

Wonderware® Historian Client Software User's Guide

Invensys Systems, Inc.

Revision E
Last Revision: 6/16/10



Copyright

© 2005-2006, 2010 Invensys Systems, Inc. All Rights Reserved.

All rights reserved. No part of this documentation shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Invensys Systems, Inc. No copyright or patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this documentation, the publisher and the author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein.

The information in this documentation is subject to change without notice and does not represent a commitment on the part of Invensys Systems, Inc. The software described in this documentation is furnished under a license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of these agreements.

Invensys Systems, Inc.
26561 Rancho Parkway South
Lake Forest, CA 92630 U.S.A.
(949) 727-3200

<http://www.wonderware.com>

For comments or suggestions about the product documentation, send an e-mail message to ProductDocumentationComments@invensys.com.

Trademarks

All terms mentioned in this documentation that are known to be trademarks or service marks have been appropriately capitalized. Invensys Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this documentation should not be regarded as affecting the validity of any trademark or service mark.

Alarm Logger, ActiveFactory, ArcestrA, Avantis, DBDump, DBLoad, DT Analyst, Factelligence, FactoryFocus, FactoryOffice, FactorySuite, FactorySuite A², InBatch, InControl, IndustrialRAD, IndustrialSQL Server, InTouch, MaintenanceSuite, MuniSuite, QI Analyst, SCADAAlarm, SCADASuite, SuiteLink, SuiteVoyager, WindowMaker, WindowViewer, Wonderware, Wonderware Factelligence, and Wonderware Logger are trademarks of Invensys plc, its subsidiaries and affiliates. All other brands may be trademarks of their respective owners.

Contents

	Welcome.....	17
	Documentation Conventions.....	17
	Technical Support	18
Chapter 1	Introduction	19
	About the Wonderware Historian Client Software	19
	Desktop Applications.....	20
	Microsoft Office Add-Ins	20
	ActiveX and .NET Controls.....	21
	About the Wonderware Historian	21
	Client/Server Architecture.....	22
	Integration with Wonderware Application Server	23
	Analyzing Process Data	26
Chapter 2	Common Client Components	27
	Server Connection Configuration	27
	Creating a New Server Connection	28
	Editing a Server Connection.....	30
	Reconnecting to a Server.....	31
	Removing a Server Connection.....	31
	Using HTTP as the Server Connection Protocol.....	32
	Considerations for VPN Access	38
	Status Bar	39
	Tag Picker	40

Servers Pane.....	41
Tags Pane	44
Filter Pane	44
Showing/Hiding the Tag Picker.....	46
Viewing the Arcestra Hierarchical Name	46
Tag Picker Views.....	47
Time Picker.....	47
Viewing Program and License Information.....	49

Chapter 3 Wonderware Historian Client Trend51

Getting Started with Trend	51
Working with Trend Files.....	52
Creating a New Trend.....	53
Configuring Default Settings for a Trend File	53
Opening an Existing Trend	54
Saving a Trend	54
Closing a Trend	55
Undoing/Redoing Actions.....	55
Configuring a Trend.....	56
Configuring a Trend to Use a Summary Tag.....	56
Working with Replicated Tags	57
Viewing Tag Definition Information	63
Viewing the Hierarchical Name in a Trend.....	64
Viewing Data in the Trend Chart	65
Refreshing the Trend Chart	66
Deleting a Tag	66
Configuring Trend Options for a Tag.....	67
Scrolling through Tags in a Trend	78
Highlighting a Tag	79
Stacking Traces	80
Showing Live Data	81
Showing Historical Data in “Replay” Mode	82
Scaling Tags.....	83
Panning in the Trend Chart	96
Using Axis Cursors.....	98
Zooming.....	99
Showing/Hiding the Chart Grid	100
Viewing Trend Data in a Table Format.....	101
Viewing the Data Log in a “Narrow” Format	101
Viewing the Data Log in a “Wide” Format	103
Viewing Statistics.....	105
Using Annotations.....	107

Adding an Annotation	108
Viewing the Annotation List.....	109
Editing an Annotation.....	110
Deleting an Annotation.....	111
Saving the Annotations List as a .CSV File.....	111
Printing Annotations.....	112
Trending Events	113
Using Absolute or Relative Times	114
Using Absolute Time	114
Using Relative Time.....	116
Switching Between Absolute and Relative Time: Example.....	117
Using Time Offsets to Compare Data	119
Configuring Trend Application Options.....	124
Configuring Retrieval Options.....	124
Configuring Color Options	126
Configuring Time Zone Options	128
Configuring Miscellaneous Options	130
Configuring Other Options	131
Configuring Trend File Properties	133
Configuring General Properties	133
Configuring Color Properties.....	134
Configuring Axis Properties	136
Configuring Limit Properties.....	138
Configuring Annotation Properties	140
Configuring Target Region Properties	141
Working with Scatter Plots.....	142
Viewing Data in a Scatter Plot.....	143
Scaling Tags in a Scatter Plot.....	145
How Are Value Pairs Matched?.....	146
Panning and Zooming in a Scatter Plot	147
Defining a Target Region for a Scatter Plot.....	148
Configuring Scatter Plot Properties	150
Other Considerations for Working with Scatter Plots	152
Outputting Trend Data	153
Printing Trend Data.....	153
Printing Trend Sets.....	153
Saving Trend Data to a .CSV File	156
Saving the Trend Chart to an Image File.....	157
E-mailing a Trend File.....	157
Copying a Trend Chart to the Windows Clipboard ...	158

Publishing Trends to the Wonderware Information Server.....	158
Publishing a Static Trend Report.....	159
Publishing a Dynamic Trend Report.....	160
Using Trend with a Tablet PC.....	161
Annotating a Chart	161
Making Chart Annotations.....	162
Selecting, Copying, and Deleting Chart Annotations	163
Saving, Printing, and E-Mailing an Annotated Chart	163
Importing .CRV Data	164

Chapter 4 Wonderware Historian Client Query 165

Getting Started with Query.....	165
Query Toolbar.....	167
Columns Pane.....	167
Results Pane	167
Status Bar.....	172
Working with Query Files.....	172
Opening an Existing Query File.....	172
Saving a Query File.....	172
Creating a Query	173
Query Types.....	175
Query Type: Aggregate Values.....	176
Query Type: Alarm History	178
Query Type: Alarm Limits.....	181
Query Type: Analog Summary Values.....	182
Query Type: Annotations.....	185
Query Type: Custom	186
Query Type: Event History Values	186
Query Type: Event Snapshot.....	188
Query Type: Favorites.....	190
Query Type: History Values	191
Query Type: IO Server	198
Query Type: Live Values.....	199
Query Type: Number of Tags.....	200
Query Type: Server Version.....	201
Query Type: State Summary Values.....	201
Query Type: Storage.....	204
Query Type: Storage Size Available.....	206
Query Type: Storage Start Date.....	206
Query Type: Summary Values.....	206
Query Type: Tag Details	208

Query Type: Tag Search.....	210
Query Type: Time Running	212
Common Tabs for Query Types	212
Chapter 5 Wonderware Historian Client Workbook... 217	
Getting Started.....	217
Managing Server Connections.....	218
Opening an Existing Workbook File	219
Manually Loading/Unloading the Add-In.....	220
Creating a Report: Overview	222
Working with Functions, Formulas, and Cells.....	223
Refreshing a Function or Array Formula	224
Editing a Function.....	224
Converting a Function to Values.....	224
Refreshing a Sheet	225
Converting a Sheet to Values	225
Manually Inserting a Function.....	225
Manually Editing a Function.....	228
Copying a Function	230
Selecting Cells	231
Verifying the Date/Time Format in Microsoft Excel ..	231
Selecting Tags for Reports	232
Selecting Analog, Discrete, String, Summary, or Event Tags	233
Selecting Summary Tags	235
Finding a Source Tag or Replicated Tag.....	236
Selecting Event Snapshot Tags	238
Retrieving Tag Configuration Information.....	239
Retrieving Configuration Details for a Tag	239
Retrieving Analog Tag Alarm Limits	245
Retrieving Tag Values.....	247
Retrieving Live Values.....	248
Retrieving History Values.....	251
Retrieving Aggregate Values	259
Retrieving Values for Summarized Tags	264
Retrieving Values for Event Snapshot Tags.....	268
Common Properties for Tag Values.....	271
Analyzing Tag Data.....	277
Analog Tag Analysis.....	278
Batch Analysis.....	286
Scatter Analysis.....	290
Discrete Tag Analysis.....	293

Analog Values at Discrete Transition Analysis.....	298
Analog/Discrete Pair Analysis.....	302
Creating a Direct Query.....	306
Configuring Workbook Options.....	309
Configuring Global Formatting Options.....	309
Referencing Formatting Options in a Query.....	312
Using a Named Range for Formatting Options.....	314
Changing Formatting Options in Named Range.....	315
Configuring Time Zone Options.....	316
Configuring Data Source Options.....	318
Configuring General Options.....	319
Setting the Base Date and Base Time Parameters....	319
Using "Binding" Tags to a Query at Run Time.....	321
Time Options for Queries.....	325
Publishing Reports.....	326
Publishing a Static Workbook Report.....	327
Publishing a Dynamic Workbook Report.....	328
Wonderware Historian Client Workbook Function Reference.....	330
Function Arguments.....	340
Error Messages for Functions.....	356
Migrating History Data Retrieval Functions.....	356
Viewing the Wonderware Historian Details.....	358

Chapter 6 Wonderware Historian Client Report 359

About Add-ins and Templates.....	359
Getting Started.....	360
Manually Loading/Unloading the Add-In.....	363
Managing Server Connections.....	366
About Field Codes.....	366
Opening an Existing Report Document.....	369
Running a Report Document.....	369
Saving Report Documents.....	370
Saving a Report Document.....	371
Saving a Configured Report Document as a Report Template.....	371
Saving a Run Report Document as an HTML File....	373
Inserting a SQL Query.....	375
Editing a Query.....	378
Using Date and Time Options.....	380
Inserting Date and Time Field Codes.....	381

About Date and Time Wildcards	382
Inserting Date and Time Wildcards	384
Configuring Report Options.....	386
Chapter 7 Introduction to Controls and Objects	389
About the Wonderware Historian Client Controls and Objects	390
About Properties, Methods, and Events.....	391
Getting Started with the Controls.....	391
Using the Controls in Different Environments	392
Using the Controls within InTouch HMI Software	393
Using the Controls in Microsoft Office.....	393
Mapping for Numerical Data Types	393
Chapter 8 aaHistClientTrend Control	395
Using aaHistClientTrend at Runtime.....	395
Using aaHistClientTrend in an Application	395
Adding aaHistClientTrend to an InTouch Window....	396
aaHistClientTrend Properties	397
aaHistClientTrend Methods	462
aaHistClientTrend Events.....	494
aaHistClientTrend Enumerations.....	496
aaChartType Enumeration.....	497
aaDashStyle Enumeration.....	497
aaDataPointLabelingType Enumeration.....	497
aaDateModeEnumeration Enumeration.....	497
aaInterpolationType Enumeration.....	498
aaQualityRules Enumeration.....	498
aaRetrievalMode Enumeration	499
aaRetrievalVersion Enumeration.....	499
aaStateCalculation Enumeration.....	500
aaTargetRegionExcursionType Enumeration	501
aaTimeStampRules Enumeration	501
aaTraceGradientType Enumeration	501
aaTrendGradientType Enumeration.....	502
aaTrendType Enumeration.....	502
aaTrendValueFormat Enumeration.....	502
aaUpdateToCurrentTimeState Enumeration.....	503

	aaValueAxisLabelEnumeration Enumeration	503
	aaHistClientTrend Unsupported Objects	503
	Using aaHistClientTrend in a Multi-Monitor Environment.....	504
Chapter 9	aaHistClientQuery Control	505
	Using aaHistClientQuery at Runtime	505
	Using aaHistClientQuery in an Application.....	506
	Adding aaHistClientQuery to an InTouch Window ...	506
	aaHistClientQuery Properties.....	507
	aaHistClientQuery Methods.....	515
	aaHistClientQuery Events	523
	aaQueryTypeEnumeration	524
Chapter 10	aaHistClientTagPicker Control	527
	Using aaHistClientTagPicker at Runtime.....	527
	Using aaHistClientTagPicker in an Application.....	527
	Adding aaHistClientTagPicker to an InTouch Window	528
	aaHistClientTagPicker Properties	529
	aaHistClientTagPicker Methods	534
	aaHistClientTagPicker Events.....	537
	aaHistClientTagPickerSplitterOrientation Enumeration 539	
Chapter 11	aaHistClientTimeRangePicker Control....	541
	Using aaHistClientTimeRangePicker at Runtime.....	541
	Using aaHistClientTimeRangePicker in an Application.....	541
	Adding aaHistClientTimeRangePicker to an InTouch Window	542
	aaHistClientTimeRangePicker Properties	543
	aaHistClientTimeRangePicker Methods	547
	aaHistClientTimeRangePicker Events	550
Chapter 12	aaHistClientActiveDataGrid Control	551
	Using aaHistClientActiveDataGrid at Runtime	552
	Using aaHistClientActiveDataGrid in an Application..	557
	Adding aaHistClientActiveDataGrid to an InTouch Window	557
	aaHistClientActiveDataGrid Properties	558

aaHistClientActiveDataGrid Methods	567
aaHistClientActiveDataGrid Events.....	573
Script Examples for aaHistClientActiveDataGrid	575
aaHistClientActiveDataGrid Error Messages	578
Chapter 13 Server Objects	579
aaServer Object	579
aaServer Properties.....	579
aaServer Methods.....	586
aaServers Object.....	587
aaServers Properties	587
aaServers Methods	588
aaServers Events.....	591
Instantiating an aaServers Object	593
aaServerListChangeArgs Object	593
Properties.....	593
aaServerStateChangeArgs Object.....	594
Properties.....	594
aaServerState Enumeration	595
aaServerType Enumeration.....	596
Chapter 14 aaHistClientSingleValueEntry Control	597
Using the aaHistClientSingleValueEntry Control at Runtime	597
Adding a Tag Value.....	598
Using the aaHistClientSingleValueEntry Control in an Application	598
Adding the aaHistClientSingleValueEntry Control to an InTouch Window.....	599
aaHistClientSingleValueEntry Control Properties....	600
aaHistClientSingleValueEntry Control Methods.....	616
aaHistClientSingleValueEntry Control Events	621
aaFieldLabelPositionEnumeration Enumeration	623
aaUseTimeZoneEnumeration Enumeration.....	623
Chapter 15 aaTag Object	625
Using aaTag in an Application	625
aaTag Properties	625

Chapter 16	aaHistClientWorkbookRunner and aaHistClientReportRunner Objects.....	631
	aaHistClientWorkbookRunner Object	631
	aaHistClientWorkbookRunner Object Properties	631
	aaHistClientWorkbookRunner Methods.....	633
	aaHistClientReportRunner Object.....	640
	aaHistClientReportRunner Object Properties.....	641
	aaHistClientReportRunner Object Methods.....	642
Chapter 17	Workbook and Report Automation Objects.....	645
	Wonderware Historian Client Workbook Object	645
	Wonderware Historian Client Workbook Object Methods	645
	Wonderware Historian Client Workbook Automation Example.....	652
	Wonderware Historian Client Report Object.....	656
	Report Object Properties.....	656
	Report Object Methods.....	657
Chapter 18	aaHistClientGlobalFunctions Object	659
	Using aaHistClientGlobalFunctions Object in an Application.....	659
	aaHistClientGlobalFunctions Methods.....	659
Chapter 19	Common Properties, Methods, Events, Enums, and Data Types	663
	Common Properties	663
	BackColor.....	664
	BackColorStyle	664
	BorderStyle.....	664
	CausesValidation	665
	Container	665
	ContextMenuEnabled	665
	DataBindings.....	665
	DragIcon	666
	DragMode	666
	Enabled	666
	Font	666
	ForeColor	667
	Height	667

HelpContextID.....	667
Index.....	667
Left	668
Name	668
Object	668
Parent.....	668
TabIndex	669
TabStop	669
Tag.....	669
ToolTipText.....	669
Top.....	670
Transparent	670
Visible.....	670
WhatsThisHelpID.....	670
Width.....	671
Common Methods.....	671
Drag.....	671
Move	671
SetFocus	671
ShowWhatsThis.....	672
ZOrder	672
Common Events.....	672
Click	673
DblClick	673
DragDrop	673
DragOver.....	673
GotFocus	673
KeyDown.....	673
KeyPress	674
KeyUp	674
LostFocus	674
MouseDown.....	674
MouseMove	674
MouseUp	674
Validate.....	675
Common Enumerations	675
aaRetrievalSource Enumeration	675
aaTagType Enumeration	675
aaTimeRangeEnumeration Enumeration.....	676
Common Data Types	678
DateTime	678
Color	678

DataSet	679
Font	679
Object	679
Appendix A Configuring an IIS Virtual Directory for SQL Server	681
Appendix B Data Retrieval Options.....	689
Understanding Retrieval Modes.....	689
Cyclic Retrieval	690
Delta Retrieval	693
Full Retrieval.....	700
Interpolated Retrieval.....	701
“Best Fit” Retrieval	707
Average Retrieval.....	712
Minimum Retrieval	719
Maximum Retrieval	725
Integral Retrieval.....	731
Slope Retrieval	733
Counter Retrieval.....	737
ValueState Retrieval.....	741
RoundTrip Retrieval	748
Understanding Retrieval Options	752
Which Options Apply to Which Retrieval Modes?.....	753
Using Retrieval Options in a Transact-SQL Statement	754
Cycle Count (X Values over Equal Time Intervals) (wwCycleCount)	755
Resolution (Values Spaced Every X ms) (wwResolution).....	757
About “Phantom” Cycles	760
Time Deadband (wwTimeDeadband).....	762
Value Deadband (wwValueDeadband).....	766
History Version (wwVersion).....	769
Interpolation Type (wwInterpolationType)	771
Time stamp Rule (wwTimestampRule).....	774
Time Zone (wwTimeZone).....	777
Quality Rule (wwQualityRule)	778
State Calculation (wwStateCalc).....	786

Analog Value Filtering (wwFilter)	788
Selecting Values for Analog Summary Tags (wwValueSelector)	794
Edge Detection for Events (wwEdgeDetection)	797
Appendix C Retrieval Styles for Trend	809
Working with Retrieval Styles.....	809
Location and Structure of Retrieval Styles.....	810
Creating and Editing Retrieval Styles	812
Retrieval Style XML Elements	813
Using the Standard Retrieval Styles.....	818
Retrieval Styles, Application Settings, and Tag Settings	819
Appendix D Control and Object Migration Reference .	821
Wonderware Historian Client Version Comparison: Overview	822
Migrating an ActiveFactory 8.5 Application.....	824
Migrating Documents to Wonderware Historian Client 10.0 SP1	826
iTrend and aaHistClientTrend Comparison.....	826
iQuery and aaHistClientQuery Control Comparison	831
TimeBar and aaHistClientTimeRangePicker Comparison.....	832
TagPicker and aaHistClientTagPicker Comparison	834
SingleValueEntry and aaHistClientSingleValueEntry Comparison.....	836
ActiveDataGrid and aaHistClientActiveDataGrid Comparison.....	837
Server and aaServer Object Comparison.....	837
Servers and aaServers Object Comparison.....	839
Tag and aaTag Object Comparison	840
Tags Object	841
GlobalFunctions Object and aaHistClientGlobalFunctions Object Comparison	841
Glossary.....	843
Index	855

Welcome

You can use the Wonderware Historian Client software to retrieve data from a Wonderware Historian. Before you can use the Wonderware Historian Client software, the Wonderware 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 Wonderware 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.

Documentation Conventions

This documentation uses the following conventions:

Convention	Used for
Initial Capitals	Paths and filenames.
Bold	Menus, commands, dialog box names, and dialog box options.
Monospace	Code samples and display text.

Technical Support

Wonderware Technical Support offers a variety of support options to answer any questions on Wonderware products and their implementation.

Before you contact Technical Support, refer to the relevant section(s) in this documentation for a possible solution to the problem. If you need to contact technical support for help, have the following information ready:

- The type and version of the operating system you are using.
- Details of how to recreate the problem.
- The exact wording of the error messages you saw.
- Any relevant output listing from the Log Viewer or any other diagnostic applications.
- Details of what you did to try to solve the problem(s) and your results.
- If known, the Wonderware Technical Support case number assigned to your problem, if this is an ongoing problem.

Chapter 1

Introduction

The Wonderware 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 Wonderware Historian Client Software

Whether you are an operator, process engineer, or manager, the Wonderware 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.

The Wonderware Historian Client software is a full-featured suite of applications that maximize the value of the data in the Wonderware Historian. The Wonderware Historian Client software integrates tightly with the most popular Microsoft Office tools. With the Wonderware 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 Wonderware Historian database.
- Visualize the current process state.
- Produce rich automated reports.

Desktop Applications

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

- Wonderware 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 Chapter 3, Wonderware Historian Client Trend.

- Wonderware Historian Client **Query**. This point-and-click tool enables complex queries to be created and executed against any Wonderware Historian. Knowledge of the database structure or SQL is not required.

For more information, see Chapter 4, Wonderware Historian Client Query.

Microsoft Office Add-Ins

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

For more information, see Chapter 5, Wonderware Historian Client Workbook.

- Wonderware Historian Client **Report**. This add-in to Microsoft Word allows sophisticated reporting from a Wonderware Historian using the Word document format (.doc).

For more information, see Chapter 6, Wonderware Historian Client Report.

ActiveX and .NET Controls

aaHistClientTrend and aaHistClientQuery are controls that provide essential functionality of Wonderware Historian Client Trend and Wonderware Historian Client Query for use in container applications, such as InTouch® HMI software and Internet Explorer. You can also use Wonderware 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 on page 389.

About the Wonderware Historian

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

In the Wonderware 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.

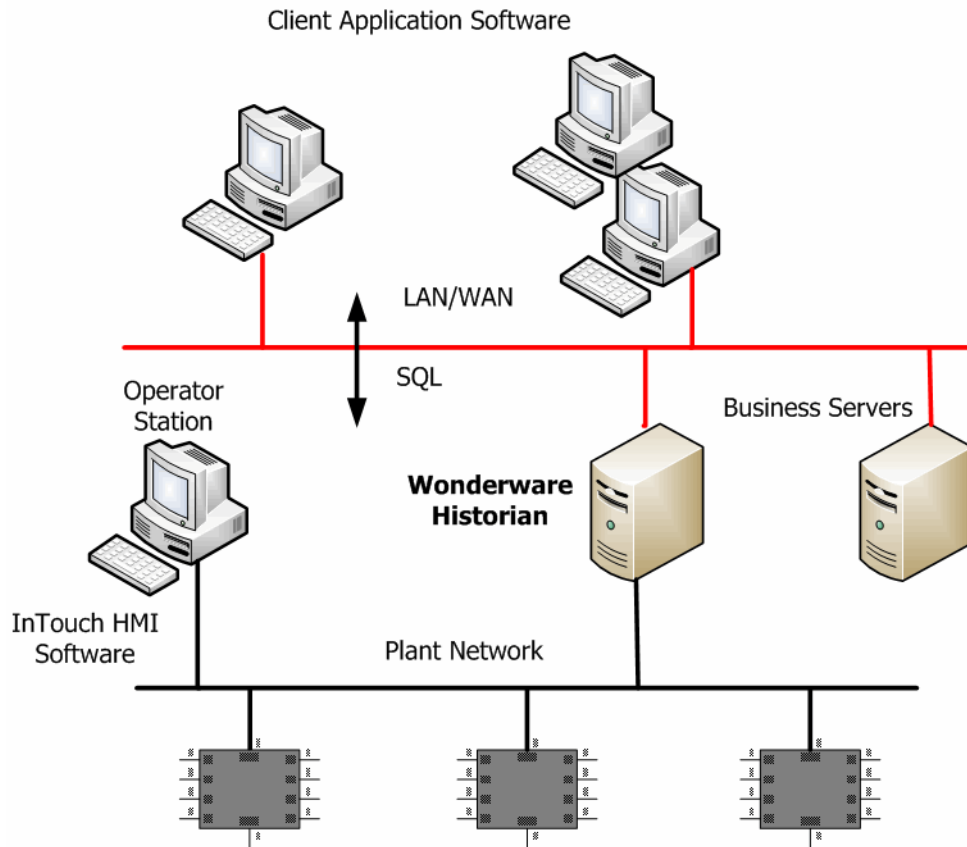
Some examples of database queries possible with the historian are:

- 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 Wonderware 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 Wonderware Historian is used as a link between the process network and the business LAN/WAN:



InControl Software, Data Acquisition, PLCs, DCSs, Process Computers, Instruments

Integration with Wonderware Application Server

The Wonderware Application Server software is ArchestrA-based and provides a unified environment for development, deployment, and maintenance of the distributed automation objects.

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

You can use the ArcestrA IDE to replicate the ArcestrA Model View to a Wonderware Historian. Galaxies and objects are represented as groups and attributes are represented as tags in the Wonderware Historian.

ArcestrA Naming Conventions

The following are the different names to refer to objects in ArcestrA:

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

`<TagName>.<AttributeName>`

For example, Reactor_002.ReactLevel

- **Contained name** is 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 ArcestrA 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** is 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 ArchestraA IDE, the hierarchical name is shown within brackets to the right of the object's tag name. For example:

Reactor_002 [R32.Reactor]

Object Viewer

Attribute Reference: Reactor_002.ReactLevel.value

Attribute Name	Value	Timestamp
AlarmMode	Enable	
AlarmModeCmd	Enable	
AlarmInhibit	false	
InAlarm	false	
ConfigVersion	2	
ContainedName	Reactor	
ExecutionRelatedObject		
ExecutionRelativeOrder	None	
HierarchicalName	R32.Reactor	
Auto	false	8/6/2009 12:43
Ref_Done	true	8/6/2009 12:43
SteamValve_InMaintenance	false	8/6/2009 12:43
TransferPump.Input.ReadStatus	Operationpen...	

AttributeReference	Value	Timestamp	Quality	Status
Reactor_002.ReactLevel	1300.0	8/12/2009 4:48:20.700 ...	C0:Good	Ok

Watch List 1

FILE: User: DefaultUser Mode: User

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

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.

Chapter 2

Common Client Components

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

Server Connection Configuration

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

You can use either your Windows user account (integrated security) or a valid SQL Server user account, depending on how the Wonderware Historian is configured. You can also access the server using HTTP. For more information on HTTP access, see *Using HTTP as the Server Connection Protocol* on page 32.

Ask your administrator what type of user account you must use to access the server. If you are accessing the server over HTTP, the administrator must provide you with the URL and virtual directory name, as well.

Server connections are shared among the Wonderware 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.

When you start a Wonderware 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:

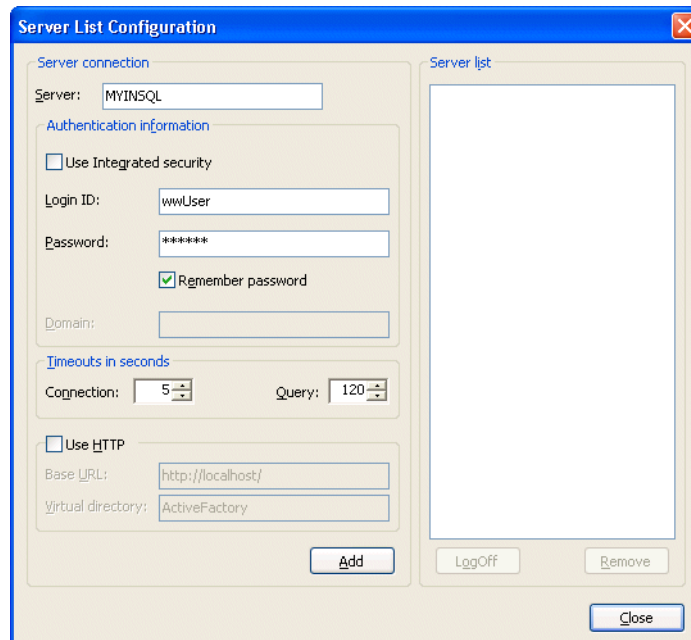
- Open a file that causes data to be retrieved for a tag on that server.
- Expand a server in the Tag Picker to view its Tag List.
- Manually log on to the server.

Creating a New Server Connection

You can create a new server configuration. To create a new connection, you need your assigned Wonderware Historian username and password.

To create a new server connection

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



The **Server list** window shows a list of connected servers or servers that were configured in the past.

- 2 In the **Server** box, type the name of the server to which you want to connect.
- 3 Make sure that you know what options to choose to access the server. For more information, see **Server Connection Configuration** on page 27.

- 4 To log on to the server using integrated security, select the **Use Integrated security** check box and then go to step 8.
- 5 To log on to the server using SQL Server credentials, clear the **Use Integrated security** check box and configure the following login details. Then, go to step 8.
 - **Login ID:** Type your assigned Wonderware 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 Wonderware 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 To log on to the server over HTTP using integrated security, select both the **Use Integrated security** check box and the **Use HTTP** check box and then configure the required options. Then, go to step 9.
 - **Login ID:** Type your Windows user name.
 - **Password:** Type your Windows password. Select the **Remember password** check box to specify for the system to remember your password.
 - **Domain:** Type the domain name in which your Windows account is validated. The domain name is only valid for HTTP connections to the Wonderware Historian when integrated security is used.
 - **Base URL:** The URL for the server.
 - **Virtual directory:** The name of the virtual directory on the server.

For more information on HTTP access, see Using HTTP as the Server Connection Protocol on page 32.

- 7 To log on to the server over HTTP using SQL Server authentication, clear the **Use Integrated security** check box, select the **Use HTTP** check box, and then configure the required options. Then, go to step 9.
 - **Login ID:** Type your assigned Wonderware 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 Wonderware 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.
 - **Base URL:** The URL for the server.
 - **Virtual directory:** The name of the virtual directory on the server.
- 8 In the **Timeouts in seconds** area, configure the time allocated for the database connection and the query execution.
 - **Connection:** The connection timeout in seconds. Valid values are 1 to 600.
 - **Query:** The query timeout in seconds. Valid values are 1 to 600.
 - 9 Click **Add**.
 - 10 Click **LogOn** to log on to the server.
 - 11 Repeat Steps 2 through 10 to add additional servers.
 - 12 Click **Close**. An error message appears if a connection cannot be made to a 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](#) on page 28.
- 4 Click **Update**.
- 5 Click **Close**.

An error message appears if a connection cannot be made to the server, but the server is added to the list.

Reconnecting to a Server

When the trend is running in live mode and a server is disconnected, the Trend application 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.

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:\WINNT or 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.

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

Using HTTP as the Server Connection Protocol

The Wonderware Historian Client software can use HTTP as the transport protocol, instead of other typical network protocols, such as TCP. HTTP (which is layered on top on TCP) usually uses port 80 and is generally open in Internet (or intranet) firewalls, so that communications can take place between the servers and the clients.

If the Wonderware Historian and the Wonderware Historian Client applications (aaTrend.exe, aaQuery.exe, aaHistClientTrend, aaHistClientQuery, and so on) are on two different sides of a firewall and TCP is the protocol, port 1433 must be opened in the firewall software/appliance or else communications cannot take place between the server and clients.

Thus, the HTTP feature allows clients to access historians located behind firewalls without requiring additional ports to be opened.

The Microsoft SQLXML technology is used to provide HTTP transport to the SQL Server, as well as XML access to the data. If you enable SQL Server access over HTTP, be aware of the limitations of this technology. For more information, see the following web resource:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsql/ac_xml1_59m4.asp

Guidelines for Accessing SQL Server over HTTP

Accessing SQL Server over HTTP provides the most benefit if you want to view data from a pre-configured client file over the Internet (or intranet). For example, you might want to open a pre-configured trend file over the Internet/intranet and view the data.

Accessing SQL Server over HTTP is not recommended if you are configuring a client file, such building and saving a trend file. The retrieval of tag information over HTTP can be very slow if there is a large number of tags in the historian database (for example, over 5,000, depending on the system).

Software Requirements for HTTP Access

To access SQL Server over HTTP, the following software must be installed and configured:

- Microsoft Internet Information Services (IIS) Server
- Microsoft SQLXML

For the required versions of the software, see the Wonderware Historian Client ReadMe file.

No software or special configuration is required on the client, other than what is generally required to support the Wonderware Historian Client installation.

Authentication for HTTP Access

The authentication mode is determined by the settings for IIS Server. The IIS Server allows for three modes:

- **No authentication.** For this mode, the SQLXML configuration in the IIS Server always uses a specified user name and password; no user name and password is required to be provided by the client. Either integrated security or SQL server authentication can be used.
- **Integrated security.** The IIS Server must be set up to use integrated security to authenticate clients. From the client application, you must choose **Integrated security** in the **Server Configuration** dialog box, and provide a user name, password, and a domain for authentication.
- **Basic authentication.** For this mode, the IIS Server passes on a user name and password supplied by the client to the SQL Server. This mode is not recommended unless you also use Secured Sockets Layer (SSL) security, which encrypts all the data going over the connection.

The integrated security and basic authentication modes require that the client provide a user account, which is passed by the IIS Server to the SQL Server.

Note that the configuration of the IIS Server governs the mode of authentication that is required of clients. For example, if the IIS Server is configured to use integrated security, then client users must access the server using integrated security.

There is no security in the access mechanism. Using HTTPS provides an additional level of security, but may impact performance.

Executing SQL Statements over HTTP

Whereas TCP is a connection-oriented protocol, and is stateful, HTTP is stateless. That is, a request is made from the client to the server, and the server then responds to the request (provided that no errors occur). The entire transaction is stateless; no state is preserved from one request to another. A request is made, and a response is received, and no state is retained.

Although HTTP is stateless, the concept of a "logged in" state to the server has been preserved to be consistent. Therefore, a client can be considered to be "logged on" to or "logged off" of an HTTP-accessed server.

Stateless SQL statements that execute in a native SQL Server access environment execute when SQL Server is accessed over HTTP. However, stateful SQL statements do not function unless they are part of a collection of statements submitted together.

If HTTP access is used, and no data rows are returned for a query, no column headers are produced.

SQL Cursors

The stateless nature of SQL Server access over HTTP affects the use of SQL cursors.

In a native SQL Server access environment, a connection is made, a cursor is created, and used subsequently until the connection is terminated. The establishment of the connection, creation of the cursor, use of the cursor, and disconnection can all be performed as separate requests.

If the SQL Server is accessed HTTP, the establishment of the connection, the execution of any SQL statements that follow, and the disconnection must all be part of a HTTP request.

Batch Statements

The stateless nature of SQL Server access over HTTP affects the use of "batch" SQL statements. For HTTP access, the last SQL statement submitted in a batch must be capable of returning a result set.

For example, the following query does not return a result set over HTTP. This is because the last statement executed does not produce a result set.

```
SET QUOTED_IDENTIFIER OFF
Select ContactName FROM Customers
SET QUOTED_IDENTIFIER ON
```

To address this problem, make sure that the last statement produces a result set. For the query example, the last statement was removed to produce the expected results:

```
SET QUOTED_IDENTIFIER OFF
```

```
Select ContactName FROM Customers
```

To get the same effect as the original batch of statements, you must modify the query so that the results are stored into a temporary table before the SET QUOTED_IDENTIFIER ON statement occurs and then return those results. For example:

```
CREATE TABLE #temp (Name NVARCHAR(255))
```

```
SET QUOTED_IDENTIFIER OFF
```

```
INSERT INTO #temp SELECT ContactName FROM Customers
```

```
SET QUOTED_IDENTIFIER ON
```

```
SELECT Name FROM #temp
```

If you have any batch queries in which the last SQL statement in the batch does not return a result set, you must restructure the query.

Column Aliases

For SQL Server access over HTTP, you must use include column aliases in SQL statements when no columns exist.

For example, when used with native SQL Server access, the following query yields the counts of all tags in the database.

```
SELECT COUNT(*) FROM Tag
```

However, the same statement does not produce results in SQL over HTTP. You need to modify the query to use column aliases. For example:

```
SELECT COUNT(*) as n FROM Tag
```

Another example is:

```
SELECT n=COUNT(*) FROM Tag
```

SQL Statements in the Trend and Query Applications

The queries used by the Wonderware Historian Client Trend and Query applications execute properly if SQL Server access over HTTP is used. However, if you execute a custom query using the Query application, be aware that not all queries that execute in the native SQL access environment execute in the SQL over HTTP environment without modifications. Those that require multiple requests in a native SQL environment (such as those using cursors) do not execute at all. For more information, see Executing SQL Statements over HTTP on page 34.

Also, if you are using the Tag Picker to search for tags, the response time can be quite slow if you are accessing a large database. This is due to the additional overhead required to access the data with SQL statements in the SQL over HTTP environment.

Error Reporting

If you are accessing SQL Server over HTTP, query-related errors are not provided. This is a limitation of the SQLXML technology. The only error condition that can be deduced is when no data (not even the table headers) is returned for a request. For this condition, a generic error is returned that prompts you to check the SQL query.

If your query is not valid, no specific error message appears, and no data is returned.

Reliability and Performance

Accessing SQL Server over HTTP is as reliable as the physical connection on which the data exchange occurs. As you can expect that a busy network may cause HTTP requests to fail or be intermittent, for the same busy network, SQL over HTTP exhibits those symptoms as well.

Important The guidelines provided here may not be achievable for your particular environment, given the wide range of factors related to network bandwidth and availability and the nature of your Wonderware Historian Client applications.

In general, equivalent operations take three to five times longer if you use SQL over HTTP, as compared to native SQL Server access. Also, it can be considerably slower if you are accessing the server from an external system over the Internet, depending on the Internet connection method. For example, if you are using a dial-up connection.

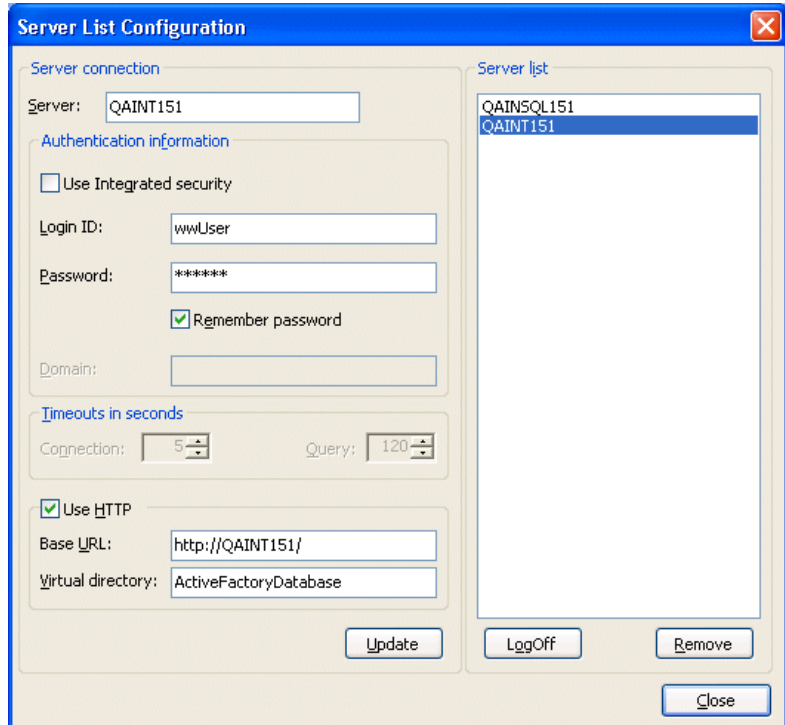
The primary use for HTTP access is for visualization of data. For this type of use, you can expect acceptable performance when updating at refresh rates of one refresh per second. Even over the Internet, visualization at rates of one time every second may be achieved during off-peak hours, and rates of one time every five seconds may result in smooth operation of your applications.

Databases that contain about 5000 tags can generally be browsed without problems using the Tag Picker, or when used in constructing trends or queries. However, larger databases are likely to slow down navigation in the Tag Picker when used across the Internet.

You may also experience slowed performance if the IIS Server and the SQL Server are not running on the same computer.

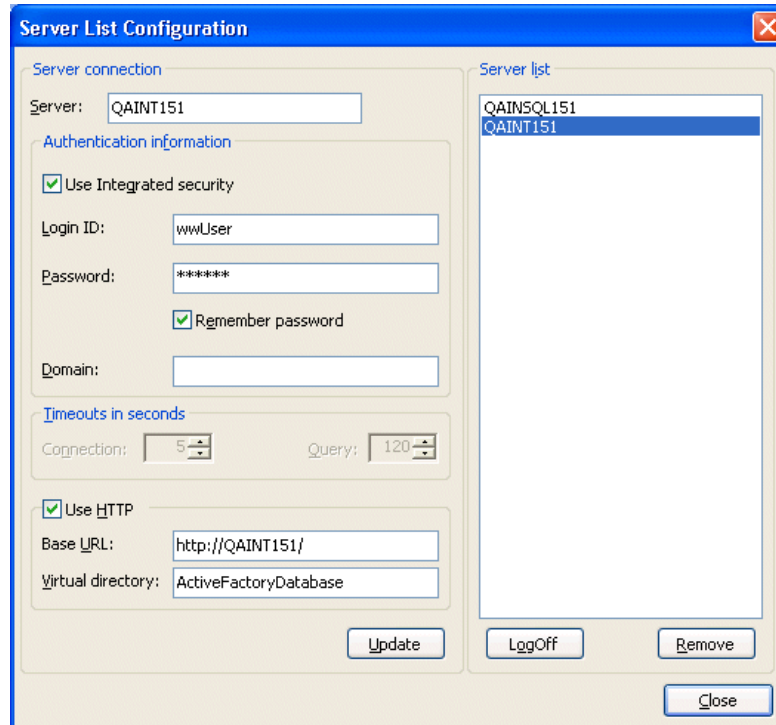
Example Server Configuration for HTTP Access

The following illustration shows how to configure a server to use HTTP for the database access protocol. For this example, the "ActiveFactoryDatabase" virtual directory was created during the installation of the SQLXML software. The local host is a computer named QAIN151.



For more information on configuring a server, see [Creating a New Server Connection](#) on page 28.

The following illustration shows the dialog box configured for integrated security. Note that even though integrated security is used, the login ID and the password are enabled. This is because this refers to the integrated security on the IIS Server.



Considerations for VPN Access

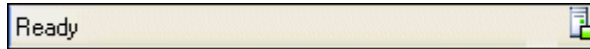
Check with your system administrator regarding how to access the server over a virtual private network (VPN).

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

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

Status Bar

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



To show the status bar

- On the **View** menu, click **Status Bar** so that a check mark appears.

To hide the status bar

- On the **View** menu, click **Status Bar** so that no check mark appears.

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

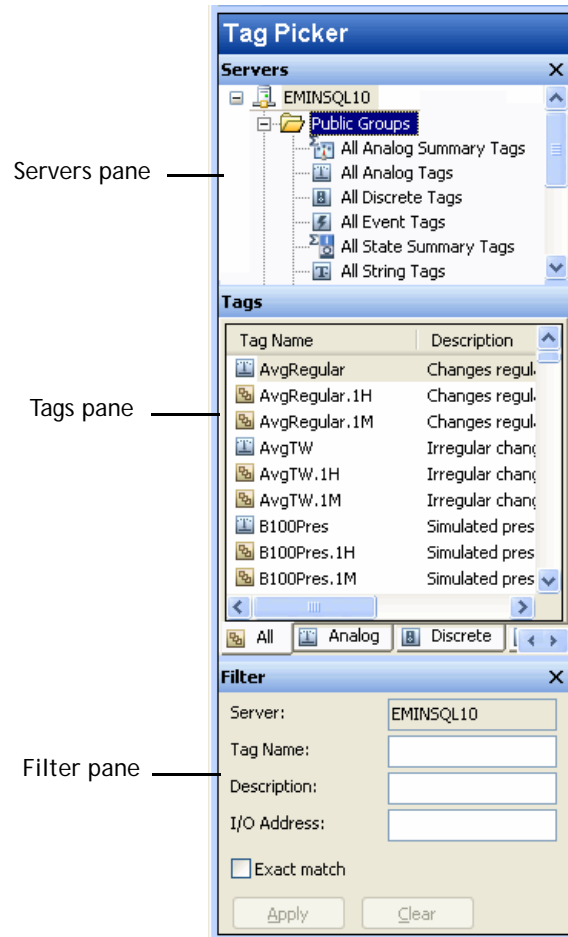
Color	Status Meaning
Gray	No servers are configured.
Green	Connections (log ons) have been established to all servers in the server configuration list.
Yellow	Connections (log ons) have been established to most of the servers in the list.
Orange	No connections (log ons) have been established to most of the servers in the list.
Red	No connections (log ons) have been established to any of the servers in the list.

Double-click the icon to access the **Server List Configuration** dialog box. For more information, see **Server Connection Configuration** on page 27.

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 and/or for tags that match a particular search pattern. You can then select the ones you want to include for the client application or control.



The Tag Picker is comprised of the following three panes:

- Servers Pane
- Tags Pane
- Filter Pane

Servers Pane

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



The Servers pane shows the following items:

Category	Description
Servers	All objects that make up the basic Wonderware 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.

Showing/Hiding the Servers Pane

To show the Servers pane

- Right-click in the Servers pane and then click Servers pane so that a check mark appears.

To hide the Servers pane

- Do one of the following:
 - Right-click in the Servers pane and then click Servers pane so that no check mark appears.
 - Click the Close button.

Editing Groups

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

Adding a Group

You can add a group.

To add a group

- 1 Right-click on the folder under which you want to create a group and then click **New Group**.

A new folder appears in the Tag Picker.

- 2 Type a name for the folder and press ENTER.

Adding a Tag to a Group

When you add tags to a new group, the original reference still appears in the default system group. Any tag can belong to any number of groups, and any group can contain any number of tags.

To add a tag to a group

- 1 Select the system group folder that contains the tag that you want to add to your new group.
- 2 In the **Tags** pane, select the tag to add.
- 3 Do any of the following:
 - Drag the tag that you want from the **Tags** pane into the folder.
 - Use the **Copy** and **Paste** commands on the **Edit** menu to copy the tag to the target folder.
 - Right-click on the selected tag in the **Tags** pane. Use the **Copy** and **Paste** commands in the shortcut menu to copy the tag to the target folder.

Deleting a Group or Tag Reference

When you delete a private group or tag reference in a private group, the group folder, any subfolders that the group folder may contain, and all references to tags are deleted. The tags themselves are not deleted, and the original references still appear in the default system group. You cannot delete public folders or the tag references contained in them.

To delete a group or tag

- 1 Select the group or tag in the pane.
- 2 Do one of the following:
 - Right-click on the group or tag and then click **Delete**.
 - Press the **DELETE** key.

Renaming a Group

You can rename a group that you have created in the Tag Picker. However, you cannot rename a public folder.

To rename a group

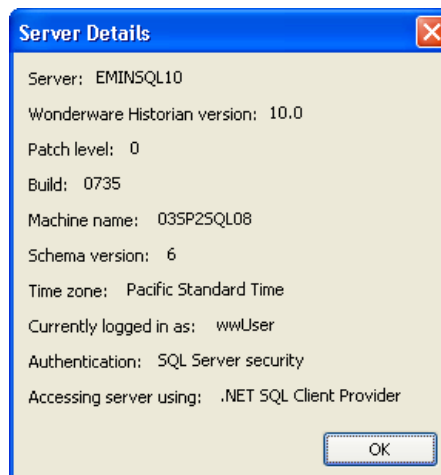
- 1 Select the group in the pane.
- 2 Do one of the following:
 - Right-click on the group and then click **Rename**.
 - Press the **F2** key.
- 3 Type a new name for the group and press **ENTER**.

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.



- 2 Click **OK**.

Tags Pane

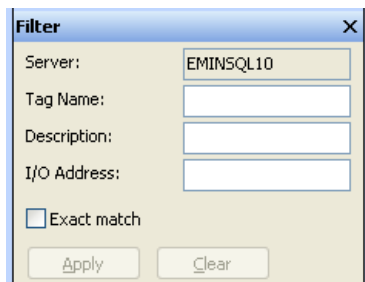
The **Tags** pane shows all the tags for the currently selected group in the **Servers** pane.



- To select multiple tags in 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 reduce the tags listed in the **Tags** pane according to criteria that you specify. You can filter the tags according to name, description, and I/O address.



The filter mechanism allows for the following "wildcard" characters as part of the filter criteria:

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: <ul style="list-style-type: none"> • [a-f] • [abcdef]
[^]	Any single character not within the specified range or set. For example: <ul style="list-style-type: none"> • [^a - f] • [^abcdef]

For example, to find all tagname ending with "level," type "%level." Filter criteria are not case-sensitive.

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 Wonderware Historian.

To apply a filter

- 1 In the **Server** box, specify or verify the server.
This box is not available if the **Servers** pane is visible.
- 2 In the **Tag name** box, type the string to match for the tagname.
- 3 In the **Description** box, type the string to match for the description.
- 4 In the **I/O Address** box, type the string to match for the I/O address.
- 5 Select the **Exact match** check box to search for tags that exactly match the entire string that you provided for the tagname and/or description options.
For example, if you specify "level" as the tagname and do not select **Exact match**, any tagname that contains the string "level" appears. For example, "ReactLevel," "ProdLevel," and "\$AccessLevel."
- 6 Click **Apply** to apply the filter criteria.
- 7 Click **Clear** to clear the **Filter** pane.

Showing/Hiding the Tag Picker

To show the Tag Picker

- ◆ Do one of the following:
 - On the **View** menu, click **Tag Picker** so that a check mark appears.
 - Click the **Show Tag Picker** toolbar button so that it is highlighted.



To hide the Tag Picker

- ◆ Do one of the following:
 - On the **View** menu, click **Tag Picker** so that no check mark appears.
 - Click the **Show Tag Picker** toolbar button so that it is not highlighted.



Viewing the Arcestra Hierarchical Name

A hierarchical name represents the fully qualified name of a contained object within Arcestra. For more information on hierarchical names, see *Integration with Wonderware Application Server* on page 23.

To view a hierarchical name

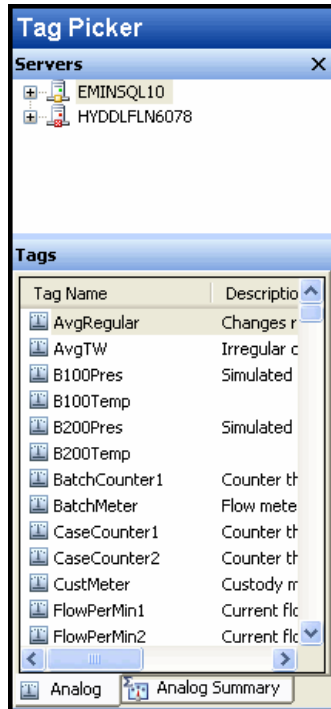
- ◆ Right-click in the Tag Picker and click **Use hierarchical name**.

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

Hierarchical Name	Description
AlarmUnAckedCntTotal	The Area represent:
IAS_Platform.CPULoadAvg	The Platform repres
IAS_Platform.RAMAvailableAvg	The Platform repres
R31.Reactor.ReactLevel	This is the Reactors
R31.Reactor.ReactTemp	This is the Reactors
SysConfigStatus	System Configuratic
SysCritErrCnt	Total Critical errors :
SysDataAcq0BadValues	Bad quality points p
SysDataAcq0OutsideRealtime	Discarded points per

Tag Picker Views

By default, all 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.

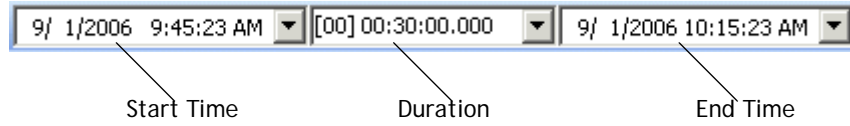
Time Picker

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 list. To select a predefined duration, click the down arrow on the duration list.



When you change one of the options, one of the other options is recalculated automatically. While you change the option, a blue frame appears around the option that will be recalculated as a result of the change.

The relation between changed and updated options 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 options in a row, which option is updated depends on which two other options 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.

Viewing Program and License Information

For each Wonderware 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**.

Chapter 3

Wonderware Historian Client Trend

Wonderware Historian Client Trend is a client application that allows you to query tags from a Wonderware 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.

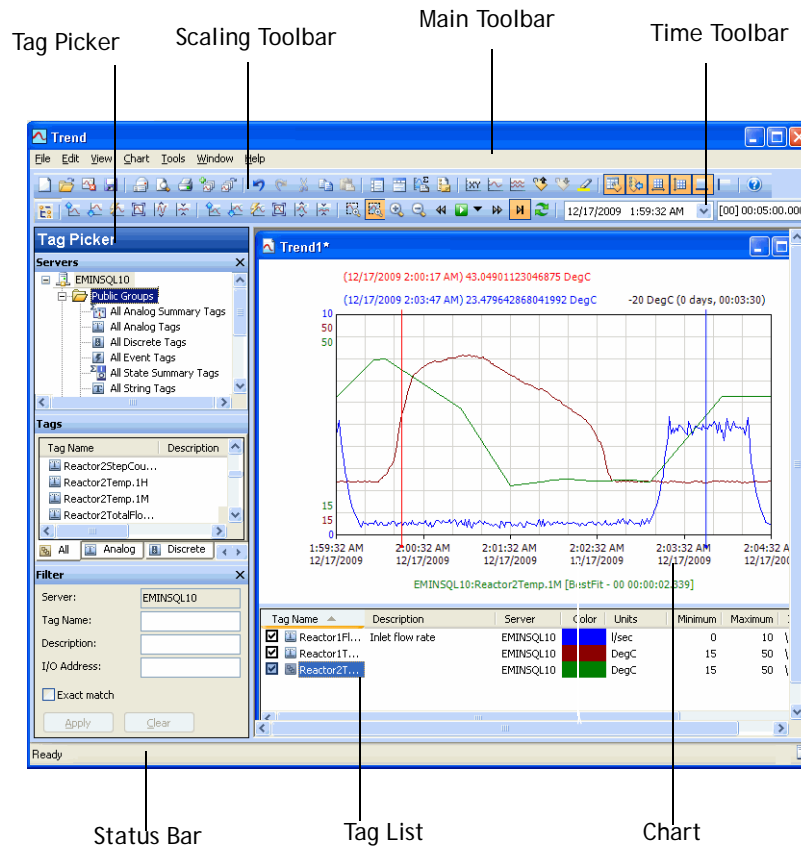
Before Trend can be used to query tag information from the database, the server must be running and you must have security access.

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 on page 142.

Getting Started with Trend

When you start the Trend application for the first time, you are immediately prompted to connect to a 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 in. For more information, see Server Connection Configuration on page 27.

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



For information on using the tag picker and the time picker, see Chapter 2, Common Client Components. To show or hide toolbars or components, use the corresponding commands on the View menu.

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.

For information on migrating trend files from previous versions of the Wonderware Historian Client software, see [Importing .CRV Data on page 164](#).

Creating a New Trend

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

To create a new trend

- ◆ Do one of the following:
 - On the File menu, click **New**.



- Click the **New Trend** toolbar button.

To configure the new trend, see [Configuring a Trend on page 56](#). By default, the new trend is in standard trend mode. To create an XY scatter plot, see [Viewing Data in a Scatter Plot on page 143](#).

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 a folder depending upon your operating system.

If you are using Windows 2003 or XP, the Default.aaTrend file is located in the following folder:

```
C:\Documents and Settings\\Local  
Settings\Application  
Data\Wonderware\ActiveFactory\Trend\
```

If you are using Windows Vista, Windows 2008, or Windows 7, the Default.aaTrend file is located in the following folder:

```
C:\Users\\AppData\Local\Wonderware\  
ActiveFactory\Trend
```

To configure the default settings

- 1 Create a new trend file. For more information, see [Creating a New Trend](#) on page 53.
- 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 Do one of the following:
 - On the **File** menu, click **Open**.
 - Click the **Open Trend** toolbar button.



The standard Windows **Open** dialog box appears.

- 2 Browse to and select the trend file to open. All trend files have the `.aaTrend` extension.
- 3 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

- ◆ On the **File** menu, point to **Recent Files**, and then click the name of the file to open.

Saving a Trend

To save a trend

- 1 Do one of the following:
 - On the **File** menu, click **Save**.
 - Click the **Save Trend** toolbar 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.

- 2 In the **Save As** dialog box, type a name for the trend. All trend files have the `.aaTrend` extension.
- 3 Click **OK**.

To save a trend as another name

- 1 On the **File** menu, click **Save As**.
The standard Windows **Save As** dialog box appears.
- 2 In the **Save As** dialog box, type a name for the trend. All trend files have the **.aaTrend** extension.
- 3 Click **OK**.

Closing a Trend

To close a trend

- ◆ Do one of the following:
 - On the **File** menu, click **Close**.
 - Click the **Close Trend** toolbar 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

- ◆ Do one of the following:
 - On the **Edit** menu, click **Undo**.
 - Click the **Undo** toolbar button.



To redo an action

- ◆ Do one of the following:
 - On the **Edit** menu, click **Redo**.
 - Click the **Redo** toolbar button.



Configuring 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 Wonderware 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.

To configure a trend

- 1 Use the Tag Picker to find tags in the Wonderware Historian database(s) that you want to include in your trend. For more information on the Tag Picker, see Tag Picker on page 40.
- 2 Drag tags from the Tag Picker to the Tag List. You can add as many tags as your system resources allow.
- 3 Select the time period for the query using the time picker. For more information, see Time Picker on page 47.

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 on page 63.

You can configure trend options for a summary tag. For more information, see [Configuring Trend Options for a Tag](#) on page 67.

For more information on retrieving summary tags, see [Configuring Retrieval Options for a Tag](#) on page 75 or [Configuring Retrieval Options](#) on page 124.

Working with Replicated Tags

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

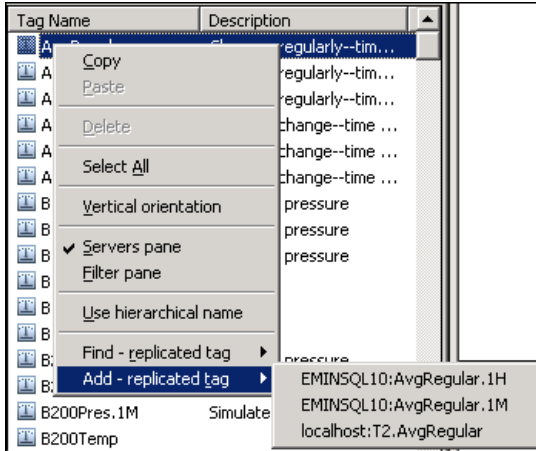
Adding a Source Tag or Replicated Tag

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, point to **Add - replicated tag**, and then click 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:

- You are connected to the IndustrialSQL Server 9.0.2
- 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.

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:

- You are connected to the IndustrialSQL Server 9.0.2
- 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 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.

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:
 - 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:

- You are connected to the IndustrialSQL Server 9.0.2
- 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 a 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.

To view tag information

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

	Description	Number	Server	Color	Units	Minimum	Maximum	IO Address	Time Offset	Source Tag	Source Server	Value at X1	Value at X2
lowIn.1H	Inlet flow r...	1	EMINSQ...	Blue	l/sec	-1	2	\\EMINSQ10\InSQL...	0:00:00.000	Reactor1FlowIn	EMINSQ10	1	1
emp.1M		2	EMINSQ...	Red	DegC	-1	2	\\EMINSQ10\InSQL...	0:00:00.000	Reactor1Temp	EMINSQ10	24	37
emp.1M		3	EMINSQ...	Green	DegC	-1	2	\\EMINSQ10\InSQL...	0:00:00.000	Reactor2Temp	EMINSQ10	38	28

The grid shows the following information:

- **Tag Name** The name of the tag within the Wonderware Historian. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
- **Description** The description for the tag.
- **Number** The serial number of the tag.
- **Server** The name of the server that contains the tag.
- **Color** The line color of the tag.
- **Units** The unit of measure of the tag value. Examples: seconds, psi, and lbs.
- **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.
- 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.

Viewing the Hierarchical Name in a Trend

A hierarchical name represents the fully qualified name of a contained object within ArcestrA. For more information, see *Integration with Wonderware Application Server* on page 23.

To view hierarchical names in a trend

- ◆ Do one of the following:
 - On the **View** menu, click **Use Hierarchical Name**.
 - Click the **Use hierarchical name** toolbar button.
 - Right-click in the Tag Picker and click **Use hierarchical name**.

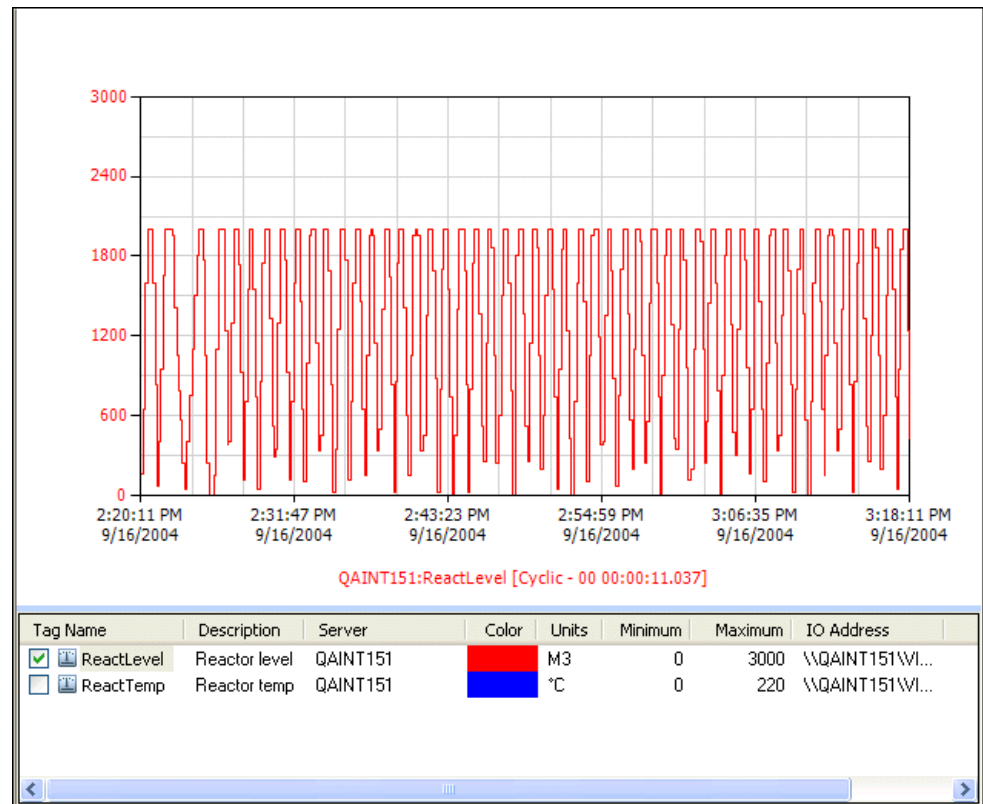


The Trend application shows the hierarchical name instead of the tag name. For example, the Tag List, Data Log dialog box, and the Trend chart area show hierarchical names.

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.



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:
 - On the **Chart** menu, click **Refresh**.
 -  • Click the **Refresh** toolbar button.
 - Press F5.

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

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.

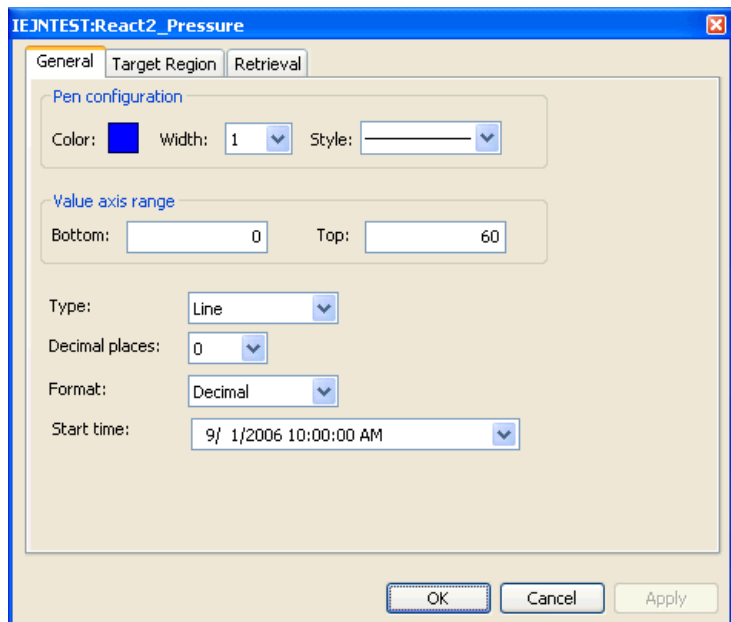
Configuring Display Options

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.
 - **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**.

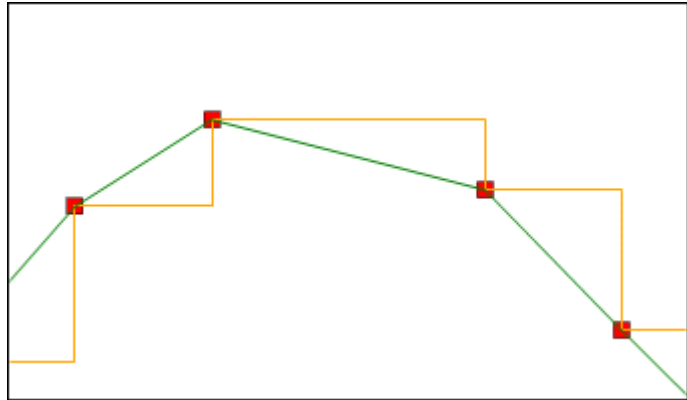
A line curve is best suited for continuously-changing analog data. A step-line curve is best suited for discrete data and for analog data that is not continuous. By default, the line curve trend is selected for the summary tags.

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

- For tags retrieved from a version 9.0 or a later Wonderware 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 Wonderware 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 Wonderware 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.

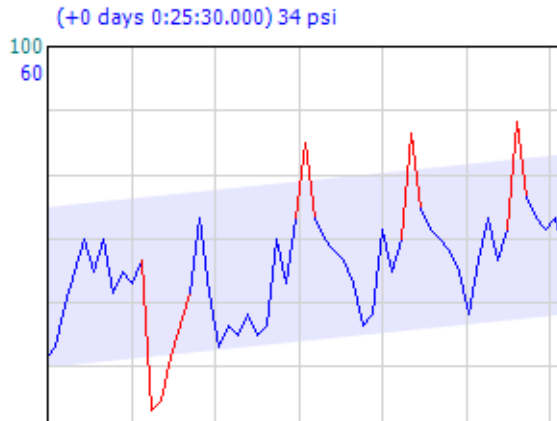


- 5 From the **Decimal places** list, select the number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags. Valid values are 0 through 15.
- 6 From the **Format** list, select how the values for the tag appear, either in decimal format or scientific format.
- 7 Configure either the time offset or start time for the data. For more information on time offsets, see *Using Absolute or Relative Times* on page 114.
 - **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* on page 118.
 - **Start time** Shown only for relative mode. The starting time stamp for the tag data in the chart.
- 8 Click **OK**.

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:



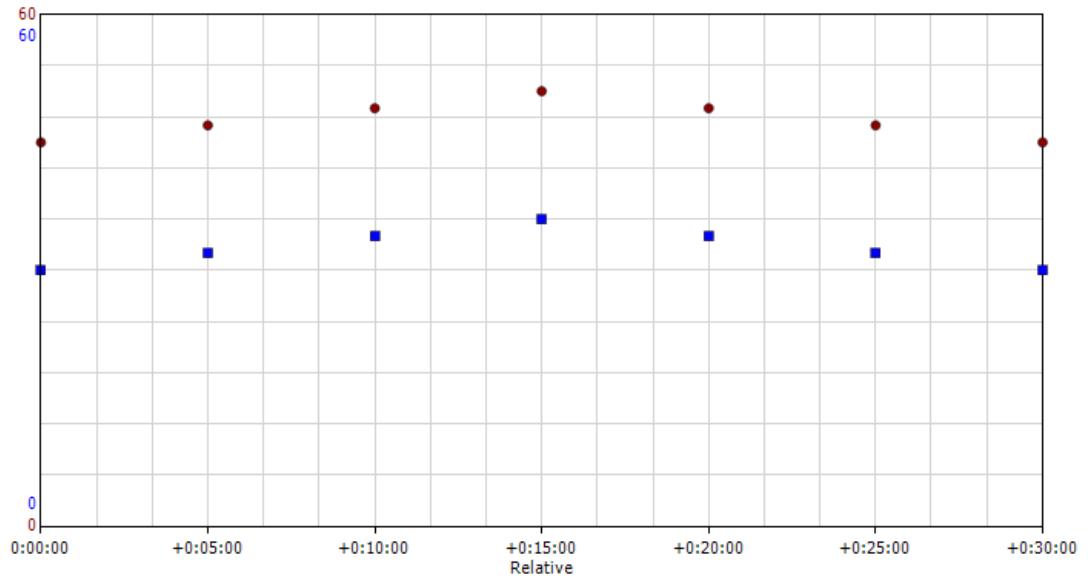
In a regular trend, you can only use target regions in relative time mode. A target region is defined by “region items,” that is, pairs of high and low values at specific time offsets.

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.

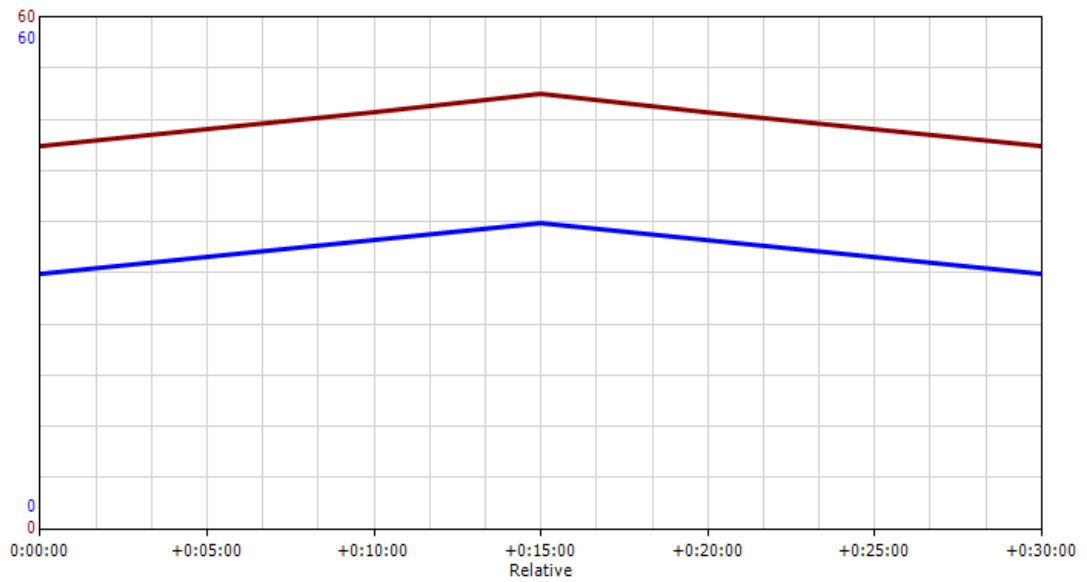
For example, assume that you define the following region items:

Time	Low	High
0:00:00.000	30	45
+0:05:00.000	32	47
+0:10:00.000	34	49
+0:15:00.000	36	51
+0:20:00.000	34	49
+0:25:00.000	32	47
+0:30:00.000	30	45

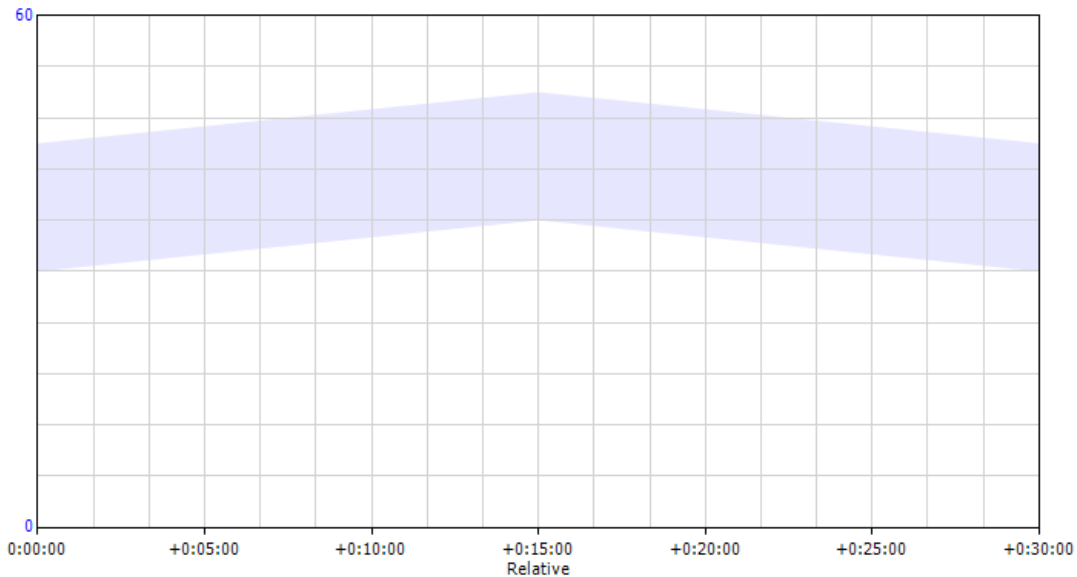
On a trend chart in relative time mode, these points look like this:



The connecting boundaries look like this:



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.

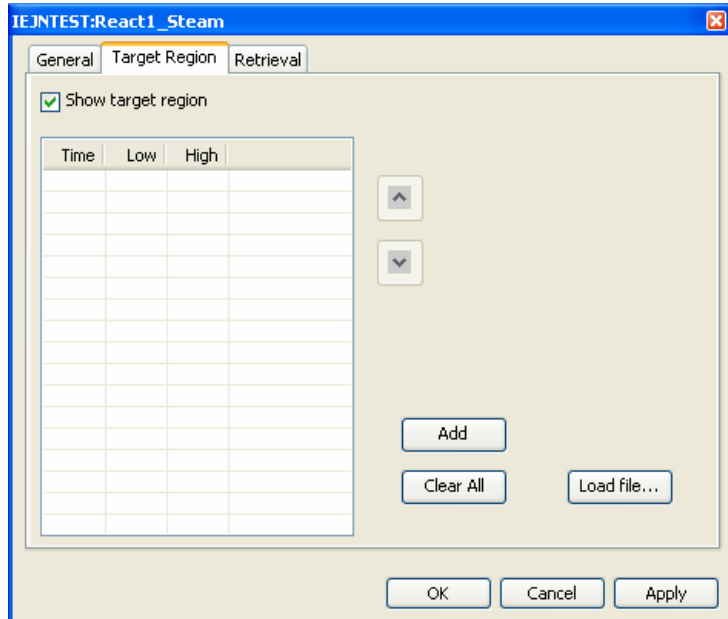
The following section explains how to set up a target region for a tag. To specify whether and how values outside the target region should be highlighted, and to set the target region's opacity, see [Configuring Target Region Properties](#) on page 141.

To configure a target region for a trend 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 Target Region tab.



- 3 Edit region items by using any of the following procedures:
 - To edit region items manually
 - To load region items from a CSV file
 - To paste region items in CSV format from another application
 - To paste region items from another tag
 - To 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* on page 118.
- 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.

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.

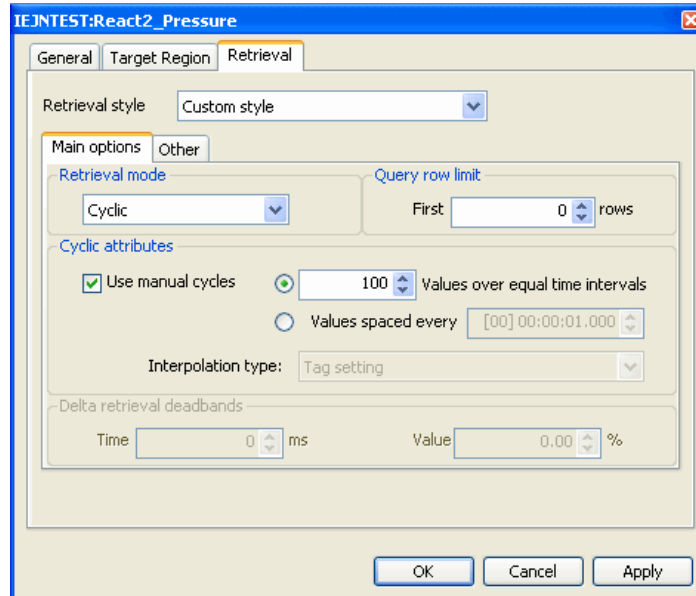
Most of the retrieval settings that you configure here only apply if you are retrieving data from a Wonderware Historian with a version of 9.0 or later.

If you are using an earlier Wonderware Historian version, see *Configuring Other Options* on page 131 and *Working with Retrieval Styles* on page 809 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* on page 809.
- To use custom retrieval settings, click **Custom style** in the **Retrieval style** list.



- 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* on page 809. 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* on page 752.

5 Specify any additional settings required on the **Other** tab.

The screenshot shows a dialog box with two tabs: 'Main options' and 'Other'. The 'Other' tab is active. It is divided into four sections: 'History version', 'Rules', 'State retrieval', and 'Transformation'.
 - **History version:** Two radio buttons, 'Latest' (selected) and 'Original'.
 - **Rules:** Two dropdown menus. 'Time stamp' is set to 'Server default' and 'Values to include in calculations' is also set to 'Server default'.
 - **State retrieval:** A dropdown menu for 'State calculation' is set to 'Percent', and a text field for 'State' is empty.
 - **Transformation:** A dropdown menu is set to 'No Transformation'.

- 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) on page 769.
- b In the **Rules** area, do the following:
 - In the **Timestamp** list, click the required timestamp value. For more information on the Time stamp rule, see Time stamp Rule (wwTimestampRule) on page 774.
 - 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 Wonderware Historian level.

For more information on each option, see Quality Rule (wwQualityRule) on page 778.

- c In the **State retrieval** area, do the following:
 - In the **State calculation** list, click the state calculation.
 - 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) on page 786.

- d 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) on page 788.

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.

To change the currently selected tag

- ◆ Do one of the following:
 - On the **Chart** menu, click either **Next Tag** or **Previous Tag** to “scroll” to the tag that you want.
 - Click the **Previous Tag** or **Next Tag** toolbar buttons to “scroll” to the tag that you want.

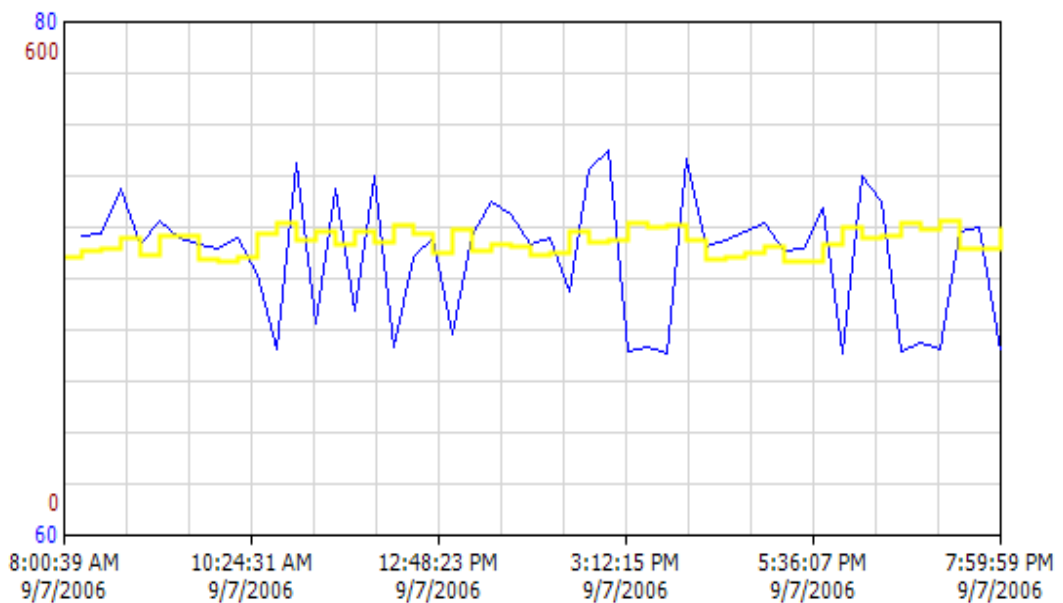


Highlighting a Tag

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 Do one of the following:
 - On the **Chart** menu, click **Highlight Tag** so that a check mark appears.
 - Click the **Highlight Tag** toolbar button so that it is highlighted.
 - The tag is highlighted in the chart.



Showing 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 single tag in the trend

- ◆ Do one of the following:
 - On the **Chart** menu, click **Single Tag Mode** so that a check mark appears.
 - Click the **Single Tag** toolbar button so that it is highlighted.



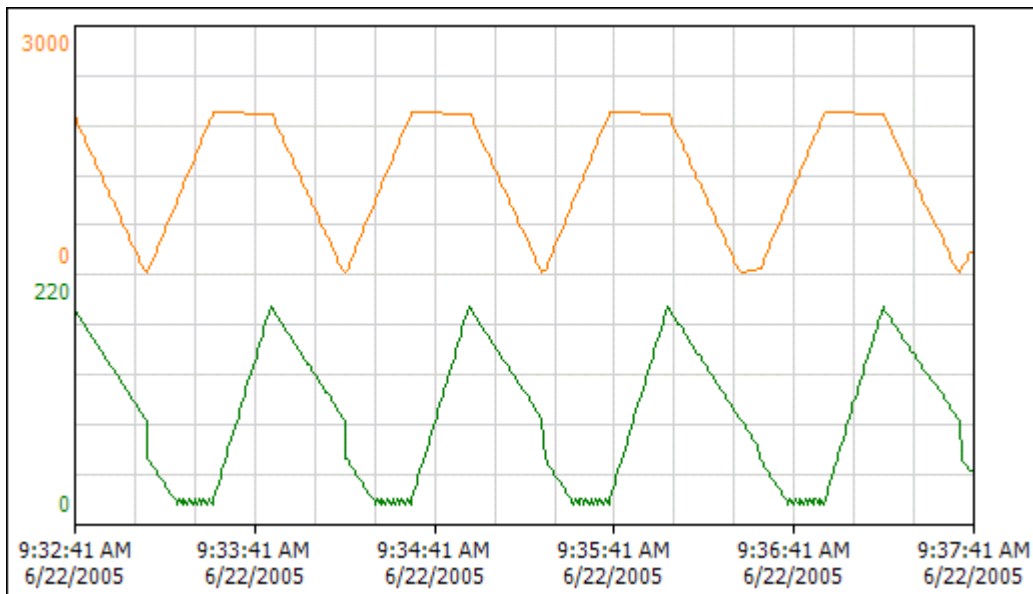
To view multiple tags again in the chart, follow the same procedure so that no check mark or highlighting appears.

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

- ◆ Do one of the following:
 - On the **Chart** menu, click **Stacked Traces**.
 - Click the **Stacked Traces** toolbar button.



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](#) on page 133.

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.

Note When retrieving live data, the time stamp rule for data retrieval is forced to "End." For more information on this setting, see [Time stamp Rule \(wwTimestampRule\)](#) on page 774.

To display "live" data

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.

3 Do one of the following:

- On the **Chart** menu, click **Live Mode** so that a check mark appears.



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

To display "static" data

◆ Do one of the following:

- On the **Chart** menu, click **Live Mode** so that no check mark appears.





- 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.


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

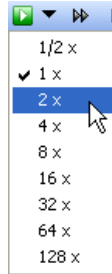
- 1 Configure a query with a time period whose end time is earlier than the current time.
-  2 Make sure the **Update to Current Time** toolbar button is *not* highlighted. If it is highlighted, click it so that it isn’t highlighted anymore, or click **Update to Current Time** on the **Chart** menu.
- 3 Do one of the following:
 - On the **Chart** menu, click **Live Mode** so that a check mark appears.
 -  • 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, or click **Update to Current Time** on the **Chart** menu.

- 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](#) on page 81.

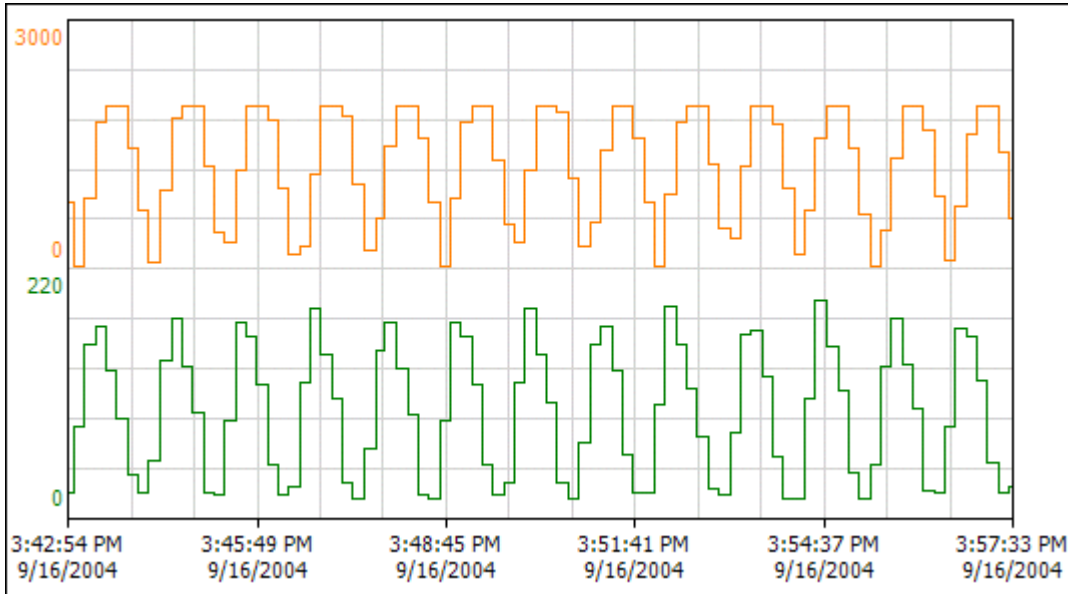
Scaling Tags

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.



The initial scale of a tag is determined by its Min/Max EU settings in the Wonderware 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* on page 67.
- Use the scaling commands to adjust the scale of single tag or all tags. For more information, see *Scaling Tags Up or Down* on page 90 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.

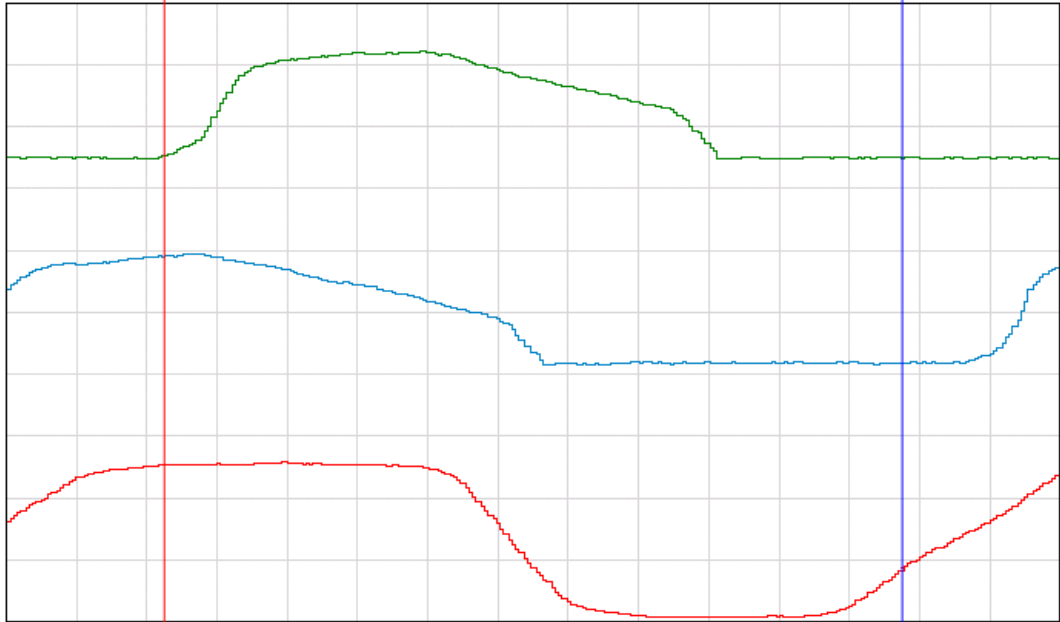
Showing No Scales on the Value Axis

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

- ◆ On the **View** menu, click **No Scales**.

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.

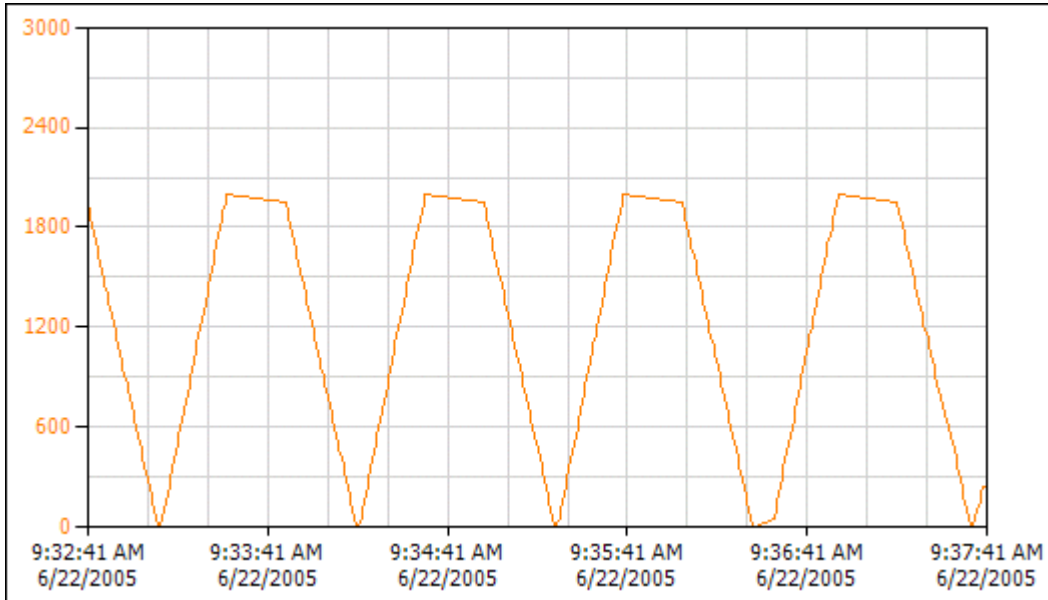
If the chart includes multiple tags, the scale of the currently selected tag in the chart is shown along the value axis. The scale label color matches the pen color of the selected tag. As you scroll through tags in the chart, the value axis always shows the scale of the selected tag. The labels along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

To show single scale

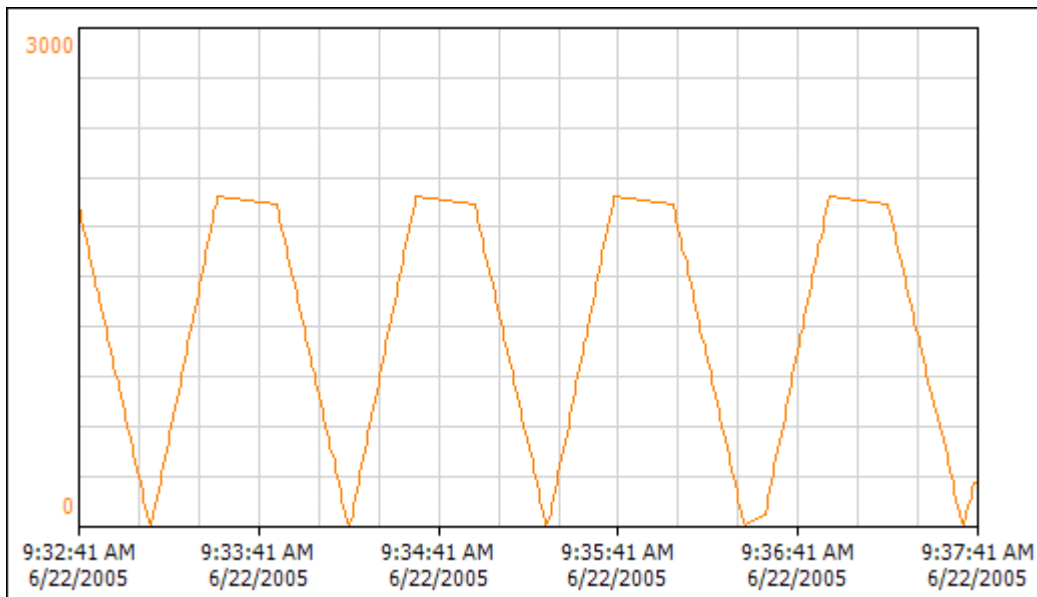
- ◆ On the **View** menu, click **Single Scale**.

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.

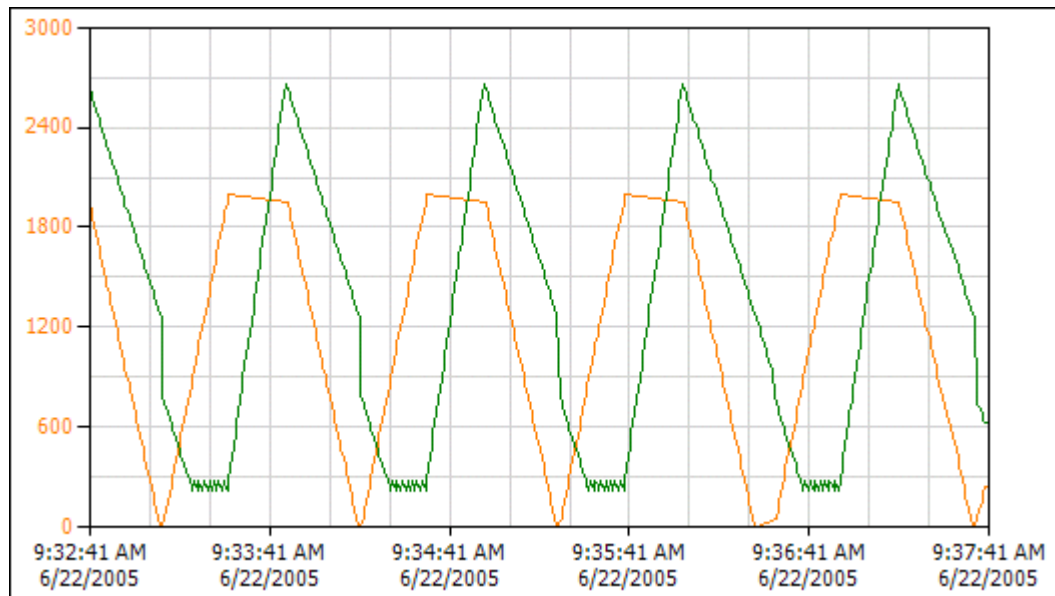
The following chart includes single tag and is configured for single scale. Stacked mode is not applied.



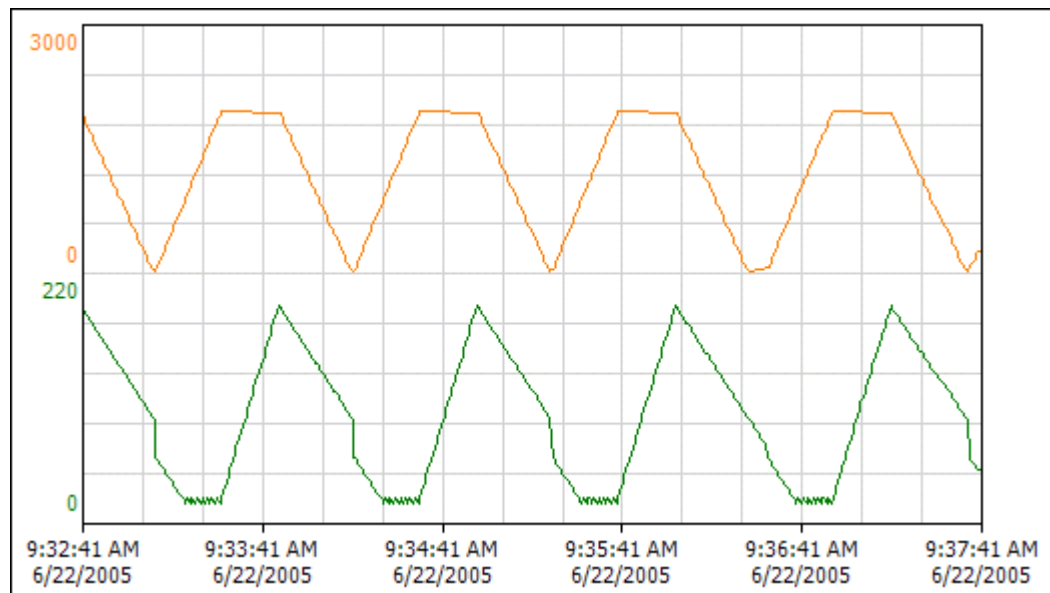
The following chart includes single tag and is configured for single scale. Stacked mode is applied. Only the minimum and maximum values are shown.



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.



Showing Multiple Scales on the Value Axis

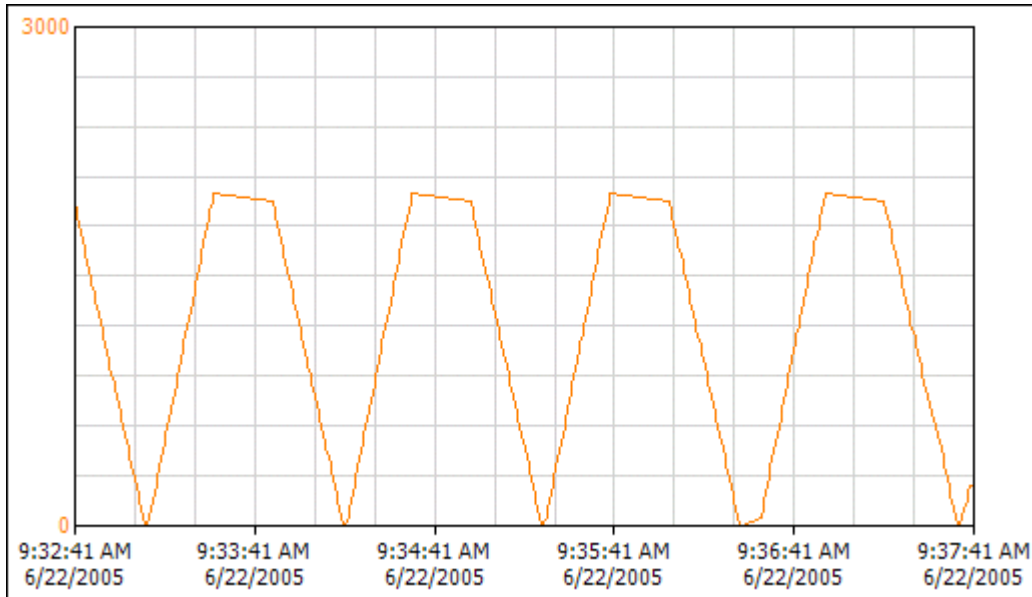
You can configure the chart to show multiple value scales. For multiple scales, only the minimum and maximum values are shown on the value axis. The scale label colors match the pen colors of the corresponding tags. The values of hidden tags are not shown.

To show multiple scales

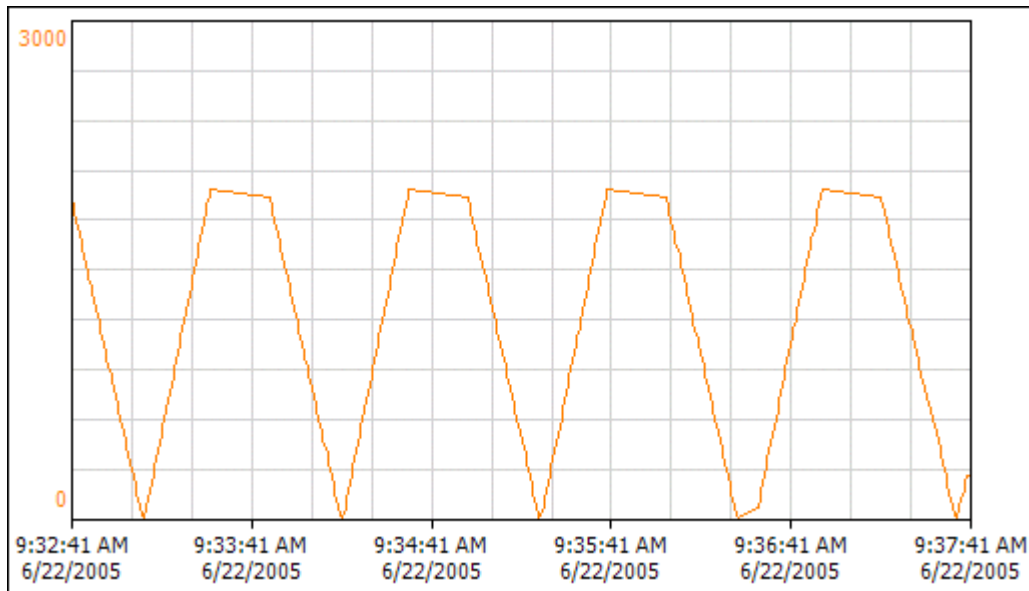
- ◆ On the View menu, click **Multiple Scales**.

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.

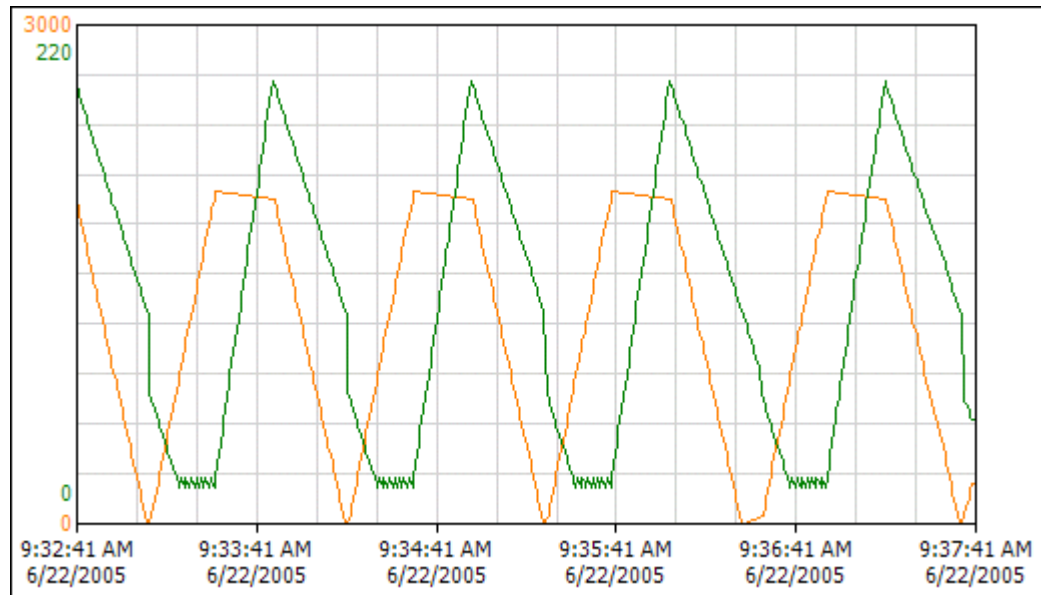
The following chart includes single tag and is configured to use multiple scales. Stacked mode is not applied.



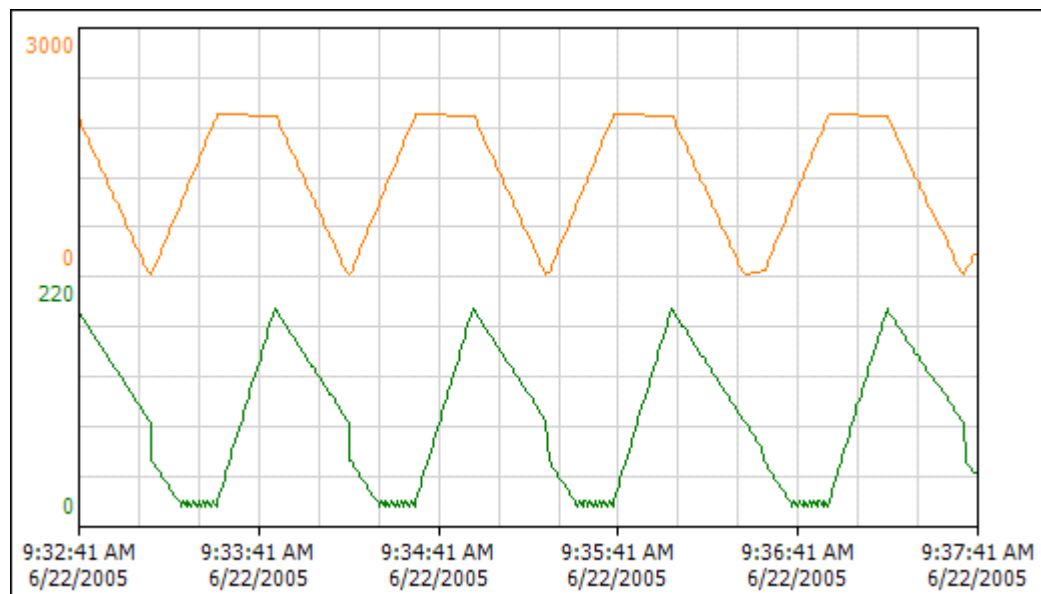
The following chart includes single tag and is configured to use multiple scales. Stacked mode is applied.



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.



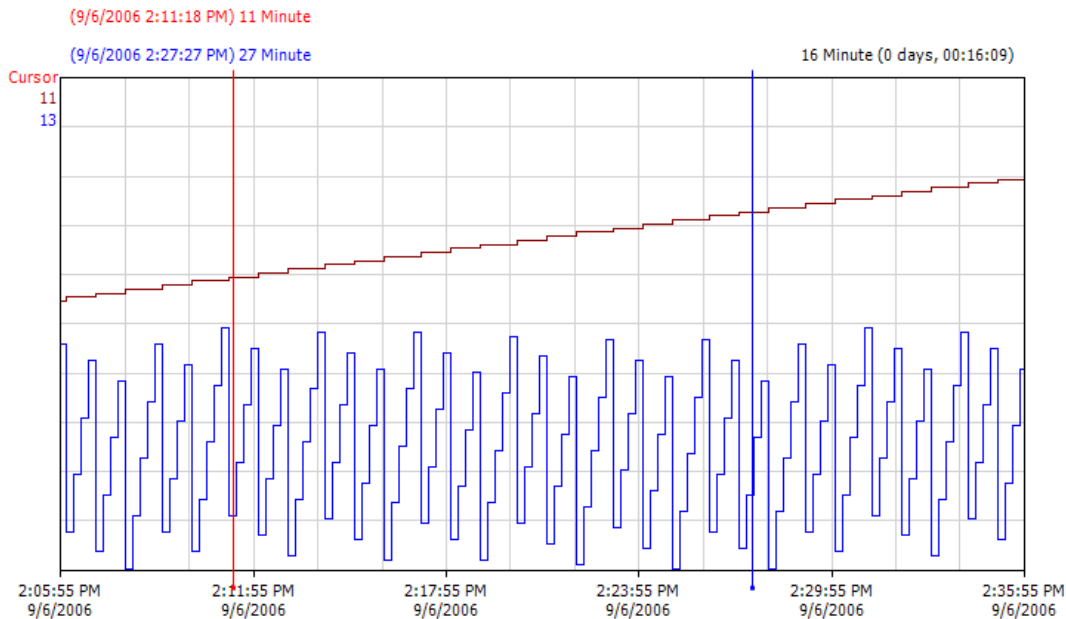
Showing Cursor Values on the Value Axis

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

- ◆ On the View menu, click Values At Cursor.

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 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 single tag up

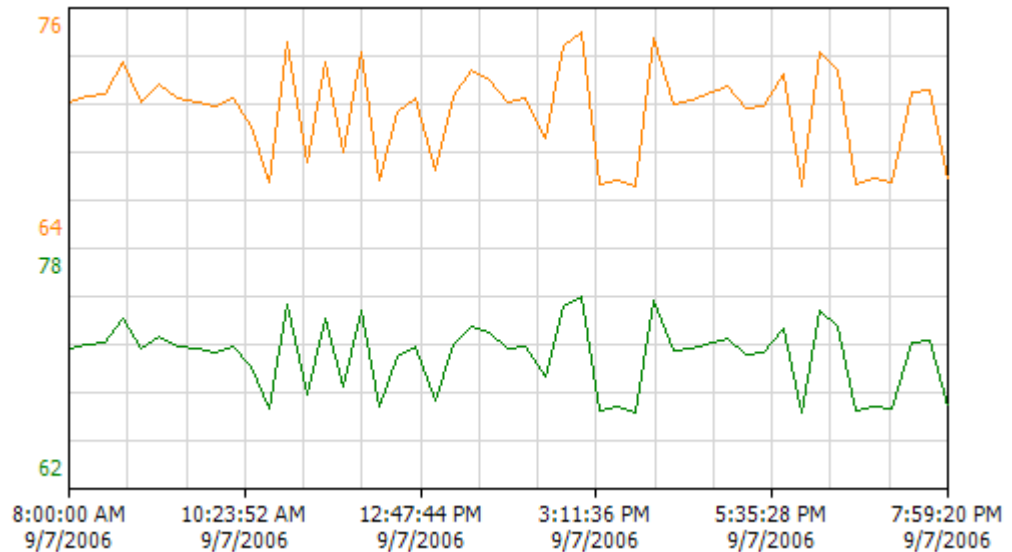
- ◆ Do one of the following:
 - On the Chart menu, point to Scale Tag and then click Scale Up.
 - Click the Scale Tag Up toolbar button.



To scale all tags up

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale All Tags** and then click **Scale Up**.
 - Click the **Scale All Tags Up** toolbar button.

The following example shows a single tag scaled up:

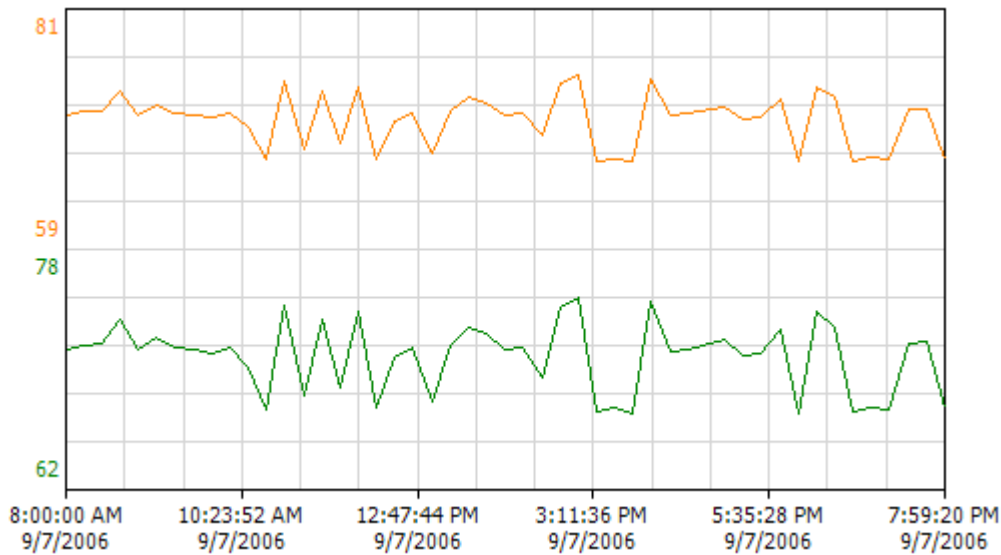
**To scale single tag down**

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale Tag** and then click **Scale Down**.
 - Click the **Scale Tag Down** button.

To scale all tags down

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale All Tags** and then click **Scale Down**.
 - Click the **Scale All Tags Down** button.

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 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](#) on page 93.

Autoscaling is not performed continually for a trend. If the trended data includes points outside of the scale region, an indicator appears to remind you to perform a second autoscale operation.

To automatically scale a single tag

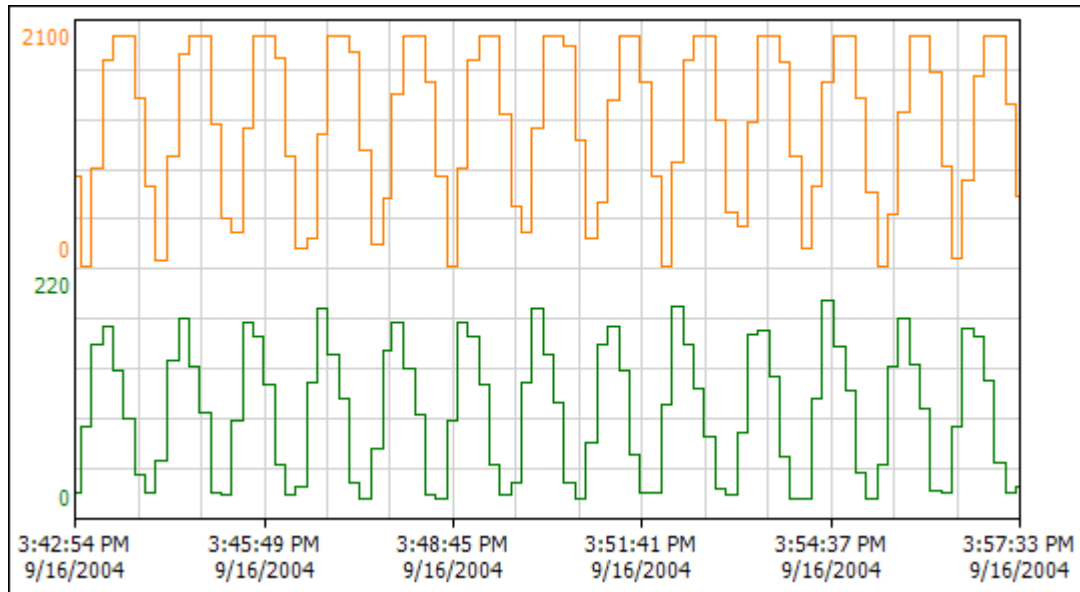
- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale Tag** and then click **Auto Scale**.
 - Click the **Auto Scale Tag** toolbar button.



To automatically scale all tags

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale All Tags** and then click **Auto Scale**.
 - Click the **Auto Scale All Tags** toolbar button.

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

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale Tag** and then click **Original Scale**.
 - Click the **Tag Original Scale** toolbar button.

To return all tags to original scale

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale All Tags** and then click **Original Scale**.
 - Click the **All Tags Original Scale** toolbar button.

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

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale Tag** and then click **Move Up**.



- Click the **Move Tag Up** toolbar button.

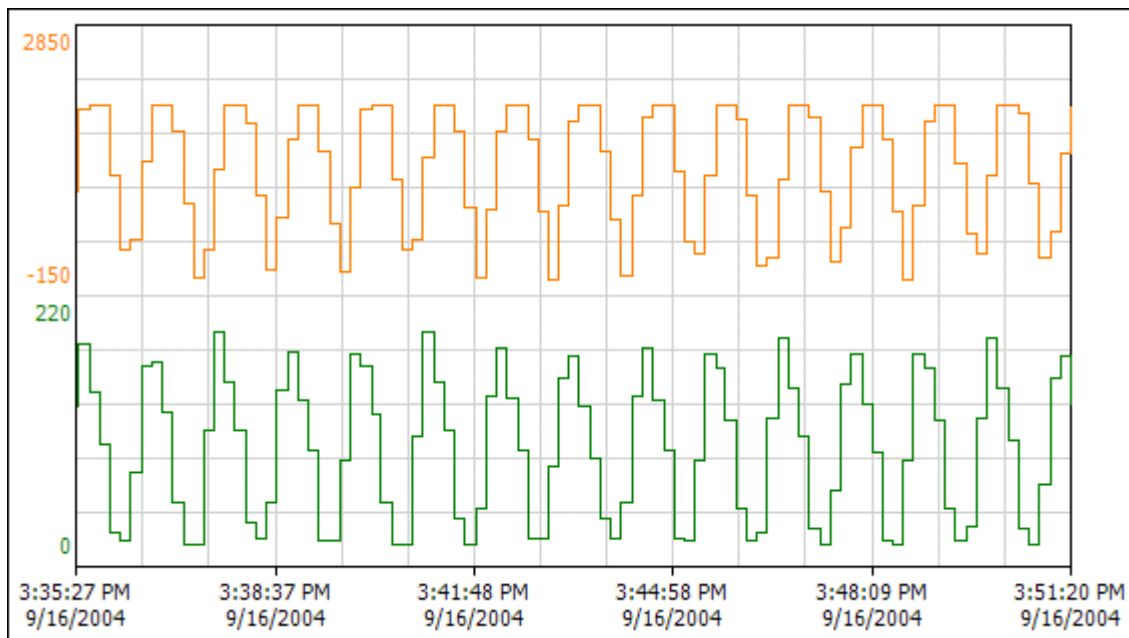
To move all tags up

- ◆ Do one of the following:
 - On the **Chart** menu, point to **Scale All Tags** and then click **Move Up**.



- Click the **Move All Tags Up** toolbar button.

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 Do one of the following:
 - On the **Chart** menu, click **Rubber Band Scaling** so that a check mark appears.



- Click the **Rubber Band Scaling** toolbar button.

- 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, do one of the following:

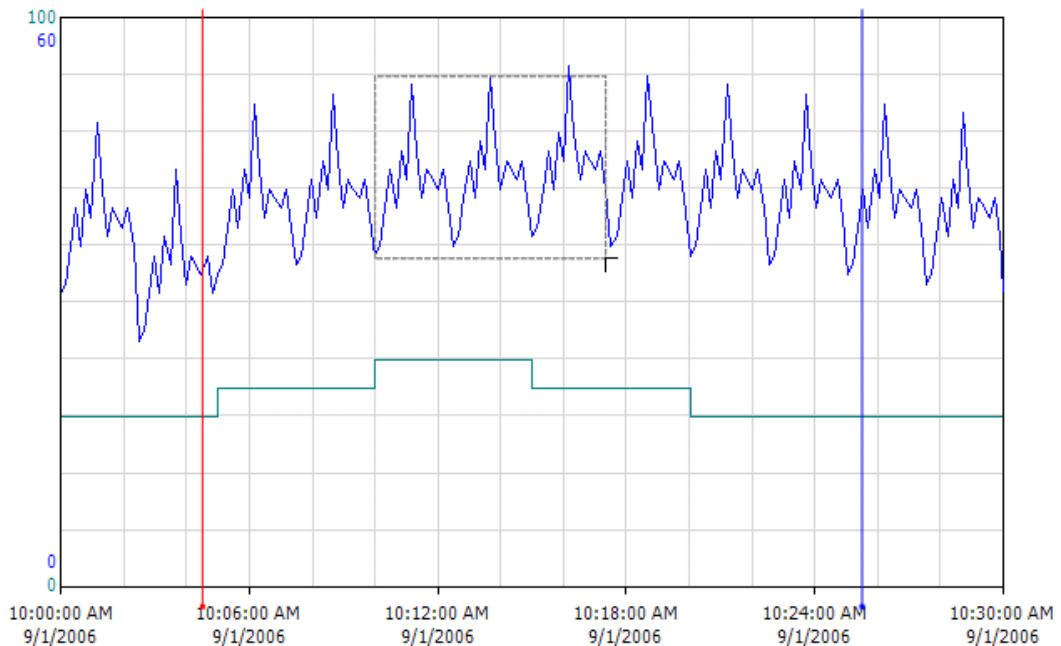
- On the **Chart** menu, click **Apply Rubber Band To All Tags** so that a check mark appears.



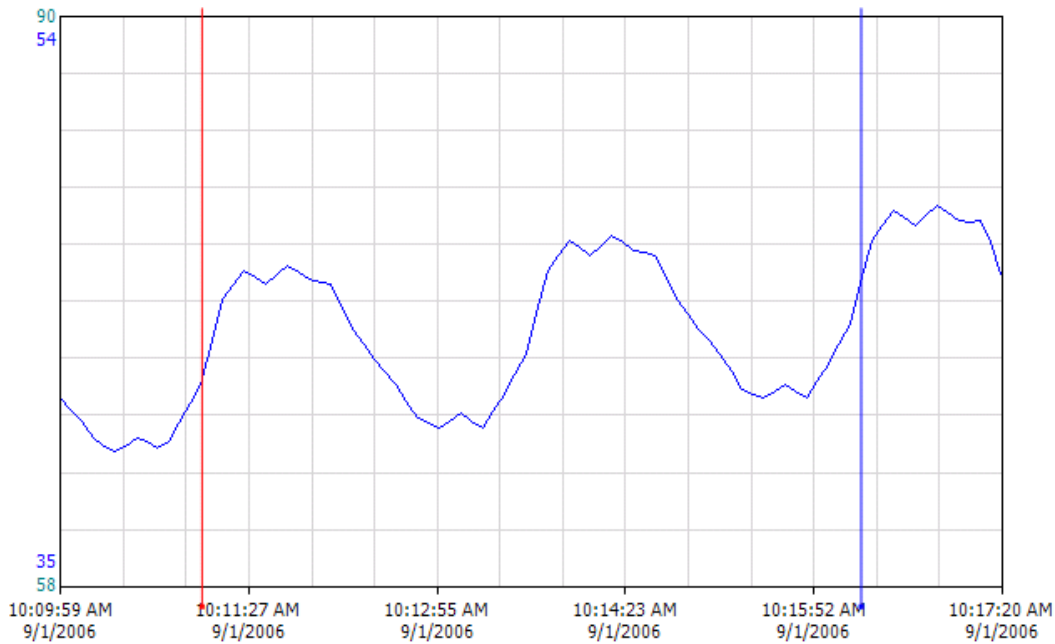
- Click the **Apply Rubber Band to All Tags** toolbar button so that it is highlighted.

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.



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



Rubber band mode remains in effect until you turn it off by clicking either the **Rubber Band Scaling** menu command or toolbar button.

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](#) on page 136.

To pan left

- ◆ Do one of the following:
 - On the **Chart** menu, click **Pan Left**.
 - Click the **Pan Left** toolbar button.



To pan right

◆ Do one of the following:

- On the **Chart** menu, click **Pan Right**.
- Click the **Pan Right** toolbar button.

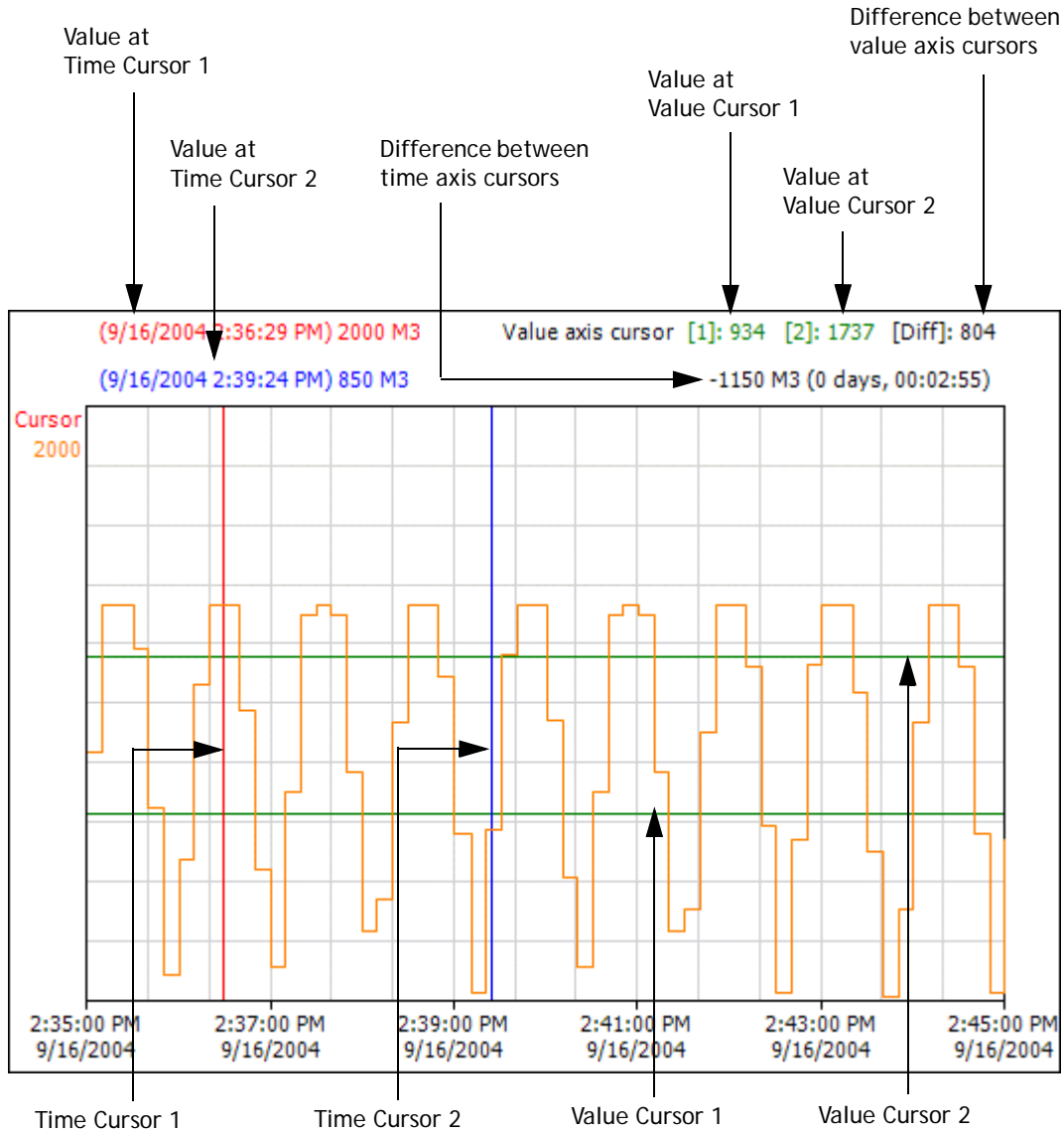


If the time scale is set into the future, then white space appears.

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* on page 136.

Moving a Cursor

To move a cursor

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

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

- ◆ Do one of the following:
 - On the **View** menu, click **Time Axis Cursors** so that a check mark appears.
 - Click the **Time Axis Cursors** toolbar button.



To show the value axis cursors

- ◆ Do one of the following:
 - On the **View** menu, click **Value Axis Cursors** so that a check mark appears.
 - Click the **Value Axis Cursors** toolbar button.



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

Showing/Hiding the Cursor Difference

To show the cursor difference

- ◆ On the **View** menu, click **Cursor Difference** so that a check mark appears.

To hide the cursor difference, follow the same procedure so that no check mark 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* on page 136.

To zoom in

- ◆ Do one of the following:
 - On the **Chart** menu, click **Zoom In**.



- Click the **Zoom In** toolbar button.

To zoom out

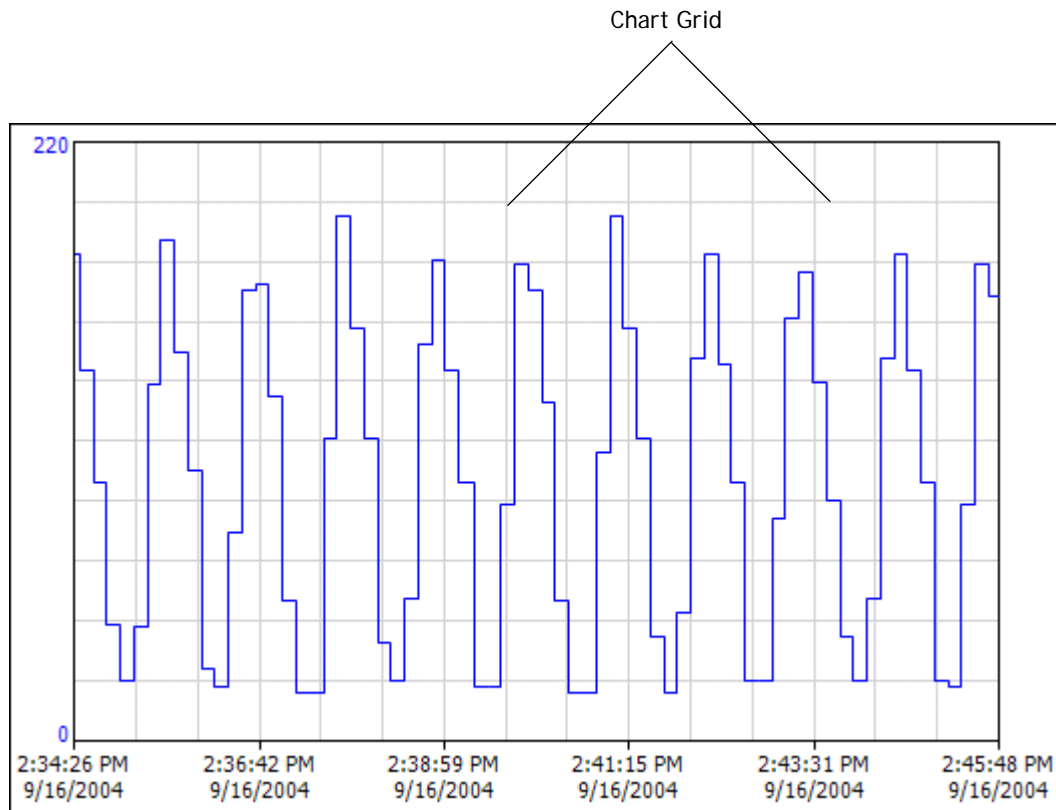
- ◆ Do one of the following:
 - On the **Chart** menu, click **Zoom Out**.



- Click the **Zoom Out** toolbar button.

Showing/Hiding the Chart Grid

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



To show the horizontal lines

- ◆ Do one of the following:
 - On the **View** menu, click **Horizontal Grid** so that a check mark appears.



- Click the **Horizontal Grid** toolbar button so that it is highlighted.

To show the vertical lines

- ◆ Do one of the following:
 - On the **View** menu, click **Vertical Grid** so that a check mark appears.
 - Click the **Vertical Grid** toolbar button so that it is highlighted.

To hide the lines, click the appropriate menu command so that no check mark appears or click the toolbar button so that it is not highlighted.

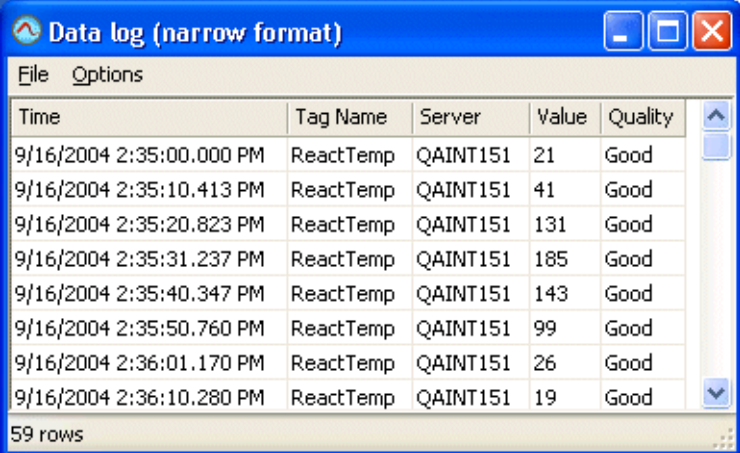
Viewing Trend Data in a Table Format

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, point to **Data Log**, and then click **Narrow**. The Data log dialog box appears.



The screenshot shows a dialog box titled "Data log (narrow format)" with a menu bar containing "File" and "Options". The main area is a table with the following data:

Time	Tag Name	Server	Value	Quality
9/16/2004 2:35:00.000 PM	ReactTemp	QAIN151	21	Good
9/16/2004 2:35:10.413 PM	ReactTemp	QAIN151	41	Good
9/16/2004 2:35:20.823 PM	ReactTemp	QAIN151	131	Good
9/16/2004 2:35:31.237 PM	ReactTemp	QAIN151	185	Good
9/16/2004 2:35:40.347 PM	ReactTemp	QAIN151	143	Good
9/16/2004 2:35:50.760 PM	ReactTemp	QAIN151	99	Good
9/16/2004 2:36:01.170 PM	ReactTemp	QAIN151	26	Good
9/16/2004 2:36:10.280 PM	ReactTemp	QAIN151	19	Good

At the bottom of the dialog box, it indicates "59 rows".

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 Wonderware 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 Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
- **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.

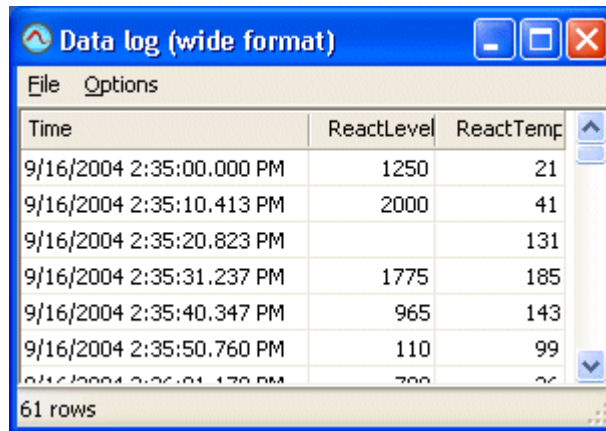
- 2 To include only the data that is between the time axis cursors on the chart, on the **Options** menu, click **Data From Between Cursors**.
- 3 To include all of the data on the chart, on the **Options** menu, click **Data From Between Graph Start/End**.
- 4 To show actual values for discrete tags (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, point to **Data Log** and then click **Wide**. The Data log dialog box appears.



Time	ReactLevel	ReactTemp
9/16/2004 2:35:00.000 PM	1250	21
9/16/2004 2:35:10.413 PM	2000	41
9/16/2004 2:35:20.823 PM		131
9/16/2004 2:35:31.237 PM	1775	185
9/16/2004 2:35:40.347 PM	965	143
9/16/2004 2:35:50.760 PM	110	99
9/16/2004 2:36:01.170 PM	700	27

61 rows

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 Wonderware 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 Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference.
For more information, see ArcestrA Naming Conventions on page 24.

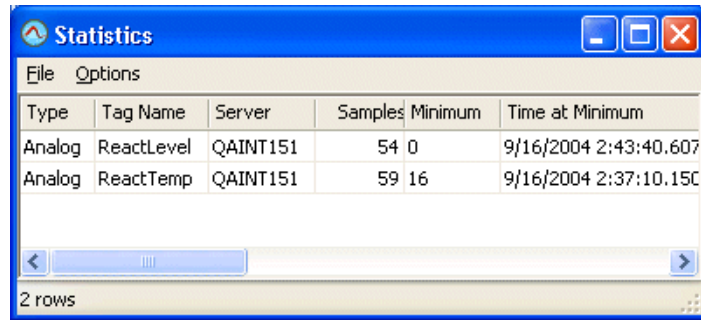
- 2 To include only the data that is between the time axis cursors on the chart, on the **Options** menu, click **Data From Between Cursors**.
- 3 To include all of the data on the chart, on the **Options** menu, click **Data From Between Graph Start/End**.
- 4 To show actual values for discrete tags (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**.

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 Wonderware Historian server. If the data values are coming from ArchedrA, the attribute reference is shown as the tag name. For ArchedrA attributes, you can also choose to show the hierarchical name along with the attribute reference.
For more information, see ArchedrA Naming Conventions on page 24.
- **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.
 - **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 Wonderware 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.

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

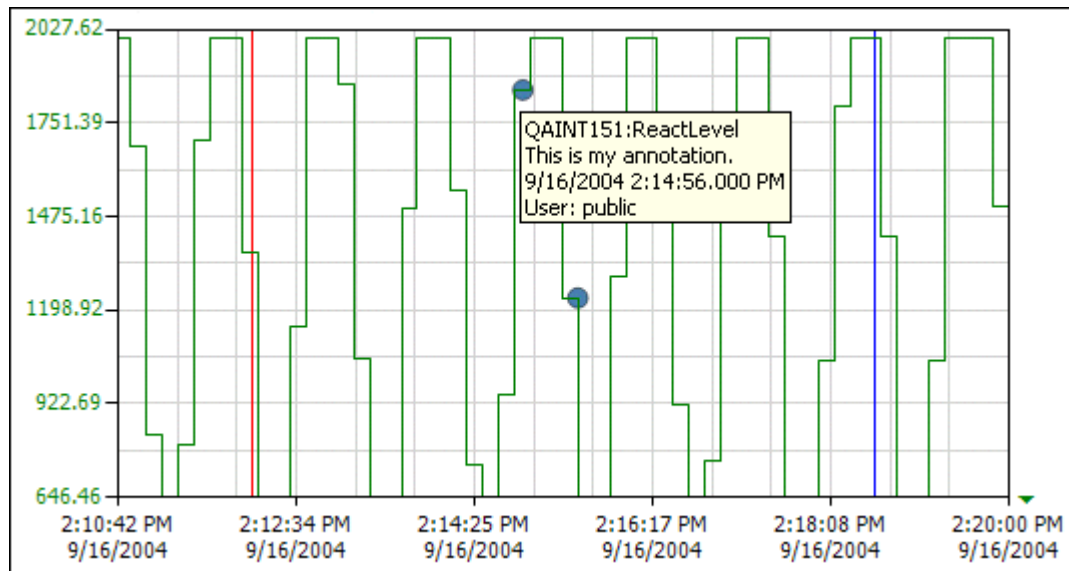
- 1 Select the tag for which you want to add an annotation. You can do this by selecting the tag in the Tag List pane.
- 2 Do one of the following:
 - On the **Chart** menu, click **Add Annotation**.
 - Right-click in the chart. In the shortcut menu that appears, click **Add Annotation**.

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 appear on the chart.



Viewing the Annotation List

To view a list of annotations

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

Type	Tag Name	Server	Content	User	Time	Created On
Public	ReactLevel	QAIN15	This is my annotation.	public	9/16/2004 2:14:56.	12/18/2004
Public	ReactLevel	QAIN15	This is another annotation.	public	9/16/2004 2:15:30.	12/18/2004

The table in the window shows the following information.

- **Type** Specifies whether the annotation is public.
- **Tag Name** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference.

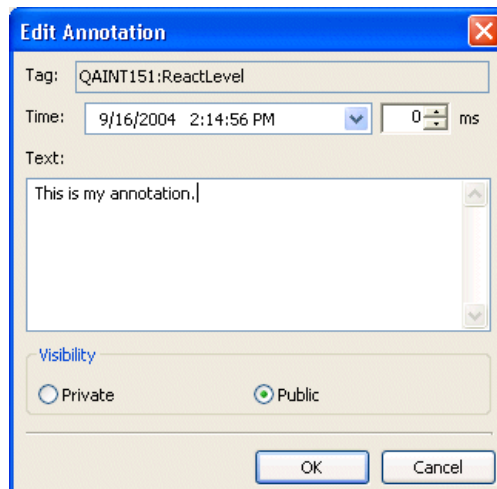
For more information, see ArchestrA Naming Conventions on page 24.

- **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 **Annotation List**. 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.



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

Deleting an Annotation

Deleting an annotation removes the annotation from the trend.

To delete an annotation

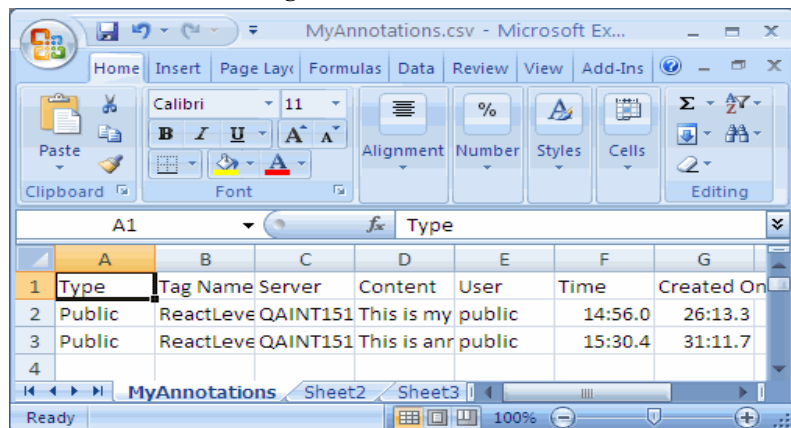
- 1 On the **View** menu, click **Annotation List**. The **Annotations** dialog box appears.
- 2 Select the annotation in the list.
- 3 On the **Annotations** menu, click **Delete**. Confirm the deletion.
- 4 Click **OK**.

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 **Annotation List**. 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.



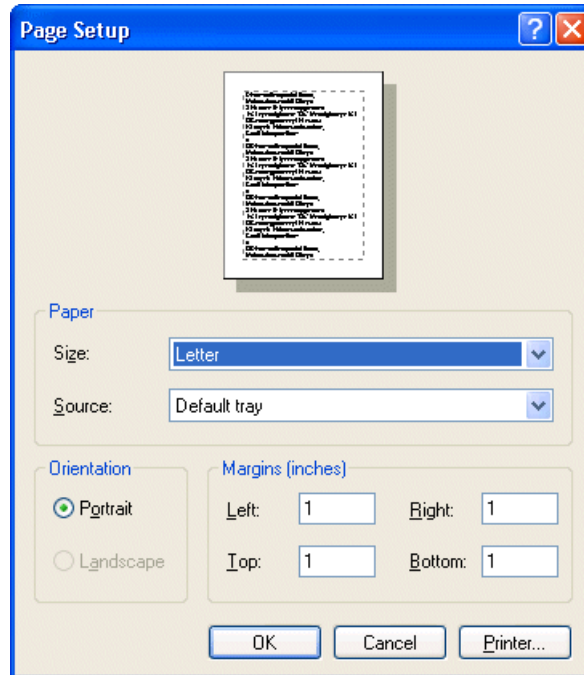
The screenshot shows a Microsoft Excel spreadsheet with the following data:

	A	B	C	D	E	F	G
1	Type	Tag Name	Server	Content	User	Time	Created On
2	Public	ReactLeve	QAINT151	This is my	public	14:56.0	26:13.3
3	Public	ReactLeve	QAINT151	This is anr	public	15:30.4	31:11.7
4							

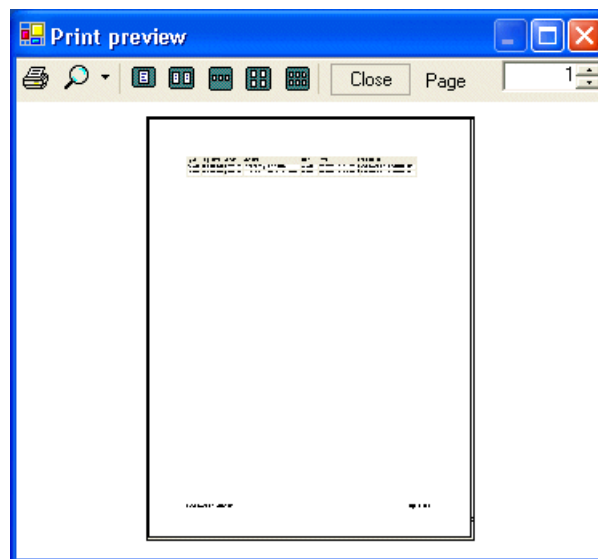
Printing Annotations

To print the list of annotations

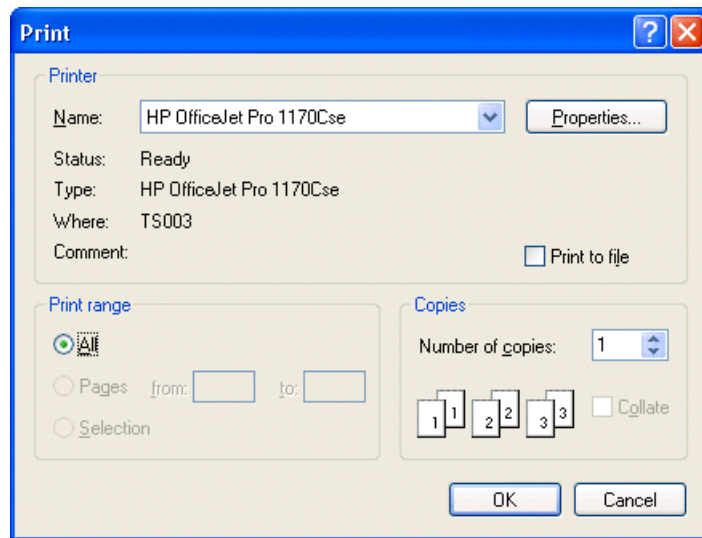
- 1 On the View menu, click **Annotation List**. The **Annotations** dialog box appears.
- 2 To configure the printing options, on the **File** menu, click **Page Setup**. The **Page Setup** dialog box appears.



- 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.



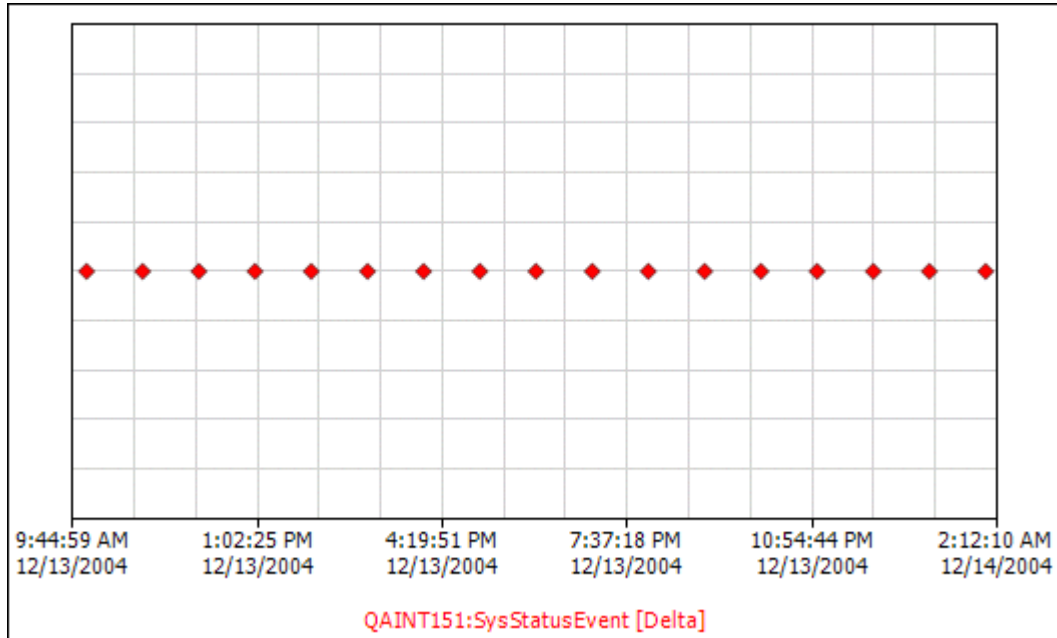
- 7 Configure the printing options and then click **OK**.

Trending Events

A trend can be configured to show event data. An event is the set of attributes describing the moment of satisfaction of a set of criteria on historical tag values in the Wonderware 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 in the same way as any other tag in the system. Events can also be displayed along with analog and discrete tags.



Using Absolute or Relative Times

The following date modes are available for the trend chart:

- Absolute time
- Relative time

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

Using Absolute Time

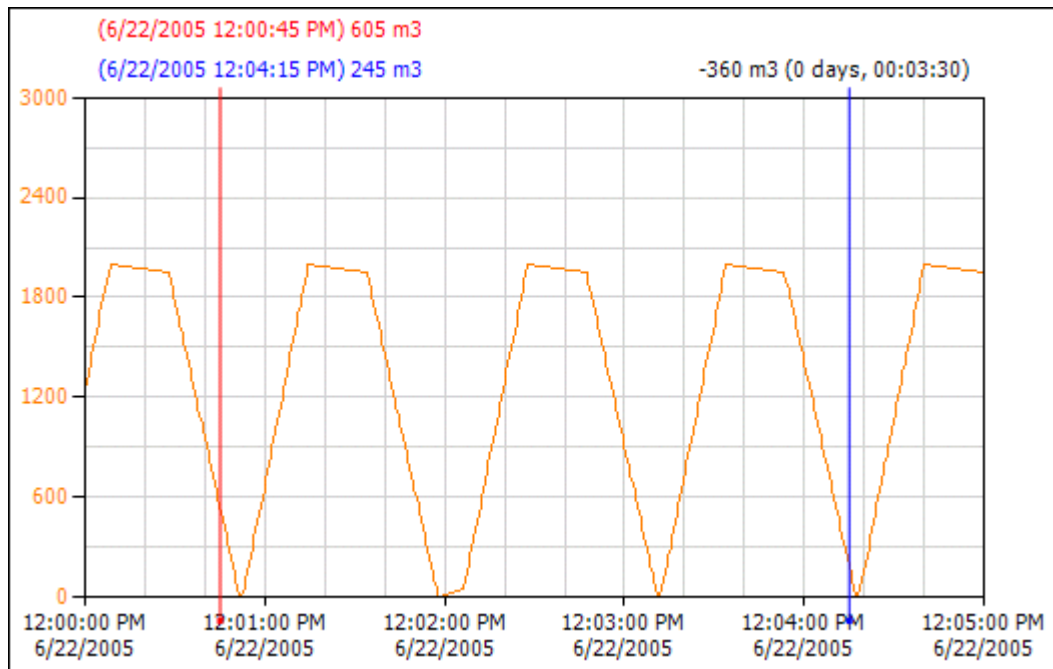
For the absolute date mode, the start and end dates in the data query are used for the start and end times in the chart, respectively.

To use absolute time

- ◆ On the **View** menu, click **Absolute Time**.

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 date mode:

- The times shown for the time axis cursors, if enabled, are absolute times.
- The times shown in the Time Bar are absolute times.

9/ 6/2006 12:00:00 PM [00] 00:05:00.000 9/ 6/2006 12:05:00 PM

- The Tag List shows the time offset for the chart data, relative to 0. In this example, there is no offset configured.

Address	Time Offset	Min Raw	M
&INT07...	0:00:00.000	0	

- The Tag Properties dialog box shows the time offset option. For more information, see [Configuring Trend Options for a Tag](#) on page 67.

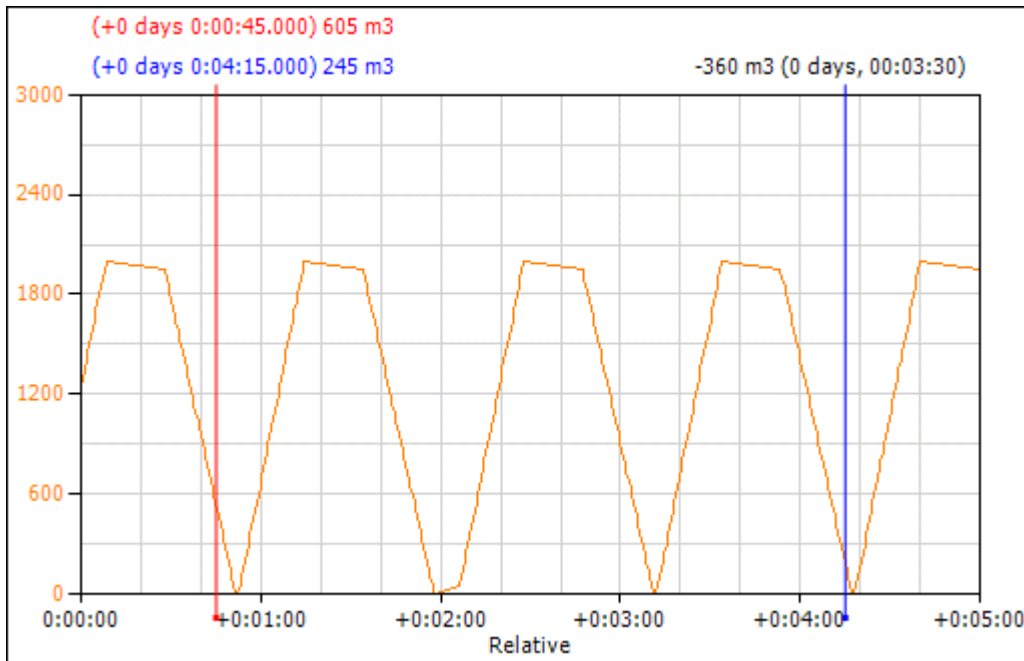
Using Relative Time

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 or the actual start and end time of the trend query. Only the time axis is updated.

To use relative time

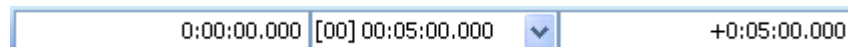
- ◆ On the View menu, click **Relative Time**.

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 more information on the display format for times, see [Time Offset Formats](#) on page 118.
- The times shown in the Time Bar are relative times. The first offset time is the base time (for example, 0:00:00.000), and the second time is the time span of the specified offset. The first offset time is always set to 0:00:00 when you transition into relative mode.



- The Tag List shows the actual start time for the tag data.

IO Address	Start Time	Min Raw
\\QAIN07...	6/22/2005 12:00:00 PM	0

- The Tag Properties dialog box shows the start time option. For more information, see Configuring Trend Options for a Tag on page 67.

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 supported notations for specifying a time offset are:

[ws][±][dws]hh:mm[:ss[.fff]][ws]

-OR-

[ws][±]HH:mm[:ss[.fff]][ws]

-OR-

[ws][±]d[.FF][ws]

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

The notation variables are as follows.

Item	Description
ws	White space.
±	Minus sign indicating a negative time. Positive time is assumed.
dws	Days, with trailing white space.
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

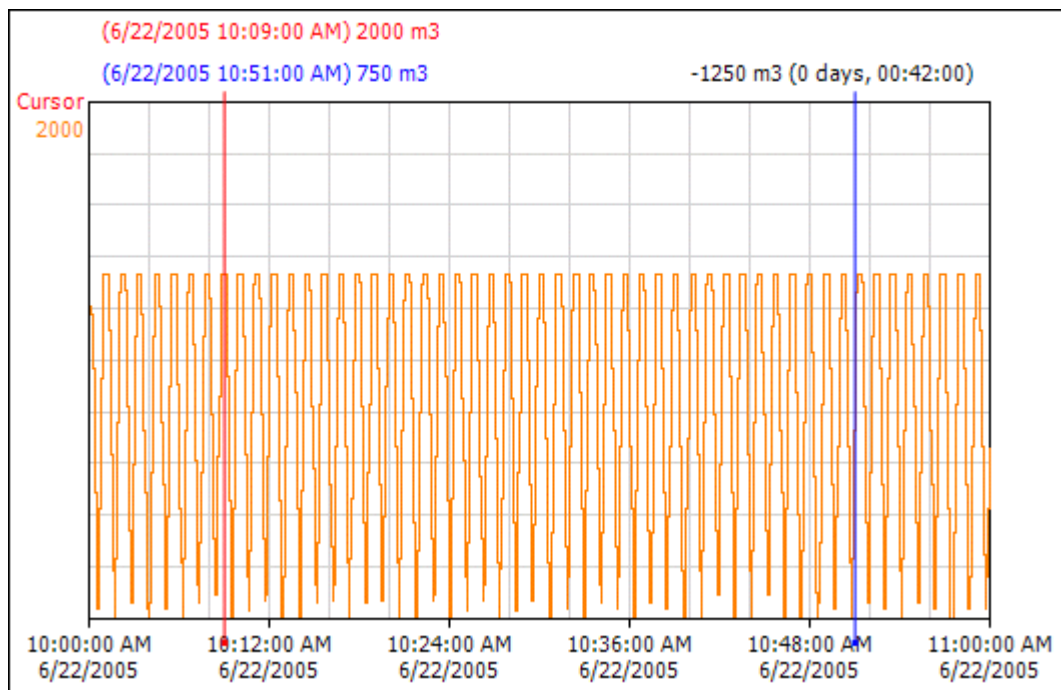
You can use a time offset to compare the same data from different time periods. For example, you may want to compare data from a shift at 10:00 a.m. to data from a shift at 11:00 a.m. The time offset feature allows you to adjust the time period for one of the shifts so that the data appears as if it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

For information on configuring a time offset, see *Configuring Trend Options for a Tag* on page 67.

To use a time offset

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

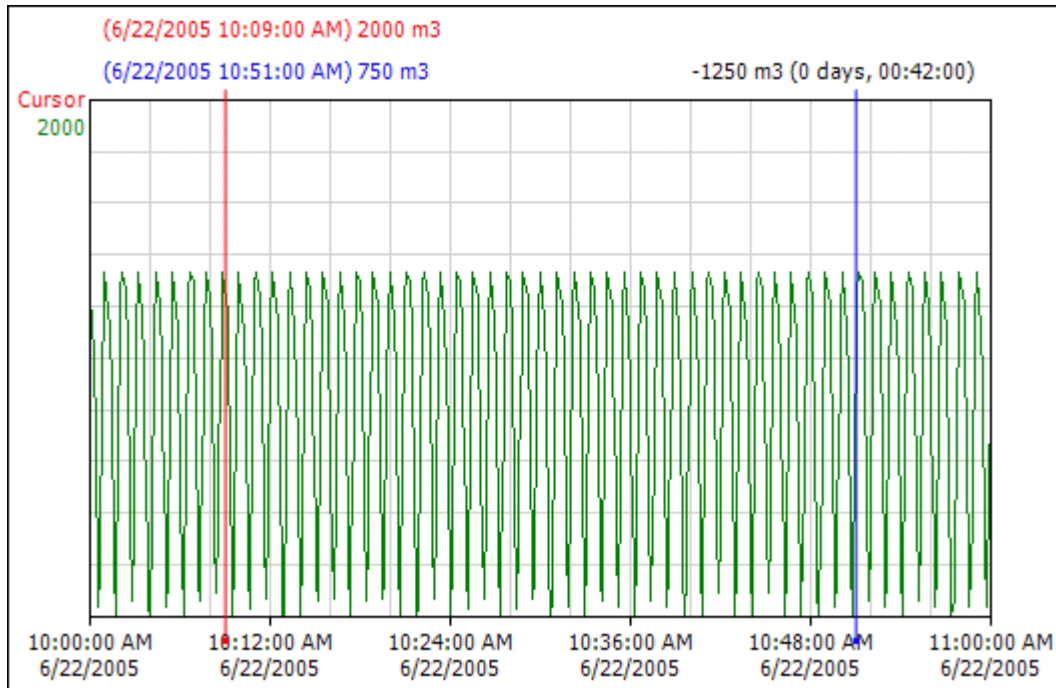
In this example, the chart is configured to show data for the ReactLevel tag between 6/22/2005 10:00:00 AM and 6/22/2005 11:00:00 AM.



- 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.

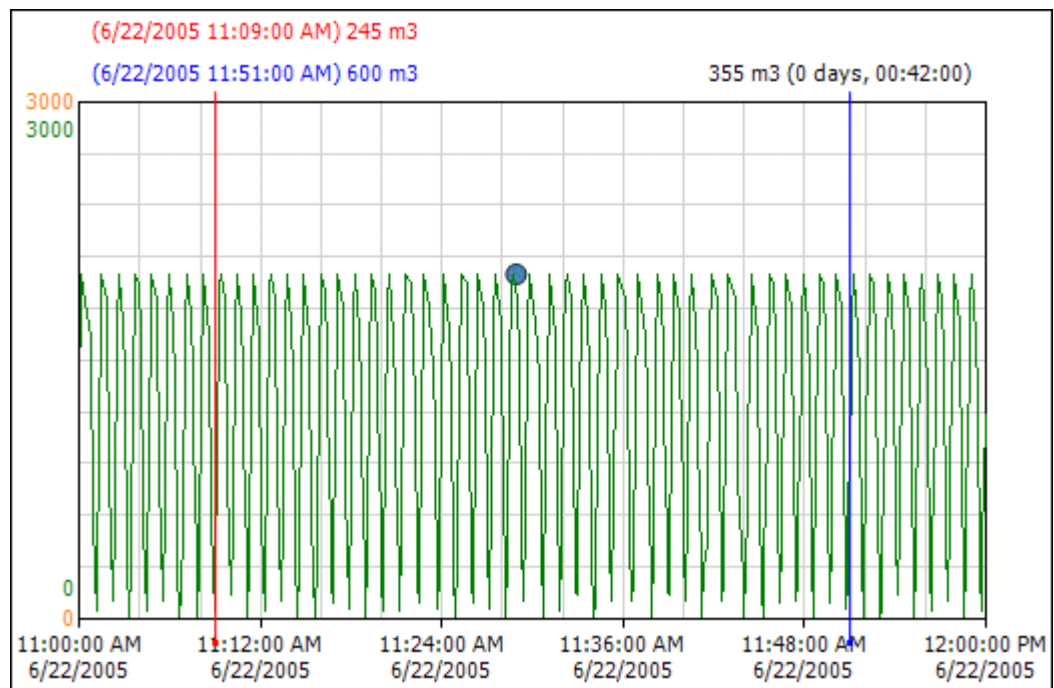
The screenshot shows the 'LOC-0322030:ReactLevel' Tag Properties dialog box. The 'General' tab is selected. The 'Pen configuration' section shows Color: Green, Width: 1, and Style: Solid. The 'Value axis range' section shows Bottom: 0 and Top: 3000. The 'Type' is set to Line, Decimal places to 0, and Format to Decimal. The 'Time offset' is set to 1:00:00.000. Buttons for OK, Cancel, and Apply are at the bottom.

- 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** on page 118.

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.

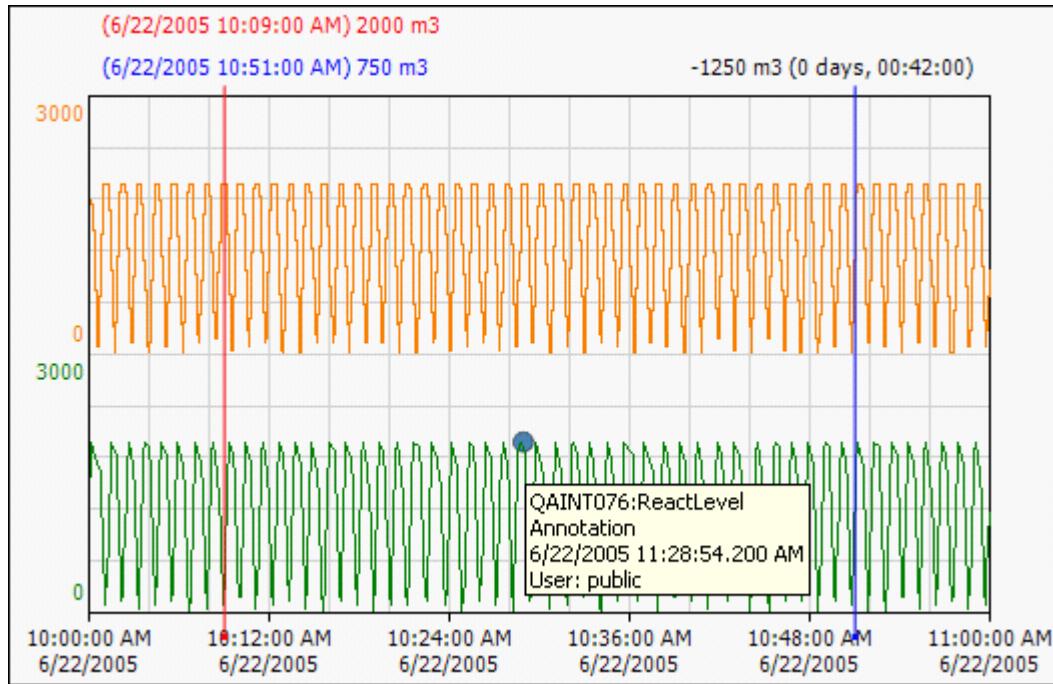
- 7 Click **OK**.

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.

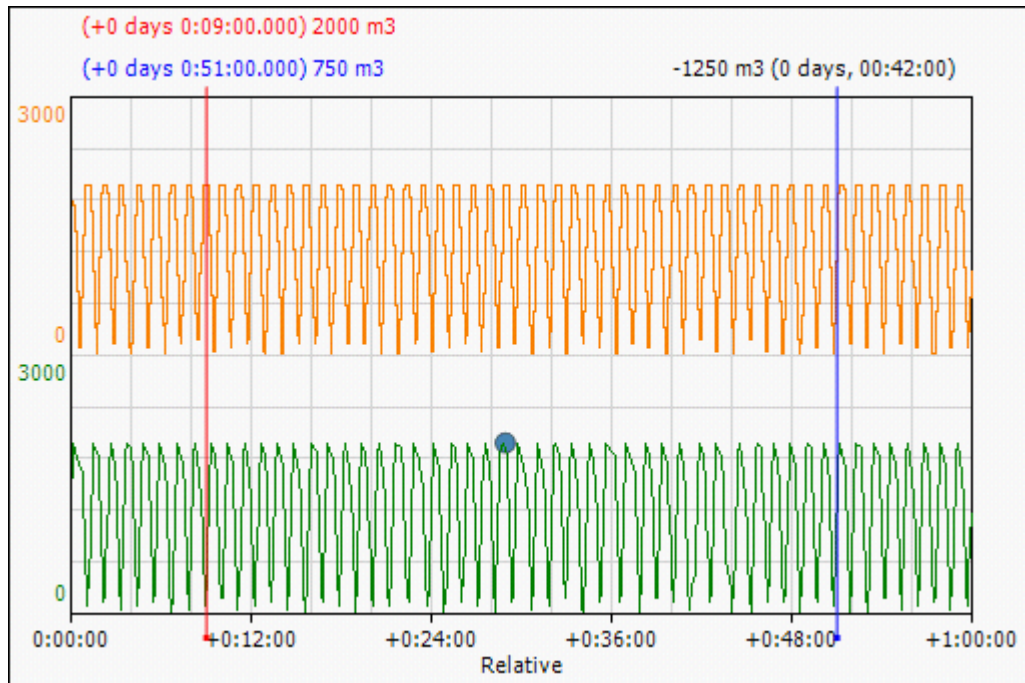


- 8 Stack the traces so that you can see both sets of data separately and then select the first tag that you added to the chart.

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).



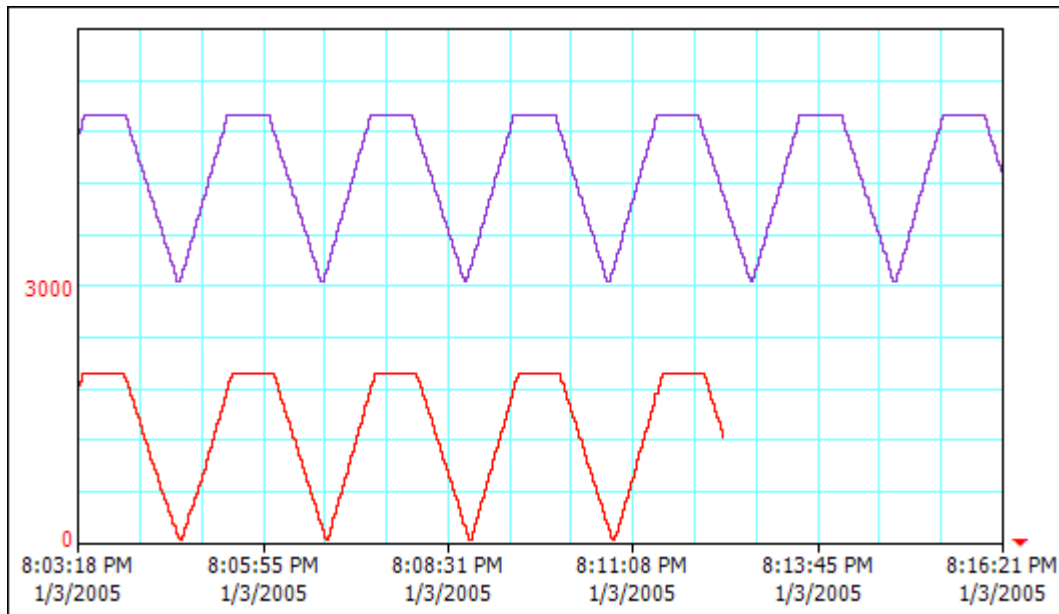
- 9 To view the chart in relative mode, on the **View** menu, click **Relative Time**. The time axis now shows the time span for the base batch starting at 0:00:00 instead of the actual time.



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

Configuring Retrieval Options

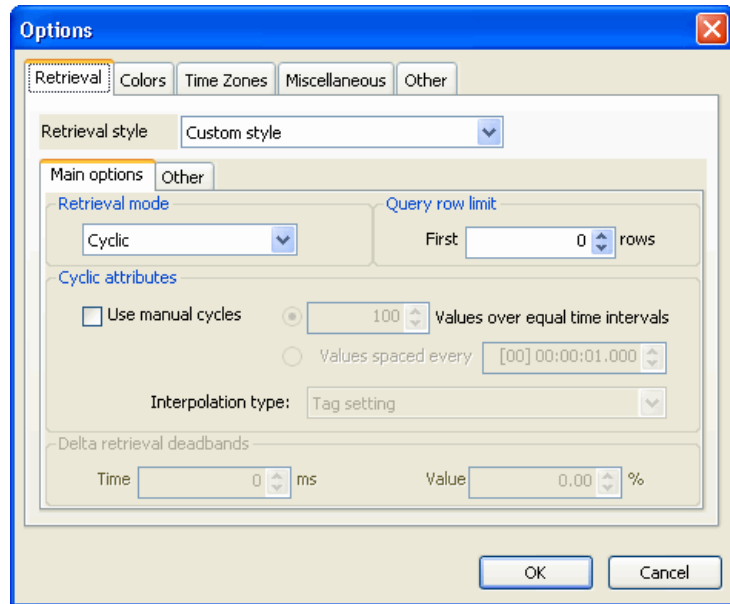
You can define data retrieval options at the application level. These options are used for all tags that do not have their own settings defined.

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* on page 67.

Most retrieval settings that you configure here only apply if you are retrieving data from a Wonderware Historian with a version of 9.0 or later. If you are using an earlier Wonderware Historian version, see [Configuring Other Options](#) on page 131 and [Working with Retrieval Styles](#) on page 809 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](#) on page 809.
 - 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](#) on page 689.
 - 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](#) on page 809.

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](#) on page 752.

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](#) on page 67.

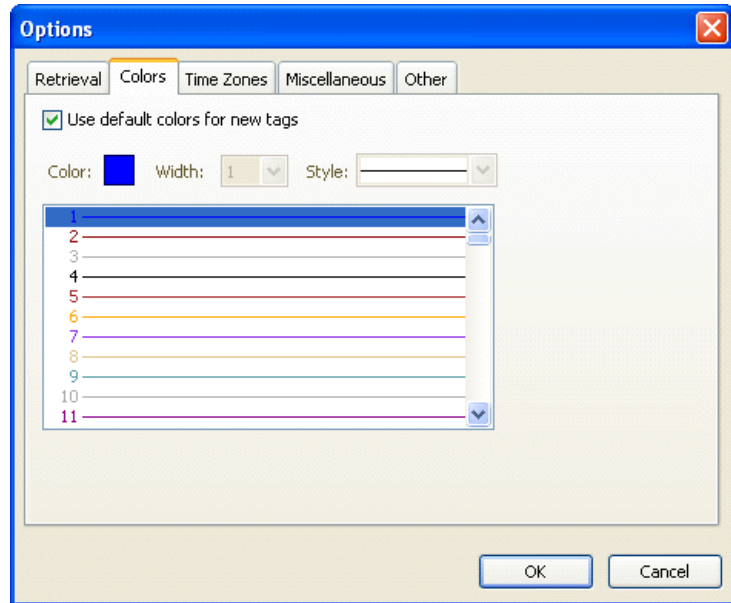
Configuring Color Options

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.



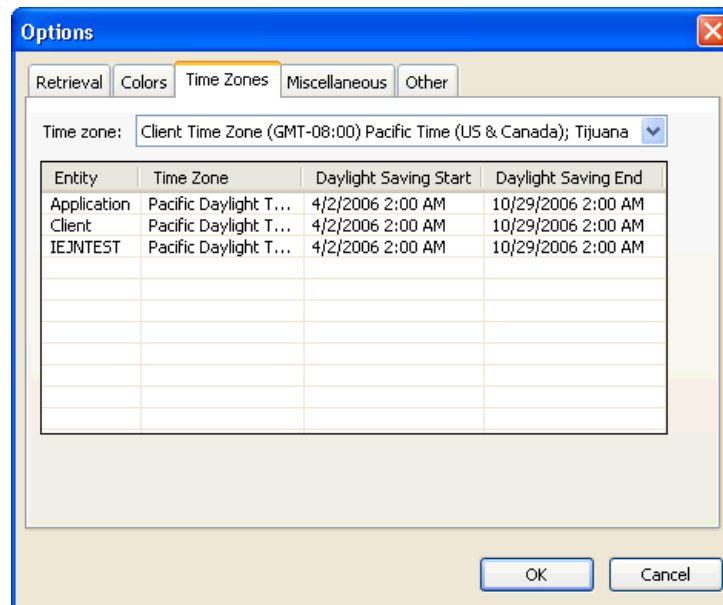
- 3 To use the default pen styles for the tags in a trend, select the **Use default colors for new tags** check box. Go to step 10.
- 4 To configure one or more pen styles, clear the **Use default colors for new tags** check box.
- 5 Select a pen number from the list.
- 6 Click the **Color** box and 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.
- 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 Trend so that it reflects the same time as the server.

To configure time zone options

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



The grid shows the current time zone and daylight savings time settings for the following entities:

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

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

The time zone for the Trend application in the grid shows the new time zone picked.

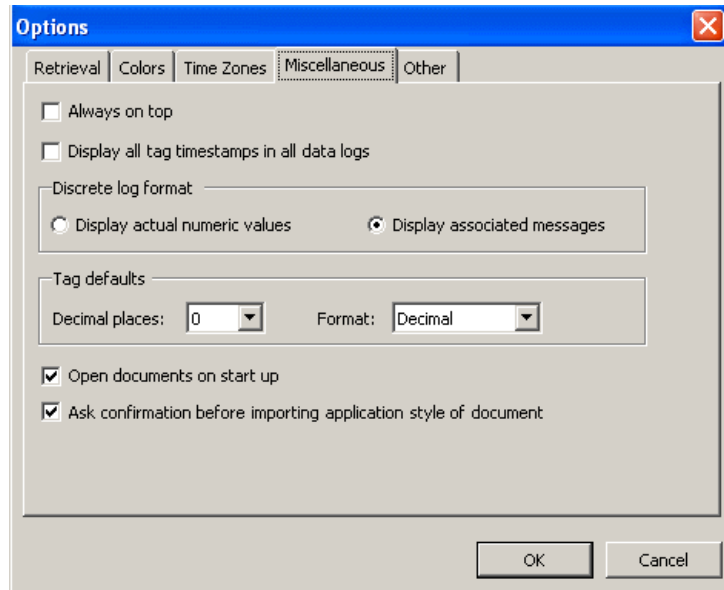
For example, consider a SCADA application that monitors a pipeline between Houston, Texas and Lake Forest, California. The Trend application is installed on a computer located in Houston, Texas. Therefore, the time zone entry for the Client entity displays Central Standard Time. The server is also located in Houston, Texas. The time zone entry for the Server entity also displays Central Standard Time. You want to send a trend 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 Trend application to reflect the time of Lake Forest, California (Pacific Standard Time), so that the trend that you send to the engineer displays data in a time zone that is relevant to him/her.

- 4 Click **OK**.

Configuring Miscellaneous Options

To configure miscellaneous options

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



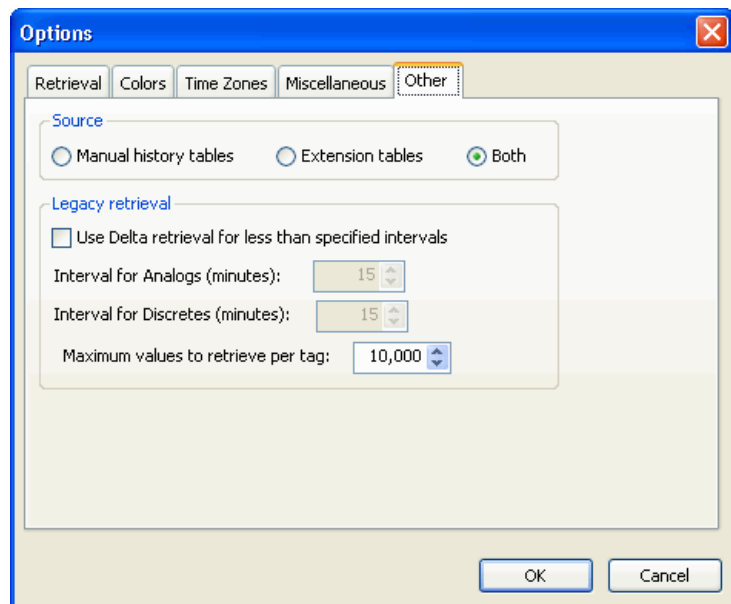
- 3 Select the **Always on top** check box to always display Trend as the top-most program on the computer desktop.
- 4 Select the **Display all tag timestamps in all data logs** check box to include the time stamps for all tags in the data log.
- 5 In the **Discrete log format** area, configure how 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.
- 7 By default, when you start the Trend application, it automatically reopens the trend files that were open when you closed it. Clear the **Open documents on startup** check box to disable this behavior.
- 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 Wonderware 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 Wonderware Historian data files. These tables support the Wonderware 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 Wonderware 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* on page 809.

- **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

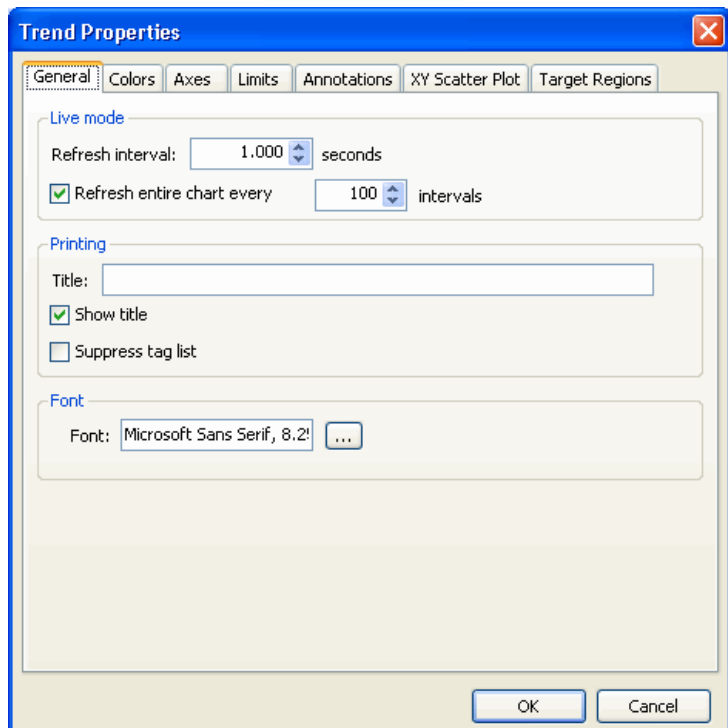
The trend properties allow you to configure the trend file. Trend file properties are saved with the trend file. Categories of trend properties that can be set include:

- Configuring General Properties
- Configuring Color Properties
- Configuring Axis Properties
- Configuring Limit Properties
- Configuring Annotation Properties
- Configuring Target Region Properties

Configuring General Properties

To configure general properties

- 1 On the Chart menu, click Properties. The Trend Properties dialog box appears.
- 2 Click the General tab.



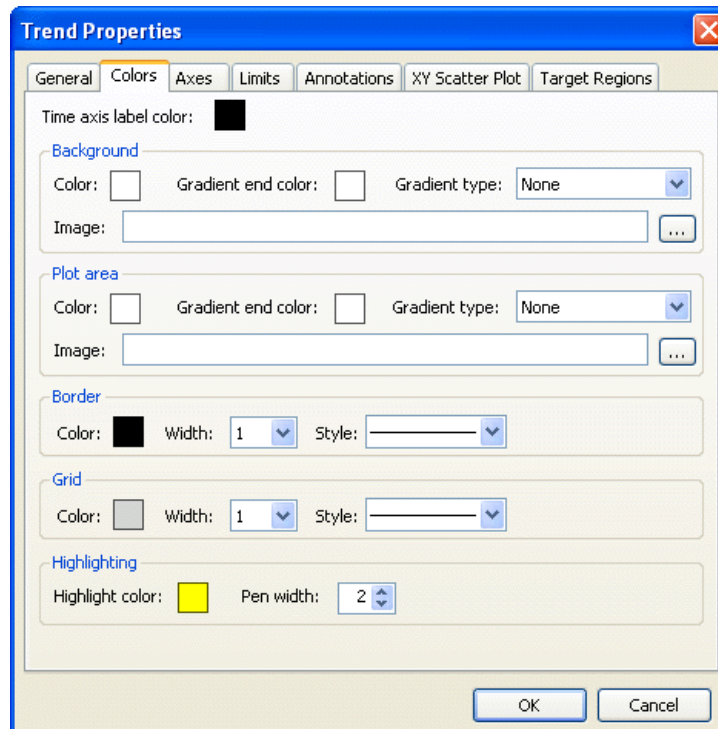
- 3 In the Refresh interval box, specify the time period, in seconds, at which the chart is refreshed if set to live mode. Valid values are 0.25 to 300. The default value is 1.

- 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.



- 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.

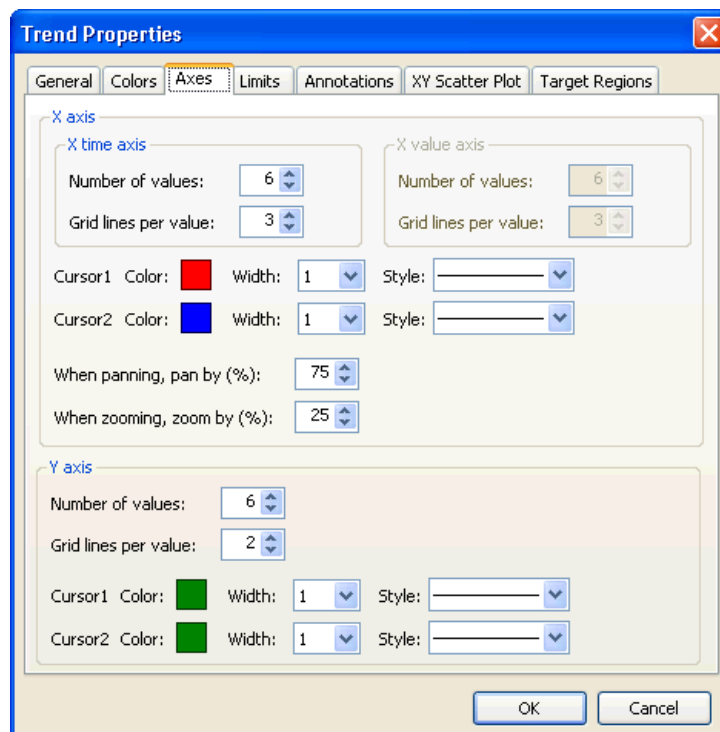
If you want to publish the trend to the Wonderware Information Server, specify the image path as a UNC path that is accessible from the Wonderware Information Server. Otherwise, the image does not appear on the clients.
- 5 In the **Plot** area, configure the colors or image to use for the chart plotting area. Options are the same as for the **Background** colors.
- 6 In the **Border** area, configure the color for the border of the chart.
 - **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.
- 9 Click **OK**.

Configuring Axis Properties

To configure axis properties

- 1 On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
- 2 Click 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.
- Color Click to select or configure the color for each time axis cursor.
- 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 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.

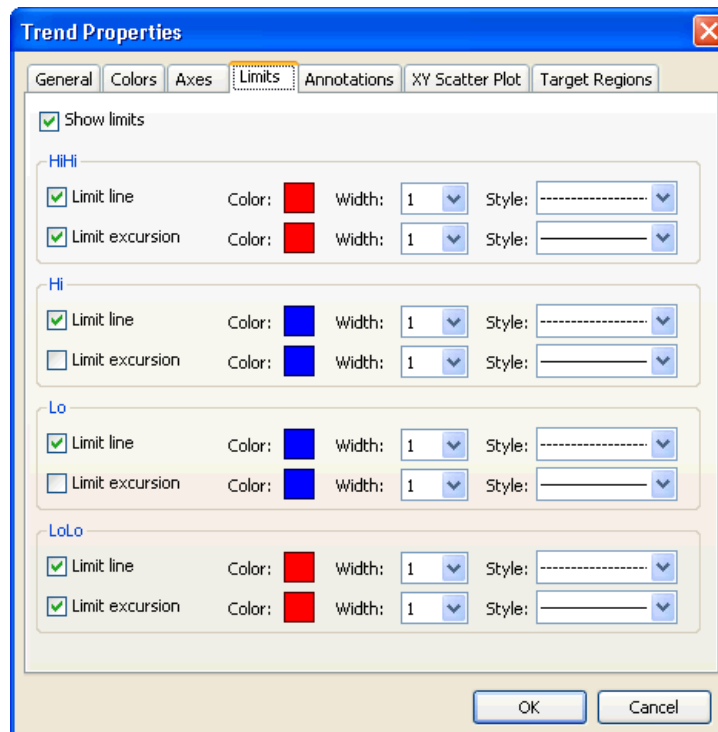
- **Width** The width of each value axis cursor.
- **Style** The line style for each value axis cursor.

5 Click OK.

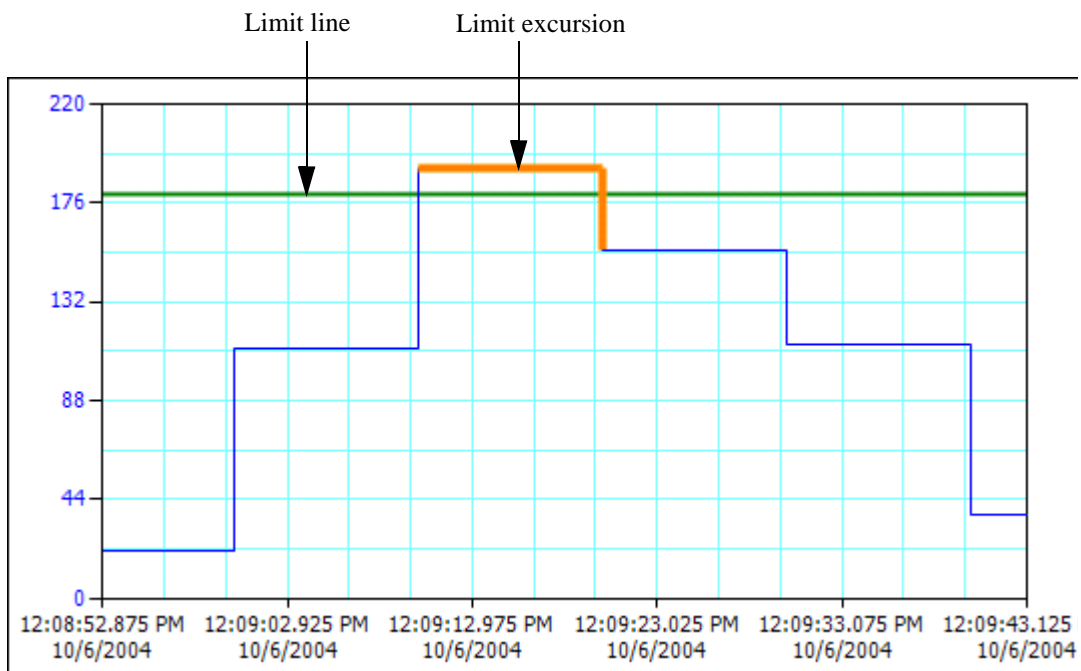
Configuring Limit Properties

To configure limit properties

- 1 On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
- 2 Click 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.

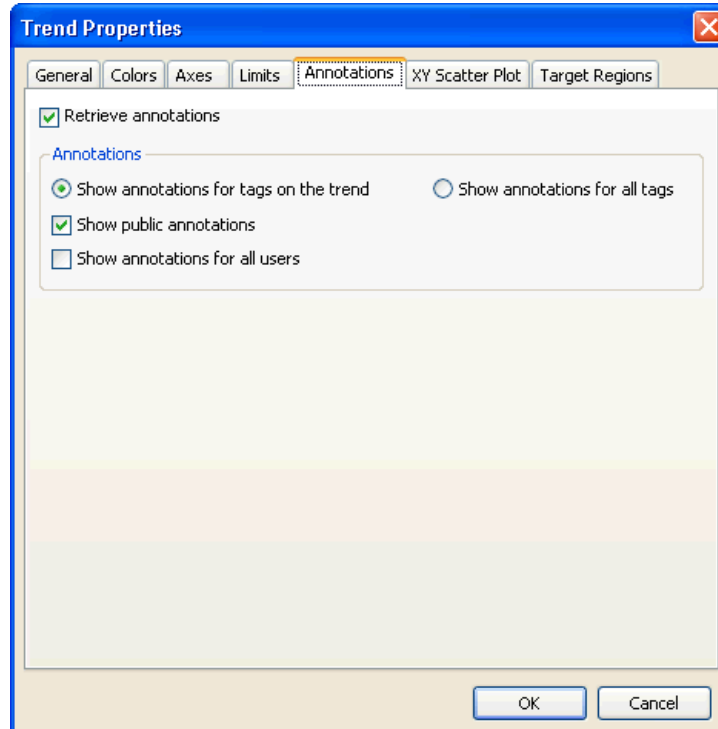


- 4 For each type of limit (HiHi, Hi, Lo, and LoLo), configure the properties of the line.
 - **Limit line** Select this check box to include a line on the chart for the limit value. For example, if an analog tag has a Hi limit of 1800, a horizontal line is drawn at the 1800 mark on the vertical axis.
 - **Limit excursion** Select this check box 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 Click the Annotations tab.



- 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 Wonderware 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.

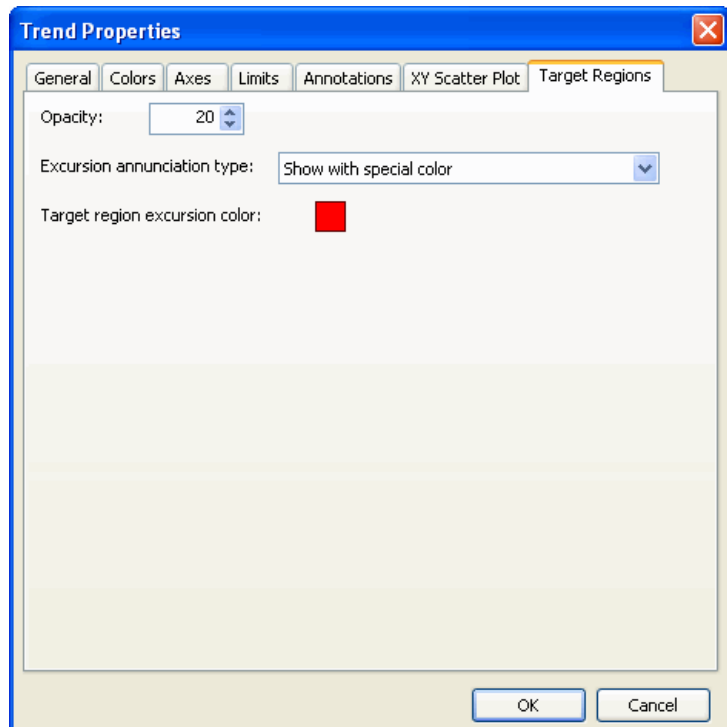
5 Click **OK**.

For information on the **XY Scatter Plot** tab, see *Configuring Scatter Plot Properties* on page 150.

Configuring Target Region Properties

To configure target region properties

- 1 On the **Chart** menu, click **Properties**. The **Trend Properties** dialog box appears.
- 2 Click the **Target Regions** tab.



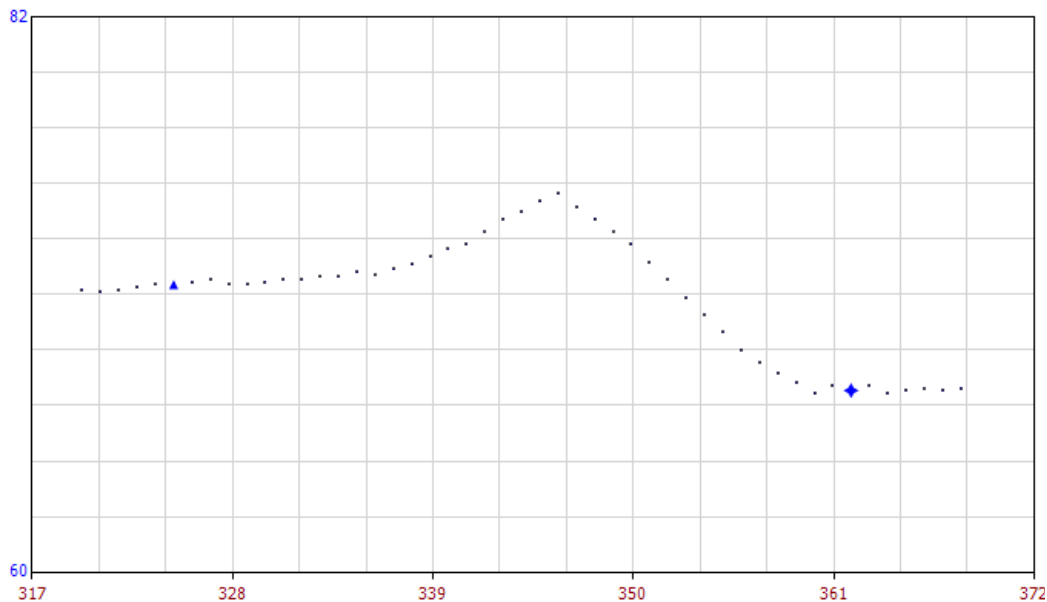
- 3 In the **Opacity** box, enter the opacity with which you want the target region to appear on the trend chart.

- 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:

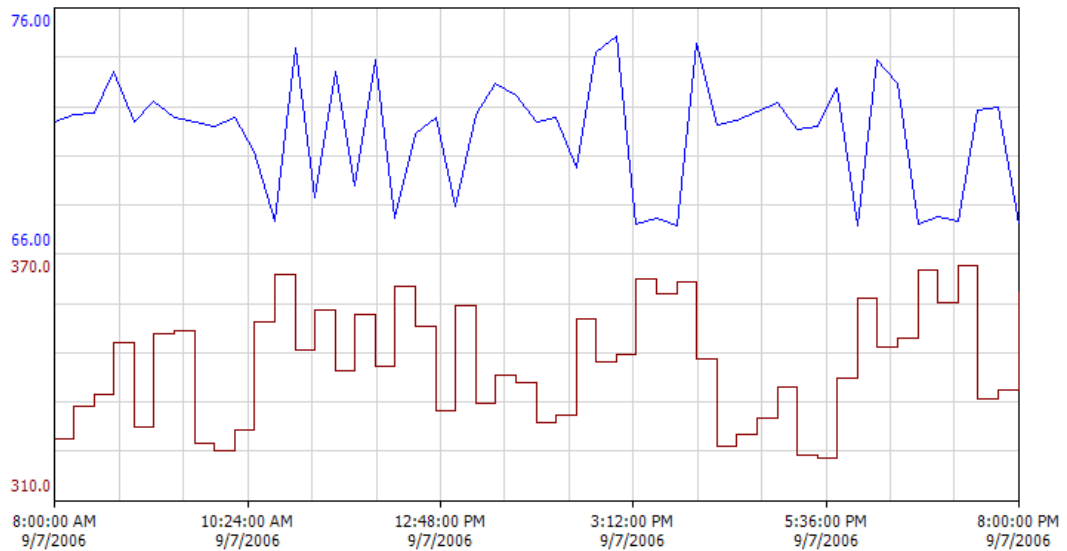


IEJNTEST:ProductGood[Full] vs IEJNTEST:ReactTemp[Full]

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?](#) on page 146.

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.



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?](#) on page 146

To configure a scatter plot

-  **1** Click the **New Chart** toolbar button. A new trend chart appears.
- 2** Do one of the following:
 - On the **Chart** menu, point to **Chart Type** and click **XY Scatter Plot**.
 -  • Click the **XY Scatter Plot** toolbar button.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](#) on page 40. 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. For more information, see [Time Picker](#) on page 47.
- 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](#) on page 67.
- 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:

- 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](#) on page 83. 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* on page 136. 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.

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.

	Good	Unknown	Uncertain	Bad
Good	Good	Unknown	Uncertain	Bad
Unknown	Unknown	Unknown	Uncertain	Bad
Uncertain	Uncertain	Uncertain	Uncertain	Bad
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** on page 96 and **Zooming** on page 99.

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](#) on page 70.

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

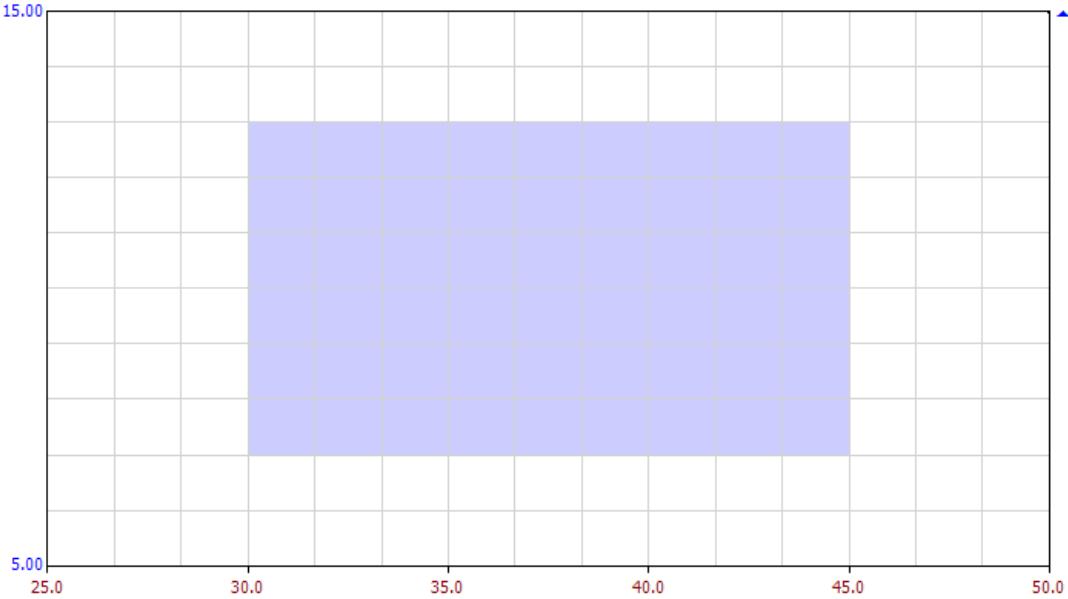
- ◆ Follow the procedure given under [To configure a target region for a trend tag](#) on page 73. The only difference is that when importing a CSV file or pasting clipboard data, each row must contain a region item that is composed of two items instead of three. The first item is the X value, the second 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

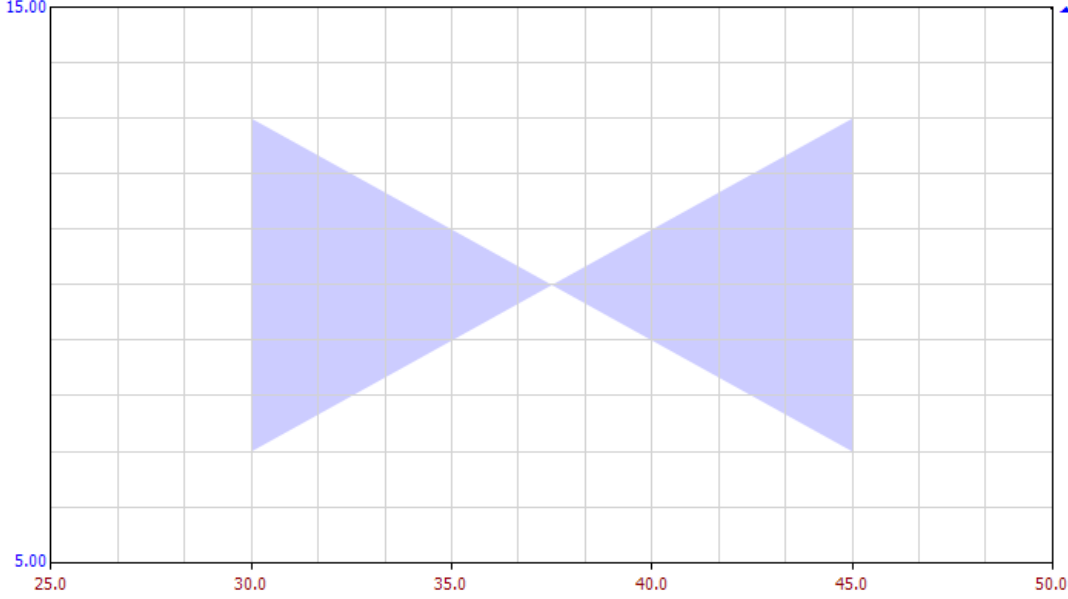
The resulting target region looks like this:



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:

X	Y
30	7
30	13
45	7
45	13

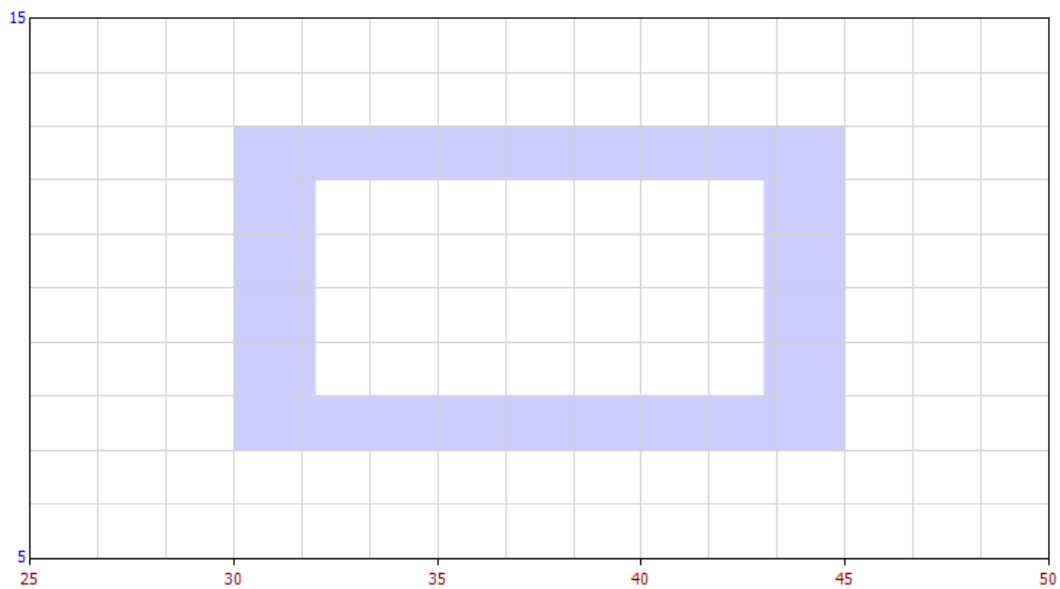
The resulting target region looks like this:



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

The resulting target region looks like this:



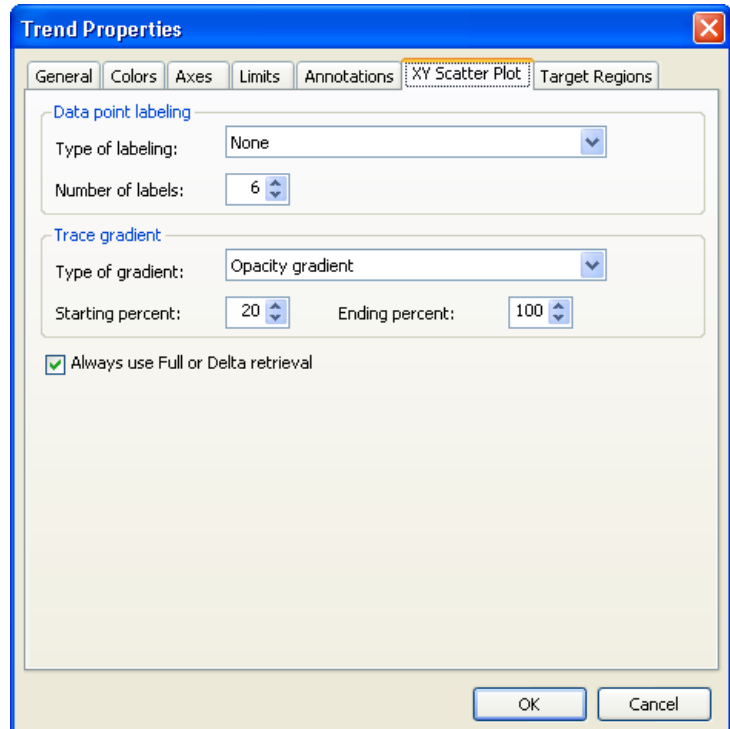
Configuring Scatter Plot Properties

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.



- 3 In the **Data point labeling** area, configure the following options:
 - **Type of labeling** Select **Time labels on current tag** to show time labels on the data points in the scatter plot. Select **None** if you do not want labels.
 - **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.)

- 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 Wonderware 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, .png, .jpeg, and .gif. You can also copy and paste the trend graph and associated Tag List to the Windows Clipboard.

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: On the **File** menu, click **Page Setup**.
- To preview the print output: On the **File** menu, click **Print Preview**.
- To print the chart: On the **File** menu, 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

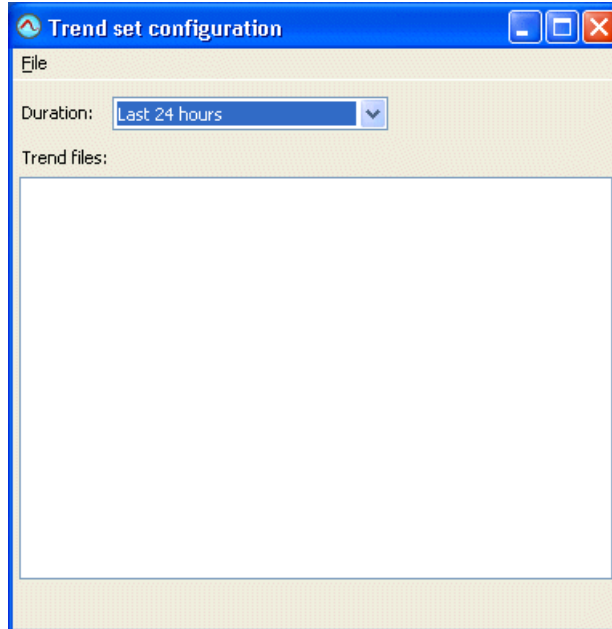
A trend set is a saved grouping of trend files. You can specify a common trend duration (for example, the last 24 hours) to apply to all of the files in the set.

This allows you to easily print information for the same duration from multiple trend files at the same time. Trend sets are only used for printing.

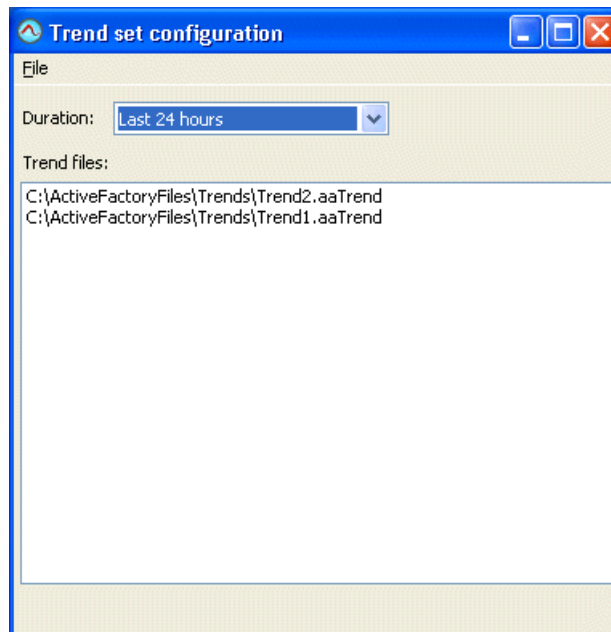
Creating a Trend Set

To create a trend set

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

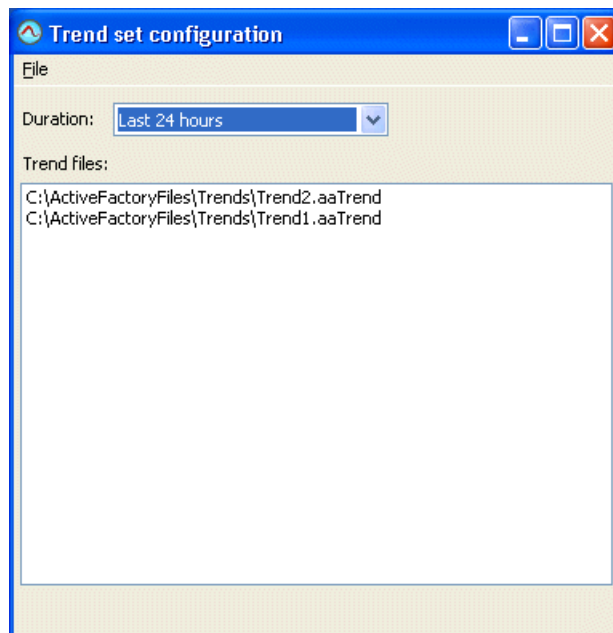


- 5 In the **Duration** list, select the time period for the trend set. The duration is applied to all of the files in the set. Data that is outside of the scope of the trend set duration is ignored.
- 6 To save the trend set, on the **File** menu, click **Save As**. The standard Windows **Save As** 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.



- 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**.

Printing a Trend Set

To print a trend set

- ◆ On the **File** menu, click **Print**.

If you double-click a trend set in Windows Explorer, the trend set opens in the Wonderware 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.

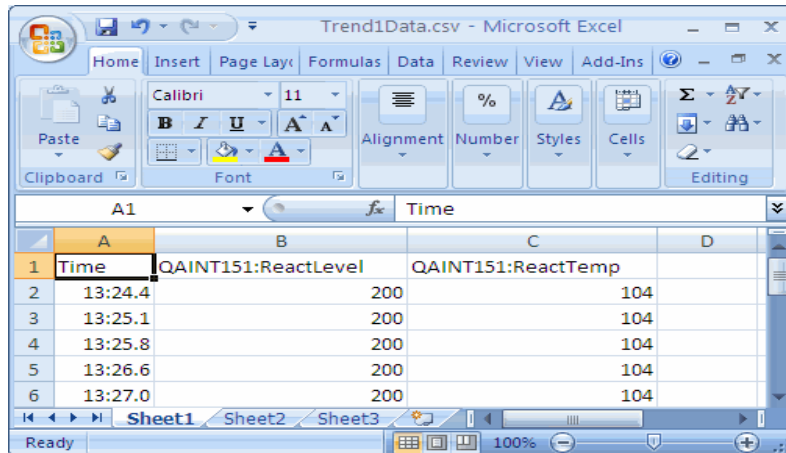
Saving Trend Data to a .CSV File

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 On the **File** menu, click **Save Data**.
The standard Windows **Save As** dialog box appears.
- 2 Browse to the location in which you want to save the file.
- 3 In the **File name** box, type a name for the trend data file.
- 4 Click **Save**.

You can view the data in any spreadsheet application that can open .csv files. For example, Microsoft Excel.



	A	B	C	D
1	Time	QAIN151:ReactLevel	QAIN151:ReactTemp	
2	13:24.4	200	104	
3	13:25.1	200	104	
4	13:25.8	200	104	
5	13:26.6	200	104	
6	13:27.0	200	104	

Saving the Trend Chart to an Image File

You can save a trend chart to a .bmp, .gif, .jpeg, .svg, or .png image file.

To save the trend chart

- 1 On the **File** menu, click **Save Image**.
The standard Windows **Save As** dialog box appears.
- 2 Browse to the location in which you want to save the file.
- 3 In the **File name** box, type a name for the trend image file.
- 4 In the **Save as Type** box, select an image type.
- 5 Click **Save**.

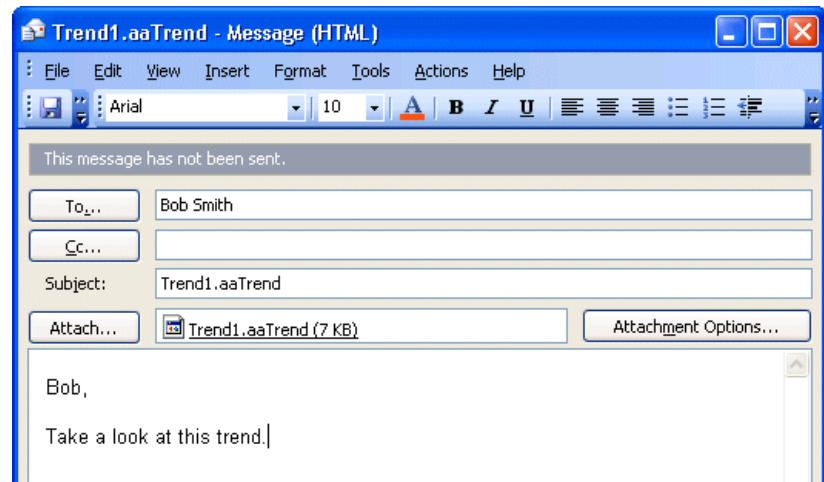
E-mailing a Trend File

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.

When you e-mail a trend, the actual trend file (.aaTrend) is attached.

To e-mail a trend file

- 1 On the **File** menu, point to **Send To** and then click **Mail Recipient**. The e-mail program starts up and a new message appears that includes the trend file as an e-mail attachment. For example:



- 2 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 On the **Edit** menu, click **Copy**.
- 3 Open the target application (for example, Microsoft Word).
- 4 Paste in the chart.

The trend chart is pasted as a graphic in the target application.

Publishing Trends to the Wonderware Information Server

You can publish trends to the Wonderware Information Server. When you publish a trend, the trend report information is stored in special tables in the Wonderware Historian, and the file is copied to a folder on the Wonderware Information Server. When you publish a trend, Wonderware 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, Wonderware 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, Wonderware 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 Wonderware Information Server, data is retrieved from the Wonderware Historian database and appears in the trend chart.

The Wonderware Information Server must be associated with the same Wonderware Historian as the trend you want to publish. If a trend references multiple Wonderware Historians, the Wonderware Information Server must be associated with at least one of the Wonderware 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.

The **Report name** box shows the name of the trend report as it appears on the Wonderware 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 Wonderware Historian on which to store the report publishing information.
- 4 In the **Report site** list, select the URL of the Wonderware Information Server to which you want to publish the trend.

The Wonderware Information Server may or may not be physically located on the Wonderware 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 Wonderware Information Server periodically scans the publishing folders for changes. A short delay may occur prior to the published report being shown on the Wonderware Information Server.

- 8 To view the Wonderware 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.

The **Report name** box shows the name of the trend report as it appears on the Wonderware 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 Wonderware Historian on which to store the report publishing information.

- 4 In the **Report site** list, select the URL of the Wonderware Information Server to which you want to publish the trend.

The report site may or may not be physically located on the Wonderware 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 Wonderware 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 Wonderware 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 Wonderware Information Server, click **Browse**. Otherwise, click **Done**.

Using Trend with a Tablet PC

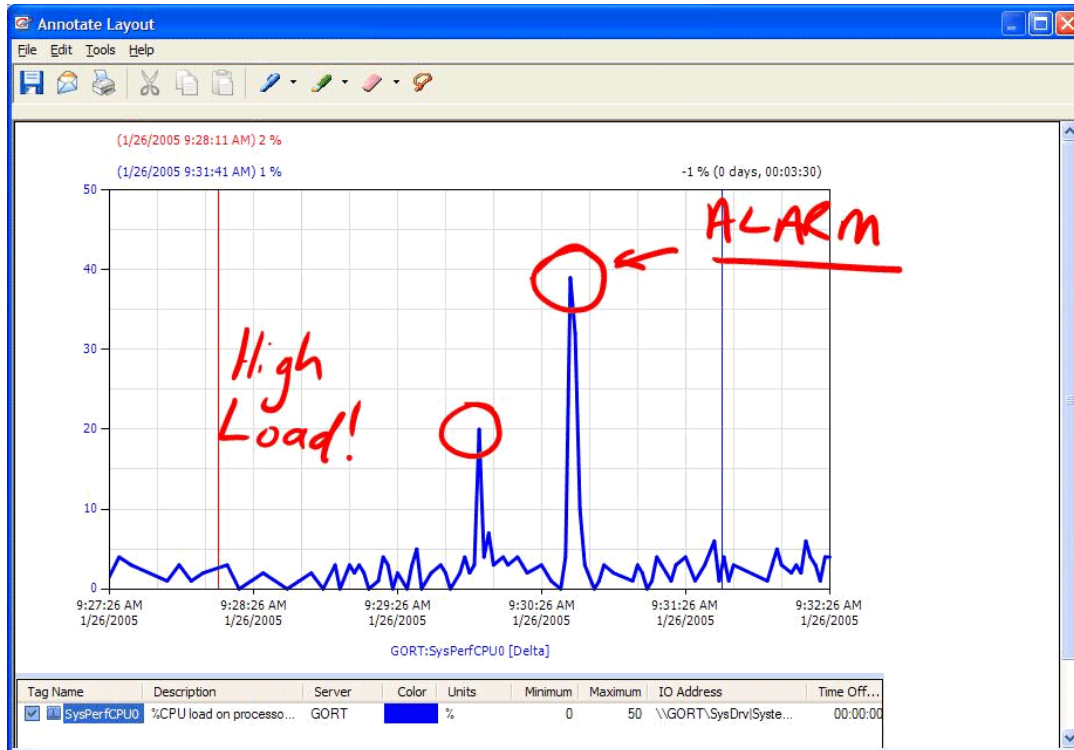
When you run the Trend application on a Tablet PC, you can capture an image of the application window or the chart area, annotate the image using various drawing tools, and save, print or e-mail the results.

Annotating a Chart

To annotate a chart or the application window

- 1 Create a trend chart.
- 2 On the **Tools** menu, click **Annotate Chart** or **Annotate Application**. The **Annotate Layout** dialog box appears.




- Use the stylus to write annotations on the image. For more information, see Making Chart Annotations.



- Save or e-mail the file. For more information, see Saving, Printing, and E-Mailing an Annotated Chart on page 163.

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 On the **File** menu, click **Save**. A standard Windows **Save As** dialog box appears.
- 2 Type a name and format for the file and click **OK**.

To print an annotated chart

- 1 On the **File** 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 You only need to configure the e-mail server one time. If you have already done this, go to step 3.

- 1 On the **Edit** menu, click **E-Mail Configuration**. The **E-Mail Configuration** dialog box appears.
- 2 Type the host name of the SMTP e-mail server to use for sending e-mail. If you are unsure, ask your administrator for assistance. Click **OK**.
- 3 On the **File** menu, click **E-Mail**. The **New Message** dialog box appears.
- 4 Type sender and recipient addresses and write a message. An image file of the annotated trend is automatically added as an attachment.
- 5 Click **Send** to send the e-mail.

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**.

Chapter 4

Wonderware Historian Client Query

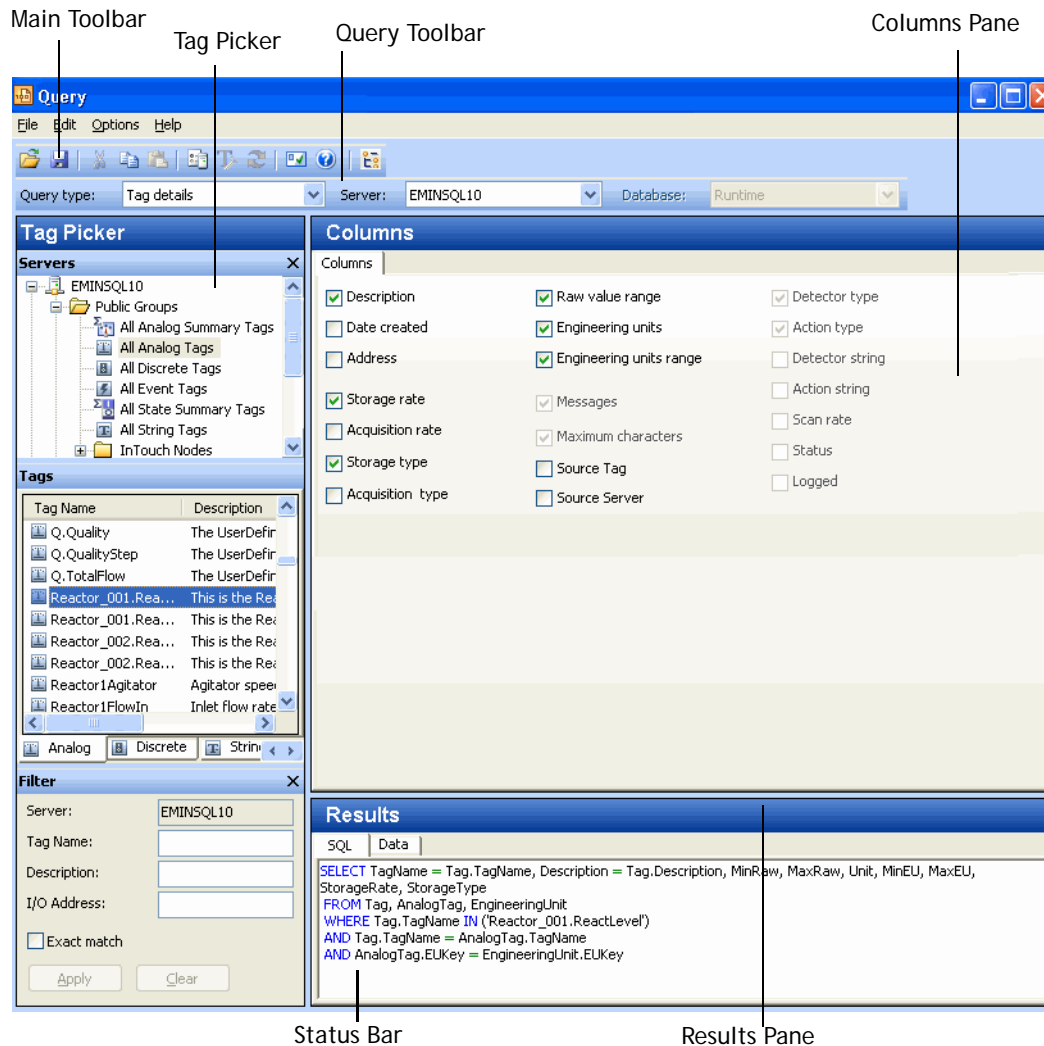
The Wonderware Historian Client Query is an application that allows you to retrieve data from a Wonderware Historian database or any SQL Server database and return the results in a table format. If you are querying a Wonderware Historian, you can choose from a number of predefined query types and easily 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 on page 27.

After you establish a connection with the server, the Query program appears, showing the main window:



The Query application user interface consists of the following components:

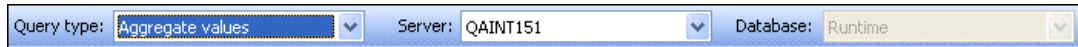
- Main Toolbar
- Query Toolbar
- Tag Picker (may not appear depending on the query type)
- Columns Pane
- Results Pane
- Status Bar

For information on using the common toolbars and the Tag Picker, see Chapter 2, Common Client Components.

Query Toolbar

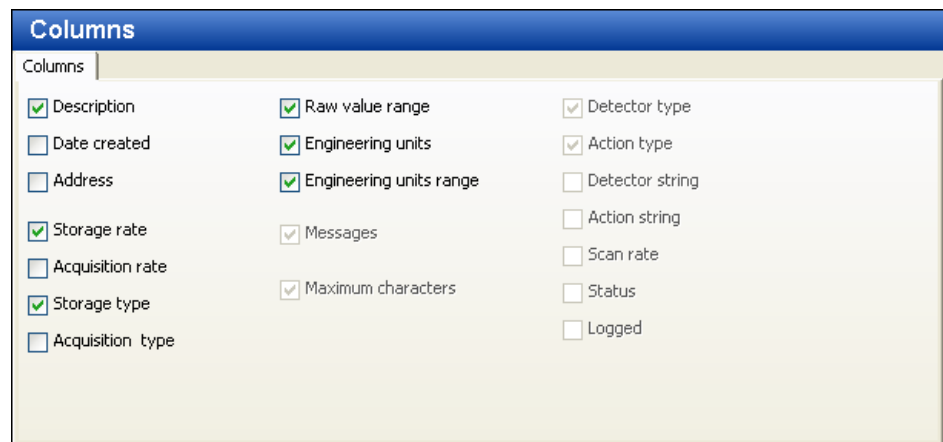
Use the query toolbar to select the query type, server, and database for the query.

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.

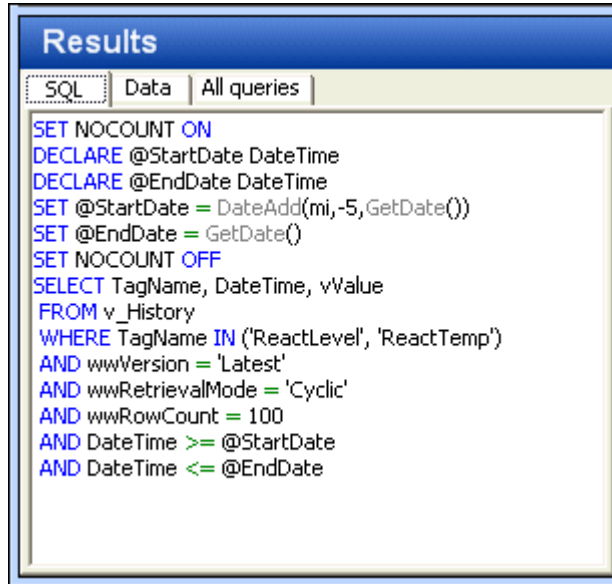


Results Pane

Use the **Results** pane to view the results of the query that you have created. The **Results** pane includes three tabs:

- SQL tab
- Data tab
- All Queries tab

The SQL tab shows the actual SQL statement that is sent to the server.



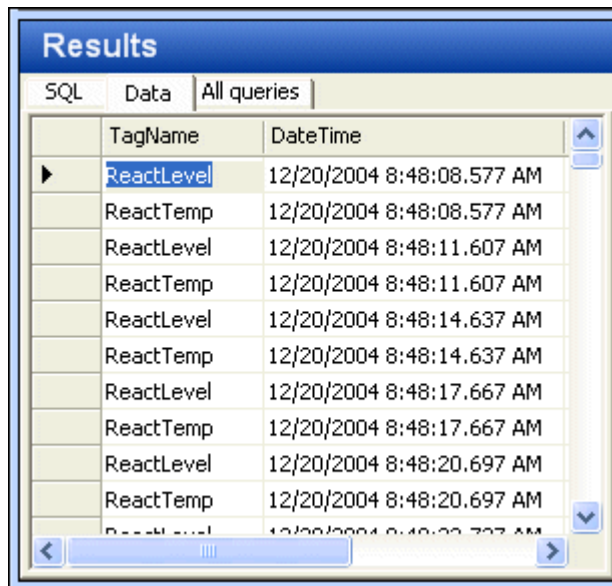
The screenshot shows a window titled "Results" with three tabs: "SQL", "Data", and "All queries". The "SQL" tab is selected, displaying the following SQL query:

```

SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-5,GetDate())
SET @EndDate = GetDate()
SET NOCOUNT OFF
SELECT TagName, DateTime, vValue
FROM v_History
WHERE TagName IN ('ReactLevel', 'ReactTemp')
AND wwVersion = 'Latest'
AND wwRetrievalMode = 'Cyclic'
AND wwRowCount = 100
AND DateTime >= @StartDate
AND DateTime <= @EndDate

```

The Data tab shows the data returned by the SQL statement.



The screenshot shows the "Results" window with the "Data" tab selected. It displays a table with two columns: "TagName" and "DateTime". The table contains 12 rows of data, alternating between "ReactLevel" and "ReactTemp" tags, with timestamps from 12/20/2004 8:48:08.577 AM to 12/20/2004 8:48:20.697 AM.

TagName	DateTime
ReactLevel	12/20/2004 8:48:08.577 AM
ReactTemp	12/20/2004 8:48:08.577 AM
ReactLevel	12/20/2004 8:48:11.607 AM
ReactTemp	12/20/2004 8:48:11.607 AM
ReactLevel	12/20/2004 8:48:14.637 AM
ReactTemp	12/20/2004 8:48:14.637 AM
ReactLevel	12/20/2004 8:48:17.667 AM
ReactTemp	12/20/2004 8:48:17.667 AM
ReactLevel	12/20/2004 8:48:20.697 AM
ReactTemp	12/20/2004 8:48:20.697 AM
ReactLevel	12/20/2004 8:48:20.697 AM

The **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, click **All queries** on the **Options** menu.

```

Results
SQL | Data | All queries
SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-5,GetDate())
SET @EndDate = GetDate()
SET NOCOUNT OFF
SELECT DateTime, Value = v_History.vValue
FROM v_History
WHERE v_History.TagName IN ('$AccessLevel')
AND v_History.vValue IS NOT NULL
AND Quality = 0
AND wwVersion = 'Latest'
AND wwRetrievalMode = 'Cyclic'
AND wwRowCount = 100
AND DateTime >= @StartDate
AND DateTime <= @EndDate

SELECT * FROM IOserver
WHERE IOserver.ComputerName IN ('QAIN151')

SET NOCOUNT ON
DECLARE @StartDate DateTime
DECLARE @EndDate DateTime
SET @StartDate = DateAdd(mi,-5,GetDate())
SET @EndDate = GetDate()

```

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 Wonderware Application Server* on page 23.

To view the hierarchical names in a query

- ◆ Do one of the following:
 - On the **Options** menu, click **Use Hierarchical Name**.
 - Click the **Use hierarchical name** toolbar button.
 - 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 Wonderware 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 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. 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, 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 **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:

- You are connected to the IndustrialSQL Server 9.0.2
- 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.

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 Wonderware 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** on page 39.

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 Do one of the following:
 - On the **File** menu, click **Open**.



- Click the **Open File** toolbar button.

The standard Windows **Open** dialog box appears.

- 2 Browse to and select the query file to open. All query files have the **.sql** extension.
- 3 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 Do one of the following:

- On the **File** menu, click **Save**.
- Click the **Save File** toolbar button.



The standard Windows **Save As** dialog box appears.

2 In the **Save As** dialog box, type a name for the query.

You can select to save the query as a SQL script file (the) or as a text file.

3 Click **OK**.

Creating a Query

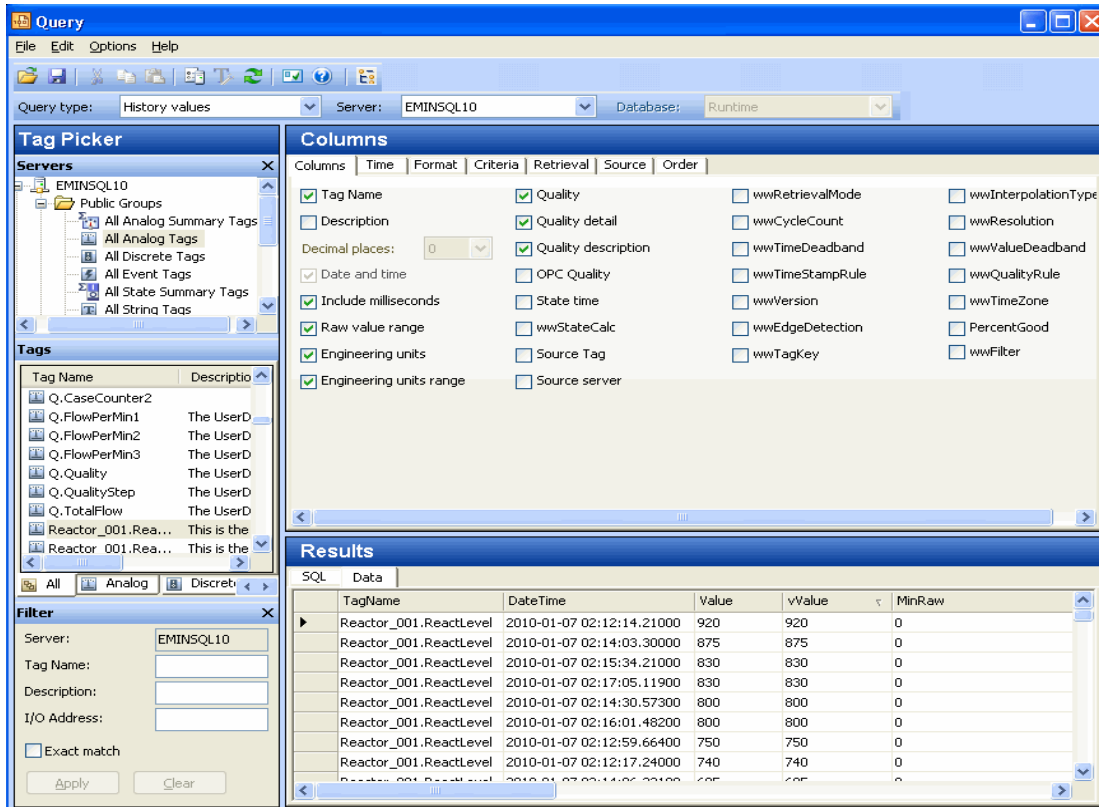
When you configure a query, you must select the tag or tags for which you want to retrieve data, the type of query, and the server(s) from which to retrieve the data. The data is queried from the database to which you are currently logged on. You can also configure additional parameters that are specific 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 In the **Query type** list in the toolbar, click the name of the type of query you want to use as a starting point. For more information on the possible types, see **Query Types** on page 175.
- 2 In the **Server** list, click the name of the server from which you want to retrieve data.

- Use the Tag Picker to find tags in the Wonderware Historian database that you want to include in your query. For more information on the Tag Picker, see Tag Picker on page 40.



- 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 on page 175.
- 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
- Query Type: Alarm Limits
- Query Type: Analog Summary Values
- Query Type: Annotations
- Query Type: Custom
- Query Type: Event History Values
- Query Type: Event Snapshot
- 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: State Summary Values
- Query Type: Storage
- Query Type: Storage Size Available
- Query Type: Storage Start Date
- Query Type: Summary Values
- 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 retrieve aggregated values from the Wonderware Historian's summary tables, use the Summary Values query type. For more information, see [Query Type: Summary Values](#) on page 206.

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](#) on page 213.
 - See [Time Tab](#) on page 212.
 - See [Criteria Tab](#) on page 176.
 - See [Calculations Tab](#) on page 178.
 - See [Retrieval Tab](#) on page 214.
 - See [Source Tab](#) on page 215.
- 4 To view the results, click the **Data** tab in the **Results** pane.

	TagName	Average
▶	ProdLevel	6515.73
	ReactLevel	1215.9

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.

Value >= 0
 and < 0
 Value contains 0
 Value not null
 Quality: Not used
 Criteria applicability: Not used

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).

- **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.
- **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

You can query the database 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 on page 179.
 - See **Time** Tab on page 212.
 - See **Alarm Limits** tab on page 180.
 - See **Retrieval** Tab on page 214.
 - See **Source** Tab on page 215.
 - See **Order** Tab on page 216.
- 4 To view the results, click the **Data** tab in the **Results** pane.

	Description	TagName	DateTime	Value	Quality
▶	Reactor level	ReactLevel	06 Oct 2004 12:04:28:787	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:29:393	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:28:180	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:26:970	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:27:577	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:26:363	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:25:150	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:25:757	2000	0
	Reactor level	ReactLevel	06 Oct 2004 12:04:24:547	2000	0

Columns tab

Use the **Columns** tab to configure the columns that are returned for the results.

<input checked="" type="checkbox"/> Tag name	<input checked="" type="checkbox"/> Quality
<input type="checkbox"/> Description	<input checked="" type="checkbox"/> Quality detail
Decimal places: <input type="text" value="0"/> ▼	<input checked="" type="checkbox"/> Quality description
<input checked="" type="checkbox"/> Date and time	<input type="checkbox"/> OPC Quality
<input checked="" type="checkbox"/> Include milliseconds	

Options are as follows:

- **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions on page 24.
- **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 Wonderware 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.
- **Quality detail:** The internal representation of data quality.
- **Quality description:** The text string that describes the quality detail value.
- **OPC Quality:** The quality value received from the data source.

Alarm Limits tab

Use the **Alarm Limits** tab to filter the alarm history values.

TagName	Name	Value
ReactLevel	Lo	200
ReactLevel	Hi	1800
ReactTemp	Hi	100

Context: InTouch

Value not null

Quality: Good

To configure alarm limits

- 1 Select the **Use alarm limits** check box to filter the alarm history according to a selected limit.
- 2 In the **Context** list, click the name of the context to which the alarm limit belongs. For example, the alarm limit can be valid within the context of an InTouch application.

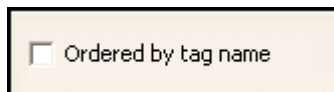
- 3 In the table, select the row that contains the limit you want to apply. The columns in the window are:
 - **TagName:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
 - **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.
 - **Unit:** The unit of measure. For example mph, grams, and pounds.
 - **LimitType:** The type of limit; that is, whether it is a rising (up) or falling (down) limit. 0 = Rising; 1 = Falling.
- 4 Select the **Value not null** check box to only return values that are not NULL.
- 5 In the **Quality** list, select the type of quality for which you want to return results. Quality values are Good (0), Bad (1), and Doubtful (16). If you do not want to filter on quality, select Not used.

Query Type: Alarm Limits

You can view the alarm limits for a tag. For example, the Hi or Lo alarm limit for an analog tag.

To view alarm limits

- 1 In the **Query Type** list in the toolbar, click **Alarm limits**.
- 2 If you want to only retrieve annotations for particular tag(s), use the Tag Picker to select one or more tags.
- 3 In the **Columns** pane, click the **Alarm limits** tab.



- 4 Select the **Ordered by tag name** check box to order the results in alphabetical order by tagname.
- 5 To view the results, click the **Data** tab in the **Results** pane.

	TagName	Name	Value	Unit
▶	ProdLevel	Hi	7000	M3
	ReactLevel	Lo	200	M3
	ReactLevel	Hi	1800	M3
	ReactTemp	Hi	100	°C
	ReactTemp	HiHi	180	°C

The columns in the result set are as follows:

- **TagName:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions on page 24.
- **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.
- **Unit:** The unit of measure. For example mph, grams, and pounds.

Query Type: Analog Summary Values

You can retrieve the summary data of analog or analog summary tags. The summary data includes 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* on page 56.

To view analog summary values

- 1 In the **Query Type** list in the toolbar, click **Analog summary values**.
- 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** on page 183.
 - See **Time Tab** on page 212.
 - See **Retrieval Tab** on page 214.

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.

<input checked="" type="checkbox"/> Tag Name	<input checked="" type="checkbox"/> OPC Quality	<input checked="" type="checkbox"/> Minimum	<input checked="" type="checkbox"/> Integral	<input type="checkbox"/> wwVersion
<input type="checkbox"/> Description	<input type="checkbox"/> PercentGood	<input type="checkbox"/> MinDateTime	<input type="checkbox"/> ValueCount	
<input type="checkbox"/> SourceTag	<input type="checkbox"/> First	<input checked="" type="checkbox"/> Maximum	<input type="checkbox"/> wwCycleCount	
<input type="checkbox"/> SourceServer	<input type="checkbox"/> FirstDateTime	<input type="checkbox"/> MaxDateTime	<input type="checkbox"/> wwResolution	
<input type="checkbox"/> StartDateTime	<input type="checkbox"/> Last	<input checked="" type="checkbox"/> Average	<input type="checkbox"/> wwTimeZone	
<input type="checkbox"/> EndDateTime	<input type="checkbox"/> LastDateTime	<input checked="" type="checkbox"/> Standard Deviation	<input type="checkbox"/> wwRetrievalMode	

The following are the column options:

- **Tag Name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information see, ArchestrA Naming Conventions on page 24.
- **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.
- **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) on page 755.
- **wwResolution:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode. For more information, see Resolution (Values Spaced Every X ms) (wwResolution) on page 757.
- **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 on page 689.
- **wwVersion:** The version of data to be used if the original data value is changed. For more information, see History Version (wwVersion) on page 769.

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** on page 185.
 - See **Time Tab** on page 212.
- 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 12:07:10.147
	ReactTemp	10/6/2004 12:04:44.617 PM	Check out the cause of this	public	10/6/2004 12:07:43.570
	ProdLevel	10/6/2004 12:04:46.953 PM	There may be a problem	dbo	10/6/2004 12:07:57.710

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.

<input type="checkbox"/> Date and time	<input type="checkbox"/> Public
<input type="checkbox"/> Date created	<input type="checkbox"/> All users
<input type="checkbox"/> Content	
<input type="checkbox"/> User name	<input type="checkbox"/> All tags on server

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 Wonderware 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 shipped with Microsoft SQL Server.)

```
USE Northwind
SELECT * FROM Employees
WHERE City = 'London';
```

Query Type: Event History Values

You can view all the events that occurred for specified event tags.

To view event history

- 1 In the **Query Type** list in the toolbar, click **Event history**.
- 2 Use the Tag Picker to select one or more event tags.
- 3 In the **Columns** pane, click on each tab and configure the parameters for the query.
 - See **Columns Tab** on page 187.
 - See **Time Tab** on page 212.
 - See **Order Tab** on page 216.

4 To view the results, click the **Data** tab in the **Results** pane.

	TagName	Description	DateTime	DetectDateTime
▶	SysStatusEvent	Status Tag snapshot event	10/8/2004 11:00:00.000 AM	10/8/2004 11:00:33.4
	SysStatusEvent	Status Tag snapshot event	10/8/2004 12:00:00.000 PM	10/8/2004 12:00:33.4
	SysStatusEvent	Status Tag snapshot event	10/8/2004 1:00:00.000 PM	10/8/2004 1:00:33.35
	SysStatusEvent	Status Tag snapshot event	10/8/2004 2:00:00.000 PM	10/8/2004 2:00:33.39
	SysStatusEvent	Status Tag snapshot event	10/8/2004 3:00:00.000 PM	10/8/2004 3:00:33.34
	SysStatusEvent	Status Tag snapshot event	10/8/2004 4:00:00.000 PM	10/8/2004 4:00:33.31
	SysStatusEvent	Status Tag snapshot event	10/8/2004 5:00:00.000 PM	10/8/2004 5:00:33.29
	SysStatusEvent	Status Tag snapshot event	10/8/2004 6:00:00.000 PM	10/8/2004 6:00:33.28
	SysStatusEvent	Status Tag snapshot event	10/8/2004 7:00:00.000 PM	10/8/2004 7:00:33.41
	SysStatusEvent	Status Tag snapshot event	10/8/2004 8:00:00.000 PM	10/8/2004 8:00:33.4

Columns Tab

Use the **Columns** tab to configure the columns to show in the results.

To configure the columns

1 Select the columns to show in the results:

- **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
- **Description:** The description of the tag.
- **Date and time:** The timestamp reflecting when the event history data was acquired. This is the time for when the event actually occurred.
- **Include milliseconds:** Used to include milliseconds in the timestamp.
- **Detect date time:** The timestamp reflecting when the event was detected by the event system.

- 2 Configure how to filter the results:
 - **Limit to XX rows:** The number of initial consecutive rows to return out of the total number of rows in the record set, starting with the first row in the record set. For example, if there are a total of 150 rows, and you set this value to 100, only the first 100 rows in the records set will be returned.

Query Type: Event Snapshot

You can view the data values for selected analog, discrete, or string tags that have the same timestamp as a detected event. This provides you with a “snapshot” of selected data values at the time of an event.

To view event snapshot information

- 1 In the **Query Type** list in the toolbar, click **Event snapshot**.
- 2 Use the **Tag Picker** to select one or more event tags that have a snapshot event action.
- 3 In the **Columns** pane, click on each tab and configure the parameters for the query.
 - See **Tag Set Tab** on page 188.
 - See **Columns Tab** on page 189.
 - See **Time Tab** on page 212.
 - See **Order Tab** on page 216.
- 4 To view the results, click the **Data** tab in the **Results** pane.

	EventTagName	TagName	Description	DateTime	DetectDateTim
▶	ReactLevelSnapshot	ReactTemp	Reactor temp	10/15/2004 12:52:17.167 P	10/15/2004 12:
	ReactLevelSnapshot	ReactTemp	Reactor temp	10/15/2004 12:53:23.333 P	10/15/2004 12:
	ReactLevelSnapshot	ReactTemp	Reactor temp	10/15/2004 12:54:29.560 P	10/15/2004 12:
	ReactLevelSnapshot	ReactTemp	Reactor temp	10/15/2004 12:55:35.723 P	10/15/2004 12:

Tag Set Tab

Use the **Tag Set** tab to select the tag(s) for which the data values are stored as a snapshot. (This is not the event tag.)

Snapshot tag type: Analog

Snapshot tags:

Tag Name	Description
ReactTemp	Reactor temp

To configure the tag set

- 1 In the **Snapshot tag type** list, click the type of snapshot, either Analog, Discrete, or String. The **Snapshot tags** window shows all of the snapshot tags for the type you have selected.
- 2 In the **Snapshot tags** window, select the snapshot tag.

Columns Tab

Use the **Columns** tab to configure the columns to show in the results.

The screenshot shows the 'Columns Tab' configuration window. It contains the following elements:

- Tag name
- Description
- Date and time
- Decimal places:
- Detect date time
- Quality
- Limit to: rows
- Quality detail
- Quality description
- Include milliseconds
- Quality:

To configure the columns

- 1 Select the columns to show in the results:
 - **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
 - **Description:** The description of the tag.
 - **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 Wonderware 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.
 - **Decimal places:** The number of decimal places to show for the data value of the currently selected tag. This applies only to analog tags.
 - **Detect date time:** The timestamp reflecting when the event was detected by the event system.
 - **Quality:** The basic data quality indicator associated with the data value.
 - **Quality detail:** The internal representation of data quality.

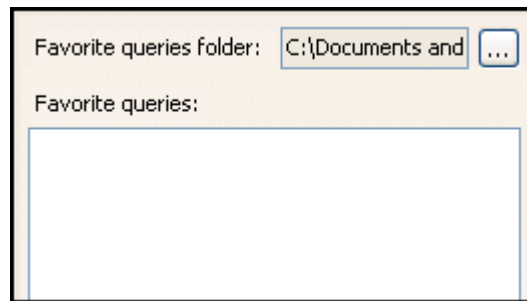
- **Quality description:** The text string that describes the quality detail value.
- 2 Configure how to filter the results:
 - **Limit to XX rows:** The number of initial consecutive rows to return out of the total number of rows in the record set, starting with the first row in the record set. For example, if there are a total of 150 rows, and you set this value to 100, only the first 100 rows in the records set will be returned.
 - **Quality:** The type of quality for which you want to return results. Quality values are Good (0), Bad (1), and Doubtful (16). If you do not want to filter on quality, select Not used.

Query Type: Favorites

You can load a saved SQL query file (.sql) and execute it against the database.

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 on page 194.

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 on page 191.
 - See Time Tab on page 212.
 - See Format Tab on page 213.
 - See Criteria Tab on page 194.
 - See Retrieval Tab on page 195.
 - See Source Tab on page 215.
 - See Order Tab on page 216.
- 4 Click the **Data** tab in the **Results** pane to view results.

TagName	DateTime	Value	vValue
Reactor_001.ReactTemp	2010-01-19 02:54:26.60700	141	141
Reactor_001.ReactTemp	2010-01-19 02:54:29.63700	161	161
Reactor_001.ReactTemp	2010-01-19 02:54:32.66700	181	181
Reactor_001.ReactTemp	2010-01-19 02:54:35.69700	193.699996948242	193.69999694824219

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.

<input checked="" type="checkbox"/> Tag Name	<input checked="" type="checkbox"/> Quality	<input type="checkbox"/> wwRetrievalMode	<input type="checkbox"/> wwInterpolationType
<input type="checkbox"/> Description	<input checked="" type="checkbox"/> Quality detail	<input type="checkbox"/> wwCycleCount	<input type="checkbox"/> wwResolution
Decimal places: <input type="text" value="0"/>	<input checked="" type="checkbox"/> Quality description	<input type="checkbox"/> wwTimeDeadband	<input type="checkbox"/> wwValueDeadband
<input checked="" type="checkbox"/> Date and time	<input type="checkbox"/> OPC Quality	<input type="checkbox"/> wwTimeStampRule	<input type="checkbox"/> wwQualityRule
<input checked="" type="checkbox"/> Include milliseconds	<input type="checkbox"/> State time	<input type="checkbox"/> wwVersion	<input type="checkbox"/> wwTimeZone
<input checked="" type="checkbox"/> Raw value range	<input type="checkbox"/> wwStateCalc	<input type="checkbox"/> wwEdgeDetection	<input type="checkbox"/> PercentGood
<input checked="" type="checkbox"/> Engineering units	<input type="checkbox"/> Source Tag	<input type="checkbox"/> wwTagKey	<input type="checkbox"/> wwFilter
<input checked="" type="checkbox"/> Engineering units range	<input type="checkbox"/> Source server		

Note Some of the columns are not available when you select **Wide query format** on the **Format** tab.

The following are the column options:

- **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
- **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 Wonderware 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) on page 786.
- **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](#) on page 689.
- **wwCycleCount:** The number of retrieval cycles (sub-intervals) for the specified time period. For more information, see [Cycle Count \(X Values over Equal Time Intervals\) \(wwCycleCount\)](#) on page 755.
- **wwTimeDeadband:** The time deadband used in data retrieval. For more information, see [Time Deadband \(wwTimeDeadband\)](#) on page 762.
- **wwTimeStampRule:** The time stamp rule used in data retrieval. For more information, see [Time stamp Rule \(wwTimestampRule\)](#) on page 774.
- **wwVersion:** The version of data to be used if the original data value is changed. For more information, see [History Version \(wwVersion\)](#) on page 769.
- **wwEdgeDetection:** The type of edge detection used in the query.
- **wwTagKey:** The unique identifier of the tag in the Wonderware Historian server.
- **InterpolationType:** The interpolation type used to calculate the value. For more information, see [Interpolation Type \(wwInterpolationType\)](#) on page 771.
- **wwResolution:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode. For more information, see [Resolution \(Values Spaced Every X ms\) \(wwResolution\)](#) on page 757.
- **wwValueDeadband:** The value deadband used in data retrieval. For more information, see [Value Deadband \(wwValueDeadband\)](#) on page 766.
- **wwQualityRule:** The quality rule used in data retrieval. For more information, see [Quality Rule \(wwQualityRule\)](#) on page 778.
- **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.
- **wwFilter:** The analog filter used to filter data during retrieval. For more information, see [Analog Value Filtering \(wwFilter\)](#) on page 788.

Criteria Tab

Use the **Criteria** tab to specify the filtering criteria for the data value(s) to be returned.

The screenshot shows the 'Criteria Tab' interface with the following elements:

- First **Value** check box: Value, operator: \geq , value: 0
- and** check box: and, operator: $<$, value: 0
- Third **Value** check box: Value, operator: contains, value: 0
- Use StringHistory** check box: Use StringHistory
- Value not null** check box: Value not null
- Quality:** dropdown menu: Not used
- Criteria applicability:** dropdown menu: Not used

To configure criteria to retrieve data values

- 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. 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.

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 a Wonderware Historian in different ways.

For more information on the retrieval options, see [Understanding Retrieval Modes](#) on page 689 and [Understanding Retrieval Options](#) on page 752.

- 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 other data 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) on page 769.
- 2 In the **Rules** area, do the following:
 - In the **Timestamp** list, click the required timestamp value. For more information on the Time stamp rule, see Time stamp Rule (wwTimestampRule) on page 774.
 - 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 Wonderware Historian level.

For more information on each option, see Quality Rule (wwQualityRule) on page 778.

- 3 In the **State retrieval** area, do the following:
 - In the **State calculation** list, click the state calculation.
 - Select the **Specify state** check box to set the value of the state. For example, you can specify either an 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) on page 786.

- 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) on page 788.

- 5 In the **Phantom cycle** area, select the **Do not include boundary values** check box to remove boundary values from the result.

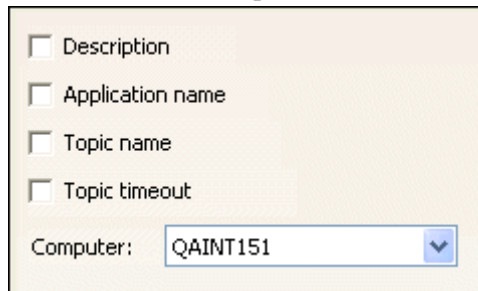
For more information on Phantom cycle filter, see About “Phantom” Cycles on page 760.

Query Type: IO Server

You can retrieve basic configuration information for all I/O Servers configured for use with the Wonderware 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.



The screenshot shows a configuration dialog box with a light beige background. It contains four unchecked checkboxes arranged vertically: "Description", "Application name", "Topic name", and "Topic timeout". Below these checkboxes is a "Computer:" label followed by a dropdown menu. The dropdown menu is currently displaying the text "QAIN151" and has a small downward-pointing arrow on its right side.

- 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.
 - **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 Wonderware 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, click on each tab and configure the parameters for the query.
 - See **Columns Tab** on page 199.
 - See **Time Tab** on page 212.
- 4 To view the results, click the **Data** tab in the **Results** pane.

	TagName	DateTime	Value	MinRaw	MaxRaw	MinEU	MaxEU	Unit
▶	ReactTemp	2009-07-27 05:47:18.843	131.600006103516	0	220	0	220	°C

Columns Tab

Use the **Columns** tab to configure the columns to show in the results.

The screenshot shows the following configuration options:

- Tag Name
- Description
- Decimal places:
- Date and time
- Include milliseconds
- Raw value range
- Engineering units
- Engineering units range
- Quality
- Quality detail
- Quality description
- OPC Quality
- Source Tag
- Source server

Options are as follows:

- **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see **ArchestrA Naming Conventions** on page 24.
- **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.

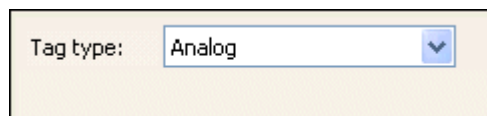
- **Date and time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the Wonderware Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
- **Include milliseconds:** Used to include milliseconds 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.
- **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 Wonderware 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.



A screenshot of a software interface showing a dropdown menu. The label "Tag type:" is on the left. The dropdown box contains the text "Analog" and a small downward-pointing arrow on the right side.

- 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.

	Analog Tags
▶	119

Query Type: Server Version

You can retrieve the version number for the currently selected Wonderware 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.

	StringValue
▶	8,0,300,0830

Query Type: State Summary Values

You can retrieve the summary data of state summary, analog, discrete or string tags. The summary data includes 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* on page 56.

To view state summary values

- 1 In the **Query Type** list in the toolbar, click **State summary values**.
- 2 Use the **Tag Picker** to select one or more state summary, analog, discrete or string tags.
- 3 In the **Columns** pane, click each tab and configure the parameters for the query.
 - See *Columns Tab* on page 202.
 - See *Time Tab* on page 212.
 - See *Retrieval Tab* on page 214.
- 4 To view the results, click the **Data** tab in the **Results** pane.

	TagName	Value	vValue	OPCQuality	StateCount	StateTimeMin	StateTimeMax	StateTimeAvg	StateTimeTotal
▶	AvgRegular	25	25	192	1	762	762	762	762
	AvgRegula	45	45	192	1	841	841	841	841
	AvgRegula	5	5	192	1	890	890	890	890

Columns Tab

Use the **Columns** tab to select which columns to include in the query results.

<input checked="" type="checkbox"/> Tag Name	<input checked="" type="checkbox"/> State Count	<input checked="" type="checkbox"/> Average	<input type="checkbox"/> wwMaxState
<input type="checkbox"/> Description	<input type="checkbox"/> Contained State Count	<input type="checkbox"/> AverageContained	<input type="checkbox"/> wwCycleCount
<input type="checkbox"/> SourceTag	<input checked="" type="checkbox"/> Minimum	<input checked="" type="checkbox"/> Total	<input type="checkbox"/> wwResolution
<input type="checkbox"/> SourceServer	<input type="checkbox"/> MinContained	<input type="checkbox"/> TotalContained	<input type="checkbox"/> wwTimeZone
<input type="checkbox"/> StartDateTime	<input checked="" type="checkbox"/> Maximum	<input checked="" type="checkbox"/> Percent	<input type="checkbox"/> wwRetrievalMode
<input type="checkbox"/> EndDateTime	<input type="checkbox"/> MaxContained	<input type="checkbox"/> PercentContained	<input type="checkbox"/> wwVersion
<input checked="" type="checkbox"/> OPC Quality			

The following are the column options:

- **Tag Name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions on page 24.
- **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-intervals) 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.

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 Wonderware 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 Wonderware Historian service user has full access to those network locations. For the Windows Server 2003 operating system, the locations must be specified using UNC notation.
 - **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.

	Storage Type	MaxMBSize	MinMBThreshold	Path
▶	Circular	0	125	C:\Historian\DATA\Circular
	Alternate	0	125	rr:\OverflowData
	Buffer	0	125	C:\Historian\DATA\Buffer
	Permanent	0	125	C:\Historian\DATA\Permanent

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.

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

	TagName	System Space (MB)
▶	SysSpaceAlt	0
	SysSpaceBuffer	148.828125
	SysSpaceMain	148.828125
	SysSpacePerm	148.828125

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

	Start Date
▶	9/15/2004 12:58:00.000 PM

Query Type: Summary Values

You can view the summarized values of specified tags as calculated by the event system. To view aggregated data as calculated by the standard SQL Server aggregation functions, use the **Aggregate Values** query type. For more information, see **Query Type: Aggregate Values** on page 176.

To view summary values

- 1 In the **Query Type** list in the toolbar, click **Summary 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** on page 207.
 - See **Time Tab** on page 212.
 - See **Calculations Tab** on page 208.
 - See **Order Tab** on page 216.
- 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.

The screenshot shows a control panel for the 'Columns' tab. It includes several checkboxes and a dropdown menu:

- Tag name
- Description
- Decimal places: 0 (dropdown menu)
- Date and time
- Include milliseconds
- Quality
- Resolution
- Timestamp
- Event tag

Options are as follows:

- **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions on page 24.
- **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 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.
- **Include milliseconds:** Used to include milliseconds in the timestamp.
- **Quality:** The basic data quality indicator associated with the data value.
- **Resolution:** The sampling rate, in milliseconds, for retrieving the data in cyclic mode.
- **Timestamp:** The timestamp used when storing the result of the calculation. This can either be the time of when the calculation period starts or the time when it ends.
- **Event tag:** The unique name of the tag within the Wonderware Historian system.

Calculations Tab

Use the **Calculations** tab to specify which calculated values to retrieve from the database.

- 1 In the **Limit to XX rows** list, specify the number of initial consecutive rows to return out of the total number of rows in the record set, starting with the first row in the record set. For example, if there are a total of 150 rows, and you set this value to 100, only the first 100 rows in the records set will be returned.
- 2 In the **Calculation type** list, click the type of calculation. Sum, Maximum, Minimum, or Average.
- 3 In the **Calculation frequency** list, click the time duration, in seconds, for which the calculation is performed.
- 4 Select the **Show** check boxes to show the calculation type and/or frequency in the result set.

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, click **Tag details**.
- 2 Use the Tag Picker to select one or more tags.
- 3 In the **Columns** pane, click on the **Columns** tab.

<input checked="" type="checkbox"/> Description	<input checked="" type="checkbox"/> Raw value range	<input checked="" type="checkbox"/> Detector type
<input type="checkbox"/> Date created	<input checked="" type="checkbox"/> Engineering units	<input checked="" type="checkbox"/> Action type
<input type="checkbox"/> Address	<input checked="" type="checkbox"/> Engineering units range	<input type="checkbox"/> Detector string
<input checked="" type="checkbox"/> Storage rate	<input checked="" type="checkbox"/> Messages	<input type="checkbox"/> Action string
<input type="checkbox"/> Acquisition rate	<input checked="" type="checkbox"/> Maximum characters	<input type="checkbox"/> Scan rate
<input checked="" type="checkbox"/> Storage type	<input type="checkbox"/> Source Tag	<input type="checkbox"/> Status
<input type="checkbox"/> Acquisition type	<input type="checkbox"/> Source Server	<input type="checkbox"/> Logged

- 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.
- **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 Wonderware Historian.
- **Action string:** The script that specifies the event action. Action scripts are executed on the local Wonderware 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, click the **Data** tab in the **Results** pane.

	TagName	Description	Address
▶	ReactLevel	Reactor level	VIEW TagName!ReactLevel

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, click on the tab and configure the parameters for the query:
 - See **Search Tab** on page 211.

- 4 To view the results, click the **Data** tab in the **Results** pane..

	TagName	Description
▶	SysCritErrCnt	Total Critical errors since startup
	SysFatalErrCnt	Total Fatal errors since startup
	SysErrErrCnt	Total non-fatal errors since startu
	SysWarnErrCnt	Total warnings since startup
	SysStatusRxTotalItems	Total items received from SysDrv :
	SysStatusRxItemsPerSec	Items per second received from S
	SysSpaceMain	Space left on Circular Data Path
	SysSpaceAlt	Space left on Alternative Data Pat
	SysSpaceBuffer	Space left on Buffer Data Path
	SysSpacePerm	Space left on Permanent Data Pat
	SysMinutesRun	Minutes since the last startup
	SysTimeSec	System Time : Seconds
	SysTimeMin	System Time : Minutes
	SysTimeHour	System Time : Hours
	SysDateDay	System Date : Day
	SysDateMonth	System Date : Month
	SysDateYear	System Date : Year
	SysConfigStatus	System Configuration Status (Re-f
	SysPerfCPUTotal	%CPU total processor load

Search Tab

Use the **Search** tab to search for a tag in the database.

Tag type:	Analog	▼
Calculation type:	All	▼ <input type="checkbox"/> Show
Calculation frequency:	All	▼ <input type="checkbox"/> Show

- 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.
- 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 Wonderware 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.

	SysMinutesRun
▶	11196

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.

Time:

Use time zone of server

Time zone:

Entity	Time Zone	Daylight Saving Start	Daylight Saving End
Application	Pacific Daylight T...	4/2/2006 2:00 AM	10/29/2006 2:00 AM
Client	Pacific Daylight T...	4/2/2006 2:00 AM	10/29/2006 2:00 AM
IEJNTEST	Pacific Daylight T...	4/2/2006 2:00 AM	10/29/2006 2:00 AM

The grid shows the time zone and daylight savings time settings for the following entities:

Entity	Description
Application	The Wonderware 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 Wonderware 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 on page 47.
- 2 To return the data with a timestamp that reflects the time zone setting of the Wonderware 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 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.

<input type="radio"/> Narrow query format			<input checked="" type="radio"/> Wide query format		
TagName	DateTime	vValue	DateTime	SysCPU0	SysCPU1
SysCPU 0	1/20/2005	2	1/20/2005	0	2
SysCPU 1	1/20/2005	3	1/20/2005	0	5
SysCPU 2	1/20/2005	0	1/20/2005	0	5

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.

Retrieval Tab

Use the **Retrieval** tab to specify the “granularity” of the data to be returned.

To configure retrieval mode options

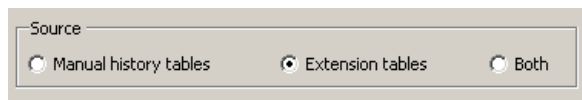
- 1 In the **Retrieval mode** list, select the retrieval mode that allows you to access the data stored in a Wonderware Historian in different ways.
- 2 In the **Query row limit** list, select the maximum number of rows for the data retrieval to avoid excessively large result sets. For example, if you 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.
- 3 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.

- 4 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** on page 196.

Source Tab

Use the **Source** tab to specify the data version and type of table for the query.

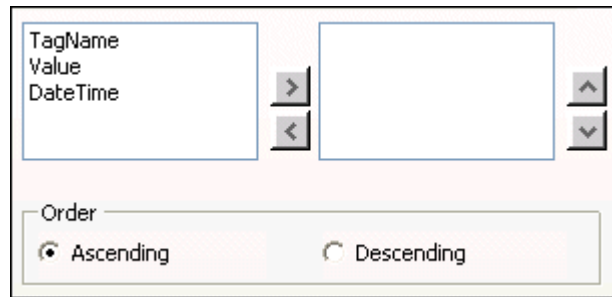


To configure the source

- 1 In the **Source** area, specify the Wonderware 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 Wonderware Historian data files. These tables support the Wonderware 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. Then click the arrow button to move the column to the right column. Repeat to add all of the columns 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.

Chapter 5

Wonderware Historian Client Workbook

The Wonderware Historian Client Workbook is an add-in to Microsoft Excel that allows you to query one or more Wonderware Historian or SQL Server databases and return results to a spreadsheet. Using the Wonderware Historian Client Workbook, you can easily create reports using Wonderware Historian data without needing in-depth knowledge of SQL scripting. The reports that you create with the Wonderware Historian Client Workbook can be saved, allowing you to run a report again at any time.

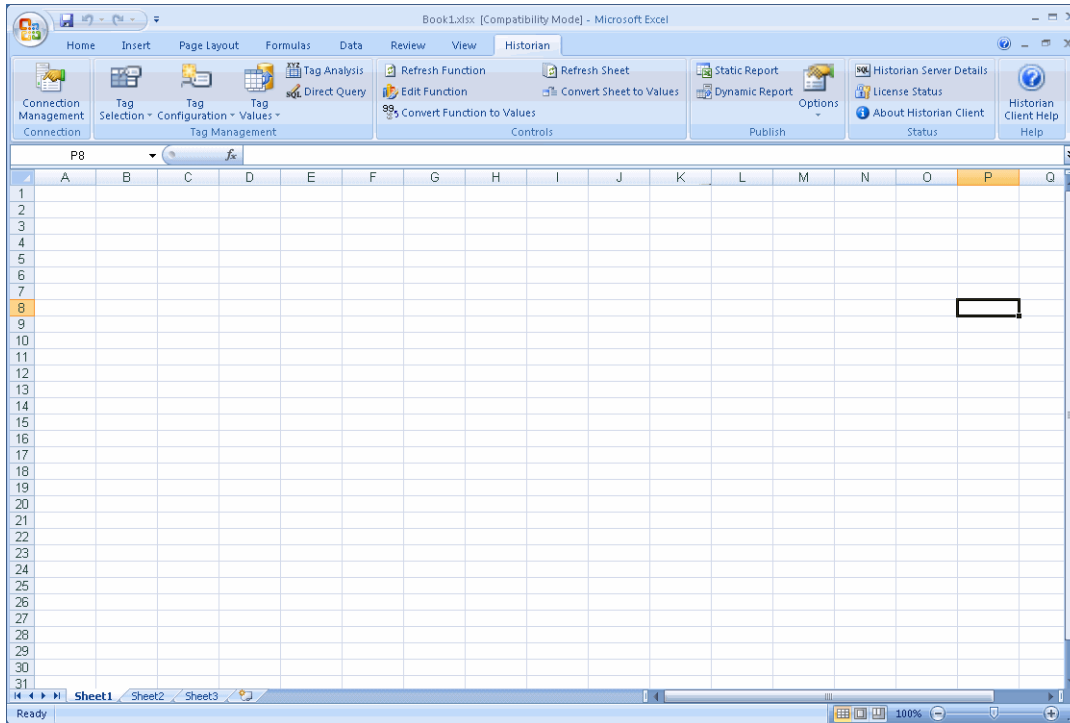
Getting Started

If the Wonderware 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 Wonderware Historian data.

The appearance of the **Historian** menu in the Wonderware Historian Client Workbook differs based on the version of Microsoft Office.

- If you are using Office 2003 or XP, the **Historian** menu appears as a menu item with drop-down options and contains a few toolbar options.
- If you are using Office 2007, the **Historian** menu is a part of the Ribbon Bar.



Managing Server Connections

You must specify one or more Wonderware Historians and/or SQL Servers as the data sources for the Wonderware Historian Client Workbook.

To manage server connections

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Connection Management**. The **Server List Configuration** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Connection** group, click **Connection Management**. The **Server List Configuration** dialog box appears.

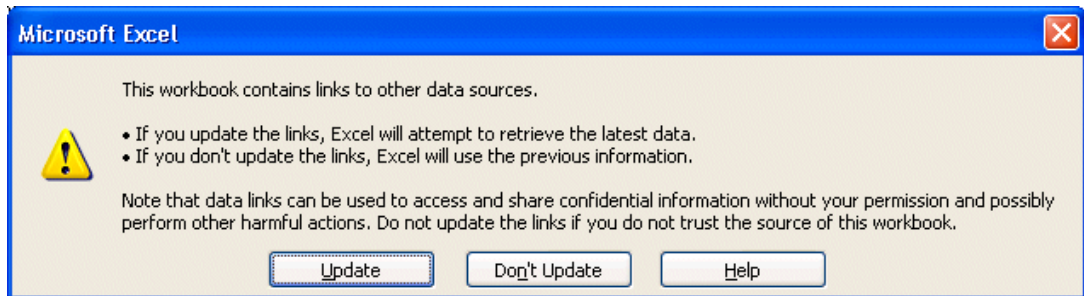
- 2 Configure the server(s) and then click **Close**. For more information, see *Server Connection Configuration* on page 27.

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 Wonderware Historian Client Workbook
- The file you are opening was saved using a different computer.

If Microsoft Excel detects that links need updating, a message box appears.



You can either update the links or keep them the same. No matter which option you select, the Wonderware Historian Client Workbook add-in automatically updates only the Wonderware Historian Client Workbook reference within the file to use the current add-in location. You can update the links or keep them the same. If you update the links, click **Continue** in the dialog box that appears.

To open an existing Workbook file

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **File** menu, click **Open**. The **Open** dialog box appears.
 - If you are using Excel 2007, click the **Microsoft Office Button**, and then click **Open**. The **Open** dialog box appears.
- 2 Select the name of the file to open.
- 3 Click **Open**.

Manually Loading/Unloading the Add-In

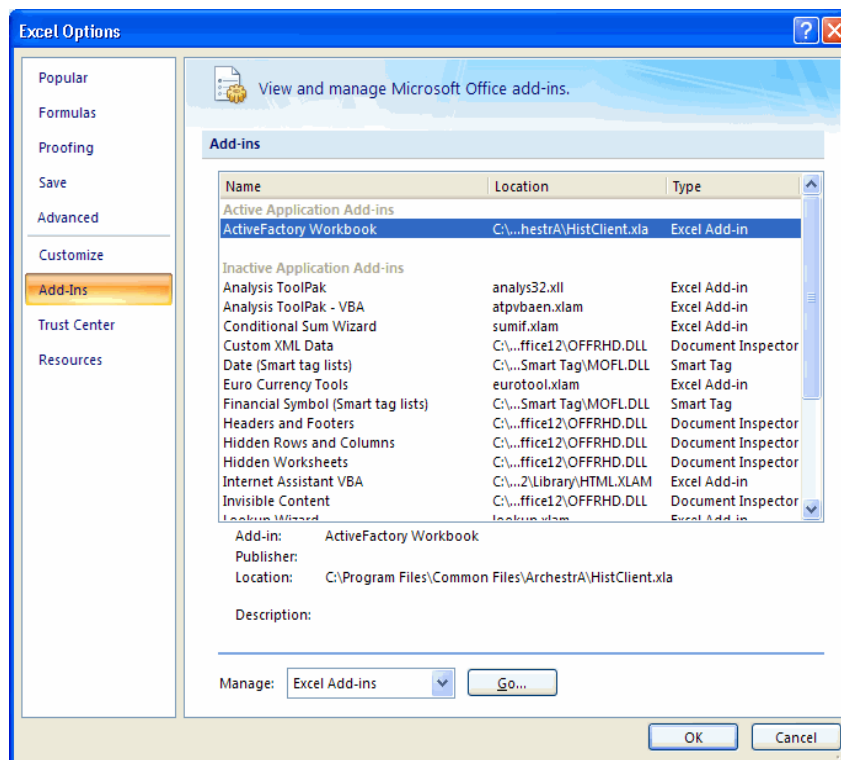
When you install the Wonderware Historian Client software after installing Microsoft Excel, the Excel add-in is automatically loaded into Excel. For Office 2003 or XP, the **Historian** menu appears in the menu bar and for Office 2007, the **Historian** tab appears in the Ribbon bar.

However, if you need to manually load or unload the add-in, use the following procedure:

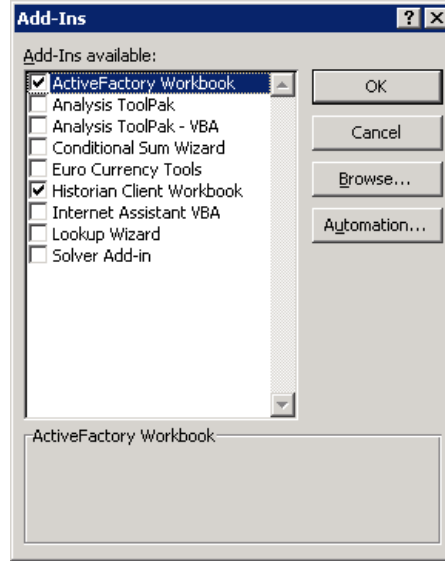
To manually load the add-in

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Tools** menu, click **Add Ins**. The **Add-Ins** dialog box appears.
 - If you are using Excel 2007, click the **Microsoft Office Button**, and then click **Excel Options**. The **Excel Options** dialog box appears.

Click **Add-Ins**.



In the Manage list, select **Excel Add-ins**, and then click **Go**. The **Add-Ins** dialog box appears.



- 2 In the **Add-Ins** dialog box, do one of the following:
 - If you are using Office 2003 or XP, browse and select the **HistClient.xla** and select the **ActiveFactory Workbook** check box.
 - If you are using Office 2007, browse and select the **HistClient.xla** and **HistClient.xlam** and select the **ActiveFactory Workbook** and **Historian Client Workbook** check boxes.

By default, the **HistClient.xla** and **HistClient.xlam** files are installed in the **C:\Program Files\Common Files\ArchestrA** folder.

- 3 Click **OK**.

To manually unload the add-in

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Tools** menu, click **Add Ins**. The **Add-Ins** dialog box appears.
 - If you are using Excel 2007, click the **Microsoft Office Button**, and then click **Excel Options**.

Click **Add-Ins**.

In the Manage list, select **Excel Add-ins**, and then click **Go**. The **Add-Ins** dialog box appears.

- 2 In the **Add-Ins** dialog box, do one of the following:
 - If you are using Excel 2003 or XP, clear the **ActiveFactory Workbook** check box.
 - If you are using Excel 2007, clear the **Historian Client Workbook** check box.

- 3 Click **OK**.

Creating a Report: Overview

Follow these general steps to create reports using the Wonderware Historian Client Workbook.

- 1 Configure a connection to one or more servers. For more information, see [Managing Server Connections](#) on page 218.
- 2 Understand how functions, formulas, and array formulas work. For more information, see [Working with Functions, Formulas, and Cells](#) on page 223.
- 3 Determine how you want to set up or use Workbook options. For more information, see [Configuring Workbook Options](#) on page 309.
- 4 Configure tags for which you want to return data. For more information, see [Selecting Tags for Reports](#) on page 232.
- 5 Create a data report for the selected tags.
 - For information on retrieving configuration data using wizards, see [Retrieving Tag Configuration Information](#) on page 239.
 - For information on retrieving current and historical data values using wizards, see [Retrieving Tag Values](#) on page 247.
 - For information on generating analysis graphs and data using wizards, see [Retrieving Tag Values](#) on page 247.
 - For information on retrieving data using a manually created SQL query, see [Creating a Direct Query](#) on page 306.
- 6 Configure other advanced optional features. For more information, see [Configuring Workbook Options](#) on page 309.
- 7 Save the report.
- 8 Optionally publish the report to the Wonderware Information Server. For more information, see [Publishing Reports](#) on page 326.

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, ...)
```

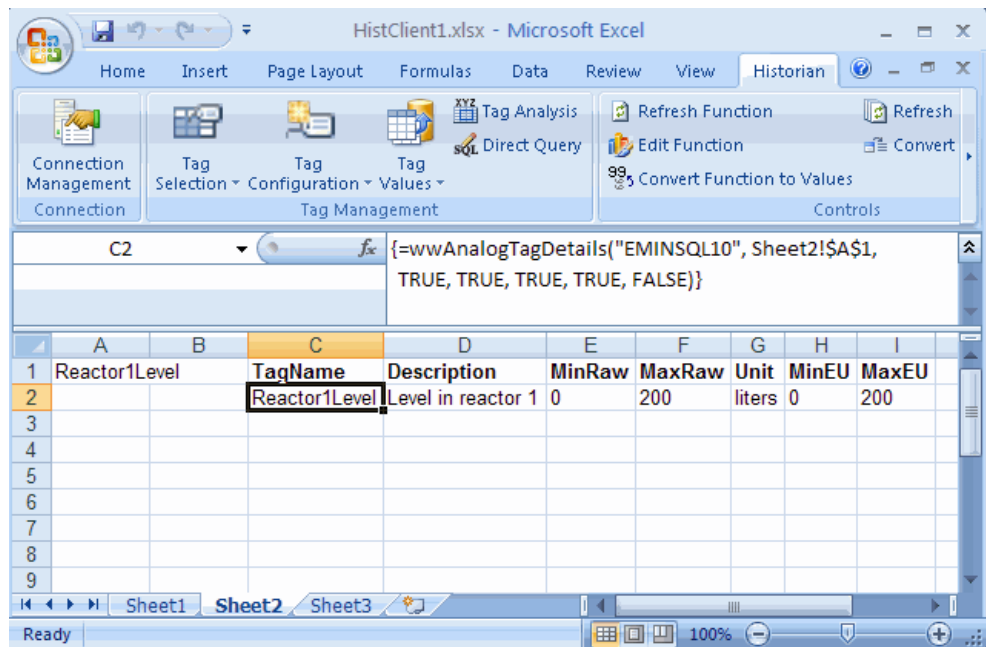
An array formula is a type of function that can perform multiple calculations and then return either single or multiple results. Array formulas act on two or more sets of values called array arguments. The arguments are the inputs to the function and are required to be in a particular order.

Most of the formulas created using the Wonderware 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.



Note There are certain limitations when working with arrays. For more information, see the following link:

<http://support.microsoft.com/kb/166342/>

Refreshing a Function or Array Formula

You can refresh any function or array formula in the worksheet.

To refresh

1 Select the function to refresh. If you want to refresh an array formula, select any cell in the array.

2 Do one of the following:



- If you are using Excel 2003 or XP, on the **Historian** menu, click **Refresh Function**.
- Click the **Refresh Function** toolbar button.
- If you are using Excel 2007, on the **Historian** tab, in the **Controls** group, click **Refresh Function**.

The function is executed and the results are returned.

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 Do one of the following:



- If you are using Excel 2003 or XP, on the **Historian** menu, click **Edit Function**.
- Click the **Edit Function** toolbar button.
- If you are using Excel 2007, 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 Do one of the following:



- If you are using Excel 2003 or XP, on the **Historian** menu, click **Convert Function to Values**.
- Click the **Convert Function to Values** toolbar button.
- If you are using Excel 2007, 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 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Refresh Sheet**.
 - Click the **Refresh Sheet** toolbar button.
 - If you are using Excel 2007, 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 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Convert Sheet to Values**.
 - Click the **Convert Sheet to Values** toolbar button.
 - If you are using Excel 2007, on the **Historian** tab, in the **Controls** group, click **Convert Sheet to Values**.



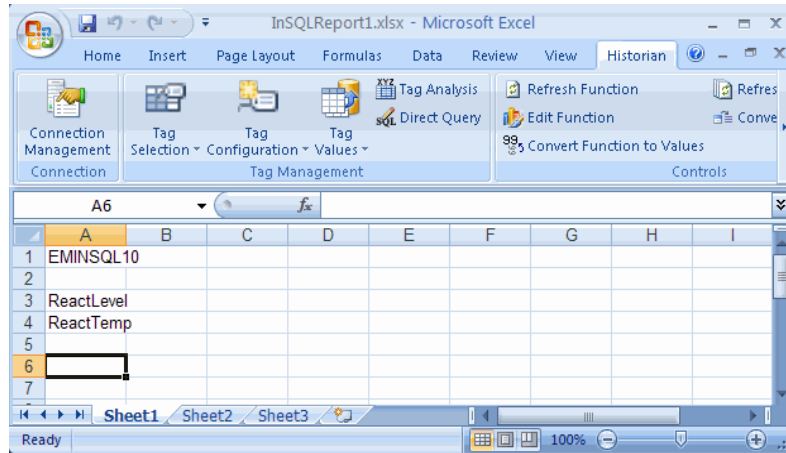
Manually Inserting a Function

You can manually insert functions instead of using the function wizards.

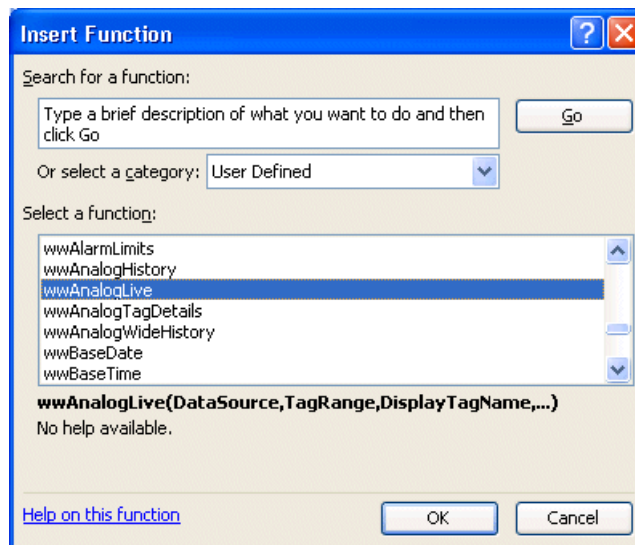
In Excel, functions are not automatically inserted as array formulas. By default, only single cell contains a value from the result set. You must type the formula as an array formula (by pressing **CTRL+SHIFT+ENTER**) so that all values in the result set appear.

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.



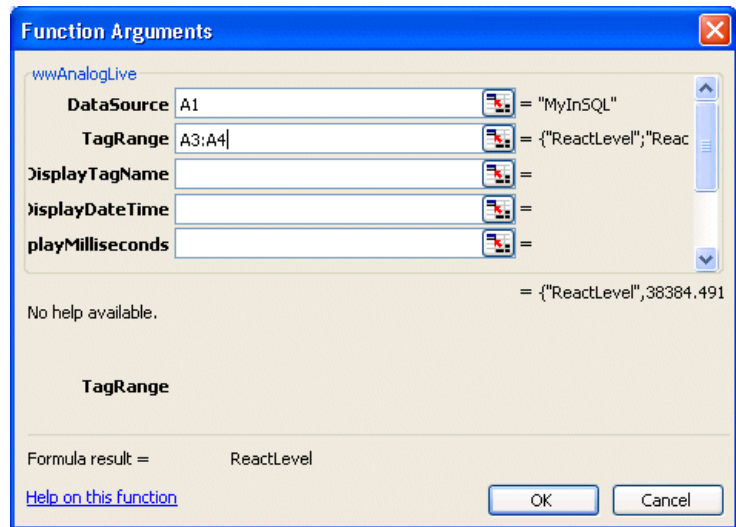
- 2 On the Formulas tab, in the Function Library group, click Insert Function. The Insert Function dialog box appears.



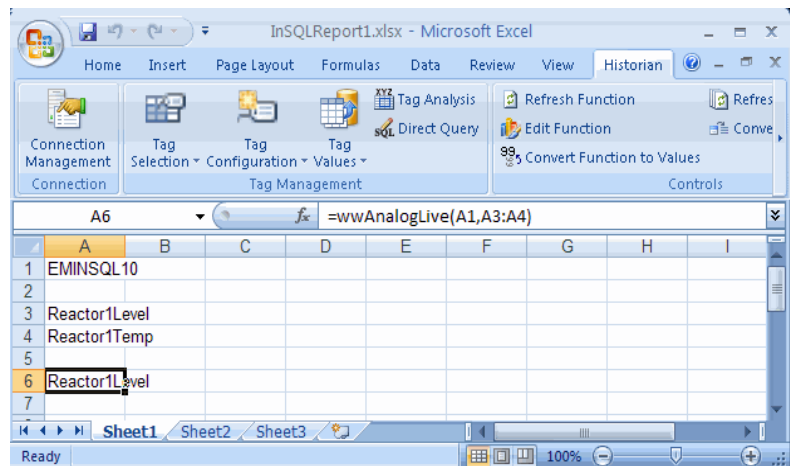
- 3 In the Or select a category list, click User Defined.
- 4 In the Select a function list, select any of the Wonderware Historian Client Workbook functions.

All of these functions are prefixed with “ww.” For more information regarding these functions and their arguments, see Wonderware Historian Client Workbook Function Reference on page 330.

- 5 Click OK. The Function Arguments dialog box appears.



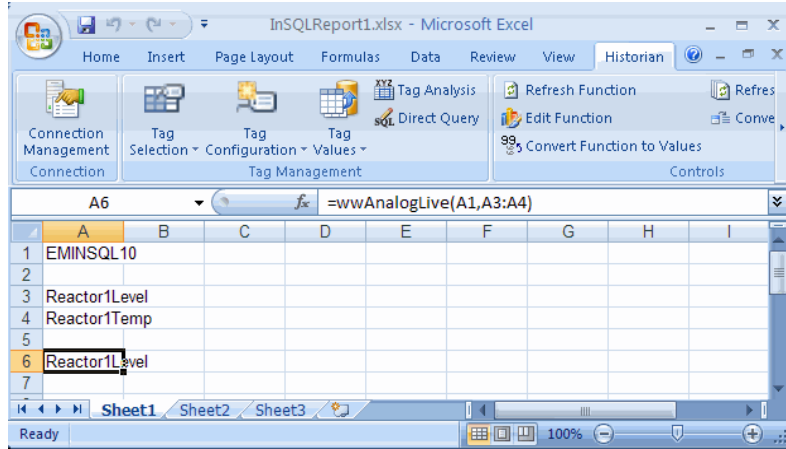
- 6 For each of the arguments, assign a cell value that contains the input.
For example, assigning A1 to the DataSource argument causes "MyInSQL" to be used for the data source.
- 7 Click OK. The function is inserted into the spreadsheet.



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.

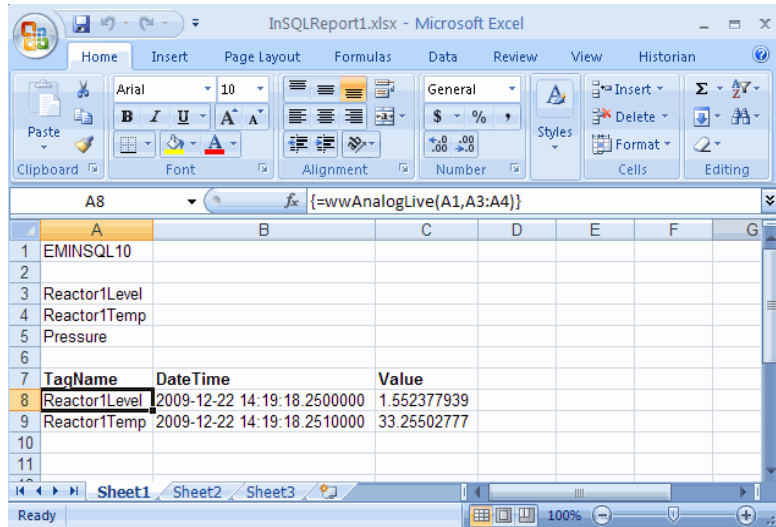
- 9 On the **Historian** tab, in the **Controls** group, click **Refresh Function**. The formula is converted to an array and you can see all of the return values.



Manually Editing a Function

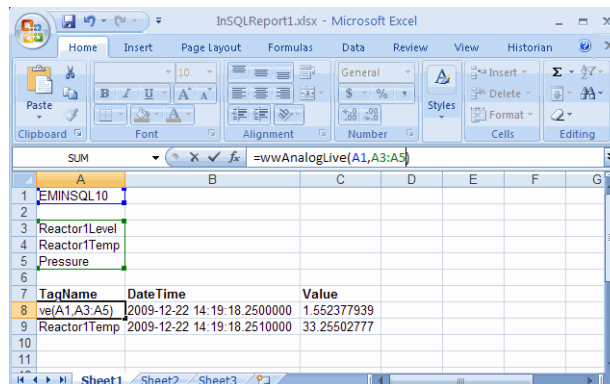
To manually edit a function

- 1 In your worksheet, select the function to edit so that it appears in the formula bar.

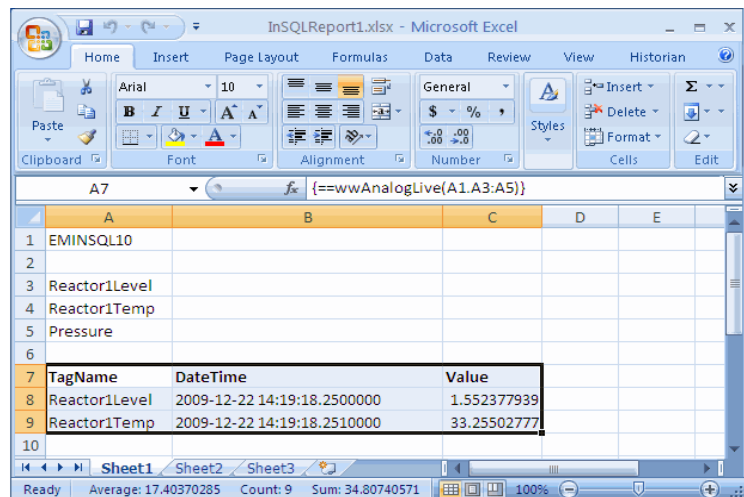


- In the formula bar, edit the argument value(s) for the function.

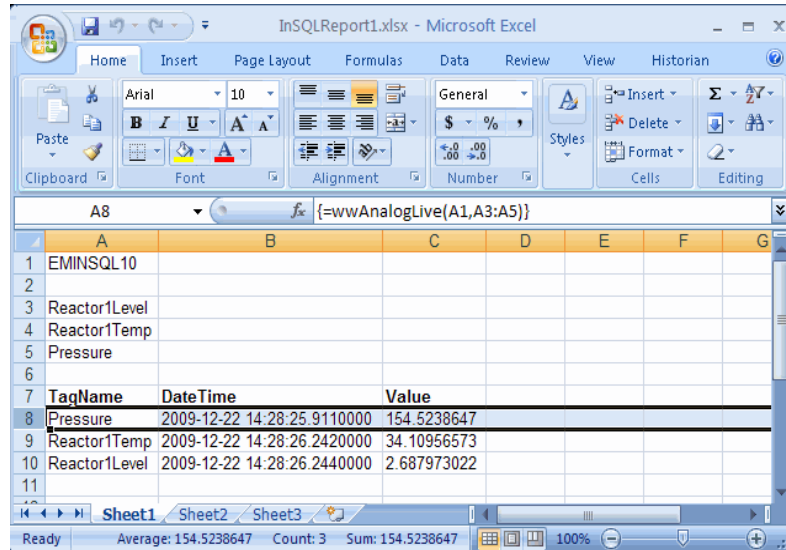
For example, you can add an additional tag by expanding the cell range.



- 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](#) on page 228.

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.

To select a cell in the worksheet

- 1 Click the button to the right of the option box that requires a cell location.



The cell selector dialog box appears.



- 2 In the spreadsheet, use your mouse to select the cell(s). The cell notation appears in the dialog box.



- 3 Click the **Notation** button to insert the notation into the option box.



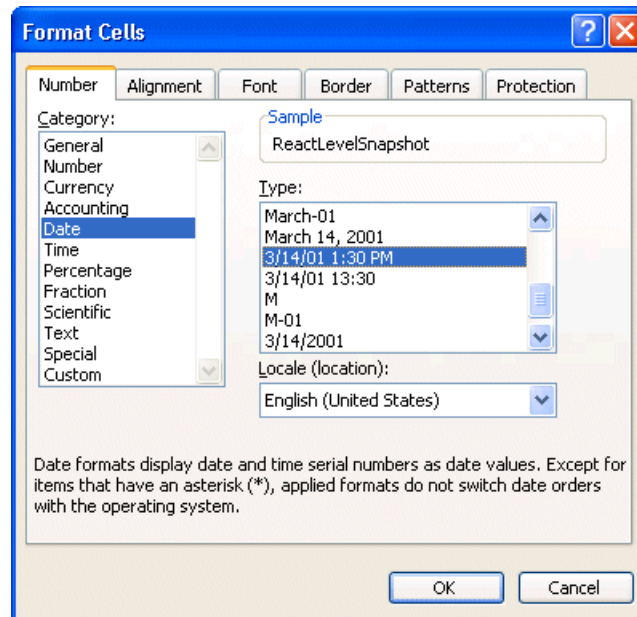
Verifying the Date/Time Format in Microsoft Excel

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.

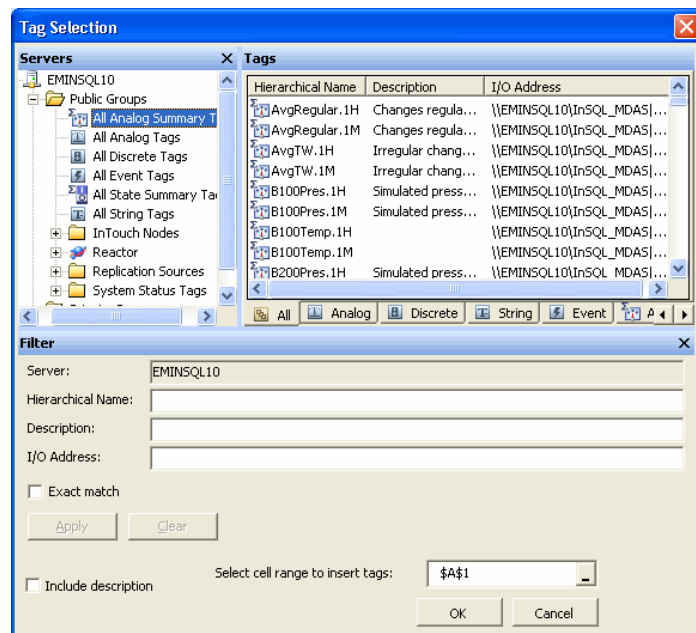
You can include the following types of tags in your worksheet:

- Analog, discrete, string, summary, and event tags. For more information, see *Selecting Analog, Discrete, String, Summary, or Event Tags* on page 233.
- 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* on page 235.
- 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* on page 238.

Selecting Analog, Discrete, String, Summary, or Event Tags

To select an analog, discrete, string, summary, or event tags

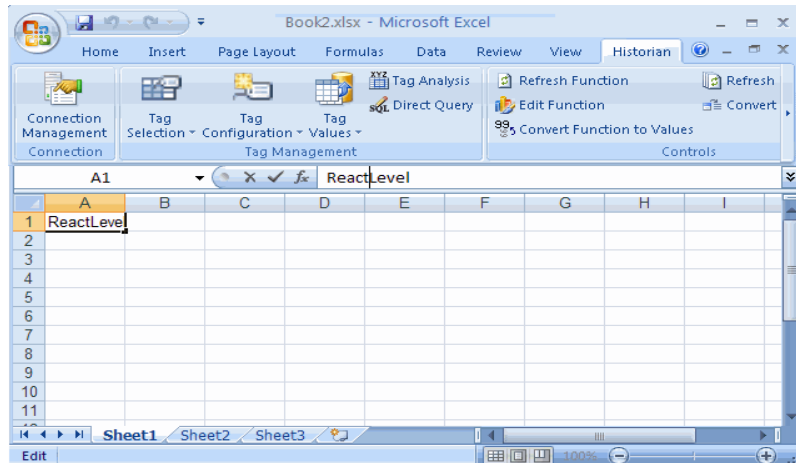
- Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, 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** on page 40.

- Select the **Include description** check box to include tag descriptions in the results.
- 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** on page 231.
- 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 Wonderware Application Server* on page 23.

To view the hierarchical name in a Sheet

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
 - If you are using Excel 2007, 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 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Selection**, and then click **Summary Tag Selection**. The **Summary Tag Selection** dialog box appears.
 - If you are using Excel 2007, 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.

Summary Tag Selection

Servers: EMINSQL10

Filter

Summarization frequency: 3600 Calculation type: Average

Tag name: % Description: %

Select tags to insert into workbook

Tag Name	Description
ReactLevel	Reactor level

Select cell range to insert tags: \$A\$1

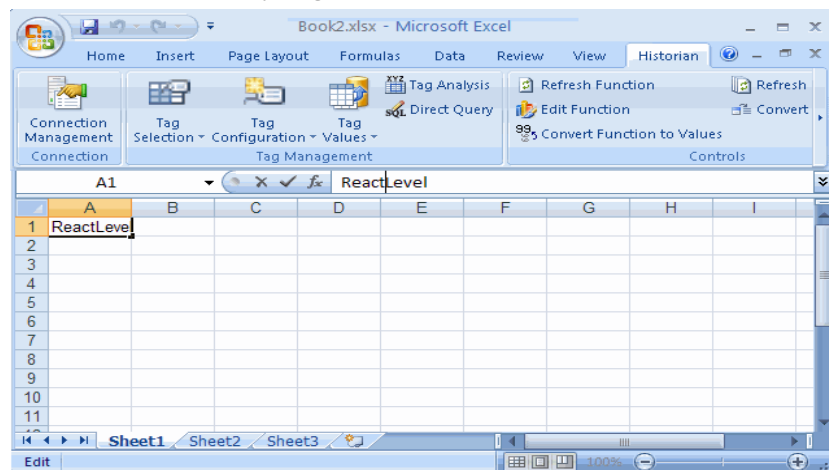
Include description

OK Cancel

- 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 Wonderware Historian server. If the data values are coming from ArchestrA, the attribute reference is shown as the tag name. For ArchestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArchestrA Naming Conventions on page 24.
 - **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** on page 231.
 - 6 Click **OK**.

The summary tags are inserted into the selected cell.



Finding a Source Tag or Replicated Tag

You can replicate tag information in a Wonderware 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 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
 - If you are using Excel 2007, 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:

- You are connected to the IndustrialSQL Server 9.0.2
- 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.

Selecting Event Snapshot Tags

To select event snapshot tags

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Selection**, and then click **Event Snapshot Tag Selection**. The **Event Snapshot Tag Selection** dialog box appears.
 - If you are using Excel 2007, 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.

Event Snapshot Tag Selection

Servers: EMINSQL10

Filter

Event tag: ReactLevelSnapshot

Snapshot tag type: Analog

Select tags to insert into workbook

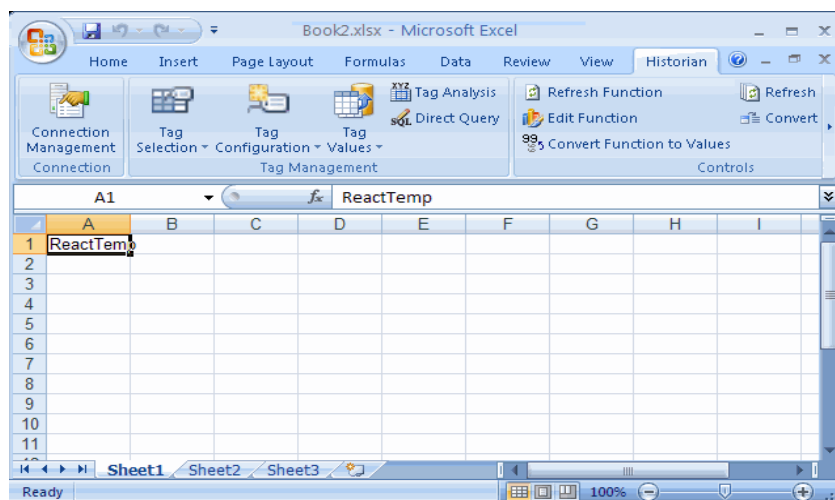
Tag Name	Description
ReactTemp	Reactor temp

Select cell range to insert tags: \$A\$1

Include description

OK Cancel

- 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](#) on page 231.
- 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.

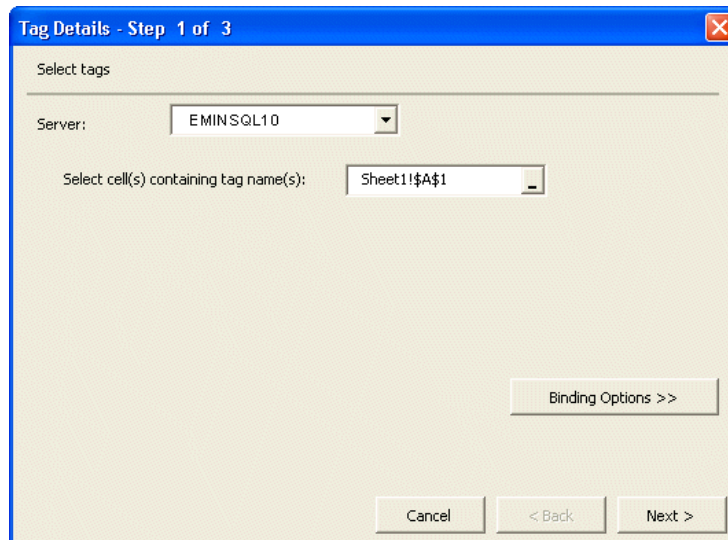
Retrieving Configuration Details for a Tag

You 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.

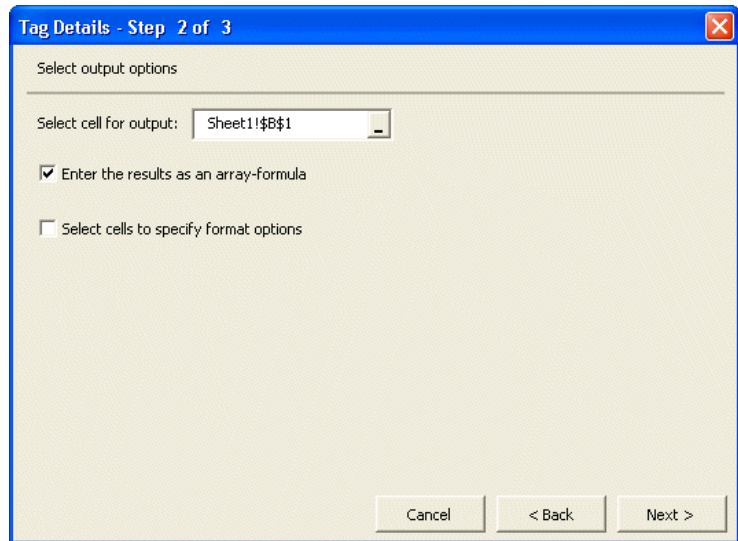
To retrieve tag details

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Configuration**, and then click **Tag Details**. The **Tag Details - Step 1 of 3** dialog box appears.
 - If you are using Excel 2007, 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.



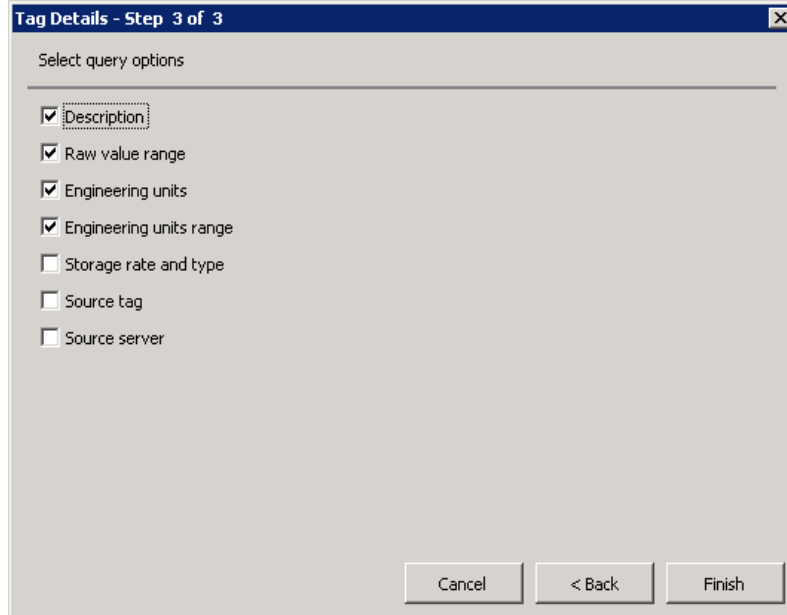
- 2 In the **Server** list, click the name of the server to use.
- 3 In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see [Selecting Cells](#) on page 231.
- 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](#) on page 321.

- 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](#) on page 231.
- 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](#) on page 223.
- 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](#) on page 231.

- 9 Click **Next**. The **Tag Details - Step 3 of 3** dialog box appears.

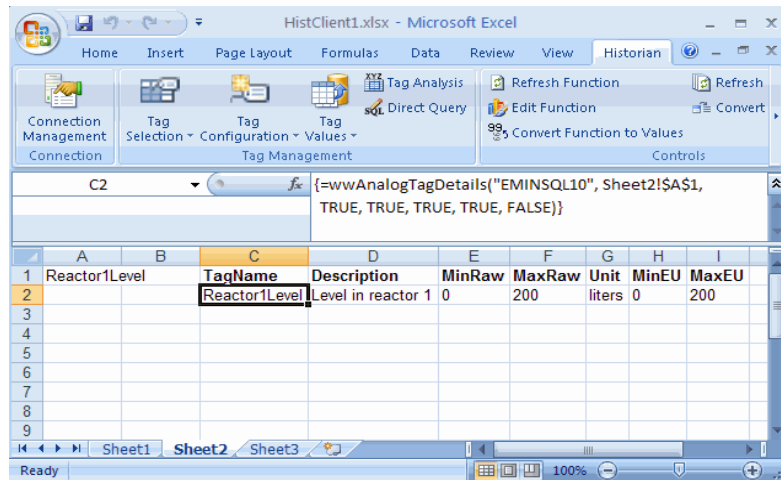


This dialog box displays different options, depending on the type of tag you have selected.

- 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).

14 Click **Finish**. The details appear in the spreadsheet.



Viewing the Arcestra Hierarchical Name in a Sheet

You can view the Arcestra hierarchical name in a sheet. For more information, see *Integration with Wonderware Application Server* on page 23.

To view the hierarchical names in a sheet

- Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Tag Management** group, click **Tag Selection**, and then click **Tag Selection**. The **Tag Selection** dialog box appears.
- Right-click a group or the tag, and click **Use hierarchical name**.

The **Tags** pane shows the hierarchical names of the selected Arcestra attributes.

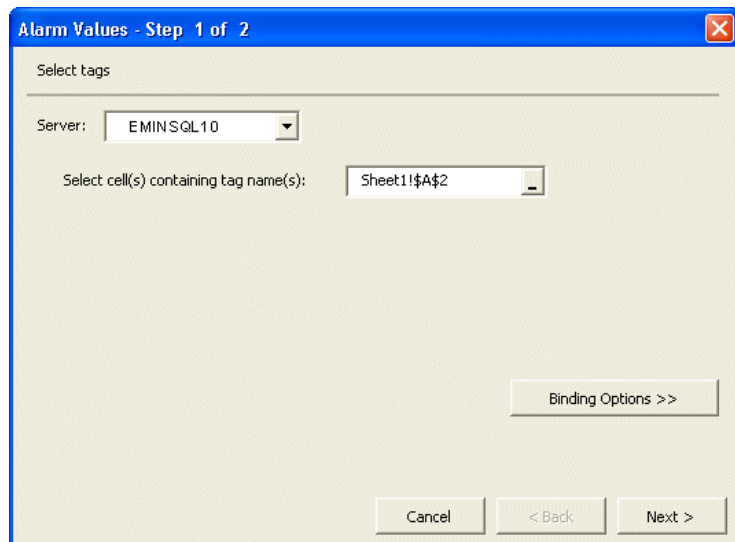
Retrieving Analog Tag Alarm Limits

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

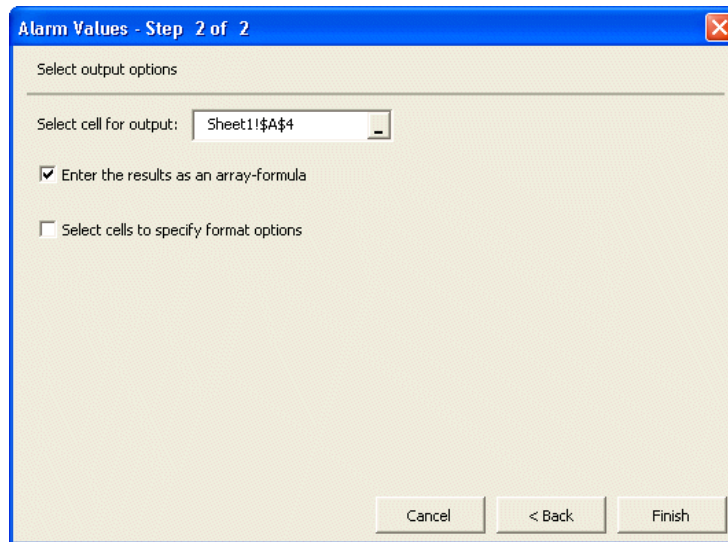
- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Configuration**, and then click **Analog Tag Alarm Limits**. The **Alarm Values - Step 1 of 2** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Tag Management** group, click **Tag Configuration**, and then click **Analog Tag Alarm Limits**.

The **Alarm Values - Step 1 of 2** dialog box appears.



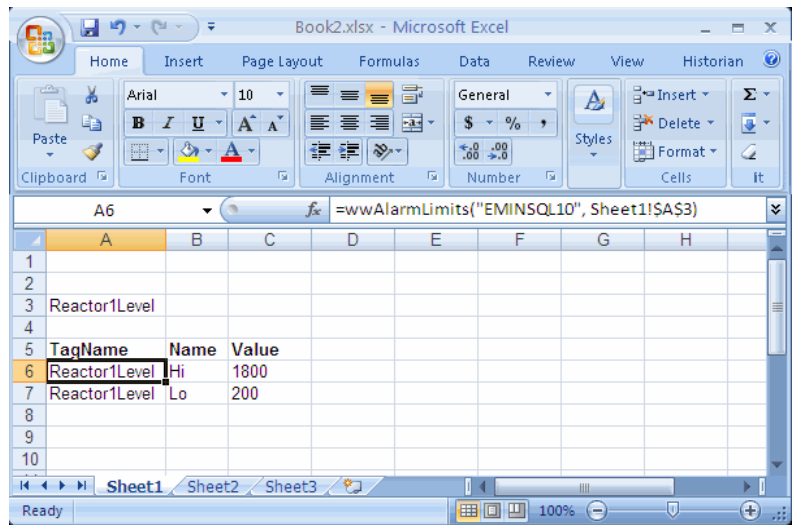
- 2 In the **Server** list, click the name of the server to use.
- 3 In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see [Selecting Cells](#) on page 231.
- 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](#) on page 321.

- 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* on page 231.
- 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* on page 223.
- 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* on page 231.

9 Click Finish. The details appear in the spreadsheet.



Retrieving Tag Values

You can retrieve the following types of values for tags:

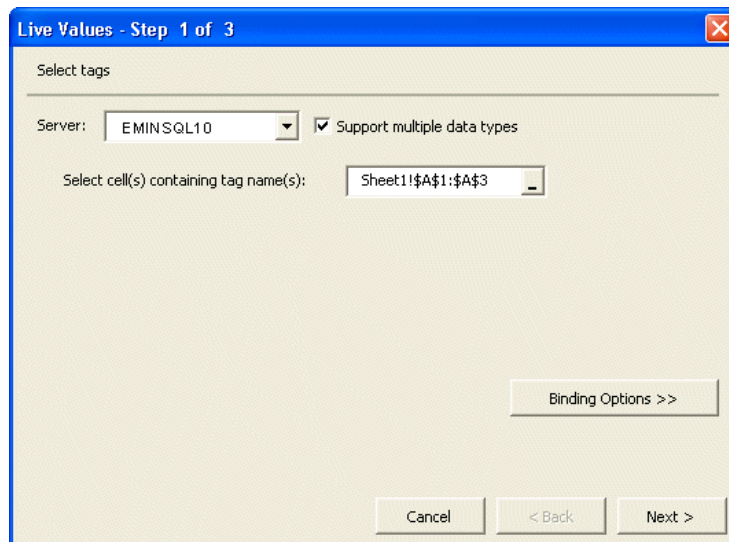
- "Live" values
- History values
- Aggregate values
- Summary System values
- Event Snapshot values

Retrieving Live Values

You can retrieve the current data values for specified tags.

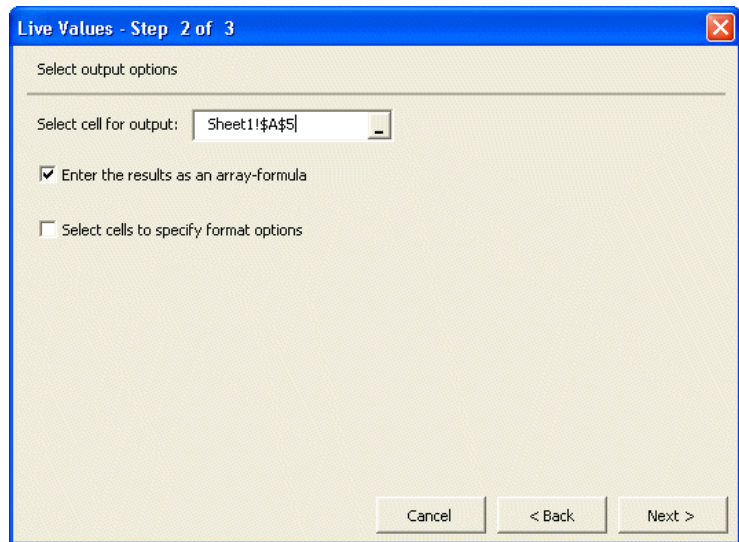
To retrieve live values

- 1 In cells in your worksheet, enter one or more tag names (one tag name per cell). For more information, see [Selecting Tags for Reports](#) on page 232.
- 2 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Values**, and then click **Live Values**. The **Live Values - Step 1 of 3** dialog box appears.
 - If you are using Excel 2007, 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.



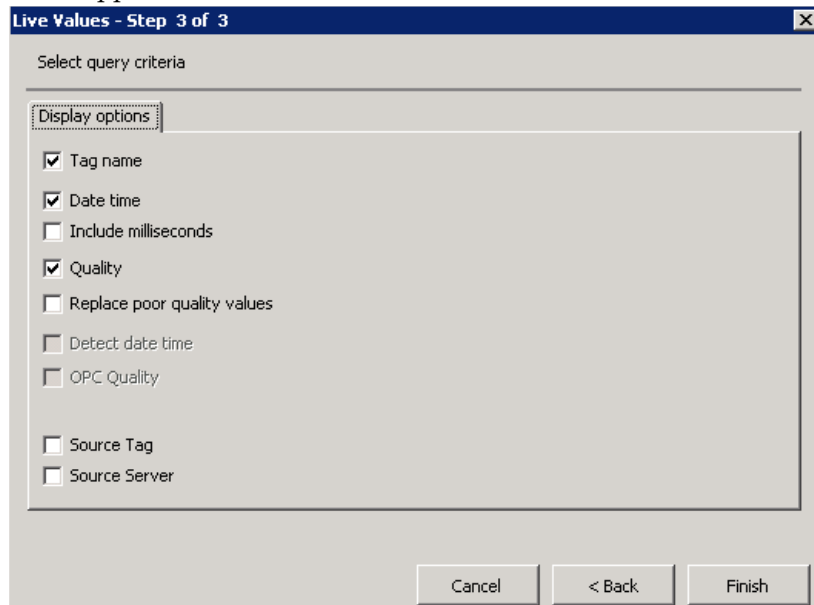
- 3 In the **Server** list, click the name of the server to use.
- 4 Select the **Support multiple data types** check box to allow for the selection of dissimilar data types for the same query. That is, a mix of analog, discrete, string, and/or event tags.
- 5 In the **Select cell(s) containing tag name(s)** 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](#) on page 231.
- 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](#) on page 321.

- 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* on page 231.
- 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* on page 223.
- 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* on page 231.

11 Click **Next**. The **Live Values - Step 3 of 3** dialog box appears.

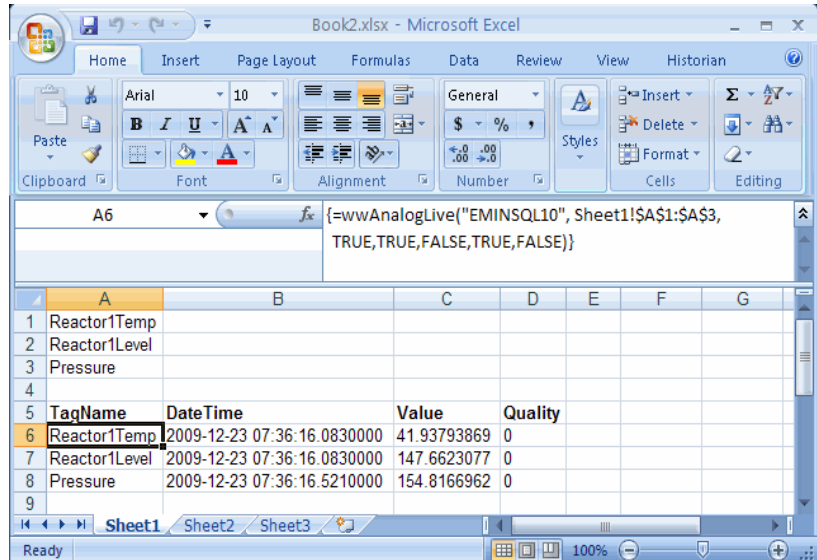


12 Configure the criteria for the query.

- **Tag name:** The name of the tag within the Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
- **Date time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the Wonderware 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 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.



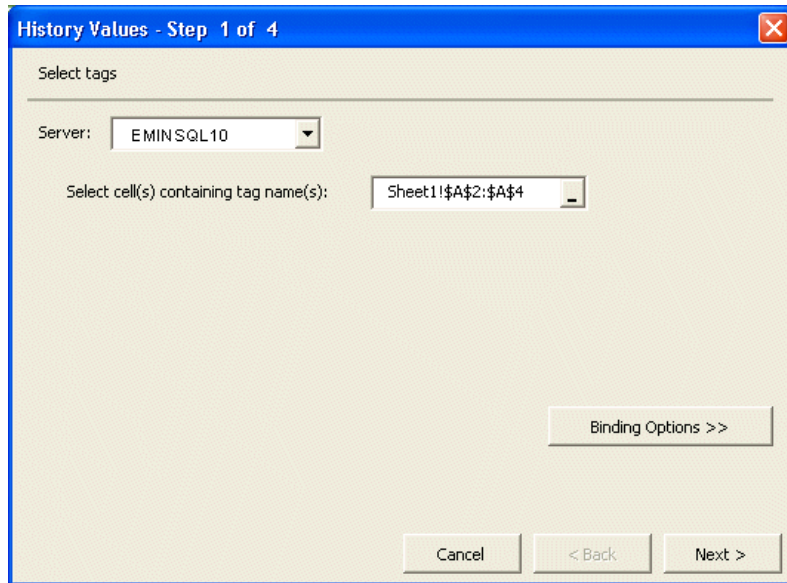
Retrieving History Values

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

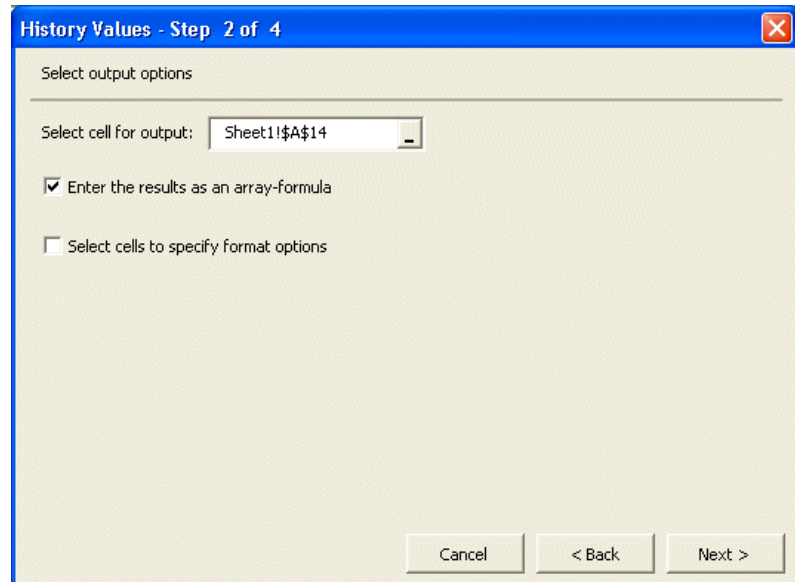
- 1 In cells in your worksheet, enter one or more tagnames (one tagname per cell).
- 2 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Values**, and then click **History Values**. The **History Values - Step 1 of 4** dialog box appears.

- If you are using Excel 2007, 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.



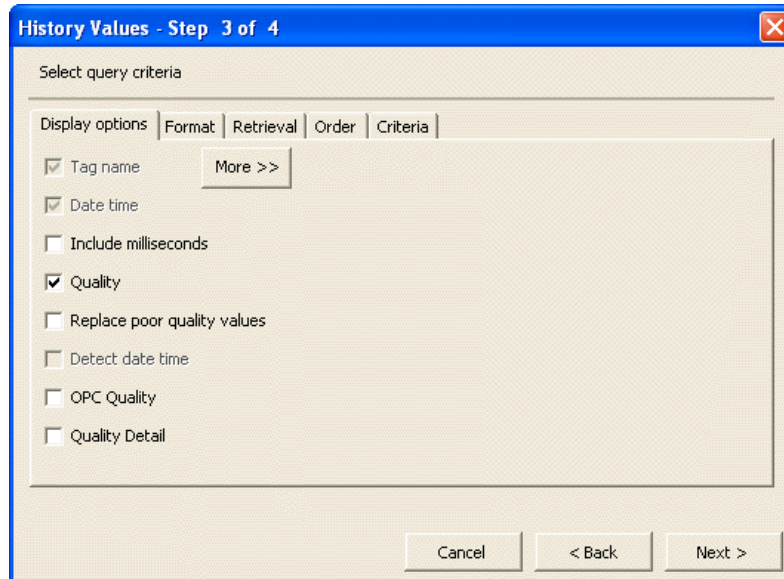
- 3 In the **Servers** list, click the name of the server to use.
- 4 In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see **Selecting Cells** on page 231.
- 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** on page 321.

- 6 Click Next. The History Values - Step 2 of 4 dialog box appears.

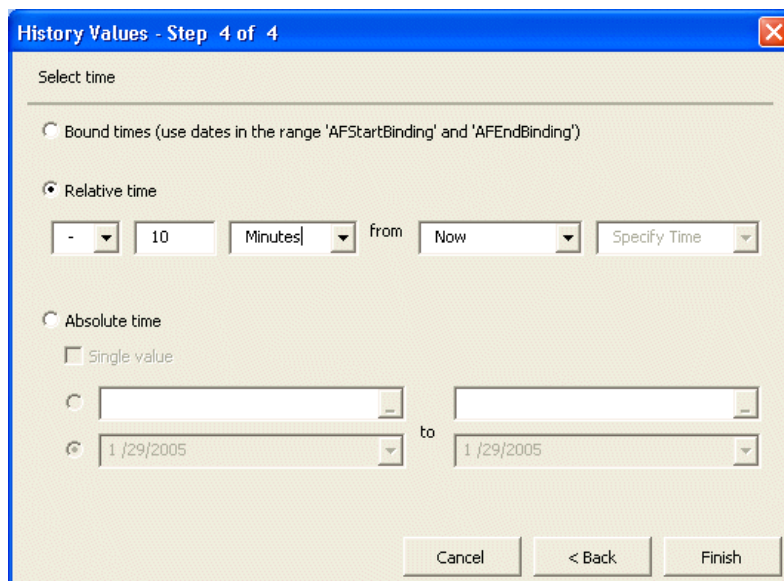


- 7 In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see [Selecting Cells](#) on page 231.
- 8 Select the **Enter the results as an array-formula** check 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](#) on page 223.
- 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](#) on page 231.

- 10 Click **Next**. The **History Values - Step 3 of 4** dialog box appears.

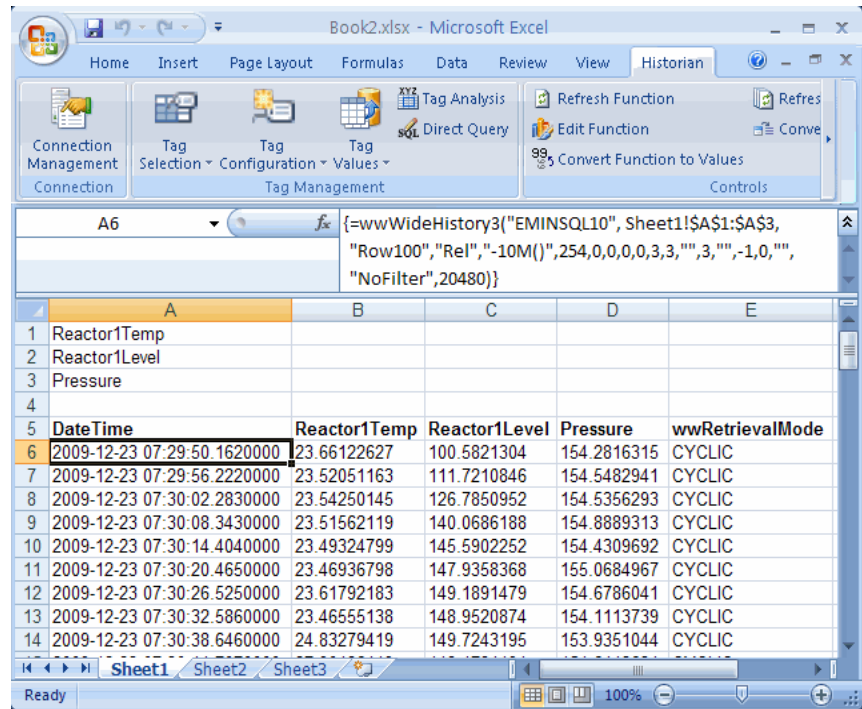


- 11 Configure the criteria for the query.
 See **Display Options Tab** on page 255.
 See **Format Tab** on page 273.
 See **Retrieval Tab** on page 257.
 See **Order Tab** on page 276.
 See **Criteria Tab** on page 274.
- 12 Click **Next**. The **History Values - Step 4 of 4** dialog box appears.



13 Configure the time for the query. For more information on configuring these options, see [Time Options for Queries](#) on page 325.

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](#) on page 271.

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](#) on page 689.
- **wwCycleCount**: The cycle count used in data retrieval. For more information, see [Cycle Count \(X Values over Equal Time Intervals\) \(wwCycleCount\)](#) on page 755.

When retrieving data from a Wonderware 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 \(wwTimeDeadband\)](#) on page 762.

- **wwTimeStampRule:** The timestamp rule used in data retrieval. For more information, see [Time stamp Rule \(wwTimeStampRule\)](#) on page 774.
- **wwVersion:** The history version of the value. For more information, see [History Version \(wwVersion\)](#) on page 769.
- **wwEdgeDetection:** The type of edge detection used in the query.
- **wwTagKey:** The unique identifier of the tag on the Wonderware 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 \(wwInterpolationType\)](#) on page 771.
- **wwResolution:** The resolution used in data retrieval. For more information, see [Resolution \(Values Spaced Every X ms\) \(wwResolution\)](#) on page 757.
- **wwValueDeadband:** The value deadband used in data retrieval. For more information, see [Value Deadband \(wwValueDeadband\)](#) on page 766.
- **wwQualityRule:** The quality rule used in data retrieval. For more information, see [Quality Rule \(wwQualityRule\)](#) on page 778.
- **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 \(wwStateCalc\)](#) on page 786.
- **wwFilter:** The analog filter used to filter data during retrieval. For more information, see [Analog Value Filtering \(wwFilter\)](#) on page 788.

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* on page 689 and *Understanding Retrieval Options* on page 752.

Other Tab

Use the **Other** tab to configure the other retrieval options.

- 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)* on page 769.
- b In the **Rules** area, do the following:
 - In the **Timestamp** list, click the required timestamp value. For more information on the Time stamp rule, see *Time stamp Rule (wwTimestampRule)* on page 774.
 - 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 Wonderware Historian level.

For more information on each option, see *Quality Rule (wwQualityRule)* on page 778.

- c 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) on page 786.

- d 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) on page 788.

- e 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 on page 760.

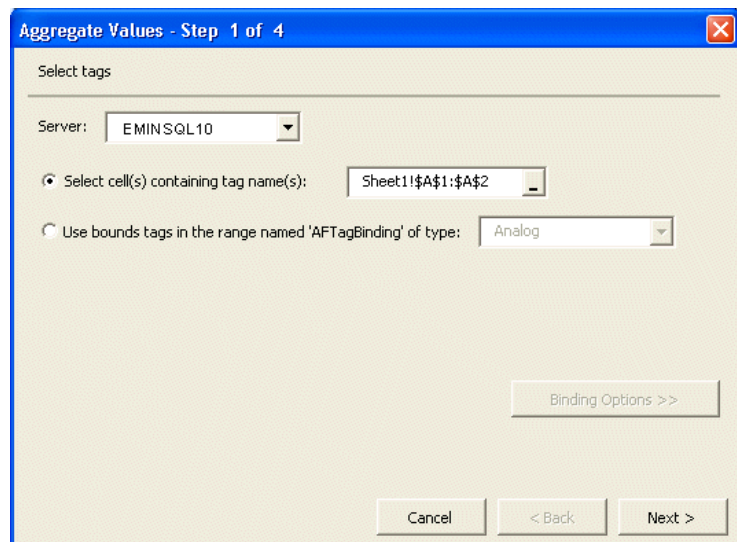
Retrieving Aggregate Values

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 Wonderware Historian's summary tables, see *Retrieving Values for Summarized Tags* on page 264.

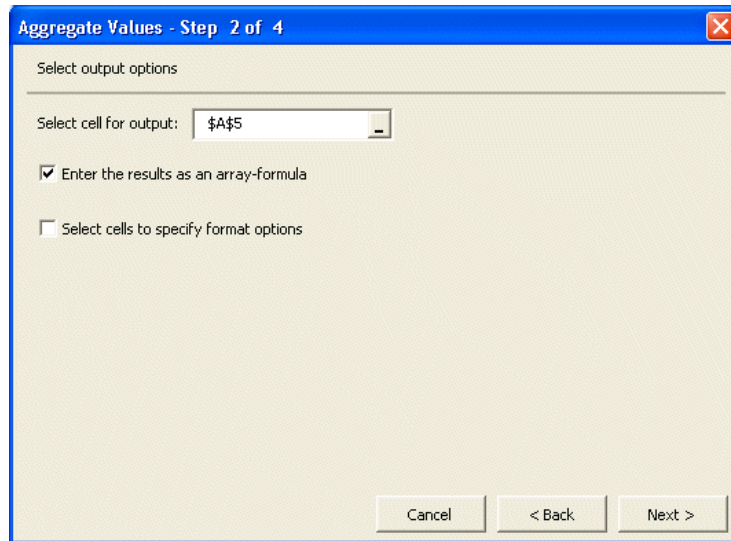
To retrieve aggregate values

- 1 In cells in your worksheet, enter one or more tag names (one tag name per cell).
- 2 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Values**, and then click **Aggregate Values**. The **Aggregate Values - Step 1 of 4** dialog box appears.
 - If you are using Excel 2007, 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.



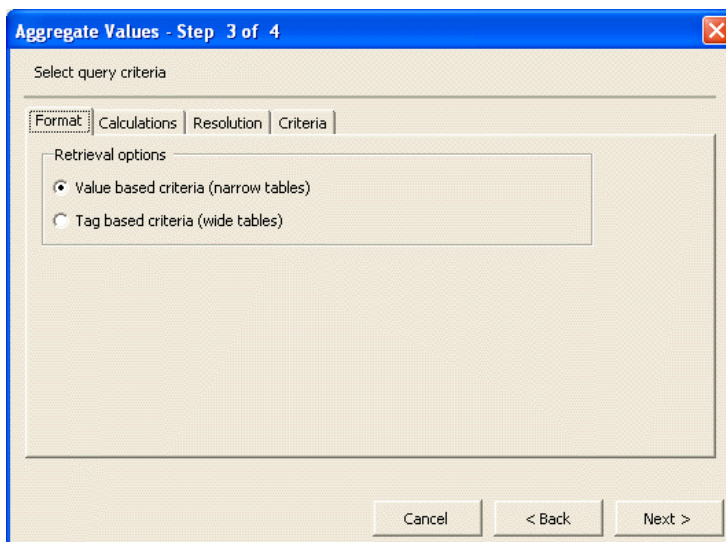
- 3 In the **Servers** list, click the name of the server to use.
- 4 In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see *Selecting Cells* on page 231.
- 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* on page 321.

- 6 Click **Next**. The **Aggregate Values - Step 2 of 4** dialog box appears.

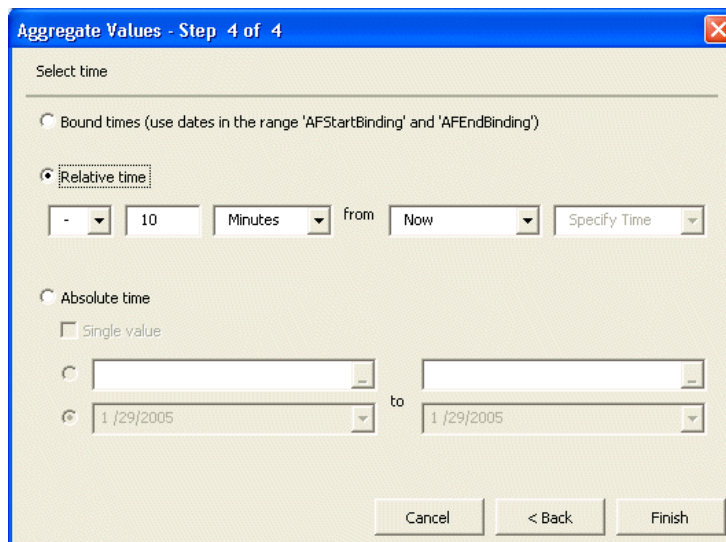


- 7 In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see [Selecting Cells](#) on page 231.
- 8 Select the **Enter the results as an array-formula** check 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](#) on page 223.
- 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](#) on page 231.

- 10 Click Next. The **Aggregate Values - Step 3 of 4** dialog box appears.

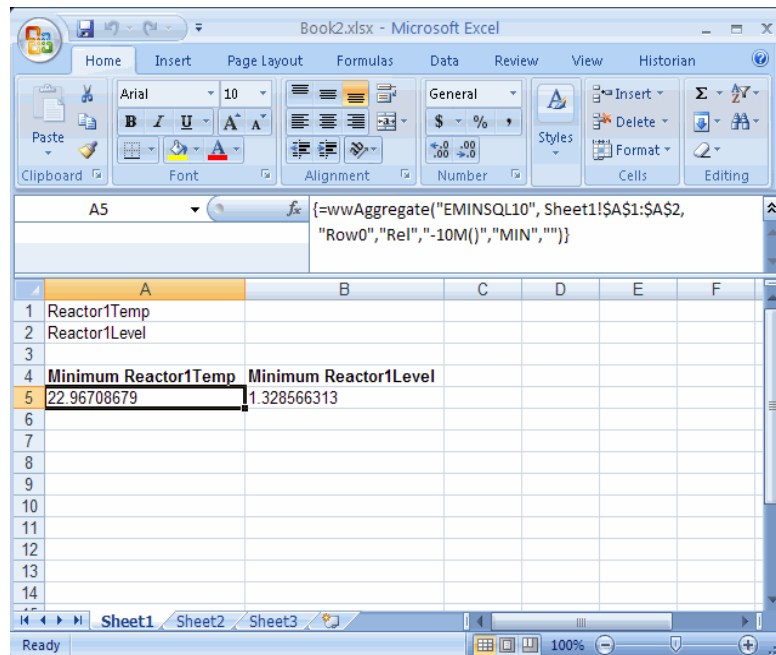


- 11 Configure the criteria for the query.
 See **Format Tab** on page 273.
 See **Calculations Tab** on page 262.
 See **Resolution Tab** on page 263.
 See **Criteria Tab** on page 274.
- 12 Click Next. The **Aggregate Values - Step 4 of 4** dialog box appears.



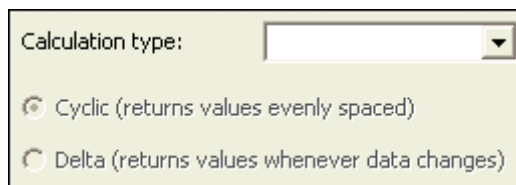
- 13 Configure the time for the query. For more information on configuring these options, see **Time Options for Queries** on page 325.

14 Click Finish.



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.

The screenshot shows the Resolution Tab configuration interface. It is divided into two main sections: Cyclic and Delta.

Mode: Radio buttons for Cyclic (selected) and Delta.

Cyclic (evenly spaced values):

- Radio button selected for "100 values over equal time intervals".
- Radio button for "Values spaced every 1000 ms".
- Radio button for "Full".
- Checkbox for "Interpolate" (unchecked).

Delta (whenever value changes):

- Radio button selected for "All rows".
- Radio button for "First 10 rows".
- Checkbox for "Time deadband 1000 ms" (unchecked).
- Checkbox for "Value deadband 10.00 %" (unchecked).

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.

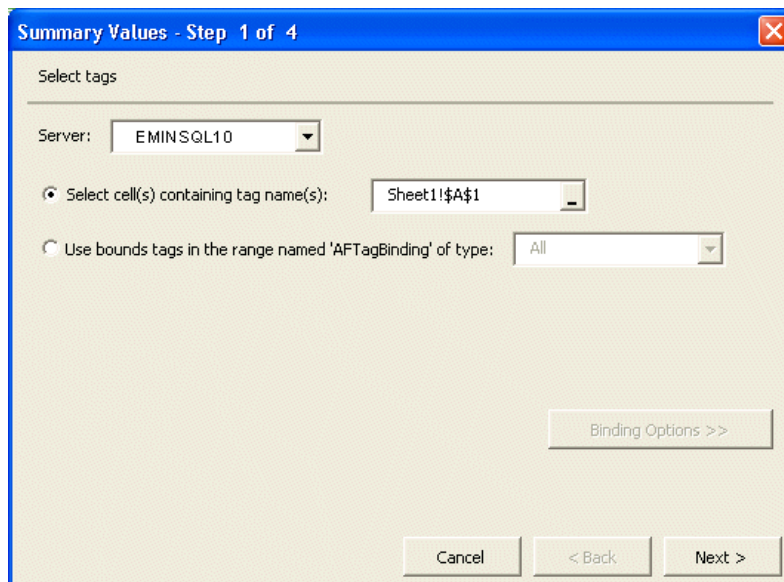
- **Value deadband:** The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval.
- **Time deadband:** The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.

Retrieving Values for Summarized Tags

You can retrieve summary values for tags that have been configured to be summarized by the Wonderware Historian event subsystem.

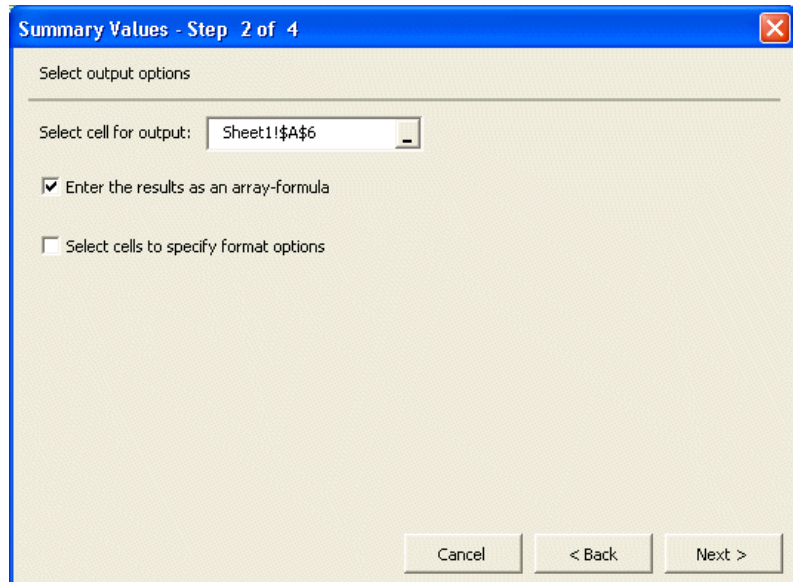
To retrieve summary system values

- 1 In cells in your worksheet, enter one or more tag names (one tag name per cell).
- 2 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Values**, and then click **Summary System Values**. The **Summary Values - Step 1 of 4** dialog box appears.
 - If you are using Excel 2007, 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.



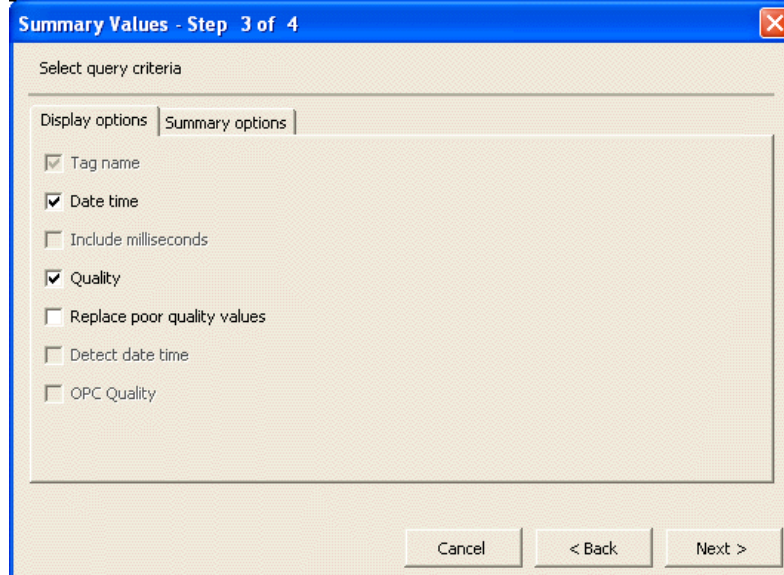
- 3 In the **Servers** list, click the name of the server to use.

- 4 In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see [Selecting Cells](#) on page 231.
- 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](#) on page 321.
- 6 Click **Next**. The **Summary Values - Step 2 of 4** dialog box appears.

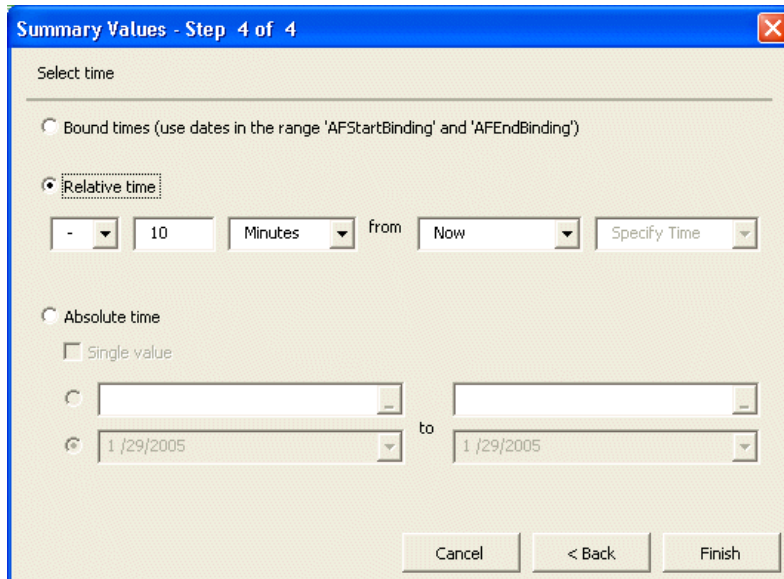


- 7 In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see [Selecting Cells](#) on page 231.
- 8 Select the **Enter the results as an array-formula** check 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](#) on page 223.
- 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](#) on page 231.

- 10 Click **Next**. The **Summary Values - Step 3 of 4** dialog box appears.

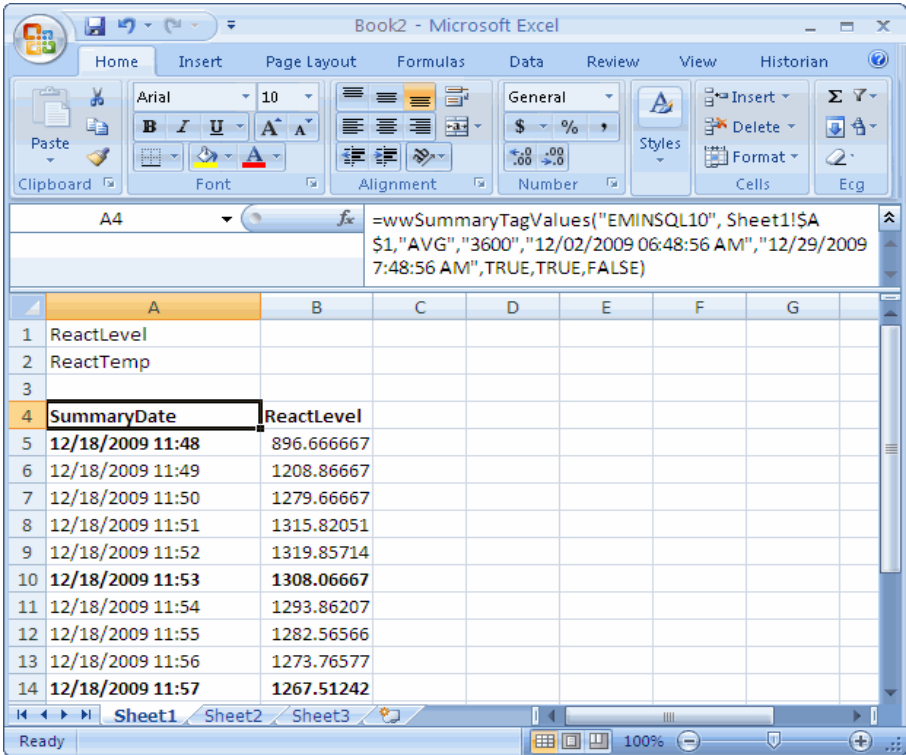


- 11 Configure the criteria for the query.
See **Display Options Tab** on page 271 and **Summary Options Tab** on page 267.
- 12 Click **Next**. The **Summary Values - Step 4 of 4** dialog box appears.



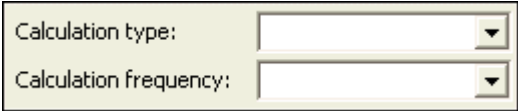
- 13 Configure the time for the query. For more information on configuring these options, see **Time Options for Queries** on page 325.

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, click 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

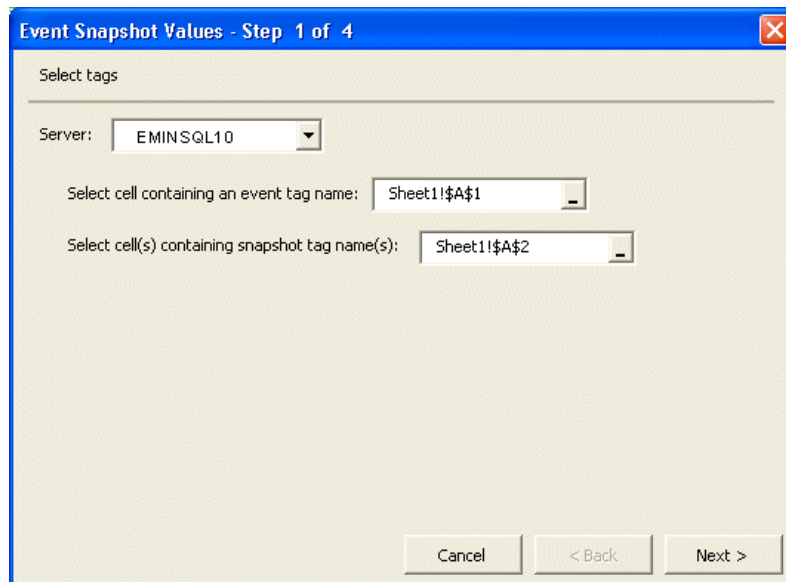
- 1 In cells in your worksheet, enter one or more tag names (one tag name per cell).

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](#) on page 233.

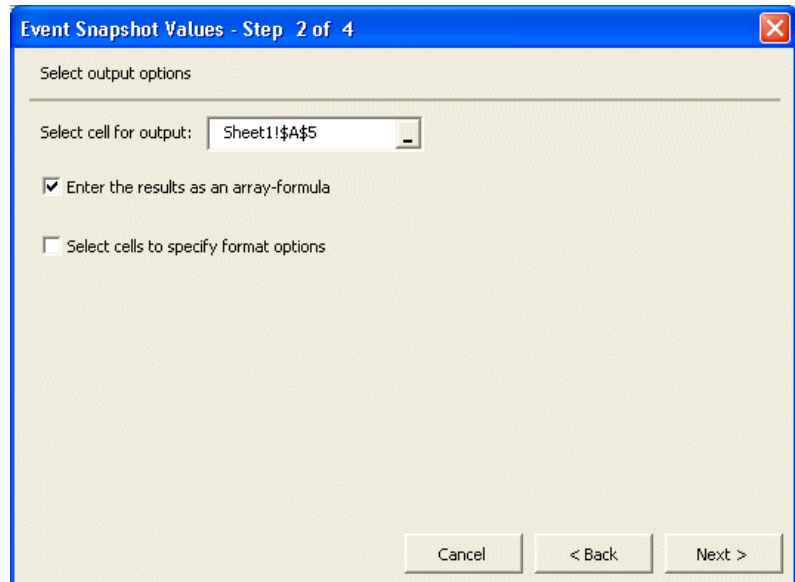
For information on selecting a snapshot tag, see [Selecting Event Snapshot Tags](#) on page 238. Select the tag(s) associated with the event tag you selected.

- 2 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Tag Values**, and then click **Event Snapshot Values**. The **Event Snapshot Values - Step 1 of 4** dialog box appears.
 - If you are using Excel 2007, 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.



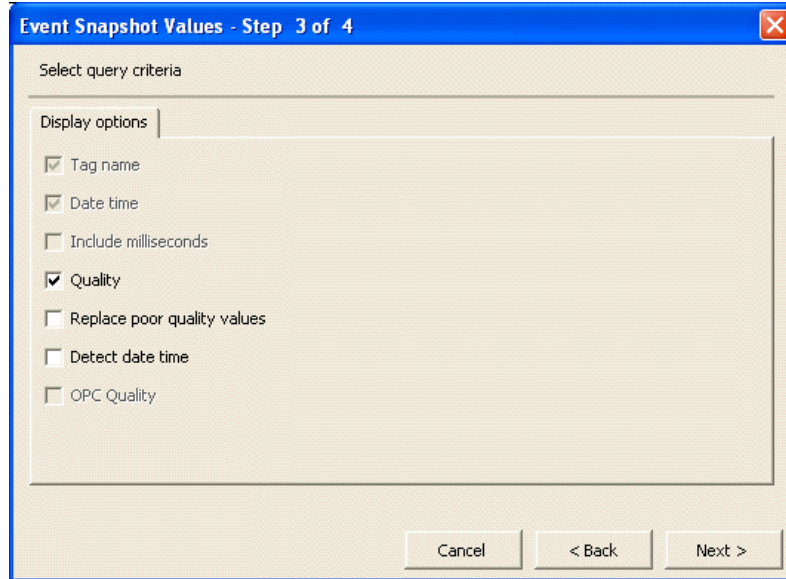
- 3 In the **Servers** list, click the name of the server to use.
- 4 In the **Select cell(s) containing tag name(s)** list, specify the location of the worksheet cell(s) that contains the tag name(s). Click on the button to select the cell(s). For more information, see [Selecting Cells](#) on page 231.

- 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](#) on page 321.
- 6 Click **Next**. The **Event Snapshot Values - Step 2 of 4** dialog box appears.

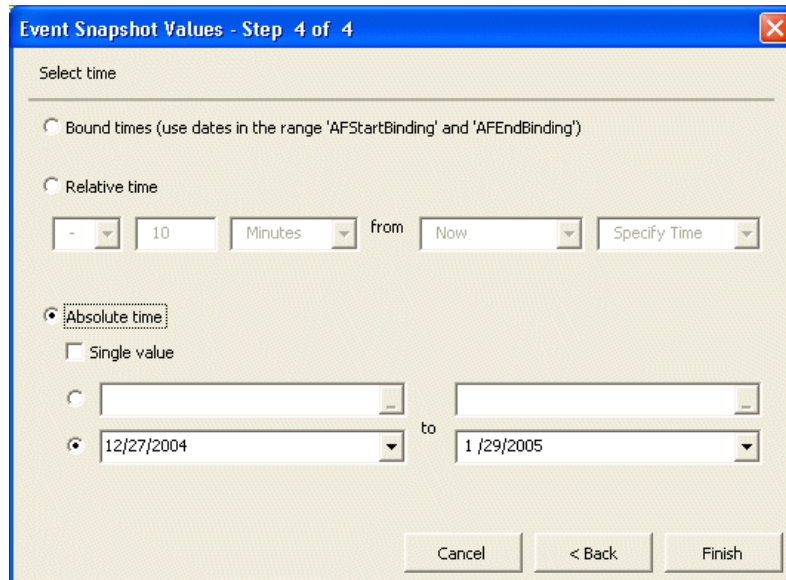


- 7 In the **Select cell for output** list, specify the location of the worksheet cell(s) that will contain the output. Click on the button to select the cell(s) using your mouse. For more information, see [Selecting Cells](#) on page 231.
- 8 Select the **Enter the results as an array-formula** check 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](#) on page 223.
- 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](#) on page 231.

- 10 Click Next. The Event Snapshot Values - Step 3 of 4 dialog box appears.

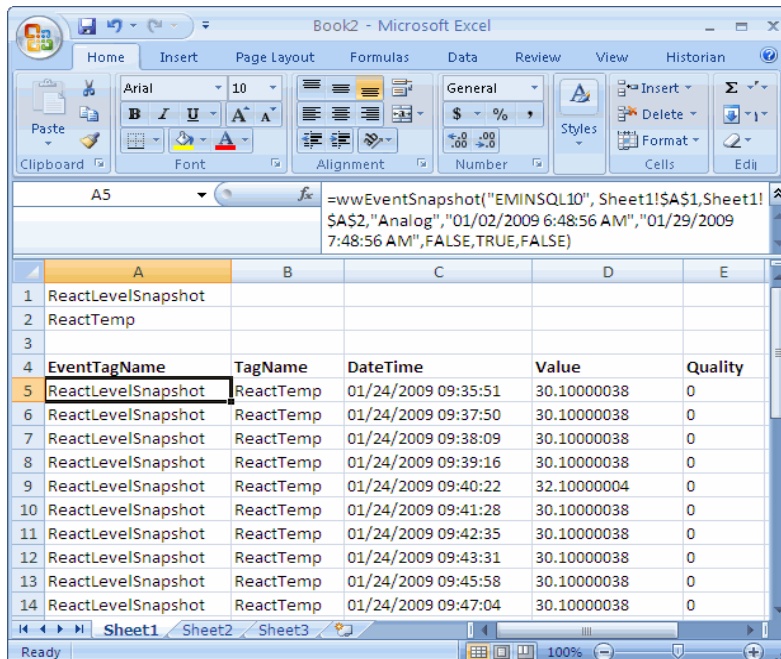


- 11 Configure the criteria for the query.
For more information, see Display Options Tab on page 271.
- 12 Click Next. The Event Snapshot Values - Step 4 of 4 dialog box appears.



- 13 Configure the time for the query. For more information on configuring these options, see Time Options for Queries on page 325.

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.

- Tag name
- Date time
- Include milliseconds
- Quality
- Replace poor quality values
- Detect date time
- OPC Quality

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 Wonderware Historian server. If the data values are coming from ArcestrA, the attribute reference is shown as the tag name. For ArcestrA attributes, you can also choose to show the hierarchical name along with the attribute reference. For more information, see ArcestrA Naming Conventions on page 24.
- **Date time:** The time stamp for the returned value. For delta retrieval, this is the time at which the value was acquired by the Wonderware 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 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** on page 274.

The screenshot shows the 'Format Tab' interface with two main sections: 'Retrieval options' and 'Presentation options'.

Retrieval options:

- Value based criteria (narrow tables)
- Tag based criteria (wide tables)

Presentation options:

- Narrow format
- Wide format

Two example tables are shown below the options:

TagName	DateTime	vValue
SysCPU 0	1/20/2005	2
SysCPU 1	1/20/2005	3
SysCPU 2	1/20/2005	0

DateTime	SysCPU0	SysCPU1
1/20/2005	0	2
1/20/2005	0	5
1/20/2005	0	5

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** on page 273.

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.

The screenshot shows a form with four criteria rows. Each row starts with a checkbox, followed by a text input field containing 'SysTimeSec', a dropdown menu with '...', and a comparison operator. The first row has the operator 'IS NOT NULL'. The second and third rows have the operator '>' and a numeric input field containing '0', followed by the text 'units'. The fourth row has the operator 'equals' and an empty text input field. At the bottom, there is a dropdown menu labeled 'Criteria applicability:' with 'Not used' selected.

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 screenshot shows a form with five criteria rows. Each row starts with a checkbox, followed by a text input field containing 'Value', a dropdown menu with '>', and a numeric input field containing '0'. The second row has a checkbox, the text 'and', a dropdown menu with '>', and a numeric input field containing '0'. The third row has a checkbox, the text 'Value', a dropdown menu with 'equals', a numeric input field containing '0', and a checkbox labeled 'Use StringHistory'. The fourth row has a checkbox, the text 'Value not null', and a dropdown menu with 'Not used'. The fifth row has a checkbox, the text 'OPC Quality:', and a dropdown menu with 'Not used'. At the bottom, there is a dropdown menu labeled 'Criteria applicability:' with 'Not used' selected.

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 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 Wonderware 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.

Analyzing Tag Data

In addition to creating value reports, you can use the Wonderware 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.

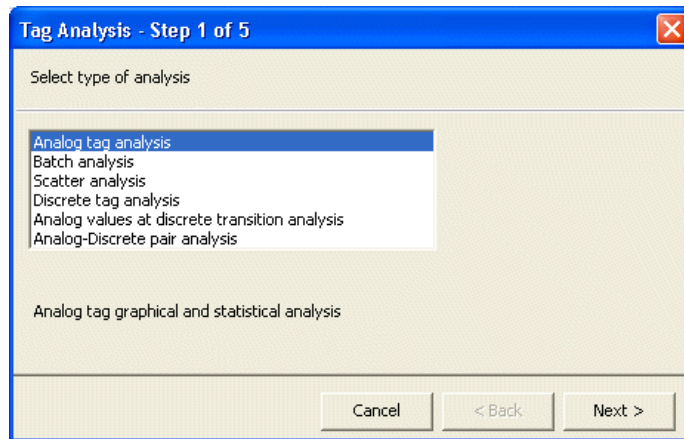
Wizards are provided to guide you through selecting the required options to create the output.

Analog Tag Analysis

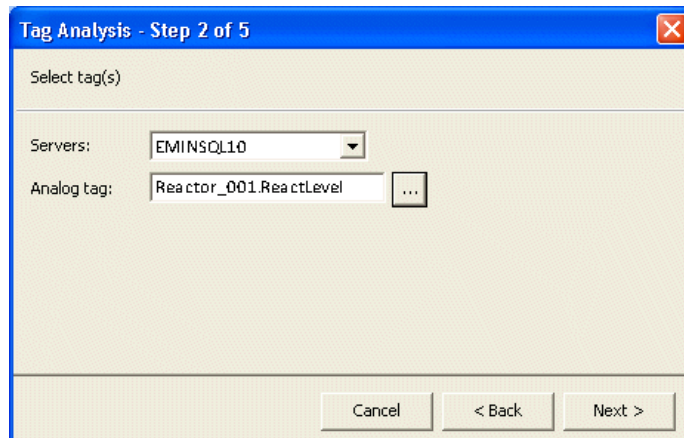
Use the Analog Tag Analysis wizard to generate graphs and statistics for an analog tag.

To analyze an analog tag

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
 - If you are using Excel 2007, 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.



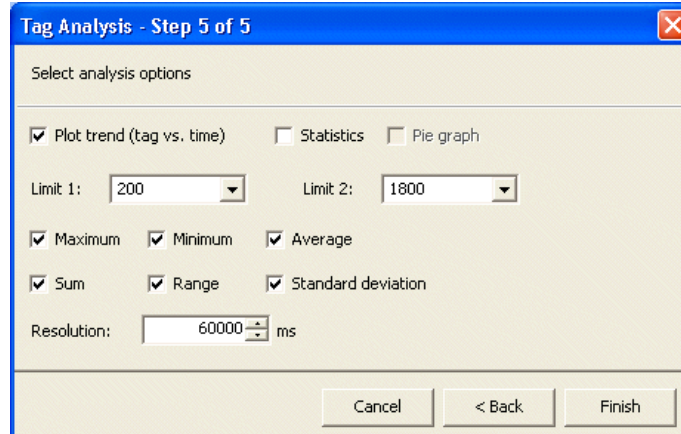
- 4 In the **Servers** list, click the name of the server to use.
- 5 In the **Analog tag** list, specify the name of the tag to analyze. Click the ellipsis button to open the **Tag Picker** and browse for the tag. For more information, see **Tag Picker** on page 40.

- 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.

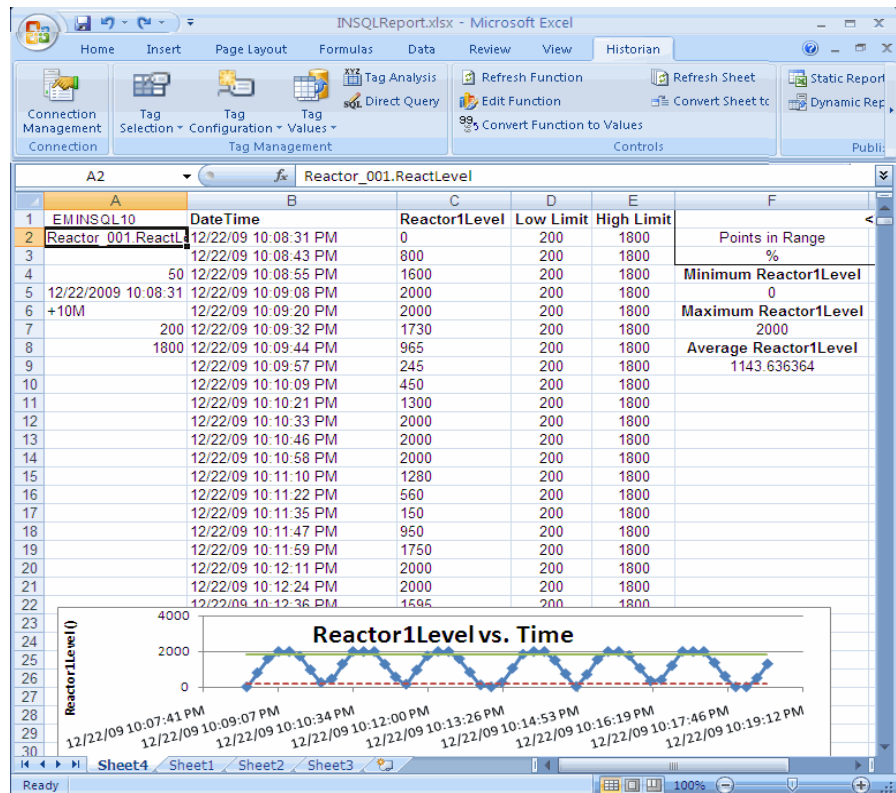
- 10 Configure the resolution for the data to be returned.
- **Number of rows:** The number of rows to be returned for a specified time period using cyclic retrieval. The rows are spaced evenly across the time period, and the default row count is 50 rows. The row count is applied for each tag in a query.
 - **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.
 - **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 are assigned to cells in the worksheet. For this particular example:

A1: Server

A2: Tag

A4: Row or resolution

A5: Start Time

A6: Duration

A7: Low Limit

A8: High Limit

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:

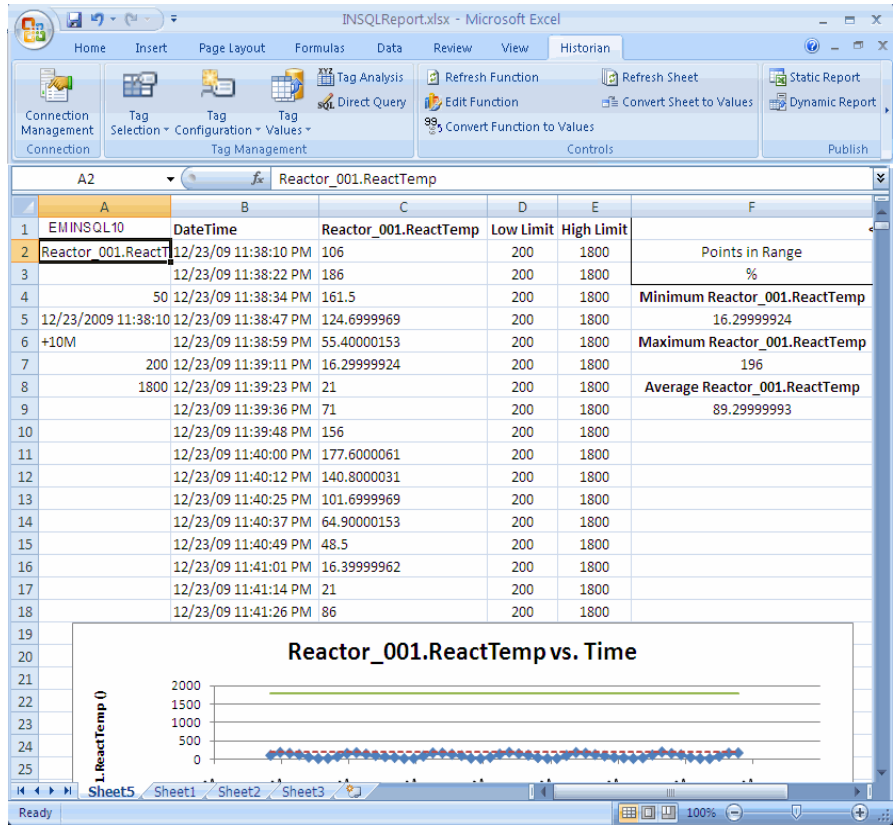
```
=wwAggregateWide(A1,A2,"ResFull","Rel",A6 & "(" & A5 & ")", "Min", "")
```

- 15 Optionally alter the results by changing values that appear in the first column of the worksheet.

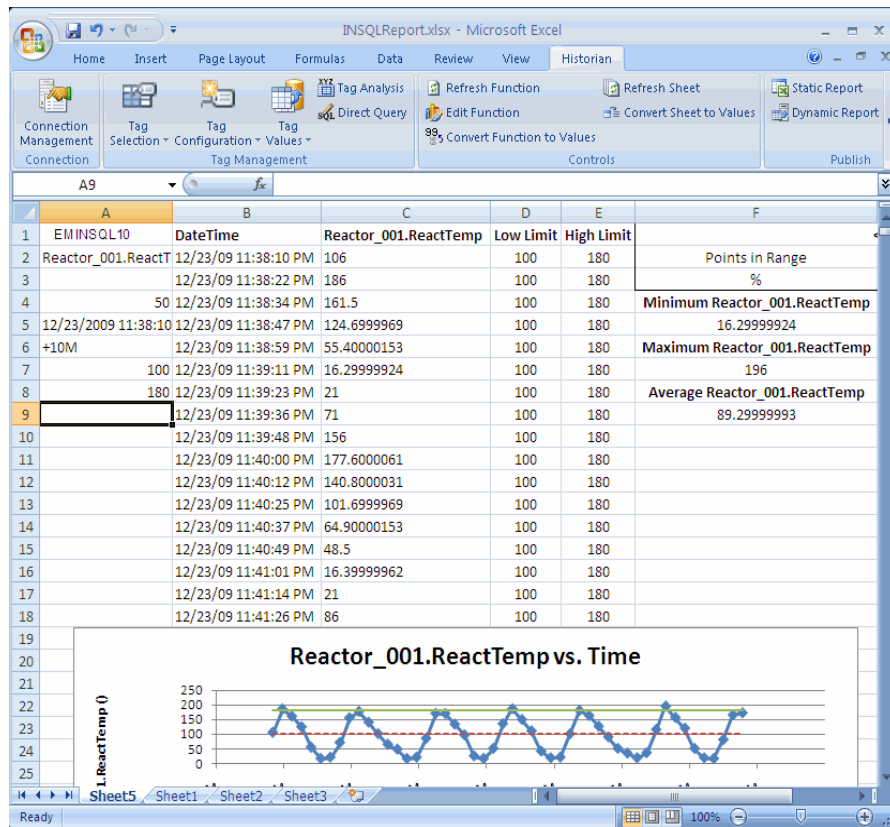
If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

Note If you change the resolution and/or duration, this may change the cells that are referenced by the graph and statistics.

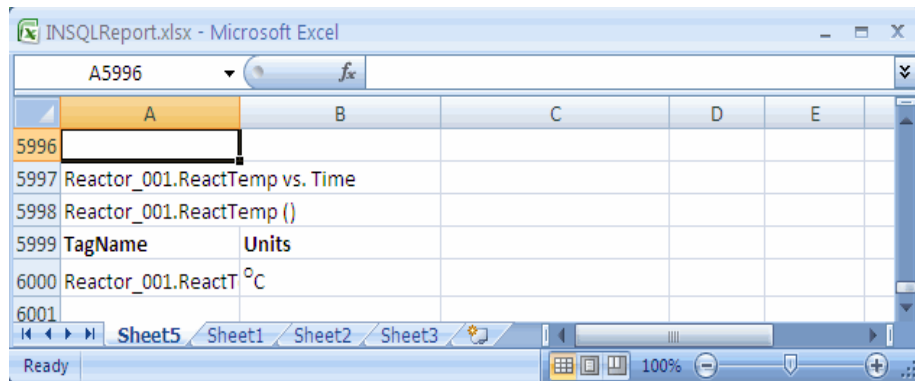
For example, you can change the tagname from ReactLevel to ReactTemp.



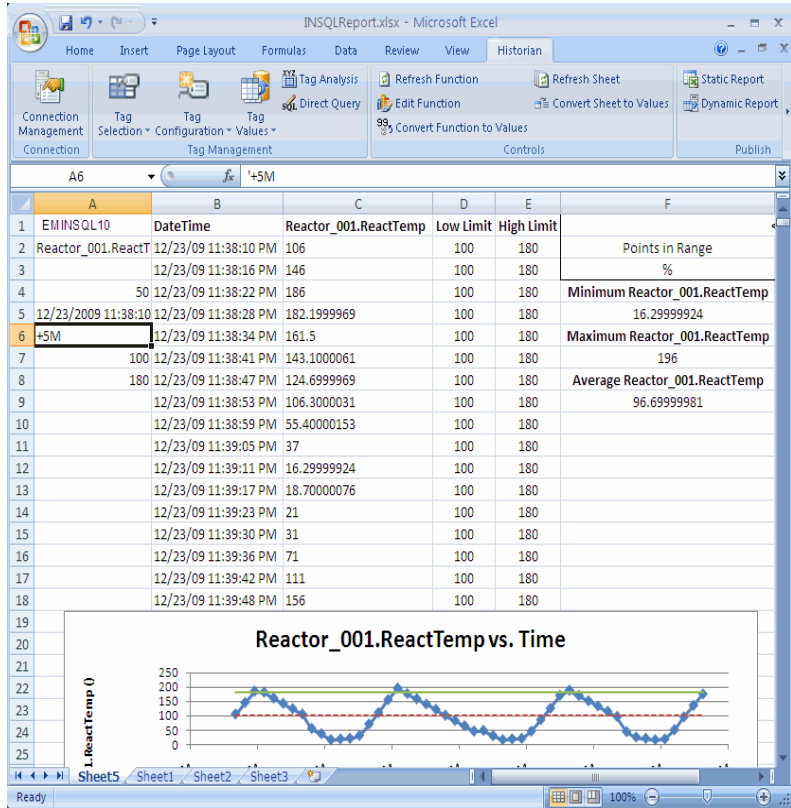
Change the limits to the correct limits so that the statistics and trend chart reflect the data.



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.

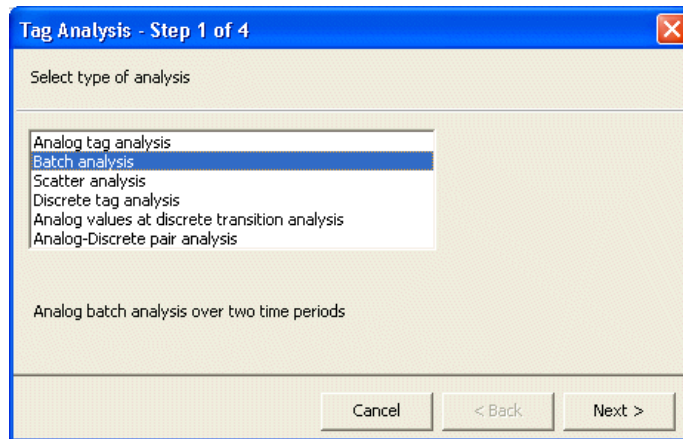


Batch Analysis

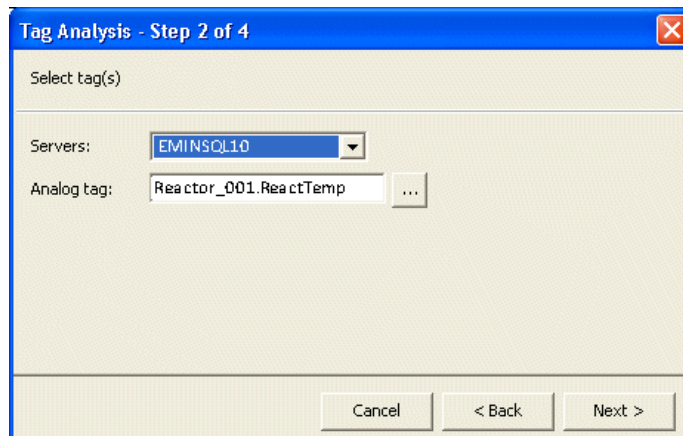
Use the Batch Analysis wizard to graph one analog tag over two time periods.

To create a batch analysis

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
 - If you are using Excel 2007, 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.



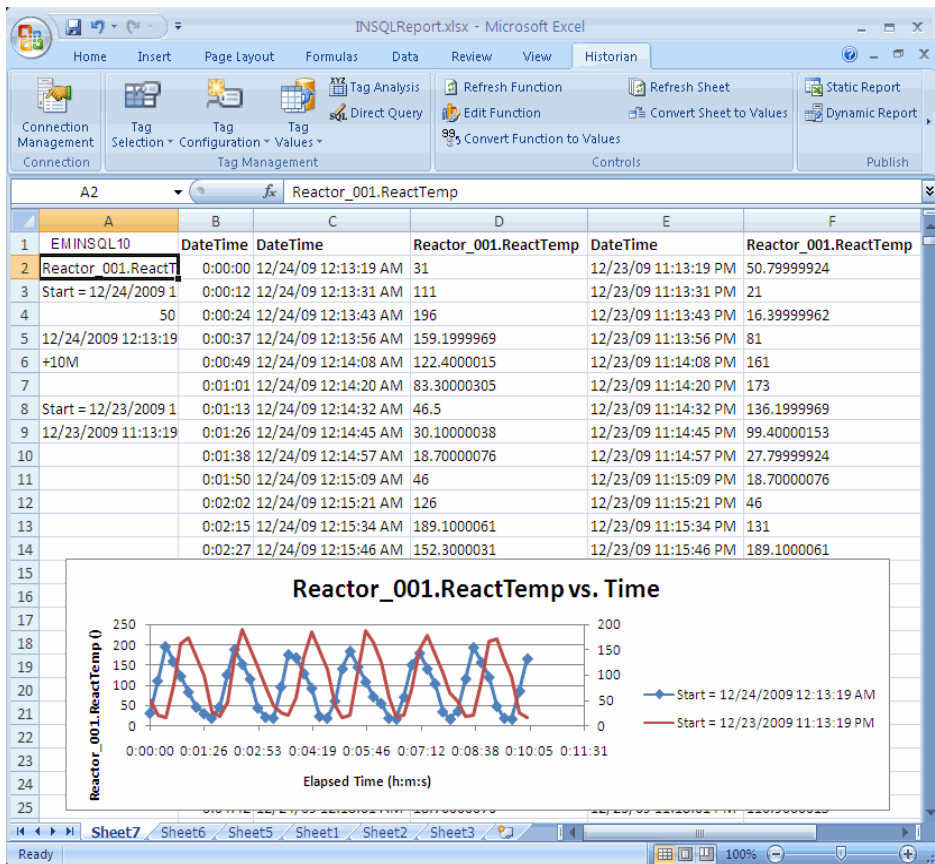
- 4 In the **Servers** list, click the name of the server to use.
- 5 In the **Analog tag** list, specify the name of the tag to analyze. Click the ellipsis button to open the **Tag Picker** and browse for the tag. For more information, see **Tag Picker** on page 40.

- 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 are assigned to cells in the worksheet. For this particular 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

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:

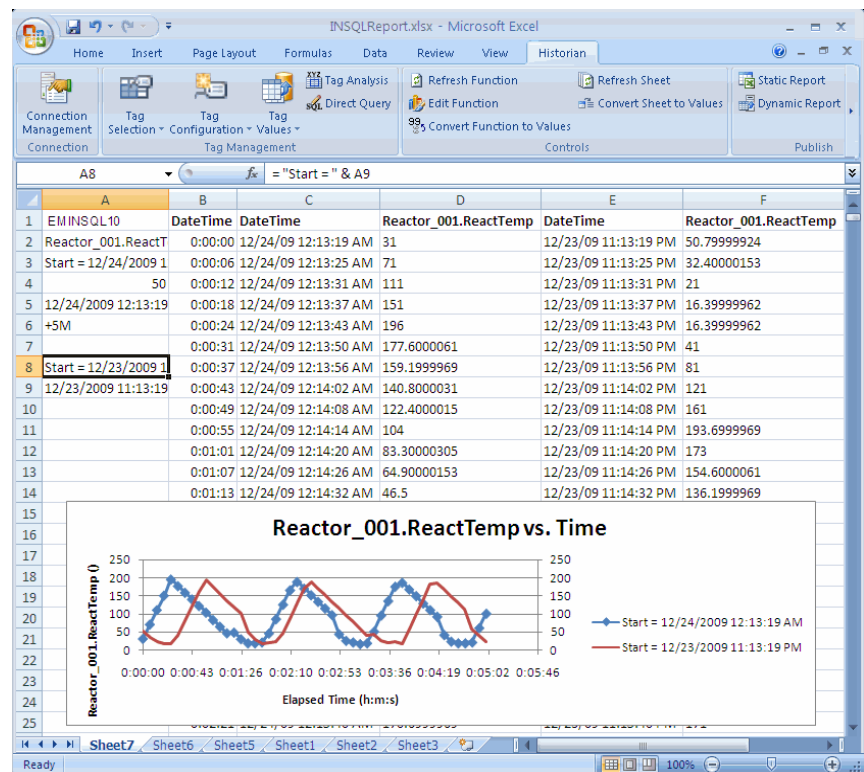
```
=wwAnalogWideHistory(A1,A2,"Res" & A4,"Rel",A6 & "(" & A9 & ")",FALSE," ",TRUE,FALSE)
```

- 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.

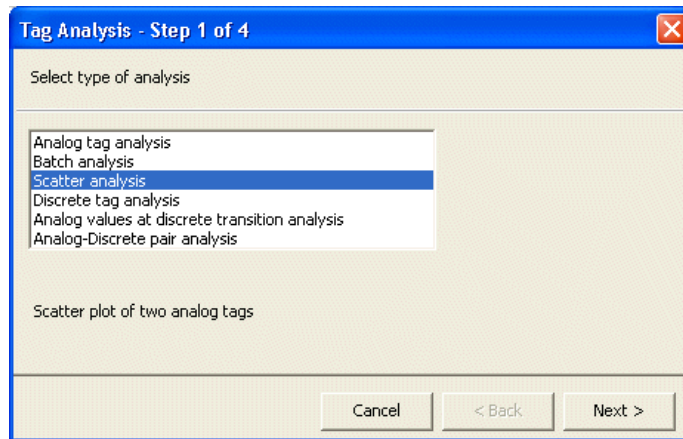


Scatter Analysis

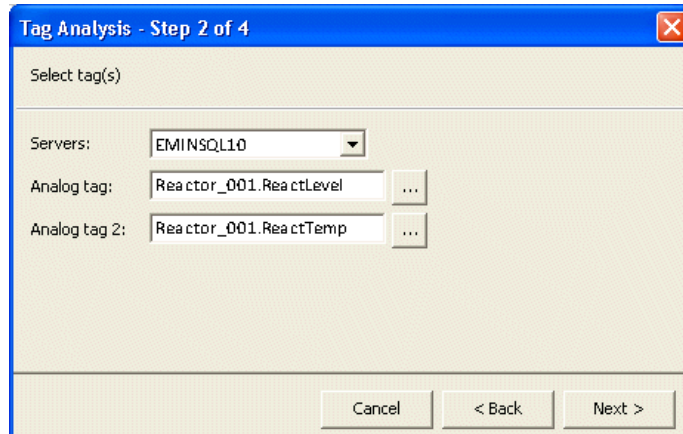
Use the Scatter Analysis wizard to create a scatter plot of two analog tags.

To create a scatter plot

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
 - If you are using Excel 2007, 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.



- 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** on page 40.

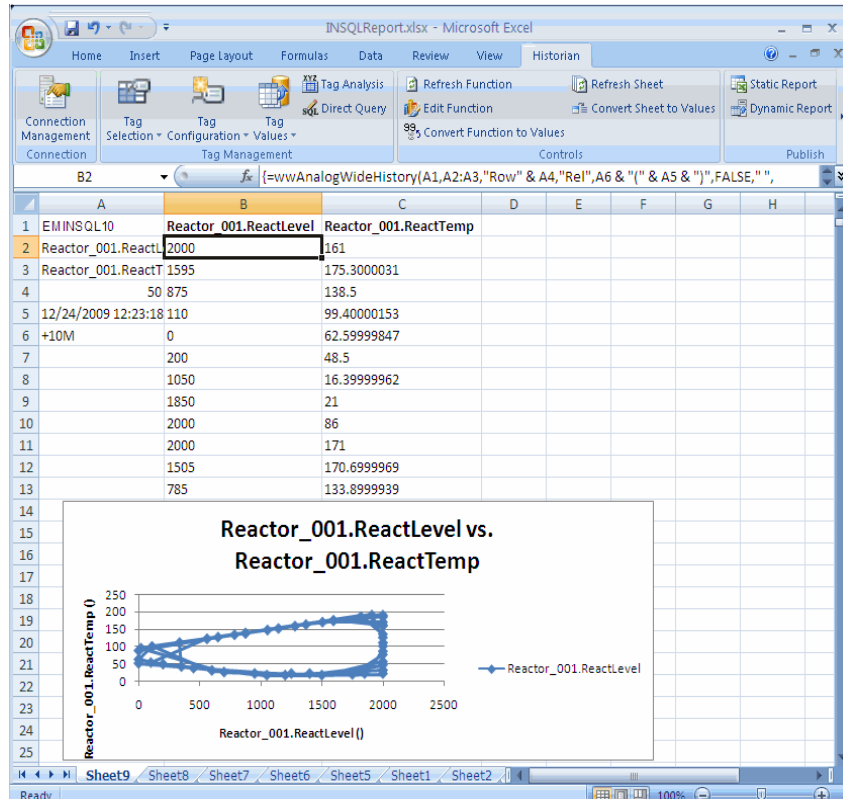
- 6 Click Next. The Tag Analysis - Step 3 of 4 dialog box appears.

- 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 are assigned to cells in the worksheet. For this particular example:

- A1: Server
- A2: Tag 1
- A3: Tag 2
- A4: Row or resolution
- 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.

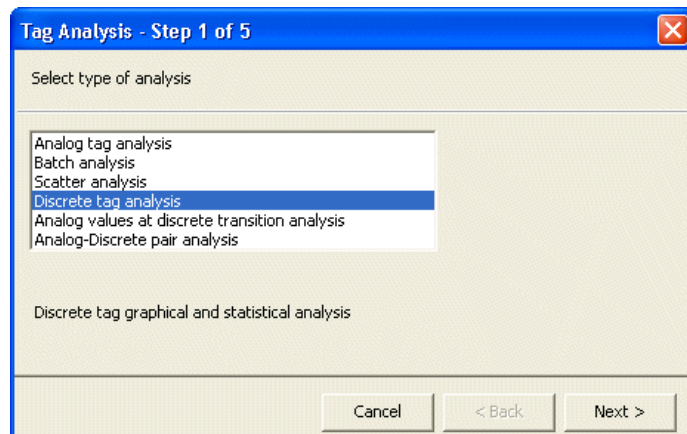
If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

Discrete Tag Analysis

Use the Discrete Tag Analysis wizard to create graphs and trends, calculate statistics, and return configuration information.

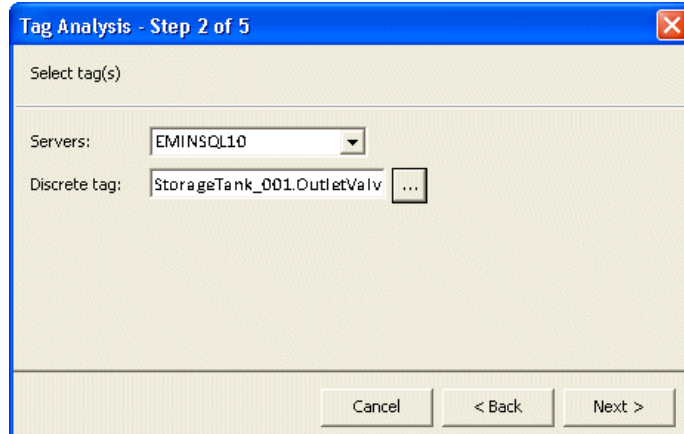
To analyze a discrete tag

- Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.

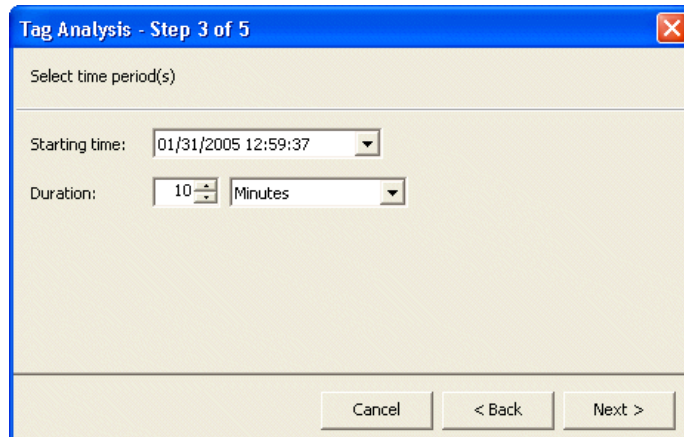


- Select **Discrete tag analysis**.

- 3 Click **Next**. The **Tag Analysis - Step 2 of 5** dialog box appears.

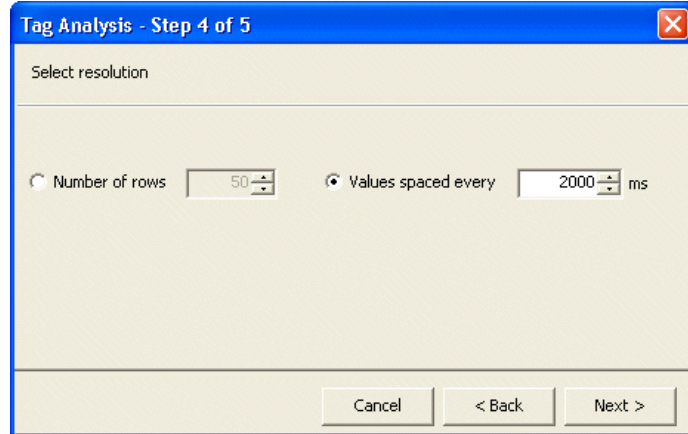


- 4 In the **Servers** list, click the name of the server to use.
- 5 In the **Discrete tag** list, specify the name of the tag to analyze. Click the ellipsis button to open the Tag Picker and browse for the tag. For more information, see Tag Picker on page 40.
- 6 Click **Next**. The **Tag Analysis - Step 3 of 5** dialog box appears.

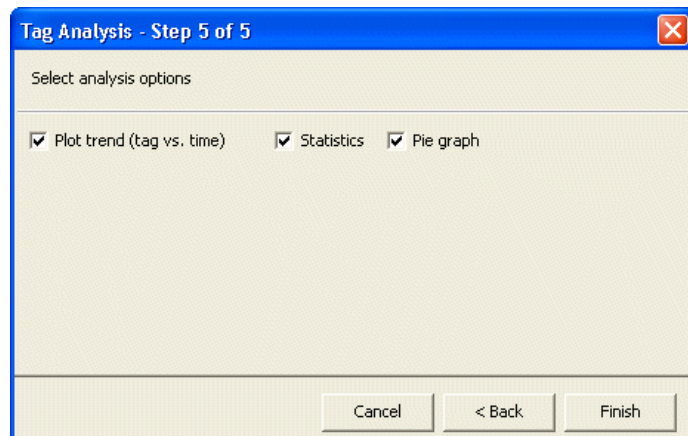


- 7 In the **Starting time** list, enter the starting time for the query. Click the arrow button to select a date from a calendar.
- 8 In the **Duration** lists, specify the duration and the duration unit. For example, 10 minutes. The duration is used to calculate the end date for the query.

- 9 Click Next. The Tag Analysis - Step 4 of 5 dialog box appears.

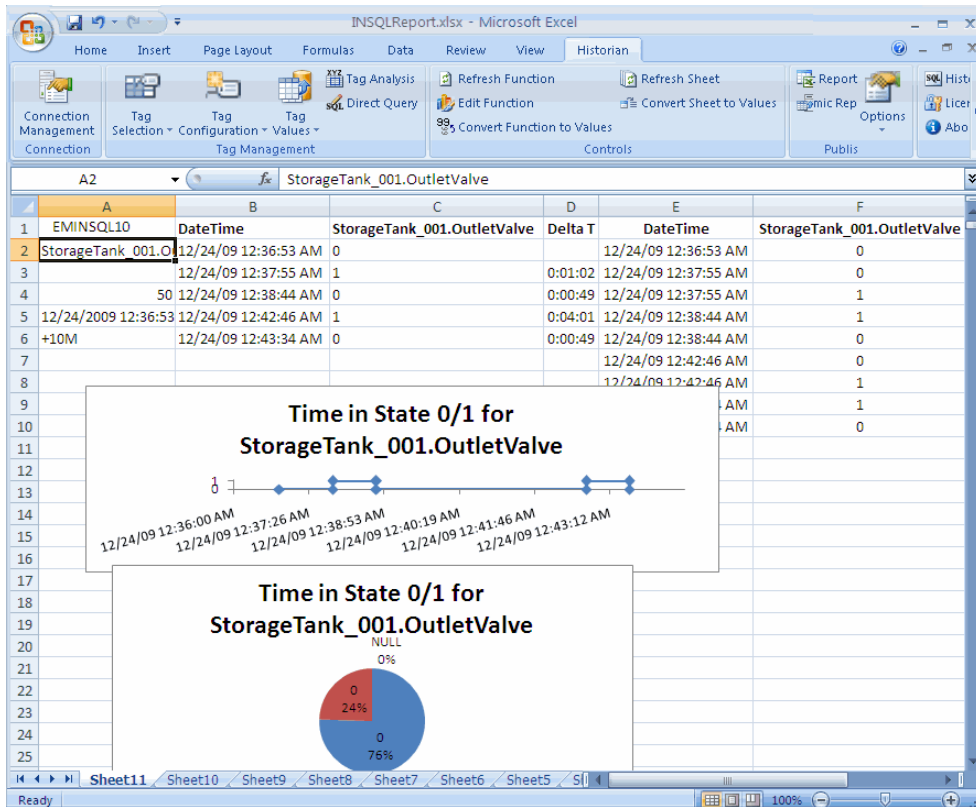


- 10 Configure the resolution for the data to be returned.
- **Number of rows:** The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 50 rows. For cyclic retrieval, the row count is applied 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.
 - **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 are assigned to cells in the worksheet. For this particular 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)

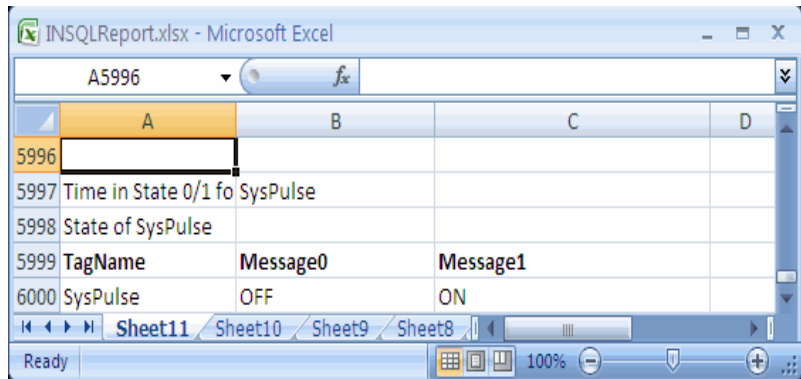
```
=COUNTIF(C3:C12, 0)
```

Columns E and F contain both transitions for each of the dates in Column B.

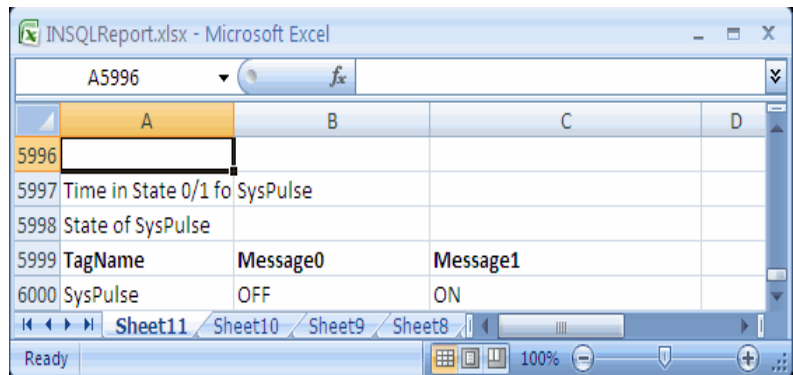
- 15 Optionally alter the results by changing values that appear in the first column of the worksheet.

If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

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.

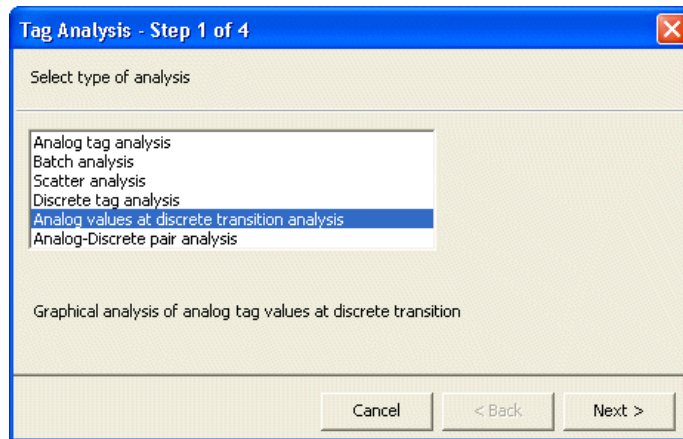


Analog Values at Discrete Transition Analysis

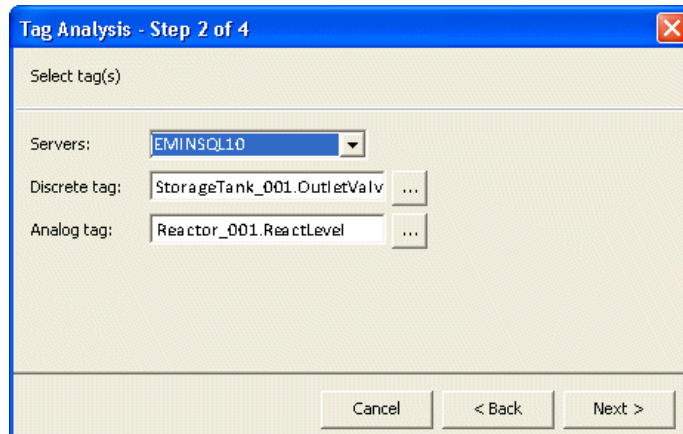
Use the Analog Values at Discrete Transition Analysis wizard to graph analog tag values at discrete tag transitions.

To analyze an analog tag at a discrete transition

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
 - If you are using Excel 2007, 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.



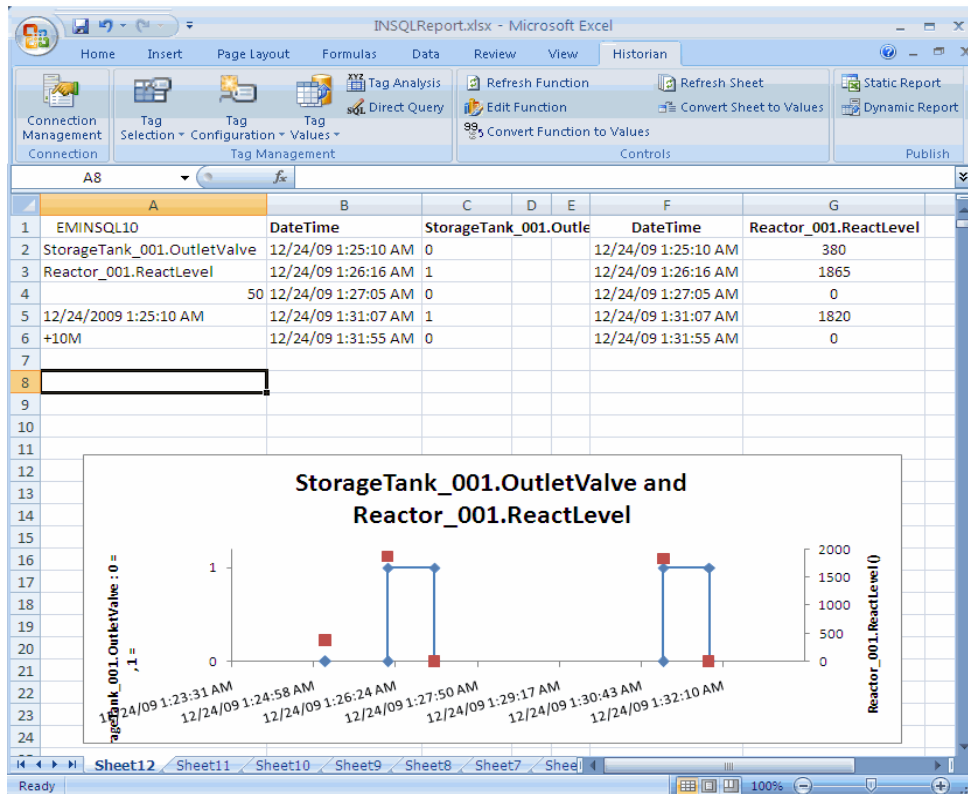
- 4 In the **Servers** list, click the name of the server to use.
- 5 In the **Analog tag** and **Discrete tag** lists, specify the names of the tags to analyze. Click the ellipsis button to open the **Tag Picker** and browse for the tags. For more information, see **Tag Picker** on page 40.

- 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 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, 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 are assigned to cells in the worksheet. For this particular example:

- A1: Server
- A2: Discrete Tag
- A3: Analog Tag
- A4: Row or resolution
- A5: Start Time
- A6: Duration

- 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:

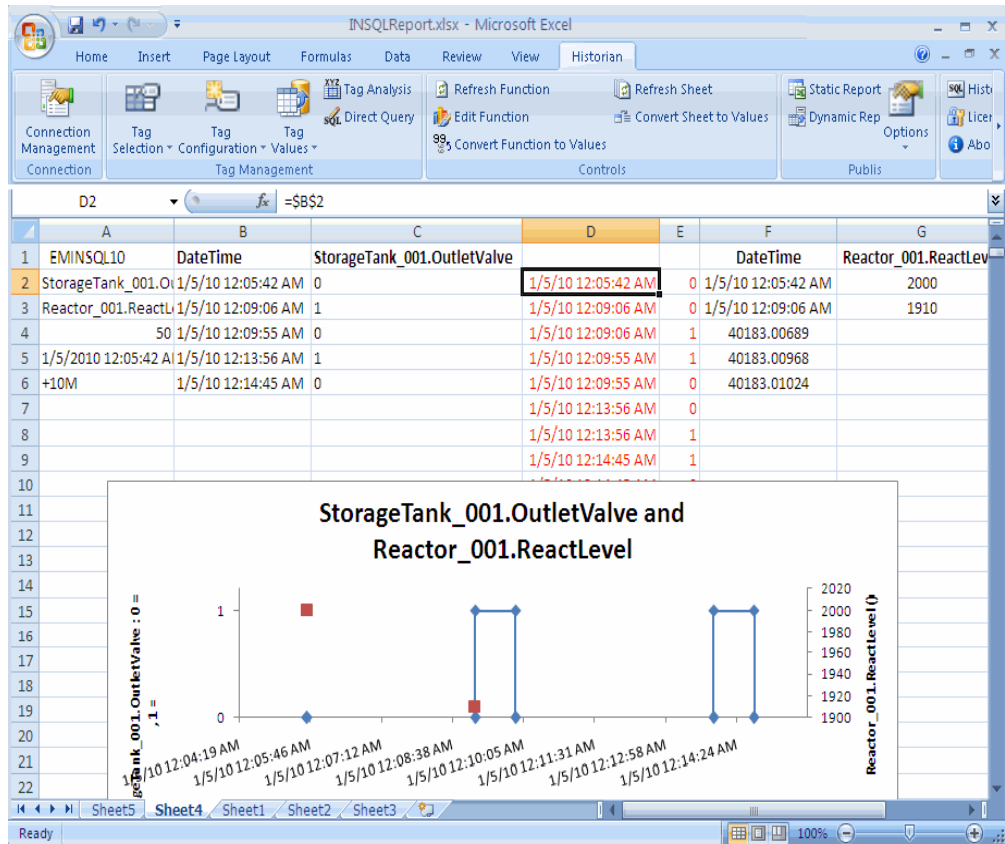
=wwDiscreteWideHistory(A1,A2,"Row" & A4,"Rel",A6 & "(" & A5 & ")",",",TRUE, FALSE)

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.



- 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.

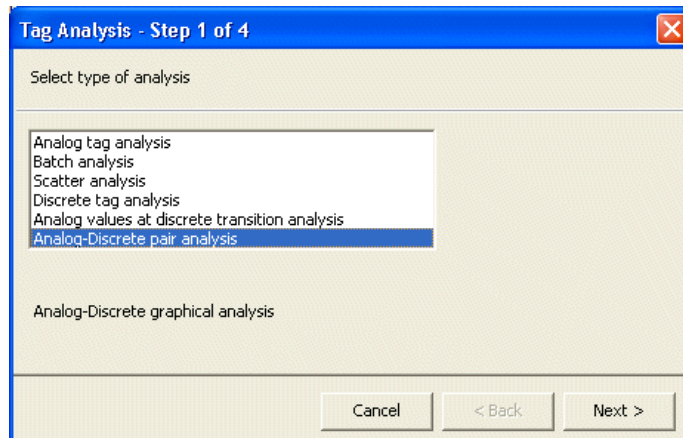
Keep in mind that changing the time arguments may return a different number of rows in the result set, causing the analysis to be incorrect.

Analog/Discrete Pair Analysis

Use the Analog/Discrete Pair Analysis wizard to graph analog vs. discrete tags.

To analyze an analog-discrete pair

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Tag Analysis**. The **Tag Analysis** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Tag Management** group, click **Tag Analysis**. The **Tag Analysis** dialog box appears.



- 2 Select **Analog-Discrete pair analysis**.

- Click Next. The Tag Analysis - Step 2 of 4 dialog box appears.

- In the Servers list, click the name of the server to use.
- 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 on page 40.
- Click Next. The Tag Analysis - Step 3 of 4 dialog box appears.

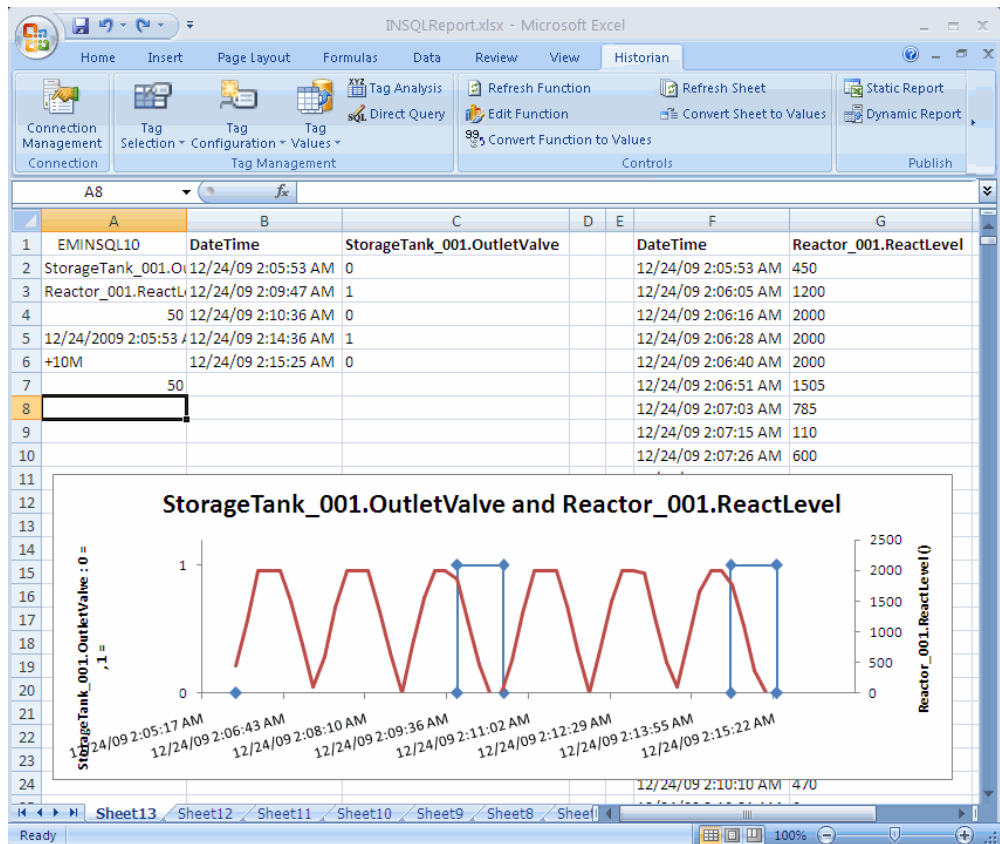
- In the Starting time list, enter the starting time for the query. Click the arrow button to select a date from a calendar.
- In the Duration lists, specify the duration and the duration unit. For example, 10 hours. The duration is used to calculate the end date for the query.

- 9 Click **Next**. The **Tag Analysis - Step 4 of 4** dialog box appears.

The screenshot shows a dialog box titled "Tag Analysis - Step 4 of 4". The dialog has a close button in the top right corner. The main area is titled "Select resolution". It contains two sections. The first section is for "WaterValve" with a radio button selected for "Number of rows" set to 20, and another radio button for "Values spaced every" set to 2000 ms. The second section is for "ReactLevel" with a radio button selected for "Number of rows" set to 20. At the bottom are "Cancel", "< Back", and "Finish" buttons.

- 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 Finish.



Information that you specified using the wizard are assigned to cells in the worksheet. For this particular 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.

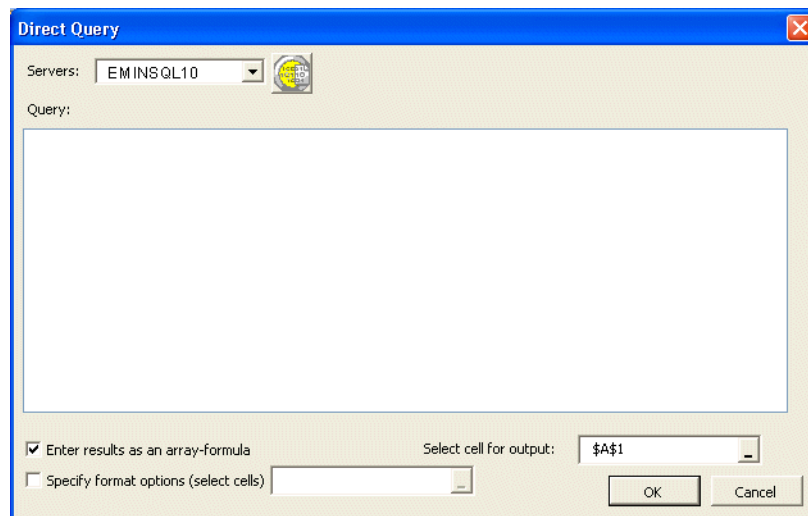
If the calculation mode is set to automatic, any changes you make to the values in this column are immediately reflected in the worksheet.

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.

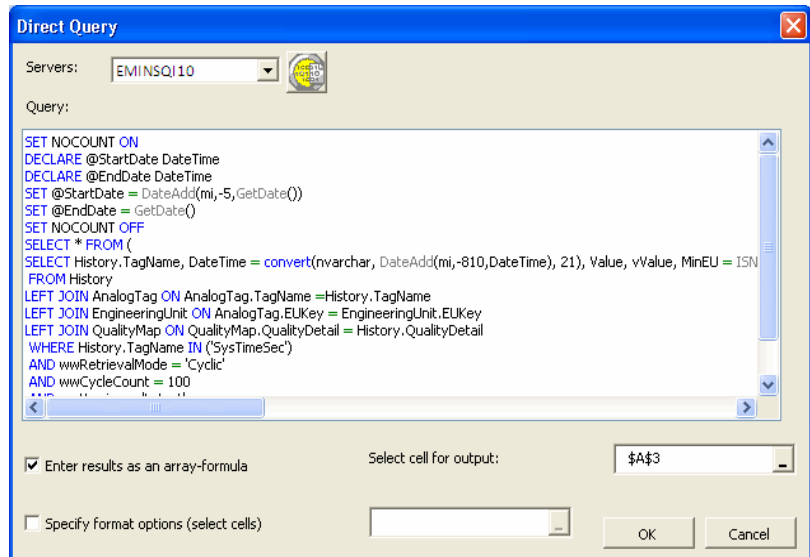
To perform a direct query

- Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Direct Query**. The **Direct Query** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Tag Management** group, click **Direct Query**. The **Direct Query** dialog box appears.



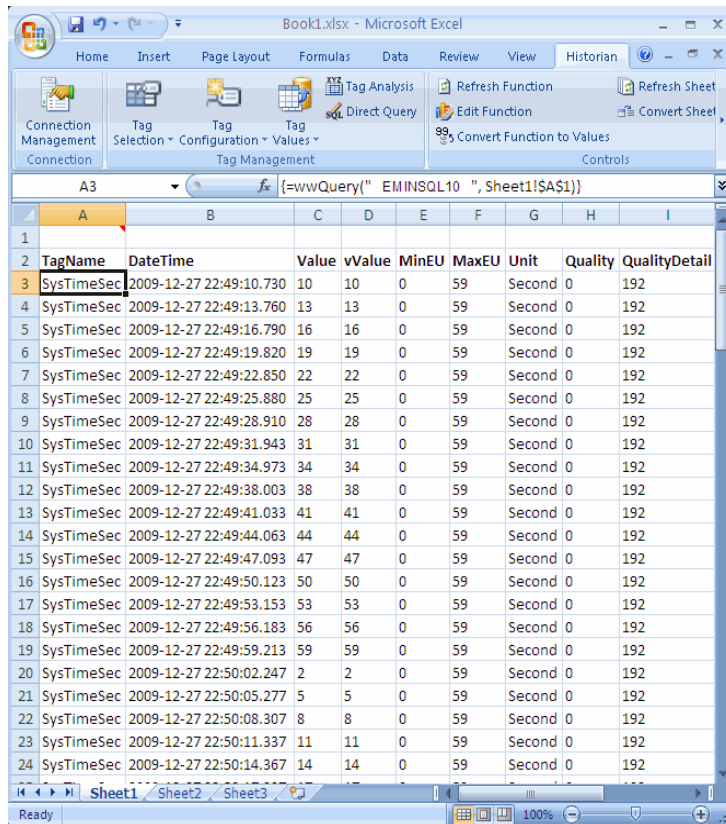
- 2 In the **Servers** list, click the name of the server to use.
- 3 In the **Query** window, type the SQL query to execute against the database.

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 window. For more information, see Chapter 4, Wonderware Historian Client Query.

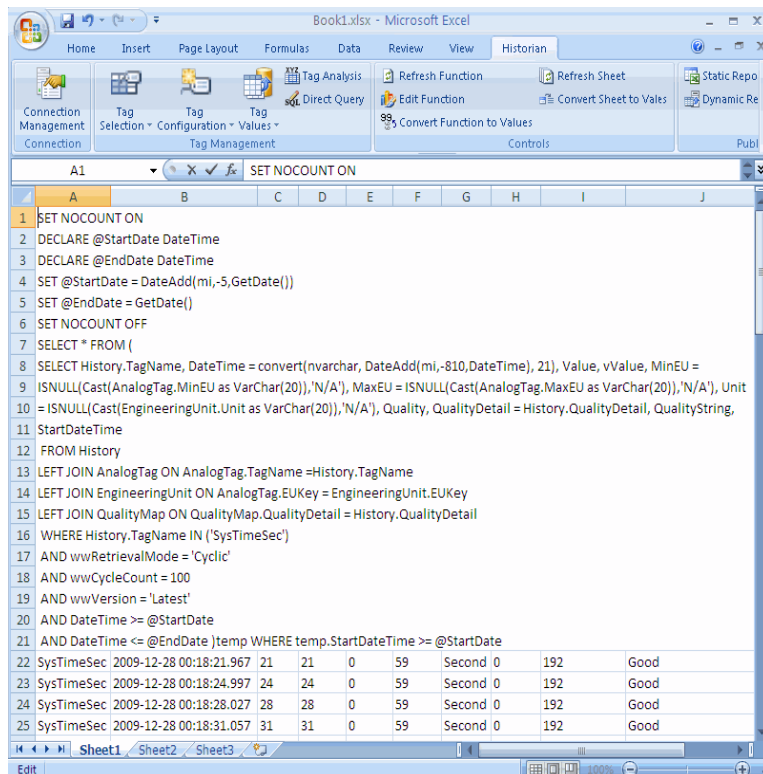


- 4 Select the **Enter the results as an array-formula** check box to insert the results as an array formula. An array formula can perform one or more calculations and then return either single result or multiple results. An array formula allows for the resending of the query, since the query parameters are included in the cells that contain the query results. For more information, see Working with Functions, Formulas, and Cells on page 223.
- 5 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 on page 231.
- 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 on page 231.

7 Click OK.



8 To edit the query, click in the cell that contains the red triangle.



Configuring Workbook Options

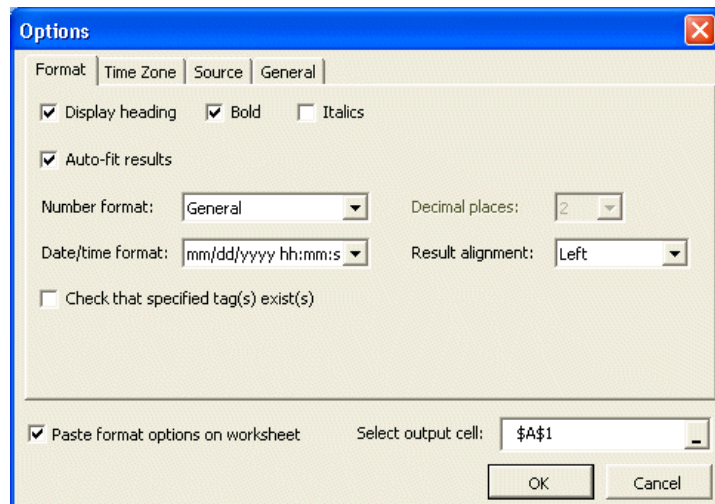
You can configure global settings related to formatting, time zone usage, data sources, and other general options. You can also set values for formatting and date/time options and then reference them from functions in your workbook. Finally, you can set up custom filters for your reports.

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 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Options** and then click **Options**. The **Options** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
- 2 Click the **Format** tab.



- 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.

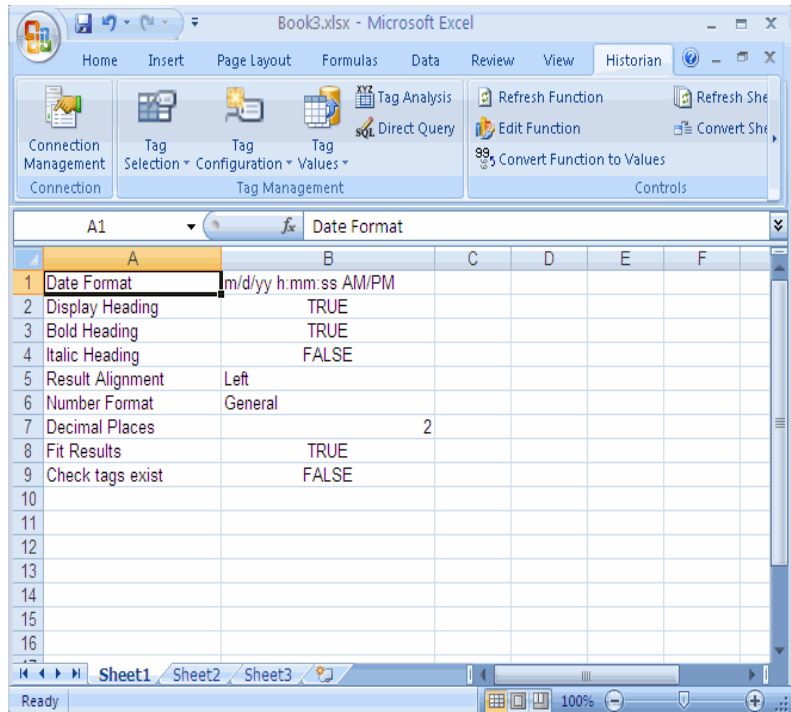
- Override the date/time format for the timestamp by using the **Select output cell list** option in Step 10 to contain the formatting settings. Then, reference the **Date Format** cell in your query to control the timestamp format for the returned data values. You can change the **Date Format** cell 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](#) on page 231.

11 Click **OK**.

If you select to output the formatting information, it appears in the sheet.



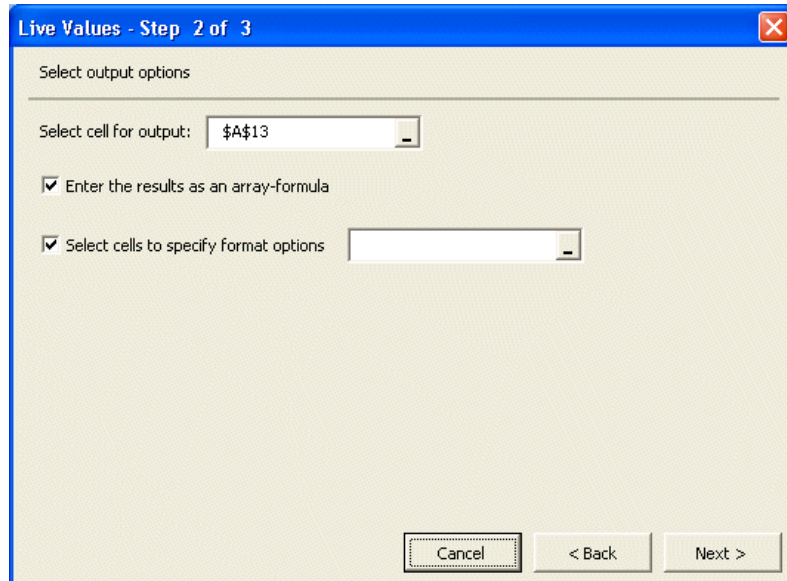
Referencing Formatting Options in a Query

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* on page 309.

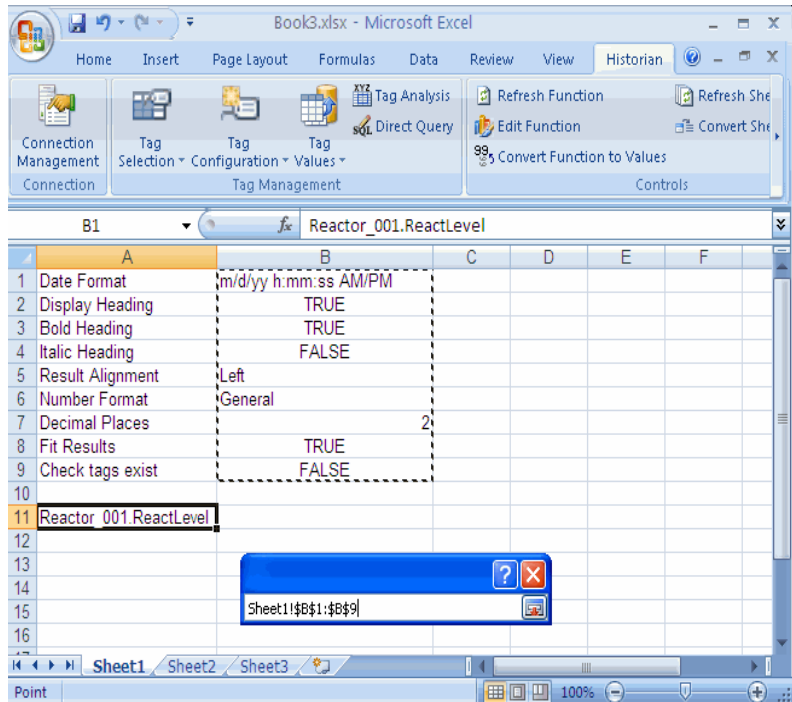
The following procedure assumes that you are using a wizard to create the query.

To reference formatting options in a query

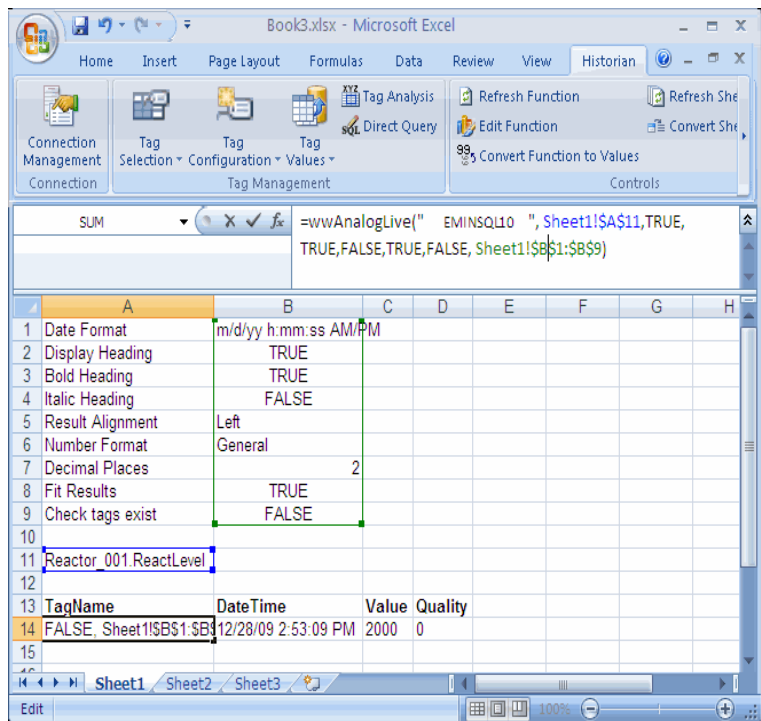
- 1 In the wizard, select the **Select cells to specify format options**.



- 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](#) on page 231.



The query results are formatted according to the option settings. Note that the function references the formatting option cells.

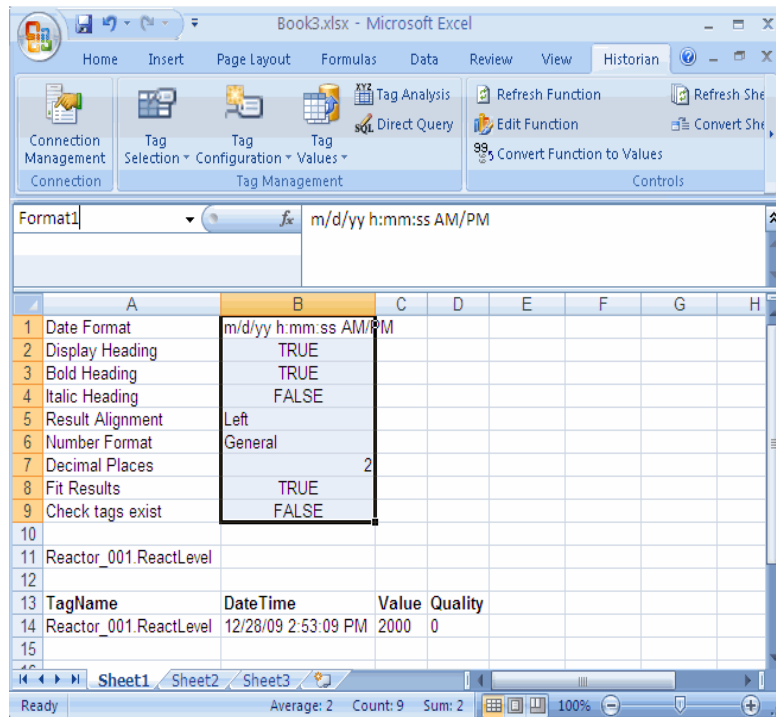


Using a Named Range for Formatting Options

Instead of referencing multiple cells that contain formatting options, you can give the group of cells a name and then reference just that name.

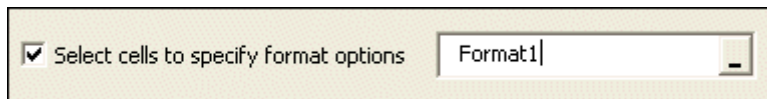
To use a named range

- 1 Insert the formatting options into a location in your worksheet. For more information, see *Configuring Global Formatting Options* on page 309.
- 2 Select the cells that contain the options.
- 3 In the Name Box list, type a name for the cell range.

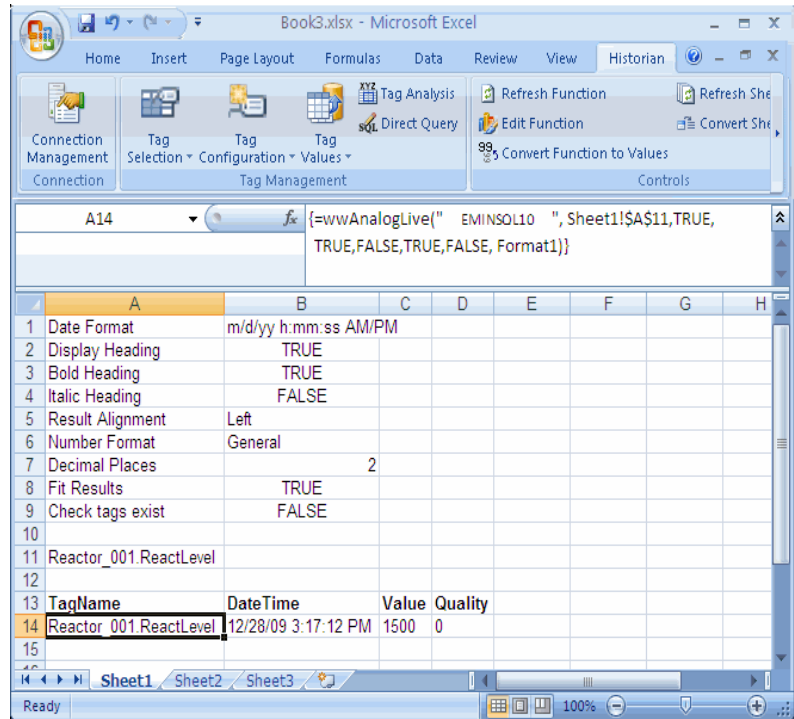


- 4 Press Enter on your keyboard.

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.

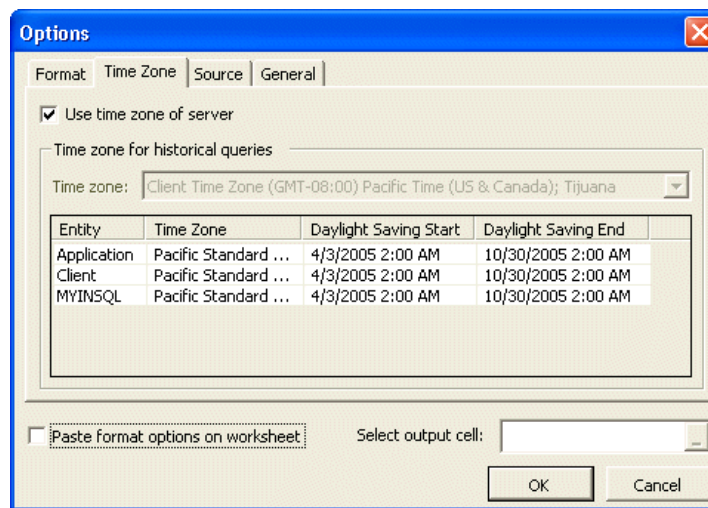
The new formatting options are applied to the results.

Configuring Time Zone Options

The time zone settings are applied to all functions.

To configure time zone options

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Options** and then click **Options**. The **Options** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Publish** group, click **Options**, and then click **Options**. The **Options** dialog box appears.
- 2 Click the **Time Zone** tab.



The grid displays the current time zone and daylight savings time settings for the following entities:

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

- 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.

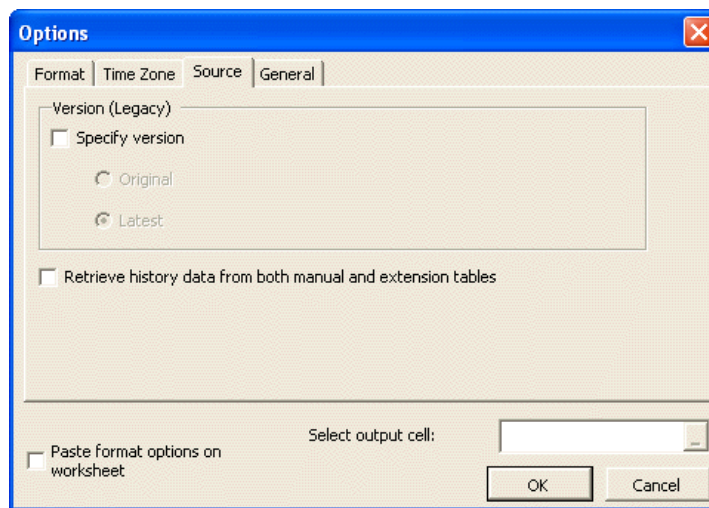
- 4 Click **OK**.

Configuring Data Source Options

The data source settings are applied to all functions.

To configure data source options

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Options** and then click **Options**. The **Options** dialog box appears.
 - If you are using Excel 2007, 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 a Wonderware 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 Wonderware Historian.
 - **Latest:** The latest value that is stored in the Wonderware 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.
 - **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 Wonderware Historian data files. These tables support the Wonderware Historian time domain extensions for handling data.

5 Click OK.

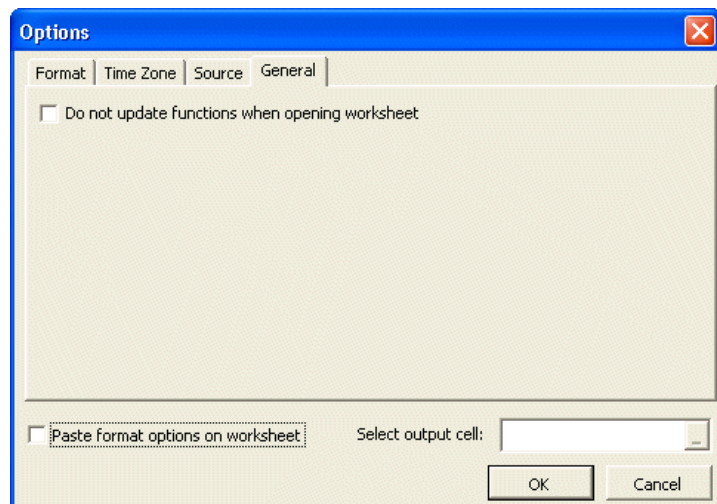
Configuring General Options

To configure general options

1 Do one of the following:

- If you are using Excel 2003 or XP, On the **Historian** menu, point to **Options** and then click **Options**. The **Options** dialog box appears.
- If you are using Excel 2007, 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.

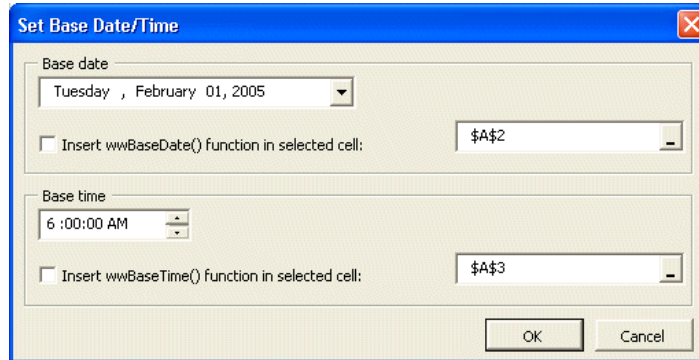
Setting the Base Date and Base Time Parameters

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.

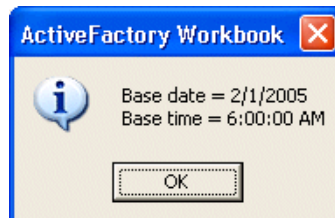
By using these parameters, you can create generic reports that accommodate any date/time; change the base date and base time for the workbook.

To set the base date and time

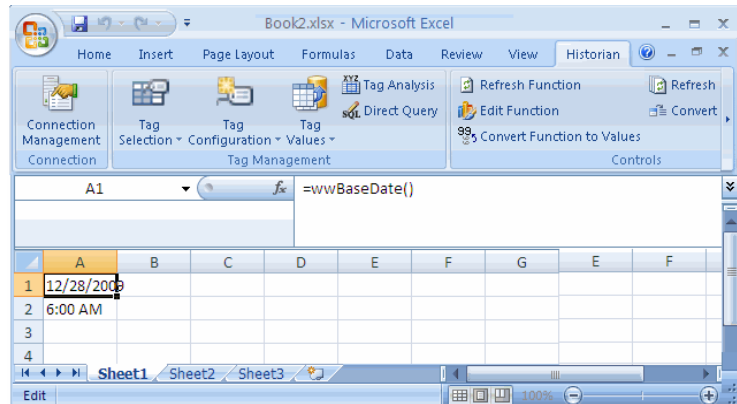
- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Options**, and then click **Set Base Date/Time**. The **Set Base Date/Time** dialog box appears.
 - If you are using Excel 2007, on the **Historian** tab, in the **Publish** group, click **Options**, 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](#) on page 231.
- 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](#) on page 231.
- 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 Wonderware 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 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.

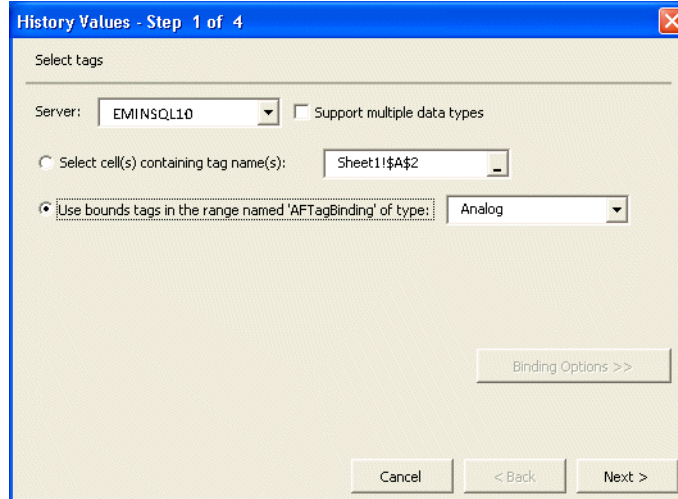
The binding values can be used in the following ways:

- Publish the workbook report as a dynamic report to the Wonderware 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 Wonderware Historian Client Workbook add-in or the WorkbookRunner object.

Creating a Bound Report

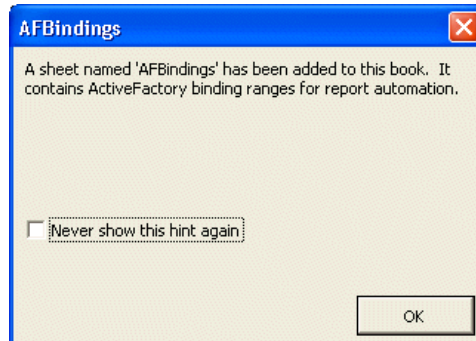
To create a "bound" report

- 1 Start one of the Workbook data retrieval wizards. For example, the History Values wizard.

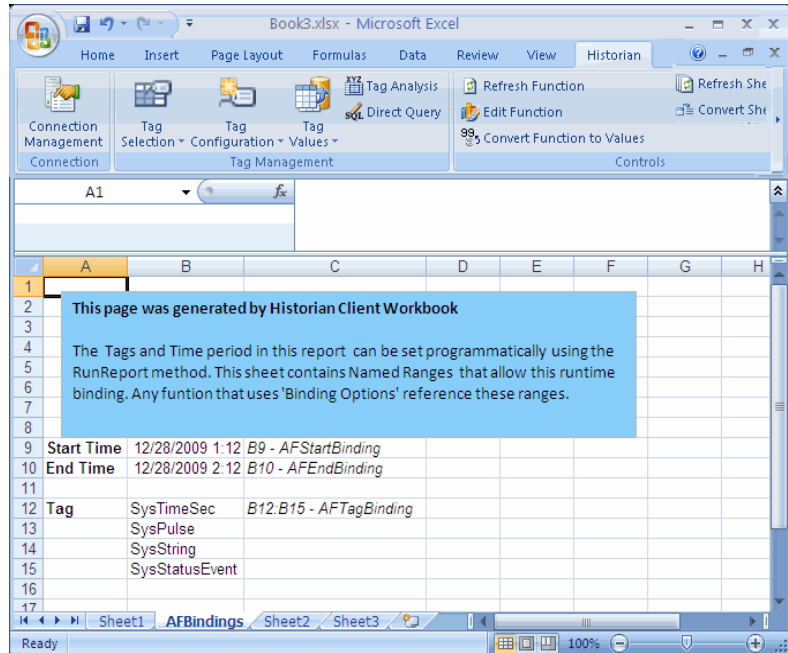


- 2 Click **Binding Options** to show the binding options.
- 3 Click **Use bound tags in the range named 'AFTagBinding' of type** and then select the type of tag from the list.

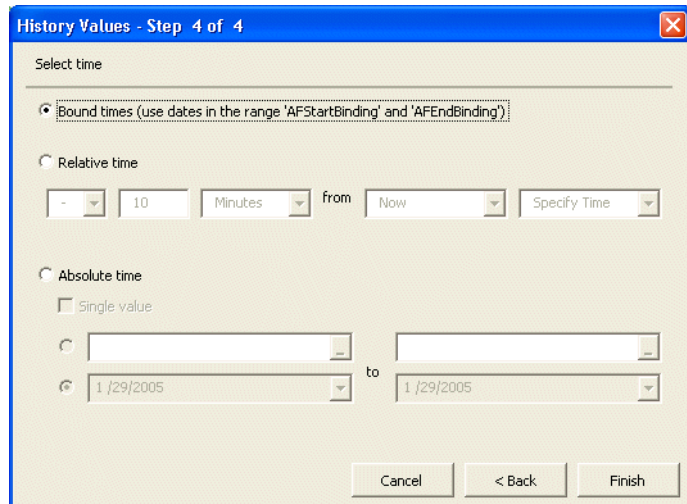
If the AFTagBinding range does not already exist for the current workbook, a new sheet is added to the workbook. You are prompted to confirm the creation.



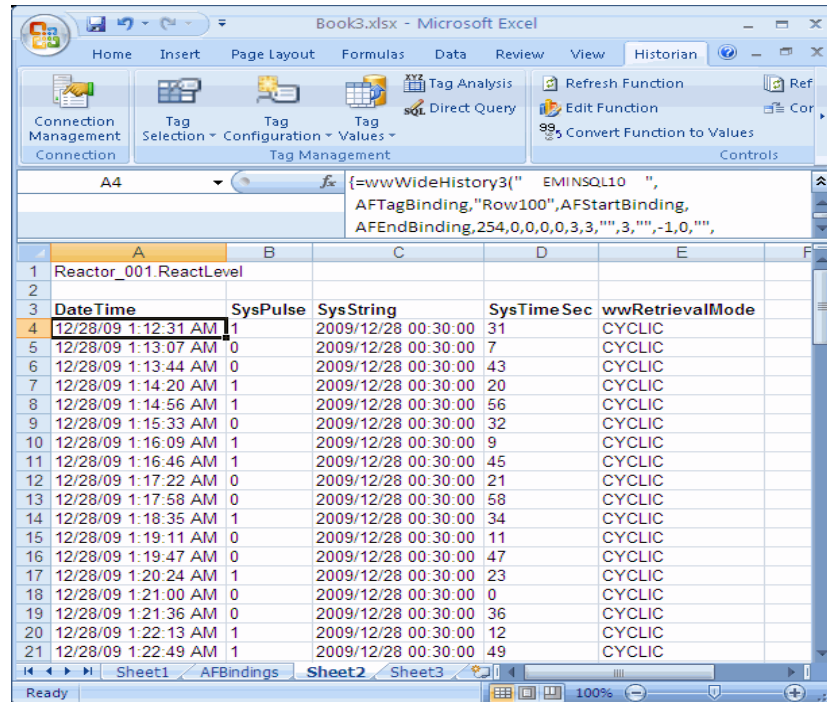
- Click OK. The AFBindings sheet is created for the current workbook.



- 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 Wonderware 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 cell, indicated by red background as unable to be resized.

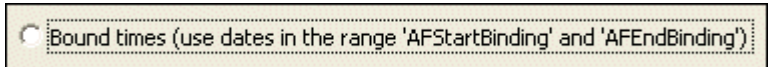
Time Options for Queries

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 on page 321.

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 on page 319.
- 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* on page 231.
 - 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.

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* on page 231.
 - 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 Wonderware Information Server. When you publish a report, the report information is stored in special tables in the Wonderware Historian, and the file is copied to a folder on the Wonderware Information Server. When you publish a report, Wonderware 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 Wonderware 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:

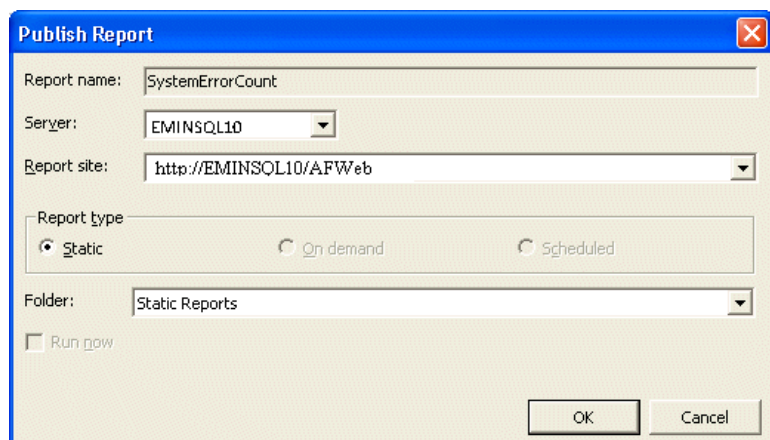
- Do not publish shared workbooks.
- If you create reports that use the `wwHistory2` and `wwWideHistory2` data retrieval functions introduced in ActiveFactory 9.2 (for example, reports created using the History Values wizard), do not publish them to a Wonderware Information Server.
- The Wonderware Information Server must be associated with the same Wonderware 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 Wonderware 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 `.xls` file.
- 2 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, point to **Publish**, and then click **Static Report**. The **Publish Report** dialog box appears.
 - If you are using Excel 2007, 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 Wonderware 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 Wonderware Historian on which to store the report publishing information.
- 4 In the **Report site** list, select the URL of the Wonderware Information Server to which you want to publish the report.

The report site may or may not be physically located on the Wonderware 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 Wonderware Information Server periodically scans the publishing folders for changes; a short delay may occur prior to the published report being displayed in the Wonderware Information Server.

- 8 To view the Wonderware 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 .xls file.
- 2 Do one of the following:
 - If you using Excel 2003 or XP, on the **Historian** menu, point to **Publish**, and then click **Dynamic Report**. The **Publish Report** dialog box appears.

- If you are using Excel 2007, 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 Wonderware 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 Wonderware Historian on which to store the report publishing information.
- 4 In the **Report site** list, select the URL of the Wonderware Information Server to which you want to publish the report.

The report site may or may not be physically located on the Wonderware 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 Wonderware 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 Wonderware Information Server periodically scans the publishing folders for changes. Therefore, a short delay may occur prior to the published report being displayed on the Wonderware Information Server.

- 10 To view the Wonderware Information Server, click **Browse**. Otherwise, click **Done**.

Wonderware Historian Client Workbook Function Reference

This section describes the Wonderware Historian Client Workbook functions and their arguments. For a description of the arguments, see *Function Arguments* on page 340.

Required arguments are shown in a bold font.

Function Name	Syntax	Comments
wwAggregate	=wwAggregate(DataSource, TagRange, RowOrRes, Time1, Time2, AggCalc, ValueCriteria, OptionRange)	<p>Returns an array.</p> <p>This function retrieves values from the AnalogHistory table for specified tags, and then performs aggregate calculations.</p> <p>Single value is returned for each of the selected tags. Quality and value criteria can be specified.</p> <p>Use a row count of 0 to ensure that all stored values are included in the calculations.</p>

Function Name	Syntax	Comments
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.

Function Name	Syntax	Comments
wwAnalogLive1	=wwAnalogLive1(DataSource, TagRange, DisplayTagName, DisplayDateTime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with Wonderware Historian 10.0 or later.
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 Wonderware 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.

Function Name	Syntax	Comments
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.
wwDiscreteLive1	=wwDiscreteLive1(DataSource, TagRange, DisplayTagName, DisplayDateTime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with Wonderware 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 Wonderware Historian 10.0 or later.

Function Name	Syntax	Comments
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.

Function Name	Syntax	Comments
wwHistory	=wwHistory(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TimeDeadband, ValueDeadband, ValueCriteria, EdgeDetection, HistoryVersion, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality, DisplayAsWide, OrderBy, OptionRange)	Returns an array. Use with IndustrialSQL Server 8.0.
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 9.0 or later.

Function Name	Syntax	Comments
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 Wonderware Historian 10.0 or later.
wwLive	=wwLive(DataSource, TagRange, DisplayTagName, DisplayDatetime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality, OptionRange)	Returns an array. Use with IndustrialSQL Server 8.0 or later.
wwLive1	=wwLive1(DataSource, TagRange, DisplayTagName, DisplayDateTime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplayOPCQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array Use with Wonderware Historian 10.0 or later.
wwQuery	=wwQuery(DataSource, SQLQuery, OptionRange)	Returns an array. Returns the results of a SQL statement.

Function Name	Syntax	Comments
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 Wonderware Historian 10.0 or later.
wwReplicatedTagDetails	=wwReplicatedTagDetails(DataSource, TagRange, Description, Storage, SourceTag, SourceServer, OptionRange)	Returns an array. Use with Wonderware 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.

Function Name	Syntax	Comments
wwStringLive1	=wwStringLive1(DataSource, TagRange, DisplayTagName, DisplayDateTime, DisplayMilliseconds, DisplayQuality, ReplacePoorQuality, DisplaySourceTag, DisplaySourceServer, OptionRange)	Returns an array. Use with Wonderware Historian 10.0 or later.
wwStringTagDetails	=wwStringTagDetails(DataSource, TagRange, Description, MaxLength, OptionRange)	Returns an array.
wwStringTagDetails1	=wwStringTagDetails1(DataSource, TagRange, Description, MaxLength, SourceTag, SourceServer, OptionRange)	Returns an array. Use with Wonderware 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.

Function Name	Syntax	Comments
wwWideHistory	=wwWideHistory(DataSource, TagRange, RowOrRes, Time1, Time2, Interpolation, RetrievalMode, TagCriteria, EdgeDetection, HistoryVersion, DetectDatetime, DisplayMilliseconds, OrderBy, OptionRange)	Returns an array. Use with Wonderware 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 Wonderware Historian version 9.0 or later.
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 IndustrialSQL Server 9.5 or later.

Function Arguments

The following arguments are used for the Wonderware 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
- 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 = 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.

Value	Description
ToDiscrete	<p>Accepts two parameters: Cutoff value and operator.</p> <p>The cutoff value signifies the boundary between values that are interpreted as ON and OFF. For double value, the default is 0.0</p> <p>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 ">".</p>
SnapTo	<p>Accepts two parameters: Tolerance and base value.</p> <p>The tolerance value is of type double. The default value is 0.01.</p> <p>Base values are comma separated double values. The default value is 0.0.</p> <p>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.</p>

For more information, see Analog Value Filtering (wwFilter) on page 788.

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.

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

When you use this parameter with the `wwWideHistory2` function, the bits are as follows (bit 0 is the rightmost bit):

Bit	Display option	Integer equivalent
13	<code>wwStateCalc</code>	8192
12	<code>wwInterpolationType</code>	4096
11	Value timestamp	2048
10	Show milliseconds in timestamp	1024
9	<code>wwRetrievalMode</code>	512
8	<code>wwCycleCount</code>	256
7	<code>wwResolution</code>	128
6	<code>wwTimeDeadband</code>	64
5	<code>wwValueDeadband</code>	32
4	<code>wwTimestampRule</code>	16
3	<code>wwQualityRule</code>	8
2	<code>wwVersion</code>	4
1	<code>wwTimeZone</code>	2
0	<code>wwEdgeDetection</code>	1

For additional information on each display option, see Display Options Tab on page 255.

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:

Bit	Display option	Integer equivalent
16	Value timestamp	65536
15	Show milliseconds in timestamp	32768
3	<code>wwVersion</code>	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:

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.

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) on page 769.

When used with the wwHistory2 or wwWideHistory2 functions, valid values are:

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 Wonderware Historian level.

For more information on each option, see Interpolation Type (wwInterpolationType) on page 771.

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 Wonderware 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.

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 on page 498.

RawRange

TRUE = Include the raw range for the tag in the results.

FALSE = Do not include the raw range in the results.

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 on page 499.

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.

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.

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 on page 500.

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.

(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 on page 501.

UseStringHistory

TRUE = Include the string history table for query.

FALSE = Do not include the string history table for query.

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 Wonderware 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.

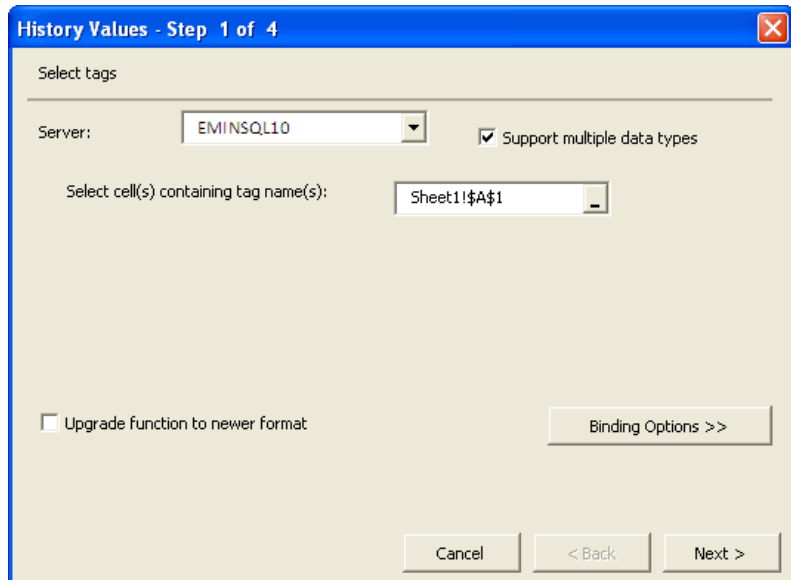
Migrating History Data Retrieval Functions

Workbooks created with earlier versions of the Wonderware 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`.
- 3 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Edit Function**. The **History Values - Step 1 of 4** dialog box appears.

- If you are using Excel 2007, on the **Historian** tab, in the **Controls** group, click **Edit Function**. The **History Values - Step 1 of 4** dialog box appears.



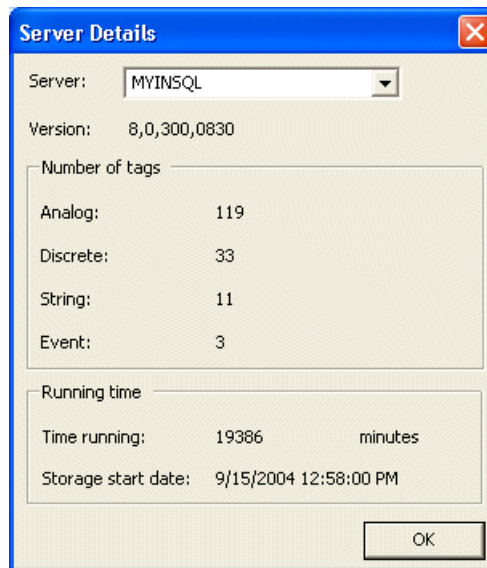
- 4 Select the **Upgrade function to newer format** check box. Click **Next**.
- 5 Go through the rest of the wizard as described in **Retrieving History Values** on page 251. After you finish the wizard, the function call is updated to use the new function.

Viewing the Wonderware Historian Details

You can view details for all of the Wonderware Historians that have been configured for use within the Wonderware Historian Client Workbook.

To view server details

- 1 Do one of the following:
 - If you are using Excel 2003 or XP, on the **Historian** menu, click **Wonderware Historian Details**. The **Server Details** dialog box appears.
 - If you are using Excel 2007, 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**.

Chapter 6

Wonderware Historian Client Report

The Wonderware Historian Client Report is an add-in to Microsoft Word that allows you to query one or more Wonderware Historian or SQL Server databases and return results to a Word document.

About Add-ins and Templates

The Wonderware 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 Wonderware 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 a Wonderware Historian or a normal SQL Server database.

Note You can manually load/unload the Wonderware Historian Client Report add-in. For more information, see [Manually Loading/Unloading the Add-In](#) on page 363.

The Wonderware 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 Wonderware Historian Client Report.

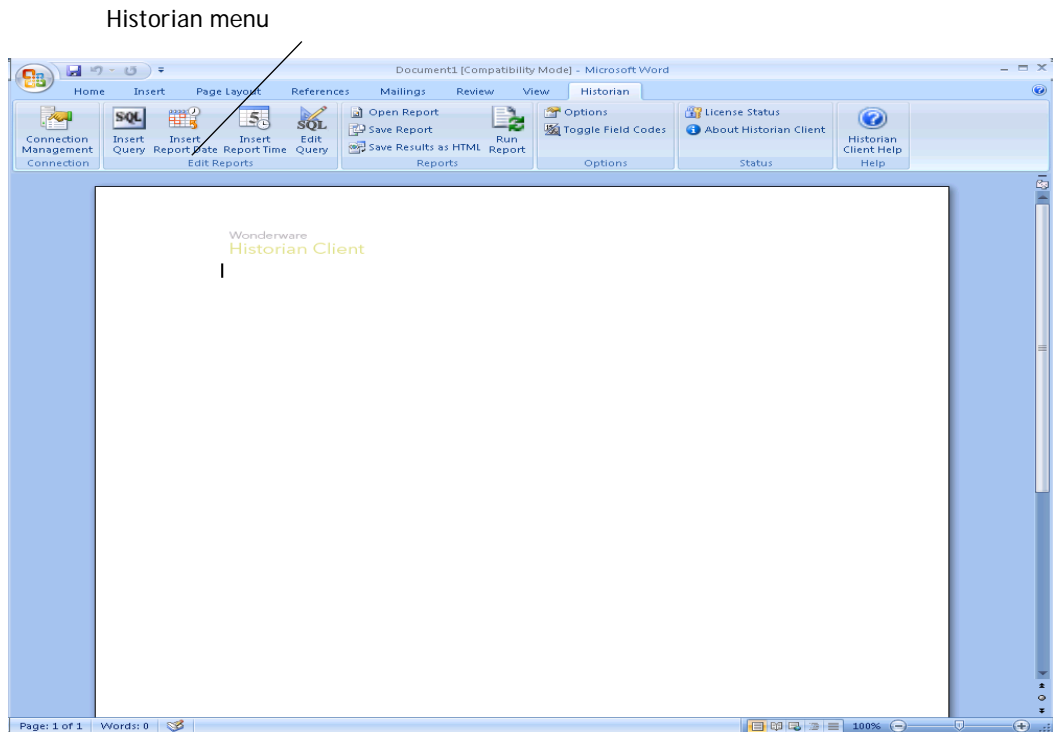
To get started with the Wonderware Historian Client Report

- 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 **Wonderware** program group, point to the **Wonderware 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**.
 - If you are using Office 2003 or XP, open Word. On the **File** menu, click **New**.
 - If you are using Office 2007, 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 C:\Program Files\Common Files\ArchestrA, for Office 2003 or XP, in the C:\Program Files\Microsoft Office\OFFICE11\STARTUP, and for Office 2007 in the C:\Program Files\Microsoft Office\OFFICE12\STARTUP folders.

A new blank document appears in Microsoft Word. The **Historian** menu appears.

The appearance of **Historian** menu in the Wonderware Historian Client Report differs based on the version of Microsoft Office.

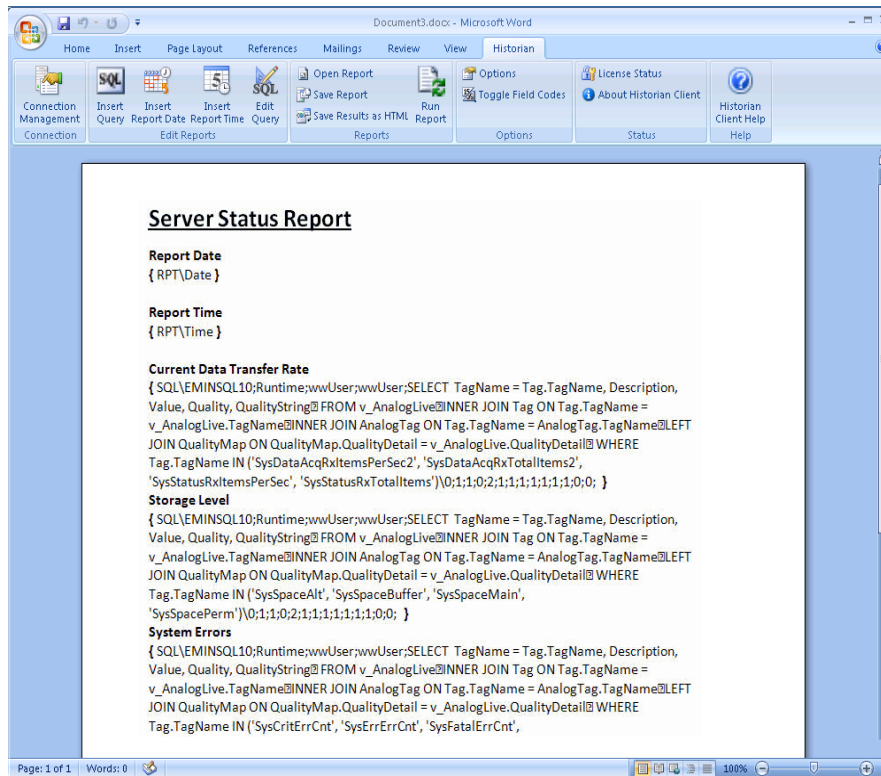
- If you are using Office 2003 or XP, the **Historian** menu appears as a menu item with drop-down options and contains a few toolbar options.
- If you are using Office 2007, the **Historian** menu is a part of the Ribbon Bar.



- 2 Configure the connection to one or more servers. For more information, see [Managing Server Connections](#) on page 366.
- 3 Create headings, explanatory paragraphs, sections, and so on, similar to a normal Word document.
- 4 Use the commands on the Wonderware 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](#) on page 371.

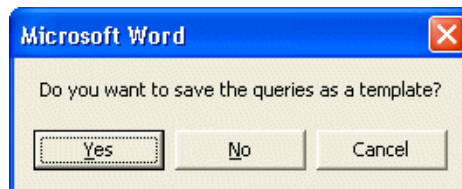
- Optionally add date and time fields to your report document. For more information, see [Inserting Date and Time Field Codes](#) on page 381.

The following example shows a configured report document that shows status information for a Wonderware Historian, as well as the date and time that the report document was run.



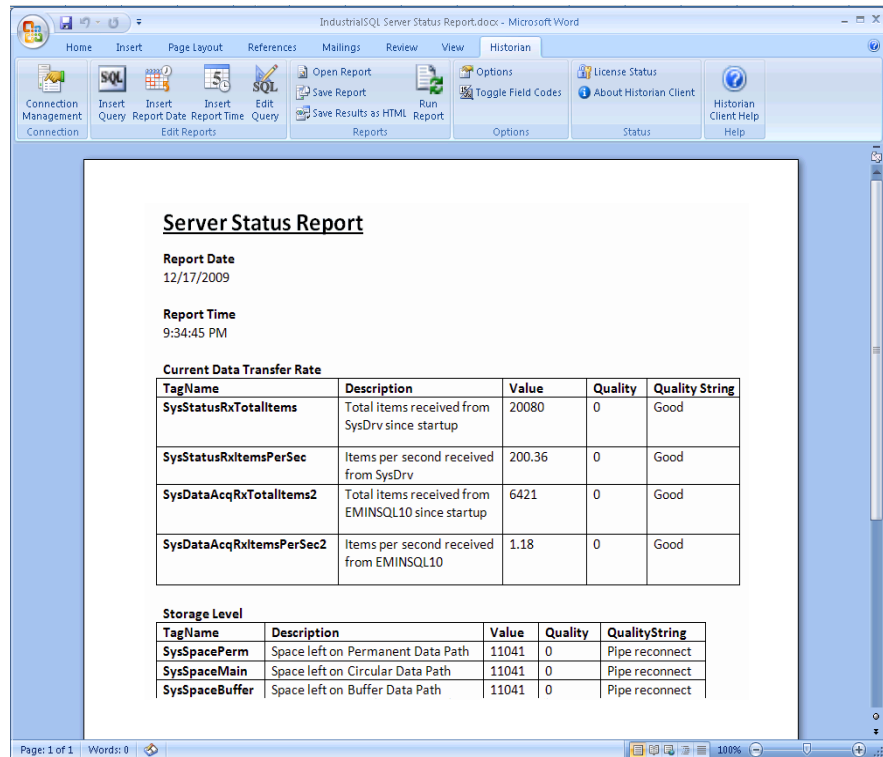
- Run the report document. For more information, see [Running a Report Document](#) on page 369.

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 Wonderware Historian Client Word add-in fills in the report document with the data and the resulting report document appears. For example:



- 7 Save the run report document. For more information, see Saving Report Documents on page 370.

Manually Loading/Unloading the Add-In

When you install the Wonderware Historian Client software, the Word add-in is automatically loaded into Microsoft Word. For Office 2003 or XP, the **Historian** menu appears in the menu bar and for Office 2007, the **Historian** tab appears in the Ribbon bar.

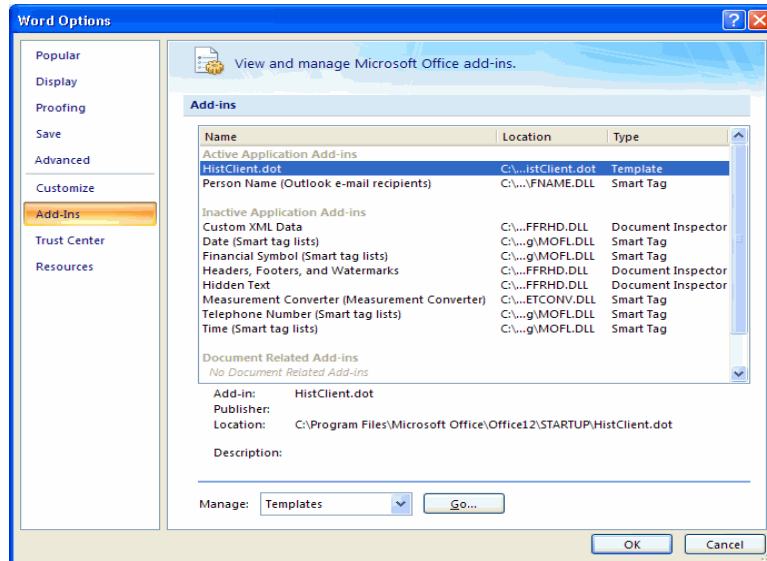
However, if you need to manually load or unload the add-in, use the following procedure:

To manually load the add-in

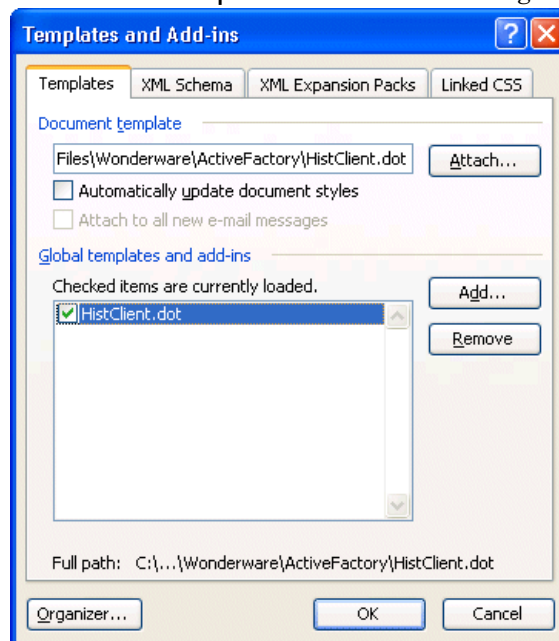
1 Do one of the following:

- If you are using Word 2003 or XP, on the Tools menu, click **Templates and Add Ins**. The **Templates and Add-Ins** dialog box appears.
- If you are using Word 2007, click the **Microsoft Office Button**, and then click **Word Options**. The **Word Options** dialog box appears.

Click **Add-Ins**.



In the **Manage** list, select **Word Add-ins**, and then click **Go**. The **Templates and Add-Ins** dialog box appears.



2 Click the **Templates** tab.

- 3 If the Wonderware Historian Client Word add-in is not listed in the **Checked items are currently loaded** window, do the following:
 - If you are using Office 2003 or XP, click **Add** and select the **HistClient.dot** file. By default, the **HistClient.dot** file is installed in the **C:\Program Files\Common Files\ArchestrA** and **C:\Program Files\Microsoft Office\OFFICE11\STARTUP** folders.
 - Select the **HistClient.dot** check box.
 - If you are using Office 2007, click **Add** and select the the **HistClient.dot** and **HistClient.dotm** files. By default the **HistClient.dot** and **HistClient.dotm** files are installed in the **C:\Program Files\Common Files\ArchestrA** and **C:\Program Files\Microsoft Office\OFFICE12\STARTUP** folders.
 - Select the **HistClient.dot** and **HistClient.dotm** check boxes.
- 4 Click **OK**.

To manually unload the add-in

- 1 Do one of the following:
 - If you are using Word 2003 or XP, on the **Tools** menu, click **Templates and Add Ins**. The **Templates and Add-Ins** dialog box appears.
 - If you are using Word 2007, click the **Microsoft Office Button**, and then click **Word Options**.
Click **Add-Ins**.
In the **Manage** list, select **Word Add-ins**, and then click **Go**. The **Templates and Add-Ins** dialog box appears.
- 2 Clear the **HistClient.dot** and/or **HistClient.dotm** check box in the **Checked items are currently loaded** window.
- 3 Click **OK**.

Managing Server Connections

You must specify one or more Wonderware Historians and/or SQL Servers as data sources for the Wonderware Historian Client Report.

To manage server connections

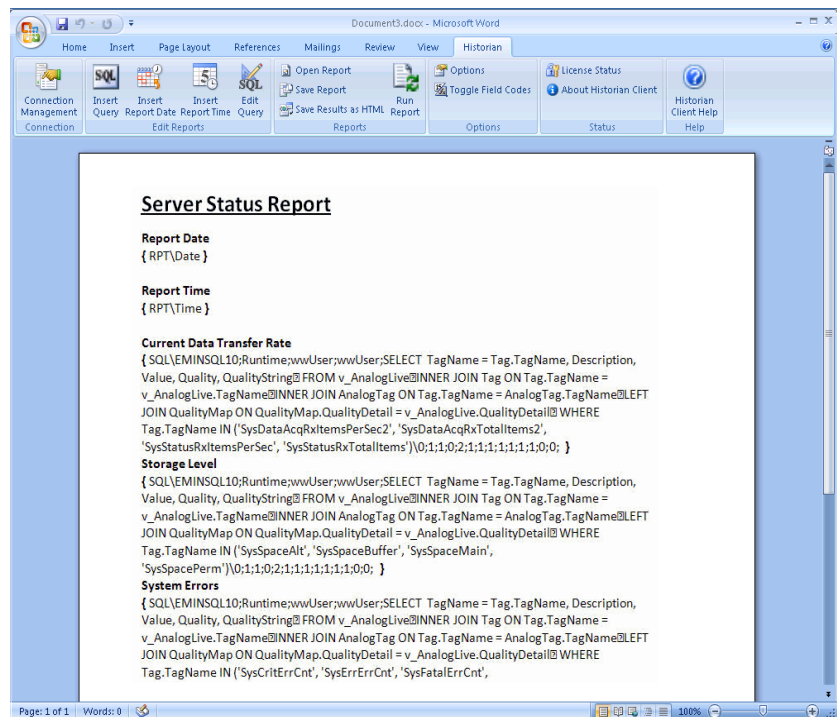
- 1 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Connection Management**. The **Server List Configuration** dialog box appears.
 - If you are using Word 2007, on the **Historian** tab, in the **Connection** group, click **Connection Management**. The **Server List Configuration** dialog box appears.
- 2 Configure the server(s) and then click **Close**. For more information, see **Server Connection Configuration** on page 27.

About Field Codes

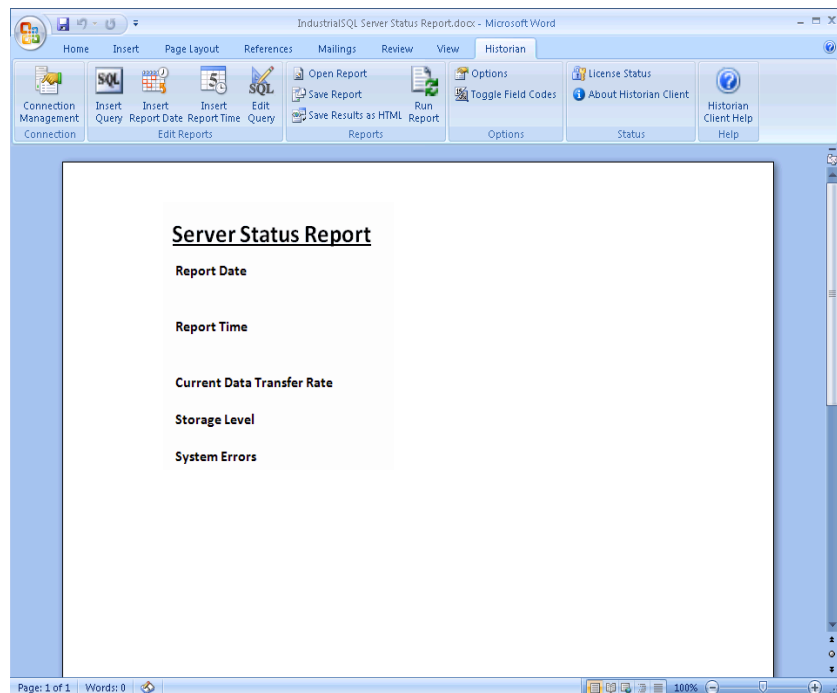
A field code is a special string of text in a Microsoft Word document that includes instructions for data processing. Field codes can process data from inside the same document or from external sources. For the Wonderware Historian Client Report, field codes are used to contain the instructions for retrieving 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 {}.



In the following graphic, the field codes are hidden. For more information on showing or hiding the field codes, see [Showing/Hiding Field Codes](#) on page 368.



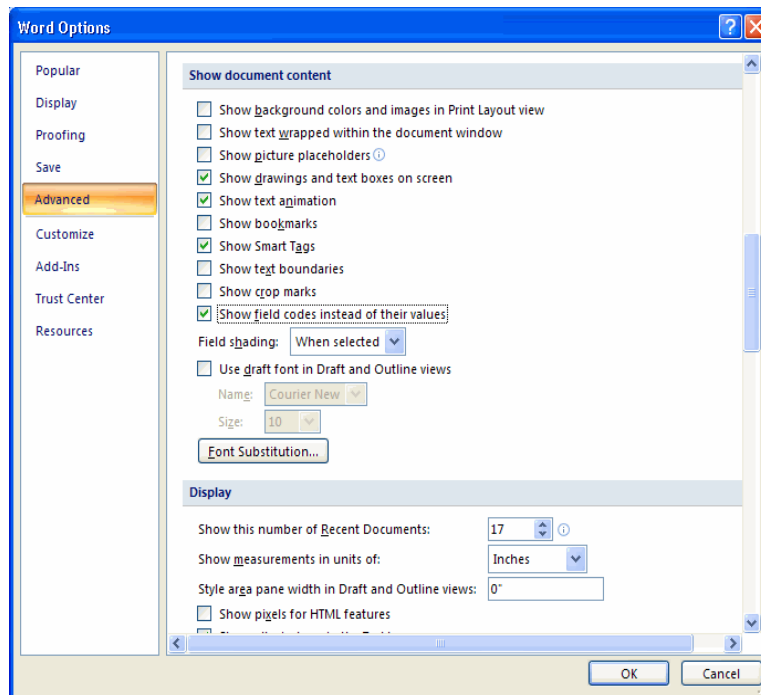
Showing/Hiding Field Codes

To show/hide the field codes using the Wonderware Historian Client menu

- ◆ Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Toggle Field Codes**.
 - If you are using Word 2007, on the **Historian** tab, in the **Options** group, click **Toggle Field Codes**.

To show/hide the field codes using Word options

- 1 Click the **Microsoft Office Button**, 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* on page 360.

To open an existing report document

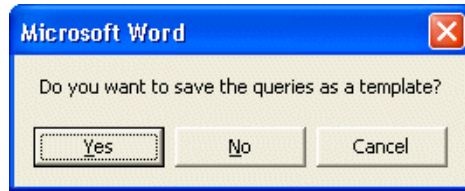
- 1 Do one of the following:
 - If you are using Word 2003 or XP, on the **File** menu, click **Open**. The **Open** dialog box appears.
 - If you are using Word 2007, click the **Microsoft Office Button**, and then click **Open**. The **Open** dialog box appears.
- 2 Browse to the report to open.

Running a Report Document

To run a report document

- 1 Open an existing report.
- 2 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Run Report**.
 - If you are using Word 2007, 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. For example:

 A screenshot of a Microsoft Word window titled "IndustrialSQL Server Status Report.docx - Microsoft Word". The ribbon shows the "Historian" tab with various options like "Open Report", "Save Report", "Run Report", "Options", and "Toggle Field Codes". The main content area displays a report titled "Server Status Report" with the following details:

Report Date
12/17/2009

Report Time
9:34:45 PM

Current Data Transfer Rate

TagName	Description	Value	Quality	Quality String
SysStatusRxTotalItems	Total items received from SysDrv since startup	20080	0	Good
SysStatusRxItemsPerSec	Items per second received from SysDrv	200.36	0	Good
SysDataAcqRxTotalItems2	Total items received from EMINSQL10 since startup	6421	0	Good
SysDataAcqRxItemsPerSec2	Items per second received from EMINSQL10	1.18	0	Good

Storage Level

TagName	Description	Value	Quality	QualityString
SysSpacePerm	Space left on Permanent Data Path	11041	0	Pipe reconnect
SysSpaceMain	Space left on Circular Data Path	11041	0	Pipe reconnect
SysSpaceBuffer	Space left on Buffer Data Path	11041	0	Pipe reconnect

 The status bar at the bottom indicates "Page: 1 of 1" and "Words: 0".

Saving Report Documents

You can save a report document as:

- A static .doc 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.

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* on page 369.
- 2 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Save Report**. The **Save As** dialog box appears.
 - If you are using Word 2007, 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.

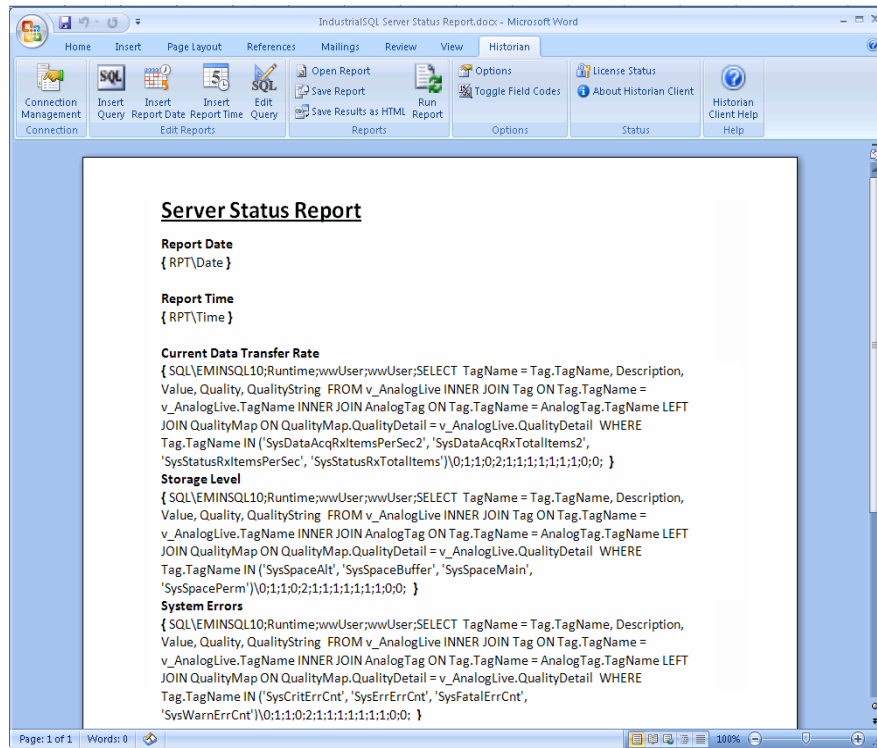
Saving a Configured Report Document as a Report Template

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 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Save Report**. The **Save As** dialog box appears.
 - If you are using Word 2007, 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 .dot file appears in Microsoft Word.



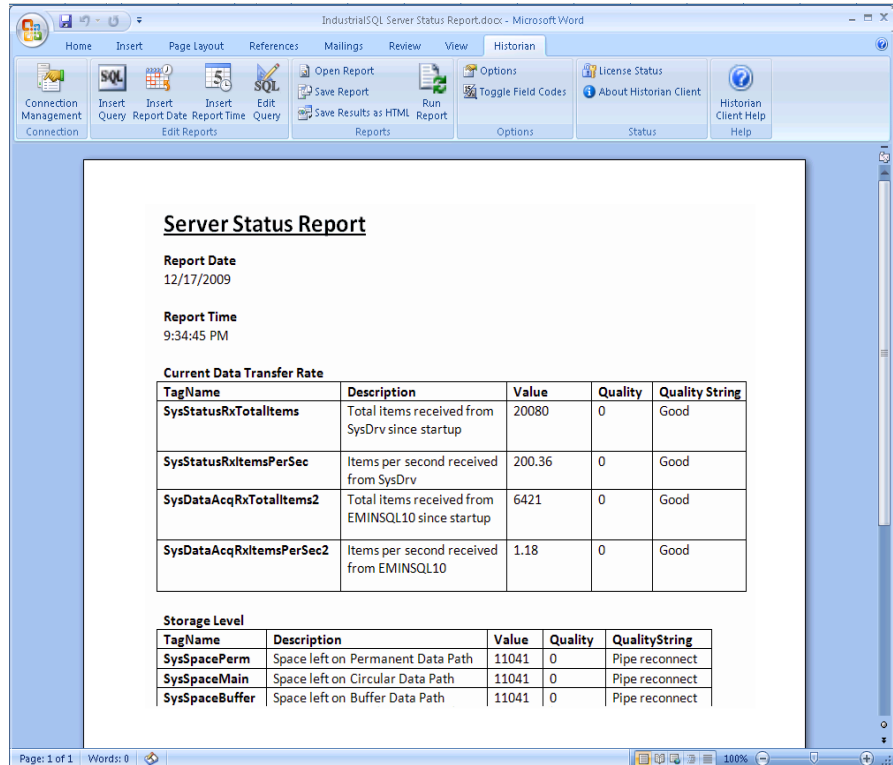
5 You can then copy the report template file into the Microsoft Word template directory and use it to create report documents or as a basis to create new report template files.

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 Wonderware 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 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Save Results as HTML**. The **Save As** dialog box appears.
 - If you are using Word 2007, 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.

Server Status Report

Report Date
12/21/2009

Report Time
1:22:07 AM

Current Data Transfer Rate

TagName	Description	Value	Quality	Quality String
SysStatusRxTotalItems	Total items received from SysDrv since startup	20080	0	Good
SysStatusRxItemsPerSec	Items per second received from SysDrv	200.36	0	Good
SysDataAcqRxTotalItems2	Total items received from EMINSQL10 since startup	6421	0	Good
SysDataAcqRxItemsPerSec2	Items per second received from EMINSQL10	1.18	0	Good

Storage Level

TagName	Description	Value	Quality	QualityString
SysSpacePerm	Space left on Permanent Data Path	11041	0	Pipe reconnect
SysSpaceMain	Space left on Circular Data Path	11041	0	Pipe reconnect
SysSpaceBuffer	Space left on Buffer Data Path	11041	0	Pipe reconnect
SysSpaceAlt	Space left on Alternative Data Path	0	0	Pipe reconnect

6 On the Quick Access Toolbar, click Web Page Preview to see the report document in a web browser.

Server Status Report

Report Date
12/21/2009

Report Time
1:22:07 AM

Current Data Transfer Rate

TagName	Description	Value	Quality	Quality String
SysStatusRxTotalItems	Total items received from SysDrv since startup	20080	0	Good
SysStatusRxItemsPerSec	Items per second received from SysDrv	200.36	0	Good
SysDataAcqRxTotalItems2	Total items received from EMINSQL10 since startup	6421	0	Good
SysDataAcqRxItemsPerSec2	Items per second received from EMINSQL10	1.18	0	Good

Storage Level

TagName	Description	Value	Quality	QualityString
SysSpacePerm	Space left on Permanent Data Path	11041	0	Pipe reconnect
SysSpaceMain	Space left on Circular Data Path	11041	0	Pipe reconnect
SysSpaceBuffer	Space left on Buffer Data Path	11041	0	Pipe reconnect
SysSpaceAlt	Space left on Alternative Data Path	0	0	Pipe reconnect

Inserting a SQL Query

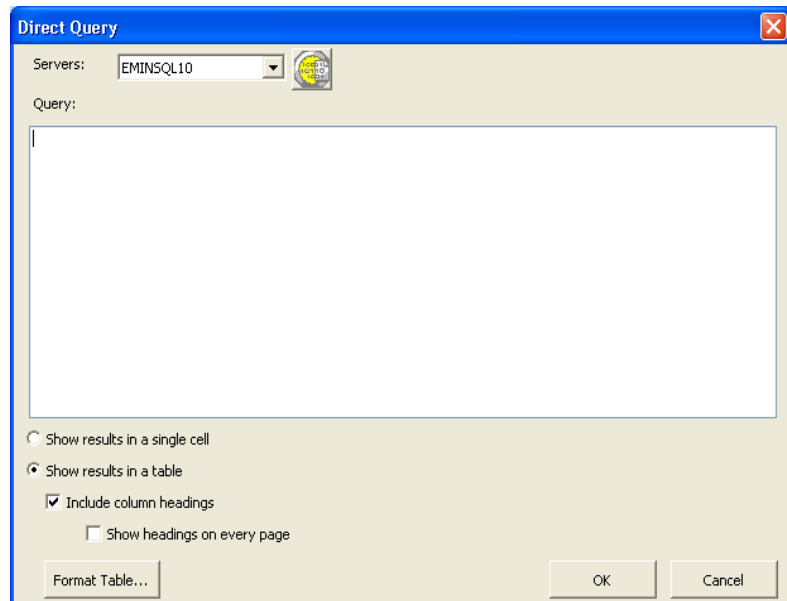
You can either type in a SQL query or launch the Query application to allow you to build the query using a point-and-click interface.

If you want to use the Query application, you must configure at least one server connection. For more information, see *Getting Started with Query* on page 165.

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 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Insert Query**. The **Direct Query** dialog box appears.
 - If you are using Word 2007, on the **Historian** tab, in the **Edit Reports** group, click **Insert Query**. The **Direct Query** dialog box appears.

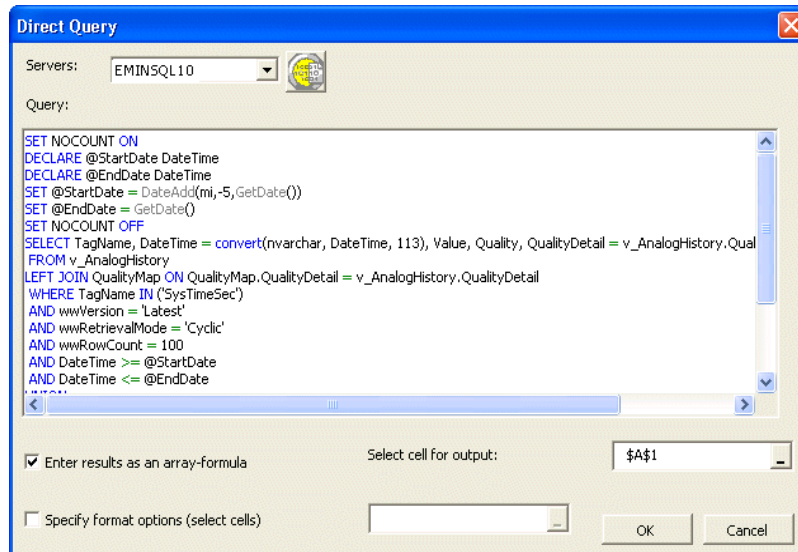


- 3 In the **Servers** list, click the name of the server to use.

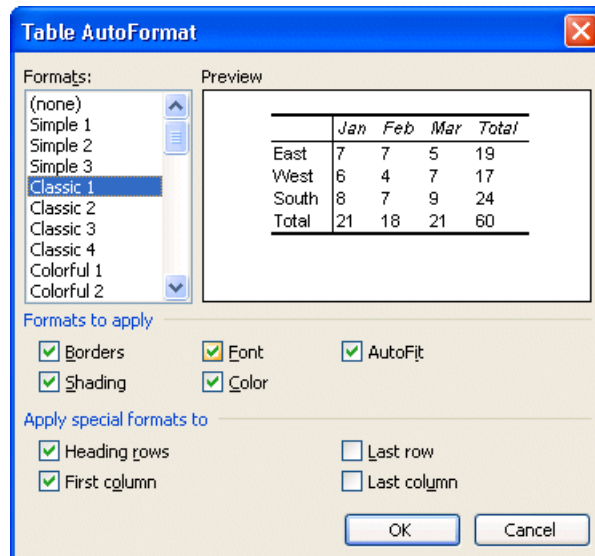
- 4 In the **Query** box, type the SQL query that are executed against the database.



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 Chapter 4, Wonderware Historian Client Query.

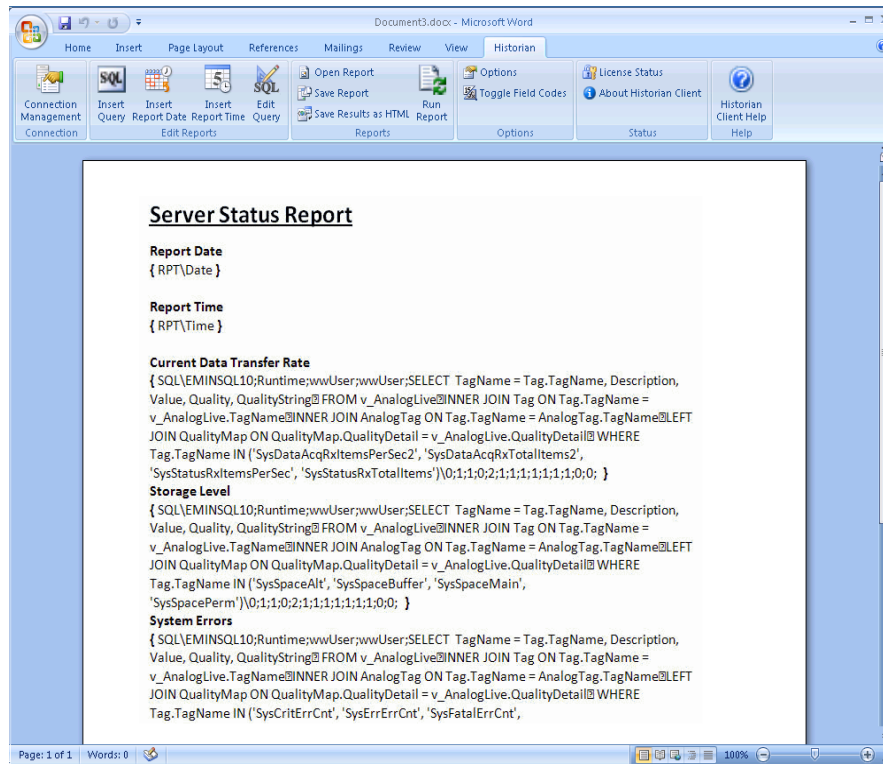


- 5 Configure how to display the results in the report document after it has been run.
 - **Show results in a cell:** Displays only the value in the first column of the first record in the returned record set. For example, if you queried the tagname and the description for a tag, only the value for the 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

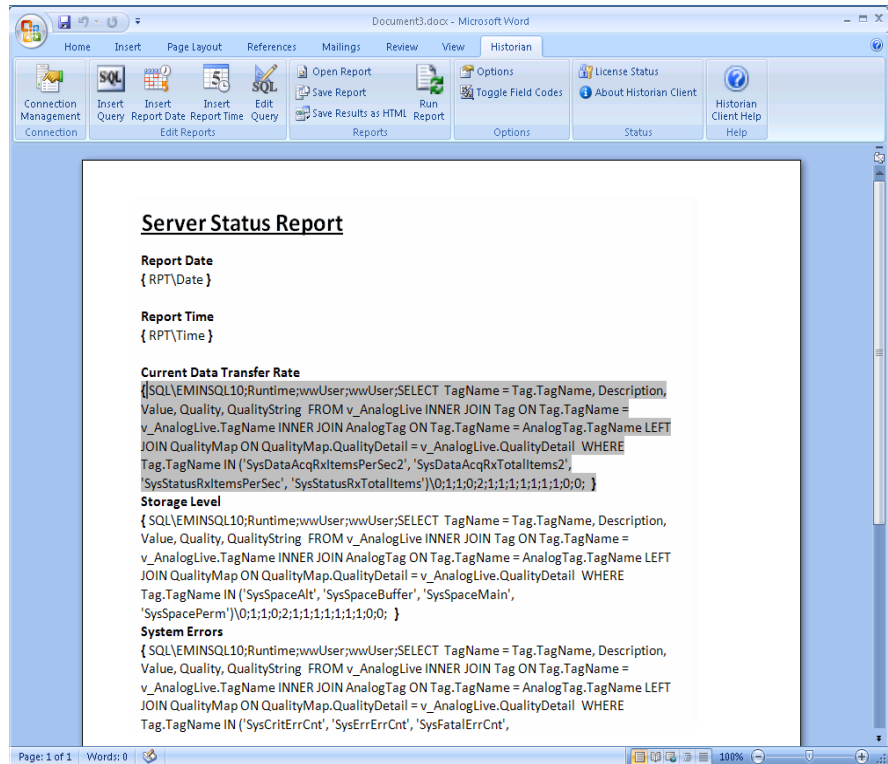
You can edit an existing query in a report template or report document that has not yet been run. (After a report document is run, all queries are converted to actual data.)

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.

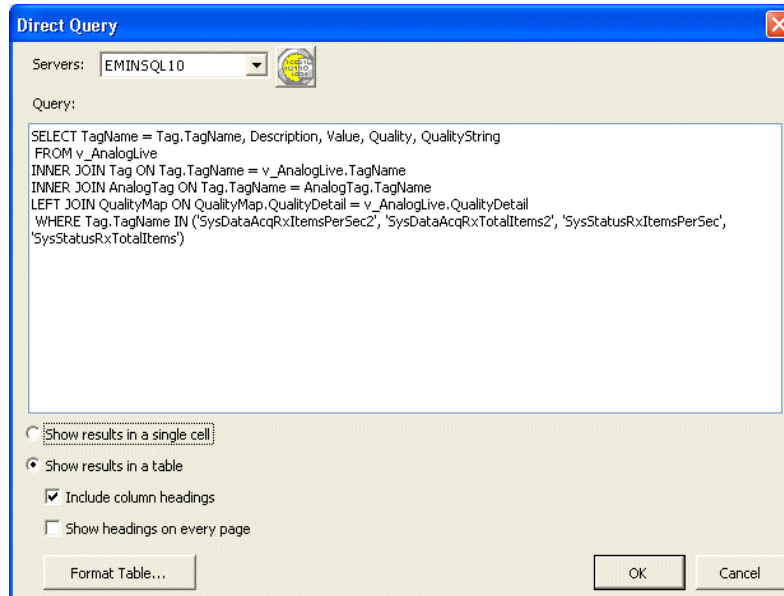
Both methods require that the field codes for the report document or template are visible. For more information, see **About Field Codes** on page 366.

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 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Edit Query**. The **Direct Query** dialog box appears.
 - If you are using Word 2007, 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 Chapter 4, *Wonderware Historian Client Query*.
- 5 Change any of the options for the results. For more information, see *Saving a Configured Report Document as a Report Template* on page 371.
- 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* on page 381.
- Variables for relative date/time parameters in the **WHERE** clause for queries. For more information, see *About Date and Time Wildcards* on page 382.

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

- ◆ Do any one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Insert Report Date**.
 - If you are using Word 2007, on the **Historian** tab, in the **Edit Reports** group, click **Insert Report Date**.

To insert a time field

- ◆ Do any one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Insert Report Time**.
 - If you are using Word 2007, 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](#) on page 368.

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.

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* on page 386.

These wildcards are handled by the Wonderware Historian Client Report; queries that include them do not work in other query tools, such as the Wonderware 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* on page 386.

The time specification for the query is specified within parenthesis.

The valid duration units for the time offset are:

- s = seconds
- mi = minutes
- 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* on page 386.

The valid duration units for the time offset are:

- s = seconds
- mi = minutes
- h = hours
- d = days
- w = weeks
- mm = months

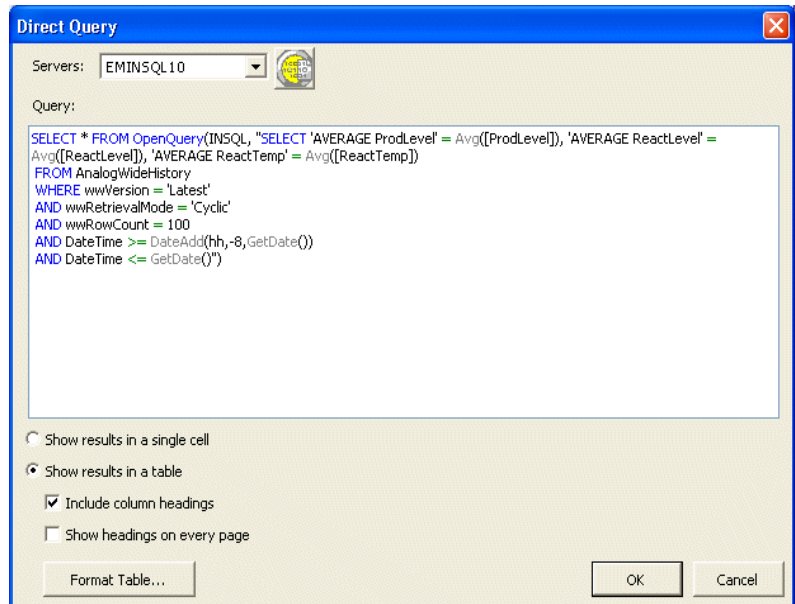
Inserting Date and Time Wildcards

For more information on wildcards, see About Date and Time Wildcards on page 382.

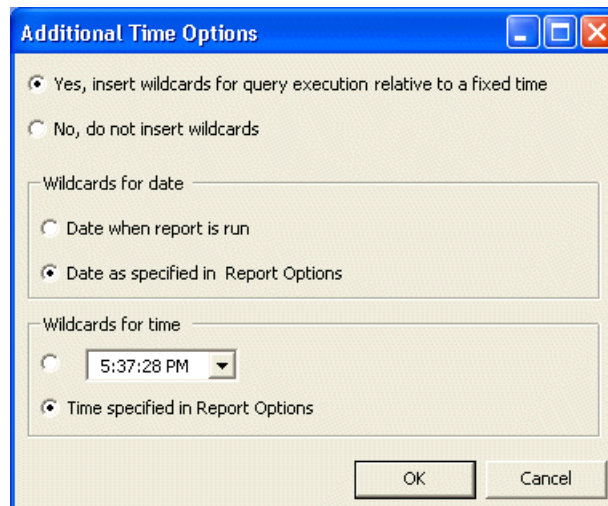
To use date and time wildcards

- 1 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Options**.
 - If you are using Word 2007, 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 on page 386.
- 3 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Insert Query**.
 - If you are using Word 2007, 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 Chapter 4, Wonderware Historian Client Query.



- 5 Click **OK**. The **Additional Time Options** dialog box appears.



- 6 Click **Yes, insert wildcards**.
- 7 In the **Wildcards for date** area, configure which date is substituted for the date in the query.
- **Date when report is run:** The day that the report document is run is used as the base date for the query. This option is used for the #.
 - **Date as specified in Report Options:** The date specified in the report options is used as the base date for the query. For more information, see *Configuring Report Options* on page 386.

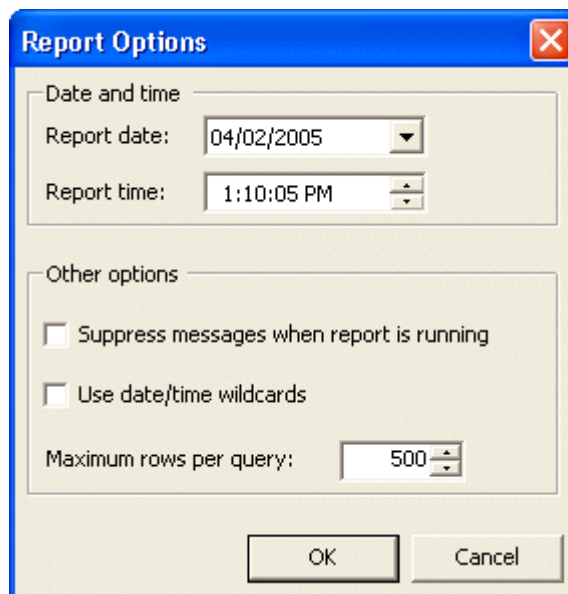
- 8 In the **Wildcards for time** area, configure which time is substituted for the base time in the query.
 - **(list box):** A base time for the query.
 - **Time specified in Report Options:** The time specified in the report options is used as the base time for the query. For more information, see *Configuring Report Options* on page 386.
- 9 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 Do one of the following:
 - If you are using Word 2003 or XP, on the **Historian** menu, click **Options**. The **Report Options** dialog box appears.
 - If you are using Word 2007, 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 on the wildcards, see **About Date and Time Wildcards** on page 382.
 - **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**.

Chapter 7

Introduction to Controls and Objects

The Wonderware 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 Wonderware 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 Wonderware Historian Client Controls and Objects

The Wonderware 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:

- 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 a Wonderware 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 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 Wonderware Historian Client controls: one for use within a ActiveX control containers and another for within .NET containers. ActiveX control containers include Visual Interdev, 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 Wonderware Historian Client software. Embedding the ActiveX versions of the controls in Internet Explorer is not recommended. If you are using Visual Interdev, manually edit the HTML to use the .NET version of the control.

The following HTML example shows how to embed the .NET control on an HTML form:

```
<html>
<body>
<object id="Trend1"
  classid="http:aaHistClientUI.dll#Archestra.HistClient
  .UI.aaTrendControl"
  height="300" width="300" VIEWASTEXT></object>
</body>
</html>
```

ActiveX controls that are embedded in HTML are loaded if you launch the .htm file within Windows Explorer (that is, if the URL starts with `file://`). However, a URL that starts with `file://` does not load .NET controls. You must use `http://`, which means you must create a web share for the folder in which the .htm file resides.

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 Wonderware 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 Wonderware Historian Client controls can be run.

In C# or .NET environments:

- Byte = 8 bits
- 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.

Chapter 8

aaHistClientTrend Control

The aaHistClientTrend control allows you to run the Wonderware Historian Client Trend program (or a functional subset) from within the Wonderware InTouch HMI software or a .NET container like Visual Basic .NET or Internet Explorer.

For more information on using the Wonderware Historian Client Trend, see Chapter 3, Wonderware Historian Client Trend.

Using aaHistClientTrend at Runtime

At runtime, aaHistClientTrend trends configured tag values from the Wonderware Historian in the container application. You can use aaHistClientTrend just as you would use the Wonderware Historian Client Trend application.

For more information on using Wonderware Historian Client Trend, see Wonderware Historian Client Trend on page 51.

Using aaHistClientTrend in an Application

aaHistClientTrend is capable of running with all of the functionality of the Wonderware 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 a Wonderware 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");

#aaHistClientTrend1.TagPickerVisible = 0;

#aaHistClientTrend1.TimeBarVisible = 0;

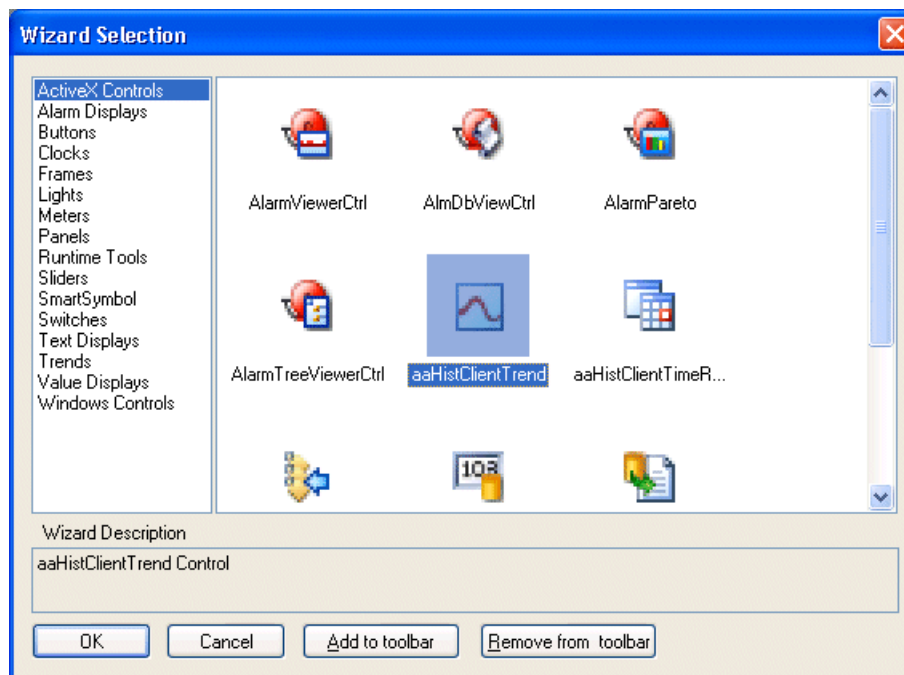
#aaHistClientTrend1.RealTimeMode = 1;
```

Adding aaHistClientTrend to an InTouch Window

To add the aaHistClientTrend control



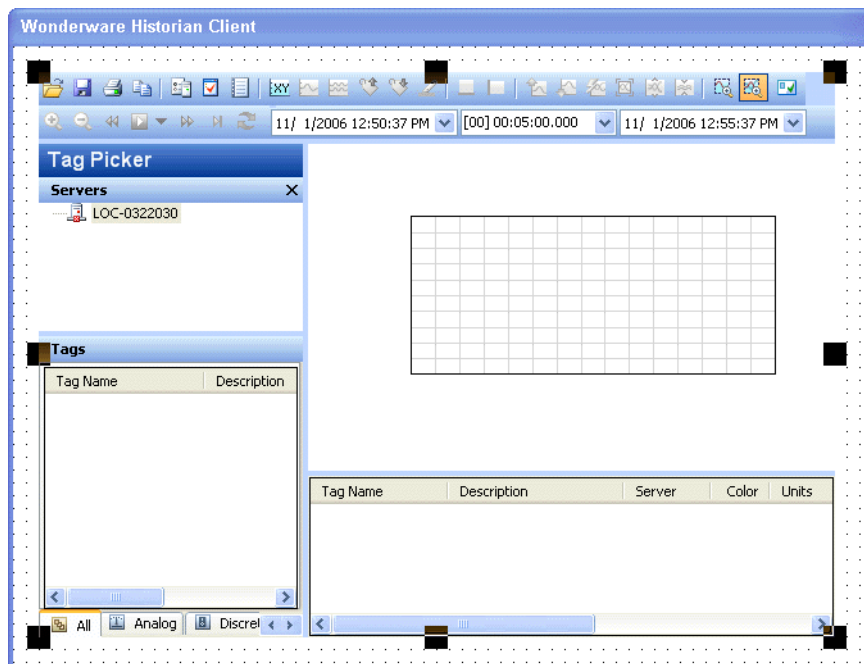
- 1 In WindowMaker, click the Wizards button. The Wizard Selection dialog box appears.



- 2 Select the aaHistClientTrend control.

3 Click OK.

The control appears in the window.



aaHistClientTrend Properties

The aaHistClientTrend properties include:

- AddMultipleTags
- AllowContextMenu
- AllowGridEditing
- AlwaysUseFullForXYScatterPlots
- AnalogPlottingAlgorithm
- ApplyRubberBandToAllTags
- AutoRefreshMode
- BackColor
- BackGradient
- BackGradientEndColor
- BackImage
- BorderColor
- BorderStyle
- BorderWidth

- ChartType
- CurrentServerName
- CurrentTagColor
- CurrentTagCycleCount
- 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
- 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
- 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
- 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.

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.

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 a Wonderware 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.

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.

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;
```

Remarks

For information on setting the color value, see Color on page 678.

The default value is 16777215.

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;
```

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 on page 502.

The default value is 0.

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 on page 678.

The default value is 16777215.

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 on page 678.

The default value is 0.

BorderStyle

The BorderStyle property is a read-write property that gets or sets the style of the border line around the trend chart.

Syntax

```
aaHistClientTrend.BorderStyle = aaDashStyle;  
Result = aaHistClientTrend.BorderStyle;
```

Remarks

For more information on the values for the border style, see the aaDashStyle Enumeration on page 497.

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 on page 497.

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.

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 on page 678.

The default value is 0.

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 more information, see Working with Retrieval Styles on page 809.

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` on page 499.

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.

Syntax

```
aaHistClientTrend.CurrentTagHistoryVersion =
    aaRetrievalVersion;
Result = aaHistClientTrend.CurrentTagHistoryVersion;
```

Remarks

For information on possible values, see `aaRetrievalVersion Enumeration` on page 499. 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` on page 498. 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).

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;
```

Remarks

The offset, expressed in milliseconds, can be positive or negative. For more information, see [Using Time Offsets to Compare Data](#) on page 119. 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:

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;
```

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 on page 498. 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.

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* on page 809.

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](#) on page 809. For information on possible values, see [aaRetrievalMode Enumeration](#) on page 499.

The default value is 0 (cyclic). Make sure that the specified retrieval mode is supported by the Wonderware 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;
```

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](#) on page 810. 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 on page 351. 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.

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 on page 678.

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 on page 500. 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 on page 809.

The default value is 5 (use application setting).

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) on page 762.

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;
```

Remarks

For information on possible values, see aaTimeStampRules Enumeration on page 501. 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 on page 502.

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)` on page 755.

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`.

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](#) on page 98.

This property has no default value.

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 on page 98.

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) on page 766.

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.

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

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

The PenWidth property is a read-write property that gets or sets the thickness of the trend curve for the currently selected tag.

Syntax

```
aaHistClientTrend.CurrentTrendItem.PenWidth = integer;  
Result = aaHistClientTrend.CurrentTrendItem.PenWidth;
```

Remarks

Valid values are 0 through 10. The default value is 0.

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

The ValueFormat property is a read-write property that gets or sets the value format of the currently selected 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

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

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;
```

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

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 on page 502.

The default value is 3.

Name

The Name property is a read-only property that gets the name of the currently selected tag.

Syntax

```
Result = aaHistClientTrend.CurrentTrendItem.Name;
```

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.

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.

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.

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](#) on page 497.

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](#) on page 497.

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.
- 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;
```

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 Wonderware Historians with a version earlier than 9.0. For more information, see Retrieval Styles, Application Settings, and Tag Settings on page 819.

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.

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 on page 809.

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 gets the timestamp at the right edge of the trend.

Syntax

```
aaHistClientTrend.EndDate = DateTime;  
Result = aaHistClientTrend.EndDate;
```

Remarks

For information on setting the date/time value, see `DateTime` on page 678.

This property has no default value.

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` on page 678.

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.

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.

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 on page 675.

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;
```

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.

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.

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 Wonderware Historians with a version earlier than 9.0.

Valid values are 0 to 100,000. The default value is 10,000.

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 Wonderware 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.

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 a Wonderware 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 a Wonderware 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;
```

Remarks

To calculate moving averages, use a retrieval style instead. For more information, see [Working with Retrieval Styles](#) on page 809.

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* on page 809.

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* on page 427.

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;
```

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;
```

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;
```

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](#) on page 82.

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](#) on page 678.

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;
```

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 on page 502.

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 on page 678.

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 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.

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.

The default value is 20.

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;
```

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](#) on page 809.

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 on page 499. 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 on page 498. 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).

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 on page 498. 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;
```

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* on page 809.

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 Styles](#) on page 809. For information on possible values, see [aaRetrievalMode Enumeration](#) on page 499.

The default value is 0 (cyclic). Make sure that the specified retrieval mode is supported by the Wonderware 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](#) on page 810. To find out how many retrieval styles are 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 on page 351. 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.

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 on page 500. 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* on page 809.

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)* on page 762.

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;

Result =
    aaHistClientTrend.RetrievalOptionsTimeStampRule;
```

Remarks

For information on possible values, see aaTimeStampRules Enumeration on page 501.

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) on page 755.

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;
```

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)` on page 766.

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.

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 Wonderware 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;
```

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.

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 on page 587.

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.

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.

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 678.

This property has no default value.

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 on page 809.

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.

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 Chapter 10, 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.

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 on page 501.

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 on page 678.

The default value is 0.

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;
```

Return Value

The return value is an aaHistClientTimeRangePicker control. For more information on this control, see Chapter 11, 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;
```

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;
```

Remarks

For information on possible values, see `aaTraceGradientType` Enumeration on page 501.

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** on page 47, **Refreshing the Trend Chart** on page 66, and **Showing Live Data** on page 81.

For information on possible values, see `aaUpdateToCurrentTimeState` Enumeration on page 503.

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.

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](#) on page 83. For more information on the [aaValueAxisLabelEnumeration](#) enumeration, see [aaValueAxisLabelEnumeration Enumeration](#) on page 503.

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](#) on page 678.
The default value is 255.

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;
```

Remarks

The value is given as a date/time value. For information on the date/time value format, see [DateTime](#) on page 678.

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 on page 678.

The default value is 16711680.

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 on page 678.

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.

Syntax

```
aaHistClientTrend.YCursor1Color = integer;
```

```
Result = aaHistClientTrend.YCursor1Color;
```

Remarks

For information on setting the color value, see Color on page 678.

The default value is 32768.

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 on page 678. The default value is 32768.

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
- 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
- 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

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 SuppressErrors property. For more information, see SuppressErrors on page 454.

AddServer

The AddServer method adds a server to the list.

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.

Return Value

Returns True if the server can be added; otherwise returns False.

AddServerEx

The AddServerEx 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.

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 single application; the persisted password is not available to all applications.

Return Value

Returns True if the server can be added; otherwise returns False.

Remarks

All parameters are required. Errors, if any, are reported.

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();
```

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();
```

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.

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.

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.

Return Value

Returns an integer that specifies the color. For information on the color value, see `Color` on page 678.

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

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](#) on page 119.

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

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.

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 on page 98. 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.

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 on page 98. 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.

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);
```

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](#) on page 70 for regular trends, and [Defining a Target Region for a Scatter Plot](#) on page 148 for scatter plots.

ManualConnect

The ManualConnect method displays the Server List Connection dialog box.

Syntax

```
[Result=] aaHistClientTrend.ManualConnect();
```

MoveNextTag

The MoveNextTag method sets the current tag to the next tag in the tag list.

Syntax

```
[Result=] aaHistClientTrend.MoveNextTag();
```

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();
```

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();
```

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.

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.

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 <i>fileName</i> 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 <i>fileName</i> 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.

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);
```

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 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();
```

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.

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();
```

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.

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);
```


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 information on setting the date/time value, see [DateTime](#) on page 678.

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.

SetDuration

The SetDuration method sets the time period for the trend based on a duration that is relative to the current time.

Syntax

```
[Result=] aaHistClientTrend.SetDuration(DateTime  
    duration);
```

Parameters*duration*

The time duration from the current time.

Remarks

For information on setting the date/time value, see `DateTime` on page 678.

Calling this method sets the end time to the current time and the start time to the current time minus the specified duration.

Example

In the following example, the duration is set for the past five minutes, relative to the current time.

```
#aaHistClientTrend1.SetDuration("00:05:00");
```

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);
```

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

Number	Corresponding menu item
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
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 on page 678.

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).

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 on page 119.

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);
```

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.

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.

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 UpdateToCurrentTimeState 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 more information on valid values for the duration, see `aaTimeRangeEnumeration Enumeration` on page 676.

Remarks

For information on setting the date/time value, see `DateTime` on page 678.

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

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 on page 105.

UnsetCurrentTagXAxisTag

This method removes any associated X axis tag from the currently selected tag in a scatter plot.

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 methods used by the aaHistClientTrend:

- 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 on page 675.

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.CurrentTagChanged.tagName;
```

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();
```

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;
```

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	XYSscatterPlot	XY scatter plot.

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.

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 Wonderware Historian level.

For more information on each option, see Interpolation Type (wwInterpolationType) on page 771.

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 Wonderware 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) on page 778.

aaRetrievalMode Enumeration

Specifies the data retrieval mode.

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](#) on page 689.

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 \(wwVersion\)](#) on page 769.

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 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) on page 786.

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.
3	ServerDefault	Use the default timestamp rule specified at the Wonderware Historian level.

For more information on each option, see [Time stamp Rule \(wwTimestampRule\)](#) on page 774.

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.

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](#) on page 67.

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.

aaValueAxisLabelEnumeration Enumeration

Specifies the value display format of the trend value. For more information on the types of scales, see [Scaling Tags](#) on page 83.

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 .

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:

- ArcestrA.HistClient.UI.aaTrend
- ArcestrA.HistClient.UI.aaTrendItemEditor
- Dundas.Charting.WinControl.Chart

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 `win.ini` file in your Windows folder using a text editor.
- 2 Look for the `[HistClient]` section. If no such section exists, create one.
- 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.

Chapter 9

aaHistClientQuery Control

The aaHistClientQuery control allows you to run the Wonderware Historian Client Query program (or a functional subset) from within the Wonderware InTouch HMI software or a .NET container like Visual Basic .NET or Internet Explorer.

For more information on using the Wonderware Historian Client Query, see Wonderware Historian Client Query on page 165.

Using aaHistClientQuery at Runtime

At runtime, aaHistClientQuery can retrieve data from the Wonderware Historian database and return the results in a table format. You can use aaHistClientQuery as you do the Wonderware Historian Client Query application.

For more information on using the Wonderware Historian Client Query, see Wonderware Historian Client Query on page 165.

Using aaHistClientQuery in an Application

aaHistClientQuery is capable of running with all of the functionality of the Wonderware 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.

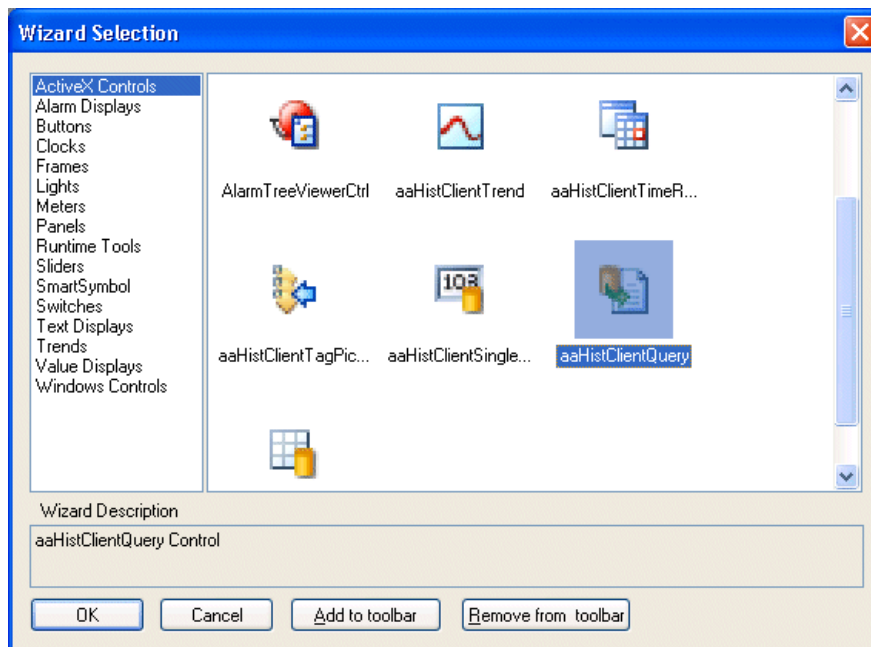
For example, maybe you want to limit the functionality of aaHistClientQuery to only allow the runtime operator to connect to a Wonderware Historian and run a particular query for a specific set of tags.

Adding aaHistClientQuery to an InTouch Window

To add the aaHistClientQuery control

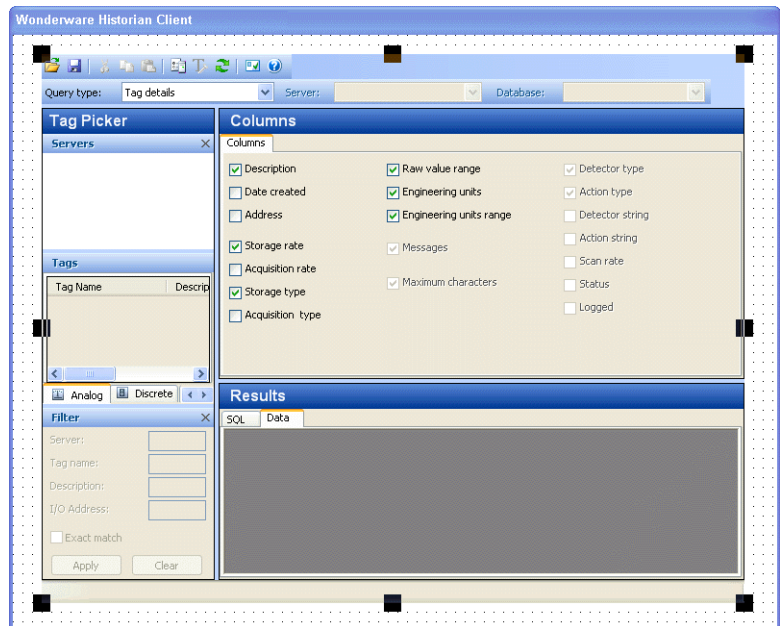


- 1 In WindowMaker, click the Wizards button. The Wizard Selection dialog box appears.



- 2 Select the aaHistClientQuery control.

3 Click OK. The control appears in the window.



aaHistClientQuery Properties

The properties for the aaHistClientQuery control are:

- ActiveServer
- AllowQueryTypeChange
- CurrentServer
- EnableAllQueriesTab
- FavoriteQueriesFolder
- FontBold
- FontCharset
- FontItalic
- FontName
- FontSize
- LockDown
- QueryFont
- QueryString
- Recordset
- Servers

- ToolbarConnectVisible
- ToolbarEditVisible
- ToolbarRequeryVisible
- ToolbarVisible
- UsePersistedServers

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.

Remarks

This property has no default value.

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

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 on page 579.

This property has no default value.

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

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.

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

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.
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

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;
```

Remarks

True = Use italics; False = Do not use italics.

The default value is False.

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

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.

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

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 on page 679.

The default font is Tahoma, 8 point (for English versions).

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

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 on page 679.

Remarks

This property is not accessible in the InTouch HMI software.

This property has no default.

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 587.

This property has no default.

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

The ToolbarEditVisible property is a read-write property that shows or hides the cut, copy, and paste toolbar buttons.

Syntax

```
aaHistClientQuery.ToolbarEditVisible = discrete;  
Result = aaHistClientQuery.ToolbarEditVisible;
```

Remarks

The default is True.

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

This read-write property shows or hides the entire toolbar.

Syntax

```
aaHistClientQuery.ToolbarVisible = discrete;  
Result = aaHistClientQuery.ToolbarVisible;
```

Remarks

The default is True, that is, the toolbar is visible.

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 Wonderware 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](#) on page 27.

aaHistClientQuery Methods

The aaHistClientQuery methods are:

- AddServer
- AddServerEx
- AddTag
- ClearTags
- CopyQuery
- CutQuery
- FileOpen
- ManualConnect
- OpenQuery
- PasteQuery
- RemoveTag
- Refresh
- SaveQuery
- SaveResults
- SetDates
- SetDuration
- SetQueryType
- SetQueryType2
- SetTimeSpan
- ShowAbout

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.

AddServerEx

The AddServerEx 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.

password

A valid password for the server.

bPersistPassword

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.

Return Value

Returns True if the server can be added; otherwise returns False.

Remarks

All parameters are required. Errors, if any, are reported.

AddTag

The AddTag method adds a tag to the tag collection.

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

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

The CopyQuery method copies the current selection in the query text box to the clipboard.

Syntax

```
[Result=] aaHistClientQuery.CopyQuery();
```

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

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).

ManualConnect

The ManualConnect method opens the **Server connection** dialog box.

Syntax

```
[Result=] aaHistClientQuery.ManualConnect();
```

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

The PasteQuery method pastes the current contents of the clipboard to the query text box.

Syntax

```
[Result=] aaHistClientQuery.PasteQuery();
```

Refresh

The Refresh method re-executes the query.

Syntax

```
[Result=] aaHistClientQuery.Refresh();
```

Remarks

The focus must be on the **Results** tab for this method to take effect.

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

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();
```

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

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` on page 678.

Return Value

Returns `True` if the dates were set. Returns `False` in case of an error.

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` on page 678.

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

The SetQueryType method selects the specified query type and tag type in the Tag Picker.

Syntax

```
[Result=]
aaHistClientQuery.SetQueryType(aaQueryTypeEnumeration
    queryType, aaTagType tagType);
```

Parameters*queryType*

The type of the query. For information on the valid enumerations, see aaQueryTypeEnumeration on page 524.

tagType

The type of the tag. For information on the valid enumerations, see aaTagType Enumeration on page 675.

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

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 `aaQueryTypeEnum` on page 524.

tagType

The type of the tag. For information on the valid values, see `aaTagType Enumeration` on page 675.

Return Value

Returns True if it can be shown; otherwise, returns False.

Remarks

Use this method in the InTouch HMI software instead of the `SetQueryType` method.

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` on page 678. For more information on setting the duration, see `aaTimeRangeEnumeration Enumeration` on page 676.

ShowAbout

The `ShowAbout` method opens the **About** dialog box.

Syntax

```
[Result=] aaHistClientQuery.ShowAbout();
```

aaHistClientQuery Events

The aaHistClientQuery events are:

- ModeChanged
- QueryChanged
- ServerChanged

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;
```

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

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.

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 Wonderware 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 Wonderware Historian.

Value	Enumeration	Description
13	ServerVersion	Retrieve the server version.
14	StorageStartDate	Retrieve the start date of data storage.
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.

Chapter 10

aaHistClientTagPicker Control

The aaHistClientTagPicker control allows you to view the hierarchy of objects in a Wonderware 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 on page 40.

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 on page 40.

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.

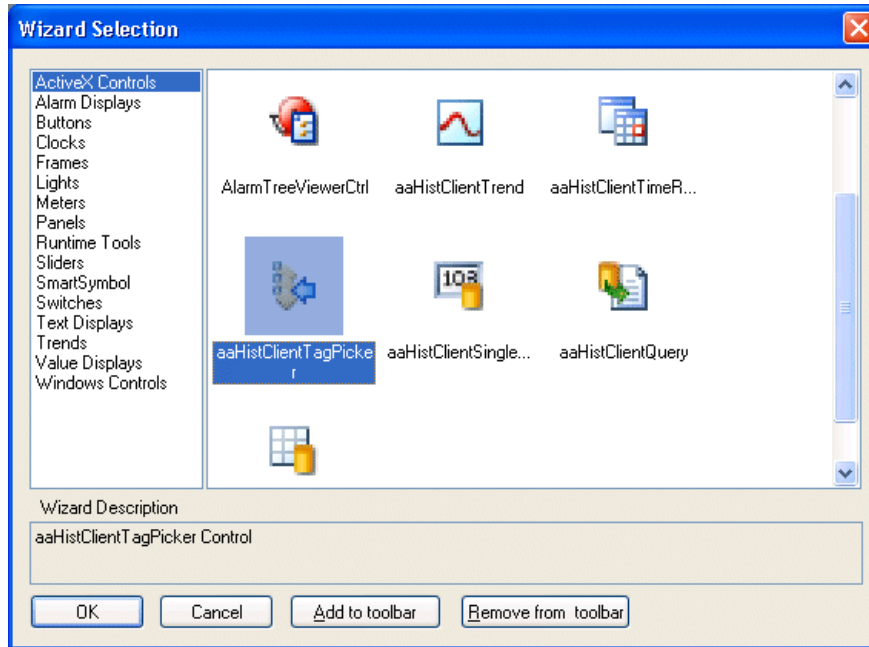
All properties, methods, and events can be controlled through scripting. In addition, some of these properties and methods are exposed through the aaHistClientTagPicker property panel available during application development.

Adding aaHistClientTagPicker to an InTouch Window

To add the aaHistClientTagPicker control



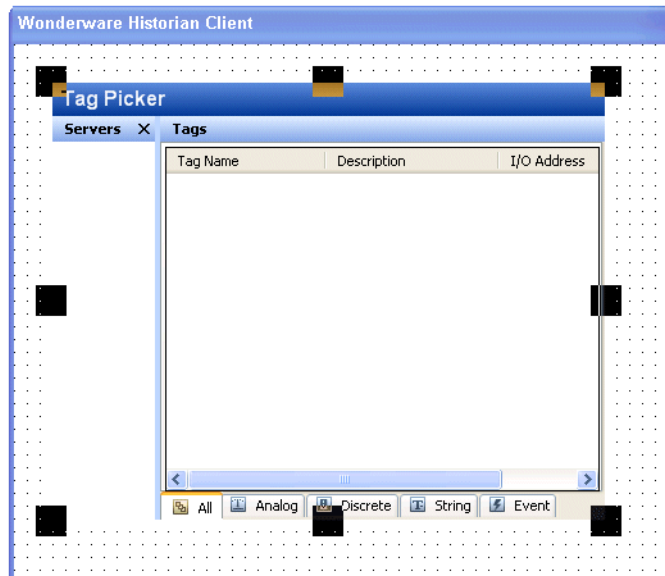
- 1 In WindowMaker, click the Wizards button. The Wizard Selection dialog box appears.



- 2 Select the aaHistClientTagPicker control.

- 3 Click OK.

The control appears in the window.



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

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 on page 579.

This property has no default value.

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

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

The FilterVisible property is a read-write property that shows or hides the Filter pane.

Syntax

```
aaHistClientTagPicker.FilterVisible = discrete;  
Result = aaHistClientTagPicker.FilterVisible;
```

Remarks

The default value is False.

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

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 ("").

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 a Wonderware 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 Wonderware Historian is installed on a case-sensitive SQL Server.

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.

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 on page 579.

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 );
```

SingleSelectionMode

The SingleSelectionMode 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.SingleSelectionMode = discrete;
Result = aaHistClientTagPicker.SingleSelectionMode;
```

Remarks

The default value is False.

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 =
  aaHistClientTagPicker.SplitterOrientation;
Result = aaHistClientTagPicker.SplitterOrientation;
```

Remarks

This aaHistClientTagPickerSplitterOrientation enumeration is used for the orientation types. For more information, see aaHistClientTagPickerSplitterOrientation Enumeration on page 539.

The default value is 0 (horizontal).

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;
```

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

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

The TreeVisible property is a read-write property that shows or hides the Servers pane.

Syntax

```
aaHistClientTagPicker.TreeVisible = discrete;  
Result = aaHistClientTagPicker.TreeVisible;
```

Remarks

The default value is True.

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

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

The `Visible` property is a read-write property that shows or hides the Tag Picker.

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

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

The LogOn method displays a dialog box for connecting to the specified server.

Syntax

```
[Result=] aaHistClientTagPicker.Logon(aaServer server);
```

Parameter

server

The server to which to connect.

Remarks

This methods uses the aaServer object. For more information, aaServer Object on page 579.

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 Wonderware Historian is installed on a case-sensitive SQL Server. Errors, if any, are logged in the SMC Logger.

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

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.

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

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.

```
#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 Wonderware Information Server portal.

aaHistClientTagPicker Events

The aaHistClientTagPicker events are:

- OnFilterChanged
- OnGroupChanged
- OnTagsPicked
- OnTagsSelected

- OnServerChanged
- OnSelectedTabChanged
- OnTagNameChanged

OnFilterChanged

The OnFilterChanged event is triggered when the filter is changed.

Syntax

```
aaHistClientTagPicker.OnFilterChanged();
```

OnGroupChanged

The OnGroupChanged event is triggered when the tag group is changed in the navigation tree in the **Servers** pane.

Syntax

```
aaHistClientTagPicker.OnGroupChanged();
```

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

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

The OnServerChanged event is triggered when the server is changed.

Syntax

```
aaHistClientTagPicker.OnServerChanged();
```

OnSelectedTabChanged

The OnSelectedTabChanged event is triggered when the user changes tabs in the Tags pane.

Syntax

```
aaHistClientTagPicker.OnSelectedTabChanged();
```

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.

Chapter 11

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 on page 47.

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.

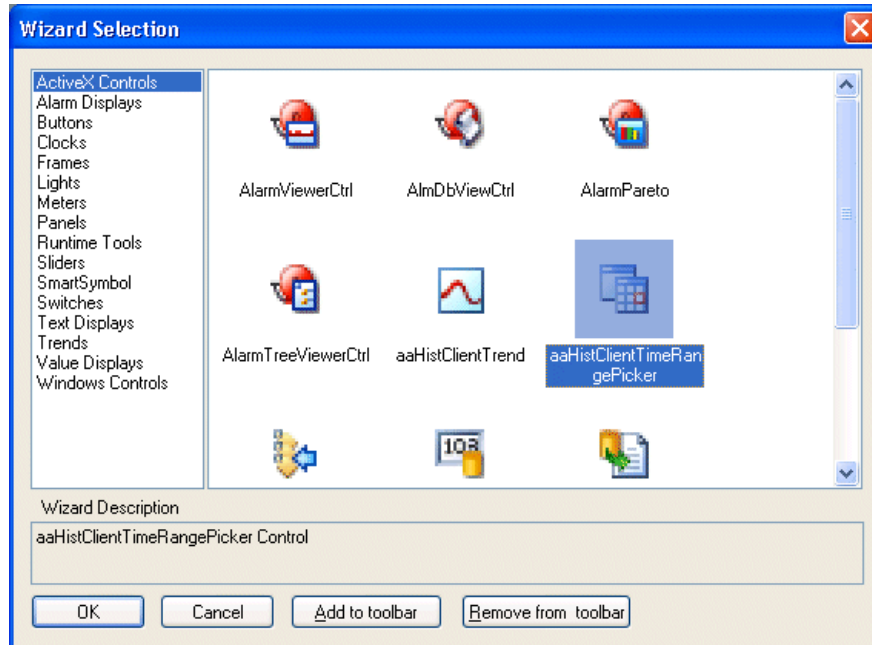
All properties, methods, and events can be controlled through scripting. In addition, some of these properties and methods are exposed through the `aaHistClientTimeRangePicker` property panel available during application development.

Adding aaHistClientTimeRangePicker to an InTouch Window

To add the aaHistClientTimeRangePicker control



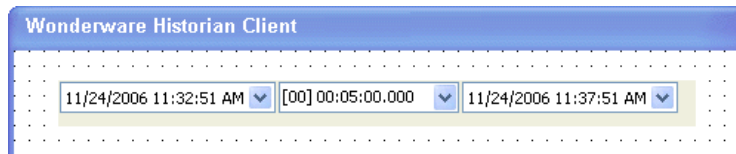
- 1 In WindowMaker, click the Wizards button. The Wizard Selection dialog box appears.



- 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

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

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

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).

Syntax

```
Result = aaHistClientTimeRangePicker.EndDateUTC;
```

Return Value

A message value in a valid date/time format in UTC is returned.

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 06/01/2001 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.
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.

Format String	Description
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

The StartDate property is a read-only property that returns the start date and time of the time range.

Syntax

```
Result = aaHistClientTimeRangePicker.StartDate;
```

Return Value

A message value in a valid date/time format is returned.

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

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` on page 676.

The default value is 18.

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

The GetStartAndEndTimes method retrieves the start and end times for the query.

Syntax

```
[aaTimeRangeEnumeration=]  
aaHistClientTimeRangePicker.GetStartAndEndTimes(Date  
Time startTime, DateTime endTime);
```

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 on page 676.

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.

Return Value

The time range enumeration (such as `Custom`, `Last5Minutes`, and so on) is returned. For more information, see `aaTimeRangeEnumeration Enumeration` on page 676.

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

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.

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](#) on page 676.

Remarks

The date and time formats are set using the `Format` property.

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` on page 676.

Remarks

The date and time formats are set using the `Format` property.

aaHistClientTimeRangePicker Events

The events for the `aaHistClientTimeRangePicker` are:

- `OnChange`

OnChange

The `OnChange` event is triggered when the start date and/or end dates are changed.

Syntax

```
aaHistClientTimeRangePicker.OnChange();
```

Chapter 12

aaHistClientActiveDataGrid Control

The aaHistClientActiveDataGrid control can execute any SQL query that returns a result set from any Wonderware Historian or Microsoft SQL Server database and returns the results in a grid.

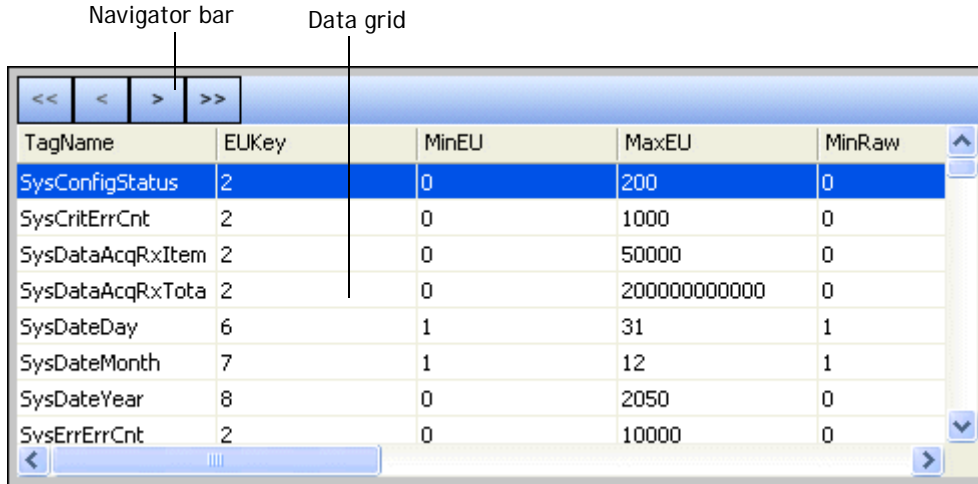
Note The aaHistClientActiveDataGrid does not support data definition or data manipulation queries.

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.





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.

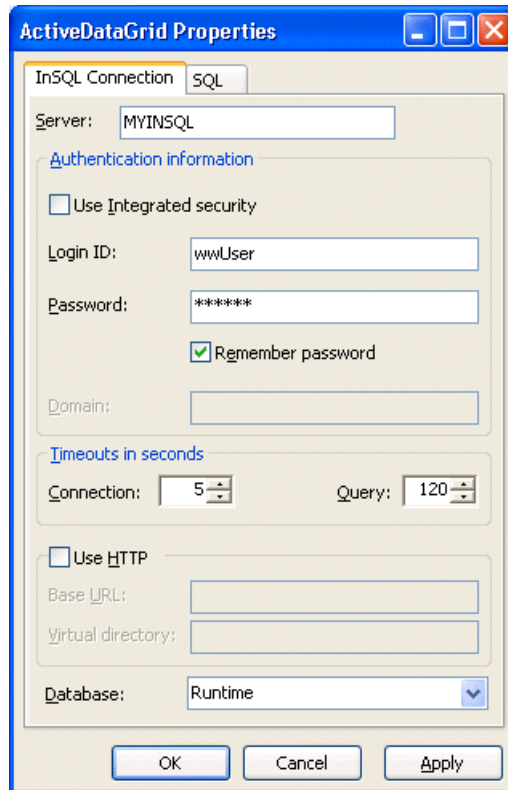
Note Depending on how the aaHistClientActiveDataGrid was configured during development, the navigator bar may not be available during runtime.

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](#) on page 27.

Note The aaHistClientActiveDataGrid control can only connect to single server. Multiple servers are not supported.

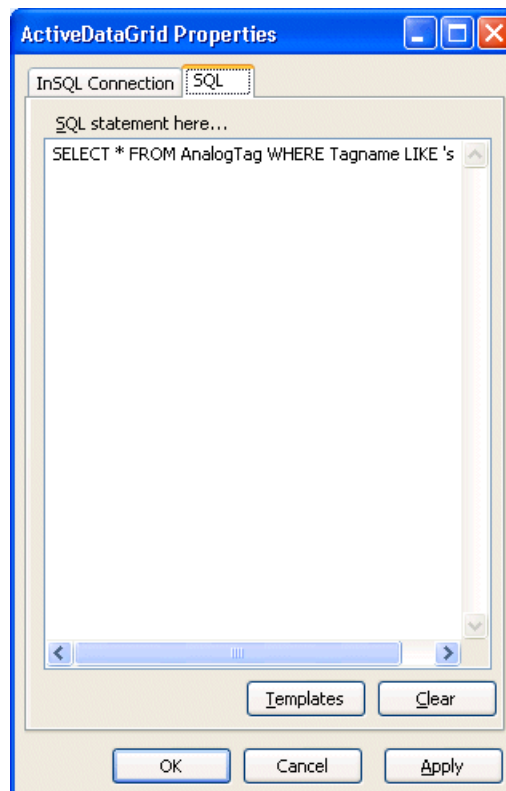
- 4 To apply the changes, click **Apply** or **OK**.
The grid is cleared, and the current SQL statement is re-executed according to the values specified.

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

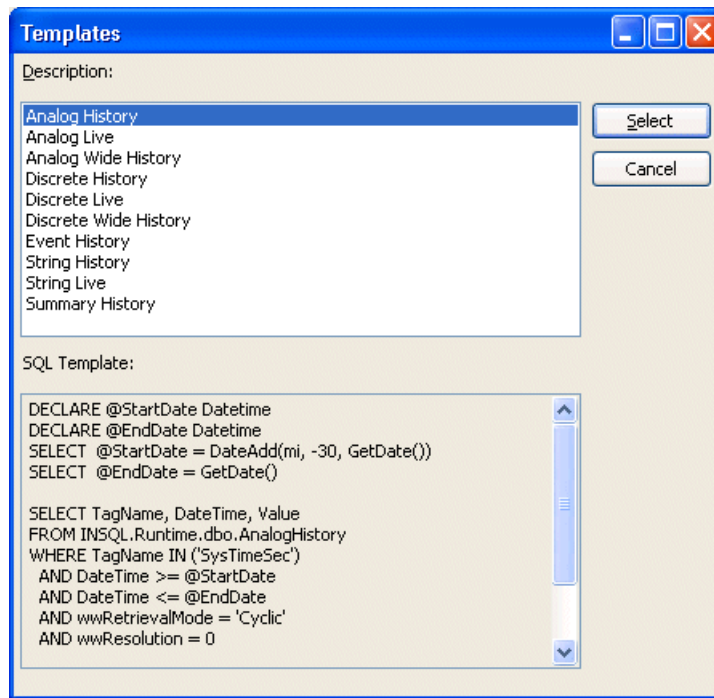
- 1 Right-click in the aaHistClientActiveDataGrid.
- 2 In the shortcut menu that appears, click SQL. The ActiveDataGrid Properties dialog box appears.
- 3 If the SQL tab is not already selected, click the SQL tab.



- 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

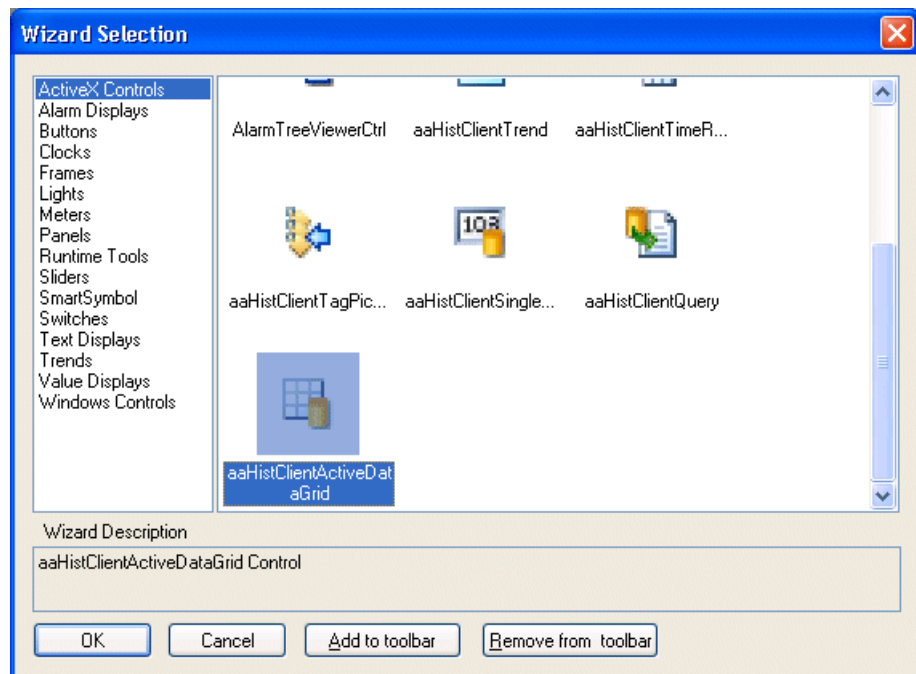
Use aaHistClientActiveDataGrid's properties, methods, and events to create scripts that set up a database connection and customize how the data grid functions during runtime.

Adding aaHistClientActiveDataGrid to an InTouch Window

To add the aaHistClientActiveDataGrid control

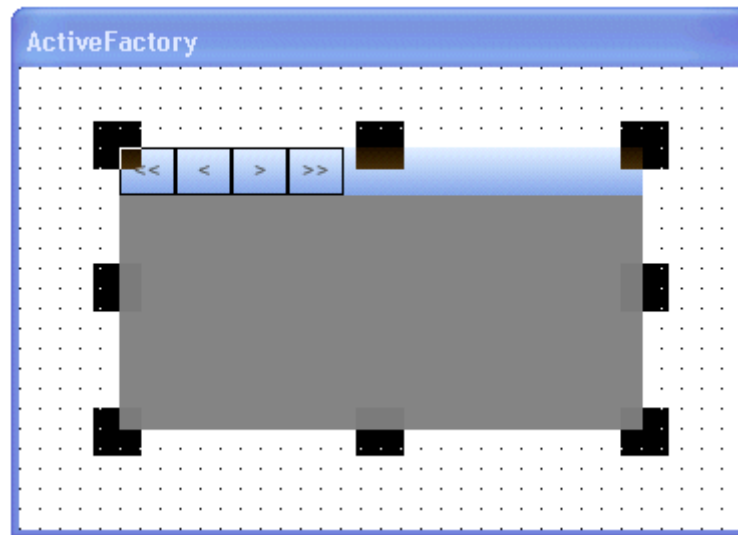


- 1 In WindowMaker, click the Wizards button. The Wizard Selection dialog box appears.



- 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

- ShowErrorDlg
- ShowNavigatorBar
- SQLString
- UserName
- VirtualDirectoryName

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.

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

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

The `BOF` property is a read-only property that returns whether the user has attempted to navigate prior to the 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.

BusinessObjectServer

This read-write property specifies the path to the HTTP server when using HTTP to access the historian.

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 [Using HTTP as the Server Connection Protocol](#) on page 32.

To enable HTTP access, you must also specify the virtual directory name using the `VirtualDirectoryName` property.

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

Use this read-write property to initiate or terminate a connection to the Wonderware Historian and to check whether a connection is currently active.

Syntax

```
aaHistClientActiveDataGrid.Connected = discrete;  
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 Wonderware 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

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 Wonderware 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

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;
```

Remarks

The default value is 100.

Domain

The `Domain` property is a read-write property that gets or sets the domain string for the connection to the server.

Syntax

```
aaHistClientActiveDataGrid.Domain = message;  
Result = aaHistClientActiveDataGrid.Domain;
```

Remarks

This property is used in cases where the Windows integrated security requires the domain name to be specified.

The default is an empty message value ("").

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.

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

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

The Handle property is a read-only property that returns the Window handle for the control.

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

The Password property is a write-only property that specifies the password for the provided username on the specified Wonderware Historian.

Syntax

```
aaHistClientActiveDataGrid.Password = message;
```

Remarks

See the Wonderware Historian documentation for the default passwords associated with the default usernames.

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;
```

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

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

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.

Remarks

The default value is 0.

ServerName

The ServerName property is a read-write property that specifies the name of the Wonderware 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 a Wonderware Historian.

This property has no default value.

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.

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

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 Wonderware 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 ("").

UserName

The `UserName` property is a read-write property that specifies the username used to logon to the Wonderware Historian specified in the `ServerName` property.

Syntax

```
aaHistClientActiveDataGrid.UserName = message;  
Result = aaHistClientActiveDataGrid.UserName;
```

Remarks

See the Wonderware Historian documentation for information on the default Wonderware Historian users.

Remarks

The default `UserName` is `wwUser`.

VirtualDirectoryName

The `VirtualDirectoryName` property is a read-write property that gets or sets the virtual directory name.

Syntax

```
aaHistClientActiveDataGrid.VirtualDirectoryName =  
    message;  
Result =  
    aaHistClientActiveDataGrid.VirtualDirectoryName;
```

Remarks

The default is an empty message value ("").

aaHistClientActiveDataGrid Methods

The aaHistClientActiveDataGrid methods are:

- ClearGrid
- ColumnName
- ColumnValue
- ColumnValueByName
- Execute
- MoveFirst
- MoveLast
- MoveNext
- MovePrevious
- RowColumnValue
- RowColumnValueByName
- ShowPropertiesDialog
- SQLAppend

ClearGrid

The ClearGrid method clears the contents of the data grid and sets the Connected property to False.

Syntax

```
aaHistClientActiveDataGrid.ClearGrid();
```

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

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.

Return Value

A message representation of the data.

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

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.

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:

- 1 The specified server is not running or connection to it is not available.
- 2 The server assigned to the ServerName property is invalid or not found.
- 3 The username assigned to the UserName property is invalid or not found.
- 4 The password assigned to the Password property is invalid or not associated with the specified UserName on the specified ServerName.
- 5 There is a syntax error in the SQLString property.
- 6 The DatabaseName property was not assigned or the wrong database was specified.
- 7 The BusinessObjectServer property is set to an HTTP server that does not exist, or the HTTP server specified is not running

MoveFirst

The MoveFirst method selects the first row in the data grid.

Syntax

```
aaHistClientActiveDataGrid.MoveFirst();
```

MoveLast

The MoveLast method selects the last row in the data grid.

Syntax

```
aaHistClientActiveDataGrid.MoveLast();
```

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

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

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);
```

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

The RowColumnValueByName method gets the value at the indicated row and column (specified by name).

Syntax

```
[Result=]  
aaHistClientActiveDataGrid.RowColumnValueByName(integ  
er 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.

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

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.

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`
- `OnDbClick`
- `OnError`

For information on ambient events, see [Common Events](#) on page 672.

OnClick

The `OnClick` event is triggered every time the user clicks on a data row in the control.

Syntax

```
aaHistClientActiveDataGrid.OnClick;
```

OnDbClick

The `OnDbClick` event is triggered every time the user double-clicks on a data row in the control.

Syntax

```
aaHistClientActiveDataGrid.OnDbClick;
```

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 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:

```

TErrorNo = #ThisEvent.OnErrorerrorNo; {Assign error
    number from the event to a tag called TErrorNo}
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 TErrorNo == 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 TErrorNo == 1 THEN
            #ThisEvent.OnErrorerrorString = "Not able to
            connect to the specified server.";
        ELSE IF TErrorNo == 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

The following example demonstrates how to connect to the Wonderware 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')");
#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

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()");
#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 Wonderware 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.
10	The column name, <Column Name>, was not found.
11	The RefreshFrequency property must be assigned a positive number.

Chapter 13

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 Wonderware Historian Client objects and controls.

aaServer Properties

The aaServer properties are:

- 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 Wonderware Historian as a message.

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 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;
```

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.

PatchLevel

The PatchLevel property is a read-only property that returns the patch level of the Wonderware 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.

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 Wonderware 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 Wonderware Historian Client schema version for the server.

Syntax

```
Result = aaServer.SchemaVersion;
```

Return Value

Returns the schema version 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.

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 on page 596.

Remarks

This property always returns a value of 1.

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 on page 595.

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 Wonderware 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.

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 on page 592.

LogOn

The LogOn method creates a connection (logs on) to the server.

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 on page 592.

aaServers Object

The aaServer 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 Wonderware 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 Wonderware Historian.

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 on page 579.

Remarks

This property is not supported in the InTouch HMI software.

This property has no default value.

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 on page 579.

GetServer

The GetServer method gets the aaServer object for a server from the server list.

Syntax

```
[Result=] aaServers.GetServer(message name);
```

name

The name of the server to get.

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 on page 579.

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.

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 Wonderware Historian Client Trend and the Wonderware 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

This parameter is not used.

args

The server state change arguments. For more information on the aaServerListChangeArgs object, see aaServerListChangeArgs Object on page 593.

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 on page 679.

args

The server state change arguments. For more information on the aaServerListChangeArgs object, see aaServerListChangeArgs Object on page 593.

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 on page 679.

args

The server state change arguments. For more information on the aaServerListChangeArgs object, see aaServerListChangeArgs Object on page 593.

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 on page 679.

args

The server state change arguments. For more information on the aaServerStateChangeArgs object, see aaServerStateChangeArgs Object on page 594.

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.

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 on page 579.

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;
```

Return Value

The aaServer instance. For more information on the aaServer object, see aaServer Object on page 579.

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 on page 595.

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 on page 678.

Remarks

This property has no default value.

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.

Value	Enumeration	Description
0	isSvrPdssrv	For backward-compatibility only.
1	isSvrSQLServer	For backward-compatibility only.
2	isSvrNotInSQL	For backward-compatibility only.

Chapter 14

aaHistClientSingleValueEntry Control

Use the aaHistClientSingleValueEntry control to manually add a tag value to the Wonderware Historian database.

Using the aaHistClientSingleValueEntry Control at Runtime

Use the aaHistClientSingleValueEntry control to manually add single data value for a tag in the Wonderware 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 Wonderware Historian's history blocks. Therefore, you can retrieve them using the same views and tables as data that is stored automatically by the Wonderware 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 on page 40.
- 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

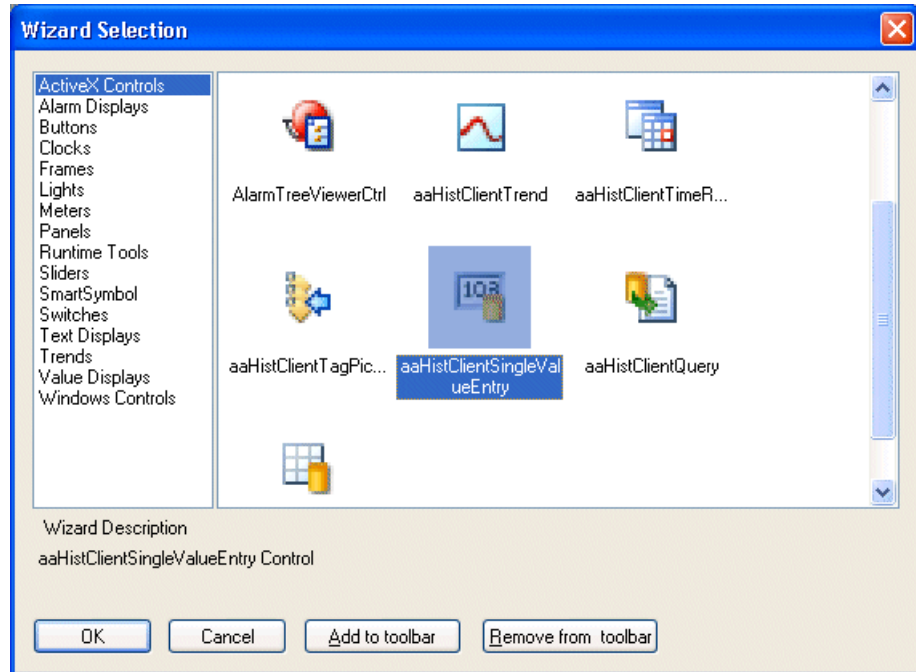
You can use the aaHistClientSingleValueEntry control's properties, methods, and events in runtime scripts in your application to control the functionality available to the runtime user.

Adding the aaHistClientSingleValueEntry Control to an InTouch Window

To add the aaHistClientSingleValueEntry control



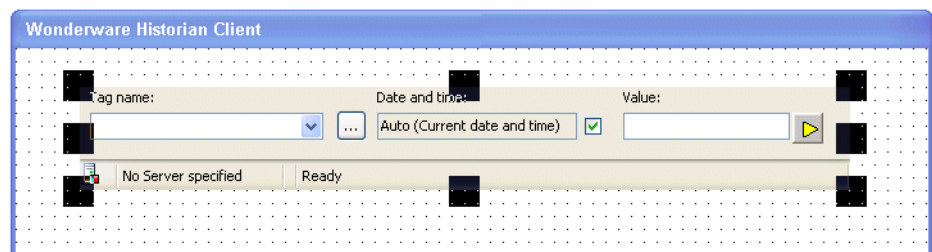
- 1 In WindowMaker, click the Wizards button. The Wizard Selection dialog box appears.



- 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
- DisplayErrorMessage
- FieldLabelPosition
- FieldLayoutHorizontal
- HideDateTimeModeTabs
- HideFieldLabels
- HideStatusBar
- InsertButtonDisable
- InsertButtonVisible
- InTouchDateTime
- LastErrorDetails
- LastErrorMessage
- LastOperationResult
- LastOperationSuccessful
- Pwd
- Quality
- QualityDetail
- QualityDetailFieldDisable
- QualityDetailFieldVisible
- QualityFieldDisable
- QualityFieldVisible

- RememberEnteredTags
- Servers
- StringValue
- TagName
- TagNameFieldDisable
- TagNameFieldVisible
- TagPickerButtonDisable
- TagPickerButtonVisible
- Tags
- TagType
- TagValid
- Transparent
- User
- UseTimezone
- Value
- ValueEx
- ValueFieldDisable

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.

CurrentServerName

The `CurrentServerName` property is a read-write property that gets or sets the name of the Wonderware 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

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` on page 678.

Remarks

The default value is 12:00:00 AM.

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

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

The DateTimeString property is a read-write property that gets and sets the timestamp as a string value to be used for the insert.

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

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.

DisplayErrorMessage

The `DisplayErrorMessage` property is a read-write property that enables or disables the display of error message dialog boxes.

Syntax

```
aaHistClientSingleValueEntry.DisplayErrorMessage =  
    discrete;  
Result =  
    aaHistClientSingleValueEntry.DisplayErrorMessage;
```

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

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` on page 623.

If the `FieldLayoutHorizontal` property is set to `True`, the `FieldLabelPosition` property has no effect.

The default value is `0`.

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

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.

If the DateTimeFieldVisible property is set to False, the HideDateTimeModeTags property is overridden.

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

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.

InsertButtonDisable

The InsertButtonDisable property is a read-write property that gets or sets whether the **Insert** button is available in the control at runtime.

Syntax

```
aaHistClientSingleValueEntry.InsertButtonDisable =  
    discrete;  
Result =  
    aaHistClientSingleValueEntry.InsertButtonDisable;
```

Remarks

If set to True, the button is not available.

The default value is False.

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

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.

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

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

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.

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

The LastOperationSuccessful property is a read-only property that gets the status of the last data value insert.

Syntax

```
Result =  
    aaHistClientSingleValueEntry.LastOperationSuccessful;
```

Remarks

If set to True, the last insert was successful.

To reset this property, use the Reset method.

The default value is False.

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

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

1 = Bad

16 = Doubtful

The default value is -1.

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 Wonderware 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

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.QualityDetailFieldDisabl
    e;
```

Remarks

The default value is **False**.

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.QualityDetailFieldVisibl
    e;
```

Remarks

The default value is **False**.

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

The `QualityFieldVisible` property is a read-write property that gets or sets whether the **Quality** box is visible in the control at runtime.

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

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

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;
```

Remarks

This property references the `aaServers` object. For more information, see `aaServer` Object on page 579.

This property has no default value.

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

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.

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

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

The TagPickerButtonDisable property is a read-write property that gets or sets whether the Tag Picker button to 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

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

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);
```

Remarks

For more information on the aaTag object, see Chapter 15, aaTag Object.

This property is not accessible in the InTouch HMI software.

This property has no default value.

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

The TagValid property is a read-only property that gets whether the current tag is valid.

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

The User property is a read-write property that gets or sets the current user for a Wonderware Historian.

Syntax

```
aaHistClientSingleValueEntry.User = message;  
Result = aaHistClientSingleValueEntry.User;
```

Remarks

Important To insert data for a tag, a user must have wwAdministrator privileges for the Wonderware 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

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;
```

Remarks

For more information on the aaUseTimeZoneEnumeration enumeration, see aaUseTimeZoneEnumeration Enumeration on page 623.

The default value is 0.

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

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.

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

AddServer

The AddServer method adds a server to the list.

Syntax

```
[Result=]
aaHistClientSingleValueEntry.AddServer(message
serverName, message loginName, message password,
[discrete bPersistPassword]);
```

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 Wonderware 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.

AddServerEx

The AddServer 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 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 Wonderware 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.

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.

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

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

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);
```

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

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

The Insert method inserts a value for a manual tag.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Insert();
```

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

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.

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

The `Refresh` method forces a repaint of the control.

Syntax

```
[Result=] aaHistClientSingleValueEntry.Refresh();
```

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`
- `InsertFail`
- `TagNameChanged`
- `ValueChanged`

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

The InsertComplete event is triggered when a data value insert operation succeeds.

Syntax

```
aaHistClientSingleValueEntry.InsertComplete();
```

Remarks

This event is not triggered by the InsertValue method.

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

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

The ValueChanged event is triggered when the Value, StringValue, or AnalogValue property changes.

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.

Chapter 15

aaTag Object

When tags are read from a Wonderware 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 Wonderware 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 Wonderware 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
- TypeAsTagType
- Units

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

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.

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

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.

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

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.

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

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.

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

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.

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

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.

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 on page 579. The server cannot be changed after construction.

Remarks

The default value is the name that was specified when the tag was created.

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.

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](#) on page 675.

The default value is 0.

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.

Chapter 16

aaHistClientWorkbookRunner and aaHistClientReportRunner Objects

The aaHistClientWorkbookRunner and aaHistClientReportRunner objects are used when reports are published to the Wonderware Information Server.

aaHistClientWorkbookRunner Object

The aaHistClientWorkbookRunner object is a control that is used to run reports created with the Wonderware 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 (.xls) and output the results (.htm).

aaHistClientWorkbookRunner Object Properties

The aaHistClientWorkbookRunner object properties include:

- ErrDescription
- ErrNumber
- OutputFile
- SourceFile
- ExcelVisible

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 ("").

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

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 ("").

SourceFile

The SourceFile property is a read-write property that specifies the name of the Excel file (.xls) to use to generate the report.

Syntax

```
aaHistClientWorkbookRunner.SourceFile = message;
```

```
Result = aaHistClientWorkbookRunner.SourceFile;
```

Remarks

You must specify the entire path and include the .xls extension.

The default is an empty message value ("").

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:

- Run
- RunReport
- RunReport2

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 (.xls) 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

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,  
    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, .xls, 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:

$$\text{OutputFile} = \text{OutputPrefix} + \text{InputFile} + \text{year} + \text{month} + \text{day} + _ + \text{hour} + \text{minute} + \text{second}$$

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 .xls or .xlt file, the output file is an .xls file.

1 = .htm

2 = .xls

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.

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 (.xls) 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 additional binding options for custom filters, use the RunReport2 method.

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, .xls, 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:

$$\text{OutputFile} = \text{OutputPrefix} + \text{InputFile} + \text{year} + \text{month} + \text{day} + _ + \text{hour} + \text{minute} + \text{second}$$

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 .xls or .xlt file, the output file is an .xls file.

1 = .htm

2 = .xls

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.

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 Wonderware 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.

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 (.xls) 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 Wonderware 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
- SourceFile
- WordVisible

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

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.

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

The SourceFile property is a read-write property that specifies the name of the Word template file (.htm) to use to generate the report.

Syntax

```
aaHistClientReportRunner.SourceFile = message;  
Result = aaHistClientReportRunner.SourceFile;
```

Remarks

You must specify the entire path and include the .htm extension.

The default is an empty message value ("").

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

The Run method processes the Word report.

Syntax

```
[Result=] aaHistClientReportRunner.Run();
```

Return Value

Returns True if the report generation was successful; otherwise returns False.

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.

Chapter 17

Workbook and Report Automation Objects

The Wonderware Historian Client Workbook and the Wonderware Historian Client automation objects allow you to automate the Wonderware Historian Client Workbook and Report from a scripting environment, such as Visual Basic for Applications.

Wonderware Historian Client Workbook Object

To automate the generation of reports from the Wonderware Historian Client Workbook, use the Wonderware Historian Client Workbook object within the scripting environment.

Wonderware Historian Client Workbook Object Methods

The Wonderware Historian Client Workbook object methods are:

- AddServer
- Auto_Close
- Auto_Open

- `GetLastError`
- `RunReport`
- Wonderware Historian Client Workbook Menu Methods
- Wonderware Historian Client Workbook Functions

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)
```

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

The `Auto_Close` method removes the Wonderware Historian Client toolbar and resets the main menu for Excel.

Syntax

```
ActiveFactoryWorkbook.Auto_Close()
```

Auto_Open

The `Auto_Open` method adds the Wonderware Historian Client toolbar and adds the Wonderware Historian Client menu to the main menu for Excel.

Syntax

```
ActiveFactoryWorkbook.Auto_Open();
```

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.

RunReport

The RunReport method processes the Workbook report. This method uses the date/time binding feature of Workbook, plus custom binding filters.

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, .xls, 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:

$$\text{OutputFile} = \text{OutputPrefix} + \text{InputFile} + _ + \text{year} + \text{month} + \text{day} + \text{hour} + \text{minute} + \text{second}$$

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 .xls or .xlt file, the output file is an .xls file.

1 = .htm

2 = .xls

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 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.

customFilters

A string of name-value pairs used to pass information from the Wonderware 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.

Wonderware 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.
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.

Method	Used to
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.

Wonderware Historian Client Workbook Functions

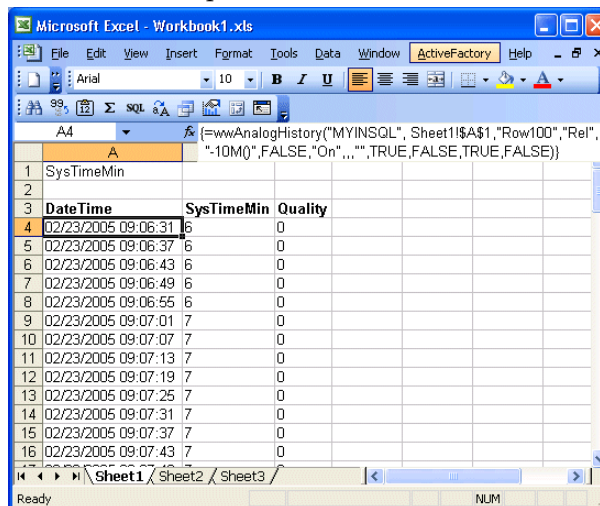
For more information see Wonderware Historian Client Workbook Function Reference on page 330.

Wonderware Historian Client Workbook Automation Example

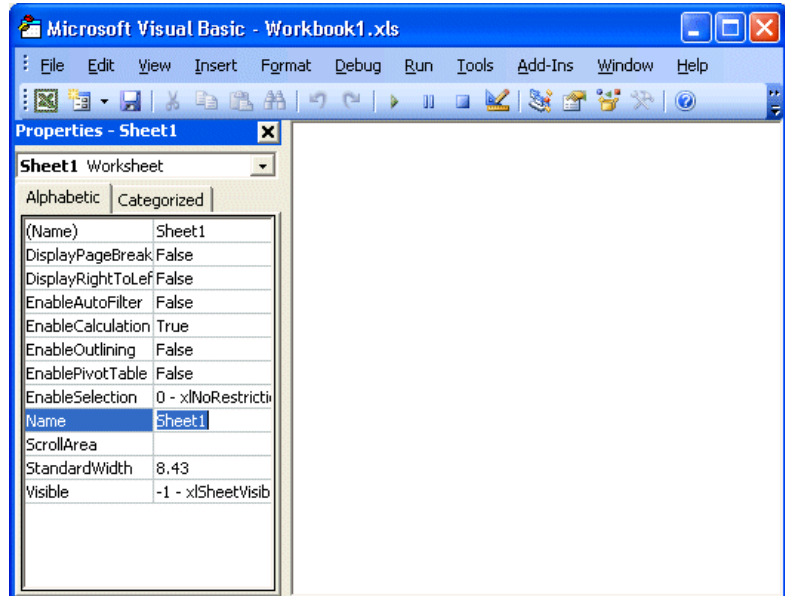
The following example illustrates how to automate the Wonderware Historian Client Workbook within Visual Basic for Applications (VBA). In this example, a button is added to the workbook that can be used to convert all of the functions in the sheet to values. This example uses the `ActiveFactoryWorkbook.mnuConvertSheet` method.

To automate the Wonderware Historian Client Workbook

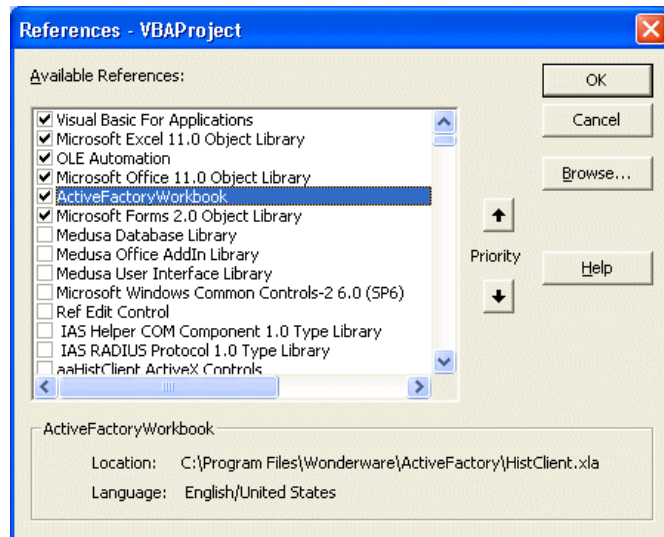
- 1 Start Excel and create a Wonderware Historian Client Workbook spreadsheet.



- 2 On the **Tools** menu, point to **Macro** and then click **Visual Basic Editor**. The Microsoft Visual Basic editor appears.

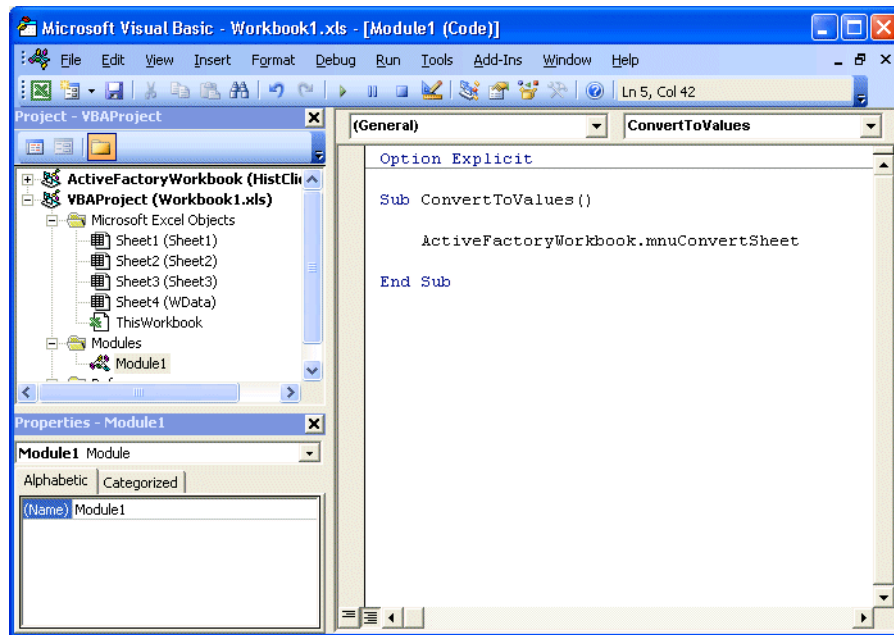


- 3 On the **Tools** menu, click **References**. The **References - VBAPProject** dialog box appears.

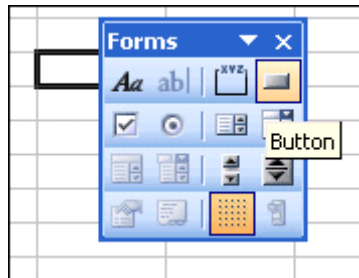


- 4 Select the **ActiveFactoryWorkbook** check box.
- 5 Click **OK**.
- 6 On the **Insert** menu, click **Module** to add a new module to the project.

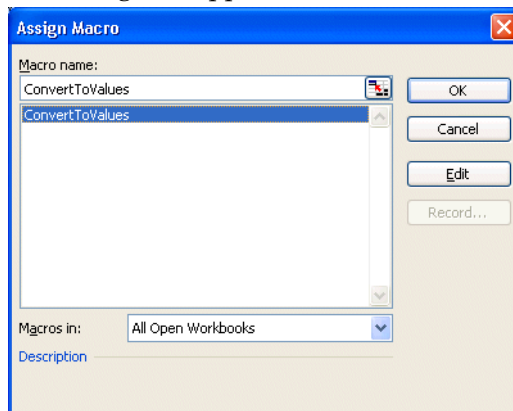
- 7 Add a subroutine that executes the `mnuConvertSheet` method.



- 8 Switch back to Excel.
- 9 On the View menu, point to Toolbars and then click Forms to open the Forms toolbar.

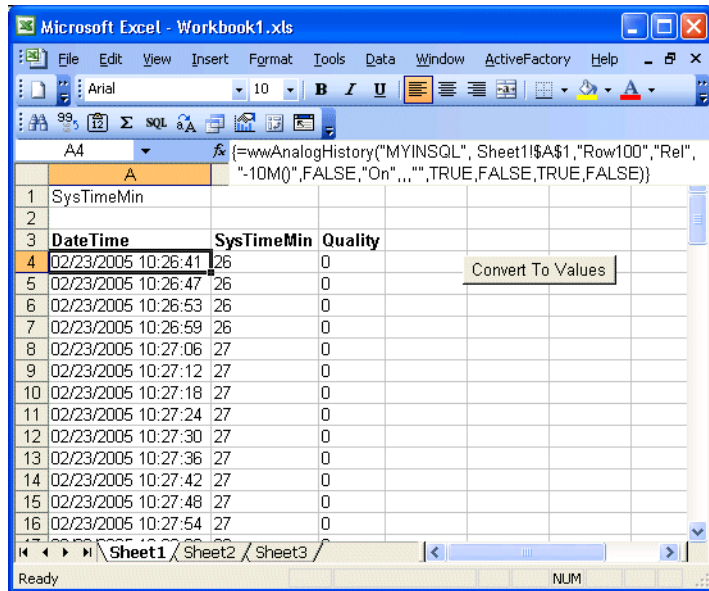


- 10 Insert a button into the worksheet. The Assign Macro dialog box appears.

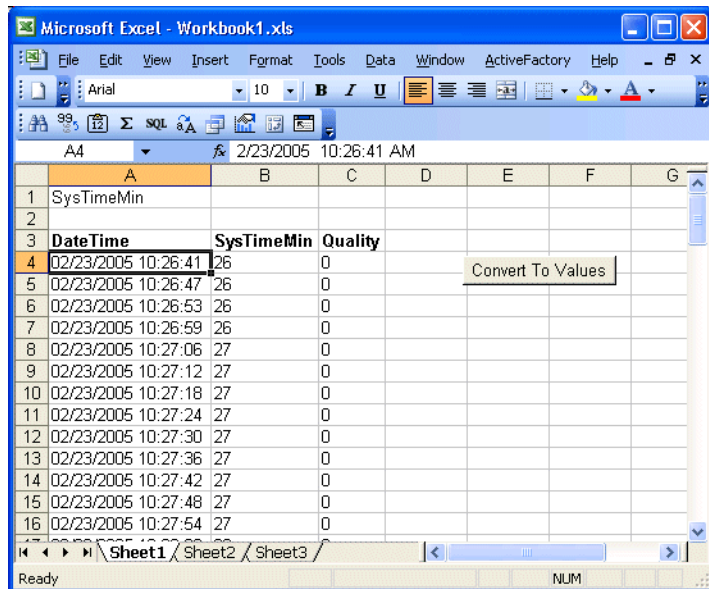


- 11 In the Macro name list, select `ConvertToValues`, which is the subroutine that you created in Step 7.

- 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.



Wonderware Historian Client Report Object

To automate the generation of reports from the Wonderware Historian Client Report, use the Wonderware Historian Client Report object within the scripting environment.

Report Object Properties

The Report object properties are:

- ReportDate
- ReportTime

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 on page 383.

The default value is the current date of when Microsoft Word was launched.

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 on page 383.

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

The AutoExec method initializes values.

Syntax

```
ActiveFactoryReport.AutoExec()
```

AutoExit

The AutoExit method removes the Wonderware Historian Client toolbar and resets the main menu for Word.

Syntax

```
ActiveFactoryReport.AutoExit()
```

RunReport

The RunReport method processes the Word report.

Syntax

```
ActiveFactoryReport.RunReport()
```

Return Value

Returns False if the report generation was successful; otherwise returns True.

Remarks

Any message dialog boxes are suppressed.

Chapter 18

aaHistClientGlobalFunctions Object

This object provides methods for accessing the information about the Wonderware 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 Wonderware Historian Client software.

The ProgID for the GlobalFunctions object is:

```
ArchestraA.HistClient.Util.aaHistClientGlobalFunctions
```

aaHistClientGlobalFunctions Methods

The aaHistClientGlobalFunctions methods are:

- GetDictionaryPath
- GetInstallPath
- GetAFVersion
- GetWorkstationName
- MDACOk

GetDictionaryPath

The GetDictionaryPath method returns the path to the dictionary file.

Syntax

```
[Result=]  
aaHistClientGlobalFunctions.GetDictionaryPath();
```

Return Value

Returns an empty string.

Remarks

This method is provided for backward compatibility only.

GetInstallPath

The GetInstallPath method returns the path where the Wonderware Historian Client software is installed.

Syntax

```
[Result=] aaHistClientGlobalFunctions.GetInstallPath();
```

Return Value

Returns a fully-qualified path to the installation folder as a string.

GetAFVersion

The GetAFVersion method returns the version of the Historian Client software. For example, this method returns 10.0.0.0 as the version of the Historian Client software.

Syntax

```
[Result=] aaHistClientGlobalFunctions.GetAFVersion();
```

Return Value

Returns a string value containing the version of the Historian Client software.

GetWorkstationName

The GetWorkstationName method returns the name of the computer on which the Historian Client software is running. For example, this method returns the computer name as HYDDLFLN5877.

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

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 Wonderware Historian Client software, this method always returns True.

Chapter 19

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.

Note Not all of the common properties, methods, events, enums, and data types are used by all of the controls.

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
- ToolTipText
- Transparent
- WhatsThisHelpID
- Tag
- Top
- Visible
- 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;
```

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.

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.

Syntax

```
<objectname>.Enabled = discrete;  
Result = <objectname>.Enabled;
```

Font

The Font property is a read-write property that gets or sets the font object.

Syntax

```
<objectname>.Font = Font;  
Result = <objectname>.Font;
```

Remarks

This property controls the font appearance for all text in the user interface.

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;  
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.

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.

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

The Object property is a read-only property that gets the object in a control.

Syntax

```
<objectname>.Object = Object;  
Result = <objectname>.Object;
```

Remarks

This property is not available in the InTouch HMI software.

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.

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 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.

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();
```

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
- 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();
```

DbClick

The DbClick event is triggered when the user double-clicks on the object at runtime with the mouse.

Syntax

```
<objectname>.DbClick();
```

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.

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);
```

Parameters

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();
```

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

aaRetrievalSource Enumeration

Specifies the source of process data to retrieve.

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.

Value	Enumeration	Description
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.
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.

Enum	Value	Description
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.
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:

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 upper 8 bits being 0 by default, but 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 hexadecimal value for this color is 0x00FF0000. The decimal value is 16711680.

0xA0FF0000 is half-transparent, half-opaque 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 (half-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*.

Appendix A

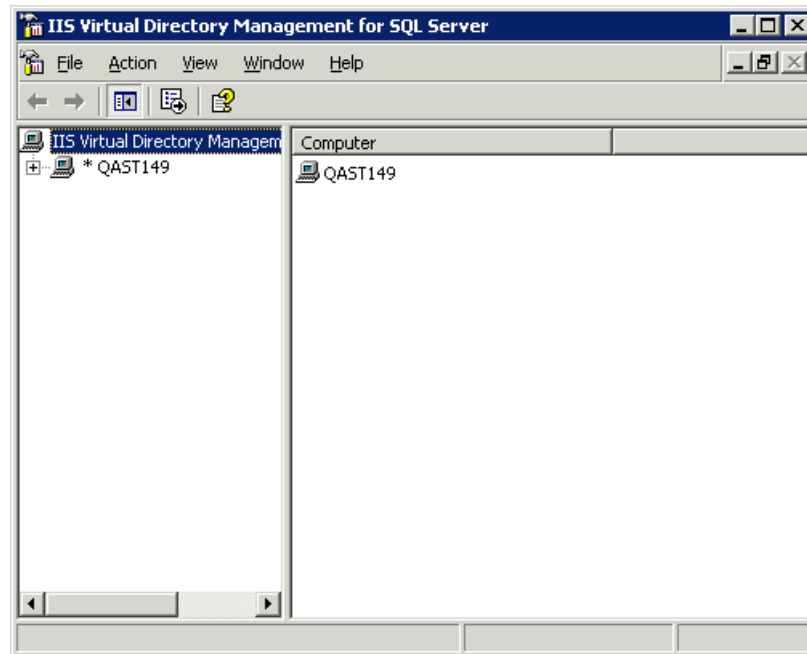
Configuring an IIS Virtual Directory for SQL Server

To enable access to SQL Server via HTTP, you must install the SQLXML software and set up a virtual directory for the Wonderware Historian Client software. The following instructions show how to set up a virtual directory on the Windows 2003 Server operating system.

To define and register a new virtual directory, use the IIS Virtual Directory Management for SQL Server utility on the computer running the Microsoft Internet Information Services (IIS) software. Use this utility to associate a new virtual directory and an instance of Microsoft SQL Server.

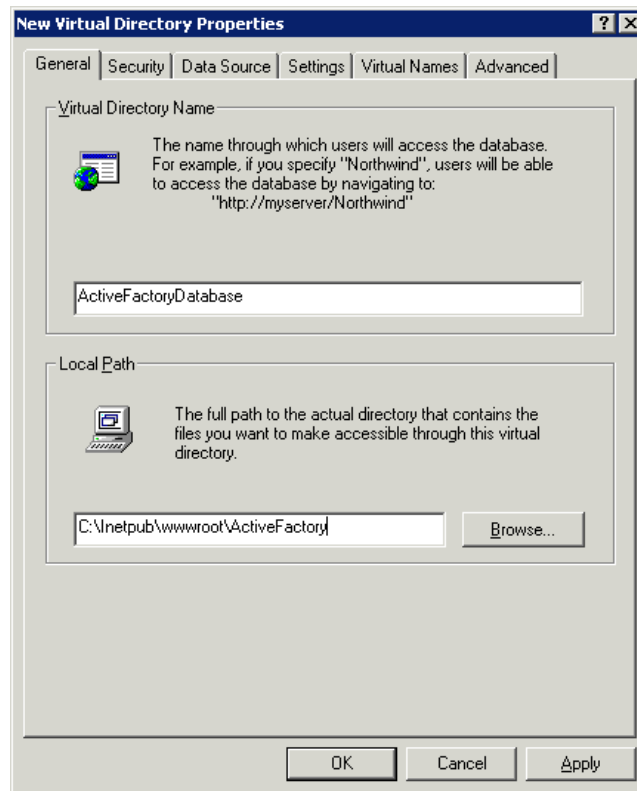
To configure an IIS virtual directory

- 1 Create or determine a physical directory to use for the virtual directory. For example:
c:\Inetpub\wwwroot\ActiveFactory.
- 2 On the Windows Start menu, point to **Programs**, point to **Microsoft SQL Server**, and then click **Configure SQL XML Support in IIS**. The **IIS Virtual Directory Management for SQL Server** console appears.



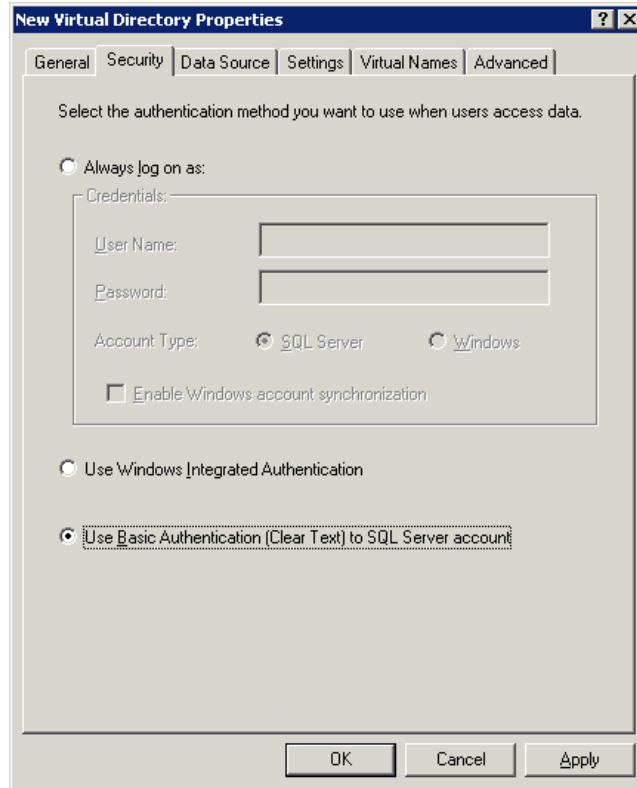
- 3 Expand the server name.
- 4 Right-click on **Default Web Site**, point to **New**, and then click **Virtual Directory**. The **New Virtual Directory Properties** dialog box appears.

5 Click the General tab.



- 6 In the **Virtual Directory Name** box, type the name through which the Wonderware Historian Client users access the Wonderware Historian. This name can be any name you choose, but keep in mind that it is used in the client-side configuration for the server.
- 7 In the **Local Path** box, type the path to the physical directory that you want to use as the virtual directory. Or, click the **Browse** button to select an existing path.

8 Click the Security tab.



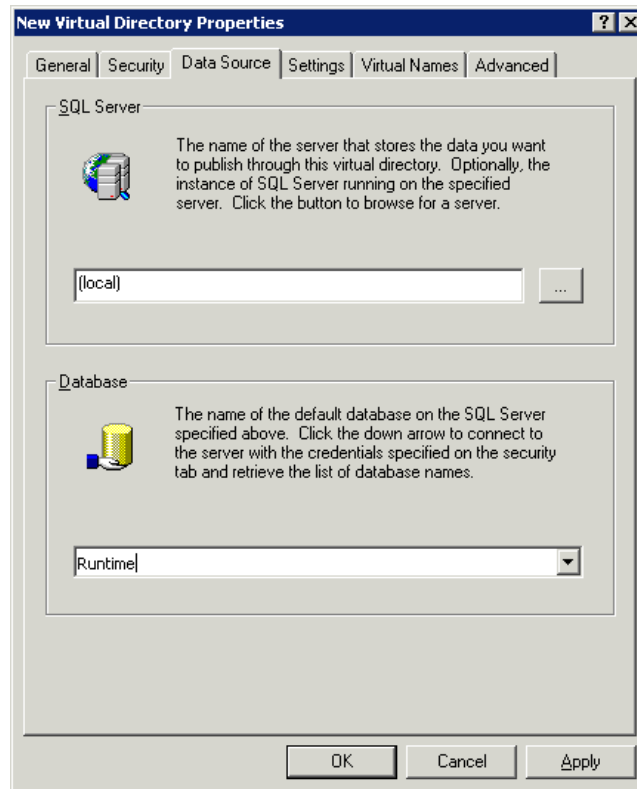
9 Select the appropriate authentication mode. For more information, see Authentication for HTTP Access on page 33.

For the **Always log on as** option, no password is required for the client, which means security has to be managed differently.

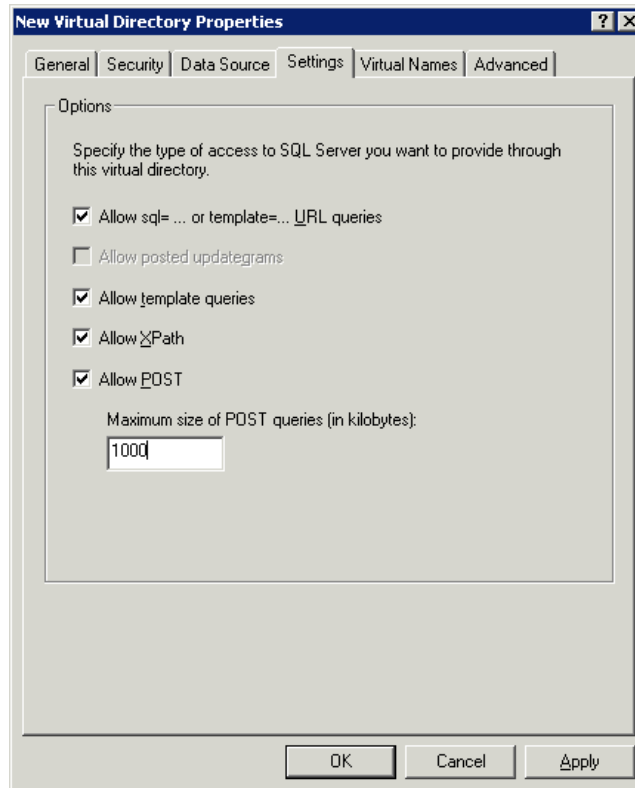
For the **Use Windows Integrated Authentication** option, integrated security is used, which means that the client must supply the user name, password and domain (if necessary).

For the **Use Basic Authentication** option, a SQL Server login is used (user name and password). The IIS Server must receive these from the client and sends them to the SQL Server.

10 Click the Data Source tab.



- 11 In the SQL Server box, type the name of the SQL Server to use. You may be prompted to provide login information to connect to the SQL Server.
- 12 In the Database list, select the Runtime database.

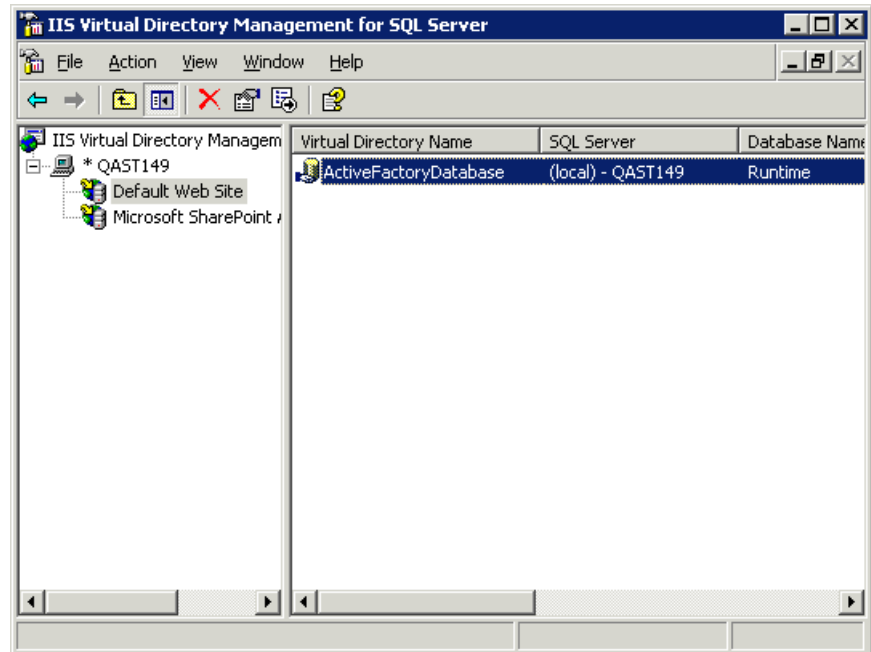
13 Click the **Settings** tab.

- 14** Select all of the check boxes except for **Allow posted updategrams** (which is unavailable).
- 15** In the **Maximum size of POST queries** box, leave the default value of 1000.

If publishing is to be done through the HTTP server, then this value must be at least as high as the largest size document published. Typically, an Excel file is between 30 and 80 kilobytes.

16 Click OK.

The new virtual directory is listed in the console window.



Appendix B

Data Retrieval 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 a Wonderware 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.

A Wonderware Historian with a version earlier than 9.0 supports two retrieval modes:

- Cyclic Retrieval
- Delta Retrieval

A Wonderware 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**

A Wonderware Historian with a version of 10.0 or higher supports the following additional mode:

- RoundTrip 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 Wonderware 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 Wonderware 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.

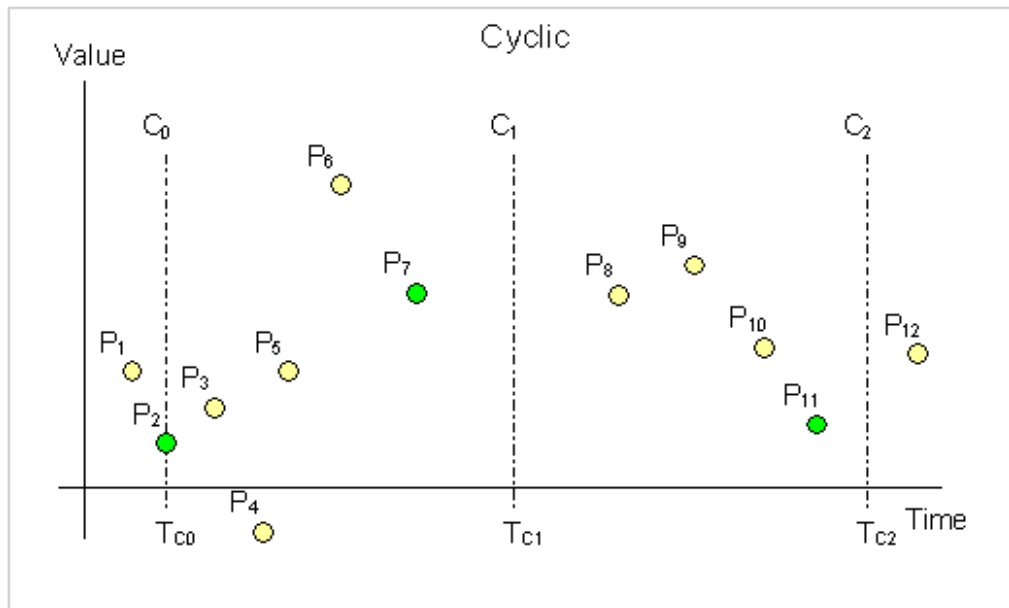
If no data value is actually stored at a cycle boundary, the last value before the boundary is returned.

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, and so on.) might fall between cycle boundaries. For an alternative, see “Best Fit” Retrieval on page 707.

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) on page 755.
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757.

- History Version (wwVersion) on page 769.
- Time stamp Rule (wwTimestampRule) on page 774 (only on IndustrialSQL Server 9.0 and later).

Cyclic Retrieval - Query Examples

To use the cyclic retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Cyclic'
```

Query 1

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 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 on page 697.

Cyclic Retrieval - Handling NULL Values

No special handling is done for NULL values. They are returned just like any other value.

Delta Retrieval

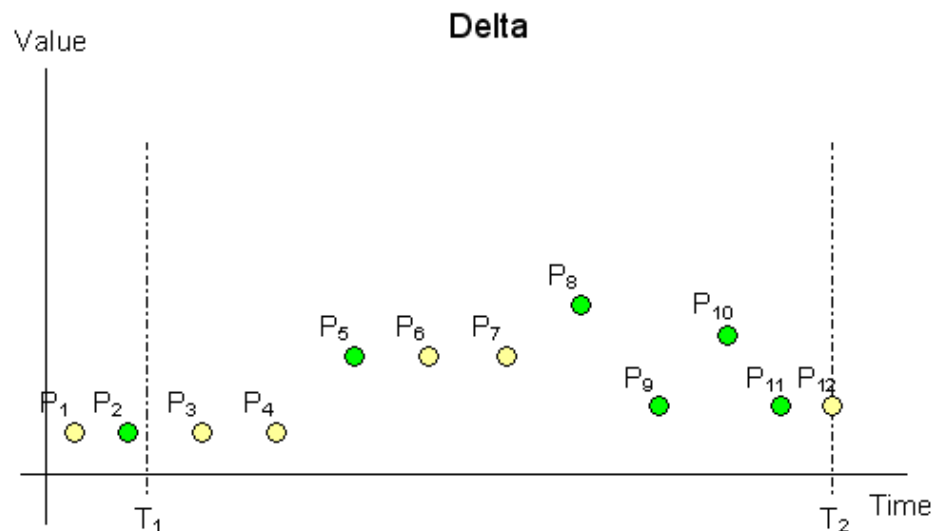
Delta retrieval, or retrieval based on exception, is the retrieval of only the changed values for a tag(s) for the given time interval. That is, duplicate values are not returned. It works with all types of tags.

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:

- 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) on page 762
- Value Deadband (wwValueDeadband) on page 766
- History Version (wwVersion) on page 769

Delta Retrieval - Query Examples

To use the delta retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Delta'
```

Query 1

As an example of how delta mode works, consider the following query:

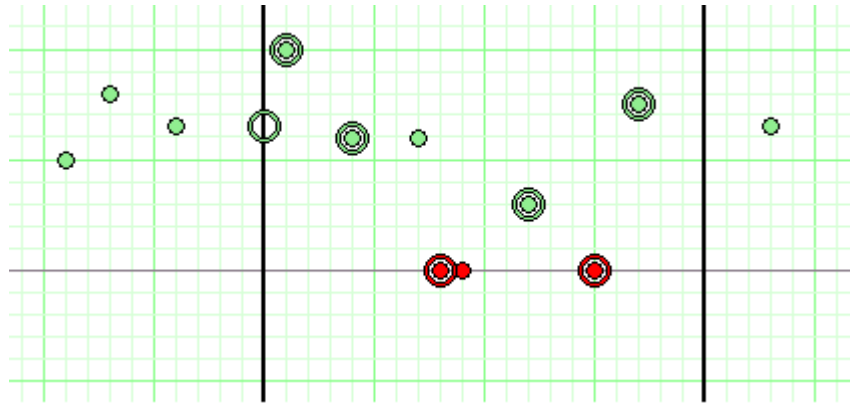
```
SELECT TagName, DateTime, Value, QualityDetail
FROM History
WHERE TagName = 'A001'
      AND DateTime >= '2009-09-12 00:20'
      AND DateTime <= '2009-09-12 00:40'
      AND wwRetrievalMode = 'Delta'
```

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

Tagname	DateTime	Value	QualityDetail
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:



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 *Retrieval Styles, Application Settings, and Tag Settings* on page 819.

Also, the use of a time deadband and/or value deadband with delta retrieval produces differing results. For more information, see *Time Deadband (wwTimeDeadband)* on page 762 and *Value Deadband (wwValueDeadband)* on page 766.

Query 1

```
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

Query 2

```
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
2001-07-27 03:03:10.000	10	0	192
2001-07-27 03:04:10.000	10	0	192

Query 3

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
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

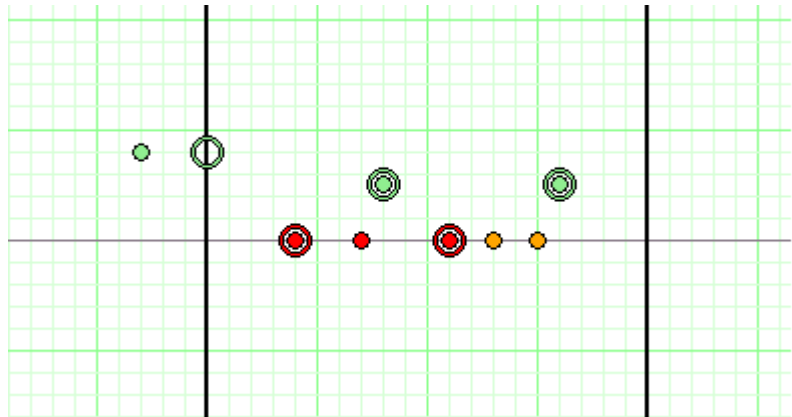
The initial NULL value after a non-NULL is always returned. Multiple NULL values are suppressed. The first non-NULL after a NULL is always returned even if it is the same as the previous non-NULL value.

```
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

The following is a graphical representation of the data:



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.

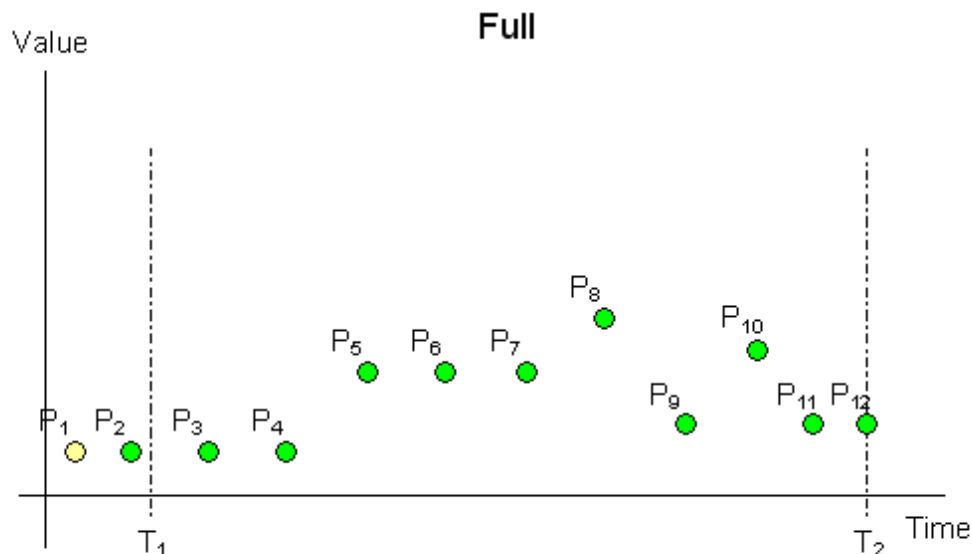
By using full retrieval in conjunction with storage without filtering (that is, no delta or cyclic storage mode is applied at the historian), you can retrieve all values that originated from the plant floor data source or from another application.

Full retrieval best represents the process measurements recorded by the Wonderware 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) on page 769

Full Retrieval - Query Examples

Query 1

```
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 on page 697.

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.

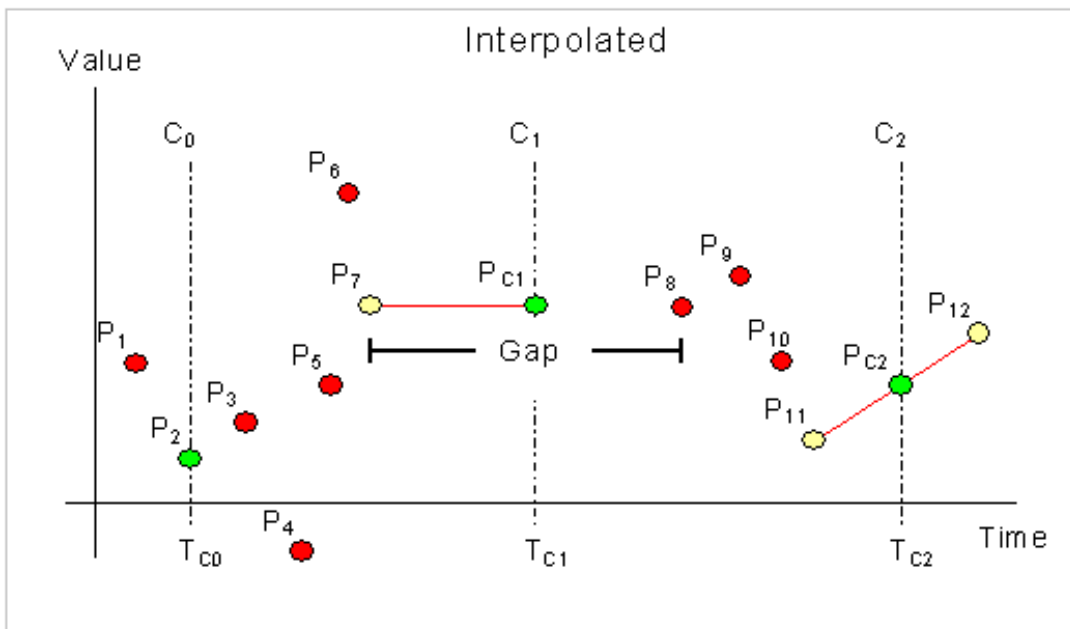
By default, interpolated retrieval uses the interpolation setting specified for the tag in the Wonderware Historian. This means that if a tag is set to use stair-step interpolation, interpolated retrieval gives the same results as cyclic retrieval.

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 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} . 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 , P_{C1} , P_{C2}) 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 P_{C1} is returned, which is the NULL value represented by P_7 shifted forward to time T_{C1} . At the last cycle boundary, point P_{C2} is returned, which has been interpolated using points P_{11} and P_{12} .

Interpolated Retrieval - Supported Value Parameters

You can use various parameters to adjust which values are returned in interpolated retrieval mode. For more information, see the following sections:

- Cycle Count (X Values over Equal Time Intervals) (wwCycleCount) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Interpolation Type (wwInterpolationType) on page 771
- Time stamp Rule (wwTimestampRule) on page 774
- Quality Rule (wwQualityRule) on page 778

Interpolated Retrieval - Query Examples

To use the interpolated mode, set the following parameter in your query.

```
wwRetrievalMode = 'Interpolated'
```

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
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

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
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

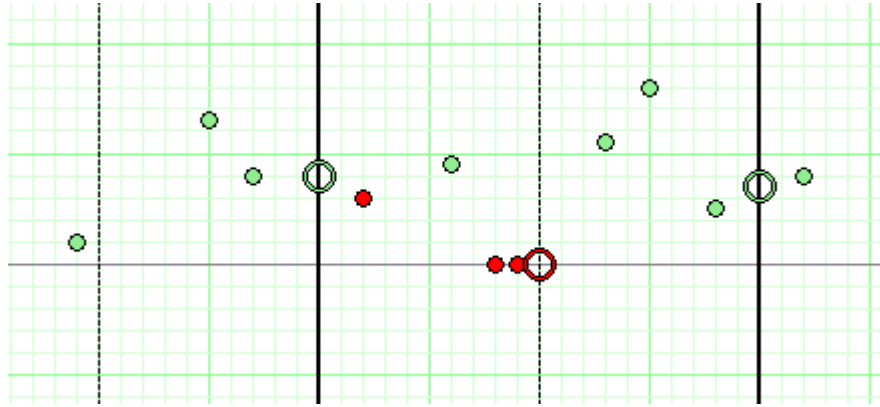
Query 3

```
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:



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.

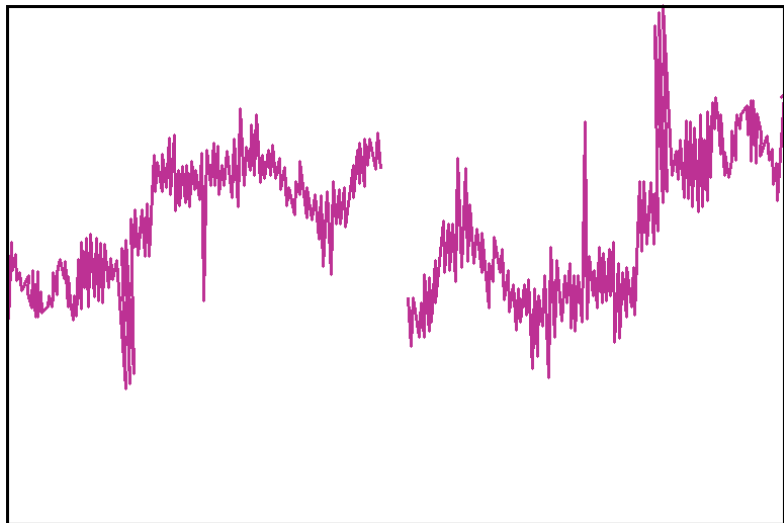
If a valid value precedes a cycle boundary, but is followed by a NULL value after the cycle boundary, no interpolation will be used and wwInterpolationType will be set to STAIRSTEP for that value.

“Best Fit” Retrieval

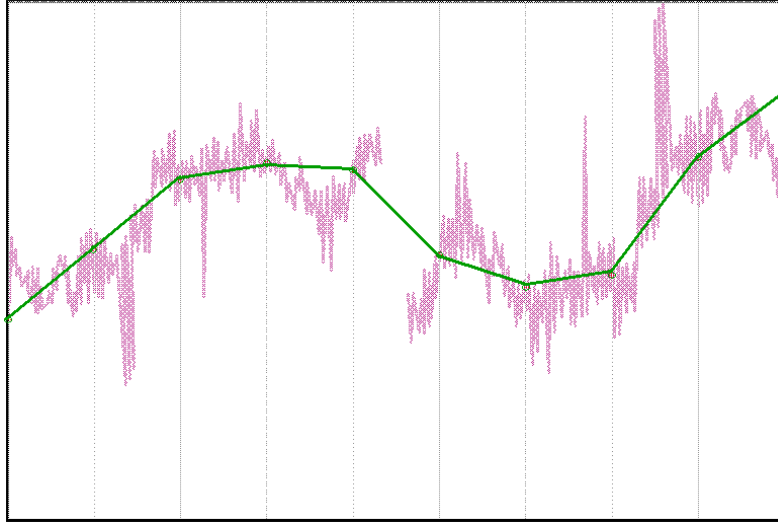
For the “best fit” retrieval mode, the total time for the query is divided into even sub-periods, and then up to five values are returned for each sub-period:

- 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)

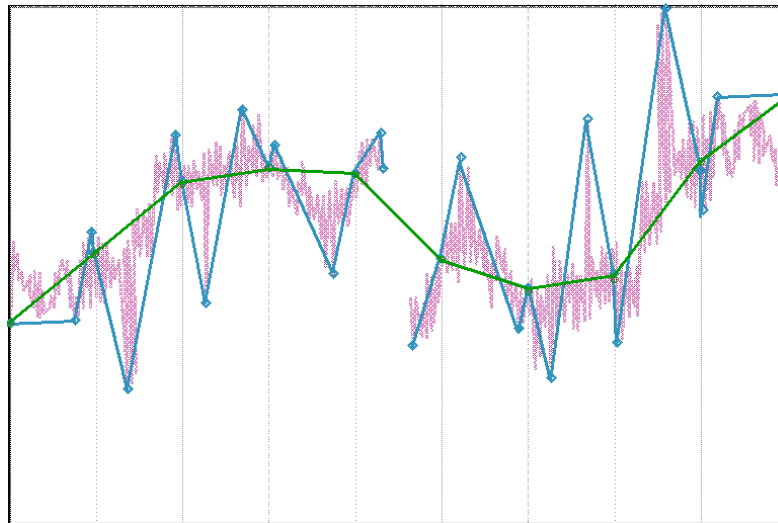
“Best fit” retrieval allows for a compromise between delta retrieval and cyclic retrieval. For example, delta retrieval can accurately represent a process over a long period of time, as shown in the following trend. However, to achieve this representation, a large number of data values must be returned.



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.



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.

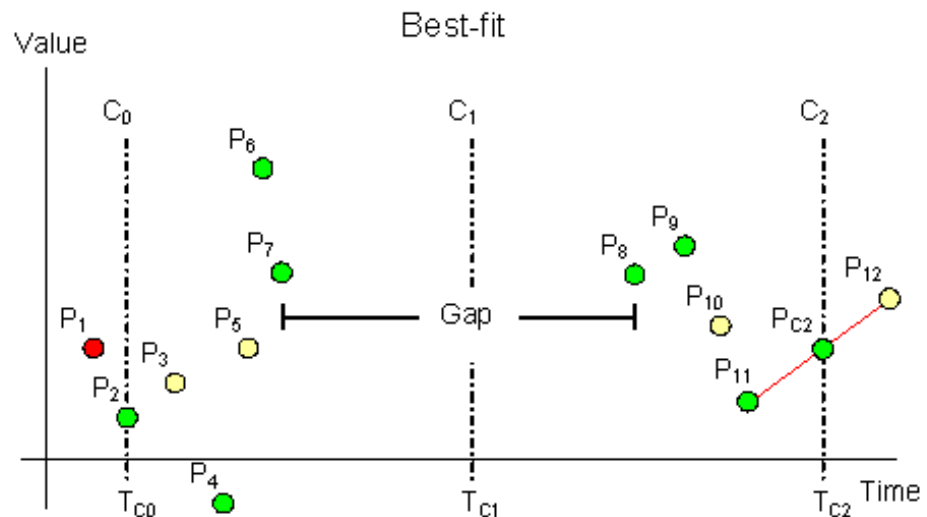
As in cyclic retrieval, the number of cycles is based on the specified resolution or cycle count. However, the number of values returned is likely to be more than one per cycle.

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 .

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`) on page 755
- Resolution (Values Spaced Every X ms) (`wwResolution`) on page 757
- History Version (`wwVersion`) on page 769
- Interpolation Type (`wwInterpolationType`) on page 771
- Quality Rule (`wwQualityRule`) on page 778

Best Fit Retrieval - Query Examples

To use the best fit retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'BestFit'
```

Query 1

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 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
(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

	DateTime	TagName	Value	IT
(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. The values of the initial and final points will be determined by interpolating the points preceding and following the query start or end time.

Standard interpolation rules will be used to return the initial and final values. For more information, see *Interpolated Retrieval* on page 701.

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.)

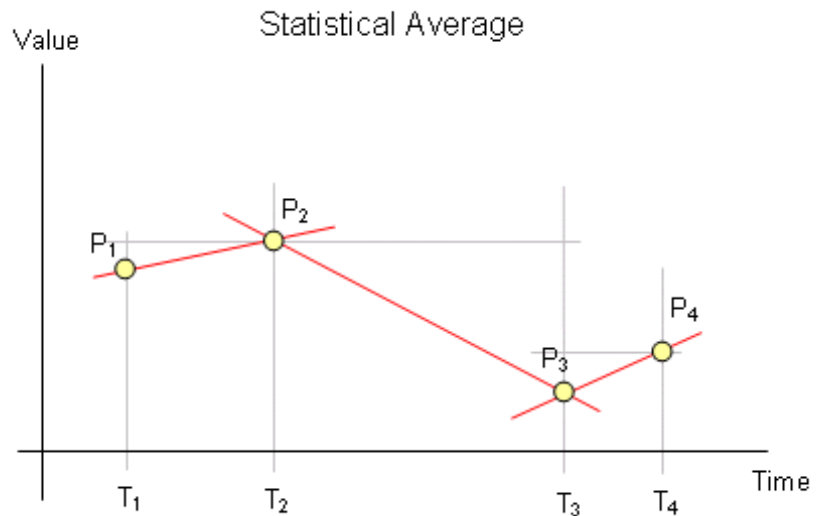
Average Retrieval

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:

$$(P_1 + P_2 + P_3 + P_4) / 4 = \text{Average}$$

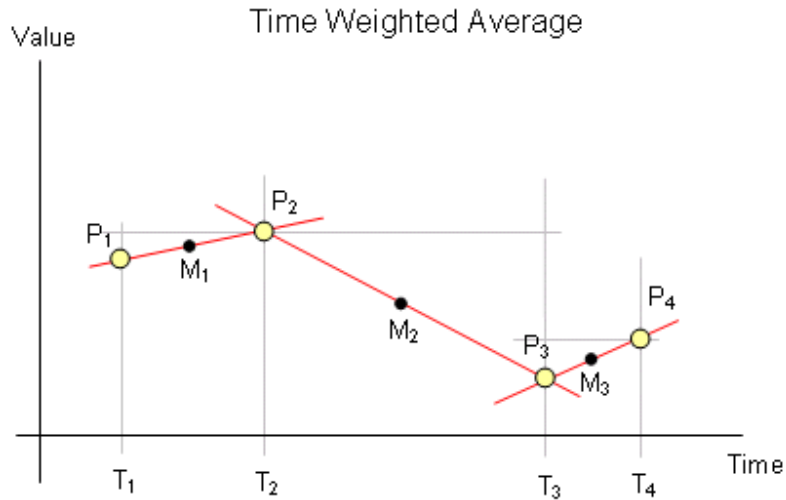


For a time-weighted average, values are multiplied by the time difference between the points to determine the time-weighted value. Therefore, the longer a tag had a particular value, the more weight that value holds in the overall average. The overall average is determined by adding all of the time-weighted 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:

$$\frac{((P_1 + P_2) / 2) \times (T_2 - T_1) + ((P_2 + P_3) / 2) \times (T_3 - T_2) + ((P_3 + P_4) / 2) \times (T_4 - T_3)}{T_4 - T_1} = \text{Average}$$



If the same tag uses stair-step interpolation, the time-weighted average is:

$$\frac{(P_1 \times (T_2 - T_1)) + (P_2 \times (T_3 - T_2)) + (P_3 \times (T_4 - T_3))}{T_4 - T_1} = \text{Average}$$

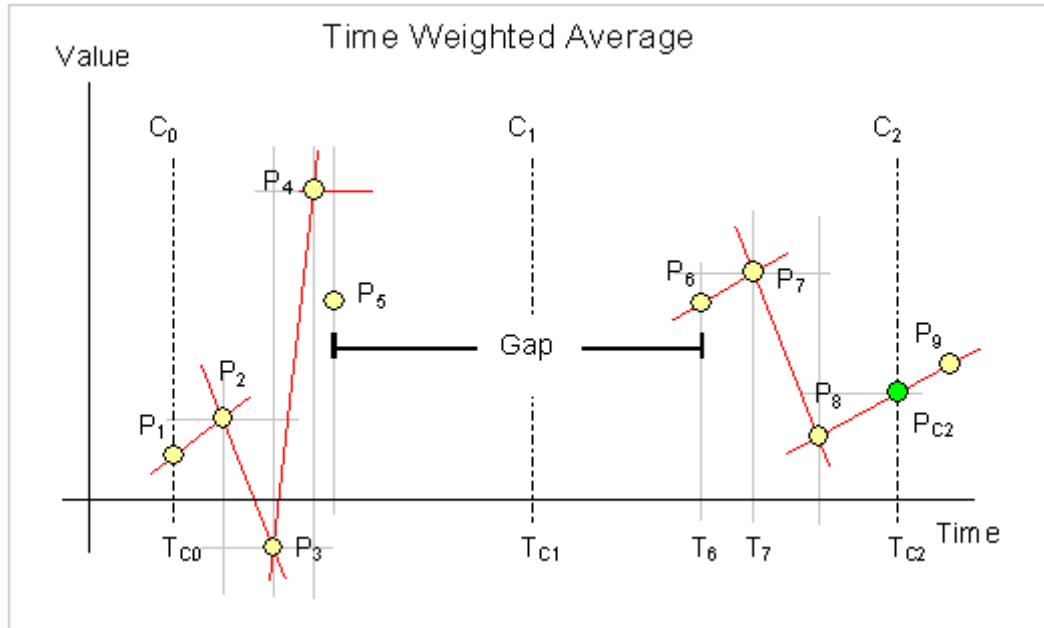
The SQL Server AVG aggregate is a simple statistical average. Using the average retrieval mode with a cycle count of 1 returns data much faster than the AVG aggregate, and usually more accurately due to the time weighting. The event subsystem also returns a simple statistical average.

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:

- 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) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Interpolation Type (wwInterpolationType) on page 771
- Time stamp Rule (wwTimestampRule) on page 774
- Quality Rule (wwQualityRule) on page 778

Average Retrieval - Query Examples

To use the average mode, set the following parameter in your query.

```
wwRetrievalMode = 'Average'
```

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

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

For an additional example, see the section [Querying Aggregate Data in Different Ways](#) in the *Wonderware Historian Server Concepts Guide*.

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.

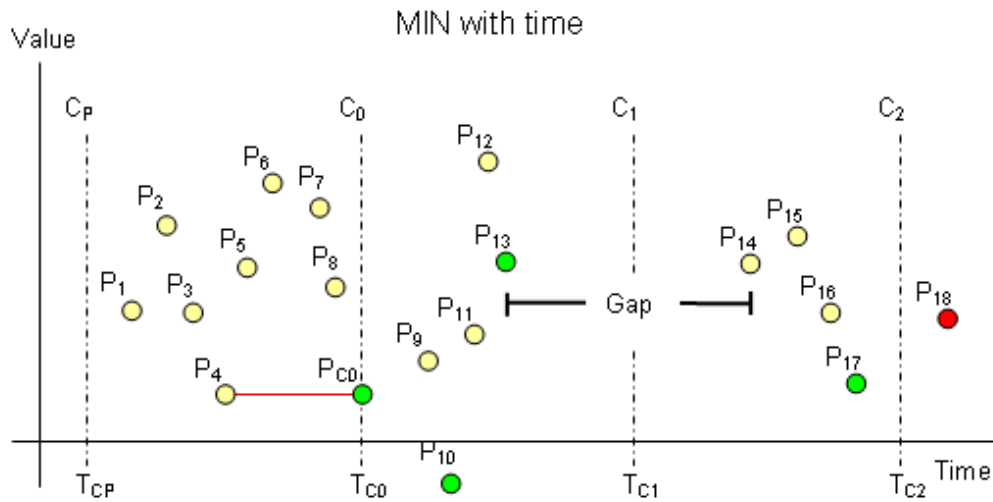
Minimum retrieval only works with analog tags. For all other tags, normal delta results are returned.

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.

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.

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) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Quality Rule (wwQualityRule) on page 778

Minimum Retrieval - Query Examples

To use the minimum mode, set the following parameter in your query:

```
wwRetrievalMode = 'Min'
```

or

```
wwRetrievalMode = 'Minimum'
```

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

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

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

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](#) on page 697.)

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.

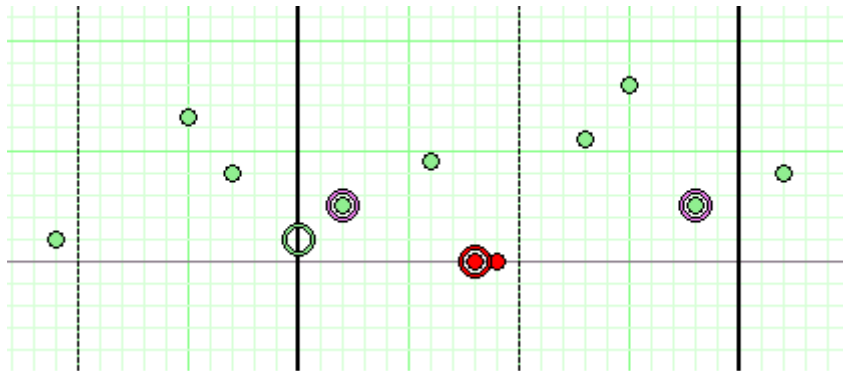
As an example of how minimum retrieval mode handles NULLs, consider the following query:

```
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
A001	2009-09-12 00:42	0.8	192

The following is a graphical representation of the data:



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.

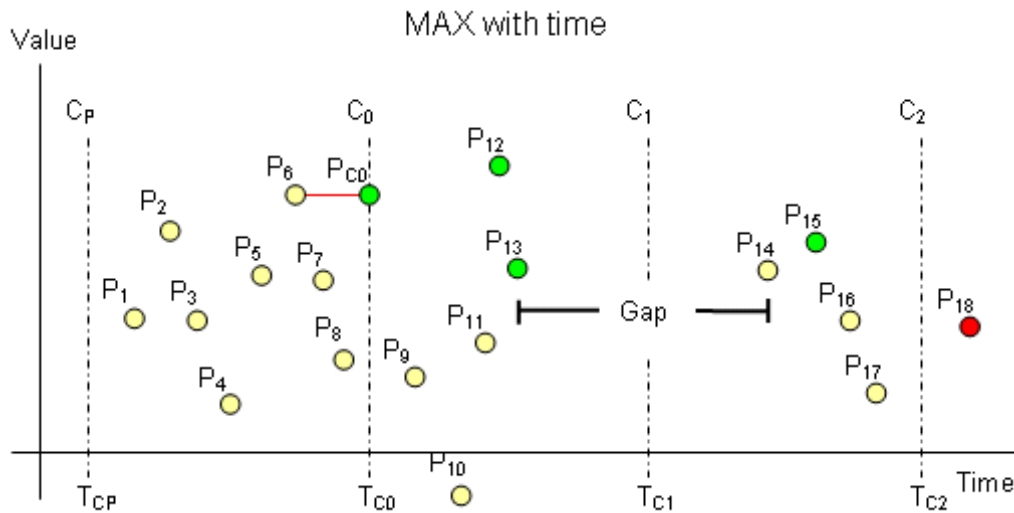
Maximum retrieval only works with analog tags. For all other tags, normal delta results are returned.

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.

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

- P₁₃ as the first and only exception occurring in the first cycle
- P₁₅ 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) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Quality Rule (wwQualityRule) on page 778

Maximum Retrieval - Query Examples

To use the maximum mode, set the following parameter in your query:

```
wwRetrievalMode = 'Max'
```

or

```
wwRetrievalMode = 'Maximum'
```

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

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

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

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 the section Determining Cycle Boundaries in the *Wonderware Historian Server Concepts Guide*.)

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

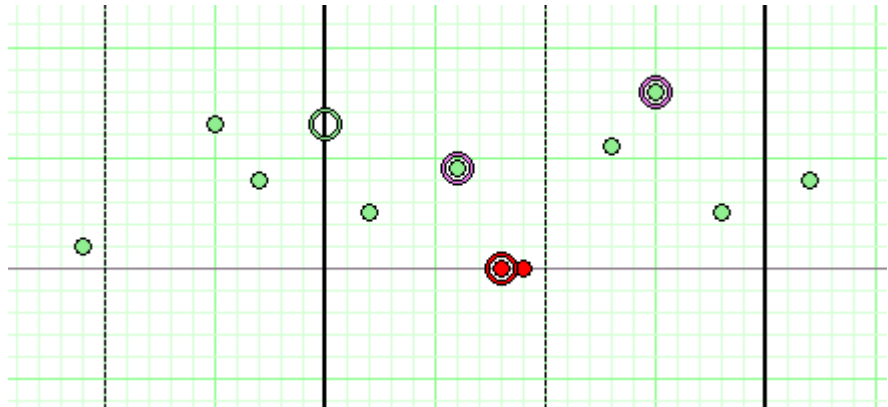
The results are:

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.



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 on page 712.
- 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) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Interpolation Type (wwInterpolationType) on page 771
- Time stamp Rule (wwTimestampRule) on page 774
- Quality Rule (wwQualityRule) on page 778

Integral Retrieval - Query Examples

To use the integral retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Integral'
```

Query 1

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

	DateTime	TagName	Flow	OPCQuality	PercentGood
(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 the section *Querying Aggregate Data in Different Ways* in the *Wonderware Historian Server Concepts Guide*.

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.

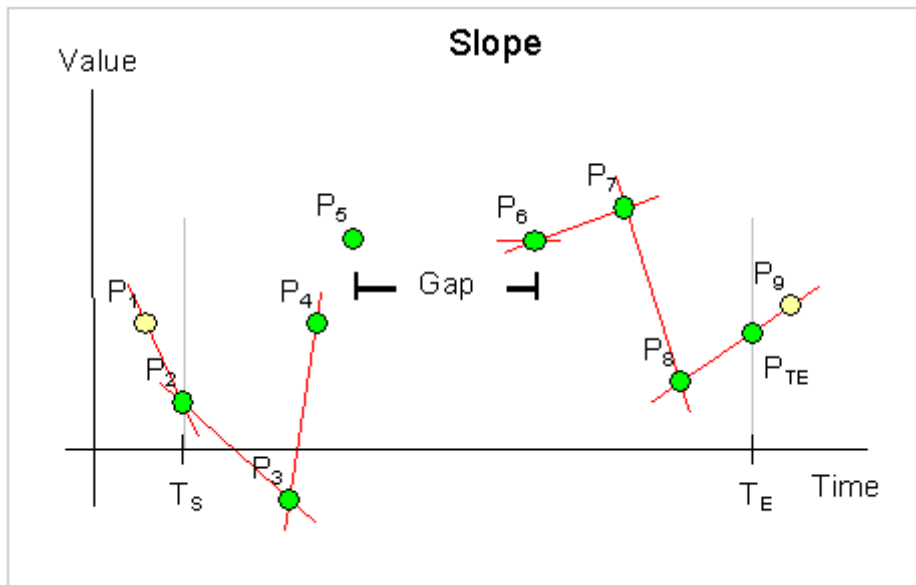
The slope retrieval mode can be considered a delta mode. Apart from the initial value and a value at the query end time, all returned points are calculated delta points returned with the timestamp of an actual delta point.

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.



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) on page 769
- Quality Rule (wwQualityRule) on page 778

Slope Retrieval - Query Example

To use the slope retrieval mode, set the following parameter in your query.

```
wwRetrievalMode = 'Slope'
```

Query 1

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 the section Determining Cycle Boundaries in the *Wonderware Historian Server Concepts Guide*.

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
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 - 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.

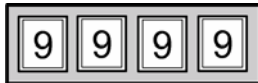
Slope Retrieval - Handling NULL Values

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 Wonderware 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.

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.

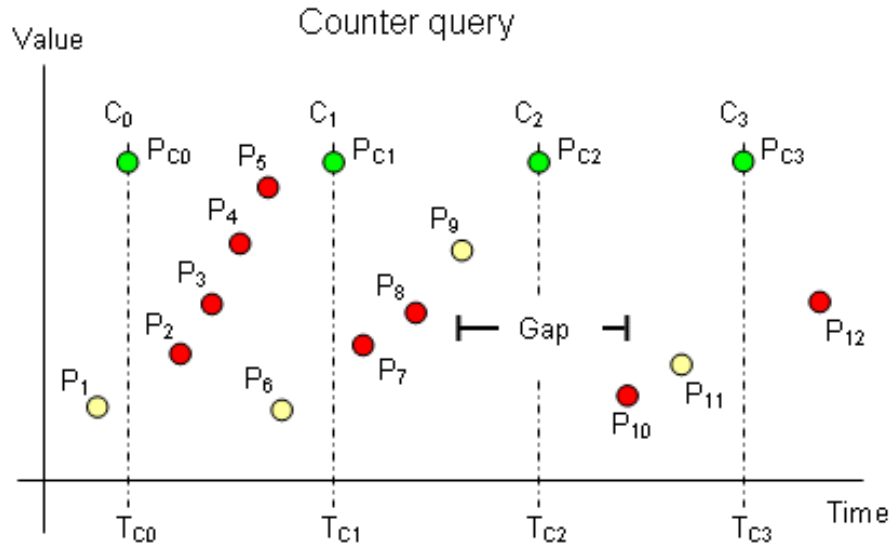
Counter retrieval is a true cyclic mode. It returns one row for each tag in the query for each cycle. The number of cycles is based on the specified resolution or cycle count.

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 \(wwQualityRule\)](#) on page 778. When a rollover has occurred in the calculation cycle, a special quality detail of 212 is returned in all non-NULL cases.

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:

$$P_{C1} = n * V_R + P_6 - P_1$$

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.

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, and 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 - 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) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Time stamp Rule (wwTimestampRule) on page 774
- Quality Rule (wwQualityRule) on page 778

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.

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 two-minute time span, the SysTimeSec tag increments from 0 to 59 two times. The following query returns the total count within the two-minute time span. The QualityDetail of 212 indicates that a counter rollover occurred during the query time range.

```
select DateTime, TagName, Value, Quality, QualityDetail
as QD from History
  where TagName = 'systimesec'
        and DateTime >= '2009-08-13 1:00'
        and DateTime < '2009-08-13 1:02'
        and wwRetrievalMode = 'counter'
        and wwCycleCount = 1
```

The results are:

DateTime	TagName	Value	Quality	QD
2009-08-13 01:00:00.0000000	SysTimeSec	120	0	212

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 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 display.

If you *do not* 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.

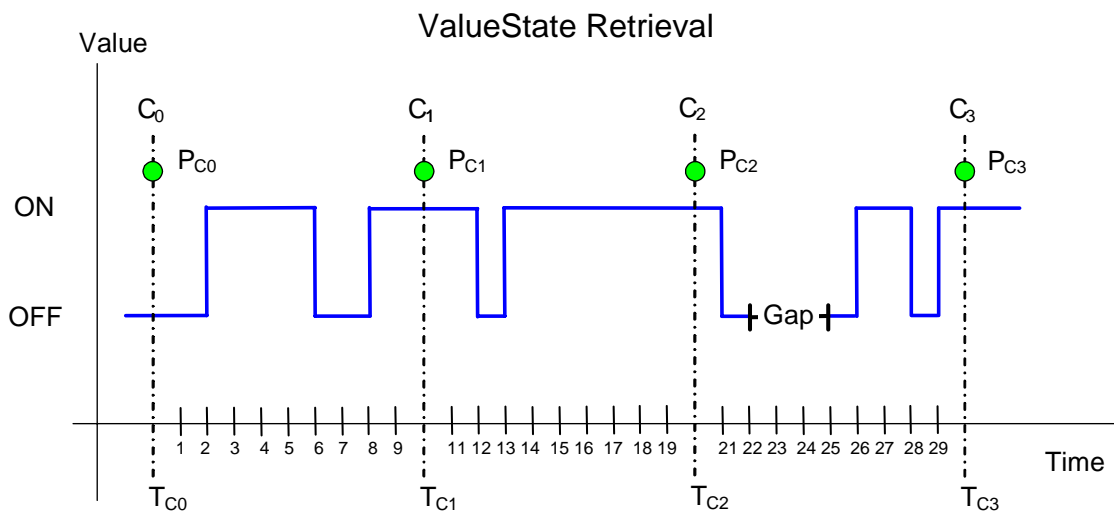
ValueState retrieval works with integer, discrete, string, and state summary tags. For other types of tags, no rows are returned. NULL values are treated like any other distinct state.

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`) on page 786.

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 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 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 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) on page 755
- Resolution (Values Spaced Every X ms) (wwResolution) on page 757
- History Version (wwVersion) on page 769
- Time stamp Rule (wwTimestampRule) on page 774
- Quality Rule (wwQualityRule) on page 778
- State Calculation (wwStateCalc) on page 786

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:

```
wwStateCalc = 'Total'
```

Be sure that you use the "<=" operator for ending date/time.

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

Query 2: Minimum Time in State for Single Tag

The following query finds the minimum time-in-state for the SteamValve 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 ('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'
      AND State = '1'
```

The results are:

DateTime	TagName	vValue	StateTime	wwStateCalc
2005-04-17 10:00:00.000	SteamValve	1	43749.0	MINIMUM
2005-04-17 10:05:00.000	SteamValve	1	43749.0	MINIMUM

Query 2

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'

```

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
2005-04-17 10:05:00.000	SteamValve	1	43750.0	MAXIMUM

Query 3

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

Query 4

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:

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

Query 5: 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.

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 rows 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.

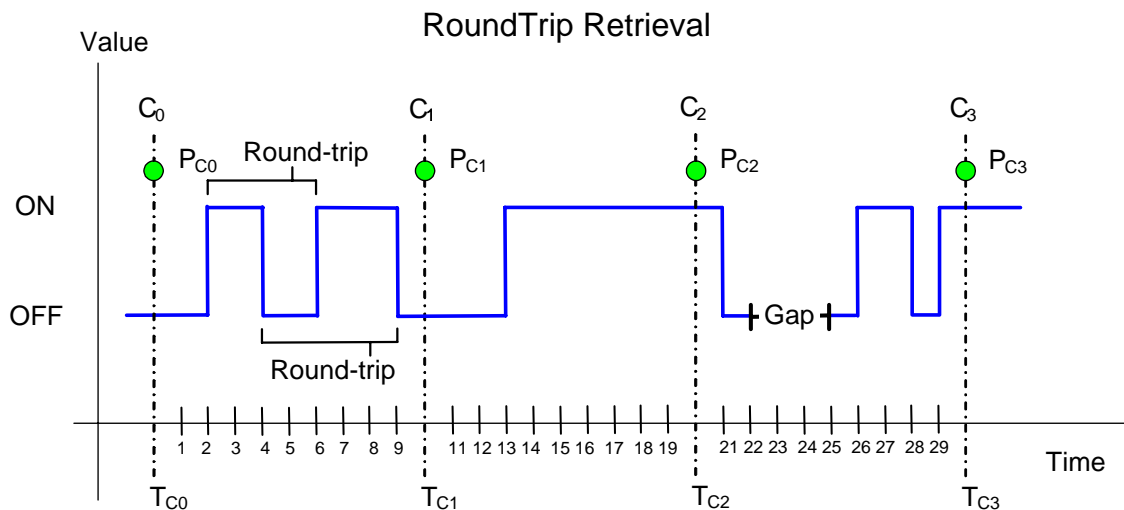
RoundTrip retrieval is supported for the History and StateWideHistory tables.

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 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 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 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 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 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 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:

- Time stamp Rule (wwTimestampRule) on page 774
- Quality Rule (wwQualityRule) on page 778
- State Calculation (wwStateCalc) on page 786

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 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 single cycle. Any state changes that occur across the cycle boundaries are not included.

```
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
```

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

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.

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? on page 753.

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.

	Cycle Count (X Values over Equal Time Intervals)	Resolution (Values Spaced Every X ms) (wwResolution)	Time Deadband (wwTimeDeadband)	Value Deadband (wwValueDeadband)	History Version (wwVersion)	Interpolation Type (wwInterpolationType)	Time stamp Rule (wwTimestampRule)	Quality Rule (wwQualityRule)	State Calculation (wwStateCalc)
Cyclic Retrieval	•	•			•		•*		
Delta Retrieval			•	•	•				
Full Retrieval					•				
Interpolated Retrieval	•	•			•	•	•	•	
"Best Fit" Retrieval	•	•			•	•		•	
Average Retrieval	•	•			•	•	•	•	
Minimum Retrieval	•	•			•			•	
Maximum Retrieval	•	•			•			•	
Integral Retrieval	•	•			•	•	•	•	
Slope Retrieval					•			•	
Counter Retrieval	•	•			•		•	•	
ValueState Retrieval	•	•			•		•	•	•
RoundTrip Retrieval	•	•			•		•	•	•

* (only on IndustrialSQL Server 9.0 and later)

Using Retrieval Options in a Transact-SQL Statement

You can retrieve data in the Wonderware Historian extension tables using normal Transact-SQL code, as well as the specialized SQL time domain extensions provided by the Wonderware Historian. The Wonderware 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 *Wonderware Historian OLE DB Provider Unsupported Syntax and Limitations in the Wonderware Historian Server Concepts Guide*.

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.

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), 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

The following queries demonstrate the cycle count behavior if applied to single tag or to multiple tags in the same query.

Query Using 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

Query 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:

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:

$$\text{number of cycles} = \text{time period} / \text{resolution}$$

The number of actual return values is not always identical with this cycle count.

In “truly cyclic” modes (Cyclic, Interpolated, Average, and Integral), 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 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) on page 755.

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) on page 762.

Resolution - Query Examples

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
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 is common to expect 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 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 returned an additional row at the query end time.

```
SELECT DateTime, Value, Quality, QualityDetail,
       OPCQuality
FROM History
WHERE TagName IN ('SysTimeSec')
      AND DateTime >= '2009-10-16 08:00:00'
      AND DateTime < '2009-10-16 09:00:00'
      AND wwRetrievalMode = 'Avg'
      AND wwResolution = 3600000
```

The results are:

DateTime	Value	Quality	QualityDetail	OPCQuality
2009-10-16 08:00:00.0000000	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 actually not required because the resolution has been 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 as QD,
       OPCQuality
FROM History
WHERE TagName IN ('SysTimeSec')
      AND DateTime >= '2007-12-11 08:00:00'
      AND DateTime <= '2007-12-11 09:00:00'
      AND wwRetrievalMode = 'Avg'
      AND wwCycleCount = 1
```

The results are:

DateTime	Value	Quality	QD	OPCQuality
2009-10-16 08:00:00.0000000	29.5	0	192	192

This second example also asks for hourly averages and it also returns only 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.

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`
- `Interpolated`
- `BestFit`
- `Min`
- `Max`
- `Average`
- `Integral`
- `Counter`
- `ValueState`
- `RoundTrip`

In the remaining retrieval modes, the `StartDateTime` column returns the same time stamp as the `DateTime` column.

For an additional example, see the section `Querying Aggregate Data in Different Ways` in the *Wonderware Historian Server Concepts Guide*.

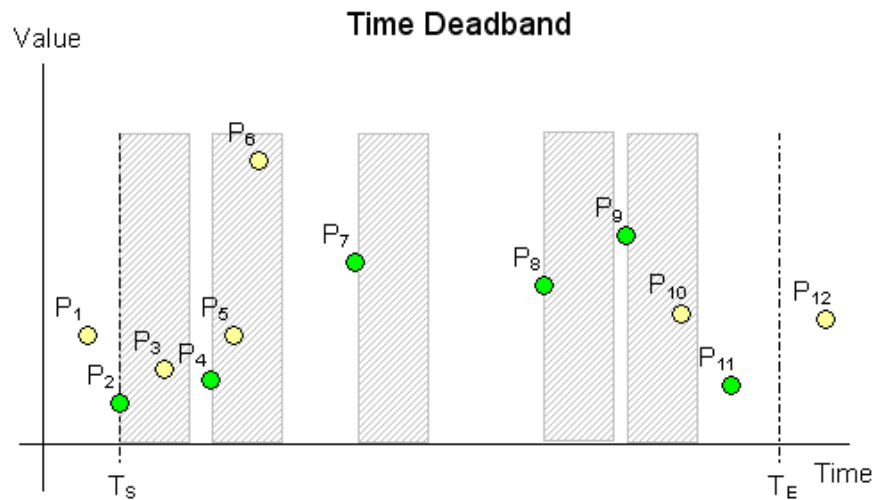
Time Deadband (`wwTimeDeadband`)

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:



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 grey 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.

The following queries return data values for the analog tag 'SysTimeSec'.

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
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

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:

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

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
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

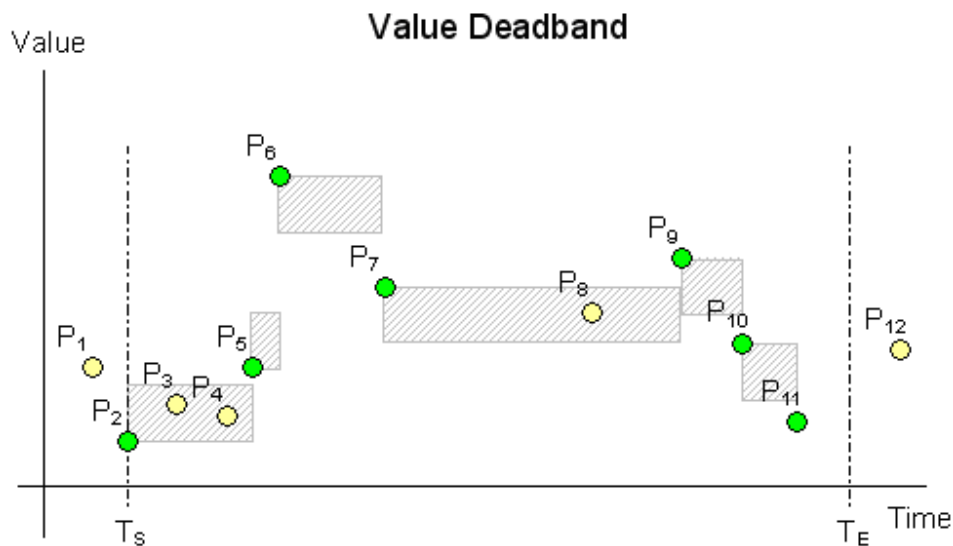
Value Deadband (wwValueDeadband)

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:



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 grey 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

The following queries return data values for the analog tag 'SysTimeSec'. The minimum engineering unit for 'SysTimeSec' is 0, and the maximum engineering unit is 59.

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
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

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:

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
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

History Version (wwVersion)

The Wonderware 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.

This option is relevant in all retrieval modes.

History Version - Query Example

For example:

```
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 00:03:00.000	3	ORIGINAL
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.

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'
```

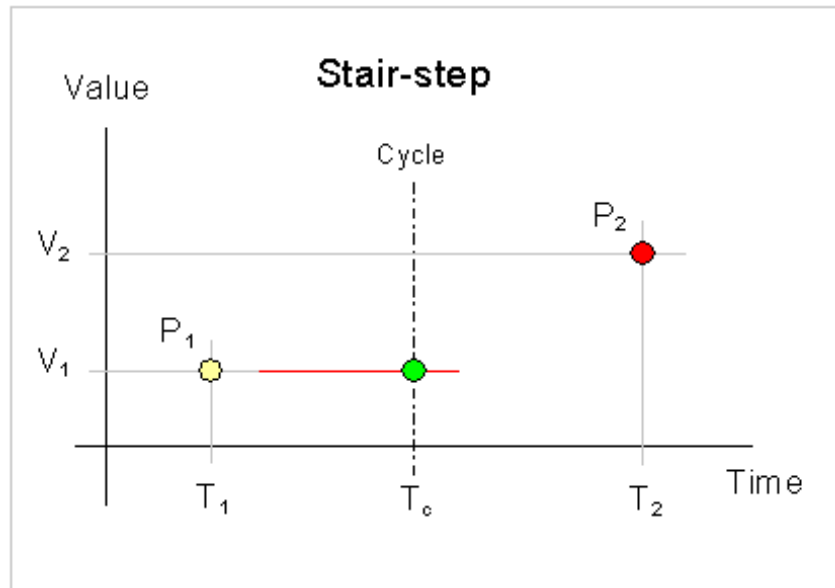
The results are:

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:

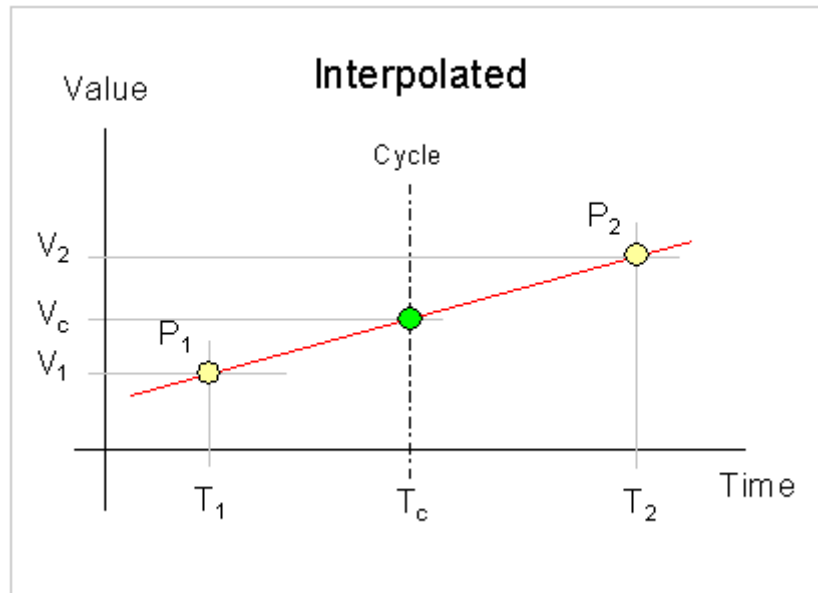
- **Stairstep:** No interpolation occurs. The value at the cycle boundary is assumed to be the same value as the last stored value before the cycle boundary. The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned.



- **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 \(wwQualityRule\)](#) on page 778.

The interpolation type can be set on three levels:

- The Wonderware Historian system-wide setting. The system-wide setting must be either stair-step or interpolated. For more information, see the chapter *System Parameters* in the *Wonderware Historian Server Concepts Guide*. This setting is configured using the Wonderware 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 Wonderware 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'
```

Time stamp 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:

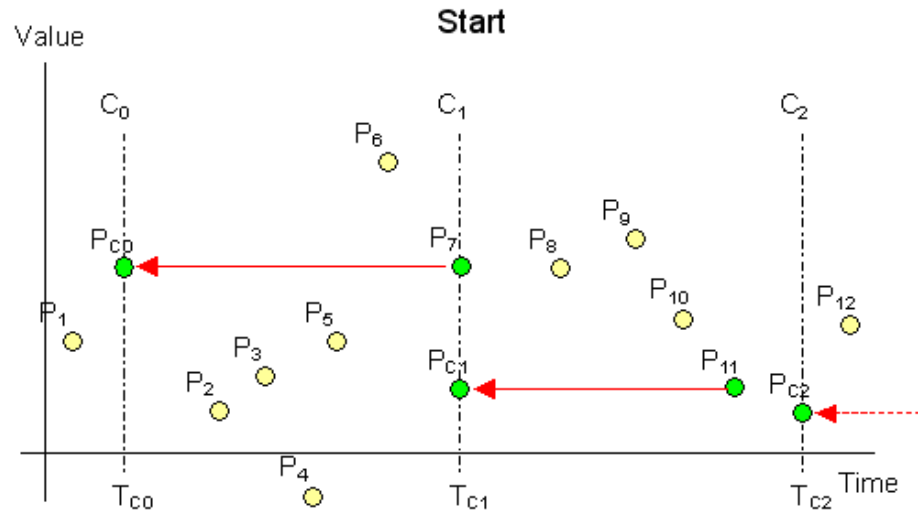
```
AND wwTimeStampRule = 'End'
```

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:

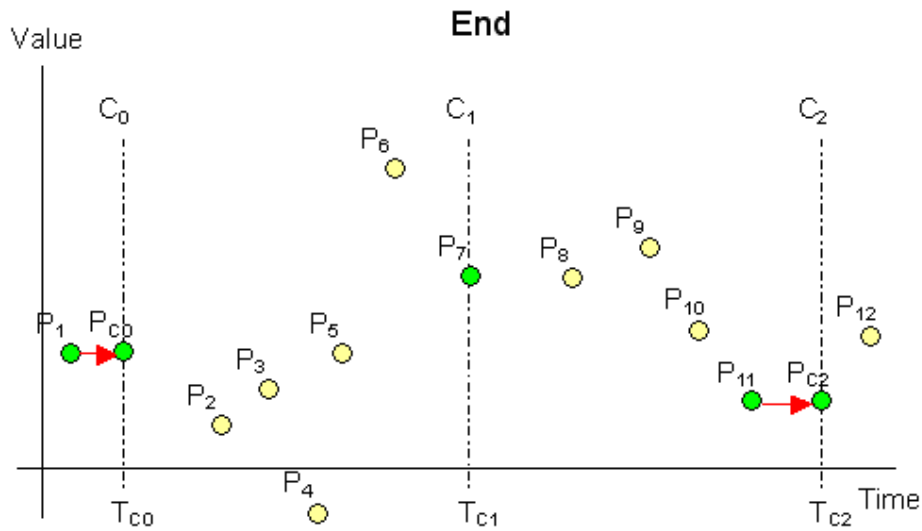
- **Start:** The value for a given cycle is stamped with the cycle start time. For example, in the following illustration of a cyclic query, the following values are returned at the cycle boundaries:
 - At T_{C0} : P_7 , because it falls on the cycle boundary. In cyclic mode, if there is a value right 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}



- **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 Wonderware Historian.

This option is relevant in the following retrieval modes:

- Cyclic Retrieval (only for IndustrialSQL Server 9.0 and later)
- Interpolated Retrieval
- Average Retrieval
- Integral Retrieval
- Counter Retrieval
- ValueState Retrieval
- RoundTrip Retrieval

Time Zone (wwTimeZone)

For Wonderware Historian version 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. In addition to specifying the name of the timezone in the wwTimeZone parameter, you can also specify the TimeZoneID (as a string). For example, on a typical US English system, specifying " wwTimeZone = 'Mountain Standard Time' " and " wwTimeZone = '64' " yields the same result.

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 sub-system 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 Wonderware 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
SysTimeMin	2001-12-20 00:05:00.000	5	W. Europe Standard Time

If you are using date/time functions and the `wwTimeZone` parameter, you will need to use the **`faatZgetdate()`** function.

Quality Rule (`wwQualityRule`)

For various retrieval modes, you can explicitly exclude values with uncertain quality from data retrieval in modes that calculate return values.

Where applicable, 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`.

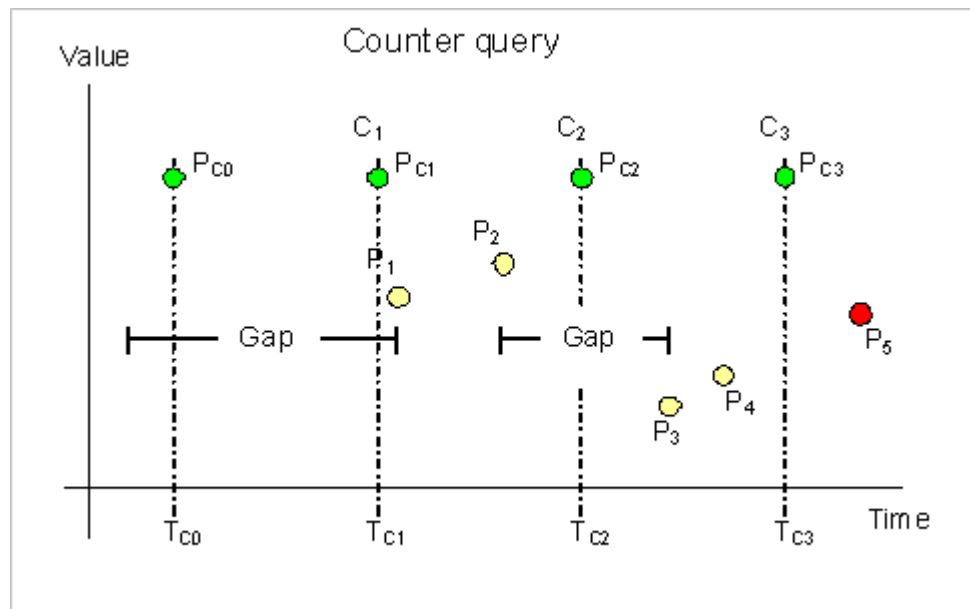
- A quality rule of `GOOD` means that data values with uncertain (64) OPC quality are not used in the retrieval calculations and are ignored. Values with bad `QualityDetail` indicate gaps in the data.
- A quality rule of `EXTENDED` means that data values with both good and uncertain OPC quality are used 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 in the counter and integral retrieval modes 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 in both retrieval modes. The integral retrieval mode is an exception to this where the integral is scaled up to cover gaps. For more information, see Integral Retrieval on page 731.

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.



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, P_1 and P_2 . The query uses P_2 as the end value of the cycle and infers a start value of the cycle from P_1 . At the third cycle boundary, T_{c2} , the query returns $P_2 - P_1$. Similarly, at the last cycle boundary, the query returns $P_4 - P_3$.

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.

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'
```

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 = 'IOR5'  
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
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

Similarly, 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
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

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 Time	EndDate Time	OPCQuality	PercentGood	wwRetrievalMode	first
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

```
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
F0R5	2009-09-12 00:40	NULL	65536	0	AVERAGE

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

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

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 role over 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

For Interpolated mode only the returned row with Linear wwInterpolationType will have combined qualities.

```
SELECT TagName, DateTime, Value, QualityDetail,
       OPCQuality, wwRetrievalMode, wwInterpolationType
FROM History
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

Tagname	DateTime	Resolution	QualityDetail
IOR5	2009-09-12 00:27	0	200
IOR5	2009-09-12 00:29	2	207
IOR5	2009-09-12 00:33	3	215
IOR5	2009-09-12 00:36	0	216
IOR5	2009-09-12 00:39	1	219

The results are:

Tagname	DateTime	Value	QualityDetail	OPCQuality
IOR5	2009-09-12 00:20	0.8	192	192
IOR5	2009-09-12 00:30	2.3	192	192
IOR5	2009-09-12 00:40	1.0	192	219

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.
- **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.

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.

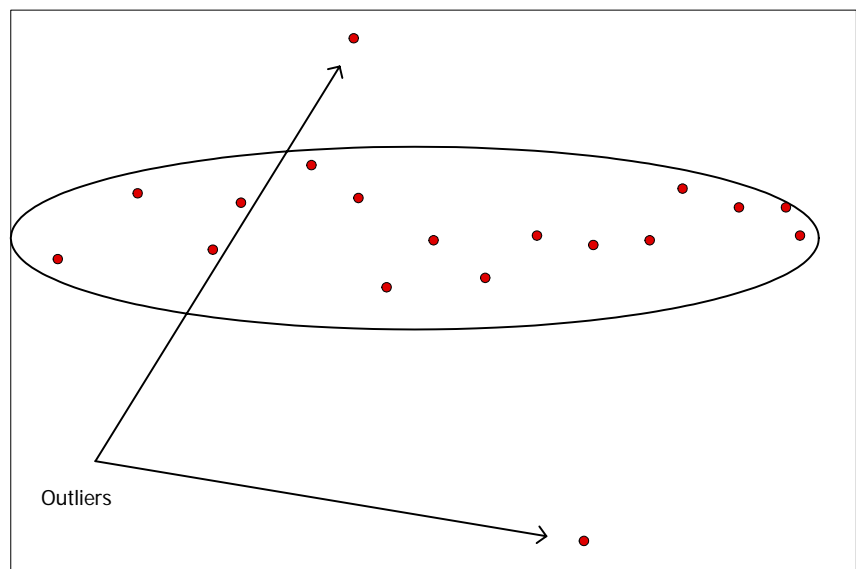
The default value for the `wwFilter` column is 'NoFilter'. If the query does not specify the `wwFilter` element at all, or if its default value is not overridden, then no filter is applied.

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.



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, μ (mu), and time weighted standard deviation, σ (sigma), are found for each analog tag for the entire query range including phantom cycles if any, and points falling outside of the range $[\mu - n\sigma, \mu + n\sigma]$ are removed from the point set before the points are processed further. In other words, the value will be filtered out if value $> \mu + n\sigma$ or value $< \mu - n\sigma$.

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:

$$s^2_{\text{weighted}} = \frac{1}{N} \sum_{i=1}^N w_i (x_i - \mu)^2$$

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'
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
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, <=)
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 \(wwEdgeDetection\)](#) on page 797.

“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(0.1, 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.

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 single summarization period. When you query analog summary data, 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.

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.

Retrieval Mode	Analog Summary Behavior
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 cycle is considered to have a NULL.
Average	<p>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.</p> <p>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.</p> <p>If the QualityDetail of any analog summary considered for a cycle is uncertain (64), the resulting quality is set to 64.</p>
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	<p>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.</p> <p>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.</p> <p>If the QualityDetail of any analog summary considered for a cycle is uncertain (64), the resulting quality will be set to 64.</p>
Slope	The last value within the summary period is used. The actual timestamp of the last value occurrence within the summary period is used.

Retrieval Mode	Analog Summary Behavior
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.

Edge Detection for Events (wwEdgeDetection)

An event is the moment at which a detection criterion is met on historical tag values in the Wonderware 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.

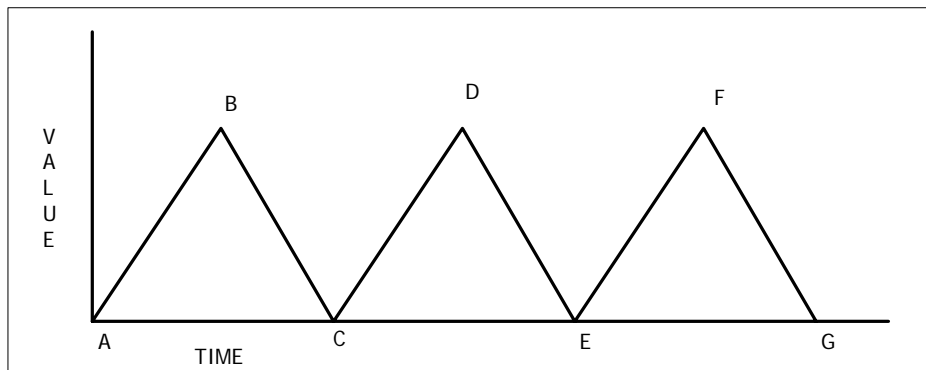
Edge detection only applies to analog and discrete value detectors. Also, edge detection is handled slightly differently based on whether you are using analog tags or discrete tags.

For more information on the event system, see the chapter on Event Subsystem in the *Wonderware Historian Server Concepts Guide*.

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)* on page 790.

Edge Detection for Analog Tags

For example, the behavior of the `WHERE` clause as it processes a result set can be illustrated as:



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
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 on page 798. 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

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 on page 798. 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
2001-12-02 10:01:00.000	0
2001-12-02 10:01:50.000	50
2001-12-02 10:02:00.000	0

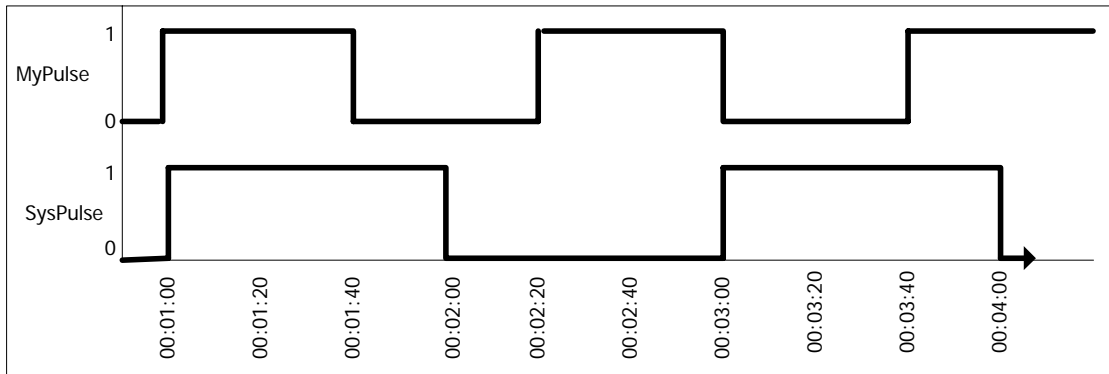
Compare these results with the same query using no edge detection, as shown in Edge Detection for Analog Tags on page 798. Notice that value of the first row in the original query is returned in the result set.

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.

Query 1

Query for *History*.

```

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
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

Query 2

Query for *WideHistory*.

```

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.

Query 1

For a query on 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
2001-12-08 00:03:40.000	MyPulse	1

Query 2

For a query on 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 on page 802. 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).

Query 1

For a query on the *History* table, if the WHERE clause criteria specifies to return 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
2001-12-08 00:03:00.000	MyPulse	1
2001-12-08 00:04:00.000	SysPulse	1

Query 2

For a query on the *WideHistory* table, applying 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 on page 802. 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.

Query 1

```
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
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

Query 2

```
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 on page 802.

Appendix C

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 Wonderware 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](#) on page 124 and [Configuring Trend Options for a Tag](#) on page 67.

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](#) on page 818. Alternatively, you can define your own retrieval styles to suit your needs. For more information, see [Location and Structure of Retrieval Styles](#) on page 810 and [Creating and Editing Retrieval Styles](#) on page 812.

Location and Structure of Retrieval Styles

Retrieval styles are stored at the application level in the following XML file:

```
C:\<Documents and Settings>\All Users\Application
Data\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.

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` on page 813.
- 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` on page 814.
- 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` on page 815.
- 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` on page 816.

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>
  <duration minSpan="POYOM1DT0H0M0S">
    <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="POYOM0DT0H0M0S">
    <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* on page 810 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` on page 814.
- 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` on page 815.
- 4 For each `duration` element, add `retrieval` elements as needed to define retrieval types. For more information, see `retrieval XML Element` on page 816.

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` on page 810.

`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`.

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 Wonderware Historian version that the retrieval style can work with, either `8.0` or `9.0`. If the Trend application is connected to a Wonderware 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 IndustrialSQL Server 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 Wonderware 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” 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.

Note 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.

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 Wonderware Historian does not return any data.

For information on each mode, see [Understanding Retrieval Modes](#) on page 689.

- `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 \(wwStateCalc\)](#) on page 786. 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 \(Values Spaced Every X ms\) \(wwResolution\)](#) on page 757.

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 Wonderware 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 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.

Style name	Description
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 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 Wonderware 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 Wonderware Historians with a version earlier than 9.0. For more information, see *Configuring Other Options* on page 131. 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.

Appendix D

Control and Object Migration Reference

This appendix provides a comparison between the main controls and objects used in different versions of the Wonderware Historian Client software. Migration notes are also included.

Wonderware Historian Client Version Comparison: Overview

The following table provides a comparison between the main controls and objects used in different versions of the Wonderware Historian Client software.

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 Software	Comments
iTrend	aaHistClientTrend	<p>All of the previous version 8.5 properties, methods, and events are included.</p> <p>Properties, methods, and events are included that either provide additional functionality or provide a more efficient means for the same functionality.</p> <p>For more information, see iTrend and aaHistClientTrend Comparison on page 826.</p>
iQuery	aaHistClientQuery	<p>All of the previous properties, methods and events are included.</p> <p>New properties, methods, and events are included that either provide additional functionality or provide a more efficient means for the same functionality.</p> <p>For more information, see iQuery and aaHistClientQuery Control Comparison on page 831.</p>
TimeBar	aaHistClientTimeRangePicker	<p>Slight changes.</p> <p>For more information, see TimeBar and aaHistClientTimeRange Picker Comparison on page 832.</p>

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 Software	Comments
TagPicker	aaHistClientTagPicker	Slight changes. For more information, see TagPicker and aaHistClientTagPicker Comparison on page 834.
ActiveDataGrid	aaHistClientActiveDataGrid	All previous version 8.5 properties, methods, and events are exactly the same. Some new properties have been added. For more information, see ActiveDataGrid and aaHistClientActiveData Grid Comparison on page 837.
SingleValueEntry	aaHistClientSingleValueEntry	Slight changes. For more information, see SingleValueEntry and aaHistClientSingleValue Entry Comparison on page 836.
Server	aaServer	Slight changes. For more information, see Server and aaServer Object Comparison on page 837.
Servers	aaServers	Slight changes. For more information, see Server and aaServer Object Comparison on page 837.
Tag	aaTag	Slight changes. For more information, see Tag and aaTag Object Comparison on page 840.
Tags	(no equivalent)	Functionality is provided by the aaTag object.

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 Software	Comments
TimeSelector	(no equivalent)	N/A
ServerTester	(no equivalent)	The server configuration is tested when a log on is initiated.
GlobalFunctions	aaHistClientGlobalFunctions	N/A
WorkbookRunner	aaHistClientWorkbookRunner	No changes. Properties, methods, and events are exactly the same.
ReportRunner	aaHistClientReportRunner	No changes. Properties, methods, and events are exactly the same.
ManualDataEntry	(no equivalent)	N/A
dbAlias	(no equivalent)	N/A
ActiveTagBrowser	(no equivalent)	N/A
ActiveTimeSelector	(no equivalent)	N/A

Migrating an ActiveFactory 8.5 Application

To migrate an ActiveFactory 8.5 application in the InTouch HMI software for use with the Wonderware Historian Client 10.0 SP1 controls, you must do the following:

- The controls do not show up immediately in WindowMaker. If you repaint the window, the controls appear.
- Open and save all of the InTouch windows containing the Wonderware Historian Client controls one time in WindowMaker before running the windows in WindowViewer, so that the controls display properly at runtime.
- If you assigned any properties to tag names for a control using the control **Properties** dialog box, you must recreate the assignments.

- For the `ActiveDataGrid` control, if you configured connection settings or SQL statements using the `ActiveDataGrid Properties` dialog box in the ActiveFactory 8.5 software, you must instead configure this information using scripts or tagname assignments. The Wonderware Historian Client 10.0 SP1 software does not provide custom tabs in the `ActiveDataGrid Properties` dialog box for this information.
- The `aaHistClientTimeRangePicker` and `aaHistClientTagPicker` controls do not have the same signatures in their properties, methods, and events as in the ActiveFactory 8.5 software. You must modify your application accordingly.
- When the Wonderware Historian Client 10.0 SP1 client application or control connects to a Wonderware Historian, it validates the presence of required Wonderware Historian Client support in the Runtime database (for example, stored procedures and tables). If the supporting entities are not found, a login dialog box appears so that you can provide an administrative login in order that the required support can be automatically configured. After this initial setup is complete, no further action is required.

If your existing InTouch application uses a Wonderware Historian Client control that only uses a user login (for example, through a script), and the Wonderware Historian Client 10.0 SP1 server-side support has not yet been configured for the historian database, it is possible that no administrative logon prompt appears after you migrate the application to the Wonderware Historian Client 10.0 SP1 version. Therefore, make sure that you have logged into the Wonderware Historian at least one time as administrator using one of the Wonderware Historian Client applications, such as Wonderware Historian Client Trend.

- Existing `.CRV` files cannot be opened with the `aaHistClientTrend` control. You must import any `.CRV` files into the Wonderware Historian Client application and then save them as `.aaTrend` files. For more information, see [Importing .CRV Data](#) on page 164. Then, change the references for the `.CRV` files in the application to the appropriate `aaTrend` files.
- You must change references to `uisupport.globalfunctions` to `ArchestrA.HistClient.Util.aaHistClientGlobalFunctions`.

Migrating Documents to Wonderware Historian Client 10.0 SP1

You can open Trend, Query, Workbook, and Report documents created using ActiveFactory versions 9.0, 9.1, 9.2, and Wonderware Historian Client versions 9.5 and 10.0. These documents are converted and saved to the format used by Wonderware Historian Client 10.0 SP1.

Note ActiveX and .NET applications built using ActiveFactory 8.5, 9.0, 9.1, 9.2, and Wonderware Historian Client 9.5, and 10.0 controls continue to operate without changes to their original functionality.

iTrend and aaHistClientTrend Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 iTrend control and the aaHistClientTrend control. The tables only list differences related to functionality present in version 8.5. Unless otherwise noted, properties, methods and events work the same way as before.

You must reconfigure the servers for the aaHistClientTrend control; the aaHistClientTrend control does not automatically use the servers as specified by the iTrend control.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	ApplyRubberBandToAllTags	This is the same as the RubberBandAll property in version 8.5.
--	BackGradient	
--	BackGradientEndColor	
--	BackImage	
--	BorderStyle	
--	BorderWidth	

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
CyclicRows	CyclicRows	<p>When retrieving data from a IndustrialSQL Server 8.0, a fixed algorithm controls how many rows are returned. It is the width of the plot area of the chart.</p> <p>For a IndustrialSQL Server 9.0 or later, use retrieval styles instead.</p>
DatePickerFormatString	DatePickerFormatString	<p>Can also be configured using a method in the aaHistClientTimeRangePicker control.</p>
EnableDeltaRetrieval	EnableDeltaRetrieval	<p>Only relevant when retrieving data from a IndustrialSQLIndustrialSQL Server 8.0. For a IndustrialSQL Server 9.0 or later, use retrieval styles instead.</p>
EnableSummaryData	EnableSummaryData	<p>Present, but ignored. Use retrieval styles instead.</p>
--	HistorySource	

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	LiveModeRate	Provides the same functionality as RealTimeRate, except this property is in milliseconds.
LoginTimeout	LoginTimeout	You can also use the Server object, or Servers object.
MaxDeltaSamples	MaxDeltaSamples	Only relevant when retrieving data from a IndustrialSQL Server 8.0. For a IndustrialSQL Server 9.0 or later, use retrieval styles instead.
MaxMinutesForDeltaAnalog	MaxMinutesForDeltaAnalog	Only relevant when retrieving data from a IndustrialSQL Server 8.0. For a IndustrialSQL Server 9.0 or later, use retrieval styles instead.
MaxMinutesForDeltaDiscrete	MaxMinutesForDeltaDiscrete	Only relevant when retrieving data from a IndustrialSQL Server 8.0. For a IndustrialSQL Server 9.0 or later, use retrieval styles instead.
--	MaxSamplesPerTag	

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
MovingAverageMode	MovingAverageMode	Present, but ignored. Use retrieval styles instead.
MovingAverageSamples	MovingAverageSamples	Present, but ignored. Use retrieval styles instead.
--	PlotColor	
--	PlotGradient	
--	PlotGradientEndColor	
--	PlotImage	
PrintShowMarkers	PrintShowMarkers	Not applicable to version 9.x.
QueryTimeout	QueryTimeout	You can also use the Server object or Servers object.
RetrieveExtensionData	RetrieveExtensionData	You can also use HistorySource.
RetrieveManualData	RetrieveManualData	You can also use HistorySource.
RTRate	RTRate	Use RealTimeRate instead.
RubberBand	RubberBand	You can use RubberBandScaling instead.
RubberBandAll	RubberBandAll	You can use ApplyRubberBandToAllTags instead.
--	RubberBandScaling	Same as RubberBand.
--	Servers	

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
SummaryDataMode	SummaryDataMode	Present, but ignored. Use retrieval styles instead.
TagGridOrientation	TagGridOrientation	Not applicable in 9.x.
TimeBarVisible2	TimeBarVisible2	Not applicable in 9.x.
TimeSelector	TimeSelector	Not applicable in 9.x.
--	ToolBarVisible	Same as ToolbarVisible2.
ToolBarVisible2	ToolBarVisible2	You can also use ToolbarVisible.
UseIniFile	UseIniFile	Not applicable in 9.x.
Width	--	
Methods		
ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
AddServer	AddServer	You can also add a server using the Servers object.
GetMenuItemEnabled	GetMenuItemEnabled	Not applicable in 9.x. The menu is automated for enabling/disabling.
LoadCRVString	LoadCRVString	Since data formats are encrypted, this is no longer applicable.
RemoveServer	RemoveServer	You can also use the Servers object.
ScaleAllTagsDlg	ScaleAllTagsDlg	Not applicable in 9.x. The TagList user interface provides this functionality.
ScaleTagDlg	ScaleTagDlg	Not applicable in 9.x. The TagList user interface provides this functionality.

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
SetMenuItemEnabled	SetMenuItemEnabled	Present, but not applicable in 9.x. The menu is automated for enabling/disabling.
Events		
ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
DatesChanged	DatesChanged	This functionality is also available through the TimeRangePicker object.

iQuery and aaHistClientQuery Control Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 iQuery control and the aaHistClientQuery control. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

You must reconfigure the servers for the aaHistClientQuery control; the aaHistClientQuery control does not automatically use the servers as specified by the iQuery control.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	CurrentServer	CurrentServer returns an aaServer object.
FontBold	FontBold	You can use QueryFont instead.
FontCharset	FontCharset	You can use QueryFont instead.
FontItalic	FontItalic	You can use QueryFont instead.
FontName	FontName	You can use QueryFont instead.

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
FontSize	FontSize	You can use QueryFont instead.
--	QueryFont	
--	Servers	
UsePersistedServers	UsePersistedServers	You can also use the Servers object to change the server set.
Methods		
ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
SetDates2	--	
SetDuration2	--	
SetQueryType	SetQueryType	Cannot be used in the InTouch HMI software. Use the SetQueryType2 method instead.
--	SetTimeSpan	Cannot be used in the InTouch HMI software. Use the SetDates and SetDuration methods instead.
ShowLicenseManager	--	Not applicable in version 9.x.
Events		
No changes.		

TimeBar and aaHistClientTimeRangePicker Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 TimeBar control and the aaHistClientTimeRangePicker control. The tables only list differences between the two versions.

Unless otherwise noted, properties, methods, and events work the same way as before.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
AllowModeChange	--	Use the time range picker to set a duration or a custom set of times.
BackColor	--	Ambient.
BackStyle	--	Ambient.
BorderStyle	--	Ambient.
DTFormat	--	Use Format.
--	DurationMS	
Enabled	--	Ambient.
--	EndDate	
--	Format	
ForeColor	--	Ambient.
Servers	--	Not needed.
--	StartDate	
--	TimeDuration	
TimeSelector	--	Use the time range picker to set a duration or a custom set of times.

Methods

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	GetStartAndEndTimes	
Refresh	--	Use RefreshTimes.
--	RefreshTimes	
SetDates	--	Use SetStartAndEndTimes.
SetDuration	--	Use SetStartAndEndTimes.
--	SetStartAndEndTimes	

Events

No changes.

TagPicker and aaHistClientTagPicker Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 TagPicker control and the aaHistClientTagPicker control. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
AllowAddToTarget	--	Drag-and-drop, cut and paste, and so on are suitable alternatives.
AllowUserPaneControl	--	No control allowed.
BackColor	--	Ambient.
BackStyle	--	Ambient.
BorderStyle	--	Ambient.
Enabled	--	Ambient.
Font	--	
ForeColor	--	Ambient.
--	HideCaption	
--	SelectedTagCount	
--	SelectedPath	
SelectedTags	--	Use the SelectedTag method.
SelectionMode	--	The determination of whether a tag is selected or picked is now managed through the OnPicked and OnSelected events.
SingleServerMode	--	Not provided. You can now add as many servers as you choose, as well as control how many servers appear for the control.
TagTypeFilter	--	Not needed. The user interface displays all tag types and includes an all-inclusive All tab.

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
TagTypeFilterEnabled	--	Not needed. The user interface displays all tag types and includes an all-inclusive All tab.
TargetName	--	Drag-and-drop, cut and paste, and so on are suitable alternatives.
Methods		
ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
AddAllTagsToTarget	--	Drag-and-drop, cut and paste, and so on are suitable alternatives.
AddTagsToTarget	--	Drag-and-drop, cut and paste, and so on are suitable alternatives.
--	RefreshTags	
Refresh	--	Use RefreshTags.
ReloadTree	--	Not needed.
--	SelectedTag	
ValidNode	--	Not implemented.
Events		
ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
Click	--	Ambient.
DbClick	--	Ambient.
KeyDown	--	Ambient.
KeyPress	--	Ambient.
KeyUp	--	Ambient.
MouseDown	--	Ambient.
MouseMove	--	Ambient.
MouseUp	--	Ambient.
--	OnFilterChanged	
--	OnGroupChanged	

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	OnSelectedTabChanged	
--	OnServerChanged	
--	OnTagsPicked	
--	OnTagsSelected	
NodeClick		Not designed. There is no need for this behavior.

SingleValueEntry and aaHistClientSingleValueEntry Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 SingleValueEntry object and the Wonderware Historian Client 10.0 SP1 aaHistClientSingleValueEntry object. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
BackColor	--	Ambient.
BackStyle	--	Ambient.
BorderStyle	--	Ambient.
Enabled	--	Ambient.
Font	--	
ForeColor	--	Ambient.

Methods

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
InsertValue	--	

Events

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
Click	--	Ambient.
DblClick	--	Ambient.
KeyDown	--	Ambient.
KeyPress	--	Ambient.
KeyUp	--	Ambient.
MouseDown	--	Ambient.
MouseMove	--	Ambient.

ActiveDataGrid and aaHistClientActiveDataGrid Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 ActiveDataGrid control and the Wonderware Historian Client 10.0 SP1 aaHistClientActiveDataGrid Control.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	Domain	Used for HTTP access.
--	VirtualDirectoryName	Used for HTTP access.
--	Visible	Ambient.

Methods

No changes.

Events

No changes.

Server and aaServer Object Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 Server object and the Wonderware Historian Client 10.0 SP1 aaServer object. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	BaseURLAddress	
--	Build	
--	Domain	
--	LoginID	
--	LoginTimeout	
--	MachineName	
--	Name	
--	Password	
--	PatchLevel	
--	QueryTimeout	
--	RetainPassword	
--	SchemaVersion	
--	State	
--	TrustedConnection	
--	UseHttp	
--	VirtualDirectoryName	

Methods

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	LogOff	
--	LogOn	

Events

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
ServerSatusChanged	--	

Servers and aaServers Object Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 Servers object and the Wonderware Historian Client 10.0 SP1 aaServers object. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
LoginTimeOut	--	Applied on a per-server basis.
QueryTimeOut	--	Applied on a per-server basis.

Methods

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	ApplicationName	
--	GetServer	
GetStatus	--	
LogOn	--	Applied on a per-server basis.
LogOff	--	Applied on a per-server basis.
UnderlyingServerName	--	
--	Update	
UpdateSchemaVersion	--	

Events

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
LicenseFailure		
--	OnServerAdded	
--	OnServerUpdated	
--	OnServerRemoved	
--	OnServerStateChanged	
ServerSatusChanged	--	

Tag and aaTag Object Comparison

This section provides a comparison for the properties, methods, and events between the ActiveFactory 8.5 Tag object and the Wonderware Historian Client 10.0 SP1 aaTag object. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

Properties

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
--	DateCreated	
--	Description	
--	IOAddress	
--	MaxRaw	
--	MinRaw	
--	MinEU	
--	MaxEU	
--	Message0	
--	Message1	
--	Mode	
--	Name	
--	RawType	
--	Server	
ServerName	--	The Server property can be used to get the name.
TagName	--	Use the Name property instead.
TagType	--	Us the TypeAsTagType property instead.
--	Type	
--	TypeAsTagType	
--	Units	

Methods

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
GetBasicInfo	--	Use the version 9.x properties.

Tags Object

The ActiveFactory 8.5 Tags object does not have an equivalent in Wonderware Historian Client 10.0 SP1. In most cases, an array of tags can be used instead. The functionality provided by this object is provided in Wonderware Historian Client 10.0 SP1 software through the SelectedTagCount and SelectedTags properties of the aaHistClientTagPicker control.

GlobalFunctions Object and aaHistClientGlobalFunctions Object Comparison

This section provides a comparison for the methods between the ActiveFactory 8.5 GlobalFunctions object and the Wonderware Historian Client 10.0 SP1 aaHistClientGlobalFunctions object. The tables only list differences between the two versions. Unless otherwise noted, properties, methods, and events work the same way as before.

Methods

ActiveFactory 8.5 Software	Wonderware Historian Client 10.0 SP1 Software	Comments
GetDictionaryPath	GetDictionaryPath	For 9.x, returns an empty string.
GetInstallPath	GetInstallPath	The installation folder for version 9.x does not have the same structure as 8.5.

Glossary

add-in	A software application that increases the capabilities of the larger application. For example, the Wonderware Historian Client Trend and Query add-ins are used to extend the functionality of Microsoft Word and Excel.
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.
analog	An analog value is a variable that measures a continuous physical quantity. For example, the temperature of a boiler would be measured as an analog value.
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.
Archestra	The architecture for open and extensible technology based on a distributed, object-oriented design.
architecture	A system's architecture describes the structure of a computer system, including the hardware and software that link the computers on a network.
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.
attribute name	An attribute name is the name of a variable exposed by an object.
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. <i>See also</i> server.
binding tags	A binding tag consists of a set of tags that you can bind to the report at run time.
block	<i>See</i> history block.
browser	A browser is a graphical representation of hierarchical groups of data. A browser is used to display disk 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. <i>See also</i> namespace.
C	
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. <i>See also</i> 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.
console	<i>See</i> 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."
contained name	The contained name is the name given to an object with considerations to its place within the overall object hierarchy. By default, the contained name is same as that of the tag

	name. Example, for a given object, the hierarchical name is Line1.Tank1.OutletValve and its contained name is OutletValve. <i>See also</i> hierarchical name, tag name.
CRV	CRV is the abbreviation for a curve. The .CRV file contains the data of the trend curve.
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. <i>See also</i> 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. <i>See also</i> delta storage.
data	The coded representation of information for use in a computer. Data has attributes, such as type and length.
data source	A data source is a database from which a client retrieves data.
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.
deadband	A deadband is the amount of increase or decrease that a value can experience before an event will occur in the system. <i>See also</i> 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. <i>See also</i> 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." <i>See also</i> 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. <i>See also</i> 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). <i>See also</i> message pair.
domain	A domain is a group of computers that share a tree or subtree of a network for security authentication.
.dot file	A .dot file is a template file in Microsoft Word.
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. <i>See also</i> leading edge, trailing edge.
engineering unit	An engineering unit is the unit of measure for a tag. For example, RPMs, milliseconds, degrees.
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. <i>See also</i> detector, action, event tag.
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." <i>See also</i> detector, action, event.
field	<i>See</i> row.
function	A function is a procedure in programming language. <i>See also</i> argument.
Galaxy	The complete ArchestrA system consisting of a single logical namespace and collection of Platforms, Engines, and Objects.
hierarchical name	A hierarchical name is the contained name for an object, preceded by the tag names of the containing objects in the hierarchy. Example, Line1.Tank1.OutletValve. <i>See also</i> tag name, contained name.

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 Wonderware 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.
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	An abbreviation for INPUT/OUTPUT.
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.
instance	An object that exists in runtime.
integer	An integer is any member of the set consisting of the positive and negative whole numbers and zero. Examples: -59, -3, 0.
interpolation	Interpolation is a method of constructing new data points within the range of a discrete set of known data points.
join	A join is a class of SQL query that queries data from one or more columns from two or more tables.
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). <i>See also</i> , edge detection, trailing edge.
limit	A limit is a user-definable maximum or minimum value for a range of values.
live	Live is a term that describes data that reflects the most current value of a tag.
live mode	Live mode is the mode in which the data is retrieved continuously in real time for a fixed duration that is relative to the current time.
local	Local is used to describe the computer that a user is currently logged on to and is physically using. <i>See also</i> remote.

local area network (LAN)	A LAN is a group of computers connected by a communications network. A LAN encompasses a relatively limited network area.
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.
login ID	<i>See</i> 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.
millisecond	One thousandth of a second, abbreviated ms or msec.
network	A network is a communications infrastructure connecting a group of physically connected computers.
node	A node is any computer or device that can be connected to an internetwork. A node is also referred to as a host.
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.
on-demand report	An on-demand report is a type of a dynamic report that is executed upon a user's request.
OPC quality	The quality of a process value or an event. The quality can be rated as Good, Bad, Doubtful, or Initial Value.
opacity	Opacity is a measure of how much an image will block the background when painted.
pan	A pan is a sweeping movement of the chart.
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.
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.
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. <i>See also</i> information system network.
protocol	A protocol is the set of rules and standards to enable computers to connect and exchange data over a network.
quality	Quality is an indicator of the accuracy, availability, and validity of acquired data. Data values stored by the Wonderware 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.
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.
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 Wonderware 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	<i>See</i> row.
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.

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. <i>See also</i> local.
replay mode	Replay mode is a mode that uses real-time speed to continuously plot historical data on the trend chart.
resolution	Resolution is the sampling interval, in milliseconds, to retrieve data from any of the history tables of the Wonderware 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.
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 Wonderware 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.
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.
scatter plot	The graphical representation of variation of a tag's value over the variation of another tag's value.
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.
scheduled report	A scheduled report is a type of a dynamic report that is executed automatically according to a defined schedule.
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 Wonderware Historian performs all of these functions, plus provides for data storage and management. <i>See also</i> client.
server name	The server name is the name of the server under which the Wonderware 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.
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.
SQL	<i>See</i> Structured Query Language.
stacked mode	A stacked mode is a mode in which a tag curve is stacked on top of the other in the trend chart.
stateful	The state of an entity is preserved from one request to another. For example, TCP/IP is a stateful communication protocol.
stateless	The state of an entity is not preserved from one request to another. For example, HTTP is a stateless communication protocol. Using HTTP, when a request is made from the client to the server, the entire transaction is stateless; no state is preserved from one request to another.
statement	An expression of instruction in a computer language.
state summary replication	State summary replication summarizes the states of a tag value. Can be applied to analog, discrete, and string tags.

storage by exception	<i>See</i> 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.
string	A string value is a text expression treated as a single data item. A string does not require a special format or syntax.
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 data	Summary data is data that is the result of an internal calculation performed by the Wonderware 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 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.
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 Wonderware 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. Wonderware Historian system tags have a SYS prefix. For example, SysTimeSec.
table	A table is a group of related data entities and their characteristics. <i>See also</i> row.
Tablet PC	A Tablet PC refers to a wireless personal computer that allows you to take notes using a touch screen or a digital pen.
tag	A tag is defined as an elemental variable of type analog, discrete, string, or complex that is stored in the Wonderware 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).
tag name	A tagname is the name assigned to an elemental variable in the Wonderware 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.
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.
trailing edge	The trailing edge is the query return only rows that are the first to fail the criteria (return false) after a row successfully met criteria (returned true). <i>See also</i> , leading edge, edge detection.

trend	A general direction in which data tends to move in the form of a curve.
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), also known as Greenwich Mean Time, 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 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.
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 can also be applied for retrieval.
wide area network (WAN)	A WAN is a group of geographically separated computers connected by a communications network.
wide tables	In the Wonderware 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. <i>See also</i> "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 "sys", search for "sys%". .xlaThe .xla is the extension of a Microsoft Excel add-in.
XML	XML is the abbreviation for Extensible Markup Language; a flexible format for creating and sharing data on the Web.

Index

Symbols

#date wildcard 383
 #ReportTime wildcard 383
 #time wildcard 382

A

aaChartType enumeration 497
 aaDashStyle enumeration 497
 aaDataPointLabelingType enumeration 497
 aaDateModeEnumeration 497
 aaFieldLabelPositionEnumeration enumeration 623
 aaHistClientActiveDataGrid
 database connection 554
 error messages 578
 methods 567
 properties 558
 aaHistClientActiveDataGrid control
 ActiveDataGrid comparison 837
 adding to an InTouch window 557
 database connection 554
 error messages 578
 events 573
 history data over a LAN 575
 methods 567
 properties 558
 retrieving data from the grid 575
 scripting examples 575
 SQL statement 555
 SQL statements 555
 using at runtime 552
 using in an application 557
 aaHistClientGlobalFunctions object
 comparison 841
 methods 659
 using in an application 659
 aaHistClientQuery control
 adding to InTouch windows 506
 comparison 831
 events 523
 methods 515
 properties 507
 using at runtime 505
 using in an application 506
 aaHistClientReportRunner object 640
 methods 642
 properties 641
 aaHistClientSingleValueEntry control
 adding to an InTouch window 599
 events 621
 methods 616
 migration 836
 properties 600
 using at runtime 597
 using in an application 598

- aaHistClientTagPicker control
 - adding to an InTouch window 528
 - events 537
 - methods 534
 - migration 834
 - properties 529
 - using at runtime 527
 - using in an application 527
- aaHistClientTagPickerSplitterOrientation enumeration 539
- aaHistClientTimeRangePicker control
 - adding to an InTouch window 542
 - events 550
 - methods 547
 - migration 832
 - properties 543
 - using at runtime 541
 - using in an application 541
- aaHistClientTrend control
 - adding to an InTouch window 396
 - enumerations 496
 - events 494
 - methods 462
 - migration 826
 - properties 397
 - using at runtime 395
 - using in an application 395
- aaHistClientWorkbookRunner object 631
 - methods 633
 - properties 631
- aaInterpolationType enumeration 498
- aaQualityRules enumeration 498
- aaQueryTypeEnumeration 524
- aaRetrievalMode enumeration 499
- aaRetrievalSource enumeration 675
- aaRetrievalVersion enumeration 499
- aaServer object 579, 837
 - methods 586
 - properties 579
- aaServerListChangeArgs object 593
- aaServers object 587, 839
 - events 591
 - instantiating 593
 - methods 588
 - properties 587
- aaServerState enumeration 595
- aaServerStateChangeArgs object 594
- aaServerType enumeration 596
- aaStateCalculation enumeration 500
- aaTag object 840
 - properties 625
 - using in an application 625
- aaTagType enumeration 675
- aaTargetRegionExcursionType enumeration 501
- aaTimeRangeEnumeration enumeration 676
- aaTimeStampRules enumeration 501
- aaTraceGradientType enumeration 501
- aaTrendGradientType enumeration 502
- aaTrendType enumeration 502
- aaTrendValueFormat enumeration 502
- aaUpdateToCurrentTimeState 503
- aaUseTimeZoneEnumeration enumeration 623
- aaValueAxisLabelEnumeration 503
- about
 - browser 40
 - controls 390
 - data grid 552
 - Wonderware Historian 21
- AboutBox 465
- absolute time 114, 117
- absolute, time 114
- acquisition 209
- action scripts 210
- actions, undoing/redoing 55
- ActiveDataGrid comparison,
 - aaHistClientActiveDataGrid control 837
- ActiveServer 508
- Add 589
- AddAnyTag 465
- adding
 - groups 42
 - tag values 598
 - value 598
- adding a tag, groups 42
- adding an annotation, trend 108
- adding for a trend, annotations 108
- adding to a group, tags 42
- adding to an InTouch window
 - aaHistClientActiveDataGrid control 557
 - aaHistClientSingleValueEntry control 599
 - aaHistClientTagPicker control 528
- add-ins

- for Microsoft office 20
 - manually loading 220, 363
 - manually unloading 220, 363
 - templates 359
 - AddMultipleTags 403
 - address string 209
 - AddServer 465, 515, 616, 646
 - AddTag 467, 517, 618
 - aggregate functions
 - defined 843
 - aggregate values
 - querying 176
 - retrieving 259
 - alarm history, querying 178
 - alarm limits 180
 - querying 181
 - retrieving 245
 - aliases 35
 - AllowContextMenu 403
 - AllowGridEditing 403
 - AllowQueryTypeChange 508
 - AllowUserConfiguration 559
 - AlwaysUseFullForXYScatterPlots 403
 - analog
 - defined 843
 - analog summary replication
 - defined 843
 - analog tag analysis 278
 - analog tags
 - edge detection 798, 800, 801
 - analog values at discrete transition analysis 298
 - analog/discrete pair analysis 302
 - AnalogPlottingAlgorithm 404
 - AnalogValue 601
 - annotations
 - adding for a trend 108
 - configuring properties 140
 - creating using a Tablet PC 161
 - defined 843
 - deleting 111
 - deleting in a trend 111
 - description 107
 - editing 110
 - for a trend 107
 - printing 112
 - querying 185
 - saving as a .csv file 111
 - trend 107
 - using 107
 - viewing list 109
 - annotations using a Tablet PC, creating 161
 - application name
 - I/O Server 198
 - application name, I/O Server 209
 - application programming interface (API)
 - defined 843
 - ApplicationName 588
 - ApplyFilter 534
 - ApplyRubberBandToAllTags 404
 - architecture
 - defined 843
 - argument
 - defined 843
 - arguments, functions 340
 - array formula, refreshing 224
 - attribute
 - defined 844
 - authentication 33
 - defined 844
 - Auto_Close 646
 - Auto_Open 646
 - AutoExec 657
 - AutoExit 657
 - automatically scaling, tags 92
 - AutoRefresh 559
 - AutoRefresh property 559
 - AutoRefreshMode 405
 - average (time-weighted), retrieval modes 712
 - averages 713
 - axis cursors
 - showing tag values on value axis 90
 - showing/hiding 99
 - using 98
 - axis properties, configuring 136
- ## B
- back end
 - defined 844
 - BackColor 405, 664
 - BackGradient 406
 - BackGradientEndColor 406
 - BackImage 406
 - BackStyle 664

- base date 319
- base time 319
- BaseURLAddress 580
- batch analysis 286
- batch statements 34
- best fit retrieval mode 707
- best fit, retrieval modes 707
- binding options 321
- binding values 324
- block
 - defined 844
- BOF 560
- BOF property 560
- BorderColor 407
- BorderStyle 407, 664
- BorderWidth 407
- bound report, creating 322
- browser
 - about 40
 - defined 844
- Build 580
- BusinessObjectServer 560
- C**
- calculation 208, 211, 235, 262, 267
- Calculations tab 178, 208, 262
- CausesValidation 665
- cells 223
- cells, selecting 231
- Change 621
- characters
 - limit for string tags 209, 243
- ChartType 407
- ClearGrid 567
- ClearGrid method 567
- clearing, data grid 567
- ClearTags 467, 517
- Click 673
- client
 - defined 844
- client tools, connecting to the server 27
- client/server
 - defined 844
- client/server architecture 22
- closing, trend 55
- Color 678
- color options, configuring 126
- color properties, configuring 134
- column 568, 570
- ColumnCount 561
- ColumnName 567
- Columns Pane 167
- Columns tab 179, 187, 189, 191, 199, 207
- ColumnValue 568
- ColumnValue method 568
- ColumnValueByName 568
- common
 - events 672
 - methods 671
 - properties 663
- common properties for , tag values 271
- comparison
 - aaHistClientGlobalFunctions object 841
 - aaHistClientQuery control 831
- configuration details, retrieving for a tag 239
- configuration, server connections 27
- configuring
 - scatter plots 143
 - target region 73
 - trend 56
- configuring , formatting options 309
- configuring axes, scatter plots 146
- configuring for scatter plot, target region 148
- configuring options
 - reports 386
 - scatter plots 150
- configuring properties
 - annotations 140
 - target region 141
- configuring the database connection 554
- Connect 619
- Connected 561
- console
 - defined 844
- Container 665
- ContextMenuEnabled 665
- contexts
 - defined 844
- controls
 - about 390
 - getting started 391
 - using in different environments 392

- using within InTouch HMI software 393
 - Controls 21
 - converting to values
 - functions 224
 - sheet 225
 - copying to the clipboard, trend chart 158
 - copying, functions 230
 - CopyQuery 517
 - copyright information 49
 - Count 588
 - counter retrieval mode 737
 - counter, retrieval modes 737
 - CreateManualTag 619
 - creating
 - groups 42
 - query 173, 186
 - reports 222
 - server connections 28
 - SQL statements 555
 - trend set 154
 - Creating a New Trend 53
 - Criteria tab 176, 185, 194, 204, 274
 - .crv data, importing 164
 - CSV data
 - defined 845
 - CurrentServer 508, 529
 - CurrentServerName 422, 602
 - CurrentTagChanged 494
 - CurrentTagColor 408
 - CurrentTagCycleCount 408
 - CurrentTagEffectiveRetrievalMode 409
 - CurrentTagFormat 409
 - CurrentTagGetStyle 467
 - CurrentTagHistoryVersion 409
 - CurrentTagIndex 410
 - CurrentTagInterpolationType 410
 - CurrentTagName 411
 - CurrentTagNumStyles 411
 - CurrentTagOffsetMS 411
 - CurrentTagPenStyle 412
 - CurrentTagPenWidth 412
 - CurrentTagPrecision 412
 - CurrentTagQualityRule 413
 - CurrentTagResolution 413
 - CurrentTagRetrievalMode 414
 - CurrentTagRetrievalStyle 414
 - CurrentTagRowLimit 415
 - CurrentTagStartDate 415
 - CurrentTagState 415
 - CurrentTagStateCalculation 416
 - CurrentTagTargetRegionVisible 416
 - CurrentTagTimeDeadband 417
 - CurrentTagTimeStampRule 417
 - CurrentTagTrendType 418
 - CurrentTagUseAutoCycles 418
 - CurrentTagUseResolution 419
 - CurrentTagValAtX1 419
 - CurrentTagValAtX2 420
 - CurrentTagValueDeadband 420
 - CurrentTrendItem 421
 - CurrentValOfX1 424
 - CurrentValOfX2 424
 - CurrentValOfY1 425
 - CurrentValOfY2 425
 - CurrentXAxisTagIndex 425
 - CurrentXAxisTagName 426
 - CurrentXAxisTagServerName 426
 - cursor difference, showing/hiding 99
 - CutQuery 517
 - cycle count 755
 - defined 850
 - cyclic retrieval
 - defined 845
 - cyclic retrieval, about 690
 - cyclic storage
 - defined 845
 - cyclic, retrieval modes 690
 - CyclicRows 426
- ## D
- data
 - defined 845
 - data grid
 - about 552
 - clearing 567
 - refreshing 556
 - data log
 - viewing in a narrow format 101
 - viewing in a wide format 103
 - data quality 102, 180, 189, 192, 200, 207, 250, 272
 - data source options, configuring 318
 - data sources
 - defined 845
 - data type

- defined 845
- data types 678
- database
 - defined 845
- database connection
 - aaHistClientActiveDataGrid 554
 - aaHistClientActiveDataGrid control 554
 - configuring the 554
- database options, configuring 124, 131
- database user 110, 185
- DatabaseName 561
- DataBindings 665
- DataPointLabelType 427
- DataSet 679
- date and time
 - options 380
 - wildcards 382
- date/time format, verifying 231
- DateCreated 626
- DateMode 427
- DatePickerFormatString 428
- DatesChanged 495
- DateTime 602, 678
- DateTimeFieldDisable 602
- DateTimeFieldVisible 602
- DateTimeString 603
- DbClick 673
- deadband 215, 264, 355, 762, 766, 853, 854
 - defined 845
 - time 853
 - value 854
- deadband, for events 243
- default.aaTrend 53
- DefaultColumnWidth 562
- DefaultTagFormat 428
- DefaultTagPrecision 429
- defining target region, scatter plots 148
- defining, retrieval styles 810
- DeleteCurrentTag 468
- deleting
 - annotations 111
 - groups 42
- deleting an annotation, trend 111
- deleting files from, trend set 155
- deleting from a trend, tags 66
- deleting in a trend, annotations 111
- delta retrieval

- defined 845
- delta retrieval, about 693
- delta storage
 - defined 845
- delta, retrieval modes 693
- Description 626
- description, annotations 107
- DescriptionFilter 530
- desktop applications 20
- detection time 187, 189, 250, 272
- detector
 - defined 846
- detector scripts 210
- deviation
 - defined 846
- direct query, creating 306
- DisableTagEntry 603
- Disconnect 619
- discrete
 - defined 846
- discrete tag analysis 293
- discrete tags
 - edge detection 802, 803, 805, 806
- DisplayErrorMessages 604
- displaying in the trend, tags 80
- displaying live data, trend 81
- displaying, live values 81
- documentation conventions 17
- Domain 562, 581
- domain
 - defined 846
- Drag 671
- DragDrop 673
- DragIcon 666
- DragMode 666
- DragOver 673
- duration 48
- duration XML element 815
- DurationMS 543
- Dynamic Link Library (.DLL)
 - defined 846

E

- edge detection 797
 - analog tags 798, 800, 801
 - defined 846
 - discrete tags 802, 803, 805, 806
- editing

- annotations 110
 - functions 224
 - groups 42
 - query 378
 - server connections 30
 - trend set 155
- e-mail 157
- EnableAllQueriestab 509
- Enabled 562, 666
- EnableDeltaRetrieval 429
- EnableShortcutMenu 562
- EnableSummaryData 429
- EnableTimeOffsets 430
- end time 48
- EndDate 430, 543
- EndDateUTC 543
- engineering units 192, 200, 209, 215, 242, 264, 355, 627, 766
 - defined 846
- enumerations, aaHistClientTrend control 496
- enumerations, common 675
- EOF 563
- EOF property 563
- ErrDescription 632, 641
- ErrNumber 632, 641
- error messages
 - aaHistClientActiveDataGrid 578
 - aaHistClientActiveDataGrid control 578
 - functions 356
- error reporting 36
- event actions
 - snapshot actions
 - about 232
- event history values, querying 186
- event snapshot
 - querying 188
 - retrieving values for tags 268
 - selecting tags 238
- event tag 113
- event tags
 - defined 846
- events
 - aaHistClientActiveDataGrid control 573
 - aaHistClientQuery control 523
 - aaHistClientSingleValueEntry control 621
 - aaHistClientTagPicker control 537
 - aaHistClientTimeRangePicker control 550
 - aaHistClientTrend control 494
 - aaServers object 591
 - common 672
 - defined 846
 - edge detection 797
 - trending 113
 - ExactMatchFilter 530
 - ExcelVisible 633
 - Execute 568
 - Execute method 568
 - exporting, trend 156
 - extension tables 754
- F**
 - FALSE 209, 628
 - FavoriteQueriesFolder 509
 - favorites, query 190
 - field
 - defined 846
 - field codes 366
 - inserting into a report 381
 - FieldLabelPosition 604
 - FieldLayoutHorizontal 604
 - file structure, retrieval styles 810
 - FileName 430
 - FileNew 468
 - FileOpen 468, 518
 - FileSave 469
 - filter pane, using 44
 - FilterVisible 530
 - Font 666, 679
 - FontBold 509
 - FontCharset 510
 - FontItalic 511
 - FontName 511
 - FontSize 511
 - for a trend, annotations 107
 - for controls and objects, migration 824
 - ForeColor 667
 - Format 544
 - formats, time offsets 118
 - formatting options 315
 - configuring 309

- referencing in a query 312
- formulas 223
- full, retrieval modes 700
- function
 - defined 846
- functions 223
 - arguments 340
 - converting to values 224
 - copying 230
 - editing 224
 - error messages 356
 - manually editing 228
 - manually inserting 225
 - refreshing 224

G

- general options, configuring 319
- general properties, configuring 133
- GetDictionaryPath 660
- GetInstallPath 660
- GetLastError 647
- GetMenuItemEnabled 469
- GetServer 589
- GetStartAndEndTimes 547
- GetStartAndEndTimesUTC 548
- GetTagColor 470
- GetTagFormat 470
- GetTagOffsetMS 470
- GetTagPenStyle 471
- GetTagPenWidth 472
- GetTagPrecision 472
- GetTagValAtX1 472
- GetTagValAtX2 473
- GetTagVisible 474
- getting started
 - controls 391
 - Wonderware Historian Client Query 165
 - Wonderware Historian Client Trend 51
- GetToolBarButtonEnabled 474
- GotFocus 673
- GraphStack 474
- grid, showing/hiding 100
- GridColor 430
- GridHorizontal 431
- GridVertical 431
- GridVisible 431
- groups 41

- adding 42
- adding a tag 42
- creating 42
- deleting 42
- editing 42
- renaming 43

H

- Handle 563
- Height 667
- HelpContextID 667
- HideCaption 530
- HideCurrentTag 431
- HideDateTimeModeTabs 605
- HideFieldLabels 605
- HideStatusBar 605
- hiding/showing in a trend, tags 80
- hiding/showing tags, trend 80
- hierarchical name 46, 64
- HighlightCurrentTag 432
- highlighting, tags 79
- historical data 65
- historical data, displaying in "replay" mode 82
- history data over a LAN,
 - aaHistClientActiveDataGrid control 575
- history data retrieval functions,
 - migration 356
- history values
 - querying 191
 - retrieving 251
- history version 769
- HistorySource 432
- HTTP 27
- HTTP access 32, 33, 37
- HTTP access, using 32
- Human-Machine Interface (HMI)
 - defined 847
- Hypertext Transfer Protocol (HTTP)
 - defined 847

I

- I/O
 - defined 847
- I/O Server, querying 198
- in applications, SQL statements 35
- Index 667
- initialization

- defined 847
 - Insert 620
 - InsertButtonDisable 606
 - InsertButtonVisible 606
 - InsertComplete 622
 - InsertFail 622
 - inserting into a report
 - field codes 381
 - InsertValue 620
 - instantiating, aaServers object 593
 - integer
 - defined 847
 - integral retrieval mode 731
 - integral, retrieval modes 731
 - interpolated, retrieval modes 701
 - interpolation 701
 - interpolation of values, scatter plots 146
 - interpolation type 771
 - InTouch window
 - aaHistClientTimeRangePicker control, adding to an 542
 - aaHistClientTrend control, adding to an 396
 - InTouch windows, aaHistClientQuery control, adding to 506
 - InTouch® HMI 21
 - InTouchDateTime 606
 - IOAddress 626
 - IOAddressFilter 531
 - Items 588
- J**
- join
 - defined 847
- K**
- KeyDown 673
 - KeyPress 674
 - KeyUp 674
- L**
- LastErrorDetails 607
 - LastErrorMessage 607
 - LastOperationResult 608
 - LastOperationSuccessful 608
 - leading edge
 - defined 847
 - leading edge detection 800, 801, 803, 806
 - Left 668
 - legacy retrieval settings 132, 820
 - licensing 49
 - limit
 - defined 847
 - limit names 181, 182
 - limit properties, configuring 138
 - limit type 181
 - limits
 - for a storage location 205
 - linear interpolation 772
 - live
 - defined 847
 - live data 65, 81
 - live values
 - displaying 81
 - querying 199
 - retrieving 248
 - LiveModeRate 432
 - LoadCRVString 475
 - LoadTargetRegionFromFile 475
 - local
 - defined 847
 - local area network (LAN)
 - defined 848
 - LockDown 433, 511
 - LoggedOn 581
 - logical operators
 - defined 848
 - login identification
 - defined 848
 - login IDs
 - defined 848
 - LoginID 581
 - LoginTimeout 433, 582
 - LogOff 586
 - LogOn 535, 587
 - logon
 - defined 848
 - LostFocus 674
- M**
- MachineName 582
 - managing, server connections 218, 366
 - ManualConnect 475, 518
 - manually editing, functions 228
 - manually inserting, functions 225
 - manually loading 220, 363

- matching of value pairs, scatter plots 146
 - MaxDeltaSamples 434
 - MaxEU 627
 - maximum retrieval mode 725
 - maximum, retrieval modes 725
 - MaxMinutesForDeltaAnalog 434
 - MaxMinutesForDeltaDiscrete 435
 - MaxRaw 627
 - MaxSamplesPerTag 435
 - MDACOk 661
 - Message 595
 - Message0 628
 - Message1 628
 - methods
 - aaHistClientActiveDataGrid 567
 - aaHistClientActiveDataGrid control 567
 - aaHistClientGlobalFunctions object 659
 - aaHistClientQuery control 515
 - aaHistClientReportRunner object 642
 - aaHistClientSingleValueEntry control 616
 - aaHistClientTagPicker control 534
 - aaHistClientTimeRangePicker control 547
 - aaHistClientTrend control 462
 - aaHistClientWorkbookRunner object 633
 - aaServer object 586
 - aaServers object 588
 - common 671
 - Report object 657
 - Microsoft office, add-ins 20
 - Microsoft SQL Server 21
 - migration
 - aaHistClientSingleValueEntry control 836
 - aaHistClientTagPicker control 834
 - aaHistClientTimeRangePicker control 832
 - aaHistClientTrend control 826
 - for controls and objects 824
 - history data retrieval functions 356
 - millisecond
 - defined 848
 - MinEU 627
 - minimum, retrieval modes 719
 - MinRaw 627
 - miscellaneous options, configuring 130
 - Mode 628
 - ModeChanged 523
 - MouseDown 674
 - MouseMove 674
 - MouseUp 674
 - Move 671
 - MoveFirst 569
 - MoveLast 569
 - MoveNext 569
 - MoveNextTag 475
 - MovePrevious 570
 - MovePrevTag 476
 - Moving a Cursor 99
 - moving tags up or down, trend 94
 - MovingAverageMode 435
 - MovingAverageSamples 436
 - multiple scales 87
 - multiple, scale 87
- ## N
- Name 582, 628, 668
 - named range 315
 - named range, formatting options 314
 - navigator bar 553, 565
 - network
 - defined 848
 - node
 - defined 848
 - NULL
 - defined 848
 - NumDataPointLabels 436
 - numerical data types 393
 - NumTimeAxisGridPerValue 436
 - NumTimeAxisValues 437
 - NumXValueAxisGridLinesPerLabel 437
 - NumXValueAxisLabels 437
 - NumYAxisGridPerValue 437
 - NumYAxisValues 438
- ## O
- Object 668, 679
 - objects 41
 - objects, about 390
 - OnChange 550
 - OnClick 573
 - OnError 573
 - OnFilterChanged 538

- OnGroupChanged 538
 - OnSelectedTabChanged 539
 - OnServerAdded 591
 - OnServerChanged 539
 - OnServerRemoved 592
 - OnServerStateChange 592
 - OnServerUpdated 591
 - OnTagsPicked 538
 - OnTagsSelected 538
 - OPC quality 778
 - defined 848
 - opening
 - report documents 369
 - trend 54, 172
 - OpenQuery 518
 - options, date and time 380
 - original data 769
 - OutputFile 632, 641
 - outputting, trend data 153
 - over HTTP 34
 - overview
 - scatter plots 142
 - target region 70
- P**
- PanLeft 476
 - panning 96
 - panning in, trend 96
 - panning, in a scatter plot 147
 - panning, scatter plots 147
 - PanPercentage 438
 - PanRight 476
 - parameter
 - defined 848
 - Parent 668
 - partial cycle
 - defined 848
 - Password 563, 583
 - passwords
 - defined 849
 - PasteQuery 518
 - PatchLevel 583
 - Pen Configuration 68
 - performance 36
 - phantom cycle
 - defined 849
 - PlaybackSpeed 438
 - PlotColor 438
 - PlotGradient 439
 - PlotGradientEndColor 439
 - PlotImage 439
 - PrintGraph 476
 - PrintGraphDlg 476
 - printing
 - annotations 112
 - trend 153
 - trend data 153
 - trend set 153, 156
 - printing, trends 153
 - PrintShowActiveTag 440
 - PrintShowMarkers 440
 - PrintShowTitle 440
 - PrintTitle 440
 - private groups 41
 - process data 26
 - process network
 - defined 849
 - program information 49
 - program information, viewing 49
 - properties 571, 593, 594
 - aaHistClientActiveDataGrid 558
 - aaHistClientActiveDataGrid control 558
 - aaHistClientQuery control 507
 - aaHistClientReportRunner object 641
 - aaHistClientSingleValueEntry control 600
 - aaHistClientTagPicker control 529
 - aaHistClientTimeRangePicker control 543
 - aaHistClientTrend control 397
 - aaServer object 579
 - aaServers object 587
 - aaTag object 625
 - common 663
 - methods
 - events 391
 - Report object 656
 - PropertiesDlg 477
 - protocols
 - defined 849
 - public groups 41
 - PublicAnnotations 441
 - publishing
 - trend reports 158
 - workbook reports 326

Pwd 609

Q

Quality 609

quality 102, 180, 189, 192, 200, 207, 250, 272

defined 849

retrieval rule 778

quality calculation, scatter plots 147

quality rule 778

QualityDetail 609

QualityDetailFieldDisable 610

QualityDetailFieldVisible 610

QualityFieldDisable 610

QualityFieldVisible 611

query

creating 173, 186

defined 849

editing 378

favorites 190

file

opening 172

saving 172

time options for 325

types 175

common tabs for 212

query files 172

QueryChanged 523

QueryFont 512

querying

aggregate values 176

alarm limits 181

annotations 185

event snapshot 188

history values 191

live values 199

queryingnumber of , tags 200

QueryString 512

QueryTimeout 441, 583

R

Random Access Memory (RAM)

defined 849

rate of change

defined 849

raw value 192, 200, 209, 242, 627

defined 849

RawType 629

real

defined 849

real-time

defined 849

RealTimeMode 441

RealTimeRate 442

Reconnection Time 31

record

defined 849

Recordset 512

referencing in a query, formatting

options 312

Refresh 519, 621

refresh,

aaHistClientActiveDataGrid 559, 564

RefreshData 477

RefreshFrequency 564

RefreshFrequency property 564

refreshing

data grid 556

functions 224

sheet 225

trend chart 66

refreshing the graph, trend 66

RefreshTags 536

RefreshTimes 548

registry

defined 849

relational databases

defined 850

relative time 114, 116, 117

relative time, time offsets 116

relative, time 116

reliability 36

RememberEnteredTags 611

remote

defined 850

Remove 589

RemoveServer 477

RemoveTag 478, 519

removing, server connections 31

renaming, groups 43

Replicated tag 57

replicated tags 57

replication latency

defined 850

Report 20

report documents

- opening 369
- running 369
- saving 370, 371
- saving as a report template 371
- saving as an HTML file 373
- Report object
 - methods 657
 - properties 656
- ReportDate 656
- reports
 - configuring options 386
 - creating 222
 - selecting tags for 232
- ReportTime 656
- Reset 621
- resolution 263, 757
 - defined 850
- result
 - defined 850
- Results pane 167
- RetainPassword 584
- retrieval 214
 - edge detection 797
 - quality rule 778
 - time deadband 762
 - time zone 777
 - value deadband 766
 - version of data 769
- retrieval modes
 - average (time-weighted) 712
 - best fit 707
 - counter 737
 - cyclic 690
 - delta 693
 - full 700
 - integral 731
 - interpolated 701
 - maximum 725
 - minimum 719
 - slope 733
 - time-in-state 741
 - ValueState 741
- retrieval options, configuring for a tag 75
- retrieval styles
 - defining 810
 - file structure 810
- Retrieval tab 195
- retrieval XML element 816
- RetrievalOptionsCycleCount 442
- RetrievalOptionsGetStyle 478
- RetrievalOptionsHistoryVersion 443
- RetrievalOptionsInterpolationType 443
- RetrievalOptionsNumStyles 443
- RetrievalOptionsQualityRule 444
- RetrievalOptionsResolution 444
- RetrievalOptionsRetrievalMode 445
- RetrievalOptionsRetrievalStyle 445
- RetrievalOptionsRowLimit 446
- RetrievalOptionsState 446
- RetrievalOptionsStateCalculation 446
- RetrievalOptionsTimeDeadband 447
- RetrievalOptionsTimeStampRule 447
- RetrievalOptionsUseAutoCycles 448
- RetrievalOptionsUseResolution 449
- RetrievalOptionsValueDeadband 449
- retrievalStyle XML element 814
- RetrieveAnnotations 450
- RetrieveExtensionData 450
- RetrieveManualData 450
- retrieving
 - aggregate values 259
 - alarm limits 245
 - history values 251
 - live values 248
 - tag values 247
- retrieving configuration information, tags 239
- retrieving data from the grid, aaHistClientActiveDataGrid control 575
- retrieving values for tags, event snapshot 268
- returning to original scale, tags 93
- RevisionNumber 584
- rollover value 737
- roperties, aaHistClientWorkbookRunner object 631
- Row 564
- row
 - defined 850
- row count 214, 263, 304
- Row property 564
- RowColumnValue 570
- RowColumnValue method 570
- RowColumnValueByName 570
- RowCount 564
- RowCount property 564

- rowset
 - defined 850
- RTRate 451
- rubber band scaling 95
- Rubberband 451
- RubberbandAll 451
- RubberBandScaling 451
- Run 633, 642
- running, report documents 369
- RunReport 634, 648, 657
- RunReport2 637
- runtime
 - defined 850
- S**
- sampling interval 757
- SaveData 479
- SaveImage 479
- SaveQuery 519
- SaveResults 519
- SaveSettings 480
- saving
 - report documents 370, 371
 - trend 54
- Saving a Trend 54
- saving as a .csv file, annotations 111
- saving as a report template, report documents 371
- saving as an HTML file, report documents 373
- saving to a .csv file, trend data 156
- scale
 - multiple 87
 - single 85
- ScaleAllTags 480
- ScaleAllTagsDlg 480
- ScaleAutoAllTags 480
- ScaleAutoTag 481
- ScaleDownAllTags 481
- ScaleDownTag 481
- ScaleMoveAllTagsDown 481
- ScaleMoveAllTagsUp 482
- ScaleMoveTagDown 482
- ScaleMoveTagUp 482
- ScaleTag 482
- ScaleTagDlg 483
- ScaleUpAllTags 483
- ScaleUpTag 483
- scaling 92
 - defined 850
- scaling tags 83, 90
- scaling tags, scatter plots 145
- scaling, tags in a scatter plot 145
- scan rate
 - for event tags 210, 243, 352
- scatter analysis 290
- scatter plots
 - configuring 143
 - configuring axes 146
 - configuring options 150
 - defining target region 148
 - interpolation of values 146
 - matching of value pairs 146
 - overview 142
 - panning 147
 - quality calculation 147
 - scaling tags 145
 - zooming 147
- SchemaVersion 584
- script
 - defined 851
- scripting examples,
 - aaHistClientActiveDataGrid control 575
- scrolling through tags, trend 78
- Search tab 211
- searching for, tags 210
- SelectedPath 531
- SelectedTag 536
- SelectedTagCount 531
- selecting , tags 233
- selecting tags for, reports 232
- selecting tags, event snapshot 238
- Server 593, 594, 629
- server
 - defined 851
- server connections
 - configuration 27
 - creating 28
 - editing 30
 - managing 218, 366
 - removing 31
- server details, viewing 43
- server name
 - defined 851
- server version, querying 201
- ServerChanged 524

- ServerName 565, 584
- Servers 452, 513, 532, 611
- Servers pane
 - showing/hiding 41
 - using 41
- ServerType 585
- services
 - defined 851
- SetCurrentTag 483
- SetCurrentTagXAxisTag 484
- SetDates 485, 520
- SetDuration 485, 520
- SetFocus 671
- SetMenuItemEnabled 486
- SetQueryType 521
- SetQueryType2 521
- SetStartAndEndTimes 548
- SetStartAndEndTimesUTC 549
- SetTagColor 487
- SetTagColorDlg 488
- SetTagFormat 488
- SetTagOffsetMS 488
- SetTagPenStyle 489
- SetTagPenWidth 490
- SetTagPrecision 490
- SetTagVisible 491
- SetTimeSpan 491, 522
- SetToolBarButtonEnabled 492
- sheet
 - converting to values 225
 - refreshing 225
- ShowAbout 522
- ShowErrorDlgs 565
- showing tag values on value axis, axis cursors 90
- showing/hiding
 - axis cursors 99
 - Servers pane 41
 - Tag Picker 46
 - target region 75
- ShowLimits 452
- ShowNavigatorBar 565
- ShowNavigatorBar property 565
- ShowPropertiesDialog 571
- ShowPropertiesDialog method 571
- ShowStatistics 493
- ShowValuesAtCursor 452
- ShowWaitCursor 452
- ShowWhatsThis 672
- ShowXAxisCursors 453
- ShowYAxisCursor 453
- simple replication
 - defined 851
- single scale 85
- single, scale 85
- SingleSelectMode 532
- SingleTagMode 453
- slope retrieval mode 733
- slope, retrieval modes 733
- snapshot
 - defined 851
- snapshot actions 232
- source and replicated tags
 - adding 57
 - finding 59, 170, 236
 - replacing 61
- Source tab 215
- SourceFile 632, 642
- SplitterOrientation 532
- SQL
 - defined 851
- SQL cursors 34
- SQL statement,
 - aaHistClientActiveDataGrid control 555
- SQL statements 34, 566, 568, 571
 - aaHistClientActiveDataGrid control 555
 - creating 555
 - in applications 35
- SQLAppend 571
- SQLString 566
- stacked mode 88
- stairstep interpolation 771
- start time 48
- StartDate 454, 545
- StartDateUTC 545
- starting, Wonderware Historian Client Trend 51
- State 585, 594
- state summary replication
 - defined 851
- StateChanged 495
- statement
 - defined 851
- statistical average 713
- statistics

- viewing 105
 - viewing in a trend 105
- status bar 39, 172
- status bar, description 39
- storage by exception
 - defined 852
- storage locations
 - defined 852
 - path 205
- storage path
 - defined 852
- storage rate 209, 242, 243
 - defined 852
- storage size available, querying 206
- storage start date, querying 206
- storage type 209
- strings
 - defined 852
- StringValue 612
- Structured Query Language (SQL)
 - defined 852
- styleCollection XML element 813
- SuiteLink
 - defined 852
- summaries
 - defined 852
- summarized tags, retrieving values for 264
- summary data
 - defined 852
- summary date 207
- Summary Options tab 267
- summary replication
 - defined 852
- summary tags, selecting 235
- summary values, querying 206
- SummaryDataMode 454
- SupressErrors 454
- system administrator (sa)
 - defined 852
- system objects 41
- system parameters
 - defined 852
- system tags
 - defined 853

T

TabIndex 669

- tables
 - defined 853
- Tablet PC 161
- TabStop 669
- Tag 669
- tag cursors 98
- tag data 277
- tag definition, viewing 63
- tag details, querying 208
- Tag Picker 40
 - showing/hiding 46
 - views 47
- tag search 40
- Tag Set tab 188
- tag values
 - adding 598
 - common properties for 271
 - retrieving 247
- TagDisplayChanged 495
- TagGridOrientation 454
- TaglistChanged 496
- TagListRows 455
- TagName 612
- tagname
 - defined 853
- TagNameChanged 622
- TagNameFieldDisable 612
- TagNameFieldVisible 613
- TagNameFilter 533
- TagPicker 455
- TagPickerButtonDisable 613
- TagPickerButtonVisible 613
- TagPickerVisible 455
- tags
 - adding to a group 42
 - automatically scaling 92
 - defined 853
 - deleting from a trend 66
 - displaying in the trend 80
 - hiding/showing in a trend 80
 - highlighting 79
 - queryingnumber of 200
 - retrieving configuration information 239
 - returning to original scale 93
 - searching for 210
 - selecting 233
- Tags object 841

- tags pane, using 44
- TagType 614
- TagValid 614
- target region
 - configuring 73
 - configuring for scatter plot 148
 - configuring properties 141
 - overview 70
 - showing/hiding 75
- TargetRegionExcursionType 455
- TargetRegionOpacity 456
- TCP 32
- TCP/IP
 - defined 853
- technical support, contacting 18
- templates, add-ins 359
- threshold
 - for storage 205
- tier-1 historian
 - defined 853
- tier-2 historian
 - defined 853
- time
 - absolute 114
 - relative 116
- time deadband 762, 853
 - defined 853
- time deadband, retrieval 762
- time domain extensions 754
- time interval
 - defined 853
 - for event tags 210, 243, 352
- time offsets 117, 119
 - formats 118
 - relative time 116
- time options for, query 325
- time running, querying 212
- Time tab 212
- Time toolbar 47
- time zone 777
- time zone options, configuring 128, 316
- TimeAxisLabelColor 456
- TimeBarVisible 456
- TimeBarVisible2 457
- TimeDuration 546
- time-in-state 786
- time-in-state retrieval mode 741
- time-in-state, retrieval modes 741
- timeout 198
- TimeSelector 457
- timestamp rule 774
- time-weighted average retrieval
 - mode 712, 731
- toolbar, Wonderware Historian Client Query 167
- ToolBarConnectVisible 513
- ToolBarEditVisible 513
- ToolBarRequeryVisible 513
- ToolBarVisible 457
- ToolBarVisible 514
- ToolBarVisible2 457
- ToolTipText 458, 669
- Top 670
- topics 198, 209
- TraceGradientEndingPercentage 458
- TraceGradientStartingPercentage 458
- TraceGradientType 458
- traces, stacking 80
- trailing edge
 - defined 853
- trailing edge detection 800, 801, 805, 806
- Transact-SQL 754
- Transparent 670
- TreeVisible 533
- TreeWidth 534
- trend
 - adding an annotation 108
 - annotations 107
 - closing 55
 - configuring 56
 - deleting an annotation 111
 - displaying live data 81
 - exporting 156
 - hiding/showing tags 80
 - moving tags up or down 94
 - opening 54, 172
 - panning in 96
 - printing 153
 - refreshing the graph 66
 - saving 54
 - scrolling through tags 78
 - trending events 113
 - viewing data 101
 - viewing statistics 105
 - zooming 99

trend application options, configuring 124
 trend chart
 copying to the clipboard 158
 refreshing 66
 viewing data 65
 trend chart , saving to an image file 157
 trend data
 outputting 153
 printing 153
 saving to a .csv file 156
 viewing in a table format 101
 trend display options, configuring for a tag 67
 trend file properties, configuring 133
 trend file, e-mailing 157
 trend files 52
 trend report, publishing 159, 160
 trend reports, publishing 158
 trend set
 creating 154
 deleting files from 155
 editing 155
 printing 153, 156
 trending , events 113
 trending events, trend 113
 trending, events 113
 TRUE 209, 628
 TrustedConnection 585
 Type 629
 TypeAsTagType 630
 types, query 175

U

unit of measure 181, 182, 192, 200, 209, 242, 630
 Units 630
 Universal Naming Convention (UNC)
 defined 854
 Update 590
 update
 defined 854
 UpdateToCurrentTimeState 459
 UseHttp 586
 UseIniFile 459
 UsePersistedServers 514
 User 615
 user name
 defined 854

UserName 566
 users 110, 185
 UseTimezone 615
 using
 annotations 107
 axis cursors 98
 Servers pane 41
 using at runtime
 aaHistClientActiveDataGrid control 552
 aaHistClientQuery control 505
 aaHistClientSingleValueEntry control 597
 using in an application
 aaHistClientActiveDataGrid control 557
 aaHistClientGlobalFunctions object 659
 aaHistClientSingleValueEntry control 598
 aaTag object 625
 using in different environments, controls 392
 using within InTouch HMI software, controls 393
 UTC 777
 defined 854

V

Validate 675
 Value 615
 value
 adding 598
 Value axis range 68
 value deadband 766, 854
 defined 854
 value deadband, retrieval 766
 ValueAxisLabel 459
 ValueChanged 623
 ValueFieldDisable 616
 ValueState retrieval mode 741
 ValueState, retrieval modes 741
 version information 49
 version of data, retrieval 769
 versioned data 769
 viewing , statistics 105
 viewing data
 trend 101
 trend chart 65

viewing details, Wonderware
Historian 358

viewing in a narrow format, data log 101

viewing in a table format, trend data 101

viewing in a trend, statistics 105

viewing in a wide format, data log 103

viewing list, annotations 109

viewing statistics, trend 105

Viewing the Hierarchical Name of a Tag
in a Trend 64

views, Tag Picker 47

VirtualDirectoryName 566, 586

Visible 421, 670

VPN access 38

W

WAN
defined 854

WhatsThisHelpID 670

When 595

wide tables
defined 854

Width 671

wildcard character
defined 854

wildcards, date and time 382

wildcards, inserting into a report 384

Wonderware Historian 17, 21, 27, 69, 217,
359, 505
about 21
viewing details 358

Wonderware Historian Client 17, 19, 49

Wonderware Historian Client Query
getting started 165
toolbar 167

Wonderware Historian Client Report,
getting started 360

Wonderware Historian Client software,
about 19

Wonderware Historian Client Trend
getting started 51
starting 51

Wonderware Historian Client Workbook,
getting started 217

WordVisible 642

Workbook 20

workbook file, opening 219

workbook options, configuring 309

workbook report, publishing 327, 328

workbook reports, publishing 326

wwEdgeDetection column
about 797

wwHistory function, updating to
wwHistory2 356

wwParameters column 754

wwTimeZone column
about 777

wwWideHistory function, updating to
wwWideHistory2 356

X

x-axis, tag cursors 98

XCursor1Color 460

XCursor1Pos 460

XCursor2Color 461

XCursor2Pos 461

XY scatter plot 51

Y

y-axis, tag cursor 98

YCursor1Color 461

YCursor2Color 462

Z

zoom factor 137

ZoomIn 493

zooming 99
scatter plots 147
trend 99

zooming, in a scatter plot 147

ZoomOut 494

ZoomOutPercentage 462

ZOrder 672

