Hyper Historian
Training Manual

Version 10.50

© 2012 ICONICS, Inc. All rights reserved worldwide. This document may not be reproduced or copied in any manner without written authorization from ICONICS, Inc.

The information contained within this document is subject to change without notice. ICONICS, Inc. does not guarantee the accuracy of the information.

GENESIS64, Hyper Historian, Pocket GENESIS, BizViz and their respective modules, OPC-To-The-Core, and Visualize Your Enterprise are trademarks of ICONICS, Inc.

Windows, Windows XP, Windows 2000, Windows 2003 Server, Windows CE, Windows NT, Windows XP-64, Windows 2003-64, Windows Vista, Windows Server 2008, Windows 7 and Windows Server 2008 R2 are registered trademarks of Microsoft Corp. in the United States and other countries.

Other product and company names mentioned herein may be trademarks or registered trademarks of their respective owners.

Document Number: TM-HH-1050

Visualize Your Enterprise™

ICONICS LICENSE AGREEMENT

YOU SHOULD CAREFULLY READ THE FOLLOWING TERMS AND CONDITIONS!

Opening and using the enclosed software for any purpose indicates your acceptance of the terms and conditions of this license agreement. If you do not agree with them you should return all software, documentation and copy protection keys within seven days of shipment unopened and your money will be refunded.

ICONICS provides this program and licenses its use in the United States, Puerto Rico, or internationally. You assume the responsibility for the selection of the program to achieve your intended results, and for the installation, use and results obtained from this program.

LICENSE

You are granted a personal license to use this program under the terms stated in this Agreement. You may: 1) Install and use the program on a single machine. 2) Make a single archival back-up copy of the program for the sole purpose of supporting your use of the single program on a single machine. 3) You may not use, copy, modify, or transfer the program, or transfer any copy, in whole or in part, except as expressly provided in this license, or with a written contractual agreement with **ICONICS,** Inc.

TERM

The license is effective until terminated. It will terminate if you fail to comply with any term or condition of this Agreement. You agree, upon such termination, to destroy the program and all copies that were made from it and to promptly return any copy protection key(s) to **ICONICS.**

LIMITED WARRANTY

ICONICS WARRANTS THE CD-ROM OR PHYSICAL DISKETTES, ON WHICH THE PROGRAM IS FURNISHED, AND PHYSICAL DOCUMENTATION TO BE FREE OF DEFECTS IN MATERIALS AND WORKMANSHIP UNDER NORMAL USE FOR A PERIOD OF NINETY (90) DAYS FROM THE DATE OF DELIVERY TO YOU AS EVIDENCED BY YOUR RECEIPT. THIS PROGRAM IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION.

ICONICS SPECIFICALLY DISCLAIMS ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, IMPLIED WARRANTIES OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN PARTICULAR, WITH RESPECT TO ANY PARTICULAR APPLICATION, USE OR PURPOSE, IN NO EVENT SHALL **ICONICS** INC. BE LIABLE FOR ANY OTHER COMMERCIAL DAMAGE, INCLUDING BUT NOT LIMITED TO SPECIAL, INCIDENTAL, CONSEQUENTIAL OR OTHER DAMAGES. **ICONICS** DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE PROGRAM WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE PROGRAM WILL BE UNINTERRUPTED OR ERROR FREE.

LIMITATION OF REMEDIES

ICONICS' entire liability and your exclusive remedy shall be the replacement of any CD-ROM or diskette not meeting **ICONICS**' "Limited Warranty" which is returned to **ICONICS** with a copy of your receipt within the warranty period. The remedy for breach of this warranty shall be limited to replacement and shall not encompass any other damages including but not limited to loss of profit, special, incidental, consequential, or other similar claims arising out of the use or inability to use such program even if **ICONICS** has been advised of the possibility of such damages, or for any claim by any other third party.

GENERAL

This Agreement will be governed by the laws of the Commonwealth of Massachusetts. Should you have any questions concerning this Agreement, you may contact **ICONICS** in writing at:

ICONICS, Inc.
100 Foxborough Boulevard
Foxborough, MA 02035

You acknowledge that you have read this agreement, understand it and agree to be bound by its terms and conditions. You further agree that it is the complete and exclusive statement of the Agreement between you and **ICONICS** which supersedes any proposal or prior Agreement, oral or written, and any other communications between you and **ICONICS** relating to the subject matter of the Agreement.

ICONICS RETURN POLICY

All sales are final (NO RETURNS) unless one of the following conditions is applicable: Program is returned in its original packaging material within 7 days after shipment. In such a case, a full refund is provided, less all incurred shipping and handling costs. Program is returned within 30 days from the original shipment date. In such a case a 20% restocking fee is applicable. No returns are allowed on OPC ToolWorX or ActiveX ToolWorX.

Contents

1. Introduction to Hyper Historian

1.1. OBJECTIVES 1-1
1.2. SYSTEM REQUIREMENTS 1-1
1.3. OVERVIEW OF GENESIS64 HYPER HISTORIAN 1-2
1.4. HYPER HISTORIAN EXAMPLES 1-7
LAB: VISUALIZING HISTORICAL DATA L1-1
REVIEW: INTRODUCTION TO HYPER HISTORIAN R1-1

2. Logger Configurations

2.1. OBJECTIVES 2-1
2.2. OVERVIEW OF LOGGERS 2-1
2.3. DISK LOGGING 2-2
2.4. MEMORY PERSISTENT LOGGING 2-5
2.1.1 ARCHIVING 2-8
LAB: LOGGER SETUP L2-1
REVIEW: LOGGER CONFIGURATIONS R2-1

3. Logging Groups

3.1. OBJECTIVES 3-1
3.2. BEFORE YOU BEGIN 3-1
3.3. LOGGING GROUPS 3-2
3.4. COLLECTOR GROUPS IN A LOGGING GROUP 3-4
3.5. HYPER HISTORIAN TAGS IN A COLLECTOR GROUP 3-7
3.6. ORGANIZING TAGS INTO FOLDERS 3-19
LAB: COLLECTOR GROUPS L3-1
REVIEW: LOGGING GROUPS R3-1

4. Aggregate Groups

4.1. OBJECTIVES 4-1
4.2. OVERVIEW OF AGGREGATE GROUPS 4-1
4.3. CREATING AN AGGREGATE GROUP 4-2
4.4. HYPER HISTORIAN TAGS IN AGGREGATE GROUPS 4-2
LAB: AGGREGATE GROUPS L4-1
REVIEW: AGGREGATE GROUPS R4-1

5. Additional Settings

5.1. Objectives 5-1
5.2. Hyper Historian in the Workbench 5-1
5.3. License Mode (Enterprise Edition) 5-1
5.4. Collector Node 5-3
5.5. The Local Logging Server 5-5
5.6. Archive Settings 5-12
5.7. Restoring Archived Files 5-13
LAB: ARCHIVE SETTINGS L5-1
REVIEW: ADDITIONAL SETTINGS R5-1

Appendix A: Hyper Historian SQL Query Engine

A.1. Objectives A-1
A.2. Running Queries against Hyper Historian Data A-1
A.3. Architecture A-2
A.4. Data Architecture A-3
A.5. Supported SQL Constructs A-4
A.6. Specifying a Connection A-11
A.7. Microsoft SQL Server Specifics A-14
A.8. Troubleshooting A-15
A.9. Examples A-16
LAB: HYPER HISTORIAN SQL QUERY ENGINE LA-1

Appendix B: Glossary B-1

Appendix C: Answers C-1

Introduction to Hyper Historian 1

Welcome to training for ICONICS' Hyper Historian™, a stand-alone module of ICONICS' GENESIS64™ Suite. This course will take you through the process for configuring and using Hyper Historian. This training manual includes hands-on exercises that provide opportunities for you to work with Hyper Historian on your own.

1.1. Objectives

After completing this chapter, you should be able to:

- Understand what Hyper Historian can do for you
- Name the Hyper Historian features that most suit your installation
- Start and stop the Hyper Historian service
- View the historical data
- Modify and annotate data

1.1.1. Before You Begin

The material in this course assumes that you are familiar with or have experience with the following:

- 64-bit Microsoft Windows operating systems such as Windows Vista, Windows 7, or Windows Server 2008; you should be able to launch and close applications
- Microsoft Internet Explorer
- The ICONICS GENESIS32 or GENESIS64 suite of software applications, including AlarmWorX and TrendWorX
- Industrial automation devices such as PLCs, controllers, and I/O hardware

1.2. System Requirements

Hyper Historian has the following operating system requirements:

- Microsoft Windows Server 2008 R2 64-bit
- Microsoft Windows Server 2008 SP2 64-bit
- Microsoft Windows 7 64-bit

- Microsoft Windows Vista SP2 64-bit
- Microsoft Windows Server 2003 64-bit*
- Microsoft Windows XP SP2 64-bit*

NOTE: * Hyper Historian leverages several advanced operating system features such as File System Transactions (Transactional NTFS) from the latest Microsoft platforms in order to help ensure the robustness and reliability of logged data. However, these features are not supported on Windows Server 2003 or Windows XP. For that reason, ICONICS strongly recommends that you deploy Hyper Historian applications on one of the latest server platforms from Microsoft for best results.

The minimum requirements for Hyper Historian are based on typical applications settings; depending on your settings, the minimum requirements may vary. Specific hardware requirements depend on the size and complexity of your Hyper Historian implementation. With any application, faster CPUs and increased physical memory (RAM) result in better performance. The minimum requirements for installing Hyper Historian are:

- 1.6 GHz Dual/Multi Core 64-bit processor (AMD Athlon 64 X2, Intel Pentium D, Intel Core 2 Duo, Intel Xeon)
- 4 GB Physical Memory (RAM)
- 160 GB free hard disk space available
- 8x speed DVD-ROM optical drive
- High resolution display at:
    - Minimum: 1024x768, 16-bit color display
    - Recommended: 1280x1024 32-bit color (or better), DirectX 9 Video Card or better
- Microsoft Internet Information Service (IIS) 7.0
- One of the following SQL Server packages installed:
    - Microsoft SQL Server 2005 Workgroup/Standard/Enterprise Edition
    - Microsoft SQL Server 2005 or later Express Edition with Microsoft SQL Server Management Studio Express

1.3. Overview of GENESIS64 Hyper Historian

ICONICS Hyper Historian is a powerful, 64-bit, high performance plant historian, designed for the most mission-critical applications that demand the highest level of availability; it provides unparalleled performance with very efficient use of computer resources.

Hyper Historian uses Swinging Door data compression for high speed data collection of up to 50,000 data events per second on reference hardware for real-time, enterprise-wide information. The Swinging Door algorithm takes advantage of 64-bit hardware and software architectures, allowing it to access more CPU power and memory than traditional 32-bit-based historians and providing the highest performance possible.

Key features include:

- OPC-UA and OPC–DA Compliance
- Industry-standard Data Connectivity (SNMP, BACnet)
- Optimistic Concurrent Multi-user Configuration
    - Web-based Configuration and Administration
    - Unattended Data Archiving and Backup
- 2D and 3D Real-time Charts
- Real-time and Historical Data Replay
- Multiple Remote Data Collectors
- Store and Forward Technology
- Integration with ICONICS BizViz™ Solutions
    - Reporting/Data Bridging
    - Dashboards and Portals
- Alarm Management
- Diagnostic and Data Tracing with NT Event Logs
- Integrated Redundancy

Hyper Historian also has robust, built-in software redundancy for mission critical applications that require uninterrupted access and collection of data. Automatic Store and Forward technology ensures data integrity in the event of system upset or communication disruption.

Hyper Historian uses advanced data integration, providing unsurpassed connectivity to any device via OPC-UA, OPC-DA, OPC XML, SNMP, BACnet, or database values, as shown in Figure 1.1. Users can easily collect information from multiple plants, facilities, or throughout the enterprise. Data sources such as PLCs, I/O devices, HMI applications, and network devices can be collected and stored for reporting and analysis.

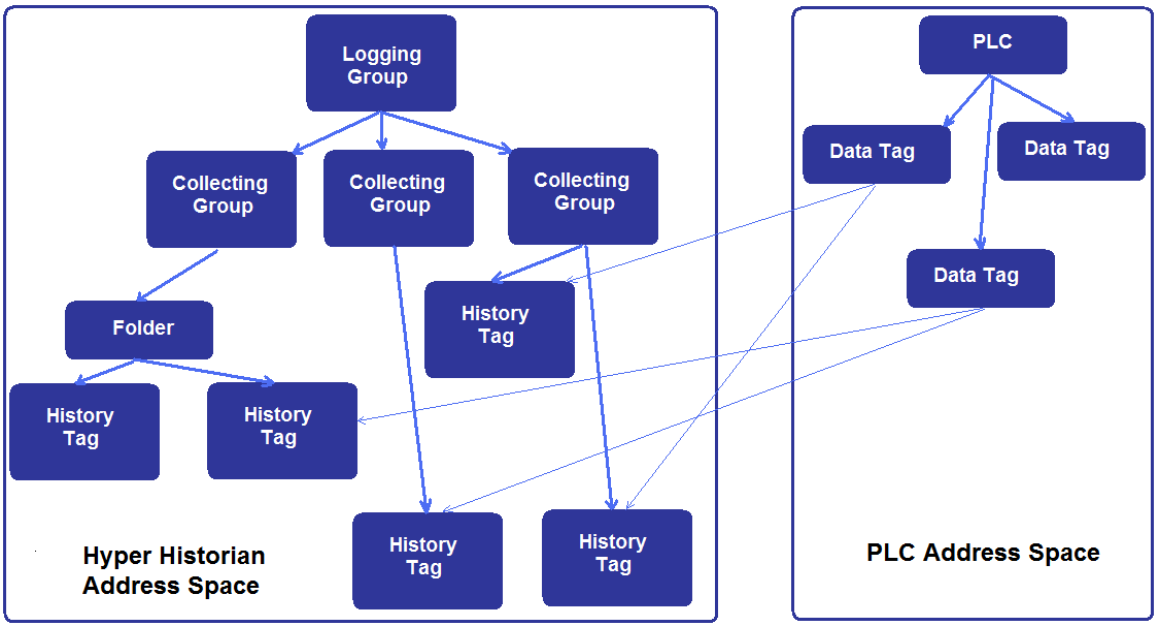


Figure 1.1 - Relationship between Hyper Historian and PLC Data

Figure 1.2 shows the general architecture of the Hyper Historian application.

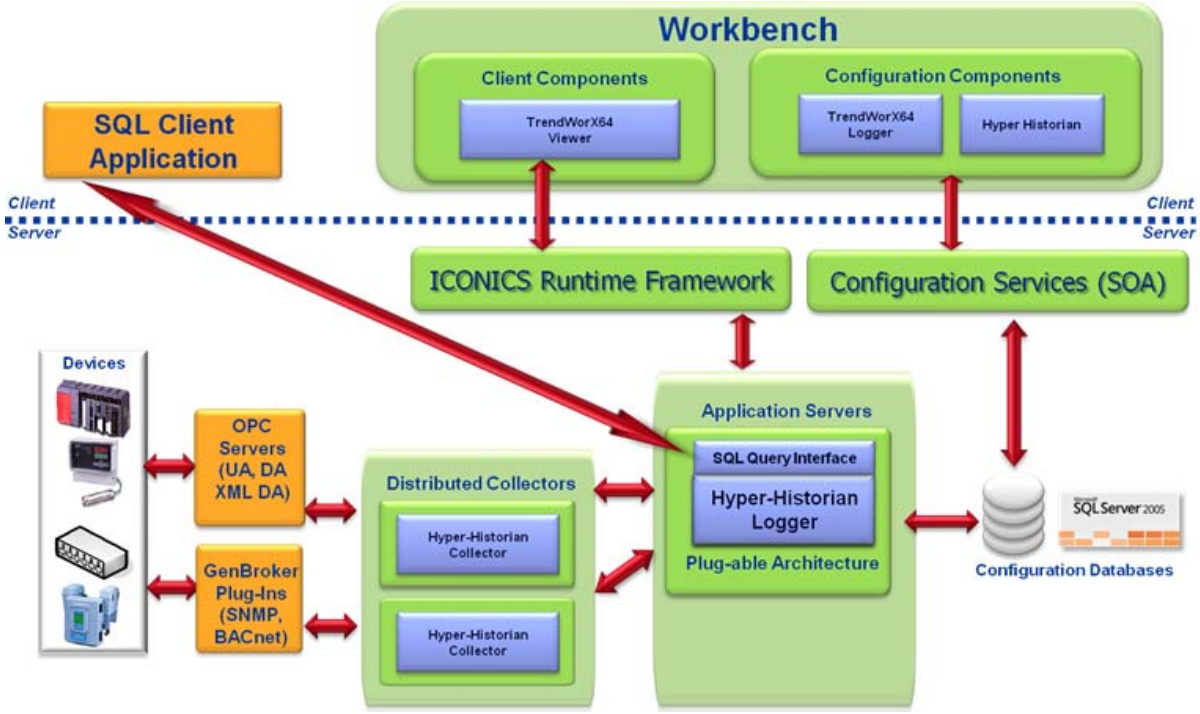


Figure 1.2 - Hyper Historian Architecture

1.3.1. SQL Query Interface

Hyper Historian includes an industry-standard SQL Query interface, enabling tight integration with Microsoft SQL-compatible databases such as Microsoft SQL Server 2005/2008. Hyper Historian also has a unique automation archiving features that allows for routine or triggered scheduling of data archives, freeing up disk space and backing up files for long term storage and/or retrieval.

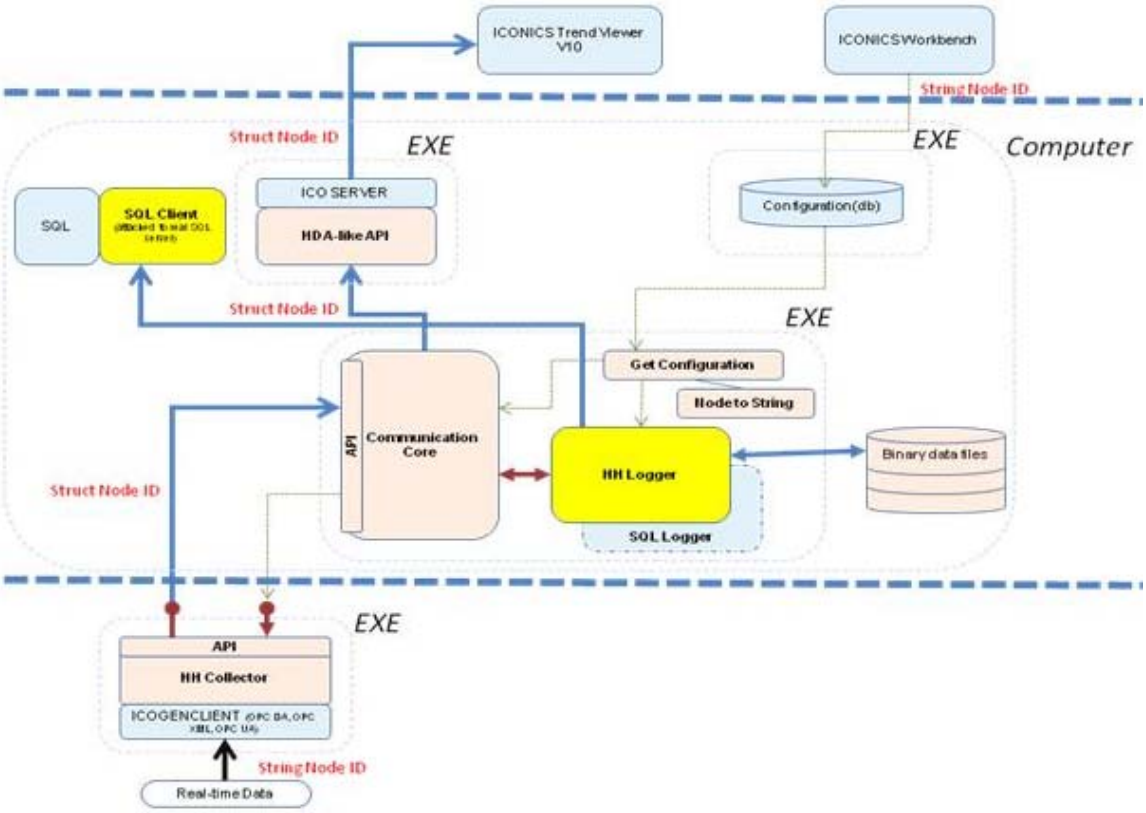


Figure 1.3 - Hyper Historian Schema

1.3.2. The Workbench

The Workbench houses the Hyper Historian Configurator, which features a thin client, optimistic concurrent design, acting as the central configuration environment and operator interface. The runtime operational interface gives you complete visualization of real-time and historical data with 2D and 3D charts. The Workbench's advanced configuration console performs complete service management and has integrated layout, project management, and remote pack-and-go deployment capabilities.

From the Workbench, operators can add fully-customized 3D trends and charts. Users can choose from a wide library of 3D charts such as X vs. Y, logarithmic, bar graph,

strip chart recorder, circular and more, to build clear and accurate representations of real-time and historical data. Intuitive ribbons and galleries help to customize trends by adding color, gradients, smooth animation, translucency/glass effects, anti-aliasing and more, making data analysis clear and straightforward. Users can drag and drop sources during runtime and view multiple trends simultaneously.

Users can trend production numbers against a target. Data can also be exported in tabular formats and it is possible to enter operator comments, as well as manage lab data and audit trails in accordance with 21 CFR Part 11 policies.

1.3.3. Connectivity to BizViz Suite

Hyper Historian also connects to the entire ICONICS BizViz Suite, creating best-in-class reporting, analysis, portal, or data bridging applications. Figure 1.4 illustrates how Hyper Historian integrates with the Workbench, ICONICS BizViz Suite, third-party applications, remote data collectors, and more.

For additional information about ICONICS products, please visit the ICONICS web site at www.iconics.com.

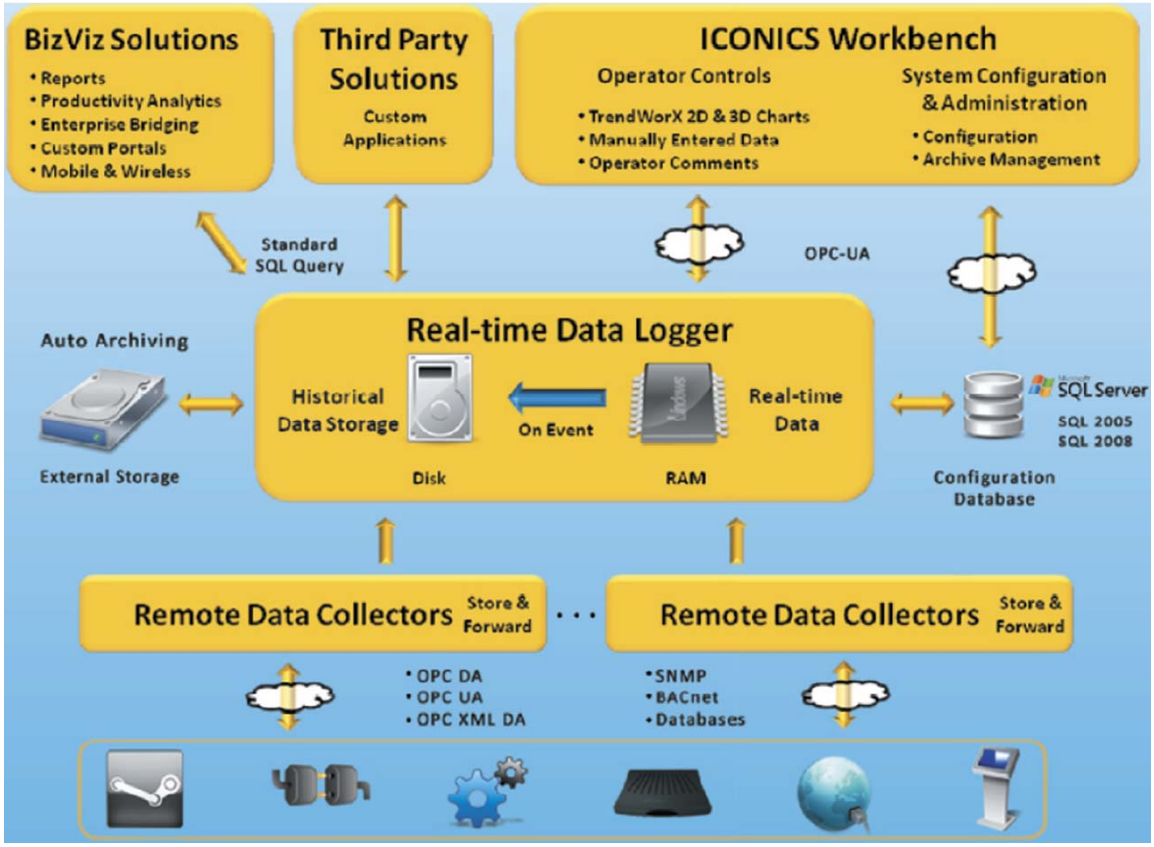


Figure 1.4 - Hyper Historian and Other Products

1.4. Hyper Historian Examples

The Hyper Historian works "out of the box". It has a default configuration that you can use; all you need to do to start logging data is to click the "traffic light" icon in the Home tab of the ribbon. Once you have logged some data, you can view the data using the TrendWorX64 Viewer.

TrendWorX64 Viewer can replay historical data or display real-time data. Note that you can only view data that are still active in the database; you cannot view archived data. If you need to view archived data, you must restore it first before you can replay it.

The following example demonstrates the use of TrendWorX64 and Hyper Historian inside Workbench.

Open the Workbench and click on TrendWorX64. Open a TrendWorX64 container by right-clicking on any node, and selecting **New trend**. Click on the **Configure** button to open the TrendWorX64 Viewer configuration window shown in Figure 1.5. Create a new plot by right-clicking on **Chart** and then selecting **Add Plot**. Then create a new pen by right-clicking on the plot and selecting **Add** > **New Pen**.

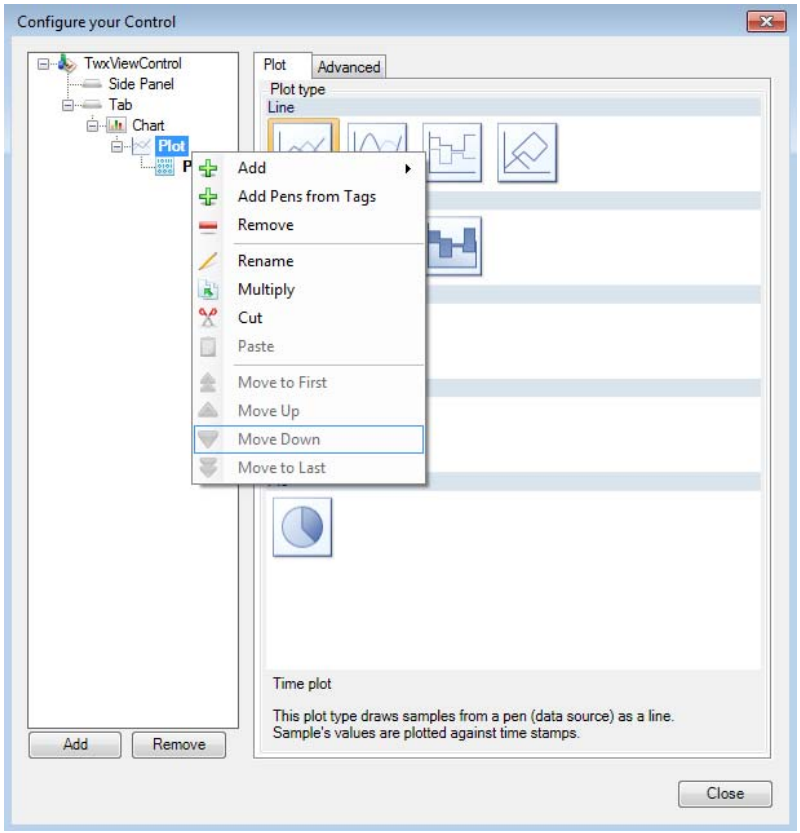


Figure 1.5 - Trend Viewer Configurator

On the right side of the form that appears, enter a **Data Source** or click on the ellipsis button to open the Data Browser, shown in Figure 1.6.

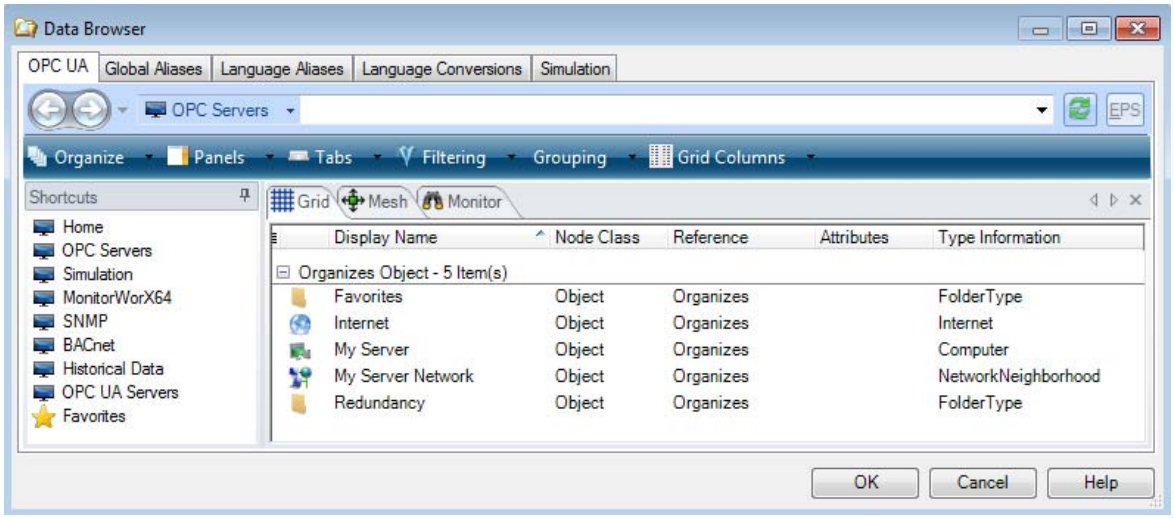


Figure 1.6 - OPC UA Tab

In the Data Browser, click on **Historical Data** in the navigation pane and browse on the right for **Hyper Historian**, as shown in Figure 1.7. Double-click on **Hyper Historian**, then **Logging Group**, then **OPC UA collecting**. For demonstration purposes, within this sample trend example you can double-click **Sine** and then click OK.

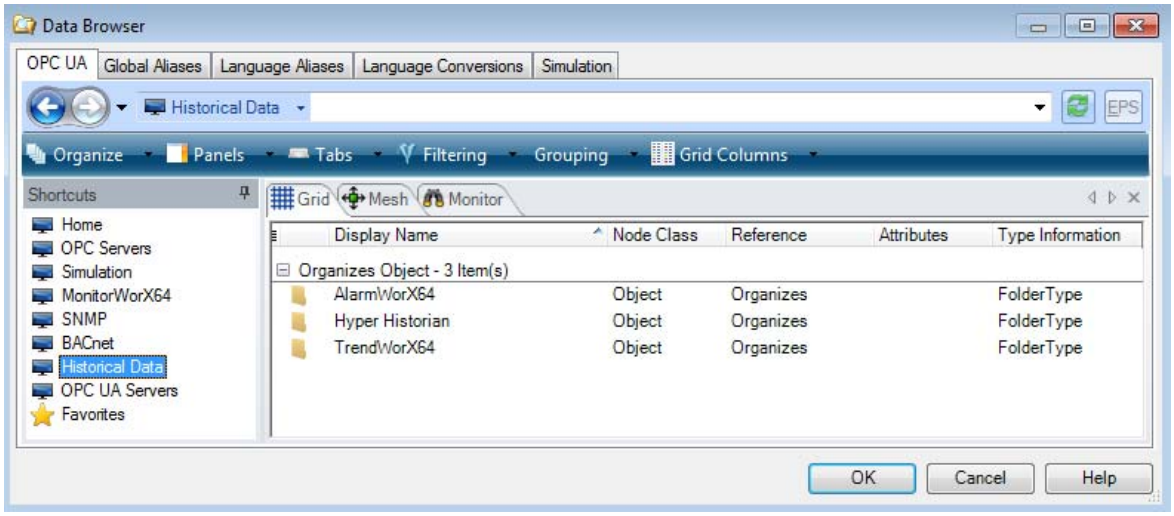


Figure 1.7 - Data Browser

As shown in the Advanced Configuration dialog box in Figure 1.8, the Use HDA Connection option is automatically chosen for you. You can change the aggregate type, indicate whether you want to read actual or modified data, and specify the maximum number of samples. Then click OK to save your changes.

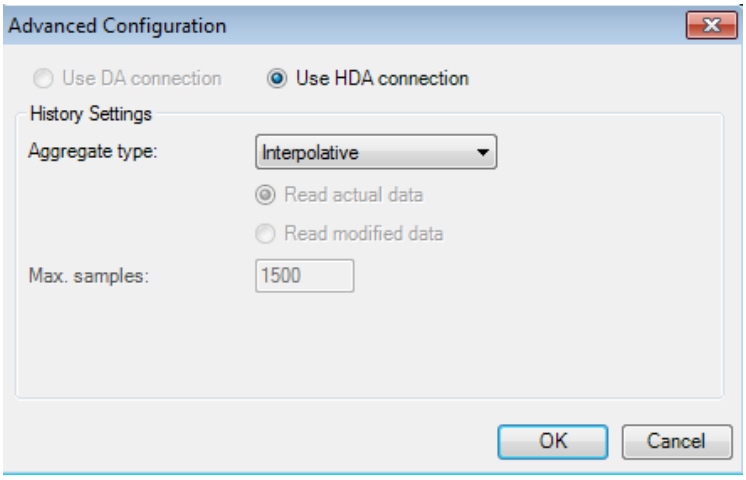


Figure 1.8 - HDA Selection

Back in the Configure your Control dialog box, you can then change the Description – to "Hyper Historian Sine", for example.

Next, click on **Chart** in the navigation pane, and select the **Time and Rate** tab, shown in Figure 1.9. Here, enter your desired time settings (trend and summary periods in days, hours, minutes, seconds and/or milliseconds), whether to use UTC time stamps, and the time and date formats to be used.

Set the Rates settings (display refresh, data collection and history refresh rates in hours, minutes, seconds and/or milliseconds), then click OK. Your Data Collection rate should match those set in the Hyper Historian Logging Group.

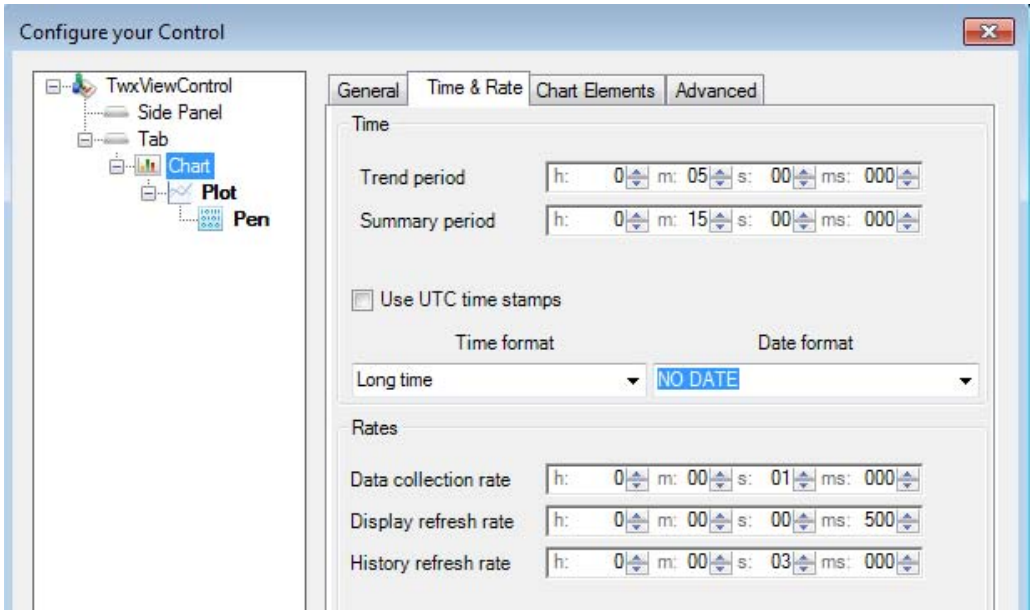


Figure 1.9 - Configuring the Chart

In the Workbench Project Explorer, in the New Trend that you just created, click on **Runtime** and you'll see the trend in motion (like the running sine wave in Figure 1.10).

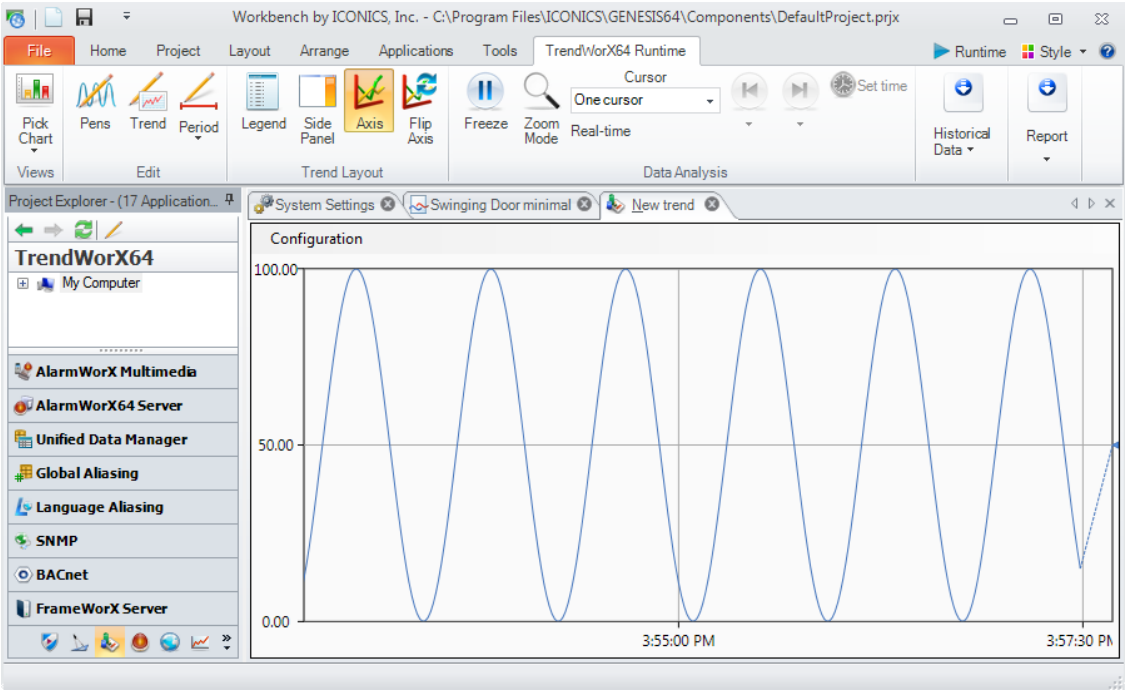


Figure 1.10 - Viewing Historical Data

Within Hyper Historian, you can now double-click anywhere within the viewer to enable the toolbar. Use the toolbar to configure the trend during runtime, change plot types (line, area, bars or pie), edit the trend's visual appearance, configure the timing for trend period and rates, show or hide axes, swap the X and Y axes, zoom within the trend viewer, freeze/unfreeze the trend, or show/hide the legend.

1.4.1. Editing and Annotating Data

You can edit the data that Hyper Historian logs to files. You may want to modify data to correct errors created by a failure or a fault in the system, or you may simply want to manually enter missing data. (The original data will always be available.)

You can enter and modify data in either of the following ways:

- From the Trend Viewer as it plays live or during replay of historical data
- Using the Hyper Historian SQL Interface

Modifications you make are stored with the data in its logged data files; you can replay both the original and modified versions in Trend Viewer.

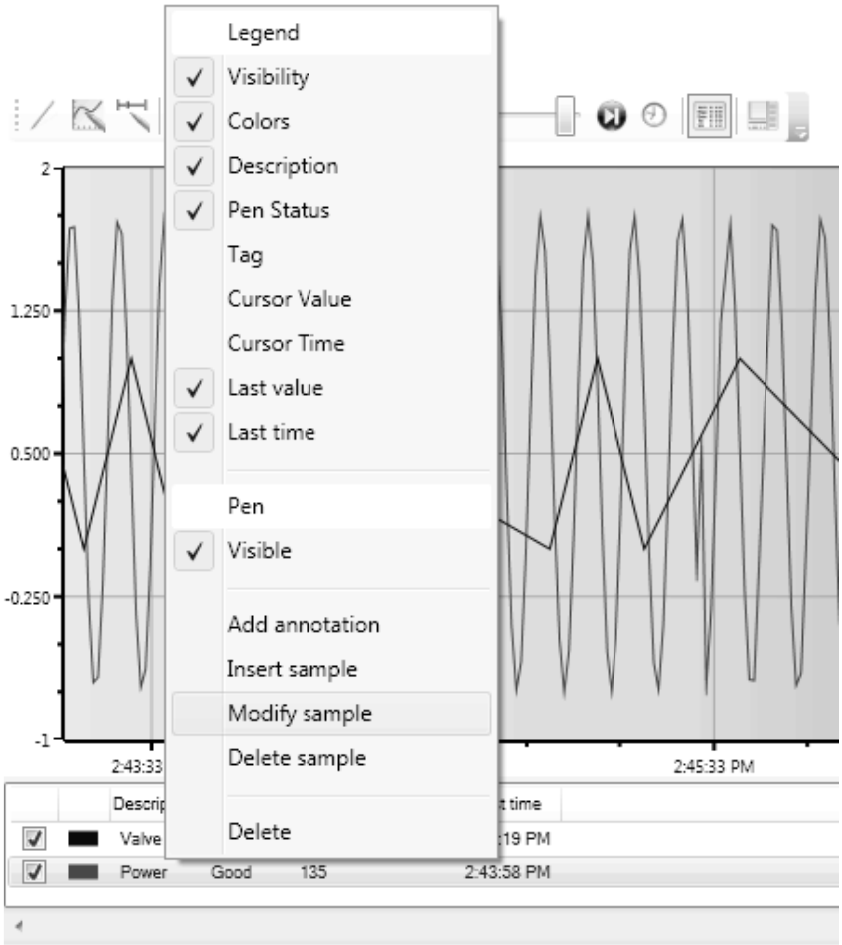


Figure 1.11 - Editing Sample or Adding Annotations

You can also annotate data while in the Trend Viewer, but only if in the Hyper Historian tag configuration the Support Operator Annotations checkbox has a checkmark in it. Annotations act like post-it notes, letting you mark or label the data, document an event, or flag a trend.

Each annotation appears in the Trend Viewer as a text box when the runtime is in freeze mode. When Show Operator Comment is enabled annotations are saved with the data in the logged data files and can be accessed using the Hyper Historian SQL Interface.

. Lab: Visualizing Historical Data L1

In this section, you will explore GenDemo samples to see the ways to visualize your data. You will learn where to start and configure Hyper Historian engine, and finally you will find out how to visualize your historical data using TrendWorX64 Viewer.

Estimated time to complete this lab: 20 minutes

1.1. Task Checklist

In this lab, you will complete the following tasks:

- Explore GenDemo
- Discover where to configure, and how to start and stop Hyper Historian
- Connect your TrendWorX64 Viewer to Hyper Historian server

1.2. Support Files

You do not need any supporting files to complete this lab.

1.3. Exploring GenDemo

There are many helpful displays in GenDemo that can provide examples when you need ideas or samples for modeling your own displays. In this section, we will explore a few of GenDemo's displays.

1. Open GenDemo by going to Start → Programs → ICONICS GENESIS64 → Tools → Edit GenDemo. We will need to open this version of GenDemo in order to go into configuration mode.

2. Once the display opens, click on the gears icon in the top-right toolbar. This opens the Carousel play list. You can pause the play list if you wish.

3. Scroll down the play list and find the display files whose names start with *TWX64*. Each of these displays provides information about TrendWorX64 Viewer. For Hyper Historian, we will want to look at the last five *TWX64* displays.

Double-click any of these displays to open it in the TrendWorX Viewer. Each display has features for making data accessible in a format that suits you best.

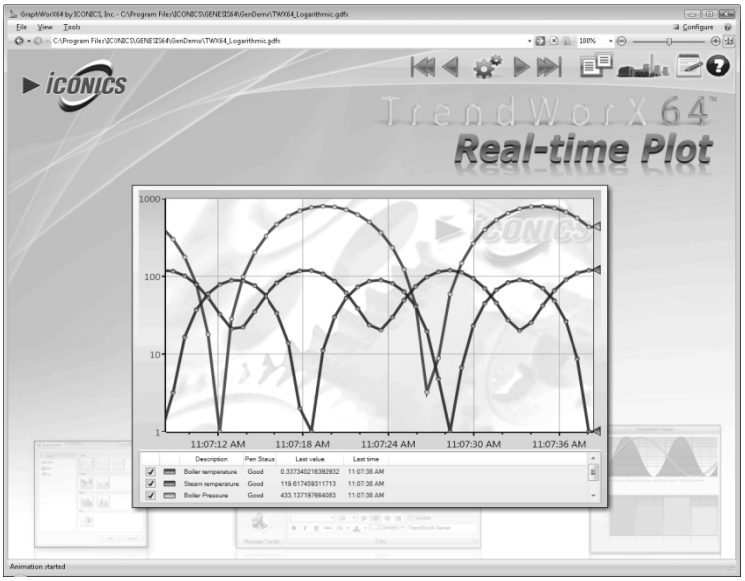


Figure L1.1 - GenDemo TrendWorX64 Viewer Example Display

4. Starting with TWX64_Logarithmic.gdfx, open the last five TWX64 displays and explore the TrendWorX Viewer. These GenDemo displays show *real-time* data. You should note that *historical* data is shown in a similar way.

5. You may wonder how the displays were configured so that you can put some of the same features to use in your own displays. So let's take a look at some of their configurations. Close the Carousel application; this prompts you to stop the Carousel, so go ahead and stop it. The Carousel application closes and the TrendWorX64 Viewer opens in a GraphWorX64 display in Configuration mode. Here you can double-click on the TrendWorX64 Viewer to see its configuration.

   While you are in GraphWorX64, you can click the **Runtime** button located in the top-right corner of the window to view the display in runtime again, and then click on the **Configuration** button (located in the same position) to return to configuration mode.

1.4. Managing Hyper Historian Logging

To store real-time data using Hyper Historian, you need to know where to start the Hyper Historian service and how to maintain the logging properties.

Since the Workbench acts as a centralized location for accessing most of the applications, we will use it throughout this course to manage our training course project.

1. Open the Workbench from Start → Programs → ICONICS GENESIS64 → Workbench.

2. To open Hyper Historian, click the Hyper Historian icon in the Project Explorer's bottom tray.

3. In the Tree control, expand the localhost node to show all available configurations on this node.

   NOTE: If you connect to a remote server, you will see the remote server's name instead of "localhost".

4. If you expand any of the available configurations, you will see the basic structure of a Hyper Historian configuration, which contains four nodes: Data Collections, Loggers, Node Setup and Redundancy, and System Administration. We will discuss these nodes later in this course.

5. For now, we will focus on the Start/Stop button in the Home ribbon. Clicking this button starts and stops the Hyper Historian service. When the traffic light is red, data is not logging; so click the button to make the light turn green. When it turns green, data is logging. In the next section, we will connect to Hyper Historian and visualize the data that we have just started to log.

Figure L1.2 - Left to Right: Started (Green), Unknown (Yellow), and Stopped (Red)

1.5. Hyper Historian Data Connectivity

We will now use TrendWorX64 to connect to historical data logged by Hyper Historian to see how you can view and control data within the application.

1. If you have closed the Workbench since the last section, open it from Start → Programs → ICONICS GENESIS64 → Workbench.

2. Click on the Applications ribbon, then click the TrendWorX64 button in the Client section, shown in Figure L1.1. Doing this opens TrendWorX.

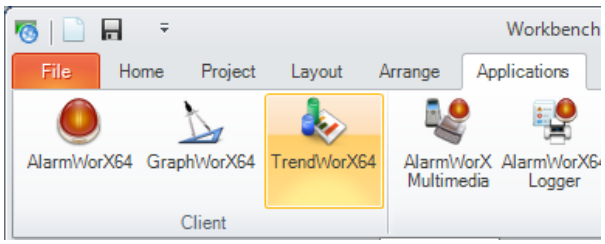


Figure L1.3 – Opening TrendWorX

3. Now go to the Home ribbon, and click the New Trend button, shown in Figure L1.4. Notice how doing this opens a New Trend tab in the work area.

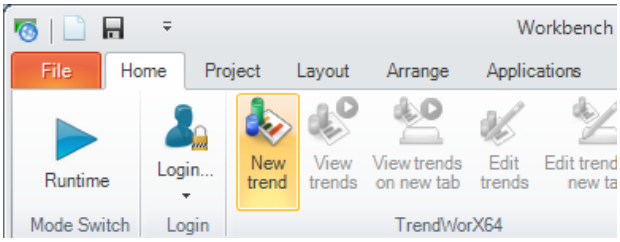


Figure L1.4 - Adding a New Trend

4. To configure the trend, in the TrendWorX64 Viewer Configuration ribbon that appears click the Configure button. A configuration dialog box opens, like the one shown in Figure L1.5.

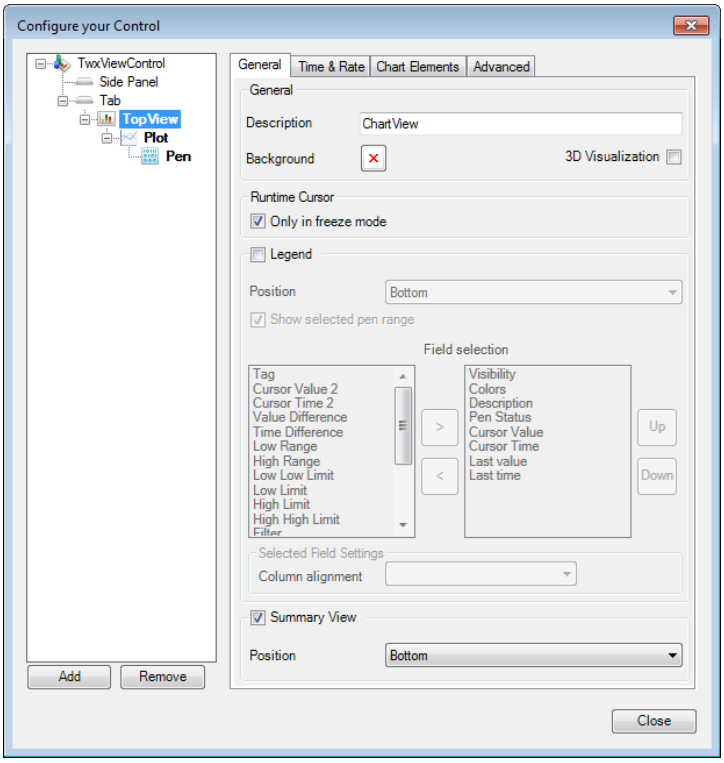


Figure L1.5 - TrendWorX64 Viewer Configuration Dialog

5. On the left of this dialog box is a tree control that allows you to create a sensible structure of your data distributed into various chart views, plots, and tabs. Notice in this tree structure that you can also add a side panel to your TrendWorX display. First, click on the Chart item and rename it to “Top View”.

6. Then, right-click the Top View chart and select Add, then Plot.

7. Click on the Plot item and change its name to “My First Plot.”

8. Next, add a Pen item by right-clicking My First Plot, then selecting Add, then Pen. On the right side of the dialog box, click the Data Source ellipsis button to open the Data Browser. In the Data Browser, make sure you are on the OPC UA tab and click on Historical Data on the left under Shortcuts; you should see something like the window shown in Figure L1.6.

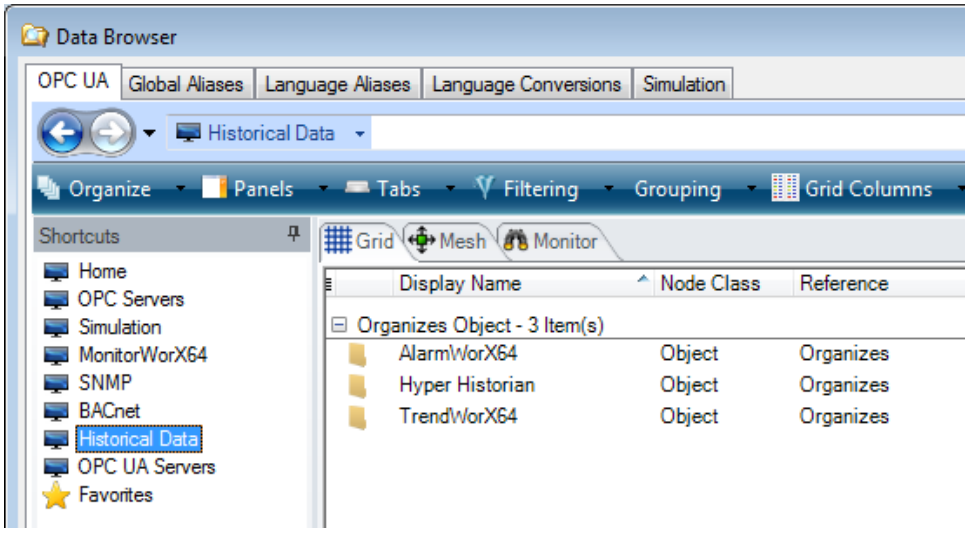


Figure L1.6 – On the OPC UA Tab, Click Historical Data

9. On the right side, double-click Hyper Historian → Logging Group → OPC UA collecting. For demonstration purposes in this trend example, we are going to double-click on Sine. So go ahead and click on it. The Browser will look something like the example shown in Figure L1.7.

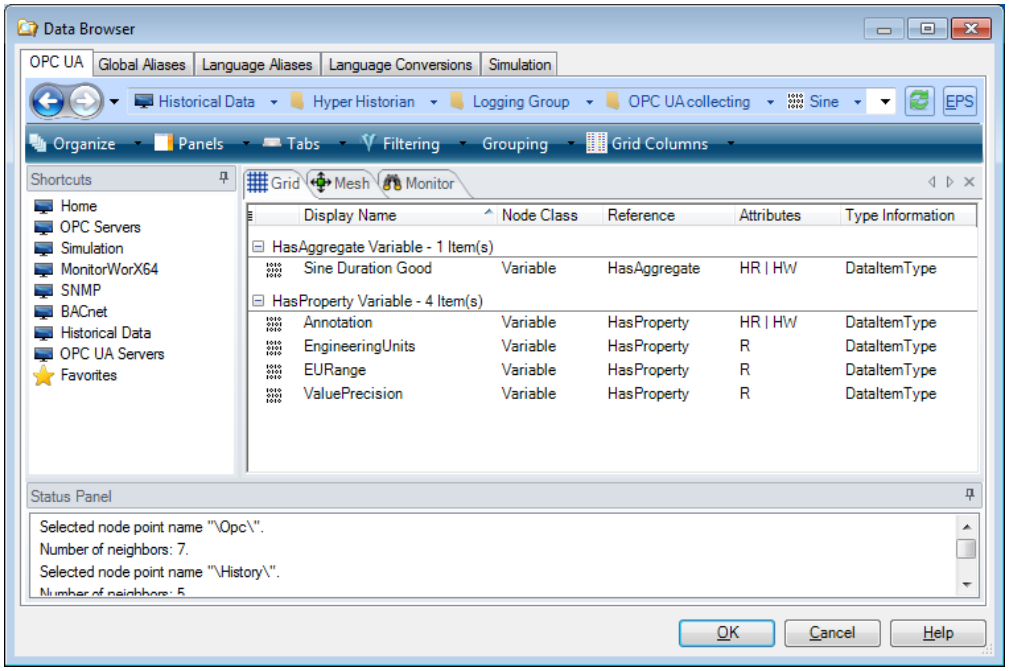


Figure L1.7 - Data Browser

10. Click OK to select Sine as the data source.

11. Click the pen's Advanced Configuration button to open the dialog box shown in Figure L1.8, and select the Use HDA Connection option. Then click OK. (By default this is set automatically when you select Historical data.)

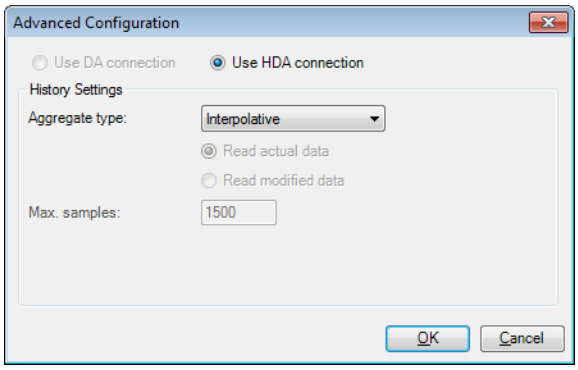


Figure L1.8 - HDA Selection

12. Right-click on Tab in the tree control and select Add, then Chart. Select the chart in the tree control and rename it to “Bottom View”.

13. Put a checkmark in the BottomView chart’s 3D Visualization checkbox, illustrated in Figure L1.9, to show data in the new chart view in 3D mode.

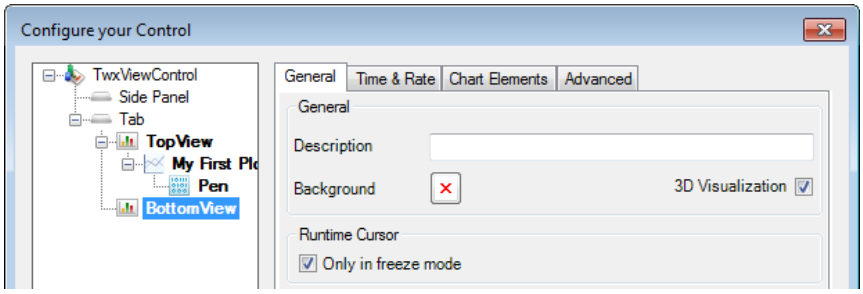


Figure L1.9 - Turning on 3D Visualization

14. Now add a Plot and a Pen to the new chart, just like you did a few minutes ago for the first chart. Then rename the new Plot to “My Second Plot”.

15. Turn this plot into an Area plot by selecting the first Area picture as shown in Figure L1.10.

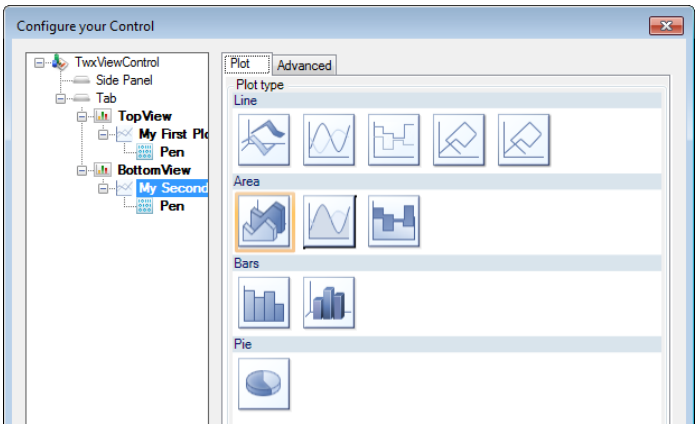


Figure L1.10 - Changing Plot Type

16. Now repeat the process of selecting a data source for the New Pen as described a moment ago, but this time add the Random signal.

17. When you finish, click Close to close the configuration dialog and save the configuration.

18. In the Workbench Project Explorer, in the New Trend that you just created, click Runtime and you'll see the trend in motion. An example is shown in Figure L1.11.

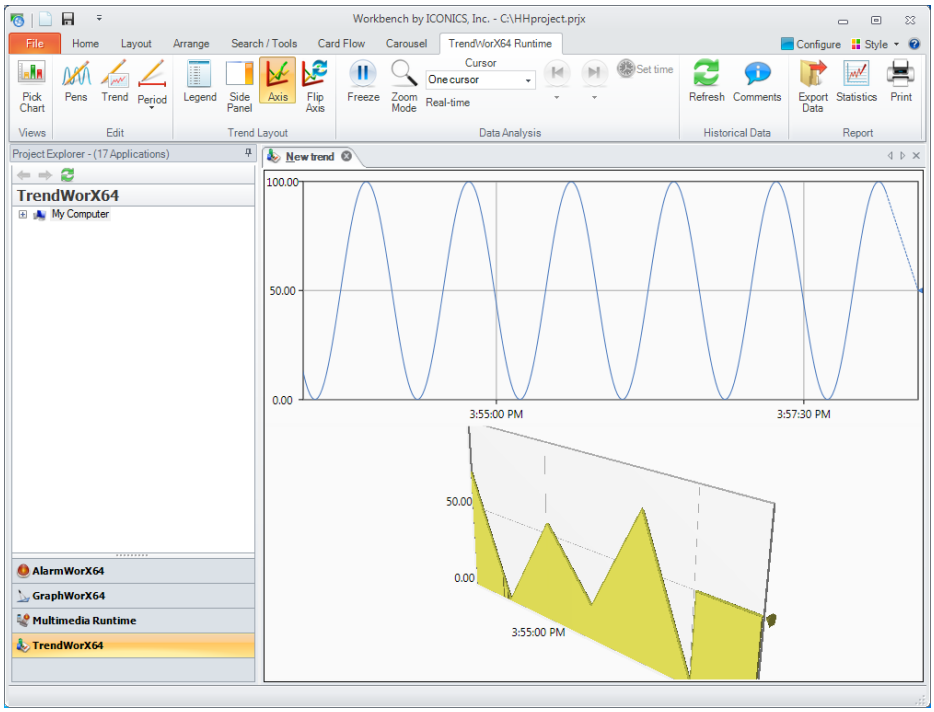


Figure L1.11 - Viewing Historical Data

19. From the TrendWorX64 Runtime ribbon (shown at the top of the Workbench in Figure L1.11), you can configure the trend during runtime, change plot types (line, area, bars or pie), edit the trend's visual appearance, configure the timing for trend period and rates, show or hide axes, swap the X and Y axes, zoom within the trend viewer, freeze and unfreeze the trend, or show and hide the legend.

Review: Introduction to Hyper Historian

R1

1.1. Chapter Objectives

You should now know and understand the following topics:

- Understand what Hyper Historian can do for you
- Name the Hyper Historian features that most suit your installation
- Start and stop the Hyper Historian service
- View the historical data
- Modify and annotate data

1.2. Summary

ICONICS Hyper Historian is a powerful, 64-bit, high performance plant historian designed for mission-critical applications that require uninterrupted access and collection of data.

Hyper Historian works out of the box. You can start data collecting after installation and configure the TrendWorX64 Viewer to experience its power immediately. You can use the TrendWorX64 Viewer to annotate data while in the Trend Viewer if, in the Hyper Historian tag configuration, the Support Operator Annotations option is enabled.

Hyper Historian also connects to the entire ICONICS BizViz Suite, creating best-in-class reporting, analysis, portal, or data bridging applications. An industry-standard SQL Query interface provides integration with Microsoft SQL-compatible databases.

Hyper Historian provides redundancy for mission critical applications. Store-and-Forward technology ensures data integrity in the event of system upset or communication disruption.

Archiving allows for routine or triggered scheduling of data archives. This frees up disk space and backs up files for long-term storage and/or retrieval.

1.3. Terms

BizViz Suite	Hyper Historian	Hyper Historian SQL Interface
OPC DA	OPC HDA	OPC UA
TrendWorX64 Viewer	Workbench	

1.4. Test Your Knowledge

1. True or False? Hyper Historian can compress data.

2. What is the name of the ICONICS tool that acts as the central configuration environment and operator interface?

    a. Hyper Historian Configurator
    b. Workbench
    c. GraphWorX64
    d. ICONICS Central Configurator

3. True or False? Hyper Historian can connect to OPC DA interface.

4. True or False? You can start and stop the Hyper Historian service inside the Workbench.

5. True or False? There is a limit to the Hyper Historian data you can view in TrendWorX64 Viewer.

6. True or False? You can add operator comments to your historical data.

7. What tool can you use to modify historical data for Hyper Historian tags?

    a. Hyper Historian Data Modifier
    b. SQL interface
    c. TrendWorX64 Viewer
    d. All of the Above
    e. A and B
    f. A and C
    g. B and C
    h. None of the above; you cannot modify logged data

Logger Configurations 2

Hyper Historian allows you to create multiple **Loggers** based on the data you are collecting. Once created, these loggers can be organized into **Logging Groups**. In addition, **Collector Groups** can be used for collecting values from multiple tags.

For each Logger that is created, the Hyper Historian creates a series of binary files to hold the Plant Historical data, which can be archived on the basis of configuration options that you set.

2.1. Objectives

After completing this chapter, you should know how to:

- Identify the uses of the two types of loggers
- Set up a Disk Logger configuration that saves collected data to a hard drive
- Set up a Memory Persistent Logger configuration that collects data in RAM
- Store and archive logged data

2.2. Overview of Loggers

A logger is the means you use for grouping tags that collect similar types of data. Let's take a look at what types of data we are talking about.

2.2.1. Grouping Tags by Logger

For each logger configuration, you determine a specific length of time over which data is to be logged and stored for the tags that are grouped in the logger. For this reason, before you even begin to create your loggers, you need to come up with a plan for the data you are collecting. You will be creating one or more loggers, and each will collect data from a set of tags for a pre-determined length of time before deleting that data or offloading using an archival process. It is your job to determine how long you need data to reside online, and to create loggers based on that need.

To do this, you should analyze the data you will be collecting to determine how long you want the data to be kept online in the Hyper Historian. Data that must be kept online for *longer* periods of time should be grouped together in one Logger.

Data you want kept online for *shorter* periods of time should be grouped in a second Logger, and so on. Once this analysis is done, you can proceed.

2.2.2. Two Types of Loggers

In Hyper Historian, there are **two types of logger configurations** that govern how data gets logged to the Logger. They are *Disk logger* configurations and *Memory-Persistent logger* configurations. You can create any number of either type of logger. Data stored with either type of logger is available for replay and reporting for a timeframe of your choosing. Stored data is automatically archived at a point determined by you, and can be easily restored from archives.

A Disk Logger collects plant data and saves it to a hard disk; this is the type of logger that is used most often. This type of logging can be used to collect:

- plant data for use in historic replay with TrendWorX64,
- plant data you need to collect for reports created using BizViz products,
- plant data you need to archive for regulatory compliance, and
- plant data you want to save for future reference and analysis.

A Memory-Persistent Logger collects plant data, stores it in RAM, and discards it after a pre-configured amount of time. Use a Memory-Persistent Logger for tags whose values you may not necessarily need to save, but whose previous values you would like to see in the viewer when going in and out of runtime.

A helpful feature of the Memory-Persistent Logger is that it can be configured to store a portion of its collected data to the hard disk when certain events occur; the data is stored in real time. This feature lets you troubleshoot events by collecting and saving high-resolution data surrounding problematic events.

Let's look at each type of logger configuration more closely.

2.3. Disk Logging

A Disk Logger collects plant data that it saves to a hard disk.

To add a new Logger configuration, right-click on **Loggers** in the navigation tree and select **Logger**. The Hyper Historian Logger configurator appears on the right. Provide a name for the logger configuration; you should make the name descriptive enough so that you understand the logger's use and content.

From the **Logger Type** pull-down list, select **Hyper Historian Disk Logger** and the configuration form is instantly generated to look like Figure 2.1.

If you choose to create a Disk File Logger, you will need to use the File Storage Information section, also shown in Figure 2.1, to specify the conditions under which you want the Hyper Historian to create a new data file and close the current, existing file. Let's look at the options that are available to you.

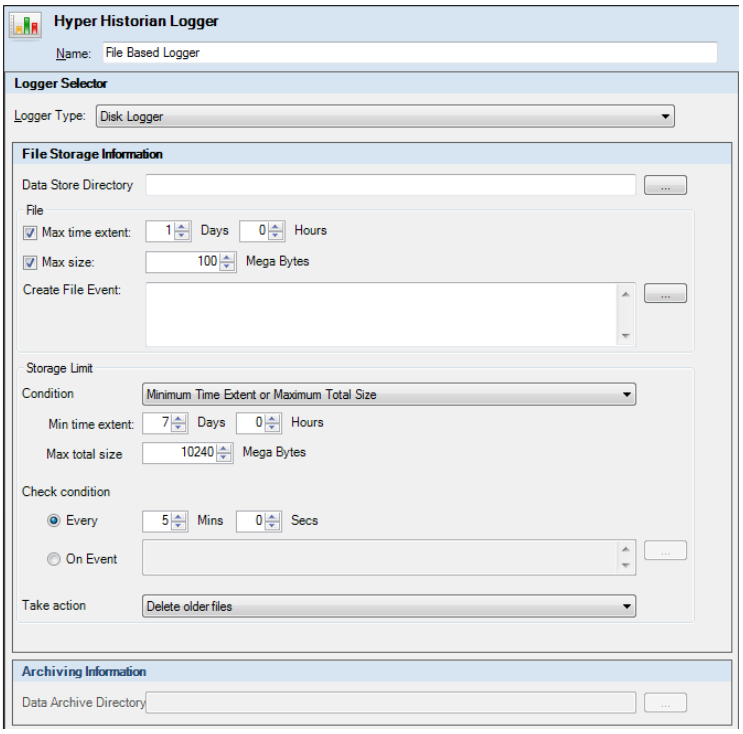


Figure 2.1 - Disk Logger Configuration

2.3.1. File Storage Information

It is with the File Storage Information options that you control the amount of plant data or the time frame for the plant data that should be logged to each file. The conditions can be based on a time period, a file size, or a Unified Data Manager triggered event. As soon as the configured conditions are reached, the current file is closed (but remains online for data retrieval) and a new file is created.

In the Data Store Dictionary field you specify the path where data files will be stored. You can click the ellipsis button to navigate to the location. Do not enter a file name because files are named automatically using the following format:

loggerConfigName_startDate_startTime_endDate_endTime

> **NOTE:** Dates are in *yyyymmdd* (year, month, day) format, and times are in *hhmmss* (hour, minute, second) format.

Next, you must specify how long the file will be held open for recording data. You can specify a timeframe, a file size, or an event that determines when the file will be saved. It is probably easiest to specify how frequently you want to save a file using the **Max Time File Extent** days and hours options. If you do not care how large the file gets, in the **Max File Size** field you can specify a value of 0 (zero) to allow the file to grow unlimited. However, if you want to limit the size of the file, specify the size limit. The file will be saved when the time limit or the file size is reached; that is, unless you specify an event in the **Create File Event** box.

Optionally, you can create a trigger that saves the file. For example, if you are collecting data from a machine that is rarely used, you might want to save the current file and create a new one when the machine is shut off by an operator. In the **Create File Event** box, type the specific path for the OPC-UA name for the event trigger or click the ellipsis button to browse for the event trigger. The Unified Data Manager opens and allows you to schedule file closing based on date, time, or an external event. Select from the data- or time-specific triggers. For instance, a time trigger may be “Five Minute On/Off Conditional” or “One Shot”, etc.

> **NOTE:** Event triggers are created and maintained in the Unified Data Manager (UDM) application. For more information on the UDM, please refer to the UDM help files.

2.3.2. Storage Limits for Archiving or Deleting Older Files

Once a data file is closed, it can remain on the system and be a data source for the runtime trend display and for SQL queries. Eventually, the stored data will have to be detached from the system and moved to external storage to make room for new incoming plant data. The process of detaching the older data and moving it is called *archiving*. However, you don’t have to archive files at all – you can choose to simply delete them. Remember, however, that if you archive files, you will have the option of restoring them at any time to replay them; if you delete them, their data cannot be restored.

In the **Storage Limit** area of the logger’s File Storage Information, specify how long the configuration’s saved files are to be kept for retrieval before they are archived or deleted. Archival or deletion occurs automatically at the time you configure it to happen.

It is with the **Take Action** option near the bottom of the form where you specify whether older files are **archived** or **deleted**.

If you choose to **Archive Older Files**, then you must also specify a path for the archival location in the **Archiving Information** section; type the path or click the ellipsis button to navigate to and select the location.

You can create a configuration to archive or delete files based on:

- **When the files have reached a certain age and/or are of a particular size.** You can use the **Condition** options to limit files to a particular age or size. From the **Condition** drop-down list, specify whether you want to use the file's age or size, or specify that you want to use both. In **Min Time Extent**, specify the age (in days and hours) at which a file is to be removed. In **Max Total Size**, specify a file size that qualifies a file for immediate removal. Be careful when using this feature because it can mean that as soon as a file of this size is saved, it is removed.
- **The condition to check files for archiving is working on a regular schedule.** To remove files on a routine schedule, use the **Check Condition** options. Specify the frequency at which files are to be removed by selecting **Every** and then specifying the frequency in minutes and seconds.

Alternatvely, you can use a trigger that, for example, archives every Saturday at 3:00 AM. Once you have created the trigger, select the **On Event** option, and specify the trigger in this option's box, either by entering a path and the trigger name or by clicking the ellipsis button to select the trigger file. If you use this option, it is used together with the other archiving options, described next.

> **NOTE:** Event triggers are created and maintained in the Unified Data Manager (UDM) application. For more information on the UDM, please refer to the UDM help files.

-

Until a file is archived or deleted, it is available to reports and charts in your ICONICS system; once it is removed in one of these ways, it is no longer available for that purpose. You can, however, restore an archived file to the system (see "**5.7 - Restoring Archived Files**" for more information on how to reconnect to archived files).

2.3.3. What Do I Do Next?

After you create a Disk Logger configuration, you can create *logging groups* for it. In the next chapter, we will look at logging groups. But first, let's look at the other type of logger, the Memory Persistent Logger.

2.4. Memory Persistent Logging

A Memory Persistent Logger collects plant data that it stores in RAM and discards after a pre-configured amount of time. Using a Memory Persistent Logger has its advantages in that you can collect more granular data (and, therefore, a greater volume of data) because you are not saving all of it. Should you choose to save data surrounding a specific event, you can do this while capturing a high resolution of data that helps you analyze the moments up to and after the event.

To add a new Logger configuration for a Memory Persistent Logger, right-click **Loggers** in the navigation pane, and then select **Logger**. The Hyper Historian Logger configurator appears on the right. Provide a name for the logger configuration, making the name descriptive enough so that you understand its use and content. Then, from the **Logger Type** pull-down list, select **Memory Persistent Logger** and the instantly-generated configuration form will look like Figure 2.2. Let's look at the options that are available for a Memory Persistent Logger.

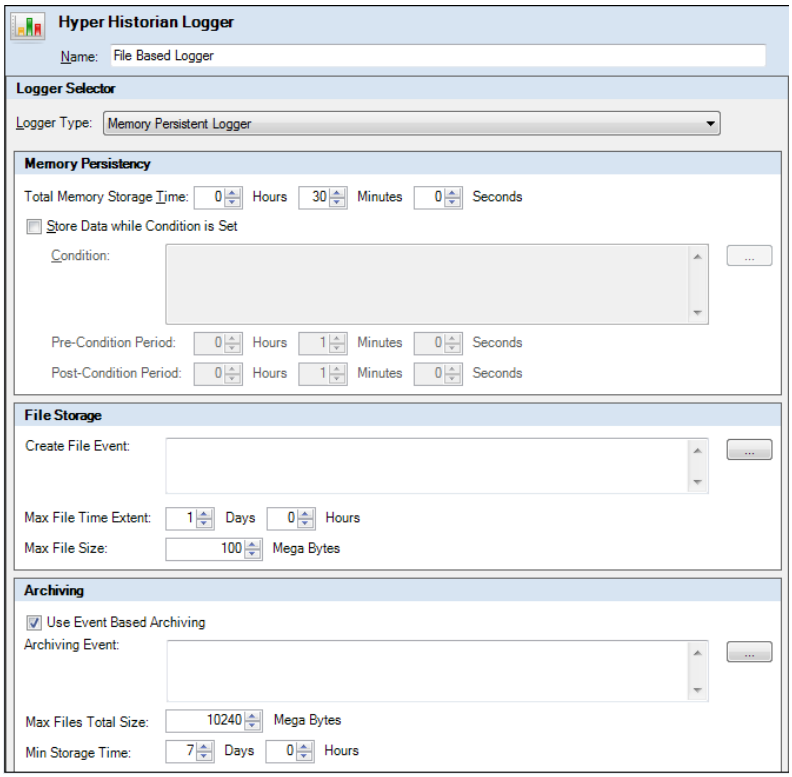


Figure 2.2 - Memory Persistent Logger Configurator

2.4.1. Memory Persistency

In the **Memory Persistency** section, you must specify the amount of data you want the Hyper Historian to collect into RAM. First, let's look at the **Total Memory Storage Time**, which is the amount of time you want the logged data to be kept in RAM. When this time expires, the data is discarded from memory. You will never get more data than the data that is collected in this about of time. In deciding how long to keep data in RAM, you should consider the following:

First, consider the *types of tags* that data will be collected from, the *volume of data* you will be collecting from these tags, and the *amount* of memory data collection from these tags will occupy. The tags you collect data from may determine if you want to create more than one logger configuration. For example, for slowly changing values of troublesome tags whose events you are trying to eliminate, you may create one logger configuration that keeps data in memory for a day. But for other troublesome tags that collect values that are more stable, you might create a second logger configuration that keeps data for an hour.

Second, keep in mind how *much data you want saved in a file if an event occurs*. This is described in detail for the Store Data While a Condition Is Set options, next.

2.4.2. Store Data When a Condition Is Set

Technically speaking, **Total Memory Storage Time** is all that is needed to create a Memory Persistent Logger configuration. However, if you want persistent trend data to be stored to the disk when an event happens, select the **Store Data While a Condition Is Set** checkbox. Using this feature lets you save the data that was collected before and after abnormal or infrequent events and store it for analysis.

In the **Condition** text box, type the event trigger or click the ellipsis button to browse for it. Note that by using a condition you are creating a configuration that looks for a very specific event; therefore, it is likely you are creating the logger configuration for use with a particular tag or type of machine. This is why you should make sure you name the configuration in a way that reflects its particular use.

Once you specify the event, you can set the **Pre-Condition** and **Post-Condition Periods** (in hours, minutes or days) to be captured along with the event itself. Capturing the pre- and post-condition periods along with the event will shed light on the circumstances surrounding the event for analysis later, so you should make sure the timeframe is long enough to be of value later but not so long that repeated captures of data overburdens your hard drive.

When using this feature, you must also make sure that the timeframe you specify for the **Total Memory Storage Time** is long enough to hold the data to be captured and saved. For example, if you want to save two hours of data prior to the event, you must hold data in memory for at least two hours.

2.4.3. File Storage and Archiving

The Memory Persistent Logger Configurator has File Storage and Archiving that is similar to, but not the same as, File Storage and Archiving for the Disk Logger Configurator. For a Memory Persistent Logger, use the **File Storage Information** section to specify how long the file will be held open for recording data when the specified trigger event happens. You can specify a timeframe, a file size, or an event when the file will be saved; as soon as one file is saved, a new one is started the next time the event occurs. Files are named like this:

loggerConfigName_startDate_startTime_endDate_endTime

NOTE: Dates are in *yyyymmdd* (year, month, day) format, and times are in *hhmmss* (hour, minute, second) format.

It is probably easiest to specify how frequently you want to save a file using the **Max Time File Extent** days and hours options. If you do not care how large the file gets, in the **Max File Size** field specify a value of zero to allow the file grow unlimited. However, if you want to limit the size of the file, specify the size limit. The file will be saved when the time limit or the file size is reached; that is, unless you specify any event in the **Create File Event** box. You can create a trigger that saves the file; for example, if you are collecting data from a machine that is rarely used, you might want to create a new file when the machine is shut off by an operator. In the **Create File Event** box, enter the OPC-UA name for the event trigger into the text box or click on the ellipsis ... button to browse for it.

NOTE: Triggers are created and maintained in the Unified Data Manager (UDM) application. For more information on the UDM, please refer to the UDM help files.

2.4.4. Archiving

In the **Archiving** area, specify how long the configuration's saved files are kept for retrieval. Archiving occurs automatically at the times you configure it to happen. You can configure to archive files using either of the following criteria:

- **On a regular schedule.** To do so, you will need to create a trigger that, for example, archives every Saturday at 3:00 AM.
  Once you have created the trigger, put a checkmark in the **Use Event Based Archiving** checkbox, and specify the trigger in the **Archiving Event** box, either by entering a string or clicking on the ellipsis button to select the trigger file. If you use this option, it is used in conjunction with the other archiving options, described next.
- **When the files have reached a certain age or are of a particular size.** In **Min Storage Time**, specify how long files must be in the system (in days and hours) before they can be archived. In **Max Files Total Size**, specify a file size that qualifies a file for immediate archival. Be careful when using this feature, because it means that as soon as a file of this size is saved, it is archived.

Until a file is archived, it is available to reports and charts in your ICONICS system; once it is archived, it is no longer available for that purpose. You can, however, restore an archived file to the system.

Lab: Logger Setup L2

In this section, you will create a Hyper Historian configuration database to be used as a building block for all of the exercises that will follow. You will add a new Disk Logger and learn how to set File Storage and Archiving options.

Estimated time to complete this lab: 15 minutes

2.1. Task Checklist

In this lab, you will complete the following tasks:

- Create a new Hyper Historian database
- Add a new Disk Logger

2.2. Support Files

You do not need any supporting files to complete this lab.

2.3. Creating Hyper Historian Database

Although a configuration database is created during installation, we will create one specifically for this project, without including Sample Data.

1. Open the Workbench from Start → Programs → ICONICS GENESIS64 → Workbench if you closed it from the previous lab.

2. Open Hyper Historian by clicking the Hyper Historian icon from the Project Explorer's bottom tray.

3. Once the Hyper Historian is open, right-click **Localhost** in the tree control and select **New Database**, as shown in Figure L2.1.

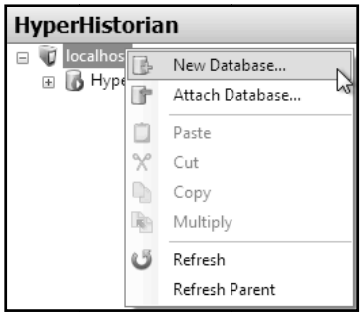


Figure L2.1 - Creating the Configuration Database

4. The **Hyper Historian – Create Database Wizard** starts.

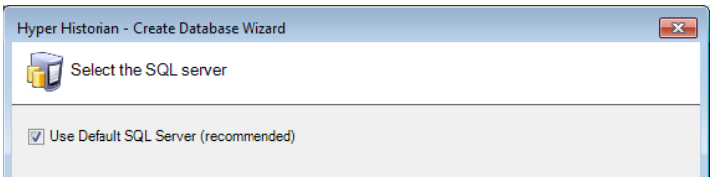


Figure L2.2 - Hyper Historian Create Database Wizard

5. Click **Next** to accept **Use Default SQL Server**, which is recommended.

6. In the next dialog, shown in Figure L2.3, uncheck **Include Sample Data** option. For the **Database Name**, type "My Database". Leave the default file settings and click **Finish**.

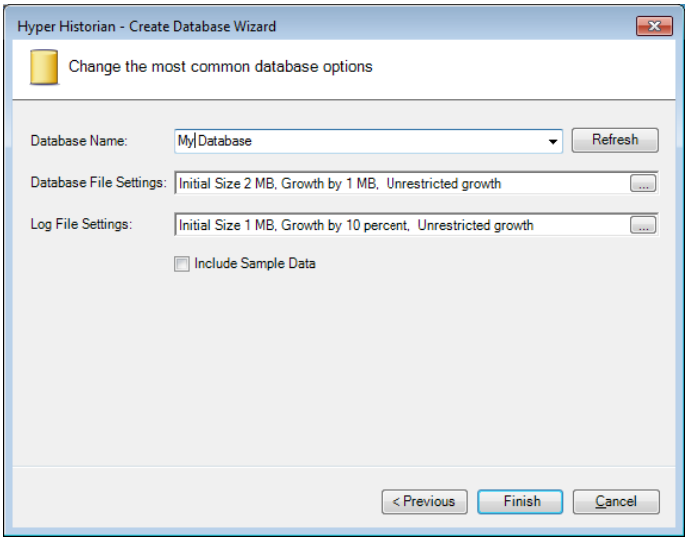


Figure L2.3 - SQL Database Wizard: Common Database Options

7. The database creation wizard now creates the new database.

8. Stop the service by selecting the **Hyper Historian Start/Stop Service** button. The traffic light should now be red.

9. Now set this new database as the **Active** database by right-clicking it and selecting **Make Active** from the drop-down menu, as shown in Figure L2.4.

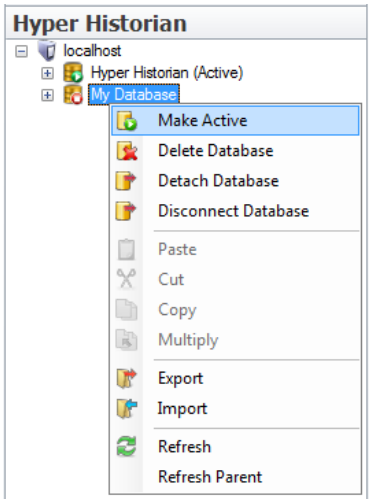


Figure L2.4 - Set Active Database

10. Start the service by clicking the **Hyper Historian Start/Stop Service** button. The traffic light should now turn green.

11. Once the traffic light is green, you should notice that your database now has the notation of "**(Active)**" to the right of its name, as illustrated in Figure L2.5.

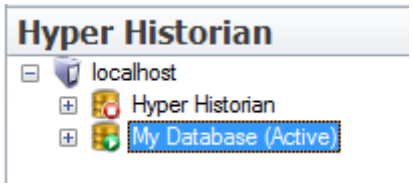


Figure L2.5 - Hyper Historian Active Database

2.4. Adding Disk Logger

A **Disk Logger** collects plant data that it saves to a hard disk. A **Memory-Persistent Logger** collects plant data that it stores in RAM and discards after a pre-configured amount of time. In this exercise, you will create a commonly used configuration based on File Logging.

1. With Hyper Historian open in the Workbench, expand your newly-created database.

2. Right-click on **Loggers** and select **Logger**, as shown in Figure L2.6.

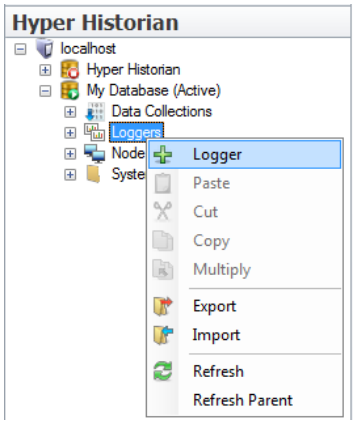


Figure L2.6 - Add New Logger

3. Once the new logger opens, select **Disk Logger** from the **Logger Type** drop-down list, as shown in Figure L2.7.

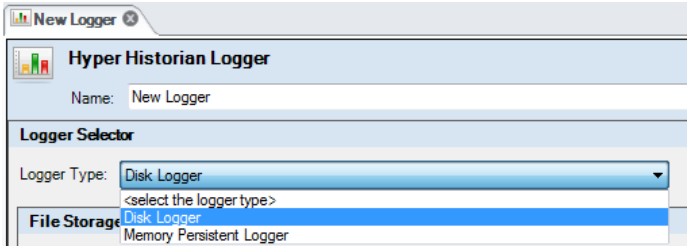


Figure L2.7 - Logger Configuration

4. Type “Disk logger” into the **Name** field.

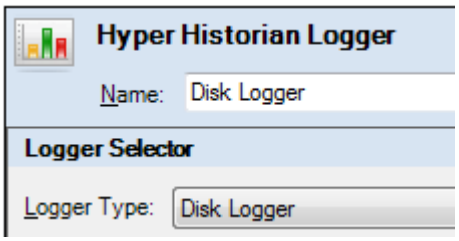


Figure L2.8 - Disk Logger

5. In the **File Storage Information**, set **Max File Time Extent** to **1 Hour**. This specifies the time after which a new Storage File will be created.

6. Under **Storage Limit**, select **Minimum Time Extent** from the **Condition** drop-down list, and set **Min Time Extent** to **2 Hours**. This specifies how long the configuration’s saved files are kept for retrieval.

Click **Apply** to save your Disk Logger configuration. (You may need to scroll down to see the Apply button.)

Review: Logger Configurations R2

2.1. Chapter Objectives

You should now know and understand the following topics:

- Identify the uses of the two types of loggers
- Set up a Disk Logger configuration for saving collected data to a hard drive
- Set up a Memory Persistent Logger configuration for collecting data in RAM
- Store and archive logged data

2.2. Summary

Hyper Historian has two types of logger configurations that govern how data gets logged to the Logger.

A Disk Logger saves collected data to a hard disk, which can be used to collect plant data for the following uses:

- for use in historic replay with TrendWorX64
- for reports created using ICONICS BizViz products
- for archival for regulatory compliance
- for future reference and analysis

A Memory-Persistent logger configuration stores collected data in RAM and discards it after a pre-configured amount of time. Using a Memory Persistent Logger has its advantages in that the data you collect can be more granular (and, therefore, of a greater volume) because you will not be saving all of it. Should you choose to save data surrounding a specific event, you can do this while capturing a high resolution of data that helps you analyze the moments up to and after the event.

Data stored with either type of logger configuration is available for replay and reporting for a timeframe of your choosing. Stored data is automatically archived at a point determined by you, and can be easily restored from archives.

2.3. Terms

Archived Data Disk Logger Memory-Persistent Logger

2.4. Test Your Knowledge

In this chapter, we talked about Hyper Historian's two types of logger configurations. Make sure you can answer these questions before you proceed:

1. What are the two types of logger configurations? Select two.

    a. Hyper Historian Disk Logger
    b. Hyper Historian Database Logger
    c. Hyper Historian Memory Persistent Logger
    d. Hyper Historian Temporary Logger

2. Which type of logger configuration stores the data in a SQL database?

    a. Hyper Historian Disk Logger
    b. Hyper Historian Memory Persistent Logger
    c. Hyper Historian Database Logger
    d. Hyper Historian Temporary Logger
    e. All of the above
    f. None of the above

2. True or False? Both types of logger configurations can store data to a hard drive.

3. True or False? It is possible to base the archiving process on an event.

4. Why would you create a Memory Persistent Logger?

    a. To save data surrounding specific events
    b. For troubleshooting purposes
    c. To capture small amounts of high-resolution data for analysis
    d. None of the above
    e. All of the above

5. True or False? A deleted data file can be restored to your ICONICS system under some circumstances.

Logging Groups 3

As we discussed in the prior chapter, a logger consists of the tags that it collects data for. Now we will look at how the logger's tags are organized into logging groups. To set up a logging group you must:

- Create and configure a **logging group** that identifies the server where the logger resides, and the logger configuration to be used for logging data.
- Create the **collector groups** to be used for collecting values from tags.
- And create the **Hyper Historian tags** for each collector group. Each Hyper Historian tag identifies the OPC tag whose values are to be collected, along with an assortment of configuration information for collecting the tag's values.

In this chapter, we will review how to complete these steps.

3.1. Objectives

After completing this chapter, you should know how to:

- Set up a logging group
- Set up collector groups for a logging group
- Create folders for organizing Hyper Historian tags in a collector group
- Add Hyper Historian tags to a collector group
- Set up filters for a Hyper Historian tag

3.2. Before You Begin

Before we begin, let's look at what logging groups are. As shown in Figure 3.1, each Logger has one or more logging groups. Note that logging groups can be enabled or disabled via the configurator, and can be enabled or disabled online as well. Each Logging Group may contain one or more *collector groups*. Collector groups are made up of *Hyper Historian tags* whose data is logged. Hyper Historian tags can optionally be organized into *tag folders* so that it is easier for you to work with them.

Each Logging Group also has a folder called **Aggregation**, which Hyper Historian adds automatically to a logging group when you create it. You can add any number of *Aggregate Groups* to this folder. Each Hyper Historian tag can reference the Logging Group's aggregate groups.

The significance of this is described in the next chapter, where we will focus on aggregate groups. This chapter, however, focuses on Logging Groups, Collector Groups, Tags, and Tag Folders.

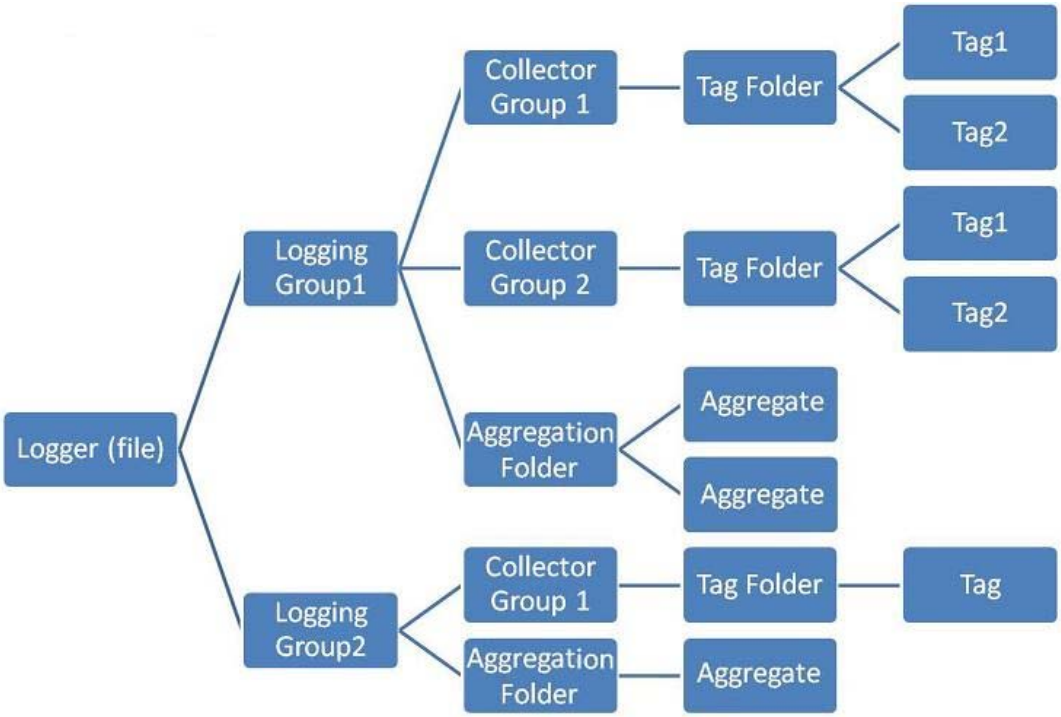


Figure 3.1 - Organizing Your Tags in Hyper Historian

3.3. Logging Groups

Once you have configured your Hyper Historian system and defined loggers, you can create logging groups. A logging group is a configuration for collecting Hyper Historian data for a set of OPC tags. Each logging group configuration identifies **the logger** configuration, which as you remember from an earlier discussion, is either a Memory Persistent logger that collects data into memory or a File Logger that saves data to the hard drive, and **an optional condition** under which data is collected. If you don't specify a condition, *all* incoming data will be collected.

You can create as many logging groups as you need for each logger. Each logging group has one or more collector groups; and each collector group is configured to collect data from one or more OPC tags. (Each logging group also has a folder for aggregate groups, which we will review in the next chapter.)

3.3.1. Create a Logging Group

To create a logging group, right-click on **Data Collections** in the navigation pane of the Workbench Project Explorer, then select **Logging Group**. The Hyper Historian Logging Group configuration appears as shown in Figure 3.2.

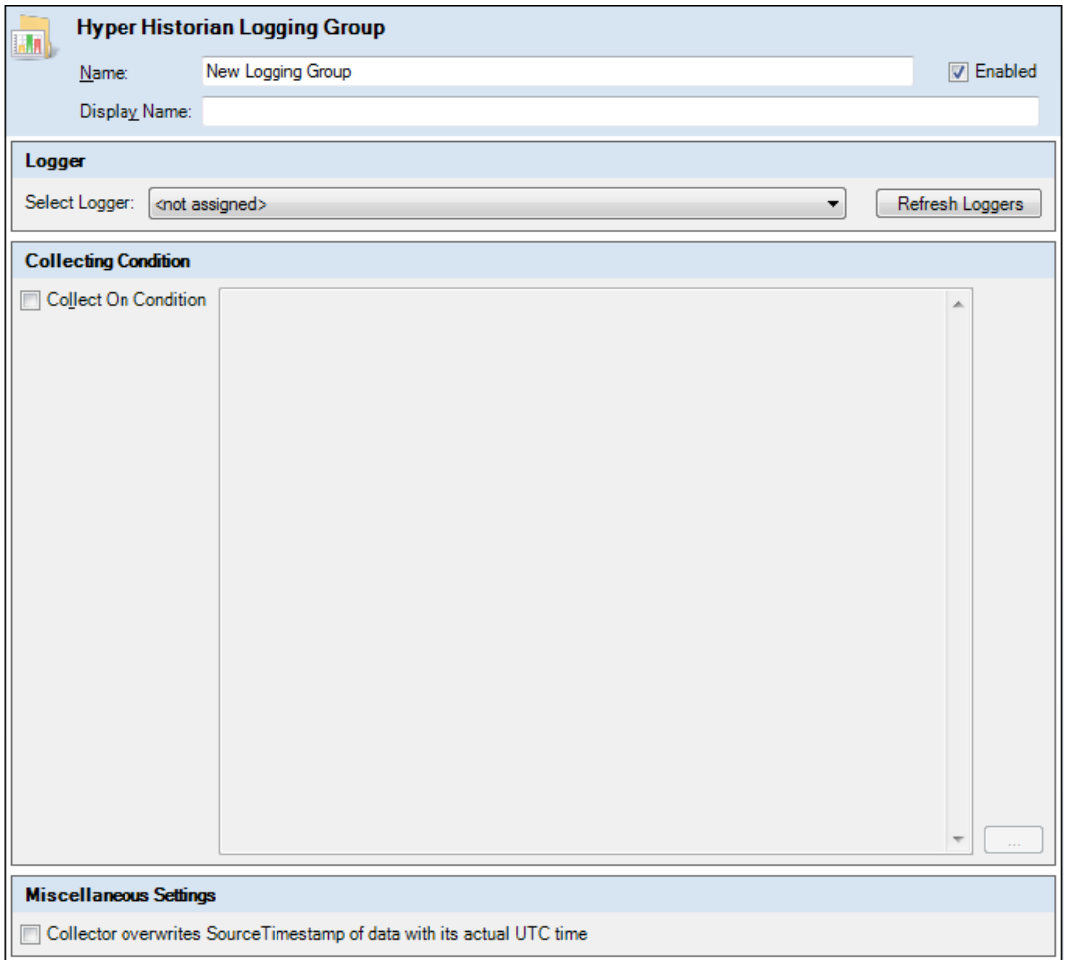


Figure 3.2 - Hyper Historian Logging Group Configuration

To define the logging group, start by giving the logging group a **Name** that makes its use easy to discern. Select the **Enable** checkbox to enable the logging group itself. As long as the logging group is enabled, any changes you apply take effect immediately.

From the **Logger** pull-down list, select the logger configuration to use for data collection. This is either a Memory Persistent Logger configuration or a Disk Logger configuration, as described in the prior chapter. The exclamation icon reminds you to assign a logger configuration. If necessary, you can create a new logger configuration, then return to this tab, click on **Refresh Loggers**, then select the new logger configuration.

If you want to collect data only under certain circumstances, set a **Collecting Condition**. To do this, put a checkmark in the **Collect On Condition** checkbox, then enter the name of the condition trigger to use for controlling the data collection, or click on the ellipsis button to browse for a conditional trigger.

If you want the tags' *local* timestamps to be collected with the data onto the server, put a checkmark in the **Collector overwrites SourceTimestamp of data with its actual UTC time** checkbox. Select this checkbox only if you want to use the timestamp from the local machine of the collector instead of your OPC Server.

3.3.2. What's Next?

Once you've defined a logging group, you can add a **collector group**, described next in this chapter, and an **aggregate group**, which allows you to log calculated data within a period of time using different mathematical functions. We'll get to aggregate groups in the next chapter.

3.4. Collector Groups in a Logging Group

A **collector group** defines the collection rate and period of calculation for the OPC tags you will be attaching to it. If you want to use more than one collection rate or calculation period, simply create an additional collector group, then add the tags to be collected at that rate. A logging group can have any number of collector groups. Once you have created the collector group, then you can specify the tags to collect samples for the collector group.

3.4.1. Create a Collector Group

To create a collector group, right-click on the logging group in the navigation true in the Project Explorer, then select **+ Collector Group**. The Hyper Historian **Collector Group** configuration appears as show in Figure 3.3.

Give it a name and, in the **Display Name** box, you have the option of entering a different name that appears only in the navigation tree for the Collector Group. In the **Info** section at the bottom of the form, specify the use for the collector group. You should make the description clear enough so that anyone who maintains the group understands its purpose.

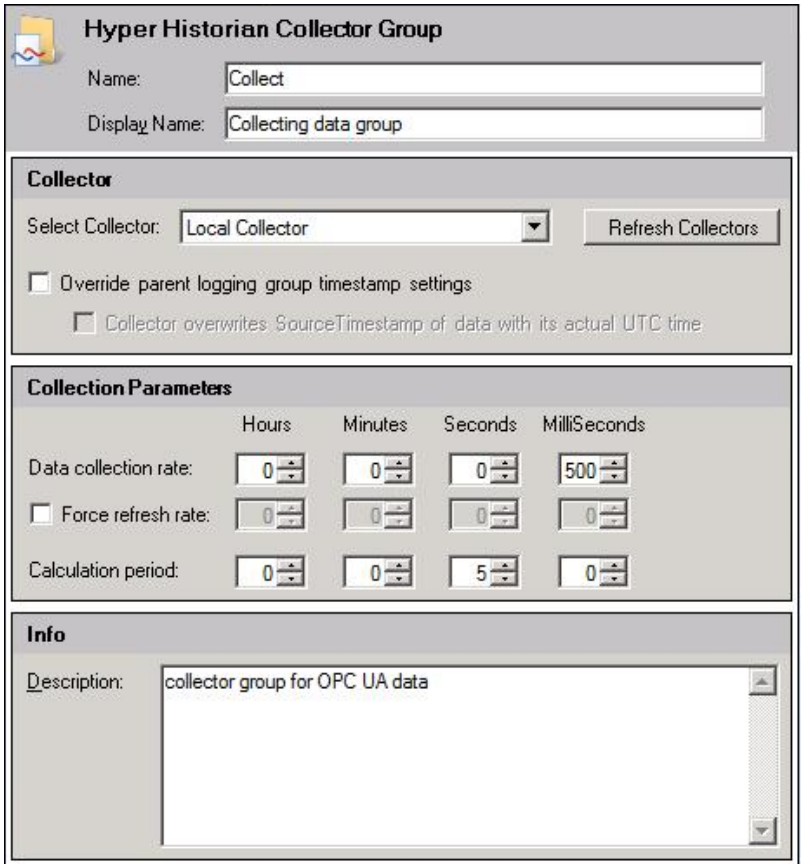


Figure 3.3 - Hyper Historian Collector Group

Select a collector from the **Select Collector** pull-down list. If you don't see the collector you are looking for, make sure it is defined, click on the **Refresh Collectors** button, and check the list again. If you choose a remote collector that physically resides in a different time zone than the logging server, you can choose which time zone's timestamps will be saved with logged values – enable the **Collector Override Parent Local Timestamps** and **Collector Overwrites Source Timestamp of Data with its Actual UTC Time** check boxes if you want the Hyper Historian Collector to apply its own time stamps to the collected data instead of using the time stamps provided by the OPC server.

For the **Data Collection Rate,** specify how frequently a sample will be collected from OPC tags that will be added to this collector group. You can enter the rate in hours, minutes, seconds, or milliseconds. The fastest data collection rate is ten milliseconds. While this option determines the collection rate, how you set each tag's filters in its configuration determines which collected values are sent to the logger.

To force the collector to refresh on a periodic basis, put a checkmark in the **Force Refresh Rate** check box and specify the interval at which it is to refresh.

The **Calculation Period** is used by some calculation filters, called **Log to Disk** filters that you can configure for Hyper Historian tags. These filters can choose a value from data received over the length of time established by the calculation period. The value used for calculation is extracted from the collector cache before the data is forwarded to the logger.

How long should the calculation period be? The length of the calculation period must take into account the data collection rate and the length that data resides in the memory buffer. Specifically, the calculation period must be longer than the data collection rate, but shorter than the rate at which memory buffer will be cleared.

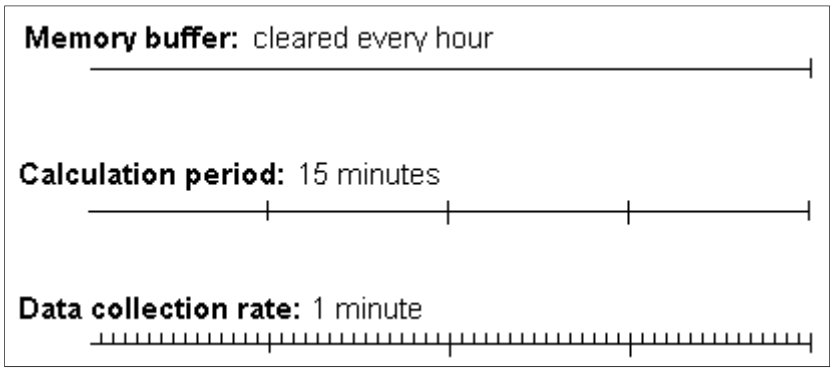


Figure 3.4 - Calculation Period, Memory Buffer, and Data Collection Rate Relationships

Within these confines, a *shorter* calculation period provides more data granularity and logs more data; a *longer* period provides less granularity but takes up less space. Regardless, if you choose to use a calculation period, you'll want to synchronize its duration with the expected duration of the cache such that data is not cleared in the middle of a calculation period.

Therefore, for the **Calculation Period**, specify in hours, minutes, seconds, or milliseconds the length of the period for which a sample can be sent to the logger. When setting a length for this period, keep in mind the type of OPC tags from which the collection group will be capturing data. For example, how often does the collector take a sample? How many samples do you want to collect? What granularity of information do you need? What sort of filtering will be used for the Hyper Historian tags added to the collector group? Once you complete this course, you will have a better understanding of how the calculation period is used in Hyper Historian and, therefore, will have a better idea of how you want to use it.

Once you have created the collector group, you can configure its Hyper Historian tags which identify the OPC tags from which you want to collect data.

3.5. Hyper Historian Tags in a Collector Group

A Hyper Historian tag is a configuration that identifies an OPC tag from which data is to be collected. You can use up to three filters to separate out a tag's values to be forwarded to the logger. Alternatively, you can choose to use *no* filters and forward *all* of a tag's sample values to the logger.

3.5.1. What's a Hyper Historian Tag?

Each Hyper Historian tag collects data from an OPC tag or an expression that calculates a value from multiple OPC tags. As you work with Hyper Historian tags, keep in mind that it is the *collector group* that determines:

- The collection rate at which values are collected for the tags and
- The calculation rate, at which calculations are performed on the collected samples, and then forwarded to the logger

For example, let's assume that the collection group's collection rate is set to collect four values per second. When you create a configuration for the tag in Hyper Historian, you can choose to send all samples to the Logger, or apply any or all of the three filters that are available. With these filters, you can: choose from a long list of options, including the minimum or maximum value for the calculation period; apply an industry-standard Deadband; apply an industry-standard Swinging Door Compression filter.

You may find that you need to filter data from an OPC tag in more than one way. In that case, you can simply configure multiple Hyper Historian tags for the same OPC tag in the collector group; each Hyper Historian tag can filter the data in a different way, sending a different subset of data to the logger.

> **NOTE:** An alternative would be to use one Hyper Historian tag to send an encompassing set of data to the logger, then using post-logging tools, such as Excel, to filter data into usable subsets.

3.5.2. Adding a Hyper Historian Tag

To add a tag to the collector group, right-click the collector group, then select **Tag** to proceed. The Hyper Historian tag configurator appears as shown in Figure 3.5.

> **NOTE:** If you would rather organize the collector group's tags into folders, you can create a folder, then right-click the folder and select **Tag**.

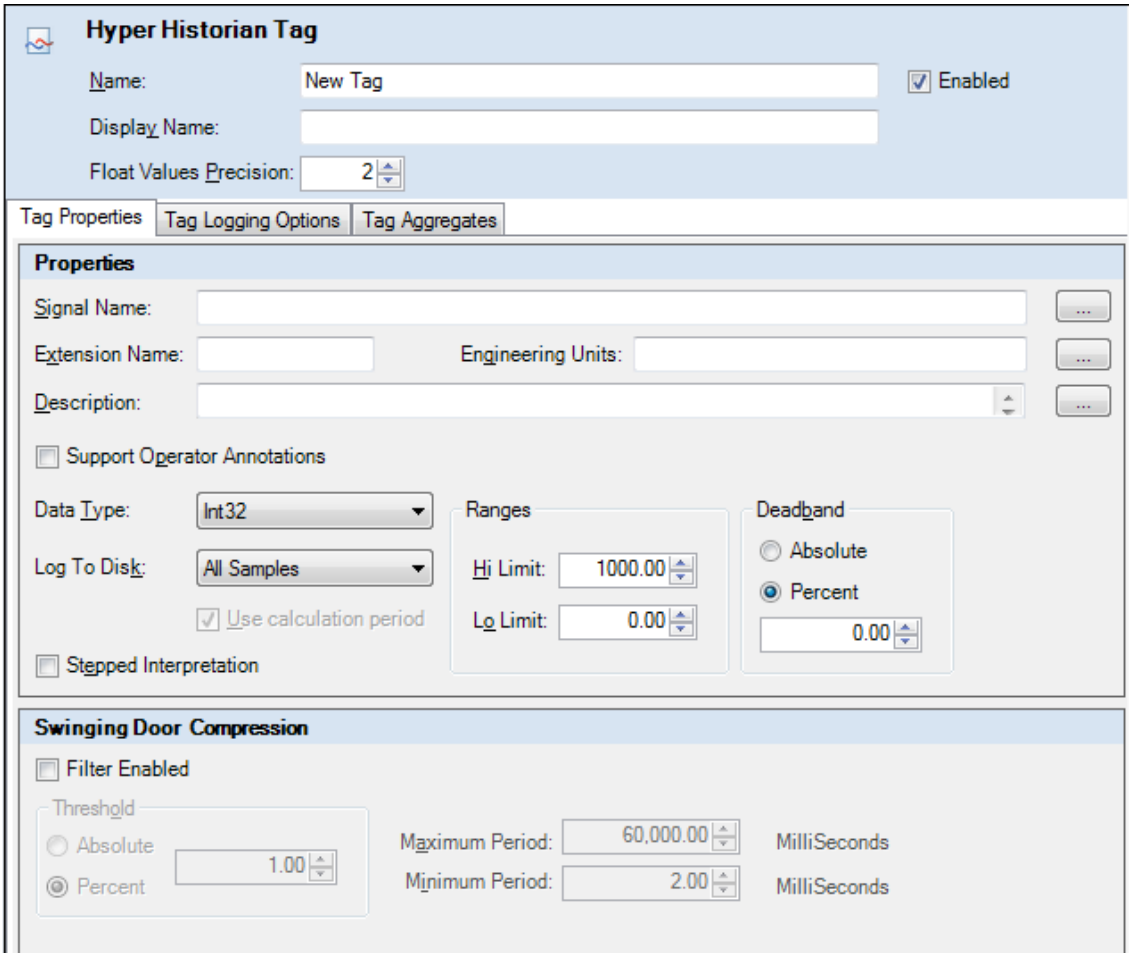


Figure 3.5 - Hyper Historian Tag Configuration

The Hyper Historian Tag Configurator has three tabs. In this chapter, we will be looking at the **Tag Properties** and **Tag Logging Options** tabs. We'll save the **Tag Aggregates** tab for the next chapter, where we'll look at Aggregate Groups.

Start by entering a **Name** for the tag. Make sure the name is descriptive and unique, but accurate so that others can recognize it. You may want to use the name of the OPC tag, itself. If you are configuring more than one Hyper Historian tag for an OPC tag, make sure you use a distinct name of each tag; for example, add "Minimum" to the name. Each tag name must be unique for the collector group. You can provide a separate name for the tag in the **Display Name** field – this name will appear in the navigation tree.

You can adjust the number of decimal places that appear on the configuration tabs by using the **Float Values Precision** field. The number of decimal places to be shown can be adjusted from 0 to 8 decimal places. This field changes the number of float positions if you need more decimal places for the numbers you are entering for filtering. You can change this number at any time without affecting the values you've entered already.

On the **Tag Properties** tab, for the **Signal Name** field select the signal (that is, the tag or point) to be collected by clicking the Signal Name's ellipsis button and browsing for the OPC tag whose data is to be collected. Alternatively, you can configure an expression in the Data Browser. If you leave this field empty, an exclamation icon reminds you that you have to enter a signal name or expression. If you are using a signal name more than once as a Hyper Historian tag, make sure you also use the Extension Name field.

The **Extension Name** field makes the signal name unique if the signal name is used for more than one Hyper Historian tag. For the *first use* of the signal name, this field can be empty. If you are using *a signal that has already been used* in another Hyper Historian tag, an extension is generated automatically; you can modify the generated extension if you like.

Enter the **Engineering Units** or click on the ellipsis button to select an alias tag for the Engineering Units using the Data Browser. Also, enter a **Description** or click the ellipsis button to select an alias tag for the description using the Data Browser.

Later, as the tag's data values play or replay in the Trend Viewer, you may want to be able annotate the tag's data values as you see them trending before you. For example, maybe you want to note an operator action that altered the data slightly, or highlight a piece of data for analysis later. Putting a checkmark in the **Support Operator Annotations** checkbox lets you do that. Such annotations are saved with the tag's data in its logger file, which makes the annotations available in a replay, accessible to the Hyper Historian SQL Server Interface, and archived with the tag's data.

> **NOTE:** Annotations are added in the Trend Viewer. To add an annotation, you freeze the view then select **Add** Annotation from the pop-up menu. Once added, the annotation appears as an icon in the data at the point at which you added the annotation.

Select the **Data Type** from the pull-down list to choose the data type in which you want the data to be logged. The default value is the tag's native data type on the OPC server. The different data types are: **Boolean**, **Int8**, **UInt8**, **Int16**, **UInt16**, **Int32**, **UInt32**, **Int64**, **UInt64***, **Float32**, **Float64***, and **String**. Obviously, you want to make sure you choose a data type that is relevant to the data being collected from the tag *and* to the way you intend to filter, display, and use the data.

> **Note:** *The **UInt64** and **Float64** data types are not supported by OPC DA servers running on Windows 2000.

3.5.3. Filtering Data

You can limit the amount of data sent to the logger by implementing different filters available in the tag's configuration. Using any of these filters is optional, but if you use them, they are applied in the following order:

- The **Log to Disk** pre-configured calculations let you identify the type of value to be forwarded to the logger; for example, you can choose to forward only the minimum, maximum, or average value for each calculation period.
- An industry-standard **Deadband** filter.
- An industry-standard **Swinging Door Compression** filter.

Let's take a look at each of these filters.

3.5.3.1. Using Log-to-Disk Filters

You can use **Log to Disk** calculations as a filter, to limit the volume of data sent to the logger (i.e., logging only a minimum, maximum, or other type of value over a period of time). Your **Log to Disk** selection, which is used by the Collector, specifies the value that gets sent to the logger. If you *don't* want to filter samples before they are sent to the logger, select the **All Samples** option; as a result, all collected values are forwarded to the Logger.

> **NOTE:** The **Log to Disk** option only determines *which values get sent to the Logger*, not whether they are saved to disk. Values are saved to disk only if the logger configuration is Disk-based or – if it is a Memory Persistent Logger configuration – is configured to save values to a file.

Even *after* a value is forwarded to the Logger, you can configure the tag so that additional filters (the Deadband and Swinging Door filters) are applied to subsequent incoming values.

*You do have the option to send **all** collected values to the logger.* However, using this filter can reduce network traffic and spare you RAM and disk space. So if you can get the data you need by using this filter, it is worth considering using it.

From the **Log to Disk** pull-down list, you can choose one of the filters described in this section. When choosing a **Log to Disk** filter, keep in mind the following:

- If you'd like to filter an OPC tag's data in multiple ways, you can create additional Hyper Historian tags, one for each Log to Disk filter you want to use for the OPC tag. For example, if you want to send the minimum, maximum, and average values to the logger, you can configure three Hyper Historian tags, one for each type of filter.
- The calculation period (which is specified in the collector group's configuration) is considered for only those options marked below with the symbol ‡. Such options consider only those values collected inside the calculation period as long as there's a checkmark in the **Use Calculation Period** checkbox, and one value is sent to the logger for each calculation period. *However*, you can disregard the calculation period for these options by removing the checkmark from the **Use Calculation Period** checkbox. If you remove the checkbox, one value is sent to the logger before the memory buffer is cleared.
- Some of the **Log to File** options do not render some data types, so always consider the tag's data type (in the **Data Type** field) when choosing from the **Log to File** list. For example, choosing an average for a String data type will result in no data being logged.
- The **Stepped Interpretation** checkbox is used primarily during playback, for interpreting the tag's data values. This feature is applied only if you request a value for a certain point of time. The returned value will be the nearest older value available.

NOTE: In general, if the data comes from a discrete signal such as an on/off switch (or a Boolean data type), put a checkmark in the Stepped Interpretation checkbox; but if data comes from a sine wave or analog signal, leave the checkbox empty.

3.5.3.2. Different Log to Disk Filters

As we said earlier, if you *don't* want to filter samples before they are sent to the logger, select the **All Samples** option; as a result, all collected values are forwarded to the Logger. But if you *do* want to send calculated values (before the Deadband and/or Swinging Door filters are applied), select one of the options offered in the **Log to Disk** pull-down list:

- Minimum, Moving Minimum, or Running Minimum
- Maximum, Moving Maximum, or Running Maximum
- Average, Moving Average, Running Average
- Standard Deviation
- Totalizer

- Most Recent On Time

Let's take a closer look at each of these values.

Logging a Minimum Value

If you want only minimum values sent to the logger, you can choose one of the following **Log to File** filter options:

- **Min‡** (minimum) – At the end of the calculation period, the lowest value collected during the calculation period is sent to the logger; all other values collected during the calculation period are discarded.
- **Moving Min‡** (moving minimum) – The calculation period moves forward each time a value is collected, creating a sliding window of time from which the lowest value is determined. The lowest value collected during the sliding calculation period is sent to the logger.
- **Running Min** (running minimum) – Each collected value is compared to the lowest value logged thus far since the beginning of data collection for the Hyper Historian tag. When a new low value is collected, it is sent to the logger. All subsequent incoming values are then compared to this new low. Values are logged at the collection rate.

Table 3.1 compares these three options. A value is collected every 15 seconds; the calculation period is 60 seconds. The top row shows incoming values; each subsequent row shows the values that would be logged for each option.

Table 3.1 – Comparison of Min, Moving Min, and Running Min

Incoming values, every 15 seconds:	**75**	**76**	**73**	**84**	**72**	**69**	**68**	**65**	**89**	**73**	**74**	**75**
Min‡				73				65				73
Moving Min‡				73	72	69	68	65	65	65	65	73
Running Min	75	75	73	73	72	69	68	65	65	65	65	65

Logging a Maximum Value

If you want only maximum values sent to the Logger, you can choose one of the following **Log to File** options:

- **Max‡** (maximum) – At the end of the calculation period, the highest value collected during the calculation period is sent to the logger; all other values collected during the calculation period are discarded.

- **Moving Max‡** (moving maximum) – The calculation period resets each time a value is collected, creating a sliding window of time from which the highest value is determined. The highest value collected during the sliding calculation period is sent to the logger.
- **Running Max** (running maximum) – Each collected value is compared to the highest value logged thus far since the beginning of data collection for the Hyper Historian tag. When a new high value is collected, it is sent to the logger. All subsequent incoming values are then compared to this new high. Values are logged at the collection rate.

Table 3.2 compares these three options. A value is collected every 15 seconds; the calculation period is 60 seconds. The top row shows incoming values; each subsequent row shows the values that would be logged for each option, assuming the **Use Calculation Rate** checkbox has a checkmark.

Table 3.2 – Comparison of Max, Moving Max, and Running Max

Incoming values, every 15 seconds:	**75**	**76**	**73**	**84**	**72**	**69**	**68**	**65**	**89**	**73**	**74**	**75**
Max‡				73				65				73
Moving Max‡				84	84	84	84	72	89	89	89	89
Running Max	75	76	76	84	84	84	84	84	89	89	89	89

Logging an Average

If you want value averages sent to the Logger, you can choose one of the following **Log to File** options:

- **Avg‡** (average) – At the end of the calculation period, the average of all values collected during the calculation period is sent to the logger. All collected values for the calculation period are discarded.
- **Moving Avg‡** (moving average) – The calculation period resets each time a value is collected, creating a sliding window of time in which the average is calculated for all values collected during the sliding calculation period.
- **Running Avg** (running average) – With each collected value, the average to-date is calculated for all values collected so far for the Hyper Historian tag.

Table 3.3 compares these three options. A value is collected every 15 seconds; the calculation period is 60 seconds. The top row shows incoming values; each subsequent row shows the values that would be logged for each option, assuming the **Use Calculation Rate** checkbox has a checkmark.

Table 3.3 – Comparison of Avg, Moving Avg, and Running Avg

Incoming values, every 15 seconds:	**75**	**76**	**73**	**84**	**72**	**69**	**68**	**65**	**89**	**73**	**74**	**75**
Avg‡				77				68.5				77.75
Moving Avg‡				77	76.25	74.5	73.25	68.5	72.75	73.75	75.25	77.75
Running Avg	75	75.5	74.66	77	76	78.83	73.86	72.75	74.56	74.4	74.36	74.42

Logging the Standard Deviation

If you want the standard deviation for the calculation period sent to the Logger, choose **Std. Dev.**

This option logs the standard deviation value of all samples collected for the tag. It calculates the standard deviation value over the calculation period if there is a checkmark in the **Use Calculation Period** checkbox. If the **Use Calculation Period** checkbox is empty, the standard deviation value for all samples ever collected is logged each time a value is collected.

Using the Totalizer to Log Counts

Think of the **Totalizer** as a counter. The Totalizer gives you a count for a particular period, expressed in the units being collected. When a value is collected, the Totalizer updates the count then logs the count to the logger.

The following examples illustrate uses for the Totalizer option:

- For a signal that comes from a flow rate gauge that measures liters, the Totalizer gives you the total number of liters that pass through during an interval that you specify, such as liters per minute.
- For an electrical power gauge that measures kilowatts, the Totalizer gives you the total energy consumed during a measured interval, such as kilowatts-hours.

The totalized count is independent of the sampling rate; for example, energy consumed is the same regardless of how often you query the power gauge. Although, if the source oscillates at about half of the sampling frequency or faster, the result it gives you may be inaccurate; for this reason, you should carefully consider the collection rate being used.

If you choose this option, you must go to the **Tag Logging Options** tab where the **Totalizer** pane becomes available for you to configure the totalizer.

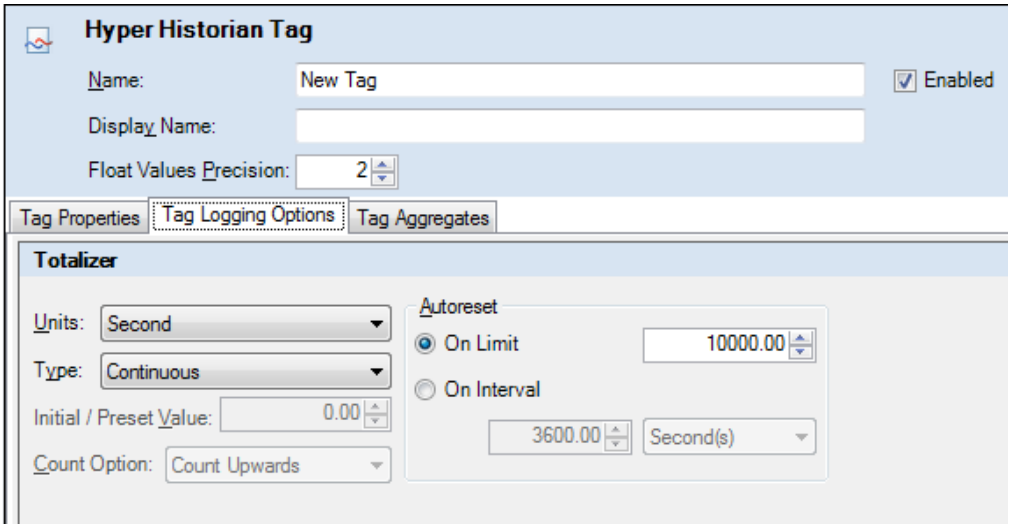


Figure 3.6 - Totalizer Options

In the **Units** field, specify the time measure for capturing the throughput rate per **Second**, **Minute**, **Hour**, or **Day**. For example, to get liters per hour select **Hour**, but to get liters per minute select **Minute**. To choose the unit that will give you an accurate totalized count, you need to *anticipate how fast your raw input signal is going to change*. For example, if the raw input signal takes more than an hour to change, then a sampling rate of once per *hour* is reasonable. However, if the raw signal changes significantly over the period of an hour, you should choose a faster collection rate of one per *minute*.

In the **Type** field, specify the starting count as Continuous, Preload or Preset. **Continuous** starts at 0 (zero) and counts continuously unless it reaches the auto-reset value. **Preload** starts at a count you specify in the **Initial/Preset Value** field; it counts continuously unless it reaches the auto-reset value. The Preload option is useful when you are bringing equipment online partially filled or when the count is suspended while an operation is being done (for example, the pipes are being cleaned) and then started from a known value.

If you choose **Preset** as the **Type**, the Totalizer counts in one of two ways:

- Upward from 0 (zero) to the preset value; when it exceeds the preset value, it resets to 0 (for example, when filling a tank).
- Downward from the preset value to 0; when it reaches 0, it resets to the preset value (for example, when draining a tank).

Specify the preset value in the **Initial/Preset Value** field; use the **Count Option** field to **Count Upwards** or **Count Downwards**.

With the **Autoreset** field, you identify the point at which you want the Totalizer to reset the counter. It can reset the counter under the following conditions:

- Once a total has been reached – Select **On Limit**, then specify the count which, when reached, the Totalizer will reset the count.
- After a specified period of time – Select **On Interval** and, in the fields that enable when you select it, specify the timeframe in seconds, hours, minutes, and/or days. For example, to reset the count once a week, specify 7 days. To get an hourly count, specify 1 hour.

When the Totalizer reaches the count limit or time interval, it restarts the count in accordance with the options selected in the **Type**, **Initial/Preset Value**, and **Count Option** fields.

Most Recent on Time

If you want the last value received in the calculation period sent to the Logger, choose **Most Recent On Time‡**. The last value collected during the calculation period is the value that is sent to the logger; all other values are discarded. You might use this if you need a second-by-second reading in memory but want to log a reading every 5 minutes.

If you choose this option, you must go to the tag's **Tag Logging Options** tab where the **Most Recent On Time** pane becomes available, as shown in Figure 3.7. Here you establish time parameters for the reading.

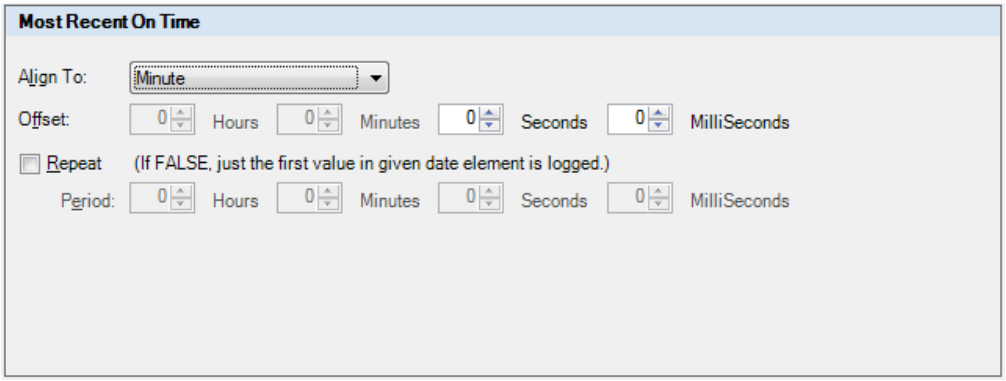


Figure 3.7 - Most Recent On Time Pane

3.5.3.3. Using the Deadband Filter

Each tag (signal) can have its own **Deadband** filter. To define the Deadband filter, you specify an amount or a percentage by which the tag's signal (that is, incoming value) must change before the collector will forward the incoming value to the logger. The Deadband compares each incoming value to the last value it sent to the Logger and sends the incoming value to the Logger only if it falls outside a margin that you specify.

You establish the Deadband's margin by specifying either an absolute value or a percentage by which the incoming value must differ from the last value forwarded to the logger. If it doesn't differ by at least that margin, it is discarded.

Ideally, you should use the Deadband filter if you are capturing the tag's running average or moving average. However, if you want to log all values, don't use the Deadband filter.

To disable the Deadband filter, enter zero as the amount or percentage. This will allow all samples to forward to the logger that were not pre-filtered by the **Log to File** filter.

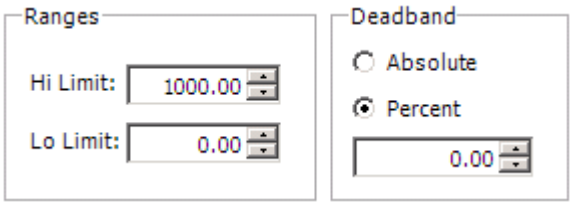


Figure 3.8 - Setting the Deadband filter

To set the Deadband filter, specify either a value or percentage:

- To specify a *value*, choose **Absolute** then specify the amount by which the incoming value must differ from the last value sent to the logger.
- To specify a *percentage*, choose **Percent**. You can specify the percentage by which the incoming value must differ from the last value forwarded to the logger. The percent is calculated from taking the absolute value of the difference of the **Lo Limit** and **Hi Limit** in the **Range** section, and taking the specify percentage of this value. For example, high and low limits are 100 and -100 and the percent of change is 5% relative to the range of values for the tag; in this case, the Deadband filter is waiting for an incoming value that is 10 higher or lower than the last value sent to the logger; when one arrives, it is forwarded to the logger.

NOTE: In the Float Position Precision field you can change the number of float positions if you need more decimal places for the numbers you are entering. Also, the Hi Limit and Lo Limit fields are used by the Deadband filter and the Swinging Door filter described next.

3.5.3.4. Using the Swinging Door Compression Filter

The Swinging Door filter discards incoming values that aren't needed in order to adequately reconstruct the trend of values graphically. This filter is useful for reconstructing sine waves in the Trend Viewer. An industry standard, the Swinging Door filter does the following:

- It uses all of the functionality used by the Deadband filter, plus
- It is the last filter that is applied, and
- It is applied by the Logger to incoming values it receives from the Collector. The Logger discards values that the Swinging Door Compression filters out, and keeps the rest in accordance with the logger's configuration (that is, either in memory or stored in files).

You can activate **Swinging Door Compression** by putting a checkmark in its **Filter Enable** checkbox on the tag's Tag Properties tab. The Swinging Door filter's **Absolute** and **Percent** options work the same way they work for the Deadband filter.

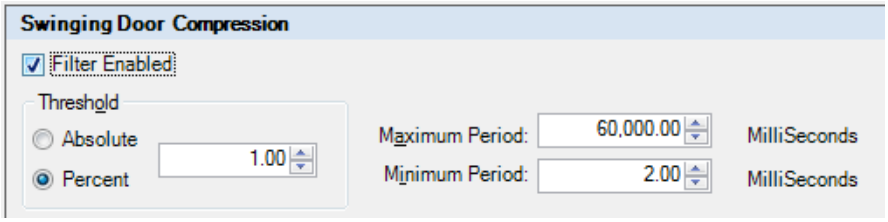


Figure 3.9 - Enabling Swinging Door Compression

Use the **Maximum Period** and **Minimum Period** fields to establish a timeframe for blocking incoming values that occurred relative to the last value kept in the logger. If two consecutive values are closer together in time than the minimum period, the new value is ignored. If the time between the last logged value and incoming value is longer than the maximum period, the filter is flushed.

3.5.4. Making Copies Quickly and Easily

You have the option, at any time, of quickly making any number of copies of a Logging Group, Collector Group, Hyper Historian Tag or Tag Folder, or Aggregate Group. Once copies are made, you can then tailor them to specific uses. This allows you to quickly create items that share characteristics, or to easily propagate characteristics throughout your system.

To make multiple copies of one of the items listed above, navigate to the item in the Project Explorer. Then right-click and select **Multiply**. This opens the Multiply Options dialog box, shown in Figure 3.10.

In the **Number of Items** box, specify the number of copies you want. Put a checkmark in the **Perform a Deep Copy of the Item** checkbox to copy *all* of the values defined for the item. This option is helpful if most of the object's properties are relevant for the copies you are making. If they aren't relevant, leave the checkbox empty.

Each copy's name will contain a number incremented by 1, starting with the value in the **First Number** box. The incremented number will have the number of digits specified in the **Numeric Places** box. You can override these increments for individual fields (the item's name, its display name, and its description) by leaving a checkmark in the field's **Modify** checkbox and entering new values for the line you want modified from the original. To prevent a field from being modified, remove the checkmark from the field's **Modify** checkbox. When you are ready, click **OK** to make the copies; the copies are added to the navigation tree, where you can access them and modify them.

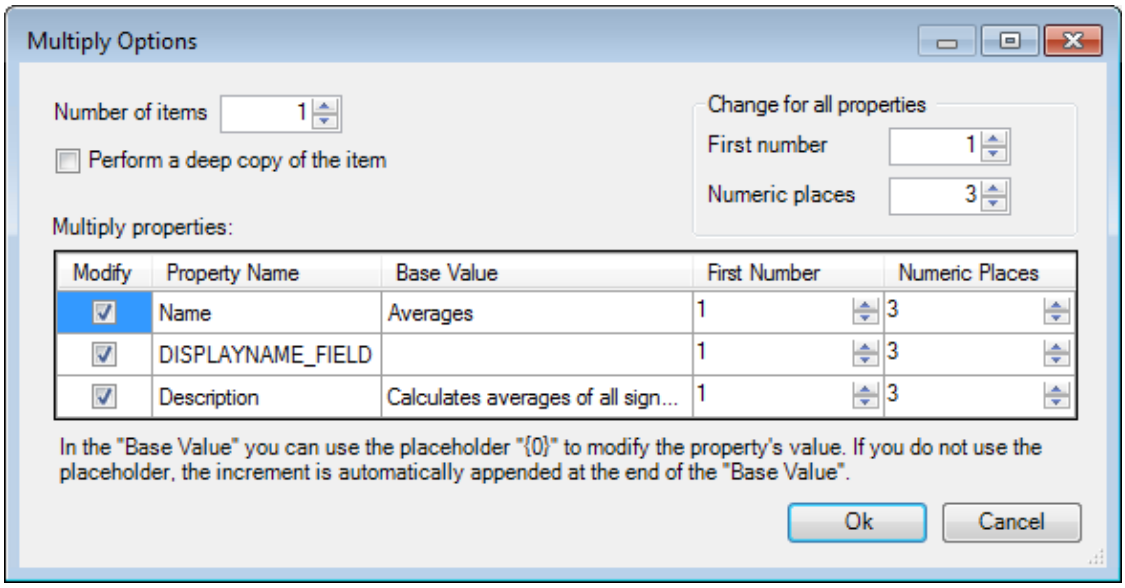


Figure 3.10 – Making Multiple Copies

3.6. Organizing Tags into Folders

If you have several tags that the collector group is collecting data from, you can organize the tags into folders if you like. To create a folder, right-click on **Collector Group**, then select **Folder**. Enter a name for the folder and a description of the tags in the folder. Click on **Apply** to save the folder. Then add tags to it by right-clicking on the folder and selecting **Tag**.

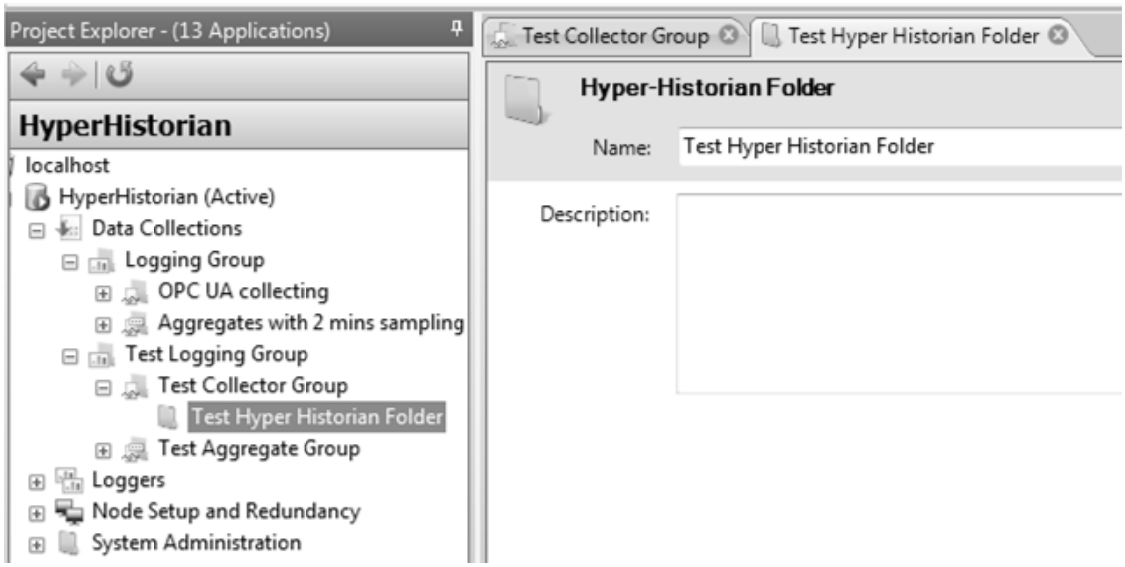


Figure 3.11 - Hyper Historian Folder Tag Folder

Lab: Collector Groups L3

Once you finish creating a Logger configuration, you can create Logging Groups. A Logging Group is a grouping of tags whose values are to be collected all at the same rate, and who all have the same filters applied to their data.

Estimated time to complete this lab: 20 minutes

3.1. Task Checklist

In this lab, you will complete the following tasks

- Create a logging group
- Define collector groups
- Add data points to a collector group
- Apply filtering algorithms

3.2. Support Files

You do not need any supporting files to complete this lab.

3.3. Create a Logging Group

For every Logging Group you need to define a Logging server and Logger that will be used for logging data that you specify in child Logging Groups.

1. Open Hyper Historian in the Workbench as explained earlier in the course. Expand your configuration database.

2. Right-click the **Data Collections** node in the Project Explorer and select **Logging Group**, as shown in Figure L3.1.

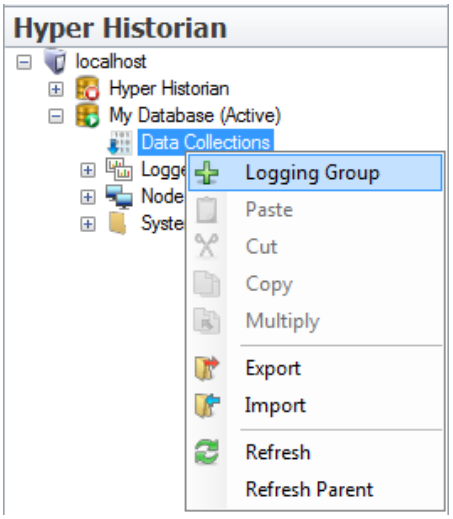


Figure L3.1 - Adding Logging Group

3. Give it a **Display Name** of "Logging Group" and make sure the **Enabled** checkbox has a checkmark in it.

4. Select the **Logger** that you created in a previous lab – **Disk Logger**. We are not going to enable Condition logging nor do we want our data to bear the Data Collector's timestamp, so we will leave the remaining checkboxes empty. Your form should look like Figure L3.2.

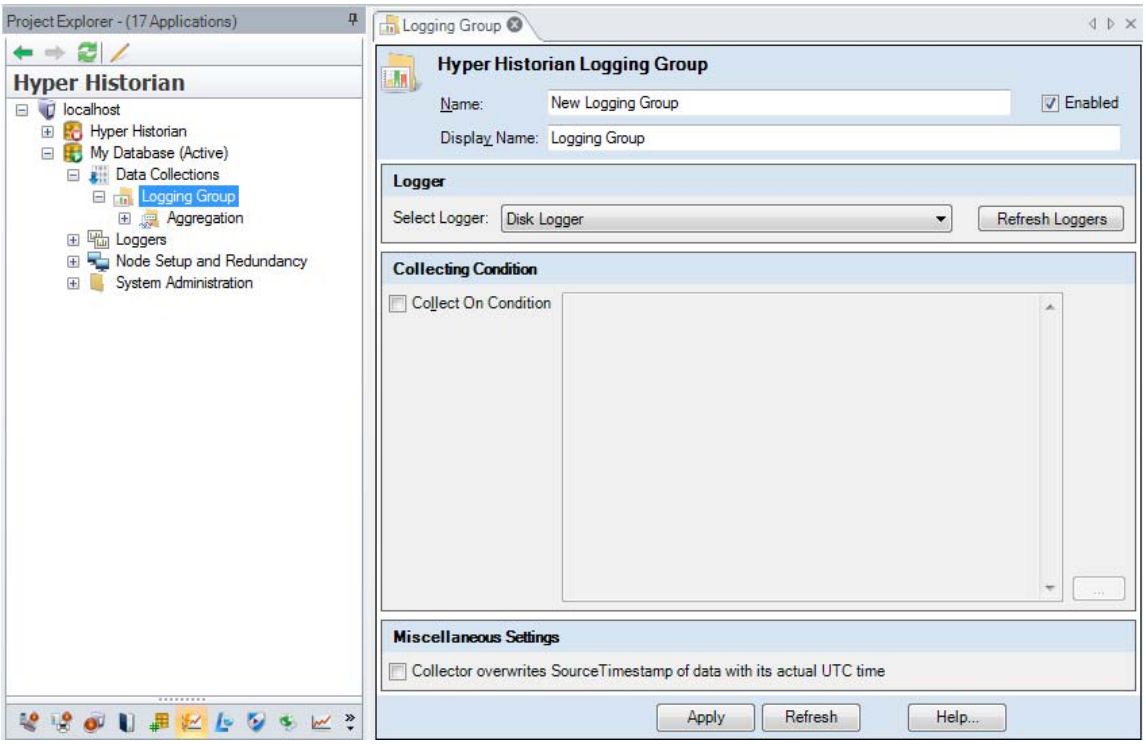


Figure L3.2 - Logging Group Configuration

5. Click **Apply** to save the new logging group. Notice that when you do this, a new node called **Aggregation** is automatically added to the new logging group's tree structure in the Project Explorer; we will take a look at the Aggregation folder later, in the next lab. Also, notice in the Project Explorer that although the new group's *name* is "New Logging Group" its *display name* of "Logging Group" is what appears here.

3.4. Setting up a Collector Group

Now let us create a Collector Group for our new Logging Group.

1. Right-click on the new Logging Group and select **+ Collector Group**, as shown in Figure L3.3.

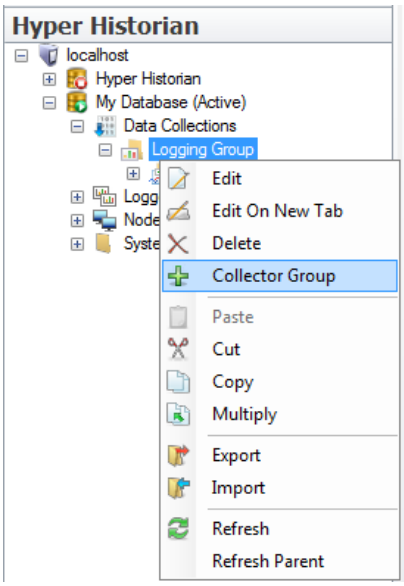


Figure L3.3 - Adding a Collector Group

2. In the New Collector Group tab (shown in Figure L3.4), provide a **Display Name** of "Collector Group". From the **Select Collector** drop-down list, select **Local Collector**. The Local Collector is created automatically when a new configuration is added. We won't be overriding timestamp settings.

3. Set the **Data Collection Rate** to be **1 second** and the **Calculation Period** to be **10 seconds**. We won't force a refresh for this lab.

4. You can add some additional info in the **Description** text box and click **Apply** to save the collector group configuration.

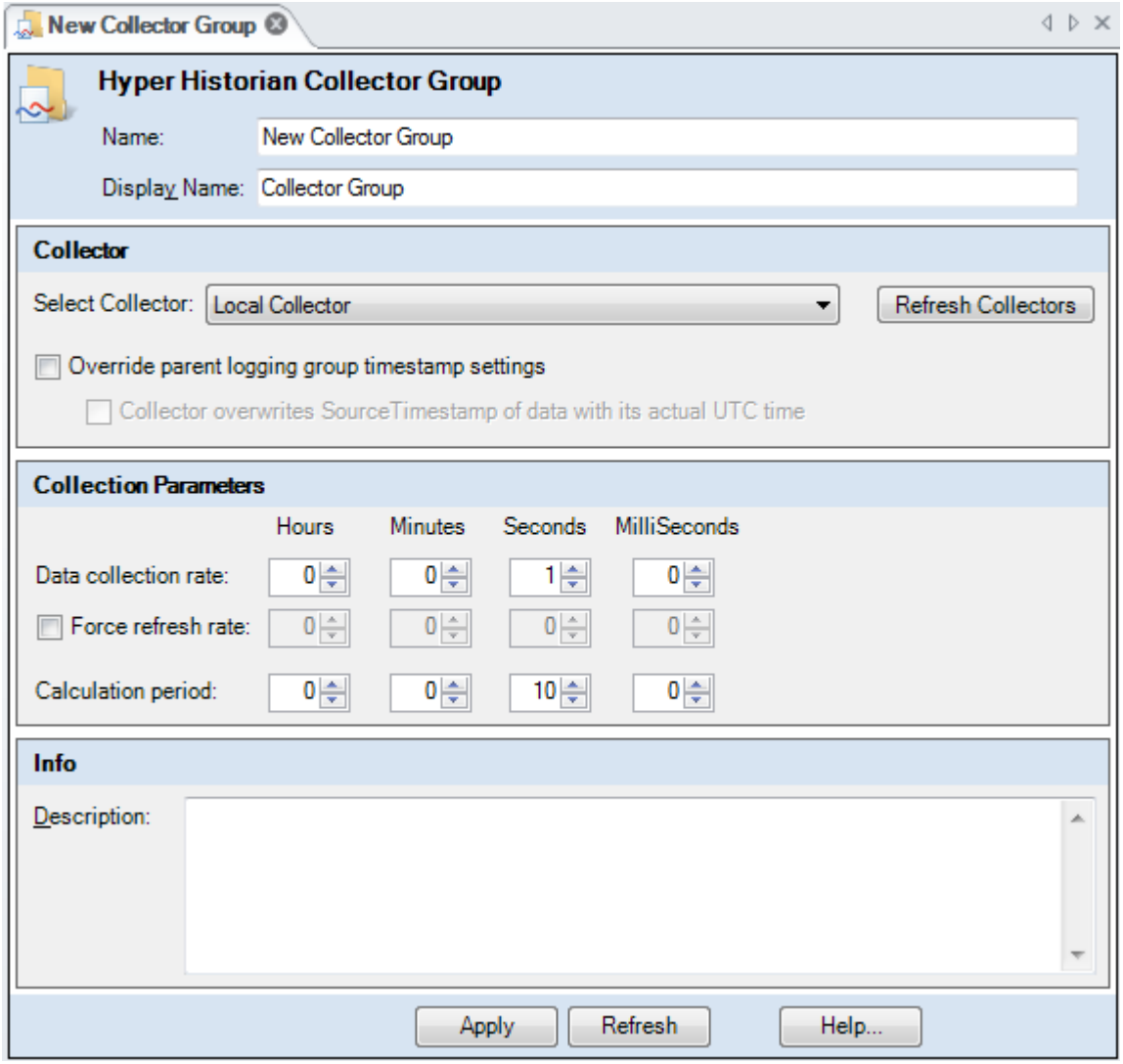


Figure L3.4 - Collector Group Configuration

3.5. Unfiltered Hyper Historian Tag

We have defined the rate at which data in the Collector Group will be acquired and stored. Now we are going to fill the collector group with monitored OPC data points. We will use one signal with different types of filters applied so that we can compare the resulting data.

1. Right-click the new **Collector Group** and select **+ Tag** from the pull-down menu. A new tag is added to the group and the new tag's configuration dialog opens, as shown in Figure L3.5. Now we are going to configure a tag.

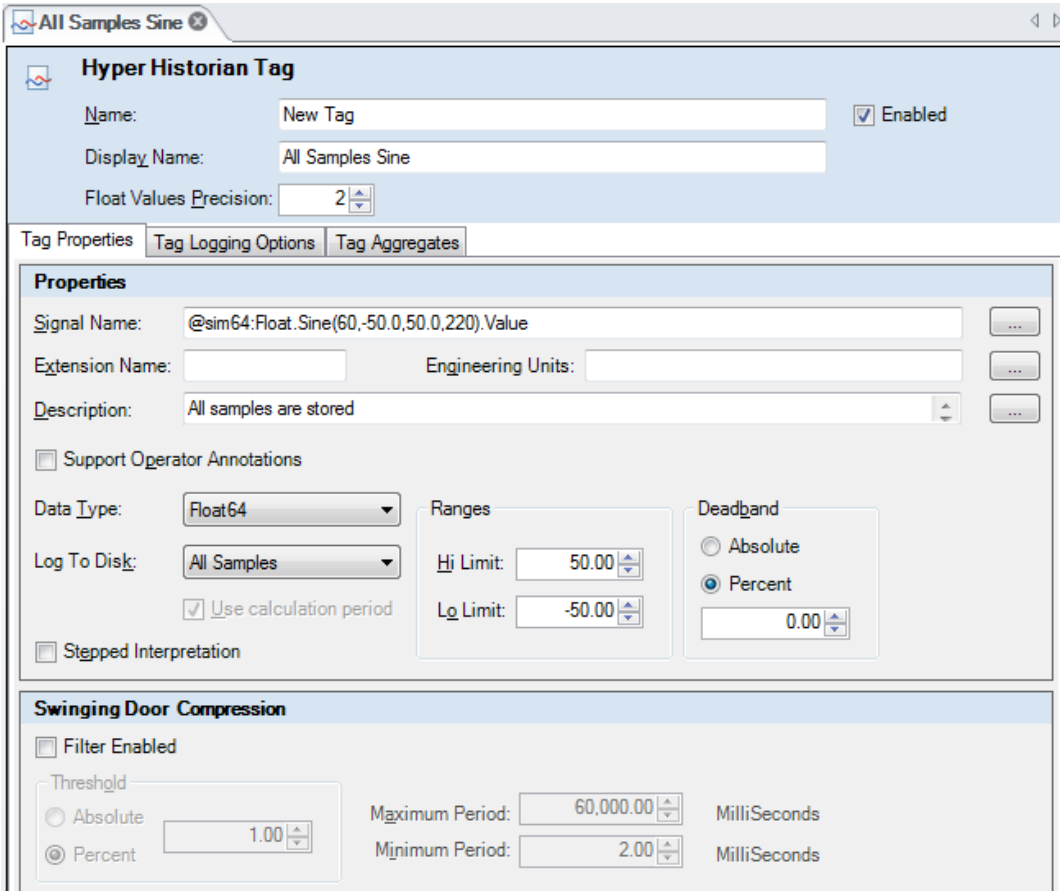


Figure L3.5 - Configuration of a Hyper Historian Tag

2. Change the tag's **Display Name** to "All Samples Sine" and click on the **Signal Name**'s ellipsis button.

3. In the Data Browser that appears, in the navigation pane, click **Simulation** and then on the right double-click the **Float** folder. The Data Browser will look something like the illustration in Figure L3.6.

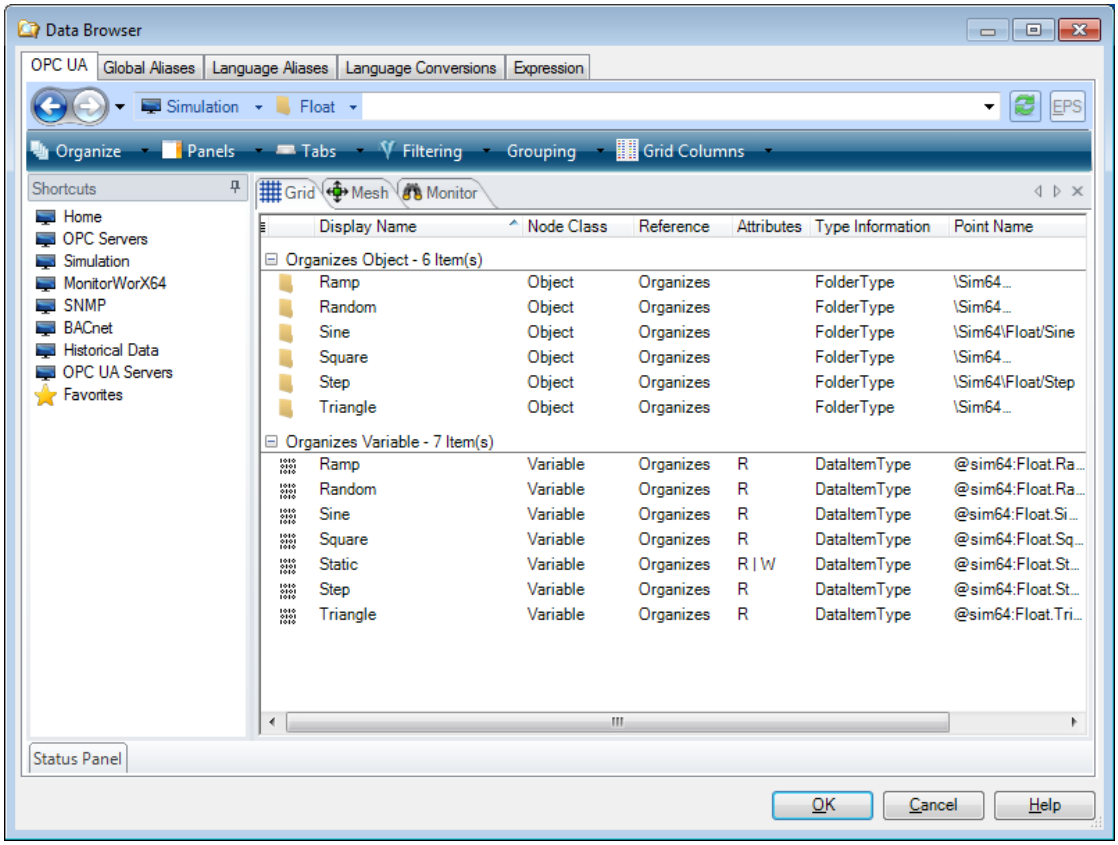


Figure L3.6 - Selecting a Data Source in the Data Browser

4. Double-click the **Sine** folder. From the list of signals, select **VerySlowXL220**, and then click **OK**.

5. Type “All samples are stored” in the **Description** field.

6. Select **Float64** as the **Data Type**.

7. Make sure the **Log To Disk** field reads **All Samples** (the default value), the **Deadband** value is **0,** and the **Filter Enabled** checkbox in the Swinging Door Compression section is **empty.**

8. Click the **Apply** button to save the configuration for this tag.

3.6. Filtered Hyper Historian Tags

We *could* create the three additional tags in the same way as the previous one, changing their properties along the way. Instead, we are going to use an easier way, by using the Multiply function.

1. Right-click the **All Samples Sine** Hyper Historian Tag in the navigation tree and select **Multiply**.

2. In the Multiply Options dialog that opens, as shown in Figure L3.7, set the **Number of Items** to **3**. Also, **remove the checkmark** from the **Modify** checkbox next to the **Signal Name** so that the signal name *does not* get incremented when we make copies. If you scroll up and down you'll notice all of the other properties that *will* be incremented. Click **OK**. Three more Hyper Historian tags are added to the navigation tree.

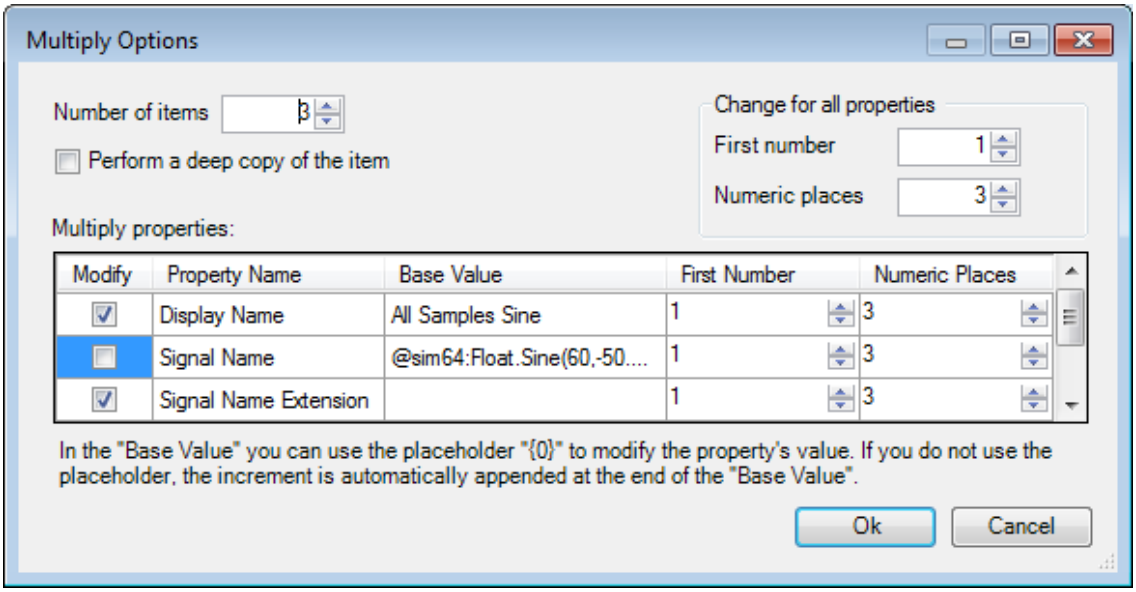


Figure L3.7 - Multiplying a Hyper Historian Tag

Log to Disk Filter

Next, we will apply a filter that will calculate an average value out of all collected samples within the Calculation Period. Only this average value will be forwarded to the Logger.

1. Double-click on the new **All Samples Sine001** Hyper Historian tag in the navigation tree to open the tag configuration form, as shown in Figure L3.8.

2. Change its **Display Name** to “Average Sine” and its description to “Average value is stored”.

3. Set the **Log To Disk** filter to **Avg**, and make sure the **Use calculation period** option is checked. Then click **Apply** to save the configuration.

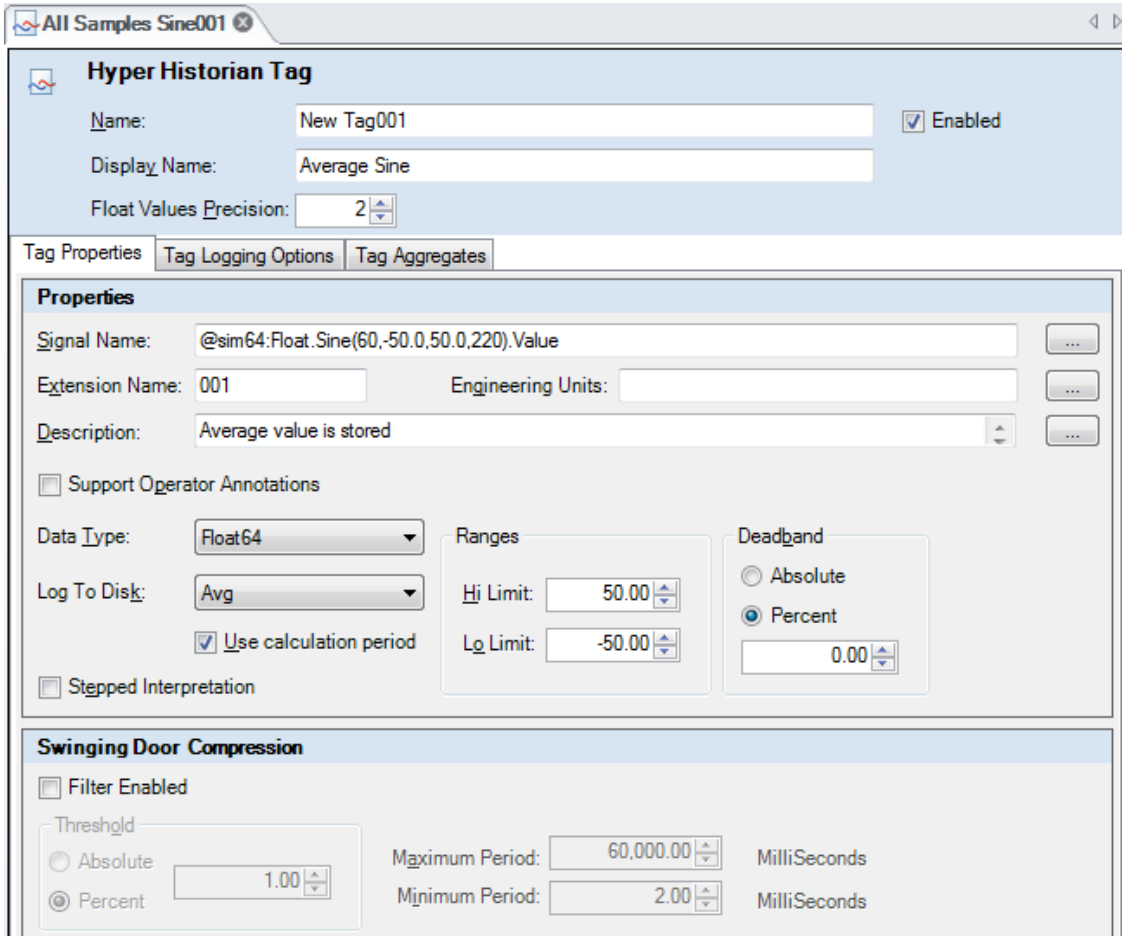


Figure L3.8 - Average Log to Disk Filter Applied on a Hyper Historian Tag

Deadband Filter

Now let us apply a deadband filter, as shown in Figure L3.9, which discards values that are fluctuating with small amounts.

1. Double-click the **All Samples Sine002** tag in the navigation tree.

2. Change its **Display Name** to “Deadband Sine” and its description to “Values beyond Deadband are stored”.

3. We are going to apply the Deadband filter based on an absolute change. Select **Absolute** in the Deadband section, and set its value to **20**. Then click **Apply**.

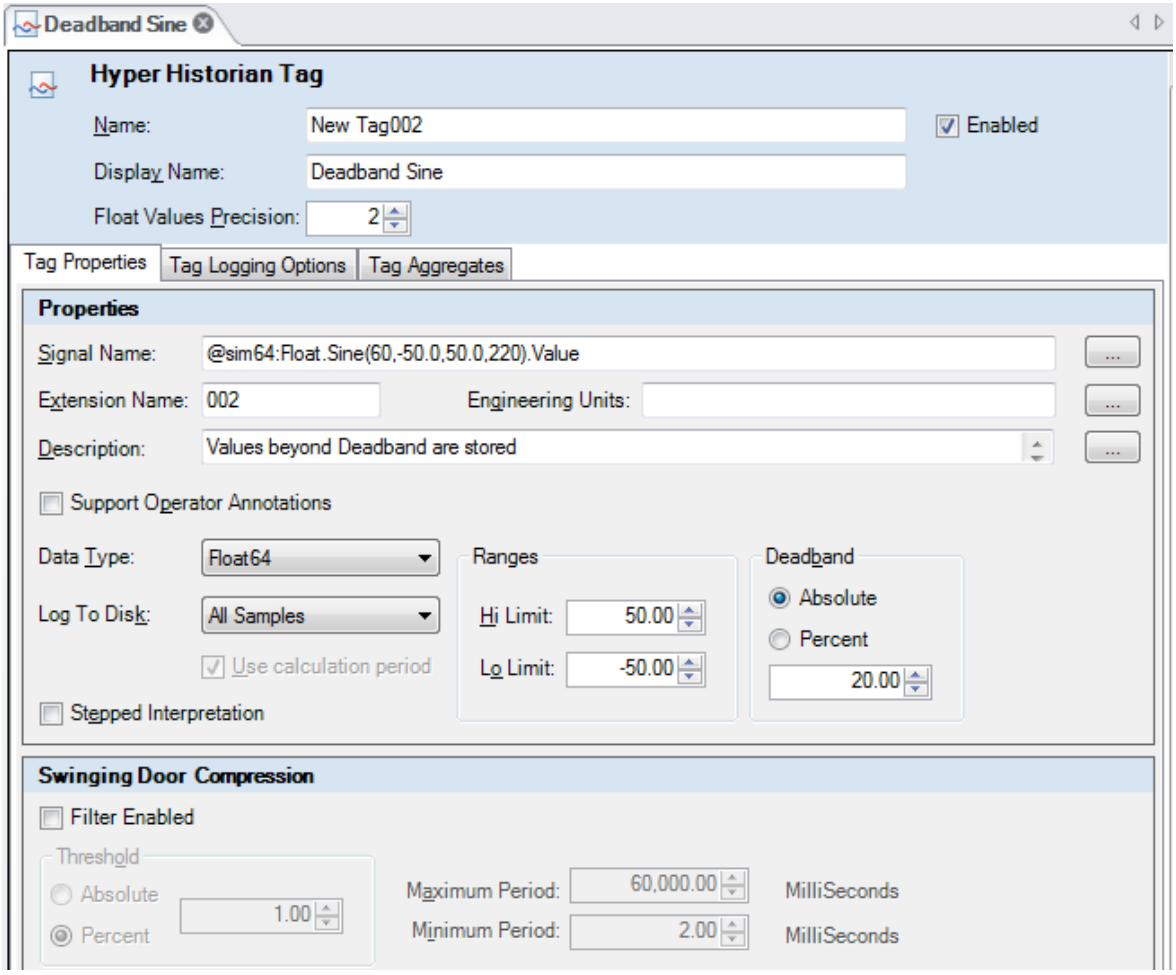


Figure L3.9 – Deadband Filter Applied on a Hyper Historian Tag

Swinging Door Compression

Swinging Door compression is very similar to Deadband filter. Now we are going to apply a Swinging Door filter to a Hyper Historian tag, shown in Figure L3.10.

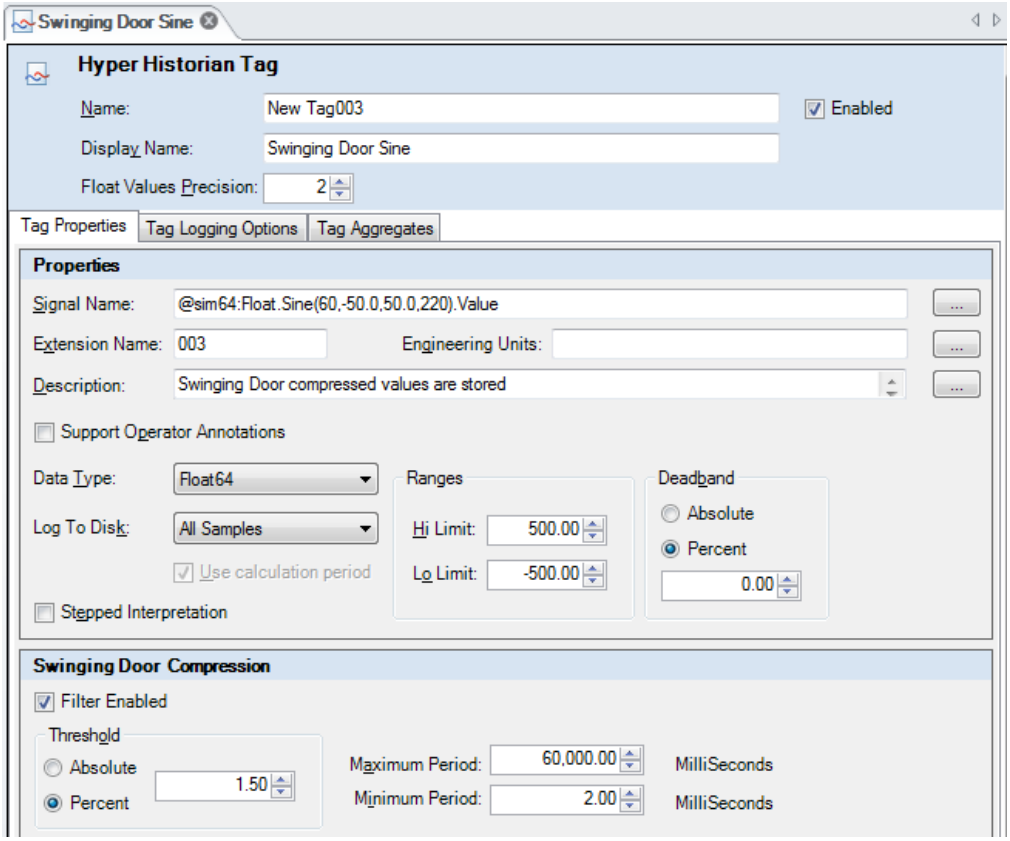


Figure L3.10 - Swinging Door Compressed Hyper Historian Tag

1. Double-click the **All Samples Sine003** Hyper Historian tag in the navigation tree.

2. Change its **Display Name** to "Swinging Door Sine" and its description to read "Swinging Door compressed values are stored".

3. Swinging Door Compression will be based on a percentage of change, so we have to set ranges. Set the **Hi Limit** to **500** and the **Lo limit** to **-500**.

4. Put a checkmark in the **Filter Enabled** checkbox in the Swinging Door Compression section of the dialog. This enables the options in this section.

5. If not already selected, select **Percent**, and set **1.50** for its value. Click **Apply** to save the configuration when you are done.

3.7. Data Comparison

Now that we have finished configuring our Hyper Historian tags, let's look at them.

1. Open the TrendWorX64 Viewer just as you did in the first lab.
2. Create a chart and then add all of the historical pens you have just created.
3. Select markers for each pen. Figure L3.11 shows a Marker selection of Square for one of the pens.

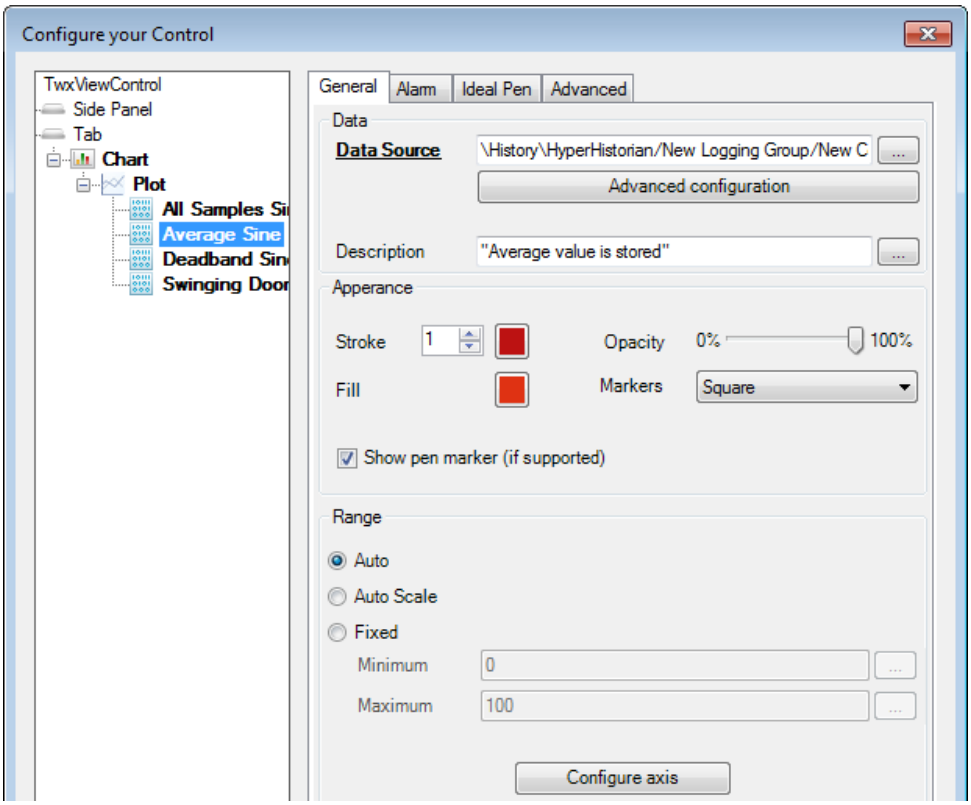


Figure L3.11 - Square Marker Selected

4. Then close the configuration and click the Runtime button.

   In runtime, shown in Figure L3.12, you can hide or show pens and compare the shapes of curves and the number of samples stored or used respectively to draw them.

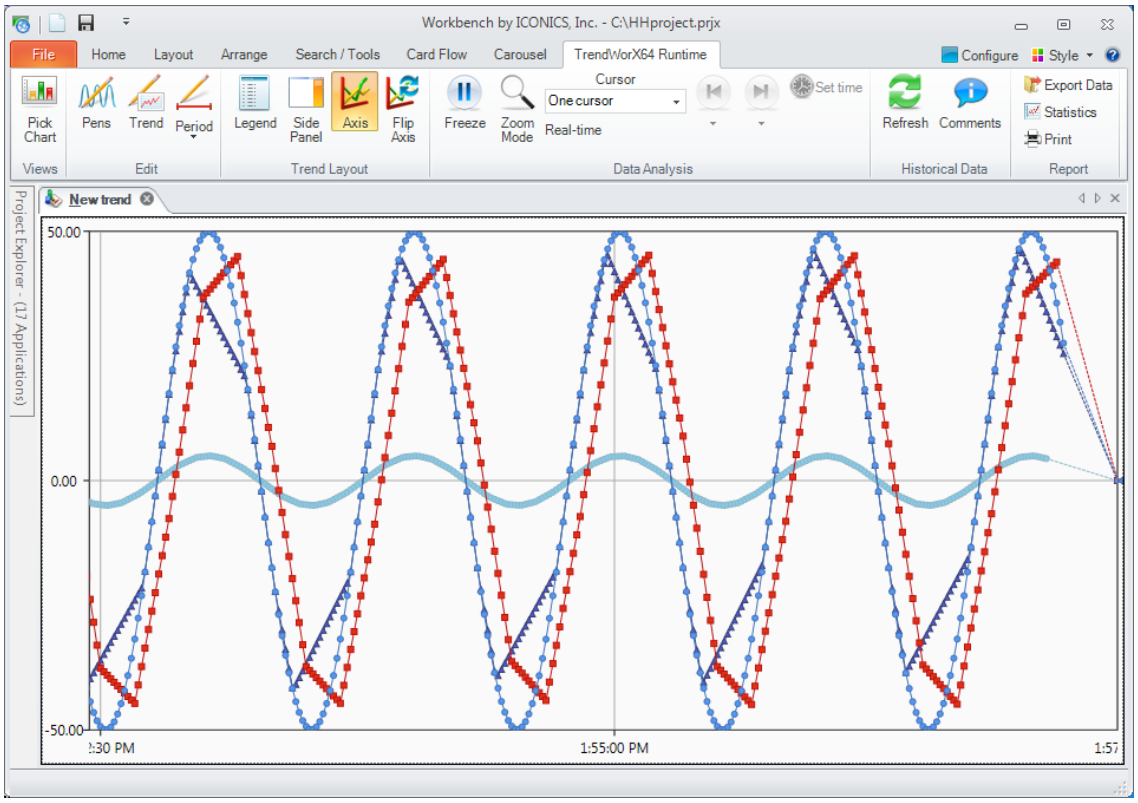


Figure L3.12 – Runtime Configuration

Review: Logging Groups R3

3.1. Chapter Objectives

You should now know and understand the following topics:

- Set up a logging group
- Set up collector groups for a logging group
- Create folders for organizing Hyper Historian tags in a collector group
- Add Hyper Historian tags to a collector group
- Set up filters for a Hyper Historian tag

3.2. Summary

To set up a logging group, you create and configure the logging group, then add the logging group's collector groups and the collector groups' Hyper Historian tags.

A logging group identifies two things: the server where the logger resides, and the logger configuration to be used for collecting and logging Hyper Historian data from a set of OPC tags. It also has any number of collection groups, each of which has its own collection rate and period of calculation for a group of Hyper Historian tags. Each Hyper Historian tag identifies the data source values that are to be collected, along with an assortment of configuration information for collecting the OPC tag's values.

Several options are available when you configure a Hyper Historian tag for collecting data. You can collect a minimum or maximum value, an average value, a standard deviation, a data count, or the most-recent value over a period of time that you specify. You can apply an industry-standard deadband or Swinging Door Compression before logging the data. You can also turn on Operator Annotation support so that operators can make comments to data, for example, as they see it occurring or replaying in the Trend Viewer.

3.3. Terms

Calculation Period	Collector Group	Data Collection Rate
Deadband	Hyper Historian Tag	Log to Disk Filters
Logging Group	Swinging Door Compression	

3.4. Test Your Knowledge

In this chapter, we reviewed logging groups, their collector groups, and the Hyper Historian tags whose data they collect. Before you proceed, make sure you can answer these questions:

1. What is automatically added to a Logging Group when you create it?

    a. Collector Group
    b. Database Group
    c. Aggregation Folder
    d. All of the above
    e. A and B
    f. A and C
    g. B and C
    h. None of the above

2. How many Collector Groups can be in one Logging Group?

    a. It depends on your license
    b. 10
    c. 15
    d. 100
    e. Unlimited

3. What filter(s) can be used for discarding unessential data for a tag?

    a. Log to Disk filter calculations
    b. Deadband filter
    c. Swinging Door Compression filter
    d. All of the above
    e. None of the above

4. True or False? The Calculation Period is used only for collector filter purposes.

5. True or False? It is possible to turn on filtering at the Logging Group level.

6. True or False? You can turn off all filtering for a tag.

7. True or False? You can use a Totalizer function to summarize values of two or more signals.

8. True or False? You cannot collect a tag's local timestamp because only the collector's timestamp can be used.

9. What Hyper Historian Elements can you quickly replicate using the Multiply feature?

    a. Logging groups and collector groups
    b. Hyper Historian tags and tag folders
    c. Aggregate groups
    d. All of the above
    e. Hyper Historian tags only
    f. A and B

10. The Deadband Filter compares each incoming value to the last value it sent to the Logger, then it sends the incoming value to the Logger under what circumstances?

    a. If the difference between the *values* falls *inside* specified limits

    b. If the difference between the *values* falls *outside* specified limits

    c. If the difference between the *timestamps* falls *outside* specified limits

    d. If the difference between the *timestamps* falls *inside* specified limits

    e. None of the above

Aggregate Groups 4

In addition to logging raw data, Hyper Historian can perform calculations to the raw samples that it collects. You can perform further analysis on logged data by using *aggregates* which perform calculations not included in a Hyper Historian tag on cumulative data. An aggregate group has its own calculation period for aggregating data, which lets you easily confine data to a chosen timeframe for performing calculations and reviewing summary information.

4.1. Objectives

After completing this chapter, you should know how to:

- Set up aggregate groups
- Add Hyper Historian tags and create aggregates

4.2. Overview of Aggregate Groups

In Hyper Historian, you can evaluate the ongoing, accumulating data that Hyper Historian tags are amassing. This evaluation lets you isolate types of predefined summary information that includes averages, totals, counts, durations, deltas, and percentages, giving you an ongoing, overall summary of a tag's data from a timeframe of your choosing.

To configure an aggregate evaluation of Hyper Historian tags, you create an **aggregate group** which identifies the timeframe for summarizing data. Then you assign Hyper Historian tags to the group – these are the tags whose logged data you want to evaluate inside that timeframe. Because you are using Hyper Historian tags, all aggregation functions are done with values forwarded to the Logger.

A Hyper Historian tag's evaluation can be disabled and re-enabled again; value changes are ignored when the tag is disabled. The tag's evaluation runs in the background independently. Its results are stored by the Hyper Historian logger based on the parameters that you define for the Hyper Historian tag.

Each logging group has its own, dedicated aggregate groups for the tags that make up that logging group. All aggregate groups are defined in the logging group's

Aggregation folder, which is a folder that is created automatically for each logging group that you define.

4.3. Creating an Aggregate Group

To create an aggregate group for a logging group, right-click the logging group's **Aggregation** folder, then select **Aggregate Group**. The Hyper Historian New Aggregate Group form appears, as shown in Figure 4.1

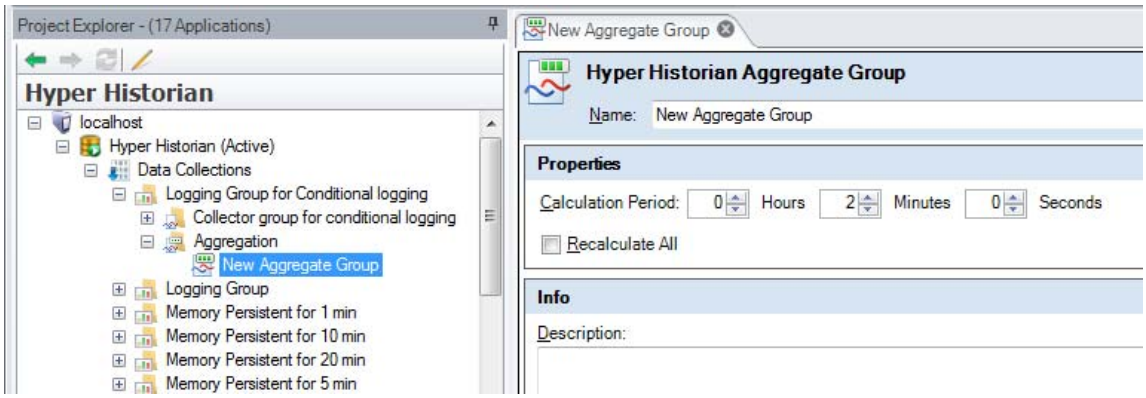


Figure 4.1 - Hyper Historian Aggregate Group

First, enter a Name for your Hyper Historian Aggregate Group in the space provided. Next, you can enter a **Calculation Period** in hours, minutes, and/or seconds; this period becomes the default timeframe for all Hyper Historian tags that are added to the group.

You can also recalculate all existing calculation periods associated with this aggregate group by selecting the **Recalculate All** checkbox. If you put a checkmark in the Recalculate All checkbox, when a Hyper Historian tag is added to the aggregate group, Hyper Historian uses the currently-stored data samples to perform an initial calculation.

4.4. Hyper Historian Tags in Aggregate Groups

You can add a Hyper Historian tag to an aggregate group as long as both the tag and the aggregate group are in the same logging group. A tag's aggregate calculation evaluation runs in the background independently and its results are stored by the Hyper Historian logger based on the parameters defined for the aggregate group. The evaluation's rules are quite simple:

- The evaluation monitors incoming data values and reevaluates aggregates whose values might change as a result of changed source data.

- Evaluation is performed periodically on time intervals detected as changed since the last evaluation; initial evaluation of an interval occurs with at least 1 minute delay.
- When a Hyper Historian tag is added to an aggregate group, the tag's values may get evaluated backwards as well, depending on value of the aggregate group's Recalculate All setting; however, changing the Recalculate All setting has no effect on tags that already belong to the group.

You can add any number of Hyper Historian tags to an aggregate group, and each Hyper Historian tag can belong to multiple aggregate groups. You can even include a tag in an aggregate group multiple times, each time collecting a different type of data.

Let's look at a very simple example of aggregate groups. Figure 4.2 highlights the Hyper Historian tag called "Sine signal" in a logging group for conditional logging. It also shows the aggregation group called "Averages" in the Aggregation folder for the same logging group. Only the tags in this logging group can be added to the Averages aggregate group. We will look at how this is done in the next figure.

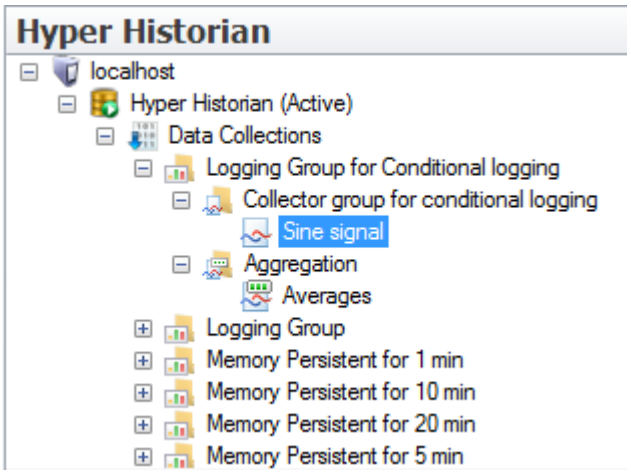


Figure 4.2 – A Hyper Historian Tag and an Aggregate Group for a Logging Group

To add a Hyper Historian tag to an aggregate group, you must go to the navigation tree in the Project Explorer, and expand the **logging group**. Then expand the **collector group**. Double-click the Hyper Historian tag to open it, and then go to the tag's **Tag Aggregates tab**, which is shown in Figure 4.3. On this tab, you can add the tag to any number of aggregate groups defined for the logging group. Each aggregate group appears on its own line in this tab. In Figure 4.3 you can see that the "Sine signal" tag has been added to the "Averages" aggregate group.

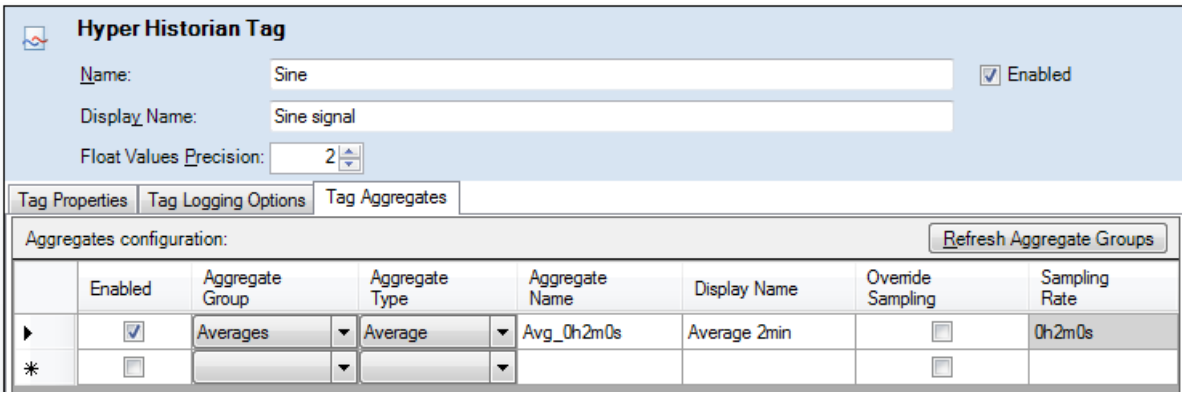


Figure 4.3 - Hyper Historian Aggregate Configurator

To add an aggregate group to the tag's list, start by selecting the group from the **Aggregate Group** drop-down list. This list includes all of the aggregate groups defined for the logging group. If an aggregate group doesn't appear in the list, you can click the **Refresh Aggregate Groups** button and then look in the list again. As soon as you select the aggregate group, the tab starts to build the **Aggregate Name** and **Display Name** for you. We will look at these two columns later.

Once you have added an aggregate group to a line, you can include the Hyper Historian tag in the aggregate group by putting a checkmark in the line's **Enabled** checkbox. Notice that you have the option of adding an aggregate group but leaving the Enable checkbox empty; this prevents the tag's data from being included in the aggregate totals, but leaves you the option of easily including it at any time.

Next, you must select an **Aggregate Type**. Valid types that appear in the drop-down list are limited by the type of signal coming from the Hyper Historian tag. Table 4.1 provides a list of aggregate types. Note that you can collect aggregate data for the tag using more than one of these aggregate types – simply add new lines for the aggregate group, but choose a different aggregate type each time. For example, add one line to collect a type of **Start**, and another line with a type of **End**.

Table 4.1 – Aggregate Types by Category

Category	Aggregate Type
Interpolation aggregate:	**Interpolative** does not retrieve an aggregate. This is used for retrieving interpolated values.

Category	Aggregate Type
Data averaging and summation aggregates:	**Average** retrieves the average data over the re-sample interval. **Time Average** retrieves the time weighted average data over the re-sample interval. **Total** retrieves the sum of the data over the re-sample interval. **Totalize Average** retrieves the totalized Value (time integral) of the data over the re-sample interval.
Data variation aggregates:	**Minimum** retrieves the minimum Value in the re-sample interval. **Maximum** retrieves the maximum Value in the re-sample interval. **Minimum Actual Time** retrieves the minimum value in the re-sample interval and the Timestamp of the minimum value. **Maximum Actual Time** retrieves the maximum value in the re-sample interval and the Timestamp of the maximum value. **Range** retrieves the difference between the minimum and maximum Value over the sample interval.
Counting aggregates:	**Annotation Count** retrieves the number of Annotations in the interval. **Count** retrieves the number of raw Values over the re-sample interval.
Time aggregates:	**Start** retrieves the Value at the beginning of the re-sample interval. The time stamp is the time stamp of the beginning of the interval. **End** retrieves the Value at the end of the re-sample interval. The time stamp is the time stamp of the end of the interval. **Delta** retrieves the difference between the first and last Value in the re-sample interval.

Category	Aggregate Type
Data quality aggregates:	**Duration Good** retrieves the duration of time in the interval during which the data is good. **Duration Bad** retrieves the duration of time in the interval during which the data is bad. **Percent Good** retrieves the percent of data (0 to 100) in the interval which has good StatusCode. **Percent Bad** retrieves the percent of data (0 to 100) in the interval which has bad StatusCode. **Worst Quality** retrieves the worst StatusCode of data in the interval.

Each Hyper Historian tag can have its own sampling rate. You can use the aggregate group's default **Calculation Period** (which initially appears in the tag's **Sampling Rate** column) *or* you can override it specifically for this tag. To override it, put a checkmark in the **Override Sampling** column, and then click in the Sampling Rate column. When you do this, an ellipsis button appears. Click the ellipsis button to open the dialog box in which you can enter an aggregate sampling rate for the tag in hours, minutes, and/or seconds; click OK to save the changes and close the dialog box. To cancel the overridden rate, simply remove the checkmark from the **Override Sampling** checkbox.

Sampling intervals are aligned to the nearest earlier occurrence of the nearest higher time bin boundary. For example, a sampling rate of seconds will yield sampling intervals aligned to a minute; having a sampling rate of 5 seconds will yield intervals starting at +0, +5, +10… +55 seconds within a minute.

> **Note:** A sampling rate that is not a divisor of the given time bin may yield anomalies in sampling interval alignment during re-evaluations. For that reason, you should restrict available sampling rates to divisors.

At this point, you might want to look at the names displayed in the **Aggregate Name** and **Display Name** columns. These names are created for you, and both tell you the aggregate's type and the sampling rate. For example, before you choose an aggregate type, the *aggregate name* might be ***None_0h2m0s***. "None" means that an aggregate type hasn't been chosen yet, and "0h2m0s" reflects the default sampling rate, which is the aggregate group's calculation rate. After you finish, though, the name might be

WstQual_8h0m0s. This means that the aggregate collects the Worst Quality in the past eight hours.

The *display name* is similar, but uses more descriptive words. If you follow the examples provided above for the aggregate name: before an aggregate type is specified the display name would be ***None 2min***, but after you finish the name would be ***Worst Quality 8hrs***.

As was stated earlier, you can aggregate the tag's data in multiple ways. You can re-use an aggregate group to do this. But you can also define multiple aggregrate groups for the logging group, and use any number of these groups to aggregate the tag's data.

For ideas of how you might use aggregates, review the aggregate types in Table 4.1.

4. Lab: Aggregate Groups L4

In this section, you will create an Aggregate Group and add Hyper Historian tags to it to provide you with statistical data based on those Hyper Historian Tags.

Estimated time to complete this lab: 20 minutes

4.1. Task Checklist

In this lab, you will complete the following tasks:

- Create an aggregate group
- Add Hyper Historian tags to it

4.2. Support Files

You do not need any supporting files to complete this lab.

4.3. Create an Aggregate Group

1. Open Hyper Historian in the Workbench as explained earlier in the course. Expand your configuration database in the Project Explorer.

2. Expand the **Data Collections** node and the **Logging Group** node. Right-click the **Aggregation** node and select **Aggregate Group**, as shown in Figure L4.1. You may remember that the Aggregation node was created automatically when you saved the logging group. The Aggregation node is permanent; it cannot be deleted. All aggregate groups must be added to this node.

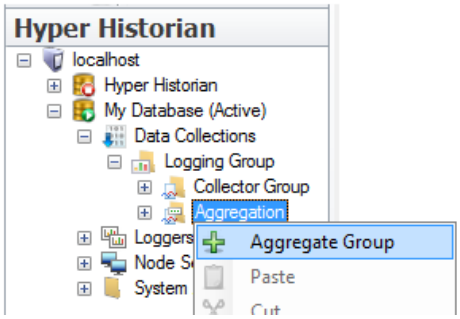


Figure L4.1 - Adding an Aggregate Group to the Aggregation Folder

3. The New Aggregate Group configuration opens on the right. First, change the new group's name to "Aggregate Group."

4. Then set the **Calculation Period** to **3 minutes**. This period is the amount of time for each block of data that we will generate an aggregate value for. Notice that the calculation period is really the only defining feature of the group; this period is used by default for Hyper Historian tags that reference this group, but it can be overridden for any tag.

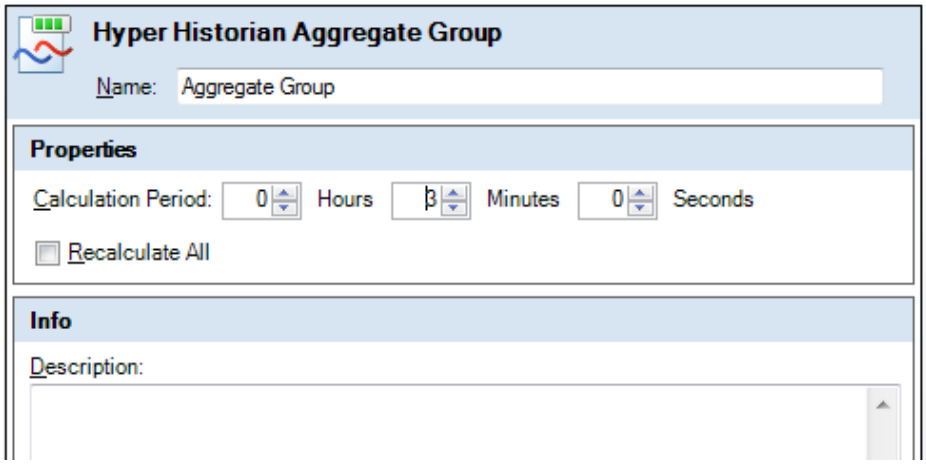


Figure L4.2 - Adding an Aggregate Group

5. Add an informative description of the group and its use. Click **Apply** to save the new aggregate group.

4.4. Adding a Hyper Historian Tag to the Group

Next we will add a Hyper Historian tag to the aggregate group. To do this, you must return to one of the Hyper Historian tags you added to the logging group earlier.

1. In the Project Explorer, expand the logging group's **Collector Group** and double-click the **All Samples Sine** Hyper Historian tag to open its configuration, and go to the configuration's **Tag Aggregates** tab, as shown in Figure L4.3.

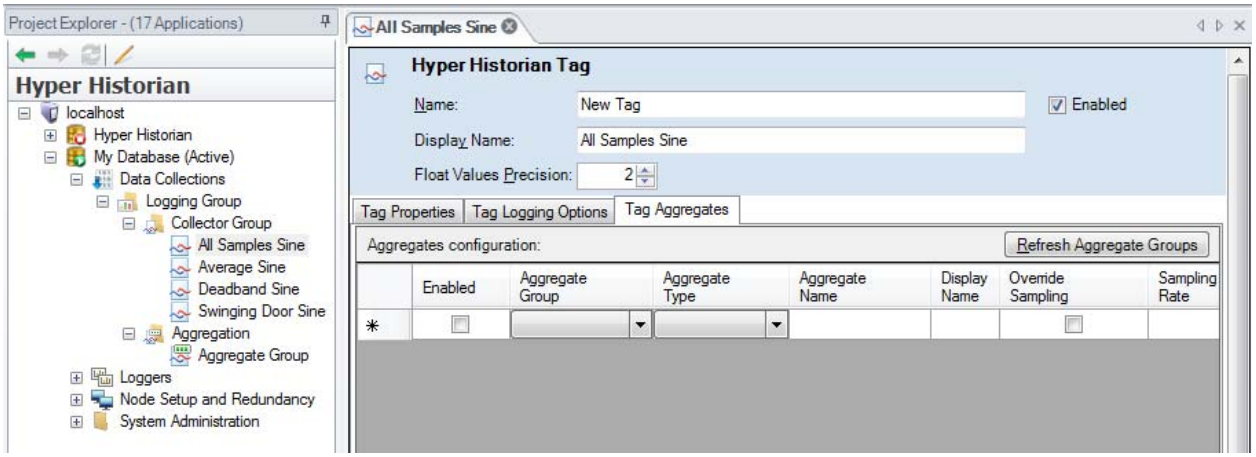


Figure L4.3 - Hyper Historian Tag Configuration, Tag Aggregates Tab

2. In the **Aggregate Group** column, select the group called **Aggregate Group** from the drop-down list.

3. In the Aggregate Type column, select **Average**.

4. Put a checkmark in the **Enabled** column.

5. We are not going to override the Calculation Period, so leave the Override Sampling checkbox empty. Note that the Sampling Rate column displays the aggregate group's calculation period of 3 minutes.

6. If you add an aggregate line that you don't want to use, leave its Enabled checkbox empty, as shown in the second line in Figure L4.4.

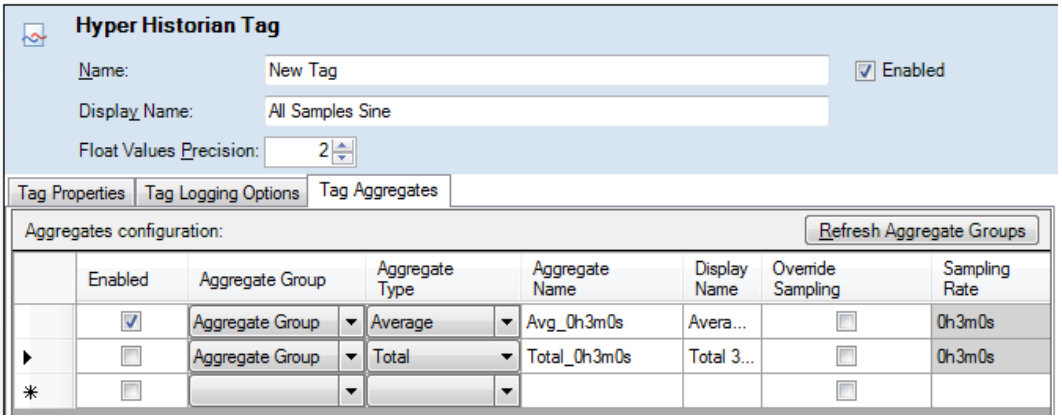


Figure L4.4 - Aggregates Defined for a Hyper Historian Tag

7. Click **Apply** to save the Hyper Historian's tag with the configuration changes you just added.

8. Repeat the procedure for the on **Average Sine** Hyper Historian tag, except for this tag, calculate its **Maximum**.

9. Later you can look at the results in TrendWorX64 Viewer. Remember that you have to wait an appropriate amount of time so that your aggregates get created.

Review: Aggregate Groups R4

4.1. Chapter Objectives

You should now know and understand the following topics:

- Set up an aggregate group
- Add Hyper Historian tags and create aggregates

4.2. Summary

In Hyper Historian, you can evaluate the ongoing, accumulating data that a Hyper Historian tag is amassing inside a specified timeframe. This evaluation lets you isolate types of predefined summary information including averages, totals, counts, durations, deltas, and percentages, giving you an ongoing, overall summary of a tag's data within a timeframe that you choose.

You create **aggregate groups** to identify the timeframe for summarizing data. Then you assign Hyper Historian tags to the group – these are the tags whose logged data you want to evaluate inside that timeframe.

A Hyper Historian tag's aggregate evaluation can be disabled and re-enabled again; value changes are ignored when the tag is disabled. The tag's evaluation runs in the background independently. Its results are stored by the Hyper Historian logger based on the parameters defined for the Hyper Historian tag.

4.3. Terms

Aggregate Aggregate Group

4.4. Test Your Knowledge

In this chapter, we discussed aggregate tags and how you configure them. To review, answer the questions below.

1. True or False? You can create an aggregate Hyper Historian tag based on a real-time data point.

2. True or False? Values of a new aggregate Hyper Historian tag may be evaluated backwards.

3. True or False? You can show the aggregate Hyper Historian tags in TrendWorX64 Viewer.

4. How do you modify a Hyper Historian tag's aggregates?

    a. You must go to the Aggregation folder and modify the tag's aggregate group

    b. You open the Hyper Historian tag and modify the settings on its Tag Aggregates tab

    c. You add an aggregate tag to the Aggregation folder

    d. You cannot modify it; instead, you must delete the tag's aggregates and then recreate them

    e. None of the above

5. True or False? You cannot retrieve a tag's values from <u>both</u> the beginning and end of a sampling period.

6. True or False? You can count the number of operator annotations for a sampling period.

7. True or False? You cannot use aggregates to retrieve data relating to the quality of a machine's output.

8. True or False? You cannot aggregate a tag's data using an aggregate group from another logging group.

Additional Settings 5

There are some additional features and settings in Hyper Historian that you should be familiar with. In this chapter we will take a look at them.

5.1. Objectives

After completing this chapter, you should understand the following topics:

- Workbench capabilities with Hyper Historian
- Choosing a license mode
- Configuring a collector node
- Configuring the local logging node
- Configuring automatic archiving
- Restoring archived files

5.2. Hyper Historian in the Workbench

You configure Hyper Historian using the Workbench. The Workbench is part of the Hyper Historian installation and is a multi-functional, centralized, web-based environment for configuring products.

The Workbench can also act as an advanced operator interface for visualization and service management. It has integrated project management and remote, pack-and-go deployment capabilities.

5.3. License Mode (Enterprise Edition)

Hyper Historian is available in two editions: *Standard* and *Enterprise*. If you are using the Enterprise edition, you must choose your license mode as will be explained in this section.

Both editions support the connectivity, data collection and logging, archiving, and data playback capabilities of the Hyper Historian.

In addition to supporting these core capabilities, the *Enterprise* edition supports:

- remote distributed data collection,
- data redundancy both at the collector level and at the data logger/database level,
- store-and-forward capability, and
- optimization for multi-core and multi-processor systems.

The *Enterprise* edition is optimized for distributed data collection in which the data collectors reside on separate machines. The *Standard* edition, on the other hand, is optimized for a single PC or workstation where all of the Hyper Historian components are running on one machine.

If you have Hyper Historian *Enterprise* edition, you can run the Hyper Historian Logger in Enterprise *or* Standard mode if you are running Hyper Historian on a single PC or workstation (that is, a non-redundant single workstation). Standard mode provides better logging performance and potentially high throughput for a single PC solution.

Under **System Administration** in the Project Explorer, double-click on **System Settings** to display a configuration. The **Options** tab displays.

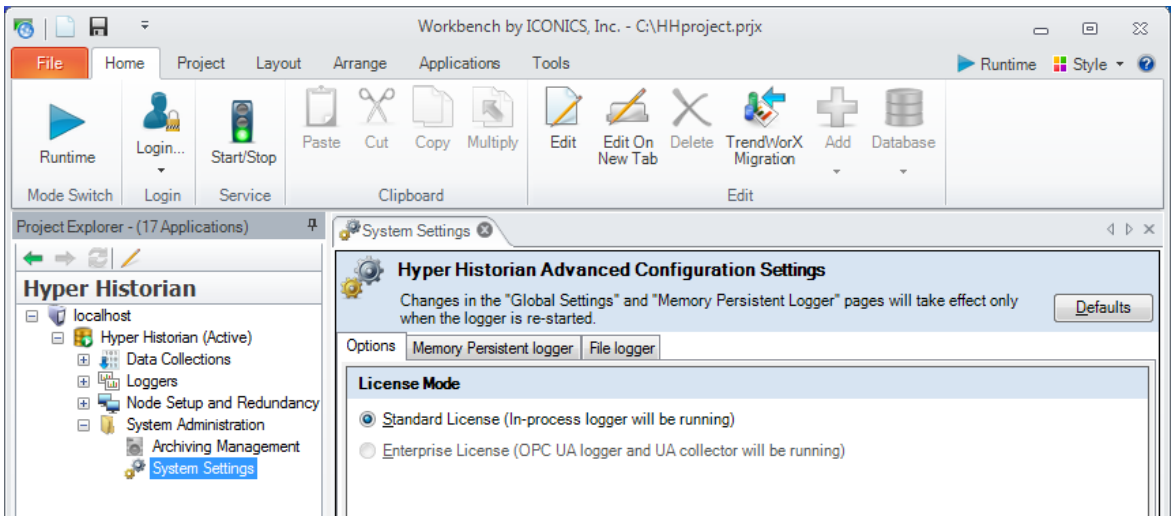


Figure 5.1 - Configuring the License Mode

On the Options tab, Enterprise edition users must set the license mode to Standard License or Enterprise License:

- Selecting **Standard License** results in Hyper Historian using the in-process version of the Hyper Historian Logger and Collector which is optimized for single-PC or -workstation performance. If you select this option, you cannot use the extra features that are available with the Enterprise license.

- Selecting **Enterprise License** results in Hyper Historian using the OPC UA version of the Hyper Historian Logger and Collector which is optimized for a distributed system – one that supports redundancy, store-and-forward, and optimized performance on multi-CPU systems.

5.4. Collector Node

One of the first settings you may notice in the Project Explorer navigation tree (on the left side of the Workbench) is **Node Setup and Redundancy**. Here you configure the local collector node. The collector node specifies the location of the data collector.

> **NOTE:** If you are making a Quick Start, it is not necessary to adjust any settings, because they are set to default upon installation of Hyper Historian; default settings are for a non-redundant single workstation setup (that is, a non-redundant Hyper Historian logger with one local collector).

To define a collector node, right-click **Node Setup and Redundancy**, then select the **Collector**. The Hyper Historian Collector configuration interface appears. Hyper Historian server will send the configuration to the remote collector machine upon entering runtime. There is no need to configure the remote collector on the remote node.

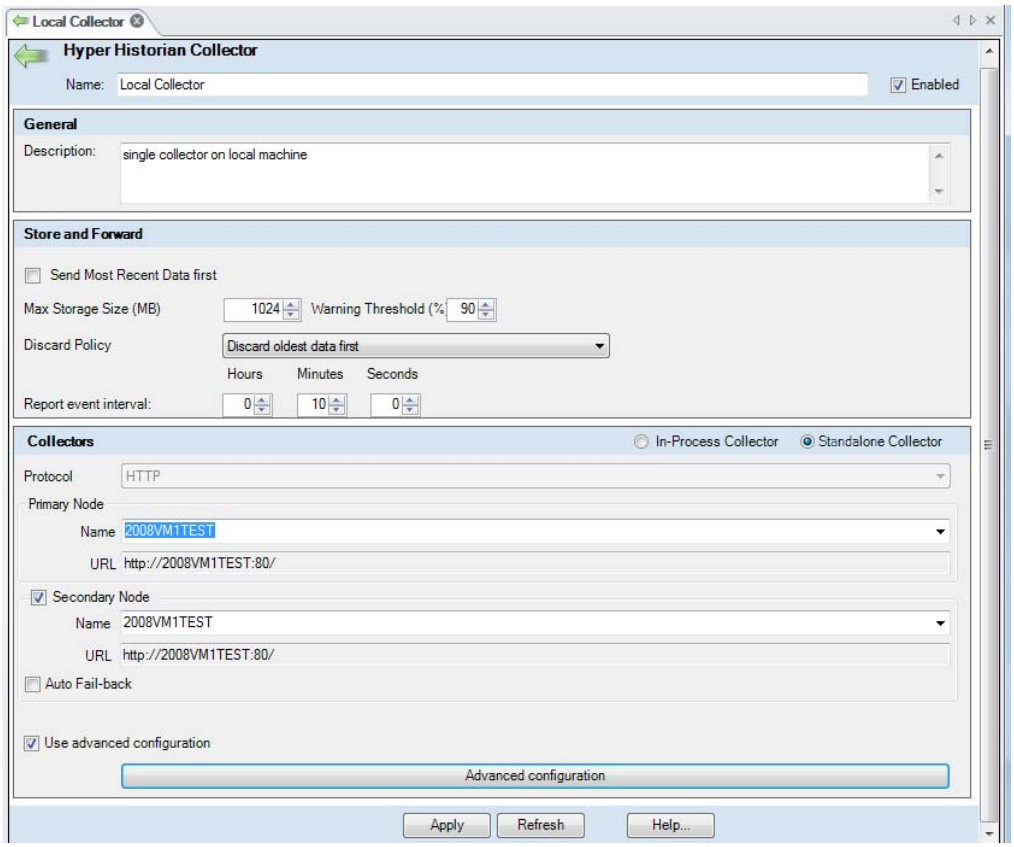


Figure 5.2 - Hyper Historian Collector Configurator

Use the configurator to enter a **Name** and **Description** for the Hyper Historian Collector for the selected host; enable the collector by putting a checkmark in the **Enabled** checkbox.

5.4.1. Store-and-Forward

Now you can optionally set your redundancy policies by specifying information for the **Store and Forward** feature. This feature caches data if the connection to a collector is lost. When the connection to the collector is restored, the cache is flushed and the two databases (that is, the primary and secondary nodes that you specify in the Collectors section) are automatically synchronized. You can use store-and-forward regardless of whether you use a backup node for redundancy in the Collectors section of the form.

If you want the most-recently collected data to be forwarded *before all other data*, click a checkmark into the **Send Most Recent Data First** checkbox. If you leave this checkbox empty, data will be forwarded in the order in which it is collected.

The following options determine how much data will be stored. Specify the **maximum storage size** for data, in megabytes; when storage reaches this limit, data will be discarded in accordance with the discard policy that you specify – you can discard the *oldest* data first, or discard the *newest* data first. Then specify the percentage of the maximum size at which a **warning** is given. Make sure this percentage gives you enough time to react to the warning and safeguard any data that is in jeopardy. Make sure you have a procedure in place for handling such an occurrence.

> **NOTE:** The collected data is stored *temporarily* on the hard drive; it is not logged to the database.

Specify the **report event interval** in hours, minutes, and/or seconds

5.4.2. Primary Collector

Next, select either **In-Process Collector** (the local collector) or **Standalone Collector** (the remote collector) as your collector. Selecting Standalone Collector enables the remaining options in the Collectors pane so that you can set up data redundancy. The remaining steps assume you have selected **Standalone Collector**. (If the Standalone Collector option is grayed out, it is because Enterprise edition is not installed, or because Enterprise edition is installed but you are using it with a Standard License.)

Use the **Protocol** drop-down list to select either HTTP or TCP.

For the primary node, enter a **Name** or select one from the drop-down list. Once you've identified the primary node, its URL appears in the **URL** field.

If the URL does not appear, or if you want to use a different URL, put a checkmark in the **Use Advanced Configuration** checkbox, and then click the **Advanced Configuration** bar at the bottom of the form. This opens a dialog box so you can change the URL. Note, however, that if you change the URL, the same change may be required in the node's configuration.

5.4.3. Secondary Collector

You now have the option to identify a secondary collector to be used as a stand-by or backup node. If the primary collector becomes unavailable for any reason, the secondary collector will take over collection of data.

To identify the secondary node to be used as backup collector, put a checkmark in the **Secondary Node** checkbox. Then select the secondary node from the **Name** drop-down list. Once you've identified the secondary node, its URL appears in the **URL** field. If the URL does not appear, or if you want to use a different URL, put a checkmark in the **Use Advanced Configuration** checkbox, and then click the **Advanced Configuration** bar at the bottom of the form. This opens a dialog box so you can change the URL. Like the primary node, if you change the URL of the secondary node, the same change may be required in the node's configuration.

By default, the secondary node will continue to be used as the primary collector, even after the primary node becomes available again. This happens because, by default, the **Auto Fail-back** checkbox is empty. However, if you want the primary node to automatically take over again as soon as it becomes available again put a checkmark in the **Auto Fail-back** checkbox.

5.5. The Local Logging Server

Next you can configure the local logging server. To do this, right-click on **Node Setup and Redundancy**, then select **Local Logging Server**. This displays a configuration form for the local logging server which is shown in Figure 5.3.

The **Local Logging Server** settings are similar to the Collector's, but not identical. Here, you enter a **Name** and **Enable** the Hyper Historian Logging Server, enter a **Description**.

Optionally, to make the server's historical data read-only, put a check mark in the **Disable HDA Editing** check box. If you do this, you won't be able to edit any Historical data being logged on the server using any programmatic interfaces.

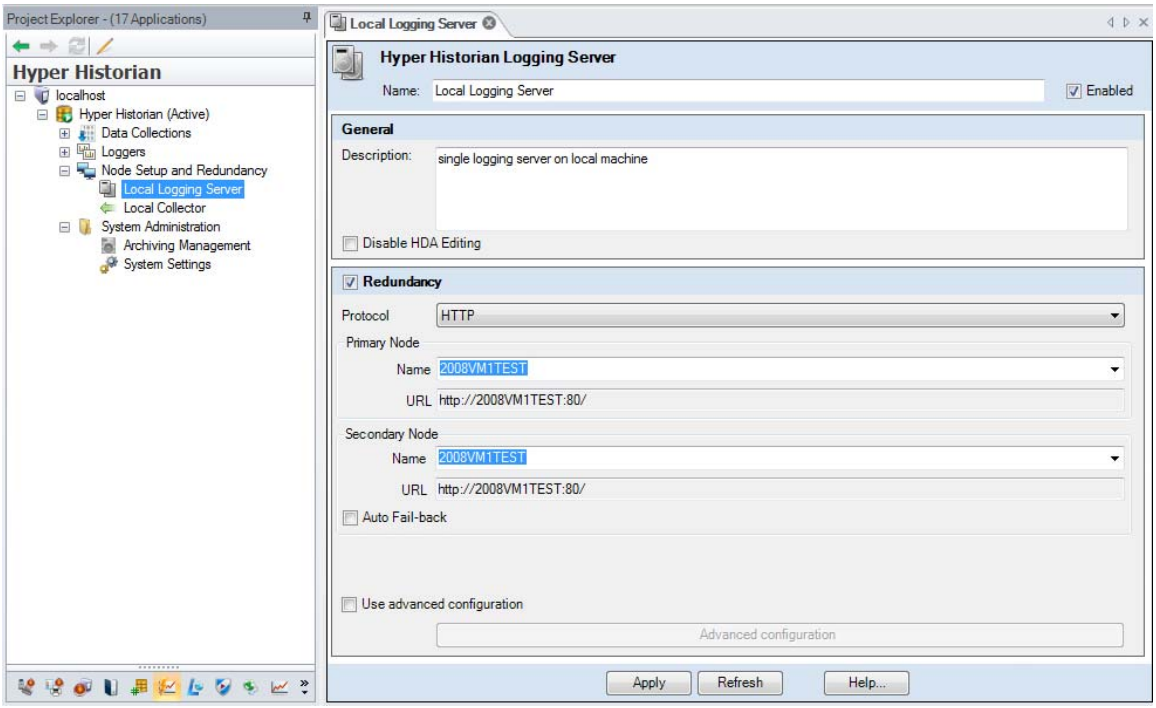


Figure 5.3 - Local Logging Server Settings

You have the option to put a checkmark next to **Redundancy** to enable it. Then specify the **Protocol**, the name of the **Primary Node** for the logging server, the name of the **Secondary Node**, and your **Auto Fail-back** option. These options, as well as the **Advanced Configuration** options, work for logging the same way that they work for collectors.

There are additional settings you need to configure for the logging server. These additional settings are what we will be discussing next.

5.5.1. Logging Server Storage Path

Before using Hyper Historian for the first time, you must set the logging server storage path. The Logging Server storage path is the location where Hyper Historian stores the plant data that it collects. The default folder for the Logging Server storage path is located on the C: drive in:

C:\Program Files\ICONICS\GENESIS64\Components\HHLogger\BaseLogger.1\Data Folder

It is important that you change this default path, especially if the C: drive has a low amount of free storage space.

While still in System Settings, click the **Memory Persistent Logger** tab; the tab appears, with its **General** sub-tab displayed.

At the bottom of the **General** sub-tab in the **Logging Server Storage Path** field (shown in Figure 5.4), change the default path by clicking the **Browse** button to find and choose your desired location.

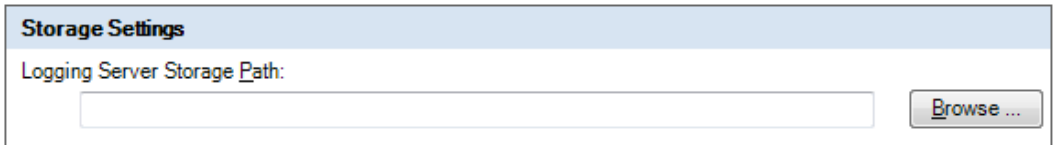


Figure 5.4 - Logging Server Storage Path

5.5.2. Managing Disk Space

It is also important to set the **Disk Space Management** settings. With these settings, you can set thresholds that warn you when disk space is getting low. To do this, expand **System Administration** in the navigation tree. Then double-click **System Settings**. This displays the System Settings Configurator with the **Options** tab on top. At the bottom of the **Options** tab, in the Disk Space Management section, specify the amount of disk space (in megabytes) that will give you:

- A warning
- An error message, in which the Data Logger stops logging

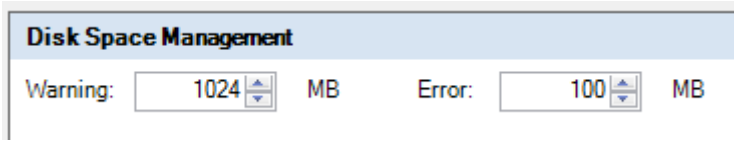


Figure 5.5 - Disk Space Management

5.5.3. Memory Persistent Tuning Settings

Now go to the Memory Persistent Logger tab's **Tuning** sub-tab where you'll find cache settings for Memory Persistent Loggers. Caching is used to house data received from lower processing levels and in preparation for forwarding to the core. When data is received from processing levels, it is stored to a cache (as much as space requirements allow), or forwarded to storage. When a client requests data, the cache is used as a data

source. When the client confirms it has received the data, cached counterparts are removed and the cache gets backfilled.

> **NOTE:** These advanced settings are quite advanced. If you are not an expert, we recommend that you do not change these settings. However, if you modify any parameter on the **Tuning** sub-tab, you must restart the logger to apply the changes. (To do this, click the traffic light icon in the **Home** tab of the ribbon and wait for it to turn red (indicating the logger has stopped). Then click it a second time to restart the logger.)

Among the parameters available on this tab are:

- Shadow Flush Time
- Index Cache Size
- Data Cache Size
- Memory Only Index Cache Size
- Memory Only Data Cache Size

Various parameters are provided on this tab. As you can see in Figure 5.6, a description of each parameter is provided on the sub-tab. Depending on the parameter, you can specify whether you want to include a Boolean Value, enter a Numeric Value or enter a Text Value. For most parameters, the default value type is Numeric.

> **NOTE:** Clicking the **Defaults** button resets *all* parameters on *all* System Settings tabs back to their original settings.

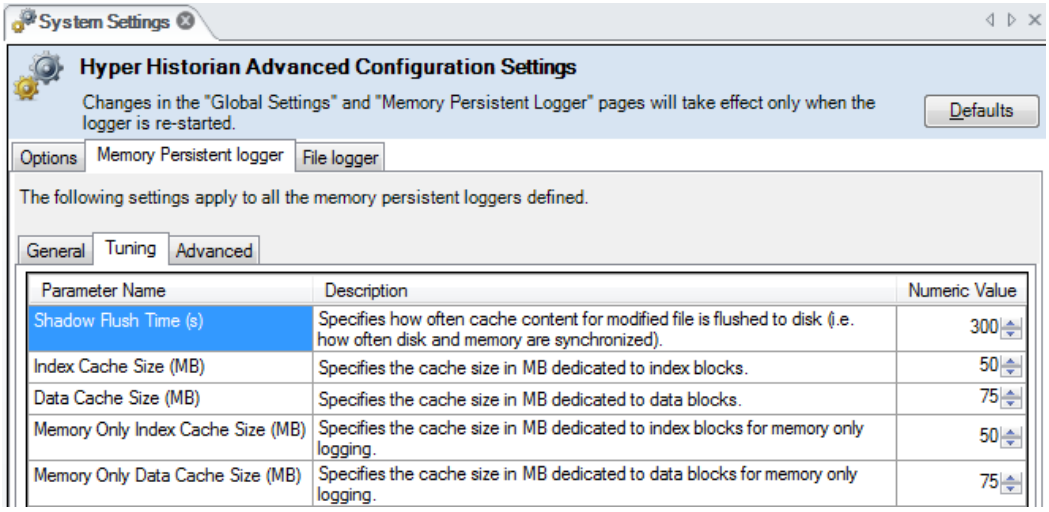


Figure 5.6 – Memory Persistent Logger's Settings for Cache Sizes

5.5.4. Memory Persistent Settings Used on Logger Startup

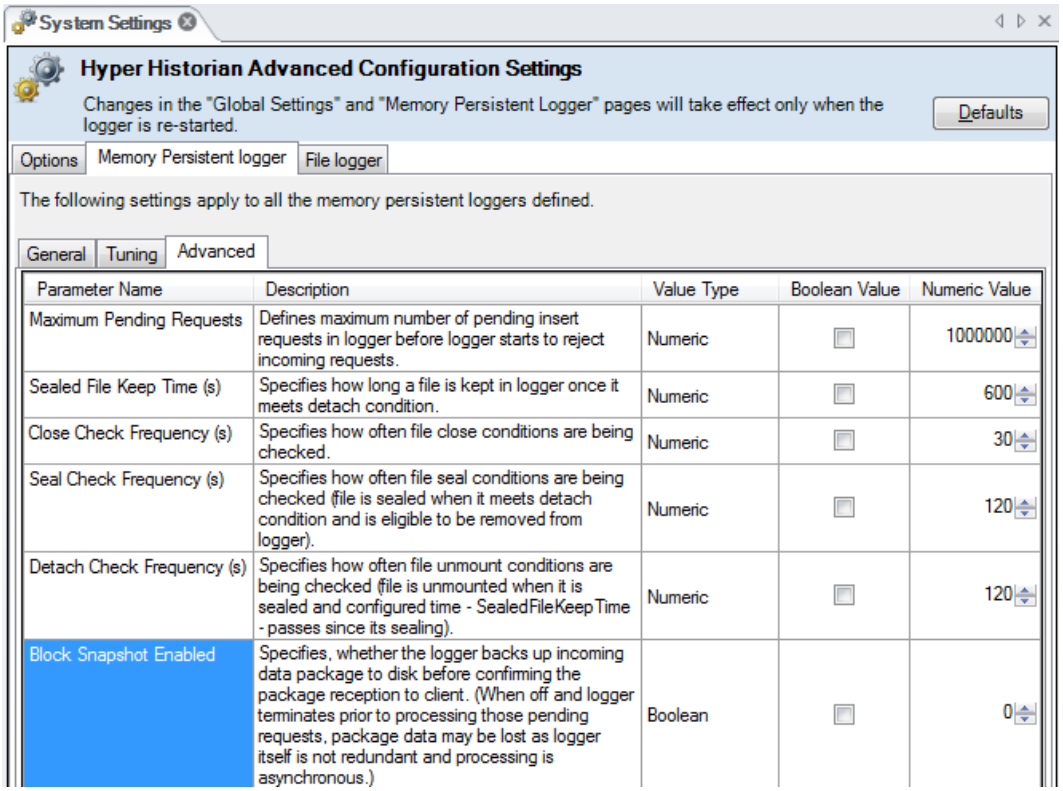
Now go to the Memory Persistent Logger tab's **Advanced** sub-tab where you'll find Hyper Historian's **Advanced Configuration Settings**. These settings are used for Memory Persistent Loggers on logger startup only.

> **NOTE:** These advanced settings are quite advanced. If you are not an expert, we recommend that you do not change these settings. However, if you modify any parameter on the **Tuning** sub-tab, you must restart the logger to apply the changes.

Among the parameters available on this tab are:

- Maximum Pending Requests
- Sealed File Keep Time
- Close Check Frequency
- Seal Check Frequency
- Detach Check Frequency
- Block Snapshot Enabled

Figure 5.7 shows the sub-tab.



Parameter Name	Description	Value Type	Boolean Value	Numeric Value
Maximum Pending Requests	Defines maximum number of pending insert requests in logger before logger starts to reject incoming requests.	Numeric		1000000
Sealed File Keep Time (s)	Specifies how long a file is kept in logger once it meets detach condition.	Numeric		600
Close Check Frequency (s)	Specifies how often file close conditions are being checked.	Numeric		30
Seal Check Frequency (s)	Specifies how often file seal conditions are being checked (file is sealed when it meets detach condition and is eligible to be removed from logger).	Numeric		120
Detach Check Frequency (s)	Specifies how often file unmount conditions are being checked (file is unmounted when it is sealed and configured time - SealedFileKeepTime - passes since its sealing).	Numeric		120
Block Snapshot Enabled	Specifies, whether the logger backs up incoming data package to disk before confirming the package reception to client. (When off and logger terminates prior to processing those pending requests, package data may be lost as logger itself is not redundant and processing is asynchronous.)	Boolean		0

Figure 5.7 – Memory Persistent Logger's Advanced Settings

5.5.5. File Logger Settings

Now go to the **File Logger** tab. On this tab there are several sub-tabs, also. These tabs provide cache settings for File Loggers, as well as other File Logger settings. These settings are used on logger startup only.

> **NOTE:** These advanced settings are quite advanced. If you are not an expert, we recommend that you do not change these settings. However, if you modify any parameter on the **Tuning** sub-tab, you must restart the logger to apply the changes.

Each tab's settings are shown in the illustrations that follow. Figure 5.8 shows the following settings in the File Logger's Cache sub-tab:

- Flush Time
- Full Garbage Collection Period
- Maximum Cache Size
- Maximum Lifetime of Unused Data Blocks
- Maximum Lifetime of Unused Index Blocks

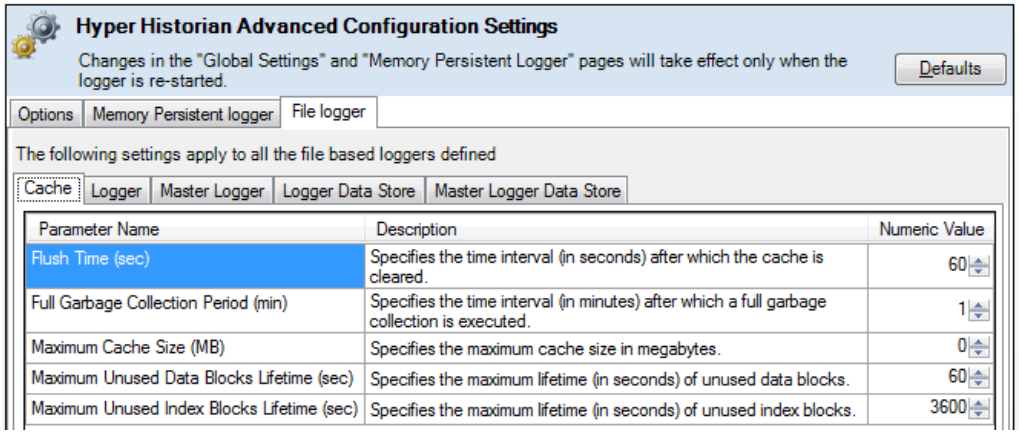


Figure 5.8 – File Logger - Cache Sub-tab

Figure 5.9 shows the following settings in the File Logger's Logger sub-tab:

- Operation Retry Interval
- Maximum Operation Queue Length
- Priority Increment Period
- Operation Queue Priority Count

Cache | Logger | Master Logger | Logger Data Store | Master Logger Data Store

Parameter Name	Description	Numeric Value
Operation Retry Interval (sec)	Specifies the time interval (in seconds) after which the logger attempts to re-execute a failed operation.	10
Maximum Operation Queue Length	Specifies the maximum operation queue length.	1000
Priority Increment Period (ms)	Specifies the time period (in milliseconds) after which queue priority is increased.	250
Operation Queue Priority Count	Specifies the count of priorities in the operation queue.	20

Figure 5.9 – File Logger - Logger Sub-tab

Figure 5.10 shows the following settings in the File Logger's Master Logger sub-tab:

- Alternative Data Index Directory
- Archive Data Store Directory
- Data File Extension
- Data Store Directory
- Index File Extension
- Master Logger Directory
- Master Logger File
- Maximum Online Loggers Count
- Maximum Inactivity Period
- Operation Retry Interval

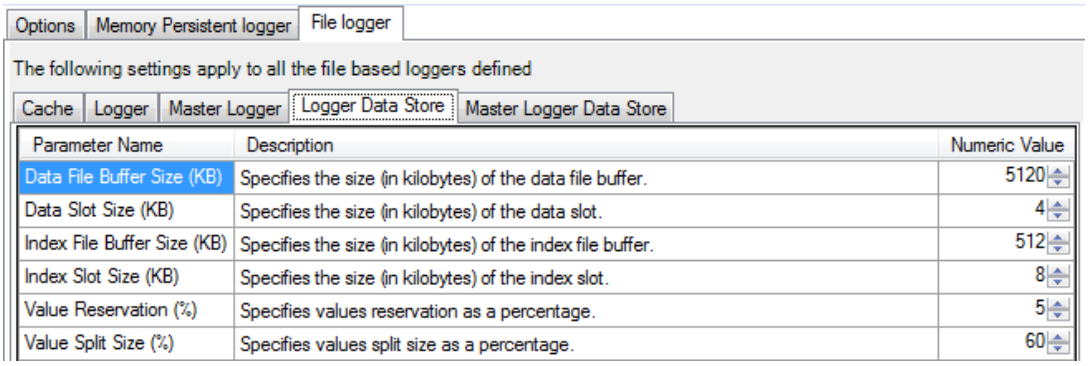
Cache | Logger | Master Logger | Logger Data Store | Master Logger Data Store

Parameter Name	Description	Value Type	Numeric Value	Text Value
Alternative Data Index Directory	Specifies an alternative directory path used to store data indices.	Text	0	
Archive Data Store Directory	Specifies the directory path used to store archive data.	Text	0	
Data File Extension	Specifies the extension for logged data files.	Text	0	hhd
Data Store Directory	Specifies the directory path used to store logged data.	Text	0	
Index File Extension	Specifies the extension for index files.	Text	0	hhi
Master Logger Directory	Specifies the directory path of the master logger.	Text	0	
Master Logger File	Specifies the pathname of the master logger binary file.	Text	0	Datastores.bin
Maximum Online Loggers Count	Specifies the maximum count of online loggers allowed.	Numeric	100	
Maximum Inactivity Period (min)	Specifies the maximum inactivity period (in minutes) allowed for the master logger.	Numeric	10	
Operation Retry Interval (sec)	Specifies the time interval (in seconds) after which the master logger attempts to re-execute a failed operation.	Numeric	10	

Figure 5.10 – File Logger – Master Logger Sub-tab

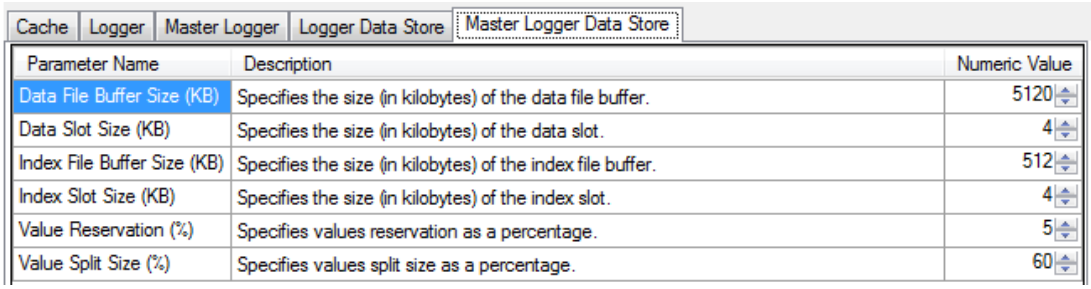
Figure 5.11 and Figure 5.12 show the following settings for the Logger's data store and the Master Logger's data store, respectively:

- Data File Buffer Size
- Data Slot Size
- Index File Buffer Size
- Index Slot Size
- Value Reservation
- Value Split Size



Parameter Name	Description	Numeric Value
Data File Buffer Size (KB)	Specifies the size (in kilobytes) of the data file buffer.	5120
Data Slot Size (KB)	Specifies the size (in kilobytes) of the data slot.	4
Index File Buffer Size (KB)	Specifies the size (in kilobytes) of the index file buffer.	512
Index Slot Size (KB)	Specifies the size (in kilobytes) of the index slot.	8
Value Reservation (%)	Specifies values reservation as a percentage.	5
Value Split Size (%)	Specifies values split size as a percentage.	60

Figure 5.11 – File Logger – Logger Data Store Sub-tab



Parameter Name	Description	Numeric Value
Data File Buffer Size (KB)	Specifies the size (in kilobytes) of the data file buffer.	5120
Data Slot Size (KB)	Specifies the size (in kilobytes) of the data slot.	4
Index File Buffer Size (KB)	Specifies the size (in kilobytes) of the index file buffer.	512
Index Slot Size (KB)	Specifies the size (in kilobytes) of the index slot.	4
Value Reservation (%)	Specifies values reservation as a percentage.	5
Value Split Size (%)	Specifies values split size as a percentage.	60

Figure 5.12 – File Logger – Master Logger Data Store Sub-tab

5.6. Archive Settings

Older data stored by Hyper Historian can be automatically archived when it is no longer needed for online playback, reports, or charts. When Hyper Historian archives data files, it detaches data from the system of ICONICS products and offloads it to a specified folder.

Archiving means another level of security and space management. The detached files may be moved to an external storage to free up space on the logger's hard drive, and secure them away from the production process. Furthermore, because Hyper Historian's archival process is automatic, you are not burdened with a manual process that diverts your attention and takes up valuable time. Finally, at any time, you can

manually restore any archived file to the online system, making it available to ICONICS products once again.

5.6.1. Automatic Archiving Settings

To configure archiving, expand **System Administration** in the Project Explorer, and double-click **System Settings**. Go to the **Memory Persistent Logger** tab where you'll see the **Archive Settings** section on the General sub-tab, shown in Figure 5.13. Archive settings for file-based loggers are configured at the file logger (see **2.11 – Archiving** for more information).

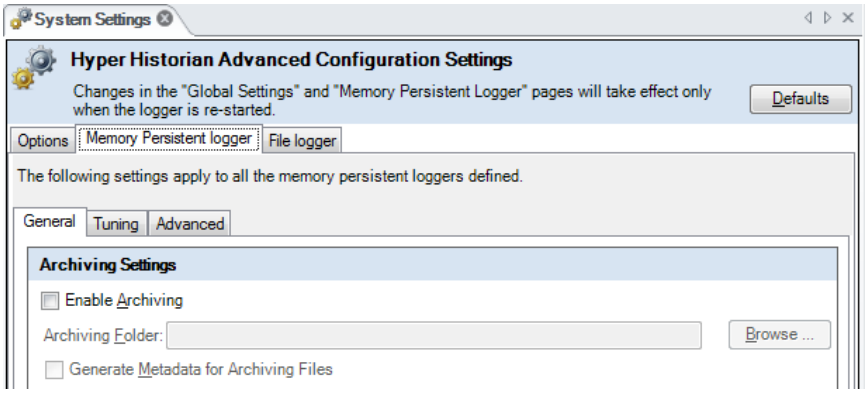


Figure 5.13 - Starting Hyper Historian Archiving

Enable archiving by putting a checkmark in the **Enable Archiving** checkbox, then entering the path and folder name for the **Archiving Folder** that is to contain the archived (detached) data files.

Hyper Historian will also generate metadata for archived files if you put a checkmark in the **Generate Metadata for Archiving Files** checkbox. An archived metadata file contains detailed information about the archived file, including a list of the tags whose data is stored in the archive file. You will find that metadata is very helpful if you need to restore data later on. You need to, of course, consider that it occupies additional storage space, but you may find it well worth it.

The archiving operation moves the file from the logger folder to the archive folder, as configured. The restoring operation makes a copy of the archived file folder back to the logger folder.

5.6.2. Important Considerations

Note that archiving will begin only after you have configured the logger with an event trigger that starts archiving. In this way, you can pinpoint data that needs to be stay online when other data gets archived immediately or only under particular circumstances.

After making changes to the system settings, make sure you restart the Hyper Historian logger. To do this, click the traffic light icon in the **Home** tab of the ribbon and wait for it to turn red (indicating the logger has stopped). Then click it a second time to restart the logger.

Once archiving is enabled and Hyper Historian has begun archiving files, make sure you routinely back up the archived files to CD, DVD, tape, or some other permanent type of media for long-term storage.

5.7. Restoring Archived Files

Once the Hyper Historian's logged data is archived, it is no longer available for playback on the trend display nor is it available through the SQL Query interface. However, it is easy to reattach archived data to the system again to make it viewable and accessible once again. You can do this by using Hyper Historian Archiving Management.

To access Archiving Management, expand the **System Administration** folder in the Project Explorer, and then double-click **Archiving Management**. The Archiving Management tab appears, as shown in Figure 5.14, and displays any files that have already been reattached. Archive files that have been restored to the online system are listed in the **Attached Archive Files** pane. If the **Generate Metadata for Archiving Files** checkbox was selected in the System Settings at the time the data was archived, metadata appears in the **Archive File Metadata Info** pane to the right.

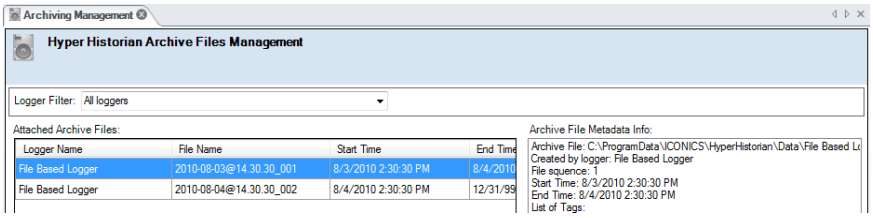


Figure 5.14 - Hyper Historian Archiving Management

It is from here that you can restore archived files to the online system by reattaching them. To select the data that you want to restore, click the **Attach New** button; this

opens a dialog box where you can browse to the folder containing the archive files to be reattached. Archived files have a file extension of **.hhd** and are named like this:

LoggerName_StartDate_StartTime_EndDate_EndTime

Start is when the file was opened to begin collection; *End* is the time at which the file was saved and closed. The date format is *yyyymmdd* (year, month, day) and the time format is *hhmmss* (hour, minute, second).

In the dialog box, select the files you want to reattach. You can select multiple files by holding down the **Shift** or **Control** key as you select files. Then click the **OK** button to restore the files.

Once a reattached file is no longer needed online, make sure you return to **Archiving Management** to detach it again. It will not be returned to the archives until you do so. To detach a reattached file, select the file in the **Attached Archive Files** pane, then click the **Detach Selected** button. If the file you are looking for is not in the list, try clicking the **Refresh Files** button first.

Lab: Archive Settings L5

This section will give you a chance to set up an archive for your data. We will also walk through the steps to restore some archive information for viewing purposes.

Estimated time to complete this lab: 20 minutes

5.1. Task Checklist

In this lab, you will complete the following tasks:

- Configure archive settings
- Restore an archived file
- View restored archive data

5.2. Support Files

You do not need any supporting files to complete this lab.

5.3. Archive Settings

1. Open the Workbench if you closed it from the previous lab and browse to the Hyper Historian configuration.
2. Expand the **System Administration** node, right-click on **System Settings**, and select **Edit On New Tab**. The Hyper Historian Advanced Configuration Settings opens in the right-hand pane.
3. Go to the **Memory Persistent Logger** tab. On the General sub-tab, put a checkmark in the **Enable Archiving** checkbox in the **Archiving Settings** section.
4. Click on the archiving folder's **Browse…** button. Browse to the GENESIS64 Training directory and select it, then click **OK**.
5. You can also check the **Generate Metadata for Archiving Files** option. Metadata can help you to understand what is in the archive file when you need to restore it. See Figure L5.1 for an example of the configuration.

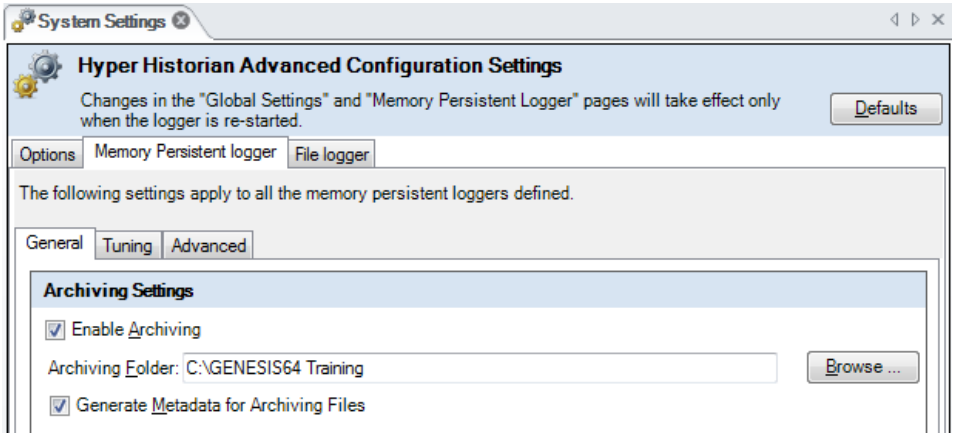


Figure L5.1 - Enable Archiving

6. Click **Apply** to save the configuration.

7. Now, we must restart the Hyper Historian service for this to take effect. Click on the Home ribbon's **Start/Stop** button (the traffic light) to stop it. Once the light turns red, you can click the button again to start the service. When it turns green again, it is running.

5.4. Restoring the Archive

In a previous lab, we set the Max File Time Extend option in the Disk Logger to one hour and the Min Storage Time option to two hours. This means that we will get our first archive file three hours later. Once the data is archived, we will not be able to view it; it will have to wait for data in the third hour before the file is archived. In this section, we will assume that you already have an archive file for restoration.

1. In the Hyper Historian configurator, expand **System Administration** and then right-click on **Archiving Management** and select **Edit on New Tab**.

2. The Archive Files Management form appears, as shown in Figure L5.2.

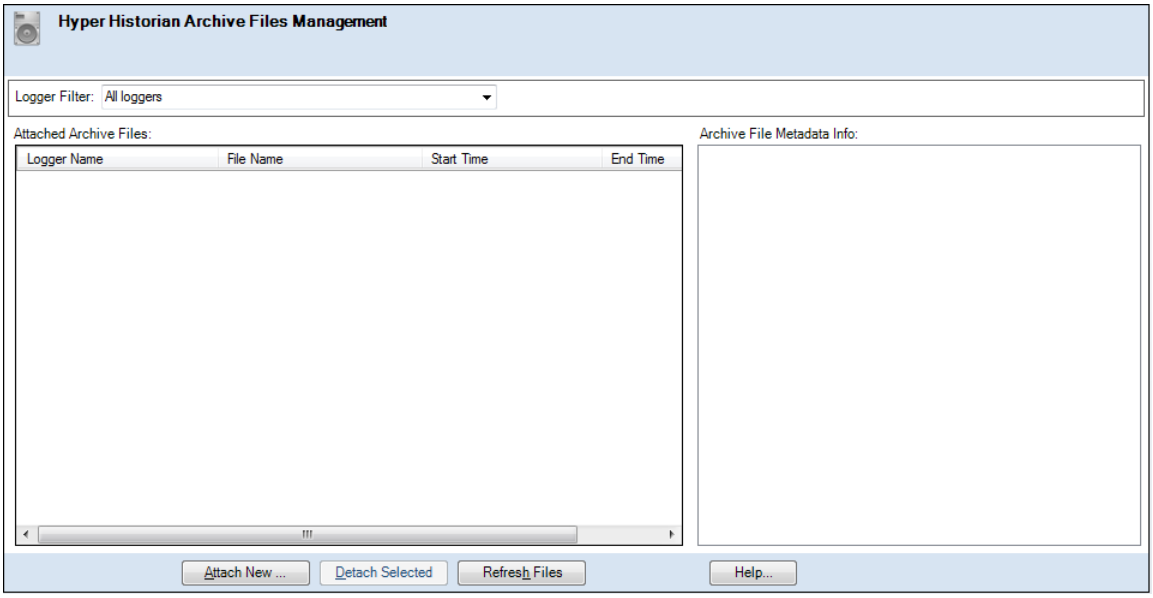


Figure L5.2 - Hyper Historian Archive Files Management

3. Click the **Attach New…** button, at the bottom of the form.

4. This opens the Attach Archive Files dialog box, pointing to the archiving folder specified in System Settings. Select the archive file that you wish to restore and click **OK**.

5. A confirmation box tells you the number of files attached.

6. Once your files have been attached, you can view the data in this archive. Remember that it will remain active until you detach it.

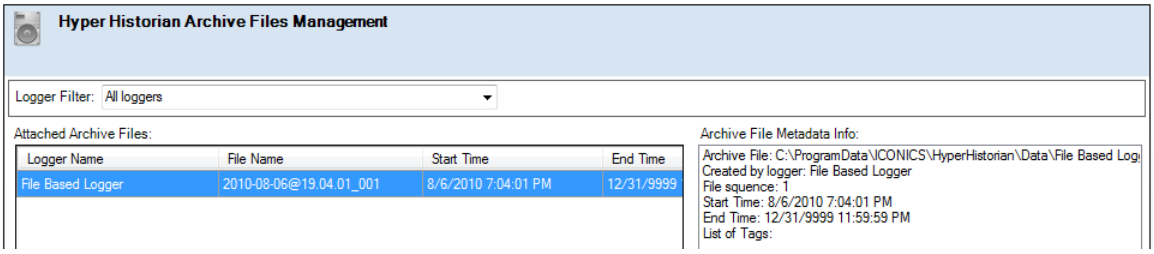


Figure L5.3 – Viewing an Attached File

7. You can now view the archived data in the TrendWorX64 Viewer.

Review: Additional Settings R5

5.1. Chapter Objectives

You should now know and understand the following topics:

- Workbench capabilities with Hyper Historian
- Choose a license mode
- Configure a collector node
- Configure the local logging node
- Configure automatic archiving
- Restore archived files

5.2. Summary

Hyper Historian is available in two editions: ***Standard*** and ***Enterprise***. The *Enterprise* edition is optimized for distributed data collection in which the data collectors reside on disparate, separate machines. The *Standard* edition, on the other hand, is optimized for a single PC or workstation where all of the Hyper Historian components are running on one machine. If you are using the Enterprise edition, you must choose your license mode.

You configure Hyper Historian using the Workbench. One of the first settings you will notice in the Project Explorer navigation tree is **Node Setup and Redundancy**. Here you primary and stand-by collectors. You can also set up Store and Forward options for collected data.

You can also set up these options for your logging server. But before using Hyper Historian for the first time, you must set the logging server storage path – the location where Hyper Historian stores the plant data that it collects – along with disk space management thresholds that warn you when disk space is getting low. Also, there are several options available to you for fine-tuning your loggers.

Older data stored by Hyper Historian can be automatically archived when it is no longer needed for online playback, reports, or charts. Hyper Historian detaches data from the system of ICONICS products and offloads it to a specified folder, freeing up space and providing another level of security and space management.

At any time, you can manually restore any archived file to the online system, making it available to ICONICS products once again.

5.3. Terms

Archive Files	Auto Fail-back	Collector Node
Enterprise License	Metadata	Primary Node
Secondary Node	Standard License	Store-and-Forward

5.4. Test Your Knowledge

In this chapter, we covered basic system setup for Hyper Historian, and looked at how archived files can be restored. Before you proceed, make sure you can answer these questions:

1. True or False? Hyper Historian can act as a distributed system of logger and remote collectors.

2. Which license version is optimized for local collection and logging?

    a. Lite
    b. Standard
    c. Enterprise

3. True or False? Hyper Historian supports automatic archiving.

4. True or False? You can restore archived files.

5. How many back-up nodes can you set up for a collector or logger?

    a. One
    b. Two
    c. Three
    d. As many as you want
    e. None

Hyper Historian SQL Query Engine A

Hyper Historian logs data to a proprietary database. While you can use the TrendWorX64 Viewer to see and edit your logged data in a graphical format, you may wish to create reports for the logged data or edit data in bulk. For this reason, Hyper Historian comes with a SQL Query Engine that allows you to retrieve and edit data.

You can use MS SQL tools such as the MS SQL 2005 Management Studio as well as the MS SQL 2005 Management Studio Express as the interface to query Hyper Historian data. In this chapter, we will discuss how you can retrieve data using MS SQL 2005 Management Studio.

A.1. Objectives

After completing this chapter, you should know:

- Interfaces for the Hyper Historian SQL Provider
- How to query for information with SQL 2005 Management Studio
- Hyper Historian SQL Provider query limitations

A.2. Running Queries against Hyper Historian Data

ICONICS Hyper Historian logs data to a proprietary database. While you can use the TrendWorX64 and TrendWorX32 Viewers to see and edit your logged data in a graphical format, you may wish to create reports for the logged data or edit data in bulk. For this reason, Hyper Historian comes with a SQL Provider that allows you to use common SQL Data Manipulation Language (DML) queries to retrieve and edit data.

A.2.1. Requirements

The following must be installed on the machine if you want to use the SQL Query Engine: VC Redistributable files for VS2008, ICONICS Hyper Historian, Microsoft SQL 2005 Management Studio, and Microsoft Excel (2003 or 2007).

A.2.2. Capabilities and Limitations of the SQL Query Engine

The Hyper Historian SQL Query Engine is developed, tested and validated against the Microsoft OLE DB RowsetViewer (version 2.70.7531.0).

Due to the provider's nature, a limited range of SQL features is supported. Catalogs, schemata and tables are predefined (by the Hyper Historian HDA Server). You can use only DML queries (SELECT, INSERT, UPDATE and DELETE). Also, the stored procedures are implemented in the Hyper Historian SQL Query Engine.

Individual query types have specific limitations to the query structure, as well. These limitations are detailed in an appropriate query section. In addition, advanced consumers (for example, Microsoft SQL Server) may perform query analysis or preprocessing and may change the original query. This may lead to unexpected results, which are described later in this appendix.

A.3. Architecture

The Hyper Historian SQL Query Engine is an in-proc COM server. It consists of several DLL libraries (modules). You can use the SQL Query Engine in 32-bit and 64-bit clients.

32-bit binaries (needed every time):

- **GenericOleDbProvider.dll** – The module that communicates with a SQL client; it processes data from the HHRawDataProvider.exe and provides the data in the form of tables and stored procedures to the client.
- **HHOleDbProvider.dll** – The main module; it performs registration of the OLE DB Provider.
- **HHRawDataProvider.exe** – The module that communicates with the Hyper Historian HDA Server. The module communicates indirectly through the IcoFwxUaClient.dll library (OPC UA communication).
- **RawDataProviderPS.dll** – The proxy/stub for the HHRawDataProvider.exe module.

64-bit binaries (needed for 64-bit clients):

- **GenericOleDbProvider64.dll** – The module that communicates with a SQL client; it processes data from the HHRawDataProvider.exe and provides the data in the form of tables and stored procedures to the client.

- **HHOleDbProvider64.dll** – The main module; it performs registration of the OLE DB Provider.
- **RawDataProviderPS64.dll** – The proxy/stub for the HHRawDataProvider.exe module.

A.4. Data Architecture

There are three tables that the Hyper Historian SQL Query Engine uses. They are the QUERYSETTINGS table, the TAGS table, and the RAWDATA table.

A.4.1. QUERYSETTINGS table

The QUERYSETTINGS table contains information which is used during the SQL query calculation. The table has the following structure:

Column Name	Data Type	Description
TIMESTAMP_START	DBTIMESTAMP	Start of the timestamp range (default value – can be overloaded in the query)
TIMESTAMP_END	DBTIMESTAMP	End of the timestamp range (default value – can be overloaded in the query)
USE_USER_TIME_RANGE	BOOL	If the flag is set to TRUE then the TIMESTAMP_START and TIMESTAMP_END timestamps are used for the queries. Otherwise for the queries are used default timestamps which are generated by the provider
MAX_ROWS_TO_RETURN	UI4	Number of rows which can be returned to the client
TREAT_UNCERTAIN_AS_GOOD	BOOL	For the HDA calculations
PERCENT_DATA_BAD	BYTE	For the HDA calculations
PERCENT_DATA_GOOD	BYTE	For the HDA calculations
ALLOW_EMPTY_TAG	BOOL	If this value is set to true you can run queries against the RawData table without the defining the tagname in WHERE clause. Otherwise these queries fail

A.4.2. TAGS table

The TAGS table contains information about tags and presents list of all tags which are available in the Hyper Historian HDA Server. The table has following structure:

Column Name	Data Type	Description
TAGNAME	WSTR | BYREF	Name of the tag
DESCRIPTION	WSTR | BYREF	Tag description
DATATYPE	UI2	Information about tag's data type
STEPPEDINTERPOLATION	BOOL	Stepped interpolation info
ACCESSRIGHTS	UI2	Information about access rights

A.4.3. RAWDATA table

The RAWDATA table contains information about all raw data which are in the Hyper Historian HDA Server. The table has following structure:

Column Name	Data Type	Description
TAGNAME	WSTR | BYREF	Name of the tag
TIMESTAMP	DBTIMESTAMP	Timestamp where the value was caught
QUALITY	UI4	Quality of the value
VALUE	VARIANT	The value itself

A.5. Supported SQL Constructs

You can run SQL queries that retrieve or edit data. This section provides details for and examples of the different SQL queries and functions that are supported for use with the Hyper Historian SQL Query Engine.

> **NOTE:** If you are unfamiliar with some SQL commands, refer to Microsoft SQL documentation to gain a better understanding. It's crucial that you have at least a beginner level of SQL knowledge before you continue; otherwise, you may find it difficult to understand terms or concepts covered in the pages that follow.

You can use the syntax that is provided below to query the Hyper Historian database, replacing all *italicized conditions* with actual values. What follows is syntax for queries (SELECT, INSERT, and UPDATE; note that the DELETE query is not supported) and functions.

A.5.1. SELECT Query

If you are familiar with SQL, you know that a SELECT statement can have the following syntax (variables appear in *italics*):

```
SELECT [lead count restriction] expression
FROM table | stored procedure
[WHERE condition];
```

The following rules apply to individual SELECT elements:

- *expression* – Defines column(s) to be retrieved by the SELECT query. These may include SQL aggregate column(s) as well, but it is not possible to combine ordinal and aggregate columns together in one SQL query. (GROUP BY is not supported).
- *table | stored procedure* – Defines table or stored procedure to select data from. It is not possible to combine table with stored procedure in one SQL query.
- **WHERE** *condition* – Defines a logical expression that must evaluate to true in order to return a table row. The expression allows only AND/OR chaining, use of brackets, and logical comparison operators (**<=**, **<**, **>**, **>=**, **<>**, **=**).
- *lead count restriction* – Can be used to restrict the number of retrieved rows. Use the form **TOP** *x* where *x* is the number of rows to return. Note that the count restriction is applied after the query is evaluated, thus, unfortunately, it won't affect performance in a positive way.

You can also use the *SQL functions* that begin on page A-7, and the *SQL stored procedures* that begin on page A-9.

You may also use the following standard SQL aggregates with the SELECT query:

- Min
- Max
- Avg
- Sum
- Count

Be careful when you are running SELECT queries against the RAWDATA table. If the ALLOW_EMPTY_TAG in the QUERYSETTINGS table is set to "true", you may see the following. You can run a query without the WHERE clause, but if you do so, the OLE DB Provider will get all data for all tags in the specified catalog (Logging group) and schema (Collector group). Then, the SQL Query Engine must sort data from all tags to one stream and this data will be sent to the output. This can take a very long

time, depending on the tag count. You can set ALLOW_EMPTY_TAG to "false". In this case, the queries without TAGNAME in the WHERE clause will fail (such queries are not permitted).

A.5.2. INSERT Query

You can insert rows into the RAWDATA table using the INSERT statement. The INSERT query is supported only for the RAWDATA table. This statement works like the standard SQL INSERT statement. The following INSERT syntax is supported:

```
INSERT INTO table (TAGNAME, TIMESTAMP, VALUE)
VALUES (tagname, timestamp, value);
```

When you want to insert a new value into the RAWDATA table, you must specify `TAGNAME`, `TIMESTAMP` and `VALUE`. (QUALITY is set automatically to "good".) When you are using the SQL Server, use OPENQUERY for the INSERT.

You can also use the *SQL functions* that begin on page A-7, and the *SQL stored procedures* that begin on page A-9.

Here's an example:

```
SELECT * FROM OPENQUERY(HH2, 'INSERT INTO LogGrp.OPCUA.RAWDATA
(TAGNAME, TIMESTAMP, VALUE) VALUES (''Sine'', ''2010-05-01
10:00:00'', 25;)');
```

A.5.3. UPDATE Query

You can update (that is, change) values logged into the database by using the UPDATE command. The UPDATE query can be used against only the QUERYSETTINGS and RAWDATA tables. The following UPDATE syntax is supported:

```
UPDATE table
SET column_name_1 = column_value_1 [, column_name_2 =
column_value_2, …]
[WHERE condition];
```

When you update the QUERYSETTINGS table, no `WHERE` clause is needed.

When you update the RAWDATA table, you must specify `TAGNAME` and `TIMESTAMP`. You can update only the `VALUE`. (QUALITY is automatically set to "good".) When you are using the SQL Server, use OPENQUERY for the UPDATE.

You can also use the *SQL functions* that begin below on page A-7, and the *SQL stored procedures* that begin on page A-9.

Here is an UPDATE example for the QUERYSETTINGS table:

```
SELECT * FROM OPENQUERY(HH2, 'UPDATE LogGrp.OPCUA.QUERYSETTINGS
SET MAX_ROWS_TO_RETURN = 500');
```

Here is an UPDATE example for the RAWDATA table:

```
SELECT * FROM OPENQUERY(HH2, 'UPDATE LogGrp.OPCUA.RAWDATA SET
VALUE = 25 WHERE TAGNAME = ''Sine'' AND TIMESTAMP = ''2010-05-01
10:00:00'';');
```

A.5.4. DELETE Query – Not Supported

At this time, the DELETE query is not supported in the Hyper Historian itself, so the Hyper Historian SQL Query Engine doesn't support it either.

A.5.5. SQL Functions

The following SQL functions are supported:

- The **QualityToStr** function can be used instead of the QUALITY column. The function converts quality from DWORD representation (number) to a string. For example:

  ```
  SELECT TAGNAME, QualityToStr(QUALITY) FROM RAWDATA …
  ```

  Output can be one of the following values:

  - Good
  - Bad
  - Uncertain

- The **DataTypeToStr** function can be used instead of the DATATYPE column. This function converts the data type from the WORD representation (number) to a string. For example:

  ```
  SELECT TAGNAME, DataTypeToStr(DATATYPE) FROM TAGS …
  ```

  Output can be one of the following values:

  - VT_BOOL

- VT_UI1
- VT_UI2
- VT_UI4
- VT_UI8
- VT_I1
- VT_I2
- VT_I4
- VT_I8
- VT_R4
- VT_R8

- The **AccessRightsToStr** function can be used instead of the ACCESSRIGHTS column. This function converts access rights from WORD representation (number) to a string. For example:

  ```
  SELECT TAGNAME, AccessRightsToStr(ACCESSRIGHTS) FROM TAGS …
  ```

  Output can be one of the following values:

  - TAG_ACCESS_RIGHTS_NONE
  - TAG_ACCESS_RIGHTS_READ
  - TAG_ACCESS_RIGHTS_WRITE
  - TAG_ACCESS_RIGHTS_READ_WRITE

- The **IsGood**, **IsBad** and **IsUncertain** functions can be used in a WHERE clause. The **IsGood()** function selects only raw data where the quality is Good. The **IsBad()** function selects only raw data where the quality is Bad. And the **IsUncertain()** function selects only raw data where the quality is Uncertain. Here's an example of syntax:

  ```
  SELECT * FROM RAWDATA WHERE IsBad(QUALITY) = 1;
  ```

- **Date / Time Functions.** In the WHERE clause, you can use date/time functions. You can use the **dateadd()** function and you can use also *now* and *nowutc* variables. Here's an example of a query that returns all raw data that is no more than two hours old:

  ```
  SELECT * FROM RAWDATA WHERE TIMESTAMP >= dateadd(hh, -2, nowutc)
  and TIMESTAMP <= nowutc;
  ```

  The function's syntax is:

```
dateadd(DATE_PART, VALUE, BASE_TIMESTAMP)
```

where the parameters can have the following values:

DATE_PART

- `ms` – milliseconds
- `ss` – seconds
- `mi` – minutes
- `hh` – hours
- `dd` – days
- `wk` – weeks
- `mm` – months
- `yy` – year

VALUE – can be any integer.

BASE_TIMESTAMP

- `now` – local time
- `nowutc` – UTC time
- or you can type your own timestamp in following format: *'yyyy-mm-dd hh:mi:ss'*

A.5.6. SQL Stored Procedures

All historical-data aggregates calculations are provided through the stored procedures. (A list of historical aggregates is provided on page A-A-10.) In addition, more complex operations above raw data are also provided through the stored procedures as well.

The stored procedures that are supported at this time are listed below. Input parameters are provided following the list.

- **HDA_BOOL** calculates Duration in State 0, Duration in State 1, Number of Transitions and Count for one tag.
- **HDA_BOOL_5** works just like HDA_BOOL but for up to five tags.
- **HDA_ANALOG** calculates Min, Max, Average, Time Average, Totalize Average, Interpolative, Last, Delta, Range, Total and Count for one tag.
- **HDA_ANALOG_5** work just like HDA_ANALOG but for up to five tags.
- **HDA_REDUCED_ANALOG** calculates Min, Max, Time Average and Count for one tag.
- **HDA_REDUCED_ANALOG_5** works just like HDA_REDUCED_ANALOG but for up to five tags.

- **SP_RAW_DATA** gets raw data for one tag. Input parameters are:
    - *Start timestamp* – start of the timestamp range
    - *End timestamp* – end of the timestamp range
    - *Tag name* – when you specify only tag name (without catalog and schema) then the catalog and schema for the tag is taken from the stored procedure specification

Input parameters for **HDA_ANALOG**, **HDA_REDUCED_ANALOG**, and **HDA_BOOL** are:

- *Start timestamp* – start of the timestamp range
- *End timestamp* – end of the timestamp range
- *Resampling period* – time interval size
- *Tag name* – when you specify only tag name (without catalog and schema) then the catalog and schema for the tag is taken from the stored procedure specification

Input parameters for **HDA_BOOL_5**, **HDA_REDUCED_ANALOG_5**, and **HDA_ANALOG_5** are:

- *Start timestamp* – start of the timestamp range
- *End timestamp* – end of the timestamp range
- *Resampling period* – time interval size
- *Tag name 1*
- *Tag name 2*
- *Tag name 3*
- *Tag name 4*
- *Tag name 5*

Note: When you specify only the tag name (without a catalog and schema), the catalog and schema for the tag is taken from the stored procedure specification.

A.5.6.1. Historical Aggregates

Historical aggregates and their behaviors are listed here:

- **Min** – Retrieve the minimum value in the re-sample interval
- **Max** – Retrieve the maximum value in the re-sample interval
- **Average** – Retrieve the average data over the re-sample interval
- **Time average** – Retrieve the time weighted average data over the re-sample interval

- **Totalize average** – Retrieve the totalized value (time integral) of the data over the re-sample interval
- **Interpolative** –
- **Last** – Retrieve the value at the end of the re-sample interval. The timestamp is the timestamp
- of the end of the interval
- **Duration in state 0** – Retrieve the duration of time in the interval during which the data is bad
- **Duration in state 1** – Retrieve the duration of time in the interval during which the data is good
- **Number of transitions** – Retrieve the number of state changes a Boolean value experienced in the interval
- **Delta** – Retrieve the difference between the first and last value in the re-sample interval
- **Range** – Retrieve the difference between the minimum and maximum value over the re-sample
- interval
- **Total** – Retrieve the sum of the data over the re-sample interval
- **Count** – Retrieve the number of raw values over the re-sample interval

A.5.6.2. Status Code Calculation for Historical Aggregates

For aggregate values, the status code for each returned aggregate is "good" if the status code for all values used in the aggregate was good.

If the status code of any value used in computing the aggregate was not good, then the server uses the TreatUncertainAsBad, PercentDataBad and PercentDataGood parameter settings to determine the status code of the resulting aggregate for the interval. Some aggregates may explicitly define their own method of determining quality.

If the percentage of good values in an interval is greater than or equal to the PercentDataGood, the aggregate is considered "good", otherwise it is "bad".

Because a value can be either good or bad only (uncertain is defined as good or bad as per "treat uncertain as bad" setting), percentage good = 100 – percentage bad. If a percentage good(X) is in the following range percentage bad < X < percentage good, the quality of the aggregate is Uncertain_DataSubNormal.

A.6. Specifying a Connection

The Hyper Historian SQL Query Engine overrides some default pages in the Data Link Properties dialog.

The first page is a default Provider tab. Select **ICONICS Hyper Historian SQL Query Engine** and click the **Next** button. The Connection tab appears.

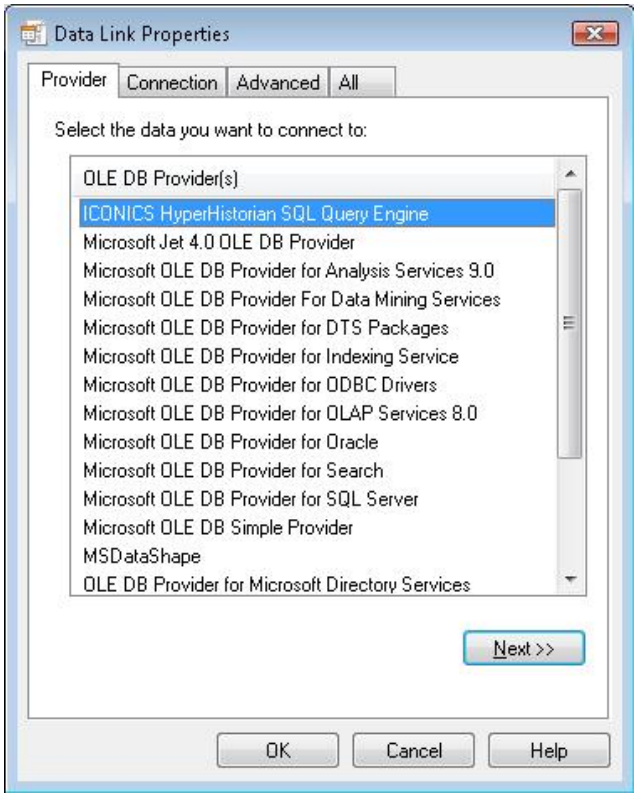


Figure A.1 – Provider Tab

The Connection tab (shown below) allows you to set the data source, initial catalog, and schema.

1. Specify the location of the Hyper Historian HDA Server. If this value is not specified, the localhost is used.

2. Specify an initial catalog (Logging group). This catalog is used as the default catalog when the SQL queries are executed. If the initial catalog is not specified, then the default catalog is the first catalog from the list. This list is gained from the Hyper Historian HDA Server.

3. You can also specify the initial schema (Collector group). The rules here are the same as for the initial catalog. The specified schema is used as the default schema when SQL queries are executed and so on.

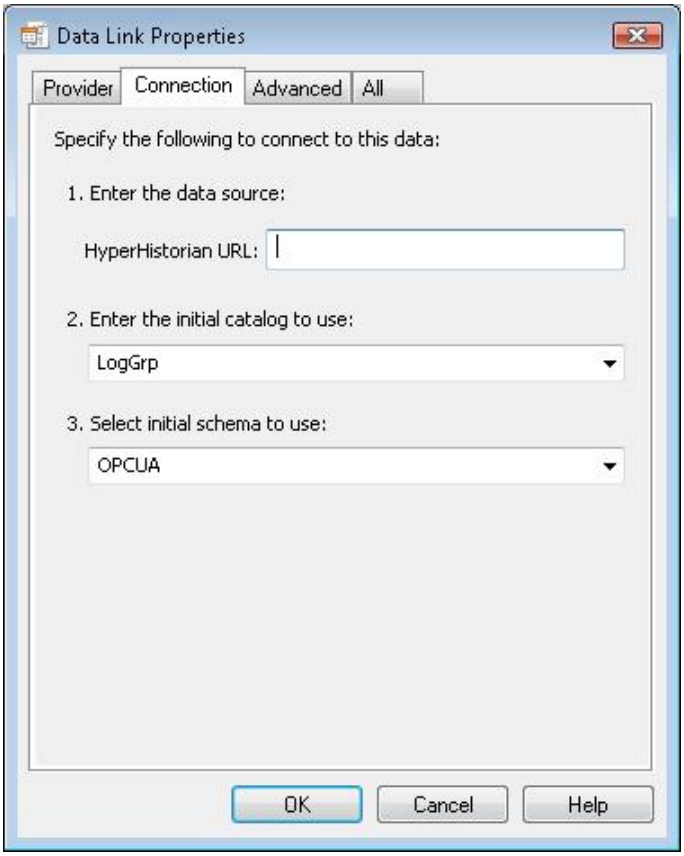


Figure A.2 – Connection Tab

The third tab is the overridden Advanced tab, shown below. On this tab, you can specify default starting and ending timestamps. If the timestamps are specified, then the timestamps are used as default timestamps when the SQL queries are executed. These settings are applied when you run SQL queries against the RAWDATA table. When you specify timestamps in the SQL query in the WHERE clause, you will override the default timestamps.

When you don't specify timestamps, then the default timestamps are generated every time when the query is started against the Hyper Historian SQL Query Engine. In this case, default timestamps are set subsequently. The ending timestamp is set to the current time. The starting timestamp is set to the current time minus three days (for example: 2009-12-01 10:00:00, 2009-12-04 10:00:00).

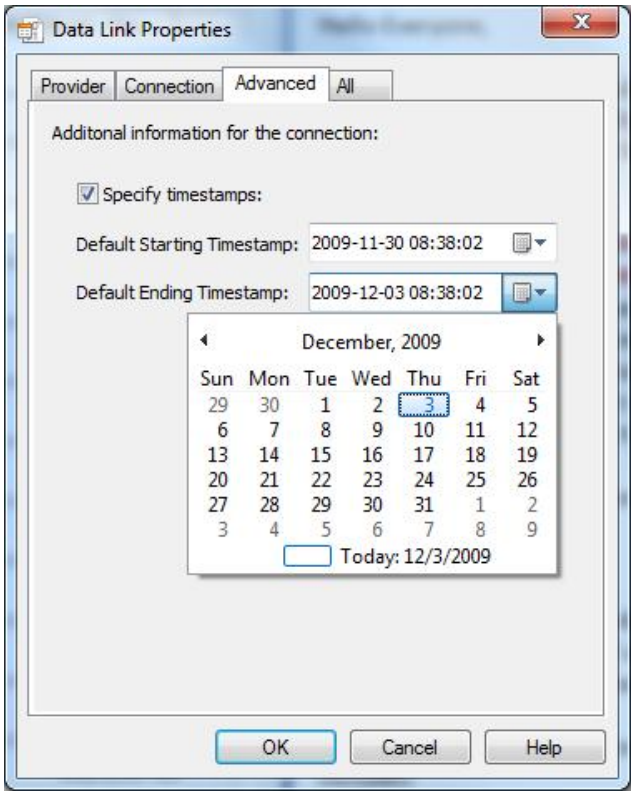


Figure A.3 – Advanced Tab

A.7. Microsoft SQL Server Specifics

Microsoft SQL Server performs specific modifications to the queries passed to the linked server (e.g., aliasing). It also performs many auxiliary operations such as data type conversions and local functions.

Microsoft SQL Server preprocessing puts limitations on the query parameters. Namely, the Microsoft SQL Server won't propagate time specified as absolute date, e.g. '2009-01-01 12:00:00', to the provider. Instead, you must:

- Use date manipulation functions, such as *dateadd()* and *getutcdate()*, or
- "Hack" the conversion using a nested SELECT statement, such as `select convert(datetime, '2009-01-01 12:00:00')`, or
- Use the OPENQUERY statement; for example, `SELECT * FROM OPENQUERY(HH2, '<SQL query>')`. This solution is recommended to use in the SQL Server. Otherwise, you can fall into trouble.

A.7.1. MS SQL Server Query Execution Problems

When Microsoft SQL Server decides not to send the entire query to the provider, the query can be still executed and can succeed. But rather than being processed by the provider it will be processed at Microsoft SQL Server.

The problems with this approach are:

- **Performance** – If a query such as `SELECT * FROM HH2.<catalog>.<schema>.RAWDATA WHERE TIMESTAMP >= '2009-01-01 12:00:00' AND TAGNAME = 'Sine';` is used, Micorosft SQL Server will execute `SELECT * FROM HH2.<catalog>.<schema>.RAWDATA;` against the provider. The provider won't get information about the tag for which you can read the data. In this case, the provider will get all data for all tags in the specified catalog and schema (this may take a very long time) and after this, the Microsoft SQL Server will do the final filtering. If you want to prevent this, set the ALLOW_EMPTY_TAG in the QUERYSETTINGS to "false". Then queries like this will not be started.
- **Runtime limitations** – Certain queries are prohibited, such as queries without certain limit parameters. Microsoft SQL Server might inadvertently use these queries when executing a subset of a user-supplied query, which would lead to query execution failure.

A.8. Troubleshooting

Make sure the Hyper Historian SQL Query Engine is correctly installed in the SQL Server:

1. Start Microsoft SQL Server Management Studio.
2. Go to **Server Objects** > **Linked Servers**.
3. You should see linked server named **HH2**.
4. Open window for new query.
5. Run following query: `EXEC sp_catalogs HH2;`
6. If the provider is installed, you should get a list of all catalogs (Hyper Historian logging groups).

If you cannot get any data in the SQL Server:

1. Run the **dcomcnfg** application.
2. Select **Component Services** > **Computers** > **My Computer** > **DCOM Config**.
3. In the application list, find the **ICONICS Hyper Historian Raw Data Provider**.
4. Open its properties and go to the **Security** tab.
5. Check if there is a user under which the SQL Server is running (usually NETWORK SERVICE). This user must exist in the Launch and Activation Permissions. Also, this user must exist in the Access Permissions.
6. If the user exists in both cases, check if there is "all allowed" for this user.
7. Confirm all changes.
8. Open Windows Task Manager and go to the **Processes** tab.
9. Find the **HHRawDataProvider.exe** process and end the process (or you can restart the computer; this would be better).
10. If you still can't get any data, check if the Hyper Historian HDA Server is running properly.

A.9. Examples

This section provides sample SQL states for specific purposes.

A.9.1. TAGS table

To select all tags:

```
SELECT * FROM LogGrp.OPCUA.TAGS;
```

To select tags with a specified name:

```
SELECT * FROM LogGrp.OPCUA.TAGS WHERE TAGNAME = 'Sine';
```

To select all tags (only some columns):

```
SELECT TAGNAME, DATATYPE FROM LogGrp.OPCUA.TAGS;
```

A.9.2. QUERYSETTINGS tables

To select from the query settings table:

```
SELECT * FROM LogGrp.OPCUA.QUERYSETTINGS;
```

To update information in the query settings table:

```
UPDATE LogGrp.OPCUA.QUERYSETTINGS SET MAX_ROWS_TO_RETURN = 1000,
ALLOW_EMPTY_TAG = 0;
```

A.9.3. RAWDATA table

To select raw data (all tags, with default TIMESTAMPs):

```
SELECT * FROM LogGrp.OPCUA.RAWDATA;
```

To select raw data with specified TAGNAME and TIMESTAMPs:

```
SELECT * FROM LogGrp.OPCUA.RAWDATA WHERE TAGNAME = 'Sine' AND
TIMESTAMP >= '2009-12-01 10:00:00' AND TIMESTAMP <= '2009-12-02
10:00:00;
```

To select raw data using DateAdd() function:

```
SELECT * FROM LogGrp.OPCUA.RAWDATA WHERE TAGNAME = 'Sine' AND
TIMESTAMP >= DateAdd(hh, -5, nowutc()) AND TIMESTAMP <= nowutc();
```

To select raw data using QualityToStr() function:

```
SELECT TAGNAME, TIMESTAMP, QualityToStr(QUALITY), VALUE FROM
LogGrp.OPCUA.RAWDATA WHERE TAGNAME = 'Sine' AND TIMESTAMP >=
DateAdd(hh, -5, nowutc()) AND TIMESTAMP <= nowutc();
```

To select raw data using IsGood() function:

```
SELECT * FROM LogGrp.OPCUA.RAWDATA WHERE TAGNAME = 'Sine' AND
TIMESTAMP >= DateAdd(hh, -5, nowutc()) AND TIMESTAMP <= nowutc()
AND IsGood(QUALITY) = 1;
```

A.9.4. Stored procedures

To run the HDA_BOOL stored procedure:

```
EXEC LogGrp.OPCUA.HDA_BOOL '2009-12-01 12:00:00', '2009-12-02
12:00:00', 300000, 'Sine';
```

To run the HDA_ANALOG stored procedure in SELECT query:

```
SELECT * FROM LogGrp.OPCUA.HDA_ANALOG('2009-12-01 12:00:00',
'2009-12-02 12:00:00', 300000, 'Sine');
```

To select the MAX, MIN and AVG values:

```
SELECT MIN(VALUE) AS min_value, MAX(VALUE) AS max_value,
AVG(VALUE) AS avg_value FROM LogGrp.OPCUA.RAWDATA WHERE TAGNAME =
'Sine' AND TIMESTAMP >= '2009-12-01 12:00:00' AND TIMESTAMP <=
'2009-12-02 12:00:00';
```

Lab: Hyper Historian SQL Query Engine

LA

In this lab, you will build a simple query in Microsoft SQL Server Management Studio Express, which is part of Microsoft SQL 2005 suite, to retrieve Hyper Historian data over its SQL interface.

Estimated time to complete this lab: 20 minutes

A.1. Task Checklist

In this lab, you will complete the following task:

- Build SQL queries to retrieve Hyper Historian data

A.2. Support Files

You do not need any supporting files to complete this lab. However, you must have SQL Server Management Studio Express or SQL Server Management Studio.

A.3. Hyper Historian Linked Server

First, open SQL Server Management Studio to see the providers, which are automatically installed and configured together with Hyper Historian.

1. Open SQL Server Management Studio Express by going to **Start → Programs → Microsoft SQL Server 2005 → SQL Server Management Studio Express**.
2. When prompted, fill in credentials to connect to your local SQL server.
3. In the Object Explorer on the left, expand **Server Objects**.
4. Double-click on **Linked Servers** to expand it. You should now see a linked server called **HH2** (HH stands for Hyper Historian).
5. Expand the **Providers** folder. You will see a provider called **ICONICS.HHOleDbProvider** which is necessary for your queries to work.

A.4. Building a SQL Query

In this exercise, you will use a simple SQL command that reads the list of all tags available in one Hyper Historian Logging Group. On top of that, you will create another simple query to retrieve logged data related to the selected tag.

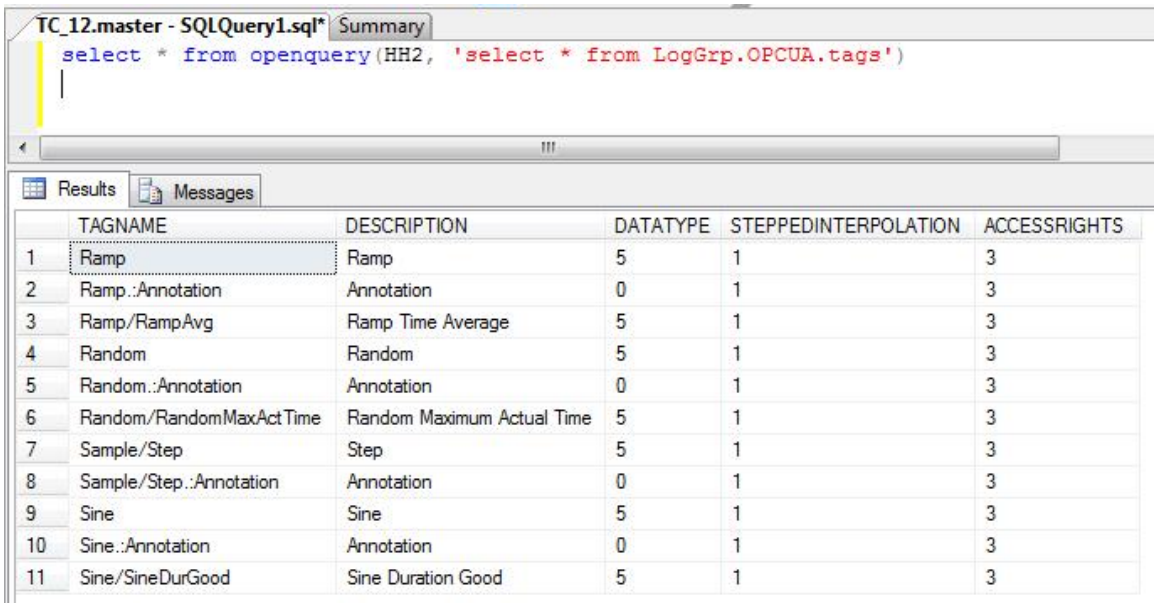
You'll use the default samples Logging Group:

1. Click on the **New Query** button, located in the standard toolbar of Microsoft SQL Server Management Studio Express.

2. In the SQL Editor that opens on the right, write the following query:

```
select * from openquery(HH2, 'select * from
LogGrp.OPCUA.tags')
```

3. Click the **Execute** button in the SQL Editor Toolbar. The result will look something like this:



	TAGNAME	DESCRIPTION	DATATYPE	STEPPEDINTERPOLATION	ACCESSRIGHTS
1	Ramp	Ramp	5	1	3
2	Ramp.:Annotation	Annotation	0	1	3
3	Ramp/RampAvg	Ramp Time Average	5	1	3
4	Random	Random	5	1	3
5	Random.:Annotation	Annotation	0	1	3
6	Random/RandomMaxActTime	Random Maximum Actual Time	5	1	3
7	Sample/Step	Step	5	1	3
8	Sample/Step.:Annotation	Annotation	0	1	3
9	Sine	Sine	5	1	3
10	Sine.:Annotation	Annotation	0	1	3
11	Sine/SineDurGood	Sine Duration Good	5	1	3

Figure LA.1 – Getting Tag Names

NOTE: If you get no results for the query, you probably have no logged data. Open Hyper Historian (with the default configuration) and start the service. Then repeat this step.

4. Now select one of the tag names to query. On the next line of the query editor add the following query, which will list all values for the specified tag:

```
select * from openquery(HH2, 'select * from
LogGrp.OPCUA.rawdata where tagname = ''Ramp''')
```

The result will look something like this:

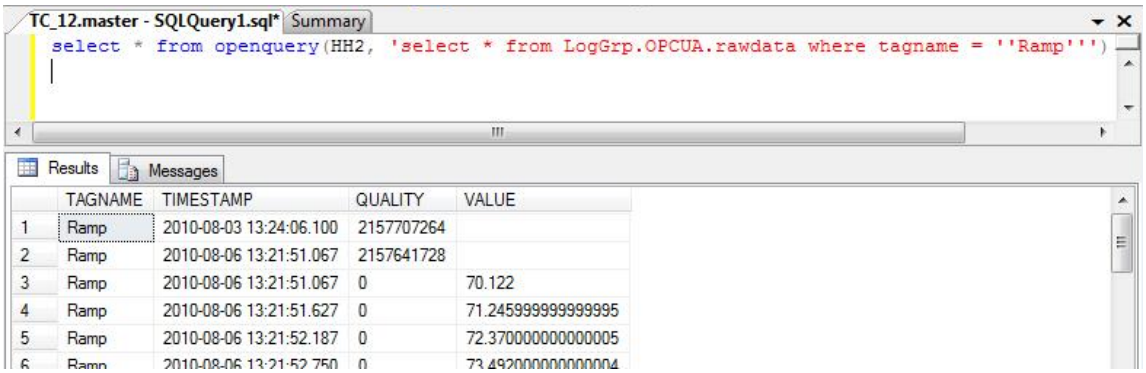


Figure LA.2 – Getting Logged Data

A.5. SQL Clients – MS Excel

The Hyper Historian SQL Query Engine can be used in other containers. Let us take a look at how to use it in Microsoft Excel.

1. Open MS Excel by going to **Start → Programs → Microsoft Office → Microsoft Office Excel 2007**.
2. In the **Data** tab, click the **From Other Sources** button.
3. Select the **From Data Connection Wizard** option.
4. Choose the **Other/Advanced** data source and click **Next**.
5. Select **ICONICS Hyper Historian SQL Query Engine** and click **Next**.
6. On the **Connection** properties tab, select the **Catalog** and **Schema** names, which correspond to the Hyper Historian logging group names. Click **OK**.
7. In the dialog box that appears, choose the database that contains the data. In this case, it's **LogGrp**. A list of tables appears as shown in Figure LA.3.

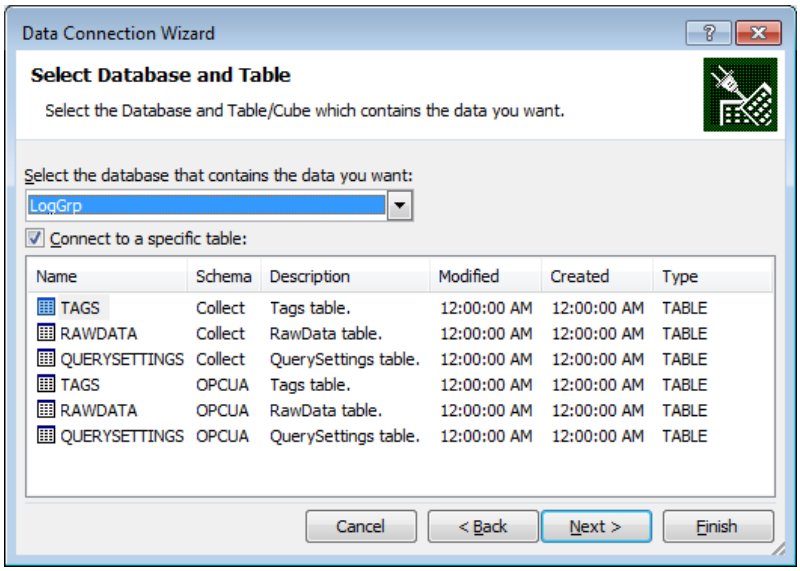


Figure LA.3 – Selecting a Table

8. Now select one of the tables: TAGS displays all tags in the Hyper Historian configuration, RAWDATA displays logged data for the selected logging group, and QUERYSETTINGS displays settings for the SQL query. Select one and click Next. The Data Connection Wizard window appears.

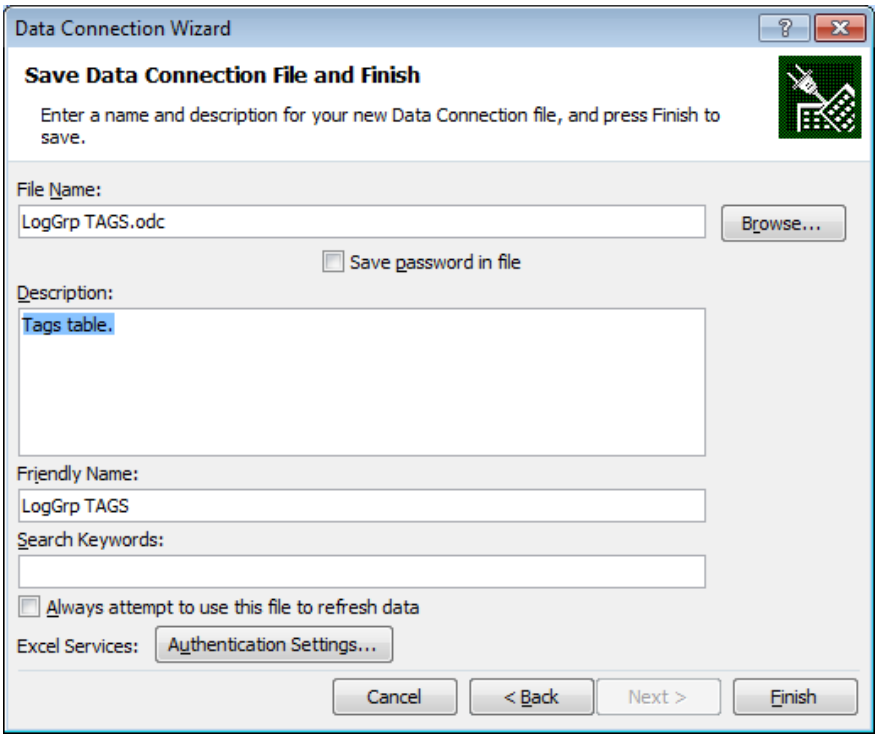


Figure LA.4 – Data Connection Wizard

9. Configure the name and description for your data connection file. Click **Finish**. The Import Data properties dialog appears; it looks like this:

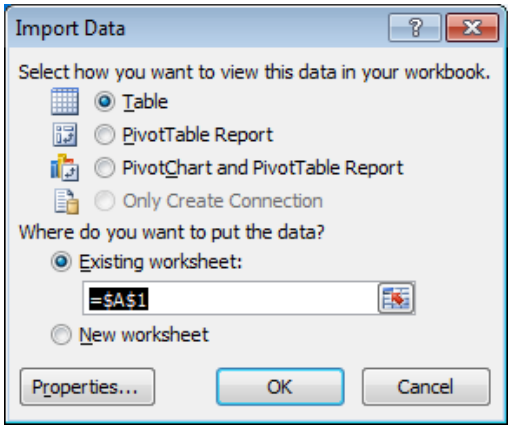


Figure LA.5 – Import Data Dialog Box

10. You can change properties to your liking or leave them as they are. Click **OK** when you are ready to continue.

11. The result of your query will look something like this:

TAGNAME	DESCRIPTION	DATATYPE	STEPPEDINTERPOLATION	ACCESSRIGHTS
Conditional	Conditional signal	11	TRUE	3
Filtering/DeadbandMax	Deadband Aggressive	5	TRUE	3
Filtering/DeadbandMin	Deadband Minimal	5	TRUE	3
Filtering/NoFilter	No Filter	5	TRUE	3
Filtering/SD_Max	Swinging Door aggressive	5	TRUE	3
Filtering/SD_Min	Swinging Door minimal	5	TRUE	3

Figure LA.6 – MS Excel Display of Logged Data

NOTE: You can use Hyper Historian SQL Query Engine in the BizViz ReportWorX application, too, for creating scheduled reports.

Glossary

B

Aggregate is a pointer from a Hyper Historian tag to an *aggregate group* along with the specification of how the tag's data is to be aggregated and the timeframe for aggregating it.

Aggregate Group is a Hyper Historian container for aggregates in a logging group's Aggregation folder. The aggregate group defines the timeframe in which aggregate information is accumulated.

Archived Data is data that has been offloaded to archivable files. Data in these files is no longer available to your ICONICS system for charts and reports, but it can be restored so that it is available whenever you want it.

Archive File is a metadata file that contains detailed information about the archived file, including a list of the tags whose data is stored in the archive file

Auto Fail-back is an option in which, after a failover has occurred, the primary node automatically takes over as soon as it becomes available again.

BizViz Suite is the suite of software from ICONICS made for Business Visualization. The products in this includes: BridgeWorX, ReportWorX, PortalWorX, Mobile HMI, Alarm Analytics, and Productivity Analytics. For more information on this product, please visit www.iconics.com/products/bizviz_suite.asp

Calculation Period is the time frame that Hyper Historian uses to calculate some preset function before logging the data. The amount of values used in this calculation depends on your Data Collection Rate.

Collector Group is A Hyper Historian configuration that defines the collection rate and period of calculation for a group of OPC tags.

Collector Node specifies the location of the data collector.

Data Collection Rate is the rate at which samples are collected. The fastest you can collect samples at is 10 milliseconds.

Deadband is the amount or percentage that a subsequent collected value must differ from the prior collected value before it is considered a good sample.

Disk Logger is a Hyper Historian logger that collects plant data that it saves to a hard disk. This type of logging is also known as a File-Based Logging. Compare to Memory-Persistent logger.

Enterprise License is one of the Hyper Historian License modes. This license lets you run in Standard mode or in Enterprise mode which allows you to run your data collector and data logger on different machines.

Hyper Historian is a powerful, 64-bit, high performance plant historian designed for mission-critical applications that require uninterrupted access and collection of data. It accesses more CPU power and memory than traditional 32-bit-based historians and provides the highest performance possible.

Hyper Historian SQL Interface is the bridge that allows you to access the data logged in our proprietary Hyper Historian database. You can use SQL 2005 Management Studio to run some standard SQL queries to obtain data.

Hyper Historian Tag is a configuration that identifies an OPC tag or expression and the type of data that is collected from it for the Hyper Historian

Logging Group is an object with a logger configuration for collecting Hyper Historian data for a set of OPC tags

Log to Disk Filters are pre-defined functions that Hyper Historian uses to calculate a value for a set period of tem (Calculation Period). The functions include: min, max, avg, etc.

Memory-Persistent Logger is a Hyper Historian logger that collects plant data that it stores in RAM and discards after a pre-configured amount of time. It can be configured to store a portion of its collected data to the hard disk when certain events occur. Compare to Disk logger.

Metadata is detailed information about an archived file, including a list of the tags whose data is stored in the archive file, which is very helpful if you need to restore data later on.

OPC DA stands for OPC Data Access, which is a standard to get real time data

OPC HDA stands for OPC Historical Data Access, which is a standard to log and retrieve historical data.

OPC UA stands for OPC Unified Architecture, which is a new standard from the OPC foundation for accessing data.

Primary Node is the primary collector, or the node that is typically used as the collector. If this node becomes unavailable for any reason, if redundancy has been set up in Hyper Historian, the *secondary node* takes over as the collector.

Secondary Node is the stand-by collector that will take over if the primary node (and thus, the primary collector) becomes unavailable.

Standard License is one of the Hyper Historian License modes. This is ideal if you want to run the logger and the collector on the same machine.

Store-and-Forward is an optional redundancy policy for caching data if the connection to a collector is lost. When this option is in use, after the connection to the collector is restored, the cache is flushed and the databases on the *primary* and *secondary nodes* are synchronized.

Swinging Door Compression is an algorithm used by Hyper Historian to determine which values must be logged to the database for a smooth curve. Compared to how TrendWorX64 logs data, this logs much less data, but produces a much smoother curve.

TrendWorX64 Viewer is the application that you can use to view real-time data, TrendWorX64 Logged data, and Hyper Historian data. You can view data in standard time plot, x-y plot, bar plot, etc.

Workbench is the multi-functional, centralized, web-based environment for configuring Hyper Historian.

Answers

C

Questions / Chapters	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Chapter 1	True	B	True	True	False	True	G			
Chapter2	A and C	F	True	True	E	False				
Chapter3	C	E	D	True	False	True	True	False	D	B
Chapter4	False	True	True	B	False	True	False	True		
Chapter5	True	B	True	True	A					

Hyper Historian™

Visualize Your Enterprise™

100 Foxborough Blvd., Foxborough, MA 02035
508.543.8600 • FAX: 508.543.1503
www.iconics.com • E-mail: info@iconics.com

© 2012 ICONICS, Inc. All Rights Reserved.
Specifications subject to change without notice.