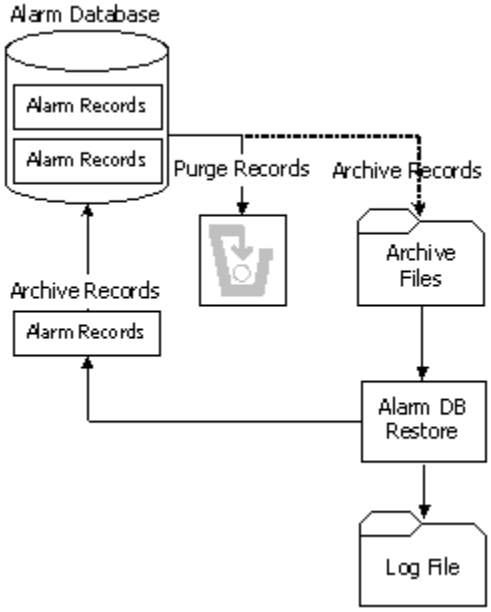
You manage the alarm database using two alarm database utilities. Use the Alarm DB Purge-Archive utility to remove records from the database permanently or archive them to files. Use the Alarm DB Restore utility to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

query previously archived data.

Purging is used to permanently remove data that is no longer required. Archiving allows data to be exported to a file so they can later be restored in needed.

The following figure shows how both utilities purge/archive records and then restore them back to the database.



You must be logged into the computer as an administrator to use the Alarm DB Purge-Archive utility.

Configuring Purge or Archive Settings

Use the Alarm DB Purge-Archive utility to:

- Manually purge or archive records from the database to free up space and improve performance.
- Automatically purge or archive records regularly.
- Optionally archive database records to files.
- Save the status of archive or purge operations to a log file to troubleshoot problems.
- Show the status of purge or archive operations.

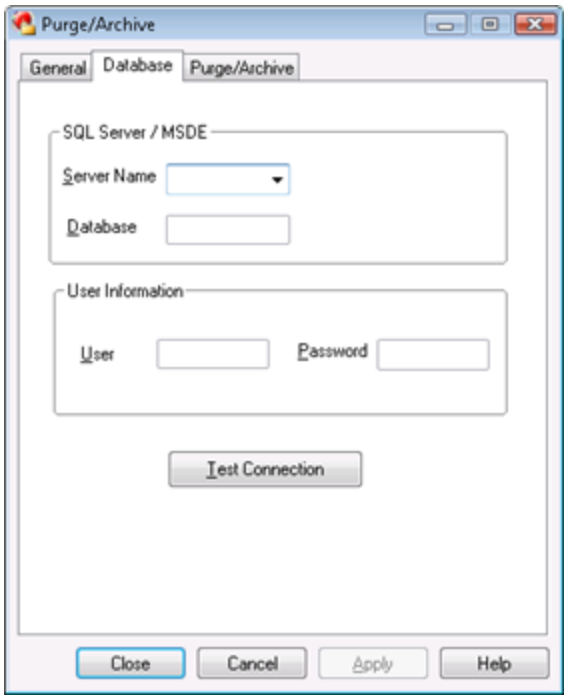
Configuring the Database Connection

You must select a database to restore the archived data to. If the specified database is not present on the server, you are prompted to create a new database with default server parameters.

To configure a database for restoring

1. Open the Alarm DB Restore utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Restore**.
2. Click the **Configure** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Configure the connection to the alarm database. Do the following:
   a. In the **SQL Server Name** list, click the node name of the server that hosts the alarm database.
   b. In the **Database Name** box, type the name of the alarm database.
   c. In the **User Information** area, type an alarm database user name and password in the respective boxes.
4. Click **Test Connection** to test your connection to the database. A message indicates whether the connection to the alarm database is successful or not. Click **OK**.
5. Click **Apply**.

Configuring How Much Data to Purge from the Server

You can:

- Select the type of alarm records to be purged from the alarm database.
- Optionally, archive purged records from the alarm database to files.
- Select the folder location to store the purge log file.

You can select the type of table that needs to be purged, either the AlarmDetail or AlarmConsolidated table.

Notes: The Historian alarm database is named *A2ALMDB*. Earlier versions of AVEVA System Platform used *WWALMDB* database rather than *A2ALMDB* for alarms and events. If you are currently using *WWALMDB* and want to to use *A2ALMDB* instead, you may need to change your alarm historization settings from within the System Platform IDE and change your alarm queries to use A2ALMDB.

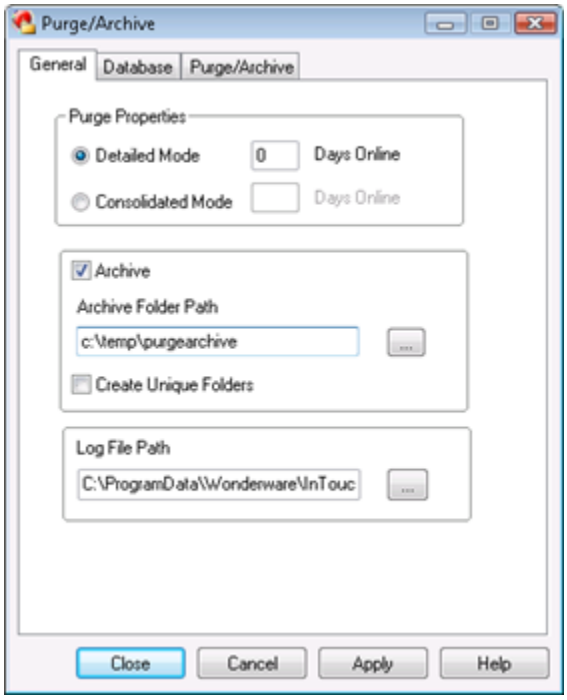
The Historian *A2ALMDB* database is created only in Detailed mode, while the InTouch *WWALMDB* database is supported in both Detailed and Consolidated modes.

All data from the day previous to the number specified is purged. Valid entries are 0-9999. If you select 0, all records are purged from the alarm database except the current day's records.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To select records to purge

1. Open the Alarm DB Purge-Archive utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Purge-Archive**.
2. Click the **General** tab.



3. In the **Purge Properties** area, configure the type of records to purge. Do either of the following:
   - Click **Detailed Mode** to purge alarm records that are saved in the database in Detailed mode.
   - Click **Consolidated Mode** to purge alarm records that are saved in the database in Consolidated mode.
4. In the **Days Online** box, type the number of days worth of records to retain in the alarm database.
5. Click **Apply**.

Configuring the Archive of Purged Data

You archive the records purged from the alarm database and then restore them using the Alarm DB Restore utility.

When you purge the alarm database, the Alarm DB Purge-Archive utility automatically creates a set of nine archive files that correspond to the purged alarm database tables. Each file contains the purged records of a single table.

The Alarm DB Purge-Archive utility assigns names to the archive files based upon the table name, date, and time when the purge operation occurred. For example, the name of the archive file for the AlarmMaster table that was purged on June 22, 2007 at 5:30 p.m. is formatted like the following:

```
AlarmMaster_06222007_1730.txt
```

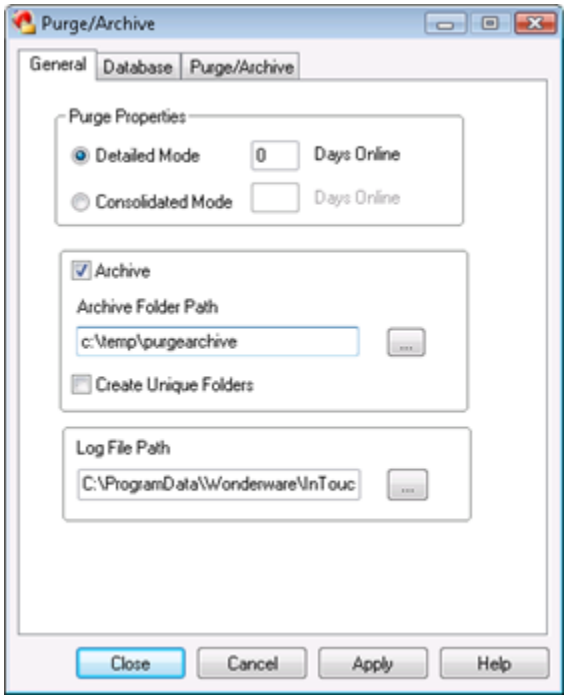
To configure the archive

1. Open the Alarm DB Purge-Archive utility. Do the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

a. In the **Tools** view, expand **Applications**.
b. Double-click **Alarm DB Purge-Archive**.

2. Click the **General** tab.



3. Select the **Archive** check box.
4. In the **Archive Folder Path** box, type the folder location where archive files should be saved or click the ellipsis button to browse for the location.
5. Select the **Create Unique Folders** check box if you want the archive files to be placed in an individual sub-folder beneath the archive file folder.
6. Click **Apply**.

Configuring Log File Settings

The Alarm DB Purge-Archive utility generates status messages during a purge operation. You can view these messages online from the utility's **Status** window. The Alarm DB Purge-Archive utility also writes purge messages to the purge log file named WWAlmPurge.log.

The example below shows the messages stored in the log file after a successful purge operation.

```
Purge Started on 12:16:48 PM 6/22/2007
Starting transaction....
Archiving Table ProviderSession...
Archiving Table Query...
Archiving Table Cause...
Archiving Table Alarm Master...
Archiving Table OperatorDetails...
Archiving Table Alarm Detail...
Archiving Table Comment...
Archiving Table Events...
Archiving Table TagStatus...
Purging records in the database...
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Committing....
Purge Completed On 12:16:52 PM 6/22/2007
144 records from AlarmMaster were purged along with the related records from other
tables.
```

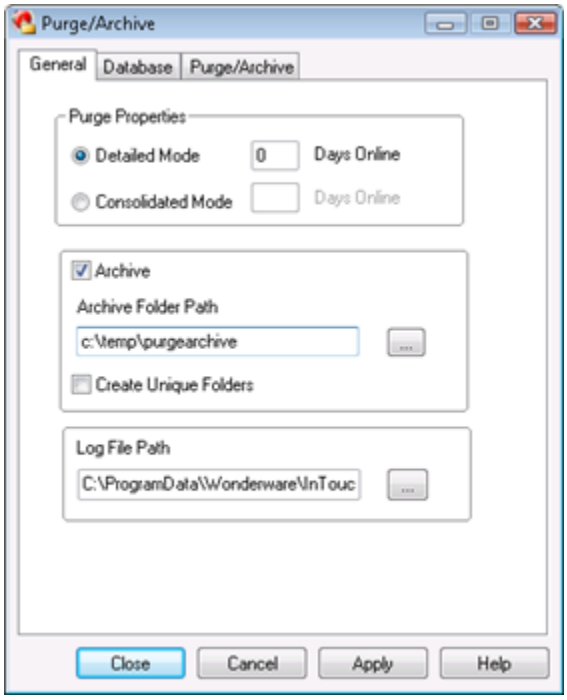
By default, the purge log file is stored in this folder: C:\Users\UserName\Documents\My InTouch Applications.

You can change the storage location of the purge log file.

The Alarm DB Purge-Archive utility appends new messages to the log file each time a purge occurs.

To set archive logging

1. Open the Alarm DB Purge-Archive utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Purge-Archive**.
2. Click the **General** tab.



3. In the **Log File Path** box, type the folder location where the purge log file should be placed or click the ellipsis button to browse for the location.
4. Click **Apply**.

Manually Purging and Archiving the Database

You can purge and archive your alarm database manually. This overrides the activation time and starts the purging and archiving immediately.

The purge operation checks for the presence of an archive file and appends to the same. If the archive file is not present, the file is created as per the naming convention and then used for archiving.

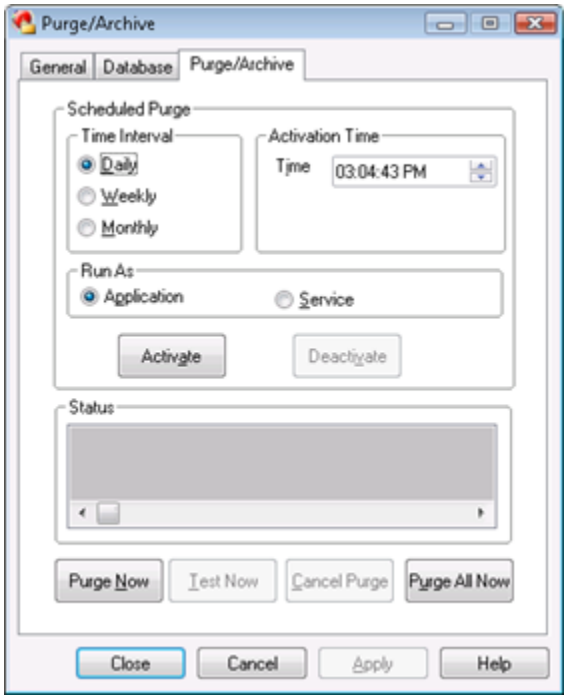
The purge operation does not delete entries in tables such as ProviderSession, Query, and Cause that are linked to the main tables such as AlarmMaster through foreign key constraints. The related records in these tables are written to the files to maintain the data consistency and also retained in the database.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Caution: Manually purge all records (the Purge All Now option) only when the Alarm DB Logger service is stopped. If the purge operation is committed successfully while the Alarm DB Logger service is running, the Alarm DB Logger service stops logging and starts caching records.

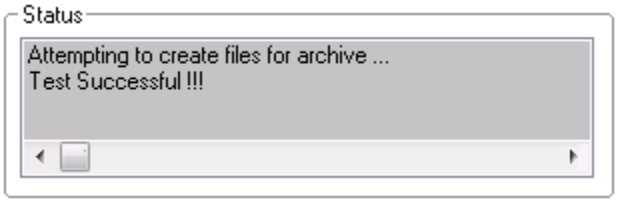
To manually purge and archive records from the alarm database

1. Open the Alarm DB Purge-Archive utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Purge-Archive**.
2. Click the **Purge/Archive** tab.



3. Click **Test Now** to perform a test purge to verify your connection to the database and archive locations.

   The test purge creates empty archive files in the specified archive folder. The **Status** area shows a message that the test was successful.



   The **Test Now** button is available only if you have chosen to archive your purged records. The Archive option is located on the **General** tab.
4. Purge the records from the database. Do either of the following:
   - Click **Purge Now** to purge the selected records.
   - Click **Purge All Now** to purge all records.
5. To stop a purge, click **Cancel Purge**. If you cancel the purge, the alarm database is rolled back to its original state.

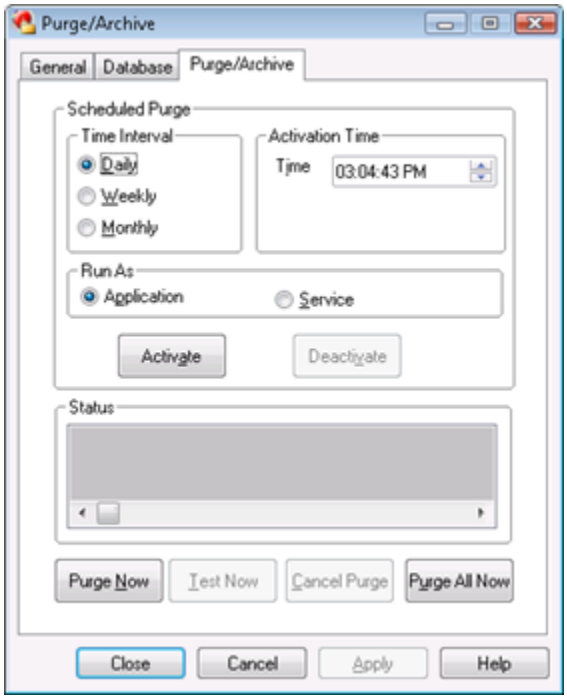
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Setting a Schedule for Automatic Purging

The Alarm DB Purge-Archive utility can automatically purge or archive records from the alarm database at scheduled intervals. You can perform a test purge to verify your connection to the database and target locations and to start and stop purging.

To set a schedule for automatic purging

1. Open the Alarm DB Purge-Archive utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Purge-Archive**.
2. Click the Purge/Archive tab.



3. In the Time Interval area, select a purge interval, either daily, weekly, or monthly.

   If you click **Weekly** or **Monthly**, a **Day** box appears in the **Activation Time** area for you to specify the day of the week or day of the month.

   If you click **Daily**, in the **Time** box, configure the time of day that you want the purge/archive operation to start.
4. In the **Run As** area, click **Service** to run the purge-archive utility as a service. It is recommended to run the scheduled utility as a service to ensure the utility restarts automatically after a computer reboot.
5. Click Apply to save your purge and archive settings.
6. Click **Activate** to place the Alarm DB Purge-Archive utility on an automatic purge schedule.
7. Click **Close**.

Restoring the Alarm Database

The Alarm DB Restore utility restores the archived alarm records in the archive files back to your alarm database. The following figure summarizes the steps to restore alarm records to the database.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AlarmDBRestore

To restore a database, you must:

- Configure the connection to the alarm database.
- Select which records to restore to the alarm database.
- Restore archived records to the alarm database.

When minimized, the Alarm DB Restore utility appears as an icon in the system tray. When you right-click the icon, a menu shows the following commands:

Command	Description
Restore	Begins the restoring process.
Cancel Restore	Cancels the restoring process.
Clear Status	Clears the status window.
Hide Window	Minimizes the Alarm DB Restore utility to an icon in the system tray.
Show Window	Opens and maximizes the Alarm DB Restore utility.
Exit	Closes the Alarm DB Restore utility.

If you right-click in the Alarm DB Restore utility, the same menu appears.

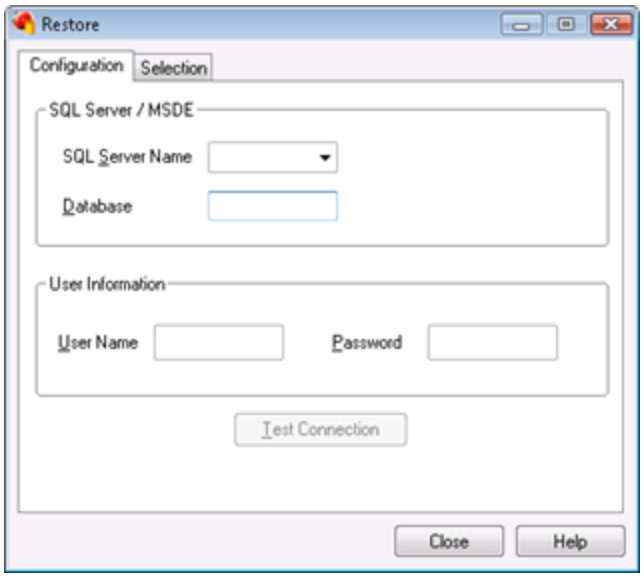
Configuring the Database Connection

You must select a database to restore the archived data to. If the specified database is not present on the server, you are prompted to create a new database with default server parameters.

To configure a database for restoring

1. Open the Alarm DB Restore utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Restore**.
2. Click the **Configure** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Configure the connection to the alarm database. Do the following:
   a. In the **SQL Server Name** list, click the node name of the server that hosts the alarm database.
   b. In the **Database Name** box, type the name of the alarm database.
   c. In the **User Information** area, type an alarm database user name and password in the respective boxes.
   d. Click **Test Connection** to test your connection to the database. A message indicates whether the connection to the alarm database is successful or not. Click **OK**.
4. Click **Close**.

Configuring Which Files to Restore

You can select a time period for the records to restore and whether you want the database tables to be recreated.

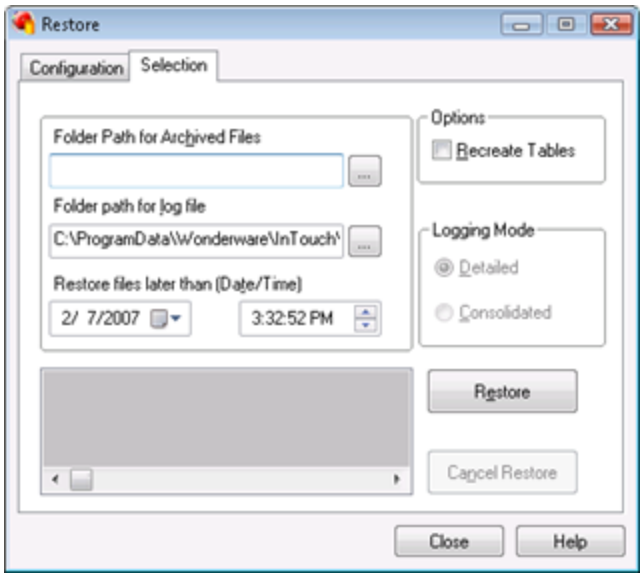
If you cancel the restore, the database is rolled back to its original state.

Caution: If you try to restore archived alarms that are already present in the database, the archived records are not restored. This avoids duplicate alarm/event entries in the database. The Alarm GUID or Event GUID associated with records determines whether an alarm or event is already present in the database.

To select database records to restore

1. Open the Alarm DB Restore utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Restore**.
2. Click the **Selection** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. In the **Folder Path for Archived Files** box, type the full path (up to 255 alphanumeric characters) to the location of the archived files or click the button to locate and select the folder where archived files are stored.
4. In the **Restore files later than (Date/Time)** area, select the date and time to start restoring records to the database.

   The starting date and time are set by default to the current date and time.
5. In the **Folder path for log file** box, type the full path (up to 255 alphanumeric characters) where the log files are created and stored or click the button to locate and select a folder.
6. If you select the **Recreate Tables** check box, the tables of the specified alarm database are recreated. Depending on the type of logging you selected for the alarm records contained in the archived files, select:
   - **Detailed** - Recreate the alarm database tables in detailed mode.
   - **Consolidated** - Recreate the alarm database tables in consolidated mode.

   Important: Recreating tables overwrites all records currently stored in the alarm database.
7. Click **Restore**.

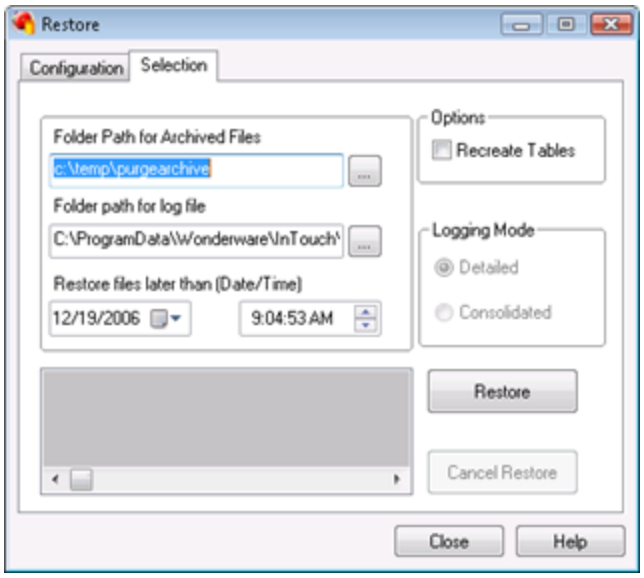
Starting a Database Restore Operation

You restore archived database records after you have established the database connection, specified the archived files folder and a time filter.

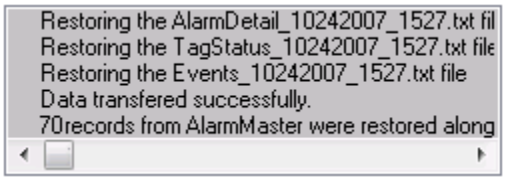
To restore database records from an archive

1. Open the Alarm DB Restore utility. Do the following:
   a. In the **Tools** view, expand **Applications**.
   b. Double-click **Alarm DB Restore**.
2. Click the **Selection** tab.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



3. Click **Restore**. A message shows whether the restoration is successful and the number of records restored to the database.



Migrating Data from the A2ALMDB Database to History Blocks

You can migrate the alarm and event data in the A2ALMDB database to history blocks. Before you start the migration, be sure that:

- The historian is running.
- The user account used to connect to the A2ALMDB database is a member of the aaAdministrators group. Members of the aaPowerUsers group can also run the utility if they are granted the VIEW DATABASE STATE permission.
- The historian is configured to store alarms and events to history blocks.
- There is at least one record in the A2ALMDB database.

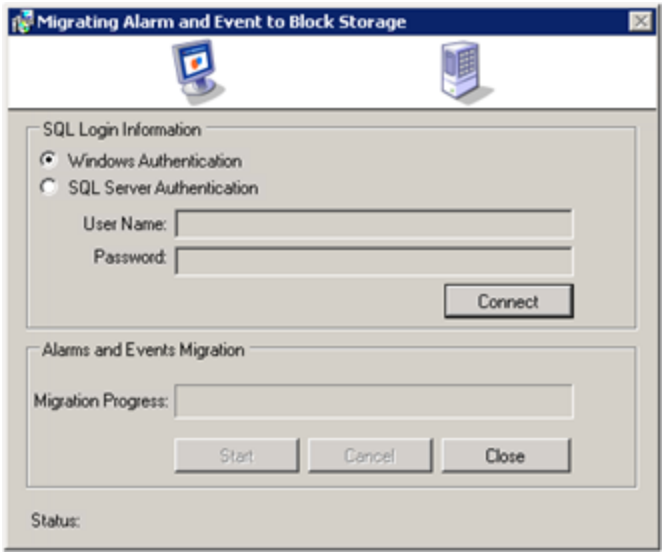
At the end of the migration, the following migration statistics are logged to the ArchestrA Logger:

- Number of alarms and events in the database
- Number of events forwarded to history block storage
- Number of events/second forwarded to history block storage

To migrate alarm and event data

1. On the **Start** menu, click **All Programs**, click **AVEVA**, click **Historian** and then click **Migrate A2ALMDB**. The **Migrating Alarm and Event to Block Storage** utility appears.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. In the SQL Login Information area, specify a user account that has administrative access to the A2ALMDB database.
3. Click **Connect**.
4. Click **Start** to begin the migration.

Historian Concepts

Welcome to AVEVA Historian

AVEVA Historian is a powerful, high-performance data historian. It can handle the immense volumes of data that modern industrial facilities generate.

Historian can move all of that data fast, acquiring your data for storage and retrieving it when you need it for process analysis and reporting.

AVEVA Historian combines the power and flexibility of Microsoft SQL Server with high speed acquisition and efficient data compression to store time-series data.

With AVEVA Historian, you can:

- **Acquire data from a various sources**
  Historian's Data Acquisition subsystem offers high-speed data capture to acquire plant data from AVEVA I/O Servers, DAServers, InTouch HMI software, Application Server, and other devices.

  AVEVA Historian Historian optimally acquires and stores analog, discrete, and string data. In addition, the SuiteLink protocol used by AVEVA I/O Servers and DAServers provides time and quality stamps as data is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

added to the system.

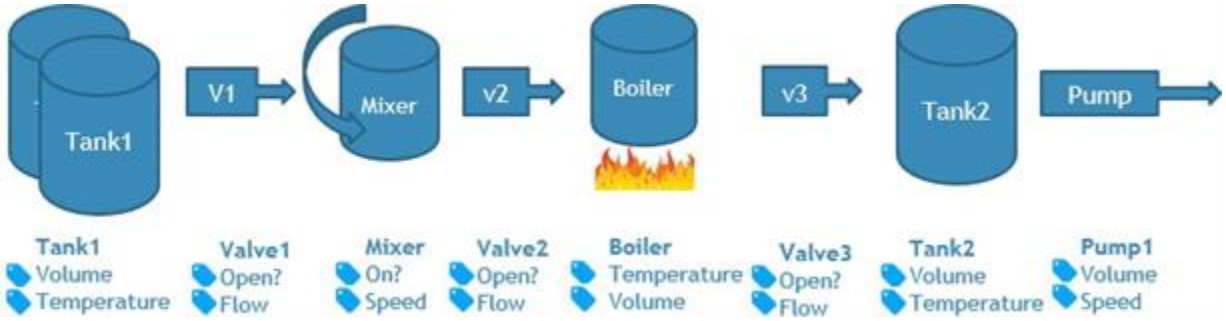
- **Store lots of data efficiently**
  The Storage subsystem compresses the data for maximum storage efficiency. In addition to your plant's data, AVEVA Historian stores alarms and events, summary data, configurations, security information, backups, and system monitoring data.
- **Retrieve data when you want it**
  The Retrieval subsystem responds to SQL requests for plant data and allows you to retrieve large amounts of data very quickly. It also supports REST and SDK data retrieval.
- **Replicate your stored data for security**
  The Replication subsystem replicates tags' values to other AVEVA Historian servers with high fidelity or calculates and replicates summaries of those values.
- **Query data using time domain extensions to SQL**
  AVEVA Historian builds on the capabilities of the SQL query language by supporting time-series data and controlling the resolution of returned data in SQL (for example, to give an evenly spaced sampling of data over a period of time).
- **Create, save, and share graphical content**
  AVEVA Historian includes a tool called Insight that lets you quickly search for data and generate clean, clear graphical content that you can save and share.

Process data: About tags and values

AVEVA Historian acquires and stores process data, which is any information related to successfully running a process. That data is stored as *tags*.

The term "tag" originally referred to physical label on a mechanical part or device on the plant floor. Each tag identifies the corresponding device to AVEVA Historian. As each device sends values to the historian, they are recorded by tag.

For example, a boiler might have two tags – one for the temperature gauge and one for the volume meter.



Using process data to answer your business questions

Historian stores and retrieves your process data in a way that allows you bring clarity to issues related to your business.

- **Real-time data**
  Historian stores data as it comes in from the plant floor.

  Real-time data answers questions like "What's the temperature of that tank right now?"

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Historical data**
  Historian can accept historical information from other systems and retains data captured in real time to use for historical reference.

  Historical data helps to answer questions like "What was the value of this tag every second last Monday?"
- **Summary data**
  Historian summarizes certain data to help answer with big-picture questions like "What is the average number of bags produced each week?"
- **Event data**
  Historian records process alarms and events as they happen. This data is used to find answers about when things happened; for example, "How often did that boiler trip last month?"
- **Configuration data**
  This data describes your system's configuration and answers questions like "What types of I/O Servers am I using?"

Process data analysis can help improve performance, enhance quality, and reduce costs.

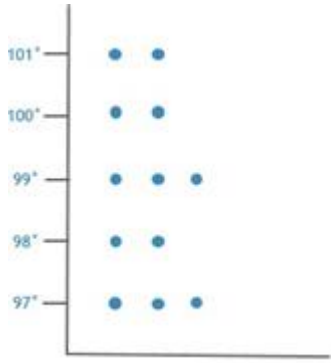
For more information about defining and using tags, see Defining Tags in the AVEVA Historian Administration Guide.

Three-dimensional data: Value, time, and quality (VTQ)

Each time Historian records a data value, it also records a corresponding timestamp and data quality rating. Together, these three things – value, time, and quality – are called a "VTQ".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Values alone: Only somewhat helpful



For example, suppose there is a tag named "tank1.temp" that measures the temperature of a tank in the plant.

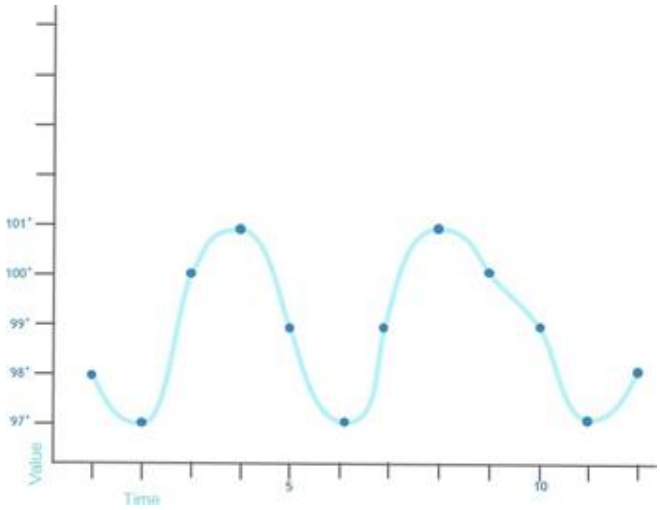
The tank's sensor would send periodic temperatures to Historian – 97 degrees, 98 degrees, 102 degrees, 108 degrees, etc.

These mean little unless you know when the temperatures were taken, so each value also has a date/time stamp.

Value + Time

Recording each value with a timestamp allows you to see trends, pinpoint process errors, etc.

Historian tracks when a record is sent by the device and when it is received by Historian. This helps to clarify the information if there is a data lag, or if values are added or updated later.

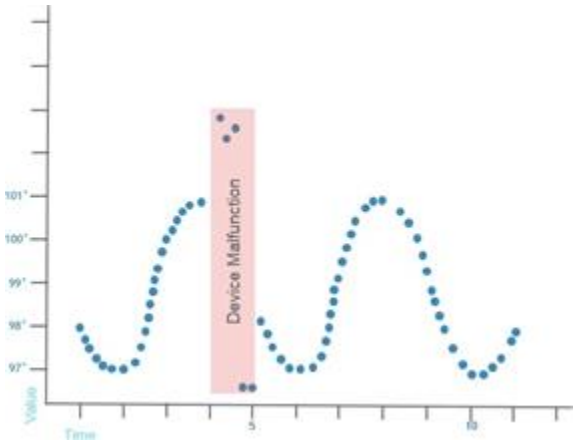


Value + Time + Quality

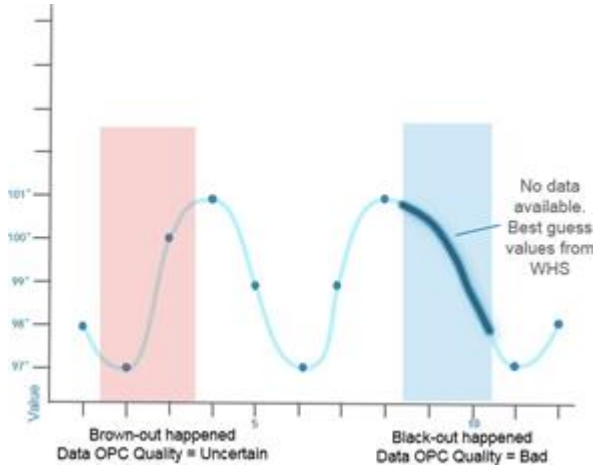
And because errors can occur – from minor mechanical hiccups to major area-wide blackouts – Historian also records a data quality indicator for each record.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If something happens that may affect the data quality, the quality indicator reflects that. That way, you can know if the quality less than optimal for some records, and use that information to report and as accurately as possible.



Where there are gaps, Historian can provide a best guess of what the values were.



Types of tags

AVEVA Historian can handle a wide range of data by supporting these tag types:

- **Analog**
  Measures a continuous physical quantity, such as a tank's volume or a boiler's temperature.

- **Discrete**
  Records one of two states for the tag. For example: on/off, open/closed, jam/cleared.

- **String**
  Captures a text expression--with no special format--that is treated as a single data item. A

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

string tag could be used to capture the state of a machine; for example: "started", "stopped", "jammed", or "cleared".

- **Event**
  Records an instance when a tag meets a preset requirement. For example, a process event tag can let you know when a batch number changes.

- **System**
  Reflects a predefined system variable. System tags are used to collect the system's performance data. AVEVA Historian system tags have a "Sys" prefix (for example, SysTimeSec).

- **Analog summary**
  Reflects summarized data (minimum, maximum, average, and so on) that is configured to be replicated from one historian to another.

- **State summary**
  Reflects summarized data (minimum time in state, maximum time in state, average time in state, and so on) that is configured to be replicated from one historian to another, or stored locally.

You can configure analog, discrete, string, and legacy history event tags through the Operations Control Management Console (OCMC). For more information, see Viewing and Configuring Tags in the *AVEVA Historian Administration Guide*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Data acquisition: Getting data into Historian

Historian acquires and processes data several times faster than a traditional relational database.

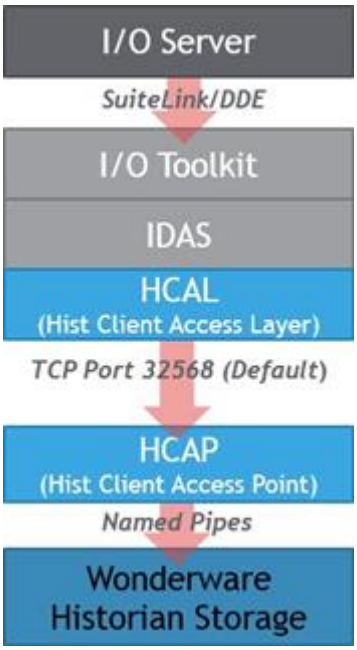
When data is acquired, Historian attaches a timestamp and quality stamp to the value before committing the record to storage.

Historian acquires both original data and revision data. Historian tracks any modifications to the data, and tracks the means by which it was modified.

Sources of data

Historian can accept data from a number of sources. The most typical scenario is data acquisition from an I/O server.

Data acquisition from I/O servers



An I/O server provides data from plant floor devices to AVEVA Historian using the SuiteLink or Dynamic Data Exchange (DDE) protocol.

First, the I/O server collects data values from programmable logic controllers (PLCs), Remote Telemetry Units (RTUs), and similar devices on the factory floor.

Then, the I/O server uses the SuiteLink or DDE protocol to time and quality stamp each data value it collect.

Next, it passes the data values through the Data Acquisition subsystem, or IDAS. IDAS seamlessly handles data values,regardless of their time.

Note: The SuiteLink protocol can be used to collect data from an I/O server on the same or different computer than the IDAS instance. The DDE protocol can be used only when the I/O server is on the same computer as the IDAS instance.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	For each data value acquired by IDAS, the timestamp, value, and quality are attached.

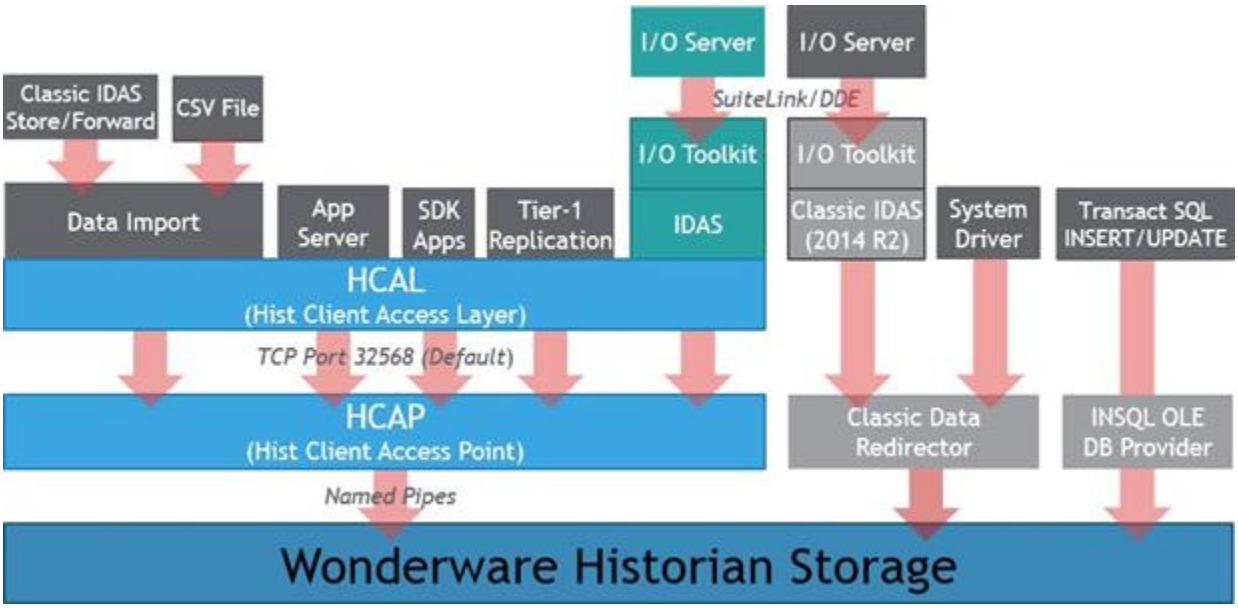
Then the values are sent through the Historian Client Access Layer (HCAL) to a Historian Client Access Point (HCAP) on the Historian server, and then to storage. HCAL is a client-side software layer that provides programmatic access to storage, retrieval, and system configuration functionality in the AVEVA Historian.

Historian accepts and historizes each data value according to the storage rules for the tag to which the data value belongs.

For more details, see Configuring Data Acquisition in the *AVEVA Historian Administration Guide*.

Other data acquisition options

As this diagram illustrates, AVEVA Historian can accept data from a range of sources.



In addition to I/O servers, Historian can acquire data from these sources:

- **TransactSQL INSERT and UPDATE statements**
  You can insert or update history data in the AVEVA Historian extension tables using Transact-SQL INSERT and UPDATE statements.

  For more information, see Importing, Inserting, or Updating History Data in the *AVEVA Historian Administration Guide*.
- **CSV and LGH files**
  Using the Historian Data Importer utility (aahImport), you can add history data from a file to AVEVA Historian. This utility reads data from InTouch history (LGH) files or comma-separated value (CSV) files, and then sends the data to the Historian server via HCAL.

  Imported data is integrated with data currently stored in history blocks, providing you with seamless access to all your data.

  For more information, see Importing, Inserting, or Updating History Data in the *AVEVA Historian Administration Guide*.
- **App Server, custom SDK client applications, tier-1 replication, and other sources**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

These sources are also able to use HCAL to send data to Historian.

- **AVEVA Historian itself**
  Configuration data comes from the Historian itself.

About AVEVA Historian data security

AVEVA Historian provides two ways to authorize access:

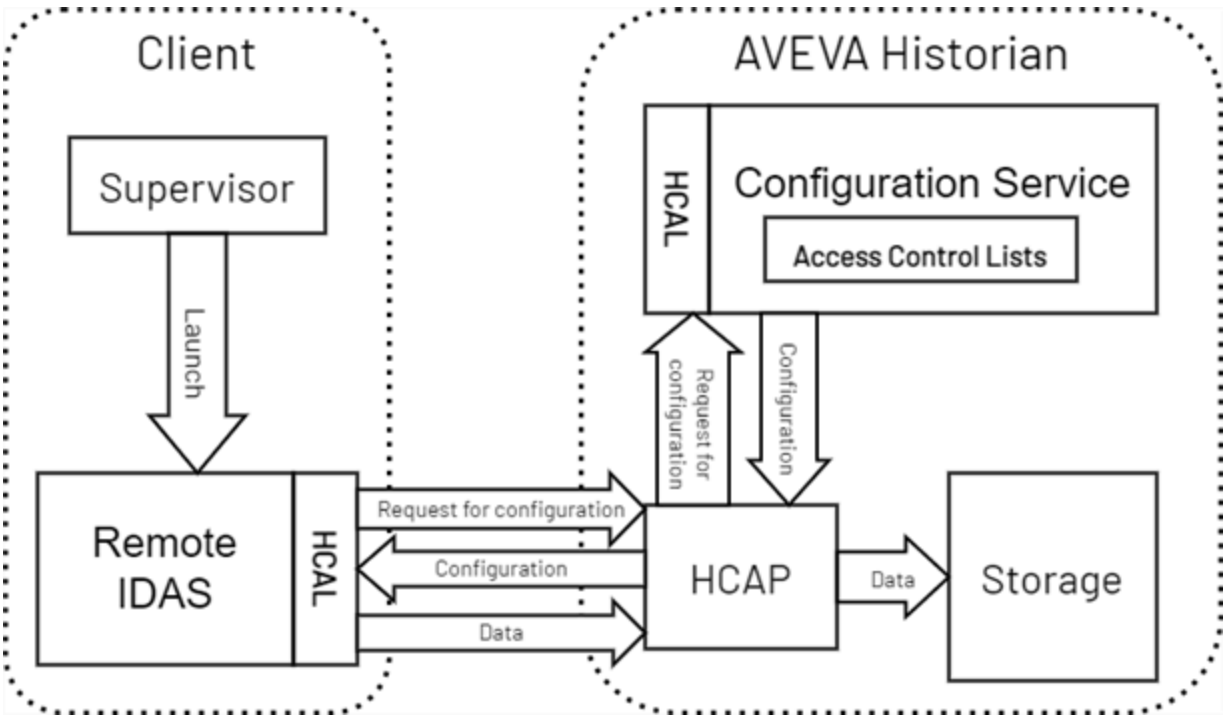
- **Integrated security.** IDAS computers in the same domain as the historian can be configured with integrated security. Using this model, all users and computers that access historian data are assigned membership to one of the following user groups:
  - Administrators (aaAdministrators)
  - Power Users (aaPowerUsers)
  - Replication Users (aaReplicationUsers)
  - Users (aaUsers)
- **Workgroup security.** IDAS computers outside of the historian's domain can use username and password as security. This username and password must match a local user on the server node, and must be provided in the format `<server node hostname>/<user name>`.

  When the IDAS is configured with this type of security, an authentication token and is defined and forwarded to the remote IDAS computer. Each time the remote computer accesses the historian, it will use the token and the historian will use it to authenticate the remote computer before allowing access.

Data can be passed from any computer that's a member of the historian's Power User or Administrator group. (Computers must be on the same domain as the historian.)

When data is ready to be sent from a remote computer, the AVEVA Historian pushes configuration information, including ACLs (access control lists) that define access permissions, to HCAP on the client computer. HCAP launches IDAS on the remote computer and data is sent through HCAL to the historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



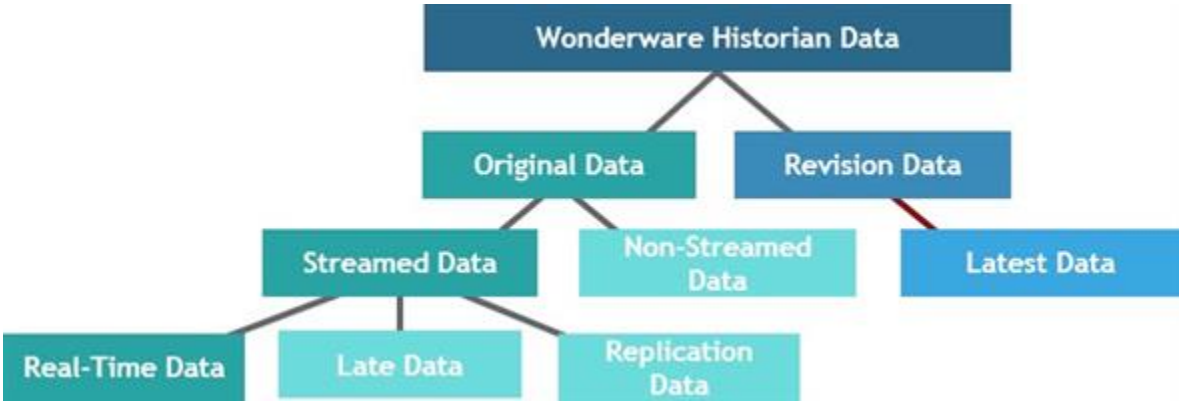
Store-and-forward safeguards

Historian uses a store-and-forward method to protect against data loss if communication is interrupted between the data source and Historian.

Systems using the Data Acquisition Subsystem (IDAS) or Historian Client Access Layer (HCAL) to send data to Historian are able to use the store-and-forward method in case of communication breaks. If the data source loses communication with Historian, the source stores the collected data until communication is reestablished. Then, it forwards the stored data to Historian.

Data categories

AVEVA Historian is able to process and store data in a variety of ways. It categorizes each data record by type to provide a consistent framework for data operations. Each category of data has a separate set of characteristics and is handled differently by the historian.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Original versus revision data

Data acquired by AVEVA Historian can be categorized as:

- **Original data** is the data received from the data source originally—that is, for the first time. For example, a real-time stream of data from an I/O server represents original data. Usually the original data arrives to a Historian in high data volumes for many tags with timestamps close to each other.
- **Revision data**, by contrast, is data that corrects or appends original data. Revision data operations are performed on a per-tag basis and typically have far lower volumes than original data.

Streamed versus non-streamed data

Original data can be streamed or non-streamed.

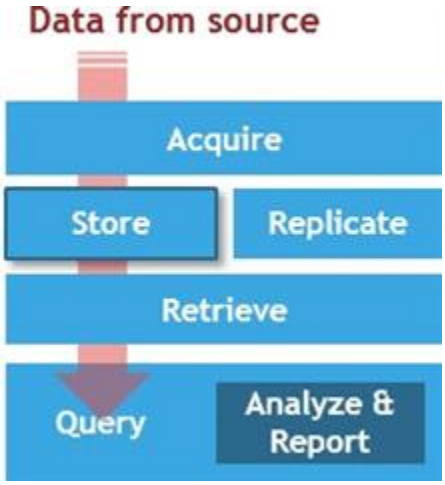
- **Streamed:** If the data source is able to enforce time order for its output, the data is streamed data. Streamed data has three subtypes:
  - Real-time data is in time order, where the timestamp is in the past relative to the current AVEVA Historian time.
  - Late data is in time order, where the timestamp is far in the past compared to the current AVEVA Historian time.
  - Replication data is data that has been replicated from a tier-1 historian to a tier-2 historian.
- **Non-streamed:** If the time order is not enforced, the data is non-streamed.

Data storage: Preserving huge amounts of data over time

Historian, through its Storage subsystem, saves plant data from various sources to disk.

Historian's efficient storage model includes:

- **A range of storage modes**
  Historian's storage modes allows storage of every meaningful value and extrapolation of all other values at retrieval time.
- **History blocks and partitions**
  Historian efficiently warehouses huge amounts of data using a compact storage format that is optimized for time-series data.
- **Auto-summarization**
  Historian calculates and stores automatic summary records as real-time analog tag values. These auto-summaries are used provide fast, seamless data retrieval, no matter what the granularity.



For information about configuring the Storage subsystem, see Managing Data Storage in the *AVEVA Historian*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

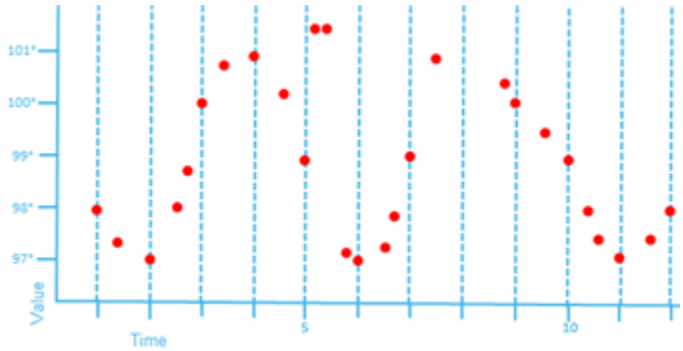
Administration Guide.

Storage modes

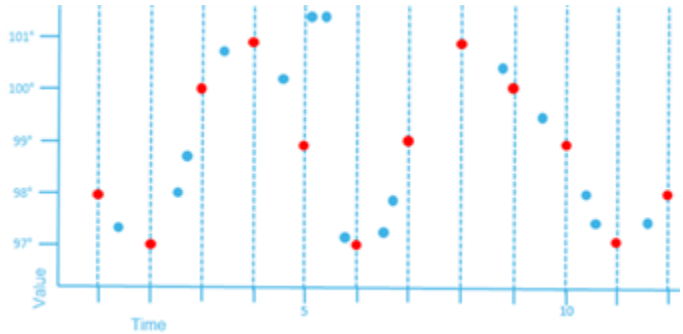
Depending on a tag's definition, Historian uses one of these storage modes to retain the values received for that tag:

- **No storage -** No values are stored.
- **Forced storage** - All collected values are stored.

  For comparison's sake, this is what forced storage looks like. The red dots represent collected values. All of these values are stored by Historian.

- **Cyclic storage** - Only values that occur at a specified time interval are stored. Using the same collected values as shown above, cyclic storage retains only the values represented by red dots.

- **Delta storage** - Only changed values are stored.

Types of delta storage

Delta storage, as a rule, requires Historian to store any value that is different than the previously received value. Time or deadband rules can be applied for delta storage to further constrain what values are stored.

These are all types of delta storage:

- **Time-enforced delta storage** -- Any changed value is recorded. Additionally, a record must be stored after a given amount of time. This storage mode is used by the Historian SDK.
- **Time deadband delta storage** -- Only changes outside of a particular time deadband are stored.
- **Value deadband delta storage** -- Only changes outside of a particular value deadband are stored.
- **Rate of change (swinging door) deadband storage** -- Only changes outside of a particular rate-of-change

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

(swinging door) deadband are stored.

Measuring change with deadbands

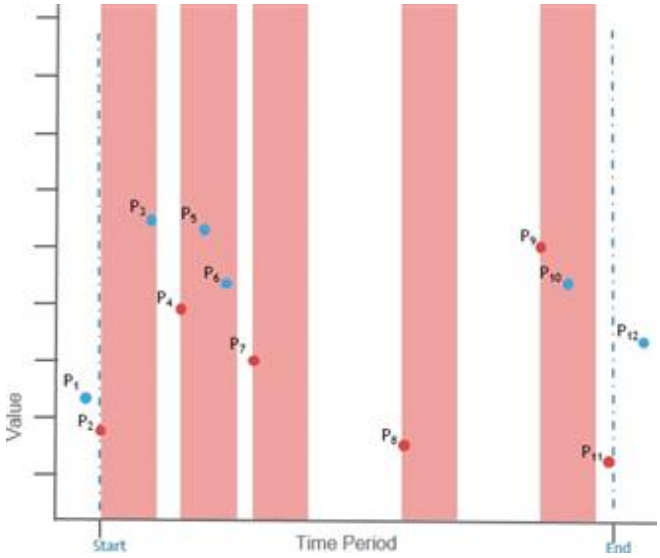
Delta storage retains only those values that have significantly changed from the previously stored value.

For example, if you had a discrete (binary) tag that reflected the state of a power switch, you may not want to record every time the system checks to see that it is switched on. You might really be interested only in when it switches off when it is supposed to be running, and when it gets switched on again.

For analog (numeric) tags, you may only care only about large changes, but not tiny ones. Or, you may want a snapshot of values at certain intervals, and not every one that is reported. You can filter out extraneous value with deadbands.

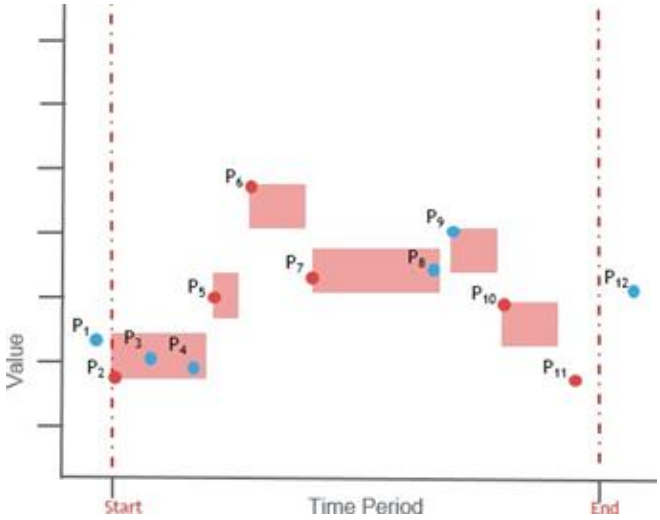
A **time deadband** is a time filter. It marks the minimum time (milliseconds) between stored values for a single tag. Any value changes that occur within the time deadband are not stored.

For example, these red and blue points are all the values reported for a certain tag. The orange bars represent the time deadband, which starts over with every reported value. Only the red points (P2, P4, P7, P8, P9, P11) are stored. The other points are excluded because they fall within a deadband or outside of the time period.



A **value deadband** is a filter that marks the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored.

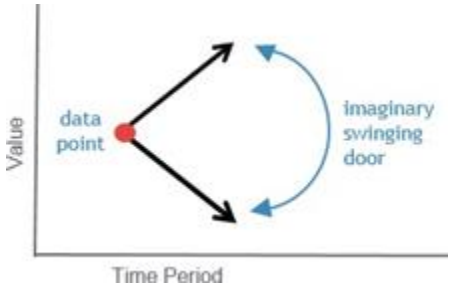
Here, the orange bars represent the value deadband, which starts over with every reported value. Only the red points (P2, P5, P6, P7, P10, P11) are stored. The other points are not stored because they fall within a deadband or outside of the time period. P9 is not stored because P8 was discarded and it is within the percentage deviation.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A **swinging-door deadband** marks a rate of change deadband, based on changes in the slope of the received values.

For example, specifying a swinging door deadband value of 10 percent means that values are stored if the percentage change in slope of the consecutive data values exceeds 10 percent.



For more information on delta storage modes, see About Delta Storage Mode in the *AVEVA Historian Administration Guide*.

History blocks and partitions

AVEVA Historian stores data in history blocks. History blocks use a proprietary file format and are essentially subfolders of the main historian storage folder.

Although the tag values are stored in history blocks on disk, the values appear to be saved to tables in the Runtime database.

Each history block stores all data for a specified duration. The default history block duration is one day, but may be as little as one hour. When there is data to be stored in that time interval, Historian creates a new history block for that data. For example, if a history block is defined for a day's worth of data, when it receives the first data value for the second day, Historian creates a new one-day history block and places the corresponding data into the new block.

As data is acquired, the size of these history blocks grows on a continual basis, being limited only by the size of the hard disk on which the historian resides.

If the historian was not running for some time, or if a history block is deleted, for a certain time period, there may be a gap in the sequence of history blocks -- also known as a *block gap*.

History block formats are specially optimized for storing time-series data, while general-purpose database management systems typically are not.

Compact storage formats reduce the storage space requirements than would be required in a general-purpose database. Upon retrieval, historical data is presented by the AVEVA Historian OLE DB provider as if it were stored in SQL Server tables.

Historian partitions

Historian organizes history blocks within partitions. As real-time data arrives, Historian stores it in history blocks located in the main data partition. At the same time, Historian automatically computes and records a corresponding hourly summary for each analog tag value received. The auto-summary values are stored in auto-summary history blocks within the auto-summary partition.

For more information on history blocks and partitions, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.

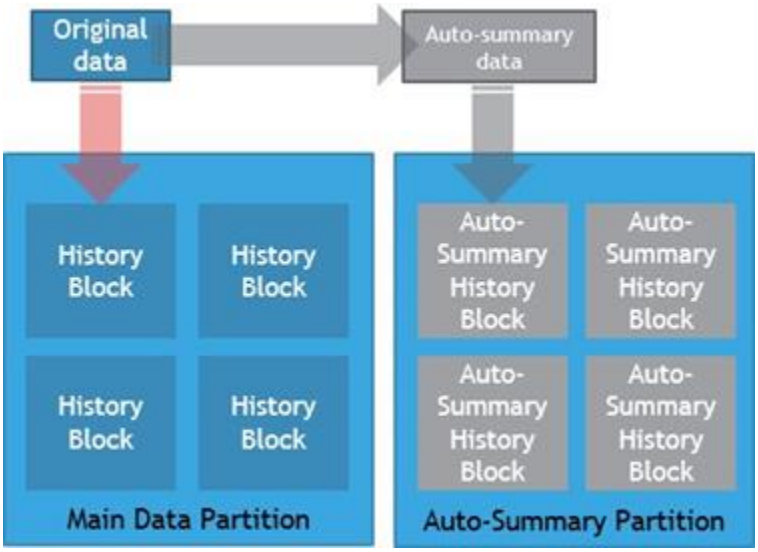
Auto-summarization

For every analog tag in the system, AVEVA Historian creates a local replication entity and a one-hour summary

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tag. As values arrive for an analog tag, Historian automatically computes and records a summary.

Auto-summary values are stored in their own history blocks within the auto-summary partition.



With auto-summarization, Historian can quickly and efficiently retrieve large-volume data for a long duration, even months or years.

Note: The auto-summarization feature is enabled by default, but can be disabled.

Data retrieval: Transforming data into information

AVEVA Historian can process SQL-based queries from any number of client applications, including: AVEVA Insight, AVEVA Historian Client, and ad hoc SQL query tools. It can also process queries via the OData interface and from SDK client applications.

When Historian receives a request for data, it performs the following steps:

1. Locates the requested data.
   - Historized process data is stored in history blocks.
   - Configuration data is stored in SQL Server database tables.
   - Replication data is stored in history blocks.
2. Apply a retrieval mode to the data.

   Because of the enormous amount of data potentially associated with a facility, Historian provides several retrieval modes that help interpret that data, turning the collected data into usable information.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

3. Returns the results to the client application.

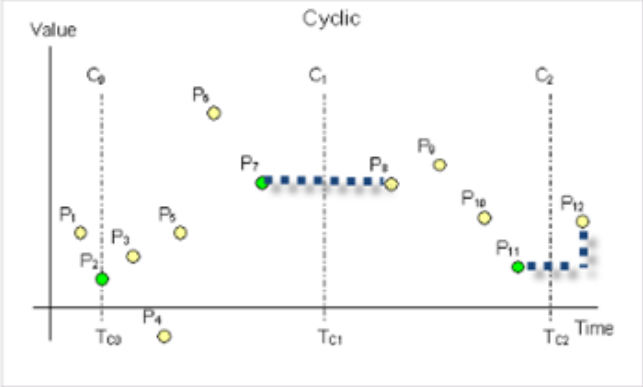
For more information about retrieving data from AVEVA Historian, see the *AVEVA Historian Retrieval Guide*.

Retrieval modes

Historian can acquire and store huge amounts of data and allows you to choose from among several retrieval modes to view and interpret the data you need.

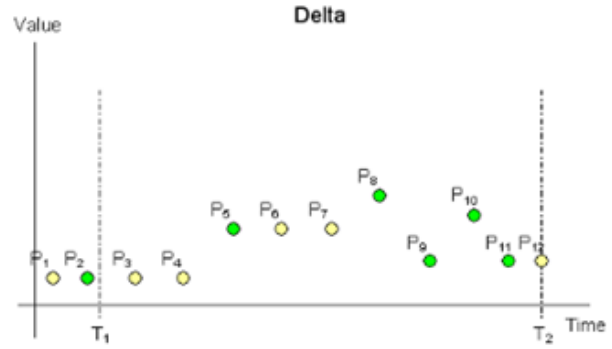
Cyclic

Retrieves one value per cycle. Whatever the value is when the cycle begins.



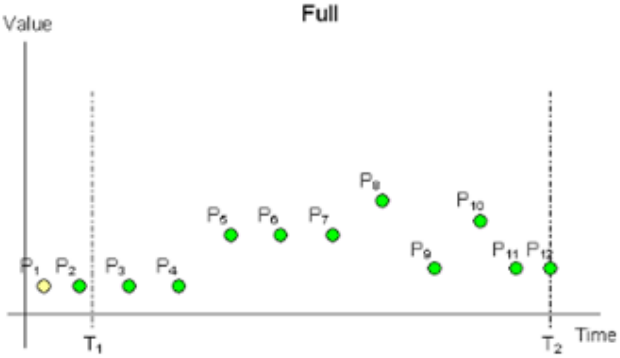
Delta

Retrieves a value each time the value changes from the previous value. For example, if the value of "4" followed an earlier value of "4", it would not be retrieved. But if "4’" followed "3", it would.



Full

Every value within a time period is retrieved.

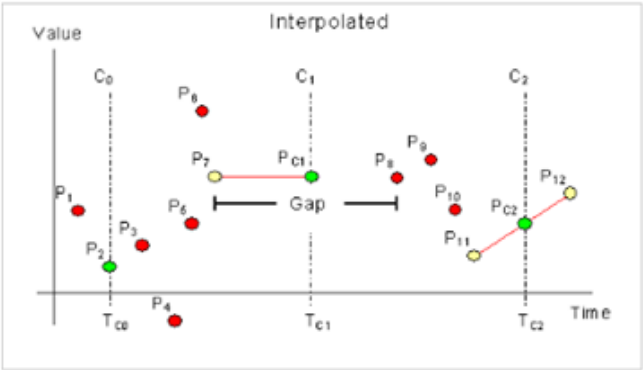


© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Interpolated

Based on values before and after a certain point in time, Historian estimates the value for that time.

In this example, P2 is located exactly at the query start time. Because of this, P2 is returned at that time without need for interpolation. At the following cycle boundary, point PC1 is returned, which is the NULL value represented by P7 shifted forward to time TC1. At the last cycle boundary, point PC2 is returned, which has been interpolated using points P11 and P12.

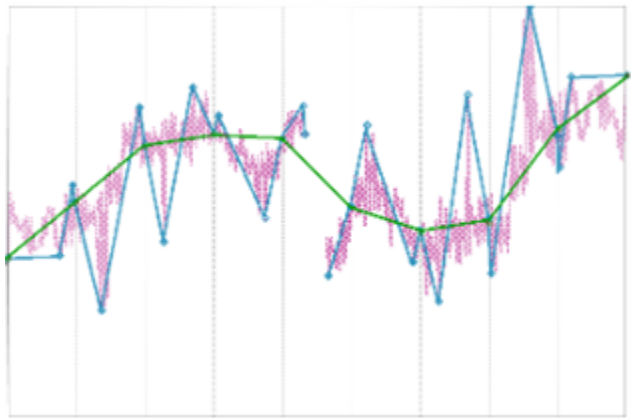


Best Fit

"Best fit" retrieval allows for a compromise between delta retrieval and cyclic retrieval.

Delta retrieval can accurately represent a process over a long period of time, but requires a large number of data values. Cyclic retrieval is much more efficient, but less accurate, because of fewer values.

Best fit provides faster retrieval, like cyclic retrieval, plus the better representation, like delta retrieval.

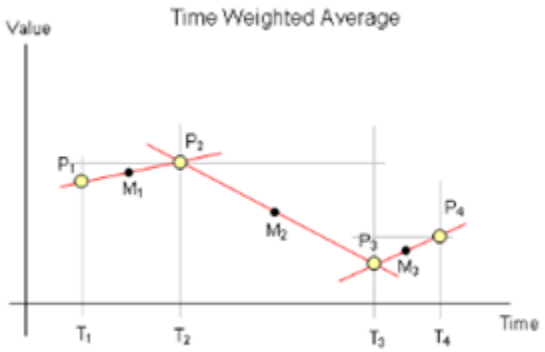


Average

Uses a time-weighted average algorithm to calculate the value for each retrieval cycle.

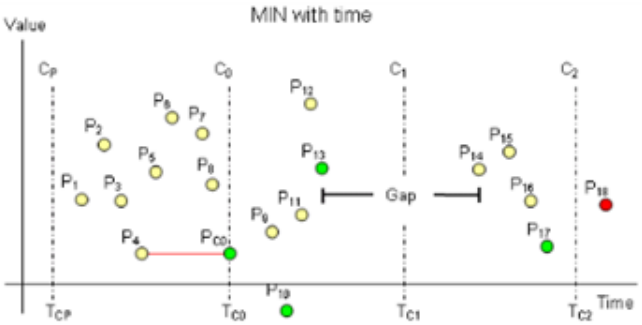
For the following data values of a tag that uses linear interpolation, the time-weighted average is computed as:

Average = (((P1 + P2) / 2) x (T2 - T1)) + (((P2 + P3) / 2) x (T3 - T2)) + (((P3 + P4) / 2) x (T4 - T3)) / (T4 - T1)



Minimum

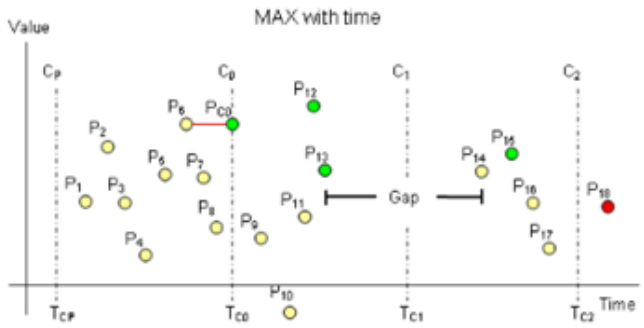
Returns the minimum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. If there are NULL values in the cycle, NULL is returned for that cycle.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Maximum

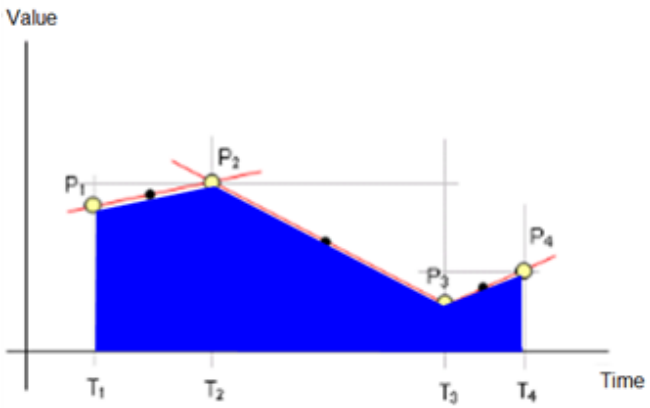
Similarly, this mode returns the maximum value of actual data for the retrieval cycle.



Integral

Calculates the values at retrieval cycle boundaries by integrating the graph described by the points stored for the tag. In other words, it works much like average retrieval, but it additionally applies a scaling factor. This retrieval mode is useful for calculating volume for a particular tag (for example, gallons of water flowing through a valve over a certain period).

Integral retrieval works with analog tags only. For all other tags, normal cyclic results are returned.



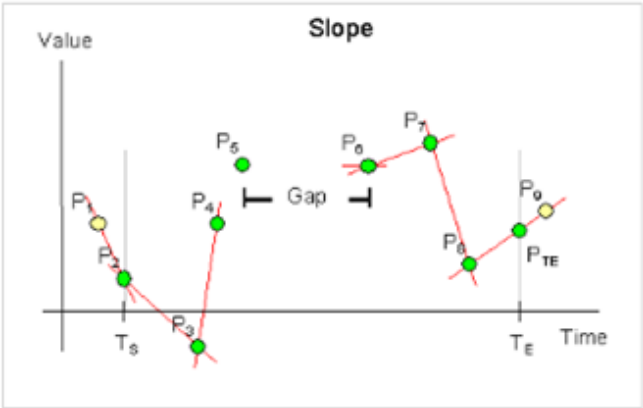
Slope

Returns the slope of a line drawn through a given point and the point immediately before it, thus expressing the rate at which values change.

For example, two points P1 and P2 occur at times T1 and T2. The slope is calculated as:

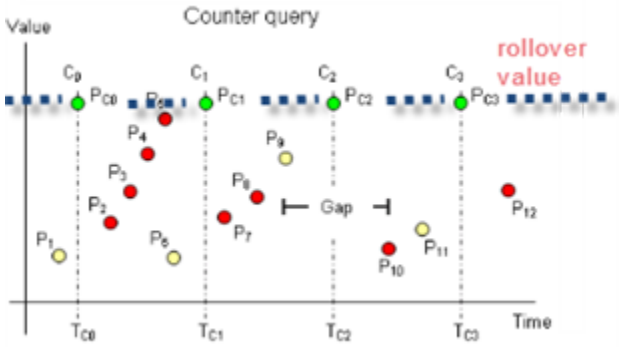
(P2 - P1) / (T2 - T1)

The difference between T1 and T2 is measured in seconds, so the returned value represents the change in engineering units per second.



Counter

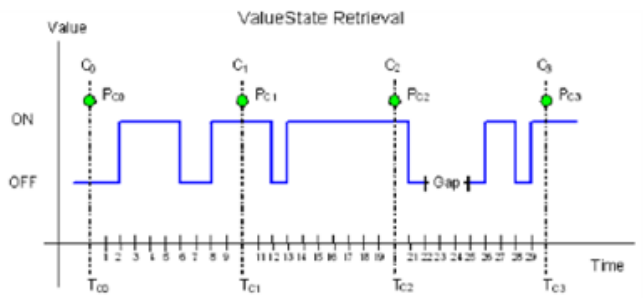
The change in a tag's value from the beginning to the end of the period, factoring in any rollover value for the counter. This retrieval mode is useful for determining how much of an item was produced during a particular time period.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

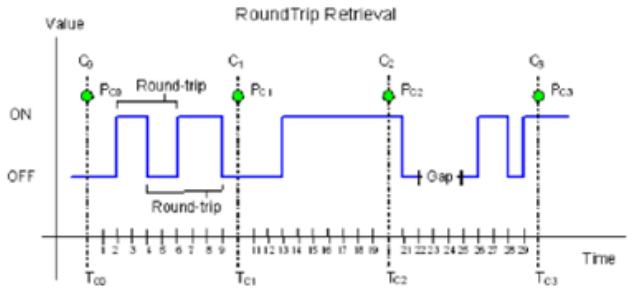
ValueState

Returns information on how long a tag has been in a particular value state during each retrieval cycle. That is, a time-in-state calculation is applied to the tag value.



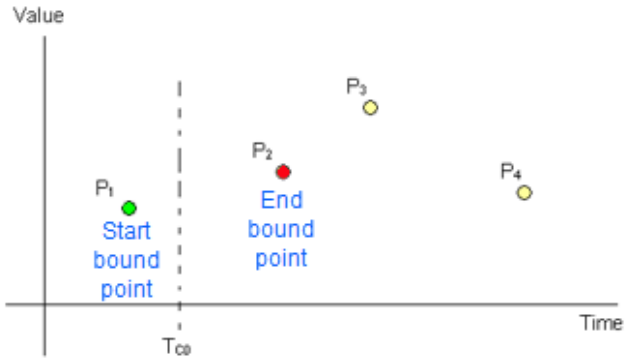
RoundTrip

Like ValueState retrieval, this mode uses state occurrences within a period for its calculations. RoundTrip retrieval calculates the time between consecutive leading edges of the same state.



Bound Value

Retrieves either the start bound point or the end bound point for a requested point in time. For a start bound point, Historian retrieves the first value on or before the requested date/time. For an end bound point, Historian retrieves the first value after the requested date/time.



Data query tools: Accessing your data

To report on the data you have stored in AVEVA Historian, you can use:

- AVEVA Insight, a tool included with AVEVA Historian
- Transact-SQL queries
- Any other query tool that can access SQL data sources

AVEVA Historian is part of a client/server architecture that supports desktop client applications, while ensuring the integrity and security of data on the server. This architecture provides common access to time-series data and associated configuration, event,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

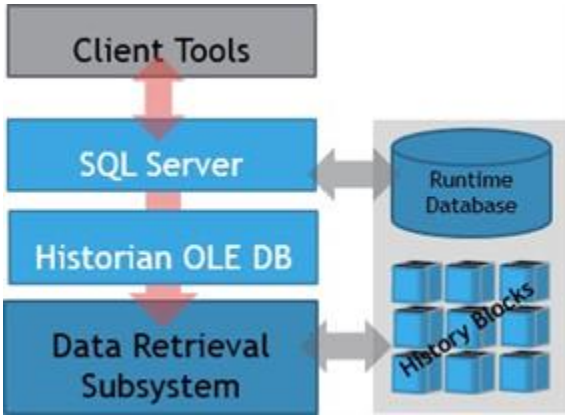
and business data.

The computing power of both the client and the server is exploited by optimizing processor intensive operations on the server and minimizing data to be transmitted on the network to improve system performance.

Microsoft SQL Server acts as the gateway for accessing any type of information in the historian. Historian uses Microsoft linked server technology to plug in its own OLE DB provider. Because of this, any client application that can connect to Microsoft SQL Server can also connect to *AVEVA Historian*.

For users with client applications, it seems that queries are made to the Runtime database. That database is, in fact, the logical interface to the data.

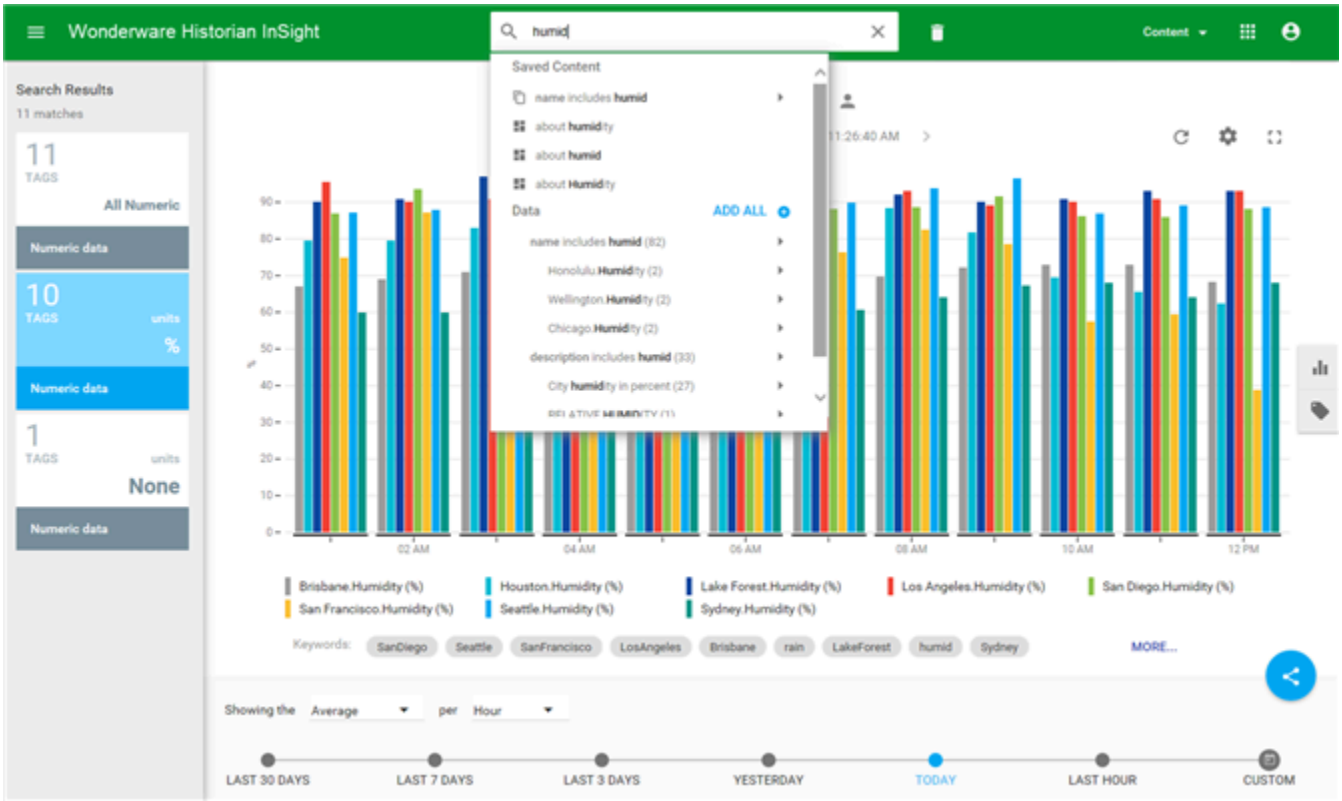
When it receives a request for data, the Retrieval subsystem retrieves the historized data from history blocks.



Querying with AVEVA Insight

AVEVA Insight (included with AVEVA Historian) is a search-based tool that lets you quickly turn your data into easy-to-read charts.

With Insight, you can type the name -- or even a part of a name -- for the tags you want to analyze. Then you can choose the chart type and timeframe to report on. Insight also lets you save and share your data.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For more information on using Insight, see the online help.

Querying with Transact-SQL

Historian can handle traditional SQL queries. For example:

SQL Query

```
select Value as Step,
Label as Label,
StateTimeAvgContained as Avg,
StateTimeMinContained as Fastest,
StateTimeMax as Longest
from StateSummaryHistory
where TagName = 'Filler1.Step'
and StartDateTime >= '2016-05-01'
and EndDateTime <= '2016-05-02'
and wwCycleCount = 1
and wwRetrievalMode='cyclic'
```

Result

Step	Label	Avg	Fastest	Longest
1	Load	1029	1020	1038
2	Open	1009	1006	1012
3	Fill	5710	5612	5842
4	Settle	2022	2018	2036
5	Close	4558	1074	5816
6	Seal	1067	1058	1176
7	Release	1120	994	1140

Data replication: Delivering information to people who need it

AVEVA Historian allows you to replicate data. That is, a tag that was collected by one historian can be replicated and stored on another historian.

AVEVA Historian supports two types of replication:

- **Simple replication**
  You can replicate tag data directly using simple replication, where the tag information is replicated directly from one historian to the other. For simple replication, every value for a tag is copied.
- **Summary replication**
  You can also set up summary tags that receive a summarized version of the tag data.

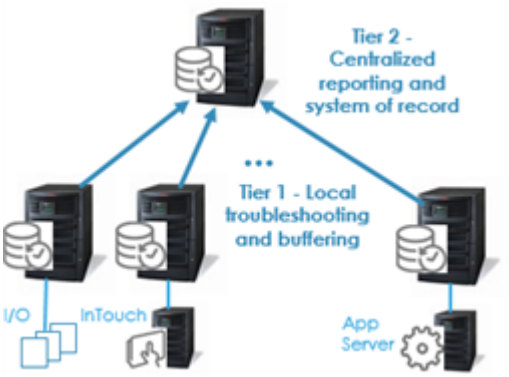
  Summary replication is useful to minimize bandwidth requirements. This is important, for example, when an application uses a wide-area network or connecting via satellite/cellular.

  Note: Summary replication happens between tier 1 and tier 2 only. All data replication to tier 3 and beyond is simple replication.

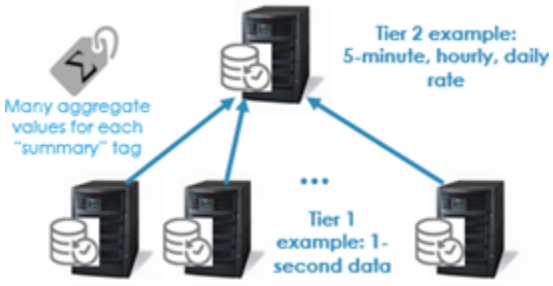
Setting up replication servers is useful for several circumstances. For example:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

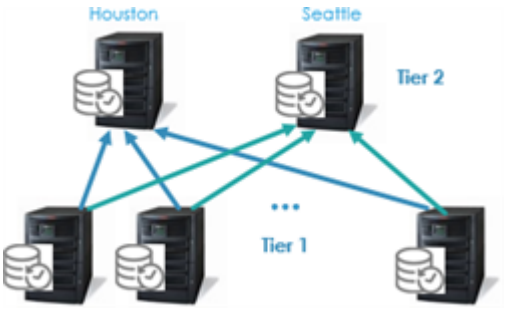
- **Centralized data**
  When you want to replicate data from multiple individual historians and sent it to a single centralized historian.



- **Replicated summaries**
  When you want summary data to be available from a centralized historian.

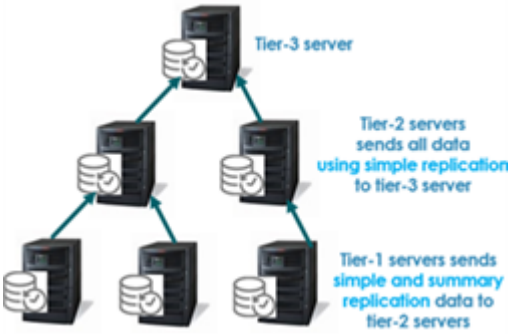


- **Many-to-many**
  When you want to set up a many-to-many relationship between tiers of historians.

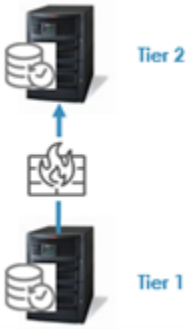


- **Cloud**
  When you want data available from the cloud.

- **Multiple levels**
  When you want to replicate data to multiple tiers of historians.



- **Across firewall**
  When your facility has a firewall and you want users on both sides of the firewall to have access to the same data.



For more information about setting up and using replication, see Managing and Configuring Replication in the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVEVA Historian Administration Guide.

Replication tiers

When you replicate data, it creates a tiered relationship between the historians.

That is, the tier-1 historian send its replicated data to a tier-2 historian.

AVEVA Historian can replicate process data as well as alarms and events.

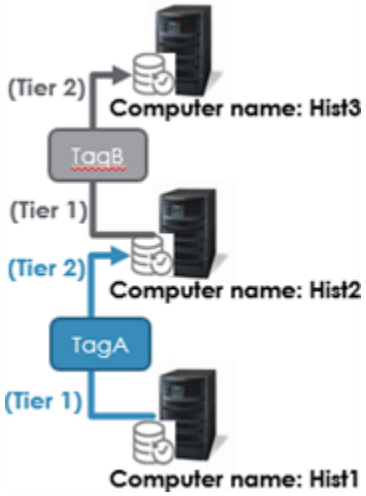
A historian can act as a tier-1 and a tier-2 historian simultaneously.

Here's a typical scenario for a tiered historian:

- TagA is collected by Hist1, a historian. It is replicated and stored on Hist2.

  In this relationship, Hist1 is a tier-1 historian and Hist2 is a tier-2 historian.
- Concurrently, TagB is collected by Hist2. It is replicated and stored on Hist3.

  In this relationship, Hist2 is a tier-1 historian and Hist3 is a tier-2 historian.

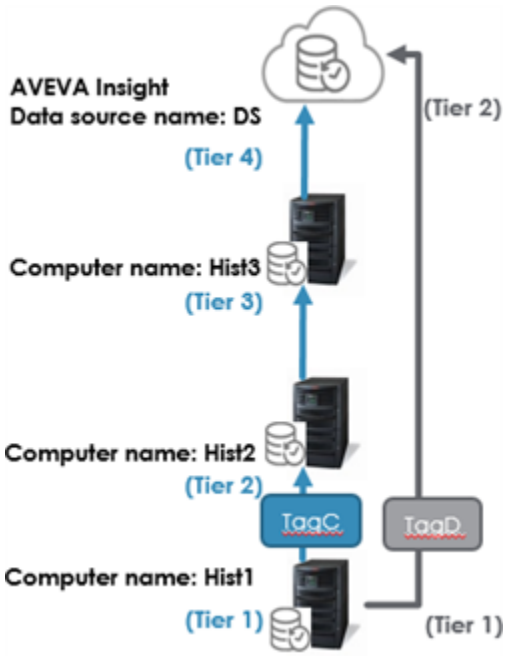


Historian supports multi-tier replication. Data originating at tier 1 can be replicated to tier 2, then again to tier 3, and so on.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

This diagram illustrates two relationships between historians.

- TagC is replicated from Hist1, a tier-l historian, to Hist2, a tier-2 historian. Hist2 replicates the tag to Hist3, a tier-3 historian. Hist3 replicates the tag once again to AVEVA Insight (tier 4 in this scenario).
- TagD is also replicated from Hist1, but this time directly to AVEVA Insight. In this case, Hist1 is the tier-1 historian and Insight is the tier-2 historian.



The following tables show how the replicated data is named as it is replicated in these two scenarios. These examples are based on the default naming scheme.

TagC replicated across 4 tiers (Hist1, Hist2, Hist3, Insight)

Tier	Computer name	Tag name	Summary tag name
Tier 1	Hist1	TagC	TagC.1M
Tier 2	Hist2	Hist1.TagC	Hist1.TagC.1M
Tier 3	Hist3	Hist1.TagC	Hist1.TagC.1M (see note below)
Tier 4	Insight (and data source = DS)	DS.Hist1.TagC	DS.Hist1.TagC.1M

Note: Summary replication happens between tier 1 and tier 2 only. All data replication to tier 3 and beyond is simple replication.

TagD replicated from Hist1 to Insight

Tier	Computer name	Tag name	Summary tag name
Tier 1	Hist1	TagD	TagD.1M

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tier 2	Insight (and data source = DS)	DS.TagD	DS.TagD.1M

Note: Before version 17.3.100, replication to AVEVA Insight used the same default naming as any other tier 2 and still included the "DS" prefix (where "DS" is the name of the data source). For example, consider how "TagC" was replicated to Insight before and since version 17.3.100:
- Before 17.3.100: TagC was replicate to Insight as "DS.Hist1.TagC".
- Since 17.3.100: TagC is replicated to Insight as "DS.TagC".

Historian Database Guide

Welcome

This guide describes the database model of the AVEVA Historian system. Each database entity is described, and the relationships between the entities are defined. It is very important that you understand these data structures and relationships to effectively query AVEVA Historian and build productive client applications that interact with it.

Table Categories

There are eight table categories within the AVEVA Historian Runtime database. Tables in a category together facilitate a particular functionality in the historian.

Note: Additional tables and views are provided for backward compatibility support. For more information, see Backward Compatibility Entities.

History Tables and Views

Because normal Microsoft SQL Server functionality cannot handle the storage and retrieval of huge quantities of rapidly changing data, plant data storage and retrieval are made possible by the AVEVA Historian storage subsystem, the history tables, and the retrieval system.

Some of the history tables are implemented as normal SQL Server tables, and the information contained in them is stored in the Runtime database file (Run110Dat.mdf). Others are implemented as a special type of table called a remote table, or extension table. Extension tables do not actually exist in the database, but rather expose data that is stored in special history files (history blocks) on disk using OLE DB technology.

For more information, see the About Data Retrieval in the *AVEVA Historian Retrieval Guide.*

Acquired tag data can be presented in the history tables in four different formats:

- Normal historical format
- "Wide" format
- "Live" format
- Analog/state summary history format

Information about the history blocks is stored in the special HistoryBlock (INSQL.Runtime.dbo.HistoryBlock)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

extension table.

AVEVA Historian also includes several views to make querying from the history tables easier. Instead of specifying the table name using the required four-part syntax (INSQL.Runtime.dbo.<tablename>), you can simply use the view name instead. The history tables and associated views are listed in the following table. (Backward compatibility tables and views are not included.)

Table Name (OLE DB Provider Syntax)	Associated View
History (INSQL.Runtime.dbo.History)	History
WideHistory (INSQL.Runtime.dbo.WideHistory)	(none)
StateWideHistory (INSQL.Runtime.dbo.StateWideHistory)	(none)
Live (INSQL.Runtime.dbo.Live)	Live
HistoryBlock (INSQL.Runtime.dbo.HistoryBlock)	HistoryBlock
AnalogSummaryHistory (INSQL.Runtime.dbo.AnalogSummaryHistory)	AnalogSummaryHistory
StateSummaryHistory (INSQL.Runtime.dbo.StateSummaryHistory)	StateSummaryHistory

The History and Live tables can accommodate a mixture of tag types and should be used for all queries. The vValue column returns a sql_variant for all tag types. The Value column returns a float value for analog and discrete tags and a NULL for string tags. The Value column is included to allow for aggregation and other operations that are not permitted on a sql_variant column.

You can relate these tables to other tables in the AVEVA Historian database.

For more information on each of these tables, see the corresponding table description in this documentation.

Note: The AnalogHistory, DiscreteHistory, StringHistory, AnalogLive, DiscreteLive, and StringLive tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you cannot mix the tag types in the query without doing a UNION.

In SQL Server Management Studio, the extension tables are listed under the INSQL or INSQLD linked servers under the **Server objects** tree item.

History Table Format

The History table presents acquired plant data in a historical format, which is shown as follows:

DateTime	TagName	Value	vValue	Quality	QualityDetail	(continued...)
2017-02-17 15:40:01.000 0000	Temp1	78	78	0	192	(continued...)
2017-02-17	Temp2	79	79	0	192	(continued...)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

15:40:31.000 0000						
2017-02-17 15:41:01.000 0000	Temp3	77	77	0	192	(continued...)
2017-02-17 15:41:31.000 0000	Temp4	80	80	0	192	(continued...)
2017-02-17 15:42:01.000 0000	Temp1	77	77	0	192	(continued...)
2017-02-17 15:42:31.000 0000	Temp2	78	78	0	192	(continued...)
2017-02-17 15:43:01.000 0000	Temp3	76	76	0	192	(continued...)
2017-02-17 15:43:31.000 0000	Temp4	79	79	0	192	(continued...)
2017-02-17 15:44:01.000 0000	Temp1	76	76	0	192	(continued...)
2017-02-17 15:44:31.000 0000	Temp2	77	77	0	192	(continued...)
2017-02-17 15:45:01.000 0000	Temp3	78	78	0	192	(continued...)
2017-02-17 15:45:31.000 0000	Temp4	80	80	0	192	(continued...)

There is one row for a single tag's value for a particular timestamp.

Note: The AnalogHistory, DiscreteHistory, and StringHistory tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you cannot mix the tag types in the query without doing a UNION. The History table, however, can accommodate a mixture of tag types and should be used instead of the AnalogHistory, DiscreteHistory, or StringHistory tables. The Value column returns a float value for analog and discrete tags, a NULL for string tags. The vValue column returns a sql_variant for all tag types.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

"Wide" History Table Format

The WideHistory table contains the same data as the History table, but in a different format. The WideHistory table presents data for one or more tag values for a single timestamp, thus providing a "wide" view of the data. To query for values in the WideHistory table, you must specify the timestamp and one or more tagnames as the column names in the query syntax. The results will contain a column for the timestamp and columns for the value of each specified tag at that timestamp. In the following example, Temp1, Temp2, Temp3, and Temp4 are tagnames:

DateTime	Temp1	Temp2	Temp3	Temp4
02:17:01:03	78	79	77	80
02:17:01:04	77	78	76	79
02:17:01:05	77	78	76	79

Using the History table to perform the same task is much more difficult.

You can also specify search criteria for the values you want to return (for example, where Temp1 > 75). The WideHistory table can only be related to other tables based on the timestamp.

Note: The AnalogWideHistory, DiscreteWideHistory, and StringWideHistory tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you can't mix the tag types in the query. The WideHistory table, however, can accommodate a mixture of tag types and should be used instead of the AnalogWideHistory, DiscreteWideHistory, or StringWideHistory tables.

The WideHistory table column type returns a SQL Server type float for analog, a SQL Server type int for discrete tags, and an nvarchar(512) for string tags. The schema of the definition table, WideHistory_OLEDB, indicates a sql_variant type. This is simply a shorthand notation; it does not represent the type actually returned.

There is no Quality column for the WideHistory table because there is more than one tag value for each row returned. However, a value returned for a specified tag will be set to NULL if the quality of the value is invalid, inhibited, or unavailable.

The following restrictions apply when performing a query against the WideHistory table:

- Column names must be specified.
- The table is only accessible using the OPENQUERY statement.

Because tagnames are used for column names, the tagname can include any characters as defined by the rules for Microsoft SQL Server identifiers. An identifier that does not comply with the rules for the format of regular identifiers must always be delimited using brackets ([ ]). For more information on identifiers and delimiters, see your Microsoft SQL Server documentation.

If you include an illegal column name in your query and do not use delimiters, no data will be returned.

The StateWideHistory table is similar to the WideHistory table, except that it allows for retrieval of calculated "time in state" values for multiple tags, instead of actual history values. This table includes a vValue column, and the tag columns contain the time in state for the corresponding value. For more information on this table, see StateWideHistory (INSQL.Runtime.dbo.StateWideHistory). For information on how to query this table, see -OLD-ValueState Retrieval in the *AVEVA Historian Concepts Guide.*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

"Live" Table Format

The Live table presents the current value of the specified tag(s).

Note: In certain situations, data can bypass the Live table. These situations include:
- Receiving non-streamed original data (store/forward or CSV);
- Receiving revision data for a Latest value;
- Receiving no new streamed values after Historian was shut down and disabled, or after the computer was rebooted.

The format of the Live table is as follows. The DateTime column will indicate the time the value was received.

DateTime	TagName	Value	vValue	Quality	QualityDetail	(continued...)
02:17:01:05	Temp1	77	77	0	192	(continued...)
02:17:01:05	Temp2	78	78	0	192	(continued...)
02:17:01:05	Temp3	76	76	0	192	(continued...)
02:17:01:05	Temp4	79	79	0	192	(continued...)

Note: The AnalogLive, DiscreteLive, and StringLive tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you can't mix the tag types in the query. The Live table, however, can accommodate a mixture of tag types and should be used instead of the AnalogLive, DiscreteLive, or StringLive tables.

Auto-Summary Replication Tables

AVEVA Historian performs two types of replication -- Standard Replication Tables and auto-summary replication.

With auto-summary replication, Historian automatically computes and records an hourly summary as its corresponding real-time data is acquired. This allows Historian to quickly and efficiently retrieve large-volume data for a long duration, even months or years.

The Auto-Summary Replication tables are:

- _AutoReplicationGroup *
- _AutoReplicationRule *
- _AutoReplicationTagEntity *
- _AutoReplicationServer *
- _AutoTag *
- _AutoTagHistory *

  * System-level table. Do not edit.

  Note: The auto-summary feature was available beginning with AVEVA Historian 2017. From the time you installed or upgraded to AVEVA Historian 2017, the system has been creating auto-summary values for your analog tags. To backfill values for time before that instillation or upgrade, you can use the Replication Backfill Manager. For more information, see Adding Auto-Summary Values for a Defined Timeframe in the *AVEVA Historian Administration Guide*.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag Definition Tables

Types of tags that can be defined in the AVEVA Historian are analog, discrete, event, and string. The Tag Definition tables describe the qualities of the tags in your system.

The Tag Definition tables are:

- _Tag
- ChannelStatus
- CurrentEditor
- EngineeringUnit
- Message
- TagHistory
- TagType

Note: The _Tag table replace the previous Tag table. A new view named Tag now provide backward-compatibly.

Tag Definition Specialty Tables

The Tag Definition Specialty tables contain information about tag definitions that are imported into the Runtime database from the InTouch tagname.x Importer.

The Tag Definition Specialty tables are:

- Context
- Deviation
- Limit
- LimitName
- RateOfChange

I/O Data Acquisition Tables

I/O Data Acquisition tables contain information about tag definitions that are imported into the Runtime database from an IDAS.

The I/O Data Acquisition tables are:

- _IODriver
- _IOServer
- _Topic
- IOServerType

Note: The _IODriver, _IOServer, and _Topic tables replace the previous IODriver, IOServer, and Topic tables. New view named IODriver, IOServer, and Topic now provide backward-compatibly.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Insight Client Content Tables

Insight Client Content tables contain information needed to create content in the AVEVA Insight or Historian Insight tool.

The Insight Client Content tables are:

- ChartConfiguration
- ChartConfigurationAuditLog
- ChartConfigurationKeyword
- ChartConfigurationProperty
- ChartConfigurationStatistics
- ChartConfigurationTag
- DashboardConfiguration

InTouch Node Detail Tables

InTouch Node Detail tables contain information about tag definitions that are imported into the Runtime database from an InTouch application.

The InTouch Node Detail tables are:

- InTouchNode
- InTouchSpecific
- TopicImportInfo

Legacy Event and Summary Tables

Legacy Event and Summary tables contain definitions for events, including tags associated with events, detectors for events, and actions for events. The event system tables can also store "snapshots" of tag values at the time of an event, as well as details about the event itself.

A special type of event action is a summarization of tag values. A subset of the event tables provide the supporting framework for fully automated summary generation for analog, discrete and string tags.

Legacy Event and Summary tables are:

- _EventTag
- ActionType
- AnalogSnapshot
- CalcType
- DetectorType
- DiscreteSnapshot
- EventHistory
- EventTagPendingDelete*
- Frequency

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- SnapshotTag
- StringSnapshot
- SummaryData
- SummaryHistory
- SummaryOperation
- SummaryTagList
- TimeDetectorDetail
- TimeDetectorDetailPendingDelete*

  * System-level table. Do not edit.

  Note: The _EventTag table replaces the previous EventTag table. New views named EventTag now provide backward-compatibly.

Manual Data Tables

Manual Data tables contain information about tag that were created manually.

The Manual Data tables are:

- ManualAnalogHistory
- ManualDiscreteHistory
- ManualStringHistory

Modification Tracking Tables

The modification tracking tables contain information about changes that are made to columns in the database.

The modification tracking tables are:

- ModLogTable
- ModLogColumn
- UserDetail

Namespace and User-SpecificTables

The namespaces and grouping tables contain information that defines how sets of tags can be grouped together for alarming, displays, event management, and batch management. These tables also define hierarchies for items in the system, public, or private namespaces.

The namespaces and grouping tables are:

- Annotation
- GroupTagList
- PrivateGroupTag
- PrivateNameSpace

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- PublicGroupTag
- PublicNameSpace
- ServerList
- TagRef
- UserDetail
- WideTableDictionary

Standard Replication Tables

AVEVA Historian performs two types of replication -- standard replication and auto-summary replication.

With standard replication, tag information can be replicated from source, or tier 1, servers to replication, or tier 2, servers. Standard replication lets you consolidate and summarize information from separate servers to a single replication server so you can then perform analyses and run reports from the replication server on the consolidated data. You can summarize tags to capture analog or state values. You can also do a simple replication, which copies tag information directly without summarizing it. For more information, see Managing and Configuring Replication in the *AVEVA Historian Administration Guide.*

The Standard Replication tables are:

- CustomReplicationSchedule
- IntervalReplicationSchedule
- ReplicationGroup
- ReplicationRule
- ReplicationScheduleType
- ReplicationSchedule
- ReplicationSyncRequest
- ReplicationTagEntity
- ReplicationType

Storage Tables

Storage tables describe the storage partitions (shards) used by AVEVA Historian to house your data.

The Storage tables are:

- AttributeType
- ReplicationShard
- ShardAssignmentRule
- StorageLocation
- StorageShard
- StructureAttributes
- StructureType

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

System Configuration Tables

All AVEVA Historian parameters are stored in system configuration tables. Parameters include information regarding the historian's physical nodes, site-specific configuration parameters, and parameters pertaining to the physical I/O equipment to which the system is connected.

The system configuration tables are:

- _IODriver
- _IOServer
- _Tag
- _Topic
- ConfigStatusPending *
- ConfigStatusSnapshot*
- ErrorLog
- IOServerType
- LocalizedText*
- OPCQualityMap
- SnapshotDetail
- StorageLocation
- StorageShard
- SystemParameter
- TimeZone*
- UserDetail

* System-level table. Do not edit.

Note: The _IODriver, _IOServer, _Tag, and _Topic tables replace the previous IODriver, IOServer, Tag, and Topic tables. New views named IODriver, IOServer, Tag, and Topic now provide backward-compatibly.

ArchestrA Browsing Tables

The ArchestrA browsing tables store information required to support the browsing of the ArchestrA model view hierarchy by AVEVA Historian clients.

The ArchestrA browsing tables are:

- aaAreaData *
- aaAttributeData *
- aaAreaXML *
- aaObjectDataPending *
- aaObjectData *

* These tables are for internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tables

All information regarding how the system is configured is stored in tables in the Runtime database. Event history, summary history, and summary data are also stored in SQL Server tables. You can view the details of all tables by using the Microsoft SQL Server Management Studio.

_AutoIntervalReplicationSchedule

Contains one row for each autosummarization schedule interval.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule.
Period	small int NOT NULL	The number of units that make up the interval.
Unit	nvarchar(32) NOT NULL	The unit of measure for this interval.

_AutoReplicationGroup

Contains one row for each tag group that is summarized for your system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) ReplicationGroupKey	int NOT NULL	The unique identifier for the replication group.
ReplicationGroupName	nvarchar(255) NOT NULL	The name of the replication group.
(PK, FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the ReplicationServer table.
(FK) ReplicationTypeKey	tinyint NOT NULL	Can be 1, 2, or 3. (1 = Simple Replication, 2 = Analog Summary Replication, 3 = State Summary Replication.) ReplicationTypeKey is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		a foreign key from the ReplicationType table.
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the replication schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
SummaryReplicationNamingSchem e	nvarchar(512) NULL	The naming scheme for the replication tags belonging to this replication group. If the summary replication naming scheme is NULL, the summary replication naming scheme from the replication server is used as the default naming scheme for summary tags.
GroupAbbreviation	nvarchar(32) NULL	The abbreviation for the replication group. If GroupAbbreviation is NULL, ScheduleAbbreviation is used as the default group abbreviation.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoReplicationRule

Contains one row for each rule that applies to autosummarization for your system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
Name	nvarchar(255), NOT NULL	The name of the replication rule.
Priority	int, NOT NULL	The priority for the rule.
TagFilter	nvarchar(4000), NOT NULL	Do not edit.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		This shows the OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
(FK) ReplicationGroupKey	int, NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
(FK) ReplicationServerKey	int, NOT NULL	The unique identifier for the replication server.
Enabled	bit, NOT NULL	Used to indicate whether the replication rule is enabled. 0 - not enabled; 1- enabled
ApplyOtherRules	bit, NOT NULL	Used to indicate whether other rules apply. 0 - other rules do not apply; 1- other rules appy.
Id	int, NOT NULL	The unique identifier for the object.
ChangeVersion	timestamp, NOT NULL	Internal use only.

_AutoReplicationSchedule

Contains one row for each autosummarization schedule.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule.
ReplicationScheduleName	nvarchar(255) NOT NULL	The name of the replication schedule.
(FK) ReplicationScheduleTypeKey	int NOT NULL	The type of autosummarization schedule. ReplicationScheduleType is a foreign key from the ReplicationScheduleType table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ReplicationScheduleAbbreviation	nvarchar(32) NOT NULL	The abbreviation for the autosummarization schedule.
CreateGroup	bit NOT NULL	If TRUE, this autosummarization schedule is automatically added to new autosummarization groups.

_AutoReplicationServer

Contains one row for each replication server used for autosummarization.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server.
ReplicationServerName	nvarchar(255) NOT NULL	The name of the replication server.
Description	nvarchar(512) NULL	The description of the replication server.
SFPath	nvarchar(260) NULL	The local store-and-forward path associated with the replication server for this instance of AVEVA Historian.
SFFreeSpace	int NOT NULL	The free space for the store-and-forward path in MB.
CompressionEnabled	bit NULL	Used to specify whether compression should be enabled for the tag. 0 = No compression; 1= Compression.
UserName	nvarchar(255) NULL	The user name for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
Password	nvarchar(512) NULL	The encrypted password for logging in to the replication server. (AuthenticateWithAAUser must be 0.)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
TCPPort	int NOT NULL	The TCP port to use to log in to the replication server.
SummaryReplicationNamingSceme	nvarchar(512) NULL	The naming rule for summary replication tags. If ReplicationGroupKey is NULL, the naming rule is used from the ReplicationServerName scheme. If ReplicationServerName is NULL, the naming rule is used from the SummaryReplicationNamingSchem e system parameter.
SimpleReplicationNamingScheme	nvarchar(512) NULL	Naming rule for simple replication tags. If NULL the naming rule specified in the simple replication naming scheme system parameter is used.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
Bandwidth	int NOT NULL	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited.
MinSFDuration	int NOT NULL	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
ConnectionDetails	nvarchar(4000) NULL	Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
IntegratedSecurity	bit, NULL	Indicates whether this will be used for local replication connection and not for remote. (For remote replication, users are expected to provide username and password.)
ReplicationEvents	bit, NOT NULL	Specifies whether events are to be replicated.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoReplicationTagEntity

Contains one row for each tag replicated by the system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
(PK) (FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the Replication Server table.
(PK) DestinationTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
DestinationTagID	uniqueidentifier NOT NULL	The unique identifier for the destination tag.
(FK) SourceTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the source tag. SourceTagName is a foreign key from the Tag table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(FK) ReplicationGroupKey	int NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
MaximumStates	tinyint NOT NULL	Maximum number of states to track for state summary tags. Discrete summary tags have a limit of 3 states. Analog summary tags of a limit of 100 states. The default is 10 states.
(FK) CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ReplicationTagEntityId	uniqueidentifier, NOT NULL	This plays a role in SyncQueue to identify the entity.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoTag

Contains one row for each tag defined in the system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
TagId	uniqueidentifier, NOT NULL	The unique identifier for the tag.
(PK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
(FK) IOServerKey	int, NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
(FK) TopicKey	int, NULL	The unique numerical identifier for the topic. TopicKey is a foreign key from the Topic table.
Description	nvarchar(512), NULL	The description of the tag.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StorageType	smallint, NOTNULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored."
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds.
ItemName	nvarchar(256), NULL	The address string of the tag.
(FK) TagType	int, NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table.
DeadbandType	smallint, NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband.
TimeDeadband	int, NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.
ServerTimeStamp	bit, NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage.
(FK) ChannelStatus	tinyint, NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table.
(FK) MessageKey	int, NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.
(FK) EUKey	int, NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
MinEU	float, NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float, NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float, NULL	The minimum value of the raw acquired value.
MaxRaw	float, NULL	The maximum value of the raw acquired value.
Scaling	int, NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use).
RawType	int, NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float, NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
IntegerSize	tinyint, NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit, NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
RateDeadband	float, NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.
InterpolationType	tinyint, NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.
RolloverValue	float, NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.
MaxLength	smallint, NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint, NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		double-byte. The default is 0.
(FK) StructureId	uniqueidentifer, NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.
SourceTag	nvarchar(256), NULL	The source (tier 1) tag for the summary tag
SourceServer	nvarchar(255), NULL	The source (tier 1) server for the summary tag.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.
(FK) CurrentEditor	tinyint, NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		Application Server.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian. wwTagKey is populated from the Tag table, but is not a foreign key.
AIHistory	bit, NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage.
DateCreated	datetime2(7), NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated. Internal use only.
CreatedBy	nvarchar(255), NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated. Internal use only.
ChangeVersion	timestamp, NOT NULL	Internal use only.
CEVersion	tinyint, NOT NULL	The version number used to track changes to the information in the Tag table. Any change to the data in a row will cause the version indicator to change. The Configuration Editor (and other client tools) can detect the changed version and reload the corresponding tag details. Changes to this column are not tracked by the modification tracking system.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

_AutoTagHistory

Contains one row for each tag metadata instance uniquely identified by the TagId column.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) TagId	uniqueidentifier, NOT NULL	The unique identifier for the tag.
TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system. Internal use only.
Description	nvarchar(512), NULL	The description of the tag. Internal use only.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver. Internal use only.
StorageType	smallint, NOTNULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." Internal use only.
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. Internal use only.
TagType	int NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table. Internal use only.
TimeDeadband	int NULL	The minimum time, in milliseconds,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. Internal use only.
DateCreated	datetime2(7) NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated. Internal use only.
CreatedBy	nvarchar(256) NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated. Internal use only.
CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server. CurrentEditor is a foreign key from CurrentEditor table. Internal use only.
ServerTimeStamp	bit NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. Internal use only.
DeadbandType	smallint NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. Internal use only.
ChannelStatus	tinyint NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table. Internal use only.
AIHistory	bit NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage. Internal use only.
Message0	nvarchar(64) NULL	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. Internal use only.
Message1	nvarchar(64) NULL	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. Internal use only.
Unit	nvarchar(32) NULL	The unit of measure. Examples are mph, grams, and pounds. Internal use only.
DefaultTagRate	init NULL	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		rate set for the tag. Internal use only.
IntegralDivisor	float NULL	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only.
MinEU	float NULL	The minimum value of the tag, measured in engineering units. Internal use only.
MaxEU	float NULL	The maximum value of the tag, measured in engineering units.Internal use only.
MinRaw	float NULL	The minimum value of the raw acquired value. Internal use only.
MaxRaw	float NULL	The maximum value of the raw acquired value. Internal use only.
Scaling	int NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use). Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
RawType	int NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes). Internal use only.
ValueDeadband	float NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. Internal use only.
IntegerSize	tinyint NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use). Internal use only.
SignedInteger	bit NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed. Internal use only.
RateDeadband	float NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. Internal use only.
InterpolationType	tinyint NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. Internal use only.
RolloverValue	float NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000. Internal use only.
MaxLength	smallint NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512. Internal use only.
DoubleByte	tinyint NULL	Used to specify whether or not to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0. Internal use only.
StructureId	uniqueidentifier NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table. Internal use only.
SourceTag	nvarchar(256) NULL	The name of the source tag used for the replication tag. Internal use only.
SourceServer	nvarchar(255) NULL	The name of the tier 1 server with the source tag. Internal use only.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).

_DeletedReplicationTagExtendedProperty

Contains one row for each deleted replication tag extended property.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(FK) ReplicationTagExtended PropertyKey	int, NOT NULL	A unique identifier for the replication tag extended property.
ChangeVersion	timestamp, NOT NULL	Internal use only.

_EventTag

Contains one row for each event definition. Configuration information specific to event tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) DetectorTypeKey	int, NULL	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6 = Time-based (schedule). DetectorTypeKey is a foreign key from the DetectorType table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
DetectorString	nvarchar(1500), NULL	The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian.
(FK) ActionTypeKey	int, NULL	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 = Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary. ActionTypeKey is a foreign key from the ActionType table.
ActionString	nvarchar(1500), NULL	The script that specifies the event action. Action scripts run on the local AVEVA Historian.
UseThreadPool	bit, NOT NULL	Used to specify how system threads are used to process events. 1 = All events are handled by a single thread and a single logon to the SQL Server; 0 = Each event uses a separate system thread and logon. This will allow the event subsystem to manage the scan rates of each detector component concurrently. (Reserved for future use.)
ScanRate	int, NULL	The interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Logged	bit, NOT NULL	Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured. 0 = Not logged; 1 = Logged. The default is 1.
PostDetectorDelay	int, NOT NULL	The amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.
Priority	tinyint, NOT NULL	The priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. 0 = Normal; 1 = Critical. The default is 0.
Edge	tinyint, NOT NULL	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector.
Status	tinyint, NOT NULL	The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.

_IODriver

Contains one row for each IDAS providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IODriverKey	int NOT NULL	The unique identifier for an IDAS. This value is automatically generated by the system when the IDAS is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
ComputerName	nvarchar(255) NOT NULL	The name of the computer on which the IDAS runs.
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, redundant IDAS runs. You must use the fully qualified name of the computer. You could also use the IP address. This should be set to an empty string if no redundant IDAS is specified. Make sure that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online. By default, this

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		column is an empty string.
StoreForwardMode	tinyint NOT NULL	Used to specify whether or not store-and-forward capability is enabled. If enabled, and the network connection between the IDAS and the storage node fails, data will be "buffered" to the location specified by the store-and-forward path. Valid values are: 0 = Disabled; 1 = Enabled; 2 = Autonomous. The Autonomous mode (2) is an extension of the normal store-and-forward mode (1). It allows the IDAS to start up using an IDAS configuration file and collect data in store-and-forward mode if the network connection to the AVEVA Historian is not available.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StoreForwardPath	nvarchar(255) NULL	Used to specify the path for the IDAS data buffer on the local hard drive of the IDAS computer. The path should be absolute (for example, c:\IDASBuffer). Data is written to this path until the minimum threshold for the buffer is reached. Remote buffer paths are not supported. When the store-and-forward path specified for the IDAS is invalid, the default path picked by the system is: <public folder>\ArchestrA\Historian\IDAS\SF where the <public folder> is dependent on the operating system. For example, for the Windows 2008 operating system, the path is C:\ProgramData\ArchestrA\Historian\IDAS\SF. When the store-and-forward path specified for the IDAS is just a folder name (without any path characters like \ and :), the default path picked by the system is: <Windows system path>\<folder name specified by the user>. For example, for the Windows Server 2008 32-bit operating system, the path is C:\WINDOWS\system32\<folder name>.
MinMBThreshold	int NOT NULL	The minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer.
Status	tinyint NULL	Automatically updated by the system if a change is made to IDAS: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
Enabled	bit NOT NULL	Used to indicate whether the IDAS is enabled or not. 0 = Not enabled; 1 = enabled. Disabling the IDAS allows for the configuration to be retained in the database, even though the IDAS is removed from

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the system.
StoreForwardDuration	int NOT NULL	The minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
AutonomousStartupTimeout	int NOT NULL	The amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates.
FileChunkSize	int NOT NULL	The size, in bytes, of the data "chunks" that are sent to the historian when store-and-forward data is forwarded. The size of the chunks can be decreased to accommodate slower networks. Decrease this number only if the forwarding delay is greater than zero.
ForwardingDelay	int NOT NULL	The time interval, in milliseconds, at which "chunks" of store-and-forward data are forwarded to the historian. The length of the interval may need to be increased to accommodate slower networks.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ConnectionTimeout	int NOT NULL	The amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/ reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks.
CompressionEnabled	bit NOT NULL	Used for HCAL connections, this specifies whether compression is enabled.
TCPPort	int NOT NULL	Used for HCAL connections, this identifies the TCP port on remote IDAS server where HCAP is listening. The default value is 32565.
IntegratedSecurity	bit NOT NULL	Specifies whether to use Integrated security for communication. Applies to Domain environment only. In case of remote IDAS, the IDAS system will need to trust the server.
UserName	nvchar(255) NULL	Identifies the login username for the domain and workgroup. This is used for connecting to remote IDAS to push configuration.
Password	nvchar(512) NULL	Identifies the login password for the domain and workgroup. This is used for connecting to remote IDAS to push configuration.
ConnectionDetails	nvchar(4000) NULL	Specifies the encrypted connection token for allowing the remote IDAS to connect to Historian. Generated by the system.
Classic	bit NOT NULL	Specifies whether this is used for classic IDAS. Used during migration. Once the system detects the new IDAS, this is set as false.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ChangeVersion	timestamp NOT NULL	For internal use only.

_IOServer

Contains one row for each I/O Server providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IOServerKey	int NOT NULL	The unique numerical identifier for the I/O Server. This value is automatically generated by the system when the I/O Server is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
(FK) IODriverKey	int NULL	The unique identifier for an IDAS. IODriverKey is a foreign key from the IODriver table.
(FK) ApplicationName	nvarchar(32) NULL	The application name of the I/O Server. This name is usually the same as the executable file name. ApplicationName is a foreign key from the IOServerType table.
Path	nvarchar(255) NULL	The full UNC path (including the filename) to locate the executable file for the I/O Server. If the I/O Server type key is specified, the filename may be omitted.
ComputerName	nvarchar(255) NULL	The name of the computer on which the I/O Server runs.
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made.
AutoStart	bit NOT NULL	Used to control how the I/O Server starts up. 0 = Automatic startup

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		when the system starts. 1 = Manual startup required. Currently not used.
ExeType	int NOT NULL	The type of executable for the I/O Server. Used by the Operations Control Management Console to determine how to start the I/O Server. 0 = Service; 1 = Console application; 2 = Windows application.
InitializationStatus	tinyint NOT NULL	A control flag used to ensure that each I/O Server has been asked for the data type (integer or real) of each tag that it will send. Only needed after a database modification.
ProtocolType	tinyint NOT NULL	The protocol used by the AVEVA Historian server to communicate with the I/O Server. 1 = DDE; 2 = SuiteLink™; 3 = AVEVA Historian named pipe driver (for compatibility with IndustrialSQL Server 3.0 and previous versions). Of the operating systems currently supported by the AVEVA Historian, DDE is only supported on the Windows XP operating system.
Description	nvarchar(50) NULL	The description of the I/O Server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_ReplicationTagExtendedProperty

Contains one entry for each property for a replicated tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ReplicationTag ExtendedProperty Key	int NOT NULL	A unique identifier for the replication tag extended property.
(FK) ReplicationServer Key	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the Replication Server table.
(FK) DestinationTag Name	nvarchar (256) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
(FK) PropertyNameKey	int NULL	A unique identifier for the extended property name. PropertyNameKey is a foreign key from the TagExtendedPropertyName table.
PropertyValue	sql_variant NOT NULL	The value of this replication tag extended property.
ChangeVersion	timestamp NOT NULL	Internal use only.
Status	tinyint NOT NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_Tag

Contains one row for each tag in the system and includes the basic definition for the tag, such as the I/O Server that supplies the values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
TagId	uniqueidentifier NOT NULL	The unique identifier for the tag.
(PK) TagName	TagNameType (nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
(FK) IOServerKey	int, NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
(FK) TopicKey	int, NULL	The unique numerical identifier for the topic. TopicKey is a foreign key from the Topic table.
Description	nvarchar(512), NULL	The description of the tag.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver.
StorageType	smallint, NOT NULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored."
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds.
ItemName	nvarchar(256), NULL	The address string of the tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(FK) TagType	int, NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table.
DeadbandType	smallint, NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband.
TimeDeadband	int, NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.
ServerTimeStamp	bit, NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage.
(FK) ChannelStatus	tinyint, NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table.
(FK) MessageKey	int, NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.
(FK) EUKey	int, NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float, NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float, NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float, NULL	The minimum value of the raw acquired value.
MaxRaw	float, NULL	The maximum value of the raw acquired value.
Scaling	int, NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		root is reserved for future use).
RawType	int, NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float, NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
IntegerSize	tinyint,, NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit, NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.
RateDeadband	float, NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.
InterpolationType	tinyint, NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.
RolloverValue	float, NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.
MaxLength	smallint, NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint, NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0.
(FK) StructureId	uniqueidentifer, NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.
SourceTag	nvarchar(256), NULL	The name of the source tag used for the replication tag.
SourceServer	nvarchar(255), NULL	The name of the tier 1 server with the source tag.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.
(FK) CurrentEditor	tinyint, NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.
AIHistory	bit, NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage.
DateCreated	datetime(2) 7, NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
CreatedBy	nvarchar(256), NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated.
ChangeVersion	timestamp, NOT NULL	Internal use only.
CEVersion	tinyint, NOT NULL	The version number used to track changes to the information in the Tag table. Any change to the data in a row will cause the version indicator to change. The Configuration Editor (and other client tools) can detect the changed version and reload the corresponding tag details. Changes to this column are not tracked by the modification tracking system.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_TagExtendedProperty

Contains one entry for each property for a tag.

Column	Data Type	Description
(FK) TagName	nvarchar (256) NOT NULL	Specifies a tag name. TagName is a foreign key from the _Tag table.
(FK) PropertyNameKey	int NULL	A unique identifier for the extended property name. PropertyNameKey is a foreign key from the TagExtendedPropertyName table.
PropertyValue	sql_variant NOT NULL	The value of the tag extended property for this tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ChangeVersion	timestamp NOT NULL	Internal use only.

_Topic

Contains one row for each topic to be read from an I/O Server.

Column	Data Type	Description
(PK) TopicKey	int, NOT NULL	The unique numerical identifier for the topic. This value is automatically generated by the system when the topic is added.
(PK) (FK) IOServerKey	int, NOT NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
Name	nvarchar(180), NOT NULL	The name of the topic.
TimeOut	int NOT NULL	The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the topic: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
LateData	bit, NOT NULL	Used to enable acquisition of "late" data. 0 = Late data disabled; 1 = Late data enabled.
IdleDuration	int, NOT NULL	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		least 60 seconds.
ProcessingInterval	int, NOT NULL	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration.

aaAreaData

Contains one row for each area referenced by an object in the ArchestrA namespace

Column	Data Type	Description
(PK) AreaKey	int, NOTNULL	The unique identifier for the item in the Area data hierarchy.
Category	int, NOTNULL	The type of the item in the Area data hierarchy. 0 = Galaxy; 1 = WinPlatform; 3 = AppEngine; 13 = Area; 11 = DDESuiteLinkClient, OPCClient or InTouchProxy; 24 = RedundantDIObject. All other values are reserved for future use.
AreaName	nvarchar(255), NOT NULL	The name of the item in the Area data hierarchy.
ContainedName	nvarchar(255), NULL	The contained name (if relevant) of the item in the Area data hierarchy.
ParentKey	int, NOTNULL	The unique identifier for the parent item of this item. For the Galaxy

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		item, this value is 0.

aaAreaXML

Contains a single row describing the latest Area data sent from ArchestrA.

Column	Data Type	Description
Version	bigint NULL	The version number of the latest ArchestrA Area data package.
AreaXML	ntext NULL	Reserved for future use.

aaAttributeData

Contains one row for each attribute referenced by an object in the ArchestrA namespace.

Column	Data Type	Description
AttributeName	nvarchar(256) NOT NULL	The ArchestrA attribute name. This name corresponds to an AVEVA Historian tagname.
(FK) ObjectKey	int NOT NULL	ObjectKey is a foreign key from the aaObjectData table.
wwDomainTagKey	int NOT NULL	The unique numerical identifier for the ArchestrA attribute (historian tag) in a specific domain.
HierarchicalAttributeName	TagNameType (nvarachar(256) NOT NULL	The hierarchical attribute name for the tag.

aaAttributeDataPending

Contains one row for each attribute in the latest ArchestrA attribute data package.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
AttributeName	nvarchar(256) NOT NULL	The ArchestrA attribute name. This name corresponds to a AVEVA Historian tagname.
(FK) ObjectKey	int NOT NULL	ObjectKey is a foreign key from the aaObjectDataPending table.

aaObjectData

Contains one row for each object in the ArchestrA namespace.

Column	Data Type	Description
(PK) ObjectKey	int NOT NULL	The unique identifier for the object. This column does not have the same numeric value as ObjectKey column of the aaObjectDataPending table.
Type	int NOT NULL	The type of the object. 0 = Area; 1 = ApplicationObject (regular); 2 = Traceability object. All other values are reserved for future use.
aaTagName	TagNameType (nvarachar(256)) NULL	The ArchestrA tag name for the object.
ContainedName	nvarchar(256) NULL	The ArchestrA contained name for the object.
(FK) ParentKey	int NOT NULL	The unique identifier for the parent of this object.
Status	tinyint NOT NULL	Used to indicate whether a name change has occurred. 0 = No change; 1 = The tag name has changed; 2 = The contained name has changed. The default is 0.

aaObjectDataPending

Contains one row for each object in the latest ArchestrA object data package.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ObjectKey	int NOT NULL	The unique identifier for the object. This identifier is unique only within an object data package and may be repeated in subsequent data packages.
Type	int NOT NULL	The type of the object. 0 = Area; 1 = ApplicationObject (regular); 2 = Traceability object. All other values are reserved for future use.
aaTagName	TagNameType (nvarachar(256)) NULL	The ArchestrA tag name for the object.
ContainedName	nvarchar(256) NULL	The ArchestrA contained name for the object.
(FK) ParentKey	int NOT NULL	The unique identifier for the parent of this object.

ActionType

Contains one row for each type of event action.

Column	Data Type	Description
(PK) ActionTypeKey	int NOT NULL	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 = Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary. This value is automatically generated when a new action is created.
Name	nvarchar(33) NOT NULL	The name given to the type of action.
Description	nvarchar(50) NULL	The description of the action.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
EditorClassName	nvarchar(80) NULL	The name by which the component is referenced by a client application, such as the Operations Control Management Console, in order to provide a visual representation.
ActionClassName	nvarchar(80) NULL	The name by which the action component (COM object) is referenced in the system in order to perform the action.

AnalogSnapshot

Contains one row for each analog tag value that was configured to be stored when a defined event occurred. To view analog, discrete, and string snapshot values at the same time, use the v_SnapshotData view instead. For more information, see v_SnapshotData.

Column	Data Type	Description
(PK) (FK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. SnapshotTagKey is a foreign key from the SnapshotTag table.
(PK) (FK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. EventLogKey is a foreign key from the EventHistory table.
Value	float NULL	The value of the tag at the time of the event occurrence. Measured in engineering units.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

AnalogSummaryHistory (INSQL.Runtime.dbo.AnalogSummaryHistory)

The AnalogSummaryHistory view returns results for analog summary points.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
TagName	nvarchar(256) NOT NULL	The name of the summary tag.
StartDateTime	datetime2(7) NOT NULL	Start time of the retrieval cycle for which this row is returned.
EndDateTime	datetime2(7) NOT NULL	End time of the retrieval cycle for which this row is returned.
SliceBy	int, Discrete, or String	Performs dynamic resolution/cycle computation by tag. Returns one Analog Summary value per tag per dynamic cycle with start and end date time.
SliceByValue	var	Specifies the filter criterion to get the summary values for SlicedBy, based on that filter value.
OPCQuality	int NULL	OPC quality. Normal OPC quality retrieval logic is applied if: • All the points found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. • If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. • For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned.
PercentGood	float NULL	Time in seconds that the value was good for the retrieval cycle (pro-rated for partial cycles).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
First	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the first point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
FirstDateTime	datetime2(7) NULL	Timestamp associated with first value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle.
Last	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the last point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
LastDateTime	datetime2(7) NULL	Timestamp associated with last value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle.
Minimum	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the minimum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
MinDateTime	datetime2(7) NULL	Timestamp associated with Min value. NULL if Min is NULL.
Maximum	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the maximum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
MaxDateTime	datetime2(7) NULL	Timestamp associated with Max value. NULL if Max is NULL.
Average	float NULL	Time weighted average value of retrieval cycle. This is calculated by using the individual summary averages. The calculation is "Sum(average * delta t) / Total time of average in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Average query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.)
StdDev	float NULL	Time weighted standard deviation value of the retrieval cycle. The value is calculated using time weighted sums (Integrals) and time weighted sums of squares (IntegralOfSquares) values, prorated for any partially contained storage cycles. For analog tags, similar StdDev values are produced for each cycle.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Integral	float NULL	Area under value curve of retrieval cycle. The calculation is "Sum(value * delta t) / Total time of integral in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Integral query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) For analog tags, similar Integral values are produced for each cycle.
ValueCount	int NULL	Number of values in a particular cycle.
SourceTag	nvarchar(256) null	The source (tier 1) tag for the summary tag.
SourceServer	nvarchar(256) null	The source (tier 1) server for the summary tag.
wwCycleCount	int NULL	The number of cycles into which the entire query time range has been divided.
wwResolution	int NULL	Length of cycles in milliseconds. The default is 3600000 (equal to 1 hour).
wwTimeZone	nvarchar(50) NULL	Time zone to use for interpreting both input and output timestamp parameters. If none is specified, then the default is set to LOCAL.
wwVersion	nvarchar(30) NULL	Data version, ORIGINAL or LATEST. If none is specified, the default is LATEST.
wwTagKey	int NOT NULL	Tag key.
wwRetrievalMode	nvarchar(16) NOT NULL	Determines whether to use CYCLIC, DAILY, or DELTA retrieval. The default is DELTA.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

AttributeType

Contains one row for each attribute type.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) AttributeTypeKey	int NOT NULL	The unique numerical identifier for the attribute. This value is automatically generated by the system when the attribute is added.
AttributeTypeName	nvarchar(255) NOT NULL	The name of the attribute type.
AttributeTypeValue	tinyint NOT NULL	The bit mask for the attribute type.

CalcType

Contains one row for each type of summary calculation that can be performed by the Event subsystem.

Column	Data Type	Description
(PK) CalcType	CalcTypes(char(3)) NOT NULL	The type of calculation to be performed: SUM, MAX, MIN, or AVG.
Description	nvarchar(50) NULL	The description of the calculation.

ChangeNotification

Contains one row for each configuration modification made for a tag.

Column	Data Type	Description
(PK) ChangeType	sysname (nvarchar(128)) NOT NULL	Internal use only.
ChangeTime	datetime2(7) NOT NULL	Internal use only.
ChangeVersion	timestamp NOT NULL	Internal use only.

ChannelStatus

Contains one row for each type of channel status.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ChannelStatus	tinyint NOT NULL	Internal use only.
Description	nvarchar(255) NOT NULL	Internal use only.

ChartConfiguration

Defines configuration settings for a particular Insight content.

Column	Data Type	Description
(PK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
ChartConfigurationName	nvarchar(200), NOT NULL	The name of the Insight content.
ChartConfigurationUrl	nvarchar(100), NOT NULL	The web address for the Insight content.
ChartConfigurationType	tinyint, NOT NULL	Specifies what type of chart was saved. For example, single chart or dashboard
(FK) ChartConfigurationOwnerKey	int, NOT NULL	A unique identifier for the Insight content owner.
ChartConfigurationShareMode	tinyint, NOT NULL	Specifies whether the Insight content is shared.
LastSharedDateTimeUtc	datetime2(7), NULL	Specifies when the Insight content was last shared.
CreationDateTimeUtc	datetime2(7), NOT NULL	Specifies when the InSight content was created.
TimePreset	nvarchar(200), NULL	Specifies the selected time frame of the saved content. For example: Last 30 days, Last hour, or specific start and end times (for Custom).
TimeAggregate	tinyint, NULL	Specifies the aggregates used by the saved content. For example, Hour/Day for a Column chart.
ChartType	nvarchar(100), NULL	The type of chart used for this Insight content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
MobileShareMode	tinyint, NOT NULL	Specifies whether this Insight content is shared with mobile users.
EmbedShareMode	tinyint, NOT NULL	Specifies whether this Insight content can be embedded into a web page or other object.

ChartConfigurationAuditLog

Contains one row for each chart configuration audit log entry.

Column	Data Type	Description
ChartConfigurationUrl	nvarchar(100), NOT NULL	The web address for this InSight content.
(PK) ChartConfigurationUserKey	int, NOT NULL	A unique identifier for the InSight content user.
AuditLogDescription	nvarchart(200), NOT NULL	A descriptive record for the log.
(PK) CreationDateTimeUtc	datetime2(7), NOT NULL	The creation date and time for the log entry.

ChartConfigurationKeyword

Contains one row for each keyword associated with a particular Insight content.

Column	Data Type	Description
(PK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
Keyword	nvarchar(50), NOT NULL	A list of keywords associated with the content.

ChartConfigurationProperty

Contains one row for configuration property used by Insight charts.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
(PK) ChartConfigurationPropertyKey	nvarchar(100), NOT NULL	The unique identifier for the configuration property.
(PK) ChartConfigurationPropertyValue	nvarchar(200), NOT NULL	A value for the property.

ChartConfigurationStatistics

Contains statistics about chart configuration access.

Column	Data Type	Description
(PK, FK) ChartConfigurationUserKey	int, NOT NULL	The unique identifier for the InSight user.
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the InSight content.
LastAccessDateTimeUtc	datetime2(7), NULL	Specifies when the InSight content was last accessed.

ChartConfigurationTag

Contains one row for each tag configuration used in an Insight chart.

Column	Data Type	Description
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
(PK, FK) FQN	TagNameType(nvarchar(256)), NOT NULL	The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName.
Selected	bit, NOT NULL	Indicates whether the tag is selected to display in the chart of saved content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Color	nvarchar(10), NOT NULL	Indicates the color associated with the charted tag.
ActiveGroup	bit, NOT NULL	Specifies whether the tag is part of the active group.
(PK) LayoutIndex	smallint, NOT NULL	Indicates the index of the layout when the content is recreated in the browser.
SelectedOrder	smallint, NULL	Insight tracks the order in which tags for a chart are selected. This column indicates the position for this particular tag in that selection order.

Comments

Contains details of comments associated with a tag.

Column	Data Type	Description
Comments Key	int NOT NULL	The unique numerical identifier for the comment.
(FK) Comments TypeKey	tinyint NOT NULL	The unique numerical identifier for the comment type. CommentsTypeKey is a foreign key from the CommentsType table.
(FK) Comments OwnerKey	int NOT NULL	The unique numerical identifier for the owner of the comment.
(FK) Comments ShareMode	tinyint NOT NULL	Indicates whether or not the comment is shared.
(FK) FQN	nvarchar(256) NOT NULL	Fully qualified name of the associated tag. A fully qualified tagname uses the format: DataSourceName.TagName. FQN is a foreign key from the ChartConfigurationTag table.
DateTime	datetime2(7) NOT NULL	The timestamp for this comment.
DateTimeUtc	datetime2(7) NOT NULL	The UTC timestamp for this

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		comment.
Content	nvarchar(1024) NOT NULL	The comment text.
CreationDateTime	datetime2(7) NOT NULL	The date and time that the comment was created
CreationDateTimeUtc	datetime2(7) NOT NULL	The UTC date and time that the comment was created
ModifiedDateTime	datetime2(7) NULL	The date and time that the comment was modified.
ModifiedDateTimeUtc	datetime2(7) NULL	The URC date and time that the comment was modified
LastShared DateTime	datetime2(7) NULL	The date and time that the comment was last shared
LastShared DateTimeUtc	datetime2(7) NULL	The UTC date and time that the comment was last shared

CommentsType

Contains comment type definitions.

Column	Data Type	Description
(PK) Comments TypeKey	tinyint NOT NULL	The unique numerical identifier for the comment type.
Comments TypeName	nvarchar(32) NULL	A name for this comment type.

ConfigStatusPending

Contains one row for each database modification that requires a reinitialization of the system.

Important: Do not edit this table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ID	int NOT NULL	The unique identifier for the database modification.
Type	tinyint NOT NULL	Used to indicate the type of object to which the modifications apply. 0 = IDAS; 1 = IOServer; 2 = Topic; 3 = Tag; 4 = StorageLocation; 5 = SnapshotDetail; 6 = NamedSystemParameter; 7 = EngineeringUnit.
ObjectKey	int NOT NULL	The unique identifier of the modified object. If the modified object is a system parameter, the value will be 0. For all other object types, the value is from one of the following tables and columns: IODriver.IODriverKey; IOServer.IOServerKey; Topic.TopicKey; Tag.wwTagKey; StorageLocation.StorageType; SnapshotDetail.StorageSize.
Status	tinyint NULL	Used to indicate the type of modification. 1 = Insert; 2 = Update; 3 = Delete; 6 = The tag's source has changed (that is, if the value of the IOServerKey or TopicKey column in the Tag table has changed).

ConfigStatusSnapshot

When changes to the historian system are committed, a snapshot of the contents of the ConfigStatusPending table are stored to this table. The internal configuration object then finishes processing the reinitialization based on the data in this table, while any new changes are being stored in the ConfigStatusPending table.

The columns in this table are identical to the columns in the ConfigStatusPending table.

Important: Do not edit this table.

Context

Contains one row for each context to which a group of limits, rates of change, or deviations can belong. Example contexts are "Normal Operation" and "Cold Shutdown."

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ContextKey	int NOT NULL	The unique numerical identifier for the context. This value is automatically generated when a new context is added.
Description	nvarchar(50) NOT NULL	The description of the context.

CurrentEditor

Contains one row for each editor.

Column	Data Type	Description
(PK) CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		Application Server.
EditorName	nvarchar(max) NOT NULL	The name of the editor.

CustomReplicationSchedule

Contains one row for each trigger time for a custom replication schedule of ScheduleType CUSTOM. (This is used exclusively for tiered historian installations.) Interval-based replication schedules are handled in the IntervalReplicationSchedule table.

Column	Data Type	Description
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
TimeOfDay	nvarchar(10) NOT NULL	The time of day (in the local time for the AVEVA Historian) for the trigger time in the custom replication schedule. This value is automatically populated based on the schedule. The format is <Hour:Minutes><AM/PM>. Time is displayed on a 12-hour clock.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DashboardConfiguration

Contains one row for each InSight dashboard configured.

Column	Data Type	Description
(PK, FK) DashboardConfigurationKey	int, NOT NULL	The unique identifier for the InSight dashboard.
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the InSight content.
Positions	smallint, NOT NULL	Indicates the position/index of the chart when the chart is displayed in the Dashboard along with the other charts.

DeletedReplicationTagEntity

Contains one row for each attribute ...

Column	Data Type	Description
ReplicationTagEntityKey	int, NOT NULL	The unique identifier for the replication tag entity.
ChangeVersion	timestamp, NOT NULL	Internal use only.

DeletedTag

Contains one row for each deleted tag.

Column	Data Type	Description
TagId	uniqueidentifier NOT NULL	Internal use only.
ChangeVersion	timestamp NOT NULL	Internal use only.

DetectorType

Contains one row for each type of event detector.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) DetectorTypeKey	int NOT NULL	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6 = Time-based (schedule). This value is automatically generated when a new detector is created.
Name	nvarchar(33) NOT NULL	The name given to the type of detector.
Description	nvarchar(50) NULL	The description of the detector.
EditorClassName	nvarchar(80) NULL	The name by which the component is referenced by a client application, such as the Operations Control Management Console, in order to provide a visual representation.
DetectorClassName	nvarchar(80) NULL	The name by which the detector component (COM object) is referenced in the system in order to perform the detection.
ExecutionMode	tinyint NOT NULL	Used to specify the manner in which the detector executes. 0 = Executed cyclically by the event subsystem according to the event tag scan rate; 1 = Asynchronous and triggered by an external mechanism. The default is 0.

Deviation

Contains one row for each defined deviation for an analog tag. The deviation is the percentage of change in a tag's value from a fixed value, called the target. Each analog tag can have two defined deviations: major and minor. This table is populated when an InTouch application is imported and is not used by the AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(PK) (FK) ContextKey	int NOT NULL	The unique numerical identifier for the context. ContextKey is a foreign key from the Context table.
MinorDeviation	real NULL	The percentage that the tag can deviate from the target value before a minor deviation alarm condition is produced.
MinorChecked	bit NOT NULL	Used to determine the alarm state of the tag based on the minor deviation. 0 = Not in an alarm condition; 1 = In an alarm condition.
MinorPriority	int NULL	The priority level for the minor deviation. Valid values are numbers between 1 and 999, with 1 being the highest priority and 999 being the lowest priority.
MajorDeviation	real NULL	The percentage that the tag can deviate from the target value before a major deviation alarm condition is produced.
MajorChecked	bit NOT NULL	Used to determine the alarm state of the tag based on the major deviation. 0 = Not in an alarm condition; 1 = In an alarm condition.
MajorPriority	int NULL	The priority level for the major deviation. Valid values are numbers between 1 and 999, with 1 being the highest priority and 999 being the lowest priority.
Target	float NULL	The reference value of the tag from which minor and/or major deviation percentages are based.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Deadband	real NULL	The deviation percentage the tag value must drop below the target before the tag is taken out of alarm.

DiscreteSnapshot

Contains one row for each discrete tag value that was configured to be stored when a defined event occurred. To view analog, discrete, and string snapshot values at the same time, use the v_SnapshotData view instead. For more information, see v_SnapshotData.

Column	Data Type	Description
(PK) (FK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. SnapshotTagKey is a foreign key from the SnapshotTag table.
(PK) (FK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. EventLogKey is a foreign key from the EventHistory table.
Value	tinyint NULL	The state of the discrete tag at the time that the event occurred. 0 = FALSE; 1 = TRUE.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

EngineeringUnit

Contains one row for each defined engineering unit (unit of measure).

Column	Data Type	Description
(PK) EUKey	int NOT NULL	The unique numerical identifier of an engineering unit. This value is automatically generated by the system when the engineering unit is added.
Unit	nvarchar(32) NULL	The unit of measure. Examples are mph, grams, and pounds.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
DefaultTagRate	int NULL	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag.
Status	tinyint NULL	Automatically updated by the system if a change is made to the engineering unit: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
(FK) EUCKey	int NOT NULL	The unique numerical identifier for the catalog unit. Foreign key to the EngineeringUnitCatalog table.
EUChangeVersion	timestamp NOT NULL	For system use. Updated automatically to indicate the date and time of the last update.

EngineeringUnitCatalog

Contains one row for each defined engineering catalog unit.

Column	Data Type	Description
(PK) EUCKey	int NOT NULL	The unique numerical identifier of an engineering catalog unit. This value is automatically generated by the system when the engineering unit is added.
(FK) EUDKey	int NOT NULL	The unique numerical identifier of the dimension the unit belongs to. Foreign key to the EngineeringUnitDimension table.
(FK) EUSKey	int NOT NULL	The unique numerical identifier of the system of measure the unit belongs to. Foreign key to the EngineeringUnitSystem table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Symbol	nvarchar(32) NOT NULL	Canonical (standard) symbol used by the system of measurement that the unit belongs to. May contain special characters, such as the degree symbol for temperature measurements (for example, 68°).
BasicSymbol	nvarchar(32) NOT NULL	Canonical (standard) symbol used by the system of measurement that the unit belongs to. May not contain special characters.
Description	nvarchar(80) NOT NULL	A more descriptive name for the unit.
IntegralDivisor	float NOT NULL	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000.
ToBaseOffset	float NOT NULL	The offset used in linear conversion with respect to the base unit (b in the following line equation: $y = m(x+b)$).
ToBaseScaleFactor	float NOT NULL	The scale factor used in linear conversion with respect to the base unit (m in the following line equation: $y = m(x+b)$).
IsVisible	bit NOT NULL	Set to 1 if this unit is displayed

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		when you are creating a new engineering unit, and trying to link it to an existing catalog unit. Default value = 1.
IsSystem	bit NOT NULL	Set to 1 if this unit is defined by the system. (For example, it was created by the RuntimePostData.sql script.) Default value = 0.

EngineeringUnitDimension

Contains one row for each defined engineering unit dimension.

Column	Data Type	Description
(PK) EUDKey	int NOT NULL	The unique numerical identifier of a unit of measure's dimension.
Dimension	nvarchar(50) NOT NULL	The unit of measure's dimension name. For example, temperature, flow rate, length. Case-sensitivity depends on the default database collation.
BaseEUCKey	int NOT NULL	The unique numerical identifier of the base catalog unit for the given dimension. Linear conversions will be performed via the base unit. Foreign key to the EngineeringUnitCatalog table. Default value is 0 when creating a new dimension.
(FK) IntegralEUDKey	int NULL	The unique numerical identifier for this dimension's integral dimension. If not null, it indicates that accumulating units over time in this dimension results in units in the integral dimension. For example, the Volumetric Flow dimension's integral dimension is Volume.
(FK) DerivativeEUDKey	int NULL	The unique numerical identifier for this dimension's derivative

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		dimension. If not null, it indicates that derivation of units over time in this unit results in units in the derivative dimension. For example, the Length dimension's derivative dimension is Speed.

EngineeringUnitSystem

Contains one row for each defined engineering unit system.

Column	Data Type	Description
(PK) EUSKey	int NOT NULL	The unique numerical identifier of an engineering unit system.
System	nvarchar(50) NOT NULL	The name of the system of measure. For example, SI, US, British Imperial. Case-sensitivity depends on the default database collation.

ErrorLog

Contains one row for each system message (or error message). Typically, this table is not used. The actual message text is stored in the LocalizedText table, and can be retrieved by specifying the error code in the SQL query. Or, you can use the v_ErrorLog view to retrieve the data included in this table, plus the actual text.

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The date that the message was written to the system log, in the local time of the AVEVA Historian.
Type	nvarchar(10) NULL	The type of system message.
ErrorCode	int NULL	The unique identifier for the message.
Parameter	nvarchar(256) NULL	Optional details pertaining to the message text. For example, for the message "Disk space remaining on circular path" the parameter would contain the number of MB.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
TotalCount	int NULL	Used to prevent "flooding" conditions in the log file. If a particular message is generated numerous times during a relatively short period of time, the message is written to the log file only once, and the total number of times that it occurred appears in this column.
ModuleID	int NULL	A unique number assigned to the AVEVA Historian subsystem that generated the message.
Host	nvarchar(32) NULL	The computer on which the AVEVA Historian subsystem runs.
FileName	nvarchar(64) NULL	Used to indicate the program file that contains the line of code that an error message comes from. Used for debugging.
Line	int NULL	Used to indicate the line of code that an error message comes from. Used for debugging.

EventHistory

Contains one row for each stored event, as labeled by the tagname. Event data must be configured to be logged into this table.

Column	Data Type	Description
(PK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. This value is automatically generated by the system when the event record is added.
(FK) TagName	TagNameType (nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the EventTag table.
DateTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the time zone of the AVEVA Historian.
DetectDateTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the event system.
Edge	tinyint NULL	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector.

EventTagPendingDelete

Contains one row for each event tag that is pending deletion. This table is used internally by the system during the deletion process. The columns in this table are the same as in the _EventTag table.

Frequency

Contains one row for each available frequency for summary operations.

Column	Data Type	Description
(PK) FrequencyID	int NOT NULL	The unique numerical identifier for the frequency. Used to link a frequency with a time-based detector. 1= Hourly; 2 = Daily; 3 = Weekly; 4 = Monthly; 5 = Periodic; 6 = Other (Reserved for future use). This value is automatically generated by the system when the summarized tag is added.
Frequency	nvarchar(12) NOT NULL	The name for the frequency.

GroupTagList

Contains one row for each identified group of tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK, FK) GroupID	int, NOT NULL	Globally unique identifier for the tag group.
(PK, FK) wwDomainTagKey	int, NOT NULL	The unique numerical identifier for a tag in a specific domain.
Triggerval	float, NULL	A value that can be read by an application as a trigger value.

History (INSQL.Runtime.dbo.History)

Contains one row for each stored tag value.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp of the returned value. For delta retrieval, this is typically the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
TagName	(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float NULL	The value of the tag at the timestamp. The value is always NULL for string tags.
vValue	nvarchar(4000) NULL	The value of the analog, discrete, or string tag stored as a sql_variant. Using this column in a query allows you to have values with mixed datatypes as a result.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
OPCQuality	int NULL	The quality value received from the data source.
wwTagKey	int NOT NULL	The unique numerical identifier of a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		tag within a single AVEVA Historian.
wwRowCount	int NULL	The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query. This parameter has been deprecated; do not use. Use the wwCycleCount parameter instead.
wwResolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
wwEdgeDetection	nvarchar(16) NULL	The type of edge detection result set that the query will return. Valid values are NONE, LEADING, TRAILING, and BOTH.
wwOption	nvarchar(512) NULL	Specifies whether to return information about original data, summary data, or gaps in the storage blocks. Valid values are: • AutoSummaryData • PrimaryData • BlockGaps

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwRetrievalMode	nvarchar(16) NULL	Used to specify how retrieved data is processed before it is returned to the client. Valid values are: CYCLIC, DELTA, FULL, INTERPOLATED, BESTFIT, AVERAGE, MINIMUM, MAXIMUM, INTEGRAL, SLOPE, COUNTER, VALUESTATE, and ROUNDTRIP. • FULL = All stored values are returned • CYCLIC = All stored data for tags during the specified time interval are returned for the number of retrieval cycles or resolution specified • DELTA = Only values that changed during the specified time interval are returned. For all other modes, a calculation is performed by the system on the data and the value(s) are returned. The default is CYCLIC for retrieval from analog tables, DELTA for retrieval from discrete and string tables, and default is DELTA for retrieval from the History table, unless the specific retrieval mode implies otherwise. For example, SLOPE always has DELTA characteristics.
wwTimeDeadband	int NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
wwValueDeadband	float NULL	The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval. The default is 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.
wwVersion	nvarchar(30) NULL	If the original data values have been modified in the database, use this column to specify which version of the stored data is to be retrieved. Valid values are: ORIGINAL or LATEST. If no parameter is specified, the latest version of the data is retrieved by default. Modification is indicated by the QualityDetail.
wwCycleCount	int NULL	The number of retrieval cycles (sub-intervals) for the specified time period. The cycles will be spaced evenly across the time period. For example, if you specify a cycle count of four, the time period will be divided into four even cycles, and one or more values (depending on the retrieval mode) will be returned per cycle.
wwTimeStampRule	nvarchar(20) NULL	Used to specify whether cyclic results are timestamped at the beginning of the cycle or at the end of the cycle. Valid values are START and END. If no timestamp rule is specified in the query, then retrieval uses the setting of the TimeStampRule system parameter.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwInterpolationType	nvarchar(20) NULL	Used to determine which analog value to return at a given cycle boundary. Valid values are STAIRSTEP and LINEAR. If STAIRSTEP is specified, no interpolation occurs. The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned. If LINEAR is specified, the system calculates a new value at the given cycle time by interpolating between the last known value prior to the cycle time and the first value after the cycle time.
wwQualityRule	nvarchar(20) NULL	Used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC. A quality rule of GOOD means that data values with doubtful (64) OPC quality will not be used in the retrieval calculations and will be ignored. Values with bad QualityDetail indicate gaps in the data. A quality rule of EXTENDED means that data values with both good and doubtful OPC quality will be used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data. A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values will not cause the overall calculations to return NULL. You can apply wwQualityRule to all retrieval modes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwStateCalc	nvarchar(20) NULL	Used to indicate the type of calculation to return in the StateTime column for the "value state" retrieval mode. Valid values are: MINIMUM, MAXIMUM, AVERAGE, TOTAL, or PERCENT. You can also use the shortened versions: MIN, MAX, AVG, or SUM. The default for this column is TOTAL.
StateTime	float NULL	The amount of time in the state, expressed as a float (64-bit) number of milliseconds, for all time-in-state modes except for "Percent." For a time-in-state percentage calculation, this value is the percentage of the total time interval, in the range 0.0 to 100.0, that the value was in the state.
PercentGood	float NULL	The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100.
wwParameters	nvarchar(128) NULL	Contains the "stream index" (used for informational purposes only) and the special index value to indicate that the value was calculated by the "SLR()" filter. SLR stands for "simple linear regression," the algorithm used for predictive retrieval. By default, the value of this parameter is an empty string.
StartDateTime	datetime2 NOT NULL	Start time of the retrieval cycle for which this row is returned.
SourceTag	nvarchar(256) NULL	Returns the name of the source tag for a replicated tag at the time this point was stored. With the SourceServer, this column uniquely identifies the tag from which this replicated point is coming.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
SourceServer	nvarchar(256) NULL	Returns the name of the server from which replication occurred for this replicated tag at the time this point was stored.
wwFilter	nvarchar(512) NULL	Gives the name of the filter. Filters are specified as C-like functions and parentheses are always required, even when the filter does not override the default parameters (no parameters are passed). Filter values are NoFilter, ToDiscrete(), SigmaLimit(), SnapTo(), and SLR(). The default value is NoFilter. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.
wwValueSelector	nvarchar(128) NOT NULL	Used to specify which column to return for specified analog summary tags in the four basic retrieval modes: DELTA, FULL, CYCLIC, and INTERPOLATED. The defined set of selectors are AUTO (the default in all modes if not overridden), MINIMUM or MIN, MAXIMUM or MAX, FIRST, LAST, AVERAGE or AVG, INTEGRAL, and STDDEV or STANDDEVIATION. The default AUTO setting returns the Last attribute in the Value column (which makes it accessible in the WideHistory table). You can only override the selector for the basic retrieval modes. FIRST, LAST, MIN, and MAX each have their own timestamp that will be used for the time part of the VTQ. AVG, INTEGRAL and STDDEV represent values that hold for the entire cycle so the summary period start time will be used for the time part of a VTQ.
wwMaxStates	int NULL	For internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

HistoryBlock (INSQL.Runtime.dbo.HistoryBlock)

Contains one row for each block of history data stored in the main storage partition's timeline.

Column	Data Type	Description
FromDate	datetime2(7) NOT NULL	The starting timestamp for the history block.
ToDate	datetime2(7) NOT NULL	The ending timestamp for the history block.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node.
Description	nvarchar(50) NULL	The description of the history block.
OnLine	tinyint NOT NULL	Deprecated.
HistoryArchived	int NOT NULL	Used to indicate whether or not the history block has been archived (backed up). 1 = No status; 2 = Archived; 3 = Restored; 4 = Deleted. Reserved for future use.
SummaryArchived	int NOT NULL	Used to indicate whether or not the tag summary has been archived (backed up). 1 = No status; 2 = Archived; 3 = Restored; 4 = Deleted. Reserved for future use.
EventArchived	int NOT NULL	Used to indicate whether or not the event has been archived (backed up). 1 = No status; 2 = Archived; 3 = Restored; 4 = Deleted. Reserved for future use.
StorageAreaType	int NOT NULL	The paradigm used for storage. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent. Reserved for future use.
ArchiveDate	datetime2(7) NULL	The date at which the history block was archived. Reserved for future use.
ArchiveLocation	nvarchar(50) NULL	The location to which the history block was archived. Reserved for future use.
Version	int NULL	The version number for the history block. 1 = Block format used until release 3.0; 2 = Block format used for releases 3.0 and later. Reserved for future use.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Compression	int NULL	The version number for cyclic compression. 1 = No compression; 2 = Huffman encoding. Reserved for future use.
Sequence	int NOT NULL	The sequence number for the data stream. (1...n) Reserved for future use.
TimeZoneOffset	int NULL	The UTC offset, in minutes, from the local timestamp for when the history block was created. For example, a value of 480 would indicate an 8-hour offset from UTC, which would be Pacific Standard Time.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.

HistorianSysObjects

Contains one row for each object in the database for which changes can be tracked.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) id	int NOT NULL	The unique identifier for the object.
Type	char(2) NULL	The type of object. C = CHECK constraint; D = Default or DEFAULT constraint; F = FOREIGN KEY constraint; K = PRIMARY KEY or UNIQUE constraint; L = Log; P = Stored procedure; R = Rule; RF = Stored procedure for replication; S = System table; TR = Trigger; U = User table; V = View; X = Extended stored procedure. Currently, only changes for the user tables (object type U) are tracked.
Name	varchar(50) NULL	The name of the modified object.

IntervalReplicationSchedule

Contains one row for each replication schedule of ScheduleType INTERVAL. (This is used exclusively for tiered historian installations.) Custom replication schedules are handled in the CustomReplicationSchedule table.

Column	Data Type	Description
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
Period	smallint NOT NULL	The period value.
Unit	nvarchar(32) NOT NULL	The name of the unit.

InTouchNode

Contains one row for each InTouch node from which a tagname data dictionary (Tagname.x) is imported into the AVEVA Historian.

Column	Data Type	Description
(PK) NodeKey	int NOT NULL	The unique numerical identifier of the named InTouch node. A node key is automatically generated by

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the system when a node is added.
MachineName	nvarchar(255) NOT NULL	The name of the computer on which the InTouch application resides.
ApplicationName	nvarchar(32) NULL	The name of the InTouch application (VIEW).
Path	nvarchar(250) NULL	The UNC path to the InTouch Tagname.X file.
Description	nvarchar(50) NULL	The description of the InTouch node.
DuplicateChar	nvarchar(12) NOT NULL	The string that was added to a tag name as a prefix or suffix to make it unique.
PrefixOrSuffix	bit NOT NULL	Used to indicate whether unique tags were created by prefixing or suffixing the unique string for the node. 0 = Suffix; 1 = Prefix. Internal use only.
AlwaysModifyName	bit NOT NULL	Used to indicate whether a uniqueness string was added to every tag for the node. Internal use only.
ImportPlantTags	tinyint NOT NULL	Used to indicate whether plant tags were imported. (In InTouch, plant tags are called I/O tags.) Internal use only.
ImportSystemTags	tinyint NOT NULL	Used to indicate whether system tags were imported. Internal use only.
ImportMemoryTags	tinyint NOT NULL	Used to indicate whether memory tags were imported. Internal use only.
ImportAllTags	int NOT NULL	Used to indicate whether all tags were imported. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
FixedStorageRate	tinyint NOT NULL	The cyclic storage rate, in seconds, for imported tags. Internal use only.
ImportRoute	tinyint NOT NULL	Used to indicate the type of import that was last performed for the node. Internal use only.

InTouchSpecific

Contains one row of import-related information for each data dictionary (Tagname.x) imported from InTouch HMI software.

Column	Data Type	Description
(PK) (FK) NodeKey	int NOT NULL	The unique numerical identifier of the named InTouch node. NodeKey is a foreign key from the InTouchNode table.
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
OriginalName	nvarchar(32) NOT NULL	The original tag name in an InTouch application. The tag name may be different than the AVEVA Historian tag name if a new name was generated to ensure uniqueness.
TypeInfo	int NOT NULL	The type of tag in an InTouch application. For more information about InTouch tag types, see your InTouch documentation. Internal use only.
InInSQL	bit NOT NULL	Used to specify whether or not the tag information has been imported from InTouch into the AVEVA Historian database. Internal use only.
Comment	nvarchar(50) NULL	The original description for the tag that was imported from InTouch.

IOServerType

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Contains one row for every known I/O Server type. Information about a new I/O Server is added to this table when a server is installed. This table is populated with the latest information about AVEVA I/O Servers at the time of shipping.

Column	Data Type	Description
(PK) ApplicationName	nvarchar(32) NOT NULL	The application name of the I/O Server. This name is usually the same as the executable file name.
Description	nvarchar(100) NULL	The description of the I/O Server type.
ExeName	nvarchar(255) NULL	The name of the I/O Server's executable file.
Revision	nchar(20) NULL	The revision number for the I/O Server.

Limit

Contains one row for each monitored limit for a specified tag. A limit can be associated with one or more tags and/or contexts.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the AnalogTag table.
(PK) (FK) ContextKey	int NOT NULL	The unique numerical identifier for the context. ContextKey is a foreign key from the Context table.
(PK) LimitType	tinyint NOT NULL	The type of limit; that is, whether it is a rising (up) or falling (down) limit. 0 = Rising; 1 = Falling.
(PK) ,Value	float NOT NULL	The value that is used as a specific limit for a tag. In theory, a tag can have an infinite number of limits defined.
(PK) (FK) LimitNameKey	int NOT NULL	The unique numerical identifier associated with a limit name. LimitNameKey is a foreign key from the LimitName table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Priority	int NOT NULL	The priority for the limit. Priorities can range from 1 to over 2 billion, with 1 being the highest priority.
Checked	bit NOT NULL	Used to specify whether a tag imported from InTouch is configured for automatic limit checking. Only checked limits are imported. 0 = Checking disabled; 1 = Checking enabled.
Description	nvarchar(50) NULL	The description of the limit.

LimitName

Contains one row for each name that is associated with a defined limit. Examples are "high," "low," and "maintenance."

Column	Data Type	Description
(PK) LimitNameKey	int NOT NULL	The unique numerical identifier associated with a limit name. This value is automatically generated by the system when a limit is added.
Name	nvarchar(20) NULL	The name for the limit.

Live (INSQL.Runtime.dbo.Live)

Contains one row for each analog, discrete, or string tag. The value of each tag in this table is updated every time a new value is received.

Note: In certain situations, data can bypass the Live table. These situations include:
- Receiving non-streamed original data (store/forward or CSV);
- Receiving revision data for a Latest value;
- Receiving no new streamed values after Historian was shut down and disabled, or after the computer was rebooted.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data last changed.
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Value	float NULL	The value of the tag at date/time. This value is always NULL for string tags.
vValue	nvarchar(256) NULL	The value of the analog, discrete, or string tag stored as a sql_variant. Using this column in a query allows you to have values with mixed datatypes as a result.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
OPCQuality	int NULL	The quality value received from the data source.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.
wwRetrievalMode	nvarchar(16) NULL	For queries against this table, the value of this column is ignored.
wwTimeDeadband	int NULL	For queries against this table, the value of this column is ignored.
wwValueDeadband	float NULL	For queries against this table, the value of this column is ignored.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwParameters	nvarchar(128) NULL	Used for additional parameters that can be specified. By default, the value of this parameter is an empty string.
SourceTag	nvarchar(256) NULL	Returns the name of the source tag for a replicated tag at the time this point was stored. With the SourceServer, this column uniquely identifies the tag from which this replicated point is coming.
SourceServer	nvarchar(256) NULL	Returns the name of the server from which replication occurred for this replicated tag at the time this point was stored.
wwValueSelector	nvarchar(128) NOT NULL	Used to specify which column to return for specified analog summary tags in the four basic retrieval modes, DELTA, FULL, CYCLIC, and INTERPOLATED. The defined set of selectors are AUTO (the default in all modes if not overridden), MINIMUM or MIN, MAXIMUM or MAX, FIRST, LAST, AVERAGE or AVG, and INTEGRAL. The default AUTO setting returns the Last attribute in the Value column (which makes it accessible in the WideHistory table). You can only override the selector for the basic retrieval modes.
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)`returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

LocalizedText

Contains one row for each string of text that can be returned to a client from AVEVA Historian (for example, error messages and status messages).

If you add new text to the LocalizedText table, you must stop and restart AVEVA Historian for the changes to go into effect.

Column	Data Type	Description
(PK) TextKey	int NOT NULL	The unique identifier for the message.
(PK) LanguageID	int NOT NULL	The locale ID for the language used. This ID is also used in the SQL Server syslanguages table.
LocalizedText	nvarchar(max) NULL	The content of the message.

ManualAnalogHistory

Contains one row for each tag history that was manually defined for analog tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) DateTime	datetime2(7), NOT NULL	The date and time that the tag history was defined.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	tinyint, NULL	The value of the tag at date/time.
Quality	tinyint, NOT NULL	An internal representation of data quality.
QualityDetail	int, NULL	An internal representation of data quality.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualDiscreteHistory

Contains one row for each tag history that was manually defined for discrete tags

Column	Data Type	Description
(PK) DateTime	datetime2(7), NOT NULL	The date and time that the tag history was defined.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	tinyint, NULL	The value of the tag at date/time.
Quality	tinyint, NOT NULL	An internal representation of data quality.
QualityDetail	int, NULL	An internal representation of data quality.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualStringHistory

Contains one row for each tag history that was manually defined for string tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) DateTime	datetime2(7), NOT NULL	The date and time that the tag history was defined.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512), NULL	The value of the tag at date/time. This value is always NULL for string tags.
Quality	tinyint, NOT NULL	An internal representation of data quality.
QualityDetail	int, NULL	An internal representation of data quality.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

Message

Contains one row for each on/off message pair that can be associated with a discrete tag. For example, a message pair may be "Open" and "Closed" and could be associated with valve and switch positions.

Column	Data Type	Description
(PK) MessageKey	int NOT NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is automatically generated by the system when the message pair is added.
Message0	nvarchar(64) NULL	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state.
Message1	nvarchar(64) NULL	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state.

ModLogColumn

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Contains one row for each database column on which an INSERT, UPDATE, or DELETE has been performed.

Column	Data Type	Description
(FK) ModTableKey	int NOT NULL	The unique numerical identifier for the modification. ModTableKey is a foreign key from the ModLogTable table.
ColumnName	nvarchar(30) NOT NULL	The name of the modified column.
OldValue	sql_variant NULL	The value stored in the column before the modification was made, if the modification was to a configuration table. For modifications to history data using SQL INSERT and UPDATE statements, this column contains the timestamp of the earliest data affected by the INSERT or UPDATE operation. If multiple changes are made to the same data, then only the most recent change will be contained in this column. This column is not used for modifications made to history data using a CSV file.
NewValue	sql_variant NULL	The new value stored in the column, if the modification was to a configuration table. For modifications to history data, this column contains the total count of consecutive value updates attempted.

ModLogTable

Contains one row for each database table on which an INSERT, UPDATE, or DELETE has been performed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ModTableKey	int NOT NULL	The unique numerical identifier for the modification. This value is automatically generated by the system when a new modification record is added.
(FK) id	int NOT NULL	The unique identifier for the object that was modified. id is a foreign key from the HistorianSysObjects table.
ModType	char(1) NOT NULL	The type of modification. U = Update; I = Insert; D = Delete; 1 = SQL insert; 2 = SQL original insert; 3 = SQL update; 4 = CSV insert; 5 = CSV original insert; 6 = CSV update; 7 = CSV multi-point update; 8 = CSV "fast load" insert.
RowKey	sql_variant NOT NULL	The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.
UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. This value is from the UserDetail table. Currently not used.
DateTime	datetime2(7) NOT NULL	The timestamp of when the modification occurred.
UserName	nvarchar(256) NOT NULL	The name of the database user that made the modification. The value of this column reflects the Windows authentication user name (for example, DOMAIN\ user_login_name) or the SQL Server authentication user name (for example, dbo), depending on how the user is logged into the SQL Server when the modification is made. In the case of a CSV file import, this column contains the user name as it appears in the CSV file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

NameSpaceIcons

Contains one row for each icon.

Column	Data Type	Description
(PK) Type	int, NOT NULL	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups. This value is of data type int, with no default.
Icon	image, NULL	Includes the image of the icon.
Name	nvarchar(30), NOT NULL	The name of the object in the hierarchy.
Description	nvarchar(50), NULL	A description of the namespace icon.

OPCQualityMap

Contains one row for each defined OPC quality.

Column	Data Type	Description
(PK) OPCQuality	tinyint NOT NULL	The quality value received from the data source.
Description	nvarchar(100) NULL	The text that describes what the OPC quality value means. Do not modify this description.

PrivateGroupTag

Contains one row for each instance of a tag in a user's private namespace.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) (FK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. NameKey is a foreign key from the PrivateNameSpace table.
(PK) (FK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(PK) (FK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain. wwDomainTagKey is a foreign key from the TagRef table.

PrivateNameSpace

Contains one row for each object in the private namespace. Objects in the private namespace can include plant machines, areas, tags, and so on, and are organized in a hierarchy. Allows for more than one name to map to a single tag.

Column	Data Type	Description
(PK) (FK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(PK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. This value is automatically generated by the system when the object is added.
Type	int NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
Name	nvarchar(255) NULL	The name of this object in the hierarchy.
ConfigStor	ntext(16) NULL	If the namespace object has configuration information associated with it (for example, configuration information for a set of trend curves, the name of the file that contains the configuration information).
ParentKey	int NOT NULL	The unique identifier for a named object in this namespace.

PublicGroupTag

Contains one row for each instance of a tag in the public namespace.

Column	Data Type	Description
(PK) (FK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. NameKey is a foreign key from the PublicNameSpace table.
(PK) (FK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain. wwDomainTagKey is a foreign key from the TagRef table.

PublicNameSpace

Contains one row for each object in the public namespace. Objects in the public namespace can include plant machines, areas, and so on, and are organized in a hierarchy. Allows more than one name to map to a single tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. This value is automatically generated by the system when the object is added.
Type	int NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
Name	nvarchar(255) NULL	The name of this object in the hierarchy.
ConfigStor	ntext NULL	If the namespace object has configuration information associated with it (for example, configuration information for a set of trend curves, the name of the file that contains the configuration information).
ParentKey	int NOT NULL	The unique identifier for a named object in this namespace.
OriginalName	nvarchar(255) NOT NULL	Internal use only.

QualityMap

Contains one row for every permutation of quality detail for a tag value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) QualityDetail	int NOT NULL	An internal representation of data quality.
QualityString	nvarchar(max) NULL	The text string that describes what the quality detail value means.

RateOfChange

Contains one row for each monitored rate of change for a tag.

Column	Data Type	Description
(PK) (FK) TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(PK) (FK) ContextKey	int NOT NULL	The unique numerical identifier for the context. ContextKey is a foreign key from the Context table.
Value	float NOT NULL	The percentage of change for a tag during the amount of time specified by the time base.
TimeBase	int NOT NULL	The unit of time against which the rate of change will be measured.
Priority	int NOT NULL	The priority for the rate of change. Priorities can range from 1 to over 2 billion, with 1 being the highest priority.
Checked	bit NOT NULL	Used to specify whether a tag imported from InTouch was configured for automatic rate of change checking. 0 = Checking disabled; 1 = Checking enabled.

ReplicationGroup

Contains one row for each replication group. (This is used exclusively for tiered historian installations.)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ReplicationGroupKey	int NOT NULL	The unique identifier for the replication group.
ReplicationGroupName	nvarchar (255) NOT NULL	The name of the replication group.
(PK) (FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the ReplicationServer table.
(FK) ReplicationTypeKey	tinyint NOT NULL	Can be 1, 2, or 3. (1 = Simple Replication, 2 = Analog Summary Replication, 3 = State Summary Replication.) ReplicationTypeKey is a foreign key from the ReplicationType table.
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the replication schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
SummaryReplicationNamingSchem e	nvarchar(512) NULL	The naming scheme for the replication tags belonging to this replication group. If the summary replication naming scheme is NULL, the summary replication naming scheme from the replication server is used as the default naming scheme for summary tags.
GroupAbbreviation	nvarchar(32) NULL	The abbreviation for the replication group. If GroupAbbreviation is NULL, ScheduleAbbreviation is used as the default group abbreviation.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ReplicationRule

Contains one row for each replication rule for your system.

Column	Data Type	Description
Name	nvarchar(255), NOT NULL	The name of the replication rule.
Priority	int, NOT NULL	The priority for the rule.
TagFilter	nvarchar(4000), NOT NULL	Do not edit. This shows the OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
(FK) ReplicationGroupKey	int, NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
(FK) ReplicationServerKey	int, NOT NULL	The unique identifier for the replication server.
Enabled	bit, NOT NULL	Used to indicate whether the replication rule is enabled. 0 - not enabled; 1- enabled
ApplyOtherRules	bit, NOT NULL	Used to indicate whether other rules apply. 0 - other rules do not apply; 1- other rules appy.
Id	int, NOT NULL	
ChangeVersion	timestamp, NOT NULL	Internal use only.

ReplicationSchedule

Contains one row for each replication schedule. (This is used exclusively for tiered historian installations.)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule.
ReplicationScheduleName	nvarchar(255) NOT NULL	The name of the replication schedule.
(FK) ReplicationScheduleTypeKey	int NOT NULL	The type of replication schedule. ReplicationScheduleType is a foreign key from the ReplicationScheduleType table.
ReplicationScheduleAbbreviation	nvarchar(32) NOT NULL	The abbreviation for the replication schedule.
CreateGroup	bit NOT NULL	If TRUE, this replication schedule is automatically added to new replication groups.

ReplicationScheduleType

Contains one row for each type of replication schedule. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationScheduleTypeKey	int NOT NULL	The unique identifier for the schedule type.
ReplicationScheduleTypeName	nvarchar(32) NOT NULL	The name of the replication schedule type, either INTERVAL or CUSTOM. The default is INTERVAL.

ReplicationServer

Contains one row for each replication server. (This is used exclusively for tiered historian installations.) The password is encrypted by an internal routine before storing in this table.

Column	Data Type	Description
(PK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server.
ReplicationServerName	nvarchar(255) NOT NULL	The name of the replication server.
Description	nvarchar(512) NULL	The description of the replication server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
SFPath	nvarchar(260) NULL	The local store-and-forward path associated with the replication server for this instance of AVEVA Historian.
SFFreeSpace	int NOT NULL	The free space for the store-and-forward path in MB.
CompressionEnabled	bit NULL	Used to specify whether compression should be enabled for the tag. 0 = No compression; 1= Compression.
UserName	nvarchar(255) NULL	The user name for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
Password	nvarchar(512) NULL	The encrypted password for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
TCPPort	int NOT NULL	The TCP port to use to log in to the replication server.
SummaryReplicationNamingSchem e	nvarchar(512) NULL	The naming rule for summary replication tags. If ReplicationGroupKey is NULL, the naming rule is used from the ReplicationServerName scheme. If ReplicationServerName is NULL, the naming rule is used from the SummaryReplicationNamingSchem e system parameter.
SimpleReplicationNamingScheme	nvarchar(512) NULL	Naming rule for simple replication tags. If NULL the naming rule specified in the simple replication naming scheme system parameter is used.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Bandwidth	int NOT NULL	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited.
MinSFDuration	int NOT NULL	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
ConnectionDetails	nvarchar(1024) NULL	Internal use only.
IntegratedSecurity	bit, NULL	Indicates whether this will be used for local replication connection and not for remote. (For remote replication, users are expected to provide username and password.)
ReplicationEvents	bit, NOT NULL	Specifies whether events are to be replicated. This applies only to remote servers.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication server: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationShard

Contains one row for each partition (shard) used for replication.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
ShardName	nvarchar (255), NOT NULL	The name of the partition.
Description	nvarchar (512), NULL	The description of the partition.
ComputerName	nvarchar (255), NULL	The name of the computer on which the partition resides.
CmdArgs	nvarchar (4000), NULL	Do not edit. These are command line parameters for customizing replication and storage execution.
CmdExtArgs	nvarchar (4000), NULL	Do not edit. These are command line parameters for customizing replication and storage execution.
Id	int, NOT NULL	The unique identifier for the object.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationSyncRequest

Contains one row for each replication synchronization request. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationSyncRequestKey	bigint NOT NULL	The unique identifier for the replication synchronization request.
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
RequestVersion	smallint NOT NULL	The version type. 0 = Initial version; 1 = Latest version.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ModStartDateTimeUtc	datetime2(7) NOT NULL	The start time (in UTC) for the replication synchronization request.
ModEndDateTimeUtc	datetime2(7) NOT NULL	The end time (in UTC) for the replication synchronization request.
EarliestExecutionDateTimeUtc	datetime2(7) NULL	The earliest execution date (in UTC) for the replication synchronization request.
ExecuteState	tinyint NOT NULL	Value automatically changes as the rep service processes the sync queue. 0 = ready to process; 1 = currently being processed; 2 = rows needs merging/unmerging.

ReplicationSyncRequestPending

Contains one row for each pending replication synchronization request. This table is used internally by the system.

The columns in this table are the same as in the ReplicationSyncRequest table.

ReplicationTagEntity

Contains one row for each replication tag entity. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
(PK) (FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the Replication Server table.
(PK) DestinationTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
DestinationTagID	uniqueidentifier NOT NULL	The unique identifier for the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		destination tag.
(FK) SourceTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the source tag. SourceTagName is a foreign key from the Tag table.
(FK) ReplicationGroupKey	int NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
MaximumStates	tinyint NOT NULL	Maximum number of states to track for state summary tags. Discrete summary tags have a limit of 3 states. Analog summary tags of a limit of 100 states. The default is 10 states.
(FK) CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication tag: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationType

Contains one row for each replication type. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationTypeKey	tinyint NOT NULL	Can be 1, 2, or 3.
ReplicationTypeName	nvarchar(255) NOT NULL	Value is determined by the ReplicationTypeKey. 1 = Simple Replication, 2 = Analogy Summary Replication, 3 = State Summary Replication.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SearchMessageSyncRequest

Contains one row for each search message synchronization request. (This is used exclusively for tiered historian installations.)

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) Id	int, NOT NULL	The unique numerical identifier for a search message synchronization request.
JobId	nvarchar(255), NOT NULL	The unique numerical identifier for a search message synchronization job.
Indexing Message	nvarchar(MAX), NOT NULL	Details from the JSON file used for search indexing to make the associated content searchable. This is an example: `{"_messageid":"test_636571577850365608","_tenantid":"","_body":[{"_keywords":[],"_fields":[{"ContentName":"test","_analyzers":"nGram"}],"_id":"afwwGwD30FUMl9UpE`
DocumentType	nvarchar(50), NOT NULL	Specifies the type of index document; for example SavedContent or Tag.
DeQueue Count	smallint, NULL	The number of times this message has been dequeued.
JobSubmisionTimeUtc	datetime2(7), NOT NULL	The time the job was submitted in UTC time.
LastModified TimeUtc	datetime2(7), NULL	The time the job was last modified in UTC time.
Status	smallint, NULL	The status of the search message synchronization job.

ServerList

Contains one row for each server used in an enterprise system. Allows for the creation of the system namespace, which contains a list of servers, and a flat namespace of tags per server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) ServerKey	int NOT NULL	The unique numerical identifier of am AVEVA Historian server. This value is automatically generated by the system when a server is added.
ComputerName	nvarchar(50) NOT NULL	The Microsoft network name of the server computer.
Description	nvarchar(50) NULL	The description of the server.

ShardAssignmentRule

Contains one row for each rule used for assigning tags to particular partitions (shards).

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
Name	nvarchar(255), NOT NULL	The name of the assignment rule.
Priority	int, NOT NULL	The priority assigned to this assignment rule.
TagFilter	nvarchar(4000), NOT NULL	Do not edit. This shows the OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
Enabled	bit, NOT NULL	Indicates whether this rule is enabled. 0 - not enabled; 1 - enabled.
Id	int, NOT NULL	The unique identifier for the object.
ChangeVersion	timestamp, NOT NULL	Internal use only.

ShareMode

Contains one row for each share mode used for InSight content.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(FK) ShareModeKey	int, NOT NULL	A unique identifier for the share mode.
ShareModeName	timestamp, NOT NULL	The name of the InSight content share mode.

SnapshotDetail

Contains one row for each storage size configuration for tags. This table is used by the Classic Storage subsystem to manage the snapshot files.

Column	Data Type	Description
(PK) StorageSize	int NOT NULL	The storage size, in bytes, of the tag value: -1 = Blob; 0 = Variable length string; 1 = 1 byte; 2 = 2 byte; 4 = 4 byte; 8 = 8 byte.
SnapshotSize	int NOT NULL	The maximum size of the snapshot, in bytes. If this limit is reached, a new snapshot is created. The default is 2,097,152 bytes (2 MB).
ImageTime	int NOT NULL	The interval, in seconds, between updates to the snapshot file. The snapshot file is updated with tag value information from the snapshot buffer, which resides in memory. The default is 30 seconds, and the maximum value is 60 seconds.
ThresholdTime	int NOT NULL	The maximum amount of time, in seconds, that can elapse before a new snapshot is automatically created, provided that the value for the snapshot size has not been reached. The default is 3600 seconds (1 hour).
Status	tinyint NULL	Automatically updated by the system if a change is made to the snapshot: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SnapshotTag

Contains one row for each tag that is included in the snapshot action associated with a given event tag.

Column	Data Type	Description
(PK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. This value is automatically generated by the system when the snapshot is added.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. This tag is the snapshot tag. TagName is a foreign key from the Tag table.
(FK) EventTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the event tag to which the snapshot tag is related. EventTagName is a foreign key from the EventTag table.
(FK) TagType	int NOT NULL	Used to indicate the type of tag. 1 = Analog; 2 = Discrete; 3 = String. The default is 1. TagType is a foreign key from the TagRef table.

SQLTemplate

Contains one row for each pre-defined SQL script, which can be copied and used as a basis for an event detection or action script.

Column	Data Type	Description
(PK) TemplateKey	int NOT NULL	The unique numerical identifier for a SQL template. This value is automatically generated when a new SQL template is created.
Description	nvarchar(50) NULL	The description of the SQL template.
Type	int NULL	The type of SQL template. 0 = Detector; 1 = Action.
Script	ntext NULL	A pre-defined SQL script. This script can be copied and used as an event detection or action script.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

StateSummaryHistory (INSQL.Runtime.dbo.StateSummaryHistory)

The StateSummaryHistory extension table returns results for state summary points.

Column	Data Type	Description
TagName	nvarchar(256) NOT NULL	The tag name.
StartDateTime	datetime2(7) NOT NULL	Start time of retrieval cycle.
EndDateTime	datetime2(7) NOT NULL	End time of retrieval cycle.
Value	float NULL	Numeric state.
vValue	nvarchar(4000) NULL	Non-numeric state.
OPCQuality	int NULL	OPC quality. Normal OPC quality retrieval logic is applied if: • All the points found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. • If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. • If the retrieval cycle contains a tag datatype transition (for example, the tag was configured to store integer values initially, then changed to store floating-point values) a DOUBTFUL OPC quality (64) is returned.
StateCount	int NULL	The number of times the state occurred within the retrieval cycle, including states that only partially occur in the cycle.
ContainedStateCount	int NULL	The number of times the state occurred fully contained within the retrieval cycle. States that only partially occur in the cycle are not counted.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StateTimeMin	float NULL	Minimum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeMinContained	float NULL	The minimum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeMax	float NULL	Maximum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeMaxContained	float NULL	The maximum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StateTimeAvg	float NULL	Average time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period.
StateTimeAvgContained	float NULL	Average time in this state among all occurrences of this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeTotal	float NULL	Total time in this state during this retrieval cycle, including state occurrences that fall only partially within the period.
StateTimeTotalContained	float NULL	Total time in this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimePercent	float NULL	Percent of the time during this retrieval cycle that the tag was in this state, including state occurrences that fall only partially within the period.
StateTimePercentContained	float NULL	The precentage of the entire retrieval cycle time that the tag was in this state, excluding state occurrences that fall only partially within the period. This is a ratio between StateTimeTotalContained and StateTimeTotal expressed as a percentage in the range 0 to 100. An occurrence that was partially contained in two or more consecutive storage cycles is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		converted to a contained state within the retrieval cycle if possible.
SourceTag	nvarchar(256) NULL	The source (tier 1) tag for the summary tag.
SourceServer	nvarchar(256) NULL	The source (tier 1) server for the summary tag.
wwCycleCount	int NULL	The number of cycles into which the entire query time range has been divided.
wwResolution	int NULL	Length of cycles in milliseconds. The default is 3600000 (equal to 1 hour).
wwTimeZone	nvarchar(50) NOT NULL	Time zone to use for interpreting both input and output timestamp parameters. If none is specified, then the default is set to LOCAL.
wwVersion	nvarchar(30) NOT NULL	Data version, ORIGINAL or LATEST. If none is specified, the default is LATEST.
wwTagKey	int NOT NULL	Tag key.
wwRetrievalMode	nvarchar(16) NOT NULL	Determines whether to use CYCLIC or DELTA retrieval. The default is DELTA.
wwMaxStates	int NULL	The maximum number of states (for state summaries) that are stored. The first N states will have summary values. For internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

StateWideHistory (INSQL.Runtime.dbo.StateWideHistory)

Contains one row for the amount of time one or more analog, discrete, or string tags have been in a particular state, thus providing a "wide" view of the data.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp for the start of the time-in-state period.
vValue	sql_variant NULL	The string representation of the state, the ordinal for state types that do not have a string representation, or NULL for a gap or "bad" value.
Tag1	float NULL	The name of a tag to query.
Tag2	float NULL	The name of a tag to query.
ManyOtherTags	float NULL	A "placeholder" column for one or more tags in the wide table format. In the wide table format, tagnames are used as column names. The ManyOtherTags column is "duplicated" for as many tags as are specified in the database query.
wwRowCount	int NULL	The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query. This parameter has been deprecated; do not use. Use the wwCycleCount parameter instead.
wwResolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the start date and ending with the end date.
wwRetrievalMode	nvarchar(16) NULL	Used to specify the time-in-state retrieval mode. The valid values are VALUESTATE and ROUNDTRIP. The default wwRetrievalMode is VALUESTATE.
wwTimeDeadband	int NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
wwValueDeadband	real NULL	The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval. The default is 0.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwVersion	nvarchar(30) NULL	If the original data values have been modified in the database, use this column to specify which version of the stored data is to be retrieved. Valid values are: ORIGINAL or LATEST. If no parameter is specified, the latest version of the data is retrieved by default. Modification is indicated by the QualityDetail.
wwCycleCount	int NULL	The number of retrieval cycles (sub-intervals) for the specified time period. The cycles will be spaced evenly across the time period. For example, if you specify a cycle count of four, the time period will be divided into four even cycles, and one or more values (depending on the retrieval mode) will be returned per cycle.
wwTimeStampRule	nvarchar(20) NULL	Used to specify whether cyclic results are timestamped at the beginning of the cycle or at the end of the cycle. Valid values are START and END. If no timestamp rule is specified in the query, then retrieval uses the setting of the TimeStampRule system parameter.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwQualityRule	nvarchar(20) NULL	Used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC. A quality rule of GOOD means that data values with doubtful (64) OPC quality will not be used in the retrieval calculations and will be ignored. Values with bad QualityDetail indicate gaps in the data. A quality rule of EXTENDED means that data values with both good and doubtful OPC quality will be used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data. A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values will not cause the overall calculations to return NULL. You can apply wwQualityRule to all retrieval modes.
wwStateCalc	nvarchar(20) NULL	Used to indicate the type of calculation to return in the StateTime column for the "value state" retrieval mode. Valid values are: MINIMUM, MAXIMUM, AVERAGE, TOTAL, CONTAINED, or PERCENT. You can also use the shortened versions: MIN, MAX, AVG, or SUM. The default for this column is TOTAL.
wwParameters	nvarchar(128) NULL	Used for additional parameters that can be specified. By default, the value of this parameter is an empty string.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StartDateTime	datetime2(7) NOT NULL	Start time of the retrieval cycle for which this row is returned.
wwFilter	nvarchar(512) NOT NULL	The name of the filter. Filters are specified as C-like functions and parentheses are always required, even when the filter does not override the default parameters (no parameters are passed). Filter values are NoFilter, ToDiscrete(), SigmaLimit(), and SnapTo(). The default value is NoFilter. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.
wwMaxStates	int NULL	For internal use only.

StorageLocation

Contains one row for each defined storage location on a specific storage partition (shard).

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	A unique identifier for the partition.
(FK) StorageType	int, NOT NULL	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent, 5 = Tag data index, 6 = Tag data index write-ahead logs, 7 = Revision tag data write-ahead logs. There can be only one storage location of each type.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Path	nvarchar(255), NOT NULL	The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different internal hard drive. If you use a network location, the InSQLConfiguration virtual service account must have full access to the location. You can grant that access by using the credentials of the computer account in the format <domain_name>\<computer_name>$. The locations must be specified using UNC notation. Mapped drives are not supported. If empty, the default <SystemDataPath>\Wonderware\Data\Circular is used.
MaxMBSize	int, NOT NULL	The limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used.
MinMBThreshold	int, NOT NULL	The minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		(before any compression) by 1.5 to determine the minimum threshold.
MaxAgeThreshold	int, NOT NULL	The age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied.
(PK) Id	int, NOT NULL	The unique identifier for the object.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the storage location: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

StorageShard

Contains one row for each storage partition used by the system.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
ShardName	nvarchar(255), NOT NULL	The name of the partition.
Description	nvarchar (512), NULL	The description of the partition.
ComputerName	nvarchar (255), NULL	The network name of the computer on which the storage partition resides.
BlockDuration	int, NOT NULL	Duration, in hours, for history blocks. Valid values are: 1, 2, 3, 4, 6, 8, 12, 24. The default is 24 hours. The history block size must always be greater than the highest scan

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		rate. For more information, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.
TimeUnitId	tinyint, NOT NULL	Foreign key to TimeUnit. Indicates whether the block duration is in hours/days.
TimeZoneId	smallint, NULL	The time zone associated with this storage partition.
AdjustToDST	tinyint, NOT NULL	Internal use only.
MaxSnapshotSize	int, NOT NULL	Maximum size, in MB, for data storage snapshots in memory. Bigger snapshots allow for faster retrieval. You might need to increase this size for systems with very high data rates. For example, if retrieval is slow from data in the current history block, try increasing this rate. Also be sure that you have enough RAM, up to 1 GB.
CmdArgs	nvarchar(4000), NULL	Do not edit. These are command line parameters for customizing replication and storage execution.
CmdExtArgs	nvarchar(4000), NULL	Do not edit. These are command line parameters for customizing replication and storage execution.
Id	int, NOT NULL	The unique identifier for the object.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
UseIndexing	bit, NOT NULL	Determines whether indexing is used during low-level retrieval. 0 = Indexing disabled; 1 = Indexing

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		enabled

StorageType

Contains one row for each type of storage used by the system.

Column	Data Type	Description
(PK) StorageTypeId	int, NOT NULL	A unique identifier for this storage type.
(FK) StorageType	int, NOT NULL	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent, 5 = Tag data index, 6 = Tag data index write-ahead logs, 7 = Revision tag data write-ahead logs. There can be only one storage location of each type.
Description	nvarchar(255), NULL	A description of the storage type.

StringSnapshot

Contains one row for each string tag value that was configured to be stored when a defined event occurred. To view analog, discrete, and string snapshot values at the same time, use the v_SnapshotData view instead. For more information, see v_SnapshotData.

Column	Data Type	Description
(PK) (FK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. SnapshotTagKey is a foreign key from the SnapshotTag table.
(PK) (FK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. EventLogKey is a foreign key from the EventHistory table.
Value	nvarchar(512) NULL	The value of the string tag at the event timestamp.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

StructureAttributes

Contains one row for each attribute definition for the StructureType read-only table.

Column	Data Type	Description
(PK) (FK) StructureId	uniqueidentifier NOT NULL	The unique identifier for the structure. StructureID is a foreign key from the StructureType table.
(PK) AttributeName	nvarchar(255) NOT NULL	The name of the structure attribute.
(FK) AttributeTypeKey	int NOT NULL	The unique identifier for the structure attribute. AttributeTypeKey is a foreign key from the AttributeType table.
AttributeOrder	tinyint NOT NULL	The order of the attribute within the structure.

StructureType

Contains one row for each structure type. Read-only table.

Column	Data Type	Description
StructureId	uniqueidentifier NOT NULL	The unique identifier for the structure.
StructureTypeName	nvarchar(255) NOT NULL	The name of the structure type.
Description	nvarchar(512) NOT NULL	The description of the structure type.

SummaryData

Contains one row for each summarized value, or result, for a tag. This table is used by the event subsystem; it is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

not used by the replication subsystem. The Quality column contains the highest quality value of the raw data from which the result is calculated.

Column	Data Type	Description
(PK) (FK) LogKey	int NOT NULL	The unique numerical identifier of the summary's historical log. LogKey is a foreign key from the SummaryHistory table.
(PK) (FK) SumVarKey	int NOT NULL	The unique numerical identifier for a summarized tag. SumVarKey is a foreign key from the SummaryTagList table.
Value	float NULL	The value of the summary.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
Modified	int NOT NULL	Used to specify whether or not the data has been modified. This value is optional. 1 = Modified; 0 = Not modified.

SummaryHistory

Contains one row for each occurrence of a summary operation. This table is used by the Event subsystem; it is not used by the replication subsystem. Rows are inserted even if the operation did not return data.

Column	Data Type	Description
(PK) LogKey	int NOT NULL	The unique numerical identifier of the summary's historical log. This value is automatically generated by the system when the record is added.
(FK) OperationKey	int NOT NULL	The unique numerical identifier for the summary operation. OperationKey is a foreign key from the SummaryOperation table.
SummaryDate	datetime2(7) NOT NULL	The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
SumDateTimeStamp	tinyint NULL	Duplication of the TimeStamp column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation timestamp setting performed, in case of a later modification of the summary operation definition.
SumDateCalcType	varchar(3) NULL	Duplication of the CalcType column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation type performed, in case of a later modification of the summary operation definition.
SumDateDuration	real NULL	Duplication of the Duration column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation duration used in case of a later modification of the summary operation definition.
SumDateResolution	int NULL	Duplication of the Resolution column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation resolution used, in case of a later modification of the summary operation definition.
Status	tinyint NOT NULL	The flag indicating the status of the operation. 0 = Operation completed successfully; Not 0 = Operation is in progress or has failed. Reserved for future use.
OperationStart	datetime2(7) NULL	The timestamp when the calculation started for the operation.
OperationEnd	datetime2(7) NULL	The timestamp when the calculation completed for the operation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SummaryOperation

Contains one row for each defined summary operation that is associated with the event tag specified in the TagName column. This table is used by the Event subsystem; it is not used by the Replication subsystem.

Column	Data Type	Description
(PK) OperationKey	int NOT NULL	The unique numerical identifier for the summary operation. This value is automatically generated by the system when the operation is added.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the EventTag table.
(FK) CalcType	CalcTypes (char(3)) NOT NULL	The type of calculation to be performed: SUM, MAX, MIN, or AVG. CalcType is a foreign key from the CalcType table.
Description	nvarchar(50) NULL	The description of the summary operation.
Duration	real NOT NULL	The period, in seconds, for which the calculation is performed.
Resolution	int NOT NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
TimeStamp	tinyint NOT NULL	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends. 0 = Beginning of the calculation period; 1 = End of the calculation period.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Frequency	nvarchar(12) NULL	The name for the frequency.
SourceType	varchar(3) NULL	The type of summary, set to 'DYN' (for "dynamic" data). Used for backward compatibility with Industrial Workbook.

SummaryTagList

Contains one row for each combination of a summarized tag and a specific summary operation. This table is used by the Event subsystem; it is not used by the Replication subsystem. This table is a linking table that allows tags to be associated with a type of operation.

Column	Data Type	Description
(PK) SumVarKey	int NOT NULL	The unique numerical identifier for a summarized tag. This value is automatically generated by the system when the summarized tag is added.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) OperationKey	int NOT NULL	The unique numerical identifier for the summary operation. OperationKey is a foreign key from the SummaryOperation table.
LowerLimit	float NULL	The lower limit of validity for the tag's value. Values lower than this limit are not used in the calculation. By default, this value is set to -1000000000.
UpperLimit	float NULL	The upper limit of validity for the tag's value. Values higher than this limit are not used in the calculation. By default, this value is set to 1000000000.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Description	nvarchar(50) NULL	The description of the summarized tag. This normally describes the result of the operation, although this description can be the same as that of the tag on which the operation is performed.

SystemParameter

Contains one row for each system parameter.

Column	Data Type	Description
(PK) Name	nvarchar(50) NOT NULL	The unique name for the system parameter.
Value	sql_variant NULL	The value of the system parameter.
Editable	bit NOT NULL	Used to determine if the value of the named system parameter can be changed using the InSQL Console. 1 = Editable; 0 = Not editable.
Description	nvarchar(255) NULL	The description of the system parameter.
Status	tinyint NULL	Automatically updated by the system if a change is made to the named system parameter: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

TagExtendedPropertyName

Contains the name of each extended tag property stored by AVEVA Historian, namely HierarchicalName and Alias.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) PropertyNameKey	int NOT NULL	A unique identifier for the extended tag property name.
PropertyName	nvarchar (256) NOT NULL	The extended tag property name.
(FK) PropertyType	int NOT NULL	Specifies a type for this extended tag property.
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
Substring Searchable	bit NOT NULL	Specifies whether the name can be located with a substring search.

TagGroup

Contains one row for each tag group used by the system.

Column	Data Type	Description
(PK) GroupID	int, NOT NULL	A unique identifier for the tag group.
Description	nvarchar(50), NULL	A description of the tag group.
CreatedDate	datetime2(7), NULL	The date the tag group was created.
CreatedBy	nvarchar(18), NULL	The person or application that created the tag group.
Type	int, NULL	The type of tags in this group.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagHistory

Historian needs some essential information about how time-series data is stored ("metadata") in order to correctly interpret that data when stored in the history blocks. Since this metadata can change over the life of a tag, the system must preserve the complete metadata record for all history blocks. The ability to store tag metadata to an alternate file location is also possible.

When a tag record gets modified or deleted in the Tag table, the system automatically preserves the previous tag version in the TagHistory table. The TagHistory table should not be modified by the user, otherwise the data stored in the history block may become inaccessible.

Contains one row for each tag metadata instance uniquely identified by the TagId column.

Important: This table is for internal use only. Do not edit this table as it may result in unpredictable behavior. Additionally, Invensys reserves the right to make modifications to the structure/schema of the table as needed.

Column	Data Type	Description
(PK) TagId	uniqueidentifier NOT NULL	The unique identifier for the tag. Internal use only.
TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. Internal use only.
Description	nvarchar(512) NULL	The description of the tag. Internal use only.
AcquisitionType	tinyint NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver. Internal use only.
StorageType	smallint NOT NULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." Internal use only.
StorageRate	int NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		in milliseconds. Internal use only.
TagType	int NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table. Internal use only.
TimeDeadband	int NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. Internal use only.
DateCreated	datetime2(7) NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated. Internal use only.
CreatedBy	nvarchar(256) NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server. CurrentEditor is a foreign key from CurrentEditor table. Internal use only.
ServerTimeStamp	bit NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		data values with the same timestamp, which may affect data calculations, such as for swinging door storage. Internal use only.
DeadbandType	smallint NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. Internal use only.
ChannelStatus	tinyint NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
AIHistory	bit NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage. Internal use only.
Message0	nvarchar(64) NULL	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. Internal use only.
Message1	nvarchar(64) NULL	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. Internal use only.
Unit	nvarchar(32) NULL	The unit of measure. Examples are mph, grams, and pounds. Internal use only.
DefaultTagRate	init NULL	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
IntegralDivisor	float NULL	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only.
MinEU	float NULL	The minimum value of the tag, measured in engineering units. Internal use only.
MaxEU	float NULL	The maximum value of the tag, measured in engineering units.Internal use only.
MinRaw	float NULL	The minimum value of the raw acquired value. Internal use only.
MaxRaw	float NULL	The maximum value of the raw acquired value. Internal use only.
Scaling	int NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use). Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
RawType	int NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes). Internal use only.
ValueDeadband	float NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. Internal use only.
IntegerSize	tinyint NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use). Internal use only.
SignedInteger	bit NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed. Internal use only.
RateDeadband	float NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
InterpolationType	tinyint NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. Internal use only.
RolloverValue	float NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000. Internal use only.
MaxLength	smallint NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512. Internal use only.
DoubleByte	tinyint NULL	Used to specify whether or not to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0. Internal use only.
StructureId	uniqueidentifier NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table. Internal use only.
SourceTag	nvarchar(256) NULL	The name of the source tag used for the replication tag. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
SourceServer	nvarchar(255) NULL	The name of the tier 1 server with the source tag. Internal use only.

TagRef

Contains one row for each tag in the system. This table is used as a reference table for the Tag table, so that the TagName column is not propagated as the primary key of child tables.

Column	Data Type	Description
(PK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain. This value is automatically generated by the system when the tag is added.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian. wwTagKey is populated from the Tag table, but is not a foreign key.
(FK) ServerKey	int NOT NULL	The unique numerical identifier of an AVEVA Historian server. ServerKey is a foreign key from the ServerList table.
(FK) TagName	TagNameType (nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
TagType	int NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 4 = Complex; 5 = Event, 7 = Summary tag (analog or state).

TagType

Contains one row for each tag type.

Column	Data Type	Description
(PK) TagTypeKey	int NOT NULL	The unique identifier for the tag type.
TagTypeName	nvarchar(32) NULL	The name of the tag type.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TimeDetectorDetail

Contains at least one row for each event tag associated with a time detector.

Column	Data Type	Description
(PK) TimeDetectorDetailKey	int NOT NULL	The unique numerical identifier for each time-based event tag. This value is automatically generated by the system when a time-based event tag is created.
(FK) FrequencyID	int NOT NULL	The unique numerical identifier for the frequency. Used to link a frequency with a time-based detector. 1= Hourly; 2 = Daily; 3 = Weekly; 4 = Monthly; 5 = Periodic; 6 = Other (Reserved for future use). FrequencyID is a foreign key from the Frequency table.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
Periodicity	int NULL	The interval period in minutes between detector events. Only used for a periodic detection.
StartDateTime	datetime2(7) NULL	The timestamp from which the time detector starts. Only used for a periodic detection.
RunTimeDay	tinyint NULL	In the context of a weekly detector, RunTimeDay maps the week day number (0 = Sunday – 6 = Saturday). In the context of a monthly detector, RunTimeDay maps to the day of the month. Not used for periodic detections.
RunTimeHour	tinyint NULL	The hour of the day at which the time detector triggers. Not used for periodic detections.
RunTimeMin	tinyint NULL	The minute of the hour at which the time detector triggers. Not used for periodic detections.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TimeDetectorDetailPendingDelete

Contains one row for each time detector that is pending deletion. This table is used internally by the system during the deletion process.

The columns in this table are the same as in the TimeDetectorDetail table.

TimeUnit

Contains one row for each unit of time used by the system.

Column	Data Type	Description
Id	tinyint, NOT NULL	The unique identifier for the object.
(PK) Name	nvarchar(32), NOT NULL	The name of the time unit.

TimeZone

Contains one row for each time zone as defined by the Windows operating system. This table is automatically populated by the system.

Column	Data Type	Description
(PK) TimeZoneID	smallint NULL	The unique numerical identifier for the time zone.
TimeZone	nvarchar(100) NULL	The name of the time zone.
Description	nvarchar(100) NULL	The description of the time zone that includes the hour offset from UTC (GMT) and major cities or regions in the time zone.
Offset	smallint NOT NULL	The offset, in minutes, for daylight savings time, when in effect.
RegistryName	nvarchar(100) NULL	The Windows registry name of the time zone, which is always in English.

TopicImportInfo

Contains one row for each topic definition imported from an InTouch node.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) (FK) NodeKey	int NOT NULL	The unique numerical identifier of the named InTouch node. This value is automatically generated by the system when the node is added.
(PK) DdeSourceKey	int NOT NULL	The unique identifier for the DDE source. Assigned by the AVEVA Historian system when data is imported.
SourceName	nchar(50) NOT NULL	The DDE Access Name from InTouch.
ApplicationName	nchar(50) NULL	The name of the InTouch application from which the topic definition is imported.
TopicName	nchar(50) NOT NULL	The name of the topic definition that is imported.
RequestInitialData	bit NOT NULL	Used to determine if the topic was configured to request initial data. See the InTouch documentation for more information. Internal use only.
AlwaysAdvise	bit NOT NULL	Used to determine if the topic was configured to poll all items for data. See the InTouch documentation for more information. Internal use only.
DefaultStorageRate	int NOT NULL	The cyclic storage rate for the topic. Internal use only.
DefaultStorageType	int NOT NULL	The default storage type for the topic. Internal use only.
TimeDeadband	int NOT NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
ValueDeadband	float NOT NULL	Either the InTouch log deadband or the AVEVA Historian deadband, as specified by the DeadbandType column. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
DeadbandType	tinyint NULL	The type of deadband used. Internal use only.
Import	bit NOT NULL	Used to indicate whether the topic has previously been imported from InTouch into AVEVA Historian. Internal use only.
ProtocolType	tinyint NOT NULL	The protocol used by the AVEVA Historian server to communicate with the I/O Server. Internal use only.
IODriverKey	int NULL	The unique identifier for an IDAS.
RateDeadband	float NOT NULL	The rate deadband that was specified during the InTouch topic import. Internal use only. This rate deadband is not used for swinging door storage. For more information on the rate deadband for swinging door storage, see AnalogSummaryTag.

UserDetail

Contains one row for each AVEVA Historian user. Used to store additional user information that is not stored in the Microsoft SQL Server Runtime.sysusers table. Applicable for both users and groups of users.

When AVEVA Historian is installed, a SQL job is created on the Microsoft SQL Server that automatically updates this table every hour. In order for this job to run, the SQL Server Agent must be running. For more information about jobs, see your Microsoft Online Books.

Column	Data Type	Description
(PK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table.
UserName	nvarchar(128) NOT NULL	The name of the database user.
AccessLevel	int NOT NULL	The security level for the user. 1 is the lowest level and 9999 is the highest. Used to limit access of certain users.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
uid	int NOT NULL	The identifier for the user. This ID is referenced from the Microsoft SQL Server sysusers table.
gid	int NOT NULL	The identifier for the group in which a user belongs. This ID is referenced from the Microsoft SQL Server sysusers table.

WideHistory (INSQL.Runtime.dbo.WideHistory)

Contains one row of values for multiple analog, discrete, or string tags for a single timestamp, thus providing a "wide" view of the data.

Because tagnames are used as column names for the returned data (indicated by Tag1, Tag2, and ManyOtherTags), the value data types will be of the appropriate type for associated tags.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp for the returned value. For delta retrieval, this is typically the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
Tag1	(as per the tag type) NULL	The name of a tag to query.
Tag2	(as per the tag type) NULL	The name of a tag to query.
ManyOtherTags	(as per the tag type) NULL	A "placeholder" column for one or more tags in the wide table format. In the wide table format, tagnames are used as column names. The ManyOtherTags column is "duplicated" for as many tags as are specified in the database query.
wwRowCount	int NULL	The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query. This

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		parameter has been deprecated; do not use. Use the wwCycleCount parameter instead.
wwResolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
wwEdgeDetection	nvarchar(16) NULL	The type of edge detection result set that the query will return. Valid values are NONE, LEADING, TRAILING, and BOTH.
wwRetrievalMode	nvarchar(16) NULL	Used to specify how retrieved data is processed before it is returned to the client. Valid values are: CYCLIC, DELTA, FULL, INTERPOLATED, BESTFIT, AVERAGE, MINIMUM, MAXIMUM, INTEGRAL, SLOPE, COUNTER, VALUESTATE, and ROUNDTRIP. FULL = All stored values are returned; CYCLIC = All stored data for tags during the specified time interval are returned for the number of retrieval cycles or resolution specified; DELTA = Only values that changed during the specified time interval are returned. For all other modes, a calculation is performed by the system on the data and the value(s) are returned. The default is CYCLIC for retrieval from analog tables,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		DELTA for retrieval from discrete and string tables, and default is DELTA for retrieval from the History table, unless the specific retrieval mode implies otherwise. For example, SLOPE always has DELTA characteristics. The default value for wwRetrievalMode is DELTA.
wwTimeDeadband	int NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
wwValueDeadband	real NULL	The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval. The default is 0.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		from the TimeZone table.
wwVersion	nvarchar(30) NULL	If the original data values have been modified in the database, use this column to specify which version of the stored data is to be retrieved. Valid values are: ORIGINAL or LATEST. If no parameter is specified, the latest version of the data is retrieved by default. Modification is indicated by the QualityDetail.
wwCycleCount	int NULL	The number of retrieval cycles (sub-intervals) for the specified time period. The cycles will be spaced evenly across the time period. For example, if you specify a cycle count of four, the time period will be divided into four even cycles, and one or more values (depending on the retrieval mode) will be returned per cycle.
wwTimeStampRule	nvarchar(20) NULL	Used to specify whether cyclic results are timestamped at the beginning of the cycle or at the end of the cycle. Valid values are START and END. If no timestamp rule is specified in the query, then retrieval uses the setting of the TimeStampRule system parameter.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwInterpolationType	nvarchar(20) NULL	Used to determine which analog value to return at a given cycle boundary. Valid values are STAIRSTEP and LINEAR. If STAIRSTEP is specified, no interpolation occurs. The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned. If LINEAR is specified, the system calculates a new value at the given cycle time by interpolating between the last known value prior to the cycle time and the first value after the cycle time.
wwQualityRule	nvarchar(20) NULL	Used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC. A quality rule of GOOD means that data values with doubtful (64) OPC quality will not be used in the retrieval calculations and will be ignored. Values with bad QualityDetail indicate gaps in the data. A quality rule of EXTENDED means that data values with both good and doubtful OPC quality will be used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data. A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values will not cause the overall calculations to return NULL. You can apply wwQualityRule to all retrieval modes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
wwValueSelector	nvarchar(20) NULL	Used to specify which column to return for specified analog summary tags in the four basic retrieval modes, DELTA, FULL, CYCLIC, and INTERPOLATED. The defined set of selectors are AUTO (the default in all modes if not overridden), MINIMUM or MIN, MAXIMUM or MAX, FIRST, LAST, AVERAGE or AVG, and INTEGRAL. The default AUTO setting returns the Last attribute in the Value column (which makes it accessible in the WideHistory table). You can only override the selector for the basic retrieval modes.
wwFilter	nvarchar(512) NOT NULL	Gives the name of the filter. Filters are specified as C-like functions and parentheses are always required, even when the filter does not override the default parameters (no parameters are passed). Filter values are NoFilter, ToDiscrete(), SigmaLimit(), SnapTo(), and SLR(). The default value is NoFilter. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.
wwParameters	nvarchar(128) NULL	Contains the "stream index" (used for informational purposes only) and the special index value to indicate that the value was calculated by the "SLR()" filter. SLR stands for "simple linear regression," the algorithm used for predictive retrieval. By default, the value of this parameter is an empty string.
StartDateTime	datetime2(7) NOT NULL	Start time of the retrieval cycle for which this row is returned.
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

WideTableDictionary

Contains unique identifiers for tags and users.

Column	Data Type	Description
(PK, FK) UserKey	int, NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Views

A view is a logical way of looking at data from one or more tables in the database. A view is a "virtual" table that does not actually exist in the database. A view contains pointers to the actual tables in the database. Views can be used to include a subset of information stored in one or more tables, while leaving out other information. Views are part of normal SQL Server functionality.

To make it easier to query data from some of the AVEVA Historian tables, a number of views onto the data are provided. Queries are performed on these views as if they were normal physical tables.

Note: Some views are included in the database for backward compatibility support only. It is recommended that you discontinue the use of these views, as they will be dropped in a future release. For details about views retained for backward compatibility, see Backward Compatibility Views.

History Table Views

Views have been created for the extension tables to make querying these tables easier. Normally, you must use the full reference for an extension table in the query, which is **linked_server.catalog.schema.objectname**. An extension table view allows you to simply use the view name instead, eliminating the need to provide the long reference.

All of the following views reflect the same table structure as the extension tables after which they are named.

This view	References this extension table
History	INSQL.Runtime.dbo.History
HistoryBlock	INSQL.Runtime.dbo.HistoryBlock
Live	INSQL.Runtime.dbo.Live
AnalogSummaryHistory	INSQL.Runtime.dbo.AnalogSummaryHistory
StateSummaryHistory	INSQL.Runtime.dbo.StateSummaryHistory

Note: Some History table views are included in the database for backward compatibility support only. It is recommended that you discontinue the use of these views, as they will be dropped in a future release. For details about History table views retained for backward compatibility, see History Table Views (Backward Compatible).

Annotation

Contains one row for each user annotation about a tag value. Users can make personal (or public) notes about a tag value. This information is stored with the tag value and timestamp to which the annotation applies. Each annotation in this view is linked to a database user.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) AnnotationKey	int NOT NULL	The unique numerical identifier for the annotation. This value is automatically generated by the system when the annotation is added.
(FK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(FK) TagName	TagNameType (nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
DateCreated	datetime(2)7 NULL	The date that the annotation was created.
Content	nvarchar(1000) NOT NULL	The annotation text.
DateTime	datetime2(7) NOT NULL	The timestamp of the tag value for which the user has made an annotation.
Value	float NULL	The value of the tag at the time of the annotation.

Events

Returns one row for each alarm and event.

Column	Data Type	Description
ID	GUID	Unique identifier for the event
EventTime	DateTime2	Time of the event, returned by default in the local time for the Historian server. **Note:** EventTime is modified by wwTimeZone criteria in the query, similar to the way DateTime is in the History table.
EventTimeUtc	DateTime2	UTC time stamp showing when the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		event occurred.
EventTimeUTCOffsetMIns	Integer	Offset, in minutes, between the Historian server time and UTC time.
Type	String	Main categorization of the event: Alarm.Set, Alarm.Clear, Alarm.Acknowledged, Alarm.Silenced.Set, Alarm.Silenced.Clear, Alarm.Suppressed.Set, Alarm.Suppressed.Clear, Alarm.Disabled.Set, Alarm.Disabled.Clear, User.Write, User.Write.Secured, User.Write.Verified, System.Deploy, System.Undeploy, Engine.Start, Engine.Stop, Engine.OnScan, Engine.OffScan, Engine.Active, Engine.Standby, or Engine.Terminated.
Priority	Int32	Event importance. Values range from 1 to 999, with lower numbers indicating higher importance.
Severity	Int32	Urgency categorization of alarms or events: 1=Critical, 2=High, 3=Medium, 4=Low.
IsAlarm	Bool	Indicator telling whether the message is an alarm type message.
Source_ProcessVariable	String	Process variable to which the event is related. For example, if "TI101" has a field attribute "PV" and this is a "Hi" alarm, this is TI101.PV". This will normally match the name of the associated Historian tagname. In a User.Write or similar event, this is the attribute being written to. Also see the Source_ConditionVariable column.
Source_Units	String	Engineering units used for process variable. For example: feet, pounds, or N/m^2.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Alarm_Acknowledged	Boolean	Indicates whether the alarm was acknowledged (true or false).
Alarm_Class	String	(Legacy) InTouch alarm class..
Alarm_Condition	String	Condition being alarmed. It should be one of these predefined values, if appropriate: Limit.HiHi, Limit.Hi, Limit.Lo, Limit.LoLo, ROC.Hi, ROC.Lo, System, Discrete.True, Discrete.False, Deviation.Minor, Deviation.Major, SPC, Batch, Process, Software, System.
Alarm_DurationMs	Int32	Time, in milliseconds, between when the alarm is raised and when it is cleared.
Alarm_ID	String	Unique identifier for an instance of an alarm. This is used to link different alarm events together from the time the alarm is raised until the time it is no longered considered an alarm.
Alarm_InAlarm	Boolean	State of the alarm after the transition (true or false).
Alarm_IsShelved	Boolean	Indicator showing whether alarm was shelved (true or false).
Alarm_IsSilenced	Boolean	Indicator showing whether alarm is silenced (true or false).
Alarm_LimitString	String	Limit being alarmed.
Alarm_OriginationTime	DateTime2	Date/Time stamp for when the initial alarm condition originated.
Alarm_ShelveDurationMs	Int32	The duration, in milliseconds, for which the alarm was shelved. Used only if alarm is currently shelved.
Alarm_ShelveReason	String	Reason for shelving. Used only if alarm is currently shelved.
Alarm_ShelveStartTime	DateTime2	Date/time stamp showing

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		scheduled start of shelve time, if the alarm is shelved. Returned by default in local time for the Historian server. Used only if alarm is currently shelved.
Alarm_State	String	State of the alarm (UNACK_ALM, UNACK_RTN, ACK_ALM, or ACK_RTN).
Alarm_TagType	String	Types of tags associated with the alarm (S, I, or F).
Alarm_Type	String	(Legacy) InTouch alarm type.
Alarm_UnAckDurationMs	Int32	Time, in milliseconds, between when the alarm is raised and when it is acknowledged. This property should not be included for an alarm until it is acknowledged.
Comment	String	Additional comments or descriptions for the event.
IntouchType	String	Legacy InTouch event type. Valid values are: ALM, RTN, ACK, and SYS.
Provider_ApplicationName	String	Name of the application that generated the event. For Application Server, this is the galaxy name. For InTouch, this is the InTouch application name.
Provider_InstanceName	String	Provider-specific string that uniquely identifies the instance on a given node name for the given application.
Provider_NodeName	String	Name of the node that generated the event. For example, "AOS-N-01".
Provider_System	String	Software system that generated the event. For example, Application Server, InTouch, or InBatch.
Provider_SystemVersion	String	Software version for the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		component identified by Provider_ApplicationName.
ReceivedTime	DateTime2	Time at which the the Historian server received the event. Returned by default in the local time for the Historian server.
Source_Area	String	Non-hierarchical area name. For example, "Area_001".
Source_ConditionVariable	String	Related condition variable. For example, if "TI101" has a field attribute "PV" and this is a "Hi" alarm, this value would be "TI101.PV.Hi". Also see the Source_ProcessVariable column.
Source_Engine	String	Non-hierarchical engine name. For example, "AppEngine_001".
Source_HierarchicalArea	String	Hierarchical area name. For example, "Plant_001.Building_002.Area_001 ".
Source_HierarchicalObject	String	Hierarchical name of source object. For example, For example, "Reactor_001.TIC".
Source_Object	String	Non-hierarchical name of source object. For example, "TIC101".
Source_Platform	String	Non-hierarchical platform name. For example, "WinPlatform_001".
User_Account	String	Login name for the operator for the specified application. Depending on the security model being used, this can be <Domain>\<Alias>, email address, or user profile name. For Application Server/ GalaxySecurity Model: User Profile Name For Application Server/OS Model: OperatorDoman\Login For InTouch: Operator Login

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		For Windows Live Login: TBD
User_Agent	String	Name of the application that the user was running when the event was generated.
User_Email	String	User email address. Note: This property is not currently in the OData metadata for events, but may be added in the future.
User_Name	String	Complete first name and last name. For example, "Chris Walton".
User_NodeName	String	Computer name from which the user performed the action. For example, "ww112.baytown.amci.com".
User_Phone	String	User's phone number. For example, "1-800-555-1212". Note: This property is not currently in the OData metadata for events, but may be added in the future.
ValueString	String	The value that triggered the alarm, represented as a string. The maximum length of this string is 329 characters. **Note:** We recommend that you do not use this property except for legacy purposes, as extensive use of this property can negatively affect performance.
Verifier_Account	String	Login name for the verifier. Depending on the security model being used, this can be <Domain>\<Alias>, email address, or user profile name. For Application Server/ GalaxySecurity Model: User Profile Name For Application Server/OS Model: OperatorDoman\Login For InTouch: Operator Login

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		For Windows Live Login: TBD
Verifier _Agent	String	Name of the application that the verifier was running.
Verifier _Email	String	Verifier email address. Note: This property is not currently in the OData metadata for events, but may be added in the future.
Verifier _Name	String	Complete first name and last name
Verifier _NodeName	String	Computer name from where the verification occurred. For example, "ww112.baytown.amci.com".
Verifier _Phone	String	Verifier's phone number. For example, "1-800-555-1212". Note: This property is not currently in the OData metadata for events, but may be added in the future.
wwTimeZone	String	Specifies the time zone for this record.

IODriver

Contains one row for each IDAS providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IODriverKey	int NOT NULL	The unique identifier for an IDAS. This value is automatically generated by the system when the IDAS is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
ComputerName	nvarchar(255) NOT NULL	The name of the computer on which the IDAS runs.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, redundant IDAS runs. You must use the fully qualified name of the computer. You could also use the IP address. This should be set to an empty string if no redundant IDAS is specified. Make sure that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online. By default, this column is an empty string.
StoreForwardMode	tinyint NOT NULL	Used to specify whether or not store-and-forward capability is enabled. If enabled, and the network connection between the IDAS and the storage node fails, data will be "buffered" to the location specified by the store-and-forward path. Valid values are: 0 = Disabled; 1 = Enabled; 2 = Autonomous. The Autonomous mode (2) is an extension of the normal store-and-forward mode (1). It allows the IDAS to start up using an IDAS configuration file and collect data in store-and-forward mode if the network connection to the AVEVA Historian is not available.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
StoreForwardPath	nvarchar(255) NULL	Used to specify the path for the IDAS data buffer on the local hard drive of the IDAS computer. The path should be absolute (for example, c:\IDASBuffer). Data is written to this path until the minimum threshold for the buffer is reached. Remote buffer paths are not supported. When the store-and-forward path specified for the IDAS is invalid, the default path picked by the system is: <public folder>\ArchestrA\Historian\IDAS\SF where the <public folder> is dependent on the operating system. For example, for the Windows 2008 operating system, the path is C:\ProgramData\ArchestrA\Historian\IDAS\SF. When the store-and-forward path specified for the IDAS is just a folder name (without any path characters like \ and :), the default path picked by the system is: <Windows system path>\<folder name specified by the user>. For example, for the Windows Server 2008 32-bit operating system, the path is C:\WINDOWS\system32\<folder name>.
MinMBThreshold	int NOT NULL	The minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer.
Status	tinyint NULL	Automatically updated by the system if a change is made to IDAS: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
Enabled	bit NOT NULL	Used to indicate whether the IDAS is enabled or not. 0 = Not enabled; 1 = enabled. Disabling the IDAS allows for the configuration to be retained in the database, even though the IDAS is removed from

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		the system.
StoreForwardDuration	int NOT NULL	The minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
AutonomousStartupTimeout	int NOT NULL	The amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates.
FileChunkSize	int NOT NULL	The size, in bytes, of the data "chunks" that are sent to the historian when store-and-forward data is forwarded. The size of the chunks can be decreased to accommodate slower networks. Decrease this number only if the forwarding delay is greater than zero.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ForwardingDelay	int NOT NULL	The time interval, in milliseconds, at which "chunks" of store-and-forward data are forwarded to the historian. The length of the interval may need to be increased to accommodate slower networks.
ConnectionTimeout	int NOT NULL	The amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/ reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks.

IOServer

Contains one row for each I/O Server providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IOServerKey	int NOT NULL	The unique numerical identifier for the I/O Server. This value is automatically generated by the system when the I/O Server is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
(FK) IODriverKey	int NULL	The unique identifier for an IDAS. IODriverKey is a foreign key from the IODriver table.
(FK) ApplicationName	nvarchar(32) NULL	The application name of the I/O Server. This name is usually the same as the executable file name. ApplicationName is a foreign key from the IOServerType table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Path	nvarchar(255) NULL	The full UNC path (including the filename) to locate the executable file for the I/O Server. If the I/O Server type key is specified, the filename may be omitted.
ComputerName	nvarchar(255) NULL	The name of the computer on which the I/O Server runs.
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made.
AutoStart	bit NOT NULL	Used to control how the I/O Server starts up. 0 = Automatic startup when the system starts. 1 = Manual startup required. Currently not used.
ExeType	int NOT NULL	The type of executable for the I/O Server. Used by the Operations Control Management Console to determine how to start the I/O Server. 0 = Service; 1 = Console application; 2 = Windows application.
InitializationStatus	tinyint NOT NULL	A control flag used to ensure that each I/O Server has been asked for the data type (integer or real) of each tag that it will send. Only needed after a database modification.
ProtocolType	tinyint NOT NULL	The protocol used by the AVEVA Historian server to communicate with the I/O Server. 1 = DDE; 2 = SuiteLink™; 3 = AVEVA Historian named pipe driver (for compatibility with IndustrialSQL Server 3.0 and previous versions). Of the operating systems currently supported by the AVEVA Historian, DDE is only supported on the Windows XP operating system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Description	nvarchar(50) NULL	The description of the I/O Server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the I/O Server: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationSyncRequestInfo

Contains one row for each replication synchronization request. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
SourceTagName	TagNameType (nvarchar(256)) NULL	The name of the source tag used for the replication tag.
ReplicationServerName	nvarchar(255) NULL	The name of the replication server.
DestinationTagName	TagNameType (nvarchar(256)) NULL	The name of the destination tag.
EarliestExecutionDateTimeUtc	datetime2(7) NULL	The earliest execution date (in UTC) for the replication synchronization request.
ModStartDateTimeUtc	datetime2(7) NOT NULL	The start time (in UTC) for the replication synchronization request.
ModEndDateTimeUtc	datetime2(7) NOT NULL	The end time (in UTC) for the replication synchronization request.
ReplicationSyncRequestKey	bigint NOT NULL	The unique identifier for the replication synchronization request.
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
RequestVersion	smallint NOT NULL	The version type. 0 = Initial version; 1 = Latest version.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ExecuteState	tinyint NOT NULL	Value automatically changes as the rep service processes the sync queue. 0 = ready to process; 1 = currently being processed; 2 = rows needs merging/unmerging.
CurrentEditor	tinyint NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
DestinationTagID	uniqueidentifier NULL	The unique identifier for the destination tag.
MaximumStates	tinyint NULL	Maximum number of states to track for state summary tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ReplicationGroupKey	int NULL	The unique identifier for the replication group.
ReplicationServerKey	int NULL	The unique identifier for the replication server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication server: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
AuthenticateWithAAUser	bit NULL	1 if the login should be authenticated using the the Network Account (previously called the ArchestrA User); otherwise, 0 to authenticate with the UserName and Password.
Bandwidth	int NULL	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited.
BufferCount	int NULL	The number of buffers.
Description	nvarchar(512) NULL	The description of the replication server.
MinSFDuration	int NULL	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
Password	nvarchar(512) NULL	The password for logging in to the replication server. (AuthenticateWithAAUser must be 0.)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
SFFreeSpace	int NULL	The free space for the store-and-forward path in MB.
SFPath	nvarchar(260) NULL	The local store-and-forward path associated with the replication server for this instance of AVEVA Historian.
ServerDefaultSimpleReplicationNamingScheme	nvarchar(512) NULL	Naming rule for simple replication tags. If NULL the naming rule specified in the SimpleReplicationNamingScheme system parameters is used.
ServerDefaultSummaryReplicationNamingScheme	nvarchar(512) NULL	The default naming rule for summary replication tags. If NULL, the naming rule specified in the SummaryReplicationNamingSchem e system parameter is used.
TCPPort	int NULL	The TCP port to use to log in to the replication server.
UserName	nvarchar(255) NULL	The user name for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
GroupAbbreviation	nvarchar(32) NULL	The abbreviation for the replication group.
ReplicationGroupName	nvarchar(255) NULL	The unique identifier for the replication group.
ReplicationScheduleKey	int NULL	The unique identifier for the schedule.
ReplicationTypeKey	tinyint NULL	Can be 1, 2, or 3. (1 = Simple Replication, 2 = Analog Summary Replication, 3 = State Summary Replication.)
GroupDefaultSummaryReplicationNamingScheme	nvarchar(512) NULL	The group default naming rule for summary replication tags.
CompressionEnabled	bit NULL	0 = Compression off. 1 = Enable compression for the packets sent to the replication server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
ConnectionDetails	nvarchar(1024) NULL	Internal use only.

ReplicationTagExtendedPropertyInfo

Contains one entry for each replicated tag extended property.

Column	Data Type	Description
ReplicationTag ExtendedProperty Key	int NOT NULL	A unique identifier for the replication tag extended property.
ReplicationServer Key	int NOT NULL	The unique identifier for the replication server.
DestinationTag Name	nvarchar (256) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
SourceTagName	nvarchar (256) NOT NULL	The unique name for the source tag.
PropertyNameKey	int NULL	A unique identifier for the extended tag property name.
PropertyName	nvarchar (256) NOT NULL	The extended tag property name.
PropertyType	int NOT NULL	Specifies a type for this extended tag property.
PropertyValue	sql_variant NOT NULL	The value of this replication tag extended property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
SubstringSearchable	bit NOT NULL	Specifies whether the name can be located with a substring search.
AttributeTypeName	nvarchar (255) NOT NULL	The name of the attribute type.
AttributeTypeValue	tinyint NOT NULL	The bit mask for the attribute type
ChangeVersion	timestamp NOT NULL	Internal use only.
Status	tinyint NOT NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

Tag

Contains one row for each tag in the system and includes the basic definition for the tag, such as the I/O Server that supplies the values.

Column	Data Type	Description
(PK) TagName	TagNameType (nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
(FK) IOServerKey	int, NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.
(FK) TopicKey	int, NULL	The unique numerical identifier for the topic. TopicKey is a foreign key from the Topic table.
Description	nvarchar(512), NULL	The description of the tag.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver.
StorageType	smallint, NOT NULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored."
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds.
ItemName	nvarchar(256), NULL	The address string of the tag.
(FK) TagType	int, NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
TimeDeadband	int, NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.
DateCreated	datetime(2) 7, NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated.
CreatedBy	nvarchar(256), NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated.
(FK) CurrentEditor	tinyint, NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
ServerTimeStamp	bit, NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage.
DeadbandType	smallint, NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband.
CEVersion	tinyint, NOT NULL	The version number used to track changes to the information in the Tag table. Any change to the data in a row will cause the version indicator to change. The Configuration Editor (and other client tools) can detect the changed version and reload the corresponding tag details. Changes to this column are not tracked by the modification tracking system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
TagId	uniqueidentifier NOT NULL	The unique identifier for the tag.
(FK) ChannelStatus	tinyint, NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table.
AIHistory	bit, NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage.
ChangeVersion	timestamp, NOT NULL	Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
ServerTimeStamp	bit, NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		timestamp, which may affect data calculations, such as for swinging door storage.
(FK) MessageKey	int, NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.
(FK) EUKey	int, NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float, NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float, NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float, NULL	The minimum value of the raw acquired value.
MaxRaw	float, NULL	The maximum value of the raw acquired value.
Scaling	int, NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use).
RawType	int, NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float, NULL	The percentage of the difference between the minimum and

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
IntegerSize	tinyint,, NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit, NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.
RateDeadband	float, NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.
InterpolationType	tinyint, NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
RolloverValue	float, NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.
MaxLength	smallint, NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint, NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0.
(FK) StructureId	uniqueidentifer, NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.
SourceTag	nvarchar(256), NULL	The name of the source tag used for the replication tag.
SourceServer	nvarchar(255), NULL	The name of the tier 1 server with the source tag.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description

TagExtendedPropertyInfo

Contains one entry for each extended property for a tag.

Column	Data Type	Description
TagName	nvarchar (256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Property NameKey	int NULL	A unique identifier for the extended tag property name.
Property Name	nvarchar (256) NOT NULL	The extended tag property name.
PropertyType	int NOT NULL	Specifies a type for this extended tag property.
Property Value	sql_variant NOT NULL	The value of this replication tag extended property.
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
Substring Searchable	bit NOT NULL	Specifies whether the name can be located with a substring search.
AttributeTypeName	nvarchar (255) NOT NULL	The name of the attribute type.
AttributeTypeValue	tinyint NOT NULL	The bit mask for the attribute type
Change Version	timestamp NOT NULL	Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagExtendedPropertyNameInfo

Contains the name of the extended properties currently stored by AVEVA Historian, namely HierarchicalName and Alias.

Column	Data Type	Description
Property Name	nvarchar (256) NOT NULL	The extended tag property name.
PropertyType	int NOT NULL	Specifies a type for this extended tag property.
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
Substring Searchable	bit NOT NULL	Specifies whether the name can be located with a substring search.
AttributeTypeName	nvarchar (255) NOT NULL	The name of the attribute type.
AttributeTypeValue	tinyint NOT NULL	The bit mask for the attribute type.

Topic

Contains one row for each topic to be read from an I/O Server.

Column	Data Type	Description
(PK) TopicKey	int NOT NULL	The unique numerical identifier for the topic. This value is automatically generated by the system when the topic is added.
(PK) (FK) IOServerKey	int NOT NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
Name	nvarchar(180) NOT NULL	The name of the topic.
TimeOut	int NOT NULL	The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic.
Status	tinyint NULL	Automatically updated by the system if a change is made to the topic: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
LateData	bit NOT NULL	Used to enable acquisition of "late" data. 0 = Late data disabled; 1 = Late data enabled.
IdleDuration	int NOT NULL	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds.
ProcessingInterval	int NOT NULL	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

v_EventSnapshot

Returns one row for each snapshot value for an analog and/or discrete tag (specified by the TagName column) associated with a particular snapshot event (specified by the Event column).

Column	Data type	Description
Event	nvarchar(256) NULL	The unique name of the tag within the AVEVA Historian system.
EventTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectionTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the event system.
Edge	nvarchar(8) NOT NULL	The "edge" for the event detection. For more information on edge detection, see Edge Detection for Events (wwEdgeDetection) in the *AVEVA Historian Retrieval Guide*.
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float(8) NULL	The value of the tag at the time of the event occurrence. Measured in engineering units.
Quality	tinyint NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

Note: When an event is not linked to a snapshot action, the TagName column is set to '-' and the Value, Quality, and QualityDetail columns are set to NULL.

v_EventStringSnapshot

Returns one row for each snapshot value for a string tag (specified by the TagName column) associated with a particular snapshot event (specified by the Event column).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	data type	Description
Event	TagNameType(nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system.
EventTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectionTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the event system.
Edge	nvarchar(8) NOT NULL	The "edge" for the event detection.
TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512) NULL	The value of the string tag at the event timestamp.
Quality	tinyint NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

v_ModTracking

Returns one row for each database modification made. For more information, see Viewing Database Modifications in the *AVEVA Historian Administration Guide.*

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The timestamp of when the modification occurred.
Table	varchar(50) NULL	The name of the modified object.
Column	nvarchar(128) NOT NULL	The name of the modified column.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
ModType	char(1) NOT NULL	The type of modification. U = Update; I = Insert; D = Delete; 1 = SQL insert; 2 = SQL original insert; 3 = SQL update; 4 = CSV insert; 5 = CSV original insert; 6 = CSV update; 7 = CSV multi-point update; 8 = CSV "fast load" insert.
RowKey	sql_variant NOT NULL	The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.
NewValue	sql_variant NULL	The new value stored in the column, if the modification was to a configuration table. For modifications to history data, this column contains the total count of consecutive value updates attempted.
OldValue	sql_variant NULL	The value stored in the column before the modification was made, if the modification was to a configuration table. For modifications to history data using SQL INSERT and UPDATE statements, this column contains the timestamp of the earliest data affected by the INSERT or UPDATE operation. If multiple changes are made to the same data, then only the most recent change will be contained in this column. This column is not used for modifications made to history data using a CSV file.
User	nvarchar(256) NOT NULL	The name of the database user that made the modification. The value of this column reflects the Windows authentication user name (for example, DOMAIN\ user_login_name) or the SQL Server authentication user name (for example, dbo), depending on how the user is logged into the SQL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
		Server when the modification is made. In the case of a CSV file import, this column contains the user name as it appears in the CSV file.

v_SnapshotData

Returns one row for each snapshot value for an analog, discrete, and/or string tag (specified by the TagName column) associated with a particular snapshot event (specified by the Event column).

Column	Data type	Description
Event	TagNameType(nvarchar(256)) NULL	The name of the event tag to which the snapshot tag is related.
EventTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectionTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the Event system.
Edge	nvarchar(8) NOT NULL	The "edge" for the event detection.
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512) NULL	The value of the snapshot tag at the event timestamp.
Quality	tinyint NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Stored Procedures

Some stored procedures are useful when performing database queries to return information about specific tags in the system. These stored procedures allow you to return information on a tag's definition or to narrow the scope of a query on a data storage table. You can use these stored procedures when querying the database using ad-hoc query tools, such as SQL Server Management Studio.

Other stored procedures are used to configure AVEVA Historian. System stored procedures are normally run during startup and during changes to the system configuration. These stored procedures are used mainly by the historian setup program, the Event subsystem, the Operations Control Management Console, and client applications.

Note: Stored procedures prefixed with "ww_" are provided only for backward compatibility and are deprecated. For more information, see Renamed Stored Procedures.

Stored Procedures

aaActionStringSelect

Selects the action string for a specified event tag.

Syntax

aaActionStringSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddAnalogSummaryTag

Add an analog summary tag.

Syntax

aaAddAnalogSummaryTag *TagName, TagId, Description, SourceTag, SourceServer, SourceTagScaling, SourceTagRawType, SourceTagIntegerSize, SourceTagSignedInteger, CreatedBy, DateCreated, StructureId, AcquisitionType, StorageNodeKey, IOServerKey, TopicKey, StorageType, EngineeringUnit, IntegralDivisor, MinEU, MaxEU, MinRaw, MaxRaw, DeadbandType, TimeDeadband, CurrentEditor, wwTagKey, ChannelStatus*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
TagId	The unique tag ID of the tag within the AVEVA Historian system. The value is of data type uniqueidentifier, with a default of NULL.
Description	The description of the analog summary tag. This value is of data type nvarchar(512), with a default of an empty string.
SourceTag	The name of the source tag to create the analog summary tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
SourceTagScaling	Used to specify whether the value is scaled. 0 = Not scaled. 1 = scaled. This value is of data type int, with a default of NULL.
SourceTagRawType	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes) 2 = MS Float (4 bytes) 3 = Integer (2 or 4 bytes) 4 = MS Double (reserved for future use) (8 bytes) This value is of data type int, with a default of 3.
SourceTagIntegerSize	The bit size of the analog tag. 12 = 12-bit 15 = 15-bit 16 = 16-bit 32 = 32-bit 64 = 64-bit (reserved for future use) This value is of data type tinyint, with a default of 16.
SourceTagSignedInteger	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	0 = Unsigned 1 = Signed This value is of data type bit, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
StructureId	The unique identifier for the structure. The value is of data type uniqueidentifier, with a default of NULL.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
EngineeringUnit	The unit of measure. Examples are mph, grams, and pounds. This value is of data type nvarchar(32), with a default of an empty string.
IntegralDivisor	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. This value is of data type float(25), with a default of 1.
MinEU	The minimum value of the tag, measured in engineering units. This value is of data type float, with a default of 0.
MaxEU	The maximum value of the tag, measured in engineering units. This value is of data type float, with a default of 100.
MinRaw	The minimum value of the raw acquired value. This value is of data type float, with a default of 0.
MaxRaw	The maximum value of the raw acquired value. This value is of data type float, with a default of 4095.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationGroup

Add or modify replication groups.

Syntax

aaAddReplicationGroup *ReplicationGroupName, ReplicationServerName, ReplicationTypeKey, ReplicationScheduleName, SummaryReplicationNamingScheme, GroupAbbreviation, ReplicationGroupKey*

where:

Argument	Description
ReplicationGroupName	The name of the replication group. This parameter has to be specified, else will return an error. This value is of data type nvarchar(255), with no default.
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 3.
ReplicationScheduleName	The name of the schedule. This value is of data type nvarchar(255), with no default.
SummaryReplicationNamingScheme	The naming scheme for summary replication tags. If not specified, the one specified in the ReplicationServer will be used. This value is of data type nvarchar(512), with a default of NULL.
GroupAbbreviation	Used as part of naming. If not specified, the one specified in the Schedule will be chosen as group abbreviation. This value is of data type nvarchar(32), with a default of NULL.
ReplicationGroupKey	Unique identifier for the replication group. If specified, this will overwrite the properties of the replication group. This value is of data type int, with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationRule

Add or modify a replication rule.

Syntax

aaAddReplicationRule *Name, Priority, TagFilter, ReplicationGroupKey, ReplicationServerKey, Enable, ApplyOtherRules, AutoReplication*

where:

Argument	Description
Name	The name of the replication rule.
Priority	The priority for the rule.
TagFilter	The OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
ReplicationGroupKey	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
ReplicationServerKey	The unique identifier for the replication server.
Enable	Used to indicate whether the replication rule is enabled. 0 - not enabled; 1- enabled
ApplyOtherRules	Used to indicate whether other rules apply. 0 - other rules do not apply; 1- other rules apply.
AutoReplication	Used to indicate whether autosummary is enabled. 0 - disabled; 1- enabled.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddReplicationSchedule

Add or modify the schedules for replication.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaAddReplicationSchedule *ReplicationScheduleName, ReplicationScheduleTypeName, ReplicationScheduleAbbreviation, CreateGroup, Period, Unit, TimesOfDay, ReplicationScheduleKey*

where:

Argument	Description
ReplicationScheduleName	The name of the schedule. This parameter is required. This value is of data type nvarchar(255), with no default.
ReplicationScheduleTypeName	The name of the schedule type. Can be either INTERVAL or CUSTOM. This value is of data type nvarchar(32), with a default of INTERVAL.
ReplicationScheduleAbbreviation	Will be used when creating groups as group abbreviation if not specified. This value is of data type nvarchar(32), with a default of the schedule abbreviation.
CreateGroup	If set to TRUE, groups will be created automatically when the replication server is created. This value is of data type bit, with a default of 1.
Period	The period value. This parameter is required when the schedule type is INTERVAL. This value is of data type smallint, with a default of 0.
Unit	The name of the unit. This parameter is required when the schedule type is INTERVAL. This value is of data type nvarchar(32), with a default of an empty string.
TimesOfDay	A semicolon-separated list of the times of day. This parameter is required when the schedule type is CUSTOM. This value is of data type nvarchar(max), with a default of an empty string.
ReplicationScheduleKey	The unique identifier for the schedule. If specified, this will overwrite the properties of the identified schedule. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaAddReplicationServer

Add or modify replication servers.

Syntax

aaAddReplicationServer *ReplicationServerName, Description, SFPath, SFFreeSpace, AuthenticateWithAAUser, UserName, Password, TCPPort, SummaryReplicationNamingScheme, SimpleReplicationNamingScheme, BufferCount, Bandwidth, MinSFDuration, ReplicationServerKey, CompressionEnabled, ConnectionDetails*

where:

Argument	Description
ReplicationServerName	Name or IP address of the tier 2 server. This value is of data type nvarchar(255), with a default of an empty string.
Description	Description of the replication server. This value is of data type nvarchar(512), with a default of an empty string.
SFPath	Store forward path for the replication server. The default is an empty string. This value is of data type nvarchar(260), with a default of an empty string.
SFFreeSpace	Free space for the store forward path in MB. This value is of data type int, with a default of 125.
AuthenticateWithAAUser	Set to 1 to authenticate with the Network Account (previously called the ArchestrA User). This value is of data type bit, with a default of 1.
UserName	User name for authenticating with the tier 2 server. This value is nvarchar(255), with a default of NULL.
Password	Password for authenticating with the tier 2 server. This value is of data type nvarchar(512), with a default of NULL.
TCPPort	TCP Port for communicating with the tier 2 server. This value is of data type int, with a default of 32565.
SummaryReplicationNamingScheme	Naming rule for summary replication tags. If this is NULL, the naming rule specified in system parameters will be used. This value is of data type nvarchar(512), with a default of *<ReplicationDefaultPrefix>.<SourceTagName>.<TypeAbbreviation><GroupAbbreviation>*.
SimpleReplicationNamingScheme	Naming rule for simple replication tags. If this is NULL,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	the naming rule specified in System parameters will be used. This value is of data type nvarchar(512), with a default of *<ReplicationDefaultPrefix>.<SourceTagName>*.
BufferCount	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
Bandwidth	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited. This value is of data type int.
MinSFDuration	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds. This value is of data type int, with a default of 180.
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with a default of NULL.
CompressionEnabled	0 = Compression off. 1 = Enable compression for the packets sent to the replication server.
ConnectionDetails	Internal use only.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationTagEntity

Add or modify a replication tag entity.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaAddReplicationTagEntity *SourceTagName, ReplicationGroupName, ReplicationServerName, ReplicationTypeKey, MaximumStates, CurrentEditor, DestinationTagId, DestinationTagName*

where:

Argument	Description
SourceTagName	The source tag name. This value is of data type nvarchar(256), with no default.
ReplicationGroupName	The name of the replication group. If this is NULL, the replication type is set to simple replication. This value is nvarchar(255), with a default of NULL.
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
ReplicationTypeKey	The type of replication. Valid values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 2.
MaximumStates	Maximum number of states to track for state summary tags. This value is of data type tinyint, with a default of 10 and a maximum of 100.
CurrentEditor	0 - Historian 2 - AVEVA Application Server This value is of data type tinyint, with a default of 0.
DestinationTagID	Unique ID of the destination tag. If NULL, the destination tag name is generated based on the naming rule. This value is of data type uniqueidentifier, with a default of NULL.
DestinationTagName	Name of the destination tag. If NULL, the destination tag name is generated based on the naming rule. This value is of data type nvarchar(256), with a default of NULL.
ReplicationTagEntityKey	The unique identifier for the replication tag entity. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaAddReplicationTagExtendedProperty

Add or modify a replication tag extended property.

Syntax

aaAddReplicationTagExtendedProperty, *ReplicationServerKey, DestinationTagName, PropertyName, PropertyValue*

where:

Argument	Description
ReplicationServerKey	The unique identifier for the replication server.
DestinationTagName	Name of the destination tag. If NULL, the destination tag name is generated based on the naming rule. This value is of data type nvarchar(256), with a default of NULL.
PropertyName	The extended tag property name.
PropertyValue	The value of this replication tag extended property.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddStateSummaryTag

Add or modify a state summary tag.

Syntax

aaStateSummaryTag *TagName, TagId, Description, SourceTag, SourceServer, CreatedBy, DateCreated, AcquisitionType, StorageNodeKey, IOServerKey, TopicKey, StorageType, DeadbandType, TimeDeadband, CurrentEditor, wwTagKey, ChannelStatus*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
TagId	The unique tag ID of the tag within the AVEVA Historian system. The value is of data type uniqueidentifier, with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
Description	The description of the analog summary tag. This value is of data type nvarchar(512), with a default of an empty string.
SourceTag	The name of the source tag to create the analog summary tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver. This value is of data type tinyint, with a default of 1.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	"not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. This value is of data type smallint, with a default of 1.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStorageLocation

Add or modify a storage location

Syntax

aaAddStorageLocation *ShardId, StorageType, Path, MaxMBSize, MinMBThreshold, MaxAgeThreshold*

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).
StorageType	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent, 5 = Tag data index, 6 = Tag data index write-ahead logs, 7 = Revision tag data write-ahead logs. There can be only one storage location of each type.
Path	The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different internal hard drive.If you use a network location, the InSQLConfiguration virtual service account must have full access to the location. You can grant that access by using the credentials of the computer account in the format <domain_name>\<computer_name>$. The locations must be specified using UNC notation. Mapped drives are not supported. If empty, the default <SystemDataPath>\Wonderware\Data\Circular is used.
MaxMBSize	The limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used.
MinMBThreshold	The minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block (before any compression) by 1.5 to determine the minimum threshold.
MaxAgeThreshold	The age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStorageShard

Returns definition information for specified storage shard (partition).

Syntax

aaAddStorageShard *ShardId, ShardName, Description, ComputerName, BlockDuration, TimeUnitId, TimeZoneId,*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AdjustToDST, MaxSnapshotSize, CmdArgs, CmdExtArgs

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).
ShardName	The name of the partition.
Description	The description of the partition.
ComputerName	The network name of the computer on which the storage partition resides.
BlockDuration	Duration, in hours, for history blocks. Valid values are: 1, 2, 3, 4, 6, 8, 12, 24. The default is 24 hours. The history block size must always be greater than the highest scan rate. For more information, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.
TimeUnitId	Foreign key to TimeUnit. Indicates whether the block duration is in hours/days.
TimeZoneId	The time zone associated with this storage partition.
AdjustToDST	Internal use only.
MaxSnapshotSize	Maximum size, in MB, for data storage snapshots in memory. Bigger snapshots allow for faster retrieval. You might need to increase this size for systems with very high data rates. For example, if retrieval is slow from data in the current history block, try increasing this rate. Also be sure that you have enough RAM, up to 1 GB.
CmdArgs	Command line parameters for customizing replication and storage execution.
CmdExtArgs	Command line parameters for customizing replication and storage execution.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStorageShardAssignmentRule

Add or modify an assignment rule for a storage partition (shard).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaAddStorageShardAssignmentRule *Name, Priority, TagFilter, ShardId, Enabled*

where:

Argument	Description
Name	The name of the rule.
Priority	The priority for the rule.
TagFilter	The OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
ShardId	The unique identifier for the partition (shard).
Enabled	Specifies whether the rule is enabled.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStructureTag

Add or modify a structure tag.

Syntax

aaStructureTag *TagName, TagId, Description, SourceTag, SourceServer, CreatedBy, DateCreated, StructureId, AcquisitionType, StorageNodeKey, IOServerKey, TopicKey, StorageType, EngineeringUnit, IntegralDivisor, MinEU, MaxEU, MinRaw, MaxRaw, DeadbandType, TimeDeadband, CurrentEditor, wwTagKey, ChannelStatus*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
TagId	The unique tag ID of the tag within the AVEVA Historian system. The value is of data type uniqueidentifier, with a default of NULL.
Description	The description of the analog summary tag. This value is of data type nvarchar(512), with a default of an empty string.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
SourceTag	The name of the source tag to create the analog summary tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
StructureId	The ID for the structure. The value is of data type uniqueidentifier, with a default of NULL.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored."

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
EngineeringUnit	The unit of measure. Examples are mph, grams, and pounds. This value is of data type nvarchar(32), with a default of an empty string.
IntegralDivisor	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. This value is of data type float(25), with a default of 1.
MinEU	The minimum value of the tag, measured in engineering units. This value is of data type float, with a default of 0.
MaxEU	The maximum value of the tag, measured in engineering units. This value is of data type float, with a default of 100.
MinRaw	The minimum value of the raw acquired value. This value is of data type float, with a default of 0.
MaxRaw	The maximum value of the raw acquired value. This value is of data type float, with a default of 4095.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. This value is of data type smallint, with a default of 1.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAnalogDetail

Returns information about one or more specified analog tags, including the name of the tag, a description, the acquisition rate, the engineering unit, and the minimum and maximum values in engineering units.

Syntax

aaAnalogDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaAnalogTagDelete

Deletes an analog tag.

Syntax

aaAnalogTagDelete *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	default.

Permission

Execute permission defaults to the **aaPowerUsers** and **aaAdministrators** groups.

aaAnalogTagInsert

Inserts an analog tag.

Syntax

aaAnalogTagInsert *TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, EUKey, MinEU, MaxEU, MinRaw, MaxRaw, Scaling, RawType, ValueDeadband, InitialValue, IntegerSize, SignedInteger, TopicKey, IOServerKey, StorageNodeKey, AIRetrievalMode, SamplesInActiveImage, RateDeadband, InterpolationType, RolloverValue, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHistory*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS, or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. This value is of data type int, with a default of 10000.
ItemName	The address string of the tag. This value is of data type nvarchar(256), with a default of an empty string.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with a default of 1.
MinEU	The minimum value of the tag, measured in engineering units. This value is of data type float, with a default of 0.
MaxEU	The maximum value of the tag, measured in engineering units. This value is of data type float, with a default of 100.
MinRaw	The minimum value of the raw acquired value. This value is of data type float, with a default of 0.
MaxRaw	The maximum value of the raw acquired value. This value is of data type float, with a default of 4095.
Scaling	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None 1 = Linear 2 = Square Root (reserved for future use) This value is of data type int, with a default of 1.
RawType	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes) 2 = MS Float (4 bytes) 3 = Integer (2 or 4 bytes) 4 = MS Double (8 bytes; reserved for future use) This value is of data type int, with a default of 3.
ValueDeadband	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	to delta storage only. A value of 0 indicates that a value deadband will not be applied. This value is of data type float, with a default of 0.
InitialValue	The initial value as imported from an external source (for example, from InTouch). This value is of data type float, with a default of 0.
IntegerSize	The bit size of the analog tag. 12 = 12-bit 15 = 15-bit 16 = 16-bit 32 = 32-bit 64 = 64-bit (reserved for future use) This value is of data type tinyint, with a default of 16.
SignedInteger	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned 1 = Signed This value is of data type bit, with a default of 0.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
AIRetrievalMode	Used to specify the behavior of retrieval for data in active image. You can either retrieve from all acquired data values that are currently in the active image, or only the data values that are configured to be stored on disk. Data on disk may be a subset of that in the active image, depending on the storage rate for the tag.Valid values are: 0 = All of the values received into the active image will be included in the returned data (default); 1 = Only the values that will be moved into storage will be included in the returned data. This value is of data type tinyint, with a default of 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
SamplesInActiveImage	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.
RateDeadband	Used to percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. This value is of data type float, with a default of 0.
InterpolationType	The interpolation type for retrieval. 0 = Stair-stepped interpolation 1 = Linear interpolation (if applicable, based on the tag type) 254 = System default interpolation mode The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. This value is of data type tinyint, with a default of 254.
RolloverValue	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000. This value is of data type int, with a default of 0.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the Storage subsystem may contain multiple data values

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
SourceTag	The name of the source tag to create the tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
AITag	Used to specify whether the tag's values are stored by the Classic Storage subsystem. 0 = Not stored by the Classic Storage subsystem 1 = Stored by the Classic Storage subsystem This value is of data type bit, with a default of 1.
TagId	The unique identifier for the tag. The value is of data type uniqueidentifier, with a default of NULL.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	value is of data type tinyint, with a default of 1.
AIHistory	Used to specify whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage. 1 = The tag may have data previously collected by classic storage. This value is of data type bit, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAnalogTagSelect

Selects an analog tag.

Syntax

aaAnalogTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaAnalogTagUpdate

Updates an analog tag.

Syntax

aaAnalogTagUpdate *wwTagKey, TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, EUKey, MinEU, MaxEU, MinRaw, MaxRaw, Scaling, RawType, ValueDeadband, InitialValue, IntegerSize, SignedInteger, TopicKey, IOServerKey, StorageNodeKey, AIRetrievalMode, SamplesInActiveImage, RateDeadband, InterpolationType, RolloverValue, ServerTimeStamp,*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHistory

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as the aaAnalogTagInsert stored procedure. However, only these arguments have defaults:

- *Description*
- *AcquisitionType*
- *ItemName*
- *CreatedBy*
- *DateCreated*
- *CurrentEditor*
- *SamplesInActiveImage*
- *RateDeadband*
- *InterpolationType*
- *RolloverValue*
- *ServerTimeStamp*
- *DeadbandType*
- *SourceTag*
- *SourceServer*
- *AITag*
- *TagId*
- *ChannelStatus*
- *AIHistory*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAnnotationDelete

Deletes an annotation.

Syntax

aaAnnotationDelete *AnnotationKey*

where:

Argument	Description
AnnotationKey	The unique numerical identifier for the annotation. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaUsers, aaPowerUsers, and aaAdministrators groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaAnnotationInsert

Inserts an annotation.

Syntax

aaAnnotationInsert *TagName, UserKey, DateTime, DateCreated, Content, Value*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
UserKey	The unique numerical identifier for a database user as defined in the UserDetail table. This value is of data type int, with a default of NULL.
DateTime	The timestamp of the tag value for which the user has made an annotation. This value is of data type datetime2(7), with a default of NULL.
DateCreated	The date that the annotation was created. This value is of data type datetime2(7), with a default of NULL.
Content	The annotation text. This value is of data type nvarchar(1000), with a default of "Annotation."
Value	The value of the tag at the time of the annotation. This value is of data type real, with a default of 0.0.

Permission

Execute permission defaults to the aaUsers, aaPowerUsers, and aaAdministrators groups.

aaAnnotationRetrieve

Retrieves one or more annotations.

Syntax

aaAnnotationRetrieve *TagList, StartTime, EndTime*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the data to query. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the data to query. This value is of data type nvarchar(50), with no default.

Permission

Execute permission defaults to the public group.

aaAnnotationSelect

Selects an annotation.

Syntax

aaAnnotationSelect *AnnotationKey*

where:

Argument	Description
AnnotationKey	The unique numerical identifier for the annotation. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaAnnotationUpdate

Updates an annotation.

Syntax

aaAnnotationUpdate *AnnotationKey, TagName, UserKey, DateTime, Content, Value*

where:

Argument	Description
AnnotationKey	The unique numerical identifier for the annotation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	This value is of data type int, with no default.

The remaining arguments are similar to those for the aaAnnotationInsert stored procedure.

Permission

Execute permission defaults to the aaUsers, aaPowerUsers, and aaAdministrators groups.

aaArchestrANSClear

Removes all ArchestrA entries from the public namespace.

Syntax

aaArchestrANSClear

Remarks

In general, using this stored procedure is not recommended. If you need to remove the ArchestrA entries because of a namespace corruption, contact Technical Support for guidance.

Permission

Execute permission defaults to the aaAdministrators group.

aaCheckChartConfigurationNameExists

Checks to see if a configuration name exists for some InSight content.

Syntax

aaCheckChartConfigurationNameExists *ChartConfigurationName*

where:

Argument	Description
ChartConfigurationName	The name of the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaCleanupAfterCommit

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Runs once after reinitialization or system startup is complete.

Syntax

aaCleanupAfterCommit

Remarks

This stored procedure does two things:

1. Sets the DbStatus column of the StorageNode table to 0.
2. Deletes the contents of the ConfigStatusSnapshot table.

Permission

Execute permission defaults to the aaAdministrators group.

aaCleanupSystemNotRunning

Runs once whether or not reinitialization or system startup is complete. For internal use only.

Syntax

aaCleanupSystemNotRunning

Remarks

This stored procedure does two things:

1. Sets the DbStatus column of the StorageNode table to 0.
2. Deletes information from the ConfigStatusSnapshot table.

Permission

Execute permission defaults to the aaAdministrators group.

aaClearDeletedTags

Used by storage for handling deleted tags. Internal use only.

Syntax

aaClearDeletedTags *ChangeVersion*
where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaClearDeletedReplicationTagEntities

Used by storage for handling deleted replication tag entities. Internal use only.

Syntax

aaClearDeleteReplicationTagEntities *TagList*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaCommitAllowed

Used to allow a reinitialization of the system.

Syntax

aaCommitAllowed allowCommit

where:

Argument	Description
allowCommit	Specifies the change. *<Is this an ID or name? Which table does it come from?>*

Permission

Execute permission defaults to the public group.

aaCommitChanges

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Used to trigger a reinitialization of the system.

Syntax

aaCommitChanges

Remarks

This stored procedure performs the following if a change is made:

1. Copies the contents of the ConfigStatusPending table to the ConfigStatusSnapshot table.
2. Resets the Status column in the applicable database table (Tag, Topic, IOServer, IODriver, StorageLocation, SnapshotDetail, or SystemParameter) to 0.
3. Deletes the contents of the ConfigStatusPending table.

Permission

Execute permission defaults to the aaAdministrators group.

aaCommitChangesAtStartup

Used to specify a reinitialization of the system at startup.

Syntax

aaCommitChangesAtStartup

Remarks

This stored procedure is used only when a modification to a storage location has been made. The functionality of this stored procedure is similar to that of **aaCommitChanges**.

Permission

Execute permission defaults to the aaAdministrators group.

aaContextDelete

Deletes a context.

Syntax

aaContextDelete *ContextKey*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ContextKey	The unique numerical identifier for the context. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers, and aaAdministrators groups.

aaContextInsert

Inserts a context.

Syntax

aaContextInsert *Description*

where:

Argument	Description
Description	The description of the context. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers, and aaAdministrators groups.

aaContextSelect

Selects a context.

Syntax

aaContextSelect *ContextKey*

where:

Argument	Description
ContextKey	The unique numerical identifier for the context. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaContextUpdate

Updates a context.

Syntax

aaContextUpdate *ContextKey, Description*

where:

Argument	Description
ContextKey	The unique numerical identifier for the context. This value is of data type int, with no default.
Description	The description of the context. This value is of data type nvarchar(50), with no default.

Permission

Execute permission defaults to the aaPowerUsers, and aaAdministrators groups.

CreateReplicationServerDefaultGroups

Used to create default replication server groups.

Syntax

CreateReplicationServerDefaultGroups *ReplicationServerKey*

where:

Argument	Description
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and aaAdministrators groups.

CreateReplicationServerSystemTags

Creates replication server default groups if the CreateGroups setting is set to true. Internal use only.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

CreateReplicationServerSystemTags *ReplicationServerKey*

where:

Argument	Description
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and aaAdministrators groups.

aaDBChangesPending

Returns a list of modifications pending, from the ConfigStatusPending table, in a readable format.

Syntax

aaDBChangesPending

Permission

Execute permission defaults to the public group.

aaDBConfig

Returns a summary of the current database configuration, such as number of tags, number of tags per type, storage configuration, event tags, summary configuration, and so on.

Syntax

aaDBConfig

Permission

Execute permission defaults to the public group.

aaDeleteChartConfiguration

Deletes a chart configuration.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaDeleteChartConfiguration *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaDeleteComment

Deletes a comment (annotation).

Syntax

aaDeleteComment *CommentsKey*

where:

Argument	Description
CommentsKey	The unique identifier for the comment.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaDeleteOlderEvents

Deletes old events from event storage.

Syntax

aaDeleteOlderEvents

Remarks

This stored procedure is executed by the aaSpaceManager stored procedure every ten minutes. The duration for which events are kept is stored in the SystemParameter table. Events will be deleted from the EventHistory table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteOlderSummaries

Deletes old summaries from summary storage.

Syntax

aaDeleteOlderSummaries

Remarks

This stored procedure is executed by the aaSpaceManager stored procedure every ten minutes. The duration for which summaries are kept is stored in the SystemParameter table. Summaries will be deleted from the SummaryHistory table.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteReplicationGroup

Deletes an existing replication group. If the group being deleted is referenced by a replication tag entity, the procedure returns an error message.

Syntax

```
aaDeleteReplicationGroup ReplicationGroupName, ReplicationServerName, ReplicationTypeKey
```

where:

Argument	Description
ReplicationGroupName	The name of the group. This parameter is required. This value is of data type nvarchar(255), with no default.
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
ReplicationTypeKey	The type of replication. Valid values are: 1 = Simple Replication 2 = Analog Summary Replication 3 = State Summary Replication

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	This value is of data type tinyint, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteReplicationSchedule

Deletes a replication schedule.

Syntax

aaDeleteReplicationSchedule *ReplicationScheduleName*

where:

Argument	Description
ReplicationScheduleName	The name of the schedule. This parameter is required. This value is of data type nvarchar(255), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteReplicationServer

Deletes an existing replication server. If the server being deleted is referenced by a replication group, the procedure returns an error message.

Syntax

aaDeleteReplicationServer *ReplicationServerName*

where:

Argument	Description
ReplicationServerName	The name of the replication server. This parameter is required. This value is of data type nvarchar(255), with

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	no default.

Permission

Execute permission defaults to the aaAdministrators group.

DeleteReplicationServerSystemTags

Deletes replication server default groups if the DeleteGroups setting is set to true. Internal use only.

Syntax

DeleteReplicationServerSystemTags *ReplicationServerKey*

where:

Argument	Description
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and aaAdministrators groups.

aaDeleteReplicationTagEntity

Deletes an existing replication entity from a tier 1 server.

Syntax

aaDeleteReplicationTagEntity *ReplicationServerName, DestinationTagName*

where:

Argument	Description
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
DestinationTagName	The name of the destination tag. This value is of data type nvarchar(255), with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteTag

Deletes a tag from the database.

Syntax

aaDeleteTag *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaDeleteTagExtendedProperty

Deletes a tag extended property.

Syntax

aaDeleteTagExtendedProperty *TagName, PropertyName*

where:

Argument	Description
TagName	The name of the tag.
PropertyName	The extended tag property name.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers group.

aaDetectorStringSelect

Selects the detector string for a specified event tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaDetectorStringSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDiscreteDetail

Returns information about one or more specified discrete tags, including the name of the tag, a description, the message for the TRUE (1) state of the tag, and the message for the FALSE (0) state of the tag.

Syntax

aaDescreteDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaDiscreteTagDelete

Deletes a discrete tag.

Syntax

aaDiscreteTagDelete *wwTagKey*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaDiscreteTagInsert

Inserts a discrete tag.

Syntax

aaDiscreteTagInsert *TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, MessageKey, InitialValue, TopicKey, IOServerKey, AIRetrievalMode, SamplesInActiveImage, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. This value is of data type int, with a default of 0.
ItemName	The address string of the tag. This value is of data type nvarchar(256), with a default of an empty string.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with a default of 1.
InitialValue	The initial value as imported from an external source (for example, from InTouch). This value is of data type tinyint, with a default of 0.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
AIRetrievalMode	Used to specify the behavior of retrieval for data in active image. You can either retrieve from all acquired data values that are currently in the active image, or only the data values that are configured to be stored on disk. Data on disk may be a subset of that in the active image, depending on the storage rate for the tag.Valid values are: 0 = All of the values received into the active image will be included in the returned data (default). 1 = Only the values that will be moved into storage will be included in the returned data. This value is of data type tinyint, with a default of 0.The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.
SamplesInActiveImage	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.
ServerTimeStamp	Used to specify whether local timestamping by the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
SourceTag	The name of the source tag to create the tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
AITag	Used to specify whether the tag's values are stored by the classic storage subsystem. 0 = Not stored by the Classic Storage subsystem; 1 = Stored by the Classic Storage subsystem. This value is of data type bit, with a default of 1.
TagId	The unique identifier for the tag. The value is of data type uniqueidentifier, with a default of NULL.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.
AIHistory	Used to specify whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage. 1 = The tag may have data previously collected by classic storage. This value is of data type bit, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaDiscreteTagSelect

Selects a discrete tag.

Syntax

aaDiscreteTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaDiscreteTagUpdate

Updates a discrete tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaDiscreteTagUpdate *wwTagKey, TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, MessageKey, InitialValue, TopicKey, IOServerKey, AIRetrievalMode, SamplesInActiveImage, ServerTimeStamp, DeadbandType, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as for the aaDiscreteTagInsert stored procedure. However, only these arguments have defaults:

- *Description*
- *AcquisitionType*
- *ItemName*
- *CreatedBy*
- *DateCreated*
- *SamplesInActiveImage*
- *ServerTimeStamp*
- *DeadbandType*
- *SourceTag*
- *SourceServer*
- *AITag*
- *TagId*
- *ChannelStatus*
- *AIHIstory*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEngineeringUnitDelete

Deletes an engineering unit.

Syntax

aaEngineeringUnitDelete *EUKey*

where:

Argument	Description
EUKey	The unique numerical identifier of an engineering

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	unit. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEngineeringUnitInsert

Inserts an engineering unit.

Syntax

aaEngineeringUnitInsert *Unit, DefaultTagRate, IntegralDivisor*

where:

Argument	Description
Unit	The unit of measure. Examples are mph, grams, and pounds. This value is of data type nvarchar(32), with no default.
DefaultTagRate	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag. This value is of data type int, with a default of 10000.
IntegralDivisor	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. This value is of data type float(25), with a default of 1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEngineeringUnitSelect

Selects an engineering unit.

Syntax

aaEngineeringUnitSelect *EUKey*

where:

Argument	Description
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaEngineeringUnitUpdate

Updates an engineering unit.

Syntax

aaEngineeringUnitUpdate *EUKey, Unit, DefaultTagRate, IntegralDivisor*

where:

Argument	Description
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with no default.

The remaining arguments are the same as for the aaEngineeringUnitInsert stored procedure. However, only the *IntegralDivisor* argument has a default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaEventDetection

Detects the number of events in history in which the data value for the specified tag matched the criteria defined by the remaining arguments. This stored procedure is used by the event subsystem and should not be modified.

Syntax

aaEventDetection *TagName, Operator, DetectValue, Edge, Resolution, StartTime, EndTime*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Operator	The comparison operator. Valid values are: • > • >= • < • <= • = • <> This value is of data type char(2), with no default.
DetectValue	The value against which the stored values for the tag are compared to determine if the event occurred. This value is of data type float(25), with a default of none.
Edge	The type of edge detection result set that the query will return. Valid values are: • NONE • LEADING • TRAILING • BOTH This value is of data type char(8), with no default.
Resolution	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	values that occur at each 5000 ms interval, starting with the start date and ending with the end date. This value is of data type int, with no default.
StartTime	The starting timestamp for the data to query. This value is of dataype varchar(30), with no default.
EndTime	The ending timestamp for the data to query. This value is of dataype varchar(30), with no default.

Remarks

You can apply a resolution only if you set the value of the Edge argument to NONE.

Permission

Execute permission defaults to the aaAdministrators group.

aaEventHistoryInsert

Inserts a row into the EventHistory table for each occurrence of an event for a specified event tag. This stored procedure is used by the event subsystem and should not be modified.

Syntax

aaEventHistoryInsert *TagName, DateTime, DetectDateTime, Edge*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
DateTime	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian. This value is of data type datetime2(7), with no default.
DetectDateTime	The timestamp reflecting when the event was detected by the event system. This value is of data type datetime2(7), with no default.
Edge	The "edge" for the event detection. 0 = Trailing; 1 =

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaEventHistorySelect

Returns information stored in the EventHistory table for each specified event tag.

Syntax

aaEventHistorySelect *TagList, StartTime, EndTime*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the data to query. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the data to query. This value is of data type nvarchar(50), with no default.

Remarks

This stored procedure will return information for all events that occurred between the starting time and the ending time.

Permission

Execute permission defaults to the public group.

aaEventSnapshotInsert

Inserts snapshot values into the AnalogSnapshot, DiscreteSnapshot, and StringSnapshot tables. This stored procedure is used by the event subsystem and should not be modified.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

```
aaEventSnapshotInsert EventLogKey, EventTime, EventTagName
```

Arguments

EventLogKey

The unique numerical identifier of an event occurrence. This value is of data type int, with no default.

EventTime

The timestamp reflecting when the event history data was acquired. This is the time for when the event actually occurred. This value is of data type datetime2(7), with no default.

EventTagName

The name of the event tag to which the snapshot tag is related. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaEventSnapshotSelect

Returns the snapshot tag values for each of the event tags specified by the tag list.

Syntax

aaEventSnapshot *TagList, StartTime, EndTime, OrderBy*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the data to query. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the data to query. This value is of data type nvarchar(50), with no default.
OrderBy	The name of the column in the v_EventSnapshot view used to order the rows in the result set. The value is of data type nvarchar(2000), with a default of 'Event'.

Remarks

This stored procedure will return information for all events that occurred between the starting time and the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ending time.

This stored procedure does not work with string snapshots.

Permission

Execute permission defaults to the public group.

aaEventTagDelete

Deletes an event tag.

Syntax

aaEventTagDelete *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEventTagDetail

Returns the details for a specified event tag, including time detector information, if applicable.

Syntax

aaEventTagDetail *TagList*

where:

Argument	Description
TagList	The list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with a default of '%'.

Permission

Execute permission defaults to the public group.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaEventTagInsert

Inserts an event tag.

Syntax

aaEventTagInsert *TagName, Description, CreatedBy, DateCreated, CurrentEditor, ScanRate, TimeDeadband, Logged, Status, PostDetectorDelay, UseThreadPool, DetectorTypeKey, DetectorString, ActionTypeKey, ActionString, Priority, Edge*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	2 = AVEVA Application Server This value is of data type int, with a default of 0.
ScanRate	The interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms). This value is of data type int, with a default of 0.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
Logged	Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured. 0 = Not logged 1 = Logged This value is of data type bit, with a default of 0.
Status	The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system. This value is of data type tinyint, with a default of 0.
PostDetectorDelay	The amount of time, in milliseconds, that must elapse after an event is detected before the event action can

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	be executed. This value is of data type int, with a default of 0.
UseThreadPool	To specify how system threads are used to process events. 1 = All events are handled by a single thread and a single logon to the SQL Server; 0 = Each event uses a separate system thread and logon. This will allow the Event subsystem to manage the scan rates of each detector component concurrently. (Reserved for future use.) This value is of data type bit, with a default of 1.
DetectorTypeKey	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System 2 = External event 3 = Generic SQL 4 = Analog specific value 5 = Discrete specific value 6 = Time-based (schedule) This value is of data type int, with a default of 0.
DetectorString	The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian. This value is of data type nvarchar(1500), with a default of NULL.
ActionTypeKey	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action 2 = Generic SQL 3 = Snapshot 4 = E-mail 5 = Deadband 6 = Summary This value is of data type int, with a default of 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ActionString	The script that specifies the event action. Action scripts run on the local AVEVA Historian. This value is of data type nvarchar(1500), with a default of NULL.
Priority	The priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. This value is of data type tinyint, with a default of 0.
Edge	The "edge" for the event detection. 0 = Trailing 1 = Leading 2 = Both 3 = None 4 = Time Detector 5 = External Detector This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEventTagSelect

Selects an event tag.

Syntax

aaEventTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	default of NULL.

Permission

Execute permission defaults to the public group.

aaEventTagSelectAll

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectAll

Permission

Execute permission defaults to the public group.

aaEventTagSelectDeleted

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectDeleted

Permission

Execute permission defaults to the public group.

aaEventTagSelectDisabled

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectDisabled

Permission

Execute permission defaults to the public group.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaEventTagSelectInserted

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectInserted

Permission

Execute permission defaults to the public group.

aaEventTagSelectUpdated

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectUpdated

Permission

Execute permission defaults to the public group.

aaEventTagUpdate

Updates an event tag.

Syntax

aaEventTagUpdate *wwTagKey, TagName, Description, CreatedBy, DateCreated, CurrentEditor, ScanRate, TimeDeadband, Logged, Status, PostDetectorDelay, UseThreadPool, DetectorTypeKey, DetectorString, ActionTypeKey, ActionString, Priority, Edge*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as for the aaEventTagInsert stored procedure. However, these arguments do not have defaults:

- wwTagKey
- DateCreated

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- TagName
- Description
- DetectorString
- ActionString

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaGetAnalogSummaryTags

Returns all the properties for the specified analog summary tag or if you don't specify a tag, returns this info for them all.

Syntax

aaGetAnalogSummaryTags *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the public group.

aaGetChangeNotification

Used internally to manage configuration changes for a tag. Internal use only.

Syntax

aaGetChangeNotification *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaGetChartConfigurationLayout

Returns chart layout information for specific InSight content.

Syntax

aaGetChartConfigurationLayout *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetChartConfigurations

Returns chart configuration information for specific InSight content.

Syntax

aaGetChartConfigurations *ChartConfigurationName, ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationName	The name of the InSight content.
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetChartConfigurationsForDashboard

Returns definition information for each specified tag.

Syntax

aaGetChartConfigurationsForDashboard *ChartConfigurationUrl*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetChartConfigurationsForKeywords

Returns keyword associated with specific InSight content.

Syntax

aaTagInfo *TagList*

where:

Argument	Description
Keyword	A list of keywords associated with the content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetDbRevision

Used to determine the current revision number of the database.

Syntax

aaGetDbRevision

Permission

Execute permission defaults to the public group.

aaGetDeletedReplicationTagEntities

Returns a list of deleted replication tag entities.

Syntax

aaGetDeletedReplicationTagEntities *ChangeVersion*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

where:

Argument	Description
ChangeVersion	Specifies the change version. Internal use only.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaGetDeletedTags

Used by storage to handle deleted tags. Internal use only.

Syntax

aaGetDeletedTags *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaGetHistorianPartners

Returns the name of the partner historian. Internal use only.

Syntax

aaGetHistorianPartners

Permission

Execute permission defaults to the aaAdministrators group.

aaGetLastTagKey

Returns the details for the last inserted tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaGetLastTagKey *TagType*

where:

Argument	Description
TagType	The type of tag. 1 = Analog 2 = Discrete 3 = String 4 = Complex 5 = Event 7 = Summary tag (analog or state) This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaGetReplicationGroups

Returns the groups configured in the Historian database for a given replication server and type. If you specify all the parameters, then the specific group identified by the parameters is returned.

Syntax

aaGetReplicationGroups *ConfigurationToReturn, ReplicationServerName, ReplicationTypeKey, ReplicationGroupName, IncludeEmptyGroups, ReplicationGroupKey, RowsToReturn*

where:

Argument	Description
ConfigurationToReturn	The return configuration for the replication service.This value is of data type tinyint, with a default of 1.
ReplicationServerName	The name of the replication server. This value is nvarchar(255), with a default of NULL.
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	This value is of data type tinyint, with a default of 2.
ReplicationGroupName	The name of the replication group. This value is nvarchar(255), with a default of NULL.
IncludeEmptyGroups	Bit that specifies whether to include empty groups in the return. This value is of data type bit, with a default of 0.
ReplicationGroupKey	Unique identifier for the replication group. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationNamingParameters

Returns the naming parameters for the specified replication type in the specified replication group.

Syntax

aaGetReplicationNamingParameters *ReplicationTypeKey, ReplicationGroupKey*

where:

Argument	Description
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 3.
ReplicationGroupKey	Unique identifier for the replication group. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaGetReplicationRule

Returns details about a replication rule

Syntax

aaGetReplicationRule *Name, AutoReplication*

where:

Argument	Description
Name	The name of the replication rule.
AutoReplication	Specifies whether this rule is used for autosummary replication. 0 - not used for autosummary replication; 1 - used for autosummary replication.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetReplicationSchedules

Returns the schedules configured in the Historian database.

Syntax

aaGetReplicationSchedules *ConfigurationToReturn, ReplicationScheduleName, ReplicationScheduleKey, RowsToReturn*

where:

Argument	Description
ConfigurationToReturn	The return configuration for the replication schedule. This value is of data type tinyint, with a default of 0.
ReplicationScheduleName	The name of the schedule. This value is nvarchar(255), with a default of NULL.
ReplicationScheduleKey	The unique identifier for the schedule. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationServers

Returns the configured replication servers in the database. If the server name is specified, then it will return only the properties of the server identified by the name.

Syntax

aaGetReplicationServers *ConfigurationToReturn, ReplicationServerName, ReplicationServerKey, RowsToReturn*

where:

Argument	Description
ConfigurationToReturn	Returns the configuration for the replication service. This value is of data type tinyint, with a default of 0.
ReplicationServerName	The name of the server. This value is nvarchar(255), with a default of NULL.
ReplicationServerKey	The unique identifier for the server. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationShard

Returns details about a replication shard.

Syntax

aaGetReplicationShard *ShardId*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ShardId	The unique identifier for the partition (shard).

Permission

Execute permission defaults to the aaAdministrators group.

aaGetReplicationTagEntities

Returns the replication entities configured in the Historian database. This procedure will return the properties of the replication tag entity based on the following parameter order:

- If a ReplicationTagEntityKey is specified, then the specific entity properties are returned.
- If a SourceTagName is specified, then all the entities with the specific SourceTagName are returned.
- If the Replication Server and GroupName are not specified, then all the entities belonging to the specific replication type are returned.
- If the Replication Server and GroupName and type are specified, then all the entities belonging to the specific group and type are returned.

Syntax

aaGetReplicationTagEntities *ConfigurationToReturn, ReplicationServerName, ReplicationGroupName, ReplicationTypeKey, SourceTagName, ReplicationTagEntityKey, RowsToReturn, FetchModified*

where:

Argument	Description
ConfigurationToReturn	The return configuration for the replication entities. This value is of data type tinyint, with a default of 1.
ReplicationServerName	The name of the server. This value is nvarchar(255), with a default of NULL.
ReplicationGroupName	The name of the replication group. This value is nvarchar(255), with a default of NULL.
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 2.
SourceTagName	The name of the source tag. This value is of data type nvarchar(256), with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ReplicationTagEntityKey	The unique identifier for the replication tag entity. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of 3.
FetchModified	Returns requested entities. This value is of data type bit. Valid values are: 1 = Fetch only modified entities 0 = Fetch all

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationTags

Returns the specified replication tag.

Syntax

aaGetReplicationTags *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaGetRowCount

Internal use only.

Syntax

aaGetRowCount *TableName*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
TableName	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaGetStateSummaryTags

Returns the specified state summary tag.

Syntax

aaGetStateSummaryTags *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the public group.

aaGetStorageShard

Returns a storage shard configuration.

Syntax

aaGetStorageShard *ShardId, ConfigurationToReturn*

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).
ConfigurationToReturn	Specifies which storage shard configuration to return.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaGetStorageShardAssignmentRule

Returns a storage shard assignment rule.

Syntax

aaGetStorageShardAssignmentRule *Name*

where:

Argument	Description
Name	The name of the assignment rule.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetTagExtendedProperties

Retrieves all tag extended properties.

Syntax

aaGetTagExtendedProperties *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetUserKey

Internal use only.

Syntax

aaGetUserKey

aaHistorianConfigNSExpand

Expands the tree view under an AVEVA Historian in the namespace. This stored procedure is used by the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Configuration Editor component of the Operations Control Management Console and should not be modified.

Syntax

aaHistorianConfigNSExpand *PKey*

where:

Argument	Description
PKey	A local variable used to identify the AVEVA Historian in the namespace. This value is of data type int, with no default.

Remarks

An object can have one or more objects below it in the namespace hierarchy.

Permission

Execute permission defaults to the public group.

aaHistorianNSExpand

Expands the tree view under an AVEVA Historian in the namespace.

Syntax

aaHistorianNSExpand *PKey*

where:

Argument	Description
PKey	A local variable used to identify the AVEVA Historian in the namespace. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaHistorianStatusSelect

Used to select the value of the database status flag, DbStatus.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaHistorianStatusSelect

Remarks

This stored procedure is used by the Operations Control Management Console to determine the state of a database modification.

Permission

Execute permission defaults to the public group.

aaHistorianStatusSet

Sets the value of the status flag, DbStatus, to a value greater than 0 when a database modification needs to be processed by the server (back end). Sets the value of DbStatus back to 0 when a database modification is complete.

Syntax

aaHistorianStatusSet *DbStatus, Acquisition, Storage, DBServer*

where:

Argument	Description
DbStatus	For releases prior to 8.0, used to store the status of server reinitializations. 2 = Certain columns in the Tag, AnalogTag, DiscreteTag, StringTag, Topic, and IOServer tables were changed 3 = Reinitialization needed. 4 = Commit phase of a database update is in progress. 0 = Reinitialization complete. A negative value indicates that an error was encountered during reinitialization. This value is of data type int, with no default.
Acquisition	Used with DbStatus to indicate to the back end whether the acquisition subsystem needs to be restarted. 0 = Restart not needed. 1 = Restart needed. Currently not used. This value is of data type int, with a default of 0.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
Storage	Used with DbStatus to indicate to the back end whether the Storage subsystem needs to be restarted. 0 = Restart not needed. 1 = Restart needed. Currently not used. This value is of data type int, with a default of 0.
DBServer	Used with DbStatus to indicate to the back end whether the database server needs to be restarted. 0 = Restart not needed. 1 = Restart needed. Currently not used. This value is of data type int, with a default of 0.

Note: Only the first argument is required; the others are used to indicate that a specific subsystem needs to be initialized.

Remarks

When a change is made to the Runtime database configuration using the Operations Control Management Console, the value of the DbStatus attribute in the StorageNode table is set to a value greater than 0, meaning that modifications are outstanding and a reinitialization has yet to occur. The Operations Control Management Console, after detecting that a change is outstanding based on the value of DbStatus, makes the required changes, reinitializes the AVEVA Historian, if necessary, and then sets the value of DbStatus back to 0, meaning that reinitialization has been completed.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaHistoryBlockSelect

Returns the list of history blocks for the selected time period. If no arguments are passed, the complete list is returned.

Syntax

aaHistoryBlockSelect *FromDate, ToDate*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
FromDate	The starting timestamp for the history block. This value is of data type datetime2(7), with a default of NULL.
ToDate	The ending timestamp for the history block. This value is of data type datetime2(7), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaInTouchNodeTagList

Used by the Operations Control Management Console to display a list of imported tags for an InTouch node.

Syntax

aaInTouchNodeTagList *NodeKey, FilterStr*

where:

Argument	Description
NodeKey	The unique numerical identifier of the named InTouch node. This value is of data type int, with a default of NULL.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Remarks

This stored procedure returns the AVEVA Historian tagname, the original InTouch tagname, and the InTouch tag type (for example, memory integer).

Permission

Execute permission defaults to the public group.

aaIODriverDelete

Deletes an IDAS.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaIODriverDelete *IODriverKey*

where:

Argument	Description
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaIODriverInsert

Inserts an IDAS.

Syntax

aaIODriverInsert *StorageNodeKey, ComputerName, StoreForwardMode, StoreForwardPath, MinMBThreshold, AltComputerName, Enabled, StoreForwardDuration, AutonomousStartupTimeout, BufferCount, FileChunkSize, ForwardingDelay, ConnectionTimeout*

where:

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
ComputerName	The name of the computer on which the IDAS runs. This value is of data type nvarchar(255), with a default of the name of the local server running Microsoft SQL Server.
StoreForwardMode	Used to specify whether or not store-and-forward capability is enabled. If enabled, and the network connection between the IDAS and the storage node fails, data will be "buffered" to the location specified by the store-and-forward path. Valid values are: = Disabled 1 = Enabled 2 = Autonomous The Autonomous mode (2) is an extension of the normal store-and-forward mode (1). It allows the IDAS to start up using an IDAS configuration file and collect

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	data in store-and-forward mode if the network connection to the AVEVA Historian is not available. This value is of data type tinyint, with a default of 0.
StoreForwardPath	Used to specify the path for the IDAS data buffer on the local hard drive of the IDAS computer. The path should be absolute (for example, c:\IDASBuffer). Data is written to this path until the minimum threshold for the buffer is reached. Remote buffer paths are not supported. When the store-and-forward path specified for the IDAS is invalid, the default path picked by the system is: <public folder>\ArchestrA\Historian\IDAS\SF where the <public folder> is dependent on the operating system. For example, for the Windows 2008 operating system, the path is C:\ProgramData\ArchestrA\Historian\IDAS\SF. When the store-and-forward path specified for the IDAS is just a folder name (without any path characters like \ and :), the default path picked by the system is: <Windows system path>\<folder name specified by the user>. For example, for the Windows Server 2008 32-bit operating system, the path is C:\WINDOWS\system32\<folder name>. This value is of data type nvarchar(255), with a default of an empty string.
MinMBThreshold	The minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer. This value is of data type int, with a default of 16.
AltComputerName	The name of the computer on which an optional, redundant IDAS runs. You must use the fully qualified name of the computer. You could also use the IP address. This should be set to an empty string if no redundant IDAS is specified. Make sure that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online. This value is of data type nvarchar(255), with a default of an empty string.
Enabled	Used to specify whether the IDAS is enabled or not. 0 = Not enabled 1 = Enabled Disabling the IDAS allows for the configuration to be

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	retained in the database, even though the IDAS is removed from the system. This value is of data type bit, with a default of 1.
StoreForwardDuration	The minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds. This value is of data type int, with a default of 180.
AutonomousStartupTimeout	The amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network. This value is of data type int, with a default of 60.
BufferCount	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
FileChunkSize	The size, in bytes, of the data "chunks" that are sent to the historian when store-and-forward data is forwarded. The size of the chunks can be decreased to accommodate slower networks. Decrease this number only if the forwarding delay is greater than zero. This value is of data type int, with a default of 65536.
ForwardingDelay	The interval, in milliseconds, at which "chunks" of store-and-forward data are forwarded to the historian. The length of the interval may need to be increased to accommodate slower networks. This value is of data type int, with a default of 0.
ConnectionTimeout	The amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks. This value is of data type int, with a default of 30.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIODriverSelect

Selects an IDAS.

Syntax

aaIODriverSelect *IODriverKey*

where:

Argument	Description
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaIODriverUpdate

Updates an IDAS.

Syntax

aaIODriverUpdate *IODriverKey, StorageNodeKey, ComputerName, StoreForwardMode, StoreForwardPath, MinMBThreshold, AltComputerName, Enabled, StoreForwardDuration, AutonomousStartupTimeout, BufferCount, FileChunkSize, ForwardingDelay, ConnectionTimeout*

where:

Argument	Description
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with no default.

The remaining arguments are the same as for the aaIODriverInsert stored procedure. However, only these have defaults:

- StorageNodeKey
- MinMBThreshold
- Enabled
- BufferCount
- FileChunkSize
- ForwardingDelay

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- StoreForwardDuration
- AutonomousStartupTimeout
- ConnectionTimeout

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerDelete

Deletes an I/O Server from the system configuration.

Syntax

aaIOServerDelete *IOServerKey*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerInsert

Inserts an I/O Server into the system configuration.

Syntax

aaIOServerInsert *StorageNodeKey, ApplicationName, Description, Path, ComputerName, AutoStart, ExeType, InitializationStatus, ProtocolType, AltComputerName, IODriverKey*

where:

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
Description	The description of the I/O Server. This value is of data type nvarchar(50), with a default of NULL.
Path	The full UNC path (including the filename) to locate the executable file for the I/O Server. If the I/O Server type key is specified, the filename may be omitted. This value is nvarchar(255), with a default of NULL.
ComputerName	The name of the computer on which the I/O Server runs.This value is of data type nvarchar(255), with no default.
AutoStart	Used to control how the I/O Server starts up. 0 = Automatic startup when the system starts. 1 = Manual startup required. Currently not used. This value is of data type bit, with a default of 0.
ExeType	The type of executable for the I/O Server. Used by the Operations Control Management Console to determine how to start the I/O Server. 0 = Service 1 = Console application 2 = Windows application This value is of data type int, with a default of 0.
InitializationStatus	The control flag used to ensure that each I/O Server has been asked for the data type (integer or real) of each tag that it will send. Only needed after a database modification. This value is of data type tinyint, with no default.
ProtocolType	The protocol used by the AVEVA Historian server to communicate with the I/O Server. 1 = DDE (supported only on Windows XP operating system) 2 = SuiteLink 3 = AVEVA Historian named pipe driver (for compatibility with IndustrialSQL Server 3.0 and previous versions) This value is of data type int, with a default of 1.
AltComputerName	The name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made. This

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	value is nvarchar(255), with a default of NULL.
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with a default of 2.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerSelect

Selects an I/O Server from the system configuration.

Syntax

aaIOServerSelect *IOServerKey*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaIOServerTypeDelete

Deletes an I/O Server type from the system configuration.

Syntax

aaIOServerTypeDelete *ApplicationName*

where:

Argument	Description
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	value is of data type nvarchar(32), with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerTypeInsert

Inserts an I/O Server type into the system configuration.

Syntax

aaIOServerTypeInsert *ApplicationName, Description, ExeName, Revision, Platform*

where:

Argument	Description
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with no default.
Description	The description of the I/O Server type. This value is of data type nvarchar(50), with a default of NULL.
ExeName	The name of the I/O Server's executable file. This value is nvarchar(255), with a default of NULL.
Revision	The revision number for the I/O Server. This value is of data type nchar(20), with a default of NULL.
Platform	The operating system required by the I/O Server. Valid operating systems are: • WINDOWS NT • WINDOWS 95 • WINDOWS 98 • WINDOWS XP • WINDOWS 2000 • WINDOWS 2003 • WINDOWS XP • WINDOWS VISTA This value is of data type nchar(20), with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerTypeSelect

Selects an I/O Server type from the system configuration.

Syntax

aaIOServerTypeSelect *ApplicationName*

where:

Argument	Description
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaIOServerTypeUpdate

Updates an I/O Server type in the system configuration.

Syntax

aaIOServerTypeUpdate *ApplicationName, Description, ExeName, Revision, Platform*

Arguments

All arguments are the same as for the aaIOServerTypeInsert stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerUpdate

Updates an I/O Server in the system configuration.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaIOServerUpdate *IOServerKey, StorageNodeKey, IODriverKey, ApplicationName, Description, Path, ComputerName, AutoStart, ExeType, InitializationStatus, ProtocolType, AltComputerName*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.

The remaining arguments are the same as for the aaIOServerInsert stored procedure. However, only the *AltComputerName* argument has a default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitDelete

Deletes a limit.

Syntax

aaLimitDelete *TagName, ContextKey, LimitNameKey*

Arguments

All arguments are the same as for the aaLimitInsert stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitInsert

Inserts a limit.

Syntax

aaLimitInsert *TagName, ContextKey, LimitType, Value, LimitNameKey, Priority, Checked, Description*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.
ContextKey	The unique numerical identifier for the context. This value is of data type int, with a default of 1.
LimitType	The type of limit; that is, whether it is a rising (up) or falling (down) limit. 0 = Rising; 1 = Falling. This value is of data type int, with a default of 1.
Value	The value that is used as a specific limit for a tag. In theory, a tag can have an infinite number of limits defined. This value is of data type real, with no default.
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with no default.
Priority	The priority for the limit. Priorities can range from 1 to over 2 billion, with 1 being the highest priority. This value is of data type int, with a default of 1.
Checked	Used to specify whether a tag imported from InTouch is configured for automatic limit checking. Only checked limits are imported. 0 = Checking disabled; 1 = Checking enabled. This value is of data type bit, with a default of 1.
Description	The description of the limit. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitNameDelete

Deletes a limit name.

Syntax

aaLimitNameDelete *LimitNameKey*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitNameInsert

Inserts a limit name.

Syntax

aaLimitNameInsert *Name*

where:

Argument	Description
Name	The name for the limit. This value is of data type nvarchar(20), with a default of an empty string.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitNameSelect

Selects a limit name.

Syntax

aaLimitNameSelect *LimitNameKey*

where:

Argument	Description
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

aaLimitNameUpdate

Updates a limit name.

Syntax

aaLimitNameUpdate *LimitNameKey, Name*

where:

Argument	Description
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with no default.
Name	The name for the limit. This value is of data type nvarchar(20), with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitSelect

Selects a limit.

Syntax

aaLimitSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. The limit will be selected for the specified tag. This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaLimitUpdate

Updates a limit.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaLimitUpdate *TagName, ContextKey, LimitType, Value, LimitNameKey, Priority, Checked, Description*

Arguments

All arguments are the same as for the aaLimitInsert stored procedure. However, only the *Description* argument has a default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaMessageDelete

Deletes a message for a discrete tag.

Syntax

aaMessageDelete *MessageKey*

where:

Argument	Description
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaMessageInsert

Inserts a message for a discrete tag.

Syntax

aaMessageInsert *Message0, Message1*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
Message0	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. This value is of data type nvarchar(64), with a default of NULL.
Message1	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. This value is of data type nvarchar(64), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaMessageSelect

Selects a message for a discrete tag.

Syntax

aaMessageSelect *MessageKey*

where:

Argument	Description
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaMessageUpdate

Updates a message for a discrete tag.

Syntax

aaMessageUpdate *MessageKey, Message0, Message1*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

where:

Argument	Description
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with no default.

The remaining arguments are the same as for the aaMessageInsert stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaModLogStatus

Used to retrieve the status of modification tracking.

Syntax

aaModLogStatus

Remarks

This stored procedure is used by the Operations Control Management Console. Modification tracking is controlled by the value of the ModLogTrackingStatus system parameter, which is stored in the Value column of the SystemParameter table. If the value of this column is set to a value from 1 to 7, then modification tracking is on (0 = off).

Permission

Execute permission defaults to the public group.

aaNotifyChange

Used internally to track configuration changes for tags. Internal use only.

Syntax

aaNotifyChange *ChangeType*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ChangeType	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaPrivateNSAddGroup

Adds a group object in the private namespace under the specified parent object in the namespace hierarchy.

Syntax

aaPrivateNSAddGroup *Name, ParentKey, Type*

where:

Argument	Description
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.
ParentKey	The unique identifier for a named object in this namespace. This value is of data type int, with no default.
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups This value is of data type int, with a default of 1000000.

Permission

Execute permission defaults to the public group.

aaPrivateNSAddLeaf

Adds a single object in the private namespace under the currently selected object in the namespace hierarchy.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaPrivateNSAddLeaf *wwTagKey, NameKey, ServerKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
ServerKey	The unique numerical identifier of an AVEVA Historian server. This value is of data type int, with a default of 1.

Permission

Execute permission defaults to the public group.

aaPrivateNSDeleteGroup

Deletes a group object, as well as any objects under it, in the private namespace.

Syntax

aaPrivateNSDeleteGroup *NameKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaPrivateNSDeleteLeaf

Deletes a single object in the private namespace.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaPrivateNSDeleteLeaf *NameKey, wwTagKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaPrivateNSExpand

Expands the tree view one level under a single parent object in the private namespace.

Syntax

aaPrivateNSExpand *PKey, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Remarks

A parent object can have one or more objects below it in the namespace hierarchy.

Permission

Execute permission defaults to the public group.

aaPrivateNSSelect

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Selects all valid group objects (items) for the current user in the private namespace.

Syntax

aaPrivateNSSelect

Permission

Execute permission defaults to the public group.

aaPrivateNSUpdateGroup

Updates a group object in the private namespace.

Syntax

aaPrivateNSUpdateGroup *NameKey, Name, Type*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaPublicNSAddGroup

Adds a group object in the public namespace under the specified parent object in the namespace hierarchy.

Syntax

aaPublicNSAddGroup *Name, ParentKey, Type*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.
ParentKey	The unique identifier for a named object in this namespace. This value is of data type int, with no default.
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy 1999001 = WinPlatform object 1999003 = AppEngine object 1999013 = Area object 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects 1999024 = RedundantDIObject object 1999033 = Undeployed object represented by a generic name 1999901 = ApplicationObject 1999902 = Traceability object This value is of data type int, with a default of 1000000.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSAddLeaf

Adds a single object in the public namespace under the currently selected object in the namespace hierarchy.

Syntax

aaPublicNSAddLeaf *wwTagKey, NameKey, ServerKey*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
ServerKey	The unique numerical identifier of an AVEVA Historian server. This value is of data type int, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSDeleteGroup

Deletes a group object, as well as any objects under it, in the public namespace.

Syntax

aaPublicNSDeleteGroup *NameKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSDeleteLeaf

Deletes a single object in the public namespace.

Syntax

aaPublicNSDeleteLeaf *NameKey, wwTagKey*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Argument	Description
NameKey		The unique identifier for the object in the namespace. This value is of data type int, with no default.
	wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSExpand

Expands the tree view one level under a single parent object in the public namespace.

Syntax

aaPublicNSExpand *PKey, FilterStr*

where:

	Argument	Description
PKey		A local variable used to identify the object in the namespace. This value is of data type int, with no default.
	FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Remarks

A parent object can have one or more objects below it in the namespace hierarchy.

Permission

Execute permission defaults to the public group.

aaPublicNSSelect

Selects all valid group objects (items) in the public namespace.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaPublicNSSelect

Permission

Execute permission defaults to the public group.

aaPublicNSUpdateGroup

Updates a group object in the public namespace.

Syntax

```
aaPublicNSUpdateGroup NameKey, Name, Type
```

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy 1999001 = WinPlatform object 1999003 = AppEngine object 1999013 = Area object 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects 1999024 = RedundantDIObject object 1999033 = Undeployed object represented by a generic name 1999901 = ApplicationObject 1999902 = Traceability object This value is of data type int, with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaRedirectToInTouch

Redirects the tag address (item name) to the InTouch node, rather than to the original I/O Server.

Syntax

aaRedirectToInTouch *IOServerKey, InTouchNodeKey*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.
InTouchNodeKey	The unique numerical identifier of the named InTouch node. This value is of data type int, with no default.

Remarks

When you redirect to InTouch HMI software, all tag values will come from the HMI, not directly from the I/O Server. If you redirect an I/O Server, all topics and tags for that particular I/O Server are affected.

Permission

Execute permission defaults to the aaAdministrators group.

aaSaveChartConfiguration

Returns configuration settings for a particular InSight chart.

Syntax

aaSaveChartConfiguration *ChartConfigurationName, ChartConfigurationUrl, ChartConfigurationType, ChartConfigurationShareMode, LastSharedDateTimeUtc, TimeAggregate, TimePreset, ChartType, MobileShareMode, EmbedShareMode, LastAccessDateTimeUtc, ChartConfiguationKeyword, ChartConfigurationTag, ChartConfigurationProperty, DashboardConfigurationDetail*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
ChartConfigurationName	The name of the InSight content.
ChartConfigurationUrl	The web address for the InSight content.
ChartConfigurationType	Specifies what type of chart was savedFor example, single chart or dashboard.
ChartConfigurationShareMode	Specifies whether the InSight content is shared.
LastSharedDateTimeUtc	Specifies when the InSight content was last shared.
TimeAggregate	Specifies the aggregates used by the saved conten. For example, Hour/Day for a Column chart.
TimePreset	Specifies the selected time frame of the saved content. For example: Last 30 days, Last hour, or specific start and end times (for Custom).
ChartType	The type of chart used for this InSight content.
MobileShareMode	Specifies whether this InSight content is shared with mobile users.
EmbedShareMode	Specifies whether this InSight content can be embedded into a web page or other object.
LastAccessDateTimeUtc	Specifies when the InSight content was last accessed.
ChartConfiguationKeyword	Specifies keywords for the chart.
ChartConfigurationTag	Specifies tags used in the chart.
ChartConfigurationProperty	Specifies a property used by the chart.
DashboardConfigurationDetail	Specifies the position/index of the chart when it is displayed in a dashboard with other charts.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaSearchMessageInsert

Inserts a document to the SearchMessageSyncRequest table so that the search indexer service can dequeue and index that document.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaSearchMessageInsert *JobId, IndexingMessage, DocumentType*

where:

Argument	Description
JobId	The unique numerical identifier for a search message synchronization job.
IndexingMessage	Details from the JSON file used for search indexing to make the associated content searchable. This is an example: `{"_messageid":"test_636571577850365608","_tenantid":"","_body":[{"_keywords":[],"_fields":[{"ContentName":"test","_analyzers":"nGram"}],"_id":"afwwGwD30FUMl9UpE`
DocumentType	Specifies the type of index document; for example SavedContent or Tag.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaSetCalculatedAISamples

Used to _____

Syntax

aaSetCalculatedAISamples

Permission

Execute permission defaults to_____.

aaSetServerTimeStamp

Specifies whether or not incoming data values will be timestamped using the time of the local AVEVA Historian.

Syntax

aaSetServerTimeStamp *TopicName, ServerTimeStamp*

where:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
TopicName	The name of the topic. This value is of data type nvarchar(80), with no default.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the Storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaSetStorageRule

Sets storage rules at various levels of the tag definition.

Syntax

aaSetStorageRule *Type, Key, StorageType, StorageRate, TimeDB, ValueDB, AcqType, DBType, RateDB, ServerTimeStamp, LateData, IdleDuration, ProcInterval*

where:

Argument	Description
Type	The level at which the new storage rule will be set for the tag definition. 1 = I/O Server 2 = Topic This value is of data type tinyint, with no default.
Key	The database key value for the relevant type, either the I/O Server key or the topic key. This value is of data type int, with no default.
StorageType	The type of storage defined for the tag. 0 = Not stored.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type tinyint, with no default.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in seconds. This value is of data type int, with a default of 0.
TimeDB	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of -1.
ValueDB	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. This value is of data type float, with a default of -1.
AcqType	Used to turn acquisition on or off. 0 = Acquisition off 1 = Acquisition on This value is of data type smallint, with a default of -1.
DBType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of -1.
RateDB	Used to percentage of deviation in the full-scale value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. This value is of data type float, with a default of -1.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type smallint, with a default of -1.
LateData	Used to enable acquisition of "late" data. 0 = Late data disabled 1 = Late data enabled This value is of data type smallint, with a default of -1.
IdleDuration	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds. This value is of data type int, with a default of 60.
ProcInterval	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration. This value is of data type int, with a default of 120.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

To ignore an argument, set the value to -1.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaSetTagStorage

Sets storage on or off from various level of the tag definition.

Syntax

```
aaSetTagStorage Type, List, Set
```

where:

Argument	Description
Type	The level at which the new storage rule will be set for the tag definition. 1 = I/O Server 2 = Topic This value is of data type tinyint, with no default.
List	If the type is an I/O Server, topic or public group, the IDENTITY key(s) of the relevant type. If the type is a tag, a list of tagnames separated by commas. This value is of data type nvarchar(4000), with no default.
Set	Used to set storage on or off. Valid values are: • ON • OFF This value is of dataype varchar(3), with no default.

Remarks

This stored procedure applies to analog, discrete, string and complex tag types.

Permissions

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Examples

The following example turns data storage off for all tags associated with I/O Servers that are identified by the IOServerKeys 2 and 3.

```
aaSetTagStorage 1, '2,3', 'OFF'
```

The following example turns data storage on for the listed tags.

```
aaSetTagStorage 3, 'Tag1, Tag2, Tag3', 'ON'
```

aaSnapshotDetailSelect

Returns snapshot information from the columns of the SnapshotDetail table, based on the storage size.

Syntax

aaSnapshotDetailSelect *StorageSize*

where:

Argument	Description
StorageSize	The storage size, in bytes, of the tag value: -1 = Blob; 0 = Variable length string; 1 = 1 byte; 2 = 2 byte; 4 = 4 byte; 8 = 8 byte. This value is of data type int, with a default of NULL.

Remarks

If you do not pass an argument for the storage size, information for all storage sizes in the table will be returned.

Permission

Execute permission defaults to the public group.

aaSnapshotDetailUpdate

Updates the SnapshotDetail table.

Syntax

aaSnapshotDetailUpdate *StorageSize, SnapshotSize, ImageTime, ThresholdTime*

where:

Argument	Description
StorageSize	The storage size, in bytes, of the tag value: -1 = Blob

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	0 = Variable length string 1 = 1 byte 2 = 2 byte 4 = 4 byte 8 = 8 byte This value is of data type int, with no default.
SnapshotSize	The maximum size of the snapshot, in bytes. If this limit is reached, a new snapshot is created. This value is of data type int, with no default.
ImageTime	The interval, in seconds, between updates to the snapshot file. The snapshot file is updated with tag value information from the snapshot buffer, which resides in memory. This value is of data type int, with no default.
ThresholdTime	The maximum amount of time, in seconds, that can elapse before a new snapshot is automatically created, provided that the value for the snapshot size has not been reached. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSnapToSummary

Used by the event system when configuring summary operations.

Syntax

aaSnapToSummary *OpKey, Start, End, DateStamp*

where:

Argument	Description
OpKey	An internal variable that identifies the summary operation to perform. This value is of data type int, with no default.
Start	The starting timestamp for the calculation. This value is of data type datetime2(7), with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
End	The ending timestamp for the calculation. This value is of data type datetime2(7), with no default.
DateStamp	The time the summary operation was performed. This value is of data type smalldatetime, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaSpaceManager

Used by the system to manage the amount of disk space used to store historical data for summaries and events.

Syntax

aaSpaceManager

Remarks

This stored procedure is automatically run by the system every ten minutes. This stored procedure executes the aaDeleteOlderEvents and aaDeleteOlderSummaries stored procedures to clear out old historical data. The duration for which event and summary history is kept is based on system parameters stored in the SystemParameter table.

Permission

Execute permission defaults to the aaAdministrators group.

aaStorageLocationSelect

Selects a storage location.

Syntax

aaStorageLocationSelect *StorageType, StorageNodeKey*

where:

Argument	Description
StorageType	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent, 5 = Tag data index, 6 = Tag data index write-ahead logs, 7

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	= Revision tag data write-ahead logs. There can be only one storage location of each type. This value is of data type int, with a default of NULL.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.

Permission

Execute permission defaults to the public group.

aaStorageLocationUpdate

Updates the storage location.

Syntax

aaStorageLocationUpdate *StorageType, StorageNodeKey, SortOrder, Path, MaxMBSize, MinMBThreshold*

where:

Argument	Description
StorageType	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent, 5 = Tag data index, 6 = Tag data index write-ahead logs, 7 = Revision tag data write-ahead logs. There can be only one storage location of each type. This value is of data type int, with no default.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with no default.
SortOrder	Applies only to the alternate area. If more than one location is defined, the sort order determines the order in which the alternate areas are used. Reserved for future use. This value is of data type int, with no default.
Path	The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different internal hard drive. If you use a network location, the InSQLConfiguration virtual service account must have full access to the location. You can grant that access by using the credentials of the computer account in the format <domain_name>\<computer_name>$. The locations must be specified using UNC notation. Mapped drives are not supported. This value is of data type nvarchar(255), with no default.
MaxMBSize	The limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used. This value is of data type int, with no default.
MinMBThreshold	The minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block (before any compression) by 1.5 to determine the minimum threshold. This value is of data type int, with no default.
MaxAgeThreshold	The age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied. This value is of data type int, with a default of 0.

Permission

Execute permission defaults to the aaAdministrators group.

aaStringDetail

Returns a description for one or more specified tags.

Syntax

aaStringDetail *TagList*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaStringTagDelete

Deletes a string tag.

Syntax

```
aaStringTagDelete wwTagKey
```

Arguments

wwTagKey

The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaStringTagInsert

Inserts a string tag.

Syntax

aaStringTagInsert *TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, MaxLength, InitialValue, TopicKey, IOServerKey, CurrentEditor, DoubleByte, SamplesInActiveImage, ServerTimeStamp, DeadbandType, AIRetrievalMode, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	Historian system. This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. This value is of data type int, with a default of 1000.
ItemName	The address string of the tag. This value is of data type nvarchar(256), with a default of an empty string.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CreatedBy	The name of the user or application that created the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
MaxLength	The maximum number of characters for the string. This value is of data type smallint, with a default of 131. Valid values are: • 8 • 16 • 24 • 32 • 48 • 64 • 128 • 131 • 256 • 512
InitialValue	The initial value as imported from an external source (for example, from InTouch). This value is of data type nvarchar(512), with a default of an empty string.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server. This value is of data type int, with a default of 0.
DoubleByte	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. This value is of data type tinyint, with a default of 0.
SamplesInActiveImage	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
AIRetrievalMode	Used to specify the behavior of retrieval for data in active image. You can either retrieve from all acquired data values that are currently in the active image, or only the data values that are configured to be stored on disk. Data on disk may be a subset of that in the active image, depending on the storage rate for the tag. This value is of datatype tinyint. Valid values are: 0 = All of the values received into the active image will be included in the returned data (default). 1 = Only the values that will be moved into storage will be included in the returned data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
SourceTag	The name of the source tag to create the tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
AITag	Used to specify whether the tag's values are stored by the Classic Storage subsystem. 0 = Not stored by the Classic Storage subsystem; 1 = Stored by the Classic Storage subsystem. This value is of data type bit, with a default of 1.
TagId	The unique identifier for the tag. The value is of data type uniqueidentifier, with a default of NULL.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.
AIHistory	Used to specify whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage. 1 = The tag may have data previously collected by classic storage. This value is of data type bit, with a default of 1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaStringTagSelect

Selects a string tag.

Syntax

aaStringTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaStringTagUpdate

Updates a string tag.

Syntax

aaStringTagUpdate *wwTagKey, TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, MaxLength, InitialValue, TopicKey, IOServerKey, CurrentEditor, DoubleByte, SamplesInActiveImage, ServerTimeStamp, DeadbandType, AIRetrievalMode, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as for the aaStringTagInsert stored procedure. However, only these have defaults:

- *AcquisitionType*
- *DeadbandType*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- *StorageType*
- *CreatedBy*
- *DateCreated*
- *MaxLength*
- *DoubleByte*
- *SamplesInActiveImage*
- *ServerTimeStamp*
- *AIRetrievalMode*
- *SourceTag*
- *SourceServer*
- *AITag*
- *TagId*
- *ChannelStatus*
- *AIHIstory*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaSummaryActionInsert

Used by the event subsystem to perform a summary operation for the specified tag.

Syntax

aaSummaryActionInsert *EventTagName, SumDateTime*

where:

Argument	Description
EventTagName	The name of the event tag with which the summary operation is associated. This value is of data type nvarchar(256), with no default.
SumDateTime	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends. This value is of data type datetime2(7), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryDetail

Returns summary details for one or more tags that are associated with a particular summary operation. The type of aggregation for the tag can optionally be included for each tag that you list.

The starting and ending times are used to specify the time at which the calculation started/ended for the operation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaSummaryDetail *TagList, StartTime, EndTime, OrderBy*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the calculation. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the calculation. This value is of data type nvarchar(50), with no default.
OrderBy	The column by which the results will be ordered. By default, the TagName column is used. This value is of data type nvarchar(500).

Permission

Execute permission defaults to the public group.

Examples

This example returns the average and minimum values for 'ReactTemp' and the maximum value for 'ReactLevel' between 12:12 p.m. and 2:14 p.m. on May 12, 2001. The returned rows are ordered by the date of the summary.

```
aaSummaryDetail "ReactTemp('AVG','MIN'), ReactLevel('MAX')", "2001-05-12 12:12:00.000",
"2001-05-12 12:14:00.000", "SummaryDate"
```

This example returns all aggregate values for 'ReactTemp' and 'ReactLevel' between 12:12 p.m. and 2:14 p.m. on May 12, 2001.

```
aaSummaryDetail "ReactTemp, ReactLevel", "2001-05-12 12:12:00.000", "2001-05-12
12:14:00.000"
```

aaSummaryOperationDelete

Deletes a summary operation.

Syntax

aaSummaryOperationDelete *OperationKey*

where:

Argument	Description
OperationKey	The unique numerical identifier for the summary

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	operation. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryOperationInsert

Inserts a summary operation that will be associated with the specified event tag.

Syntax

aaSummaryOperationInsert *TagName, CalcType, Duration, Resolution, TimeStamp, Description*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.
CalcType	The type of calculation to be performed: SUM, MAX, MIN, or AVG. This value is of dataype char(3), with no default.
Duration	The period, in seconds, for which the calculation is performed. This value is of data type real, with no default.
Resolution	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date. This value is of data type int, with no default.
TimeStamp	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends. 0 = Beginning of the calculation period 1 = End of the calculation period

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	This value is of data type tinyint, with no default.
Description	The description of the summary operation. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryOperationSelect

Selects a summary operation.

Syntax

```
aaSummaryOperationSelect EventTagName, CalcType, Duration, Resolution, TimeStamp
```

Arguments

EventTagName

The name of the event tag with which the summary operation is associated. This value is of data type nvarchar(256), with a default of NULL.

The remaining arguments are the same as for the aaSummaryOperationInsert stored procedure. However, all of the arguments have a default of NULL.

Remarks

The arguments of this stored procedure are used in three ways: (1) if no arguments are specified, all summary operations will be returned; (2) if the EventTagName argument is specified, all summary operations for that event tag will be returned; (3) if all arguments are specified, only the summary operation that matches the criteria will be returned.

Permission

Execute permission defaults to the public group.

aaSummaryOperationUpdate

Updates the summary operation that is associated with a specified event tag.

Syntax

aaSummaryOperationUpdate *OperationKey, TagName, CalcType, Duration, Resolution, TimeStamp, Description*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

where:

Argument	Description
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.

The remaining arguments are the same as for the aaSummaryOperationInsert stored procedure. However, only the *Description* argument has a default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryTagListDelete

Deletes summary information for a tag.

Syntax

aaSummaryTagListDelete *SumVarKey*

where:

Argument	Description
SumVarKey	The unique numerical identifier for a summarized tag. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryTagListInsert

Inserts summary information for a specified tag.

Syntax

aaSummaryTagListInsert *TagName, OperationKey, LowerLimit, UpperLimit, Description*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	nvarchar(256), with no default.
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.
LowerLimit	The lower limit of validity for the tag's value. Values lower than this limit are not used in the calculation. By default, this value is set to -1000000000. This value is of data type real.
UpperLimit	The upper limit of validity for the tag's value. Values higher than this limit are not used in the calculation. By default, this value is set to 1000000000. This value is of data type real.
Description	The description of the summarized tag. This normally describes the result of the operation, although this description can be the same as that of the tag on which the operation is performed. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryTagListSelect

Selects summary information for a tag.

Syntax

aaSummaryTagListSelect *OperationKey, TagName*

where:

Argument	Description
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the public group.

aaSummaryTagListUpdate

Updates summary information for a specified tag.

Syntax

aaSummaryTagListUpdate *SumVarKey, TagName, OperationKey, LowerLimit, UpperLimit, Description*

where:

Argument	Description
SumVarKey	The unique numerical identifier for a summarized tag. This value is of data type int, with no default.

The remaining arguments are the same as for the aaSummaryTagListInsert stored procedure.

Permission

Execute permission defaults to the aaAdministrators group.

aaSystemConfigNSExpand

Expands the tree view under a single object in the system namespace. This stored procedure is used by the Operations Control Management Console.

Syntax

aaSystemNSExpand *PKey, FKey1, FKey2, FKey3, TokenType, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FKey1-FKey3	A local variable used to determine the position of the object in the tree view. This value is of data type int, with no default.
TokenType	The type of system namespace group. 1000010 = Data Acquisition

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	1000017 = System Driver 1000018 = IDASs 1000019 = I/O Servers This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemNSExpand

Expands the tree view under a single object in the system namespace.

Syntax

aaSystemNSExpand *PKey, FKey1, FKey2, FKey3, TokenType, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FKey1-FKey3	A local variable used to determine the position of the object in the tree view. This value is of data type int, with no default.
TokenType	The type of system namespace group. 1000010 = Data Acquisition 1000017 = System Driver 1000018 = IDASs 1000019 = I/O Servers This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the public group.

aaSystemNSExpand2

Expands the tree view under a single object in the system namespace.

Note: This stored procedure is a simpler version of the aaSystemNSExpand stored procedure.

Syntax

aaSystemNSExpand2 *PKey, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemParameterSelect

Returns details for a specified system parameter, such as a description of the parameter, the current value, and so on. If you do not specify a name, the stored procedure returns details for all defined system parameters.

Syntax

aaSystemParameterSelect *name*

where:

Argument	Description
Name	The unique name for the system parameter. This value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemParameterUpdate

Updates the value and description for a specified system parameter. If you do not provide a description, the previous description is used.

Syntax

aaSystemParameterUpdate *Name, Value, Description*

where:

Argument	Description
Name	The unique name for the system parameter. This value is of data type nvarchar(50), with no default.
Value	The value of the system parameter. This value is of data type sql_variant, with no default.
Description	The description of the system parameter. This value is nvarchar(255), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTagConfig

Used by the internal Configuration Manager when the AVEVA Historian starts.

Syntax

aaTagConfig

Remarks

This stored procedure takes a snapshot of the system configuration at the tag level.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the public group.

aaTagConfigModified

Used by the internal configuration object.

Syntax

aaTagConfigModified

Remarks

This stored procedure has the same functionality as the aaTagConfig stored procedure, but only retrieves the database modifications pending when a commit of changes is performed.

Permission

Execute permission defaults to the public group.

aaTagConfigSelect

Used by the Operations Control Management Console to return a list of tags associated with a particular engineering unit (for analog tags) or message (for discrete tags).

Syntax

aaTagConfigSelect, *TagType, Key, FilterStr*

where:

Argument	Description
TagType	The type of tag to retrieve. 1 = Analog 2 = Discrete This value is of data type int, with no default.
Key	The database key value for the relevant type, either the message key or the engineering unit key. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of N%.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to public group.

aaTagInfo

Returns definition information for each specified tag.

Syntax

aaTagInfo *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaTagType

Returns the tag type for each specified tag.

Syntax

aaTagType *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaTimeDetectorDetailInsert

Inserts time detector details that are associated with a specified event tag.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaTimeDetectorDetailInsert *FrequencyID, TagName, Periodicity, StartDateTime, RunTimeDay, RunTimeHour, RunTimeMin*

where:

Argument	Description
FrequencyID	The unique numerical identifier for the frequency. Used to link a frequency with a time-based detector. 1= Hourl 2 = Daily 3 = Weekly 4 = Monthly 5 = Periodi 6 = Other (Reserved for future use) This value is of data type int, with no default.
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Periodicity	The interval period in minutes between detector events. Only used for a periodic detection. This value is of data type int, with no default.
StartDateTime	The timestamp from which the time detector starts. Only used for a periodic detection. This value is of data type datetime2(7), with no default.
RunTimeDay	In the context of a weekly detector, RunTimeDay maps the week day number (0 = Sunday – 6 = Saturday). In the context of a monthly detector, RunTimeDay maps to the day of the month. Not used for periodic detections. This value is of data type tinyint, with no default.
RunTimeHour	The hour of the day at which the time detector triggers. Not used for periodic detections. This value is of data type tinyint, with no default.
RunTimeMin	The minute of the hour at which the time detector triggers. Not used for periodic detections. This value is of data type tinyint, with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTimeDetectorDetailSelect

Selects the time detector from the TimeDetectorDetail table that is associated with the specified event tag.

Syntax

aaTimeDetectorDetailSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaTimeDetectorDetailUpdate

Updates the time detector in the TimeDetectorDetail table that is associated with the specified event tag.

Syntax

aaTimeDetectorDetailUpdate *FrequencyID, TagName, Periodicity, StartDateTime, RunTimeDay, RunTimeHour, RunTimeMin*

Arguments

The arguments are the same as for the aaTimeDetectorDetailUpdate stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTopicDelete

Deletes an I/O topic.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaTopicDelete *TopicKey*

where:

Argument	Description
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTopicInsert

Inserts an I/O topic.

Syntax

aaTopicInsert *StorageNodeKey, IOServerKey, Name, TimeOut, LateData, IdleDuration, ProcessingInterval*

where:

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with no default.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.
Name	The name of the topic. This value is of data type nvarchar(80), with no default.
TimeOut	The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic. This value is of data type int, with a default of 60000.
LateData	Used to enable acquisition of "late" data. 0 = Late data disabled 1 = Late data enabled This value is of data type bit, with a default of 0.
IdleDuration	The amount of time, in seconds, before data is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds. This value is of data type int, with a default of 60.
ProcessingInterval	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration. This value is of data type int, with a default of 120.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTopicSelect

Selects an I/O topic.

Syntax

aaTopicSelect *TopicKey*

where:

Argument	Description
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaTopicUpdate

Updates an I/O topic.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

aaTopicUpdate *TopicKey, StorageNodeKey, IOServerKey, Name, TimeOut, LateData, IdleDuration, ProcessingInterval*

where:

Argument	Description
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with no default.

The remaining arguments are the same as for the aaTopicInsert stored procedure. However, only these have defaults:

- *TimeOut*
- *LateData*
- *IdleDuration*
- *ProcessingInterval*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaUpdateCalculatedAISamples

Used by the system to write the optimum number of samples in the active image to the CalculatedAISamples column in the Tag table. This stored procedure is used by the AVEVA Historian and should not be executed by users.

Syntax

aaSetCalculatedAISamples *TagKey, Samples*

where:

Argument	Description
TagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.
Samples	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

If the AIAutoResize system parameter is set to 1, the system continuously recalculates the optimum number of samples for each tag based on the data rates received. If the calculated value exceeds the current value in the database, then the system updates the CalculatedAISamples column in the _Tag table.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaUpdateChartConfigurationStatistics

Returns statistics about chart configuration access.

Syntax

aaUpdateChartConfigurationStatistics *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The URL for the chart configuration.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaUserAccessLevelSelect

Returns the access level associated with the currently logged on user.

Access levels are:

1 = Undefined AVEVA Historian user (for example, public)

2 = User (hUser permissions)

3 = PowerUser (hPowerUser permission)

3 = Admin (hAdmin permissions)

9999 = dbo

Syntax

aaUserAccessLevelSelect

Remarks

The access level values correspond to values in the UserDetail table, which is populated during installation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Permission

Execute permission defaults to the public group.

aaUserDetailUpdate

Allows the UserDetail table to be populated from information contained in the sysusers table.

Syntax

aaUserDetailUpdate

Permission

Execute permission defaults to the aaAdministrators group.

Stored Procedures for Internal Use

Stored procedures that are used internally by the system are prefixed with "aaInternal". For example, aaInternalAnalogTagExport. Do not use these stored procedures or change them in any way. Internal stored procedures may change from release to release, and no legacy support will be provided.

Creating Stored Procedures

You can create your own stored procedures for use with the AVEVA Historian. All procedure names will be stored in the Runtime database. The stored procedure text will be stored in the Microsoft SQL Server and retrieved at startup (from procedures created in an older session) as well as at creation time (from procedures created in the current session). Temporary procedures will not be supported. No arguments are allowed.

As with Microsoft SQL Server support, support for dynamic stored procedures for the historian will be such that when defining a stored procedure, you can create a stored procedure only in the current database, and the CREATE PROCEDURE statement cannot be combined with other SQL statements in a single batch.

Creating your own stored procedures is useful when you want to execute certain types of queries through a typical ODBC connection. The historian requires a specific ODBC configuration unless you create a stored procedure to execute the query.

For example, the following query creates a stored procedure that returns the timestamp and value for the tag 'ReactLevel" for the last 15 minutes.

```
CREATE PROCedure MyProc
AS
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'ReactLevel'
         AND DateTime >= DATEADD(mi, -15, GETDATE())
         AND DateTime <= GETDATE()
```

User-Defined Functions

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A SQL Server function is a subroutine containing one or more Transact-SQL statements. Functions can be used to encapsulate code for reuse.

faaCheckLicenseViolation

Checks to see if the total number of tags in the AVEVA Historian is less than or equal to the number allowed by the current license.

Syntax

```
SELECT dbo.faaCheckLicenseViolation()
```

Return Type

Integer.

Remarks

If the total number of tags in the system is below the amount allowed, the result of this function will be 0. If not, the number of tags that exceed the allocated amount will be returned. For example, if a system has 100 tags, but the license only allows for 60, a value of 40 will be returned.

faaContainedName

Given a string in the form of "TagName [ContainedName]," returns the ContainedName.

Syntax

```
SELECT dbo.faaContainedName(DisplayName)
```

Arguments

DisplayName

The name as it appears in the model view hierarchy. The display name format is: TagName [ContainedName].

Return Type

Nvarchar(255).

Remarks

The maximum number of characters for both the display name and the returned contained name is 255.

faaGetHierarchicalAttributeNames

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Returns the ArchestrA hierarchical name plus the attribute name, when provided a historian tagname.

Syntax

```
SELECT dbo.faaGetHierarchicalAttributeNames(HistorianTagname)
```

Arguments

HistorianTagname

Tagname within the historian for which you want to return the hierarchical name. This value is of data type nvarchar(256).

faaGetHistorianTagNames

Returns the historian tagname, when provided an ArchestrA hierarchical attribute name starting with tagname as the input.

Syntax

```
SELECT dbo.faaGetHistorianTagNames(HierarchicalAttributeName)
```

Arguments

HierarchicalAttributeName

An ArchestrA hierarchical attribute name starting with tagname as the input. This value is of data type nvarchar(256).

faaLicensedTagDetails

Returns the total number of tags and the number of licensed tags in the system, as well as for each tag type.

Syntax

```
SELECT * FROM dbo.faaLicensedTagDetails()
```

Table Returned

The result is returned in a table format. For example:

Tag Type	Tag Count	Licensed Tags
Analog	213	121
Discrete	68	60

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag Type	Tag Count	Licensed Tags
String	27	26
Event	3	0
Total	311	207

Remarks

System tags and event tags are not included in the total tag count for licensing purposes.

faaLicensedTagTotal

Returns the total number of tags in the system for the specified tag type or for all tags.

Syntax

```
SELECT dbo.faaLicensedTagTotal(TagType)
```

Arguments

TagType
The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 4 = Complex; 0 = All tags.

Return Type

Integer.

Remarks

System tags and event tags are not included in the total tag count for licensing purposes.

faaObjectTagName

Given a string in the form of "TagName [ContainedName]," returns the Tagname.

Syntax

```
SELECT dbo.faaObjectTagName(DisplayName)
```

Arguments

DisplayName

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The name as it appears in the model view hierarchy. The display name format is: TagName [ContainedName].

Return Type

Nvarchar(255).

Remarks

The maximum number of characters for both the display name and the returned tagname is 255.

faaTagsInLicenseViolation

Returns information about tags that have been disabled due to violation of the licensed tag count. The information is returned in a table format.

Syntax

```
SELECT * FROM dbo.faaTagsInLicenseViolation()
```

Table Returned

Column	Data type	Description
TagName	nvarchar(256)	The unique name of the tag within the AVEVA Historian system.
wwTagKey	int	The unique numerical identifier of a tag within a single AVEVA Historian.
Description	nvarchar(512)	The description of the tag.
Address	nvarchar(512)	The address information for the tag, which consists of the computer name, application name, topic, and item name. For example, \\kc1\VIEW!Tagname:ReactLevel.

Remarks

If the licensed tag count has been exceeded, the historian will disable enough tags to run with the allowed licensed tag count. To determine the tags that are in violation of the license, the system first generates the total number of analog, discrete, complex, and string tags. (System and event tags are not included in the total tag count for licensing.) If the total number of tags exceeds the number allowed by the license, the most recent tag additions to the system are disabled. The most recent additions are indicated by the wwTagKey column of the Tag table; the higher the number, the more recent the addition.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

faaTZgetdate

Returns the date/time, in the appropriate time zone.

Syntax

```
SELECT dbo.faaTZgetdate(TimeZone)
```

Arguments

TimeZone

The name of the time zone.

Return Type

nvarchar(100).

Remarks

Use this function instead of the SQL **GetDate()** function to specify a time zone other than the server time zone in a query. To retrieve data in the time zone of the server, just use the SQL **GetDate()** function.

Example

```
DECLARE @starttime datetime
SET @starttime = dbo.faaTZgetdate('eastern daylight time')
SELECT DateTime, TagName, Value FROM History
   WHERE TagName IN ('SysTimeHour', 'SysTimeMin', 'SysTimeSec')
      AND DateTime > DateAdd(mi, -30, @starttime)
      AND DateTime < DateAdd(mi, -5, @starttime)
      AND wwTimeZone = 'eastern daylight time'
```

For more information on using date/time functions in a query, see Using DateTime Functions in the *AVEVA Historian Concepts Guide.*

faaUser_ID

Returns the database user ID (in the Runtime database) for the current user, if the user has an individual login. Returns the database ID of the appropriate Windows security group, if the current user is a group member and does have an individual login (that is, the current user logs in by virtue of being a member of the Windows group).

Syntax

```
SELECT dbo.faaUser_ID()
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Return Type

Integer.

Remarks

This function is used for processing annotations and for support of private namespaces.

- If a user has their own database login, the user has a completely private namespace and private annotations.
- If the user is part of a Windows security group, and logs in only by virtue of being part of the group, the private namespace and annotations are shared with all members of that group.

This stored procedure assumes that Windows users that are logged in are only members of a single Windows group (configured in SQL Server). If a user is found in more than one group, the ID of the last group found is used. This could be a problem if you are expecting a given user to have access to a particular private group.

Also, it is possible that annotations and namespace entries are created under names that you might not expect. For example, a user is a local administrator on a computer, and the user's login has also been added to the aaUsers local group. When logging in to SQL Server, the user will be mapped to the sysadmin fixed server role, by virtue of the user's membership in the BUILTIN\Administrators group. (This assumes that the BUILTIN\Administrators login has not been modified or disabled for security reasons). If this user creates annotations or private namespace entries, these appear as if they had been created by "dbo," rather than by a member of the local aaUsers group.

fww_GetLocalizedText

Returns the strings from the LocalizedText table for the requested language. If the specified translation is not found, English strings are returned.

Syntax

```
SELECT * FROM dbo.fww_GetLocalizedText(LangID)
```

Arguments

LangID

The locale ID for the language used. This ID is also used in the SQL Server syslanguages table. This value is of data type int.

Table Returned

The results are returned as a table that has the same columns as the LocalizedText table. However, the returned table will only include those rows containing strings in the specified language.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Backward Compatibility Entities

Some entities are included in the database for backward compatibility support only. It is recommended that you discontinue the use of these entities, as they will be dropped in a future release.

Note: Utility extended stored procedures are no longer supported by Microsoft SQL Server nor by AVEVA Historian.

Backward Compatibility Views

Backward compatibility views include:

- History Table Views (Backward Compatible)
- Summary Views
- NamedSystemParameter
- SystemNameSpace
- InSQLSysObjects
- v_ErrorLog

History Table Views (Backward Compatible)

The following views reflect the same table structure as the extension tables for which they are named.

These views	Reference this extension table
AnalogHistory, v_AnalogHistory	INSQL.Runtime.dbo.AnalogHistory
AnalogLive, v_AnalogLive	INSQL.Runtime.dbo.AnalogLive
DiscreteHistory, v_DiscreteHistory	INSQLD.Runtime.dbo.DiscreteHistory
DiscreteLive v_DiscreteLive	INSQLD.Runtime.dbo.DiscreteLive
v_History	INSQL.Runtime.dbo.History
v_HistoryBlock	INSQL.Runtime.dbo.HistoryBlock
v_Live	INSQL.Runtime.dbo.Live
StringHistory, v_StringHistory	INSQL.Runtime.dbo.StringHistory
StringLive, v_StringLive	INSQL.Runtime.dbo.StringLive

To allow joins between the analog, string, and discrete tables, the analog and string views reference the OLE DB linked server "INSQL," while the discrete views reference the OLE DB linked server "INSQLD."

Note: In SQL Server Management Studio, the extension tables are listed under the INSQL or INSQLD linked servers under the Server objects tree item.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tag Table Views

The following views are included for backward compatibility. They have the same names and structures as tables that were included in AVEVA Historian before version 2017. These views all reference the _Tag table:

- AnalogSummaryTag
- AnalogTag
- DiscreteTag
- ReplicationTag
- StringTag
- StructureTag

AnalogSummaryTag

Contains one row for each defined analog summary tag. (This is used exclusively for tiered historian installations.) Configuration information specific to analog summary tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) EUKey	int NOT NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float NOT NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float NOT NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float NULL	The minimum value of the raw acquired value.
MaxRaw	float NULL	The maximum value of the raw acquired value.

AnalogTag

Contains one row for each defined analog tag. Configuration information specific to analog tags is stored in this table, while general information for all tag types is stored in the Tag table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) EUKey	int NOT NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float NOT NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float NOT NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float NULL	The minimum value of the raw acquired value.
MaxRaw	float NULL	The maximum value of the raw acquired value.
Scaling	int NOT NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use).
RawType	int NOT NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float NOT NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
IntegerSize	tinyint NOT NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit NOT NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.
RateDeadband	float NOT NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.
InterpolationType	tinyint NOT NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.
RolloverValue	float NOT NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		Therefore, set the rollover value to 10,000.

DiscreteTag

Contains one row for each defined discrete tag. Configuration information specific to discrete tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) MessageKey	int NOT NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.

EventTag

Contains one row for each event definition. Configuration information specific to event tags is stored in the _Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) DetectorTypeKey	int NULL	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
		= Time-based (schedule). DetectorTypeKey is a foreign key from the DetectorType table.
(FK) ActionTypeKey	int NULL	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 = Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary. ActionTypeKey is a foreign key from the ActionType table.
ScanRate	int NULL	The interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).
TimeDeadband	int NOT NULL	The minimum time, in milliseconds, between stored events. If more than one event occurs during the deadband, only the most recent will be stored. The system will not store another event until the specified time has elapsed. A time deadband of 0 indicates that the system will store all events. Reserved for future use.
Logged	bit NOT NULL	Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured. 0 = Not logged; 1 = Logged. The default is 1.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Status	tinyint NOT NULL	The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.
PostDetectorDelay	int NOT NULL	The amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.
UseThreadPool	bit NOT NULL	Used to specify how system threads are used to process events. 1 = All events are handled by a single thread and a single logon to the SQL Server; 0 = Each event uses a separate system thread and logon. This will allow the event subsystem to manage the scan rates of each detector component concurrently. (Reserved for future use.)
DetectorString	nvarchar(1500) NULL	The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian.
ActionString	nvarchar(1500) NULL	The script that specifies the event action. Action scripts run on the local AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
Priority	tinyint NOT NULL	The priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. 0 = Normal; 1 = Critical. The default is 0.
Edge	tinyint NOT NULL	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector.

ReplicationTag

Contains one row for each replication tag. (This is used exclusively for tiered historian installations.) Replication tags follow the same naming convention as regular tags.

Column	Data Type	Description
(PK) (FK) TagName	TagName Type (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
SourceTag	nvarchar(256) NOT NULL	The name of the source tag used for the replication tag.
SourceServer	nvarchar(255) NOT NULL	The name of the tier 1 server with the source tag.

StringTag

Contains one row for each defined string tag. Configuration information specific to string tags is stored in this table, while general information for all tag types is stored in the Tag table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Element	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
MaxLength	smallint NOT NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint NOT NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0.

StructureTag

Contains one row for each summary tag.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique numerical identifier for a SQL template. TagName is a foreign key from the Tag table.
(FK) StructureId	uniqueidentifier NOT NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.

Alarm and Event Views (Backward Compatible)

The following Alarm and Event views are included for backward compatibility:

- Events *
- v_EventHistory *
- v_AlarmHistory *
- v_AlarmHistory2 *
- v_AlarmEventHistory2 *
- v_AlarmEventHistoryInternal2 *

* These views reflect the tables within the WWALMDB and A2ALMDB databases, or history blocks. For details about the related and tables, see "Recording Alarms into an Alarm Database" in the *AVEVA InTouch HMI Alarms and Events Guide*.

The columns of these views are generally compatible between history blocks, and the WWALMDB and A2ALMDB

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

databases, except for the following exceptions in history blocks:

- User1 for alarms: Alarm Severity
- User2: Not populated
- User3: Not populated
- UnAckDuration: Format is in milliseconds.

Note: Earlier versions of AVEVA Historian stored alarm and event in history blocks. They could alternatively be stored in the A2ALMDB database. Earlier versions of AVEVA System Platform used WWALMDB database to store alarms and events. For more information, see A2ALMDB Database in the *AVEVA Historian Administration Guide*.

Summary Views

The summary views allow you to query for data that was summarized by the event subsystem. Each of the views contains data for a specific source, frequency, and operation.

View	Contains One Row For Each
DynDailyAvg	Daily average value for a tag.
DynDailySum	Daily summary value for a tag.
DynHourlyAvg	Hourly average value for a tag.
DynHourlyMax	Hourly maximum value for a tag.
DynHourlyMin	Hourly minimum value for a tag.
DynHourlySum	Hourly summary value for a tag.
v_SummaryData	Returns one row for each summarization of a tag for an associated summary event tag.

Each table view contains the following columns:

Column	Data type	Description
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
SummaryDate	datetime2(7) NOT NULL	The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.
Value	float NULL	The value of the summary.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

v_SummaryData

Returns one row for each summarization of a tag (specified by the TagName column) for an associated summary event tag (specified by the EventTag column). The resolution is applied to data before the calculation is performed. The quality value returned is the highest quality value of the raw data from which the result is calculated.

Column	Data type	Description
TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
CalcType	varchar(3) NULL	The type of calculation to be performed: SUM, MAX, MIN, or AVG.
SummaryDate	datetime2(7) NOT NULL	The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.
Value	float NULL	The value of the summary.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
Duration	real NULL	The period, in seconds, for which the calculation is performed.
Resolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
TimeStamp	tinyint NULL	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
EventTag	TagNameType(nvarchar(256)) NOT NULL	The name of the event tag to which the snapshot tag is related.

NamedSystemParameter

Contains one row for each system parameter. This view provides backward compatibility support for the NamedSystemParameter table, which has been replaced by the SystemParameter table.

Column	Data type	Description
Name	nvarchar(50) NOT NULL	The unique name for the system parameter.
Type	varchar(7) NULL	Used to specify the datatype for the system parameter value. Valid values are: NUMERIC, STRING.
StringValue	varchar(255) NULL	The value of the system parameter. This column only contains values of type STRING.
NumericValue	real NULL	The value of the system parameter. This column only contains values of type NUMERIC.
Editable	bit NULL	Used to determine if the value of the named system parameter can be changed using the InSQL Console. 1 = Editable; 0 = Not editable.
Description	nvarchar(255) NULL	The description of the system parameter.

SystemNameSpace

Contains one row for each item in a single system namespace. Items in the system namespace include servers, topics, and users. The items are organized in a hierarchy. This view provides backward compatibility support for the SystemNameSpace table, which has been deleted.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
NameKey	int NULL	The unique identifier for the object in the namespace.
Type	int NOT NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
ParentKey	int NULL	The unique identifier for a named object in this namespace.
Name	nvarchar(290) NULL	The name of this object in the hierarchy.
PKey	int NULL	The primary key reference for other tables.

InSQLSysObjects

Contains one row for each object in the database for which changes can be tracked. This view provides backward compatibility support for the InSQLSysObjects table, which has been renamed to HistorianSysObjects.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data Type	Description
id	int NOT NULL	The unique identifier for the object.
Type	char(2) NULL	The type of object. C = CHECK constraint; D = Default or DEFAULT constraint; F = FOREIGN KEY constraint; K = PRIMARY KEY or UNIQUE constraint; L = Log; P = Stored procedure; R = Rule; RF = Stored procedure for replication; S = System table; TR = Trigger; U = User table; V = View; X = Extended stored procedure. Currently, only changes for the user tables (object type U) are tracked.
Name	varchar(50) NULL	The name of the modified object.

v_ErrorLog

Contains one row for each system message (error message), if this functionality was enabled. By default, this table is not used.

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The date that the message was written to the system log, in the local time of the AVEVA Historian.
Type	nvarchar(10) NULL	The type of system message.
LocalizedText	nvarchar(256) NULL	The content of the message.
Parameter	nvarchar(256) NULL	Optional details pertaining to the message text. For example, for the message "Disk space remaining on circular path" the parameter would contain the number of MB.
TotalCount	int NULL	Used to prevent "flooding" conditions in the log file. If a particular message is generated numerous times during a relatively short period of time, the message is written to the log file only once, and the total number of times that it occurred appears in this column.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
ModuleID	int NULL	A unique number assigned to the AVEVA Historian subsystem that generated the message.
Host	nvarchar(32) NULL	The computer on which the AVEVA Historian subsystem runs.
FileName	nvarchar(64) NULL	Used to indicate the program file that contains the line of code that an error message comes from. Used for debugging.
Line	int NULL	Used to indicate the line of code that an error message comes from. Used for debugging.

Backward Compatibility Tables

The backward compatibility tables include:

- AnalogHistory (INSQL.Runtime.dbo.AnalogHistory)
- AnalogLive (INSQL.Runtime.dbo.AnalogLive)
- AnalogWideHistory
- DiscreteHistory (INSQL.Runtime.dbo.DiscreteHistory)
- DiscreteLive (INSQL.Runtime.dbo.DiscreteLive)
- DiscreteWideHistory
- GroupTagList
- ManualAnalogHistory
- ManualDiscreteHistory
- ManualStringHistory
- NameSpaceIcons
- StringHistory (INSQL.Runtime.dbo.StringHistory)
- StringLive (INSQL.Runtime.dbo.StringLive)
- StringWideHistory
- WideTableDictionary

AnalogHistory (INSQL.Runtime.dbo.AnalogHistory)

This table has been superceded by the History (INSQL.Runtime.dbo.History) table.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(16) NULL
wwTimeDeadband	int NULL
wwValueDeadband	float NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwInterpolationType	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

AnalogLive (INSQL.Runtime.dbo.AnalogLive)

This table has been superceded by the Live (INSQL.Runtime.dbo.Live) table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRetrievalMode	nvarchar(16) NULL
wwTimeDeadband	int NULL
wwValueDeadband	float NULL
wwTimeZone	nvarchar(50) NULL
wwParameters	nvarchar(128) NULL

AnalogWideHistory

This table (INSQL.Runtime.dbo.AnalogWideHistory) has been superceded by the WideHistory (INSQL.Runtime.dbo.WideHistory) table. AnalogWideHistory is the wide version of AnalogHistory. In a query, this table must be referenced using an OPENQUERY statement.

Column	Data type
DateTime	datetime NOT NULL
TagA1	float NULL
TagA2	float NULL
ManyOtherTags	float NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(20) NULL (default wwRetrievalMode is CYCLIC)
wwTimeDeadband	int NULL
wwValueDeadband	real NULL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwInterpolationType	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

DiscreteHistory (INSQL.Runtime.dbo.DiscreteHistory)

This table has been superceded by the History (INSQL.Runtime.dbo.History) table.

Column	Data type
DateTime	datetime NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(16) NULL
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type
wwTimeStampRule	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

DiscreteLive (INSQL.Runtime.dbo.DiscreteLive)

This table has been superceded by the Live (INSQL.Runtime.dbo.Live) table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRetrievalMode	nvarchar(16) NULL The default mode is DELTA. No other retrieval mode is allowed.
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwParameters	nvarchar(128) NULL

DiscreteWideHistory

This table (INSQL.Runtime.dbo.DiscreteWideHistory) has been superceded by the WideHistory_OLEDB table. DiscreteWideHistory is the wide version of DiscreteHistory, where only discrete tags are considered. It is the same as WideHistory applied to discrete tags. In a query, this table must be referenced using an OPENQUERY statement.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagD1	tinyint NULL
TagD2	tinyint NULL
ManyOtherTags	tinyint NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(20) NULL The default is DELTA.
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

GroupTagList

Contains one row for each defined group of tags.

Column	Data type	Description
(PK) GroupID	int NOT NULL	The identifier for a group of tags.
(PK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain.
Triggerval	float NULL	A value that can be read by an application as a trigger value.

ManualAnalogHistory

This table can be used by custom client applications to store values for analog tags. By default, this table is empty. If written to by a client application, this table will contain one row for each defined analog tag per sample period. ManualAnalogHistory is a normal SQL Server table and does not support any of the AVEVA Historian

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

extensions for handling data.

Column	Data type	Description
(PK) DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data was acquired or stored.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float NULL	The value of the tag at the timestamp. Measured in engineering units.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualDiscreteHistory

This table can be used by custom client applications to store values for discrete tags. By default, this table is empty. If written to by a client application, this table will contain one row for each defined discrete tag per sample period. ManualDiscreteHistory is a normal SQL Server table and does not support any of the AVEVA Historian extensions for handling data.

Column	Data type	Description
(PK) DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data was acquired or stored.
(PK) TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	tinyint NULL	The value of the discrete tag at timestamp. 0 = FALSE; 1 = TRUE; NULL = No data.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ManualStringHistory

This table can be used by custom client applications to store values for string tags. By default, this table is empty. If written to by a client application, this table will contain one row for each defined string tag per sample period. ManualStringHistory is a normal SQL Server table and does not support any of the AVEVA Historian extensions for handling data.

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data was acquired or stored.
(PK) TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512) NULL	The value of the string tag at the timestamp.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

NameSpaceIcons

Contains one row for each defined namespace icon. Namespace icons can be shown in an application browser for each level of the namespace (system, public, and private).

Column	Data type	Description
(PK) Type	int NOT NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name;

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
		1999901 = ApplicationObject; 1999902 = Traceability object.
Icon	image NULL	The icon for the namespace.
Name	nvarchar(30) NOT NULL	The name of the icon.
Description	nvarchar(50) NULL	The description of the icon.

StringHistory (INSQL.Runtime.dbo.StringHistory)

This table has been superceded by the History table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	nvarchar(512) NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(16) NULL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

StringLive (INSQL.Runtime.dbo.StringLive)

This table has been superceded by the Live (INSQL.Runtime.dbo.Live) table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	nvarchar(512) NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRetrievalMode	nvarchar(16) NULL The default mode is DELTA. No other retrieval mode is allowed.
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwParameters	nvarchar(128) NULL

StringWideHistory

This table (INSQL.Runtime.dbo.StringWideHistory) was implemented for completeness. However, you should use the WideHistory table instead of this table. In a query, this table must be referenced using an OPENQUERY

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

statement.

Column	Data type
DateTime	datetime NOT NULL
TagS1	nvarchar(512) NULL
TagS2	nvarchar(512) NULL
ManyOtherTags	nvarchar(512) NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(20) NULL (default wwRetrievalMode is DELTA)
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

TagGroup

Contains one row for each defined tag group. A tag group is a simple, non-hierarchical grouping of tags that can be used by the system.

Column	Data type	Description
(PK) GroupID	int NOT NULL	The identifier for a group of tags.
Description	nvarchar(50) NULL	The description for the group of tags.
CreatedDate	datetime2(7) NULL	The creation date for the tag grouping.
CreatedBy	nchar(18) NULL	The name of the user or application that created the group of tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Column	Data type	Description
Type	int NULL	The type of tag group. 1 to 100 = System use. 100 = Users and third-party client applications.

WideTableDictionary

Contains one row of values for up to 249 tags. These 249 tags appear as columns that will be visible in the data dictionary for each user. Mainly used by ad-hoc query tools. Does not affect the ability to access the values stored for a tag.

Column	Data type	Description
(PK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(PK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.

Renamed Tables

The following table has been renamed. A view named InSQLSysObjects has been created for backward compatibility.

Old Name	New Name
InSQLSysObjects	HistorianSysObjects

Backward Compatibility Stored Procedures

Stored procedures that have been retained for backward compatibility are:

- aaAnalogDetail
- aaDiscreteDetail
- aaStringDetail
- ww_CheckClientVersion
- ww_CheckWhichDb
- ww_dbCheck
- ww_LoadInSQLProcedureBody

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- ww_MDASAnalogTagInsert
- ww_MDASAnalogTagUpdate
- ww_MDASDiscreteTagInsert
- ww_MDASDiscreteTagUpdate
- ww_MDASStringTagInsert
- ww_MDASStringTagUpdate

Backward compatibility extended stored procedures are discussed later in this chapter.

aaAnalogDetail

Returns information about one or more specified analog tags, including the name of the tag, a description, the acquisition rate, the engineering unit, and the minimum and maximum values in engineering units.

Syntax

aaAnalogDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaDiscreteDetail

Returns information about one or more specified discrete tags, including the name of the tag, a description, the message for the TRUE (1) state of the tag, and the message for the FALSE (0) state of the tag.

Syntax

aaDescreteDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Argument	Description
	data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaStringDetail

Returns a description for one or more specified tags.

Syntax

aaStringDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

ww_CheckClientVersion

Checks which version of the client application is running.

Syntax

```
ww_CheckClientVersion AppName, AppVersion
```

Arguments

AppName

The name of the application. This value is of data type nvarchar(255), with a default of an empty string.

AppVersion

The version number of the application. This value is of data type nvarchar(255), with a default of an empty string.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This stored procedure is used by the AVEVA Historian to ensure that a version of an AVEVA client application will run against the database. A client application will not be allowed to run against a database version that does not support that client.

Important: This stored procedure is for AVEVA use only. Do not attempt to use this stored procedure for any third-party client application.

Permission

Execute permission defaults to the public group.

ww_CheckWhichDb

Used to determine if querying the correct database.

Syntax

```
ww_CheckWhichDb dbType
```

Arguments

dbType

The identifier for the database. 1 = Runtime; 2 = Holding; 3 = Development. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

ww_dbCheck

Used to invalidate FactorySuite 1000 clients.

Syntax

```
ww_dbCheck dbType
```

Arguments

dbType

The identifier for the database. 1 = Runtime; 2 = Holding; 3 = Development.

This value is of data type int, with no default.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Remarks

This stored procedure is only used by client applications released prior to FactorySuite 2000.

Permission

Execute permission defaults to the public group.

ww_DBConfig

Returns a summary of the current database configuration, such as number of tags, number of tags per type, storage configuration, event tags, and summary configuration.

Syntax

```
ww_DBConfig
```

Permission

Execute permission defaults to the public group.

ww_LoadInSQLProcedureBody

Used internally to track which stored procedures reference the extension tables.

Syntax

```
ww_LoadSQLProcedureBody ObjName
```

Arguments

ObjName
The name of the stored procedure to load. This value is of data type varchar(92), with no default.

Permission

Execute permission defaults to the public group.

ww_MDASAnalogTagInsert

Used by the Manual Data Acquisition Service to add an analog tag.

This stored procedure calls the aaInternalMDASAnalogTagInsert stored procedure, which is for internal use only.

ww_MDASAnalogTagUpdate

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Used by the Manual Data Acquisition Service to update an analog tag.

This stored procedure calls the aaInternalMDASAnalogTagUpdate stored procedure, which is for internal use only.

ww_MDASDiscreteTagInsert

Used by the Manual Data Acquisition Service to add a discrete tag.

This stored procedure calls the aaInternalMDASDiscreteTagInsert stored procedure, which is for internal use only.

ww_MDASDiscreteTagUpdate

Used by the Manual Data Acquisition Service to update a discrete tag.

This stored procedure calls the aaInternalMDASDiscreteTagUpdate stored procedure, which is for internal use only.

ww_MDASStringTagInsert

Used by the Manual Data Acquisition Service to add a string tag.

This stored procedure calls the aaInternalMDASStringTagInsert stored procedure, which is for internal use only.

ww_MDASStringTagUpdate

Used by the Manual Data Acquisition Service to update a string tag.

This stored procedure calls the aaInternalMDASStringTagUpdate stored procedure, which is for internal use only.

Renamed Stored Procedures

The following stored procedures have been renamed. The old stored procedures have been retained in the system for backward compatibility.

Old Name	New Name
ww_ActionStringSelect	aaAddAnalogSummaryTag
ww_AddTag	aaAddStructureTag
ww_AnalogDetail	aaAnalogDetail
ww_AnalogTagDelete	aaAnalogTagDelete
ww_AnalogTagInsert	aaAnalogTagInsert
ww_AnalogTagSelect	aaAnalogTagSelect
ww_AnalogTagUpdate	aaAnalogTagUpdate
ww_Annotation	aaAnnotationRetrieve

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
ww_AnnotationDelete	aaAnnotationDelete
ww_AnnotationInsert	aaAnnotationInsert
ww_AnnotationSelect	aaAnnotationSelect
ww_AnnotationUpdate	aaAnnotationUpdate
ww_CheckClientVersion	--
ww_CheckWhichDb	--
ww_CleanupAfterCommit	aaCleanupAfterCommit
ww_CommitChanges	aaCommitChanges
ww_CommitChangesAtStartup	aaCommitChangesAtStartup
ww_ContextDelete	aaContextDelete
ww_ContextInsert	aaContextInsert
ww_ContextSelect	aaContextSelect
ww_ContextUpdate	aaContextUpdate
ww_DBChangesPending	aaDBChangesPending
ww_dbCheck	--
ww_DBConfig	aaDBConfig
ww_DeleteOlderEvents	aaDeleteOlderEvents
ww_DeleteOlderSummaries	aaDeleteOlderSummaries
ww_DeleteTag	aaDeleteTag
ww_DetectorStringSelect	aaDetectorStringSelect
ww_DiscreteDetail	aaDiscreteDetail
ww_DiscreteTagDelete	aaDiscreteTagDelete
ww_DiscreteTagInsert	aaDiscreteTagInsert
ww_DiscreteTagSelect	aaDiscreteTagSelect
ww_DiscreteTagUpdate	aaDiscreteTagUpdate
ww_EngineeringUnitDelete	aaEngineeringUnitDelete

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
ww_EngineeringUnitInsert	aaEngineeringUnitInsert
ww_EngineeringUnitSelect	aaEngineeringUnitSelect
ww_EngineeringUnitUpdate	aaEngineeringUnitUpdate
ww_EventDetection	aaEventDetection
ww_EventHistory	aaEventHistorySelect
ww_EventHistoryInsert	aaEventHistoryInsert
ww_EventSnapshot	aaEventSnapshotSelect
ww_EventSnapshotInsert	aaEventSnapshotInsert
ww_EventTagDelete	aaEventTagDelete
ww_EventTagDetail	aaEventTagDetail
ww_EventTagInsert	aaEventTagInsert
ww_EventTagSelect	aaEventTagSelect
ww_EventTagSelectAll	aaEventTagSelectAll
ww_EventTagSelectDeleted	aaEventTagSelectDeleted
ww_EventTagSelectDisabled	aaEventTagSelectDisabled
ww_EventTagSelectInserted	aaEventTagSelectInserted
ww_EventTagSelectUpdated	aaEventTagSelectUpdated
ww_EventTagUpdate	aaEventTagUpdate
ww_GetDbRevision	aaGetDbRevision
ww_GetLastTagKey	aaGetLastTagKey
ww_HistoryBlockSelect	aaHistoryBlockSelect
ww_InSQLConfigNSExpand	aaHistorianConfigNSExpand
ww_InSQLNSExpand	aaHistorianNSExpand
ww_InSQLStatusSelect	aaHistorianStatusSelect
ww_InSQLStatusSet	aaHistorianStatusSet
ww_InTouchNodeTagList	aaInTouchNodeTagList

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
ww_IODriverDelete	aaIODriverDelete
ww_IODriverInsert	aaIODriverInsert
ww_IODriverSelect	aaIODriverSelect
ww_IODriverUpdate	aaIODriverUpdate
ww_IOServerDelete	aaIOServerDelete
ww_IOServerInsert	aaIOServerInsert
ww_IOServerSelect	aaIOServerSelect
ww_IOServerTypeDelete	aaIOServerTypeDelete
ww_IOServerTypeInsert	aaIOServerTypeInsert
ww_IOServerTypeSelect	aaIOServerTypeSelect
ww_IOServerTypeUpdate	aaIOServerTypeUpdate
ww_IOServerUpdate	aaIOServerUpdate
ww_LimitDelete	aaLimitDelete
ww_LimitInsert	aaLimitInsert
ww_LimitNameDelete	aaLimitNameDelete
ww_LimitNameInsert	aaLimitNameInsert
ww_LimitNameSelect	aaLimitNameSelect
ww_LimitNameUpdate	aaLimitNameUpdate
ww_LimitSelect	aaLimitSelect
ww_LimitUpdate	aaLimitUpdate
ww_LoadInSQLProcedureBody	--
ww_MessageDelete	aaMessageDelete
ww_MessageInsert	aaMessageInsert
ww_MessageSelect	aaMessageSelect
ww_MessageUpdate	aaMessageUpdate
ww_ModLogStatus	aaModLogStatus

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
ww_PrivateNSAddGroup	aaPrivateNSAddGroup
ww_PrivateNSAddLeaf	aaPrivateNSAddLeaf
ww_PrivateNSDeleteGroup	aaPrivateNSDeleteGroup
ww_PrivateNSDeleteLeaf	aaPrivateNSDeleteLeaf
ww_PrivateNSExpand	aaPrivateNSExpand
ww_PrivateNSSelect	aaPrivateNSSelect
ww_PrivateNSUpdateGroup	aaPrivateNSUpdateGroup
ww_PublicNSAddGroup	aaPublicNSAddGroup
ww_PublicNSAddLeaf	aaPublicNSAddLeaf
ww_PublicNSDeleteGroup	aaPublicNSDeleteGroup
ww_PublicNSDeleteLeaf	aaPublicNSDeleteLeaf
ww_PublicNSExpand	aaPublicNSExpand
ww_PublicNSSelect	aaPublicNSSelect
ww_PublicNSUpdateGroup	aaPublicNSUpdateGroup
ww_RedirectToInTouch	aaRedirectToInTouch
ww_SetStorageRule	aaSetStorageRule
ww_SetTagStorage	aaSetTagStorage
ww_SnapshotDetailSelect	aaSnapshotDetailSelect
ww_SnapshotDetailUpdate	aaSnapshotDetailUpdate
ww_SnapToSummary	aaSnapToSummary
ww_SpaceManager	aaSpaceManager
ww_StorageLocationSelect	aaStorageLocationSelect
ww_StorageLocationUpdate	aaStorageLocationUpdate
ww_StringDetail	aaStringDetail
ww_StringTagDelete	aaStringTagDelete
ww_StringTagInsert	aaStringTagInsert

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
ww_StringTagSelect	aaStringTagSelect
ww_StringTagUpdate	aaStringTagUpdate
ww_SummaryActionInsert	aaSummaryActionInsert
ww_SummaryDetail	aaSummaryDetail
ww_SummaryOperationDelete	aaSummaryOperationDelete
ww_SummaryOperationInsert	aaSummaryOperationInsert
ww_SummaryOperationSelect	aaSummaryOperationSelect
ww_SummaryOperationUpdate	aaSummaryOperationUpdate
ww_SummaryTagListDelete	aaSummaryTagListDelete
ww_SummaryTagListInsert	aaSummaryTagListInsert
ww_SummaryTagListSelect	aaSummaryTagListSelect
ww_SummaryTagListUpdate	aaSummaryTagListUpdate
ww_SystemConfigNSExpand	aaSystemConfigNSExpand
ww_SystemNSExpand	aaSystemNSExpand
ww_SystemNSExpand2	aaSystemNSExpand2
ww_SystemParameterSelect	aaSystemParameterSelect
ww_SystemParameterUpdate	aaSystemParameterUpdate
ww_TagConfig	aaTagConfig
ww_TagConfigModified	aaTagConfigModified
ww_TagConfigSelect	aaTagConfigSelect
ww_TagInfo	aaTagInfo
ww_TagType	aaTagType
ww_TimeDetectorDetailInsert	aaTimeDetectorDetailInsert
ww_TimeDetectorDetailSelect	aaTimeDetectorDetailSelect
ww_TimeDetectorDetailUpdate	aaTimeDetectorDetailUpdate
ww_TopicDelete	aaTopicDelete

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
ww_TopicInsert	aaTopicInsert
ww_TopicSelect	aaTopicSelect
ww_TopicUpdate	aaTopicUpdate
ww_UpdateCalculatedAISamples	aaUpdateCalculatedAISamples
ww_UserAccessLevelSelect	aaUserAccessLevelSelect
ww_UserDetailUpdate	aaUserDetailUpdate

Extended Stored Procedure Arguments

Note: Extended stored procedures are no longer supported.

Most of the extended stored procedures for the AVEVA Historian use one or more of the following arguments:

StartTime, EndTime

The StartTime string value represents the starting timestamp for the data to query. The EndTime string value represents the ending timestamp for the data to query. The date/time value can be any valid SQL Server date/time string.
The notion of specifying a time zone is not supported. All date/time strings passed as parameters to an extended stored procedure are considered as local server time.
For start and end times, the GetDate() and DateAdd(...) functions are supported, as well as literal dates. For more information, see Literal Date Expressions, GetDate() Expressions, and DateAdd(...) Expressions.
The extended stored procedures round timestamps up or down to the next supported millisecond value: 0, 3, or 7. The standard four-part query and open query do not round timestamps, so if you retrieve data with the extended stored procedure and the four-query, you can have different timestamps for the same data value.

Resolution

The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.

MaxRowCount

The maximum number of rows to be returned for a specified time period.

ValueDeadBand

The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.

TimeDeadband

The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

Description

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The description of the history data that will be copied.

TagN

Tag1, Tag2… TagN are tagname values. Each tagname must be comma delimited and can optionally be surrounded with quotes. For example:

```
exec xp_AnalogHistory "DateAdd(HOUR, -1, GetDate())", "GetDate()", 1000, SysTimeSec,
SysTimeMin
exec xp_DiscreteHistory "DateAdd(HOUR, -1, GetDate())", "GetDate()", 1000, "SysPulse"
```

If a tagname is not of the same type as expected for the named stored procedure then it is ignored. For example, if you pass an analog tagname as a parameter to xp_DiscreteHistory, it will be ignored.
For information regarding valid tagnames, see "Naming Conventions for Tagnames" in Chapter 2, "System-Level Concepts and Functionality" in the AVEVA Historian Concepts Guide.

Literal Date Expressions

Note: Extended stored procedures are no longer supported.

Date expressions can be any valid SQL Server date expression. Here are some examples

```
"4/2/2001 13:00:00:00"
"4/2/2001 12:00 PM"
"2001-4-2 1:00 AM"
```

Years expressed as two digits are interpreted as years in the 1900s. The SQL Server configuration option that supports a two-digit year cutoff is not used.

GetDate() Expressions

Note: Extended stored procedures are no longer supported.

Date/time values can have a string value expression containing the string "GetDate". This is not the same as the SQL Server GetDate() function, although the effect is the same. Example expressions are:

```
GetDate
"GetDate"
"GetDate()"
```

For example:

```
exec xp_AnalogHistory GetDate, "GetDate()", 1000, 'SysTimeSec'
```

DateAdd(...) Expressions

Note: Extended stored procedures are no longer supported.

Date/time values can have a string value expression containing the string "DateAdd(…)". This is not the same as the SQL Server DateAdd() function, although the effect is very similar.

Syntax

```
"DATEADD (datepart, number, date)"
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameters

DatePart

Specifies on which part of the date to return a new value. The following table lists the date parts and abbreviations recognized by the extended stored procedures for the AVEVA Historian.

MILLISECOND	MS
SECOND	SS
MINUTE	MI
HOUR	HH
DAY	DD
WEEKDAY	DW
WEEK	WK
DAYOFYEAR	DY
MONTH	MM
QUARTER	QQ
YEAR	YY

Number

The value used to increment datepart. If you specify a value that is not an integer, the fractional part of the value is discarded. For example, if you specify day for datepart and 1.75 for number, date is incremented by 1.

Date

Either a literal date value without quotes (see Literal Date Expressions) or a GetDate() expression also without quotes (see GetDate() Expressions).

Example Expressions

```
"DateAdd(HOUR, -1, GetDate())"
"DateAdd(MINUTE, -30, 4/2/2001 13:00:00:00)"
```

Extended Stored Procedure Date Expression Examples

```
xp_DiscreteHistory "DateAdd(HOUR, -1, GetDate())", "GetDate()", 1000, 'SysPulse'
xp_DiscreteHistoryDelta "DateAdd(DAY, -1, 4/2/2001)", "GetDate()", 100, 'SysPulse'
```

Backward Compatibility Functions

The following functions have been renamed. The old functions have been retained in the system for backward compatibility.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Old Name	New Name
fww_CheckLicenseViolation	faaCheckLicenseViolation
fww_GetLocalizedText	faaGetLocalizedText
fww_InSQLgetdate	faaTZgetdate
fww_LicensedTagDetails	faaLicensedTagDetails
fww_LicensedTagTotal	faaLicensedTagTotal
fww_TagsInLicenseViolation	faaTagsInLicenseViolation

Historian Retrieval Guide

Welcome

This guide describes how to retrieve data that is stored by an AVEVA Historian server.

You can retrieve data by using:

- Transact-SQL queries.
- Historian Client tools for query construction, queries within Excel workbooks, and trend mapping.
- Historian Insight, a web-based tool for tag-based searches and charting. With Insight, you can save and reuse content (sets of tags and defined timeframes), and can share, embed, and download the results. Historian Insight is installed as a part of AVEVA Historian.
- Historian SDK.
- Tools that use the REST OData interface.

AVEVA Historian Documentation Set

The AVEVA Historian documentation set includes the following guides:

- *AVEVA System Platform Installation Guide*
  This guide provides information on installing the AVEVA Historian, including hardware and software requirements and migration instructions.
- *AVEVA Historian Concepts Guide*
  This guide provides an overview of the entire AVEVA Historian system and its key components.
- *AVEVA Historian Scenarios Guide*
  This guide discusses how to use AVEVA Historian to address some common customer scenarios.
- *AVEVA Historian Administration Guide*
  This guide describes how to administer and maintain an installed AVEVA Historian, such as configuring data acquisition and storage, managing security, and monitoring the system.
- *AVEVA Historian Retrieval Guide*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

This guide describes the retrieval modes and options that you can use to retrieve your data.

- *AVEVA Historian Database Reference*
  This guide provides documentation for all of the AVEVA Historian database entities, such as tables, views, and stored procedures.
- *AVEVA Historian Glossary*
  This guide provides definitions for terms used throughout the documentation set.

In addition, the *AVEVA License Manager Guide* describes the AVEVA License Manager and how to use it to install, maintain, and delete licenses and license servers on local and remote computers.

About Data Retrieval

Through the Data Retrieval subsystem, AVEVA Historian receives SQL queries from clients, locates the requested data, performs necessary processing, and then returns the results.

For configuration and event data, retrieval is made possible by normal SQL queries, because these types of data are stored in SQL Server database tables. Historical data, however, must be retrieved from history blocks and then sent to clients as if it is stored in SQL Server tables.

To accomplish retrieval from both data repositories, the Data Retrieval subsystem includes:

- **An implementation of a SQL Server data provider.**
  This determines whether the requested data is saved in SQL Server tables or in history blocks.
- **Retrieval subsystem.**
  This subsystem is responsible for extracting the requested data from the history blocks and presenting to the AVEVA Historian OLE DB provider as "virtual" history tables.
- **A set of SQL Server extensions.**
  These are implemented as columns in the history tables. You can use these extensions to specify the nature of the rowset that is returned, such as the number of rows returned, the resolution of the data, or the retrieval mode.

For more information on data storage, see Managing Data Storage.

Data Retrieval Subsystem Features

Data Retrieval subsystem features include support for:

- **Queries with all tag types**
  You can include all tag types in the same query when retrieving from the History table. Any combination of tags can be submitted in a single query.
- **Both fixed- and variable-length strings**
- **FILETIME for time computations**
  All internal time computation and manipulation is done using the Win32 FILETIME type. The resolution of FILETIME is 100 nanoseconds.
- **Time handled as Universal Time Coordinated**
  All times are handled internally as UTC. Conversions to and from local time are handled going in and out of retrieval so the external interface is local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Retrieval of different versions**

  Note: If you have an application that uses the older SQL Server datetime format, be aware that some rounding can occur as compared to the newer datetime2 format (for example, 3.3ms vs. 100ns).

History Blocks: A SQL Server Remote Data Source

History blocks are remote data sources used by AVEVA Historian. That is, they are data repositories that exist outside of a SQL Server database file (.MDF). Microsoft sometimes refers to these types of data sources as "non-local data stores."

All tag data is stored in history blocks. For more information on history blocks, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.

OLE DB technology can be used to access data in any remote data store. This access is accomplished though a software component called an OLE DB provider.

Retrieval subsystem

The Retrieval subsystem does the following:

- Fetches history data from history blocks on disk.
- Formats data so that it can be passed up through the system to the AVEVA Historian OLE DB provider or other HCAL-enabled client applications.
- Returns information regarding the history blocks, such as the start and end dates and the location.

About the AVEVA Historian OLE DB Provider

OLE DB (short for "Object Linking and Embedding for Databases") is an application programming interface (API) that allows COM-based client applications to access data that is not physically stored in the SQL Server to which they are connecting.

OLE DB provides access to different types of data in a broader manner. By using OLE DB, you can simultaneously access data from a variety of sources. A query that accesses data from multiple, dissimilar data sources such as these is called a "heterogeneous" or "distributed" query.

SQL Server uses OLE DB to make linking data between the data sources easier. Through OLE DB, Microsoft SQL Server supports Transact-SQL queries against data stored in one or more SQL Server and heterogeneous databases without any need for specialized gateway server applications.

The interface required to access data in a non-local data store (such as the AVEVA Historian history blocks) is provided by a "virtual" server, called an OLE DB provider. OLE DB providers allow you to use the power of the SQL Server query processor to make linking data stored in the SQL Server databases and from history blocks much easier and more robust. Also, the AVEVA Historian OLE DB provider has a rich set of query capabilities.

The name of the AVEVA Historian OLE DB provider is "INSQL". The AVEVA Historian OLE DB provider is installed during AVEVA Historian installation and then associated, or linked, with the Microsoft SQL Server. For information on the syntax for linking the AVEVA Historian OLE DB provider, see Linking the AVEVA Historian OLE DB Provider to the Microsoft SQL Server.

Note: The INSQL OLE DB provider cannot be used in a standalone mode.

To access AVEVA Historian historical data using OLE DB, any COM-based client application must connect directly

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

to the SQL Server and then specify to use the AVEVA Historian OLE DB provider in the syntax of the query.

When you execute a query and specify the AVEVA Historian OLE DB provider in the syntax, the Microsoft SQL Server parser will pass the appropriate parts of the data request to the AVEVA Historian OLE DB provider. The AVEVA Historian OLE DB provider will then interface with the retrieval service to locate the data store, extract the requested information, and return the data to the Microsoft SQL Server as a rowset. Microsoft SQL Server will perform any other processing required on the data and return the data to the client application as a result set and a set of output parameters, if applicable.

The AVEVA Historian OLE DB provider must be present on the server running Microsoft SQL Server. The set of Transact-SQL operations that can be used to retrieve data in the history blocks depends on the capabilities of the AVEVA Historian OLE DB provider.

For more information on OLE DB, see your Microsoft documentation.

Extension Tables for History Data

Many of Historian's tables are implemented as extension tables. That is, they are logical tables that are actually populated from data in history blocks.

Note: Extension tables are also called remote tables.

Data access from the history blocks is made possible by SQL Server's OLE DB provider technology. Client applications must connect directly to the Microsoft SQL Server and then specify to use the AVEVA Historian OLE DB provider in the syntax of the query.

The extension tables are:

History	[INSQL].Runtime.dbo.History
Live	[INSQL].Runtime.dbo.Live
AnalogSummaryHistory	[INSQL].Runtime.dbo.AnalogSummaryHistory
StateSummaryHistory	[INSQL].Runtime.dbo.StateSummaryHistory
HistoryBlock	[INSQL].Runtime.dbo.HistoryBlock
Events	[INSQL].Runtime.dbo.Events

For more information on the history extension tables, see History Tables and Views in the *Historian Database Reference*.

Legacy Process Data Extension Tables

These are legacy (backward compatible) extension tables for process data:

AnalogHistory	[INSQL].Runtime.dbo.AnalogHistory
DiscreteHistory	[INSQL].Runtime.dbo.DiscreteHistory
StringHistory	[INSQL].Runtime.dbo.StringHistory
AnalogLive	[INSQL].Runtime.dbo.AnalogLive

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DiscreteLive	[INSQL].Runtime.dbo.DiscreteLive
StringLive	[INSQL].Runtime.dbo.StringLive

The AnalogHistory, DiscreteHistory, StringHistory, and History tables are the only tables which are updateable. The remaining tables are read-only.

For more information about these tables, see Backward Compatibility Entities in the *Historian Database Reference*.

Legacy Event Data Extension Tables

These are legacy (backward compatible) extension tables for events:

v_AlarmEventHistoryInternal2	[INSQL].Runtime.dbo.LegacyAlarmEventHistory
v_AlarmHistory	[INSQL].Runtime.dbo.LegacyAlarmHistory
v_AlarmHistory2	[INSQL].Runtime.dbo.LegacyAlarmHistory2
v_EventHistory	[INSQL].Runtime.dbo.LegacyEventHistory
v_AlarmEventHistory2	Same as v_AlarmEventHistoryInternal2

Linking the AVEVA Historian OLE DB Provider to the Microsoft SQL Server

Because the AVEVA Historian OLE DB provider retrieves data from the history blocks and presents it to Microsoft SQL Server as a table, it can be thought of as a type of server. The AVEVA Historian OLE DB provider must be added to the Microsoft SQL Server as a "linked" server before it can be used to process queries.

This linking is performed automatically during the AVEVA Historian installation. If, for some reason, you need to re-link the AVEVA Historian OLE DB provider to the Microsoft SQL Server, the statements for linking are as follows:

```
sp_addlinkedserver
   @server = 'INSQL',
   @srvproduct = '',
   @provider = 'INSQL'
   go
   sp_serveroption 'INSQL','collation compatible',true
   go
   sp_addlinkedsrvlogin 'INSQL','TRUE',NULL,NULL,NULL
   go
```

"INSQL" is the name of the AVEVA Historian OLE DB provider as the linked server. Use this name to specify the AVEVA Historian OLE DB provider in a query.

To perform joins between the legacy analog history tables and discrete history tables, the installation program also creates an alias for the same AVEVA Historian OLE DB provider:

```
sp_addlinkedserver
   @server = 'INSQLD',
   @srvproduct = '',
   @provider = 'INSQL'
   go
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
sp_serveroption 'INSQLD','collation compatible',true
go
sp_addlinkedsrvlogin 'INSQLD','TRUE',NULL,NULL,NULL
go
```

For example, if you want to execute a query that performs this type of join, use the normal alias in specifying the first table (the analog history table), and use the second alias in specifying the second table (the discrete history table, hence the "D" added to the alias name).

AVEVA Historian I/O Server

The AVEVA Historian I/O Server (aahIDataService.exe) is the interface for clients to access current data using an IData-compatible protocol. The AVEVA Historian I/O Server can update items with current values for given topics, providing "real-time" I/O Server functionality.

By default, the AVEVA Historian I/O Server runs as a Windows service and can be started and stopped using the Operations Control Management Console. You can also monitor the AVEVA Historian I/O Server from within the Operations Control Management Console. For more information on the Operations Control Management Console, see About Administrative Tools in the *AVEVA Historian Administration Guide.*

The AVEVA HistorianAVEVA Historian I/O Server is a read-only server; clients cannot update data.

The AVEVA Historian I/O Server sends the original OPC quality as it was stored in the AVEVA Historian. The OPC quality remains the same throughout the system, including storage, retrieval, and the AVEVA Historian I/O Server.

AVEVA Historian I/O Server (classic)

The AVEVA Historian I/O Server (classic) (aahIOSvrSvc.exe) is the interface for clients to access current data using the SuiteLink protocol. The AVEVA Historian I/O Server (classic) can update items with current values for given topics, providing "real-time" I/O Server functionality.

The AVEVA Historian I/O Server (classic) is pre-configured with a single topic, Tagname. The AVEVA Historian I/O Server (classic) will listen for clients (such as WWClient or WindowViewer™) that are attempting to establish a connection using the pre-configured topic. After a client connects with the AVEVA Historian I/O Server (classic), a "hot" link is established between the client and the AVEVA Historian I/O Server. For more information on I/O Server addressing conventions, see I/O Server Addressing in the *AVEVA Historian Administration Guide.*

For example, the AVEVA Historian I/O Server (classic) could be used by InTouch WindowViewer to access system tag values provided by the AVEVA Historian to monitor system health. You could configure WindowViewer to generate an alarm when abnormal behavior is detected within the AVEVA Historian.

By default, the AVEVA Historian I/O Server (classic) runs as a Windows service and can be started and stopped using the Operations Control Management Console. You can also monitor the AVEVA Historian I/O Server (classic) from within the Operations Control Management Console. For more information on the Operations Control Management Console, see About Administrative Tools in the *AVEVA Historian Administration Guide.*

The AVEVA Historian I/O Server (classic) is a read-only server; clients cannot update data.

The AVEVA Historian I/O Server (classic) sends the original OPC quality as it was stored in the AVEVA Historian. The OPC quality remains the same throughout the system, including storage, retrieval, and the AVEVA Historian I/O Server (classic).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using SELECT to Retrieve Data

The most common AVEVA Historian query is a SELECT statement:

```
SELECT select_list
   FROM table_source
   WHERE search_condition
      [ GROUP BY group_by_expression ]
      [ HAVING search_condition ]
      [ ORDER BY order_expression [ ASC | DESC ] ]
```

A WHERE clause is mandatory when issuing a SELECT query against any extension table except HistoryBlock.

There are four variations for issuing a SELECT statement to the AVEVA Historian OLE DB provider to retrieve history data:

- Using the Four-Part Naming Convention
- Using an AVEVA Historian OLE DB Provider View
- Using the OPENQUERY Function
- Using the OPENROWSET Function

You should use the four-part name or a provider view to specify the extension table, whenever possible. However, there are instances when the OPENQUERY or OPENROWSET function must be used, such as for queries on wide tables.

For general information on creating SQL queries, see your Microsoft SQL Server documentation.

Using the Four-Part Naming Convention

The linked server name is simply a name by which the AVEVA Historian OLE DB provider is known to the Microsoft SQL Server. In order for a query to be passed on to the AVEVA Historian OLE DB provider, you must specify the linked server name and the extension table name as part of a four-part naming convention.

For example, this query specifies to retrieve data from the History extension table in the AVEVA Historian OLE DB provider:

```
SELECT * FROM INSQL.Runtime.dbo.History
   WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2001-09-12 12:59:00'
      AND DateTime <= '2001-09-12 13:00:00'
```

The four-part naming convention is described in the following table:

Part Name	Description
linked_server	Linked server name. By default, INSQL.
catalog	Catalog in the OLE DB data source that contains the object from which you want to retrieve data. For Microsoft SQL Server type databases, this is the name of the database. To use the AVEVA Historian OLE DB provider, the catalog name will always be "Runtime."

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Part Name	Description
schema	Schema in the catalog that contains the object. For Microsoft SQL Server type databases, this is the name of the login ID for accessing the data. To use the AVEVA Historian OLE DB provider, the catalog name will always be "dbo."
object_name	Data object that the OLE DB provider can expose as a rowset. For the AVEVA Historian OLE DB provider, the object name is the name of the remote table that contains the data you want to retrieve. For example, the History table.

In the case of four-part queries, SQL Server produces the statement that is sent to the AVEVA Historian OLE DB provider from the statement that the user executes. Sometimes this produced statement is incorrect, too complex, or lacks portions of the WHERE clause required for the AVEVA Historian OLE DB provider to return data.

A typical error message when executing unsupported syntax is:

```
Server: Msg 7320, Level 16, State 2, Line 1
Could not execute query against OLE DB provider 'INSQL'.
[OLE/DB provider returned message: InSQL did not receive a WHERE clause from SQL Server. If
one was specified, refer to the InSQL OLE DB documentation]
```

For four-part queries against non-English SQL Servers running on non-English operating systems, the default date format might differ from the English versions. For example, for a French or German SQL Server running on the corresponding operating system, the date/time in a four-part query must be:

yyyy-dd-mm hh:mm:ss.fff

For example:

2003-28-09 09:00:00.000

The default SQL date format is dependent on SQL Server and not on the operating system used. However, you can modify the format using the SQL Server Convert() method. The output of this method can be determined by the regional settings configured for the operating system.

Using an AVEVA Historian OLE DB Provider View

Microsoft SQL Server views have been provided that will access each of the extension tables, eliminating the need to type the four-part server name in the query. These views are named the same as the provider table name.

Note: Backward compatibility views are named according to the v_ProviderTableName convention.

For example:

```
SELECT * FROM History
   WHERE TagName = 'SysTimeSec'
       AND DateTime >= '2001-09-12 12:59:00'
       AND DateTime <= '2001-09-12 13:00:00'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using the OPENQUERY Function

You can use the linked server name in an OPENQUERY function to retrieve data from an extension table. The OPENQUERY function is required for retrieving from the wide table. For example:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT * FROM History
   WHERE TagName = "SysTimeSec"
      AND DateTime >= "2001-09-12 12:59:00"
      AND DateTime <= "2001-09-12 13:00:00"
      ')
```

The following example retrieves data from a wide table:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT DateTime, SysTimeSec
   FROM WideHistory
      WHERE DateTime >= "2001-09-12 12:59:00"
         AND DateTime <= "2001-09-12 13:00:00"
      ')
```

The OPENQUERY portion of the statement is treated as a table by SQL Server, and can also be used in joins, views, and stored procedures. SQL Server sends the quoted statement, unchanged and as a string, to the AVEVA Historian OLE DB provider. Consequently, only the syntax that the AVEVA Historian OLE DB provider can parse is supported. Also, be sure that you do not exceed the 8000 character limit for the statement. Consider the following example:

```
   SELECT * FROM OpenQuery(INSQL, 'XYZ')
```

where "XYZ" is the statement to pass. You should be sure that the value of "XYZ" is not more than 8000 characters. This limit is most likely to cause a problem if you are querying many tags from a wide table.

Also, you should supply the datetime in an OPENQUERY statement in the following format:

yyyy-mm-dd hh:mm:ss.fff

For example:

2001-01-01 09:00:00.000

You cannot use variables in an OPENQUERY statement. For more information, see -old-Using Variables with the Wide Table.

Using the OPENROWSET Function

The linked server name can be used as an input parameter to an OPENROWSET function. The OPENROWSET function sends the OLE DB provider a command to execute. The returned rowset can then be used as a table or view reference in a Transact-SQL statement. For example:

```
SELECT * FROM OPENROWSET('INSQL',' ', 'SELECT DateTime, Quality, QualityDetail, Value
   FROM History
   WHERE TagName in ("SysTimeSec")
   AND DateTime >= "2001-09-12 12:59:00"
   AND DateTime <= "2001-09-12 13:00:00"
   ')
```

Note: If the OpenRowSet/OpenDatasource component is turned off as part of the security configuration for the server, you will receive an error when you try to run this query. If necessary, a system administrator can reset SQL Server settings to enable use of ad hoc queries by executing the sp_configure command.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Supported Syntax Options

The following table indicates the syntax options that are available for queries that use either the four-part naming convention (or corresponding view name) or the OPENQUERY function.

Syntax Element	Four-Part Query	OPENQUERY
ORDER BY	Yes	No. Does not work within the OPENQUERY function. However, will work if used outside of the function.
GROUP BY	Yes	No
TagName IN (..)	Yes	Yes
TagName LIKE '..'	Yes	Yes
Date and time functions (for example, DateAdd)	Yes	Yes
MIN, MAX, AVG, SUM, STDEV	Yes	MIN, MAX, AVG, SUM only
Sub-SELECT with one normal SQL Server table and one extension table	Yes, with restrictions	No
Sub-SELECT with two extension tables	No	No

Unsupported or Limited Syntax Options

The AVEVA Historian OLE DB provider does not support certain syntax options in queries. In general, these limitations are due to underlying limitations in the current Microsoft SQL Server OLE DB Provider implementation.

For general information on creating SQL queries, see your Microsoft SQL Server documentation.

No Notion of Client Context

The OLE DB provider has no notion of a client context. The OLE DB provider is entirely stateless, and there is no persistence across queries in the same connection. This means that you must set the value of a AVEVA Historian time domain extension (for example, cycle count) each time you execute a query.

Also, the OLE DB provider cannot continuously return data (similar to a "hot" link in InTouch HMI software). The OLE DB specification (as defined by Microsoft) does not permit a provider to return rows to a consumer without a request from the consumer.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Limitations on Wide Tables

Wide tables do not have a fixed schema, but a schema which varies from query to query. They are transient tables, existing for the duration of one query only. For this reason, they must be accessed using the OPENQUERY function, which bypasses many of the tests and requirements associated with fixed tables. Wide tables support up to 1024 columns.

For more information on wide tables, see "Wide" History Table Format in the *AVEVA Historian Database Reference*.

LIKE Clause Limitations

The LIKE clause is only supported for the TagName and Value columns. The syntax " `... Value LIKE 'a string' ...` " is only supported for a string table. For example:

```
SELECT TagName, Value FROM History
   WHERE TagName LIKE 'Sys%'
      AND DateTime > '1999-05-24 14:30:00'
      AND DateTime < '1999-05-24 14:32:00'
```

IN Clause Limitations

If you are querying analog, discrete, or string tags from the AnalogTag, DiscreteTag, or StringTag tables (respectively), you cannot use the LIKE clause within an IN clause to condition the tagname unless you are returning the vValue column. This restriction applies if you are using the four-part naming convention or an extension table view.

For example:

```
SELECT DateTime, TagName, vValue, Quality, QualityDetail
   FROM History
      WHERE TagName IN (SELECT TagName FROM StringTag WHERE TagName LIKE 'SysString')
         AND DateTime >='2001-06-21 16:00:00.000'
         AND DateTime <='2001-06-21 16:40:00.000'
         AND wwRetrievalMode = 'Delta'
```

However, it is more efficient to use an INNER REMOTE JOIN to achieve the same results. For more information, see Using an INNER REMOTE JOIN.

OR Clause Limitations

You cannot use the OR clause to specify more than one condition for a time domain extension. For more information, see AVEVA Historian Time Domain Extensions.

Using Joins within an OPENQUERY Function

Joins are not supported within a single OPENQUERY statement. For example, the following query contains an implicit join between the Tag and Live tables, and will fail:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT v.DateTime, v.TagName, v.Value, t.Description
   FROM Tag t, Live v
      WHERE t.TagName LIKE "%Date%"
         AND v.TagName = t.TagName
   ')
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A workaround is to place the join outside of the OPENQUERY. For example:

```
SELECT v.DateTime, v.TagName, v.Value, t.Description
   FROM OPENQUERY(INSQL, 'SELECT DateTime, TagName, Value
   FROM Live
      WHERE TagName LIKE "%Date%"
   ') v, Tag t
      WHERE v.TagName = t.TagName
```

Explicit joins are also not supported within OPENQUERY. For example, the following query will fail:

```
SELECT * FROM OPENQUERY(INSQL, 'SELECT v.DateTime, v.TagName, v.Value, e.Unit
   FROM Live v
   JOIN AnalogTag t ON v.TagName = t.TagName
   JOIN EngineeringUnit e ON t.EUKey = e.EUKey
      WHERE v.TagName LIKE "%Date%"
   ')
```

A work-around is to place the join outside the OPENQUERY. For example:

```
SELECT v.DateTime, v.TagName, v.Value, e.Unit
   FROM OPENQUERY(INSQL, 'SELECT DateTime, TagName, Value FROM Live
      WHERE TagName LIKE "%Date%"
   ') v
   JOIN AnalogTag t ON v.TagName = t.TagName
   JOIN EngineeringUnit e ON t.EUKey = e.EUKey
      ORDER BY t.TagName
```

In general, use four-part syntax wherever possible. All of the previous queries are more conveniently expressed in four-part syntax. For example, the syntax for the preceding query would be:

```
SELECT v.DateTime, v.TagName, v.Value, e.Unit
   FROM INSQL.Runtime.dbo.History v
   JOIN AnalogTag t ON v.TagName = t.TagName
   JOIN EngineeringUnit e ON t.EUKey = e.EUKey
      WHERE v.TagName LIKE '%Date%'
      ORDER BY t.TagName
```

Using Complicated Joins

You can only use simple joins between SQL Server tables and the AVEVA Historian OLE DB extension tables. Joins typically require use of the INNER REMOTE JOIN syntax.

For an example of the INNER REMOTE JOIN syntax, see -old-Using an INNER REMOTE JOIN.

Using a Sub-SELECT with a SQL Server Table and an Extension Table

Using a sub-SELECT with a query on a normal SQL Server table and an extension table should be avoided; it is very inefficient due to the way SQL Server executes the query. For example:

```
SELECT TagName, DateTime, Value
   FROM INSQL.Runtime.dbo.History
      WHERE TagName IN (select TagName FROM SnapshotTag WHERE EventTagName =
      'SysStatusEvent')
         AND DateTime = '2001-12-20 0:00'
```

Instead, it is recommended that you use the INNER REMOTE JOIN syntax:

```
SELECT h.TagName, DateTime, Value
   FROM SnapshotTag st INNER REMOTE JOIN INSQL.Runtime.dbo.History h
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      ON st.TagName = h.TagName
         AND EventTagName = 'SysStatusEvent'
         AND DateTime = '2001-12-20 0:00'
```

The results are:

TagName	DateTime	Value
SysPerfCPUTotal	2001-12-20 00:00:00.000	15.0
SysSpaceMain	2001-12-20 00:00:00.000	1302.0

In general, use the following pattern for INNER REMOTE JOIN queries against the historian is:

<SQLServerTable> INNER REMOTE JOIN <HistorianExtensionTable>

For more information on INNER REMOTE JOIN, see your Microsoft SQL Server documentation.

WHERE Clause Anomalies

In some rare cases, the SQL Server query processor truncates the WHERE clause in an attempt to optimize the query. If you execute a query with a WHERE clause, but an error message is returned stating that no WHERE clause was received by the SQL Server, simply add another condition clause to the query.

For example, in the following query, the SQL Server query processor optimizes out the WHERE clause, because it is superfluous.

```
SELECT DateTime, Value, QualityDetail
   FROM History
      WHERE TagName LIKE '%'
```

A workaround is to add another condition clause. For example:

```
SELECT DateTime, Value, QualityDetail
   FROM History
      WHERE TagName LIKE '%'
      AND wwRetrievalMode = 'delta'
```

CONVERT Function Limitations

The CONVERT function is not supported on the vValue column in an OPENQUERY statement. If you are using OPENQUERY on the History table, you must filter on the vValue column outside of the query.

In the following example, the value of the vValue column is converted to a float. Note that no string tags are included in the query.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime, Quality, OPCQuality, QualityDetail, Value,
vValue, TagName
   FROM History
      WHERE TagName IN ("SysTimeMin", "SysPulse")
         AND DateTime >= "2001-12-30 04:00:00.000"
         AND DateTime <= "2001-12-30 09:00:00.000"
         AND wwRetrievalMode = "Delta"
   ')
   WHERE convert(float, vValue) = 20.0
```

You can also use the following formats:

```
WHERE convert(float, vValue) = 0
WHERE convert(float, vValue) = 0.0
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
WHERE convert(float, vValue) = 1.0
WHERE convert(float, vValue) = 1
WHERE convert(float, vValue) = 20
WHERE convert(float, vValue) = 2.0000e01
```

The following example includes a string tag and converts the vValue value to a char or varchar datatype. All values returned can be converted to a string.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime, Quality, OPCQuality, QualityDetail, Value,
vValue, TagName
   FROM History
      WHERE TagName IN ("SysString", "SysTimeMin", "SysPulse")
         AND DateTime >= "2001-12-30 04:00:00.000"
         AND DateTime <= "2001-12-30 09:00:00.000"
         AND wwRetrievalMode = "Cyclic"
         AND wwCycleCount = 300
   ')
   WHERE convert(varchar(30), vValue) = '2001-12-30 14:00:00'
```

You can also use the following formats:

```
WHERE convert(varchar(30), vValue) = '20'
WHERE convert(varchar(30), vValue) = '1'
WHERE convert(varchar(30), vValue) = '0'
```

SQL Server Optimization of Complex Queries

The SQL Server query optimizer may incorrectly parse a complex query and not send certain query criteria to the Historian OLE DB provider for handling. This can cause unexpected results for the data.

If you suspect that this is happening, use SQL Server Management Studio tools to examine the query plan that the optimizer is using and then modify your query so that the needed criteria gets directed to the Historian OLE DB provider.

For example, the following query will be incorrectly parsed:

```
SELECT GETDATE()
DECLARE @TagList TABLE (TagName nvarchar(256))
INSERT @TagList
   SELECT 'SysTimeSec' UNION
   SELECT 'SysPerfCPUTotal'
-- Prevent the TagName criteria from being sent to the Historian OLE DB provider
(incorrect)
SELECT DateTime, h.vValue, h.TagName
   FROM History h
   INNER REMOTE JOIN @TagList l
   ON h.TagName = l.TagName
   WHERE DateTime >= DATEADD(hour,-1,GETDATE())
      AND DateTime < GETDATE()
      AND wwRetrievalMode = 'AVG'
      AND wwCycleCount=1
GO
```

To correct this issue, rewrite the query so that the tagname criteria is passed to the Historian OLE DB provider correctly.

```
SELECT GETDATE()
DECLARE @TagList TABLE (TagName nvarchar(256))
INSERT @TagList
   SELECT 'SysTimeSec' UNION
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   SELECT 'SysPerfCPUTotal'
-- Force the TagName criteria to be sent to the InSQL OLE DB Provider (correct)
SELECT DateTime, h.vValue, h.TagName
   FROM @TagList l
   INNER REMOTE JOIN History h
   ON h.TagName = l.TagName
   WHERE DateTime >= DATEADD(hour,-1,GETDATE())
      AND DateTime < GETDATE()
      AND wwRetrievalMode = 'AVG'
      AND wwCycleCount=1
GO
```

Using Columns of a Variant Type with Functions

If you use a column of a variant type as the parameter for some functions, SQL Server returns a syntax error. However, the error is not passed to the Historian OLE DB provider to return to clients.

For example, in the following query, the rounding is specified for the vValue column, which is of type variant. The query does not work, but no error is returned by the Historian OLE DB provider.

```
SELECT DateTime, round(vValue, 2)
   FROM History
   WHERE TagName IN ('SysTimeSec')
      AND DateTime = getdate()
      AND wwRetrievalMode = 'Cyclic'
```

Using StartDateTime in the Query Criteria

You cannot use StartDateTime in the query criteria instead of DateTime. For example, the following query works, except that it does not apply the StartDateTime >= @StartDate clause.

```
SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-30,GetDate())
SET @EndDate = GetDate()
SET NOCOUNT OFF
SELECT History.TagName, DateTime = convert(nvarchar, DateTime, 21), Value, vValue,
StateTime, StartDateTime
   FROM History
      WHERE History.TagName IN ('Reactor1Level')
         AND wwRetrievalMode = 'RoundTrip'
         AND wwStateCalc = 'AvgContained'
         AND vValue = convert(SQL_VARIANT, '1')
         AND wwCycleCount = 1
         AND wwTimeStampRule = 'Start'
         AND wwQualityRule = 'Good'
         AND wwFilter = 'ToDiscrete(5.0,>)'
         AND wwVersion = 'Latest'
         AND DateTime >= @StartDate
         AND DateTime <= @EndDate
         AND StartDateTime >= @StartDate
```

Comparison Statements and NULL Values

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQL Server returns an error for a query that contains a comparison statement like 'Value > 0' whenever a NULL is returned. Be sure that you always include 'AND Value IS NOT NULL', so that the NULL values are filtered out.

OPENQUERY and Microsoft Query

Microsoft Query is not able to process an OPENQUERY statement.

AVEVA Historian Time Domain Extensions

Data in the extension tables can be manipulated by using normal Transact-SQL code, as well as the specialized SQL time domain extensions provided by the AVEVA Historian. The AVEVA Historian extensions provide an easy way to query time-based data from the history tables. They also provide additional functionality not supported by Transact-SQL.

The time domain extensions are:

- wwCycleCount
- wwEdgeDetection
- wwFilter
- wwInterpolationType
- wwOption
- wwQualityRule
- wwResolution
- wwRetrievalMode
- wwStateCalc
- wwTimeDeadband
- wwTimeZone
- wwValueDeadband
- wwVersion
- wwTimeStampRule
- wwValueSelector

Note: The wwParameters and wwMaxStates parameters are reserved for future use. The wwRowCount parameter is still supported, but is deprecated in favor of wwCycleCount.

The extensions are implemented as "virtual" columns in the extension tables. When you query an extension table, you can specify values for these column parameters to manipulate the data that will be returned. You will need to specify any real-time extension parameters each time that you execute the query.

For example, you could specify a value for the wwResolution column in the query so that a resolution is applied to the returned data set:

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND Value >= 50
         AND wwResolution = 10
         AND wwRetrievalMode = 'cyclic'
```

Because the extension tables provide additional functionality that is not possible in a normal SQL Server, certain limitations apply to the Transact-SQL supported by these tables. For more information, see Unsupported or Limited Syntax Options.

Although the Microsoft SQL Server may be configured to be case-sensitive, the values for the virtual columns in the extension tables are always case-insensitive.

Note: You cannot use the IN clause or OR clause to specify more than one condition for a time domain extension.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For example, "`wwVersion IN ('original', 'latest')`" and "`wwRetrievalMode = 'Delta' OR wwVersion = 'latest'`" are not supported.

For general information on creating SQL queries, see your Microsoft SQL Server documentation.

Data Retrieval Modes and Options

You can use a variety of retrieval modes and options to suit different reporting needs and applications.

Understanding Retrieval Modes

Different retrieval modes allow you to access the data stored in an AVEVA Historian in different ways. For example, if you retrieve data for a long time period, you might want to retrieve only a few hundred evenly spaced data points to minimize response time. For a shorter time period, you might want to retrieve all values that are stored on the server to get more accurate results.

An AVEVA Historian with a version earlier than 9.0 supports two retrieval modes:

- Cyclic Retrieval
- Delta Retrieval

An AVEVA Historian with a version of 9.0 or higher supports various additional modes:

- Full Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Slope Retrieval
- Counter Retrieval
- ValueState Retrieval

An AVEVA Historian with a version of 10.0 or higher supports the following additional mode:

- RoundTrip Retrieval

An AVEVA Historian version 11.6.14101 or higher supports the following additional mode:

- Predictive Retrieval

An AVEVA Historian with a version of 17.3.100 or higher supports the following additional mode:

- Bounding Value Retrieval

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Cyclic Retrieval

Cyclic retrieval is the retrieval of stored data for the given time period based on a specified cyclic retrieval resolution, regardless of whether or not the value of the tag(s) has changed. It works with all types of tags. Cyclic retrieval produces a virtual rowset, which may or may not correspond to the actual data rows stored on the AVEVA Historian.

In cyclic retrieval, one row is returned for each "cycle boundary." You specify the number of cycles either directly or by means of a time resolution, that is, the spacing of cycle boundaries in time. If you specify a number of cycles, the AVEVA Historian returns that number of rows, evenly spaced in time over the requested period. The cyclic resolution is calculated by dividing the requested time period by the number of cycle boundaries. If you specify a resolution, the number of cycles is calculated by dividing the time period by the resolution.

If no data value is actually stored at a cycle boundary, the last value before the boundary is returned.

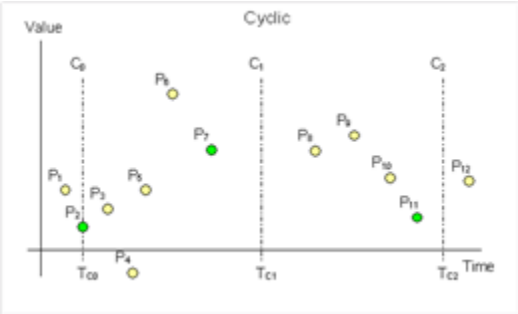
Beginning with AVEVA System Platform 2014 R2 SP1, Historian cyclic storage rules improve the handling of "slow rate change" data tags. Instead of delaying posts to the database if tag values do not arrive in a timely manner, new rules define a cyclic timeout when the database will be updated anyway. That timeout is typically one-half the period for the tag's cycle storage rate or the database server's maximum cyclic storage timeout, whichever is shorter. The time out is controlled by a the system parameter MaxCyclicStorageTimeout. For more information, see System Parameters in the *AVEVA Historian Administration Guide*.

The default retrieval mode is cyclic for retrieval from analog tables, including analog and state summary tables.

Cyclic retrieval is fast and therefore consumes little server resources. However, it may not correctly reflect the stored data because important process values (gaps, spikes, etc.) might fall between cycle boundaries. For an alternative, see Best Fit Retrieval.

Cyclic Retrieval - How It Works

The following illustration shows how values are returned for cyclic retrieval:



Data is retrieved in cyclic mode with a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for three cycle boundaries at T_{C0}, T_{C1}, and T_{C2}. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

- At T_{C0}: P_2, because it falls right on the cycle boundary
- At T_{C1}: P_7, because it is the last point before the cycle boundary
- At T_{C2}: P_{11}, for the same reason

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Cyclic Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in cyclic retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule), for AVEVA Historian 9.0 and above
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Cyclic Retrieval - Query Example

To use the cyclic retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Cyclic'
```

For example, the following query returns data values for the analog tag 'ReactLevel'. If you do not specify a wwCycleCount or wwResolution, the query will return 100 rows (the default).

```
SELECT DateTime, Sec = DATEPART(ss, DateTime), TagName, Value
   FROM History
      WHERE TagName = 'ReactLevel'
         AND DateTime >= '2001-03-13 1:15:00pm'
         AND DateTime <= '2001-03-13 2:15:00pm'
         AND wwRetrievalMode = 'Cyclic'
```

The results are:

DateTime	Sec	TagName	Value
2001-03-13 13:15:00.000	0	ReactLevel	1775.0
2001-03-13 13:15:00.000	36	ReactLevel	1260.0
2001-03-13 13:16:00.000	12	ReactLevel	1650.0
2001-03-13 13:16:00.000	49	ReactLevel	1280.0
2001-03-13 13:17:00.000	25	ReactLevel	1525.0
2001-03-13 13:18:00.000	1	ReactLevel	585.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-03-13 13:18:00.000	38	ReactLevel	1400.0
2001-03-13 13:19:00.000	14	ReactLevel	650.0
2001-03-13 13:19:00.000	50	ReactLevel	2025.0
2001-03-13 13:20:00.000	27	ReactLevel	765.0
2001-03-13 13:21:00.000	3	ReactLevel	2000.0
2001-03-13 13:21:00.000	39	ReactLevel	830.0
2001-03-13 13:22:00.000	16	ReactLevel	1925.0
...			
(100 row(s) affected)			

Cyclic Retrieval - Initial Values

No special handling is done for initial values. The initial value will behave like a normal cycle boundary at the start time. For information on initial values, see Delta Retrieval - Initial Values.

Cyclic Retrieval - Handling NULL Values

No special handling is done for NULL values. They are returned just like any other value.

Daily Retrieval

Daily retrieval is the retrieval of stored data for the given time period based on a resolution of one day, regardless of whether or not the value of the tag(s) has changed. Daily retrieval works similarly to Cyclic retrieval, except that the resolution is always a full local time day (86400000 milliseconds, except across changes between standard and daylight saving time). It works with all types of tags. Daily retrieval produces a virtual rowset, which may or may not correspond to the actual data rows stored on the AVEVA Historian.

In daily retrieval, one row is returned for each cycle. Each cycle represents a day that starts at the local time specified in StartDateTime, and ends when that time next occurs. If the cycle falls on a day that gains or loses an hour due to the start/end of daylight saving time (based on server time), that is accounted for.

If the last cycle ends with less than a full day's data, either due to ending at a time for which data for the tag is not yet stored, or a time that is less than 24 hours after the previous cycle, that will be reflected in the PercentGood. Time for which data is not available will be treated as "not good." For example, if only 12 hours of data are available for the cycle, and the day is 24 hours (i.e. not a day with a daylight-saving related time change),

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the PercentGood will normally be 50, because there is only good data for half of the cycle.

Daily retrieval is only supported with the AnalogSummaryHistory table.

Daily Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in daily retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule), for AVEVA Historian 9.0 and above
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Daily Retrieval - Query Example

To use the daily retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Daily'
```

For example, the following query returns the average for the analog tag 'ReactLevel'. Note that on the 2021-11-06, daylight saving time ends, so the resolution reflects that the cycle is 25 hours rather than 24.

```
SELECT StartDateTime, wwResolution, TagName, Average
   FROM AnalogSummaryHistory
      WHERE TagName = 'ReactLevel'
         AND StartDateTime >= '2021-11-04 08:00:00am'
         AND EndDateTime <= '2001-11-08 08:00:00am'
         AND wwRetrievalMode = 'Daily'
```

The results are:

StartDateTime	wwResolution	TagName	Average
2021-11-04 08:00:00.000	86400000	ReactLevel	1738.0
2021-11-05 08:00:00.000	86400000	ReactLevel	1654.0
2021-11-06 08:00:00.000	90000000	ReactLevel	1804.0
2021-11-07 08:00:00.000	86400000	ReactLevel	1682.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Daily Retrieval - Initial Values

No special handling is done for initial values. The initial value will behave like a normal cycle boundary at the start time. For information on initial values, see Delta Retrieval - Initial Values.

Daily Retrieval - Handling NULL Values

No special handling is done for NULL values. They are returned just like any other value.

Delta Retrieval

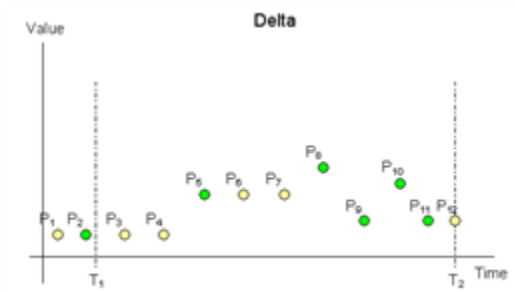
Delta retrieval, or retrieval based on exception, is the retrieval of only the changed values for a tag(s) for the given time interval. That is, duplicate values are not returned. It works with all types of tags.

Delta retrieval always produces a rowset comprised of only rows that are actually stored on the historian; that is, a delta query returns all of the physical rows in history for the specified tags, over the specified period, minus any duplicate values. If there is no actual data point at the start time, the last data point before the start time is returned.

Delta retrieval is the default mode for discrete and string tables and from the History table.

Delta Retrieval - How It Works

The following illustration shows how values are returned for delta retrieval:



Data is retrieved in delta mode with a start time of T_1 and an end time of T_2. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

- P_2, because there is no actual data point at T_1
- P_5, P_8, P_9, P_{10}, and P_{11}, because they represent changed values during the time period

For delta retrieval for replicated summary tags on a tier-2 historian, if a point with doubtful quality is returned as the result of a value selection from an input summary point with a contained gap, the same point can be returned again with good quality if the same value is selected again from the next input summary point that has good quality.

Delta Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in delta retrieval mode. For more information, see the following sections:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Time Deadband (wwTimeDeadband)
- Value Deadband (wwValueDeadband)
- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Delta Retrieval - Query Examples

To use the delta retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Delta'
```

For examples, see the following:

- Delta Retrieval - Query 1
- Delta Retrieval - Query 2
- Delta Retrieval - Query 3
- Delta Retrieval - Query 4

Delta Retrieval - Query 1

As an example of how delta mode works, consider the following query:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwRetrievalMode = 'Delta'
```

This query can be run against the following sample data:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:11	1.0	192
A001	2009-09-12 00:13	1.6	192
A001	2009-09-12 00:16	1.3	192
A001	2009-09-12 00:21	2.0	192
A001	2009-09-12 00:24	1.2	192
A001	2009-09-12 00:27	1.2	192
A001	2009-09-12 00:28	0.0	249

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:32	0.6	192
A001	2009-09-12 00:35	0.0	249
A001	2009-09-12 00:37	1.5	192
A001	2009-09-12 00:43	1.3	192

A graphical representation of the data is as follows:

delta_mode_new_example_query

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	1.3	192
A001	2009-09-12 00:21	2.0	192
A001	2009-09-12 00:24	1.2	192
A001	2009-09-12 00:28	NULL	249
A001	2009-09-12 00:32	0.6	192
A001	2009-09-12 00:35	NULL	249
A001	2009-09-12 00:37	1.5	192

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 2009-09-12 00:20 and 2009-09-12 00:40 (shown on the chart as dark vertical lines) are returned by the query.

Because there is no value that matches the start time, an initial value at 2009-09-12 00:20 is returned in the results based on the value of the preceding data point at 2009-09-12 00:16. Because there is no change in the value at 2009-09-12 00:27 from the value at 2009-09-12 00:24, the data point appears on the chart but does not appear in the results. Similarly, the second 0.0 value at 2009-09-12 00:29 is also excluded from the results.

You can further control the number of rows returned by using the wwTimeDeadband, wwValueDeadband, and wwCycleCount extensions. The use of a cycle count returns the first number of rows within the time range of the query. For more information, see -old-Using wwResolution, wwCycleCount, and wwRetrievalMode in the Same Query.

Also, the use of a time deadband and/or value deadband with delta retrieval produces differing results. For more information, see Time Deadband (wwTimeDeadband) and Value Deadband (wwValueDeadband).

Delta Retrieval - Query 2

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeSec','SysTimeMin')
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Delta'
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:01.000	SysTimeSec	1
2001-12-09 11:35:02.000	SysTimeSec	2
2001-12-09 11:35:03.000	SysTimeSec	3
2001-12-09 11:35:04.000	SysTimeSec	4
…		
2001-12-09 11:35:58.000	SysTimeSec	58
2001-12-09 11:35:59.000	SysTimeSec	59
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:00.000	SysTimeMin	36

Delta Retrieval - Query 3

```
SELECT * FROM OpenQuery(INSQL,'SELECT DateTime, Value, Quality, QualityDetail
   FROM AnalogHistory
      WHERE TagName = "SysTimeSec"
         AND wwRetrievalMode = "Delta"
         AND Value = 10
         AND DateTime >="2001-07-27 03:00:00.000"
         AND DateTime <="2001-07-27 03:05:00.000"
      ')
```

The results are:

DateTime	Value	Quality	QualityDetail
2001-07-27 03:00:10.000	10	0	192
2001-07-27 03:01:10.000	10	0	192
2001-07-27 03:02:10.000	10	0	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-07-27 03:03:10.000	10	0	192
2001-07-27 03:04:10.000	10	0	192

Delta Retrieval - Query 4

For a delta query, if both a wwCycleCount and a Value comparison are specified, the query will return the first number of rows (if available) that meet the value indicated.

```
SELECT * FROM OpenQuery(INSQL,'SELECT DateTime, Value, Quality, QualityDetail
   FROM AnalogHistory
      WHERE TagName = "SysTimeSec"
         AND wwRetrievalMode = "Delta"
         AND Value = 20
         AND wwCycleCount = 10
         AND DateTime >="2001-07-27 03:00:00.000"
         AND DateTime <="2001-07-27 03:20:00.000"
   ')
```

The results are:

DateTime	Value	Quality	QualityDetail
2001-07-27 03:00:20.000	20	0	192
2001-07-27 03:01:20.000	20	0	192
2001-07-27 03:02:20.000	20	0	192
2001-07-27 03:03:20.000	20	0	192
2001-07-27 03:04:20.000	20	0	192
2001-07-27 03:05:20.000	20	0	192
2001-07-27 03:06:20.000	20	0	192
2001-07-27 03:07:20.000	20	0	192
2001-07-27 03:08:20.000	20	0	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-07-27 03:09:20.000	20	0	192

Delta Retrieval - Initial Values

Initial values are special values that can be returned from queries that lie exactly on the query start time, even if there is not a data point that specifically matches the specified start time. If there is not a value exactly on the query start time, the last point before the start time will be returned with its DateTime set to the query start time and its Quality set to 133. If no value exists at or prior to the query start time, a NULL value will be returned at start time with QualityDetail set to 65536, OPCQuality set to 0, and Quality set to 1.

Querying the start time in exclusive form with the > operator indicates that a value should not be returned for the query start time if one does not exist. Querying the start time in inclusive form with the >= operator indicates that an initial value should be returned.

For example, the following exclusive query statement does not return an initial value for 2009-01-01 02:00:00.

```
   DateTime > '2009-01-01 02:00:00'
```

However, the following inclusive query statement does return an initial value for 2009-01-01 02:00:00.

```
   DateTime >= '2009-01-01 02:00:00'
```

No special final value is returned.

Delta Retrieval - Handling NULL Values

The initial NULL value after a non-NULL is always returned. Multiple NULL values are suppressed. The first non-NULL after a NULL is always returned even if it is the same as the previous non-NULL value.

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwRetrievalMode = 'Delta'
```

This query can be run against the following sample data:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:24	0.0	249
A001	2009-09-12 00:27	0.0	249
A001	2009-09-12 00:28	0.5	192
A001	2009-09-12 00:31	0.0	249
A001	2009-09-12 00:33	0.0	24
A001	2009-09-12 00:35	0.0	24
A001	2009-09-12 00:36	0.5	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The following is a graphical representation of the data:

delta_mode_new_example_NULL_handling

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	0.8	192
A001	2009-09-12 00:24	NULL	249
A001	2009-09-12 00:28	0.5	192
A001	2009-09-12 00:31	NULL	249
A001	2009-09-12 00:36	0.5	192

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query.

Because there is no value that matches the start time, an initial value at 00:20 is returned in the results based on the value of the preceding data point at 00:16. Because there is no change in the value at 00:27 from the value at 00:24, the data point appears on the chart but does not appear in the results. Similarly, the two 0.0 values at 00:33 and 00:35 are also excluded from the results. However, the non-NULL value at 00:36 is returned, even though it is the same as the value at 00:28, because it represents a delta from the preceding (NULL) value at 00:35.

Full Retrieval

In full retrieval mode, all stored data points are returned, regardless of whether a value or quality has changed since the last value. This mode allows the same value and quality pair (or NULL value) to be returned consecutively with their actual timestamps. It works with all types of tags.

By using full retrieval in conjunction with storage without filtering (that is, no delta or cyclic storage mode is applied at the historian), you can retrieve all values that originated from the plant floor data source or from another application.

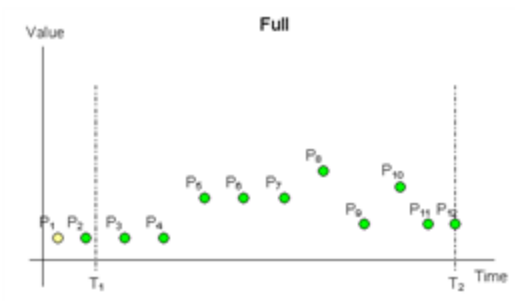
Full retrieval best represents the process measurements recorded by the AVEVA Historian. However, it creates a higher load for the server, the network and the client system because a very large number of records may be returned for longer time periods.

For full retrieval for replicated summary tags on a tier-2 historian, if a point with doubtful quality is returned as the result of a value selection from an input summary point with a contained gap, the same point can be returned again with good quality if the same value is selected again from the next input summary point that has good quality.

Full Retrieval - How It Works

The following illustration shows how values are returned for full retrieval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Data is retrieved in full mode with a start time of T_1 and an end time of T_2. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

- P_2, because there is no actual data point at T_1
- P_3 through P_{12}, because they represent stored data points during the time period

Full Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in full retrieval mode. For more information, see the following sections:

- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Full Retrieval - Query Example

For example, the following query uses full retrieval mode:

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeSec','SysTimeMin')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Full'
```

Full Retrieval - Initial Values

Full retrieval mode handles initial values the same way as delta mode. For more information on initial values, see Delta Retrieval - Initial Values.

Interpolated Retrieval

Interpolated retrieval works like cyclic retrieval, except that interpolated values are returned if there is no actual data point stored at the cycle boundary.

This retrieval mode is useful if you want to retrieve cyclic data for slow-changing tags. For a trend, interpolated

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

retrieval results in a smoother curve instead of a "stair-stepped" curve. This mode is also useful if you have a slow-changing tag and a fast-changing tag and want to retrieve data for both. Finally, some advanced applications require more evenly spaced values than would be returned if interpolation was not applied.

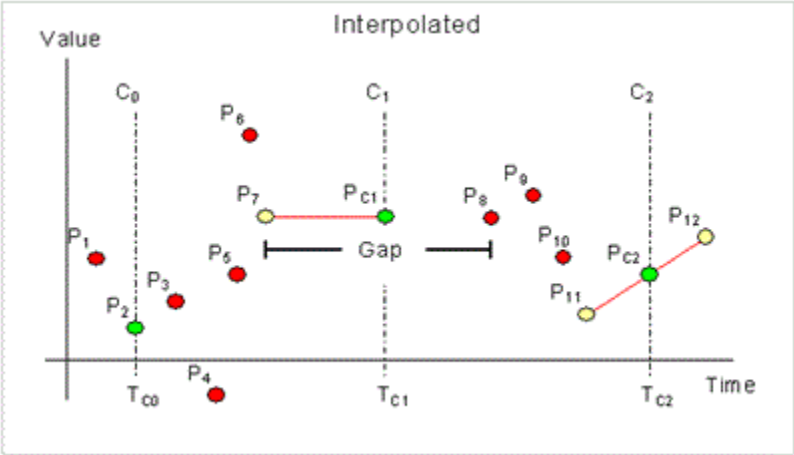
By default, interpolated retrieval uses the interpolation setting specified for the tag in the AVEVA Historian. This means that if a tag is set to use stair-step interpolation, interpolated retrieval gives the same results as cyclic retrieval.

Interpolation is only applied to analog tags. If you retrieve data for other types of tags, stair-step interpolation is used, and the results are the same as for cyclic retrieval.

Interpolated retrieval is a bit slower than cyclic retrieval. It shares the limitations of cyclic retrieval in that it may not accurately represent the stored process data.

Interpolated Retrieval - How It Works

The following illustration shows how the values for an analog tag that is configured for linear interpolation are returned when using interpolated retrieval.



Data is retrieved in interpolated mode with a start time of TC_0 and an end time of TC_2. The resolution has been set in such a way that the historian returns data for three cycle boundaries at TC_0, TC_1, and TC_2. P_1 to P_{12} represent actual data points stored on the historian. Of these points, eleven represent normal analog values, and one, P_7, represents a NULL value due to an I/O Server disconnect, which causes a gap in the data between P_7 and P_8.

The green points (P_2, PC_1, PC_2) are returned. The yellow points (P_7, P_{11}, P_{12}) are used to interpolate the returned value for each cycle. The red points are considered, but not used in calculating the points to return.

Because P_2 is located exactly at the query start time, it is returned at that time without the need for any interpolation. At the following cycle boundary, point PC1 is returned, which is the NULL value represented by P_7 shifted forward to time TC_1. At the last cycle boundary, point PC_2 is returned, which has been interpolated using points P_{11} and P_{12}.

Interpolated Retrieval - Query Examples

To use the interpolated mode, set the following parameter in your query.

```
wwRetrievalMode = 'Interpolated'
```

For examples, see the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Interpolated Retrieval - Query 1
- Interpolated Retrieval - Query 2
- Interpolated Retrieval - Query 3

Interpolated Retrieval - Query 1

Two analog tags and a discrete tag are retrieved from the History table, using linear interpolation. The start and end times are offset to show interpolation of the SysTimeMin tag. The data points at all cycle boundaries are interpolated for the two analog tags, while the values returned for the discrete tag are stair-stepped.

```
SELECT DateTime, TagName, Value, wwInterpolationType FROM History
   WHERE TagName IN ('SysTimeMin', 'ReactTemp', 'SysPulse')
      AND DateTime >= '2005-04-11 12:02:30'
      AND DateTime <= '2005-04-11 12:06:30'
      AND wwRetrievalMode = 'Interpolated'
      AND wwInterpolationType = 'Linear'
      AND wwResolution = 60000
```

The results are:

DateTime	TagName	Value	wwInterpolationType
2005-04-11 12:02:30.000	SysTimeMin	2.5	LINEAR
2005-04-11 12:02:30.000	ReactTemp	23.2	LINEAR
2005-04-11 12:02:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:03:30.000	SysTimeMin	3.5	LINEAR
2005-04-11 12:03:30.000	ReactTemp	139.96753	LINEAR
2005-04-11 12:03:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:04:30.000	SysTimeMin	4.5	LINEAR
2005-04-11 12:04:30.000	ReactTemp	111.49636	LINEAR
2005-04-11 12:04:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:05:30.000	SysTimeMin	5.5	LINEAR

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2005-04-11 12:05:30.000	ReactTemp	17.00238	LINEAR
2005-04-11 12:05:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:06:30.000	SysTimeMin	6.5	LINEAR
2005-04-11 12:06:30.000	ReactTemp	168.99531	LINEAR
2005-04-11 12:06:30.000	SysPulse	1.0	STAIRSTEP

Interpolated Retrieval - Query 2

If you omit the interpolation type in the query, the historian determines which interpolation type to use for an analog tag based on the value of the InterpolationType column in the AnalogTag table, in conjunction with the InterpolationTypeInteger and InterpolationTypeReal system parameters.

In the following query both analog tags are set to use the system default through the AnalogTag table, while the InterpolationTypeInteger and InterpolationTypeReal system parameters are set to 0 and 1, respectively. Because SysTimeMin is defined as a 2-byte integer and ReactTemp is defined as a real we see that only rows for ReactTemp are interpolated.

```
SELECT DateTime, TagName, Value, wwInterpolationType FROM History
   WHERE TagName IN ('SysTimeMin', 'ReactTemp', 'SysPulse')
      AND DateTime >= '2005-04-11 12:02:30'
      AND DateTime <= '2005-04-11 12:06:30'
      AND wwRetrievalMode = 'Interpolated'
      AND wwResolution = 60000
```

The results are:

DateTime	TagName	Value	wwInterpolationType
2005-04-11 12:02:30.000	SysTimeMin	2.0	STAIRSTEP
2005-04-11 12:02:30.000	ReactTemp	23.2	LINEAR
2005-04-11 12:02:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:03:30.000	SysTimeMin	3.0	STAIRSTEP
2005-04-11 12:03:30.000	ReactTemp	139.96753	LINEAR

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2005-04-11 12:03:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:04:30.000	SysTimeMin	4.0	STAIRSTEP
2005-04-11 12:04:30.000	ReactTemp	111.49636	LINEAR
2005-04-11 12:04:30.000	SysPulse	1.0	STAIRSTEP
2005-04-11 12:05:30.000	SysTimeMin	5.0	STAIRSTEP
2005-04-11 12:05:30.000	ReactTemp	17.00238	LINEAR
2005-04-11 12:05:30.000	SysPulse	0.0	STAIRSTEP
2005-04-11 12:06:30.000	SysTimeMin	6.0	STAIRSTEP
2005-04-11 12:06:30.000	ReactTemp	168.99531	LINEAR
2005-04-11 12:06:30.000	SysPulse	1.0	STAIRSTEP

Interpolated Retrieval - Query 3

```
SELECT TagName, DateTime, Value, QualityDetail, wwInterpolationType
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwRetrievalMode = 'Interpolated'
         AND wwResolution = '10000'
```

This query can be run against the following sample data:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:09	0.2	192
A001	2009-09-12 00:15	1.3	192
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:22	0.6	249

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

A001	2009-09-12 00:26	0.9	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:33	1.1	192
A001	2009-09-12 00:35	1.6	192
A001	2009-09-12 00:38	0.5	192
A001	2009-09-12 00:42	0.8	192

The following is a graphical representation of the data:

interpolated_mode_new_second_example

The results are:

Tagname	DateTime	Value	QualityDetail	wwInterpolationType
A001	2009-09-12 00:20	0.8	192	STAIRSTEP
A001	2009-09-12 00:30	NULL	249	STAIRSTEP
A001	2009-09-12 00:40	0.5	192	LINEAR

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query.

Because there is no value that matches the start time, an initial value at 00:20 is returned in the results based on the preceding data point at 00:17 because the following data point at 00:22 is NULL. Because a NULL value precedes the 00:30 cycle boundary at 00:29, the NULL is returned at the cycle boundary. The value at 00:40 is an interpolation of the data points at 00:38 and 00:42.

Interpolated Retrieval - Initial and Final Values

A value is returned at the start time and end time of the query using interpolation of the surrounding points.

Interpolated Retrieval - Handling NULL Values

When a NULL value precedes a cycle boundary, that NULL will be returned at the cycle boundary.

If a valid value precedes a cycle boundary, but is followed by a NULL value after the cycle boundary, no interpolation will be used and wwInterpolationType will be set to STAIRSTEP for that value.

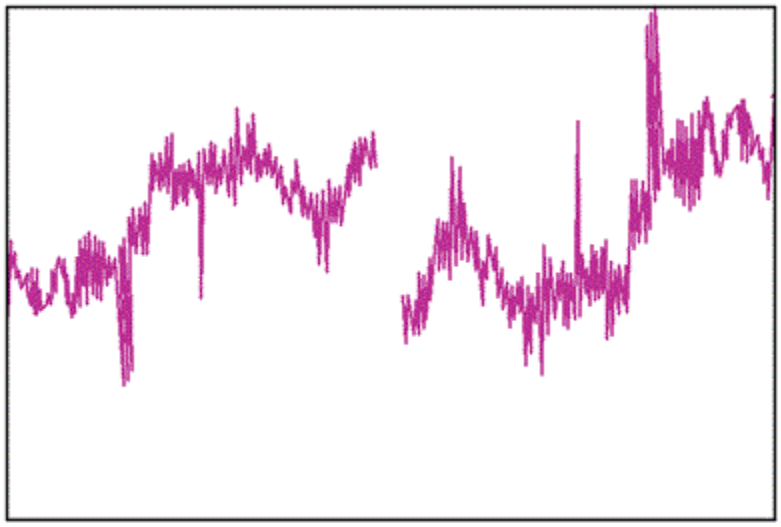
Best Fit Retrieval

For the " best fit" retrieval mode, the total time for the query is divided into even sub-periods, and then up to five values are returned for each sub-period:

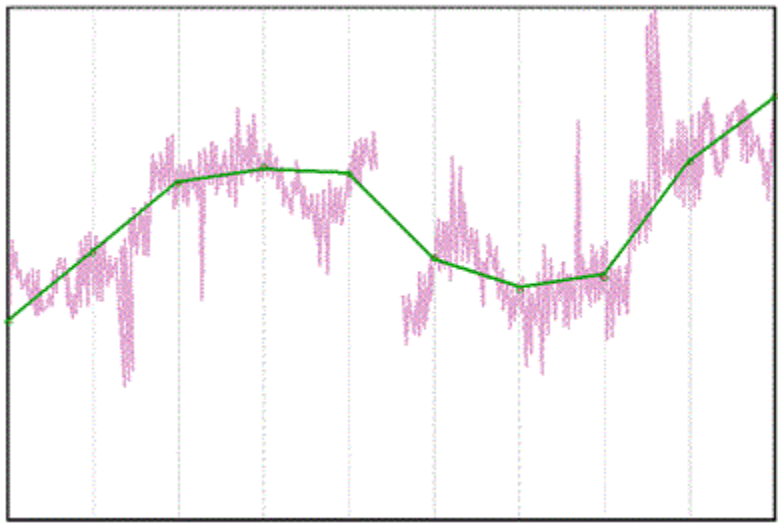
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- First value in the period
- Last value in the period
- Minimum value in the period, with its actual time
- Maximum value in the period, with its actual time
- The first "exception" in the period (non-Good quality)

"Best fit" retrieval allows for a compromise between delta retrieval and cyclic retrieval. For example, delta retrieval can accurately represent a process over a long period of time, as shown in the following trend. However, to achieve this representation, a large number of data values must be returned.

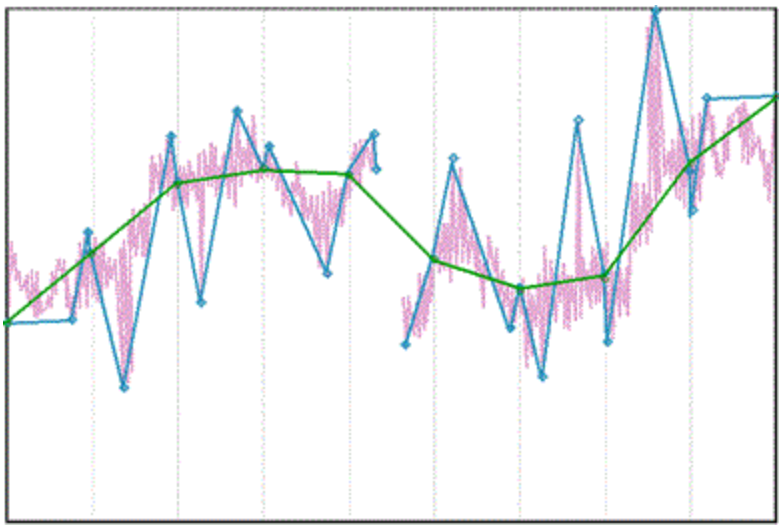


If cyclic retrieval is used to retrieve the data, the retrieval is much more efficient, because fewer values are returned. However, the representation is not as accurate, as the following trend shows.



"Best fit" retrieval allows for faster retrieval, as typically achieved by using cyclic retrieval, plus the better representation typically achieved by using delta retrieval. This is shown in the following trend.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



For example, for one week of five-second data, 120,960 values would be returned for delta retrieval, versus around 300 values for best-fit retrieval.

Best-fit retrieval uses retrieval cycles, but it is not a true cyclic mode. Apart from the initial value, it only returns actual delta points. For example, if one point is both the first value and the minimum value in a cycle, it is returned only one time. In a cycle where a tag has no points, nothing is returned.

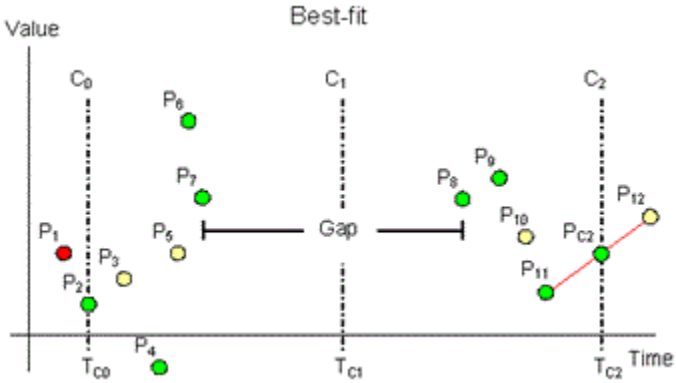
As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, the number of values returned is likely to be more than one per cycle.

All points are returned in chronological order. If multiple points are to be returned for a particular timestamp, then those points are returned in the order in which the corresponding tags were specified in the query.

The best-fit algorithm is only applied to analog and analog summary tags. For all other tags, delta results are returned.

Best Fit Retrieval - How It Works

The following illustration shows how the best-fit algorithm selects points for an analog tag.



Data is retrieved in best-fit mode with a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}. P_1 to P_{12} represent actual data points stored on the historian. Of these points, eleven represent normal analog values, and one, P_7, represents a NULL value due to an I/O Server disconnect, which causes a gap in the data between P_7 and P_8.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Because P_2 is located exactly at the start time, no initial value needs to be interpolated at the start time. Therefore, point P_1 is not considered at all. All other points are considered, but only the points indicated by green markers on the graph are returned.

From the first cycle, four points are returned:

- P_2 as the initial value of the query, as well as the first value in the cycle
- P_4 as the minimum value in the cycle
- P_6 as both the maximum value and the last value in the cycle
- P_7 as the first (and only) occurring exception in the cycle

From the second cycle, three points are returned:

- P_8 as the first value in the cycle
- P_9 as the maximum value in the cycle
- P_{11} as both the minimum value and the last value in the cycle
- As no exception occurs in the second cycle, none is returned.

Because the tag does not have a point exactly at the query end time, where an incomplete third cycle starts, the end value P_{C2} is interpolated between P_{11} and P_{12}, assuming that linear interpolation is used.

Best Fit Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in best-fit retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Interpolation Type (wwInterpolationType)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)
- Expressions (wwExpression)

Best Fit Retrieval - Query Example

To use the best fit retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'BestFit'
```

For example, an analog tag is retrieved over a five-minute period using the best-fit retrieval mode. The wwResolution parameter is set to 60000, thus specifying five 1-minute cycles. Within each cycle, the retrieval sub-system returns the first, minimum, maximum, and last data points. There are no exception (NULL) points in the time period. Notice how the points at the query start time and at the query end time are interpolated, while all other points are actual delta points.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 1), Value) AS Value, wwInterpolationType AS
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
IT FROM History
      WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 12:15:00'
      AND DateTime <= '2005-04-11 12:20:00'
      AND wwRetrievalMode = 'BestFit'
      AND wwResolution = 60000
```

The results are:

	DateTime	TagName	Value	IT
(initial, first, min)	2005-04-11 12:15:00.000	ReactTemp	40.7	LINEAR
(max in interval 1)	2005-04-11 12:15:38.793	ReactTemp	196.0	STAIRSTEP
(last in interval 1)	2005-04-11 12:15:58.810	ReactTemp	159.2	STAIRSTEP
(first, max in interval 2)	2005-04-11 12:16:00.013	ReactTemp	156.9	STAIRSTEP
(last, min in interval 2)	2005-04-11 12:16:58.857	ReactTemp	16.3	STAIRSTEP
(first, min in interval 3)	2005-04-11 12:17:00.060	ReactTemp	14.0	STAIRSTEP
(last, max in interval 3)	2005-04-11 12:17:58.793	ReactTemp	151.0	STAIRSTEP
(first in interval 4)	2005-04-11 12:18:00.107	ReactTemp	156.0	STAIRSTEP
(max in interval 4)	2005-04-11 12:18:10.057	ReactTemp	196.0	STAIRSTEP
(last, min in interval 4)	2005-04-11 12:18:58.837	ReactTemp	106.3	STAIRSTEP
(first, max in interval 5)	2005-04-11 12:19:00.040	ReactTemp	104.0	STAIRSTEP
(min in interval 5)	2005-04-11 12:19:31.320	ReactTemp	14.0	STAIRSTEP
(last in interval 5)	2005-04-11 12:19:58.773	ReactTemp	26.0	STAIRSTEP
(end bounding value)	2005-04-11 12:20:00.000	ReactTemp	30.7	LINEAR

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Best Fit Retrieval - Initial and Final Values

A point will be returned at the query start time and the query end time for each tag queried, if a point exists for that tag at or after the end time of the query. The values of the initial and final points will be determined by interpolating the points preceding and following the query start or query end time.

Standard interpolation rules will be used to return the initial and final values. For more information, see Interpolated Retrieval.

Best Fit Retrieval - Handling NULL Values

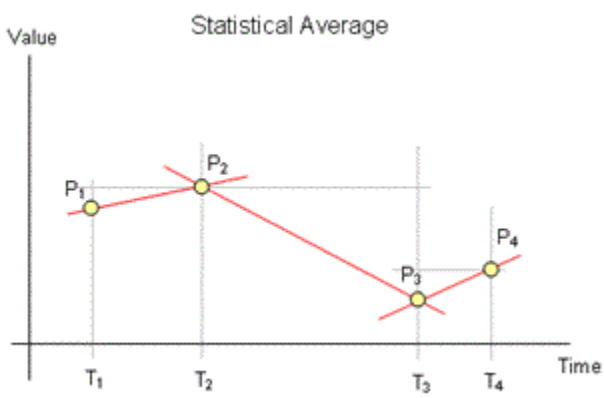
When any of the four good points are returned from a cycle that contains gaps or from an incomplete cycle with the query end time located inside of the calculation cycle the quality detail of each of the non-null points returned is modified to alert the user to this fact. This is done by performing a logical OR operation of the value 4096, which means partial cycle, onto the existing quality detail. (This is the delta point equivalent to the use of PercentGood for cyclic.)

Average Retrieval

For the time-weighted average (in short: "average") retrieval mode, a time-weighted average algorithm is used to calculate the value to be returned for each retrieval cycle.

For a statistical average, the actual data values are used to calculate the average. The average is the sum of the data values divided by the number of data values. For the following data values, the statistical average is computed as:

$(P_1 + P_2 + P_3 + P_4) / 4) = \text{Average}$



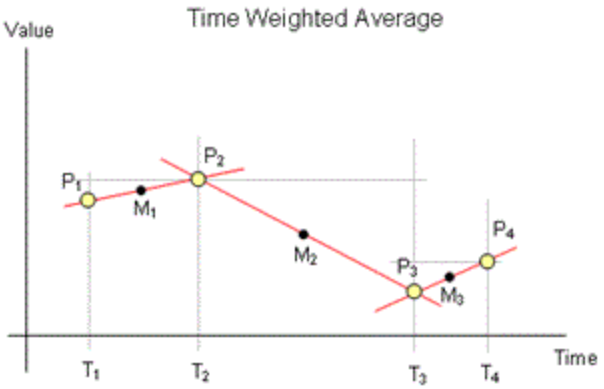
For a time-weighted average, values are multiplied by the time difference between the points to determine the time-weighted value. Therefore, the longer a tag had a particular value, the more weight that value holds in the overall average. The overall average is determined by adding all of the time-weighted values and then dividing that number by the total amount of time.

Which values are weighted depends on the interpolation setting of the tag. For a tag that uses linear interpolation, the midpoints between values are weighted. For a tag that uses stair-step interpolation, the earlier of two values is weighted.

For the following data values of a tag that uses linear interpolation, the time-weighted average is computed as:

$(((P_1 + P_2) / 2) \times (T_2 - T_1)) + (((P_2 + P_3) / 2) \times (T_3 - T_2)) + (((P_3 + P_4) / 2) \times (T_4 - T_3)) / (T_4 - T_1) = \text{Average}$

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



If the same tag uses stair-step interpolation, the time-weighted average is:

$((P_1 \times (T_2 - T_1)) + (P_2 \times (T_3 - T_2)) + (P_3 \times (T_4 - T_3))) / (T_4 - T_1) = \text{Average}$

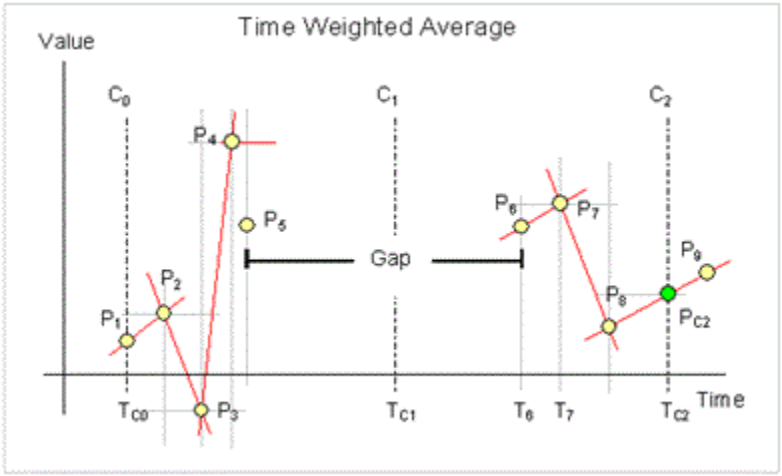
The SQL Server AVG aggregate is a simple statistical average. Using the average retrieval mode with a cycle count of 1 returns data much faster than the AVG aggregate, and usually more accurately due to the time weighting. The Event subsystem also returns a simple statistical average.

Average retrieval returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

The time-weighted average algorithm is only applied to analog and analog summary tags. If you use average retrieval with other tags, the results are the same as when using regular cyclic retrieval.

Average Retrieval - How It Works

The following illustration shows how the time-weighted average is calculated for an analog tag that uses linear interpolation.



Data is retrieved in average mode with a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}. P_1 to P_9 represent actual data points stored on the historian. Of these points, eight represent normal analog values, and one, P_5, represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_5 and P_6. Assume that the query calls for timestamping at the end of the cycle.

Results are calculated as follows:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- The "initial value" returned at the query start time (T_{C0}) is the time-weighted average of the points in the last cycle preceding T_{C0}.
- The value returned at T_{C1} is the time-weighted average of the points in the cycle starting at T_{C0}.
- The value returned at the query end time (T_{C2}) is the time-weighted average of the points in the cycle starting at T_{C1}.

To understand how the time-weighted average is calculated, observe the last cycle as an example. First, the area under the curve must be calculated. This curve is indicated by the red line through P_6, P_7, P_8 and P_{C2}, where P_{C2} represents the interpolated value at time T_{C2} using points P_8 and P_9. The data gap caused by the I/O Server disconnect does not contribute anything to this area. If a quality rule of "good" has been specified, then points with doubtful quality will not contribute anything to the area, either.

To understand how the area is calculated, consider points P_6 and P_7. The area contribution between these two points is calculated by multiplying the arithmetic average of value P_6 and value P_7 by the time difference between the two points. The formula is:

$((P_6 + P_7) / 2) \times (T_7 - T_6)$

When the area for the whole cycle has been calculated, the time-weighted average is calculated by dividing that area by the cycle time, less any periods within the cycle that did not contribute anything to the area calculation. The result is returned at the cycle end time.

If you take a closer look at points P_4 and P_5 in the example, you can see that the red line through point P_4 is parallel to the x-axis. This is because P_5 represents a NULL, which cannot be used to calculate an arithmetic average. Instead, the value P_4 is used for the whole time period between points P_4 and P_5.

The area calculation is signed. If the arithmetic average between two points is negative, then the contribution to the area is negative.

Average Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in average retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Interpolation Type (wwInterpolationType)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Average Retrieval - Query Examples

To use the average mode, set the following parameter in your query.

```
wwRetrievalMode = 'Average'
```

For examples, see the following:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- Average Retrieval - Query 1
- Average Retrieval - Query 2

For an additional example, see Querying Aggregate Data in Different Ways.

Average Retrieval - Query 1

The time-weighted average is computed for each of five 1-minute long cycles.

Note that the wwTimeStampRule parameter is set to "Start" in the query. This means that the value stamped at 11:18:00.000 represents the average for the interval 11:18 to 11:19, the value stamped at 11:19:00.000 represents the average for the interval 11:19 to 11:20, and so on. If no timestamp rule is specified in the query, then the default setting in the TimeStampRule system parameter is used.

In the first cycle there are no points, so a NULL is returned. In the second cycle value points are found covering 77.72 percent of the time as returned in PercentGood. This means that the returned average is calculated based on 77.72 percent of the cycle time. Because the same OPCQuality is not found for all the points in the cycle, OPCQuality is set to Doubtful. In the remaining three cycles, only good points occur, all with an OPCQuality of 192.

Because no quality rule is specified in the query using the wwQualityRule parameter, the query uses the default as specified by the QualityRule system parameter. If a quality rule of Extended is specified, any point stored with doubtful OPCQuality will be used to calculate the average, and the point time will therefore be included in the calculation of PercentGood.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value, OPCQuality, PercentGood
FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:18:00'
      AND DateTime < '2005-04-11 11:23:00'
      AND wwRetrievalMode = 'Average'
      AND wwCycleCount = 5
      AND wwTimeStampRule = 'Start'
```

The results are:

	DateTime	TagName	Value	OPCQuality	PercentGood
(cycle 1)	2005-04-11 11:18:00.000	ReactTemp	NULL	0	0.0
(cycle 2)	2005-04-11 11:19:00.000	ReactTemp	70.00	64	77.72
(cycle 3)	2005-04-11 11:20:00.000	ReactTemp	153.99	192	100.0
(cycle 4)	2005-04-11 11:21:00.000	ReactTemp	34.31	192	100.0
(cycle 5)	2005-04-11 11:22:00.000	ReactTemp	134.75	192	100.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Average Retrieval - Query 2

This query demonstrates the use of the average retrieval mode in a wide query. Time-weighted average values are returned for the analog tags ReactTemp and ReactLevel, while regular cyclic points are returned for the discrete tag, WaterValve.

```
SELECT * FROM OpenQuery(INSQL,
   'SELECT DateTime, ReactTemp, ReactLevel, WaterValve FROM WideHistory
      WHERE DateTime >= "2004-06-07 08:00"
         AND DateTime < "2004-06-07 08:05"
         AND wwRetrievalMode = "Average"
         AND wwCycleCount = 5
   ')
```

The results are:

DateTime	ReactTemp	ReactLevel	WaterValve
2004-06-07 08:00:00.000	47.71621	1676.69716	1
2004-06-07 08:01:00.000	157.28076	1370.88097	0
2004-06-07 08:02:00.000	41.33734	797.67296	1
2004-06-07 08:03:00.000	122.99525	1921.66771	0
2004-06-07 08:04:00.000	105.28866	606.40488	1

Average Retrieval - Initial and Final Values

If wwTimeStampRule = END, the initial value is calculated by performing an average calculation on the cycle leading up to the query start time. No special handling is done for the final value.

If wwTimeStampRule = START, the final value is calculated by performing an average calculation on the cycle following the query end time. No special handling is done for the initial value.

Average Retrieval - Handling NULL Values

Gaps introduced by NULL values are not included in the average calculations. The average only considers the time ranges with good values. TimeGood indicates the total time per cycle that the tags value was good.

Minimum Retrieval

The minimum value retrieval mode returns the minimum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. NULL is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

returned if the cycle contains one or more NULL values.

As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, minimum retrieval is not a true cyclic mode. Apart from the initial value, all points returned are delta points.

Minimum retrieval only works with analog tags. For all other tags, normal delta results are returned.

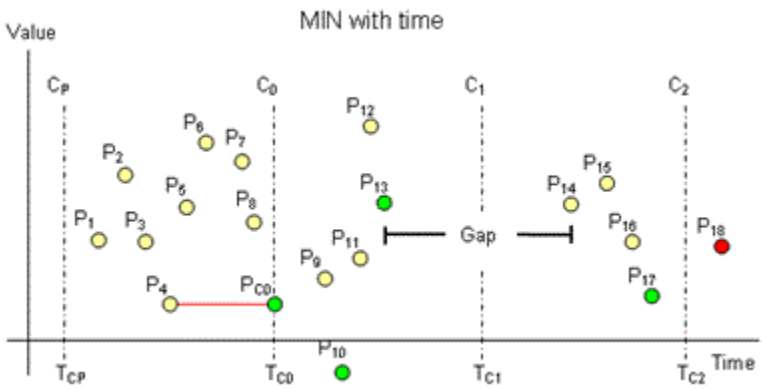
All returned values are in chronological order. If multiple points are to be returned for a particular timestamp, they are returned in the order in which the tags were specified in the query. If the minimum value occurs several times in a cycle, the minimum value with the earliest timestamp is returned.

The minimum retrieval mode must use the "<=" operator for the ending date/time.

Using the minimum retrieval mode with a cycle count of 1 returns the same results as the SQL Server MIN aggregate; however, the data is returned much faster.

Minimum Retrieval - How It Works

The following illustration shows how the minimum value is selected for an analog tag.



This example has a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1}, a "phantom" cycle starting at T_{CP}, and an incomplete cycle starting at T_{C2}. The phantom cycle has the same duration as the first cycle in the query period, extending back in time from the query start time.

For the queried tag, a total of 18 points are found throughout the cycles, represented by the markers P_1 through P_{18}. Of these points, 17 represent normal analog values. The point P_{13} represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_{13} and P_{14}.

The minimum value for the "phantom" cycle starting at T_{CP} is returned as the initial value at T_{C0}. Point P_{18} is not considered at all because it is outside of the query time frame. All other points are considered, but only the points indicated by green markers on the graph are returned (P_{10}, P_{13}, and P_{17}).

In total, four points are returned:

- P_4 as the minimum value of the "phantom" cycle and the initial point
- P_{10} as the minimum value in the first cycle
- P_{13} as the first and only exception occurring in the first cycle
- P_{17} as the minimum value in the second cycle

No points are returned for the incomplete third cycle starting at the query end time, because the tag does not have a point exactly at that time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If the minimum value of the first cycle is located exactly at the query start time, both this value and the minimum value of the phantom cycle are returned.

Minimum Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in minimum retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Minimum Retrieval - Query Examples

To use the minimum mode, set the following parameter in your query:

```
wwRetrievalMode = 'Min'
```

or

```
wwRetrievalMode = 'Minimum'
```

For examples, see the following:

- Minimum Retrieval - Query 1
- Minimum Retrieval - Query 2
- Minimum Retrieval - Query 3

Minimum Retrieval - Query 1

In this example, an analog tag is retrieved over a five minute period, using the minimum retrieval mode. Because the wwResolution parameter is set to 60000, each cycle is exactly one minute long. The minimum data value is returned from each of these cycles.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
   AND DateTime >= '2005-04-11 11:21:00'
   AND DateTime <= '2005-04-11 11:26:00'
   AND wwRetrievalMode = 'Minimum'
   AND wwResolution = 60000
```

The initial value at the query start time is the minimum value found in the phantom cycle before the start time of the query.

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:21:00.000	ReactTemp	104.00
(cycle 1)	2005-04-11 11:21:30.837	ReactTemp	14.00
(cycle 2)	2005-04-11 11:22:00.897	ReactTemp	36.00
(cycle 3)	2005-04-11 11:23:59.567	ReactTemp	18.60
(cycle 4)	2005-04-11 11:24:02.083	ReactTemp	14.00
(cycle 5)	2005-04-11 11:25:59.550	ReactTemp	108.60

Minimum Retrieval - Query 2

In this example, the minimum retrieval mode is used in a manner equivalent to using the SQL Server MIN aggregate. Note that the cycle producing the result is the five-minute phantom cycle just before the query start time.

```
SELECT TOP 1 DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:31:00'
      AND DateTime <= '2005-04-11 11:31:00'
      AND wwRetrievalMode = 'Minimum'
      AND wwResolution = 300000
```

The results are:

	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:31:00.000	ReactTemp	14.00

Minimum Retrieval - Query 3

This example shows how the minimum retrieval mode marks the QualityDetail column to indicate that a minimum value is returned based on an incomplete cycle. In this case, an incomplete cycle is a cycle that either contained periods with no values stored or a cycle that was cut short because the query end time was located inside the cycle. All values returned for the QualityDetail column are documented in the QualityMap table in the Runtime database.

```
SELECT DateTime, TagName, Value, QualityDetail FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2005-04-11 11:18:50'
      AND DateTime <= '2005-04-11 11:20:50'
      AND wwRetrievalMode = 'Minimum'
      AND wwResolution = 60000
```

The results are:

	DateTime	TagName	Value	QualityDetail
(phantom cycle)	2005-04-11 11:18:50.000	SysTimeSec	NULL	65536
(cycle 1)	2005-04-11 11:19:13.000	SysTimeSec	13.0	4140
(cycle 2)	2005-04-11 11:20:00.000	SysTimeSec	0.0	192
(cycle 3)	2005-04-11 11:20:50.000	SysTimeSec	50.0	4288

Minimum Retrieval - Initial and Final Values

For analog tags, the minimum value of the tag in the cycle leading up to the query start time is returned with its timestamp changed to the query start time. If there is no point exactly at the "phantom" cycle start time, the point leading up to the phantom cycle is also considered for the minimum calculation.(No adjustments are made to the quality of the initial point even though the timestamp may have been altered.) Apart from the initial value, all points returned are delta points. (For more information on initial values, see Delta Retrieval - Initial Values.)

If a point occurs exactly on the query end time, that point will be returned with the partial cycle bit, 4096, set in quality detail. If there is more than one such point, only the first point will be returned.

Minimum Retrieval - Handling NULL Values and Incomplete Cycles

The first NULL value in a cycle is returned.

When a minimum value is returned from a cycle that contains gaps (including a gap extended from the previous cycle) or from an incomplete cycle with the query end time located inside of the calculation cycle, the point's quality detail is modified to flag this. This is done by performing a logical OR operation of the value 4096, which indicates a partial cycle, onto the existing quality detail.

As an example of how minimum retrieval mode handles NULLs, consider the following query:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Minimum'
```

This query can be run against the following sample data:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:09	0.2	192
A001	2009-09-12 00:15	1.3	192
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:22	0.5	192
A001	2009-09-12 00:26	0.9	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:33	1.1	192
A001	2009-09-12 00:35	1.6	192
A001	2009-09-12 00:38	0.5	192
A001	2009-09-12 00:42	0.8	192

The following is a graphical representation of the data:

minimum_mode_new_example

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	0.2	192
A001	2009-09-12 00:22	0.5	4288
A001	2009-09-12 00:28	NULL	249
A001	2009-09-12 00:38	0.5	4288

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query. The resolution is set at 10,000 milliseconds.

Because there is no value that matches the start time, an initial value at 00:20 is returned based on the minimum value of the preceding cycle, which is the data point at 00:09. In the two subsequent cycles, the minimum values are at 00:22 and 00:38. The quality for these two values is set to 4288 (4096 + 192). The remaining data points are excluded because they are not minimums. In addition, the first NULL at 00:28 is included, but the second NULL (at 00:29) is not.

Maximum Retrieval

The maximum value retrieval mode returns the maximum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. NULL is

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

returned if the cycle contains one or more NULL values.

As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, maximum retrieval is not a true cyclic mode. Apart from the initial value, all points returned are delta points.

Maximum retrieval only works with analog tags. For all other tags, normal delta results are returned.

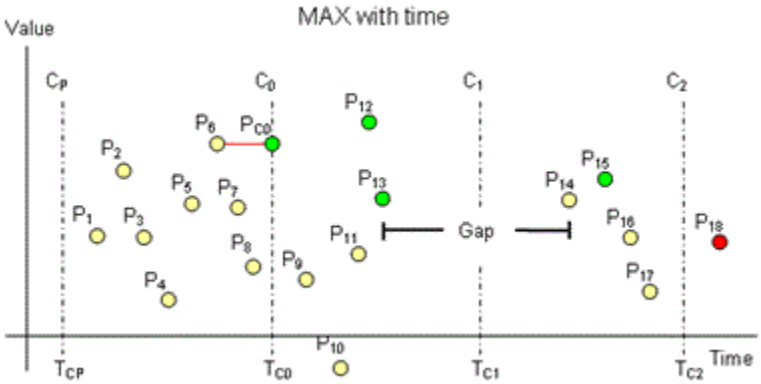
All returned values are in chronological order. If multiple points are to be returned for a particular timestamp, they are returned in the order in which the tags were specified in the query. If the maximum value occurs several times in a cycle, the maximum value with the earliest timestamp is returned.

The maximum retrieval mode must use the "<=" operator for the ending date/time.

Using the maximum retrieval mode with a cycle count of 1 returns the same results as the SQL Server MAX aggregate; however, the data is returned much faster.

Maximum Retrieval - How It Works

The following illustration shows how the maximum value is selected for an analog tag.



This example has a start time of T_{C0} and an end time of T_{C2}. The resolution has been set in such a way that the historian returns data for two complete cycles starting at T_{C0} and T_{C1}, a "phantom" cycle starting at T_{CP}, and an incomplete cycle starting at T_{C2}. The phantom cycle has the same duration as the first cycle in the query period, extending back in time from the query start time.

For the queried tag, a total of 18 points are found throughout the cycles, represented by the markers P_1 through P_{18}. Of these points, 17 represent normal analog values. The point P_{13} represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_{13} and P_{14}.

The maximum value for the "phantom" cycle starting at T_{CP} is returned as the initial value at T_{C0}. Point P_{18} is not considered at all because it is outside of the query time frame. All other points are considered, but only the points indicated by green markers on the graph are returned (P_{12}, P_{13}, and P_{15}).

In total, four points are returned:

- P_6 as the maximum value of the "phantom" cycle and the initial point
- P_{12} as the maximum value in the first cycle
- P_{13} as the first and only exception occurring in the first cycle
- P_{15} as the maximum value in the second cycle

No points are returned for the incomplete third cycle starting at the query end time, because the tag does not have a point exactly at that time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If the maximum value of the first cycle is located exactly at the query start time, this value and the maximum value of the phantom cycle are returned.

Maximum Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in maximum retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Maximum Retrieval - Query Examples

To use the maximum mode, set the following parameter in your query:

```
   wwRetrievalMode = 'Max'
```

or

```
   wwRetrievalMode = 'Maximum'
```

For examples, see the following:

- Maximum Retrieval - Query 1
- Maximum Retrieval - Query 2
- Maximum Retrieval - Query 3

Maximum Retrieval - Query 1

In this example, an analog tag is retrieved over a five-minute period, using the maximum retrieval mode. Because the wwResolution parameter is set to 60000, each cycle is exactly one minute long. The maximum data value is returned from each of these cycles.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:21:00'
      AND DateTime <= '2005-04-11 11:26:00'
      AND wwRetrievalMode = 'Maximum'
      AND wwResolution = 60000
```

The initial value at the query start time is the maximum value found in the phantom cycle before the start time of the query.

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Cycle	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:21:00.000	ReactTemp	196.00
(cycle 1)	2005-04-11 11:21:00.853	ReactTemp	101.70
(cycle 2)	2005-04-11 11:22:40.837	ReactTemp	196.00
(cycle 3)	2005-04-11 11:23:00.833	ReactTemp	159.20
(cycle 4)	2005-04-11 11:24:59.613	ReactTemp	146.00
(cycle 5)	2005-04-11 11:25:12.083	ReactTemp	196.00

Maximum Retrieval - Query 2

In this example, the maximum retrieval mode is used in a manner equivalent to using the SQL Server MIN aggregate. Note that the cycle producing the result is the five-minute phantom cycle just before the query start time.

```
SELECT TOP 1 DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Value FROM History
   WHERE TagName = 'ReactTemp'
      AND DateTime >= '2005-04-11 11:31:00'
      AND DateTime <= '2005-04-11 11:31:00'
      AND wwRetrievalMode = 'Maximum'
      AND wwResolution = 300000
```

The results are:

	DateTime	TagName	Value
(phantom cycle)	2005-04-11 11:31:00.000	ReactTemp	196.00

Maximum Retrieval - Query 3

This example shows how the maximum retrieval mode marks the QualityDetail column to indicate that a maximum value is returned based on an incomplete cycle. In this case, an incomplete cycle is a cycle that either contained periods with no values stored or a cycle that was cut short because the query end time was located inside the cycle. All values returned for the QualityDetail column are documented in the QualityMap table in the Runtime database.

```
SELECT DateTime, TagName, Value, QualityDetail FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   WHERE TagName = 'SysTimeSec'
      AND DateTime >= '2005-04-11 11:19:10'
      AND DateTime <= '2005-04-11 11:21:10'
      AND wwRetrievalMode = 'Maximum'
      AND wwResolution = 60000
```

The results are:

	DateTime	TagName	Value	QualityDetail
(phantom cycle)	2005-04-11 11:19:10.000	SysTimeSec	NULL	65536
(cycle 1)	2005-04-11 11:19:59.000	SysTimeSec	59	4288
(cycle 2)	2005-04-11 11:20:59.000	SysTimeSec	59	192
(cycle 3)	2005-04-11 11:21:10.000	SysTimeSec	10	4288

Maximum Retrieval - Initial and Final Values

For analog tags, the maximum value of the tag in the cycle leading up to the query start time is returned with its timestamp changed to the query start time. If there is no point exactly at the phantom cycle start time, the point leading up to the phantom cycle is also considered for the maximum calculation. No adjustments are made to the quality of the initial point even though the timestamp may have been altered. Apart from the initial value, all points returned are delta points. (For more information on initial values, see Determining Cycle Boundaries.)

If a point occurs exactly on the query end time, that point is returned with the partial cycle bit, 4096, set in quality detail. If there is more than one such point, only the first point is returned.

Maximum Retrieval - Handling NULL Values and Incomplete Cycles

The first NULL value in a cycle is returned.

When a maximum value is returned from a cycle that contains gaps (including a gap extended from the previous cycle) or from an incomplete cycle with the query end time located inside of the calculation cycle, the point's quality detail is modified to flag this. This is done by performing a logical OR operation of the value 4096, which indicates a partial cycle, onto the existing quality detail.

As an example of how maximum retrieval mode handles NULLs, consider the following query:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'A001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Maximum'
```

If you run this query against the following sample data:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:09	0.2	192
A001	2009-09-12 00:15	1.3	192
A001	2009-09-12 00:17	0.8	192
A001	2009-09-12 00:22	0.5	192
A001	2009-09-12 00:26	0.9	192
A001	2009-09-12 00:28	0.0	249
A001	2009-09-12 00:29	0.0	249
A001	2009-09-12 00:33	1.1	192
A001	2009-09-12 00:35	1.6	192
A001	2009-09-12 00:38	0.5	192
A001	2009-09-12 00:42	0.8	192

The results are:

Tagname	DateTime	Value	QualityDetail
A001	2009-09-12 00:20	1.3	192
A001	2009-09-12 00:26	0.9	4288
A001	2009-09-12 00:28	NULL	249
A001	2009-09-12 00:35	1.6	4288

The sample data points and the results are mapped on the following chart. Only the data falling between the time start and end marks at 00:20 and 00:40 (shown on the chart as dark vertical lines) are returned by the query. The resolution is set at 10,000 milliseconds.

Because there is no value that matches the start time, an initial value at 00:20 is returned based on the maximum value of the preceding cycle, which is the data point at 00:15. In the two subsequent cycles, the maximum values are at 00:26 and 00:35. The quality for these two values is set to 4288 (4096 + 192). The remaining data points are excluded because they are not maximums. In addition, the first NULL at 00:28 is included, but the second NULL (at 00:29) is not.

maximum_mode_new_example

Integral Retrieval

Integral retrieval calculates the values at retrieval cycle boundaries by integrating the graph described by the points stored for the tag. Therefore, it works much like average retrieval, but it additionally applies a scaling factor. This retrieval mode is useful for calculating volume for a particular tag. For example, if one of your tags

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

represents product flow in gallons per second, integral retrieval allows you to retrieve the total product flow in gallons during a certain time period.

Integral retrieval is a true cyclic mode. It returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

Integral retrieval only works with analog tags. For all other tags, normal cyclic results are returned.

Integral Retrieval - How It Works

Calculating values for a cycle in integral retrieval is a two-step process:

- First, the historian calculates the area under the graph created by the data points. This works the same as in average retrieval. For more information, see Average Retrieval.
- After this area has been found, it is scaled using the value of the IntegralDivisor column in the EngineeringUnit table. This divisor expresses the conversion factor from the actual rate to one of units per second.

For example, if the time-weighted average for a tag during a 1-minute cycle is 3.5 liters per second, integral retrieval returns a value of 210 for that cycle (3.5 liters per second multiplied by 60 seconds).

Integral Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in integral retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- Interpolation Type (wwInterpolationType)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Integral Retrieval - Query Example

To use the integral retrieval mode, set the following parameter in your query.

```
   wwRetrievalMode = 'Integral'
```

In this example, the integral is computed for each of five 1-minute long cycles. The wwQualityRule parameter is used to ensure that only points with good quality are used in the computation, which means that points with doubtful quality are discarded. The rules used to determine the returned OPCQuality are the same as described for a time weighted average query.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 2), Value) AS Flow, OPCQuality, PercentGood
FROM History
   WHERE TagName = 'FlowRate'
      AND DateTime >= '2004-06-07 08:00'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      AND DateTime < '2004-06-07 08:05'
      AND wwRetrievalMode = 'Integral'
      AND wwCycleCount = 5
      AND wwQualityRule = 'Good'
```

The results are:

	DateTime	TagName	Flow	OPCQuality	PercentGood
(interval 1)	2004-06-07 08:00:00.000	FlowRate	2862.97	192	100.0
(interval 2)	2004-06-07 08:01:00.000	FlowRate	9436.85	192	100.0
(interval 3)	2004-06-07 08:02:00.000	FlowRate	2480.24	192	100.0
(interval 4)	2004-06-07 08:03:00.000	FlowRate	7379.71	192	100.0
(interval 5)	2004-06-07 08:04:00.000	FlowRate	6317.32	192	100.0

Also, the "phantom" cycle affects the integral retrieval mode just as it does the average retrieval mode. For examples, see Querying Aggregate Data in Different Ways.

Integral Retrieval - wwExpression Query Example

In this example, the integral of a tag named Speed (measured in m/s), is computed over two 1-minute long intervals (wwResolution = 60000 milliseconds) to measure the distance traveled in meters over the interval.

```
SELECT StartDateTime, DateTime, TagName, Value, Quality, QualityDetail as QD, wwUnit
   FROM History
   WHERE TagName = 'Speed'
      AND DateTime >= '2020-12-03 10:00'
      AND DateTime < '2020-12-03 10:02'
      AND wwRetrievalMode = 'Integral'
      AND wwResolution = 60000
```

The results are:

StartDateTime	DateTime	TagName	Value	Quality	QD	wwUnit
2020-12-03 09:59:00.000	2020-12-03 10:00:00.000	Speed	-2906.88948	0	192	m
2020-12-03 10:00:00.000	2020-12-03 10:01:00.000	Speed	877.57619	0	192	m

To display the results using a different unit of measure, you can use the wwExpression query parameter with the following syntax:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   wwExpression = 'UOM(TagName, target unit)'
```

For example, the integral of the Speed tag is computed again over two 1-minute long intervals, but this time the value is converted to the distance in centimeters.

```
SELECT StartDateTime, DateTime, TagName, Value, Quality, QualityDetail as QD, wwUnit
   FROM History
   WHERE TagName = 'Speed'
      AND DateTime >= '2020-12-03 10:00'
      AND DateTime < '2020-12-03 10:02'
      AND wwRetrievalMode = 'Integral'
      AND wwResolution = 60000
      AND wwExpression = 'UOM([Speed],cm)'
```

The results are:

StartDateTime	DateTime	TagName	Value	Quality	QD	wwUnit
2020-12-03 09:59:00.000	2020-12-03 10:00:00.000	Speed	-290688.948	192	192	cm
2020-12-03 10:00:00.000	2020-12-03 10:01:00.000	Speed	87757.619	192	192	cm

Note: Engineering unit conversion requires the advanced feature license. If your Historian does not have the advanced feature license activated, the results are returned in the original units.

Integral Retrieval - Initial and Final Values

If wwTimeStampRule = END, the initial value is calculated by performing an integral calculation on the cycle leading up to the query start time. No special handling is done for the final value.

If wwTimeStampRule = START, the final value is calculated by performing an integral calculation on the cycle following the query end time. No special handling is done for the initial value.

Integral Retrieval - Handling NULL Values

Gaps introduced by NULL values are not included in the integral calculations. The average only considers the time ranges with good values. TimeGood indicates the total time per cycle that the tags value was good.

Slope Retrieval

Slope retrieval returns the slope of a line drawn through a given point and the point immediately before it, thus expressing the rate at which values change.

This retrieval mode is useful for detecting if a tag is changing at too great a rate. For example, you might have a temperature that should steadily rise and fall by a small amount, and a sharp increase or decrease could point to a potential problem.

The slope retrieval mode can be considered a delta mode. Apart from the initial value and a value at the query end time, all returned points are calculated delta points returned with the timestamp of an actual delta point.

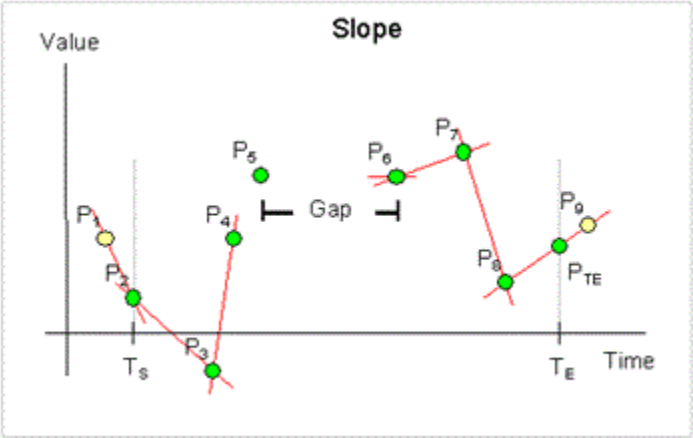
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Slope retrieval only works with analog tags. For all other tags, normal delta results are returned.

All returned values are in chronological order. If multiple points are to be returned for a particular timestamp, they are returned in the order in which the tags were specified in the query.

Slope Retrieval - How It Works

The following illustration shows how the slope is calculated for an analog tag.



This example has a start time of T_S and an end time of T_E.

For the queried tag, a total of nine points are found, represented by the markers P_1 through P_9. Of these points, eight represent normal analog values. The point P_5 represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_5 and P_6.

For every point in the time period, slope retrieval returns the slope of the line going through that point and the point immediately before it. For two points P_1 and P_2 occurring at times T_1 and T_2, the slope formula is as follows:

$(P_2 - P_1) / (T_2 - T_1)$

The difference between T_1 and T_2 is measured in seconds. Therefore, the returned value represents the change in Engineering Units per second.

In this example, point P_2 is located at the query start time, and because there is a prior value (P_1), the slope of the line through both points is calculated and returned at time T_S. Similarly, slopes are calculated to be returned at times T_3, T_4, T_7, and T_8. The slope is also calculated for the line through P_8 and P_9, but that value is returned as point P_{TE} at the query end time.

For point P_6, there is no prior point with which to perform a slope calculation. Instead, the slope of the flat line going through the point (that is, the value 0) is calculated. At the time of P_5, NULL is returned.

The quality detail and OPC quality returned with a slope point is always directly inherited from the point that also provides the time stamp. In this example, this means that point P_2 provides the qualities for the slope point returned at the query start time, T_S.

Slope Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in slope retrieval mode. For more information, see the following sections:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- History Version (wwVersion)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)
- Selecting Values for Analog Summary Tags (wwValueSelector)

Slope Retrieval - Query Example

To use the slope retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Slope'
```

For example, the following query calculates and returns the rate of change of the ReactTemp tag in °C/second. The initial value in the Quality column at the query start time shows no value is located exactly at that time, so the slope returned is the same as the one returned at the next delta point. (For more information on initial values, see Determining Cycle Boundaries.)

At 08:01:17.947 the tag has two delta points, so a slope is calculated and returned for the first point, while a NULL is returned at the second one with a special QualityDetail of 17, indicating that no slope can be calculated as it is either plus or minus infinite.

```
SELECT DateTime, TagName, CONVERT(DECIMAL(10, 4), Value) AS Slope, Quality, QualityDetail
FROM History
   WHERE TagName = 'ReactTemp'
         AND DateTime >= '2005-04-17 08:00'
         AND DateTime <= '2005-04-17 08:05'
         AND wwRetrievalMode = 'Slope'
```

The results are:

DateTime	TagName	Slope	Quality	QualityDetail
2005-04-17 08:00:00.000	ReactTemp	3.8110	133	192
2005-04-17 08:00:00.510	ReactTemp	3.8110	0	192
2005-04-17 08:00:01.713	ReactTemp	4.1563	0	192
2005-04-17 08:00:02.917	ReactTemp	4.1563	0	192
2005-04-17 08:00:04.230	ReactTemp	3.8081	0	192
2005-04-17 08:00:05.433	ReactTemp	4.1563	0	192
…	…	…		
2005-04-17 08:01:16.743	ReactTemp	-1.7517	0	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Slope	Quality	QualityDetail
2005-04-17 08:01:17.947	ReactTemp	-27.0158	0	192
2005-04-17 08:01:17.947	ReactTemp	NULL	1	17
2005-04-17 08:01:19.260	ReactTemp	-1.7530	0	192
2005-04-17 08:01:20.463	ReactTemp	-1.9119	0	192
2005-04-17 08:01:21.667	ReactTemp	-1.9119	0	192
2005-04-17 08:01:22.977	ReactTemp	-1.7517	0	192
…	…	…		

Slope Retrieval - wwExpression Query Example

The following query calculates and returns the rate of change of the Distance tag. The results are returned in the tag's default unit of measure, which in this case is feet/second.

```
SELECT DateTime, TagName, Value, QualityDetail, wwUnit FROM History
   WHERE TagName = 'Distance'
         AND wwRetrievalMode = 'Slope'
```

The results are:

DateTime	TagName	Value	QualityDetail	wwUnit
2020-07-29 06:50:53.748	Distance	1	192	ft/s
2020-07-29 06:50:54.748	Distance	2	192	ft/s

To display the results using a different unit of measure, you can use the wwExpression query parameter with the following syntax:

```
   wwExpression = 'UOM(TagName, target unit)'
```

For example, the following query calculates and returns the rate of change of the Distance tag in meters/second.

```
SELECT DateTime, TagName, Value, QualityDetail, wwUnit FROM History
   WHERE TagName = 'Distance'
         AND wwExpression = 'UOM(Distance, m/s)'
         AND wwRetrievalMode = 'Slope'
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value	QualityDetail	wwUnit
2020-07-29 06:50:53.748	Distance	0.3048	192	m/s
2020-07-29 06:50:54.748	Distance	0.6096	192	m/s

Note: Engineering unit conversion requires the advanced feature license. If your Historian does not have the advanced feature license activated, the results are returned in the original units.

Slope Retrieval - Initial and Final Values

An initial value is always generated. If a point is stored exactly at the query start time, the slope is returned as the slope between that point and the previous point. Otherwise, the slope is calculated using the slope of the points before and after the query start time.

A final value is always generated. If a point is stored exactly at the query end time, the slope is returned as the slope between that point and the previous point. Otherwise, the slope is calculated using the slope of the points before and after the query end time.

Slope Retrieval - Handling NULL Values

The first NULL following a non-NULL value is returned. Subsequent NULL values are not. If a point is preceded by a NULL, the slope for that point will be zero.

Counter Retrieval

Counter retrieval allows you to accurately retrieve the delta change of a tag's value over a period of time even for tags that are reset upon reaching a "rollover value." The rollover value is defined in the AVEVA Historian for each tag.

This retrieval mode is useful for determining how much of an item was produced during a particular time period. For example, you might have an integer counter that keeps track of how many cartons were produced. The counter has an indicator like this:

The next value after the highest value that can be physically shown by the counter is called the rollover value. In this example, the rollover value is 10,000. When the counter reaches the 9,999th value, the counter rolls back to 0. Therefore, a counter value of 9,900 at one time and a value of 100 at a later time means that you have produced 200 units during that period, even though the counter value has dropped by 9,800 (9,900 minus 100). Counter retrieval allows you to handle this situation and receive the correct value. For each cycle, the counter retrieval mode shows the increase in that counter during the cycle, including rollovers.

Note: If Historian receives a data tag value that is outside of the specified engineering unit boundaries, Historian ignores the received value and instead returns a 0. For example, if a value can be an integer between 1 and 8, and the received value is 9, Historian returns 0 as the value for that tag during that cycle.

Counter retrieval also works with floating point counters, which is useful for flow meter data. Similar to the carton counter, some flow meters "roll over" after a certain amount of flow accumulates. For both examples, the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

need is to convert the accumulating measure to a "delta change" value over a given period.

Counter retrieval is a true cyclic mode. It returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

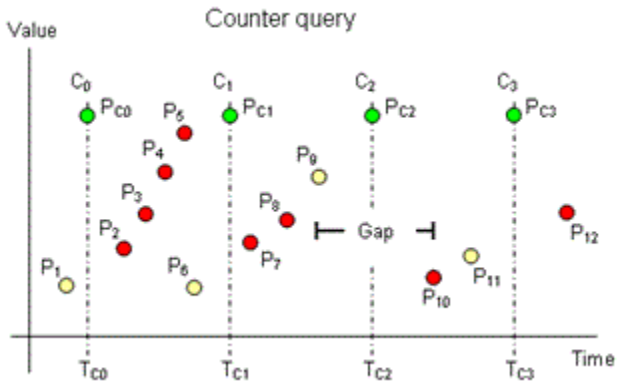
The counter algorithm is only applied to analog tags and to discrete tags. For integer analog tags, the result will be an integer returned as a float data type. For a real analog tag, the rollover value and the result may be real values and can include fractional values. If a query contains tags of other types, then no rows are returned for those tags. For discrete tags, the rollover value is assumed to be 2.

The rules used to determine the OPCQuality returned with a counter value are the same as for a time weighted average query. For more information, see Quality Rule (wwQualityRule). When a rollover has occurred in the calculation cycle, a special quality detail of 212 is returned in all non-NULL cases.

CTU counters will default to "signed integer" tags when imported into the Historian, giving a normal range of -2147483648 to 2147483647 (for a 32-bit integer). In operation, these counters will count up to the upper limit and "rollover" to the lower limit on the next increment. If these tags are changed to be "unsigned integers" the normal range will be 0 to 4294967295 and values will rollover to "0", conforming to the expected behavior of a tag used with "counter" retrieval. When used with a 16-bit CTU counter, the same rules apply, but the range of values is -32768 to 32767 as "signed" and 0 to 65535 for an "unsigned".

Counter Retrieval - How It Works

The following illustration shows how the counter algorithm determines the count for an analog tag.



This example has a start time of T_{C0} and an end time of T_{C3}. The resolution has been set in such a way that the historian returns data for three complete cycles starting at T_{C0}, T_{C1}, and T_{C2}, and an incomplete cycle starting at T_{C3}.

For the queried tag, a total of twelve points are found throughout the cycles represented by the markers P_1 through P_{12}. Of these points, eleven represent normal analog values. The point P_9 represents a NULL due to an I/O Server disconnect, which causes a gap in the data between P_9 and P_{10}. Point P_{12} is not considered because it is outside of the query time frame.

All points are considered in the counter calculation, but only the yellow ones are actually used to determine which values to return to the client. The returned points are P_{C0}, P_{C1}, P_{C2} and P_{C3}, shown in green at the top to indicate that there is no simple relationship between them and any of the actual points.

All cycle values are calculated as the delta change between the cycle time in question and the previous cycle time, taking into account the number of rollovers between the two points in time. The counter algorithm assumes that a rollover occurred if the current value is lower than the previous value. The initial value at the query start time (P_{C1}) is calculated the same way, only based on a phantom cycle before the query start time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For example, the formula to calculate P_{C1} is as follows:

$P_{C1} = n * V_R + P_6 - P_1$

where:

- n = the number of rollovers that have occurred during the cycle
- V_R = the rollover value for the tag

If either n or V_R are equal to zero, P_{C1} is simply the difference between the values P_1 and P_6.

In the case of cycle C_2, there is no value at the cycle time, so the NULL value represented by point P_9 is returned. In the case of cycle C_3, a NULL is again returned, because there is no counter value at the previous cycle boundary to use in the calculation. There must be a full cycle of values in order for the counter to be calculated.

If a gap is fully contained inside a cycle, and if points occur within the cycle on both sides of the gap, then a counter value is returned, even though it may occasionally be erroneous. Zero or one rollovers are assumed, even though the counter might have rolled over multiple times.

Counter Retrieval - Calculations for a Manually Reset Counter

If you have a counter that you typically reset manually before it rolls over, you must set the rollover value for the tag to 0 so that the count is simply how much change occurred since the manual reset.

For example, assume that you have the following data values for five consecutive cycle boundaries, and that the value 0 occurs as the first value within the last cycle:

100, 110, 117, 123, 3

If you set the rollover value to 0, the counter retrieval mode assumes that the 0 following the value 123 represents a manual reset, and returns a value of 3 for the last cycle, which is assumed to be the count after the manual reset. The value 0 itself does not contribute 1 to the counter value in this case.

If the rollover value is instead set to 200, then the counter retrieval mode assumes that the value 0 represents a normal rollover, and a count of 80 is calculated and returned (200 - 123 + 3). In this case, the value 0 contributes 1 to the counter value, and that is the change from the value 199 to the value 200.

Counter Retrieval - Using a Counter Deadband

You can set a deadband for counter retrieval to better handle reporting of rates and quantities. For example, setting a counter deadband can help to:

- Distinguish between counter resets and rollovers.
- Filter out counter reversals.

The counter deadband is the percentage of the full range of the counter.

For example, if you set the counter deadband to be 10, then any counter reset that occurs within the top 10% of the range is assumed to be a rollover and is counted as such.

For a reversal, you might have a conveyor belt carrying boxes that are counted as they pass by a station. If a jam occurs, you might reverse the conveyor belt to clear it, resulting in the counter decrementing and then incrementing again. In this case, a counter deadband of 10 would discard any duplicate counts within 10% of the range starting from the point of reversal. Only one reversal can be detected until the rollover occurs.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Counter Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in integral retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- Analog Value Filtering (wwFilter)

Counter Retrieval - Initial and Final Values

An initial value is returned using the period leading up to the query start time.

A data point that has a cycle time is used to generate the counter value for its preceding cycle. A NULL point with cycle time will cause the preceding cycle to end in a gap and the following cycle to start with a gap.

Counter Retrieval - Handling NULL Values

If wwQualityRule is configured as OPTIMISTIC, NULL data points will not be used in calculation. 0.0 will be used as the starting base value for the query unless the query data starts with a NULL. If the query starts with a NULL, the value change for the cycle is calculated from the first actual value in the cycle, rather than 0.

Otherwise, if any points considered in a cycle have UNCERTAIN quality, the result for that row will also have UNCERTAIN quality. Any cycle that starts or ends in a gap will have a quality detail of 65536.

The quality detail of DOUBTFUL will be used with the counter result for the cycles, if a NULL point is considered for the cycle and the counter result is not NULL.

Counter Retrieval - Handling Illegal Values

If the configured rollover value is larger than 0.0, then the data points whose values are greater than or equal to the rollover value causes the counter value for the cycle to be set to 0.0, with qdIO_FILTEREDPOINT applied to the quality detail.

Similarly, if any data point with a value less than 0.0 is found in a cycle, the counter value for the cycle is set to 0.0, with qdIO_FILTEREDPOINT applied to the quality detail.

Counter Retrieval - Query Example

To use the counter mode, set the following parameter in your query.

```
wwRetrievalMode = 'Counter'
```

In the following example, the rollover value for the SysTimeSec system tag is set to 0. In a five-minute time span, the SysTimeSec tag increments a total of 300 times. The following query returns the total count within the five-minute time span, which begins with StartTime and ends with DateTime. The QualityDetail of 212 indicates that a counter rollover occurred during the query time range.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
select StartDateTime, DateTime, TagName, Value, wwUnit, QualityDetail as QD from History
   where TagName = 'SysTimeSec'
      and DateTime > '2020-12-02 1:00'
      and DateTime <= '2020-12-02 1:05'
      and wwRetrievalMode = 'counter'
      and wwCycleCount = 1
```

The results are:

StartTime	DateTime	TagName	Value	wwUnit	QD
2020-12-02 01:00:00.0000000	2020-12-02 01:05:00.0000000	SysTimeSec	300	Second	212

To display the results using a different unit of measure, you can use the wwExpression query parameter with the following syntax:

```
   wwExpression = 'UOM(TagName, target unit)'
```

The following example is similar to the previous example. The following query returns the total count within the five-minute time span, which begins with StartTime and ends with DateTime, and converts the retrieved value from its original units (seconds) to minutes.

```
select StartDateTime, DateTime, TagName, Value, Quality, QualityDetail as QD from History
   where TagName = 'SysTimeSec'
      and DateTime > '2020-12-02 1:00'
      and DateTime <= '2020-12-02 1:05'
      and wwRetrievalMode = 'counter'
      and wwCycleCount = 1
      and wwExpression = 'UOM(SysTimeSec,Minute)'
```

The results are:

StartTime	DateTime	TagName	Value	wwUnit	QD
2020-12-02 01:00:00.0000000	2020-12-02 01:05:00.0000000	SysTimeSec	5	Minute	212

Note: Engineering unit conversion requires the advanced feature license. If your Historian does not have the advanced feature license activated, the results are returned in the original units.

ValueState Retrieval

ValueState retrieval returns information on how long a tag has been in a particular value state during each retrieval cycle. That is, a time-in-state calculation is applied to the tag value.

This retrieval mode is useful for determining how long a machine has been running or stopped, how much time a process spent in a particular state, how long a valve has been open or closed, and so on. For example, you might have a steam valve that releases steam into a reactor, and you want to know the average amount of time the valve was in the open position during the last hour. ValueState retrieval can return the shortest, longest, average, or total time a tag spent in a state, or the time spent in a state as a percentage of the total cycle length.

When you use ValueState retrieval for a tag in a trend chart, you must specify a single value state for which to retrieve information. ValueState retrieval then returns one value for each cycle—for example, the total amount of time that the valve was in the "open" state during each 1-hour cycle. This information is suitable for trend

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

display.

If you *don't* specify a state, ValueState retrieval returns one row of information for each value that the tag was in during each cycle. For example, this would return not only the time a valve was in the "open" state, but also the time it was in the "closed" state. This information is not suitable for meaningful display in a regular trend. You can, however, retrieve this type of information in a query and view it as a table.

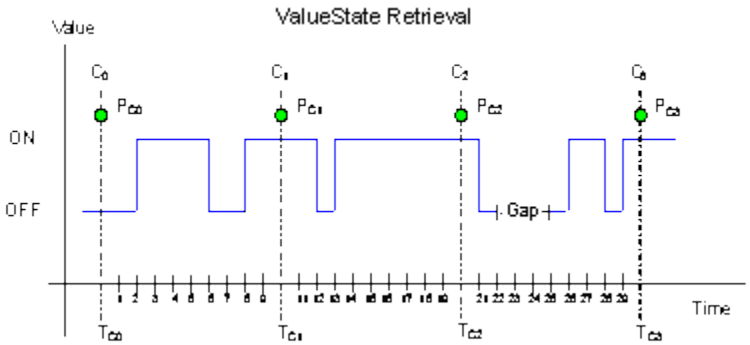
ValueState retrieval works with integer, discrete, string, and state summary tags. For other types of tags, no rows are returned. NULL values are treated like any other distinct state.

The values returned at the query start time are the result of applying the algorithm to a "phantom" cycle preceding the query range. It is assumed that the tag value at the start of the cycle is located at that point in time.

To specify the type of calculation, set the wwStateCalc parameter in the query. For more information, see State Calculation (wwStateCalc).

ValueState Retrieval - How It Works

The following illustration shows how ValueState retrieval returns values for a discrete tag.



This example has a start time of T_{C0} and an end time of T_{C3}. The resolution has been set in such a way that the historian returns data for three complete cycles starting at T_{C0}, T_{C1}, and T_{C2}, and an incomplete cycle starting at T_{C3}. Time is measured seconds.

A gap in the data occurs in the third cycle due to an I/O Server disconnect.

The state calculation is based on each cycle, and the values returned at the query start time are not regular initial values, but are the resulting values that occur after applying the algorithm to the last cycle preceding the query range. The returned points are P_{C0}, P_{C1}, P_{C2} and P_{C3}, shown in green at the top to indicate that there is no simple relationship between the calculated values and any of the actual points.

Assume the query is set so that the total time (wwStateCalc = 'Total')in the two states are returned. The timestamping is set to use the cycle end time.

- For T_{C0}, the query returns two rows (one for the "on" state and one for the "off" state), calculated as a result of the "phantom" cycle that precedes the query start time. The values have a timestamp of the query start time.
- For T_{C1}, one row is returned for the "on" state. The "on" state occurred twice during the cycle--one time for four seconds and again for two seconds before the cycle boundary, and the total time returned is six seconds. The state was "off" twice during the cycle for a total time of four seconds, and one row is returned with that value.
- For T_{C2}, one row is returned for the "on" state, and one row is returned for the "off" state. The "on" state

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

occurred for a total of nine seconds between the cycle boundaries, and the "off" state occurred for a total of one second.

- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The "on" state occurred for a total of four seconds between the cycle boundaries, and the "off" state occurred for a total of three seconds. An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect.

Using the same data, if you queried the **total contained** time for the states, the following is returned:

- For T_{C0}, the query returns two values (one for the "on" state and one for the "off" state), calculated as a result of the "phantom" cycle the precedes the query start time. The value has a timestamp of the query start time.
- For T_{C1}, one row is returned for the "on" state, and one row is returned for the "off" state. The "on" state occurred one time for four seconds within the cycle. The two seconds of "on" time that crosses the cycle boundary does not contribute to the total time. The state was "off" one time during the cycle for two seconds completely within the cycle boundary.
- For T_{C2}, the state was not "on" for any contained time between the cycle. Both occurrences of the "on" state cross over a cycle boundary, so no rows are returned for this state. One row is returned for the "off" state. The state was "off" one time during the cycle for one seconds completely within the cycle boundary.
- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The state was "on" for a single contained time of two seconds between the cycle boundaries. The state was "off" three times during the cycle for three seconds completely within the cycle boundary. An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect. The state was NULL for a total of three seconds. The I/O Server disconnect that "disrupted" the off state is treated as its own state, thereby changing what would have been a single "off" state instance of five seconds into two instances of the "off" state for one second each.

ValueState Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in ValueState retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- State Calculation (wwStateCalc)
- Analog Value Filtering (wwFilter)

ValueState Retrieval - Query Examples

To use the ValueState retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'ValueState'
```

To specify the type of aggregation, set the wwStateCalc parameter in the query, such as:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   wwStateCalc = 'Total'
```

Be sure that you use the "<=" operator for ending date/time.

For examples, see the following:

- ValueState Retrieval Query 1: Minimum Time in State
- ValueState Retrieval Query 2: Minimum Time in State for a Single Tag
- ValueState Retrieval Query 3
- ValueState Retrieval Query 4
- ValueState Retrieval Query 5
- ValueState Retrieval Query 6: Querying State Summary Values

ValueState Retrieval Query 1: Minimum Time in State

The following query finds the minimum time-in-state for the SteamValve discrete tag. Note that minimum times are returned for each state for both the five-minute phantom cycle before the query start time and for the single retrieval cycle between 10:00 and 10:05.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc FROM History
   WHERE TagName IN ('SteamValve')
      AND DateTime >= '2005-04-17 10:00:00'
      AND DateTime <= '2005-04-17 10:05:00'
      AND wwCycleCount = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Min'
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	0	35359.0	MINIMUM
2005-04-17 10:00:00.000	SteamValve	1	43749.0	MINIMUM
2005-04-17 10:05:00.000	SteamValve	0	37887.0	MINIMUM
2005-04-17 10:05:00.000	SteamValve	1	43749.0	MINIMUM

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueState Retrieval Query 2: Minimum Time in State for a Single Tag

The following query finds the minimum time-in-state for the Mixer discrete tag for the "on" state. Note that minimum times are returned for each state for both the five-minute phantom cycle before the query start time and for the single retrieval cycle between 10:00 and 10:05.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc
   FROM History
      WHERE TagName IN ('Mixer')
         AND DateTime >= '2017-12-11 08:00:00'
         AND DateTime < '2017-12-12 08:00:00'
         AND wwCycleCount= '1'
         AND wwRetrievalMode = 'ValueState'
         AND wwStateCalc = 'Min'
         AND vValue = 1
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2017-12-11 08:00:00.0000000	Mixer	1	35906	Min

ValueState Retrieval Query 3

The following query finds the maximum time-in-state for the SteamValve discrete tag in the same time period as in Query 1. Note how both the minimum and maximum values for the "1" state are very similar, while they are very different for the "0" state. This is due to the "cut-off" effect.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc FROM History
   WHERE TagName IN ('SteamValve')
      AND DateTime >= '2005-04-17 10:00:00'
      AND DateTime <= '2005-04-17 10:05:00'
      AND wwCycleCount = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Max'
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	0	107514.0	MAXIMUM
2005-04-17 10:00:00.000	SteamValve	1	43750.0	MAXIMUM
2005-04-17 10:05:00.000	SteamValve	0	107514.0	MAXIMUM

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2005-04-17 10:05:00.000	SteamValve	1	43750.0	MAXIMUM

ValueState Retrieval Query 4

The following query returns the total of time in state for a discrete tag. In this example, the TimeStampRule system parameter is set to "End" (the default setting), so the returned values are timestamped at the end of the cycle. The returned rows represent the time-in-state behavior during the period starting at 2005-04-13 00:00:00.000 and ending at 2005-04-14 00:00:00.000.

```
SELECT DateTime, vValue, StateTime, wwStateCalc FROM History
   WHERE DateTime > '2005-04-13 00:00:00.000'
      AND DateTime <= '2005-04-14 00:00:00.000'
      AND TagName IN ('PumpStatus')
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Total'
      AND wwCycleCount = 1
```

The results are:

DateTime	vValue	StateTime	wwStateCalc
2005-04-14 00:00:00	NULL	1041674.0	TOTAL
2005-04-14 00:00:00	On	56337454.0	TOTAL
2005-04-14 00:00:00	Off	29020872.0	TOTAL

ValueState Retrieval Query 5

The following query returns the percentage of time in state for a discrete tag for multiple retrieval cycles. The TimeStampRule system parameter is set to "End" (the default setting), so the returned values are timestamped at the end of the cycle. Note that the first row returned represents the results for the period starting at 2003-07-03 22:00:00.000 and ending at 2003-07-04 00:00:00.000.

The "Percent" time-in-state retrieval mode is the only mode in which the StateTime column does not return time data. Instead, it returns percentage data (in the range of 0 to 100 percent) representing the percentage of time in state.

```
SELECT DateTime, vValue, StateTime, wwStateCalc FROM History
   WHERE DateTime >= '2003-07-04 00:00:00.000'
      AND DateTime <= '2003-07-05 00:00:00.000'
      AND TagName IN ('PumpStatus')
      AND Value = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Percent'
      AND wwCycleCount = 13
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	vValue	StateTime	wwStateCalc
2003-07-04 00:00:00	1	50.885	PERCENT
2003-07-04 02:00:00	1	82.656	PERCENT
2003-07-04 04:00:00	1	7.082	PERCENT
2003-07-04 06:00:00	1	7.157	PERCENT
2003-07-04 08:00:00	1	55.580	PERCENT
2003-07-04 10:00:00	1	28.047	PERCENT
2003-07-04 12:00:00	1	47.562	PERCENT
2003-07-04 14:00:00	1	74.477	PERCENT
2003-07-04 16:00:00	1	40.450	PERCENT
2003-07-04 18:00:00	1	78.313	PERCENT
2003-07-04 20:00:00	1	54.886	PERCENT
2003-07-04 22:00:00	1	39.569	PERCENT
2003-07-05 00:00:00	1	50.072	PERCENT

ValueState Retrieval Query 6: Querying State Summary Values

If state summary values are queried and the cycle boundaries match the summary periods, the ValueState calculations are supported and return valid results.

If state summary points are queried and the cycle boundaries do not match the summary periods, the ValueState calculations are supported, but they return DOUBTFUL (QualityDetail = 64) results.

State summaries are included in the cycle where the summary end time occurs. This causes results that do not match queries against the source tag and may cause inaccurate results, such as a total state time that is greater than the cycle time.

For example, this can occur if SysTimeSec is summarized with a state summary with one minute resolution, but then queried with 10 second intervals. In most of the retrieval cycles, there will be no values, but in the cycle that includes the summary end time (one in six of the retrieval cycles), all 60 states would be returned with each state having a state time of 1 second for a total of 60 seconds of state time in a 10 second retrieval cycle.

ValueState Retrieval - Initial and Final Values

The values returned at the query start time are the result of applying the algorithm to the last cycle preceding the query range.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ValueState Retrieval - Handling NULL Values

NULLs are considered a state and are reported along with the other states.

RoundTrip Retrieval

RoundTrip retrieval is very similar to ValueState retrieval in that it performs calculations on state occurrences in the within a cycle period you specify. However, ValueState retrieval uses the time spent in a certain state for the calculation, and RoundTrip retrieval uses the time between consecutive leading edges of the same state for its calculations.

You can use the RoundTrip retrieval mode for increasing the efficiency of a process. For example, if a process produces one item per cycle, then you would want to minimize the time lapse between two consecutive cycles.

The RoundTrip mode returns a row for each state in any given cycle. RoundTrip retrieval only works with integer analog tags, discrete tags, and string tags. If real analog tags are specified in the query, then no rows are returned for these tags. RoundTrip retrieval is not applied to state summary or analog summary tags. NULL values are treated as any other distinct value and are used to analyze the round trip for disturbances.

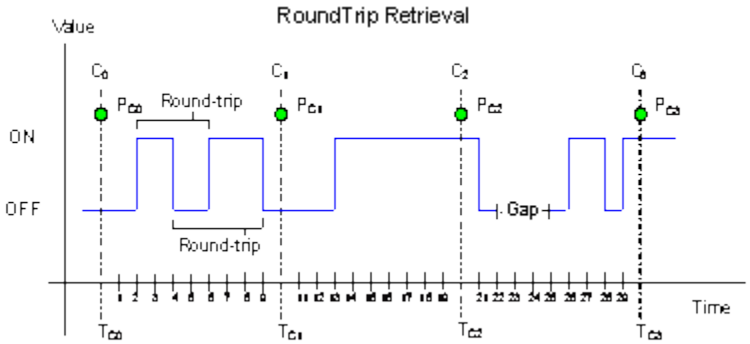
RoundTrip retrieval is supported for the History and StateWideHistory tables.

Any point on the boundary of the end cycle will be considered to the next cycle. The point on the boundary of the end query range is not counted in the calculation except that it is used to indicate that the previous state is a contained state.

If no roundtrip state is found within the cycle for a supported tag, a NULL StateTime value is returned. If there is no valid point prior to the phantom cycle, a NULL state is returned for the phantom cycle.

RoundTrip Retrieval - How It Works

The following illustration shows how RoundTrip retrieval returns values for a discrete tag.



This example has a start time of T_{C0} and an end time of T_{C3}. The resolution has been set in such a way that the historian returns data for three complete cycles starting at T_{C0}, T_{C1}, and T_{C2}, and an incomplete cycle starting at T_{C3}. Time is measured seconds.

A gap in the data occurs in the third cycle due to an I/O Server disconnect.

The state calculation is based on each cycle, and the values returned at the query start time are not regular initial values, but are the resulting values that occur after applying the algorithm to the last cycle preceding the query range. The returned points are P_{C0}, P_{C1}, P_{C2} and P_{C3}, shown in green at the top to indicate that there is no simple relationship between the calculated values and any of the actual points.

Assume the query is set so that the total contained time in the two states are returned. The timestamping is set

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

to use the cycle end time. The RoundTrip retrieval mode returns values for states that are completely contained within the cycle boundaries. The following is returned:

- For T_{C0}, the query returns two values (one for the "on" state and one for the "off" state), calculated as a result of the "phantom" cycle that preceeds the query start time. The value has a timestamp of the query start time.
- For T_{C1}, one row is returned for the "on" state, and one row is returned for the "off" state. The round-trip for the "on" state occurred one time for four seconds completely within the cycle boundary. The round-trip for the "off" state occurred one time during the cycle for five seconds.
- For T_{C2}, a round-trip did not occur for either state within the cycle boundaries. One NULL row is returned for this cycle.
- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The state was "on" for a single contained time of two seconds between the cycle boundaries. The state was "off" one time during the cycle for one second completely within the cycle boundary. An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect.
- For T_{C3}, one row is returned for the "on" state, and one row is returned for the "off" state. The state was "on" for a single contained time of three seconds between the cycle boundaries. One row is returned for the "off" state for a total contained time of seven seconds. (The first round-trip for the "off" state includes the I/O Server disconnect for a length of four seconds. The second round-trip has a length of three seconds.) An additional row is returned for the NULL state occurring as a result of the I/O Server disconnect, and the returned value is NULL because there is no round-trip during the cycle for it. The I/O Server disconnect that "disrupted" the off state is treated as its own state, thereby changing what would have been a single "off" state instance of five seconds into two instances of the "off" state for one second each.

RoundTrip Retrieval - Supported Value Parameters

You can use various parameters to adjust the values that must be returned in the RoundTrip retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)
- Resolution (Values Spaced Every X ms) (wwResolution)
- History Version (wwVersion)
- TimeStamp Rule (wwTimeStampRule)
- Quality Rule (wwQualityRule)
- State Calculation (wwStateCalc)
- Analog Value Filtering (wwFilter)

RoundTrip Retrieval - Query Examples

To use the RoundTrip retrieval mode, set the following parameter in your query:

```
wwRetrievalMode = ‘RoundTrip’
```

The following queries compare the results between ValueState retrieval and RoundTrip retrieval.

This first ValueState retrieval query returns the average amount of time that the 'Reactor1OutletValve' tag is in "on" state and the average amount of time it is in the "off" state for a single cycle. Any state changes that occur across the cycle boundaries are not included.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
SELECT DateTime, vValue, StateTime
   FROM History
      WHERE TagName IN ('Reactor1OutletValve')
      AND DateTime >= '2009-09-16 12:35:00'
      AND DateTime <= '2009-09-16 12:55:00'
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'AvgContained'
      AND wwCycleCount = 1
```

The results are:

DateTime	vValue	StateTime
2009-09-16 12:35:00.0000000	0	215878
2009-09-16 12:35:00.0000000	1	61729
2009-09-16 12:55:00.0000000	1	62827.5
2009-09-16 12:55:00.0000000	0	212856

The first two rows are for the "phantom" cycle leading up to the query start time and have a timestamp of the query start time.

The second two rows show the average amount of time for each state and have a timestamp of the query end time (the default).

Compare these results to same basic query that instead uses RoundTrip retrieval:

```
SELECT DateTime, vValue, StateTime
   FROM History
      WHERE TagName IN ('Reactor1OutletValve')
      AND DateTime >= '2009-09-16 12:35:00'
      AND DateTime <= '2009-09-16 12:55:00'
      AND wwRetrievalMode = 'RoundTrip'
      AND wwStateCalc = 'AvgContained'
      AND wwCycleCount = 1
```

The results are:

DateTime	vValue	StateTime
2009-09-16 12:35:00.0000000	1	277607
2009-09-16 12:35:00.0000000	0	278580
2009-09-16 12:55:00.0000000	0	275683.5
2009-09-16 12:55:00.0000000	1	273845

RoundTrip Retrieval - Initial and Final Values

The values returned at the query start time are the result of applying the algorithm to the last cycle preceding the query range.

RoundTrip Retrieval - Handling NULL Values

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Like in the ValueState retrieval mode, the NULL state is treated as a valid distinct value. This allows you to analyze round trips for disturbances.

Edge Detection for Events (wwEdgeDetection)

For AVEVA Historian, an event is the moment at which a detection criterion is met on historical tag values in the AVEVA Historian. At a basic level, anything that can be determined by looking at stored data can be used as an event.

When detecting events, it is useful to pinpoint rows in a result set where the satisfaction of criteria in a WHERE clause changed from true to false, or vice versa. For example, you may want to know when the level of a tank went above 5 feet. The WHERE clause in a query for this example might be TANKLEVEL > 5. As the tank level approaches 5 feet, the criterion does not return true. Only when the level crosses the line from not satisfying the criterion to satisfying it, does the event actually occur. This imaginary "line" where the change occurs is called the edge.

Over a period of time, there may be many instances where the criteria cross the "edge" from being satisfied to not satisfied, and vice-versa. The values on either side of this "edge" can be detected if you configure your event tag to include this information. There are four possible options for edge detection: none, leading, trailing, or both. You will get differing results based on which option you use:

Option	Results
None	Returns all rows that successfully meet the criteria; no edge detection is implemented at the specified resolution.
Leading	Returns only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false).
Trailing	Returns only rows that are the first to fail the criteria (return false) after a row successfully met the criteria (returned true).
Both	All rows satisfying both the leading and trailing conditions are returned.

Edge detection only applies to analog and discrete value detectors. Also, edge detection is handled slightly differently based on whether you are using analog tags or discrete tags.

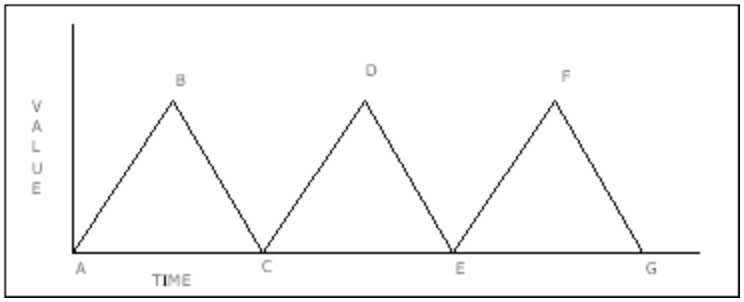
You can also use the ToDiscrete() query filter to determine when data values cross a particular threshold. For more information, see Converting Analog Values to Discrete Values (ToDiscrete).

For more information on event detection with the Classic Event subsystem, see Classic Event Subsystem in the *AVEVA Historian Administration Guide*.

Edge Detection for Analog Tags

For example, the behavior of the WHERE clause as it processes a result set can be illustrated as:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Slopes A-B, C-D and E-F are rising edges, while slopes B-C, D-E and F-G are falling edges. The slopes are affected by the WHERE clause, which is a combination of the wwEdgeDetection option and the comparison operator used against the value.

The following table describes the rows that would be returned for the various edge detection settings:

Operator	**=**	**<**	**>**	**<=**	**>=**
Leading	Falling and rising edges; first value that meets the criteria.	Falling edge only; first value to meet the criteria.*	Rising edge only; first value to meet the criteria.	Falling edge only; first value to meet the criteria. *	Rising edge only; first value to meet the criteria.
Trailing	Falling and rising edges; first value to fail the criteria after a value meets the criteria.	Rising edge only; equal to the first value to fail the criteria.	Falling edge only; first value to fail the criteria.*	Rising edge only; first value to fail the criteria.	Falling edge only; first value to fail the criteria.*

* If the falling edge is a vertical edge with no slope, the query will return the lowest value of that edge.

The following query selects all values of "SysTimeSec" that are greater than or equal to 50 from the History table between 10:00 and 10:02 a.m. on December 2, 2001. No edge detection is specified.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'None'
```

The results are:

DateTime	Value
2001-12-02 10:00:50.000	50

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-02 10:00:52.000	52
2001-12-02 10:00:54.000	54
2001-12-02 10:00:56.000	56
2001-12-02 10:00:58.000	58
2001-12-02 10:01:50.000	50
2001-12-02 10:01:52.000	52
2001-12-02 10:01:54.000	54
2001-12-02 10:01:56.000	56
2001-12-02 10:01:58.000	58

Leading Edge Detection for Analog Tags

If Leading is specified as the parameter in the edge detection time domain extension, the only rows in the result set are those that are the first to successfully meet the WHERE clause criteria (returned true) after a row did not successfully meet the WHERE clause criteria (returned false).

The following query selects the first values of "SysTimeSec" from the History table to meet the Value criterion between 10:00 and 10:02 a.m. on December 2, 2001.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'Leading'
```

The query will return only the two values that were greater than or equal to 50 for the time period specified:

DateTime	Value
2001-12-02 10:00:50.000	50
2001-12-02 10:01:50.000	50

Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags. Notice that even though the original query returned ten rows, the edge detection only returns the first row recorded after the event criteria returned true.

Trailing Edge Detection for Analog Tags

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

If Trailing is specified as the parameter in the edge detection extension, the only rows in the result set are those that are the first to fail the criteria in the WHERE clause (returned false) after a row successfully met the WHERE clause criteria (returned true).

The following query selects the first values of "SysTimeSec" from the History table to fail the Value criterion between 10:00 and 10:02 a.m. on December 2, 2001.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'Trailing'
```

The query returns only the two values that were the first to fail the criteria in the WHERE clause for the time period specified:

DateTime	Value
2001-12-02 10:01:00.000	0
2001-12-02 10:02:00.000	0

Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags. Notice that even though the original query returned ten recorded rows for each value, the edge detection only returns the first row recorded after the event criteria returned false.

Both Leading and Trailing Edge Detection for Analog Tags

If Both is specified as the parameter in the edge detection extension, all rows satisfying both the leading and trailing conditions are returned.

The following query selects values of "SysTimeSec" from the History table that meet both the Leading and Trailing criteria between 10:00 and 10:02 a.m. on December 2, 2001.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
         AND Value >= 50
         AND wwEdgeDetection = 'Both'
```

The results are:

DateTime	Value
2001-12-02 10:00:50.000	50

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-02 10:01:00.000	0
2001-12-02 10:01:50.000	50
2001-12-02 10:02:00.000	0

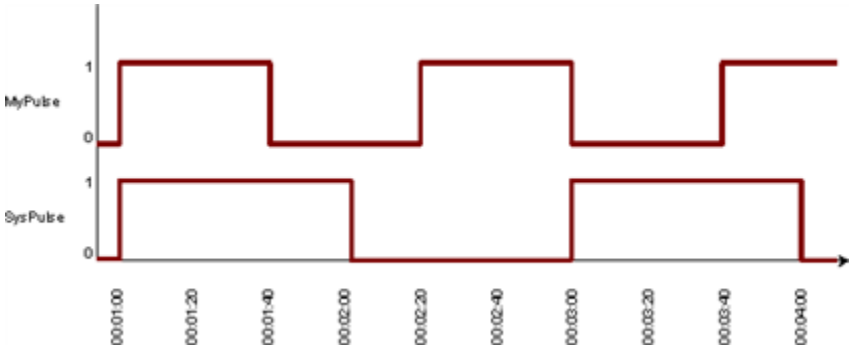
Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags. Notice that value of the first row in the original query is returned in the result set.

Edge Detection for Discrete Tags

Edge detection for discrete tags operates differently than for analog tags. For example, assume the following discrete tags are stored.

Tag	Description
SysPulse	Transitions between 1 and 0 every minute.
MyPulse	Transitions between 1 and 0 every 40 seconds.

A representation of the data stored in the system is as follows:



The following queries select values of "SysPulse" and "MyPulse" that have a value of 1 (On) from the History and WideHistory tables between 12:59 and 1:04 a.m. on December 8, 2001. No edge detection is specified.

This is a query of the History table:

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND wwRetrievalMode = 'Delta'
         AND Value = 1
         AND wwEdgeDetection = 'None'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1
2001-12-08 00:01:00.000	MyPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1
2001-12-08 00:03:40.000	MyPulse	1

The following is a query of the WideHistory table:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse, MyPulse FROM WideHistory
   WHERE DateTime > "2001-12-08 00:59:00"
      AND DateTime < "2001-12-08 01:05:00"
      AND SysPulse = 1
      AND MyPulse = 1
      AND wwRetrievalMode = "Delta"
      AND wwEdgeDetection = "None"
   ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1

Leading Edge Detection for Discrete Tags

If Leading is specified as the parameter in the edge detection time domain extension, the only rows in the result set are those that are the first to successfully meet the WHERE clause criteria (returned true) after a row did not successfully meet the WHERE clause criteria (returned false).

The following queries select values of "SysPulse" and "MyPulse" that have a value of 1 (On) from the History and WideHistory tables between 12:59 and 1:04 a.m. on December 8, 2001.

This example queries the History table, if the WHERE clause criteria specify to return only discrete values equal to 1 (On), then applying a leading edge detection does not change the result set.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND Value = 1
         AND wwEdgeDetection = 'Leading'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1
2001-12-08 00:01:00.000	MyPulse	1
2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:03:40.000	MyPulse	1

This example queries the WideHistory table, applying a leading edge detection requires that the condition only evaluate to true if both values are equal to 1 (On).

```
SELECT DateTime, SysPulse, MyPulse FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse,
MyPulse
   FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
         AND DateTime <= "2001-12-08 01:04:00"
         AND SysPulse = 1
         AND MyPulse = 1
         AND wwEdgeDetection = "Leading"
      ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1
2001-12-08 00:03:40.000	1	1

Compare these results with the same query using no edge detection, as shown in Edge Detection for Discrete Tags. If you look at the diagram, you might think that a row could be returned at 00:03:00, but because both tags change at exactly this instant, their values are not returned. In a normal process, it is unlikely that two tags would change at exactly at the same instant.

Trailing Edge Detection for Discrete Tags

If Trailing is specified as the parameter in the edge detection extension, the only rows in the result set are those that are the first to fail the criteria in the WHERE clause (returned false) after a row successfully met the WHERE clause criteria (returned true).

This example queries the History table. If the WHERE clause criteria specifies returning only discrete values equal to 1 (On), then applying a trailing edge detection is the same as reversing the WHERE clause (as if Value = 0 was applied).

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND Value = 1
         AND wwEdgeDetection = 'Trailing'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:40.000	MyPulse	1
2001-12-08 00:02:00.000	SysPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:03:00.000	MyPulse	1
2001-12-08 00:04:00.000	SysPulse	1

This example queries the WideHistory table. It applies a trailing edge detection returns the boundaries where the condition ceases to be true (one of the values is equal to 0).

```
SELECT DateTime, SysPulse, MyPulse FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse,
MyPulse
   FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
         AND DateTime <= "2001-12-08 01:04:00"
         AND SysPulse = 1
         AND MyPulse = 1
         AND wwEdgeDetection = "Trailing"
      ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:40.000	1	1
2001-12-08 00:04:00.000	1	1

Compare these results with the same query using no edge detection, as shown in Edge Detection for Discrete Tags. If you look at the diagram, you might think that a row could be returned at 00:03:00, but because both tags change at exactly this instant, their values are not returned. In a normal process, it is unlikely that two tags would change at exactly at the same instant.

Both Leading and Trailing Edge Detection for Discrete Tags

If Both is specified as the parameter in the edge detection extension, all rows satisfying both the leading and trailing conditions are returned.

The following queries select values of "SysPulse" and "MyPulse" that meet an edge detection of Both for a value criterion of 1 (On) from the History and WideHistory tables between 12:59 and 1:04 a.m. on December 8, 2001.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysPulse','MyPulse')
         AND DateTime > '2001-12-08 00:59:00'
         AND DateTime <= '2001-12-08 01:04:00'
         AND Value = 1
         AND wwEdgeDetection = 'Both'
```

The results are:

DateTime	TagName	Value
2001-12-08 00:01:00.000	SysPulse	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-08 00:01:00.000	MyPulse	1
2001-12-08 00:01:40.000	MyPulse	1
2001-12-08 00:02:00.000	SysPulse	1
2001-12-08 00:02:20.000	MyPulse	1
2001-12-08 00:03:00.000	SysPulse	1
2001-12-08 00:03:00.000	MyPulse	1
2001-12-08 00:03:40.000	MyPulse	1
2001-12-08 00:04:00.000	SysPulse	1

```
SELECT DateTime, SysPulse, MyPulse FROM OpenQuery(INSQL, 'SELECT DateTime, SysPulse,
MyPulse
   FROM WideHistory
      WHERE DateTime > "2001-12-08 00:59:00"
         AND DateTime <= "2001-12-08 01:04:00"
         AND SysPulse = 1
         AND MyPulse = 1
         AND wwEdgeDetection = "Both"
      ')
```

The results are:

DateTime	SysPulse	MyPulse
2001-12-08 00:01:00.000	1	1
2001-12-08 00:01:40.000	1	1
2001-12-08 00:03:40.000	1	1
2001-12-08 00:04:00.000	1	1

Compare these results with the same query using no edge detection, as shown in the Edge Detection for Discrete Tags.

Predictive Filter

AVEVA Historian supports predictive retrieval. Beginning with AVEVA Historian 2014 R2 Patch 01, the historian can return predictive data based on a "simple linear regression" (SLR) algorithm. More capabilities will be added in future releases.

With AVEVA Historian, you can create a query based on data you have stored to predict additional values in a trend. Historian returns predictive data based on a "simple linear regression" (SLR) algorithm.

For example, based on your currently stored values, you could use the predictive retrieval feature to help predict if a certain production target will be met by the end of the shift. Or, if the Historian loses communication with the data source, you could use predictive retrieval to determine whether and when a tank is likely to become empty.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

You can predict:

- Values in between other values.
- Values that extend beyond stored values.

For example, suppose you already captured data for a tag with timestamps up to 3 p.m. on a certain day, but not for the rest of the shift, which ran until 5 p.m., because of a power cut. With predictive retrieval, you can view the interpolated results based between 3 p.m. and 5 p.m. These results are based on the data you received through 3 p.m.

The following is an example of a query that retrieves stored values and reports both those values and additional predictive data:

```
SELECT DateTime, Value, wwFilter
FROM History
   WHERE TagName = 'Tag1'
      AND DateTime >= '2014-01-01 0:00:00.000'
      and DateTime < '2014-01-01 1:00:00.000'
      and wwFilter = 'SLR()'
```

In this example, "SLR" stands for "simple linear regression," the algorithm used by AVEVA Historian to analyze currently stored values and predict other values within the detected trend.

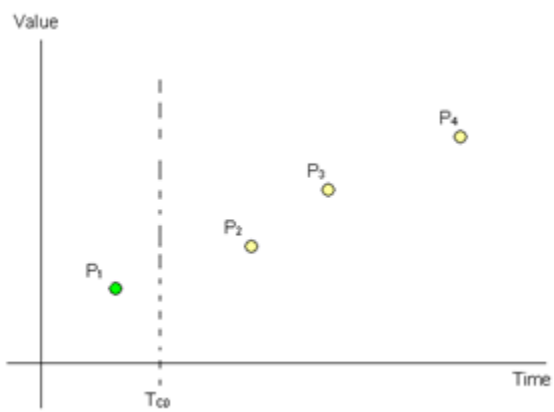
Bounding Value Retrieval

The bounding value retrieval mode returns either the start bound point or the end bound point for a requested point in time. For a start bound point, Historian retrieves the first value on or before the requested date/time. For an end bound point, Historian retrieves the first value after the requested date/time.

If no time is specified, Historian returns the bounding point at the current time.

Bounding Value Retrieval - How It Works

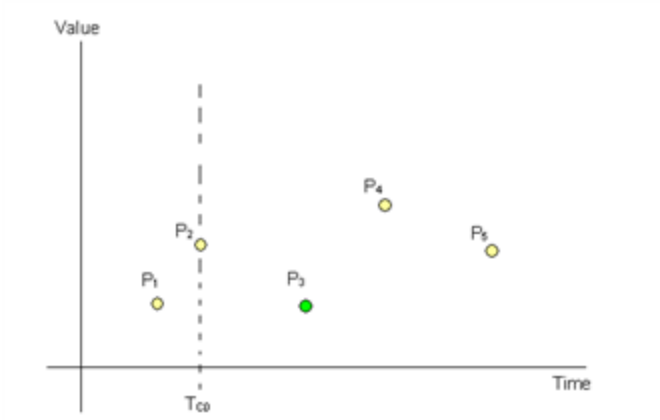
The following illustration shows how bounding value retrieval returns a start bound point:



In this case, Historian retrieves the first point on or before the datetime requested in the query. The line (T_{C0}) is the timestamp for which the start bound point is requested. P_1 is returned because that is the start or first point for the query date time.

Historian can also use bounding value retrieval to return an end bound point, as in the following illustration:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



In this case, Historian returns first point after the datetime requested in the query. T_{C0} is the timestamp for which the end bound point is requested and P_3 is returned as the ending bound point because this is the first point after the query date time.

Bounding Value Retrieval - Query Examples

You can use the Bounding Value retrieval mode to return a start bound point or an end bound point for a specified date and time. If no time is specified, Historian returns the bounding point at the current time.

To return a start bound point, set the following parameter in your query.

```
wwRetrievalMode = 'StartBound'
```

To return an end bound point, set the following parameter in your query.

```
wwRetrievalMode = 'EndBound'
```

Example 1 - Retrieve start bound point

```
select DateTime,TagName,Value
where TagName 'Plant2.R31.BatchNum'
and wwRetrievalMode = 'StartBound'
and DateTime >= '2019-04-24 12:00:00'
```

The results are:

DateTime	TagName	Value
2019-04-24 11:53:08.5430000	Plant2.R31.BatchNum	912

Example 2 - Retrieve end bound point

```
select DateTime,TagName,Value
where TagName in 'Plant2.R31.BatchNum'
and wwRetrievalMode = 'EndBound'
and DateTime >= '2019-04-24 12:00:00'
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value
2019-04-24 14:11:13.3840000	Plant2.R31.BatchNum	926

Understanding Retrieval Options

In all retrieval modes, you can adjust which values the historian returns by specifying retrieval options. The retrieval options are implemented as special parameters that you set as part of the retrieval query. This section explains each of these options. For an overview of which options apply to which retrieval modes, see Which Options Apply to Which Retrieval Modes?.

Which Options Apply to Which Retrieval Modes?

The following table shows which retrieval options apply to which modes. If you specify an option in a mode where it isn't applicable, the option is ignored.

	wwCycleCount	wwResolution	wwTimeDeadband	wwValueDeadband	wwVersion	wwInterpolationType	wwTimeStampRule	wwQualityRule	wwStateCalc	wwFilter	wwValueSelector	Predictive Filter	wwExpression	Bounding
Cyclic	Y	Y			Y		Y	Y		Y	Y		Y	
Daily	Y				Y								Y	
Delta			Y	Y	Y			Y		Y	Y	Y	Y	Y
Full					Y			Y		Y	Y	Y	Y	Y
Interpolated	Y	Y			Y	Y	Y	Y		Y	Y			
Best Fit	Y	Y			Y	Y		Y		Y	Y	Y	Y	
Average	Y	Y			Y	Y	Y	Y		Y	Y			
Minimum	Y	Y			Y			Y		Y	Y			
Maximum	Y	Y			Y			Y		Y	Y			
Integ	Y	Y			Y	Y	Y	Y		Y	Y			

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ral														
Slope					Y			Y		Y	Y			
Counter	Y	Y			Y		Y	Y		Y				
Value State	Y	Y			Y		Y	Y	Y	Y				
RoundTrip	Y	Y			Y		Y	Y	Y	Y				
Bounding	Y	Y												

Using Retrieval Options in a Transact-SQL Statement

You can retrieve data in the AVEVA Historian extension tables using normal Transact-SQL code, as well as the specialized SQL time domain extensions provided by the AVEVA Historian. The AVEVA Historian extensions provide an easy way to query time-based data from the history tables. They also provide additional functionality not supported by Transact-SQL.

Note: The wwParameters extension is reserved for future use. The wwRowCount parameter is still supported, but is deprecated in favor of wwCycleCount.

The extensions are implemented as "virtual" columns in the extension tables. When you query an extension table, you can specify values for these column parameters to manipulate the data that will be returned. You will need to specify any real-time extension parameters each time that you execute the query.

For example, you could specify a value for the wwResolution column in the query so that a resolution is applied to the returned data set:

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-02 10:00:00'
         AND DateTime <= '2001-12-02 10:02:00'
         AND Value >= 50
         AND wwResolution = 10
         AND wwRetrievalMode = 'cyclic'
```

Because the extension tables provide additional functionality that is not possible in a normal SQL Server, certain limitations apply to the Transact-SQL supported by these tables. For more information, see Unsupported or Limited Syntax Options.

Although the Microsoft SQL Server may be configured to be case-sensitive, the values for the virtual columns in the extension tables are always case-insensitive.

Note: You cannot use the IN clause or OR clause to specify more than one condition for a time domain extension. For example, "`wwVersion IN ('original', 'latest')`" and "`wwRetrievalMode = 'Delta' OR wwVersion = 'latest'`" are not supported.

For general information on creating SQL queries, see your Microsoft SQL Server documentation.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)

In retrieval modes that use cycles, the cycle count determines the number of cycles for which data is retrieved. The number of actual return values is not always identical with this cycle count. In "truly cyclic" modes (Cyclic, Interpolated, Average, and Integral), a single data point is returned for every cycle boundary. However, in other cycle-based modes (Best Fit, Minimum, Maximum, Counter, ValueState, and RoundTrip), multiple or no data points may be returned for a cycle, depending on the nature of the data.

The length of each cycle (the "resolution" of the returned values) is calculated as follows:

$$D_C = D_Q / (n - 1)$$

Where D_C is the length of the cycle, D_Q is the duration of the query, and n is the cycle count.

Instead of specifying a cycle count, you can specify the resolution. In that case, the cycle count is calculated based on the resolution and the query duration. For more information, see Resolution (Values Spaced Every X ms) (wwResolution).

This option is relevant in the following retrieval modes:

- Cyclic Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

The application of the cycle count also depends on whether you are querying a wide table. If you are querying the History table, the cycle count determines the number of rows returned per tag. For example, a query that applies a cycle count of 20 to two tags will return 40 rows of data (20 rows for each tag). If you are querying a WideHistory table, the cycle count specifies the total number of rows to be returned, regardless of how many tags were queried. For example, a query that applies a cycle count of 20 to two tags returns 20 rows of data.

Values chosen:

- If wwResolution and wwCycleCount are not specified, then a default of 100 cycles are chosen.
- If wwResolution and wwCycleCount are set to 0, then a default of 100000 cycles are chosen.
- If wwResolution and wwCycleCount are both set, then wwCycleCount is ignored.
- If wwCycleCount is specified and is less than 0, then a default of 100 cycles are chosen.
- For ValueState retrieval, if the start time of the cycle is excluded, no states are returned for the first cycle.
- For ValueState retrieval, if the end time of the cycle is excluded, no states are returned for the last cycle.

Cycle Count - Query Examples

See the following examples of queries that demonstrate the cycle count behavior if applied to a single tag or to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

multiple tags in the same query:

- Cycle Count - Query 1: Using a Single Tag
- Cycle Count - Query 2: Using Multiple Tags

Cycle Count - Query 1: Using a Single Tag

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime < '2001-12-09 11:36'
         AND wwRetrievalMode = 'Cyclic'
         AND wwCycleCount = 300
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.200	SysTimeSec	0
2001-12-09 11:35:00.400	SysTimeSec	0
2001-12-09 11:35:00.600	SysTimeSec	0
…		
2001-12-09 11:35:59.200	SysTimeSec	59
2001-12-09 11:35:59.400	SysTimeSec	59
2001-12-09 11:35:59.600	SysTimeSec	59
2001-12-09 11:35:59.800	SysTimeSec	59

Cycle Count - Query 2: Using Multiple Tags

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeMin','SysTimeSec')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime < '2001-12-09 11:36'
         AND wwRetrievalMode = 'Cyclic'
         AND wwCycleCount = 300
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.200	SysTimeMin	35
2001-12-09 11:35:00.200	SysTimeSec	0
2001-12-09 11:35:00.400	SysTimeMin	35
2001-12-09 11:35:00.400	SysTimeSec	0
2001-12-09 11:35:00.600	SysTimeMin	35
2001-12-09 11:35:00.600	SysTimeSec	0
...		
2001-12-09 11:35:59.200	SysTimeMin	35
2001-12-09 11:35:59.200	SysTimeSec	59
2001-12-09 11:35:59.400	SysTimeMin	35
2001-12-09 11:35:59.400	SysTimeSec	59
2001-12-09 11:35:59.600	SysTimeMin	35
2001-12-09 11:35:59.600	SysTimeSec	59
2001-12-09 11:35:59.800	SysTimeMin	35
2001-12-09 11:35:59.800	SysTimeSec	59

Notice that the values of the two tags are mixed together in the same column.

Resolution (Values Spaced Every X ms) (wwResolution)

In retrieval modes that use cycles, the resolution is the sampling interval for retrieving data, that is, the length of each cycle.

The number of cycles, therefore, depends on the time period and the resolution:

number of cycles = time period / resolution

The number of actual return values is not always identical with this cycle count. In "truly cyclic" modes (Cyclic, Interpolated, Average, and Integral), a single data point is returned for every cycle boundary. However, in other cycle-based modes (Best Fit, Minimum, Maximum, Counter, and ValueState), multiple or no data points may be returned for a cycle, depending on the nature of the data.

Note: The rowset is guaranteed to contain one row for each tag in the normalized query at every resolution interval, regardless of whether a physical row exists in history at that particular instance in time. The value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

contained in the row is the last known physical value in history, at that instant, for the relevant tag.

Instead of specifying a resolution, you can specify the cycle count directly. In that case, the resolution is calculated based on the cycle count and the query duration. For more information, see Cycle Count (X Values over Equal Time Intervals) (wwCycleCount).

This option is relevant in the following retrieval modes:

- Cyclic Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

For delta retrieval, you can achieve similar results by using a time deadband. For more information, see Time Deadband (wwTimeDeadband).

Resolution - Query Example

The following query returns rows that are spaced at 2 sec (2000 msec) intervals over the requested time period. Data is retrieved cyclically.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeMin','SysTimeSec')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Cyclic'
         AND wwResolution = 2000
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:02.000	SysTimeMin	35
2001-12-09 11:35:02.000	SysTimeSec	2
2001-12-09 11:35:04.000	SysTimeMin	35
2001-12-09 11:35:04.000	SysTimeSec	4

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:06.000	SysTimeMin	35
...		
2001-12-09 11:35:54.000	SysTimeMin	35
2001-12-09 11:35:54.000	SysTimeSec	54
2001-12-09 11:35:56.000	SysTimeMin	35
2001-12-09 11:35:56.000	SysTimeSec	56
2001-12-09 11:35:58.000	SysTimeMin	35
2001-12-09 11:35:58.000	SysTimeSec	58
2001-12-09 11:36:00.000	SysTimeMin	36
2001-12-09 11:36:00.000	SysTimeSec	0

About Phantom Cycles

The phantom cycle is the name given to the cycle that leads up to the query start time. It is used to calculate which initial value to return at the query start time for all retrieval modes. Some retrieval modes use the phantom cycle to simply find the last known value prior to the query start time, whereas other retrieval modes use the entire cycle to calculate aggregates. The different uses of the phantom cycle can be seen in the following table.

Simple use of phantom cycle	Cycles not defined, but similar simple use of time before query start time	Phantom cycle used to calculate aggregates
Cyclic	Delta	Min
Interpolated	Full	Max
Best Fit	Slope	Average
		Integral
		Counter
		ValueState
		RoundTrip

It's common to expect a single aggregate row returned for a particular time interval. You can accomplish this in several ways.

The following example is querying for hourly averages. It returns a single row time stamped at the query start time. If the query included the query end point by including an equal sign for it, the query would also have

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

returned an additional row at the query end time.

```
SELECT DateTime, Value, Quality, QualityDetail, OPCQuality
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime >= '2017-12-12 08:00:00'
         AND DateTime < '2017-12-12 09:00:00'
         AND wwRetrievalMode = 'Avg'
         AND wwResolution = 3600000
```

The results are:

DateTime	Value	Quality	QualityDetail	OPCQuality
2017-12-1208:00:00.0000000	29.5	0	192	192

What may be confusing in this example is the calculation of the average in the returned row for the phantom cycle leading up to the query start time. The query specifies a positive one hour time interval between the query start time and the query end time. You may therefore expect that the calculated and returned average should be for the specified interval.

However, the time difference between start and end time in the above query is not actually required because the resolution is provided explicitly (wwResolution = 36000000). If the query specified an end time equal to the specified start time and if it included the equal sign for the end time, the query would still return the same single row of data.

```
SELECT DateTime, Value, Quality, QualityDetail, OPCQuality
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime = '2017-10-16 08:00:00'
         AND DateTime <= '2017-10-16 09:00:00'
         AND wwRetrievalMode = 'Avg'
         AND wwCycleCount = 1
```

The results are:

DateTime	Value	Quality	QualityDetail	OPCQuality
2017-10-16 08:00:00.0000000	29.5	0	192	192

This second example also asks for hourly averages and it also returns only a single row of data stamped at the query start time. This query, however, must specify a time difference between the start and end time, because the resolution is not explicitly defined in the query.

As in the preceding query, the specified interval and cycle count of 1 may look like the returned row has been calculated for the specified interval, but the returned row is once again for the phantom cycle leading up to the start time.

For some queries, you may want to be certain to include values on a cycle boundary. For example, the following query is looking for a minimum value within a cycle. In this query, the beginning DateTime statement uses ">=" to ensure that the entire cycle is queried. Even if the minimum value happens to be at the beginning of the cycle, the following query will provide an accurate result:

```
   SELECT StartDateTime, *
      FROM History
         WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2016-03-31 15:41:10'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND DateTime < '2016-03-31 15:41:20'
        AND wwRetrievalMode = 'Min'
        AND Quality <> 133
        AND wwCycleCount = 1
```

The StartDateTime makes it easier to see which time interval was used to calculate the returned aggregate. This column returns the time stamp of the beginning of the cycle used for the aggregate calculation. The time stamp is always returned in accordance with the specified time zone and always has the same offset as the time stamp returned in the DateTime column, even when the two time stamps are on different sides of a DST change.

Assuming results are timestamped at the end of the cycle (as is done by default when wwTimeStampRule is set to END), the initial rows in the examples above would return a DateTime equal to '2009-10-16 08:00:00', and the StartDateTime column would return '2009-10-16 07:00:00' making it easy to interpret the result.

If instead the query were to ask for results time stamped at the beginning of the cycle with wwTimeStampRule set to START, the initial rows in the same examples would still return a DateTime equal to '2009-10-16 08:00:00', but the time stamp has now been shifted in accordance with the time stamp request. The result is therefore calculated for the specified time interval between 8 a.m. and 9 a.m. In this example, the new StartDateTime column would return the same time stamp as the DateTime column, '2009-10-16 08:00:00', again making it easier to interpret the result.

For retrieval modes for which cycles are defined, the StartDateTime column returns the cycle start time. These modes are:

- Cyclic Retrieval
- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Minimum Retrieval
- Maximum Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

In the remaining retrieval modes, the StartDateTime column returns the same time stamp as the DateTime column.

For an additional example, see Querying Aggregate Data in Different Ways.

Value Deadband (wwValueDeadband)

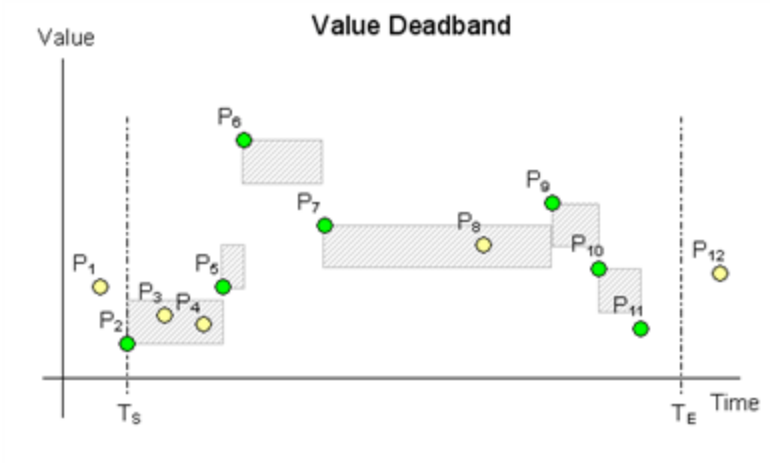
A value deadband for retrieval controls the value resolution of data returned in delta mode. Any data values that change less than the specified deadband are not returned. The deadband is a percentage of a tag's full scale in engineering units.

The deadband "base value" is reset each time that a value is returned, so that the last value returned acts as the basis for the deadband.

Changes in quality will force a value to be returned even if the value deadband has not been met.

The following illustration shows an example of applying a value deadband:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Data is retrieved for the time period starting with T_S and ending with T_E. All points in the graphic represent data values stored on the historian. The gray areas represent the value deadband, which starts anew with every returned value. Only the green points (P_2, P_5, P_6, P_7, P_9, P_{10}, P_{11}) are returned. The other points are not returned because they fall within a deadband.

Value Deadband - Query Examples

See the following examples of queries that use value deadband:

- Value Deadband - Query 1
- Value Deadband - Query 2

Note: Each of these examples returns data values for the analog tag 'SysTimeSec'. The minimum engineering unit for 'SysTimeSec' is 0, and the maximum engineering unit is 59.

Value Deadband - Query 1

This query specifies to return only data that changed by more than 10 percent of the tag's full engineering unit range. Using a value deadband of 10 percent equates to an absolute change of 5.9 for 'SysTimeSec.'

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwValueDeadband = 10
```

The results are:

DateTime	Value
2001-12-09 11:35:00.000	0
2001-12-09 11:35:06.000	6

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:12.000	12
2001-12-09 11:35:18.000	18
2001-12-09 11:35:24.000	24
2001-12-09 11:35:30.000	30
2001-12-09 11:35:36.000	36
2001-12-09 11:35:42.000	42
2001-12-09 11:35:48.000	48
2001-12-09 11:35:54.000	54
2001-12-09 11:36:00.000	0
2001-12-09 11:36:06.000	6
2001-12-09 11:36:12.000	12
2001-12-09 11:36:18.000	18
2001-12-09 11:36:24.000	24
2001-12-09 11:36:30.000	30
2001-12-09 11:36:36.000	36
2001-12-09 11:36:42.000	42
2001-12-09 11:36:48.000	48
2001-12-09 11:36:54.000	54
2001-12-09 11:37:00.000	0

Value Deadband - Query 2

This query specifies to only return data that changed by more than 5 percent of the tag's full engineering unit range. Using a value deadband of 5 percent equates to an absolute change of 2.95 for 'SysTimeSec.'

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwValueDeadband = 5
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	Value
2001-12-09 11:35:00.000	0
2001-12-09 11:35:03.000	3
2001-12-09 11:35:06.000	6
2001-12-09 11:35:09.000	9
2001-12-09 11:35:12.000	12
2001-12-09 11:35:15.000	15
2001-12-09 11:35:18.000	18
2001-12-09 11:35:21.000	21
2001-12-09 11:35:24.000	24
2001-12-09 11:35:27.000	27
2001-12-09 11:35:30.000	30
2001-12-09 11:35:33.000	33
2001-12-09 11:35:36.000	36
2001-12-09 11:35:39.000	39
2001-12-09 11:35:42.000	42
2001-12-09 11:35:45.000	45
2001-12-09 11:35:48.000	48
2001-12-09 11:35:51.000	51
2001-12-09 11:35:54.000	54
2001-12-09 11:35:57.000	57
2001-12-09 11:36:00.000	0
2001-12-09 11:36:03.000	3
2001-12-09 11:36:06.000	6
2001-12-09 11:36:09.000	9
2001-12-09 11:36:12.000	12
2001-12-09 11:36:15.000	15

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:36:18.000	18
2001-12-09 11:36:21.000	21
2001-12-09 11:36:24.000	24
2001-12-09 11:36:27.000	27
2001-12-09 11:36:30.000	30
2001-12-09 11:36:33.000	33
2001-12-09 11:36:36.000	36
2001-12-09 11:36:39.000	39
2001-12-09 11:36:42.000	42
2001-12-09 11:36:45.000	45
2001-12-09 11:36:48.000	48
2001-12-09 11:36:51.000	51
2001-12-09 11:36:54.000	54
2001-12-09 11:36:57.000	57
2001-12-09 11:37:00.000	0

Time Deadband (wwTimeDeadband)

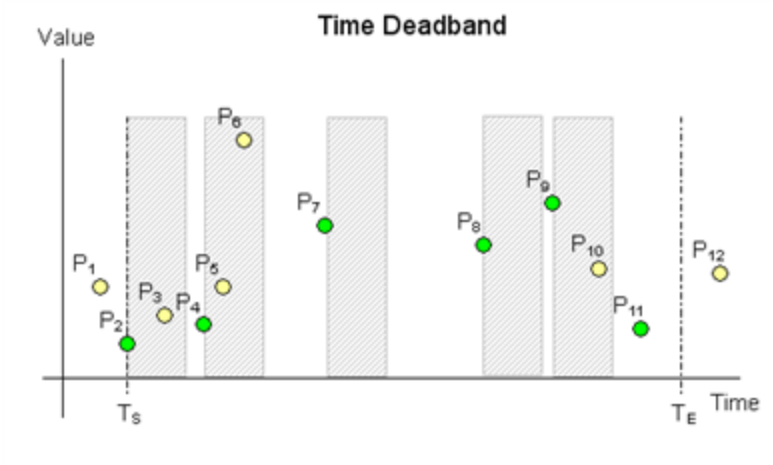
A time deadband for retrieval controls the time resolution of data returned in delta mode. Any value changes that occur within the time deadband are not returned.

Time deadbands can be applied to analog, discrete, and string tags.

The deadband "base value" is reset each time that a value is returned, so that the last value returned acts as the basis for the deadband.

The following illustration shows an example of applying a time deadband:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Data is retrieved for the time period starting with T_S and ending with T_E. All points in the graphic represent data values stored on the historian. The gray areas represent the time deadband, which starts anew with every returned value. Only the green points (P_2, P_4, P_7, P_8, P_9, P_{11}) are returned. The other points are not returned because they fall within a deadband.

Time Deadband - Query Examples

To apply a time deadband, set the wwTimeDeadband parameter in your query.

For examples, see the following:

- Time Deadband - Query 1
- Time Deadband - Query 2
- Time Deadband - Query 3

Note: All of these example queries return data values for the analog tag 'SysTimeSec'.

Time Deadband - Query 1

This query specifies to only return data that changed during a 5 second time deadband.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 5000
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:06.000	SysTimeSec	6

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:12.000	SysTimeSec	12
2001-12-09 11:35:18.000	SysTimeSec	18
2001-12-09 11:35:24.000	SysTimeSec	24
2001-12-09 11:35:30.000	SysTimeSec	30
2001-12-09 11:35:36.000	SysTimeSec	36
2001-12-09 11:35:42.000	SysTimeSec	42
2001-12-09 11:35:48.000	SysTimeSec	48
2001-12-09 11:35:54.000	SysTimeSec	54
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:06.000	SysTimeSec	6
2001-12-09 11:36:12.000	SysTimeSec	12
2001-12-09 11:36:18.000	SysTimeSec	18
2001-12-09 11:36:24.000	SysTimeSec	24
2001-12-09 11:36:30.000	SysTimeSec	30
2001-12-09 11:36:36.000	SysTimeSec	36
2001-12-09 11:36:42.000	SysTimeSec	42
2001-12-09 11:36:48.000	SysTimeSec	48
2001-12-09 11:36:54.000	SysTimeSec	54
2001-12-09 11:37:00.000	SysTimeSec	0

Time Deadband - Query 2

This query specifies to only return data that changed during a 4900 millisecond time deadband.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:37'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 4900
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:05.000	SysTimeSec	5
2001-12-09 11:35:10.000	SysTimeSec	10
2001-12-09 11:35:15.000	SysTimeSec	15
2001-12-09 11:35:20.000	SysTimeSec	20
2001-12-09 11:35:25.000	SysTimeSec	25
2001-12-09 11:35:30.000	SysTimeSec	30
2001-12-09 11:35:35.000	SysTimeSec	35
2001-12-09 11:35:40.000	SysTimeSec	40
2001-12-09 11:35:45.000	SysTimeSec	45
2001-12-09 11:35:50.000	SysTimeSec	50
2001-12-09 11:35:55.000	SysTimeSec	55
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:05.000	SysTimeSec	5
2001-12-09 11:36:10.000	SysTimeSec	10
2001-12-09 11:36:15.000	SysTimeSec	15
2001-12-09 11:36:20.000	SysTimeSec	20
2001-12-09 11:36:25.000	SysTimeSec	25
2001-12-09 11:36:30.000	SysTimeSec	30
2001-12-09 11:36:35.000	SysTimeSec	35
2001-12-09 11:36:40.000	SysTimeSec	40
2001-12-09 11:36:45.000	SysTimeSec	45
2001-12-09 11:36:50.000	SysTimeSec	50
2001-12-09 11:36:55.000	SysTimeSec	55
2001-12-09 11:37:00.000	SysTimeSec	0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Time Deadband - Query 3

This query specifies to only return data that changed during a 2000 millisecond time deadband.

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName IN ('SysTimeSec','SysTimeMin')
         AND DateTime >= '2001-12-09 11:35'
         AND DateTime <= '2001-12-09 11:36'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 2000
```

The results are:

DateTime	TagName	Value
2001-12-09 11:35:00.000	SysTimeSec	0
2001-12-09 11:35:00.000	SysTimeMin	35
2001-12-09 11:35:03.000	SysTimeSec	3
2001-12-09 11:35:06.000	SysTimeSec	6
2001-12-09 11:35:09.000	SysTimeSec	9
2001-12-09 11:35:12.000	SysTimeSec	12
2001-12-09 11:35:15.000	SysTimeSec	15
2001-12-09 11:35:18.000	SysTimeSec	18
2001-12-09 11:35:21.000	SysTimeSec	21
2001-12-09 11:35:24.000	SysTimeSec	24
2001-12-09 11:35:27.000	SysTimeSec	27
2001-12-09 11:35:30.000	SysTimeSec	30
2001-12-09 11:35:33.000	SysTimeSec	33
2001-12-09 11:35:36.000	SysTimeSec	36
2001-12-09 11:35:39.000	SysTimeSec	39
2001-12-09 11:35:42.000	SysTimeSec	42
2001-12-09 11:35:45.000	SysTimeSec	45
2001-12-09 11:35:48.000	SysTimeSec	48
2001-12-09 11:35:51.000	SysTimeSec	51

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-09 11:35:54.000	SysTimeSec	54
2001-12-09 11:35:57.000	SysTimeSec	57
2001-12-09 11:36:00.000	SysTimeSec	0
2001-12-09 11:36:00.000	SysTimeMin	36

History Version (wwVersion)

The AVEVA Historian allows you to overwrite a stored tag value with later versions of the value. The original version of the value is still maintained, so that effectively, multiple versions of the tag value exist at the same point in time.

When retrieving data, you can specify whether to retrieve the originally stored version or the latest version that is available. To do this, set the history version option to "Original" for the original version or "Latest" for the latest available version. If you do not specify the version, the latest version is returned.

To distinguish between a latest value and an original value, the historian returns a special QualityDetail value of 202 for a latest point with good quality. Note, however, that replicated data may show a QualityDetail of 202 for points other than the latest point.

This option is relevant in all retrieval modes.

History Version - Query Example

This example illustrates using history version. First, consider this query:

```
SELECT TagName, DateTime, Value, wwVersion
   FROM History
      WHERE TagName IN ('SysTimeHour', 'SysTimeMin')
         AND DateTime >= '2001-12-20 0:00'
         AND DateTime <= '2001-12-20 0:05'
         AND wwRetrievalMode = 'Delta'
         AND wwVersion = 'Original'
```

The results are:

TagName	DateTime	Value	wwVersion
SysTimeMin	2001-12-20 00:00:00.000	0	ORIGINAL
SysTimeHour	2001-12-20 00:00:00.000	0	ORIGINAL
SysTimeMin	2001-12-20 00:01:00.000	1	ORIGINAL
SysTimeMin	2001-12-20 00:02:00.000	2	ORIGINAL
SysTimeMin	2001-12-20	3	ORIGINAL

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	00:03:00.000		
SysTimeMin	2001-12-20 00:04:00.000	4	ORIGINAL
SysTimeMin	2001-12-20 00:05:00.000	5	ORIGINAL

When retrieving the latest version, the wwVersion parameter always returns with a value of LATEST for all values, even though many of the values may actually be the original values that came from the I/O Server. To distinguish between an actual latest value and an original value, a special QualityDetail of 202 is returned for a good, latest point. Note, however, that replicated data may show a QualityDetail of 202 for points other than the latest point.

For example:

```
SELECT DateTime, Value, Quality, QualityDetail, OPCQuality, wwVersion
   FROM History
      WHERE TagName IN ('PV')
         AND DateTime >= '2005-04-17 11:35:00'
         AND DateTime <= '2005-04-17 11:36:00'
         AND wwRetrievalMode = 'Delta'
         AND wwVersion = 'Latest'
```

The results are:

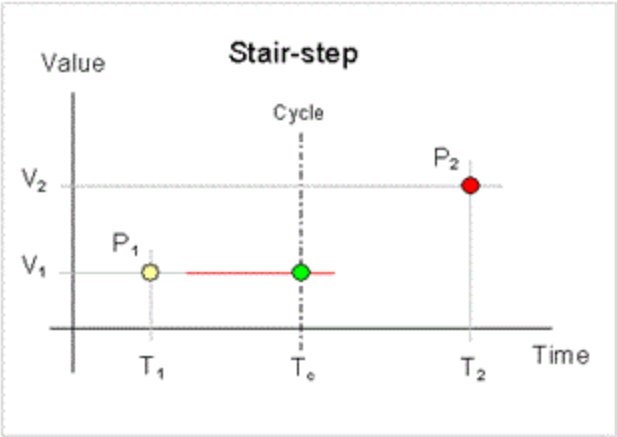
DateTime	Value	Quality	QualityDetail	OPCQuality	wwVersion
2005-04-17 11:35:00.000	12.5	0	192	192	LATEST
2005-04-17 11:35:15.000	17.3	0	192	192	LATEST
2005-04-17 11:35:30.000	34.0	0	202	192	LATEST
2005-04-17 11:35:45.000	43.1	0	192	192	LATEST
2005-04-17 11:36:00.000	51.2	0	192	192	LATEST

Interpolation Type (wwInterpolationType)

For various retrieval modes, you can control how analog tag values at cycle boundaries are calculated if there is no actual value stored at that point in time. The options are as follows:

- **Stairstep:** No interpolation occurs. The value at the cycle boundary is assumed to be the same value as the last stored value before the cycle boundary.The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned.

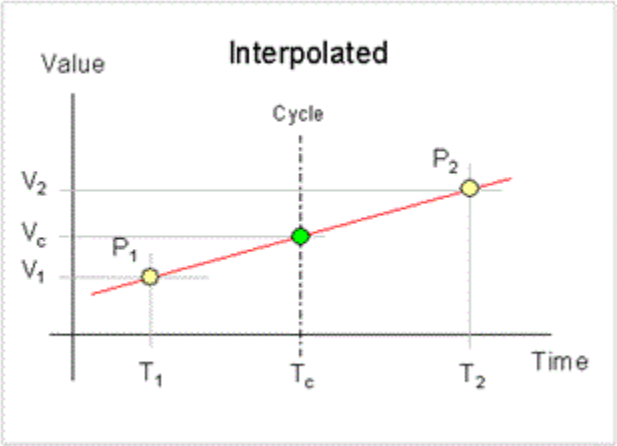
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- **Linear:** The historian calculates a new value at the cycle boundary by interpolating between the last stored value before the boundary and the first stored value after the boundary. If either of these values is NULL, it returns the last stored value before the boundary.

  Expressed as a formula, V_c is calculated as:

  $V_c = V_1 + ((V_2 - V_1) * ((T_c - T_1) / (T_2 - T_1)))$



The type of data that you want to retrieve usually determines the interpolation type to use. For example, if you have a thermocouple, the temperature change is linear, so it's best to use linear interpolation. If you have a tag that contains discrete measurements, such as a set point, then you probably want to use stair-stepped values. In general, it is recommended that you use linear interpolation as the general setting, and use stair-stepped values for the exceptions.

This option is relevant in the following retrieval modes:

- Interpolated Retrieval
- Best Fit Retrieval
- Average Retrieval
- Integral Retrieval

The quality of an interpolated point is determined by the wwQualityRule setting. For more information, see Quality Rule (wwQualityRule).

The interpolation type can be set on three levels:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- The AVEVA Historian system-wide setting. The system-wide setting must be either stair-step or interpolated. For more information, see "System Parameters" on page 36. This setting is configured using the AVEVA Historian Configuration Editor.
- The individual analog tag setting. You can configure an individual analog tag to use the system-wide setting or either stair-stepped values or linear interpolation. The individual tag setting will override the system-wide setting. This setting is configured using the AVEVA Historian Configuration Editor.
- The setting for the wwInterpolationType parameter in the query. This setting overrides any other setting for all tags in the query.

The wwInterpolationType parameter is dynamically used both for input for the query, when you need to override the individual tag settings, and for output for each individual row to show whether a particular row value was calculated using linear interpolation (returned as "LINEAR") or if it is a stair-stepped value (returned as "STAIRSTEP").

To force a query to always use linear interpolation whenever applicable, specify the following in the query:

```
AND wwInterpolationType = 'Linear'
```

To force a query to always return stair-stepped values, specify the following in the query:

```
AND wwInterpolationType = 'StairStep'
```

TimeStamp Rule (wwTimeStampRule)

For various cycle-based retrieval modes, you can control whether the returned values are timestamped at the beginning or at the end of each cycle.

To force a query to timestamp results at the start of a cycle, specify the following in the query:

```
AND wwTimeStampRule = 'Start'
```

To force a query to timestamp results at the end of a cycle, specify the following in the query:

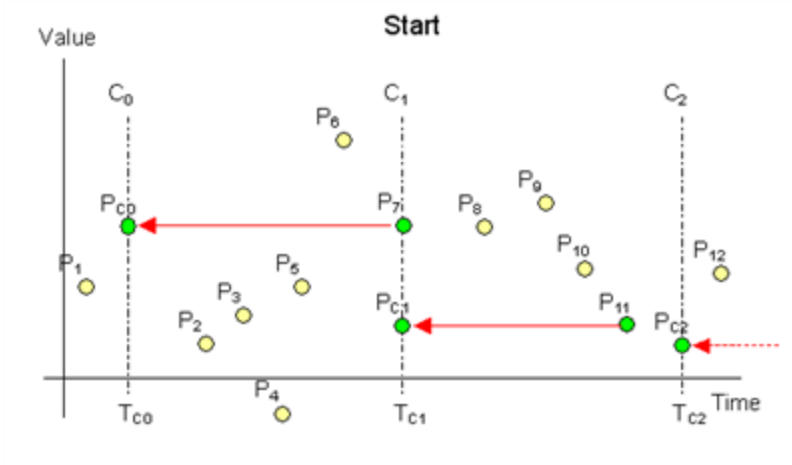
```
AND wwTimeStampRule = 'End'
```

If you include the wwTimeStampRule column in your SELECT statement, it will show which timestamp rule has been applied for the individual row, if applicable.

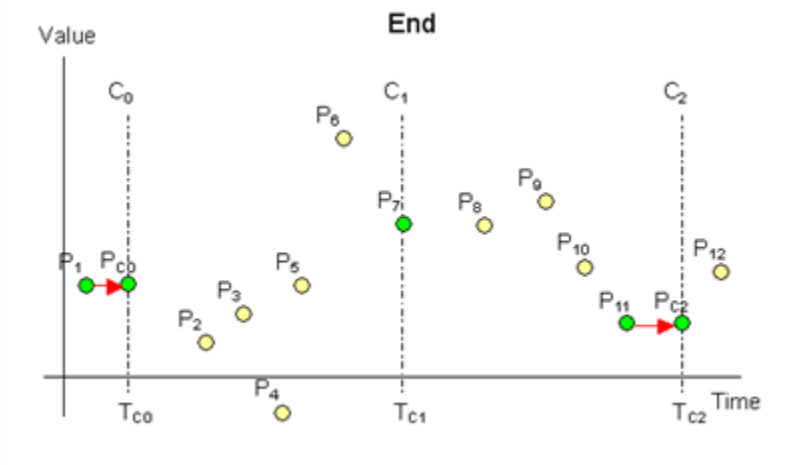
The options are as follows:

- **Start:** The value for a given cycle is stamped with the cycle start time. For example, in the following illustration of a cyclic query, the following values are returned at the cycle boundaries:
  - At T_{C0}: P_7, because it falls on the cycle boundary. In cyclic mode, if there is a value right on the cycle boundary, it is considered to belong to the cycle before the boundary. In this case, this is the cycle starting at T_{C0} and ending at T_{C1}, and because the Start timestamp rule is used, the value is timestamped at T_{C0}.
  - At T_{C1}: P_{11}, because it is the last value in the cycle starting at T_{C1} and ending at T_{C2}
  - At T_{C2}: The last value in the "phantom" cycle starting at T_{C2}

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



- **End:** The value for a given cycle is stamped with the cycle end time. For example, in the following illustration of a cyclic query, the following values are returned at the cycle boundaries:
  - At T_{C0}: P_1, because it is the last value in the "phantom" cycle ending at T_{C0}. Because the End timestamp rule is used, the value is timestamped at T_{C0}.
  - At T_{C1}: P_7, because it falls on the cycle boundary. In cyclic mode, if there is a value right on the cycle boundary, it is considered to belong to the cycle before the boundary. In this case, this is the cycle starting at T_{C0} and ending at T_{C1}, and because the End timestamp rule is used, the value is timestamped at T_{C1}.
  - At T_{C2}: P_{11}, because it is the last value in the cycle ending at T_{C2}



- **Server default:** Either Start or End is used, depending on the system parameter setting on the AVEVA Historian.

This option is relevant in the following retrieval modes:

- Cyclic Retrieval (AVEVA Historian 9.0 and later)
- Interpolated Retrieval
- Average Retrieval
- Integral Retrieval
- Counter Retrieval

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- ValueState Retrieval
- RoundTrip Retrieval

Time Zone (wwTimeZone)

For AVEVA Historianversion 8.0 and later, all history data is stored in Coordinated Universal Time (UTC). The wwTimeZone extension allows you to specify the time zone to be used for the timestamps of the returned data values. The Retrieval subsystem will convert the timestamps to local time in the specified time zone.

The wwTimeZone extension may be assigned any of the values stored in the TimeZone column of the TimeZone table in the Runtime database.

The TimeZone table is repopulated at every system startup from Microsoft operating system registry entries, and will therefore reflect the time zones available from the server operating system, including any new or custom time zones which might be added by operating system service packs or installed software.

The Retrieval subsystem will automatically correct for daylight savings time in the requested time zone. When computing daylight savings and time zone parameters, the settings of the server operating system are used. The Retrieval subsystem does not provide any means for using client-side settings.

If wwTimeZone is not specified, the time zone for retrieval defaults to the time zone of the AVEVA Historian computer.

For example:

```
SELECT TagName, DateTime, Value, wwTimeZone
   FROM History
      WHERE TagName IN ('SysTimeHour', 'SysTimeMin')
         AND DateTime >= '2001-12-20 0:00'
         AND DateTime <= '2001-12-20 0:05'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeZone = 'W. Europe Standard Time'
```

The results are:

TagName	DateTime	Value	wwTimeZone
SysTimeMin	2001-12-20 00:00:00.000	0	W. Europe Standard Time
SysTimeHour	2001-12-20 00:00:00.000	15	W. Europe Standard Time
SysTimeMin	2001-12-20 00:01:00.000	1	W. Europe Standard Time
SysTimeMin	2001-12-20 00:02:00.000	2	W. Europe Standard Time
SysTimeMin	2001-12-20 00:03:00.000	3	W. Europe Standard Time
SysTimeMin	2001-12-20 00:04:00.000	4	W. Europe Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SysTimeMin	2001-12-20 00:05:00.000	5	W. Europe Standard Time

Quality Rule (wwQualityRule)

The quality rule can be used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC.

- A quality rule of GOOD means that data values with uncertain (64) OPC quality are not used in the retrieval calculations and are ignored. Values with bad QualityDetail indicate gaps in the data.
- A quality rule of EXTENDED means that data values with both good and uncertain OPC quality are used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data.
- A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values do not cause the overall calculations to return NULL.

You can apply a quality rule to all retrieval modes.

The OPTIMISTIC setting for the quality rule lets you retrieve information that is possibly incomplete but may nevertheless provide better results where the calculation cycle contains data gaps. This setting calculates using the last known good value prior to the gap (if possible). The logic for determining the quality of the points returned remains unchanged. The integral retrieval mode is an exception to this where the integral is scaled up to cover gaps. For more information, see Integral Retrieval.

The following figure shows a counter retrieval situation in which three of the four shown cycle boundaries are located in data gaps. Without using OPTIMISTIC, counter queries would return a NULL at all cycle boundaries because the mode needs valid good values at each end of the cycle calculate a precise difference.

filter_counter_query

If the query were to specify OPTIMISTIC, the counter mode will always return rows with numeric counter values and good quality. These rows may or may not be precise. The PercentGood column of the row returns the percentage of time in each cycle in which retrieval was able to find values stored with good quality, so if the PercentGood is anything less than 100, then the returned row may be incorrect. Quality is returned as uncertain if percent good is not 100 percent.

Now look at the counter values that are returned using OPTIMISTIC quality in the preceding illustration. The query skips the value to be returned at the first cycle boundary, because there is not enough information about the cycle prior to that boundary. At the second cycle boundary, the value 0 will be returned, because there was a gap in the data for the entire first cycle. In the second cycle, there are two points, P1 and P2. The query uses P2 as the end value of the cycle and infers a start value of the cycle from P1. At the third cycle boundary, T_{c2}, the query returns P2 – P1. Similarly, at the last cycle boundary, the query returns P4 – P3.

For the integral retrieval mode, the query does not summarize data for gaps because there is no way to know which value to use for the summarization. However, if the query specifies OPTIMISTIC quality, the query uses the last known good value for the summarization in the gap. As described for the counter retrieval example, the PercentGood column also expresses the quality of the calculated value in integral retrieval, so if the PercentGood is anything less than 100, then the returned row may be incorrect.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

To force a query to exclude points with doubtful OPC quality, specify the following in the query:

```
   AND wwQualityRule = 'Good'
```

To force a query to use points with both good and doubtful OPC quality, specify the following in the query:

```
   AND wwQualityRule = 'Extended'
```

For examples, see the following:

- Quality Rule - Query 1
- Quality Rule - Query 2
- Quality Rule - Query 3
- Quality Rule - Query 4
- Quality Rule - Query 5
- Quality Rule - Query 6
- Quality Rule - Query 7
- Quality Rule - Query 8

Quality Rule - Query 1

If you include the wwQualityRule column in a SELECT statement, it will show which quality rule was used for the individual row, if applicable.

You can combine OPC qualities in a query. For example, if you combine a mixture of good OPC qualities (such as 192 to 219), a good OPC quality (192) will be returned as a combined result.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
         WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Avg'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	193
I0R5	2009-09-12 00:14	3	195
I0R5	2009-09-12 00:22	0	196
I0R5	2009-09-12 00:25	1	199
I0R5	2009-09-12 00:27	0	200
I0R5	2009-09-12 00:29	2	207

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

I0R5	2009-09-12 00:33	3	215
I0R5	2009-09-12 00:36	0	216
I0R5	2009-09-12 00:39	1	219

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
I0R5	2009-09-12 00:20	2.6	192	192	AVERAGE
I0R5	2009-09-12 00:30	1.0	192	192	AVERAGE
I0R5	2009-09-12 00:40	1.6	192	192	AVERAGE

Quality Rule - Query 2

Similar to Quality Rule - Query 1, if you combine a mixture of doubtful OPC qualities, a doubtful OPC quality (64) will be returned as the combined OPC quality.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
         WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Integral'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	65
I0R5	2009-09-12 00:14	3	68
I0R5	2009-09-12 00:22	0	71
I0R5	2009-09-12 00:25	1	74
I0R5	2009-09-12 00:27	0	79
I0R5	2009-09-12 00:29	2	80
I0R5	2009-09-12 00:33	3	88
I0R5	2009-09-12 00:36	0	92

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

I0R5	2009-09-12 00:39	1	64

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
I0R5	00:20	26.0	64	64	INTEGRAL
I0R5	00:30	10.0	64	64	INTEGRAL
I0R5	00:40	16.0	64	64	INTEGRAL

Quality Rule - Query 3

When you combine the same OPC quality then that OPC quality will be returned. However, when there is no good point in a cycle for cyclic modes such as Integral, Average, Counter, or AnalogSummary, the returned NULL value will have an OPC quality of 0 and a Quality Detail of 65536, regardless of combined qualities.

```
SELECT TagName, StartDateTime, EndDateTime, OPCQuality, PercentGood, wwRetrievalMode, first
   FROM AnalogSummaryHistory
         WHERE TagName = 'F0R5'
         AND StartDateTime >= '2009-09-12 00:20'
         AND EndDateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Cyclic'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
F0R5	2009-09-12 00:07	1.6	78
F0R5	2009-09-12 00:14	3.1	78
F0R5	2009-09-12 00:22	0.2	78
F0R5	2009-09-12 00:25	0.8	78
F0R5	2009-09-12 00:27	0.4	78
F0R5	2009-09-12 00:29	2.2	78
F0R5	2009-09-12 00:33	3.3	78
F0R5	2009-09-12 00:36	0.3	78
F0R5	2009-09-12 00:39	1.2	78

The results are:

Tagname	StartDate	EndDate	OPCQuality	PercentGood	wwRetrieval	first

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Time	Time			Mode	
F0R5	2009-09-12 00:20	2009-09-12 00:30	78	100	CYCLIC	0.200
F0R5	2009-09-12 00:30	2009-09-12 00:40	78	100	CYCLIC	3.300

Quality Rule - Query 4

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
      WHERE TagName = 'F0R5'
      AND DateTime >= '2009-09-12 00:20'
      AND DateTime <= '2009-09-12 00:40'
      AND wwResolution = 10000
      AND wwRetrievalMode = 'Avg'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
F0R5	2009-09-12 00:07	1.6	15
F0R5	2009-09-12 00:14	3.1	15
F0R5	2009-09-12 00:22	0.2	15
F0R5	2009-09-12 00:25	0.8	15
F0R5	2009-09-12 00:27	0.4	15
F0R5	2009-09-12 00:29	2.2	15
F0R5	2009-09-12 00:33	3.3	15
F0R5	2009-09-12 00:36	0.3	15
F0R5	2009-09-12 00:39	1.2	15

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
F0R5	2009-09-12 00:20	NULL	65536	0	AVERAGE
F0R5	2009-09-12 00:30	NULL	65536	0	AVERAGE

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

F0R5	2009-09-12 00:40	NULL	65536	0	AVERAGE

Quality Rule - Query 5

When you combine a mixture of good, bad, and uncertain OPC qualities, a doubtful OPC quality (64) will be returned as a combined result.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode
   FROM History
      WHERE TagName = 'F0R5'
      AND DateTime >= '2009-09-12 00:20'
      AND DateTime <= '2009-09-12 00:40'
      AND wwResolution = 10000
      AND wwRetrievalMode = 'Avg'
      AND wwQualityRule = 'Optimistic'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
F0R5	2009-09-12 00:07	1.6	15
F0R5	2009-09-12 00:14	3.1	69
F0R5	2009-09-12 00:22	0.2	78
F0R5	2009-09-12 00:25	0.8	200
F0R5	2009-09-12 00:27	0.4	15
F0R5	2009-09-12 00:29	2.2	92
F0R5	2009-09-12 00:33	3.3	88
F0R5	2009-09-12 00:36	0.3	199
F0R5	2009-09-12 00:39	1.2	196

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality	wwRetrievalMode
F0R5	2009-09-12 00:20	2.012	64	64	AVERAGE
F0R5	2009-09-12 00:30	0.820	64	64	AVERAGE
F0R5	2009-09-12 00:40	1.751	64	64	AVERAGE

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Quality Rule - Query 6

For RoundTrip, StateSummary, and ValueState modes, the OPC qualities are only combined with the same state in a cycle. If the state only occurs once in a cycle, then the qualities of that state will be returned. The returned NULL state will always have an OPC quality of 0 and Quality Detail of 65536. The same qualities are returned for a state that has no roundtrip in RoundTrip mode.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, StateTime
   FROM History
      WHERE TagName = 'I001'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'RoundTrip'
         AND wwStateCalc = 'MaxContained'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I001	2009-09-12 00:12	1	90
I001	2009-09-12 00:15	2	65
I001	2009-09-12 00:22	1	85
I001	2009-09-12 00:23	2	75
I001	2009-09-12 00:26	1	75
I001	2009-09-12 00:29	2	70

The results are:

Tagname	DateTime	Value	QualityDetail	OPC-Quality	StateTime
I001	2009-09-12 00:20	NULL	65536	0	NULL
I001	2009-09-12 00:20	1.0	90	90	NULL
I001	2009-09-12 00:20	2.0	65	65	NULL
I001	2009-09-12 00:20	1.0	64	64	4000
I001	2009-09-12 00:20	2.0	64	64	6000

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Quality Rule - Query 7

The returned Quality Detail is the same as OPC quality unless there is special flag for certain indication; for example, when there is indication for rollover in counter mode.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality
   FROM History
      WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Avg'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	218
I0R5	2009-09-12 00:14	3	218
I0R5	2009-09-12 00:22	0	218
I0R5	2009-09-12 00:25	1	218
I0R5	2009-09-12 00:27	0	218
I0R5	2009-09-12 00:29	2	218
I0R5	2009-09-12 00:33	3	218
I0R5	2009-09-12 00:36	0	218
I0R5	2009-09-12 00:39	1	218

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality
I0R5	2009-09-12 00:20	2.6	218	218
I0R5	2009-09-12 00:30	1.0	218	218
I0R5	2009-09-12 00:40	1.6	218	218

Quality Rule - Query 8

For Interpolated mode only, the returned row with Linear wwInterpolationType will have combined qualities.

```
SELECT TagName, DateTime, Value, QualityDetail, OPCQuality, wwRetrievalMode,
wwInterpolationType
   FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      WHERE TagName = 'I0R5'
         AND DateTime >= '2009-09-12 00:20'
         AND DateTime <= '2009-09-12 00:40'
         AND wwResolution = 10000
         AND wwRetrievalMode = 'Interpolated'
         AND wwInterpolationType = 'Linear'
```

If you run this query against the following sample data:

Tagname	DateTime	Resolution	QualityDetail
I0R5	2009-09-12 00:07	2	193
I0R5	2009-09-12 00:14	3	195
I0R5	2009-09-12 00:22	0	196
I0R5	2009-09-12 00:25	1	199
I0R5	2009-09-12 00:27	0	200
I0R5	2009-09-12 00:29	2	207
I0R5	2009-09-12 00:33	3	215
I0R5	2009-09-12 00:36	0	216
I0R5	2009-09-12 00:39	1	219

The results are:

Tagname	DateTime	Value	QualityDetail	OPC Quality	wwRetrievalMode	wwInterpolationType
I0R5	2009-09-12 00:20	0.8	192	192	Interpolated	Linear
I0R5	2009-09-12 00:30	2.3	192	192	Interpolated	Linear
I0R5	2009-09-12 00:40	1.0	192	219	Interpolated	Linear

Note: Cyclic, Full, Delta, Maximum, Minimum, and BestFit do not have combined qualities; therefore, the rules are not applied to these modes..

State Calculation (wwStateCalc)

The state calculation setting applies to ValueState and RoundTrip retrieval.

For ValueState retrieval, you can choose the type of state calculation (aggregation) to be performed on the data:

- **Minimum:** The shortest amount of time that the tag has been in each unique state.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- **Maximum:** The longest amount of time that the tag has been in each unique state.
- **Average:** The average amount of time that the tag has been in each unique state.
- **Total:** The total amount of time that the tag has been in each unique state.
- **Percent:** The total percentage of time that the tag has been in each unique state.
- **MinContained**: The shortest amount of time each tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **MaxContained**: The longest amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **AvgContained**: The average amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **TotalContained**: The total amount of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.
- **PercentContained**: The percentage of time that the tag has been in each unique state for each cycle, disregarding the occurrences that are not fully contained with the calculation cycle.

All results except Percent are in milliseconds. Percent is a percentage typically between 0.0 and 100.0. The percentage can be higher than 100 in certain circumstances.

The nature of the data and how you set the cycle count determines whether you should use a "contained" version of the aggregation. The calculations apply to each unique value state that the tag was in during each retrieval cycle for the query. The total and percent calculations are always exact, but the minimum, maximum, and average calculations are subject to "arbitrary" cycle boundaries that do not coincide with the value changes. Therefore, non-intuitive results may be returned. This is most apparent for slowly-changing tags queried over long cycles.

For example, a string tag that assumes only two distinct values changing every 10 minutes is queried with a cycle time of two hours. Going into a cycle, the value (state) at the cycle boundary is recorded. If the value then changes a short while into the cycle, the state found at the cycle start time will most likely end up being the minimum value. Likewise, the state at the cycle end time is cut short at the cycle end time. The two cut-off occurrences in turn skew the average calculation.

For RoundTrip retrieval, you can only specify the following types of state calculations (aggregations) to be performed on the data. The calculations are for each unique state within each retrieval cycle for the query.

- **MinContained**. The shortest time span between consecutive leading edges of any state that occurs multiple times within the cycle, while disregarding state occurrences that are not fully contained inside of the cycle.
- **MaxContained**. The longest time span between consecutive leading edges of any state that occurs multiple times within the cycle, while disregarding state occurrences that are not fully contained inside of the cycle.
- **AvgContained**. The average time span between consecutive leading edges of any state that occurs multiple times within the cycle, while disregarding state occurrences that are not fully contained inside of the cycle. (This is the default.)
- **TotalContained**. The total time span between consecutive leading edges of any state that occurs multiple times within the cycle while disregarding state occurrences that are not fully contained inside of the cycle.
- **PercentContained**. The percentage of the cycle time spent in time span between consecutive leading edges for a state that occurs multiple times within the cycle while disregarding value occurrences that are not fully contained inside of the cycle.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Analog Value Filtering (wwFilter)

You can use the following analog filters for all retrieval modes:

- Statistical removal of outliers
- Analog to discrete conversion
- Zero around a base value

These filters are applied in a query to retrieve data from the History table, WideHistory table, or StateWideHistory table. These filter only apply to analog tags. All other types of tags, including analog summary tags, are not supported.

You need to specify a filter name in the virtual column wwFilter, with or without an override, to the set of parameters that are defined for the specified filter. The filters are specified as C-like functions: parentheses are always required, even when you choose not to override the default parameters by passing no parameters.

The default value for the wwFilter column is 'NoFilter'. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.

When you use the analog filters in a query that uses wwQualityRule, wwQualityRule is applied first and wwFilter is applied later. You can only use one filter per query.

Statistically Removing Outliers (SigmaLimit)

This analog filter removes outliers from a set of analog points based on the assumption that the distribution of point values in the set is a normal distribution.

The following illustration shows an example of outliers.

filter_outliers

You can filter outliers by specifying a filter called 'SigmaLimit()'. This filter has one parameter defined for specifying the value of n. This parameter is of type double. If the parameter is omitted, then a default parameter of 2.0 is used.

When this filter is specified in any retrieval mode, a time weighted mean, ì (mu), and time weighted standard deviation, ó (sigma), are found for each analog tag for the entire query range including phantom cycles if any, and points falling outside of the range [ì - nó, ì + nó] are removed from the point set before the points are processed further. In other words, the value will be filtered out if value > ì + nó or value < ì – nó.

Time weighted standard deviation is calculated as:

Math.Sqrt((integralOfSquares - 2 * timeWeightedAverage * integral + totalTime * timeWeightedAverage * timeWeightedAverage)/totalTime);

This is the single pass equivalent to the formula:

filter_equation

Ranges where the value is NULL are excluded from these calculations.

A cyclic query example using a 'SigmaLimit()' filter without specifying the n value would look like this:

```
SELECT DateTime, Value, wwFilter
   FROM History
      WHERE TagName = ('TankLevel')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'Cyclic'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND wwFilter = 'SigmaLimit()'
```

Not specifying the n-value as done here is the same as specifying 'SigmaLimit(2)'. The result set might look like this:

DateTime	Value	wwFilter
2008-01-15 15:00:00.000	34.56	SigmaLimit()
2008-01-15 16:00:00.000	78.90	SigmaLimit()
2008-01-15 17:00:00.000	12.34	SigmaLimit()

If the first value would be filtered out by the SigmaLimit filter, the value will be replaced with the time weighted mean.

Converting Analog Values to Discrete Values (ToDiscrete)

The analog to discrete conversion filter allows you to convert value streams for any analog tag in the query tag list into discrete value streams. The filter can be used with all the retrieval modes.

To convert analog values to discrete values, specify the ToDiscrete() filter in the wwFilter column. This filter has two parameters:

Parameter	Valid Values	Default Value
CutoffValue	any double value	0.0
Operator	>, >=, or <=	>

The following are supported syntaxes.

- ToDiscrete()
- ToDiscrete(2)
- ToDiscrete(2, >=)

The following are unsupported syntaxes.

- ToDiscrete(2,)
- ToDiscrete(, >=)
- ToDiscrete(>=)

The cutoff value holds the value that signifies the boundary between values that are to be interpreted as OFF and values that are to be interpreted as ON.

The operator parameter controls the value range relative to the cutoff value to convert to the ON value and vice versa.

NULLs encountered in the analog value stream are copied unchanged to the discrete value stream. The quality of each discrete point is copied from the analog point that causes its production. However, the quality detail for values modified with this filter will have the QualityDetail flag 0x2000 (value changed by filter) set. For example, consider the following ValueState query:

```
SELECT DateTime, vValue, StateTime, wwFilter
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'ValueState'
         AND wwStateCalc = 'MinContained'
         AND wwResolution = 7200000
         AND wwFilter = 'ToDiscrete(29, >)'
```

Here the operator is specified as >, so values greater than but not including 29 are internally converted to ON, whereas values from 0 to 29 are converted to OFF. This query could return the following rows:

DateTime	vValue	StateTime	wwFilter
2008-01-15 15:00:00.000	0	30000	ToDiscrete(29, >)
2008-01-15 15:00:00.000	1	30000	ToDiscrete(29, >)
2008-01-15 17:00:00.000	0	30000	ToDiscrete(29, >)
2008-01-15 17:00:00.000	1	30000	ToDiscrete(29, >)

The values returned in the StateTime column show that the shortest amount of time that SysTimeSec had values equivalent to either ON or OFF and remained in that state was 30 seconds. A similar RoundTrip query would look like this:

```
SELECT DateTime, vValue, StateTime, wwFilter
   FROM History
      WHERE TagName IN ('SysTimeSec')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'RoundTrip'
         AND wwStateCalc = 'MaxContained'
         AND wwResolution = 7200000
         AND wwFilter = 'ToDiscrete(29, <=)'
```

Here the operator is specified as <=, so the resulting conversion is exactly opposite to that performed in the previous query. Now values smaller than or equal to 29 are internally converted to ON, whereas values from 30 to 59 are converted to OFF. This query could return the following rows:

DateTime	vValue	StateTime	wwFilter
2008-01-15 15:00:00.000	0	60000	ToDiscrete(29, <=)
2008-01-15 15:00:00.000	1	60000	ToDiscrete(29, <=)
2008-01-15 17:00:00.000	0	60000	ToDiscrete(29, <=)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2008-01-15 17:00:00.000	1	60000	ToDiscrete(29, <=)

The values returned in the StateTime column now show that the longest amount of time found between roundtrips for both the OFF and the ON state within the 2-hour cycles was 60 seconds.

Using the ToDiscrete() filter is similar to using edge detection for event tags. Edge detection returns the actual value with a timestamp in history for when a value matched a certain criteria. The ToDiscrete() filter returns either a 1 or 0 to show that the criteria threshold was crossed. The ToDiscrete() filter is more flexible, however, in the following ways:

- You can use it with delta and full retrieval.
- You can combine it with "time-in-state" calculations to determine how long a value is above a certain threshold or the duration between threshold times.

Use the ToDiscrete() filter if you are mostly interested in when something occurred, and not necessarily the exact value of the event.

For more information on edge detection, see Edge Detection for Events (wwEdgeDetection).

"Zeroing" Around a Base Value (SnapTo)

This analog filter lets you force values in a well-defined range around one or more base values to "snap to" that base value. For example, you can use this filter when a tank is known to be empty, but the tag that stores the tank level returns a "noisy" value close to zero.

The filter can be used with all retrieval modes, but its main benefits are in the aggregate retrieval modes: average, integral, minimum, and maximum.

To zero values around the base value, specify the SnapTo() filter in the wwFilter column of the query.

The syntax for this filter is:

```
SnapTo([tolerance[,base_value_1[, base_value_2]…]])
```

This filter has two parameters:

Parameter	Valid Values	Default Value
Tolerance	any double value	0.01
BaseValue	zero, one, or up to 100 comma-separated double values	single base value of 0.0

The following are supported syntaxes.

- SnapTo() – Same as SnapTo(0.01, 0.0)
- SnapTo(3.7) – Same as SnapTo(3.7, 0.0)
- SnapTo(3,) – Syntax Error
- SnapTo(,0) – Syntax error
- SnapTo(,) – Syntax error
- SnapTo(3, 4, -5) – Tolerance=3, Base Values 4 and -5.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

When the Snap to filter is specified, point values falling inside any of the ranges [Base value – Tolerance, Base value + Tolerance] will be forced to the base value before the point goes into further retrieval processing. The result is undefined if the base value +/- tolerance exceeds the range of the double data type. The range is calculated using this expression:

```
If (x <= Base value + Tolerance AND x >= Base value – Tolerance)
x = Base value
```

where *x* is the value of the point then if ranges overlap, the first matching base value will be used.

A query example from the History table looks like this:

```
SELECT DateTime, Value, wwFilter
   FROM History
      WHERE TagName = ('TankLevel')
         AND DateTime >= '2008-01-15 15:00:00'
         AND DateTime <= '2008-01-15 17:00:00'
         AND wwRetrievalMode = 'Average'
         AND wwResolution = 3600000
         AND wwFilter = 'SnapTo(0.01, 0, 1000)'
```

The following rows might be returned:

DateTime	Value	wwFilter
2008-01-15 15:00:00.000	0	SnapTo(0.01, 0, 1000)
2008-01-15 16:00:00.000	875.66	SnapTo(0.01, 0, 1000)
2008-01-15 17:00:00.000	502.3	SnapTo(0.01, 0, 1000)

When a value is snapped, the QualityDetail bit flag 0x2000 is set.

If the filter syntax is not correct, a syntax error is returned and no rows are returned.

Selecting Values for Analog Summary Tags (wwValueSelector)

For an analog summary tag, multiple summarized values can be stored in the historian for a single summarization period. When you query analog summary data, a single value, time, and quality (VTQ) must first be extrapolated from the summarized values.

You set the value selector in the query to specify which summarized value to return. The possible values are as follows:

Value Selector Setting	Value Returned	Timestamp Returned
AUTO	The retrieval mode determines the value. See the following table for how AUTO applies to the value selection. This is the default value.	The retrieval mode determines the timestamp. See the following table for how AUTO applies to the value selection. This is the default value.
FIRST	The first value that occurs within the summary period.	The actual timestamp of the first value occurrence within the summary period.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Value Selector Setting	Value Returned	Timestamp Returned
LAST	The last value that occurs within the summary period.	The actual timestamp of the last value occurrence within the summary period.
MIN or MINIMUM	The first minimum value that occurs within the summary period.	The actual timestamp of the first minimum value occurrence within the summary period.
MAX or MAXIMUM	The first maximum value that occurs within the summary period.	The actual timestamp of the first maximum value occurrence within the summary period.
AVG or AVERAGE	A time-weighted average calculated from values within the summary period.	The summary period start time.
INTEGRAL	An integral value calculated from values within the summary period.	The summary period start time.
STDDEV or STANDARDDEVIATION	A standard deviation calculated from values within the summary period.	The summary period start time.

The following table describes the value to be considered if the value selector is set to AUTO:

Retrieval Mode	Analog Summary Behavior
Cyclic	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
Delta	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
Full	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
Interpolated	The retrieval mode determines the appropriate value to return. See the following table for how AUTO applies to the value selection. This is the default value.
Best Fit	The first, last, min, and max points from analog summaries are all considered as analog input points. Best fit analysis is done with these points. If the analog summary percentage good is not 100%, the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval Mode	Analog Summary Behavior
	cycle is considered to have a NULL.
Average	The averages of analog summaries are calculated using the values from the Average column of the AnalogSummaryHistory table. Interpolation type is ignored for analog summary values, and STAIRSTEP interpolation is always used. PercentGood is calculated by considering the TimeGood of each analog summary. If cycle boundaries do not exactly match the summary periods of the stored analog summaries, the averages and time good are calculated by prorating the average and time good values for the portion of the time the summary period overlaps with the cycle. Quality will be set to 64 (uncertain) if cycle boundaries do not match summary periods. If the QualityDetail of any analog summary considered for a cycle is uncertain (64), the resulting quality is set to 64.
Minimum	The first minimum value within the summary period is used. The actual timestamp of the first minimum value occurrence within the summary period is used.
Maximum	The first maximum value within the summary period is used. The actual timestamp of the first maximum value occurrence within the summary period is used.
Integral	The integrals of analog summaries are calculated using the Integral column of the AnalogSummaryHistory table. Interpolation type is ignored for analog summary values, and STAIRSTEP interpolation is always used. PercentGood is calculated by considering the TimeGood of each analog summary. If cycle boundaries do not exactly match the summary periods of the stored analog summaries, the integrals and time good are calculated by prorating the integral and time good values for the portion of the time the summary period overlaps with the cycle. Quality is set to 64 (uncertain) if cycle boundaries do not match summary periods. If the QualityDetail of any analog summary considered for a cycle is uncertain (64), the resulting quality will

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval Mode	Analog Summary Behavior
	be set to 64.
Slope	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.
ValueState	Cannot be used with analog summary data. No values are returned.
Counter	Cannot be used with analog summary data. No values are returned.
RoundTrip	Cannot be used with analog summary data. No values are returned.

For an analog summary tag, if any of the data within a summary period has an OPCQuality other than Good, the OPCQuality returned will be Uncertain. This is true even for summary values that are not calculated, such as first, last, minimum, maximum, and so on. For example, if the OPCQuality for a last value is actually Good, but there was a I/O Server disconnect during the summary calculation period, the OPCQuality for the last value is returned as Uncertain. A QualityDetail of 202 is used to distinguish between the original point and the latest point. (Note, however, that replicated data may show a QualityDetail of 202 for points other than the latest point.)

Expressions (wwExpression)

Expressions allow you to perform mathematical and logical operations on tag data.

This option is relevant in the following retrieval modes:

- Cyclic Retrieval
- Delta Retrieval
- Full Retrieval
- Best Fit Retrieval

Note: Using expressions requires the "advanced" license feature to be enabled. If the advanced license is not available when an expression is used, no data is retrieved and an error message is displayed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

What is an expression?

An expression is made up of a combination of the following elements:

1. Values
   - A typical value is made up of three components:
     - A value with a base type of Boolean, integer, or double
     - A date and time in UTC format
     - A 16-bit integer representing the OPC quality of the value
   - Literal values are also supported, which have no date and time associated with them, and are always assumed to be "good" quality.
2. Streams
   - A stream is a time-ordered collection of values. A stream has either a stair-step or linear interpolation model, and an associated engineering unit.
   - In the context of an expression, one tag is the initial stream, but the expression can produce additional streams derived from a tag. For example, the expression `TI101.PV + 4.0` produces a result stream, but the initial stream is the stream associated with `TI101.PV`.
3. Arithmetic and logical operators
   - Standard arithmetic operators such as addition, subtraction, multiplication and division are supported, along with logical operators for comparison, such as greater than (>) and less than (<). For a full list of supported operators, see Supported arithmetic operators.
4. Scalar functions
   - A variety of scalar functions are supported for manipulating data, such as calculating the absolute value, rounding values to the nearest integer, or unit of measure conversion. For a full list of supported scalar functions, see Supported scalar functions.
5. Time-series functions
   - A variety of time-series functions are supported for comparing data, such as calculating minimum/maximum values, or time in state. For a full list of supported functions, see Supported time-series functions.
   - Time intervals can be expressed in either of two ways:
     a. As a number of days. For example, a value of 1 represents a daily period, while a value of .125 represents a period of three hours.

        Note: Daily time periods support Daylight Saving Time. This means the "1 day" period is 24 hours, except on days when the Daylight Saving Time change is made; on those days, it will be either 23 or 25 hours.

     b. As a named period. The following named periods are supported: *minute*, *hour*, *day*, *week*, *month*, *fixedday*, *fixedweek*, *fixedmonth*. These periods are anchored to a rounded time period. For example, "hour" produces hourly intervals where the minutes, seconds and milliseconds are all 0 in local time, while "minute" produces intervals where the seconds and milliseconds are all 0 in local time.

        Note: The *day*, *week* and *month* named periods support Daylight Saving Time, while the *fixedday*, *fixedweek* and *fixedmonth* named periods do not. For example, the length of a *day* period can be 23, 24, or 25 hours, depending on whether a Daylight Saving Time transition takes place. The length of a *fixedday* period is always 24 hours, *fixedweek* is always 7 days times 24 hours, and *fixedmonth* is always the number of calendar days in the month times 24 hours.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Implicit unit conversion

The following table describes the relationships between engineering unit dimensions when they are multiplied or divided by each other.

Factor1	Factor2	Product
Data Rate	Time	Data
Density	Volume	Mass
Energy Density	Volume	Energy
Length	Area	Volume
Length	Length	Area
Mass Flow	Time	Mass
Pressure	Area	Force
Pressure Rate	Time	Pressure
Resistance	Current	Electric Potential
Speed	Time	Length
Temperature Rate	Time	Temperature
Unknown Rate	Time	Unknown
Volumetric Flow	Time	Volume
Radiant Exposure	Time	Irradiance
Rate	Time	Ratio
Rotation Speed	Time	Rotation
Electric Potential	Power	Current
Acceleration	Mass	Force
Force	Speed	Power
Mass	Specific Energy	Energy

Each row in the table corresponds to four equivalent dimension relationships:

- *Dimension 1 * Dimension 2 = Result Dimension*
- *Dimension 2 * Dimension 1 = Result Dimension*
- *Result Dimension / Dimension 1 = Dimension 2*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

- *Result Dimension / Dimension 2 = Dimension 1.*

When working with expressions involving tags or streams with different engineering units, the engineering units of the results are determined as follows:

1. When adding or subtracting streams with units in the same dimension, all streams are first converted to the same unit as the first stream, and then the operations are performed. The results are in the same unit as the first stream. For example, using the expression `SysTimeHour + SysTimeMin`, both tags use units in the *Time* dimension. When the expression is evaluated, the units from the first tag (hours) are used, so `SysTimeMin` is first converted to hours, and the results are in hours.
2. When adding or subtracting streams with one or more streams having "undefined" units, the results are in the same unit as the first stream.
3. When multiplying or dividing a dimensionless stream or numeric constant with a stream that has an associated unit, the results are in that unit.
4. When multiplying or dividing streams where both units are related by one of the entries in the *DimensionMath* table, the result is in the associated result dimension's unit.
5. When multiplying streams where both units are in the same dimension, the scale factors of the two units are multiplied together. If the scale factor matches a unit in the result dimension, the results are in this unit. Otherwise, the original units are first converted to their dimension's base units, and the results are in the base units of the result dimension.
6. Any other combinations cannot be implicitly converted, so the results are in "undefined" units.

Note: AVEVA Historian 2023 updates the base units for some dimensions to enable implicit unit conversion. If you have additional Historian nodes configured for replication, the engineering unit catalog will be out of sync on tier 2 nodes until they are upgraded to the 2023 version.

Interpolation with Expressions

When retrieving data with expressions, the system determines the interpolation type for the expression based on the interpolation type of the individual tags in the expression.

If the interpolation type in the tag metadata is linear then linear interpolation is used, otherwise stair-step interpolation is used. For more information, see Interpolated Retrieval.

Special rules apply when using some expression functions:

1. An expression using the SIGN function uses stair-step interpolation.

   For example, Pump101.Flow is an analog tag representing the flow rate for a pump. While the tag uses linear interpolation, the expression `SIGN(Pump101.Flow)` uses stair-step interpolation.
2. An expression using the ROUND function uses stair-step interpolation.

   For example, Tank100.Level is an analog tag representing the level of fluid in a tank. While the tag uses linear interpolation, the expression `ROUND(Tank100.Level)` uses stair-step interpolation.

Supported arithmetic operators

AVEVA Historian supports the use of standard mathematical operators in expressions. The following operators are supported:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Symbol	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
%	Modulo
( )	Operator grouping

Arithmetic Operator Examples

The following examples show expressions that use arithmetic operators. The tag names used in these examples are:

- *Pump1.FlowRate* measures the flow rate of a pump in liters per second (L/s)
- *Pump2.FlowRate* measures the flow rate of a pump in liters per minute (L/min)
- *TankCapacity* measures the available capacity of a tank in kiloliters (kL)
- *TimeElapsed* measures elapsed time in seconds (s)

Example 1

The following expression returns the sum of flow rates for two pumps.

```
Pump1.FlowRate + Pump2.FlowRate
```

Because the tags use different units, and there is no explicit unit conversion specified, the units of *Pump2.FlowRate* are implicitly converted to liters/second to match the units of *Pump1.FlowRate* before the values are added together.

Example 2

The following expression returns a value per minute representing the total volume flowing through both pumps during that minute.

```
TOTAL(Pump1.FlowRate + Pump2.FlowRate, 1 minute)
```

Example 3

The following expression calculates a pump's efficiency as a percentage. This example uses water at 50 degrees Celsius, which has a density of 992.24 kg/m^3 and head of 3.5m, the gravitational constant of 9.81 m/s^2, and "Pump1.Power" represents the power used by the pump in Watts.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
Pump1.FlowRate * 3.5 * 992.24 * 9.81 / 100.0 / Pump1.Power
```

Example 4

The following expression returns the travel time in seconds for material in a pipe. The pipe has a diameter of 0.15m, and a length of 100m.

```
3.1415 * 0.15 ^ 2.0 * 100 / UOM(Pump1.FlowRate, m3/sec)
```

Supported logical operators

AVEVA Historian supports the use of standard mathematical operators in expressions. The following operators are supported:

Symbol	Operation
=	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

Supported logical functions

In addition to logical operators, AVEVA Historian also supports the use of logical functions in expressions. The following logical functions are supported:

Function	Usage
AND(*arguments*)	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.
OR(*arguments*)	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.
NOT(*argument*)	Evaluates the argument to a Boolean *TRUE* or *FALSE* value, and returns the opposite.

AND

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **AND**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if all arguments evaluate to *TRUE*, otherwise *FALSE*.

OR

Format	*Stream* = **OR**(*expression1, expression2, ...*)
Arguments	A comma-separated list of expressions.
Description	*TRUE* if any arguments evaluate to *TRUE*, otherwise *FALSE*.

NOT

Format	*Stream* = **NOT**(*expression*)
Arguments	Any expression.
Description	Evaluates the expression to a boolean *TRUE* or *FALSE* value, then returns the opposite.

Supported scalar functions

AVEVA Historian supports using scalar functions in expressions. These functions have a 1:1 relationship between the values in the input and output streams.

The following scalar functions are supported:

Function	Purpose
ABS	Calculates the absolute value of a stream.
CASTUOM	Changes a stream's unit of measure without changing the values.
IF	Produces a stream based on the results of a logical expression.
IFBAD	Overrides "bad" quality values in a stream.
IFLONGER	Evaluates whether or not a logical expression is true over a period greater than or equal to a specified time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	interval, and returns a TRUE or FALSE value or stream.
IFSHORTER	Evaluates whether or not a logical expression is true over a period shorter than a specified time interval, and returns a TRUE or FALSE value or stream.
IFSTALE	Replaces values from the input stream that are older than a specified interval with an alternative value.
LOG	Calculates the base 10 logarithm of a stream's values.
ROUND	Rounds a stream's values to two decimal places.
SIGN	Replaces a stream's values with 1 or -1 based on the current value's sign.
SQRT	Calculates the square root of a stream's values.
TIMESHIFT	Shifts the time stamps in a stream by a specified time interval.
TRUNCATE	Removes the fractional portion of a stream's values.
UOM	Converts a stream's values to a specified unit of measure.

ABS

The ABS() function produces a stream of results calculating the absolute value of a stream, converting negative values into positive values.

Syntax

Format	*Stream* = **ABS**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all negative values converted to positive values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ABS Examples

ABS Examples

Example 1

The following expression returns the difference between the flow rates of two pumps.

```
ABS( Pump101.Flow - Pump102.Flow )
```

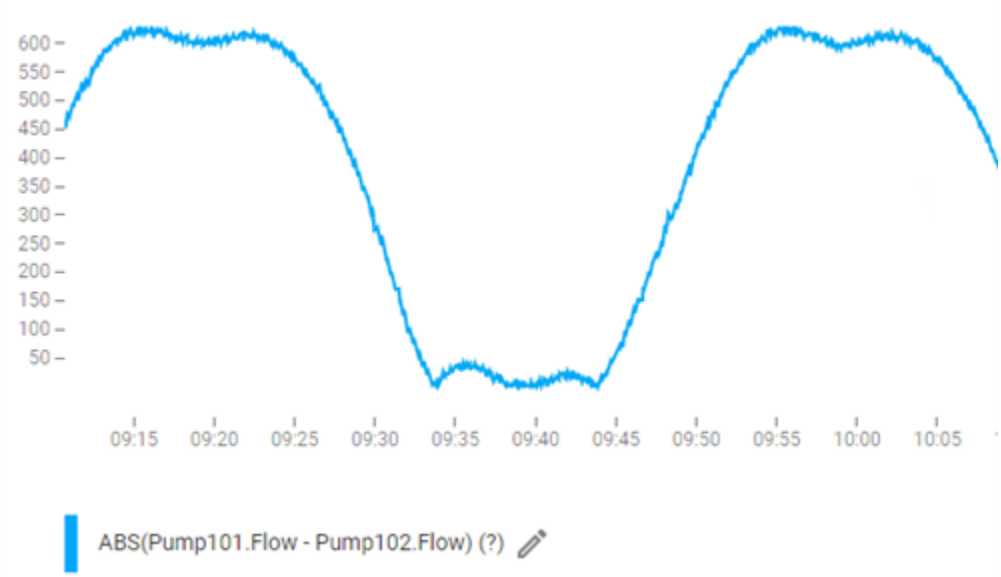
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( Pump101.Flow - Pump102.Flow )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:07:51.4320000	259.439523422241	192
2022-03-31 09:07:53.2590000	268.941925048828	192
2022-03-31 09:07:55.2590000	264.847595214844	192
...	...	...
2022-03-31 10:07:49.2590000	457.803894042969	192

Represented as a line chart over a 1 hour interval:



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example 2

The following expression produces a oscillating stream of values from 0 to 30, and back to 0:

```
ABS( SysTimeSec - 30 )
```

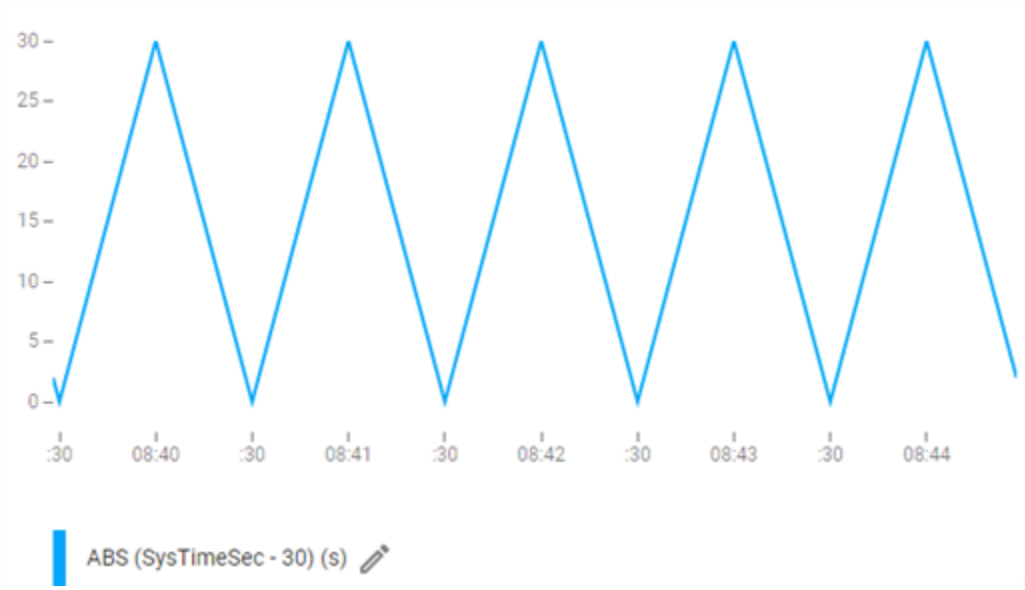
Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'ABS( SysTimeSec - 30 )'
```

Results over a 1 hour interval:

DateTime	Value	OPCQuality
2022-03-31 09:47:52.3500000	22	192
2022-03-31 09:47:53.0000000	23	192
2022-03-31 09:47:54.0000000	24	192
...	...	...
2022-03-31 10:47:51.0000000	21	192

Represented as a line chart over a 5 minute interval:



CASTUOM

The CASTUOM() function changes a stream's unit of measure to the specified unit of measure without any data conversion, preserving the original values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **CASTUOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the the provided stream to use the specified unit of measure while preserving the existing values.

IF

The IF() function produces a stream based on the results of a logical expression.

Syntax

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*, *FalseStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE *FalseStream* - A stream of values from which to select when the expression evaluates to FALSE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected for the output stream. Otherwise, the corresponding value from the *FalseStream* is selected.

Format	*Stream* = **IF**(*LogicalExpression*, *TrueStream*)
Arguments	*LogicalExpression* - A logical expression to be evaluated *TrueStream* - A stream of values from which to select when the expression evaluates to TRUE
Description	When the *LogicalExpression* evaluates to a true value, the corresponding value from *TrueStream* is selected

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	for the output stream. Otherwise, a NULL or "bad quality" value is selected.

Format	*Stream* = **IF**(*LogicalExpression*)
Arguments	*LogicalExpression* - A logical expression to be evaluated
Description	Produces an output stream of Boolean values representing the results of *LogicalExpression*.

IF Examples

IF Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, but any negative values are replaced with 0.

```
IF( Pump101.Flow < 0, 0, Pump101.Flow )
```

Represented as a query:

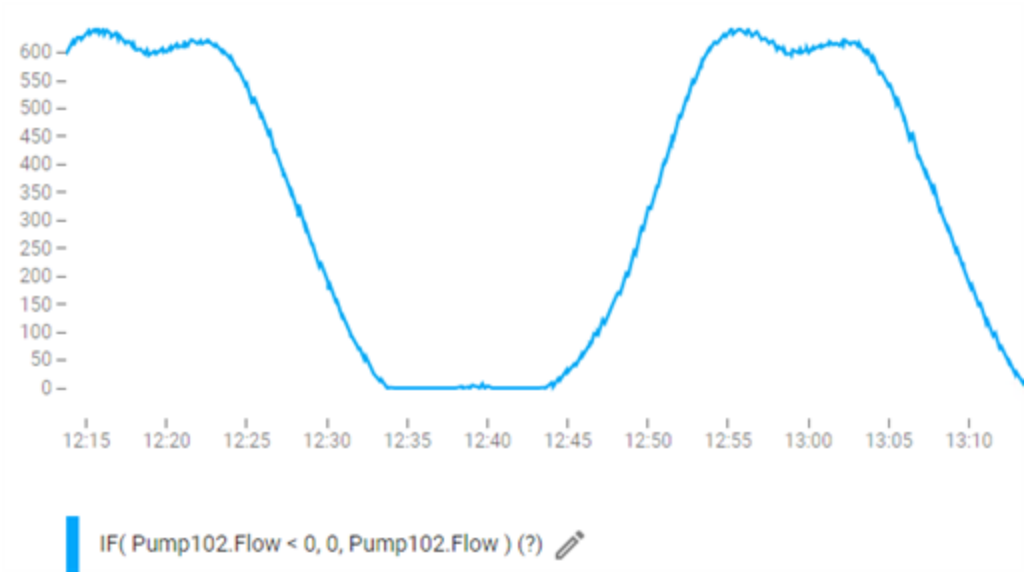
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'IF( Pump102.Flow < 0, 0, Pump102.Flow )'
```

Results over a 5 minute interval:

DateTime	Value	OPCQuality
2022-03-31 12:40:01.2590000	4.00229740142822	192
2022-03-31 12:40:03.2590000	0.365068227052689	192
2022-03-31 12:40:05.2590000	0	192
...	...	...
2022-03-31 12:44:59.2590000	27.5106182098389	192

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 2

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns 0.

```
IF( SysPerfCPUTotal > 90, 1, 0 )
```

Example 3

The following expression returns 1 when system CPU usage exceeds 90%, otherwise it returns a NULL value.

```
IF( SysPerfCPUTotal > 90, 1 )
```

Example 4

The following expression returns a true value (1) when system CPU usage exceeds 90%, otherwise it returns a false value (0).

```
IF( SysPerfCPUTotal > 90 )
```

IFBAD

The IFBAD() function replaces values from the input stream that have a bad OPC Quality with an alternative value.

Syntax

Format	*Stream* = **IFBAD**(*Stream*, *AltValue*)
Arguments	*Stream* - A stream of values *AltValue* - A replacement value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	Replaces every bad value in the input stream with *AltValue*.

Format	*Stream* = **IFBAD**(*Stream*, *AltStream*)
Arguments	*Stream* - A stream of values *AltStream* - A stream of alternative values
Description	Replaces every bad value in the input stream with the value from *AltStream* at the corresponding time.

IFBAD Examples

IFBAD Examples

Example 1

The following expression replaces all bad values for the input stream Pump102.Flow with "500."

```
IFBAD(Pump102.Flow, 500)
```

Example 2

The following expression replaces all bad values in the input stream Pump102.Flow with values from the alternative stream Pump115.Flow at the corresponding time.

```
IFBAD(Pump102.Flow, Pump115.Flow)
```

LOG

The LOG() function produces a stream of results calculating the base 10 logarithm of a stream's values.

Syntax

Format	*Stream* = **LOG**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values converted to their base 10 logarithm.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

LOG Examples

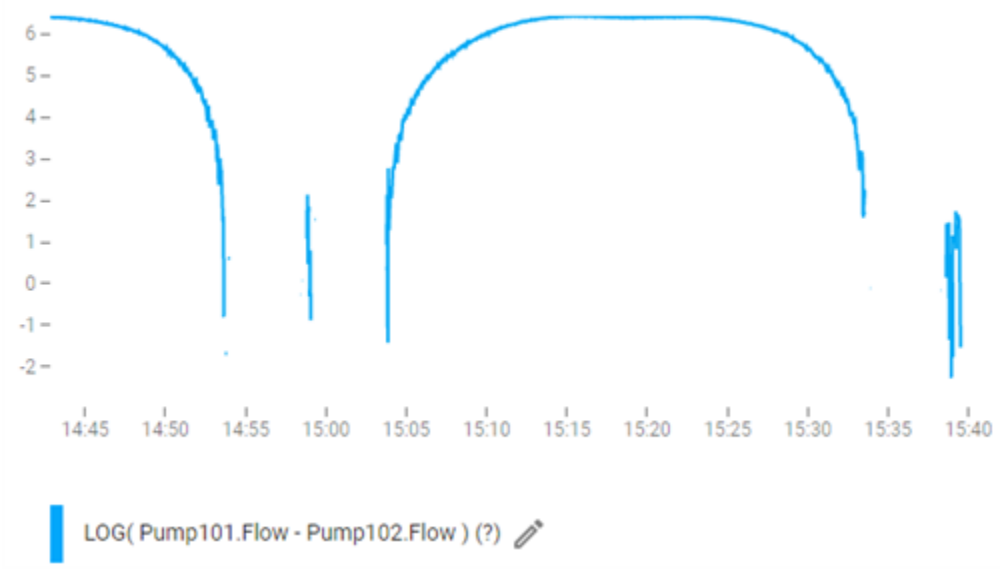
LOG Examples

Example 1

The following expression returns the base 10 logarithm of the difference in flow rates for two pumps.

```
LOG( Pump101.Flow - Pump102.Flow )
```

Represented as a line chart over a 1 hour interval:



ROUND

The ROUND() function produces a stream of results rounding a stream's values to the nearest integer.

Syntax

Format	*Stream* = **ROUND**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values rounded to the nearest integer.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

ROUND Examples

ROUND Examples

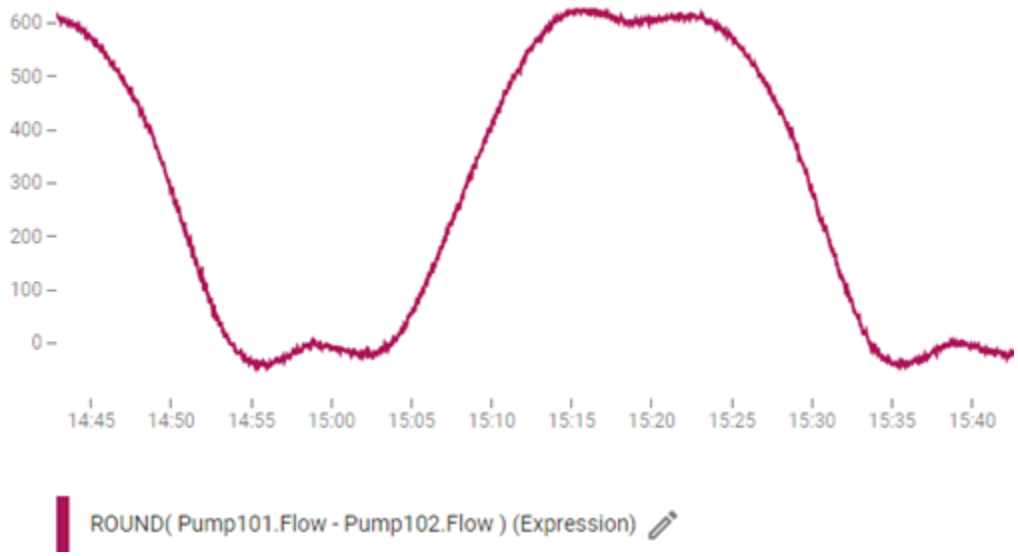
Example 1

The following expression returns the difference in flow rates for two pumps, rounded to the nearest integer.

```
ROUND( Pump101.Flow - Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag.

Represented as a line chart over a 1 hour interval:



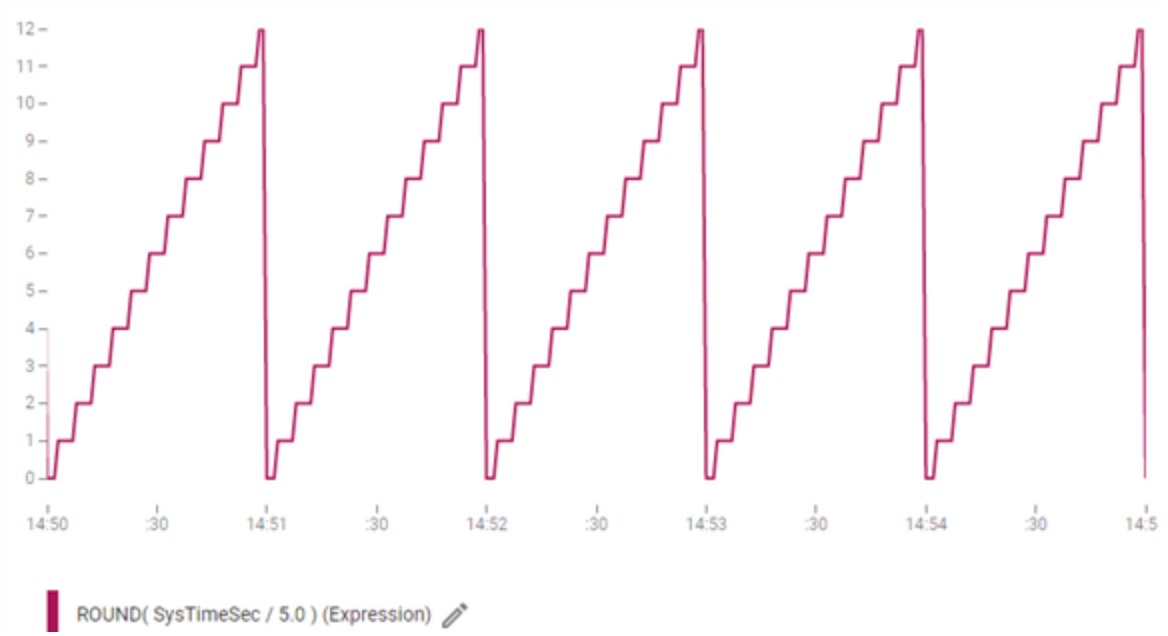
Example 2

The following expression returns the second value of the time, divided by 5 and rounded to the nearest integer.

```
ROUND( SysTimeSec / 5.0 )
```

Represented as a line chart over a 5 minute interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



SIGN

The SIGN() function produces a stream of results indicating the sign of a stream's values.

Syntax

Format	*Stream* = **SIGN**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with all values replaced based on their sign. Positive values are replaced with 1, negative values are replaced with -1, and zero values remain zero.

SIGN Examples

SIGN Examples

Example 1

The following expression returns the sign of the value of the difference in flow rates between two pumps.

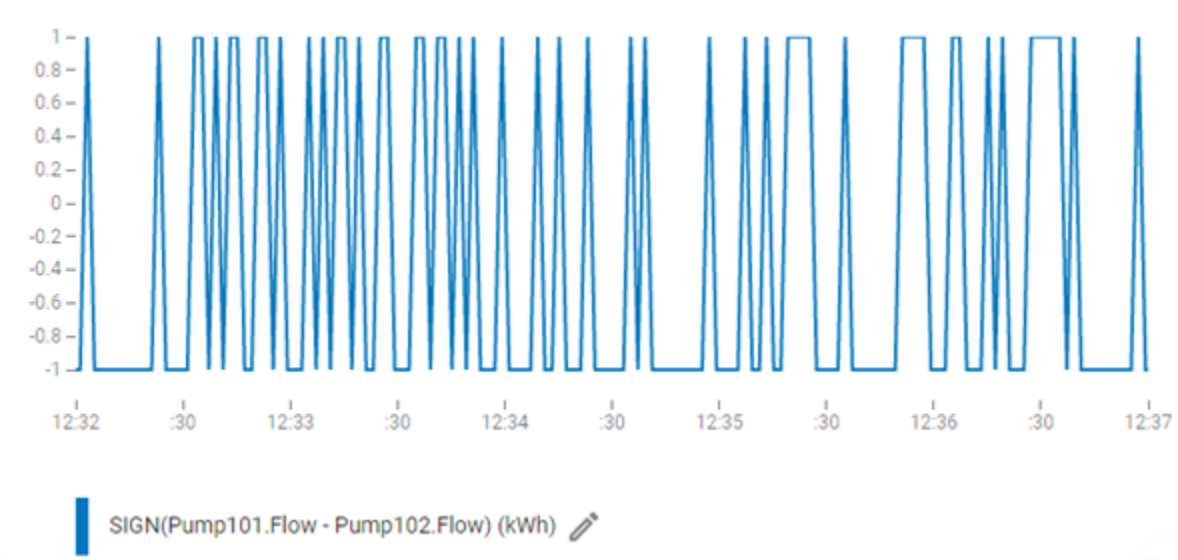
```
SIGN( Pump101.Flow - Pump102.Flow )
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

For each data point in the results, the value will be:

- -1 when the second pump has a higher flow rate
- 0 when the flow rates are equal
- 1 when the first pump has a higher flow rate.

Represented as a line chart over a 5 minute interval:



SQRT

The SQRT() function produces a stream of results calculating the square root of a stream's values.

Syntax

Format	*Stream* = **SQRT**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, but with all values being replaced with their square roots.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQRT Examples

SQRT Examples

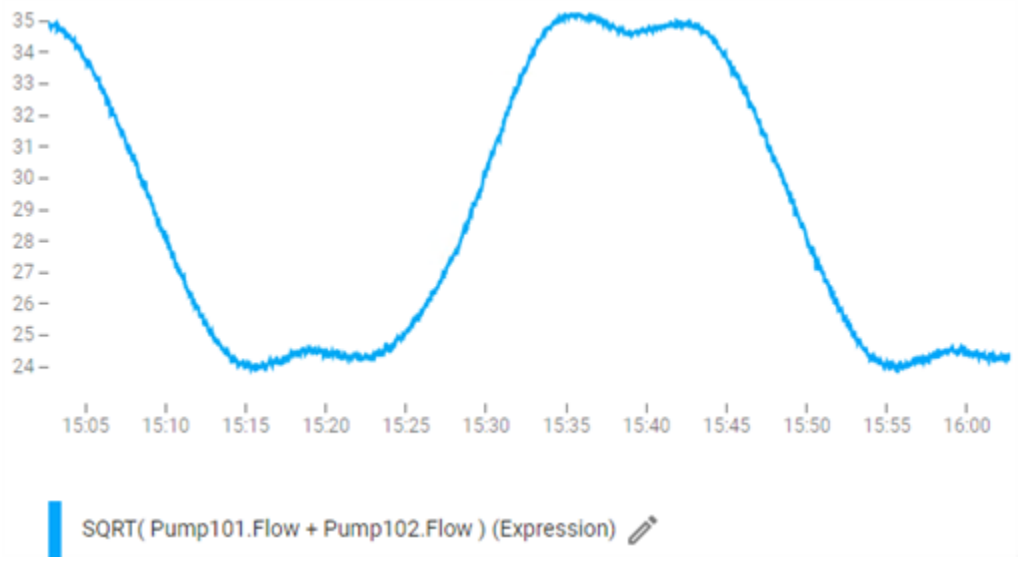
Example 1

The following expression returns the square root of the sum of flow rates for two pumps.

```
SQRT( Pump101.Flow + Pump102.Flow )
```

The units for each tag in the expression are implicitly converted to match the units of the first tag, so the expression's results are in liters per second.

Represented as a line chart over a 1 hour interval:



TIMESHIFT

The TIMESHIFT() function shifts the time stamps in the input stream by adding or subtracting a specified time interval.

Syntax

Format	*Stream* = **TIMESHIFT**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A stream of values *TimeInterval* - The time period to add (or subtract if negative)
Description	The specified interval of time is added to (or

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	subtracted from) the time stamps in the input stream.

TIMESHIFT Examples

TIMESHIFT Examples

Example

The following expression shifts the time stamp of each flow rate value forward by one hour.

```
TIMESHIFT( Pump101.Flow, Hour )
```

Example 2

The following expression shifts the time stamp of the current time's minute value back by two minutes.

```
TIMESHIFT( SysTimeMin, -2 minute )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TIMESHIFT( SysTimeMin, -2 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:57:00.0000000	59	192
2022-03-31 15:58:00.0000000	0	192
2022-03-31 15:59:00.0000000	1	192
2022-03-31 16:00:00.0000000	2	192
...	...	...

TRUNCATE

The TRUNCATE() function produces a stream of results removing the fractional portion of a stream's values.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **TRUNCATE**(*Stream*)
Arguments	*Stream* - A stream of values
Description	Returns a stream that is a copy of the input stream, with the fractional part of all values removed.

TRUNCATE Examples

TRUNCATE Examples

Example 1

The following expression returns the product of flow rates for two pumps, with the fractional portion of each value removed.

```
TRUNCATE( Pump101.Flow * Pump102.Flow )
```

Represented as a query:

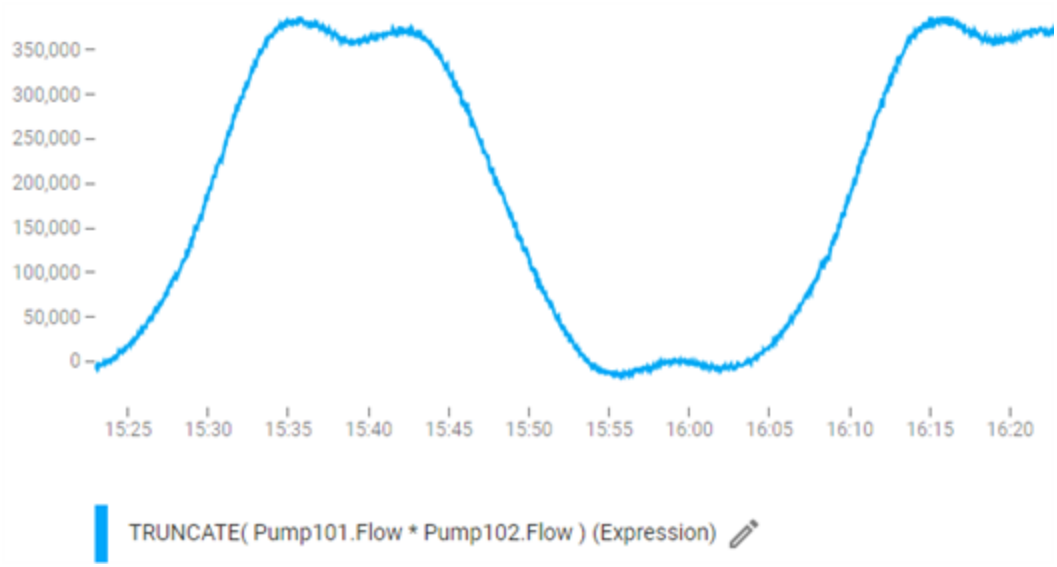
```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( Pump101.Flow * Pump102.Flow )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:24:49.2590000	11030	192
2022-03-31 15:24:51.2590000	14974	192
2022-03-31 15:24:53.2590000	14542	192
2022-03-31 15:24:55.2590000	14887	192
...	...	...

Represented as a line chart over a 1 hour interval:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Example 2

The following expression divides the current time's second value by 5, removing the fractional portion.

```
TRUNCATE( SysTimeSec / 5.0 )
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'TRUNCATE( SysTimeSec / 5.0 )'
AND wwRetrievalMode = 'Full'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 15:31:39.0000000	7	192
2022-03-31 15:31:40.0000000	8	192
2022-03-31 15:31:41.0000000	8	192
2022-03-31 15:31:42.0000000	8	192
...	...	...

UOM (Unit of Measure conversion)

The UOM() function converts the values in a stream to a specified unit of measure.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **UOM**(*Stream*, *Unit*)
Arguments	*Stream* - A stream of values *Unit* - The engineering unit to which values are converted
Description	Converts the values in the provided stream to use the specified unit of measure. If the specified unit does not exist, or is not correctly mapped to a catalog unit for conversion, the original unit is returned instead.

UOM Examples

UOM Examples

Example 1

This example uses the following tag name:

- *Pump101.Flow* measures the flow rate of a pump in liters per second (L/s)

The following expression returns the flow rate of a pump, with the units converted to gallons per second.

```
UOM( Pump101.Flow, gal/min )
```

Supported time-series functions

AVEVA Historian supports using time-series functions in expressions. These functions enable you to compare and manipulate data.

The following time-series functions are supported:

Function	Purpose
AVERAGE	Calculates the time-weighted average for a stream.
AVERAGES	Calculates the statistical average for a stream.
COUNT	Returns a count of the delta points for a stream.
COUNTALL	Returns a count of the data points for a stream.
DURATION	Calculates the duration of *TRUE* states in an

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	expression.
METER	Returns the delta change of a stream's value.
MIN/MAX	Selects the minimum/maximum value of a stream.
PREV/PREVGOOD	Returns the last value from the previous time period.
TOTAL	Calculates the time-weighted total for a stream.
TOTALS	Calculates the statistical total for a stream.

AVERAGE

The AVERAGE() function calculates the time-weighted average value of a stream or array of streams.

When using AVERAGE() in a query, if no resolution is specified, then the behavior is the same as average retrieval mode with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the average value is calculated for each cycle.

Syntax

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the time-weighted average value of a stream or array of streams.

Format	*Stream* = **AVERAGE**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the time-weighted average of a stream or array of streams over the specified time interval, based on local time. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVERAGE Examples

AVERAGE Examples

Example 1

The following expression returns the average flow rate of a pump.

```
AVERAGE(Pump101.Flow)
```

Example 2

The following expression returns the average flow rate of a pump over 10 minute intervals.

```
AVERAGE(Pump101.Flow, 10 Minute)
```

Example 3

The following expression returns the average flow rate of all streams that include ".Flow" in their tag name at 10 minute intervals.

```
AVERAGE(SELECT("TagName",".Flow"), 10 Minute)
```

Example 4

The following expression returns the average minute value of the time over 5 minute intervals.

```
AVERAGE(SysTimeMin, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'AVERAGE( SysTimeMin, 5 minute )'
```

Results:

DateTime	Value	OPCQuality
2022-03-31 12:00:00.0000000	2	192
2022-03-31 12:05:00.0000000	7	192
2022-03-31 12:10:00.0000000	12	192
2022-03-31 12:15:00.0000000	17	192
...	...	...

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVERAGES

The AVERAGES() function calculates the statistical average value of a stream or array of streams over a specified time interval.

Note: This is the preferred method for calculating averages from discrete samples such as those from a checkweigher. For most measurements, however, AVERAGE is preferred.

Syntax

Format	*Stream* = **AVERAGES**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which average values are calculated.
Description	Returns the statistical average for a stream or array of streams over the specified time interval. The start and end times are rounded to the start and end times of the interval. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNT

The COUNT() function returns a count of the delta points for a stream. Note that if the quality detail of a point differs from the previous point, this counts as a delta point, even if the value doesn't change.

Syntax

Format	*Stream* = **COUNT**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of delta points during the query period. The query period is based on local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **COUNT**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the delta points are counted
Description	Returns a count of delta points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNT Examples

COUNT Examples

Example 1

The following expression returns the number of delta points in a freezer unit's temperature.

```
COUNT(Freezer15.Temperature)
```

Example 2

The following expression returns the number of delta points in a freezer unit's temperature over hour intervals.

```
COUNT(Freezer15.FlowTotal, hour)
```

COUNTALL

The COUNTALL() function returns a count of all data points for a stream.

Syntax

Format	*Stream* = **COUNTALL**(*Stream*)
Arguments	*Stream* - A time-ordered vector of values
Description	Returns a count of all data points during the query period. The query period is based on local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **COUNTALL**(*Stream, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time interval over which the points are counted
Description	Returns a count of all data points over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

COUNTALL Examples

COUNTALL Examples

Example 1

The following expression returns the number of data points reported by a pressure alert on a boiler.

```
COUNT(Boiler03.PressureAlert)
```

Example 2

The following expression returns the number of points reported by a pressure alert on a boiler over a one hour interval.

```
COUNT(Boiler03.PressureAlert, hour)
```

DURATION (Time in State)

The DURATION() function returns a stream representing the duration in seconds of each "true" state in an expression.

Syntax

Format	*Stream* = **DURATION**(*StreamLogicalExpression*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated
Description	The logical expression represented by

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	StreamLogicalExpression is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results.

Format	*Stream* = **DURATION**(*StreamLogicalExpression*, *TimeInterval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *TimeInterval* - The time interval over which to calculate durations
Description	The logical expression represented by *StreamLogicalExpression* is evaluated, and a stream is returned representing the duration in seconds of each *TRUE* state in the results, with durations being split across the specified time period. For example, if StreamLogicalExpression evaluates to a *TRUE* value for a time period of 07:55-08:10, and the specified *Interval* is hourly, the output contains one "5 minute" and one "10 minute" result. Without specifying an *Interval*, the output would instead contain one "15 minute" result.

DURATION Examples

DURATION Examples

Example 1

The following expression returns the amount of time in seconds over the specified interval that the number of seconds in the time is 20 or greater.

```
DURATION(SysTimeSec >= 20)
```

Example 2

The following expression returns the amount of time per one hour interval that a tank's level is at or above 500 kL.

```
DURATION(Tank400.Level > 500, hour)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(Tank400.Level >= 500, hour)'
AND DateTime > '2022-04-03 00:00:00'
```

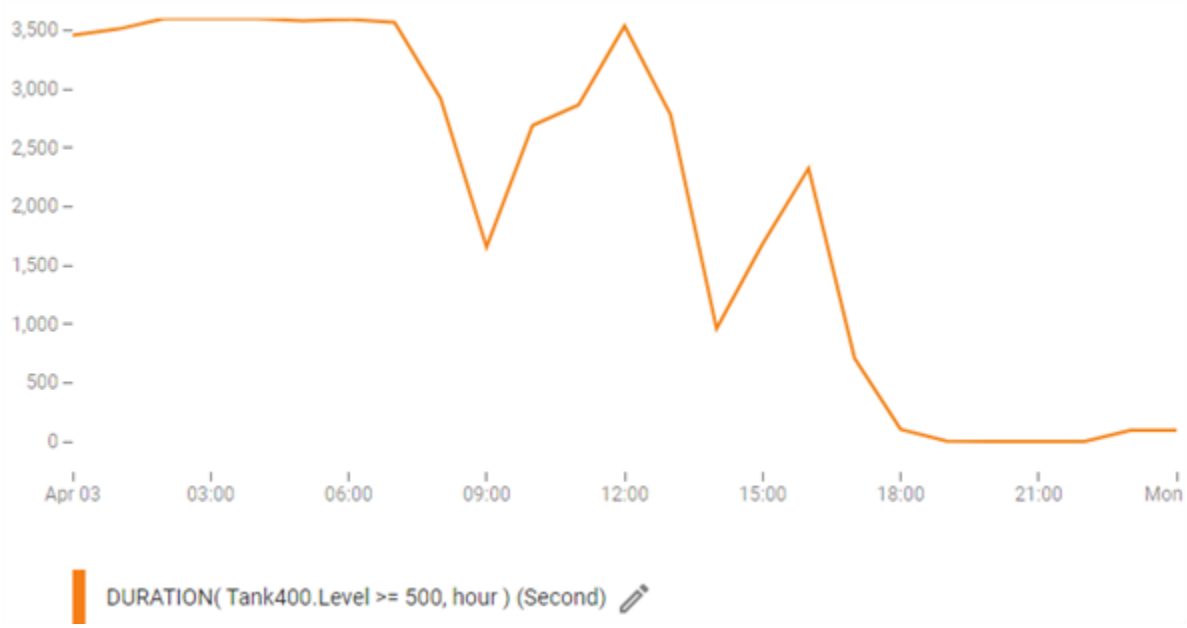
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
AND DateTime < '2022-04-04 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-04-03 01:00:00.0000000	3600	192
2022-04-03 02:00:00.0000000	3578	192
2022-04-03 03:00:00.0000000	3592	192
...	...	...
2022-04-03 23:00:00.0000000	94	192

Represented as a line chart over a 1 day interval:



Example 3

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater, over 5 minute intervals.

```
DURATION(SysTimeSec >= 20, 5 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'DURATION(SysTimeSec >= 20, 5 minute)'
```

Results:

DateTime	Value	OPCQuality
2022-04-01 08:45:00.0000000	200	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2022-04-01 08:50:00.0000000	200	192
2022-04-01 08:55:00.0000000	200	192
2022-04-01 09:00:00.0000000	200	192
...	...	...

Example 4

The following expression returns the amount of time in seconds that the second value of the time is 20 or greater over 30 second intervals, and the results are converted to minutes.

```
UOM(DURATION(SysTimeSec >= 20, 0.5 minute), Minute)
```

IFLONGER

The IFLONGER() function evaluates whether or not a logical expression is true over a period greater than or equal to a specified time interval, and returns a TRUE or FALSE value or stream.

Syntax

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the boolean TRUE is returned. Otherwise the boolean FALSE is returned.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	when the expression evaluates to *TRUE* for the length of the specified interval or longer
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. If it is not *TRUE*, a "bad quality" value is selected.

Format	*Stream* = **IFLONGER**(*StreamLogicalExpression*, *Interval*, *TrueStream*, *FalseStream*)
Arguments	*StreamLogicalExpression* - A logical expression to be evaluated *Interval* - The time interval over which the expression's state is compared *TrueStream* - A stream of values from which to select when the expression evaluates to *TRUE* for the length of the specified interval or longer *FalseStream* - A stream of values from which to select when the expression does not evaluate to *TRUE* for at least the length of the specified interval
Description	The expression represented by *StreamLogicalExpression* is evaluated, and if it remains *TRUE* for a period of time equal to or greater than the specified *Interval*, the corresponding value from *TrueStream* is selected. Otherwise the corresponding value from *FalseStream* is selected.

IFSTALE

The IFSTALE() function replaces values from the input stream that are older than a specified interval with an alternative value.

Syntax

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltValue*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltValue* - A replacement value

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Description	Replaces every value in the input stream older than *Interval* with *AltValue*.

Format	*Stream* = **IFSTALE**(*Stream*, *Interval*, *AltStream*)
Arguments	*Stream* - A stream of values *Interval* - Values older than this time interval are replaced *AltStream* - A stream of alternative values
Description	Replaces every value in the input stream older than *Interval* with the value from *AltStream* at the corresponding time.

METER

The METER() function returns the delta change of the value of a stream or array of streams. This works both for tags without rollover, and tags that reset upon reaching a rollover value.

If an expression includes a single tag, the rollover value defined in the AVEVA Historian will be used. If an expression includes multiple tags, or performs an operation on a tag, a rollover value of 0 is assumed. For example, the following would use the Historian-defined rollover value:

- METER(Tag1, hour)

The following would assume a rollover value of 0:

- METER(Tag1 + Tag2, hour)
- METER(Tag1 + 5, hour)

Otherwise, the METER function works similarly to counter retrieval mode. For more details about rollover, delta change, and common use cases for this function, see Counter Retrieval.

Syntax

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the delta change in the stream or array of streams during the query period. The query period is based on local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Format	*Stream* = **METER**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which the delta change is calculated.
Description	Returns the delta change in the stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

METER Example

METER Example

Example 1

The following expression returns the changes in a pump's total flow.

```
METER(Pump101.FlowTotal)
```

Example 2

A common use of this function is for counters with rollover. The following expression returns the number of units to come out of a production line at one hour intervals.

```
METER(ProductionLine2.UnitCount, hour)
```

Example 3

The following expression returns the number of units to come out of all streams that include "UnitCount" in their tag name at one hour intervals.

```
METER(SELECT("TagName","UnitCount"), hour)
```

MIN/MAX

The MIN() and MAX() functions produce a stream of results selecting the minimum or maximum value of a stream or array of streams. The returned timestamps correspond to the minimum/maximum values retrieved, rather than the beginning of the time interval.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

When using MIN() or MAX() in a query, if no resolution is specified, the behavior is the same as minimum or maximum retrieval modes with no resolution specified; the duration between the start and end times is divided into 100 equal cycles, and the minimum/maximum values are calculated for each cycle.

Syntax

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Returns the minimum (or maximum) value of a stream or array of streams.

Format	*Stream* = **MIN**(*Stream* or *ArrayOfStreams*, *TimeInterval*) *Stream* = **MAX**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which minimum or maximum values are calculated.
Description	Returns the minimum (or maximum) value of a stream or array of streams over the specified time interval.

MIN/MAX Examples

MIN/MAX Examples

Example 1

The following expressions return the minimum (or maximum) flow rate of a pump.

```
MIN(Pump101.Flow)
MAX(Pump101.Flow)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'MIN(Pump101.Flow)'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Result:

DateTime	Value	OPCQuality
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 2

The following expressions return the minimum (or maximum) flow rate of a pump over 30 minute time intervals.

```
MIN(Pump101.Flow, 30 minute)
MAX(Pump101.Flow, 30 minute)
```

Represented as a query:

```
SELECT DateTime, Value, OPCQuality
FROM History
WHERE wwExpression = 'MIN(Pump101.Flow, 30 minute)'
AND DateTime > '2022-03-30 00:00:00'
AND DateTime < '2022-03-31 00:00:00'
```

Results:

DateTime	Value	OPCQuality
2022-03-30 00:13:45.2590000	592.065856933594	192
2022-03-30 00:43:07.2590000	592.34765625	192
2022-03-30 01:01:07.2590000	593.351257324219	192
...	...	...
2022-03-30 23:43:53.2590000	590.4365234375	192

Example 3

The following expression returns the minimum flow rate of all streams measured in the volumetric flow dimension over 30 minute time intervals.

```
MIN(SELECT("Dimension","Volumetric Flow"), 30 Minute)
```

PREV/PREVGOOD

The PREV() and PREVGOOD() functions return the last value from the previous time period for the provided stream.

When using PREV() or PREVGOOD() in a query, if no resolution is specified then the query returns an error.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **PREV**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns the last value from the previous time period, where the period is specified by *TimeInterval*.

Format	*Stream* = **PREVGOOD**(*Stream*, *TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values *TimeInterval* - The time period for which to return the last value
Description	Returns similar results to **PREV**(), but values with "bad" or "uncertain" quality are skipped. **PREVGOOD** will continue seeking backwards until a "good" value is found, going beyond the previous period if necessary. This means **PREVGOOD** always returns a value with "good" quality.

PREV/PREVGOOD Examples

PREV/PREVGOOD Examples

Example 1

The following expression

```
Line1.Total - PREV( Line1.Total, hour )
```

TOTAL

The TOTAL() function produces a stream of results calculating the time-weighted total for a stream or array of streams.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the time-weighted total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTAL**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated.
Description	Calculates the time-weighted total for a stream or array of streams over the specified time interval during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

TOTAL Examples

TOTAL Examples

Example 1

The following expression returns the total volume flowing through a pump over 1 hour intervals.

```
TOTAL(Pump101.Flow, hour)
```

Example 2

The following expression returns the total volume of all streams measured in the volumetric flow dimension over

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

1 hour intervals.

```
TOTAL(SELECT("Dimension","Volumetric Flow"), hour)
```

Example 3

The following expression returns the total energy produced by a turbine over the query period.

```
TOTAL(Turbine306.Output)
```

Example 4

The following expression returns the total energy for all streams with a tag name that starts with "Turbine" and ends with ".Output" over the query period.

```
TOTAL(SELECT("TagName","Turbine*.Output"))
```

TOTALS

The TOTALS() function produces a stream of results calculating the statistical total for a stream or array of streams.

Syntax

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams.
Description	Calculates the statistical total for a stream or array of streams during the query period. The query period is based on local time.

Format	*Stream* = **TOTALS**(*Stream* or *ArrayOfStreams, TimeInterval*)
Arguments	*Stream* - A time-ordered vector of values. *ArrayOfStreams* - An array of streams. See Selecting an array of streams. *TimeInterval* - The time interval over which value totals are calculated
Description	Calculates the statistical total for a stream or array of streams over the specified time interval

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	during the query period. The query period is based on local time. The returned start time is rounded down to the time interval's unit. For example, if the interval is hourly, the time stamps of all values end with "00:00.000".

Bounding

The bounding option is available for delta and full retrieval modes. If the query specifies a start point, this option shows the first data point that occurs on or before the query start. If the query specifies an end point, this option shows the first data point that occurs on or after the query end. The actual date and time of the start/end point are shown, even if it's outside the bounds of the query.

The bounding option works similarly to bounding value retrieval mode, except that it is applied to other modes. For further details on bounding, including illustrations, see Bounding Value Retrieval.

Bounding - Query Example

This example illustrates using the bounding option. First, consider this query, run on a tag that returns data at 3 minute intervals:

```
SELECT DateTime, Value, Quality, OPCQuality
   FROM History
      WHERE TagName = 'Pump106.flow'
         AND wwRetrievalMode = 'Full'
         AND DateTime >= '2022-07-06 07:00:00.000'
         AND DateTime <= '2022-07-06 07:09:00.000'
         AND wwOption = 'Bounding'
```

The results are:

DateTime	Value	Quality	OPCQuality
2022-07-06 06:58:00.000	76.29	0	192
2022-07-06 07:01:00.000	74.35	0	192
2022-07-06 07:04:00.000	72.47	0	192
2022-07-06 07:07:00.000	73.84	0	192
2022-07-06 07:10:00.000	75.62	0	192

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SQL Query Examples

In addition to query examples that use the AVEVA Historian time domain extensions, other query examples are provided to demonstrate how to perform more complex queries or to further explain how retrieval works.

The examples provided are not exhaustive of all possible database queries, but they should give you an idea of the kinds of queries that you could write.

For general information on creating SQL queries, see your Microsoft SQL Server documentation.

Note: If you have configured SQL Server to be case-sensitive, be sure that you use the correct case when writing queries.

Querying the History Table

The History table presents acquired plant data in a historical format. For more information, see History Tables and Views in the *AVEVA Historian Database Reference*.

The following query returns the date/time stamp and value for the tag "ReactLevel." The query uses the remote table view (History is used in place of INSQL.Runtime.dbo.History).

If you do not specify a wwCycleCount or wwResolution, the query will return 100 rows (the default).

```
SELECT DateTime, Sec = DATEPART(ss, DateTime), TagName, Value
   FROM History
      WHERE TagName = 'ReactLevel'
         AND DateTime >= '2001-03-13 1:15:00pm'
         AND DateTime <= '2001-03-13 2:15:00pm'
         AND wwRetrievalMode = 'Cyclic'
```

The results are:

DateTime	Sec	TagName	Value
2001-03-13 13:15:00.000	0	ReactLevel	1775.0
2001-03-13 13:15:00.000	36	ReactLevel	1260.0
2001-03-13 13:16:00.000	12	ReactLevel	1650.0
2001-03-13 13:16:00.000	49	ReactLevel	1280.0
2001-03-13 13:17:00.000	25	ReactLevel	1525.0
2001-03-13 13:18:00.000	1	ReactLevel	585.0
2001-03-13 13:18:00.000	38	ReactLevel	1400.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-03-13 13:19:00.000	14	ReactLevel	650.0
2001-03-13 13:19:00.000	50	ReactLevel	2025.0
2001-03-13 13:20:00.000	27	ReactLevel	765.0
2001-03-13 13:21:00.000	3	ReactLevel	2000.0
2001-03-13 13:21:00.000	39	ReactLevel	830.0
2001-03-13 13:22:00.000	16	ReactLevel	1925.0
...			
(100 row(s) affected)			

Querying the Live Table

The Live table presents the latest available streamed data for each tag in the table.

Note: In certain situations, data can bypass the Live table. These situations include:
- Receiving non-streamed original data (store/forward or CSV);
- Receiving revision data for a Latest value;
- Receiving no new streamed values after Historian was shut down and disabled, or after the computer was rebooted.

For more information, see History Tables and Views in the *AVEVA Historian Database Reference*.

The following query returns the current value of the specified tag. The query uses the remote table view (Live is used in place of INSQL.Runtime.dbo.Live).

```
SELECT TagName, Value
   FROM Live
      WHERE TagName = 'ReactLevel'
```

The result is:

TagName	Value
ReactLevel	1145.0
(1 row(s) affected)	

Querying the WideHistory Table

The wide extension table is a transposition of the History table. Use the wide history tables any time you want to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

find the value of one or more tags over time and need to specify different filter criteria for each tag.

For more information, see History Tables and Views in the *AVEVA Historian Database Reference*.

The following query returns the value of two tags from the WideHistory table. The WideHistory table can only be accessed using the OPENQUERY function. The "Runtime.dbo." qualifier is optional.

```
SELECT * FROM OpenQuery(INSQL,'
   SELECT DateTime, ReactLevel, ReactTemp
   FROM Runtime.dbo.WideHistory
      WHERE Reactlevel > 1500
      AND ReactTemp > 150
   ')
```

The results are:

DateTime	ReactLevel	ReactTemp
2001-03-02 06:20:00.000	1865.0	191.3
2001-03-02 06:21:00.000	2025.0	195.9
2001-03-02 06:22:00.000	2000.0	195.9
2001-03-02 06:23:00.000	2025.0	180.9
2001-03-02 06:27:00.000	1505.0	177.5
(5 row(s) affected).		

In the WideHistory table, the column type is determined by the tag type.

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime, SysTimeMin, SysPulse, SysString FROM
WideHistory
      WHERE DateTime >= "2001-12-20 0:00"
         AND DateTime <= "2001-12-20 0:05"
         AND wwRetrievalMode = "delta"
      ')
```

The results are:

DateTime	SysTimeMin	SysPulse	SysString
2001-12-20 00:00:00.000	0	0	2001/12/20 08:00:00
2001-12-20 00:01:00.000	1	1	2001/12/20 08:00:00
2001-12-20 00:02:00.000	2	0	2001/12/20 08:00:00
2001-12-20 00:03:00.000	3	1	2001/12/20 08:00:00
2001-12-20 00:04:00.000	4	0	2001/12/20 08:00:00

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-12-20 00:05:00.000	5	1	2001/12/20 08:00:00

Querying Wide Tables in Delta Retrieval Mode

Wide tables in delta retrieval mode will behave normally if only one tag is returned. However, for a multiple tag display, a complete row is returned to the client for each instance in which one or more of the tags in the query returns a different value. The row will reflect the actual values being returned for the tags returning results, and will reflect the previous values for the remaining tags in the result set (similar to cyclic retrieval).

Note: The value can be "invalid" or some other quality value.

The following query returns values for three tags from the WideHistory table. "MyTagName" is a tag that periodically is invalid.

```
SELECT * FROM OpenQuery(INSQL,'
   SELECT DateTime, SysTimeSec, SysTimeMin, MyTagName
      FROM WideHistory
         WHERE DateTime >= "2001-05-12 13:00:00"
         AND DateTime <= "2001-05-12 13:02:00"
         AND wwRetrievalMode = "Delta"
   ')
```

The results are:

DateTime	SysTimeSec	SysTimeMin	MyTagName
...			
2001-05-12 13:00:55.000	55	00	1
2001-05-12 13:00:56.000	56	00	1
2001-05-12 13:00:57.000	57	00	1
2001-05-12 13:00:57.500	57	00	null
2001-05-12 13:00:58.000	58	00	null
2001-05-12 13:00:59.000	59	00	null
2001-05-12 13:01:00.000	00	01	null
2001-05-12 13:01:00.500	00	01	2

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-05-12 13:01:01.000	01	01	2
2001-05-12 13:01:02.000	02	01	2
2001-05-12 13:01:03.000	03	01	2
...			

Notice that 57 appears twice since the occurrence of 1 changing to NULL for tag "MyTagName" occurs sometime between the 57th and 58th second. The same applies for NULL changing to 2. The same behavior applies to discrete values.

Querying the AnalogSummaryHistory View

The AnalogSummaryHistory view is a "wide" view that allows you to return multiple statistics for a single tag within a single query.

The following query returns the minimum, maximum, and average values for the SysTimeSec tag for the last hour.

```
declare @End datetime
set @End = left(convert(varchar(30),getdate(),120),14)+'00:00'
SELECT Tagname, OPCQuality, Minimum as MIN, Maximum as MAX, Average as AVG
   FROM AnalogSummaryHistory
      WHERE TagName = 'SysTimeSec'
         AND StartDateTime >= dateadd(minute,-60,@End)
         AND EndDateTime < @End
         AND wwCycleCount = 2
```

The results are:

Tagname	OPCQuality	MIN	MAX	AVG
SysTimeSec	192	0	59	29.5

Note: When querying the AnalogSummaryHistory view, a data point occuring at the same timestamp as the `EndDateTime` is not considered part of the query interval. Instead, it is treated as the first data point in the next interval beginning at `EndDateTime`, and so is not included in the maximum or average value calculations.

Querying the StateSummaryHistory View

The StateSummaryHistory view is a "wide" view that allows you to return multiple statistics for a single tag within a single query.

The following query returns the state count, total time in state, and the percentage of time in state for the SysPulse system tag for the last hour. One row is returned for each state.

```
DECLARE @End DateTime
SET @END = getdate()
   SELECT
      TagName,
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      Value,
      OPCQuality,
      StateCount,
      StateTimeTotal,
      StateTimePercent
   FROM dbo.StateSummaryHistory
   WHERE TagName = N'SysPulse'
      AND StartDateTime >= dateadd(minute,-60,@End)
      AND EndDateTime <= @End
      AND wwCycleCount = 1
```

The results are:

TagName	Value	OPCQuality	StateCount	StateTimeTotal	StateTimePercent
SysPulse	1	192	30	1800000	50
SysPulse	0	192	30	1800000	50

The following query returns the minimum time in state, the minimum contained time in state, and value for the SysTimeSec system tag.

```
SELECT
   TagName,
   StartDateTime,
   EndDateTime,
   StateTimeMin as STM,
   StateTimeMinContained as STMC,
   Value
FROM StateSummaryHistory
WHERE TagName='SysTimeSec'
   AND wwRetrievalMode='Cyclic'
   AND wwResolution=5000
   AND StartDateTime>='2009-10-21 17:40:00.123'
   AND StartDateTime<='2009-10-21 17:40:05.000'
```

The results are:

TagName	StartDateTime	EndDateTime	STM	STMC	Value
SysTimeSec	2009-10-21 17:40:00.123	2009-10-21 17:40:05.123	877	0	0
SysTimeSec	2009-10-21 17:40:00.123	2009-10-21 17:40:05.123	1000	1000	1
SysTimeSec	2009-10-21 17:40:00.123	2009-10-21 17:40:05.123	1000	1000	2
SysTimeSec	2009-10-21 17:40:00.123	2009-10-21 17:40:05.123	1000	1000	3
SysTimeSec	2009-10-21 17:40:00.123	2009-10-21 17:40:05.123	1000	1000	4

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SysTimeSec	2009-10-21 17:40:00.123	2009-10-21 17:40:05.123	123	0	5

Using SliceBy

You can retrieve summary statistics for a tag per batch by using the **SliceBy** parameter in a query. Batches can be defined by changes in a different tag, such as a batch ID or valve position.

For example, suppose you wanted statistics on flow rate depending on the position of a valve. Each time the valve position changes, a new batch is reported. Here is a query to retrieve that information:

```
select SliceByValue, TagName, StartDateTime, EndDateTime, OPCQuality, PercentGood,
wwResolution, Average
from AnalogSummaryHistory
where TagName='M31.FlowIn'
and SliceBy='M31.ValveIn'
and StartDateTime>='2018-11-27 0:00'
and EndDateTime<='2018-11-28 0:00'
```

The results are:

SliceByValue	TagName	StartDateTime	EndDateTime	OPCQuality	PercentGood	wwResolution	Average
0	M31.FlowIn	2018-11-27 07:56:11	2018-11-27 09:49:47	192	100	6816000	57.4379
1	M31.FlowIn	2018-11-27 09:49:47	2018-11-27 15:23:23	192	100	20016000	0.5108
0	M31.FlowIn	2018-11-27 15:23:23	2018-11-27 17:24:17	192	100	7254000	50.3615
1	M31.FlowIn	2018-11-27 17:24:17	2018-11-27 21:50:59	192	100	16002000	0.5220
0	M31.FlowIn	2018-11-27 21:50:59	2018-11-27 23:47:17	192	100	6975000	51.4363

This query uses the same data. This time, using SliceByValue, the query retrieves statistics on the flow rate, but only for those batches when the valve is open.

```
select SliceByValue, TagName, StartDateTime, EndDateTime, OPCQuality, PercentGood,
wwResolution, Average
from AnalogSummaryHistory
where TagName='M31.FlowIn'
and SliceBy='M31.ValveIn'
and SliceByValue=1
and StartDateTime>='2018-11-27 0:00'
```

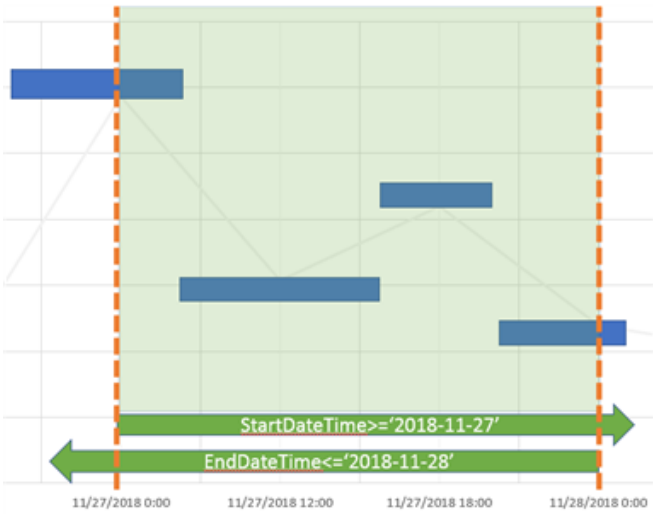
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
and EndDateTime<='2018-11-28 0:00'
```

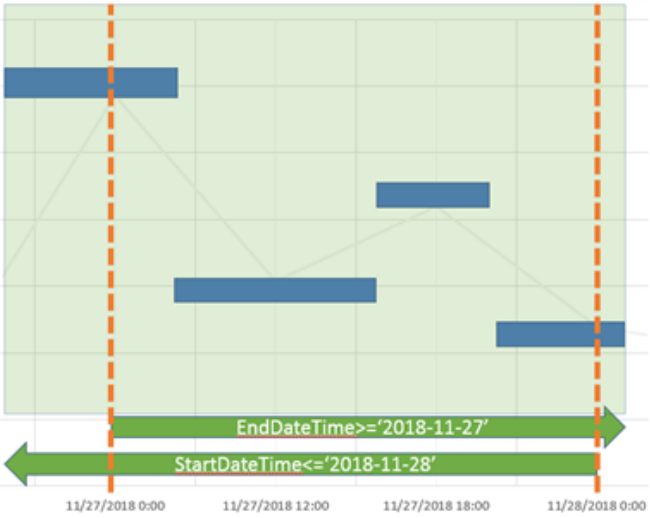
The results this time are:

SliceByValue	TagName	StartDateTime	EndDateTime	OPCQuality	PercentGood	wwResolution	Average
1	M31.FlowIn	2018-11-27 09:49:47	2018-11-27 15:23:23	192	100	20016000	0.5108
1	M31.FlowIn	2018-11-27 17:24:17	2018-11-27 21:50:59	192	100	16002000	0.5220

In the previous two queries, "StartDateTime>=" and "EndDateTime<=" define boundaries for the query to include only those batches that both start and end within the boundaries set. Any batches that begin before or end after the reporting period are excluded.



This next query is the same as the previous one, but swaps the start and end time criteria to include batches that begin before or end after the reporting period:




```
select SliceByValue, TagName, StartDateTime, EndDateTime, OPCQuality, PercentGood,
wwResolution, Average
from AnalogSummaryHistory
where TagName='M31.FlowIn'
and SliceBy='M31.ValveIn'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
and SliceByValue=1
and EndDateTime>='2018-11-27 0:00'
and StartDateTime<='2018-11-28 0:00'
```

The results are:

SliceByValue	TagName	StartDateTime	EndDateTime	OPCQuality	PercentGood	wwResolution	Average
1	M31.FlowIn	2018-11-26 23:08:08	2018-11-27 07:56:11	192	100	31683000	0.5386
1	M31.FlowIn	2018-11-27 09:49:47	2018-11-27 15:23:23	192	100	20016000	0.5108
1	M31.FlowIn	2018-11-27 17:24:17	2018-11-27 21:50:59	192	100	16002000	0.5220
1	M31.FlowIn	2018-11-27 23:47:14	2018-11-28 10:53:20	192	100	39966000	0.5039

SliceBy Queries That Include Future

If you define a timeframe that includes a batch that is not completed, you will get an summary statictics for the batch data currently available, but it will be marked with "OPCQuality=64", meaning the quality is uncertain because the end time is unknown. An ending batch can be incomplete because:

- The batch is currently in progress and the next batch has not yet initiated. Batches are defined by changes in a different tag. If that tag has not yet changed, the current batch is in progress and the next batch has not initiated.
- The batch did physically completed, but the data hasn’t yet been written to the server due to latency, power-outage, etc.

Here is an example. This query retrieves average values for batches that completed on Nov 28. The "select getdate()" line shows that the current timestamp for the query is Nov 28, 2018 at 4:31. Since the day is not over, the system cannot know for sure that another batch won’t be initiated. Therefore, the last retrieved batch is incomplete.

```
select SliceByValue, TagName, StartDateTime, EndDateTime, OPCQuality, PercentGood,
wwResolution, Average
from AnalogSummaryHistory
where TagName='M31.Level'
and SliceBy='M31.Batch'
and EndDateTime>='2018-11-28 0:00'
and StartDateTime<='2018-11-29 0:00'
select getdate()
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SliceByValue	TagName	StartDateTime	EndDateTime	OPCQuality	PercentGood	wwResolution	Average
CRC-4	M31.Level	2018-11-27 21:44:23	2018-11-28 03:03:41	192	100	19158000	94.9624
	M31.Level	2018-11-28 03:03:41	2018-11-28 04:12:20	192	100	4119000	1.6700
BLB-7	M31.Level	2018-11-28 04:12:20	2018-11-28 00:00:00	64	1.7020	NULL	134.8897

(No column name)
2018-11-28 04:31:33

Using Group By with SliceBy

You can use **Group By** in a query to get an overall summary off all occurrences of a state. For example, you might want an overall summary of production when a piece of equipment was in a certain position.

Here is an example of using Group By to organize data by tag state. This query retrieves an overall summary of statistics for each position of the valve. One line summarizes all batches for the day where the valve was closed (0) and another line for all batches with the valve open (1).

```
select SliceByValue, TagName, StartDateTime=min(StartDateTime),
EndDateTime=max(EndDateTime), TotalTime=sum(wwResolution), Maxiumum=max(Maximum),
Total=sum(Integral), Average=sum(Average*wwResolution)/sum(wwResolution),
AvgOfAvg=avg(Average)
from AnalogSummaryHistory
where TagName='M31.FlowIn'
and SliceBy='M31.ValveIn'
and EndDateTime>='2018-11-27 0:00'
and StartDateTime<='2018-11-28 0:00'
group by TagName, SliceByValue
```

The results are:

TagName	SliceByValue	StartTime	EndTime	TotalTime	Maximum	Total	Average	AvgOfAvg
M31.FlowIn	0	2018-11-27 09:49:47	2018-11-27 21:50:59	36018000	6.75	309.66	0.51	0.51

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

M31.FlowIn	1	2018-11-27 07:56:11	2018-11-27 23:47:14	21045000	114.92	18593.13	53.00	53.078

Note: This example correctly calculates the overall average for each state in the "Average" column by weighting the duration of each state. As explained by Simpson's Paradox, the simpler, "AvgOfAvg" calculation is not statistically accurate and can differ significantly with some data sets.

Querying Metadata with the Latest Value

Extended properties can be included in `History` queries with the `TaxExtendedPropertyInfo` view. This view returns the localized version of the alias if one is defined for the current SQL Server connection language (e.g. the @@LANGID). The localized alias comes from the _TagLocalizedProperty table. If no localized alias is defined, the general alias from the _TagExtendedProperty table is used.

The following is an example with the `Alias` extended property:

```
SELECT Alias=isnull(i.PropertyValue,h.TagName), h.*
   FROM History h
   LEFT OUTER JOIN TagExtendedPropertyInfo i
   ON i.TagName = h.TagName and i.PropertyName='Alias'
      WHERE h.DateTime > '2023-08-07 0:00'
         AND h.DateTime < '2023-08-07 5:00'
         AND wwRetrievalMode='delta'
         AND h.TagName='FI101.PV'
```

If the alias is not configured, you can instead use the `ISNULL()` function to use the Tag Name.

To include rows even when there isn't a matching extended property defined, use the `LEFT OUTER JOIN` clause

The following is an example of a query with multiple extended properties:

```
SELECT Alias=isnull(i.Alias,h.TagName), Location, h.*
   FROM History h
   LEFT OUTER JOIN
      (
         SELECT TagName, PropertyName, PropertyValue
         FROM TagExtendedPropertyInfo) temp
         PIVOT
         ( max(PropertyValue)
         FOR PropertyName in ( [Location], [Dimension], [Alias], [Geolocation] )
      ) i
      ON i.TagName = l.TagName
         WHERE h.DateTime > '2023-12-11 0:00'
            AND h.DateTime < '2023-12-11 10:00'
            AND wwRetrievalMode='delta'
            AND h.TagName='SysTimeSec'
   ORDER BY TagName
```

Using an Unconventional Tagname in a Wide Table Query

In a SQL query against a wide table, unconventional tag names must be delimited with brackets ([ ]), because the tagname is used as a column name. For example, tagnames containing a minus (-) or a forward slash (/) must be delimited, otherwise the parser will attempt to perform the corresponding arithmetic operation. No error will result from using brackets where not strictly necessary. For more information on unconventional

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tagnames, see Tag Naming Conventions.

The following is an example of how to delimit a tagname in a query on a wide table. "ReactTemp-2" and "ReactTemp+2" are tagnames. Without the delimiters, the parser would attempt to include the "-2" and "+2" suffixes on the tagnames as part of the arithmetic operation.

For clarity and maintainability of your queries, however, it is recommended that you do not use special characters in tagnames unless strictly necessary.

```
SELECT * FROM OpenQuery(INSQL,
   'SELECT ReactTemp, [ReactTemp-2]-2, [ReactTemp+2]+2 FROM WideHistory WHERE ... ')
```

Using an INNER REMOTE JOIN

Instead of using " … `WHERE TagName IN (SELECT TagName … )` ", it is more efficient to use INNER REMOTE JOIN syntax.

In general, use the following pattern for INNER REMOTE JOIN queries against the historian:

<SQLServerTable> INNER REMOTE JOIN <HistorianExtensionTable>

This query returns data from the history table, based on a string tag that you filter for from the StringTag table:

```
SELECT DateTime, T.TagName, vValue, Quality, QualityDetail
   FROM StringTag T inner remote join History H
   ON T.TagName = H.TagName
      WHERE T.MaxLength = 64
         AND DateTime >='2002-03-10 12:00:00.000'
         AND DateTime <='2002-03-10 16:40:00.000'
         AND wwRetrievalMode = 'Delta'
```

This query returns data from the history table, based on a discrete tag that you filter for from the Tag table:

```
SELECT DateTime, T.TagName, vValue, Quality, QualityDetail
   FROM Tag T inner remote join History H
   ON T.TagName = H.TagName
      WHERE T.TagType = 2
         AND T.Description like 'Discrete%'
         AND DateTime >='2002-03-10 12:00:00.000'
         AND DateTime <='2002-03-10 16:40:00.000'
         AND wwRetrievalMode = 'Delta'
```

Setting Both a Time and Value Deadband for Retrieval

If both time and value deadbands are specified, then every sample is checked for both deadbands, against the current basis value (the last sample returned).

If it passes both tests, then it is returned and acts as the basis for checking the next sample.

For example:

```
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'ReactTemp'
         AND DateTime >= '2002-03-13 10:08'
         AND DateTime <= '2002-03-13 10:28'
         AND wwRetrievalMode = 'Delta'
         AND wwTimeDeadband = 5000
         AND wwValueDeadband = 5
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The tag selected, ReactTemp, has a MinEU value of 0 and a MaxEU value of 220. Thus, the value deadband will be 5 percent of (220 - 0), which equals 11. ReactTemp changes rapidly between its extreme values, but the value remains constant for short periods near the high and low temperature limits. Therefore, when changes are rapid, the value deadband condition is satisfied first, then the time deadband is satisfied. In this region, the behavior is dominated by the time deadband, and the returned rows are spaced at 5 second intervals. Where the temperature is more constant (particularly at the low temperature end), the time deadband is satisfied first, followed by the value deadband. Both deadbands are satisfied only when the value of a row is more than 11 degrees different from the previous row. Thus, the effect of value deadband can be seen to dominate near the low and high temperature extremes of the tag.

The results are:

DateTime	TagName	Value
2002-03-13 10:08:00.000	ReactTemp	121.0
2002-03-13 10:08:10.000	ReactTemp	189.10000610351562
2002-03-13 10:08:20.000	ReactTemp	147.69999694824219
2002-03-13 10:08:30.000	ReactTemp	106.30000305175781
2002-03-13 10:08:40.000	ReactTemp	30.100000381469727
2002-03-13 10:08:50.000	ReactTemp	16.399999618530273
2002-03-13 10:09:00.000	ReactTemp	61.0
2002-03-13 10:09:10.000	ReactTemp	151.0
2002-03-13 10:09:20.000	ReactTemp	173.0
2002-03-13 10:09:30.000	ReactTemp	131.60000610351562
2002-03-13 10:09:40.000	ReactTemp	57.700000762939453
2002-03-13 10:09:50.000	ReactTemp	16.299999237060547
2002-03-13 10:10:10.000	ReactTemp	96.0
2002-03-13 10:10:20.000	ReactTemp	186.0
2002-03-13 10:10:30.000	ReactTemp	156.89999389648437
2002-03-13 10:10:40.000	ReactTemp	115.5
2002-03-13 10:10:50.000	ReactTemp	41.599998474121094
2002-03-13 10:11:00.000	ReactTemp	21.0
2002-03-13 10:11:10.000	ReactTemp	41.0
2002-03-13 10:11:20.000	ReactTemp	131.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2002-03-13 10:11:30.000	ReactTemp	184.5
2002-03-13 10:11:40.000	ReactTemp	140.80000305175781
2002-03-13 10:11:50.000	ReactTemp	99.400001525878906
2002-03-13 10:12:00.000	ReactTemp	25.5
2002-03-13 10:12:20.000	ReactTemp	76.0
2002-03-13 10:12:30.000	ReactTemp	166.0
2002-03-13 10:12:50.000	ReactTemp	124.69999694824219
2002-03-13 10:13:00.000	ReactTemp	50.799999237060547
2002-03-13 10:13:10.000	ReactTemp	16.399999618530273
2002-03-13 10:13:30.000	ReactTemp	111.0
2002-03-13 10:13:40.000	ReactTemp	193.69999694824219
2002-03-13 10:13:50.000	ReactTemp	152.30000305175781
2002-03-13 10:14:00.000	ReactTemp	108.59999847412109
2002-03-13 10:14:10.000	ReactTemp	34.700000762939453
2002-03-13 10:14:20.000	ReactTemp	21.0
2002-03-13 10:14:30.000	ReactTemp	51.0
2002-03-13 10:14:40.000	ReactTemp	146.0
2002-03-13 10:14:50.000	ReactTemp	177.60000610351562
2002-03-13 10:15:00.000	ReactTemp	136.19999694824219
2002-03-13 10:15:10.000	ReactTemp	92.5
2002-03-13 10:15:20.000	ReactTemp	18.600000381469727
2002-03-13 10:15:40.000	ReactTemp	86.0
2002-03-13 10:15:50.000	ReactTemp	181.0
2002-03-13 10:16:00.000	ReactTemp	161.5
2002-03-13 10:16:10.000	ReactTemp	120.09999847412109
2002-03-13 10:16:20.000	ReactTemp	76.400001525878906
2002-03-13 10:16:30.000	ReactTemp	20.899999618530273

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2002-03-13 10:16:50.000	ReactTemp	81.0
2002-03-13 10:17:00.000	ReactTemp	176.0
2002-03-13 10:17:10.000	ReactTemp	163.80000305175781
2002-03-13 10:17:20.000	ReactTemp	122.40000152587891
2002-03-13 10:17:30.000	ReactTemp	46.200000762939453
2002-03-13 10:17:40.000	ReactTemp	18.700000762939453
2002-03-13 10:18:00.000	ReactTemp	116.0
2002-03-13 10:18:10.000	ReactTemp	189.10000610351562
2002-03-13 10:18:20.000	ReactTemp	147.69999694824219
...		

Using wwResolution, wwCycleCount, and wwRetrievalMode in the Same Query

The results of a database query will vary depending on the combination of resolution, cycle count, and retrieval mode that you use in the query. These results are summarized in the following table (where *N* is a numeric value). Note that wwResolution is not supported with DAILY retrieval mode; that mode's resolution is always a full day.

Retrieval Mode	Resolution	Cycle Count	Results
CYCLIC	*N*	0 (or no value)	All stored data for tags during the specified time interval are queried, and then a resolution of N ms applied.
CYCLIC	0 (or no value)	0	The server will return 100,000 rows per tag specified.
CYCLIC	0 (or no value)	*N*	All stored data for tags during the specified time interval are queried, and then a cycle count of N evenly spaced rows is applied.
CYCLIC	*N*	(any value is ignored)	All stored data for tags during the specified time interval are queried, and

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieval Mode	Resolution	Cycle Count	Results
			then a resolution of N ms applied.
CYCLIC	(no value)	(no value or a value less than 0)	The server will return 100 rows per tag specified.
DAILY	n/a	0	The server will return 0 rows per tag specified.
DAILY	n/a	*N*	The server will return N rows per tag specified.
DAILY	n/a	(no value or a value less than 0)	The server will return rows from the StartDateTime to present for each tag specified.
DELTA	(any value is ignored)	0	All values that changed during the specified time interval are returned (up to 100,000 rows total).
DELTA	(any value is ignored)	*N*	Values that changed during the specified time interval are queried, and then a cycle count (first N rows) is applied. The cycle count limits the maximum number of rows returned, regardless of how many tags were queried. For example, a query that applies a cycle count of 20 to four tags will return a maximum of 20 rows of data. An initial row will be returned for each tag, and the remaining 16 rows will be based on subsequent value changes for any tag.
DELTA	(any value is ignored)	(no value)	All values that changed during the specified time interval are returned (no row limit).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

In general, if there is an error in the virtual columns, or an unresolvable conflict, then zero rows are returned.

Determining Cycle Boundaries

With the exception of DAILY retrieval mode, cycle boundaries are calculated based on the query start and end times, wwCycleCount, and wwResolution. (With DAILY retrieval mode, wwResolution is not supported; each cycle is a full day.)

If you only specify wwCycleCount, evenly spaced cycles are returned based on the value of wwCycleCount.

If you only specify wwResolution, cycles are spaced wwResolution milliseconds apart starting at the query start time until query end time is reached. The last cycle will have whatever duration is required to end exactly at the query end time. If this last duration is shortened by this rule, it is known as a partial cycle. Because of this, the final cycle duration may not match wwResolution.

If both wwCycleCount and wwResolution are specified, no result rows will be returned. If you specify neither wwCycleCount nor wwResolution in the query, the query will return 100 rows.

Unless otherwise specified, a value is considered in a given full or partial cycle if its timestamp occurs at or after the cycle start (timestamp >= cycle start) and before the cycle end (timestamp < cycle end).

Mixing Tag Types in the Same Query

The History and Live tables use the sql_variant data type for the vValue column, allowing the return of various data types in a single column. In other words, these tables allow values for tags of different types to be retrieved with a simple query, without the need for a JOIN operation.

For example:

```
SELECT TagName, DateTime, vValue
   FROM History
      WHERE TagName IN ('SysTimeMin', 'SysPulse', 'SysString')
         AND DateTime >= '2001-12-20 0:00'
         AND DateTime <= '2001-12-20 0:05'
         AND wwRetrievalMode = 'delta'
```

The results are:

TagName	DateTime	vValue
SysTimeMin	2001-12-20 00:00:00.000	0
SysPulse	2001-12-20 00:00:00.000	0
SysString	2001-12-20 00:00:00.000	2001/12/20 08:00:00
SysTimeMin	2001-12-20 00:01:00.000	1
SysPulse	2001-12-20 00:01:00.000	1
SysTimeMin	2001-12-20 00:02:00.000	2
SysPulse	2001-12-20 00:02:00.000	0
SysTimeMin	2001-12-20 00:03:00.000	3

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

SysPulse	2001-12-20 00:03:00.000	1
SysTimeMin	2001-12-20 00:04:00.000	4
SysPulse	2001-12-20 00:04:00.000	0
SysTimeMin	2001-12-20 00:05:00.000	5
SysPulse	2001-12-20 00:05:00.000	1

Using a Criteria Condition on a Column of Variant Data

The AVEVA Historian OLE DB provider sends variant data to the SQL Server as a string. If the query contains a criteria condition on a column containing variant type data, the filtering is handled by SQL Server. An example of a criteria condition is:

```
   WHERE ... vValue = 2
```

To perform the filtering, the SQL Server must determine the data type of the constant (in this example, 2), and attempt to convert the variant (string) to this destination type. The SQL Server assumes that a constant without a decimal is an integer, and attempts to convert the string to an integer type. This conversion will fail in SQL Server if the string actually represents a float (for example, 2.00123).

You should explicitly state the destination type by means of a CONVERT function. This is the only reliable way of filtering on the vValue column, which contains variant data.

For example:

```
SELECT DateTime, Quality, OPCQuality, QualityDetail, Value, vValue, TagName
   FROM History
      WHERE TagName IN ('ADxxxF36', 'SysTimeMin', 'SysPulse')
         AND DateTime >= '12-04-2001 04:00:00.000'
         AND DateTime <= '12-04-2001 04:03:00.000'
         AND wwRetrievalMode = 'Delta'
         AND convert(float, vValue) = 2
```

The following is another example:

```
SELECT DateTime, Quality, OPCQuality, QualityDetail, Value, vValue, TagName
   FROM History
      WHERE TagName IN ('VectorX', 'SysTimeMin', 'SysPulse')
         AND DateTime >= '20020313 04:00:07.000'
         AND DateTime <= '20020313 04:01:00.000'
         AND wwRetrievalMode = 'Delta'
         AND convert(float, vValue) > 1
         AND convert(float, vValue) < 2
```

Using DateTime Functions

Date functions perform an operation on a date and time input value and return either a string, numeric, or date and time value.

The following query returns the date/time stamp and value for the SysTimeSec tag for the last 10 minutes.

```
SELECT DateTime, TagName, Value, Quality
   FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= dateadd(Minute, -10, GetDate())
         AND DateTime <= GetDate()
         AND wwRetrievalMode = 'Cyclic'
```

The results are:

DateTime	TagName	Value	Quality
2001-12-15 13:00:00.000	SysTimeSec	0.0	0
2001-12-15 13:00:06.060	SysTimeSec	6.0	0
2001-12-15 13:00:12.120	SysTimeSec	12.0	0
2001-12-15 13:00:18.180	SysTimeSec	18.0	0
2001-12-15 13:00:24.240	SysTimeSec	24.0	0
2001-12-15 13:00:30.300	SysTimeSec	30.0	0
2001-12-15 13:00:36.360	SysTimeSec	36.0	0
2001-12-15 13:00:42.420	SysTimeSec	42.0	0
...			

For any query, the SQL Server performs all date/time computations in local server time, reformulates the query with specific dates, and sends it on to the AVEVA Historian OLE DB provider. The AVEVA Historian OLE DB provider then applies the wwTimeZone parameter in determining the result set.

For example, the following query requests the last 30 minutes of data, expressed in Eastern Daylight Time (EDT). The server is located in the Pacific Daylight Time (PDT) zone.

```
SELECT DateTime, TagName, Value FROM History
   WHERE TagName IN ('SysTimeHour', 'SysTimeMin', 'SysTimeSec')
      AND DateTime > DateAdd(mi, -30, GetDate())
      AND wwTimeZone = 'eastern daylight time'
```

If it is currently 14:00:00 in the Pacific Daylight Time zone, then it is 17:00:00 in the Eastern Daylight Time zone. You would expect the query to return data from 16:30:00 to 17:00:00 EDT, representing the last 30 minutes in the Eastern Daylight Time zone.

However, the data that is returned is from 13:30:00 to 17:00:00 EDT. This is because the SQL Server computes the "`DateAdd(mi, -30, GetDate())`" part of the query assuming the local server time zone (in this example, PDT). It then passes the AVEVA Historian OLE DB provider a query similar to the following:

```
SELECT DateTime, TagName, Value FROM History
   WHERE TagName IN ('SysTimeHour', 'SysTimeMin', 'SysTimeSec')
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      AND DateTime > 'YYYY-MM-DD 13:30:00.000'
      AND wwTimeZone = 'eastern daylight time'
```

Because the OLE DB provider is not provided an end date, it assumes the end date to be the current time in the specified time zone, which is 17:00:00 EDT.

Using the GROUP BY Clause

The GROUP BY clause works if the query uses the four-part naming convention or one of the associated views.

The following example will find the highest value of a specified set of tags over a time period.

```
SELECT TagName, Max(Value)
   FROM INSQL.Runtime.dbo.History
      WHERE TagName IN ('ReactTemp','ReactLevel','SysTimeSec')
         AND DateTime > '2001-12-20 0:00'
         AND DateTime < '2001-12-20 0:05'
         GROUP BY TagName
```

The results are:

SysTimeSec	59.0

Using the COUNT() Function

The COUNT(*) function works directly in a four-part query, but is not supported inside of the OPENQUERY function.

For example:

```
SELECT count(*)
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-12-20 0:00'
         AND DateTime <= '2001-12-20 0:05'
         AND wwRetrievalMode = 'delta'
         AND Value >= 30
```

The result is:

150	

If you use the OPENQUERY function, you cannot perform arithmetic functions on the COUNT(*) column. However, you can perform the count outside of the OPENQUERY, as follows:

```
SELECT count(*), count(*)/2 FROM OPENQUERY(INSQL, 'SELECT DateTime, vValue, Quality,
QualityDetail
   FROM History
      WHERE TagName IN ("SysTimeSec")
         AND DateTime >= "2002-04-16 03:00:00.000"
         AND DateTime <= "2002-04-16 06:00:00.000"
         AND wwRetrievalMode = "Delta"
   ')
```

The result is:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

10801	5400
(1 row(s) affected)	

Using an Arithmetic Function

The following query adds the values of two tags from the WideHistory table.

```
SELECT * FROM OpenQuery(INSQL,'
   SELECT DateTime, ReactLevel, ProdLevel, "Sum" = ReactLevel+Prodlevel
      FROM WideHistory
         WHERE DateTime > "2001-02-28 18:56"
         AND DateTime < "2001-02-28 19:00"
         AND wwRetrievalMode = "Cyclic"
      ')
```

The results are:

DateTime	ReactLevel	Prodlevel	Sum
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	1525.0	2343.0	3868.0
2001-02-28 18:56:00.000	2025.0	2343.0	4368.0
2001-02-28 18:56:00.000	2025.0	2343.0	4368.0
...			
(100 row(s) affected)			

If you use a math operator, such as plus (+), minus (-), multiply (*), or divide (/), you will need to add a blank space in front of and after the operator. For example, "Value - 2" instead of "Value-2".

Using an Aggregate Function

The following query returns the minimum, maximum, average, and sum of the tag 'ReactLevel' from the WideHistory table.

```
SELECT * FROM OpenQuery(INSQL,'
   SELECT "Minimum" = min(ReactLevel),
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   "Maximum" = max(ReactLevel),
   "Average" = avg(ReactLevel),
   "Sum" = sum(ReactLevel)
      FROM WideHistory
         WHERE DateTime > "2001-02-28 18:55:00 "
         AND DateTime < "2001-02-28 19:00:00"
         AND wwRetrievalMode = "Cyclic"
   ')
```

The results are:

Minimum	Maximum	Average	Sum
-25.0	2025.0	1181.2	118120.0

(1 row(s) affected)

If you perform a SUM or AVG in delta retrieval mode against the Wide table, the aggregation will only be performed when the value has changed. The aggregation will not apply to all of the rows returned for each column.

For example, the following query has no aggregation applied:

```
SELECT * FROM OpenQuery(INSQL, 'SELECT DateTime, SysTimeHour, SysTimeMin, SysTimeSec,
SysDateDay
   FROM AnalogWideHistory
      WHERE DateTime >= "2001-08-15 13:20:57.345"
         AND DateTime < "2001-08-15 13:21:03.345"
         AND wwRetrievalMode = "Delta"
      ')
GO
```

The results are:

DateTime	SysTimeHour	SysTimeMin	SysTimeSec	SysDateDay
2001-08-15 13:20:57.343	13	20	57	15
2001-08-15 13:20:58.000	13	20	58	15
2001-08-15 13:20:59.000	13	20	59	15
2001-08-15 13:21:00.000	13	21	0	15
2001-08-15 13:21:01.000	13	21	1	15
2001-08-15 13:21:02.000	13	21	2	15
2001-08-15 13:21:03.000	13	21	3	15

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

(7 row(s) affected)				

Then, a SUM is applied to all of the returned column values:

```
SELECT * FROM OpenQuery(INSQL,'SELECT Sum(SysTimeHour), Sum(SysTimeMin), Sum(SysTimeSec),
Sum(SysDateDay)
   FROM WideHistory
      WHERE DateTime >= "2001-08-15 13:20:57.345"
         AND DateTime < "2001-08-15 13:21:03.345"
         AND wwRetrievalMode = "Delta"
      ')
GO
```

The results are:

SysTimeHour	SysTimeMin	SysTimeSec	SysDateDay
13	41	180	15

Thus, for delta retrieval mode, a SUM or AVG is applied only if the value has changed from the previous row.

If you perform an AVG in delta retrieval mode, AVG will be computed as:

SUM of delta values/number of delta values

For example, an AVG is applied to all of the returned column values:

```
SELECT * FROM OpenQuery(INSQL,'SELECT Avg(SysTimeHour), Avg(SysTimeMin), Avg(SysTimeSec),
Avg(SysDateDay)
   FROM WideHistory
      WHERE DateTime >= "2001-08-15 13:20:57.345"
         AND DateTime < "2001-08-15 13:21:03.345"
         AND wwRetrievalMode = "Delta"
      ')
GO
```

The results are:

SysTimeMin	SysTimeSec
20.5	25.714285714285715

Making and Querying Annotations

The following query creates an annotation for the specified tag. The annotation is made in response to a pump turning off. Then, the annotations for a particular tag are returned.

```
DECLARE @@UserKey INT
SELECT @@UserKey = UserKey
   FROM UserDetail
      WHERE UserName = 'wwAdmin'
INSERT INTO Annotation (TagName, UserKey, DateTime, Content)
      VALUES ('ReactLevel', @@UserKey, GetDate(), 'The Pump is off')
SELECT DateTime, TagName, Content
   FROM Annotation
      WHERE Annotation.TagName = 'ReactLevel'
         AND DateTime > '27 Feb 01'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
         AND DateTime <= GetDate()
```

The results are:

DateTime	TagName	Content
2001-02-28 19:18:00.000	ReactLevel	The Pump is off
(1 row(s) affected)		

Using Comparison Operators with Delta Retrieval

The system behaves differently when doing typical delta-based queries where a start date and end date are specified using the comparison operators >=, >, <= and <. The comparison operators can be used on the History and WideHistory tables. The comparison operators also apply regardless of how the query is executed (for example, four-part naming, OLE DB provider views, and so on).

Delta queries that use the comparison operators return all the valid changes to a set of tags over the specified time span. Using deadbands and other filters may modify the set of valid changes.

Specifying the Start Date with ">="

If the start date is specified using >= (greater than or equal to), then a row is always returned for the specified start date. If the start date/time coincides exactly with a valid value change, then the Quality is normal (0). Otherwise, the value at the start date is returned, and the Quality value is 133 (because the length of time that the tag's value was at *X* is unknown).

Query 1

For this query, the start date will not correspond to a data change:

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime >= '2001-01-13 12:00:30'
         AND DateTime < '2001-01-13 12:10:00'
```

The start time (12:00:30) does not correspond with an actual change in value, and is therefore marked with the initial quality of 133:

DateTime	Value	Quality
2001-01-13 12:00:30.000	0	133
2001-01-13 12:01:00.000	1	0
2001-01-13 12:02:00.000	2	0
2001-01-13 12:03:00.000	3	0
2001-01-13 12:04:00.000	4	0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-01-13 12:05:00.000	5	0
2001-01-13 12:06:00.000	6	0
2001-01-13 12:07:00.000	7	0
2001-01-13 12:08:00.000	8	0
2001-01-13 12:09:00.000	9	0

(10 row(s) affected)

Query 2

For this query, the start date will correspond to a data change:

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime >= '2001-01-13 12:01:00'
         AND DateTime < '2001-01-13 12:10:00'
```

The start time (12:01:00) does correspond exactly with an actual change in value, and is therefore marked with the normal quality of 0.

DateTime	Value	Quality
2001-01-13 12:01:00.000	1	0
2001-01-13 12:02:00.000	2	0
2001-01-13 12:03:00.000	3	0
2001-01-13 12:04:00.000	4	0
2001-01-13 12:05:00.000	5	0
2001-01-13 12:06:00.000	6	0
2001-01-13 12:07:00.000	7	0
2001-01-13 12:08:00.000	8	0
2001-01-13 12:09:00.000	9	0

(9 row(s) affected)

Query 3

For this query, the start date will return at least one row, even though the query captures no data changes:

```
SELECT DateTime, Value, Quality
   FROM History
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime >= '2001-01-13 12:00:30'
         AND DateTime < '2001-01-13 12:01:00'
```

The query does not capture an actual change in value, and is therefore marked with the initial value quality of 133 for the start time of the query:

DateTime	Value	Quality
2001-01-13 12:00:30.000	0	133

(1 row(s) affected)

Specifying the Start Date with ">"

If the start date is specified using > (greater than), then the first row returned is the first valid change after (but not including) the start date. No initial value row is returned. A query that uses > to specify its start date may return zero rows.

Query 1

For this query, the first row that will be returned will be the first valid change after (but not including) the start time (12:00:30):

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime > '2001-01-13 12:00:30'
         AND DateTime < '2001-01-13 12:10:00'
```

The first row returned is the first valid change after (but not including) the start time (12:00:30):

DateTime	Value	Quality
2001-01-13 12:01:00.000	1	0
2001-01-13 12:02:00.000	2	0
2001-01-13 12:03:00.000	3	0
2001-01-13 12:04:00.000	4	0
2001-01-13 12:05:00.000	5	0
2001-01-13 12:06:00.000	6	0
2001-01-13 12:07:00.000	7	0
2001-01-13 12:08:00.000	8	0
2001-01-13 12:09:00.000	9	0

(9 row(s) affected)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Query 2

For this query, the start date will correspond to a data change, but it will be excluded from the result set because the operator used is greater than, not greater than or equal to.

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime > '2001-01-13 12:01:00'
         AND DateTime < '2001-01-13 12:10:00'
```

The start time (12:01:00) corresponds exactly with an actual change in value, but it is excluded from the result set because the operator used is greater than, not greater than or equal to.

DateTime	Value	Quality
2001-01-13 12:02:00.000	2	0
2001-01-13 12:03:00.000	3	0
2001-01-13 12:04:00.000	4	0
2001-01-13 12:05:00.000	5	0
2001-01-13 12:06:00.000	6	0
2001-01-13 12:07:00.000	7	0
2001-01-13 12:08:00.000	8	0
2001-01-13 12:09:00.000	9	0

(8 row(s) affected)

Query 3

This query will return no rows, because no data changes are captured:

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime > '2001-01-13 12:00:30'
         AND DateTime < '2001-01-13 12:01:00'
```

The query does not capture an actual change in value; therefore, no rows are returned.

DateTime	Value	Quality

(0 row(s) affected)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Specifying the End Date with "<="

If the end date is specified using <= (less than or equal to) then the last row returned is the last valid change up to, and including, the end date. If the end date uses "<=" then the last change returned may have a date/time exactly at the end date. If there is a value exactly at the end date, it will be returned.

This query uses the remote table view.

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime > '2001-01-13 12:00:30'
         AND DateTime <= '2001-01-13 12:10:00'
```

Note that there is a valid change at exactly the end time of the query (12:10:00):

DateTime	Value	Quality
2001-01-13 12:01:00.000	1	0
2001-01-13 12:02:00.000	2	0
2001-01-13 12:03:00.000	3	0
2001-01-13 12:04:00.000	4	0
2001-01-13 12:05:00.000	5	0
2001-01-13 12:06:00.000	6	0
2001-01-13 12:07:00.000	7	0
2001-01-13 12:08:00.000	8	0
2001-01-13 12:09:00.000	9	0
2001-01-13 12:10:00.000	10	0

(10 row(s) affected)

Specifying the End Date with "<"

If the end date is specified using < (less than), then the last row returned is the last valid change up to (but not including) the end date. If the end date uses "<" then the last event returned will have a date/time less than the end date. If there is an event exactly at the end date, it will not be returned.

This query uses the remote table view.

```
SELECT DateTime, Value, Quality
   FROM History
      WHERE TagName = 'SysTimeMin'
         AND wwRetrievalMode = 'Delta'
         AND DateTime > '2001-01-13 12:00:30'
         AND DateTime < '2001-01-13 12:10:00'
```

Note that there is a valid change at exactly the end time of the query (12:10:00), but it is excluded from the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

result set.

DateTime	Value	Quality
2001-01-13 12:01:00.000	1	0
2001-01-13 12:02:00.000	2	0
2001-01-13 12:03:00.000	3	0
2001-01-13 12:04:00.000	4	0
2001-01-13 12:05:00.000	5	0
2001-01-13 12:06:00.000	6	0
2001-01-13 12:07:00.000	7	0
2001-01-13 12:08:00.000	8	0
2001-01-13 12:09:00.000	9	0

(9 row(s) affected)

Using Comparison Operators with Cyclic Retrieval and Cycle Count

Cyclic queries with the wwCycleCount time domain extension return a set of evenly spaced rows over the specified time span. The result set will always return the number of rows specified by the cycle count extension for each tag in the query. The resolution for these rows is calculated by dividing the time span by the cycle count.

Specifying Cycle Count with Two Equity Operators

If the time range is specified using >= and <=, then the first row falls exactly on the start time, and the last row falls exactly on the end time. In this case, the resolution used is (end date – start date) / (cyclecount – 1).

This query uses a cycle count of 60, resulting in a 1 second resolution for the data. The query uses the remote table view.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2001-01-13 12:00:00'
         AND DateTime <= '2001-01-13 12:00:59'
         AND wwCycleCount = 60
         AND wwRetrievalMode = 'Cyclic'
```

The results are:

DateTime	Value
2001-01-13 12:00:00.000	0
2001-01-13 12:00:01.000	1

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59

(60 row(s) affected)

Specifying Cycle Count with One Equity Operator

If one end of the time range is excluded (by using > instead of >= or < instead of <=). then a gap of "resolution" is left at the beginning (or end) of the result set.

The resolution is calculated as (end date – start date) / (cyclecount).

The row that equates to the time which is designated using the < (or >) operator is not returned.

These queries use the remote table view.

Query 1

This query uses a cycle count of 60, resulting in a 1 second resolution for the data. The starting time is set to >=.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND wwCycleCount = 60
         AND DateTime >= '2001-01-13 12:00:00'
         AND DateTime < '2001-01-13 12:01:00'
```

The results are:

DateTime	Value
2001-01-13 12:00:00.000	0
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59

(60 row(s) affected)

Query 2

This query also uses a cycle count of 60, resulting in a 1 second resolution for the data. The ending time is set to <=.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND wwCycleCount = 60
         AND DateTime > '2001-01-13 12:00:00'
         AND DateTime <= '2001-01-13 12:01:00'
```

The results are:

DateTime	Value
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59
2001-01-13 12:01:00.000	0

(60 row(s) affected)

Specifying Cycle Count with No Equity Operators

If both ends of the time range are excluded (by using > and <) then a gap of "resolution" is left at the beginning and end of the result set.

The resolution is calculated as (end date – start date) / (cyclecount + 1).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The row(s) that equate to the start and end times are not returned.

This query uses the remote table view.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND wwCycleCount = 60
         AND DateTime > '2001-01-13 12:00:00'
         AND DateTime < '2001-01-13 12:01:01'
```

The results are:

DateTime	Value
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59
2001-01-13 12:01:00.000	0

```
(60 row(s) affected)
```

Using Comparison Operators with Cyclic Retrieval and Resolution

Cyclic queries that use comparison operators and the resolution time domain extension return a set of evenly spaced rows over the specified time span. The resolution for these rows is specified in the query.

Using Two Equality Operators for Comparison with Cyclic Retrieval and Resolution

If the time range is specified using >= and <=, then the first row falls exactly on the start time. The last row will fall exactly on the end time, if the resolution divides exactly into the specified time duration. If the resolution does not divide exactly into the specified time duration, then the last row returned will be the last row satisfying (start date + N*resolution) which has a timestamp less than the end date.

In short:

- <= endtime MAY return a last row containing the exact endtime (but it is not guaranteed to do so)
- < endtime is guaranteed NOT to return a last row containing the exact endtime

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

This query sets the resolution to 1 second.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND wwResolution = 1000
         AND DateTime >= '2001-01-13 12:00:00'
         AND DateTime <= '2001-01-13 12:01:00'
```

The results are:

DateTime	Value
2001-01-13 12:00:00.000	0
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59
2001-01-13 12:01:00.000	0

(61 row(s) affected)

Using One Equality Operator for Comparison with Cyclic Retrieval and Resolution

If the start time is excluded (by using > instead of >=), then a gap of "resolution" is left at the beginning of the result set. In this case, the first row returned will have the timestamp of the (start date + resolution). If the end date uses "<" then the last row returned will be the last row defined by (start date + N*resolution) which has a timestamp less than the end date.

The row that equates to the time that is designated using the < (or >) operator is not returned.

Query 1

This query uses a resolution of 1000, resulting in a 1 second resolution for the data.. The starting time is set to >=.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND wwResolution = 1000
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
         AND DateTime >= '2001-01-13 12:00:00'
         AND DateTime < '2001-01-13 12:01:00'
```

The results are:

DateTime	Value
2001-01-13 12:00:00.000	0
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	
2001-01-13 12:00:55.000	55
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59

(60 row(s) affected)

Query 2

This query also uses a row resolution of 1000, resulting in a 1 second resolution for the data. The starting time is set to <=.

```
SELECT DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND wwResolution = 1000
         AND DateTime > '2001-01-13 12:00:00'
         AND DateTime <= '2001-01-13 12:01:00'
```

The results are:

DateTime	Value
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

...	
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59
2001-01-13 12:01:00.000	0

(60 row(s) affected)

Using No Equality Operators for Comparison with Cyclic Retrieval and Resolution

If both ends of the time range are excluded (by using > and <), then a gap of resolution is left at the beginning and end of the result set.

The row(s) that equate to the start and end times are not returned.

This query uses a resolution of 1000, resulting in a 1 second resolution for the data.

```
SELECT DateTime, Value
   FROM v_AnalogHistory
      WHERE TagName = 'SysTimeSec'
         AND wwResolution = 1000
         AND DateTime > '2001-01-13 12:00:00'
         AND DateTime < '2001-01-13 12:01:01'
```

The results are:

DateTime	Value
2001-01-13 12:00:01.000	1
2001-01-13 12:00:02.000	2
2001-01-13 12:00:03.000	3
2001-01-13 12:00:04.000	4
...	
2001-01-13 12:00:56.000	56
2001-01-13 12:00:57.000	57
2001-01-13 12:00:58.000	58
2001-01-13 12:00:59.000	59
2001-01-13 12:01:00.000	0

(60 row(s) affected)

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Returning Time Between Value Changes

You can return the amount of time before a tag's value changed to a subsequent value. This time is returned using the wwResolution column.

This functionality works with the cyclic, delta, and full retrieval modes. The delta and full mode behavior of wwResolution does not apply to the AnalogSummaryHistory and StateSummaryHistory tables.

If the time change value is greater than 2,147,000,000 milliseconds (~25 days), then the value of wwResolution column is -1.

- Example 1: Cyclic Retrieval
- Example 2: Delta and Full Retrieval
- Example 3: Querying the WideHistory Table
- Example 4: Querying the History Table with the wwValueSelector Parameter
- Example 5: Calculating Total Time Between Value Changes

Example 1: Cyclic Retrieval

For this example, the following data is stored in the Historian:

DateTime	Value
2012-01-01 07:59:53	34.42384
2012-01-01 08:00:13	15.02637
2012-01-01 08:00:33	20.29732
2012-01-01 08:00:53	37.40273
2012-01-01 08:01:13	24.31662

For cyclic retrieval, cycles start at the start DateTime and occur at intervals specified by wwResolution in the query. If you query for the data 2012-01-01 08:00:00 to 2012-01-01 08:01:00 with four cycles, the wwResolution column in the results show the time in milliseconds until the next point.

DateTime	Value	wwResolution
2012-01-01 08:00:00	34.42384	20000
2012-01-01 08:00:20	15.02637	20000
2012-01-01 08:00:40	20.29732	20000
2012-01-01 08:01:00	37.40273	20000

Example 2: Delta and Full Retrieval

In delta and full mode, the wwResolution column is used in the query results to show the time in milliseconds

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

until the next point. When the first point in the result occurs before the query start time, the wwResolution column shows the time from the start of the query to the next point. If there are no more points after the first point, the wwResolution column is NULL.

For example, the following data is stored:

DateTime	Value
2012-01-01 07:59:53	34.42384

If you query for the data between 2012-01-01 08:00:00 to 2012-01-01 08:00:10, the results are:

DateTime	Value	wwResolution
2012-01-01 08:00:00	34.42384	NULL

However, if the data stored is:

DateTime	Value
2012-01-01 07:59:53	34.42384
2012-01-01 08:00:13	15.02637

Then the results are:

DateTime	Value	wwResolution
2012-01-01 08:00:00	34.42384	10000

When the last point in the result occurs before the query end time, the wwResolution column shows the time until the end of the query when there is a next available point. If there are no more points, then the wwResolution column shows NULL.

For example, the following is stored:

DateTime	Value
2012-01-01 07:59:53	34.42384
2012-01-01 08:00:13	15.02637
2012-01-01 08:00:33	20.29732
2012-01-01 08:00:53	37.40273
2012-01-01 08:01:13	24.31662

If you query for data from 2012-01-01 08:00:00 to 2012-01-01 08:05:00, the results are:

DateTime	Value	wwResolution
2012-01-01 08:00:00	34.42384	13000
2012-01-01 08:00:13	15.02637	20000

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2012-01-01 08:00:33	20.29732	20000
2012-01-01 08:00:53	37.40273	20000
2012-01-01 08:01:13	24.31662	NULL

If you query for data from 2012-01-01 08:00:00 to 2012-01-01 08:01:00, the results are:

DateTime	Value	wwResolution
2012-01-01 08:00:00	34.42384	13000
2012-01-01 08:00:13	15.02637	20000
2012-01-01 08:00:33	20.29732	20000
2012-01-01 08:00:53	37.40273	7000

If the last point happens to be end of the query, then the wwResolution value is zero, even when there are no more points after the last point. For example, if you query for data from 2012-01-01 08:00:00 to 2012-01-01 08:01:13, the results are:

DateTime	Value	wwResolution
2012-01-01 08:00:00	34.42384	13000
2012-01-01 08:00:13	15.02637	20000
2012-01-01 08:00:33	20.29732	20000
2012-01-01 08:00:53	37.40273	20000
2012-01-01 08:01:13	24.31662	0

Example 3: Querying the WideHistory Table

If you execute a query on the WideHistory table for analog tags, wwResolution shows the time between the first value change for ANY of the tags.

```
SELECT * FROM OpenQuery(INSQL,'
   SELECT DateTime, SysTimeSec, SysTimeMin, wwResolution
      FROM WideHistory
         WHERE DateTime >= "20120119 12:44:00.000"
         AND DateTime <= "20120119 12:45:00.000"
         AND wwRetrievalMode = "Delta"
   ')
```

The results are:

DateTime	SysTimeSec	SysTimeMin	wwResolution
2012-01-19 12:44:00.0000000	0	44	1000

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

2012-01-19 12:44:01.0000000	1	44	1000
2012-01-19 12:44:02.0000000	2	44	1000
2012-01-19 12:44:03.0000000	3	44	1000
2012-01-19 12:44:04.0000000	4	44	1000
2012-01-19 12:44:05.0000000	5	44	1000
...			

The wwResolution column shows 1000 milliseconds because the smallest time change is for the SysTimeSec tag, which is changing every second.

If you run the same query using the SysTimeHour tag instead of the SysTimeSec tag, the results are:

DateTime	SysTimeHour	SysTimeMin	wwResolution
2012-01-19 12:44:00.0000000	12	44	60000
2012-01-19 12:45:00.0000000	12	45	0

The wwResolution column shows 60000 milliseconds because the smallest time change is for the SysTimeMin tag, which is changing every minute (every 60 seconds). Because the query ended at the time of the last value, a 0 is shown for wwResolution for the ending value.

Example 4: Querying the History Table with the wwValueSelector Parameter

You can query the History table with the wwValueSelector parameter.

```
SELECT DateTime, TagName, Value, wwResolution
   FROM History
      WHERE Tagname like 'MyTag'
      AND DateTime >= '2012-01-19 10:00:00'
      AND DateTime <= '2012-01-19 11:00:00'
      AND wwValueSelector = 'STDDEV'
```

The results are:

DateTime	TagName	Value	wwResolution
2012-01-19 10:00:00.0000000	MyTag	977.157928752564	60000
2012-01-19	MyTag	16.5619987924163	60000

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

10:01:00.0000000			
2012-01-19 10:02:00.0000000	MyTag	16.5619987924163	60000
2012-01-19 10:03:00.0000000	MyTag	16.5619987924165	60000
2012-01-19 10:04:00.0000000	MyTag	16.5619987924163	180000
2012-01-19 10:07:00.0000000	MyTag	16.5619987924171	60000
2012-01-19 10:08:00.0000000	MyTag	16.5619987924163	60000
2012-01-19 10:09:00.0000000	MyTag	16.5619987924179	60000
2012-01-19 10:10:00.0000000	MyTag	16.5619987924163	180000
2012-01-19 10:13:00.0000000	MyTag	16.5619987924195	60000
2012-01-19 10:14:00.0000000	MyTag	16.5619987924163	240000
2012-01-19 10:18:00.0000000	MyTag	16.5619987924226	180000
2012-01-19 10:21:00.0000000	MyTag	16.5619987924163	60000
2012-01-19 10:22:00.0000000	MyTag	16.5619987924226	60000
2012-01-19 10:23:00.0000000	MyTag	16.5619987924038	60000
...			

Example 5: Calculating Total Time Between Value Changes

You can calculate the total time the multiple discrete tags are in a certain state. For example, you want to know the total time that two pumps were on during a 24-hour period.

The following query returns a dataset that shows the time when both discrete tags had a value of 0:

```
SELECT * FROM OPENQUERY(INSQL,'
   SELECT DateTime, Pump1, Pump2, wwResolution
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      FROM WideHistory
         WHERE DateTime >= "2012-03-08 16:00"
         AND DateTime < "2012-03-08 17:00"
         AND wwRetrievalMode="DELTA"
   ')
WHERE Pump1+Pump2=0
```

The results are:

DateTime	Pump1	Pump2	wwResolution
2012-03-08 16:00:00.0000000	0	0	67
2012-03-08 16:00:01.5980000	0	0	2521
2012-03-08 16:00:04.4470000	0	0	18500
2012-03-08 16:00:23.6000000	0	0	13995
2012-03-08 16:00:37.9140000	0	0	2625
...			

The following query shows how to return the total time when both tags had a value of 0:

```
SELECT SUM(wwResolution) FROM OPENQUERY(INSQL,'
   SELECT DateTime, Total=Pump1+Pump2, wwResolution
      FROM WideHistory
         WHERE DateTime >= "2012-03-08 16:00"
         AND DateTime < "2012-03-08 17:00"
         AND wwRetrievalMode="DELTA"
   ')
WHERE Total=0
```

The results are:

(No column name)
2551289

If you changed the ending WHERE clause to Total>0, the returned time would be for when more than one discrete tag was true.

SELECT INTO from a History Table

The following query inserts the specified data from the WideHistory table into another table called MyTable. Then, the data in the MyTable table is queried. This query uses the OPENQUERY function.

```
DROP TABLE MyTable
SELECT DateTime,
      "Sec" = datepart(ss, DateTime),
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
      "mS" = datepart(ms, DateTime),
      ReactTemp, ReactLevel
   INTO MyTable
      FROM OpenQuery(INSQL, 'SELECT DateTime, ReactTemp, ReactLevel FROM WideHistory
         WHERE wwResolution = 5000
         AND DateTime >= "2001-03-13 1:58pm"
         AND DateTime <= "2001-03-13 2:00pm" ')
SELECT * FROM MyTable
```

The results are:

DateTime	Sec	mS	ReactTemp	ReactLevel
2001-03-13 13:58:00.000	0	0	190.9	2025.0
2001-03-13 13:58:00.000	5	0	190.9	2025.0
2001-03-13 13:58:00.000	10	0	168.3	1215.0
2001-03-13 13:58:00.000	15	0	168.3	1215.0
2001-03-13 13:58:00.000	20	0	133.8	315.0
2001-03-13 13:58:00.000	25	0	133.8	315.0
2001-03-13 13:58:00.000	30	0	101.6	0.0
2001-03-13 13:58:00.000	35	0	101.6	0.0
2001-03-13 13:58:00.000	40	0	32.4	750.0
2001-03-13 13:58:00.000	45	0	32.4	750.0
2001-03-13 13:58:00.000	50	0	20.9	1700.0
2001-03-13 13:58:00.000	55	0	20.9	1700.0
2001-03-13 13:59:00.000	0	0	85.9	2000.0
2001-03-13	5	0	85.9	2000.0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

13:59:00.000				
2001-03-13 13:59:00.000	10	0	185.9	2000.0
2001-03-13 13:59:00.000	15	0	185.9	2000.0
2001-03-13 13:59:00.000	20	0	168.3	1235.0
2001-03-13 13:59:00.000	25	0	168.3	1235.0
2001-03-13 13:59:00.000	30	0	136.1	335.0
2001-03-13 13:59:00.000	35	0	136.1	335.0
2001-03-13 13:59:00.000	40	0	103.9	-25.0
2001-03-13 13:59:00.000	45	0	103.9	-25.0
2001-03-13 13:59:00.000	50	0	34.7	625.0
2001-03-13 13:59:00.000	55	0	34.7	625.0
2001-03-13 14:00:00.000	0	0	20.9	1575.0

(25 row(s) affected)

Moving Data from a SQL Server Table to an Extension Table

The following queries show how to insert manual data into a normal SQL Server table and then move it into the History extension table.

First, insert the data into the SQL Server table. The following query inserts two minutes of existing data for the SysTimeSec tag into the ManualAnalogHistory table:

```
INSERT INTO ManualAnalogHistory (DateTime, TagName, Value, Quality, QualityDetail,
wwTagKey)
   SELECT DateTime, TagName, Value, Quality, QualityDetail, wwTagKey
      FROM History WHERE TagName = 'SysTimeSec'
         AND DateTime >= '20050329 12:00:00'
         AND DateTime <= '20050329 12:02:00'
```

Then, create a manual tag using the Operations Control Management Console. For a manual tag, "MDAS/Manual

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Acquisition" is specified as the acquisition type. Be sure to commit the changes to the system. In this example, a manual analog tag named MDAS1 was created.

Finally, insert the data from the ManualAnalogHistory table into History:

```
INSERT INTO History (TagName, DateTime, Value, QualityDetail)
   SELECT 'MDAS1', DateTime, Value, QualityDetail FROM ManualAnalogHistory
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '20050329 12:00:00'
         AND DateTime <= '20050329 12:02:00'
```

Using Server-Side Cursors

Cursors are a very powerful feature of SQL Server. They permit controlled movement through a record set that results from a query.

For in-depth information on cursors, see your Microsoft SQL Server documentation.

The AVEVA Historian OLE DB Provider provides server-side cursors. Cursors can be used to do joins that are not possible in any other way. They can be used to join date/times from any source with date/times in the history tables.

The following query provides an example of using a server-side cursor. This query:

- Fetches all of the events in the EventHistory table.
- Shows a "snapshot" of three tags at the time of each event.
- Shows the event tag and its associated key value.

This query could easily be encapsulated into a stored procedure. The query uses the four-part naming convention.

```
SET QUOTED_IDENTIFIER OFF
DECLARE @DateValue DateTime
DECLARE @EventTag nvarchar(256)
DECLARE @EventKey int
DECLARE @Qry1 nvarchar(500)
DECLARE @Qry2 nvarchar(500)
DECLARE @Qry3 nvarchar(500)
SELECT @Qry1 = N'SELECT EventTag = @EventTag, EventKey = @EventKey, DateTime, TagName,
Value, Quality
   FROM History
      WHERE TagName IN (N''SysTimeSec'', N''SysTimeMin'', N''SysTimeHour'')
         AND DateTime = '''
SELECT @Qry2 = N''''
SELECT @Qry3 = N''
DECLARE Hist_Cursor CURSOR FOR
SELECT DateTime, TagName, EventLogKey
      FROM Runtime.dbo.EventHistory
OPEN Hist_Cursor
FETCH NEXT FROM Hist_Cursor INTO @DateValue, @EventTag, @EventKey
WHILE @@FETCH_STATUS = 0
BEGIN
   SELECT @Qry3 = @Qry1 + convert(nvarchar, @DateValue, 121) + @Qry2
   --PRINT @Qry3
   EXEC sp_executesql @Qry3, N'@EventTag nvarchar(256),
   @EventKey int', @EventTag, @EventKey
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
   FETCH NEXT FROM Hist_Cursor INTO @DateValue, @EventTag, @EventKey
END
CLOSE Hist_Cursor
DEALLOCATE Hist_Cursor
```

The results are:

EventTag	EventKey	DateTime	TagName	Value	Quality
SysStatusEvent	3	2001-01-12 13:00:27.000	SysTimeSec	27.0	0
SysStatusEvent	3	2001-01-12 13:00:27.000	SysTimeMin	0.0	0
SysStatusEvent	3	2001-01-12 13:00:27.000	SysTimeHour	13.0	0

(3 row(s) affected)

EventTag	EventKey	DateTime	TagName	Value	Quality
SysStatusEvent	4	2001-01-12 14:00:28.000	SysTimeSec	28.0	0
SysStatusEvent	4	2001-01-12 14:00:28.000	SysTimeMin	0.0	0
SysStatusEvent	4	2001-01-12 14:00:28.000	SysTimeHour	14.0	0

(3 row(s) affected)

Using Stored Procedures in OLE DB Queries

Any normal SQL Server stored procedure can make use of the tables exposed by the AVEVA Historian OLE DB Provider. Stored procedures can use any valid Transact-SQL syntax to access AVEVA Historian historical data.

In other words, stored procedures can make use of four-part-queries, OPENQUERY and OPENROWSET functions, cursors, parameterized queries and views. Stored procedures can be used to encapsulate complex joins and other operations for easy re-use by applications and end users.

Getting Data from the OPCQualityMap Table

In general, an OPC quality has 16 significant bits. The lower 8 bits contain the quality as described in the table, while the upper 8 bits hold server-specific information. To ensure correct results, it is important to consider only the lower 8 bits in a query or join involving the OPCQualityMap table.

For example:

```
SELECT h.DateTime, h.TagName, h.Value, o.Description FROM History h
   INNER JOIN OPCQualityMap o
   ON (h.OPCQuality & 255) = o.OPCQuality
      WHERE TagName in (…)
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND …
```

Using Variables with the Wide Table

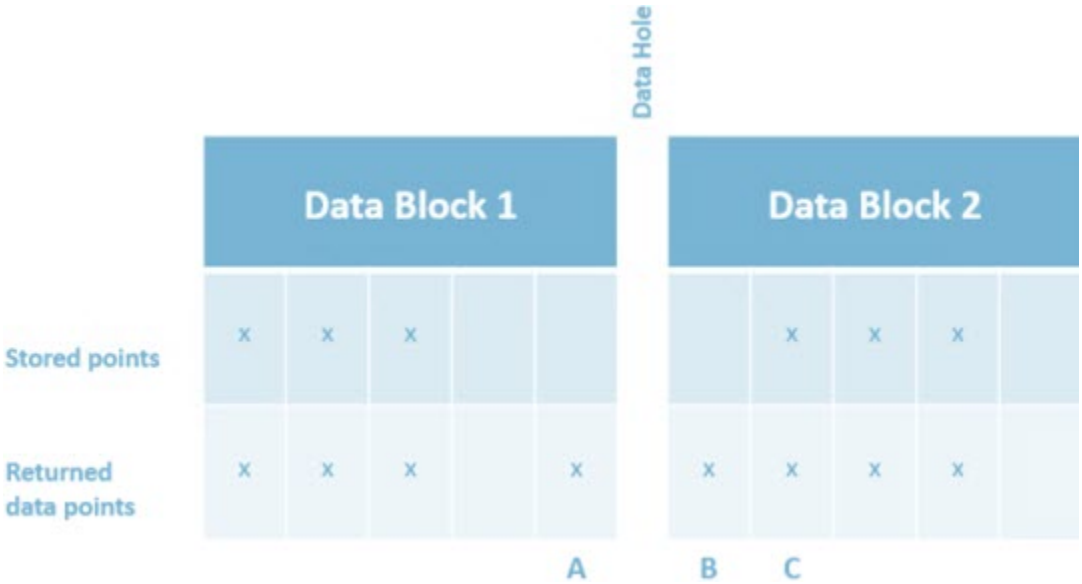
You cannot use variables in an OPENQUERY statement. Therefore, if you want to use variables in a query on the wide table, you must first build up the OPENQUERY statement "on the fly" as a string, and then execute it.

```
DECLARE @sql nvarchar(1000)
DECLARE @DateStart datetime
DECLARE @DateEnd datetime
SET @DateStart = '2001-8-29 11:00:00'
SET @DateEnd = '2001-8-29 11:11:00'
SET @sql = N'select *
FROM OPENQUERY(INSQL, ''SELECT DateTime, ReactLevel, ReactTemp, ProdLevel, BatchNumber,
ConcPump, Mixer, TransferValve, TransferPump, WaterValve, ConcValve, OutputValve,
SteamValve
   FROM WideHistory
      WHERE DateTime >= "' + CONVERT(varchar(26), @DateStart, 113) + '"
         and DateTime <= "' + CONVERT(varchar(26), @DateEnd, 113) + '"
         AND wwResolution = 1000
         AND wwRetrievalMode = "cyclic"'') '
EXEC sp_executesql @sql
```

Retrieval Across a Data Gap in Classically Stored Data

For blocks created by the Classic Storage subsystem, if the data to be retrieved spans more than one history block, and the start time of the later block is equal (within one tick) to the end time of the first block, you will not notice any difference than when querying within a single block.

However, if the system was stopped between history blocks, there will be a gap in the data, as shown in the following diagram:



Upon retrieval, additional data points (labeled A and B) will be added to mark the end of the first block's data and the beginning of the second block's data. Point C is a stored point generated by the Storage subsystem. (Upon a restart, the first value from each IDAS will be offset from the start time by 2 seconds and have a quality

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

detail of 252.)

The following paragraphs explain this in more detail.

For delta retrieval, the data values in the first block are returned as stored. After the end of the block is reached and all of the points have been retrieved, an additional data point (A) will be inserted by retrieval to mark the end of the data. The value for point A will be

Point A attribute	Value(Hex)	Value(Dec)
Value	0	0
Quality	100	256
Quality Detail	0	0

If there is no value stored at the beginning of the next block, an initial data point (B) will be inserted by retrieval and will have the snapshot initial value as stored. The quality and quality detail values are as follows:

Point B attribute	Value(Hex)	Value(Dec)
Value	Snapshot	Snapshot
Quality	0	0
Quality Detail	96	150

In the case of cyclic retrieval, a point is required for each specified time. If the time coincides with the data gap, a NULL point for that time will be generated. The inserted points will have the values defined in the following table.

Cyclic NULL point	Value(Hex)	Value(Dec)
Value	0	0
Quality	100	256
Quality Detail	0	0

If you are using time or value deadbands for delta retrieval across a data gap, the behavior is as follows:

- For a value deadband, all NULLs will be returned and all values immediately after a NULL will be returned. That is, the deadband is not applied to values separated by a NULL.
- For a time deadband, null values are treated like any other value. Time deadbands are not affected by NULLs.

Returned Values for Non-Valid Start Times

One example of a non-valid query start time is a start time that is earlier than the start time of the first history block. For delta retrieval, the first row returned will be NULL. The timestamp will be that of the query start time. The next row returned will be timestamped at the start of the history block and have the following attributes:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Point attribute	Value(Hex)	Value(Dec)
Value	Snapshot	Snapshot
Quality	0	0
Quality Detail	96	150

For cyclic retrieval, NULL will be returned for data values that occur before the start of the history block.

Another non-valid start time is a start time that is later than the current time of the AVEVA Historian computer. For delta retrieval, a single NULL value will be returned. For cyclic retrieval, a NULL will be returned for each data value requested.

Querying Aggregate Data in Different Ways

There are four different ways you can retrieve summary data, such as an average, using the Historian.

- Using the SQL Server average function. This is appropriate for discrete samples. For example, a check weigher, where you are measuring individual units against a target weight.
- Using the average retrieval mode. This is appropriate for most situations where you want to find an average, as it is weighted according to time. For example, if you want to find the average for a flow rate or a temperature.
- Setting up summary replication and then querying the AnalogSummaryHistory table. Replication uses the average retrieval mode to do the calculations.
- Setting up a summary event and then querying the SummaryData table. The Event subsystem uses the SQL Server average function.

The following examples show how you can get the same data using these different methods. All examples use the SysTimeSec system tag, which has a range of 0 to 59.

Query 1

The following query uses the SQL Server average function to return the average value of the SysTimeSec tag over the span of one minute.

```
SELECT AVG(Value) as 'SysTimeSec AVG'
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime > '2009-11-15 6:30:00'
         AND DateTime < '2009-11-15 6:31:00'
         AND wwRetrievalMode = 'Full'
```

The results are:

SysTimeSec AVG
29.5

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Query 2

The following query uses the historian time-weighted average retrieval mode to return the average for the same time period. Because the cycle count is set to 2, a first row is returned for the "phantom"cycle leading up to the query start time. The StartDateTime column shows the time stamp at the start of the data sampling, which is the start time of the phantom cycle. The second row returned reflects is the actual data that you expect. The time stamp for the data value is 2009-11-15 06:31:00 because the default time stamping rule is set so that the ending time stamp for the cycle is returned. For more information about the phantom cycle, see About Phantom Cycles.

```
SELECT StartDateTime, DateTime, TagName, Value
   FROM History
      WHERE TagName = 'SysTimeSec'
         AND DateTime >= '2009-11-15 6:30:00'
         AND DateTime <= '2009-11-15 6:31:00'
         AND wwRetrievalMode = 'Average'
         AND wwCycleCount = 2
         AND wwTimeStampRule = 'end'
```

The results are:

StartDateTime	DateTime	TagName	Value
2009-11-15 06:29:00	2009-11-15 06:30:00	SysTimeSec	29.5
2009-11-15 06:30:00	2009-11-15 06:31:00	SysTimeSec	29.5

Query 3

For the following query, local replication has been set up so that the average of the SysTimeSec tag is calculated every minute and stored to the SysTimeSec.1M analog summary tag. The query returns the value of the SysTimeSec.1M tag for the time period specified.

```
SELECT TagName, StartDateTime, EndDateTime, Average as AVG
   FROM AnalogSummaryHistory
      WHERE TagName = 'SysTimeSec.1M'
         AND StartDateTime >= '2009-11-15 6:30:00'
         AND EndDateTime <= '2009-11-15 6:31:00'
```

The results are:

TagName	StartDateTime	EndDateTime	AVG
SysTimeSec.1M	2009-11-15 06:30:00	2009-11-15 06:31:00	29.5

Query 4

The following query, the History table is used instead of the AnalogSummaryHistory table. Because the cycle count is set to 2, this query returns a row for the phantom cycle. The time stamp for the data value is 2009-11-15 06:31:00 because the default time stamping rule is set so that the ending time stamp for the cycle is returned.

```
SELECT TagName, DateTime, Value
   FROM History
      WHERE TagName = 'SysTimeSec.1M'
         AND DateTime >= '2009-11-15 6:30:00'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND DateTime <= '2009-11-15 6:31:00'
        AND wwRetrievalMode = 'avg'
        AND wwCycleCount = 2
```

The results are:

TagName	DateTime	Value
SysTimeSec.1M	2009-11-15 06:30:00	29.5
SysTimeSec.1M	2009-11-15 06:31:00	29.5

Query 5

The following query returns five minutes of summary data for an event tag that has been configured to store the average value of the SysTimeSec tag every minute.

```
SELECT TagName, CalcType, SummaryDate, Value
   FROM v_SummaryData
      WHERE TagName = 'SysTimeSec'
         AND SummaryDate >= '2009-11-15 18:30:00'
         AND SummaryDate <= '2009-11-15 18:31:00'
```

The results are:

TagName	CalcType	SummaryDate	Value
SysTimeSec	AVG	2009-11-15 18:30:00.000	29.5
SysTimeSec	AVG	2009-11-15 18:31:00.000	29.5

Bitwise Retrieval for Process Data

It is common to pack multiple digital states into the same PLC register as an integer rather than as individual bits. You can still map the individual bits to separate Historian tags for most DAServers/PLCs, but if you instead map the entire integer to a single Historian tag, you can address individual bits using standard SQL Server queries.

For example, consider the following query that returns process data values for the 'SysTimeMin' tag:

```
SELECT Value FROM dbo.History WHERE TagName = 'SysTimeMin'
```

However to get more bits of data, you can add 2 (bitposition-1) and use bitwise & operator on the Value column as shown in the following query. Using the the Integer cast, you can query a maximum of 32 bits.

```
SELECT
CONVERT(BIT, CAST(Value AS INT) & 1) As 'Bit0',
CONVERT(BIT, CAST(Value AS INT) & 2) As 'Bit1',
CONVERT(BIT, CAST(Value AS INT) & 4) As 'Bit2',
CONVERT(BIT, CAST(Value AS INT) & 8) As 'Bit3',
CONVERT(BIT, CAST(Value AS INT) & 16) As 'Bit4',
CONVERT(BIT, CAST(Value AS INT) & 32) As 'Bit5',
CONVERT(BIT, CAST(Value AS INT) & 64) As 'Bit6',
CONVERT(BIT, CAST(Value AS INT) & 128) As 'Bit7'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
FROM dbo.History WHERE TagName = 'SysTimeMin'
```

The results are:

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7
0	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
0	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0
1	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
1	0	1	1	0	0	0	0
0	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0

SQL Queries for Alarms and Events

Note: The alarm and event history functionality described in this chapter captures detailed histories from Application Server. This functionality should not be confused with the Classic Event subsystem, which allows for some basic events tracking and is based on historical data. The examples provided in this chapter only work when the Historian is configured to use high-speed alarms and events storage.

AVEVA Historian captures process data about your plant. In addition to real-time and historical data, this includes information about events.

Events are like other process data – for example, temperature – because their values can change over time. Events differ from other process data in these ways:

- Events usually change more slowly.
- Events usually are more complex than simply a value, time, and quality.

Event data answers questions like "When did this setpoint change and who changed it?" The event record could include the name of the operator, the workstation from where the change was made, any comment about the change, the name of the person who verified the change, and other related details.

Alarms are a specific kind of event. They represent state changes and have an associated lifecycle. This lifecycle includes these states (usually in this order):

- Set – For example, when a temperature goes too high.
- Acknowledged – That is, when an operator recognizes it as an alarm and, ideally, addresses it.
- Clear – For example, when the temperature returns to normal.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Alarms may also have other states, but these are the key ones.

You can query the Events view, which references the History table, to track and analyze alarms and other events. Because Events is actually an extension table (like History), its data is stored in blocks, not in SQL Server tables.

Note: The Events view does not expose all application-specific columns that may be stored by Historian. (Such columns can be queried from the REST/OData interface.) Also, it is not unusual for Events columns to contain many NULL values.

For more information about the Events view, see Events in the *AVEVA Historian Database Reference*.

Querying Alarms and Events

Querying Alarm and Event data is helpful in keeping track of your production environment. For example, as an operator, you could query the data to find out answers to these questions:

- How often alarm events occur?
- Where in the facility they occur?
- How critical the alarms are?
- How quickly they are addressed and cleared?

  Datetime in Alarm and Event Queries

All queries of alarm and event data must include at least one datetime column.

Alarm and event queries use two ways to express time:

- **UTC time**
  This format is used by EventTimeUtc.
- **Local time for the location of the Historian server**
  This is used by columns like EventTime.

The value of wwTimeZone affects any datetime recorded in local time, but does not affect EventTimeUtc.

Example: Listing all events

One of the simplest queries for alarms and events data is to get a list of all events.

For example,a query like this one would list all events between the dates and times specified:

```
SELECT *
   FROM Events
   WHERE EventTime between '2015-10-25 0:00' and '2015-10-26 0:00'
```

Example: How often alarms occur

This query reports the average alarm rate on hourly basis.

```
DECLARE @StartTime as varchar(60)
DECLARE @EndTime as varchar(60)
SET @StartTime = '2015-10-25 12:00:00'
SET @EndTime = '2015-10-26 12:00:00'
DECLARE @AlarmRaise table
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
(
   EventTime nvarchar(60),
   ID nvarchar(50),
   AlarmState nvarchar(20),
   SourceArea nvarchar(20),
   SourceObject nvarchar(20)
)
INSERT @AlarmRaise select EventTime,Alarm_ID,Alarm_State,Source_Area,Source_Object from
Events where EventTime > @StartTime and EventTime < @EndTime and Alarm_State='UNACK_ALM'
--=======================--
DECLARE @AlarmCounts table
(
   ForDate nvarchar(60),
   OnHour nvarchar(50),
   CountperHour nvarchar(20)
)
INSERT @AlarmCounts SELECT CAST(EventTime as date) AS ForDate,
DATEPART(hour,EventTime) AS OnHour,
Count(*) AS "CountperHour"
FROM @AlarmRaise
GROUP BY CAST(EventTime as date),
DATEPART(hour,EventTime)
SELECT Avg(CAST(CountperHour as INT)) as "Average Alarm Rate on Hourly Basis" from
@AlarmCounts
```

The resulting report looks like this:

Average Alarm Rate on Hourly Basis
6

Example: Most frequent alarm per hour

This query reports the most frequent alarms for each hour:

```
DECLARE @StartTime as varchar(60)
DECLARE @EndTime as varchar(60)
SET @StartTime = '2017-11-10 12:00:00'
SET @EndTime = '2017-11-10 12:10:00'
DECLARE @AlarmRaise table
(
   EventTime nvarchar(60),
   ID nvarchar(50),
   AlarmState nvarchar(20),
   SourceArea nvarchar(20),
   SourceObject nvarchar(20),
   SourceConditionVariable nvarchar(40)
)
INSERT @AlarmRaise select
EventTime,Alarm_ID,Alarm_State,Source_Area,Source_Object,Source_ConditionVariable from
Events where EventTime > @StartTime and EventTime < @EndTime and Alarm_State='UNACK_ALM'
--=======================--
SELECT CAST(EventTime as date) AS ForDate,
   DATEPART(hour,EventTime) AS OnHour,
   Count(*) AS "Count per Hour",
      SourceObject + SourceConditionVariable AS "Alarm Attribute"
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
FROM @AlarmRaise
GROUP BY CAST(EventTime as date),
   DATEPART(hour,EventTime),
      SourceObject,
      SourceConditionVariable
ORDER BY ForDate ASC,OnHour,[Alarm Attribute]
```

This resulting report looks like this:

ForDate	OnHour	Count per Hour	Alarm Attribute
2017-11-10	12	10	AlarmHeartBeatAlarmHeartBeat.AlmHeartBeat.Hi
2017-11-10	12	2	Reactor_31Reactor_31.ReactLevel.Hi
2017-11-10	12	2	Reactor_31Reactor_31.ReactLevel.Lo
2017-11-10	12	2	Reactor_31Reactor_31.ReactTemp.Hi
2017-11-10	12	1	StorageTank_31StorageTank_31.ProdLevel.Lo
2017-11-10	12	6	VectorTagsVectorTags.VectorX.Hi
2017-11-10	12	3	VectorTagsVectorTags.VectorX.HiHi
2017-11-10	12	6	VectorTagsVectorTags.VectorX.Lo
2017-11-10	12	3	VectorTagsVectorTags.VectorX.LoLo
2017-11-10	12	2	VectorTagsVectorTags.VectorZ.Hi

Example: Pinpointing where alarms occur

This query reports the number of alarms raised from each source by area and by object.

```
DECLARE @StartTime as varchar(60)
DECLARE @EndTime as varchar(60)
SET @StartTime = '2015-10-25 12:00:00'
SET @EndTime = '2015-10-26 12:00:00'
DECLARE @AlarmRaise table
(
   EventTime nvarchar(60),
   ID nvarchar(50),
   AlarmState nvarchar(20),
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
    SourceArea nvarchar(20),
    SourceObject nvarchar(20)
)
INSERT @AlarmRaise select EventTime,Alarm_ID,Alarm_State,Source_Area,Source_Object from
Events where EventTime > @StartTime and EventTime < @EndTime and Alarm_State='UNACK_ALM'
--=======================--
SELECT SourceArea AS "Source Area/Object" , count(*) AS "Total Number of Alarms" from
@AlarmRaise GROUP BY SourceArea UNION
SELECT SourceObject AS "Source Area/Object" , count(*) AS "Total Number of Alarms" from
@AlarmRaise GROUP BY SourceObject;
```

The results look like this:

Source Area/Object	Total Number of Alarms
Area_001	6
UserDefined_001	6

Example: Showing average time to clearing an alarm

This query reports the average time to clear Critical, High, Medium, and Low alarms per hour.

```
DECLARE @start DateTime2
SET @start = '2017-12-11'
DECLARE @end DateTime2
SET @end = '2017-12-12'
-- ack time per severity per hour for high, medium and low
SELECT DATEADD(hour, DATEDIFF(hour, 0, e.EventTime), 0) as hour,
    e.Severity,
    avg(Alarm_UnAckDurationMs) as avg_unack,
    count(*) as count
FROM Events e
    WHERE
        e.EventTime < @end
        AND e.EventTime >= @start
        AND e.Severity <=3 -- critical = 1, high = 2 medium = 3 low = 4
        AND e.Type = 'Alarm.Acknowledged'
GROUP BY
    DATEADD(hour, DATEDIFF(hour, 0, e.EventTime), 0),
        severity
ORDER BY
    DATEADD(hour, DATEDIFF(hour, 0, e.EventTime), 0),
        e.severity
-- ack time by user by hour
SELECT DATEADD(hour, DATEDIFF(hour, 0, e.EventTime), 0) as hour,
    avg(Alarm_UnAckDurationMs) as avg_unack,
    e.User_Name,
    count(*) as count
FROM Events e
    WHERE
        e.EventTime < @end
        AND e.EventTime >= @start
        AND e.Type = 'Alarm.Acknowledged'
GROUP BY
    DATEADD(hour, DATEDIFF(hour, 0, e.EventTime), 0),
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        e.User_Name
ORDER BY
   DATEADD(hour, DATEDIFF(hour, 0, e.EventTime), 0),
        e.User_Name
```

This query results in two reports. The first one looks like this:

hour	Severity	avg_unack	count
2017-12-11 08:00:00.000	2	330949	73
2017-12-11 08:00:00.000	3	13723786	1
2017-12-11 09:00:00.000	2	23524	195
2017-12-11 09:00:00.000	3	4931	1
2017-12-11 10:00:00.000	2	22550	182
2017-12-11 11:00:00.000	2	24552	189
2017-12-11 12:00:00.000	2	22474	189
2017-12-11 13:00:00.000	2	23492	192
…			

The second report looks like this:

hour	avg_unack	User_Name	count
2017-12-11 08:00:00.000	453722	DefaultUser	92
2017-12-11 09:00:00.000	24997	DefaultUser	239
2017-12-11 10:00:00.000	22751	DefaultUser	222
2017-12-11 11:00:00.000	25528	DefaultUser	231
2017-12-11 12:00:00.000	23549	DefaultUser	233
2017-12-11 13:00:00.000	23807	DefaultUser	236
2017-12-11 14:00:00.000	25472	DefaultUser	237
2017-12-11 15:00:00.000	25350	DefaultUser	237
…			

Example: Evaluating response time for alarms

This query reports when an alarm is raised, acknowledged, and cleared. The report lists both times and duration.

```
DECLARE @StartTime datetime2
DECLARE @EndTime datetime2
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
DECLARE @TagName nvarchar(100)
SET @EndTime = getutcdate()
SET @StartTime = dateadd(n,-5000,@EndTime)
DECLARE @AlarmState TABLE
(
   EventTime datetime2 NOT NULL,
   Alarm_ID uniqueidentifier NOT NULL,
   TypeID int,
   Duration int,
   Alarm_State nvarchar(10),
   AckedBy nvarchar(50),
   AckedFromNode nvarchar(50),
   primary key clustered (EventTime, Alarm_ID, TypeID)
)
INSERT @AlarmState
   SELECT EventTime, Alarm_ID,
      TypeID=CASE Type WHEN 'Alarm.Clear' then 1 WHEN 'Alarm.Acknowledged' then 2 END,
      Duration=CASE Type WHEN 'Alarm.Clear' then Alarm_DurationMs WHEN
      'Alarm.Acknowledged' then Alarm_UnAckDurationMs END,
      Alarm_State, User_Account, User_NodeName
   FROM Events
   WHERE EventTime > @StartTime AND EventTime < @EndTime
      AND Type IN ('Alarm.Clear','Alarm.Acknowledged')
--=======================--
SELECT
   e.EventTime as AlarmTime,
   e.Source_ProcessVariable as TagName,
   e.Alarm_Type as Type,
   e.ValueString as Value,
   e.Alarm_LimitString as Limit,
   e.Priority,
   e.Source_Area as Area,
   CASE
      WHEN a.EventTime IS NULL and c.EventTime IS NOT NULL then c.Alarm_State
      WHEN a.EventTime IS NOT NULL and c.EventTime IS NULL then a.Alarm_State
      WHEN a.EventTime > c.EventTime then a.Alarm_State
      WHEN a.EventTime < c.EventTime then c.Alarm_State
      ELSE e.Alarm_State END as AlarmState,
   c.EventTime as ClearTime,
   a.EventTime as AcknowledgeTime,
   c.Duration/1000.0 as SecsInAlarm,
   a.Duration/1000.0 as SecsUnAck,
   e.Source_Object as Object,
   e.Source_HierarchicalArea as HierarchicalArea,
   a.AckedFromNode,
   a.AckedBy
FROM Events e
   LEFT OUTER JOIN @AlarmState c
      ON c.Alarm_ID=e.Alarm_ID
      AND c.TypeID=1
   LEFT OUTER JOIN @AlarmState a
      ON a.Alarm_ID=e.Alarm_ID
      AND a.TypeID=2
   WHERE e.EventTimeUtc BETWEEN @StartTime AND @EndTime
      AND e.Type='Alarm.Set'
ORDER BY AlarmTime ASC
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The results look like this:

AlarmTime	TagName	Type	Value	Limit	Priority	Area	AlarmState	ClearTime	AcknowledgeTime	SecsInAlarm	SecsUnAck	Object	HierarchicalArea	AckedFromNode	AckedBy
2022-02-05 23:28:11.0...	LocalRamp.Float	Hi	91.0	90.0	700	Site1	UNACK_RTN	2022-02-05 23:28:30.0...	NULL	18.999000	NULL	LocalRamp	Main.Site1	NULL	NULL
2022-02-05 23:28:16.0...	LocalRamp.Float	HiHi	96.0	95.0	950	Site1	UNACK_RTN	2022-02-05 23:28:25.0...	NULL	8.999000	NULL	LocalRamp	Main.Site1	NULL	NULL
2022-02-05 23:29:51.0...	LocalRamp.Float	Lo	9.0	10.0	300	Site1	UNACK_RTN	2022-02-05 23:30:10.0...	NULL	19.000000	NULL	LocalRamp	Main.Site1	NULL	NULL
2022-02-05 23:29:56.0...	LocalRamp.Float	LoLo	4.0	5.0	50	Site1	UNACK_RTN	2022-02-05 23:30:05.0...	NULL	8.999000	NULL	LocalRamp	Main.Site1	NULL	NULL
2022-02-05 23:31:03.1...	M22.Temp	ROCLo	-0.207247973358425	0.2	100	Site1	ACK_RTN	2022-02-05 23:31:13.5...	2022-02-05 23:35:24.0...	10.483000	260.973000	M22	Main.Site1	NULL	NULL
2022-02-05 23:31:31.0...	LocalRamp.Float	Hi	91.0	90.0	700	Site1	UNACK_RTN	2022-02-05 23:31:50.0...	NULL	19.000000	NULL	LocalRamp	Main.Site1	NULL	NULL
2022-02-05 23:31:36.0...	LocalRamp.Float	HiHi	96.0	95.0	950	Site1	UNACK_RTN	2022-02-05 23:31:45.0...	NULL	9.000000	NULL	LocalRamp	Main.Site1	NULL	NULL

Browser-Friendly Data Retrieval

Historian Data REST API: Customize data retrieval

With the Historian Data REST API, you can make data retrieval requests.You can submit requests to the Historian Data REST API using a web browser or client-side applications such as:

- Microsoft Excel (2013, 2016, or Office 365)
- Business Intelligence (BI) systems, such as Tableau and Microsoft Power BI

Supported OData features

OData is an industry standard for querying and updating data from a variety of sources. The implementation of OData for the Historian Data REST API includes support for:

- JSON and atom formats.
- OData version 4 with version 2 of the Historian Data REST API.
- OData version 3 with version 1 of the Historian Data REST API (not recommended).
- Pagination. That is, if your request returns more than 5000 results, they will be returned in pages of up to 5000 records. Each page will include a link to retrieve the next page of records.
- A subset of the OData system query options.

For more information, see OData.org JSON Verbose Format specification.

Recommendation: For best results, when you want to view the data returned by the Historian Data REST API, use the JSONView extension for the Chrome browser.

Supported versions

AVEVA Historian Client Web supports versions 1 and 2 of the Historian Data REST API.

Version 2

This is the current and recommended version of the Historian Data REST API.

Version 2 of this REST AP is based on version 1 and includes further enhancements. Version 2 includes these differences from version 1:

- The `datetimeoffset` parameter is not supported as part of the `DateTime` syntax.
- While Raw and ProcessValue entities use `DateTime`, summary entities use `StartDateTime` and

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

EndDateTime.

- For most version 2 queries, single quotes are not used for DateTime. For example:

```
http://localhost:32569/Historian/v2/ProcessValues
?$filter=DateTime+gt+2017-07-13T00:00:00
```

However, when querying events, single quotes are required for DateTime. For example:

```
http://localhost:32569/Historian/v2/Events
?$filter=EventTime+gt+'2017-07-13T00:00:00'
```

By contrast, version 1 queries do use single quotes for DateTime. For example:

```
http://localhost:32569/Historian/v1/ProcessValues
?$filter=DateTime+gt+datetimeoffset'2017-07-13T00:00:00'
```

- TagProperty and Events entity types are now open type. That is, dynamic properties can be added to the response at runtime. This can be verified using $metadata endpoint URL:

```
http://localhost:32569/Historian/v2/$metadata
```

where there will be additional attribute, OpenType="true", under <EntityType> section.

- The Tags resource returns all the properties (fixed and extended) for a tag. A tag's extended properties will be added to the response only if they exist. (The extended property name will not be listed for tags that do not have a given extended property.)
- The combined Summary resource is not supported in version 2. Use the individual AnalogSummary and StateSummary resources instead to retrieve the summary of a tag.
- Version 2 adds support for the OData contains function for applicable resources and properties.

Version 1

This is the original version of Historian Data REST API based on the OData v4 specification.

- Version 1 uses the DateTime format used in this example:

```
http://localhost:32569/Historian/v1
/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'
+and+StartDateTime+ge+datetimeoffset'2016-05-14T00:00:00.000-07:00'
+and+EndDateTime+le+datetimeoffset'2016-05-16T00:00:00.000-07:00'
```

iHistory and Account Authentication

The iHistory web service requires users to be authenticated via a login process before they can retrieve data. The process for this differs between AVEVA Historian Client Web and AVEVA Insight:

- **AVEVA Historian Client Web**
  Uses Windows integrated security. A user must belong to the aaAdministrators, aaPowerUsers, or aaUsers Windows group. The iHistory web service uses Negotiate authentication, which is supported by most modern browsers and web service clients (such as Microsoft Excel).
- **AVEVA Insight (cloud-based)**
  Uses OpenID Connect and Basic authentication. Users must be invited to a specific account within AVEVA Insight and can then access all data published to that account. When using OpenID Connect, the iHistory web service uses "bearer token" authentication, per the OpenID Connect standard. As relatively new standard, some web service clients (for example, Microsoft Excel) do not have native support for it. However, some of those same clients (for example, Microsoft Power BI) support Basic authentication.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

When you query on-premises data via iHistory, you will be prompted for authentication to the Historian.

- If you have a valid Windows login on the historian, you can retrieve general information about the services and the kinds of data available.
- If you belong to the aaAdministrators, aaPowerUsers, or aaUsers Windows group, you can retrieve actual application data -- including events, process history, tag information, and so on.

Data retrieval

The Historian Data REST API allows you to retrieve data from your solution. Data retrieval with the Historian Data REST API is available through both Historian Client Web and Historian Client Web, so ensure that you refer to the correct syntax for your product from the examples below.

Forming retrieval requests

Use the following syntax for retrieval requests, depending on whether you are using HTTP or HTTPS:

```
http://<servername>:32569/Historian/<api_version>/<resource>?<query>
https://<servername>:32573/Historian/<api_version>/<resource>?<query>
```

Note: 32569 and 32573 are the default ports for HTTP and HTTPS. If your environment is configured to use different ports for either protocol, replace the ports used in this document with the appropriate ports for your environment.

The URL includes the following elements:

Syntax element	Explanation
<*servername*>:32569 or <*servername*>:32573	The servername and port number used by AVEVA Historian Client Web. If the server is the local computer, this could be "localhost:32569" for HTTP or "localhost:32573" for HTTPS.
Historian/<*api_version*>	The API version. For example, "v2".
<*resource*>	The Historian Data REST API retrieval resource. See the complete list.
?<*query*>	A query using OData parameters and operators

For example, this retrieval request gets analog summary data via an endpoint URL using basic authentication:

```
http://localhost:32569/Historian/v2/AnalogSummary
?$filter=FQN+eq+'Depot.Train09'+and+StartDateTime+ge+2017-06-09T09:00:00-07:00+and
+EndDateTime+ge+2017-06-09T10:00:00-07:00&Resolution=3600000
```

Supported OData parameters

These parameters are supported for system query via the Historian Data REST API. They are all case-sensitive.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Parameter	Description
$filter	Specifies an expression or function that must evaluate to true for a record to be returned in the collection. A $filter expression supports references to properties and literals. Literal values include: • Strings enclosed in single quotes • Numbers and Boolean values (true or false) Filtering for process value and summary data is case-sensitive. Filtering for other values, however, is case-insensitive. For example, if you filter property values based on a value of "true," values such as "TRUE," "True," and "true" could be returned. The case returned in the query results reflects the case of the stored value. Note that you can use $filter for FQN, but only for exact matches. Use the TagFilter parameter for partial FQN matches. $filter is typically used to specify a start and end date and time for the queried period. This is required for endpoints that return time-series data, but is recommended for most queries in order to reduce server load and make the number of returned results easier to work with.
$select	Specifies a subset of properties to return.
$skip	Specifies the number of records to skip from the beginning of the result set.
$skiptoken	Used to get the next record set that satisfies the query conditions. This parameter cannot be used in queries, but you will see it upon query execution.
$top	Specifies the maximum number of records to return. This subset is formed by selecting only the first *N* items of the set, where *N* is a positive integer specified by this query parameter.

Time-series extension parameters

The Historian Data REST API supports the following custom extension parameters in retrieval queries to better control your search results.

- **RetrievalMode** specifies how the resulting data is calculated for Raw and ProcessValues entities. Valid values are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

• Average	• Cyclic	• Integral	• Minimum
• BestFit	• Delta	• Interpolated	• Slope
• Counter	• Full	• Maximum	

- **Resolution** specifies the granularity of data returned for Raw, ProcessValues, and Summary entities.
- **TagFilter** is used to specify which tags to return data on. TagFilter must be used with $filter, which defines the range of results. TagFilter only works with the ProcessValues call. TagFilter supports the following operators:
  - startswith
  - endswith
  - contains

Supported operators

These logical operators are supported for queries:

Operator	Description
eq	Equal
ne	Not equal
gt	Greater than
ge	Greater than or equal
lt	Less than
le	Less than or equal
and	Logical and
or	Logical or
not	Logical negation

Note: Use "%20" to indicate a space. Use "%27" to indicate a single quote.
If you are using the JSONView viewer in the Chrome browser, you can use a plus sign (+) to indicate a space to make the URI string more readable.

If the expression includes multiple values for the criteria, you must specify each criteria separately using the "or" operator. For example:

```
...
((Priority+eq+100+or+Priority+eq+200+or+Priority+eq+500
+or+Priority+eq+700)+and+(filter …))
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Syntax for combining expressions

- Use **and** to use multiple terms for the same parameter. For example:
  $filter=DateTime+ge+2023-08-31T00:00:00z+**and**+DateTime+le+2023-08-31T12:00:00z
- Use **&** to join separate parameters. For example:
  $filter=DateTime+ge+2023-08-31T00:00:00z**&**TagFilter=startswith(FQN, 'turbine')
- In a query, you can have only a single expression specifying a period of time combined with a single expression filtering specific tags. Combine expressions with the "and" operator. The expression itself can be complex, using any of the supported logical operators. Use parentheses ( ) to create precedence groups within an expression.

Retrieval resources

The Historian Data REST API exposes various resources through an endpoint URL that is specific to your Insight solution.

This API includes the following resources for retrieving data:

Process Data and Events Resources	Tag Property Resources	Resources for Version 1 Only
• ProcessValues • AnalogSummary • StateSummary • Events	• Tags • TagProperties • TagPropertyValues • TagGroups • TagSuggest • TagSearch • TagExtendedProperties	• Summary • Daily • Hourly • Minutely

Note: All property names are case-sensitive. Event storage preserves the case that you provide for any property value. For example, a property value of "TRUE" is different than "True" and "true."

ProcessValues

Description	Retrieves a set of process value records (where each record includes value + time + quality, or VTQ) for the specified tags.
URL	`/ProcessValues`
Method	`GET`
Required Parameters	• **FQN=**[string]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z • **EndDateTime=**[DateTimeOffset] The ending date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z
Optional Parameters	• **OPCQuality=**[Int32] The data quality as reported by the source. • **Value=**[Double] 0 or 1. • **Bounding=**[true] If the query specifies a start point, this parameter shows the first data point that occurs on or before the query start. If the query specifies an end point, this parameter shows the first data point that occurs on or after the query end. The actual date and time of the start/end point are shown, even if it's outside the bounds of the query. • **Text=**[String] For string tags, contain the value. For discrete tags, contains the message associated the value (0 or 1). • **TagFilter=**[String] Allows you to use a query string using OData filter query notation to filter results by tag attributes. Supported tag attributes include FQN, description. TagFilter can include up to 20 AND clauses, or up to 20 OR clauses. You cannot mix AND and OR in the same query. Operators should be lowercase, and searches are case-insensitive. *Valid Format:* `Historian/v2/ProcessValues?TagFilter=startswith(Source,'MVDS') and TagType eq 'string'`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Invalid Format:
	`Historian/v2/ProcessValues?TagFilter=startswith(Source,'MVDS') And TagType eq 'string'`
	• **Expression=**[String]
	For analog tags, you can perform UOM conversion on retrieved tag data by including a string expression in the following format:
	`UOM([tag FQN 1],[engineering unit name 1];[tag FQN 2],[engineering unit name 2];...)`
	You can provide one or more pairs of tag names and engineering unit names, and during retrieval the data will be converted to the specified units, if valid. If the unit conversion is not valid, the tag's base unit is used instead.
	Note: This feature requires special licensing. If a license is not available and the "Expression" parameter is used, no data is retrieved and an error message is displayed.

Success Response	`Code: 200` `Content: { fqn: plant12.pump6, datetime: 2016-09-03T18:44:09.352247Z}`

Error Response	`Code: 404 NOT FOUND`
	`Content: { error : "FQN doesn't exist" }`
	or
	`Code: 401 UNAUTHORIZED`
	`Content: { error : "You are unauthorized to make this request." }`

Sample Query	**Scenario 1**
	This query returns process values for a specific tag identified by its fully qualified name (*datasource.tagname*). Using a "$filter" clause, it specifies a tag named tank_level within the Baytown data source. The result is a list of values for the tank_level tag.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'`
	Scenario 2
	This query includes a start date time, end date time, and other query parameters.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_l`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`evel'+and+DateTime+ge+2014-07-04T23:57:29Z+and+DateTime+le+2014-07-05T00:02:29Z&RetrievalMode=BestFit&Resolution=6500`

Sample Query	**Scenario 3**
	This query returns the current day of the month, and uses the Expression parameter to retrieve the value in hours.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'SysDateDay'&Expression=UOM([SysDateDay],[Hour])`
	Sample Output
	`{ "odata.metadata": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [ { "FQN": "SysDateDay", "DateTime": "2020-10-29T16:57:29.0128451Z", "OpcQuality": 192, "Value": 696.0, "Text": "696", "Unit": "Hour" } ] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 4**
	This query returns process values for a list of tags with fully qualified names (FQN) beginning with "sys", with values falling within a specific date range. If you don't know the fully qualified name of a specific tag, you can use the TagFilter parameter to get a list of possible matches.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=startswith(FQN,'sys')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [{ "FQN": "SysClassicDataRedirector", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "" }, { "FQN": "SysClientAccessPoint", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "" }, { "FQN": "SysConfigStatus", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units" }] }`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 5**
	This query returns process values for a list of tags whose FQN ends with "sec", with values falling within a specific date range.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=endswith(FQN,'sec')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [{ "FQN": "SysDataAcqOverallItemsPerSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units/s" }, { "FQN": "SysStatusRxEventsPerSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "units/s" }] }`

Sample Query	**Scenario 6**
	This query returns process values for a list of tags whose FQN contains "timesec", with values falling within a specific date range.
	`http://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=contains(FQN,'timesec')`
	Sample Output
	`{ "@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues", "value": [ { "FQN": "SysTimeSec", "DateTime": "2019-01-15T03:57:29Z", "OpcQuality": 0, "Value": null, "Text": null, "Unit": "Second"`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`}`
	`]`
	`}`

Sample Query	**Scenario 7**
	This query returns process values for a list of tags whose description contains "timesec", with values falling within a specific date range.
	`https://localhost:32569/Historian/v2/ProcessValues?$filter=DateTime+ge+2019-01-15T03:57:29Z+and+DateTime+le+2021-01-17T03:50:54.881Z&TagFilter=Contains(Description,'timesec')`
	Sample Output
	`{`
	`"@odata.context": "http://localhost:32569/Historian/v2/$metadata#ProcessValues",`
	`"value": [`
	`{`
	`"FQN": "SysTimeSec",`
	`"DateTime": "2019-01-15T03:57:29Z",`
	`"OpcQuality": 0,`
	`"Value": null,`
	`"Text": null,`
	`"Unit": "Second"`
	`}`
	`]`
	`}`
	Similar to the previous examples, you can search for tags with descriptions that start with, end with, or contain specific text.

AnalogSummary

Description	Retrieves analog statistics for the specified tags.
URL	`/AnalogSummary`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Method	GET
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z • **EndDateTime=**[DateTimeOffset] The ending date and time. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 1985-04-12T23:20:50.52435Z
Optional Parameters	• **RetrievalMode=**[string] Possible values are: Cyclic, Full. Default is Cyclic. • **Resolution=**[Int] In milliseconds. Any positive integer. • **SliceBy=**[Int|Discrete|String] Performs dynamic resolution/cycle computation by tags. Returns one Analog Summary value per tag per dynamic cycle with start and end date time. SliceBy can support up to 10 tags. For more information on using SliceBy, see Using SliceBy. • **SliceByValue=**[string] Specifies the filter criterion to get the summary values for SlicedBy, based on that filter value. • **OPCQuality=**[Int32] Defines the OPC quality for the data. Normal OPC quality retrieval logic is applied if all the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **First=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the first point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **FirstDateTime=**[DateTimeOffset] Time stamp associated with first value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle. • **Last=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the last point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **LastDateTime=**[DateTimeOffset] Time stamp associated with last value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle. • **Minimum=**[Double] If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the minimum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **MinDateTime=**[DateTimeOffset] Time stamp associated with Min value. NULL if Min is NULL. • **Maximum=**[Double]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the maximum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned. • **MaxDateTime=**[DateTimeOffset] Time stamp associated with Max value. NULL if Max is NULL.
	• **Average=**[Double] Time weighted average value of retrieval cycle. This is calculated by using the individual summary averages. The calculation is "Sum(average * delta t) / Total time of average in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Average query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) • **StdDev=**[Double] Time weighted standard deviation value of the retrieval cycle. The value is calculated using time weighted sums (Integrals) and time weighted sums of squares (IntegralOfSquares) values, prorated for any partially contained storage cycles. For analog tags, similar StdDev values are produced for each cycle. • **Integral=**[Double] Area under value curve of retrieval cycle. The calculation is "Sum(value * delta t) / Total time of integral in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Integral query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) For analog tags, similar Integral values are produced for each cycle.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Count=**[Double] Number of values in a particular cycle. • **Expression=**[String] For analog tags, you can perform UOM conversion on retrieved tag data by including a string expression in the following format: `UOM([tag FQN 1],[engineering unit name 1];[tag FQN 2],[engineering unit name 2];...)` You can provide one or more pairs of tag names and engineering unit names, and during retrieval the data will be converted to the specified units, if valid. If the unit conversion is not valid, the tag's original units are used instead. **Note:** This feature requires special licensing. If a license is not available, the data is returned in the original units without conversion.

Sample Query	**Scenario 1**
	This query produces a list of tags that end with "level". In this case, the user doesn't know the fully qualified name (FQN) of a specific tag and wants a short list of possible matches.
	`https://online.wonderware.[DOMAINSUFFIX]/apis/historian/v2/AnalogSummary?TagFilter=endswith(FQN,'level').`
	See more TagFilter examples.
	Scenario 2
	This query produces a list of values with analog summary for the tank_level tag.
	Notice that this example uses a fully qualified name ("Baytown.tank_level"), which is a combination of a data source name ("Baytown") and a tagname ("tank_level").
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'`
	Because this query specifies no start or end time and no resolution, these defaults are used for the returned results:
	• EndTime defaults to DateTime.UtcNow
	• StartTime defaults to one hour before EndTime

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• Resolution defaults toTimespan • Count defaults to1 (number of returned rows)

	Scenario 3
	This query produces a list of tag values with analog summary data. The "$filter" clause narrows the results further by specifying these parameters:
	• Fully qualified name of the tag (FQN+eq+'Baytown.tank_level')
	• Start and end times
	• Result set returned at 1 hour intervals (Resolution=3600000). There are 3.6 million milliseconds in an hour.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'+and+StartDateTime+ge+2017-05-14T00:00:00.000Z+and+EndDateTime+le+2017-05-16T00:00:00.000Z&Resolution=3600000`
	Scenario 4
	This query specifies only StartTime and Resolution (600000 ms, or 10 minutes), but no EndTime.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level' and+StartDateTime+ge+2017-06-29T00:00:00Z&Resolution=600000`
	In this case, Insight assumes:
	• EndTime defaults to DateTime.UtcNow
	The number of rows returned (Count) depends on the StartTime and EndTime.
	Scenario 5
	This query specifies only EndTime, but no StartTime or Resolution.
	`https://online.wonderware.[DOMAINSUFFIX]/s/ik97r5/apis/historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'and+EndDateTime+le+2017-06-29T00:00:00Z'`
	In this case, Insight assumes:
	• StartTime defaults to one hour before EndTime

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• Resolution defaults to Timespan • Count defaults to 1 (number of returned rows)
	Scenario 6 This query specifies uses SliceBy. `https://online.wonderware.com/apis/Historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.R21.Level'+and+StartDateTime+ge+2019-03-05T00:00:00.000Z+and+EndDateTime+le+2019-03-05T12:00:00.000Z&SliceBy=Baytown.R21.Batch` In this case, Insight calculates an analog summary for the Baytown..R21.Level tag and batches the results per value for the Baytown.R21.Batch tag.

StateSummary

Description	Retrieves state summary values for the specified tags.
URL	`/StateSummary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **RetrievalMode=**[string] Possible values are: Cyclic, Full. Default is Cyclic. • **Resolution=**[Int]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	In milliseconds. Any positive integer. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **Text=**[string] Non-numeric state. • **Average=**[Double] Average time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. • **AverageContained=**[Double] Average time in this state among all occurrences of this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Minimum=**[Double] Minimum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **MinimumContained=**[Double] The minimum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Maximum=**[Double] Maximum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **MaximumContained=**[Double] The maximum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Total=**[Double] Total time in this state during this retrieval cycle, including state occurrences that fall only partially within the period. • **TotalContained=**[Double] Total time in this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Percent=**[Double] Percent of the time during this retrieval cycle that the tag was in this state, including state occurrences that fall only partially within the period. • **PercentContained=**[Double] The precentage of the entire retrieval cycle time that the tag was in this state, excluding state occurrences that fall only partially within the period. This is a ratio between StateTimeTotalContained and StateTimeTotal expressed as a percentage in the range 0 to 100. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible. • **Count=**[Int64]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The number of times the state occurred within the retrieval cycle, including states that only partially occur in the cycle. • **CountContained=**[Int64] The number of times the state occurred fully contained within the retrieval cycle. States that only partially occur in the cycle are not counted. • **LastDateTime=**[DateTimeOffset] The time stamp for the last received value.

Sample Query	**Scenario 1**
	This query specifies no start or end time and no resolution.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03'`
	In this case, these defaults are used for the returned results:
	• EndTime defaults to DateTime.UtcNow
	• StartTime defaults to one hour before EndTime
	• Resolution defaults toTimespan
	• Count defaults to1 (number of returned rows)
	Scenario 2
	This query specifies only StartTime and Resolution (600000ms, or 10 minutes), but no EndTime.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03' and StartDateTime ge 2017-06-29T00:00:00Z&Resolution=600000`
	In this case, Insight assumes:
	• EndTime defaults to DateTime.UtcNow
	The number of rows returned (Count) depends on the StartTime and EndTime.
	Scenario 3
	This query specifies only EndTime, but no StartTime or Resolution.
	`http://localhost:32569/Historian/v2/StateSummary ?$filter=FQN eq 'Baytown.pump_03' and EndDateTime le 2017-06-29T00:00:00Z`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	In this case, Insight assumes: • StartTime defaults to one hour before EndTime • Resolution defaults toTimespan • Count defaults to1 (number of returned rows)

Events

Description	Retrieves information about events and alarms. **Note:** All property names are case-sensitive. Event storage preserves the case that you provide for any property value. For example, a property value of "TRUE" is different than "True" and "true."
URL	`/Events`
Method	`GET`
Optional Parameters	• **ID**=[GUID] Globally unique identifier for the event. • **EventTime**=[DateTime] UTC time stamp indicating when the event occurred. • **Type**=[String] Main categorization of the event. Examples of valid values: - Alarm.Acknowledge - Application.Write - Alarm.Clear - User.Write - Alarm.Set - User.Write.Secured - Alarm.Write - User.Write.Verified • **Priority**=[Int32] Value indicating the importance of the event. Values range from 1 to 999, with lower numbers indicating higher importance. • **Namespace** =[String] The namespace for this event tag. • **Severity**=[Int32] Categorization of the urgency of the event: 1 - Critical 2 - Major 3 - Minor 4 - Informational

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **EventTimeUTCOffsetMins**=[Int32] For local time, the offset in minutes from UTC time. • **ReceivedTime**=[DateTime] UTC time stamp indicating when the event was received by the Historian server. • **IsAlarm**=[Bool] "true" or "false" indicating whether this event is an alarm. • **Comment**=[String] A comment providing more information about the event. **Note:** If an alarm comment from Application Server contains a backslash (for example, \n), then an extra backslash appears in the event result (for example, \\n) in the browser. • **InTouchType**=[String] InTouch Type value. Examples include: - ALM - RTN - ACK - SYS • **ValueString**=[String] The current value string. • **PreviousValueString**=[String] The previous value string.
Optional Alarm Parameters	• **Alarm_ID**=[String] ID of the original Alarm event. For "Alarm.Set" events, this will be the same as the event ID. • **Alarm_Class**=[String] InTouch alarm classification. Examples include "DSC" (discrete), "VALUE" (LoLo, Hi, etc.), "DEV" (deviation), and "ROC" (rate of change). • **Alarm_Type**=[String] InTouch alarm type. • **Alarm_InAlarm**=[Boolean] "true" or "false" indicating whether the Alarm is still in the active state. • **Alarm_Acknowledged**=[Boolean] "true" or "false" indicating whether the user has acknowledged this alarm.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Alarm_Condition**=[String] The condition being alarmed. Examples include "Limit.Hi", "ROC.Lo", and "System" among others. • **Alarm_ValueString**=[String] Value logged for the variable related to the event. • **Alarm_LimitString**=[String] Limit being alarmed. • **Alarm_UnAckDuration**=[Int32] The duration, in milliseconds, for which the alarm went un-acknowledged. • **Alarm_Duration**=[Int32] The duration, in milliseconds, for which the alarm was active. • **Alarm_IsSilenced**=[Boolean] "true" or "false" indicating whether the alarm was silenced. • **Alarm_IsShelved**=[Boolean] "true" or "false" indicating whether the alarm was shelved. • **Alarm_ShelveStartTimeUTC**=[DateTime] Scheduled start of the shelve time if the alarm has been shelved. • **Alarm_ShelveEndTimeUTC**=[DateTime] Scheduled end of the shelve time if the alarm has been shelved. • **Alarm_ShelveReason**=[String] The reason the alarm was shelved. • **Alarm_ShelveUserLogin**=[String] The login ID for the person who shelved the alarm. • **Alarm_ShelveNode**=[String] The node of the shelved alarm.
Optional Provider Parameters	• **Provider_NodeName**=[String] Name of the node that generated the event. • **Provider_System**=[String] Software system that generated the event. Examples include "Application Server", "InTouch", and "InBatch". • **Provider_ApplicationName**=[String] Application that generated the event. For Application Server, this will be the galaxy name.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	For InTouch, it will be the InTouch application name. • **Provider_SystemVersion**=[String] Software version (for example, 4566.1210.5811.1) for the component identified by Provider_ApplicationName. • **Provider_InstanceNames**=[String] Provider-specific string that uniquely identifies the instance an application on a given node.
Optional Source Parameters	• **Source_Name**=[String] The name of the data source for this event tag. • **Source_ProcessVariable**=[String] Process variable to which the event is related. • **Source_ProcessVariable_Units**=[String] Engineering units used for the process variable. • **Source_ConditionVariable**=[String] Condition variable related to the event. For example, if "TIC101" has a field attribute "PV" and this is a "Hi" alarm, this value will be "TIC101.PV.Hi". • **Source_Object**=[String] Non-hierarchical name for the object to which the event is related, for example, "TIC101". • **Source_HierarchicalObject**=[String] Hierarchical name for the source object. For example, "Reactor_001.TIC". • **Source_Area**=[String] Non-hierarchical Area name. For example, "Bottling_Zone". • **Source_HierarchicalArea**=[String] Hierarchical Area name. For example, "Plant_001.Building_002.Mixing". • **Source_Engine**=[String] Non-hierarchical engine name. For example, "AppEngine_001". • **Source_Platform**=[String] Non-hierarchical platform name. For example "WinPlatform_001".
Optional User Parameters	• **User_Account**=[String] This is the login name for the operator for the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	given application. • **User_Name**=[String] Name of the user. • **User_NodeName**=[String] Computer name from which a user action was executed. • **User_Email**=[String] The user's email address. • **User_Phone**=[String] The user's phone number. • **User_InstanceName**=[String] An instance name for the user. • **User_Agent**=[String] Application name that the user was running when the event was generated.
Optional Verifier Parameters	• **Verifier_Account**=[String] This is the login name for theverifier. • **Verifier_Name**=[String] Name of the verifier. • **Verifier_NodeName**=[String] Computer name from which a verifier action was executed.

Sample Queries	**Sample 1**
	This query lists all events that occur from 2017-07-13 onward.
	`https://localhost:32573/Historian/v2/Events ?$filter=EventTime+gt+'2017-07-13T00:00:00'`
	Sample 2
	This query lists all alarm clearing events.
	`http://localhost:32569/Historian/v2/Events ?$filter=type+eq+'Alarm.Clear'`
	Sample 3
	This query lists events from 2017-05-14 to 2017-05-15 at 9 am. It skips the first 1000 events, then lists the top 2000 events after that.
	`http://localhost:32569/Historian/v2/ Events/?$filter=EventTime gt 2017-05-14T09:00:00.000Z and EventTime lt 2017-05-15T09:00:00.000Z&$skip1000&$top=2000`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Output	{ @odata.context: "http://localhost:32569/Historian/v2/Events/$metadata#Events", value: [{ id: "ee9b042c-181e-4872-b4e8-cb9575c1f4f9", eventtime: "2017-07-13T00:00:01.047Z", type: "Alarm.Clear", receivedtime: "2017-07-13T00:00:01.2553666Z", source_name: "AaEvents", namespace: "AaEvents", alarm_acknowledged: false, alarm_class: "VALUE", alarm_condition: "Limit.LoLo", alarm_durationms: 5006, alarm_id: "f1e21e43-352e-d3db-7552-6e77f193d02d", alarm_inalarm: false, alarm_isshelved: false, alarm_issilenced: false, alarm_limitstring: "10.0", alarm_originationtime: "2017-07-12T23:59:56.041Z", alarm_state: "UNACK_RTN", alarm_tagtype: "S", alarm_type: "LoLo", comment: "Severity 1", eventtimeutcoffsetmins: -420, intouchtype: "LoLo", isalarm: true, priority: 1, provider_applicationname: "AlarmsandEvents", provider_nodename: "INSIGHTCHARTAPP", provider_system: "Application Server", provider_systemversion: "4966.1210.14578.2", severity: 1, source_area: "Plant_Area", source_conditionvariable: "Drum_Conveyor.HorizontalMovement.LoLo", source_hierarchicalarea: "Enterprise.Site.Plant.Plant_Area", source_hierarchicalobject: "Drum_Conveyor", source_object: "Drum_Conveyor", source_processvariable: "Drum_Conveyor.HorizontalMovement", valuestring: "30.0" },

Tags

Description	Using the GET verb, retrieves tag metadata. For

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	version 2, this also includes tag extended properties. Using the POSTverb, manages multiple tags at once. Using the DELETE verb, deletes a specified tag.
URL	`/Tags`

Methods	`GET`
	`POST`
	`DELETE`

Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **Source=**[string] The data source. • **Description=**[string] The description of the tag. • **EngUnit=**[string] The engineering units used for the tag's recorded values. **EngUnitMax=**[Double] The maximum value of the tag, measured in engineering units. • **EngUnitMin=**[Double] The minimum value of the tag, measured in engineering units. The minimum value of the tag, measured in engineering units. • **InterpolationType=**[string] The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. • **IntegralDivisor=**[Double] The factor to be applied when integrating a rate

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only. • **RolloverValue=**[Double] The rollover value for the tag. • **MessageOff=**[string] The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. • **MessageOn=**[string] The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. • **TagName=**[string] The unique name of the tag within the AVEVA Historian system. • **TagType=**[string] The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the AVEVA Historian TagRef table. • **Raw** Identifies related the raw acquired value. • **Minutely** Identifies related minutely summary. • **Hourly** Identifies related hourly summary. • **Daily** Identifies related daily summary.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Get tags

Sample Query	**Scenario 1**
	This query lists tag metadata for all tags in all data sources for your Insight solution.
	`http://localhost:32569/Historian/v2/Tags`
	Scenario 2
	This query lists metadata for tags in a particular data source. The data source in this example is named Baytown.
	`http://localhost:32569/Historian/v2/Tags ?$filter=Source+eq+'Baytown'`

Sample Output	{
	@odata.context: "http://localhost:32569/ Historian/v2/$metadata#Tags",
	value: [
	{
	FQN: "14th aug app.Auto",
	Source: "14th aug app",
	Description: "Automatic mode",
	EngUnit: "",
	EngUnitMax: 0,
	EngUnitMin: 0,
	InterpolationType: "None",
	MessageOff: "Manu",
	MessageOn: "Auto",
	TagName: "Auto",
	TagType: "Discrete",
	Alias: "nareshtag-discrete",
	Location: "/Application Tag 2",
	Raw@odata.navigationLink: "http://localhost:32569/Historian/v2/Tags ('14th%252baug%252bapp.Auto')/Raw",
	Minutely@odata.navigationLink: "http://localhost:32569/Historian/v2/Tags ('14th%252baug%252bapp.Auto')/Minutely",
	Hourly@odata.navigationLink: "http://localhost:32569/Historian/v2 /Tags('14th%252baug%252bapp.Auto')/Hourly",
	Daily@odata.navigationLink: "http://localhost:32569/Historian/v2/Tags ('14th%252baug%252bapp.Auto')/Daily"
	}
	],
	@odata.nextLink: "http://localhost:32569/ Historian/v2/Tags ?$skiptoken=koen66nr4nGCReC02FqdxO4Y62C7zBVs v3uf7mO8XMQqxZveqw8 +v2/bVCsFdevT"

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Delete a tag

Sample Query	This query deletes a tag named Weather.Brisbane.Cloudiness.
	`DELETE /Historian/v2/Tags('Weather.Brisbane.Cloudiness') HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Content-Type: application/json`
	`{`
	`"delete":`
	`{`
	`"FQN":["datasourcename.tagname"]`
	`}`
	`}`

Manage multiple tags

Sample Query	**Scenario 1: Bulk query**
	This query retrieves multiple tags.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchqaruntime.cloudapp.net:8080`
	`Authorization: Basic …….`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"query":`
	`{`
	`"FQN":['Weather.Brisbane.Cloudiness','Weather.Auckland.Cloudiness'],`
	`"select":"FQN,TagName"`
	`}`
	`}`
	Scenario 2: Bulk edit
	This makes updates to multiple tags. You can edit up to 100 tags at a time, with maximum of 50 properties per tag.
	Note: Historian Data REST API extended properties map to the AVEVA Historian SDK tag extended properties API.
	This makes updates to a tag's standard and extended properties.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Authorization: Basic …`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"properties":`
	`[`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`{`
	`"FQN": "18Jul_kc.test1",`
	`"PropertyName": "Alias",`
	`"Text": "SysTimeSec",`
	`"Type": "String",`
	`"Searchable":false`
	`}`
	Scenario 3: Bulk delete
	This deletes multiple tags.
	`POST /Historian/v2/Tags HTTP/1.1`
	`Host: nchdevruntime.cloudapp.net:8080`
	`Content-Type: application/json`
	`Cache-Control: no-cache`
	`{`
	`"delete":`
	`{`
	`"FQN":["datasourcename.tagname1",`
	`"datasourcename.tagname2"]`
	`}`
	`}`

Sample Output	The output from a bulk edit includes results per tag. If any of a tag's properties is invalid, the tag fails and results in an error message.
	`{`
	`"value":`
	`[`
	`{`
	`"FQN": "System.Tag1",`
	`"$StatusCode": 200`
	`},`
	`{`
	`"FQN": "System.Tag2",`
	`"$StatusCode":500,`
	`"$ErrorMessage":"Alias property contains illegal characters"`
	`}`
	`]`
	`}`

TagProperties

Description	Lists all properties used in the system. Tag properties include both base and extended properties of a tag. Examples of TagProperties are DataSourceName and Unit.
URL	`/TagProperties`
Method	`GET`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Required Parameters	• **Name=**[string] The unique name of the tag property.
Optional Parameters	• **Values** Valid values for this property.
Sample Query	This query lists all properties and extended properties for your Insight solution: `http://localhost:32569/Historian/v2/TagProperties`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagProperties",`
	`value: [`
	`{`
	`Name: "DataSourceName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "Namespace",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "TagName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`{`
	`Name: "FullyQualifiedName",`
	`Type: "String",`
	`ReadOnly: true`
	`},`
	`...`

TagPropertyValues

Description	Retrieves values for one or more tag properties. Tag properties include base and extended properties of Tag. Examples of TagProperties are DataSourceName and Unit.
URL	`/TagpropertyValues`
Method	`GET`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Required Parameters	• **Name=**[string] Unique name of the tag property value. • **Value=**[string]
Optional Parameters	• **Tags** Identifies related Tags.
Sample Query	This query lists the property name ("Source") and value (that is, the data source name) for all data sources: `http://localhost:32569/Historian/v2/TagPropertyValues`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagPropertyValues",`
	`value: [`
	`{`
	`Name: "Source",`
	`Value: "Meter",`
	`Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Meter')/Tags"`
	`},`
	`{`
	`Name: "Source",`
	`Value: "Weather",`
	`Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Weather')/Tags"`
	`}`
	`]`
	`}`

TagGroups

Description	Retrieves information about data sources. Each data source contains a group of tags.
URL	`/TagGroups`
Method	`GET`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Required Parameters	• **GroupID=**[string] Globally unique identifier for the tag group.
Optional Parameters	• **GroupName=**[string] The name of the data source. • **TypeID=**[string] The type of data source. • **ParentID=**[string] • **Scope=**[string] Used for tag-level security, this defines a location within the data source. • **Tags** Identifies related Tags. • **Groups** Identifies related TagGroups.
Sample Query	This query lists the data sources (labeled "TagGroups" in this API) for your Insight solution identified by solution ID. In this example, the solution ID used is "ik97r5". `http://localhost:32569/Historian/v2/TagGroups`

Sample Output	`{`
	`@odata.context: "http://localhost:32569/Historian/v2/$metadata#TagPropertyValues",`
	`value: [`
	`{`
	`Name: "Source",`
	`Value: "Meter",`
	`Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagPropertyValues(Name='Source',Value='Meter')/Tags"`
	`},`
	`{`
	`GroupID: "1cd29ff2-f7f3-428b-a593-55c917136a39",`
	`GroupName: "Weather",`
	`Type: "1000000",`
	`ParentID: "0",`
	`Scope: "Public",`
	`Tags@odata.navigationLink: "http://localhost:32569/Historian/v2/TagGroups('1cd29ff2-f7f3-428b-a593-55c917136a39')/Tags",`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	`Groups@odata.navigationLink: "http://localhost:32569/Historian/v2/TagGroups ('1cd29ff2-f7f3-428b-a593-55c917136a39')/Groups" } ] }`

TagSuggest

Description	Retrieves content or tags based on search criteria from Insight. It returns the search results based on tag name, content name, keywords, description, and so on. It also returns the summary of the search result -- for example, total matching search results for the given search text. You can use the these query options: • q -- (query) Limits the search to only specified fields instead of searching all fields. • Type -- Limits the search to only a certain type of data (Tag or SavedContent).
URL	`/TagSuggest`
Method	`GET`
Required Parameters	• **Value=**[string] The suggested value.
Optional Parameters	• **FieldName=**[string] The related field name. • **Count=**[Double] The hit count for this query string. • **SearchRanking=**[Double] The search ranking for the tag. • **DisplayText=**[string] The associated display text. • **Search** • **Tags** Identifies related Tags. • **ExpandSuggest**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Sample Query	**Scenario 1**
	This query returns values from a search of all fields (default).
	`http://localhost:32569/Historian/v2/TagSuggest`
	Scenario 2
	This query uses the "q" query option to limit the search to only the "location" field.
	`http://localhost:32569/Historian/v2/TagSuggest ?q=&searchfields=location`
	Scenario 3
	This query uses the "Type" option to limit the search to only a certain type of data (Tag or SavedContent).
	`http://localhost:32569/Historian/v2/TagSuggest?q=d&Type=Tag`

TagSearch

Description	Provides tagname results based on provided search parameters. The data returned by this entity is dependent on the historian's implementation of search functionality. Some historians will return empty result sets. It finds all the matching results for a given query string. It is basically used to filter the results based on the suggestion results. It searches for the matching record only on the fields provided in the key name as part of the query parameter. The results contain basic tag metadata information; such as FQN, tagname, source, and search ranking. You can use these query options: • q - Specifies the query string (for example, the value typed by the user into the search box). • kn - Specifies the key name (field name) to which the search will be applied. • kv - Specifies the key value which will be used for the search.
URL	`/TagSearch`
Method	`GET`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname.
Optional Parameters	• **Source=**[string] The data source. • **TagName=**[string] The name of the tag. • **DisplayText=**[string] The displayed text for the tag. • **SearchRanking=**[Double] The search ranking for this tag. • **Tag** Returns a URL to retrieve a list of Tag entity values for the tags that match.
Sample Query	This example uses the "q", "kn", and "kv" query options. `http://localhost:32569/Historian/v2/TagSearch ?q=r&kn=source&kv=atron`

TagExtendedProperties

Note: This resource has been deprecated. See sample queries under Tags for current information on retrieving properties associated with a tag.

Description	Retrieves both standard and extended properties for a tag.
URL	`/TagExtendedProperties`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified name uses the format: datasource.tagname. • **PropertyName=**[string] The name of the property.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Optional Parameters	• **Value=**[Double]
	The value of the property.
	• **Text=**[string]
	The associated text for the property.
	• **Type=**[string]
	Can be one of these:
	  • [Edm.]String
	  • [Edm.]Int16, Int32, Int64
	  • [Edm.]Double
	  • [Edm.]DateTimeOffset
	  • [Edm.]Guid
	  • [Edm.]Boolean
	  • [Edm.]Geography types
	  • [Edm.]Geometry types

Summary (v1 only)

Description	Retrieves summary data for a user-defined interval. The additional query option "Resolution" is used to specify the interval to be used for this entity. The earliest and latest requested values of StartDateTime and/or EndDateTime are used for determining the span of time covered by the summary intervals returned.
URL	`/Summary`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset]

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog** An embedded collection that maps to AnalogSummary. • **Split** An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained** An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Daily (v1 only)

Description	Retrieves daily (24-hour resolution) summary values for the tags specified. The default for Daily is to report summary values for the last week.
URL	`/Daily`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **TimeZone=**[string] The timezone used for the timeframe. See "Accepted values for TimeZone" below. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	100. • **Analog=**[analog statistics] An embedded collection that maps to AnalogSummary. • **Split=**[collection of state statistics] An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained=**[collection of state statistics] An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query. • **Minute** • **Hourly**

Accepted values for TimeZone

UTC offset in minutes	TimeZone value
0	UTC
0	Morocco Standard Time
0	GMT Standard Time
0	Greenwich Standard Time
60	W. Europe Standard Time
60	Central Europe Standard Time
60	Romance Standard Time
60	Central European Standard Time
60	W. Central Africa Standard Time
60	Namibia Standard Time
120	Jordan Standard Time
120	GTB Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

120	Middle East Standard Time
120	Egypt Standard Time
120	E. Europe Standard Time
120	Syria Standard Time
120	West Bank Standard Time
120	South Africa Standard Time
120	FLE Standard Time
120	Israel Standard Time
120	Kaliningrad Standard Time
120	Libya Standard Time
180	Arabic Standard Time
180	Turkey Standard Time
180	Arab Standard Time
180	Belarus Standard Time
180	Russian Standard Time
180	E. Africa Standard Time
210	Iran Standard Time
240	Arabian Standard Time
240	Astrakhan Standard Time
240	Azerbaijan Standard Time
240	Russia Time Zone 3
240	Mauritius Standard Time
240	Georgian Standard Time
240	Caucasus Standard Time
270	Afghanistan Standard Time
300	West Asia Standard Time
300	Ekaterinburg Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

300	Pakistan Standard Time
330	India Standard Time
330	Sri Lanka Standard Time
345	Nepal Standard Time
360	Central Asia Standard Time
360	Bangladesh Standard Time
360	Omsk Standard Time
390	Myanmar Standard Time
420	SE Asia Standard Time
420	Altai Standard Time
420	W. Mongolia Standard Time
420	North Asia Standard Time
420	N. Central Asia Standard Time
420	Tomsk Standard Time
480	China Standard Time
480	North Asia East Standard Time
480	Singapore Standard Time
480	W. Australia Standard Time
480	Taipei Standard Time
480	Ulaanbaatar Standard Time
510	North Korea Standard Time
525	Aus Central W. Standard Time
540	Transbaikal Standard Time
540	Tokyo Standard Time
540	Korea Standard Time
540	Yakutsk Standard Time
570	Cen. Australia Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

570	AUS Central Standard Time
600	E. Australia Standard Time
600	AUS Eastern Standard Time
600	West Pacific Standard Time
600	Tasmania Standard Time
600	Vladivostok Standard Time
630	Lord Howe Standard Time
660	Bougainville Standard Time
660	Russia Time Zone 10
660	Magadan Standard Time
660	Norfolk Standard Time
660	Sakhalin Standard Time
660	Central Pacific Standard Time
720	Russia Time Zone 11
720	New Zealand Standard Time
720	UTC+12
720	Fiji Standard Time
720	Kamchatka Standard Time
765	Chatham Islands Standard Time
780	Tonga Standard Time
780	Samoa Standard Time
840	Line Islands Standard Time
-60	Azores Standard Time
-60	Cape Verde Standard Time
-120	UTC-02
-120	Mid-Atlantic Standard Time
-180	Tocantins Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

-180	E. South America Standard Time
-180	SA Eastern Standard Time
-180	Argentina Standard Time
-180	Greenland Standard Time
-180	Montevideo Standard Time
-180	Saint Pierre Standard Time
-180	Bahia Standard Time
-210	Newfoundland Standard Time
-240	Paraguay Standard Time
-240	Atlantic Standard Time
-240	Venezuela Standard Time
-240	Central Brazilian Standard Time
-240	SA Western Standard Time
-240	Pacific SA Standard Time
-240	Turks And Caicos Standard Time
-300	SA Pacific Standard Time
-300	Eastern Standard Time (Mexico)
-300	Eastern Standard Time
-300	Haiti Standard Time
-300	Cuba Standard Time
-300	US Eastern Standard Time
-360	Central America Standard Time
-360	Central Standard Time
-360	Easter Island Standard Time
-360	Central Standard Time (Mexico)
-360	Canada Central Standard Time
-420	US Mountain Standard Time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

-420	Mountain Standard Time (Mexico)
-420	Mountain Standard Time
-480	Pacific Standard Time (Mexico)
-480	UTC-08
-480	Pacific Standard Time
-540	Alaskan Standard Time
-540	UTC-09
-570	Marquesas Standard Time
-600	Aleutian Standard Time
-600	Hawaiian Standard Time
-660	UTC-11
-720	Dateline Standard Time

Hourly (v1 only)

Description	Retrieves hourly summary values for the tags specified. The default for Hourly is to report summary values for the last 24 hours.
URL	`/Hourly`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **TimeZone=**[string] The timezone used for the timeframe. See "Accepted values for TimeZone" below. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog=**[analog statistics] An embedded collection that maps to AnalogSummary. • **Split=**[collection of state statistics] An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained=**[collection of state statistics] An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query. • **Minute**

Minutely (v1 only)

Description	Retrieves summary values with a 1-minute resolution for the tags specified. Default duration for minutely

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	summaries is the last 1 hour. The default for Minutely is to report summary values for the last hour.
URL	`/Minutely`
Method	`GET`
Required Parameters	• **FQN=**[string] The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName. • **StartDateTime=**[DateTimeOffset] The starting date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z • **EndDateTime=**[DateTimeOffset] The ending date and time for the retrieval cycle. This is always specified in UTC using the RFC3339 / ISO8601 format with the Z time zone designator. For example: 2016-09-03T18:44:09.352247Z
Optional Parameters	• **TimeZone=**[string] The timezone used for the timeframe. See "Accepted values for TimeZone" below. • **OPCQuality=**[Int32] OPC quality. Normal OPC quality retrieval logic is applied if: All the point found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned. • **PercentGood=**[Double] The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100. • **Analog=**[analog statistics] An embedded collection that maps to AnalogSummary.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **Split=**[collection of state statistics] An embedded collection that maps to StateSummary for the ValueState retrieval mode. • **Contained=**[collection of state statistics] An embedded collection that maps to StateSummary for the ContainedState retrieval mode. • **Raw** A URL used to retrieve all stored values for a specified time period and tag. This is equivalent to a preconfigured ProcessValue query.

SystemParameters (on-premises only)

Description	Retrieves names and values for system parameters. (On-premises AVEVA Historian Client Web only)
URL	`/SystemParameters`
Method	`GET`
Required Parameters	• **Name=**[string] The unique name for the system parameter.
Optional Parameters	• **Value=**[string] The value of this system parameter. • **Tags** Identifies related tags.

Retrieval examples

You can send retrieval requests to the Historian Data REST API from any modern web browser and from OData client-side tools. Examples of such tools include Microsoft Excel (2013, 2016, or Office 365), and Business Intelligence (BI) systems, such as Tableau and Microsoft Power BI.

Select below to see query examples using specific tools:

- Browser query examples
- Postman query examples
- Excel query examples
- PowerBI query example

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Retrieve data via browser query

You can simply type an OData HTTP request in the address bar of any modern browser to query Historian Data API.

- **List root entities**

  This example lists the entities you can explore and submit additional queries against.

  ```
  http://localhost:32569/Historian/v2
  ```

  For more information, see the Data Dictionary for Historian Data API.

- **List data sources (tag groups)**

  This shows the data sources (labeled "TagGroups" in this API) for your Insight solution. If you have accounts for more than one solution, this shows the data sources for your original Insight solution.

  ```
  http://localhost:32569/Historian/v2/TagGroups
  ```

- **List tag metadata for all tags**

  This shows metadata for all tags in all data sources for your original Insight solution.

  ```
  http://localhost:32569/Historian/v1/historian/v2/Tags
  ```

- **List tag metadata for a data source**

  This shows metadata for tags in a particular data source. The data source in this example is named Baytown.

  ```
  http://localhost:32569/Historian/v2/Tags?$filter=Source+eq+'Baytown'
  ```

- **List tag values for a data source**

  This example expands on the last one. It uses a "$filter" clause that specifies a particular tag named tank_level within the Baytown data source. The result is a list of values for the tank_level tag.

  ```
  http://localhost:32569/Historian/v2/ProcessValues?$filter=FQN+eq+'Baytown.tank_level'
  ```

- **List tag values with analog summary data for a data source**

  This example result is a list of values with analog summary for the tank_level tag.

  ```
  http://localhost:32569/Historian/v2/AnalogSummary?$filter=FQN+eq+'Baytown.tank_level'
  ```

  Notice that this example uses a fully qualified name ("Baytown.tank_level"), which is a combination of a data source name ("Baytown") and a tagname ("tank_level").

Retrieve data using Postman

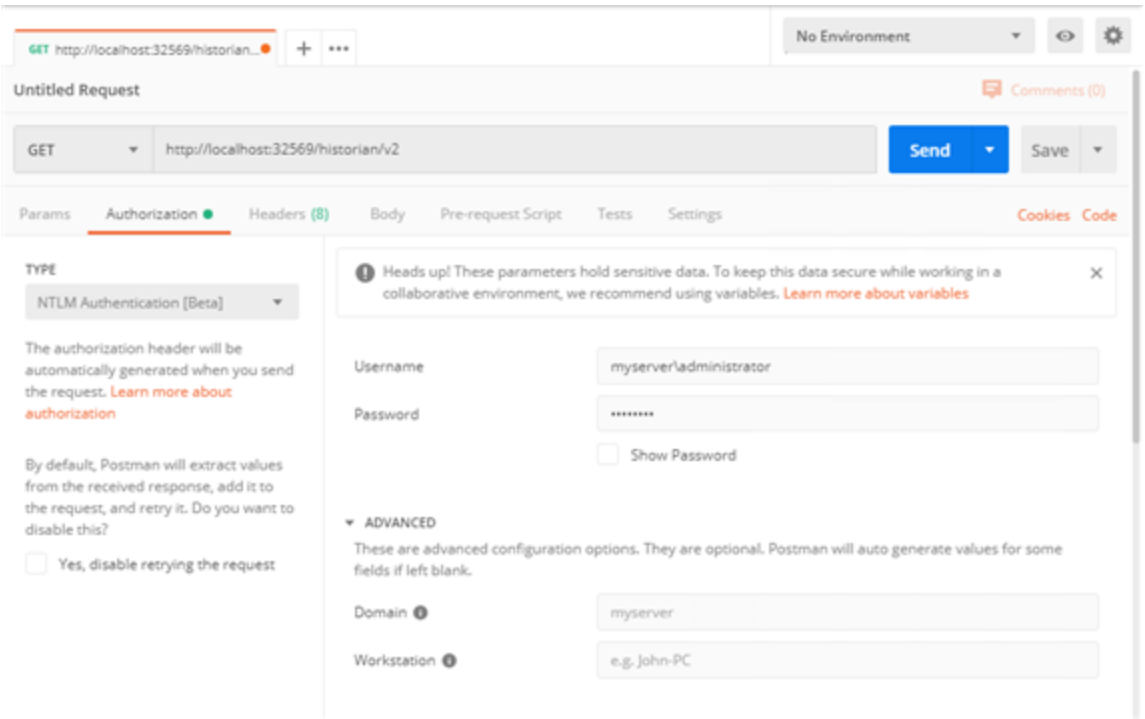
A great way to explore and learn about the AVEVA Insight APIs is by using a simple client, such as Postman. Postman is a free developer tool allows you to query any API. You can download Postman from https://postman.com/.

To query the AVEVA Insight APIs using Postman

1. Next to the GET button, type this address:

   ```
   http://localhost:32569/historian/v2
   ```

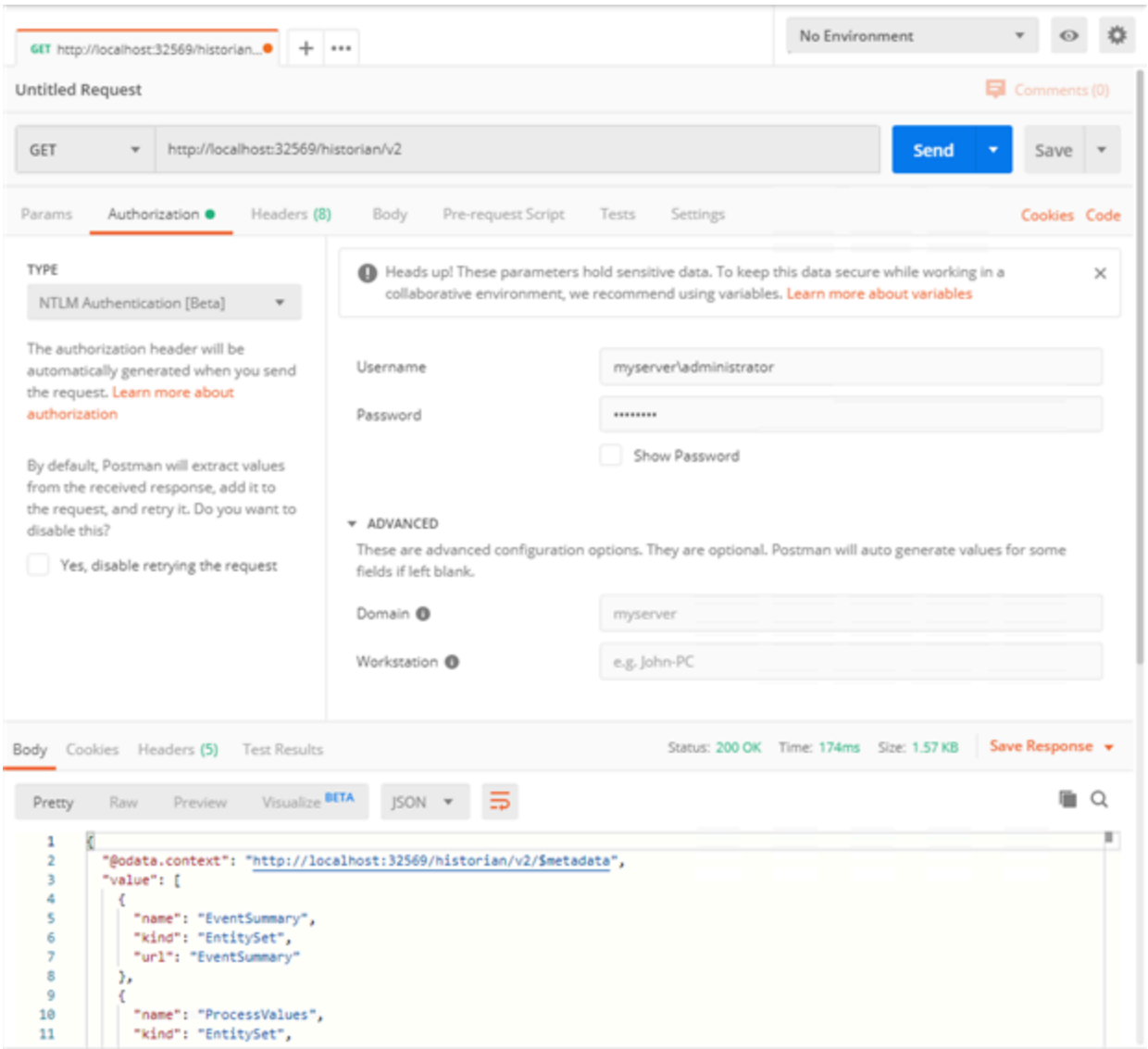
© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



2. From the **Type** dropdown list, select **NTLM Authentication [Beta]** to specify the authentication to use.
3. Enter the username and password for a user account with sufficient privileges to access the API.
4. In the upper-right of the screen, select **Send**.

   The result is a catalog of data entities from AVEVA Insight.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



Excel query examples

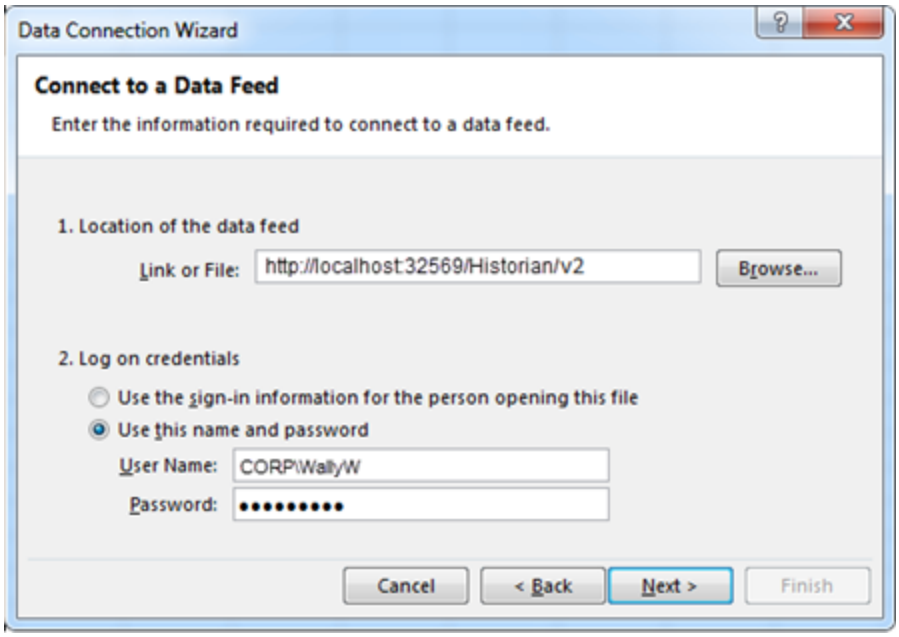
To manually import Insight data to Excel

1. Open Excel 2013 or later.
2. From **Data**,choose **Get External Data**, choose **From Other Sources**, and then choose **From OData Data Feed**.
3. In the **Link or File** box, specify a URL for your query, formatted as an atom feed (that is, end with "$format=atom'").

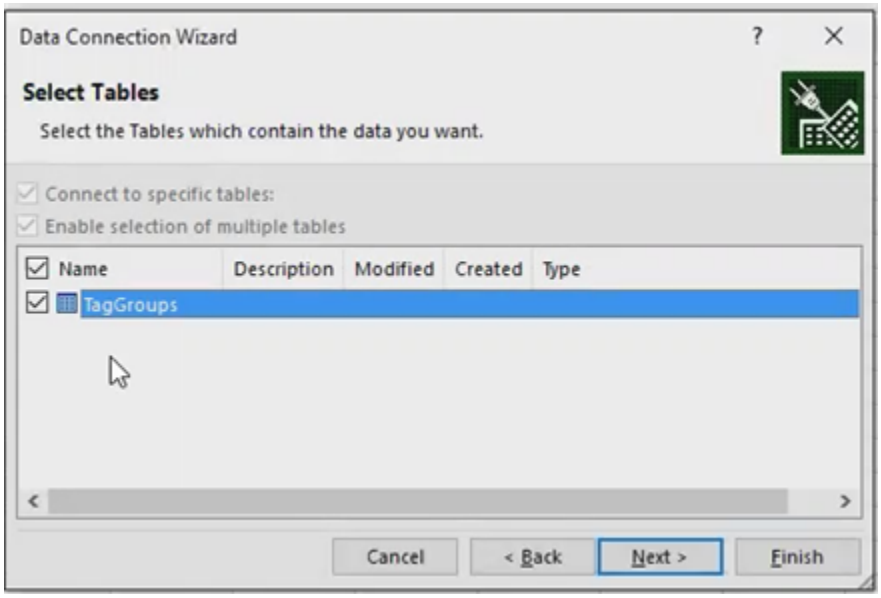
   Here is a query example:

```
http://localhost:32569/Historian/v2/TagGroups?$format=atom
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

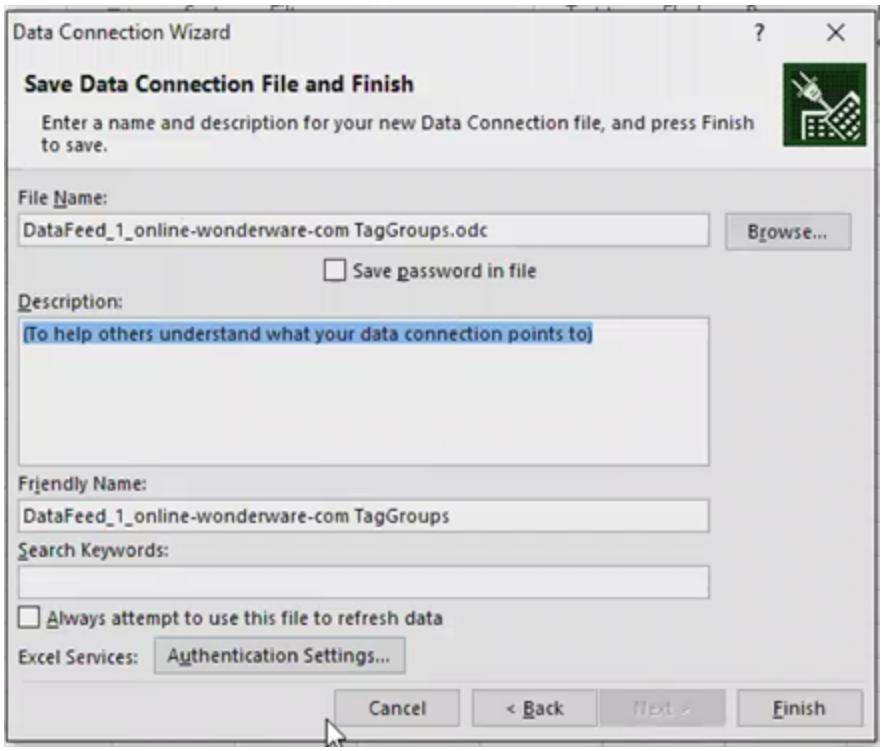


4. Click **Use this name and password**, and specify your Insight user name and password. Click **Next**.
5. In the **Select Tables** box, mark the element to include. Click **Next**.

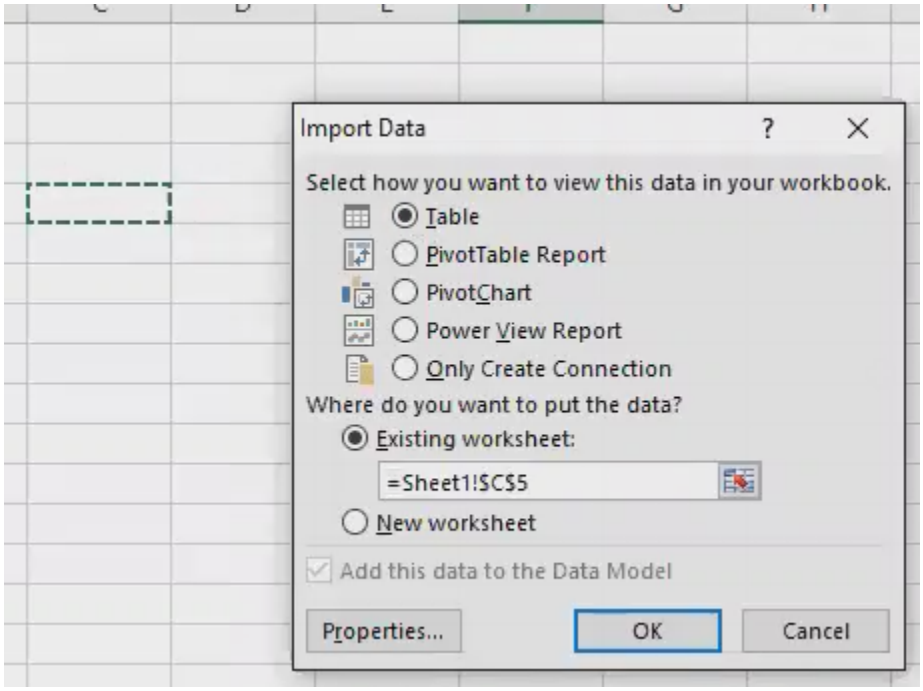


6. Review the information displayed on **Save Data Connection File and Finish** and then click **Finish**. Excel creates an ODC (OData Connection) file.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



7. Specify how and where to place your data in the spreadsheet. Click **OK**.



Excel will place the data where you specified:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

GroupID	GroupName	Type	ParentID	Scope
5d181b0f-ba93-4c81-83de-0aae9f946b5b	Wylie	1000000	0	Public
503f8aab-35c0-434a-a256-1a493ac1b539	TestNew	1000000	0	Public
42cc15b9-3bc9-417c-8957-97f22db2acfa	mwoTest	1000000	0	Public
2bb8b991-671b-4835-9207-5b27ff54a2f5	Frankfurt	1000000	0	Public
2b89caa2-e00c-497a-8a73-c897e1084da7	SAMC2016	1000000	0	Public
fb167546-fb19-4405-a17b-c4a5a31eed0f	Canterbury	1000000	0	Public
d8ec8390-c9ba-4192-850f-8790b09a3256	test	1000000	0	Public
8f707b8f-94a8-4a97-adfa-99f2c0518617	InTouch	1000000	0	Public
747fe221-5d98-4c3d-bae8-5ea268d40538	Baytown	1000000	0	Public

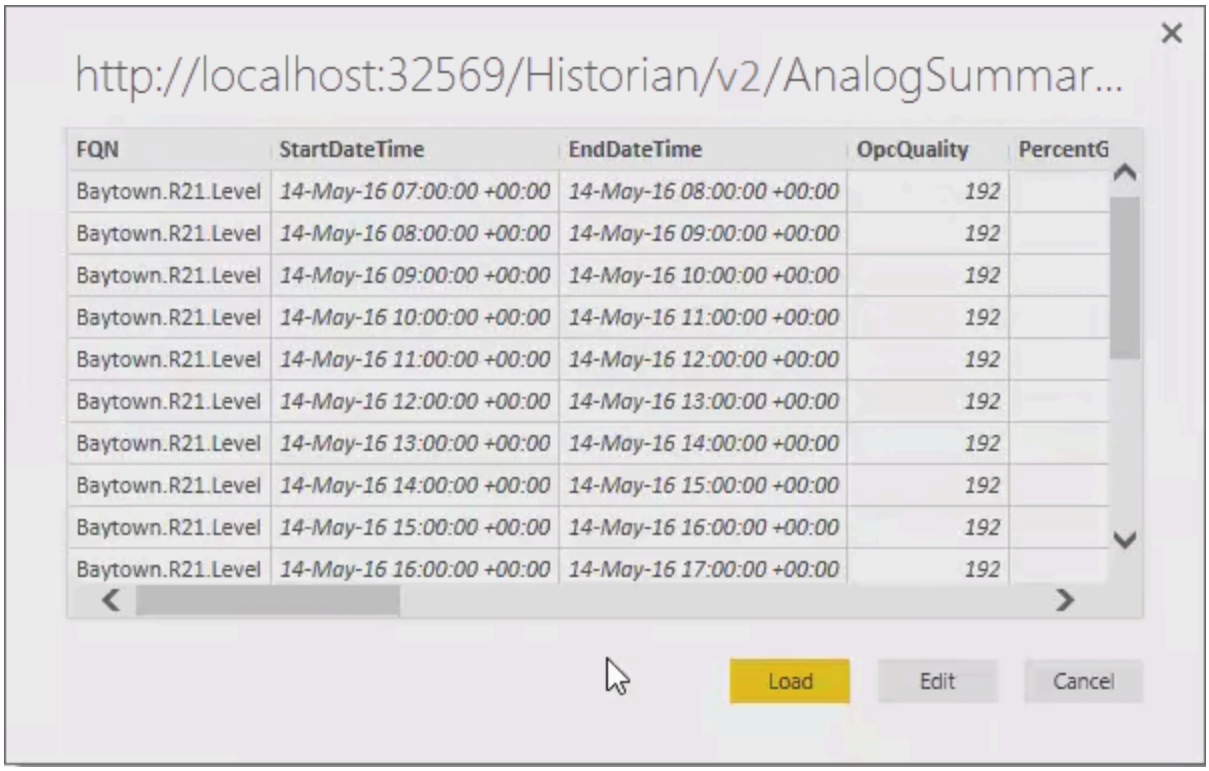
PowerBI query example

To import Insight data to PowerBI

1. Open PowerBI.
2. Click **Get Data**, and choose **OData Feed**.
3. Specify the query URL and click **OK**.

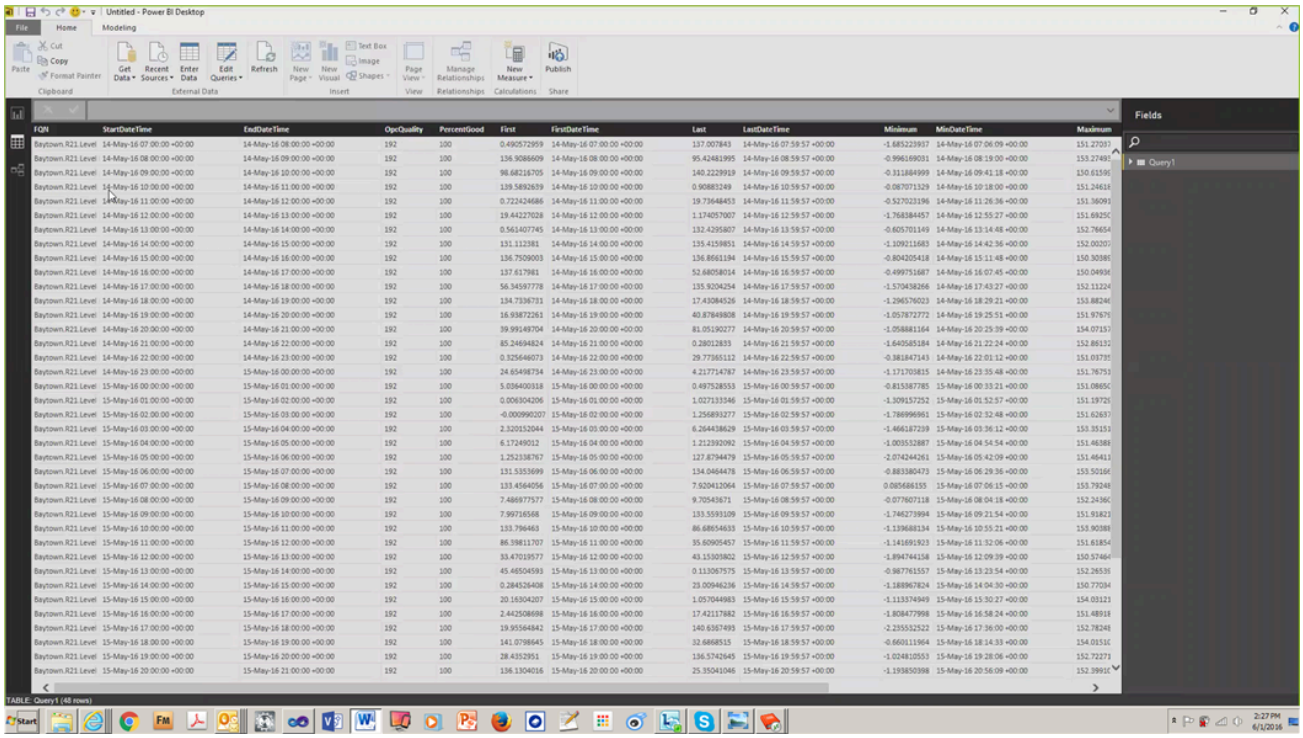
4. Specify **Basic Authentication**, and provide your Insight username and password.
5. Click **Connect**.
6. Power BI shows a preview of your data retrieved from Insight.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.



FQN	StartDateTime	EndDateTime	OpcQuality	PercentG
Baytown.R21.Level	14-May-16 07:00:00 +00:00	14-May-16 08:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 08:00:00 +00:00	14-May-16 09:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 09:00:00 +00:00	14-May-16 10:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 10:00:00 +00:00	14-May-16 11:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 11:00:00 +00:00	14-May-16 12:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 12:00:00 +00:00	14-May-16 13:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 13:00:00 +00:00	14-May-16 14:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 14:00:00 +00:00	14-May-16 15:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 15:00:00 +00:00	14-May-16 16:00:00 +00:00	192	
Baytown.R21.Level	14-May-16 16:00:00 +00:00	14-May-16 17:00:00 +00:00	192	

7. Click **Load**.
8. In the upper-left corner, click to view the table of results.



© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Querying History Blocks via SQL Server Reporting Services Extension

Historian includes an extension for SQL Server Reporting Services to enable SQL queries against the OData interface used for data stored in history blocks.

To configure the extension

1. Open the Configurator.
2. Under Historian, click the **Reporting** node.
3. Select the versions of Visual Studio installed with your SQL Server Reporting Services.
4. Click **Configure**.

To use the extension

1. Define a new Reporting Services data source.
2. Select "Historian OData Provider" as the source type.
3. Click **Historian** and then click **Extension**.
4. Enter a connection string in the form "Server=host:port"; for example:

```
Server=localhost:32569
```

Three Ways To Query the Source

You can query the source using one of these options:

- **Use a SQL-style query**. For example:

```
select * from Events where Source_Area='Site2' and EventTime >=
'2014-08-10T05:56:21.302Z'
```

- **Use an OData query**. This OData query example is equivalent to the query above:

```
http://Server1:32569/Historian/v1/Events?$filter=Source_Area+eq+%27Site2%27+and
EventTime+ge+datetimeoffset%272014-08-10T05:56:21.302Z%27
```

Note: For queries from a web browser, use "%20" to indicate a space. Use "%27" to indicate a single quote. If you are using the JSONView viewer in the Chrome browser, you can use a plus sign (+) to indicate a space to make the URI string more readable.

- **Use an SQL syntax used within the URL**. For example, this is equivalent to the other two queries above:

```
http://Server1:32569/Historian/v1/Events?sql=select+*+from Events+where
Source_Area=%27ite2%27+and+EventTime+>=+%272014-08-10T05:56:21.302Z%27
```

Retrieval errors

The OData error codes listed in the following table may be returned by an operation on any of the storage services.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Error code	HTTP status code	User message
ConditionNotMet	Not Modified (304)	The condition specified in the conditional header(s) was not met for a read operation.
MissingRequiredQueryParameter	Bad Request (400)	A required query parameter was not specified for this request.
UnsupportedQueryParameter	Bad Request (400)	One of the query parameters specified in the request URI is not supported.
InvalidQueryParameterValue	Bad Request (400)	An invalid value was specified for one of the query parameters in the request URI.
OutOfRangeQueryParameterValue	Bad Request (400)	A query parameter specified in the request URI is outside the permissible range.
RequestUrlFailedToParse	Bad Request (400)	The url in the request could not be parsed.
InvalidUri	Bad Request (400)	The requested URI does not represent any resource on the server.
InvalidHttpVerb	Bad Request (400)	The HTTP verb specified was not recognized by the server.
OutOfRangeInput	Bad Request (400)	One of the request inputs is out of range.
InvalidAuthenticationInfo	Bad Request (400)	The authentication information was not provided in the correct format. Verify the value of Authorizationheader.
InvalidInput	Bad Request (400)	One of the request inputs is not valid.
Unauthorized	Unauthorized (401)	Unauthorized: Access is denied due to invalid credentials.
ResourceNotFound	Not Found (404)	The specified resource does not exist.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Error code	HTTP status code	User message
InternalError	Internal Server Error (500)	The server encountered an internal error. Please retry the request.
OperationTimedOut	Internal Server Error (500)	The operation could not be completed within the permitted time.
ServerBusy	Service Unavailable (503)	The server is currently unable to receive requests. Please retry your request.

Historian Scenarios Guide

Welcome

In your plant, you work hard to manage large amounts of data. While you and other AVEVA Historian customers may have very different processes and products, you probably do share some key issues around managing your data and using it to better run your operations.

This book discusses ten common scenarios where AVEVA Historian can help. They are:

1. Using hourly and daily reports.
2. Tracking inventory and production with packing line counters
3. Tracking liquid and gas use with custody meters.
4. Analyzing cycle time for your process.
5. Calculating the total flow.
6. Calculating downtime statistics.
7. Pinpointing reasons for downtime.
8. Filling in the gaps in your data.
9. Understanding and using data.
10. Getting summary statistics when a certain condition applies

Scenario 1: Use hourly and daily reports

You can use the AnalogSummaryHistory view to regularly track your plant's productivity. For example, you can answer questions such as:

- What was the peak temperature each hour?
- How much product did we produce today?

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

What is Analog Summary?

For analog data (such as temperature of a boiler or volume within a tank), you can retrieve summary data using the AnalogSummaryHistory view. For even faster retrieval, you can configure the tags to be stored with specific summary data and retrieve that summary data later. You can use a variety of available summary statistics, including:

- Time-weighted average
- Standard deviation
- First, last, minimum, or maximum value for a timestamped period

About the AnalogSummaryHistory View

AVEVA Historian stores historical data in history blocks rather than in the database itself. The data is accessible via extension tables.

With AnalogSummaryHistory, you can query summary statistics for analog tags. The AnalogSummaryHistory view lets you return multiple statistics for a single tag within one query.

Example 1: Creating a Daily Report with Delta Retrieval Mode

The following query returns the minimum, maximum, and average values for the R31.ReactTemp tag for two days of data at 24-hour resolution. Because no retrieval mode is specified, the retrieval mode is delta (the default retrieval mode for the AnalogSummaryHistory table).

```
SELECT StartDateTime, TagName, Minimum, Maximum, Average
   FROM AnalogSummaryHistory
      WHERE TagName='R31.ReactTemp'
      AND StartDateTime >= '2015-08-21'
      AND EndDateTime <= '2015-08-23'
      AND wwResolution=24*60*60*1000
```

The results are:

StartDateTime	TagName	Minimum	Maximum	Average
2015-08-21 01:00:00.0000000	R31.ReactTemp	181	211	201
2015-08-22 01:00:00.0000000	R31.ReactTemp	177	219	200

Example 2: Creating a Daily Report with Daily Retrieval Mode

The following query returns the average values for the R31.ReactTemp tag for four days of data at 24-hour resolution. Unlike the previous example, this uses the daily retrieval mode, which always has a resolution of a full local time day (86400000 milliseconds, except on days when daylight saving time begins or ends). Daily mode accounts for daylight saving time changes and adjusts the resolution for time change days accordingly. In the example below, daylight saving time begins on 2021-11-06, so that day has a resolution of 90000000 milliseconds (25 hours).

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
SELECT StartDateTime, wwResolution, TagName, Average
   FROM AnalogSummaryHistory
      WHERE TagName='R31.ReactTemp'
      AND StartDateTime >= '2021-11-04'
      AND EndDateTime <= '2021-11-08'
      AND wwRetrievalMode='daily'
```

The results are:

StartDateTime	wwResolution	TagName	Average
2021-11-04 00:00:00.0000000	86400000	R31.ReactTemp	201
2021-11-05 00:00:00.0000000	86400000	R31.ReactTemp	200
2021-11-06 00:00:00.0000000	90000000	R31.ReactTemp	198
2021-11-07 00:00:00.0000000	86400000	R31.ReactTemp	203

Scenario 2: Track inventory and production with packing line counters

You can use the Counter retrieval feature to calculate production rates and totals. For example, you could answer questions like:

- How many screw tops have we used in the last week?
- How many products have we produced since Monday?

What is Counter Retrieval?

With Counter retrieval, you can easily calculate the amount of increase in a tag value (and therefore the number of items produced) over time. For example, you might have an integer counter that keeps track of how many cartons were produced. The counter has an indicator like this:

```
[ 9 ][ 9 ][ 9 ][ 9 ]
```

Some counters are reset manually -- for example, at the end of a shift before restocking. Others may never be reset, and instead will simply rollover, for example from 9999 to 0. Calculating totals accurately depends on:

- Whether the counter has rolled over
- Whether the counter gets reset manually

About Rollover Values

The rollover value is defined in the AVEVA Historian for each tag.

The next value after the highest value that can be physically shown by the counter is called the rollover value. For example, the rollover value for a four-digit integer counter is 10,000. When the counter reaches the 9,999th

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

value, the counter rolls back to 0. Therefore, a counter value of 9,900 at one time and a value of 100 at a later time means that you have produced 200 units during that period, even though the counter value has dropped by 9,800 (9,900 minus 100). Counter retrieval allows you to handle this situation and calculate the correct value.

Similarly, if you had a PLC register that is a 2-byte unsigned integer, the rollover value would be 65,536.

For each cycle, the counter retrieval mode shows the increase in that counter during the cycle, including rollovers.

Counter retrieval is a true cyclic mode. It returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

The counter algorithm is applied only to analog and discrete tags. For integer analog tags, the result will be an integer returned as a float data type. For a real analog tag, the rollover value and the result may be real values and can include fractional values. For discrete tags, the rollover value is assumed to be 2. If a query contains tags of other types, then no rows are returned for those tags.

Resetting a Counter

If you have a counter that normally gets reset manually before it rolls over, you must set the rollover value for the tag to 0 so that the count is simply how much change occurred since the manual reset.

For example, assume that you have the following data values for five consecutive cycle boundaries, and that the value 0 occurs as the first value within the last cycle:

```
100, 110, 117, 123, 3
```

If you set the rollover value to 0, the counter retrieval mode assumes that the 0 following the value 123 represents a manual reset, and returns a value of 3 for the last cycle, which is assumed to be the count after the manual reset. The value 0 itself does not contribute 1 to the counter value in this case.

If the rollover value is instead set to 200, then the counter retrieval mode assumes that the value 0 represents a normal rollover, and a count of 80 is calculated and returned (200 - 123 + 3). In this case, the value 0 contributes 1 to the counter value, and that is the change from the value 199 to the value 200.

Handling Exceptions

To handle reversals instead of treating them as rollovers, you can use a counter deadband. For more information, see Counter Retrieval - Using a Counter Deadband in the *AVEVA Historian Retrieval Guide.*

For details on how counter retrieval handles "gaps" in data and how to compensate for that in your queries, see Counter Retrieval and Counter Retrieval - How It Works in the *AVEVA Historian Retrieval Guide.*

Example: Counting Items with Counter Retrieval

To use the counter mode, set the following parameter in your query:

```
   wwRetrievalMode = 'Counter'
```

In the following example, the rollover value for the TotalProduced.Counter tag is set to 0. In a time span of 3 hours, the tag increments from 0 to 9999 twice. The following query returns the an hourly total count within the queried time span. The QualityDetail of 212 indicates that a counter rollover occurred during the reported time range.

```
   SELECT DateTime, TagName, Value, Quality, QualityDetail AS QD
      FROM History
         WHERE TagName like 'TotalProduced.Counter'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
        AND DateTime >= '2015-08-17 13:00'
        AND DateTime < '2015-08-17 16:00'
        AND wwResolution=3600000
        AND wwRetrievalMode='counter'
        AND wwQualityRule='optimistic'
        AND wwTimeStampRule='start'
```

The results are:

DateTime	TagName	Value	Quality	QD
2015-08-17 13:00:00.0000000	TotalProduced.Counter	4460	0	212
2015-08-17 14:00:00.0000000	TotalProduced.Counter	4544	0	64
2015-08-17 15:00:00.0000000	TotalProduced.Counter	4481	0	212

Scenario 3: Track liquid and gas use with custody meters

You can use the Counter retrieval feature to keep track of how much liquid or gas a process uses. For example, you could answer questions like:

- How much syrup was dispensed from Tank 3?
- How much propane have we used so far this month?

Using Counter Retrieval with Floating Point Counters

Counter retrieval also works with floating point counters, which is useful for flow meter data. Similar to the carton counter, some flow meters report accumulated flow and "roll over" after a certain amount of flow accumulates. For both examples, the need is to convert the accumulating measure to a "delta change" value over a given period. (For an example of totalizing a flow rate, see Scenario 5.)

Example: Tracking Liquid or Gas Volumes with Counter Retrieval

```
   SELECT DateTime, TagName, Value, Quality, QualityDetail AS QD
      FROM History
         WHERE TagName like 'ReceivedVolume.Counter'
         AND DateTime >= '2015-09-01 13:00'
         AND DateTime < '2015-09-03 18:00'
         AND wwResolution=3600000
         AND wwRetrievalMode='counter'
         AND wwQualityRule='optimistic'
         AND wwTimeStampRule='start'
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	Value	Quality	QD
2015-09-01 13:00:00.0000000	ReceivedVolume.Counter	8965.23345496515	0	212
2015-09-01 14:00:00.0000000	ReceivedVolume.Counter	8977.65306903776	0	212
2015-09-02 15:00:00.0000000	ReceivedVolume.Counter	8959.81258651845	0	212
2015-09-02 16:00:00.0000000	ReceivedVolume.Counter	8990.74717176479	0	212
2015-09-03 17:00:00.0000000	ReceivedVolume.Counter	8992.89247964001	0	212

Scenario 4: Analyze cycle time for your process

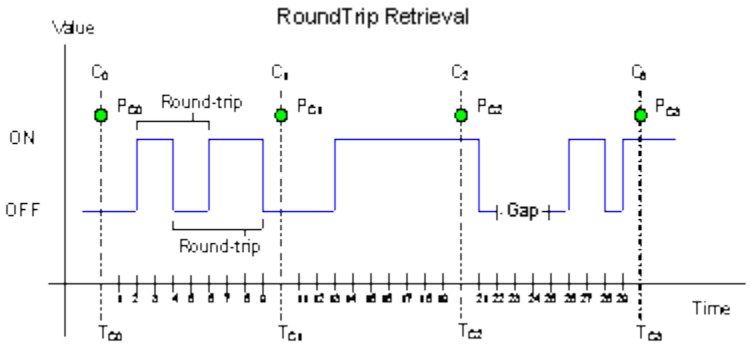
You can use the Roundtrip retrieval feature to analyze cycle time. For example, you could answer questions like:

- How long does it take between starting to fill one bag and starting to fill the next one?
- How long is the period between downtimes (or, the Mean Time Between Failures, MTBF)?
- How long between when a pump starts and the next time it starts?

What is RoundTrip Retrieval?

With RoundTrip retrieval, you can determine how long it take for the total cycle of a process to complete, through each "state" in the process. For example, a switch or a valve has two states -- on and off -- and therefore can be tracked using a discrete tag.

Here's an illustration of RoundTrip retrieval for a discrete tag that cycles through on and off states.



RoundTrip calculates the time a state begins until the time the same state begins again. For a discrete tag that simply tracks on and off, RoundTrip mode returns two rows per cycle. For example, when querying a discrete tag, RoundTrip can calculate the time between when a motor starts until the next time it starts, and when the motor stops until the next time it stops.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example: Analyzing Cycle Time with RoundTrip Retrieval

To use the RoundTrip retrieval mode, set the following parameter in your query:

```
wwRetrievalMode = ‘RoundTrip’
```

This query uses RoundTrip retrieval:

```
SELECT DateTime, TagName, vValue, StateTime
   FROM History
      WHERE TagName like 'Motor1.State'
      AND DateTime >= '2015-09-17 0:00'
      AND DateTime < '2015-09-17 2:00'
      AND wwResolution=3600000
      AND wwRetrievalMode='RoundTrip'
      AND wwStateCalc='MinContained'
      AND wwTimeStampRule='start'
```

The results are:

DateTime	TagName	vValue	StateTime
2015-09-17 00:00:00.0000000	Motor1.State	1	22001
2015-09-17 00:00:00.0000000	Motor1.State	0	21981
2015-09-17 01:00:00.0000000	Motor1.State	1	22001
2015-09-17 01:00:00.0000000	Motor1.State	0	24000

In these results, the value "0" shows when the valve was off and "1" when it was on.

The resulting rows show the average amount of time for each state and have a timestamp of the query end time (the default).

Scenario 5: Calculate the total flow

You can use the Integral retrieval feature to convert a rate into a quantity, such as totalizing a flow. For example, you can answer questions such as:

- How much vanilla extract have we used in the last month to produce ice cream?
- How much orange soda did we bottle yesterday?

What is Integral Retrieval?

Integral retrieval calculates the values at retrieval cycle boundaries by integrating the graph described by the points stored for the tag. It works much like average retrieval, but also applies a scaling factor.

Integral retrieval mode is ideal for calculating volume for a particular tag. For example, if a data tag represents product flow in liters per second, you can use integral retrieval to retrieve the total product flow in liters during a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

given time period.

Integral retrieval returns one row for each tag in the query for each cycle. You can specify cycles using resolution or cycle count.

Integral retrieval works only with analog tags.

Using Integral Instead of Counter

Integral retrieval and counter retrieval both deal in rates and quantities, but in different ways:

- **Counter retrieval** converts a total quantity measurement into a rate.

  For example, counter retrieval can take liters and a rate of 60000 to determine a rate in liters per minute.
- **Integral retrieval** converts a rate measurement into a quantity.

  For example, a rate in liters per minute can be translated into total liters for a given period.

Example: Calculating the Total Flow with Integral Retrieval

To use the integral retrieval mode, set the following parameter in your query:

```
wwRetrievalMode = 'Integral'
```

In this example, the value is computed for each of five one-minute cycles. The wwQualityRule parameter ensures that only points with good quality are used in the computation; data points with doubtful quality are discarded.

```
SELECT DateTime, TagName, Value, OPCQuality, PercentGood
   FROM History
      WHERE TagName like 'Flow1.PV'
      AND DateTime >= '2015-09-24 12:00'
      AND DateTime < '2015-09-25 8:00'
      AND wwRetrievalMode = 'Integral'
      AND wwCycleCount = 6
      AND wwQualityRule = 'Good'
```

The results of this query are:

DateTime	TagName	Value	OPCQuality	PercentGood
2015-09-24 12:00:00.000	Flow1.PV	15820.7228325108	192	100.0
2015-09-24 16:00:00.000	Flow1.PV	7776.62767927986	192	100.0
2015-09-24 20:00:00.000	Flow1.PV	14535.3472187875	192	79.0
2015-09-25 00:00:00.000	Flow1.PV	17058.0302965352	192	100.0
2015-09-25 04:00:00.000	Flow1.PV	18737.2427425968	192	100.0

The Value column shows the total flow for each 4-hour cycle. To calculate the total flow for the entire period, add

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

the flow values together.

Scenario 6: Calculate down time statistics

You can use the RoundTrip retrieval feature to calculate periodic downtime, including:

- What is the mean time between failures for my operation?
- What is the mean time to repair (or replace) for a certain component?

What are Mean Time Between Failures and Mean Time to Repair?

Both mean time between failures (MTBF) and mean time to repair or replace (MTTR) view the total cycle of a process.

- MTBF describes how long a process will likely run before the system goes down. To calculate this number, divide the total time processing by the number of breakdowns.
- MTTR describes how long it will probably take for a particular part to need repair or replacement. (For example, a pump would be repaired, but a paper filter would be replaced.) To calculate this number, divide the total downtime by the number of breakdowns.

Tracking States with RoundTrip Retrieval

Some parts in your process have just two states -- on and off -- while others have more. For example, an engine might have four states -- switch on, warm up, run, switch off. RoundTrip retrieval works with integer analog tags, discrete tags, and string tags to track a variety of states. You can use this type of retrieval with History and StateWideHistory tables.

RoundTrip retrieval performs calculations on state occurrences in the within a specified cycle period. RoundTrip retrieval uses the time between consecutive leading edges of the same state for its calculations.

The RoundTrip mode returns a row for each state in any given cycle. For example, if your engine has four states in a cycle, RoundTrip retrieval will return four rows per cycle.

Points on the boundary of the end cycle are considered part of the next cycle and are not counted in the calculation.

Example: Tracking MTBF with RoundTrip Retrieval

If the tag used is "1" when the equipment is down, this returns the MTBF over the period queried:

```
SELECT DateTime, TagName, StateTime, OPCQuality, PercentGood
   FROM History
   WHERE TagName like 'PLCSim.Boolean6'
      AND DateTime >= '2016-03-15 06:00'
      AND DateTime < '2016-03-15 08:00'
      AND wwRetrievalMode = 'RoundTrip'
      AND wwTimeStampRule='start'
      AND wwStateCalc='avgcontained'
      AND wwCycleCount = 1
      AND Value=1
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

The results of this query are:

DateTime	TagName	StateTime	OPCQuality	PercentGood
2016-03-15 06:00:00.0000000	PLCSim.Boolean6	12201.5602716469	192	100

Example: Tracking MTTR with RoundTrip Retrieval

This example shows how to track MTTR.

```
SELECT DateTime, TagName, StateTime, OPCQuality, PercentGood
   FROM History
   WHERE TagName like 'PLCSim.Boolean6'
      AND DateTime >= '2016-03-15 06:00'
      AND DateTime < '2016-03-15 08:00'AND wwRetrievalMode = 'ValueState'
      AND wwTimeStampRule='start'
      AND wwStateCalc='avgcontained'
      AND wwCycleCount = 1
      AND Value=1
```

The results of this query are:

DateTime	TagName	StateTime	OPCQuality	PercentGood
2016-03-15 06:00:00.0000000	PLCSim.Boolean6	6100.29661016949	192	100

Scenario 7: Pinpoint reasons for downtime

You can use State Summary information to help you answer questions like:

- How much downtime was due to feeder jams?
- What else is causing downtime?

What is State Summary Data?

State summary data summarizes the states of a tag value. You can use this to analyze process variables with a limited number of states, such as a machine's state of running/starting/stopping/off or a string that represents a downtime reason.

For each distinct state within a cycle, state summary replication also provides:

- Total time
- Percent of the cycle
- Shortest time
- Longest time
- Average time

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Using the StateSummaryHistory View

You can use the StateSummaryHistory view to retrieve state summary data.

A state summary results in a series of values, each representing a different state, for the same tag and time period. You configure the maximum states when you create the state summary tag. For more information, see State Summary Replication in the *AVEVA Historian Administration Guide.*

Example: Querying to Pinpoint Downtime

Suppose you know ahead of time what the likely causes for downtime might be in your plant. You could create a string or integer tag that identifies those causes. Then you can use a query like this one to track actual reasons for downtime in your facility:

```
SELECT DateTime, TagName, vValue, StateTime, OPCQuality AS 'OPC', PercentGood AS '%Good'
   FROM History
      WHERE TagName like 'Downtime.Status'
      AND DateTime >= '2015-10-07 0:00'
      AND DateTime < '2015-10-08 0:00'
      AND wwRetrievalMode = 'ValueState'
      AND wwTimeStampRule='start'
      AND wwStateCalc='percentcontained'
      AND wwCycleCount = 1
      ORDER BY StateTime DESC
```

The results of this query are:

DateTime	TagName	vValue	StateTime	OPC	%Good
2015-10-07 00:00:00.000	Downtime.Status	Running	55.173125	0	0
2015-10-07 00:00:00.000	Downtime.Status	Idle	15.4923715277778	0	0
2015-10-07 00:00:00.000	Downtime.Status	Material Jam	9.52366666666667	0	0
2015-10-07 00:00:00.000	Downtime.Status	Electrical	6.60920138888889	0	0
2015-10-07 00:00:00.000	Downtime.Status	Backed Up	5.18525925925926	0	0
2015-10-07 00:00:00.000	Downtime.Status	Equipment Jam	4.21300347222222	0	0
2015-10-07 00:00:00.000	Downtime.Status	Starved	3.33917083333333	0	0
2015-10-07 00:00:00.000	Downtime.Status	NULL	0.446497685185185	0	0

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Example: Finding the Minimum Time in a State

The following query finds the shortest closed/open time for the SteamValve discrete tag.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc
   FROM History
      WHERE TagName IN ('SteamValve')
      AND DateTime > '2015-10-04 0:00:00'
      AND DateTime <= '2015-10-08 0:00:00'
      AND wwCycleCount = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'MinContained'
```

The results of this query are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2015-10-08 00:00:00.000	SteamValve	0	20688	MinContained
2015-10-08 00:00:00.000	SteamValve	1	19105	MinContained

Example: Finding the Percentage of Time in a State

The following query returns the percentage of time in state for a discrete tag for multiple retrieval cycles. The TimeStampRule system parameter is set to "End" (the default setting), so the returned values are timestamped at the end of the cycle. Note that the first row returned represents the results for the period starting after 2015-10-05 0:00 and ending at 2015-10-08 0:00.

The "Percent" time-in-state retrieval mode is the only mode in which the StateTime column does not return time data. Instead, it returns percentage data (in the range of 0 to 100 percent) representing the percentage of time in state.

```
SELECT DateTime, TagName, vValue, StateTime, wwStateCalc
   FROM History
      WHERE TagName IN ('PLCSim.Boolean6')
      AND DateTime >= '2016-03-15 06:00'
      AND DateTime < '2016-03-15 08:00'
      AND Value = 1
      AND wwRetrievalMode = 'ValueState'
      AND wwStateCalc = 'Percent'
      AND wwCycleCount = 6
```

The results of this query are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2016-03-15 06:00:00.0000000	PLCSim.Boolean6	1	50.18475	Percent
2016-03-15 06:20:00.0000000	PLCSim.Boolean6	1	49.8515	Percent
2016-03-15	PLCSim.Boolean6	1	49.9315833333333	Percent

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	TagName	vValue	StateTime	wwStateCalc
06:40:00.0000000				
2016-03-15 07:00:00.0000000	PLCSim.Boolean6	1	50.1493333333333	Percent
2016-03-15 07:20:00.0000000	PLCSim.Boolean6	1	49.9735	Percent
2016-03-15 07:40:00.0000000	PLCSim.Boolean6	1	49.8709166666667	Percent

Scenario 8: Fill in data gaps

In a perfect world, instruments and data acquisition always capture measurements reliably, but the real world is full of interruptions. In some environments, these interruptions are so routine that they interfere with operational reporting. You can use the optimistic retrieval feature to help fill data gaps. For example, you could answer:

- If there is no data available at midnight when my report runs, how can find the most recent value before that?
- How can I fill in the gaps in my trend caused by unreliable communications?
- How do I make a "best guess" for information lost because of a communication failure between my equipment and the application server?

What is Optimistic Retrieval?

In addition to capturing data values, AVEVA Historian records the data quality (OPC quality) of a value. AVEVA Historian also marks any disconnects with NULL values and an associated QualityDetail of 10 and 24. By default, these appear in trends as gaps and in reports as NULL values .

Details about data gaps and less-than-optimal data quality are important for diagnosing communications problems and for supervisory control. However, there may be gaps in the good quality data, making it hard to create the reports you need to have. For example, it would be difficult to create a report with a total for yesterday's production or the raw material consumption at midnight if there were communications problems.

Using optimistic retrieval allows you to make a best guess without tampering with the underlying data.

Using the Optimistic quality rule lets you retrieve information that is possibly incomplete but likely to help fill in some gaps in the good quality data. This setting calculates using the last known good value before the gap (if possible). The logic for determining the percent good, value count, and OPC quality remains unchanged.

How Optimistic Retrieval Works

The following illustration shows a Counter retrieval situation where three of the four shown cycle boundaries are located in data gaps. Without using the Optimistic quality rule, Counter queries return NULL at all cycle boundaries because the mode needs valid good values at each end of the cycle calculate a precise difference.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

filter_counter_query

If the query specifies Optimistic, the results include a PercentGood column indicating whether all the data was "good" (100) or included some combination of "bad" and "uncertain" data (anything less than 100).

For the illustration above, there is no previous data and no available data in the first cycle, so it is skipped. At the second cycle boundary, the value 0 is returned, because there was a gap in the data for the entire first cycle. In the second cycle, there are two points, P1 and P2. The query uses P2 as the end value of the cycle and infers a start value of the cycle from P1. At the third cycle boundary, Tc2, the query returns P2 – P1. Similarly, at the last cycle boundary, the query returns P4 – P3.

Example: Masking Gaps in Your Data

At times, the results of a query might return null values, as in the following example:

```
SELECT TagName, DateTime, Value, PercentGood, QualityDetail
   FROM History
      WHERE TagName = 'BatchPercentConc'
      AND DateTime >= '2015-09-16 08:30'
      AND DateTime <= '2015-09-17 08:30'
      AND wwRetrievalMode = 'Delta'
```

The results are:

TagName	DateTime	Value	PercentGood	QualityDetail
BatchPercentConc	2015-09-16 08:30:00	(null)	0	65536
BatchPercentConc	2015-09-16 14:54:33	50	100	44
BatchPercentConc	2015-09-16 15:25:39	(null)	0	24
BatchPercentConc	2015-09-17 07:55:59	50	100	252
BatchPercentConc	2015-09-17 08:01:25	30	100	192
BatchPercentConc	2015-09-17 08:15:50	(null)	0	24
BatchPercentConc	2015-09-17 08:16:05	50	100	252

You can improve the usefulness of your data by adding this statement:

```
      wwQualityRule = 'Optimistic'
```

For example:

```
SELECT TagName, DateTime, Value, QualityDetail
   FROM History
      WHERE TagName = 'BatchPercentConc'
      AND DateTime >= '2015-09-16 08:30'
      AND DateTime <= '2015-09-17 08:30'
      AND wwRetrievalMode = 'Delta'
      AND wwQualityRule = 'Optimistic'
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

TagName	DateTime	Value	QualityDetail
BatchPercentConc	2015-09-16 08:30:00	(null)	65536
BatchPercentConc	2015-09-16 14:54:33	50	44
BatchPercentConc	2015-09-17 07:55:59	50	8444
BatchPercentConc	2015-09-17 08:01:25	30	192
BatchPercentConc	2015-09-17 08:16:05	50	8444

Scenario 9: Understand and use data

Data that is not 100% reliable can still be of value. Using data quality parameters, you can find out why data quality was marked uncertain, and when you find it useful, add data marked "uncertain" to your results.

You can use data quality indicators to answer questions like:

- What data did we lose because of a device failure?
- What was the volume readout before the power outage?

Data Quality in AVEVA Historian

When AVEVA Historian receives data from a data source, it adds a timestamp and quality indicator to the data value. Then, it stores the combined value, timestamp, and quality indicator (VTQ) as a data tag. The quality indicator has two parts, exposed by two separate columns:

- **OPCQuality:** What was the quality of this data as reported by the source?
- **QualityDetail:** What special processing has AVEVA Historian performed on the data?

OPC Data Quality: The Good, the Bad, and the Uncertain

OPC Data Access is an industry standard for real-time data exchange between hardware and software components. This standard defines a consistent way of exchanging live values by appending a data value with a timestamp and a quality value. In AVEVA Historian, this quality is exposed in the OPCQuality column. OPC defines quality as a 16-bit integer. The most significant bits for Historian are bits 6 and 7. They indicate whether the data's overall quality is "good", "bad" or "uncertain".

Historian stores and retrieves the timestamps and qualities for "good", "bad", and "uncertain" data. Historian always converts values with "bad" quality to NULL to protect users and applications from unwittingly propagating these values known to be bad.

The OPC standard allows status of "bad" to include a substatus. For example, a substatus of "5" means "Bad, Last Known Value" and "3" means "Bad, Device Failure". In a strict "current value only" protocol (like OPC DA), the best you can do is provide the quality and that previous value. However, AVEVA Historian will also store that previous value (with its quality and timestamp).

Note: Although in many cases OPCQuality and QualityDetail use the same codes to indicate similar concepts (for example, "192" indicates normal "good" values), they will often be different and should not be used

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

interchangeably.

QualityDetail: More to the Story

QualityDetail tells what happened with data while it was processed by various AVEVA Historian subsystems. For example, it can indicate that some data was stored as the first value after the connection to the historian was restored after a temporary disconnect, or it can tell that the retrieved value is not the actual value generated by an I/O Server or DAServer, but a manually corrected one.

Quality detail codes are metadata used to describe data values stored or retrieved using the AVEVA Historian. Quality details are 2 byte values that use the low order byte for the basic quality detail and the high order byte to store quality detail flags. The basic quality detail and quality detail flags are OR'd together.

Viewing Quality Values and Details

Descriptive labels for the many OPC quality codes are stored in the OPCQualityMap table of the Runtime database. To view OPCQuality values and descriptions, execute the following query:

```
SELECT OPCQuality, Description FROM OPCQualityMap
```

Similarly, descriptive labels for the many quality detail codes are stored in the QualityMap table of the Runtime database. To view the complete list of QualityDetail values and descriptions, execute the following query:

```
SELECT QualityDetail, QualityString FROM QualityMap
```

Note: Although a quality detail of 65536 is used to indicate block gaps, NULL values are not produced for block gaps.

Using WWQualityRule

AVEVA Historian provides three different options for interpreting the stored OPCQuality:

- **Good:** Filters out "uncertain" values, as if they did not exist, and converts "bad" values to NULL.
- **Extended**: Includes "uncertain" values, but takes them out of the "PercentGood" calculations and makes the OPCQuality "uncertain" for calculated results that include included them. Extended also converts any "bad" valuesto NULL.
- **Optimistic**: Filters out "bad", but is otherwise similar to "extended".

Examples: Using OPCQuality to Refine Results

Suppose you wanted a report to tell you the average production level for a certain storage tank for each 8-hour shift. You could run this query:

```
select DateTime, Value, OPCQuality, PercentGood
   from History
      where TagName like 'Flow273.PV'
      and DateTime > '2016-03-02 0:00'
      and DateTime <= '2016-03-02 3:00'
      and wwRetrievalMode = 'integral'
      and wwResolution = 60*60*1000
      and wwInterpolationType = 'stairstep'
      --and wwQualityRule = 'extended'
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Note that the last line is commented out because "QualityRule = 'extended'" is the default.

The results are:

DateTime	Value	OPCQuality	PercentGood
2016-03-02 01:00:00.0000000	255932.995605469	192	100
2016-03-02 02:00:00.0000000	263511.000823975	64	100
2016-03-02 03:00:00.0000000	190458.002471924	64	75

To force a query to exclude points with uncertain OPC quality, set wwQualityRule to Good. For example, you can change the previous example to this:

```
select DateTime, Value, OPCQuality, PercentGood
   from History
      where TagName like 'Flow273.PV'
      and DateTime > '2016-03-02 0:00'
      and DateTime <= '2016-03-02 3:00'
      and wwRetrievalMode = 'integral'
      and wwResolution = 60*60*1000
      and wwInterpolationType = 'stairstep'
      and wwQualityRule = 'good'
```

The results are:

DateTime	Value	OPCQuality	PercentGood
2016-03-02 01:00:00.0000000	255932.995605469	192	100
2016-03-02 02:00:00.0000000	261971.994781494	192	100
2016-03-02 03:00:00.0000000	190458.002471924	64	75

A third option is to change the quality rule to Optimistic:

```
select DateTime, Value, OPCQuality, PercentGood
   from History
      where TagName like 'Flow273.PV'
      and DateTime > '2016-03-02 0:00'
      and DateTime <= '2016-03-02 3:00'
      and wwRetrievalMode = 'integral'
      and wwResolution = 60*60*1000
      and wwInterpolationType = 'stairstep'
      and wwQualityRule = 'optimistic'
```

The results are:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

DateTime	Value	OPCQuality	PercentGood
2016-03-02 01:00:00.0000000	255932.995605469	192	100
2016-03-02 02:00:00.0000000	263511.000823975	64	100
2016-03-02 03:00:00.0000000	255474.000549316	192	75

Finally, queries to AnalogSummaryHistory use the Optimistic quality rule exclusively . For example:

```
select StartDateTime, Value, OPCQuality, PercentGood
   from AnalogSummaryHistory
      where TagName like 'Flow273.PV'
      and DateTime > '2016-03-02 0:00'
      and DateTime <= '2016-03-02 3:00'
      and wwRetrievalMode = 'integral'
      and wwResolution = 60*60*1000
      and wwInterpolationType = 'stairstep'
      and wwQualityRule = 'optimistic'
```

The results are:

StartDateTime	Value	OPCQuality	PercentGood
2016-03-02 01:00:00.0000000	263511.000823975	64	100
2016-03-02 02:00:00.0000000	255474.000549316	192	100

Scenario 10: Get summary statistics when a certain condition applies

You can use SliceBy to answer questions like:

- How much energy was consumed per batch?
- What was the average flow rate when a valve was open?
- What was lowest pressure today while a pump was running?

Using the SliceBy Parameter

SliceBy lets you define batches to be used with an Analog Summary query to get averages by batch. When you also use SliceByValue, you can get summary information per batch when a certain condition exists.

Example: Finding Total Energy Consumed per Batch

This query finds the total energy consumed per batch.

```
select SliceBy, SliceByValue, TagName, MIN(StartDateTime) as PeriodStart,
```

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

```
MAX(EndDateTime) as PeriodEnd, SUM(wwResolution)/1000 as [Duration(Sec)],
MIN(Minimum) as [Min], MAX(Maximum) as [Max], SUM(Integral) as [Total]
from AnalogSummaryHistory
where TagName='PowerMeter'
and SliceBy='R31.BatchNum'
and StartDateTime>='2019-04-17 0:00'
and EndDateTime<='2019-04-17 2:00'
group by TagName, SliceBy, SliceByValue
```

The results are:

SliceBy	SliceBy Value	TagName	PeriodStart	PeriodEnd	Duration(Sec)	Min	Max	Total
R31.BatchNum	844	PowerMeter	4/17/2019 0:02:42	4/17/2019 0:11:57	554	8.4944	15.1857	610.6690
R31.BatchNum	845	PowerMeter	4/17/2019 0:11:57	4/17/2019 0:25:30	812	6.7169	14.8926	1267.8577
R31.BatchNum	846	PowerMeter	4/17/2019 0:25:30	4/17/2019 0:34:45	554	5.3918	17.2337	431.7148

Example: Finding Average Flow Rate When Valve is Open

This query retrieves statistics on the flow rate, but only while the valve is open. The query finds an average of weighted averages.

First, the query defines a batch by instance that the valve is open. Then, it calculates the total volume of the flow for that batch and calculates a weighted average by duration. Finally, it calculates an "average of averages" for the batch.

```
select SliceBy, SliceByValue, TagName, MIN(StartDateTime) as PeriodStart,
MAX(EndDateTime) as PeriodEnd, SUM(wwResolution)/1000 as [Duration(Sec)], MAX(Maximum)
as [Max],
SUM(Integral) as [Total], AVG=SUM(Average*wwResolution)/SUM(wwResolution)
from AnalogSummaryHistory
where TagName='WaterFlow'
and SliceBy='R31_PLC.WaterValve'
and SliceByValue = 1 -- Valve Open
and StartDateTime>='2019-04-17 0:00'
and EndDateTime<='2019-04-17 2:00'
group by TagName, SliceBy, SliceByValue
```

The results are:

SliceBy	SliceBy Value	TagName	PeriodStart	PeriodEnd	Duration (Sec)	Max	Total	Avg
R31_PLC.WaterValve	1	WaterFlow	4/17/2019 0:02:45	4/17/2019 1:54:42	1199	1000	614995.85	512.51234

Note: This example correctly calculates the overall average for each state in the "Average" column by weighting the duration of each state. As explained by Simpson's Paradox, the simpler, "AvgOfAvg" calculation is not

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

statistically accurate and can differ significantly with some data sets.

Example: Finding the Lowest Pressure Today While a Pump Was Running

This example finds today's lowest pressure reading for a pump while it was running.

```
declare @date datetime
set @date = convert(date, getdate())
select SliceBy, SliceByValue, TagName, MIN(StartDateTime) as PeriodStart,
MAX(EndDateTime) as PeriodEnd, SUM(wwResolution)/1000 as [Duration(Sec)], MIN(Minimum)
[Min],
MAX(Maximum) as [Max]
from AnalogSummaryHistory
where TagName='LinePressure'
and SliceBy='StorageTank_001.OutletValve'
and SliceByValue = 1 -- Valve Open
and StartDateTime>= @date
group by TagName, SliceBy, SliceByValue
```

Where results might be:

SliceBy	SliceBy Value	TagName	PeriodStart	PeriodEnd	Duration (Sec)	Min	Max
Storage Tank_001. OutletValve	1	LinePressure	4/19/2019 0:25:10	4/19/2019 8:05:07	3094	9.0150	15.4113

Glossary

This section provides a glossary for the Historian server guides.

Welcome

This document provides a glossary for the AVEVA Historian Server documentation set.

AVEVA Historian Documentation Set

The AVEVA Historian documentation set includes the following guides:

- *AVEVA System Platform Installation Guide*
  This guide provides information on installing the AVEVA Historian, including hardware and software requirements and migration instructions.
- *AVEVA Historian Concepts Guide*
  This guide provides an overview of the entire AVEVA Historian system and describes each of the subsystems in detail.
- *AVEVA Historian Administration Guide*
  This guide describes how to administer and maintain an installed AVEVA Historian, such as configuring data

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

acquisition and storage, managing security, and monitoring the system.

- AVEVA Historian *Database Reference*
  This guide provides documentation for all of the AVEVA Historian database entities, such as tables, views, and stored procedures.
- *AVEVA Historian Scenarios Guide*
  This guide discusses how to use AVEVA Historian Server to address common customer scenarios.
- *AVEVA Historian Glossary*
  This guide provides definitions for terms used throughout the documentation set.

In addition, the *Invensys License Manager Guide* (InvensysLicenseManagerGuide.pdf) describes the Invensys License Manager and how to use it to install, maintain, and delete licenses and license servers on local and remote computers.

The AVEVA Historian documentation is also provided as an online help file, which can be accessed from the System Management Console management tool.

Documentation Conventions

This documentation uses the following conventions:

Convention	Used for
Initial Capitals	Paths and file names.
Bold	Menus, commands, dialog box names, and dialog box options.
`Monospace`	Code samples and display text.

Glossary

A B C D E F G H I J K L M N O P Q R S T U V WXYZ

action

An event action is the action that is configured to take place when the event detector determines that the event occurred. Event actions are not required; there are times when you may want to simply store when events happened. *See also* detector, event tag.

action queue

An event system action queue is a space in the action thread where a particular type of action is assigned before execution. An event action can be to either a critical, delayed, or normal action queue. Each type has its own queue.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

active image

The active image is an allocation of memory in which copies of values of acquired data are held to service client requests faster. The active image typically holds either the last 65 values for each data point or the number of values required to hold one minute of data, plus 10 percent. String tags configured with a length greater than 64 characters are not held in the active image.

aggregate functions

Aggregate functions are SQL functions that perform numerical calculations on a column in a set of data. Available aggregate functions include: SUM, AVG, MIN, MAX, and COUNT.

alarm

An alarm warns an operator about a process condition that could potentially cause a problem. Typically an alarm is set up to be triggered when a process value exceeds a defined limit. Alarms are detected by InTouch or AVEVA Application Server and recorded by AVEVA Historian's Alarm & Event History system.

alias

An alias is a name by which a networked server is known to clients on the network. A server can have multiple aliases.

alternate storage location

The alternate storage location is a directory on a computer that is used to store files that have been moved out of the circular storage location.

analog summary replication

Produces summary statistics for analog tags for a recorded interval.

annotation

An annotation is a user comment about a tag at a point in time.

application programming interface (API)

An application programming interface is a set of routines that an application can use to request lower-level services. APIs are available for use by application programmers when creating an application interface.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

architecture

A system's architecture describes the structure of a computer system, including the hardware and software that link the computers on a network.

archiving

Archiving is the process of removing records from the system and saving them to files so they can be restored later if needed. Alarm and event history stored in a SQL Server database may be archived using the Purge/Archive utility. Process history stored in history blocks can be archived and restored using Windows Explorer.

argument

An argument is the actual value passed to a function parameter.

attribute

In a database, attributes represent characteristics or properties associated with an entity, such as a tag, and usually correspond to column headings in a table.

authentication

Authentication is the process by which the logon information for a user is validated. Authentication is typically performed by a server on the network domain by comparing the logon information to an authorized list.

back end

Back end is a term that refers to the server in a client/server architecture. Data retrieval, processing, and storage occur at the back end, or server. *See also* server.

bandwidth

Bandwidth relates to the amount of data that can be transferred across a computer network in a given amount of time. Bandwidth is usually expressed in bits per second (bps) or baud.

block

See *history block*.

browser

A browser is a graphical representation of hierarchical groups of data. A browser is used to display disk

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

directories, folders, or files. For example, the System Management Console uses a browser to display servers, tags, and groups of tags in the system, private, and public namespaces. *See also* namespace.

buffer storage location

The buffer storage location is a directory on a computer that is used to store files temporarily, such as for retrieval from a data archive.

byte

A byte is a unit of information that consists of 8 bits. For data storage, a byte is equal to a single character, such as a number or a letter.

cache

Cache is special subsystem of memory in which frequently accessed data values are stored. A cache can be used as a buffer to hold data during transfers between a hard disk and random access memory (RAM). Cache memory is usually faster than RAM.

circular storage location

The circular storage location is a directory on the computer that stores historical data in files called history blocks.

client

A client is a computer that uses network services shared by the server computer. A client has full power and features for running applications, but is enhanced by the processing power of the server. The server provides data management, network administration, and security. Client computers are typically optimized for user interaction. *See also* server.

client/server

Client/server is a hardware and software architecture where the client (a user or program) makes requests (to the server) for resources or information. In this way, client/server computing enables two or more computers to share processing across a network.

Component Object Model (COM)

COM is a collection of services that allows software components to interoperate in a networked environment.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

configuration tables

Configuration tables contain information that defines most of the configuration aspects of AVEVA Historian. For example, definitions for tags, I/O Servers, and users are stored in configuration tables. *See also* extension table.

console

See *Microsoft Management Console*.

context

Meaningful description of the event or grouping to which a group of limits, rates of change, or deviations can belong. Examples are "Normal Operation," "Cold Shutdown," "My Personal Concerns."

CSV

CSV is the abbreviation for the comma-separated values format. In a file formatting according to CSV, data values are separated by commas.

cyclic retrieval

Cyclic-based retrieval is the retrieval of stored data for the given time period based on a specified cyclic retrieval resolution, regardless of whether or not the value of the tag(s) has changed. *See also* delta retrieval, resolution.

cyclic storage

Cyclic storage is the storing of analog data based on a time interval. Cyclic storage writes a record to history at the specified interval, only if a data change occurred during this time interval. *See also* delta storage.

data

The coded representation of information for use in a computer. Data has attributes, such as type and length.

data acquisition

Data acquisition is the process by which tag values are captured from various sources, such as from I/O Servers.

data dictionary

A data dictionary is a group of tables that contain information about all of the objects in the database.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

data integrity

Data integrity is the reliability and accuracy of data stored in the database.

data source

A data source is a database from which a client retrieves data.

data store

A data store is a file that contains data. A non-local data store is a data repository that exists outside of a Microsoft® SQL Server™ database.

data type

A data type specifies what type of information a table column can hold and how it is stored. There are two sources of data types: system-supplied data types and user-defined data types.

database

A database is a system repository of common types of data, sorted by unique identifiers and organized into tables. Databases are stored in files.

database name

The database name is used to identify a database. This name is used when establishing a connection from a client.

database object

See object.

database object owner

A database object owner is a user who creates a database object, such as a table, index, view, or rule. Database object owners have full permissions on any objects that they create, including the right to assign permissions for that object to other users.

database owner (dbo)

A database owner is a user who creates a database. Database owners have full access permissions for the database that they create, including the right to assign permissions for that database to other users. There can

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

be only one dbo for a database. Database ownership can be transferred between users, and multiple login IDs can be aliased to the dbo.

database query

See query.

DDE tags

A DDE tag is a tag that reads or writes its values to or from another application by means of the Microsoft Dynamic Data Exchange protocol. *See also* Dynamic Data Exchange.

deadband

A deadband is the amount of increase or decrease that a value can experience before an event will occur in the system. *See also* time deadband, value deadband.

delta retrieval

Delta retrieval, or retrieval based on exception, is the retrieval of only the changed tag values for a tag(s) for the given time interval. That is, duplicate values are not returned. *See also* cyclic retrieval.

delta storage

Delta storage is the storing of data based on a change in a value. Delta storage writes a record to history only if the value has changed since the last time it was written to history. Delta storage is also called "storage by exception." *See also* cyclic storage.

detector

An event detector is a mechanism for determining when the set of event criteria within the system has been satisfied for history data. *See also* event tag, action.

deviation

The deviation is the percentage of change in a tag's value from a fixed value, called the target. Each analog tag can have two defined deviations: major and minor.

discrete

A discrete value is a variable which only has two states: '1' (True, On) or '0' (False, Off). *See also* message pair.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Distributed Component Object Model (DCOM)

DCOM is a protocol that enables software components to communicate directly over a network.

domain

A domain is a group of computers that share a tree or subtree of a network for security authentication.

dynamic configuration

Dynamic configuration is the process of modifying the configuration of tags and other objects in the AVEVA Historian database while the system is running.

Dynamic Data Exchange (DDE)

DDE is the passage of data between applications, accomplished without user involvement or monitoring. In the Windows operating system environment, DDE is achieved through a set of message types, recommended procedures (protocols) for processing these message types, and some newly defined data types. By following the protocols, applications that were written independently of each other can pass data between themselves without involvement on the part of the user. For example, InTouch® HMI software and Microsoft Excel. *See also* topic, item, I/O Server, SuiteLink.

Dynamic Link Library (DLL)

A DLL is a software library of functions stored in a file and loaded into memory at execution time in order to be accessed by other functions or modules.

edge detection

Edge detection is the determination of the edge for a particular set of data. The edge is the imaginary "line" where, in a result set, the satisfaction of criteria changed from true to false, or vice-versa. *See also* leading edge, trailing edge.

engineering unit

An engineering unit is the unit of measure for a tag. For example, RPMs, milliseconds, degrees.

Note: Engineering units are case-sensitive. That means that AVEVA Historian can differentiate between ml (milliliters) and ML (megaliters), for example.

event

An event is a historical occurrence of a defined activity in the system. The definition for an event is stored as an event tag. Events are detected by event detectors and may be responded to by an event action. *See also*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

detector, action, event tag.

event block

Event blocks are the basic partitioning unit that users interact with. An event block is stored in a folder and generally has a collection of snapshots.

event tag

An event tag is a name for an event definition in the system. For example, if you wanted to detect how many times in history the temperature of tank reached 100 degrees, you might define an event tag and name it "TankAt100." *See also* detector, action, event.

extension table

In the AVEVA Historian, an extension table is a table that provide a means for handling acquired plant data. Extension tables are not part of normal SQL Server functionality. A normal SQL Server table stores data directly to the database file. An extension table, however, is a logical table that is populated from the data files created by the historian storage system. Data in the extension tables can be manipulated by using normal Transact-SQL code, as well as the specialized time domain extensions provided by the historian. *See also* view, configuration table.

failover

Failover is the process of substituting a backup resource, such as an IDAS, for a resource that is no longer functioning.

field

See row.

foreign key (FK)

A foreign key is one or more columns whose values match the primary key (PK) of some other table. A single primary key may have a foreign key in more than one table. *See also* key, primary key.

front end

Front end is a term that refers to the client in a client/server architecture. Database access or data input occurs at the front end, or client. *See also* client.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

frozen snapshot

Before being sent to disk, live snapshots can be frozen. After being frozen, events can no longer be added to the snapshot. This allows many optimizations to take place that allow efficient snapshot representation in both memory and disk. *See also* snapshot.

function

A function is a procedure in programming language. *See also* argument.

group

See user group.

HCAL

The Historian Client Access Layer (HCAL) is a client-side software layer that provides programmatic access to storage, retrieval, and system configuration functionality in the AVEVA Historian.

HCAP

The Historian Client Access Point (HCAP) is server-side software in the AVEVA Historian that accepts data from the Historian Client Access Layer (HCAL).

heterogeneous query

A heterogeneous query is a query that accesses data from multiple, dissimilar data sources.

history block

A history block is a group of data storage files that are written in a separate directory identified by a date stamp and a letter suffix. The AVEVA Historian stores acquired data to disk in blocks. History blocks are created when the historian starts, at a designated time interval, or when manually requested.

history block duration

The number of hours or days worth of data stored by a history block. Sometimes known as "hours per block".

history tables

In the AVEVA Historian, history tables present acquired plant data in a historical format, where there is one row for each stored tag value. *See also* "live" tables, "wide" tables.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Holding database

In an AVEVA Historian, the Holding database contains tables to temporarily store information imported from an InTouch data dictionary before it is transferred to the Runtime database. *See also* Runtime database.

Human-Machine Interface (HMI)

A human-machine interface is a software interface that allows plant floor operators to view, manipulate, and store plant data. An HMI can run on a PC or other industrial terminal.

Hypertext Transfer Protocol (HTTP)

HTTP is a protocol that enables the transfer of information over the Internet.

I/O

This is an abbreviation for Input/Output.

I/O Driver

See IDAS.

I/O Server

An I/O Server is an application that provides data to a client over a network by means of the DDE or SuiteLink protocol. *See also* Dynamic Data Exchange, SuiteLink.

IDAS

Industrial Server Data Acquisition Service (IDAS). This service accepts data values coming from SuiteLink and/or I/O servers, and forwards those values to AVEVA Historian.

IDENTITY column

An identity column contains a system-generated value that is used to uniquely identify each row in a table. If data is inserted into a table that has an identity column defined, the SQL Server will automatically generate a value. This value is based on the last assigned identity value, plus a pre-defined identity value increment.

index

An index is a set of pointers that provides faster access to data in rows of a table than a table scan. The concept of a table index is similar to an index at the back of a book; index entries make it much faster to find data than

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

starting at the beginning of the book and scanning until you find the information you are looking for. Indexes can also enforce uniqueness on rows in the table. There are two types of indexes: clustered and nonclustered.

information system (IS) network

The information system (IS) network is the business local or wide area network of a distributed AVEVA Historian. Computer workstations running AVEVA Historian client applications are most often connected to this network. *See also* process network.

initial value

Initial values are special values that can be returned from queries that lie exactly on the query start time, even if there is not a data point that specifically matches the specified start time.

initialization

Initialization refers to starting the AVEVA Historian.

in-memory snapshot

The in-memory snapshot stores the events which fall in the current block snapshot range.

integer

An integer is any member of the set consisting of the positive and negative whole numbers and zero. Examples: -59, -3, 0.

IP address

An IP address is a 32-bit address (Internet protocol address) that identifies a computer on a TCP/IP network. An IP address is normally written as four decimal numbers delimited by periods (.).

IPX/SPX

IPX/SPX is a transport protocol used in Novell networks.

item

In the DDE or SuiteLink protocol, a item is a data value placeholder. DDE protocol uses a three-part naming convention to locate information between applications. In order for an application (such as InTouch HMI software) to retrieve data, it must know the name of the application, the topic, and the item. An example of an item is the name of a cell in an Excel spreadsheet. Another example of an item is an InTouch tag. *See also* Dynamic Data Exchange, SuiteLink, topic.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

join

A join is a class of SQL query that queries data from one or more columns from two or more tables.

key

A key is a column that is used to identify a row. A row's key must be unique within the table. *See also* primary key, foreign key.

latency

Latency is the period between when an event actually occurs in the system and when it is detected by an event detector. *See also*, replication latency.

leading edge

The leading edge is the query return of only rows that are the first to successfully meet the criteria (return true) after a row did not successfully meet the criteria (returned false). *See also*, edge detection, trailing edge.

limit

A limit is a user-definable maximum or minimum value for a range of values.

linked server

A linked server is a SQL Server or an OLE DB provider that has been associated with a SQL Server. *See also* OLE DB provider.

live

Live describes data that reflects the most current value of a tag.

live tables

In the AVEVA Historian, "live" tables present the current (latest) values of acquired plant data for analog, discrete, or string tags. Live tables include streamed data, but do not include non-streamed data. *See also* history tables, ”wide” tables.

local

Local is used to describe the computer that a user is currently logged on to and is physically using. *See also* remote.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

local area network (LAN)

A LAN is a group of computers connected by a communications network. A LAN encompasses a relatively limited network area.

log file

A log file is a file that contains a database's transaction log. *See also* transaction log.

logical operators

A logical operator is used to calculate or compare data. Examples of logical operators are AND, OR, and NOT. The logical operators AND, OR, and NOT can be used in WHERE clauses to specify search conditions. AND means that both conditions are met. OR means that either of the conditions are met. NOT means that all conditions are met except those to the right of this operator.

logical tables

See view, extension table.

logical view

See view.

login ID

See login identification.

login identification

The login identification, or login ID, is a unique name that a database user uses to log on to the server.

logon

Logging on is the process of supplying a user name and password to obtain access to network resources.

MDAS

The Manual Data Access Service (MDAS) is a client-side software module that provides programmatic access to storage, retrieval, and system configuration functionality in the AVEVA Historian.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

memory tag

Memory tags are tag types that exist internally within an InTouch application. They can be used for creating system constants and simulations. They are also useful in creating calculated variables to be accessed by other Windows programs. There are four memory types: memory discrete, memory integer, memory real, and memory message.

message pair

A message pair includes the display strings associated with the TRUE (ON) or FALSE (OFF) states of a discrete value. *See also* discrete.

Microsoft Management Console (MMC)

Microsoft Management Console (MMC) is a container application that can host one or more third-party applications, called "snap-ins." The System Management Console is an MMC snap-in.

millisecond

One thousandth of a second, abbreviated ms or msec.

modification tracking

Modification tracking allows for the tracking of modifications to columns for certain tables in the database.

multi-protocol

The multi-protocol network library provides the capability for a server to listen for incoming network connections on named pipes, a TCP/IP port and an SPX socket. No ports or sockets need to be specified in the connection string, because a local RPC database is used to resolve the names over the supported protocols.

named pipe

Named pipes is an interprocess communication (IPC) mechanism used to transfer data between separate processes, usually on separate computers. In named pipes, a channel (pipe) is established by both processes for the transfer of data.

namespace

A namespace is a named set of objects. A namespace is simply a logical "area" that holds hierarchical groupings of objects. For example, servers, tags, or topics. There are three namespaces defined in the AVEVA Historian: the system namespace, the public namespace, and the private namespace. The hierarchical contents of a namespace are exposed in the browser of a client application. In the historian, the definition for what is included in a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

namespace is controlled by internal stored procedures. *See also* system namespace, public namespace, private namespace, browser, stored procedures.

network

A network is a communications infrastructure connecting a group of physically connected computers.

network address

A network address is a set of characters that uniquely identify a computer on a network.

network card

A network card is a physical extension card or device that provides a connection to a local area network (LAN) or wide area network (WAN).

node

A node is any computer or device that can be connected to an internetwork. A node is also referred to as a host.

node identifier

A node identifier locates a computer on the network. For example, an IP address. Used in conjunction with a process identifier for establishing client/server connections. *See also* process identifier.

NULL

NULL means that a column entry that has no assigned value. NULL is not equivalent to having a numeric value of zero or an empty string value. NULL is essentially the absence of a value. Unless a column is defined to allow NULLs, a value must be entered for the column.

object

An object is any of the components that constitute a database. Table, views, keys, defaults, triggers, indexes, stored procedures are all examples of database objects. Also called a database object.

Object Linking and Embedding for Databases (OLE DB)

Object Linking and Embedding for Databases (OLE DB) is an application programming interface (API) that allows COM-based client applications to access data that is not physically stored in the Microsoft SQL Server to which they are connecting.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

object owner

See database object owner.

object permission

Object permissions determine which statements can be used on database objects. Object permissions are managed by the database object owner for that object. *See also* object, database object owner.

OLE DB provider

A "virtual" server that provides an interface to access data in an OLE DB data store.

OPC quality

The quality of a process value or an event. The quality can be rated as Good, Bad, Doubtful, or Initial Value.

parameter

A parameter is an informational element that has a value. Parameters define the values to be written to or returned from the database.

partial cycle

Any cycle that is shortened so the cycle's duration ends exactly at the query end time.

password

A password is a unique set of characters used to authenticate a user and log on to a server.

permanent storage location

The permanent storage location is a directory on a computer that is used to store critical data (for example, reactor trips) that must not be overwritten. This storage location is the target directory used by the xp_DiskCopy extended stored procedure.

permission

Permissions restrict the actions that a database user can perform on a database. For example, a user may have permission to SELECT on all database tables, but not to INSERT any data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

phantom cycle

A phantom cycle is the name given to the cycle that leads up to the query start time. It is used to calculate which initial value to return at the query start time for all retrieval modes.

poll rate

The poll rate is the rate at which data is read from an acquisition device.

port number

A port number is a number from 0 to about 32,768 that identifies a particular application on a particular computer.

primary key (PK)

A primary key is one or more columns that uniquely identify a row in a table. A primary key is used for joins with foreign keys in other tables. *See also* key, foreign key.

priority

The event priority determines how events will be executed if the system becomes overloaded and cannot process all of the events. Those events that have been assigned a "critical" priority will be executed before events of a "normal" priority.

private namespace

The private namespace is a user-defined set of plant components, such as plant areas, machine names, and tags associated with a particular machine or process. The private namespace works in the same way as the public namespace, except that the private namespace is not available to all users of the AVEVA Historian. The private namespace is defined by a user of the historian. The hierarchical contents of the private namespace can be exposed in a directory tree of an application, but will only be visible to the user who created it. *See also* namespace, system namespace, public namespace, browser.

process

A process is a task, or service, that is performed by a computer's central processing unit (CPU).

process identifier

A process identifier locates a software process on a computer. For example, a pipe name or socket number. Used in conjunction with a node identifier for establishing client/server connections. *See also* node identifier.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

process network

A process network is the network to which plant floor control devices are physically attached. Devices on a process network include PLCs, DCSs, and HMI systems. *See also* information system network.

protocol

A protocol is the set of rules and standards to enable computers to connect and exchange data over a network.

public namespace

The public namespace is an administrator-defined set of plant components. Just as the system namespace includes information about an AVEVA Historian, the public namespace includes information about the plant on which the historian is running. The public namespace includes objects such as plant areas, machine names, and tags associated with a particular machine or process. The public namespace is defined by the historian administrator. The hierarchical contents of the public namespace can be exposed to all users in a directory tree of an application. *See also* namespace, system namespace, private namespace, browser.

purging

Purging is the process of permanently removing data from the system that is no longer required. *See also* archiving.

quality

Quality is an indicator of the accuracy, availability, and validity of acquired data. Data values stored by the AVEVA Historian have an associated quality value.

query

A query is a SQL script statement issued to the database by a client that searches for objects in a database table.

RAID

Redundant Array of Inexpensive Disks. RAID is a technology used to implement fault tolerance, or protection of data in the event of a hardware failure through the use of one or more physical disk drives. With fault tolerance, data is fully recovered with no downtime for the system.

Random Access Memory (RAM)

Random Access Memory is a type of memory that temporarily stores data while the computer on which it resides is running. When the computer is shut down, all data in RAM is erased. RAM can be written to or read from by the computer or other devices.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

rate of change

Rate of change is the rate that a tag value changes during a defined period of time, usually expressed as a percentage.

raw value

A raw value is the value of a data item when it was acquired. Calculations and conversions may be performed on raw data before it is used by the AVEVA Historian.

real

A real number is a floating point number represented by digits with a fixed base, such as the decimal system. A real number can be made up of either a finite or infinite set of digits.

real-time

Real-time operations occur at the same rate as a physical process. In a real-time environment, a computer must respond to situations as they occur. These situations can include a switch tripping or a furnace tapping.

record

See row.

redundant historian

A "partner" AVEVA Historian that can be used as a hot backup if the primary historian is not available.

referential integrity (RI)

Referential integrity is a mechanism that ensures that all foreign keys have an associated primary key in related tables. Referential integrity constraints prevent having foreign keys pointing to non-existent primary keys and enforces the relationship between tables. *See also* primary key, foreign key.

registry

The Windows registry is a database that contains all configuration information for a computer. The registry is organized hierarchically, and is comprised of hives, keys, sub-keys, and registry values.

reinitialization

Reinitialization is the restarting of one or more processes in the AVEVA Historian. When you make edits to the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Runtime database, a restart may be required in order for the changes to take effect.

relational database

A relational database is a database structure that organizes data according to the relationships between the data. In a relational database, relationships between data items are expressed by means of tables. For example, queries can be performed that search a single table, plus any related tables.

remote

Remote is used to describe a computer that is accessible from physically separated computers on a network. *See also* local.

remote table

A remote table is a data presentation table for a non-local data store accessed through OLE DB. *See also* data store, Object Linking and Embedding for Databases, extension table.

replication delay

The replication delay applies only to queued replication. This delay identifies how frequently "old" data, which includes inserts, updates, and store-and-forward data, is sent from the tier1 historian to the tier-2 historian.

replication group

A replication group abstracts a summary tag from a replication schedule, to make application maintenance easier. You can assign multiple summary tags to a single replication group, and assign multiple replication groups to a single schedule.

replication latency

The replication latency is the time it takes for a value to be made available for retrieval from the tier-2 historian from the moment it was stored or calculated on the tier-1 historian.

replication schedule

A replication schedule defines the specific times, in minutes or hours, for replication summary periods. For an interval-based replication schedule, cycle boundaries are calculated starting at midnight, tier-1 server local time, and continue in fixed time increments. For a custom replication schedule, replication cycles are forced to occur at user-defined fixed times of the day in tier-1 server local time.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

replication server

A replication server is the historian to which data is configured to be replicated. Also called a "tier-2" historian.

replication tag

A replication tag is a tag defined for a destination historian (tier-2) for which data from a source historian (tier-1) is copied or summarized.

resolution

Resolution is the sampling interval, in milliseconds, to retrieve data from any of the history tables of the AVEVA Historian. The resolution time domain extension is a feature provided by the historian and is not supported by normal SQL Server functionality. The number of rows returned is dependent upon the time period for the query and the resolution (number of rows = time period / resolution). Resolution only applies to cyclic retrieval.

restoring

Restoring is the process of adding records back to the database from archived files.

result

A result is the characteristics, or object attributes, of any object located by a database query.

row

In a table, a row is a set of related columns of information that describe a specific database entity. For example, for the entity "person," the row could contain column information for height, weight, hair color, or age.

row count

A row count determines the number of rows to be retrieved from any of the history tables of the AVEVA Historian. The row count time domain extension is a feature provided by the historian which, for cyclic retrieval, differs from normal SQL Server row count behavior. The application of the time domain row count extension depends whether you are using cyclic or delta retrieval, and whether you are querying a "wide" table.

rowset

Conceptually, a rowset is a set of rows in which each row contains columns of data.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

rule

A database rule is an object that is bound to a table column or to a user-defined data type. Rules determine what types of data can be entered in a column. For example, a rule can specify that the number for a unit of hours must be between 0 and 23. Only one rule can be applied to a column.

run time

Run time is the time during which data is fetched by the control unit and actual processing is performed in the arithmetic-logic unit. Also, the time during which a program is executing.

runtime database

In AVEVA Historian, the Runtime database contains tables that store all configuration, historical, and current process data. *See also* Holding database.

scaling

Scaling is the process of increasing or reducing the value of a variable (or a group of variables) by a given ratio.

script

A script is a collection of SQL statements used to perform actions on a database, such as change data or add new database objects. Scripts can be saved as stored procedures or files.

server

A server is a computer that shares resources, such as processing power and administration functions, for other computers on a network. Computers that use server resources are called clients. A server computer is typically responsible for data management, network administration, and security. A server computer also makes available to clients the processing power that was traditionally offered only by mainframes and minicomputers. The AVEVA Historian performs all of these functions, plus provides for data storage and management. *See also* client.

server name

The server name is the name of the server under which the AVEVA Historian is running. The server name must be a valid SQL identifier.

service

A service is a process that performs a specific function within the computer system.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

simple replication

Simple replication is a type of transformation that retains the data's original resolution. Analog, discrete, and string tags configured for simple replication replicate all values stored in the tier-1 historian to the tier-2 historian.

Small Computer Systems Interface (SCSI)

SCSI is a standard for a high-speed interface for connections between computers and peripheral devices, such as a hard drive.

snapshot

A snapshot is a collection of tag data values at a single point in time. When an event is detected in history data, the values of tags at the time of the event can be captured and stored. Snapshot data is useful in determining the state of a production environment at the time of a defined occurrence in history.

socket

A socket is a bi-directional channel, or "pipe," through which computers on a network can exchange information. The socket number identifies the channel and is made up of the IP address plus the port number. For example, 204.192.78.125,25. *See also* IP address.

sort order

A sort order is a set of definitions that specify how the SQL Server will organize and present data as a result of database queries. The sort order determines how the SQL Server will handle the collation of characters for both data storage and data retrieval operations involving the GROUP BY, ORDER BY, and DISTINCT statements. The SQL Server also uses the underlying sort order to resolve queries involving the WHERE and DISTINCT statements.

SQL

See Structured Query Language.

stand-alone installation

A stand-alone architecture consists of a single, non-networked computer that functions as the primary operator interface. This computer is connected to the industrial process by a direct connection, such as a serial cable.

state summary replication

State summary replication summarizes the states of a tag value. Can be applied to analog, discrete, and string tags.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

statement

An expression of instruction in a computer language.

storage by exception

See delta storage.

storage location

The storage location is the directory in which historical data files are stored.

storage path

The storage path is the path to the directory in which historical data files are stored.

storage rate

The storage rate is the time interval at which values for tags are periodically stored.

store-and-forward

Store-and-forward is a data caching process used by software applications (such as a remote IDAS) that automatically send data to a target computer on the network. If the remote application cannot communicate with the target computer, data will be cached locally until the connection is restored, at which time the cached data will be forwarded.

stored procedure

A stored procedure is a pre-compiled group of SQL statements. Stored procedures allow a group of sequentially performed actions to be executed using a single SQL statement. A stored procedure is usually called by another program to be executed; it is not automatically executed in response to an event. Stored procedures can be used as shortcuts for frequently used collections of SQL statements or to provide additional functionality. Users of the AVEVA Historian can use any of the system stored procedures provided by Microsoft SQL Server, plus the system stored procedures supplied with the historian. System stored procedures that are provided with the historian begin with "aa" or "we". User-defined stored procedures are also supported.

string

A string value is a text expression treated as a single data item. A string does not require a special format or syntax.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Structured Query Language (SQL)

SQL is a language used in relational database systems for defining, searching for, and manipulating data.

SuiteLink

SuiteLink is a network protocol designed specifically for high speed industrial applications. SuiteLink features Value Time Quality (VTQ) and places a timestamp and quality indicator on all data values delivered to VTQ-aware clients. SuiteLink uses a TCP/IP-based protocol.

summary

A summary (such as MIN, MAX, SUM, AVG) of a tag that is scheduled by the user and performed automatically according.

summary calculation queue

The summary calculation queue stores a record if a tier-1 historian is unable to perform a scheduled replication summary calculation for any reason.

summary data

Summary data is data that is the result of an internal calculation performed by the AVEVA Historian (maximum, average, sum). Summary data preserves a high-level view of data and allows for reduced storage space requirements for data kept for long amounts of time. For example, the average of five tags.

summary replication

Summary replication is a type of replication that provides low resolution summaries of high resolution data. During summary replication, statistics for a tag value are calculated at the tier-1 historian and then sent to the tier-2 historian.

summary tag

A summary tag contains the calculated data values, on a tier-2 historian, of information from a source tag on a tier-1 historian. Summary tag types are analog summary tags and state summary tags.

system administrator (sa)

The system administrator is the person responsible for administering and maintaining a SQL server database. Administration and maintenance functions include changing the database, administering database security, performing data and database backups.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

system namespace

The system namespace is a system-defined set of AVEVA Historian system components. The system namespace contains defined objects that make up a historian system, such as I/O Servers, or nodes. The historian populates the system namespace based on the current configuration for the plant, which is stored in the system configuration tables in the database. The hierarchical contents of the system namespace can be exposed to all users in a directory tree of an application. *See also* namespace, public namespace, private namespace, browser, stored procedures.

system parameter

A system parameter is a numeric or string value used for system configuration. System parameters are stored in the SystemParameter table in the AVEVA Historian database. For example, the version of the system or the version of the database is stored as a system parameter.

system tags

A system tag is a pre-defined system variable. InTouch system tags have a $ prefix. For example, $DateTime. AVEVA Historian system tags have a SYS prefix. For example, SysTimeSec.

table

A table is a group of related data entities and their characteristics. *See also* row.

tag

A tag is defined as an elemental variable of type analog, discrete, string, or complex that is stored in the AVEVA Historian database. In real terms, a tag typically refers to an instrument or device in your plant. It may also refer to system variables, such as the system time (SysTimeSec).

tagname

A tagname is the name assigned to an elemental variable in the AVEVA Historian database (a tag).

TCP/IP

Transmission Control Protocol/Internet Protocol (TCP/IP) is a group of networking protocols that allow communications across dissimilar networks. TCP/IP can route packet information across different hardware architectures and operating systems.

thread

A system thread is an object that independently performs an particular function within a process.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

tier-1 historian

A tier-1 historian is an individual historian that sends replicated data to a destination historian, called a tier-2 historian.

tier-2 historian

A tier-2 historian is a historian that accepts replicated data from one or more tier-1 historians.

time deadband

A time deadband is the minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. A time deadband also be applied at retrieval, in which case any value changes within the deadband will be ignored.

time interval

In the event system, the time interval is the rate at which configured event detectors check to see if an event has occurred in history. The time interval is also known as the scan rate.

time synchronization

Time synchronization is a mechanism by which the AVEVA Historian sends out a message to the I/O Servers to synchronize the I/O Server timestamps of data to the historian time.

topic

In the DDE or SuiteLink protocol, a topic is an application-specific subgroup of data elements. These protocols use a three-part naming convention to locate information in applications. In order for an application (such as InTouch HMI software) to retrieve data, it must know the name of the application, the topic, and the item. An example of a topic is the name of an Excel spreadsheet. *See also* Dynamic Data Exchange, SuiteLink.

trailing edge

The trailing edge is the query return of only rows that are the first to fail the criteria (return false) after a row successfully met criteria (returned true). *See also*, leading edge, edge detection.

transaction

A transaction is a collection of one or more command scripts that read and/or write to the relational database. A transaction is a request to the AVEVA Historian to find, enter, change, or return information about an object the relational database. All transactions are processed at runtime and are performed as a single unit of work. If a

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

single script fails at any point in the transaction, the entire transaction will be rolled back and the original state of the database before the transaction started will be restored. *See also* transaction log.

transaction log

A transaction log is a record of all database changes. *See also* transaction, log file.

Universal Naming Convention (UNC)

The universal naming convention is a standard for pointing to a file on a network. A UNC path consists of the following format: \\servername\sharename\path\filename

Universal Time Coordinate (UTC)

Universal Time Coordinate (UTC) is an absolute time designation used throughout the world.

update

An update is the alteration of data in a database, such as adding, deleting, or changing data.

user group

A user group is a group of database users that have permissions to perform certain actions on the database. User groups are implemented as part of database security. Any database user that is added to a user group inherits the permissions associated with that group.

user name

A user name identifies a database user for security purposes. A user name is assigned a login ID to allow a particular user access to the database.

user-defined data type

A user-defined data type is the definition of a type of data and is created by a user. User-defined data types exist in addition to predefined system data types. The type of data that can be stored in a column of a database table is determined by the data type defined for that column. Defaults and rules only apply to user-defined data types, not system data types.

value deadband

A value deadband is the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. A value deadband

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

can also be applied for retrieval.

view

A view is a logical way of looking at data from one or more tables in the database. A view is a "virtual" table; that is, it does not actually exist in the database. A view contains pointers to the actual tables in the database. Views can be used to include a subset of information stored in one or more tables, while leaving out other information. This is especially useful if some of the columns in a table contain sensitive information. Queries are performed on a view as if the view were a normal physical table. Views are part of normal SQL Server functionality. In the AVEVA Historian, however, data can be accessed using extension tables, which differ from normal views. *See also* extension table.

wide area network (WAN)

A WAN is a group of geographically separated computers connected by a communications network.

wide tables

In the AVEVA Historian, "wide" tables present the values of one or more tags over time. Each row contains the date/time stamp for the data and values for one or more tags specified in the query. *See also* "live" tables, history tables.

wildcard character

A wildcard character is a keyboard character that is used to represent one or more characters. When searching a SQL Server database, use the underscore (_), the percent sign (%), and brackets ([ ]), with the LIKE keyword to match patterns in the database. For example, to search for all tags in the system that started with "says", search for "sys%".

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Release Notes

This section includes the release notes for Historian and Historian Client.

Historian Release Notes

This section includes the release notes for Historian.

Historian 2023 R2 Patch 01 Readme

About this Readme - 2023 R2 Patch 01

This document describes the resolved issues and known issues in AVEVA Historian 2023 R2 Patch 01. It also includes the entire AVEVA Historian 2023 R2 Readme after the Patch 01 Resolved Issues.

Known Issues

Issue ID	Description
-	**Issue** The store and forward path for AutoSummary replication cannot be changed using the configurator or management console. **Workaround** The store and forward path for AutoSummary replication is stored in the _AutoReplicationServer table of the runtime database. Use the following steps to change the path: 1. Shut down and disable the Historian. 2. Create a backup of the Runtime database. 3. Execute the following SQL script in the Runtime database, replacing the value of *SFPath* with the path to the folder where you want AutoSummary replication data to be stored: `UPDATE _AutoReplicationServer SET SFPath = 'D:\Historian\StoreForward\LocalAutoSummaryReplication'`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	4. Enable and restart the Historian.
-	**Issue** Secure communication from Historian Client nodes to Historian server nodes may fail in workgroup setups with DNS enabled. This includes cloud-based environments such as Azure. **Workaround** In the hosts file on the Historian Client node, add an entry for the IP address of the the machine name (not the FQDN) of the Historian server. For example (where "10.x.x.x" is replaced with the IP address of your Historian server, and "HISTORIANSERVER1" is replaced with the name of that machine): `10.x.x.x HISTORIANSERVER1`

Resolved Issues - 2023 R2 Patch 01

This patch resolves the following issues:

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
IMS-2966258		960343330	SQL Server memory consumption increases, and Historian queries take longer.
IMS-2968797		960365321	Insight Publisher does not correctly publish discrete values from Intouch 2023
IMS-2997661		960373670	The in-program help doesn't work.
IMS-2982098		960370481	The Configurator fails with the message "Object reference not set to an instance of an object."
IMS-2895591		960355209	The auto summary backfill stalls.
IMS-2996548		960373266	Each time a full reimport is executed, tagimporter increments the value by one for MaxEU and MaxRaw.
IMS-3029223		960381994	ConfigStatusPending table is overruns the INT Size on the table identity.
IMS-3281342	IMS-2997831	960407831 960372728	The time zone for analogsummary queries is not provided and is considered as local.
IMS-3028192		960381805	In work group networks, when accessing Historian Client Web in SSO mode when Historian is configured with Flex license mode, the application loads in demo mode.
IMS-3061742	IMS-3190476	960389860	Security has been enhanced in Historian Client Web.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
IMS-3103049		960386536	When configuring IDAS in the configurator, the user name and password fields aren't editable.
IMS-3188323		960387013	Line charts on the dashboard change to column charts for shared links.
IMS-3238886		960399586	When running an "Average" retrieval query, it fails with an aaRetrieval access violation.
IMS-2997831		960372728	The Minutely API (- /apis/Historian/v2/Tags('TagName')/Minutely) shows the wrong start and end time.
IMS-3289466		960410095	Queries on [Runtime].[dbo].[StringHistory] show deviating results.
IMS-3309400		960410783	In Historian Client Web, when a URL includes tags, the content can't be saved. When saving, the error "The content couldn't be saved" occurs.

--- End of 2023 R2 Patch 01 Readme ---

Historian 2023 R2 Readme

About this Readme - 2023 R2

Latest revision: 12/15/2023

About This Readme

This Readme provides information about AVEVA Historian 2023 R2.

For information about using the product, see the AVEVA Historian documentation.

Known Issues

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Description
-	**Issue** The store and forward path for AutoSummary replication cannot be changed using the configurator or management console. **Workaround** The store and forward path for AutoSummary replication is stored in the _AutoReplicationServer table of the runtime database. Use the following steps to change the path: 1. Shut down and disable the Historian. 2. Create a backup of the Runtime database. 3. Execute the following SQL script in the Runtime database, replacing the value of *SFPath* with the path to the folder where you want AutoSummary replication data to be stored: `UPDATE _AutoReplicationServer SET SFPath = 'D:\Historian\StoreForward\LocalAutoSummaryReplication'` 4. Enable and restart the Historian.
-	**Issue** Secure communication from Historian Client nodes to Historian server nodes may fail in workgroup setups with DNS enabled. This includes cloud-based environments such as Azure. **Workaround** In the hosts file on the Historian Client node, add an entry for the IP address of the the machine name (not the FQDN) of the Historian server. For example (where "10.x.x.x" is replaced with the IP address of your Historian server, and "HISTORIANSERVER1" is replaced with the name of that machine): `10.x.x.x HISTORIANSERVER1`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

New Features in Historian 2023 R2

AVEVA Operations Control connected experience sign-on support

Historian supports the AVEVA Operations Control connected experience with AVEVA System Platform.

The server must be configured with an advanced license to retrieve data for expressions.

Note that the following still require Windows credentials:

- HTTP communication (HTTPS is required)
- Replication
- SDK connection using Classic Historian (WCF)
- Managing Historian with System Management Server

Changes to default Historian communications protocol and ports

The Historian communications protocol has been updated. The port for the new protocol is 32565 by default. The previous port, 32568, is supported as a legacy port for connecting servers that are running older software that doesn't support the new protocol.

Tier 1 Historians, AppEngines, and Platforms that support the new protocol and use the default 32568 port are updated automatically when the software is updated to the latest version. The local IDAS is also updated automatically. Existing connections that use legacy custom ports will need to be updated manually. External firewalls may need to be updated to allow connections on the new port.

- For a remote IDAS, change the port with the remote IDAS configurator.
- Update replication servers via the OCMC.
- Update application servers that use old ports in the engine definitions in the Galaxy.

Connecting servers that support the new protocol but are still using a custom legacy port will show as using the HCAL (Classic) protocol on the Data Acquisition page. (Note that connecting servers on older software will also use the HCAL (Classic) protocol. This is expected. Older software does not support the new protocol, and should still connect on the legacy port and use the classic protocol.)

If you have connecting servers that support the new protocol but still use the legacy port, the following warning appears once each hour in the Operations Control Logger:

```
The Historian Client Access Point running on XXX:XXX is an older version using a legacy protocol. Please update the port in the connection definition to point to the new Historian Client Access Point (port 32565 by default).
```

For more information, see the following sections in the *Historian Administrator Guide*:

- *IDAS Security and Firewalls*
- *Configuring IDAS on a Remote Node*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tags support multiple languages in Historian Client Web, Historian Client Query, and Historian Client Trend

You can now configure multiple translations for the same tag in Historian Client Web, Historian Client Query, and Historian Client Trend.

The language for these applications will be set automatically based on the operating system or browser language.

HTTPS-based communications

Internal Historian network communication is more secure and easily integrated with HTTP proxies.

Live data via iData

View latest historical values as the “live” values on OMI displays.

Alarm grid filter and sort in Historian Client Web

Browser-based alarm history in Historian Client Web supports more filtering and sorting options.

Historian Client Web / Excel integration

Easily transition from Historian Client Web charts to detailed analysis in Excel.

New look for Historian Client Web

The Historian Client Web interface is updated to be more powerful and easier to use.

Export / import Historian Client Web content

Include saved charts in CSV configuration import/export.

Resolved Issues in Historian 2023 R2

This release resolves the following issues:

Issue ID	Related Issues	SR/Case Number	Description
IMS-2633905		960334311	If an average() or total() expression includes cycles for which data is not yet available (typically cycles in the future, or cycles that are not available yet due to a network

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			disruption), the cycles without data now won't impact the time-weighting of the last value. This resolves an issue where Total Expression returns an incorrect value for the current period.
IMS-2638294		960333217	There is a one hour delay in the Alarms EventStamp (A2ALMDB).
IMS-2637686		960326700	SQL query results are not filtered consistently depending on the order of AND statements.
IMS-2482924		960296889	Rolling Sum chart doesn't display correctly when it's shared.
IMS-2504652		960303657	Using aahBlockMigrator.exe to convert a2almcb results in the columns "operator" and "UserFullName" being swapped.
IMS-2513233	IMS-2492540	960300539	There is an issue with the message events log in A2ALMDB when it's configured through the Configurator Utility.
IMS-2489147	IMS-2455484	960264222	Endbound retrieval has an extremely long running time compared to startbound retrieval.
IMS-2389867	IMS-2362433	960261473	An error shows in the configurator when configuring remote IDAS.
IMS-2416708	IMS-2396761	960278834	Publisher fails during configuration with the error `Invalid Value.`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			`The statement has been terminated. String or binary data would be truncated.`
IMS-2412336		960279736	IDAS Installation Conflict.
IMS-2329328		960261568	Incorrect Values in Insight.
IMS-2468109	IMS-2445491	960288997	Trends in Historian Client Web created prior to 2023 Patch 01 have color issues.
IMS-2455226		960287532	AppEngine "Store forward threshold" doesn't prevent StoreForward block from growing.
IMS-1984324	IMS-1974801	960174397	When alarms are migrated from alarm database to history blocks, the alarm duration field is not calculated correctly.
IMS-2008742		960204912	CheckValue is NULL in v_AlarmEventHistory2 and v_EventHistory views when alarms and events are stored in history blocks.
IMS-2046349		960211555	Retrieving data using OData in Power BI results in an error.
IMS-2072197		960212595	Migrating a runtime database from AVEVA Historian 2012 R2 to the current version results in a failure.
IMS-2094160		960220716	The Local Replication properties can't be edited.
IMS-2164943		960229221	ValueCount changes

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			significantly if query is for a time span longer then 5 hours.
IMS-2178880		960234046	The path for **xx:\Historian\StoreForward\LocalAutoSummaryReplication** can't be edited via the user interface.
IMS-2161218		960221324	Historian is one hour behind the System time.
IMS-2181475		960237600	Migrating a runtime database from AVEVA Historian 2017 U3 SP1 to certain versions results in a failure.
IMS-2086324		960218650	IDAS fails to configure after upgrading from AVEVA Historian 2020.
IMS-2179860	IMS-2179861	960232900	URL-based charts always open to the status board.
IMS-2204257		960223124	Retrieval using Integral with Auto Summary enabled produces errors.
IMS-2254783	IMS-2138814 IMS-2178159	960228162	Certain queries cause a memory leak in aahStorage, resulting in performance issues and eventually a crash.
IMS-2223710		960242318	Query with "Like" expression fails for certain History blocks.
IMS-2136268		960209303	When replicating a large volume of tags, the extended properties don't replicate correctly.
IMS-2290118		960209303	When replicating a large volume of tags, the error

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			`Main Metadata Server COM Exception caught, error = 38` occurred repeatedly.
IMS-2198072	IMS-2164943	960229221	ValueCount changes significantly if a query is for a period longer than 5 hours.
IMS-2311026			When automatic deletion of a history block is triggered by the Age Threshold setting, warnings and errors are recorded in the log.
IMS-2866275		960347144	Historian replication is not possible for Data Hub and Insight using DMZ Secure Link.

Historian 2023 R2 Readme

About this Readme - 2023 R2

Latest revision: 12/15/2023

About This Readme

This Readme provides information about AVEVA Historian 2023 R2.

For information about using the product, see the AVEVA Historian documentation.

Known Issues

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Description
-	**Issue** The store and forward path for AutoSummary replication cannot be changed using the configurator or management console. **Workaround** The store and forward path for AutoSummary replication is stored in the _AutoReplicationServer table of the runtime database. Use the following steps to change the path: 1. Shut down and disable the Historian. 2. Create a backup of the Runtime database. 3. Execute the following SQL script in the Runtime database, replacing the value of *SFPath* with the path to the folder where you want AutoSummary replication data to be stored: `UPDATE _AutoReplicationServer SET SFPath = 'D:\Historian\StoreForward\LocalAutoSummaryReplication'` 4. Enable and restart the Historian.
-	**Issue** Secure communication from Historian Client nodes to Historian server nodes may fail in workgroup setups with DNS enabled. This includes cloud-based environments such as Azure. **Workaround** In the hosts file on the Historian Client node, add an entry for the IP address of the the machine name (not the FQDN) of the Historian server. For example (where "10.x.x.x" is replaced with the IP address of your Historian server, and "HISTORIANSERVER1" is replaced with the name of that machine): `10.x.x.x HISTORIANSERVER1`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

New Features in Historian 2023 R2

AVEVA Operations Control connected experience sign-on support

Historian supports the AVEVA Operations Control connected experience with AVEVA System Platform.

The server must be configured with an advanced license to retrieve data for expressions.

Note that the following still require Windows credentials:

- HTTP communication (HTTPS is required)
- Replication
- SDK connection using Classic Historian (WCF)
- Managing Historian with System Management Server

Changes to default Historian communications protocol and ports

The Historian communications protocol has been updated. The port for the new protocol is 32565 by default. The previous port, 32568, is supported as a legacy port for connecting servers that are running older software that doesn't support the new protocol.

Tier 1 Historians, AppEngines, and Platforms that support the new protocol and use the default 32568 port are updated automatically when the software is updated to the latest version. The local IDAS is also updated automatically. Existing connections that use legacy custom ports will need to be updated manually. External firewalls may need to be updated to allow connections on the new port.

- For a remote IDAS, change the port with the remote IDAS configurator.
- Update replication servers via the OCMC.
- Update application servers that use old ports in the engine definitions in the Galaxy.

Connecting servers that support the new protocol but are still using a custom legacy port will show as using the HCAL (Classic) protocol on the Data Acquisition page. (Note that connecting servers on older software will also use the HCAL (Classic) protocol. This is expected. Older software does not support the new protocol, and should still connect on the legacy port and use the classic protocol.)

If you have connecting servers that support the new protocol but still use the legacy port, the following warning appears once each hour in the Operations Control Logger:

```
The Historian Client Access Point running on XXX:XXX is an older version using a legacy
protocol. Please update the port in the connection definition to point to the new Historian
Client Access Point (port 32565 by default).
```

For more information, see the following sections in the *Historian Administrator Guide*:

- *IDAS Security and Firewalls*
- *Configuring IDAS on a Remote Node*

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Tags support multiple languages in Historian Client Web, Historian Client Query, and Historian Client Trend

You can now configure multiple translations for the same tag in Historian Client Web, Historian Client Query, and Historian Client Trend.

The language for these applications will be set automatically based on the operating system or browser language.

HTTPS-based communications

Internal Historian network communication is more secure and easily integrated with HTTP proxies.

Live data via iData

View latest historical values as the “live” values on OMI displays.

Alarm grid filter and sort in Historian Client Web

Browser-based alarm history in Historian Client Web supports more filtering and sorting options.

Historian Client Web / Excel integration

Easily transition from Historian Client Web charts to detailed analysis in Excel.

New look for Historian Client Web

The Historian Client Web interface is updated to be more powerful and easier to use.

Export / import Historian Client Web content

Include saved charts in CSV configuration import/export.

Resolved Issues in Historian 2023 R2

This release resolves the following issues:

Issue ID	Related Issues	SR/Case Number	Description
IMS-2633905		960334311	If an average() or total() expression includes cycles for which data is not yet available (typically cycles in the future, or cycles that are not available yet due to a network

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			disruption), the cycles without data now won't impact the time-weighting of the last value. This resolves an issue where Total Expression returns an incorrect value for the current period.
IMS-2638294		960333217	There is a one hour delay in the Alarms EventStamp (A2ALMDB).
IMS-2637686		960326700	SQL query results are not filtered consistently depending on the order of AND statements.
IMS-2482924		960296889	Rolling Sum chart doesn't display correctly when it's shared.
IMS-2504652		960303657	Using aahBlockMigrator.exe to convert a2almcb results in the columns "operator" and "UserFullName" being swapped.
IMS-2513233	IMS-2492540	960300539	There is an issue with the message events log in A2ALMDB when it's configured through the Configurator Utility.
IMS-2489147	IMS-2455484	960264222	Endbound retrieval has an extremely long running time compared to startbound retrieval.
IMS-2389867	IMS-2362433	960261473	An error shows in the configurator when configuring remote IDAS.
IMS-2416708	IMS-2396761	960278834	Publisher fails during configuration with the error `Invalid Value.`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			`The statement has been terminated. String or binary data would be truncated.`
IMS-2412336		960279736	IDAS Installation Conflict.
IMS-2329328		960261568	Incorrect Values in Insight.
IMS-2468109	IMS-2445491	960288997	Trends in Historian Client Web created prior to 2023 Patch 01 have color issues.
IMS-2455226		960287532	AppEngine "Store forward threshold" doesn't prevent StoreForward block from growing.
IMS-1984324	IMS-1974801	960174397	When alarms are migrated from alarm database to history blocks, the alarm duration field is not calculated correctly.
IMS-2008742		960204912	CheckValue is NULL in v_AlarmEventHistory2 and v_EventHistory views when alarms and events are stored in history blocks.
IMS-2046349		960211555	Retrieving data using OData in Power BI results in an error.
IMS-2072197		960212595	Migrating a runtime database from AVEVA Historian 2012 R2 to the current version results in a failure.
IMS-2094160		960220716	The Local Replication properties can't be edited.
IMS-2164943		960229221	ValueCount changes

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			significantly if query is for a time span longer then 5 hours.
IMS-2178880		960234046	The path for **xx:\Historian\StoreForward\LocalAutoSummaryReplication** can't be edited via the user interface.
IMS-2161218		960221324	Historian is one hour behind the System time.
IMS-2181475		960237600	Migrating a runtime database from AVEVA Historian 2017 U3 SP1 to certain versions results in a failure.
IMS-2086324		960218650	IDAS fails to configure after upgrading from AVEVA Historian 2020.
IMS-2179860	IMS-2179861	960232900	URL-based charts always open to the status board.
IMS-2204257		960223124	Retrieval using Integral with Auto Summary enabled produces errors.
IMS-2254783	IMS-2138814 IMS-2178159	960228162	Certain queries cause a memory leak in aahStorage, resulting in performance issues and eventually a crash.
IMS-2223710		960242318	Query with "Like" expression fails for certain History blocks.
IMS-2136268		960209303	When replicating a large volume of tags, the extended properties don't replicate correctly.
IMS-2290118		960209303	When replicating a large volume of tags, the error

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			`Main Metadata Server COM Exception caught, error = 38` occurred repeatedly.
IMS-2198072	IMS-2164943	960229221	ValueCount changes significantly if a query is for a period longer than 5 hours.
IMS-2311026			When automatic deletion of a history block is triggered by the Age Threshold setting, warnings and errors are recorded in the log.
IMS-2866275		960347144	Historian replication is not possible for Data Hub and Insight using DMZ Secure Link.

Historian Client Release Notes

This section includes the release notes for Historian Client.

Historian Client 2023 R2 Patch 01 Readme

Last revision: 10/23/2024

About this Readme - 2023 R2 Patch 01

This document describes the resolved issues and known issues in AVEVA Historian Client 2023 R2 Patch 01. It also includes the entire AVEVA Historian Client 2023 R2 Readme after the Patch 01 Resolved Issues.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Known Issues

Issue ID	Description
-	Client Controls which include any of the Historian Client Controls (Trend, Query, TimePicker, etc.) require an additional step to import correctly. Import the aaPKG files from **C:\Program Files (x86)\ArchestrA\Framework\Bin** first. The individual Client Control dll files can then be imported normally afterwards.
-	**Runtime language switching limitations** • Runtime language switching is not supported for embedded Query controls. • For the Query tool, runtime language switching is only applicable to the Description field. • For SQL connections, runtime language switching is only available for SQL-supported languages.
-	When running a query from the Historian tab in Microsoft Word, a runtime error occurs if macros are not enabled. **Workaround** Macros must be enabled when using the Historian Client Report add-in for Microsoft Word.
-	Under certain circumstances, when migrating a CAB file to AVEVA System Platform 2023 R2 or later from a galaxy older than System Platform 2020 that contains ArchestrA Graphics and utilizes the Historical Trend control, the migration may stall and be unable to recover. No error message is displayed.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	Workaround 1. Create a backup of the existing galaxy. 2. Save a copy of the following Historian Client binaries from an installation of AVEVA System Platform that matches the version your galaxy is migrating from: • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaHistClientDatabase.dll** • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaAFCommonTypes.dll** • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaHistClientUtil.dll** 3. Install AVEVA System Platform 2023 R2 or later. 4. Open the **System Platform IDE**. 5. Create a new galaxy with the same name as the previous galaxy. Select the **default_empty.cab** type. 6. Close the **System Platform IDE**. 7. Copy the binaries saved from the previous version of AVEVA System Platform (from step 2) to: **%ProgramFiles(x86)%\Common Files\ArchestrA\Framework\Bin** 8. Open **aaSMC** and restore the galaxy. 9. Open the **System Platform IDE** and connect to the galaxy. This completes the migration. After successful migration, there may be errors reported from the aahEventEnum and aahEventSvc components, and warnings from the aahReplication component. These errors and warnings can be ignored.
-	When importing AVEVA .NET Apps that use AVEVA Historian Client controls, some of the required files are not imported correctly.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Workaround	1. Import the App using the System Platform IDE by selecting **Import** from the **Galaxy** menu. 2. Select **AVEVA App**. 3. Select the folder containing the App to be imported. 4. The App is imported and all the dependent files are copied to a new folder, **C:\Program Files (x86)\ArchestrA\Framework\FileRepository\<GalaxyName>\ObjectFileStorage2\xxx**, where **xxx** is a 3-digit number. **Tip:** Sorting the **ObjectFileStorage2** folder by date modified can make it easier to locate the correct numbered folder. 5. Some of the required files for AVEVA Historian Client controls may not be copied. For each of the controls your imported App uses, locate the corresponding folder(s) from the following list, and copy all the files from those folders into the folder from step 4: • Query control: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientQueryControl.dll** • Tag Picker: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTagPicker.dll** • Time Range Picker: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTimeRangePicker.dll** • Trend control: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTrendControl.dll**

Resolved Issues - 2023 R2 Patch 01

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
IMS-2996637		960373381	In Historian Client Trend, the options for tags display incorrectly.
IMS-2987403		960333105	Historian Client control tooltip may fail to work sometimes.
IMS-2995299		960369817	When an alarm is inhibited then uninhibited, the entire curve is highlighted in yellow.
IMS-2995897		960372801	Discrete values are not displayed in the Query client when the "ALL" tab is selected in the tag picker for history values.
IMS-3018608		960379143	The FROM clause disappears from the SQL query code if you uncheck either the QualityDetail or QualityDescription option for "Live Values" when using Historian Client Query.
IMS-2960326	IMS-3025318	960364301	Historian Client Trend and Query hang on opening.
IMS-3053957		960386657	The event tag isn't displaying in Trend when non-event tags are also displayed.
IMS-3025741		960378382	When creating a new galaxy, the client controls aaHistClientTrendControl, aaHistClientActiveDataGrid, aaHistClientSingleValueEntry, aaTagPicker, and aaTimeRangePicker are missing.
IMS-3186974		960393140	An error occurs when

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Issue ID	Related Issues	SR/Case Number	Description
			retrieving tag values from Historian Server 2017 Update 3 SP1.
IMS-3198534		960396138	After migrating a galaxy from SP2023 P02 to 2023 R2, scripts time out. This is particularly prevalent with scripts that call out a Namespace attribute.
IMS-3207981		960397172	In the Query tool, text pasted into the filter fields "Tag Name", "Description", and "I/O Address" pastes twice.
IMS-3258436		960396124	When switching from Windows to SQL authentication in the Excel add-in, the following Visual Basic error occurs: "Run-time error '-2147467261 (80004003)' Object reference not set to an instance of an object".
IMS-3265360		960390581	In the Query tool, queries fail with the error "Error executing query: Invalid column name 'wwUnit'."
IMS-3314949		960414166	When using AVEVA Window Maker, the follow error occurs: "Unhandled exception (Micorsoft .NET Framework) - ArchestrA.HistClient.UI.TrendAction.LicenseTimer_Tick".
IMS-3338549			The aaTrendClient control doesn't show tag properties.

--- End of 2023 R2 Patch 01 Readme ---

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Historian Client 2023 R2 Readme

Historian Client 2023 R2

Last revision: 11/30/2023

About This Readme

This Readme provides information about AVEVA Historian Client 2023 R2.

For information about using the product, see the AVEVA Historian Client documentation.

Known Issues in This Release

Issue ID	Description
-	**Issue** When running a query from the Historian tab in Microsoft Word, a runtime error occurs if macros are not enabled. **Workaround** Macros must be enabled when using the Historian Client Report add-in for Microsoft Word.
-	**Issue** Under certain circumstances, when migrating a CAB file to AVEVA System Platform 2023 R2 from a galaxy older than System Platform 2020 that contains ArchestrA Graphics and utilizes the Historical Trend control, the migration may stall and be unable to recover. No error message is displayed. **Workaround** 1. Create a backup of the existing galaxy. 2. Save a copy of the following Historian Client binaries from an installation of AVEVA System Platform that matches the version your galaxy is migrating from: • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaHistClientDatabase.dll** • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaAFCommonTypes.dll**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	• **%ProgramFiles(x86)%\Common Files\ArchestrA\aaHistClientUtil.dll** 3. Install AVEVA System Platform 2023 R2. 4. Open the **System Platform IDE**. 5. Create a new galaxy with the same name as the previous galaxy. Select the **default_empty.cab** type. 6. Close the **System Platform IDE**. 7. Copy the binaries saved from the previous version of AVEVA System Platform (from step 2) to: **%ProgramFiles(x86)%\Common Files\ArchestrA\Framework\Bin** 8. Open **aaSMC** and restore the galaxy. 9. Open the **System Platform IDE** and connect to the galaxy. This completes the migration. After successful migration, there may be errors reported from the aahEventEnum and aahEventSvc components, and warnings from the aahReplication component. These errors and warnings can be ignored.
-	When importing AVEVA .NET Apps that use AVEVA Historian Client controls, some of the required files are not imported correctly. **Workaround:** 1. Import the App using the System Platform IDE by selecting **Import** from the **Galaxy** menu. 2. Select **AVEVA App**. 3. Select the folder containing the App to be imported. 4. The App is imported and all the dependent files are copied to a new folder, **C:\Program Files (x86)\ArchestrA\Framework\FileRepository\<GalaxyName>\ObjectFileStorage2\xxx**, where **xxx** is a 3-digit number. **Tip:** Sorting the **ObjectFileStorage2** folder by date modified can make it easier to locate the correct numbered folder. 5. Some of the required files for AVEVA Historian Client controls may not be copied. For each of the controls your imported App uses, locate the corresponding folder(s) from the following list,

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	and copy all the files from those folders into the folder from step 4: • Query control: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientQueryControl.dll** • Tag Picker: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTagPicker.dll** • Time Range Picker: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTimeRangePicker.dll** • Trend control: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTrendControl.dll**

New features in Historian Client 2023 R2

Ad-Hoc expressions in Trend

Historian Client Desktop Trend control and its embedded ActiveX controls now support Ad-Hoc expressions. The Expression application and pane now appear in the Tag Picker with the Textbox editor.

For embedded ActiveX controls, scripting is supported for ad-hoc expressions in InTouch applications.

Ad-hoc expressions are supported with both SQL and REST connections.

Expanded ad-hoc expression functions

New functions added to ad hoc expressions make results simpler and more valuable.

Runtime language switching

Runtime language switching is supported for Trend Control, Query Control, Embedded control, and the Excel and Word ActiveX controls. It's supported for the following fields:

- Description
- EngineeringUnit
- Message On/Message Off
- Alias

If there is no value for the requested language for a field, the default language is used.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Improved Trend control for InTouch

Configuration is simplified for the Trend control within InTouch. The control is available in the graphic editor and can be placed in InTouch directly.

AVEVA Identity Manager Server login

Trend and Query support single sign-on authentication with AVEVA Identity Manager. This is restricted to REST-based connections and is not available for SQL-based connections.

Additional script functions for OMI Trend App and Graphic Editor support

The event tags TagAdded and TagRemoved have been added and are available at the Graphic level and in the HistoricalTrendApp control.

General performance improvements

Performance has been improved across the Historian Client controls.

Microsoft Office Historian Client Workbook improvements

The classic Historian Client Workbook add-in has been updated. Note the following:

- Microsoft Office is optional. If you install Office after installing Historian Client, the Historian Client add-ins do not appear in the list of Office add-ins. Run the Historian Client installation program and repair the installation to load the Office add-ins.
- When you perform a custom installation of Microsoft Office (32-bit), you must install Office Shared Features and Office Tools so that the Historian add-ins for Workbook or Report can be loaded.

  Note: Make sure that you are not using the 64-bit version of Microsoft Office.

Tool for migrating from Historian Client Workbook to the newer Excel task pane add-in

The Excel Migration Tool updates Excel workbooks created in Historian Client Workbook to formulas compatible with the newer Historian Excel task pane add-in.

This tool is supported for files created in Excel 2016 or later.

For more details, see the *Migrate data from Historian Client Workbook to the Excel add-in* section in the *AVEVA Historian Client User Guide*.

Resolved issues in Historian Client 2023 R2

This release resolves the following issues:

Issue ID	Related Issues	SR/Case Number	Description
IMS-2868362		960348145	The Historian Client Trend

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

			MovingAverage retrieval style fails. The following warning message appears in the SMC log: "Error creating dataview: System.Data.SqlClient.Sql Exception (0x80131904): Incorrect syntax near the keyword 'AND..."
IMS-2829074		960338977	Trend Client refresh does not work as expected.
IMS-1434259		960200215 960274781 960277990 960273174	The Servers.xml file was encrypted, so couldn't be shared between user profiles. To address this issue, now only the password is encrypted for SQL authentication. The non-sensitive information is no longer encrypted.
Bug-2600099			If servers.xml is upgraded from a previous version, the Historian Client Trend control crashes.
IMS-2616467		960329790	Setting a tag to a moving average in Historian Trend causes all other tag trends to disappear.
IMS-2599739	IMS-2590504	960322009	Windows that open from built-in buttons on the Trend App are not brought to the front.
IMS-2842698	IMS-2635827	960324888	When the the properties of an engineering unit are changed, the changes don't propagate to retrieval until the Historian services are restarted.
IMS-2510162		960305816	Unable to retrieve discrete live or historical values from Historian Client Query.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

IMS-2602879	IMS-1500586	960327042	OMI times out when opening a window with Historian Client Trend.
IMS-2412412		960276859	HistClient 2023 VBA function ActiveFactoryWorkbook.Addserver fails with the error `Runtime error 5, Invalid procedure call or argument.`
IMS-2361029		960254917	ShowGraphic function to open an embedded Historian Client trend in the application from a saved aatrend file is very slow.
IMS-2447731	IMS-2426902 IMS-2457568 IMS-2473685	960284737	aaTrend & aaQuery client tool buttons are purple.
IMS-2468523		960293562	When using the Historical Trend App, if more than one instance is opened at a time, the settings affect each other.
IMS-2111271		960223078	When Start Date, End Date, or Duration is changed, Update To Current Time is enabled.
IMS-2221595		960244148	Historian Client (Query & Trend) “File” menu item not visible if the Format in the Regional settings is set different from “English (United States)”
IMS-2200586		960237237	Historian Client Trend application always prompts for unnecessary save dialog for last added trend
IMS-2260942		960249515	Historian Client aaTrendControl does not

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

			accept the Bottom value entered for Value Axis Range
IMS-2201698		960239189	The trend duration is not properly applied to trends in a trend set
IMS-2308959		960256488	Trend control buttons disappear when the control is used within InTouch Access Anywhere
			When using the Trend application on a tablet PC, the Annotate Chart and Annotate Application options are missing from the Tools menu.
IMS-2333570		960264275	When using the aaHistClientActiveDataGrid ActiveX control in InTouch with a SQL login, the control fails to connect to the server.

Historian Client 2023 R2 Readme

Historian Client 2023 R2

Last revision: 11/30/2023

About This Readme

This Readme provides information about AVEVA Historian Client 2023 R2.

For information about using the product, see the AVEVA Historian Client documentation.

Known Issues in This Release

Issue ID	Description
- **Issue**	When running a query from the Historian tab in Microsoft Word, a runtime error occurs if macros are not enabled.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Workaround	Macros must be enabled when using the Historian Client Report add-in for Microsoft Word.
- **Issue**	Under certain circumstances, when migrating a CAB file to AVEVA System Platform 2023 R2 from a galaxy older than System Platform 2020 that contains ArchestrA Graphics and utilizes the Historical Trend control, the migration may stall and be unable to recover. No error message is displayed.
Workaround	1. Create a backup of the existing galaxy. 2. Save a copy of the following Historian Client binaries from an installation of AVEVA System Platform that matches the version your galaxy is migrating from: • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaHistClientDatabase.dll** • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaAFCommonTypes.dll** • **%ProgramFiles(x86)%\Common Files\ArchestrA\aaHistClientUtil.dll** 3. Install AVEVA System Platform 2023 R2. 4. Open the **System Platform IDE**. 5. Create a new galaxy with the same name as the previous galaxy. Select the **default_empty.cab** type. 6. Close the **System Platform IDE**. 7. Copy the binaries saved from the previous version of AVEVA System Platform (from step 2) to: **%ProgramFiles(x86)%\Common Files\ArchestrA\Framework\Bin** 8. Open **aaSMC** and restore the galaxy. 9. Open the **System Platform IDE** and connect to the galaxy. This completes the migration.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	After successful migration, there may be errors reported from the aahEventEnum and aahEventSvc components, and warnings from the aahReplication component. These errors and warnings can be ignored.
-	When importing AVEVA .NET Apps that use AVEVA Historian Client controls, some of the required files are not imported correctly. **Workaround:** 1. Import the App using the System Platform IDE by selecting **Import** from the **Galaxy** menu. 2. Select **AVEVA App**. 3. Select the folder containing the App to be imported. 4. The App is imported and all the dependent files are copied to a new folder, **C:\Program Files (x86)\ArchestrA\Framework\FileRepository\<GalaxyName>\ObjectFileStorage2\xxx**, where **xxx** is a 3-digit number. **Tip:** Sorting the **ObjectFileStorage2** folder by date modified can make it easier to locate the correct numbered folder. 5. Some of the required files for AVEVA Historian Client controls may not be copied. For each of the controls your imported App uses, locate the corresponding folder(s) from the following list, and copy all the files from those folders into the folder from step 4: • Query control: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientQueryControl.dll** • Tag Picker: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTagPicker.dll** • Time Range Picker: **C:\Program Files (x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTimeRangePicker.dll** • Trend control: **C:\Program Files**

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

	(x86)\ArchestrA\Framework\Bin\AVEVA_Software__LLC__aaHistClientTrendControl.dll

New features in Historian Client 2023 R2

Ad-Hoc expressions in Trend

Historian Client Desktop Trend control and its embedded ActiveX controls now support Ad-Hoc expressions. The Expression application and pane now appear in the Tag Picker with the Textbox editor.

For embedded ActiveX controls, scripting is supported for ad-hoc expressions in InTouch applications.

Ad-hoc expressions are supported with both SQL and REST connections.

Expanded ad-hoc expression functions

New functions added to ad hoc expressions make results simpler and more valuable.

Runtime language switching

Runtime language switching is supported for Trend Control, Query Control, Embedded control, and the Excel and Word ActiveX controls. It's supported for the following fields:

- Description
- EngineeringUnit
- Message On/Message Off
- Alias

If there is no value for the requested language for a field, the default language is used.

Improved Trend control for InTouch

Configuration is simplified for the Trend control within InTouch. The control is available in the graphic editor and can be placed in InTouch directly.

AVEVA Identity Manager Server login

Trend and Query support single sign-on authentication with AVEVA Identity Manager. This is restricted to REST-based connections and is not available for SQL-based connections.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

Additional script functions for OMI Trend App and Graphic Editor support

The event tags TagAdded and TagRemoved have been added and are available at the Graphic level and in the HistoricalTrendApp control.

General performance improvements

Performance has been improved across the Historian Client controls.

Microsoft Office Historian Client Workbook improvements

The classic Historian Client Workbook add-in has been updated. Note the following:

- Microsoft Office is optional. If you install Office after installing Historian Client, the Historian Client add-ins do not appear in the list of Office add-ins. Run the Historian Client installation program and repair the installation to load the Office add-ins.
- When you perform a custom installation of Microsoft Office (32-bit), you must install Office Shared Features and Office Tools so that the Historian add-ins for Workbook or Report can be loaded.

  Note: Make sure that you are not using the 64-bit version of Microsoft Office.

Tool for migrating from Historian Client Workbook to the newer Excel task pane add-in

The Excel Migration Tool updates Excel workbooks created in Historian Client Workbook to formulas compatible with the newer Historian Excel task pane add-in.

This tool is supported for files created in Excel 2016 or later.

For more details, see the *Migrate data from Historian Client Workbook to the Excel add-in* section in the *AVEVA Historian Client User Guide*.

Resolved issues in Historian Client 2023 R2

This release resolves the following issues:

Issue ID	Related Issues	SR/Case Number	Description
IMS-2868362		960348145	The Historian Client Trend MovingAverage retrieval style fails. The following warning message appears in the SMC log: "Error creating dataview: System.Data.SqlClient.Sql Exception (0x80131904): Incorrect syntax near the keyword 'AND..."
IMS-2829074		960338977	Trend Client refresh does

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

			not work as expected.
IMS-1434259		960200215 960274781 960277990 960273174	The Servers.xml file was encrypted, so couldn't be shared between user profiles. To address this issue, now only the password is encrypted for SQL authentication. The non-sensitive information is no longer encrypted.
Bug-2600099			If servers.xml is upgraded from a previous version, the Historian Client Trend control crashes.
IMS-2616467		960329790	Setting a tag to a moving average in Historian Trend causes all other tag trends to disappear.
IMS-2599739	IMS-2590504	960322009	Windows that open from built-in buttons on the Trend App are not brought to the front.
IMS-2842698	IMS-2635827	960324888	When the the properties of an engineering unit are changed, the changes don't propagate to retrieval until the Historian services are restarted.
IMS-2510162		960305816	Unable to retrieve discrete live or historical values from Historian Client Query.
IMS-2602879	IMS-1500586	960327042	OMI times out when opening a window with Historian Client Trend.
IMS-2412412		960276859	HistClient 2023 VBA function ActiveFactoryWorkbook.Addserver fails with the error `Runtime error 5, Invalid procedure`

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

			`call or argument.`
IMS-2361029		960254917	ShowGraphic function to open an embedded Historian Client trend in the application from a saved aatrend file is very slow.
IMS-2447731	IMS-2426902 IMS-2457568 IMS-2473685	960284737	aaTrend & aaQuery client tool buttons are purple.
IMS-2468523		960293562	When using the Historical Trend App, if more than one instance is opened at a time, the settings affect each other.
IMS-2111271		960223078	When Start Date, End Date, or Duration is changed, Update To Current Time is enabled.
IMS-2221595		960244148	Historian Client (Query & Trend) “File” menu item not visible if the Format in the Regional settings is set different from “English (United States)”
IMS-2200586		960237237	Historian Client Trend application always prompts for unnecessary save dialog for last added trend
IMS-2260942		960249515	Historian Client aaTrendControl does not accept the Bottom value entered for Value Axis Range
IMS-2201698		960239189	The trend duration is not properly applied to trends in a trend set
IMS-2308959		960256488	Trend control buttons disappear when the

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

			control is used within InTouch Access Anywhere
			When using the Trend application on a tablet PC, the Annotate Chart and Annotate Application options are missing from the Tools menu.
IMS-2333570		960264275	When using the aaHistClientActiveDataGrid ActiveX control in InTouch with a SQL login, the control fails to connect to the server.

© 2015-2024 AVEVA Group Limited and its subsidiaries. All rights reserved.

AVEVA Group plc
High Cross
Madingley Road
Cambridge
CB3 0HB
UK

Tel +44 (0)1223 556655

www.aveva.com
To find your local AVEVA office, visit **www.aveva.com/offices**

AVEVA believes the information in this publication is correct as of its publication date. As part of continued product development, such information is subject to change without prior notice and is related to the current software release. AVEVA is not responsible for any inadvertent errors. All product names mentioned are the trademarks of their respective holders.