AVEVA™ Historian Concepts Guide
formerly Wonderware

Publication date: Monday, August 30, 2021

Contact Information

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

https://sw.aveva.com/

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

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

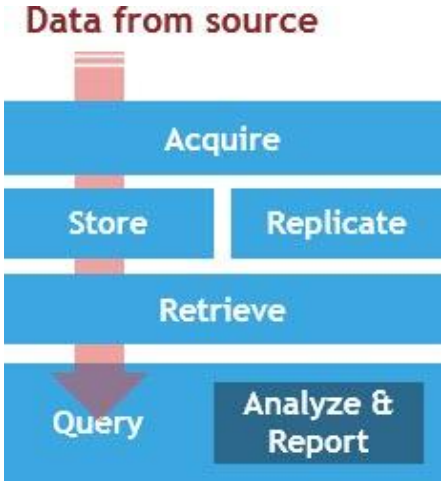
Contents

Welcome to AVEVA Historian

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

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

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



With AVEVA Historian, you can:

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

  AVEVA Historian Historian optimally acquires and stores analog, discrete, and string data. In addition, the SuiteLink protocol used by AVEVA I/O Servers and DAServers provides time and quality stamps as data is added to the system.

- **Store lots of data efficiently**
  The Storage subsystem compresses the data for maximum storage efficiency. In addition to your plant's data, AVEVA Historian stores alarms and events, summary data, configurations, security information, backups, and system monitoring data.

- **Retrieve data when you want it**
  The Retrieval subsystem responds to SQL requests for plant data and allows you to retrieve large amounts of data very quickly. It also supports REST and SDK data retrieval.

- **Replicate your stored data for security**
  The Replication subsystem replicates tags' values to other AVEVA Historian servers with high fidelity or calculates and replicates summaries of those values.

- **Query data using time domain extensions to SQL**
  AVEVA Historian builds on the capabilities of the SQL query language by supporting time-series data and controlling the resolution of returned data in SQL (for example, to give an evenly spaced sampling of data over a period of time).
- **Create, save, and share graphical content**
  AVEVA Historian includes a tool called Insight that lets you quickly search for data and generate clean, clear graphical content that you can save and share.

AVEVA Historian Documentation Set

The AVEVA Historian documentation set includes the following guides:

- *AVEVA System Platform Installation Guide*
  This guide provides information on installing the AVEVA Historian, including hardware and software requirements and migration instructions.
- *AVEVA Historian Concepts Guide*
  This guide provides an overview of the entire AVEVA Historian system and its key components.
- *AVEVA Historian Scenarios Guide*
  This guide discusses how to use AVEVA Historian to address some common customer scenarios.
- *AVEVA Historian Administration Guide*
  This guide describes how to administer and maintain an installed AVEVA Historian, such as configuring data acquisition and storage, managing security, and monitoring the system.
- *AVEVA Historian Retrieval Guide*
  This guide describes the retrieval modes and options that you can use to retrieve your data.
- *AVEVA Historian Database Reference*
  This guide provides documentation for all of the AVEVA Historian database entities, such as tables, views, and stored procedures.
- *AVEVA Historian Glossary*
  This guide provides definitions for terms used throughout the documentation set.

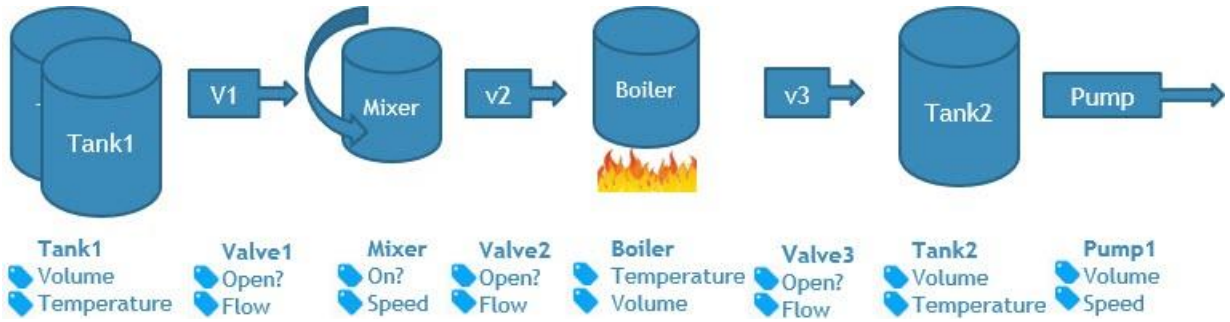
In addition, the *AVEVA License Manager Guide* describes the AVEVA License Manager and how to use it to install, maintain, and delete licenses and license servers on local and remote computers.

Process data: About tags and values

AVEVA Historian acquires and stores process data, which is any information related to successfully running a process. That data is stored as *tags*.

The term "tag" originally referred to physical label on a mechanical part or device on the plant floor. Each tag identifies the corresponding device to AVEVA Historian. As each device sends values to the historian, they are recorded by tag.

For example, a boiler might have two tags – one for the temperature gauge and one for the volume meter.



Using process data to answer your business questions

Historian stores and retrieves your process data in a way that allows you bring clarity to issues related to your business.

- **Real-time data**
  Historian stores data as it comes in from the plant floor.

  Real-time data answers questions like "What's the temperature of that tank right now?"

- **Historical data**
  Historian can accept historical information from other systems and retains data captured in real time to use for historical reference.

  Historical data helps to answer questions like "What was the value of this tag every second last Monday?"

- **Summary data**
  Historian summarizes certain data to help answer with big-picture questions like "What is the average number of bags produced each week?"

- **Event data**
  Historian records process alarms and events as they happen. This data is used to find answers about when things happened; for example, "How often did that boiler trip last month?"

- **Configuration data**
  This data describes your system's configuration and answers questions like "What types of I/O Servers am I using?"

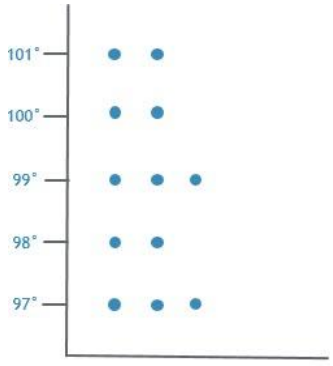
Process data analysis can help improve performance, enhance quality, and reduce costs.

For more information about defining and using tags, see Defining Tags in the AVEVA Historian Administration Guide.

Three-dimensional data: Value, time, and quality (VTQ)

Each time Historian records a data value, it also records a corresponding timestamp and data quality rating. Together, these three things – value, time, and quality – are called a "VTQ".

Values alone: Only somewhat helpful



For example, suppose there is a tag named "tank1.temp" that measures the temperature of a tank in the plant.

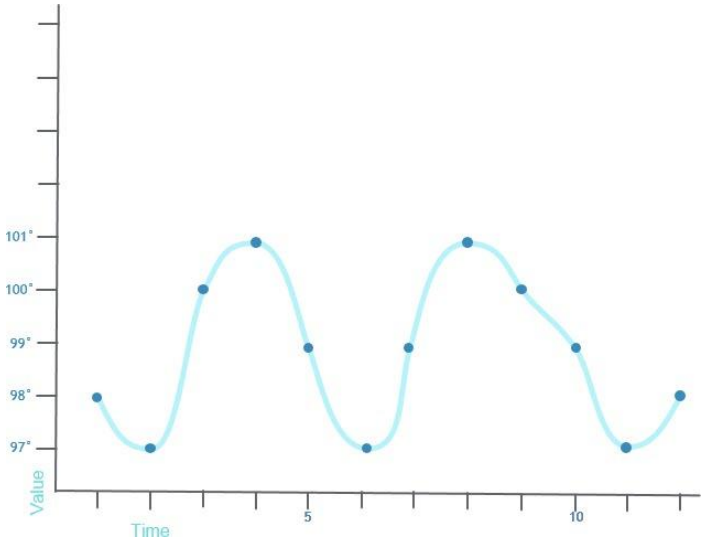
The tank's sensor would send periodic temperatures to Historian – 97 degrees, 98 degrees, 102 degrees, 108 degrees, etc.

These mean little unless you know when the temperatures were taken, so each value also has a date/time stamp.

Value + Time

Recording each value with a timestamp allows you to see trends, pinpoint process errors, etc.

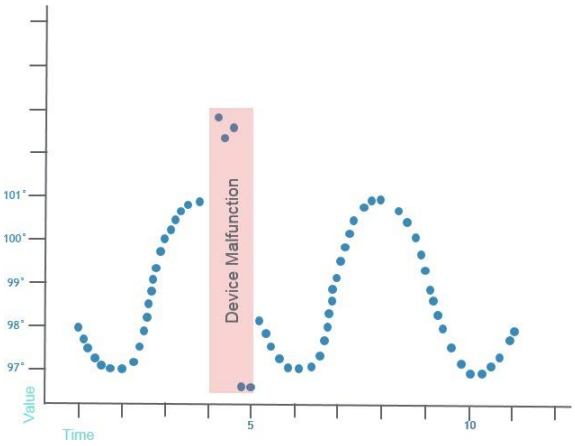
Historian tracks when a record is sent by the device and when it is received by Historian. This helps to clarify the information if there is a data lag, or if values are added or updated later.



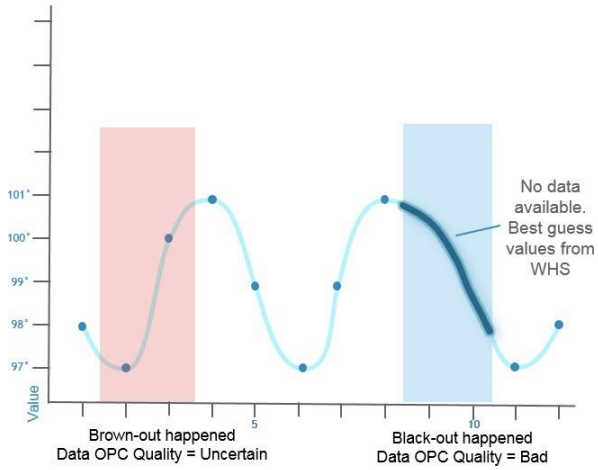
Value + Time + Quality

And because errors can occur – from minor mechanical hiccups to major area-wide blackouts – Historian also records a data quality indicator for each record.

If something happens that may affect the data quality, the quality indicator reflects that. That way, you can know if the quality less than optimal for some records, and use that information to report and as accurately as possible.



Where there are gaps, Historian can provide a best guess of what the values were.



Types of tags

AVEVA Historian can handle a wide range of data by supporting these tag types:

- **Analog**
  Measures a continuous physical quantity, such as a tank's volume or a boiler's temperature.

- **Discrete**
  Records one of two states for the tag. For example: on/off, open/closed, jam/cleared.

- **String**
  Captures a text expression--with no special format--that is treated as a single data item. A string tag could be used to capture the state of a machine; for example: "started", "stopped", "jammed", or "cleared".

- **Event**
  Records an instance when a tag meets a preset requirement. For example, a process event tag can let you know when a batch number changes.

- **System**
  Reflects a predefined system variable. System tags are used to collect the system's performance data. AVEVA Historian system tags have a "Sys" prefix (for example, SysTimeSec).

- **Analog summary**
  Reflects summarized data (minimum, maximum, average, and so on) that is configured to be replicated from one historian to another.
- **State summary**
  Reflects summarized data (minimum time in state, maximum time in state, average time in state, and so on) that is configured to be replicated from one historian to another, or stored locally.

You can configure analog, discrete, string, and legacy history event tags through the SMC. For more information, see Viewing and Configuring Tags in the *AVEVA Historian Administration Guide*.

About System Driver and System Tags

The system driver is an internal process that monitors key variables within an operating AVEVA Historian and outputs the values by means of a set of system tags. The system driver runs as a Windows service and starts automatically when the historian is started.

The system tags are automatically created when you install the historian. Also, additional system tags are created for each IDAS and replication server you configure.

The current value for an analog system tag is sent to the Storage subsystem according to a specified rate, in milliseconds. All date/time tags report the local time for the historian.

Legacy tags from upgraded systems may be retained.

Error Count Tags

The following analog tags have a storage rate of 1 minute (60000 ms). All error counts are since the AVEVA Historian is restarted or since or the last error count reset.

TagName	Description
SysCritErrCnt	Number of critical errors
SysErrErrCnt	Number of non-fatal errors
SysFatalErrCnt	Number of fatal errors

TagName	Description
SysWarnErrCnt	Number of warnings

Date Tags

The following analog tags have a storage rate of 5 minutes (300000 ms).

TagName	Description
SysDateDay	Day of the month
SysDateMonth	Month of the year
SysDateYear	Four-digit year

Time Tags

All of the following tags are analog tags. Each value change is stored (delta storage).

TagName	Description
SysTimeHour	Hour of the day
SysTimeMin	Minute of the hour
SysTimeSec	Second of the minute

Storage Space Tags

The following analog tags have a storage rate of 5 minutes (300000 milliseconds). Space remaining is measured in MB.

TagName	Description
SysSpaceAlt	Space left in the alternate storage path
SysSpaceBuffer	Space left in the buffer storage path
SysSpaceMain	Space left in the circular storage path
SysSpacePerm	Space left in the permanent storage path

I/O Statistics Tags

The following analog tags can be used to monitor key I/O information.

TagName	Description
SysDataAcq*N*BadValues*	Number of data values with bad quality received. This tag has a storage rate of 5 seconds. The maximum is 1,000,000.
SysDataAcq*N*OutsideRealtime*	The number of values per second that were discarded because they arrived outside of the real-time data window. This tag has a storage rate of 5 seconds. The maximum is 1,000,000. This tag has been deprecated and will only be available in systems migrated from AVEVA Historian 2014 and earlier.
SysDataAcqOverallItemsPerSec	The number of items received from all data sources, including HCAP. This tag has a storage rate of 10 seconds. The maximum is 100,000.
SysDataAcqRxItemPerSecN*	Tag value update received per second. This tag has a storage rate of 10 seconds.
SysDataAcqRxTotalItemsN*	Total number of tag updates received since last startup for this IDAS. This tag has a storage rate of 10 seconds.
SysPerfDataAcqNBadValues*	Number of data values with bad quality received. This tag has a storage rate of 5 seconds. The maximum is 1,000,000.
SysStatusAverageEventCommitSize	Number of events written to the A2ALMDB database per minute.
SysStatusAverageEventCommitTime	Average time, in seconds, it takes to write events to the A2ALMDB database.
SysStatusEventCommitPending	Number of events that have not yet been written to the A2ALMDB database.
SysStatusRxEventsPerSec	Number of events received per second, calculated every 10 seconds.
SysStatusRxItemsPerSec	Tag value update received per second for the system driver. This tag has a storage rate of 10 seconds.
SysStatusRxTotalDuplicateEvents	Total number of duplicate events received through different channels since startup (and discarded as duplicates).
SysStatusRxTotalEvents	Total number of events received since startup.
SysStatusRxTotalItems	Total number of tag updates received since last startup for the system driver. This tag has a storage rate of 10 seconds.
SysStatusTopicsRxData	Total number of topics receiving data. Each active IDAS "topic" and each active HCAL connection are counted. Note that

TagName	Description
	process and event history, even from the same source, count as separate connections.

*This status tag will exist for each defined IDAS. The identifying number (N) in the is the IODriverKey from the IODriver table. The number 0 designates MDAS and only applies to the SysDataAcq*N*BadValues and SysDataAcq*N*OutsideRealtime tags.

System Monitoring Tags

Unless otherwise noted, for the following discrete tags, 0 = Bad; 1 = Good.

Tag	Description
SysClassicManual Storage	Status of the data import service (aahManStSvc.exe).
SysClassicStorage	Status of the classic data redirector service (aahStoreSvc.exe).
SysClientAccessPoint	Status of the Client Access Point service (aahClientAccessPoint.exe).
SysConfiguration	Status of the configuration service (aahCfgSvc.exe). This parameter is set to 1 as long as a dynamic configuration is required or in progress.
SysDataAcqN*	Status of the IDAS service (aahIDASSvc.exe).
SysEventStorage	Status of the event storage service (aahEventStorage.exe).
SysEventSystem	Status of the classic event system service (aahEventSvc.exe).
SysIndexing	Status of the indexing service (aahIndexSvc.exe).
SysInSQLIOS	Status of the AVEVA Historian I/O Server (aahIOSvrSvc.exe).
SysMetadataServer	Status of the metadata server process (aahMetadataServer.exe)
SysOLEDB	Status of the OLE DB provider (loaded by SQL Server).
SysPulse	Discrete "pulse" tag that changes every minute.
SysReplication	Status of Replication service (aahReplSvc.exe).
SysRetrieval	Status of the retrieval service (aahRetSvc.exe).
SysStatusSFDataPending	Discrete tag indicating if one or more HCAL clients have store-and-forward data that needs to be sent to the historian. NULL = Unknown; 0 = No store-and-forward data; 1 =At least one HCAL client has data.
SysStorage	Status of the storage process (aahStorage.exe).

Tag	Description
SysSystemDriver	Status of the system driver (aahDrvSvc.exe).
SysStatusMode	Analog tag indicating the operational state of the historian. If the value is NULL, the historian is stopped. 0 = Read-only mode. 1 = Read/write mode.

*This status tag will exist for each defined IDAS. The identifying number (N) appended to the end of the tag is the IODriverKey from the IODriver table.

Miscellaneous (Other) Tags

The following table describes miscellaneous tags.

TagName	Description
SysConfigStatus	Number of database items affected by a dynamic configuration (that is, the number of entries in the ConfigStatusPending table when the commit is performed). This value is cumulative and not reset until the system is completely restarted.
SysHistoryCacheFaults	The number of history blocks loaded from disk per minute. The maximum value is 1,000. The storage rate for this analog tag is 60 seconds. For more information on the history cache, see Memory Management for Retrieval of Classic Storage Data in the *AVEVA Historian Supplemental Guide.*
SysHistoryCacheUsed	Number of bytes used for history block information. The maximum value is 3,000,000,000. The storage rate for this analog tag is 30 seconds.
SysHistoryClients	The number of clients that are connected to the Indexing service. The maximum value is 200. The storage rate for this analog tag is 30 seconds.
SysMinutesRun	Minutes since the last startup. The storage rate is 60 seconds for this analog tag.
SysRateDeadbandForcedValues	The total number of values that were forced to be stored as a result of using a swinging door storage deadband. This number reflects all forced values for all tags since the system was started.
SysString	String tag whose value changes every hour.

TagName	Description
SysTagHoursQueried	A floating point value updated every minute that indicates the total number of "tag retrieval hours" queried by all client applications during that minute. For example, if a single trend queries four tags for a 15-minute period, that is "1.0 tag retrieval hours". All tags, including replication sync queue tags and non-existent tags, are counted. Unlicensed tags are not counted.

Classic Event Subsystem Tags

The following table describes the Classic Event subsystem tags.

TagName	Description
SysEventCritActionQSize	Size of the critical action queue. For more information, see -old-Action Thread Pooling in the *AVEVA Historian Supplemental Guide.*
SysEventDelayedActionQSize	Number of entries in the delayed action queue.
SysEventNormActionQSize	Size of the normal action queue.
SysEventSystem	A discrete tag that indicates the status of the event system service (aahEventSvc.exe). 0 = Bad; 1 = Good.
SysStatusEvent	Snapshot event tag whose value changes every hour.

Replication Subsystem Tags

The Replication Service collects the following custom performance counters about its own operation, where N is a primary key of the tier-2 historian in the Runtime database of the tier-1 historian. These values are stored cyclically every 10 seconds.

TagName	Description
SysReplicationSummaryCalcQueueItemsTotal	Current number of summary calculations stored in the summary calculation queue of all tier-2 historians.
SysReplicationSummaryClientsTotal	Current number of concurrent retrieval clients performing summary calculations on the tier-1 historian for all tier-2 historians.

TagName	Description
SysReplicationSyncQueueItems*N*	Current number of items stored in the synchronization queue on the tier-2 historian of key *N*.
SysReplicationSyncQueueItemsTotal	Current number of items stored in the synchronization queue on the tier-1 for all tier-2 historians.
SysReplicationSyncQueueValuesPerSec*N*	Average synchronization queue values per second sent to the tier-2 historian of key *N*.
SysReplicationSyncQueueValuesPerSecTotal	Average values processed by the replication synchronization queue processor for all tier-2 historians.
SysReplicationTotalTags*N*	Total number of tags being replicated to the tier-2 historian of key *N*.
SysReplicationTotalValues*N*	Total number of values sent to the tier-2 historian of key *N* since the startup of the replication service.
SysReplicationTotalValuesTotal	Total number of values sent to all tier-2 historians since the startup of the replication service.
SysReplicationValuesPerSec*N*	Average values per second sent to the tier-2 historian of key *N*.
SysReplicationValuesPerSecTotal	Average values per second sent to all tier-2 historians.

Performance Monitoring Tags

You use performance monitoring tags to monitor CPU loading and other performance parameters for various AVEVA Historian processes. (All of these values map to equivalent counters that are used in the Microsoft Performance Logs and Alerts application.)

The following tags allow you to monitor the percentage CPU load for all processors:

System Tag	Description
SysPerfAvailableBytes	Amount of free memory (RAM). If the amount of available memory is over 4,294,967,296, then the tag shows the remainder of the amount of memory divided by 4,294,967,296.

System Tag	Description
SysPerfAvailableMBytes	Amount of free memory (RAM). Use this tag to monitor systems that have a larger amount of memory. The value for this tag is the amount of available memory divided by 1 million.
SysPerfCPUMax	The highest CPU load of any single core, expressed as a percentage (0-100). For example, on a quad core system where the current loads for each core are 25%, 40%, 60% and 10%, this tag will be "60".
SysPerfCPUTotal	The overall processor load as a percentage of all cores (0-100).
SysPerfDiskTime	Percentage of elapsed time that the disk drive was busy servicing read or write requests.
SysPerfMemoryPages	Rate at which pages are read from or written to disk to resolve hard page faults.

The remaining system tags are used to monitor performance for each historian service or process and for the Microsoft SQL Server service. For more information on services, see AVEVA Historian Processes.

There are six system performance tags per each service or process. These tags adhere to the following naming convention:

- SysPerf<*service*>CPU
- SysPerf<*service*>HandleCount
- SysPerf<*service*>PageFaults
- SysPerf<*service*>PrivateBytes
- SysPerf<*service*>PrivateMBytes
- SysPerf<*service*>ThreadCount
- SysPerf<*service*>VirtualBytes
- SysPerf<*service*>VirtualMBytes

where <*service*> can be any of the following:

- ClassicManualStorage
- ClassicStorage
- ClientAccessPoint
- Config
- DataAcq
- EventStorage
- EventSys
- Indexing
- InSQLIOS
- MetadataServer
- Replication
- Retrieval
- SQLServer
- Storage
- SysDrv

These tags have a cyclic storage rate of 5 seconds.

Note: The six performance tags will exist for each defined IDAS. The identifying number (N) appended to the end of the "DataAcq" portion of the tagname is the IODriverKey from the IODriver table. For example, 'SysPerfDataAcq1CPU'.

The following table describes the suffixes assigned to the names of system performance tags:

Suffix	Description
CPU	Current percentage load on the service, expressed as a percentage of total CPU load. For example, on a quad core system, if the service is using 20% of one core, 40% of another core, and 0% of the other two cores, this tag will be 15%.
HandleCount	Total number of handles currently open by each thread in the service. A handle is a identifier for a particular resource in the system, such as a registry key or file.
PageFaults	Rate, per second, at which page faults occur in the threads executing the service. A page fault will occur if a thread refers to a virtual memory page that is not in its working set in main memory. Thus, the page will not be fetched from disk if it is on the standby list (and already in main memory) or if it is being used by another process.
PrivateBytes	Current number of bytes allocated by the service that cannot be shared with any other processes. If the amount is over 4,294,967,296, then the tag shows the remainder of the amount divided by 4,294,967,296.

Suffix	Description
PrivateMBytes	Current number of Mbytes allocated by the service that cannot be shared with any other processes.
ThreadCount	Current number of active threads in the service. A thread executes instructions, which are the basic units of execution in a processor.
VirtualBytes	Current size, in bytes, of the virtual address space that is being used by the service. If the amount is over 4,294,967,296, then the tag shows the remainder of the amount divided by 4,294,967,296.
VirtualMBytes	Current size, in Mbytes, of the virtual address space that is being used by the service.

Important: You need to ensure that the memory that SQL Server reserves for the AVEVA Historian is adequate for the expected load. Based on your particular environment, you may need to adjust the SQL Server MemToLeave allocation. For more information on MemToLeave, see the Microsoft documentation.

Deprecated Tags

The following tags introduced in previous versions of Historian have been deprecated. They may still be present in upgraded systems, but will have static values of "0" or NULL.

TagName	Description

Data Quality

There are three aspects of quality handling for the system:

- Data quality assignment by data acquisition devices (OPCQuality)
- Storage subsystem quality for the AVEVA Historian (QualityDetail)

- Client-side quality definitions (Quality)

For more information on quality for tags, see Value, Time, and Quality (VTQ) for Tags.

The historian uses three distinct parameters to indicate the quality of every historized data value: Quality, QualityDetail, and OPCQuality. OPCQuality is strongly related to QualityDetail. Quality is a derived measure of the validity of the data value, while QualityDetail and OPCQuality indicate either a good quality for the data value, or the specific cause for degraded quality of the data value.

The historian persists four bytes of quality information for every data value that is historized. (This does not imply that four bytes of quality data are actually stored for every data value, but it guarantees that every data value can be associated with 4 bytes of quality information when the value is retrieved.) The 4 bytes of quality information comprises 2 bytes for QualityDetail and 2 bytes for OPCQuality. QualityDetail and OPCQuality are related, but may have different values.

In essence, OPCQuality is provided as for client applications that are fully aware of the OPC data quality standard, while QualityDetail (and Quality) are maintained to ensure proper operation of existing and legacy client applications. OPCQuality is sent by the data source, and QualityDetail is added by the AVEVA Historian.

If you import history blocks from IndustrialSQL Server 7.1 or earlier (that do not use OPC data quality), the OPCQuality is determined by using the top 2 bits of the lower 8 bits of the QualityDetail.

Viewing Quality Values and Details

The currently supported OPCQuality values are stored in the OPCQualityMap table of the Runtime database. To view OPCQuality values and descriptions, execute the following query:

```
SELECT OPCQuality, Description FROM OPCQualityMap
```

The currently supported QualityDetail values are stored in the QualityMap table of the Runtime database. To view the complete list of QualityDetail values and descriptions, execute the following query:

```
SELECT QualityDetail, QualityString FROM QualityMap
```

The following notes apply to the QualityDetail values:

- The boundary point for a QualityDetail of 448 is 1 second. QualityDetail of 448 is a concatenation of quality states of good quality 192 and the bit flag indicating a value in the future. For more information, see *QualityDetail Bit Layout* on page 23.
- Although a quality detail of 65536 is used to indicate block gaps for tier-2 tags, NULL values are not produced for block gaps for tier-2 tags.

Basic QualityDetail Codes

Quality detail codes are metadata used to describe data values stored or retrieved using the AVEVA Historian. Quality details are 2 byte values that use the low order byte for the basic quality detail and the high order byte to store quality detail flags. The basic quality detail and quality detail flags are OR'd together.

The following table lists the basic quality detail codes. These codes use the low order byte, and the individual bit values do not have independent significance. For information on how to retrieve the complete list of quality detail codes, see *Viewing Quality Values and Details* on page 20.

QualityDetail	Quality String	Description	Created By
0	Bad Quality of undetermined state	Bad quality.	Storage
1	No data available, tag did not exist at the time	For cyclic, delta, full, interpolated, and best-fit, retrieval started before the tag was created (NULL value).	Retrieval
10	Communication loss	Disconnect between a client and the historian server.	Classic data redirector, HCAP
17	Duplicate time stamp; infinite slope	Slope calculation of a vertical line.	Retrieval
20	IDAS overflow recovery	An IDAS data buffer overflow was detected, and all overflow values were discarded to recover.	IDAS
24	IOServer communication failed	I/O Server communication failure. Application Server sends quality of 24 or 32.	Classic data redirector, HCAL, MDAS
33	Violation of History Duration license feature	You have queried beyond the licensed time range limit.	Retrieval
44	Pipe reconnect	First VTQ sent after the client has reconnected to the historian.	Classic data redirector, HCAL, MDAS
64	Cannot convert	The calculated point was based on points of different QualityDetails: Good, Doubtful, or Bad. This is, for example, used in the integral and average retrieval modes.	Retrieval
150	Storage startup	Storage startup.	Classic data redirector

QualityDetail	Quality String	Description	Created By
151	Store forward storage startup	Store-and-forward storage startup.	Classic data redirector
152	Incomplete calculated value	Incomplete calculated value.	Replication
192	Good	Good value.	HCAL, MDAS
202	Good point of version Latest	Current value is the latest update which "masks" a previous original value.	Classic data redirector, retrieval
212	Counter rollover has occurred	Counter count reached the roll-over value.	Retrieval
248	First value received in store forward mode	First VTQ stored in store-and-forward mode after the client has disconnected from the historian.	HCAL, MDAS
249	Not a number	Value received was not a number, infinite value (NaN).	Classic data redirector, HCAL, MDAS
252	First value received from IOServer	First value received after an I/O Server reconnect.	Classic data redirector
65536	No data stored in history, NULL	NULL value. **Note:** Although this quality detail is used to indicate block gaps on Tier 1 tags, NULL values are not produced for block gaps for Tier 2 tags.	Retrieval

QualityDetail Flags

The following bit flags are stored in the high order byte of the quality detail. Each bit is a flag that carries specific meaning.

- 0x0000 - Basic QualityDetail.

The following hi-byte flags are or'ed with the basic QualityDetail. The three flags cannot coexist; only one is set at any point in time.

- 0x0100 - Value received in the future.
- 0x0200 - Value received out of sequence.
- 0x0400 - Configured for server time stamping.

The following hi-byte flag can coexist with any of the prior four combinations.

- 0x0800 - Point generated by Rate Deadband filter.

The following hi-byte flag can coexist with any of the prior eight combinations.

- 0x1000 - Partial cycle, advance retrieval results.

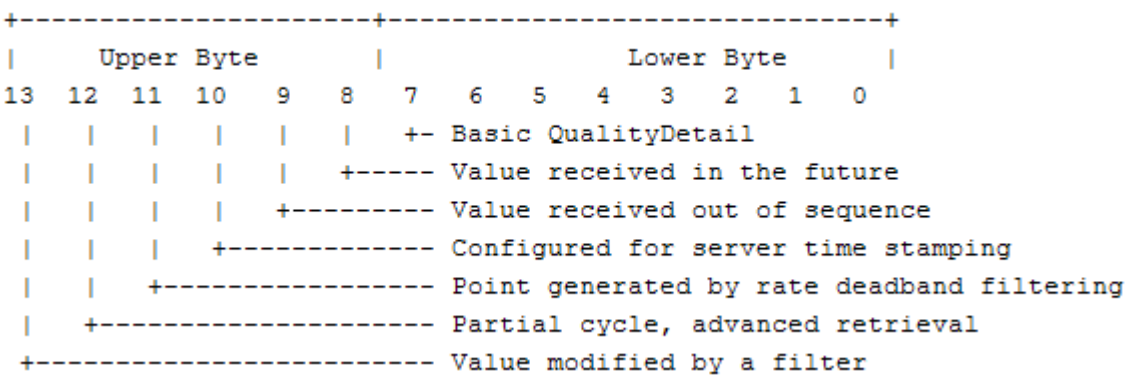
The following hi-byte flag can coexist with any of the prior eight combinations.

- 0x2000 - Value has been modified by filter.

This bit is set during retrieval to indicate that retrieval has modified a value or time stamp due to applying a filter such as SnapTo() or ToDiscrete().

QualityDetail Bit Layout

The QualityDetail bit layout is as follows:



Acquisition and Storage of Quality Information

When a data value is acquired for storage in the AVEVA Historian, it is usually accompanied by a quality indicator generated at the source of the data. In general, the historian passes the source-supplied quality indicator to the Storage subsystem unmodified, but there are instances where the Data Acquisition subsystem actually modifies the quality of the data value.

Quality for Data Acquired from I/O Servers

A data value acquired from an AVEVA I/O Server by an IDAS always has 2 bytes of quality information attached to it by the I/O Server. The IDAS presents the 2 bytes of quality information supplied by the I/O Server to the AVEVA Historian Storage subsystem as OPCQuality, and the QualityDetail is set to 192. An exception to this is if the quality value sent by the I/O Server is 24, in which case both OPCQuality and QualityDetail are set to 24.

However, in some instances, IDAS will overwrite QualityDetail with a special value to indicate a specific event or condition. For more information on these values, see *Viewing Quality Values and Details* on page 20. When IDAS overwrites QualityDetail with one of the reserved values, it does not modify the OPCQuality value. Thus, QualityDetail contains the reserved value set by IDAS, while OPCQuality retains the value provided by the data source.

The SysDataAcq*N*BadValues system tag tracks the number of data values with bad quality that are coming from an IDAS. If this value is high, you should make sure that there is not problem with the IDAS or I/O Server. For more information on system tags, see *About System Driver and System Tags* on page 10.

Quality for Data Not Acquired from I/O Servers

The AVEVA Historian supports data acquisition from a variety of sources other than AVEVA I/O Servers. These sources include properly formatted CSV files, SQL queries, and the Application Server.

Quality for data acquired from these sources is handled the same as for data received from AVEVA I/O Servers, with a few exceptions:

- For data acquired from CSV files, the specified quality is always interpreted as OPCQuality, and the QualityDetail is set to 192, unless the specified quality is 24, in which case both OPCQuality and QualityDetail are set to 24.
- If MDAS or HCAL is responsible for acquiring the data (as is the case for SQL queries and AVEVA Application Server), the quality value presented by the source is preserved in OPCQuality, and QualityDetail is set to 192. Exceptions to this are:
    - If a quality value of 32 presented by the source, OPCQuality is set to 32 and QualityDetail is set to 24.
    - If the data value presented by the source is infinite or NaN (not a number), OPCQuality contains the actual quality presented by the source, and QualityDetail is set to 249.
    - If a special condition or event occurs, MDAS or HCAL will substitute a reserved value for QualityDetail. For more information, see *Viewing Quality Values and Details* on page 20.

Client-Side Quality Values

Three quality indicators are exposed to clients:

- A 1-byte short quality indicator (Quality)
- A 4-byte long quality detail indicator (QualityDetail)
- A 4-byte long OPC quality indicator (OPCQuality)

Quality contains summary information that essentially falls into the categories of Good, Bad, Doubtful, or InitialValue. This implicit information is derived from the data source as well as the AVEVA Historian.

Quality is a 1-byte indicator that provides a summary meaning for the related data item. Quality is an enumerated type with the following values:

Hex	Dec	Name	Description
0x00	0	Good	Good value.
0x01	1	Bad	Value marked as invalid.
0x10	16	Doubtful	Value is uncertain.
0x85	133	Initial Value (Good)	Initial value for a delta request.

Initial Value and Doubtful are derived from QualityDetail. The Initial Value is dependent on the type of query that is executed. For more information, see:

- -old-Using Comparison Operators with Delta Retrieval
- -old-Using Comparison Operators with Cyclic Retrieval and Cycle Count
- -old-Using Comparison Operators with Cyclic Retrieval and Resolution

QualityDetail contains detailed information that is potentially specific to the device which supplied it. Quality values may be derived from various sources, such as from I/O Servers or from the AVEVA Historian Historian storage system.

To retrieve a listing of the quality values that are used by the historian, see *Viewing Quality Values and Details* on page 20.

Data acquisition: Getting data into Historian

Historian acquires and processes data several times faster than a traditional relational database.

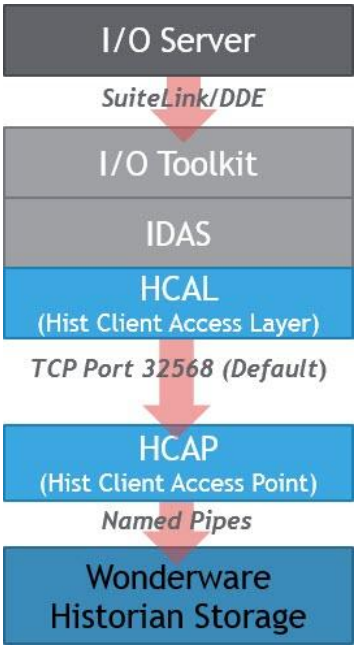
When data is acquired, Historian attaches a timestamp and quality stamp to the value before committing the record to storage.

Historian acquires both original data and revision data. Historian tracks any modifications to the data, and tracks the means by which it was modified.

Sources of data

Historian can accept data from a number of sources. The most typical scenario is data acquisition from an I/O server.

Data acquisition from I/O servers



An I/O server provides data from plant floor devices to AVEVA Historian using the SuiteLink or Dynamic Data Exchange (DDE) protocol.

First, the I/O server collects data values from programmable logic controllers (PLCs), Remote Telemetry Units (RTUs), and similar devices on the factory floor.

Then, the I/O server uses the SuiteLink or DDE protocol to time and quality stamp each data value it collect.

Next, it passes the data values through the Data Acquisition subsystem, or IDAS. IDAS seamlessly handles data values,regardless of their time.

Note: The SuiteLink protocol can be used to collect data from an I/O server on the same or different computer than the IDAS instance. The DDE protocol can be used only when the I/O server is on the same computer as the IDAS instance.

For each data value acquired by IDAS, the timestamp, value, and quality are attached.

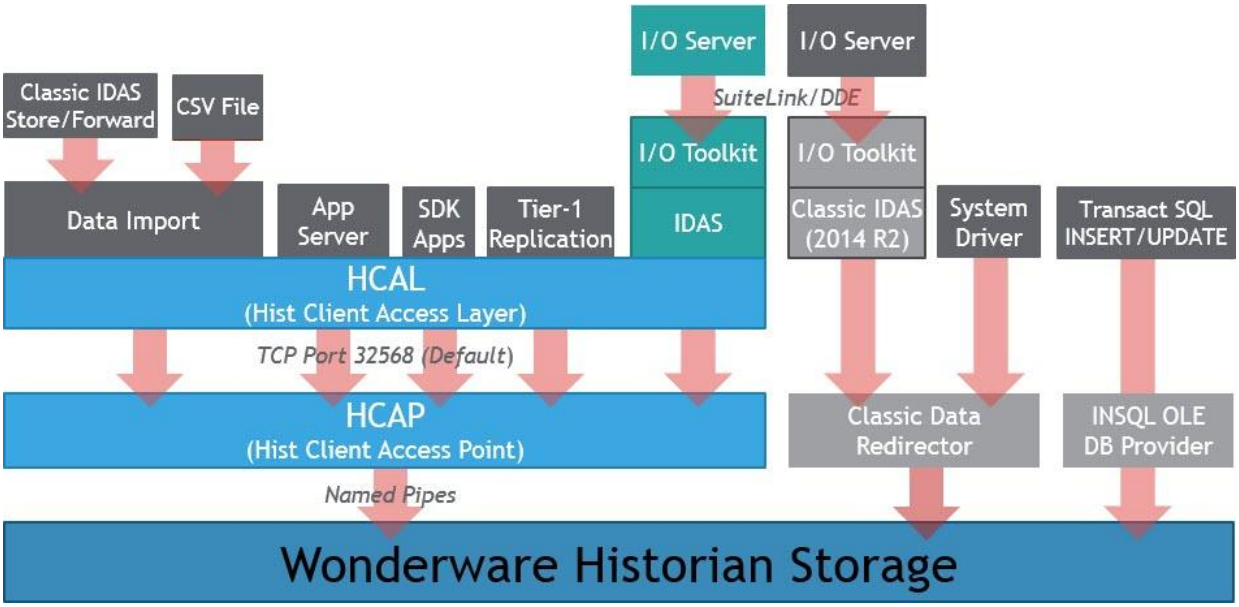
Then the values are sent through the Historian Client Access Layer (HCAL) to a Historian Client Access Point (HCAP) on the Historian server, and then to storage. HCAL is a client-side software layer that provides programmatic access to storage, retrieval, and system configuration functionality in the AVEVA Historian.

Historian accepts and historizes each data value according to the storage rules for the tag to which the data value belongs.

For more details, see Configuring Data Acquisition in the *AVEVA Historian Administration Guide*.

Other data acquisition options

As this diagram illustrates, AVEVA Historian can accept data from a range of sources.



In addition to I/O servers, Historian can acquire data from these sources:

- **TransactSQL INSERT and UPDATE statements**
  You can insert or update history data in the AVEVA Historian extension tables using Transact-SQL INSERT and UPDATE statements.

  For more information, see Importing, Inserting, or Updating History Data in the *AVEVA Historian Administration Guide*.

- **CSV and LGH files**
  Using the Historian Data Importer utility (aahImport), you can add history data from a file to AVEVA Historian. This utility reads data from InTouch history (LGH) files or comma-separated value (CSV) files, and then sends the data to the Historian server via HCAL.

  Imported data is integrated with data currently stored in history blocks, providing you with seamless access to all your data.

  For more information, see Importing, Inserting, or Updating History Data in the *AVEVA Historian Administration Guide*.

- **App Server, custom SDK client applications, tier-1 replication, and other sources**
  These sources are also able to use HCAL to send data to Historian.

- **AVEVA Historian itself**
  Configuration data comes from the Historian itself.

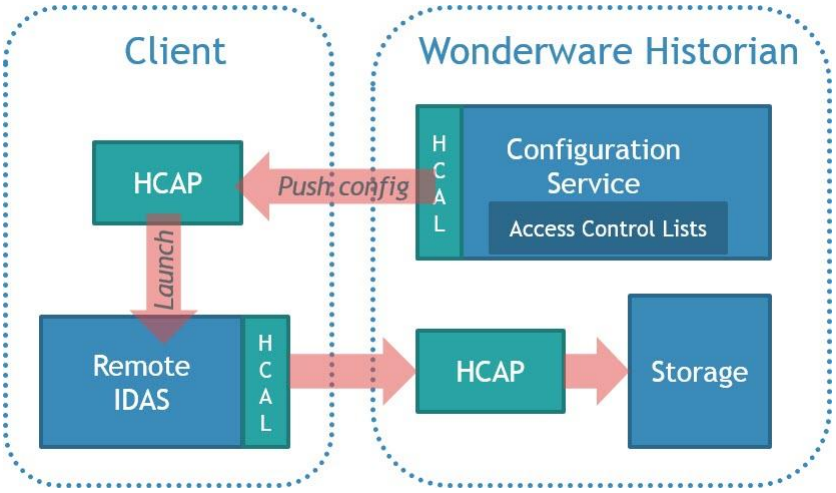
About AVEVA Historian data security

AVEVA Historian uses an integrated security model to control who can access and update data for the historian. Using this model, each person and computer that accesses or updates historian data is assigned membership in one of three security groups:

- Administrators
- Power Users
- Users

Data can be passed from any computer that's a member of the historian's Power User or Administrator group. (Computers must be on the same domain as the historian.)

When data is ready to be sent from a remote computer, the AVEVA Historian pushes configuration information, including ACLs (access control lists) that define access permissions, to HCAP on the client computer. HCAP launches IDAS on the remote computer and data is sent through HCAL to the historian.



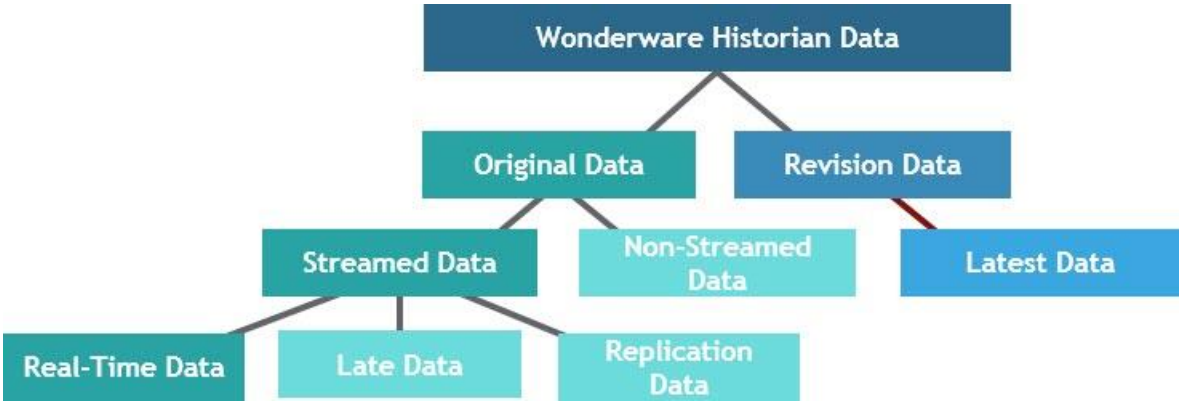
Store-and-forward safeguards

Historian uses a store-and-forward method to protect against data loss if communication is interrupted between the data source and Historian.

Systems using the Data Acquisition Subsystem (IDAS) or Historian Client Access Layer (HCAL) to send data to Historian are able to use the store-and-forward method in case of communication breaks. If the data source loses communication with Historian, the source stores the collected data until communication is reestablished. Then, it forwards the stored data to Historian.

Data categories

AVEVA Historian is able to process and store data in a variety of ways. It categorizes each data record by type to provide a consistent framework for data operations. Each category of data has a separate set of characteristics and is handled differently by the historian.



Original versus revision data

Data acquired by AVEVA Historian can be categorized as:

- **Original data** is the data received from the data source originally—that is, for the first time. For example, a real-time stream of data from an I/O server represents original data. Usually the original data arrives to a Historian in high data volumes for many tags with timestamps close to each other.
- **Revision data**, by contrast, is data that corrects or appends original data. Revision data operations are performed on a per-tag basis and typically have far lower volumes than original data.

Streamed versus non-streamed data

Original data can be streamed or non-streamed.

- **Streamed:** If the data source is able to enforce time order for its output, the data is streamed data. Streamed data has three subtypes:
  - Real-time data is in time order, where the timestamp is in the past relative to the current AVEVA Historian time.
  - Late data is in time order, where the timestamp is far in the past compared to the current AVEVA Historian time.
  - Replication data is data that has been replicated from a tier-1 historian to a tier-2 historian.
- **Non-streamed:** If the time order is not enforced, the data is non-streamed.

Data storage: Preserving huge amounts of data over time

Historian, through its Storage subsystem, saves plant data from various sources to disk.

Historian's efficient storage model includes:

- **A range of storage modes**
  Historian's storage modes allows storage of every meaningful value and extrapolation of all other values at retrieval time.
- **History blocks and partitions**
  Historian efficiently warehouses huge amounts of data using a compact storage format that is optimized for time-series data.
- **Auto-summarization**
  Historian calculates and stores automatic summary records as real-time analog tag values. These auto-summaries are used provide fast, seamless data retrieval, no matter what the granularity.

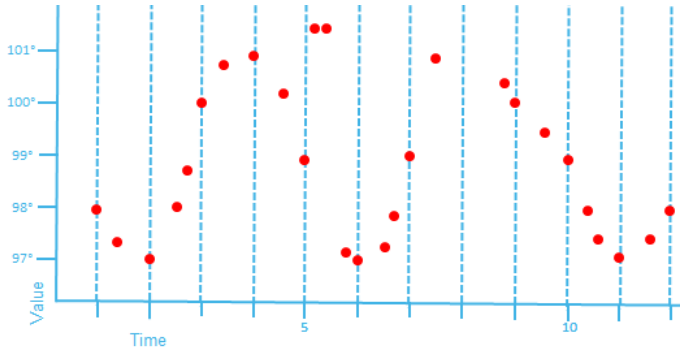
For information about configuring the Storage subsystem, see Managing Data Storage in the *AVEVA Historian Administration Guide*.

Storage modes

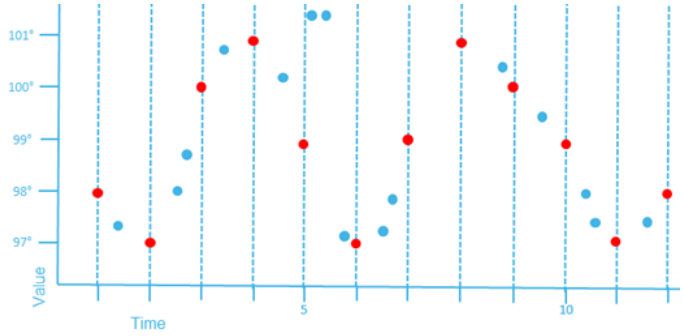
Depending on a tag's definition, Historian uses one of these storage modes to retain the values received for that tag:

- **No storage -** No values are stored.
- **Forced storage** - All collected values are stored.

For comparison's sake, this is what forced storage looks like. The red dots represent collected values. All of these values are stored by Historian.



- **Cyclic storage** - Only values that occur at a specified time interval are stored. Using the same collected values as shown above, cyclic storage retains only the values represented by red dots.



- **Delta storage** - Only changed values are stored.

Types of delta storage

Delta storage, as a rule, requires Historian to store any value that is different than the previously received value. Time or deadband rules can be applied for delta storage to further constrain what values are stored.

These are all types of delta storage:

- **Time-enforced delta storage** -- Any changed value is recorded. Additionally, a record must be stored after a given amount of time. This storage mode is used by the Historian SDK.
- **Time deadband delta storage** -- Only changes outside of a particular time deadband are stored.
- **Value deadband delta storage** -- Only changes outside of a particular value deadband are stored.
- **Rate of change (swinging door) deadband storage** -- Only changes outside of a particular rate-of-change (swinging door) deadband are stored.

Measuring change with deadbands

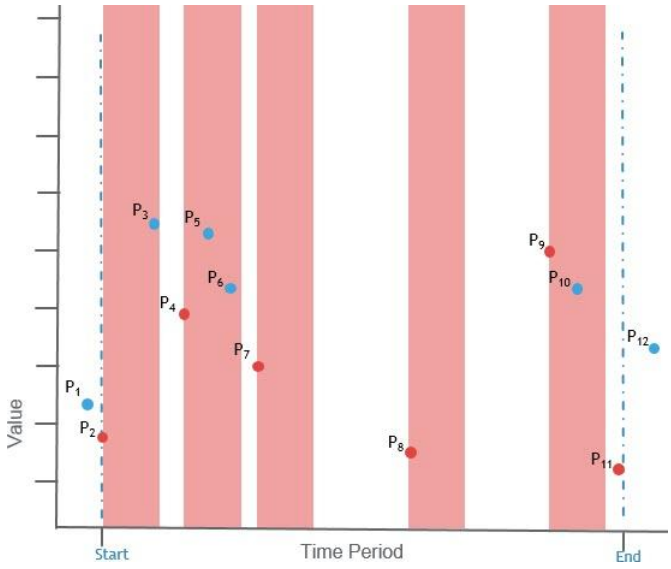
Delta storage retains only those values that have significantly changed from the previously stored value.

For example, if you had a discrete (binary) tag that reflected the state of a power switch, you may not want to record every time the system checks to see that it is switched on. You might really be interested only in when it switches off when it is supposed to be running, and when it gets switched on again.

For analog (numeric) tags, you may only care only about large changes, but not tiny ones. Or, you may want a snapshot of values at certain intervals, and not every one that is reported. You can filter out extraneous value with deadbands.

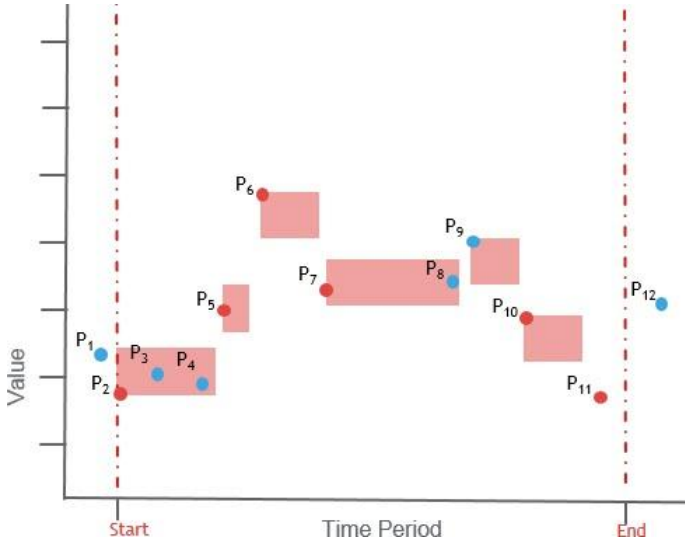
A **time deadband** is a time filter. It marks the minimum time (milliseconds) between stored values for a single tag. Any value changes that occur within the time deadband are not stored.

For example, these red and blue points are all the values reported for a certain tag. The orange bars represent the time deadband, which starts over with every reported value. Only the red points (P2, P4, P7, P8, P9, P11) are stored. The other points are excluded because they fall within a deadband or outside of the time period.



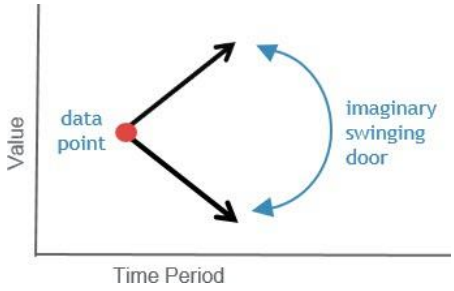
A **value deadband** is a filter that marks the percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored.

Here, the orange bars represent the value deadband, which starts over with every reported value. Only the red points (P2, P5, P6, P7, P10, P11) are stored. The other points are not stored because they fall within a deadband or outside of the time period. P9 is not stored because P8 was discarded and it is within the percentage deviation.



A **swinging-door deadband** marks a rate of change deadband, based on changes in the slope of the received values.

For example, specifying a swinging door deadband value of 10 percent means that values are stored if the percentage change in slope of the consecutive data values exceeds 10 percent.



For more information on delta storage modes, see About Delta Storage Mode in the *AVEVA Historian Administration Guide*.

History blocks and partitions

AVEVA Historian stores data in history blocks. History blocks use a proprietary file format and are essentially subfolders of the main historian storage folder.

Although the tag values are stored in history blocks on disk, the values appear to be saved to tables in the Runtime database.

Each history block stores all data for a specified duration. The default history block duration is one day, but may be as little as one hour. When there is data to be stored in that time interval, Historian creates a new history block for that data. For example, if a history block is defined for a day's worth of data, when it receives the first data value for the second day, Historian creates a new one-day history block and places the corresponding data into the new block.

As data is acquired, the size of these history blocks grows on a continual basis, being limited only by the size of the hard disk on which the historian resides.

If the historian was not running for some time, or if a history block is deleted, for a certain time period, there may be a gap in the sequence of history blocks -- also known as a *block gap*.

History block formats are specially optimized for storing time-series data, while general-purpose database management systems typically are not.

Compact storage formats reduce the storage space requirements than would be required in a general-purpose database. Upon retrieval, historical data is presented by the AVEVA Historian OLE DB provider as if it were stored in SQL Server tables.

Historian partitions

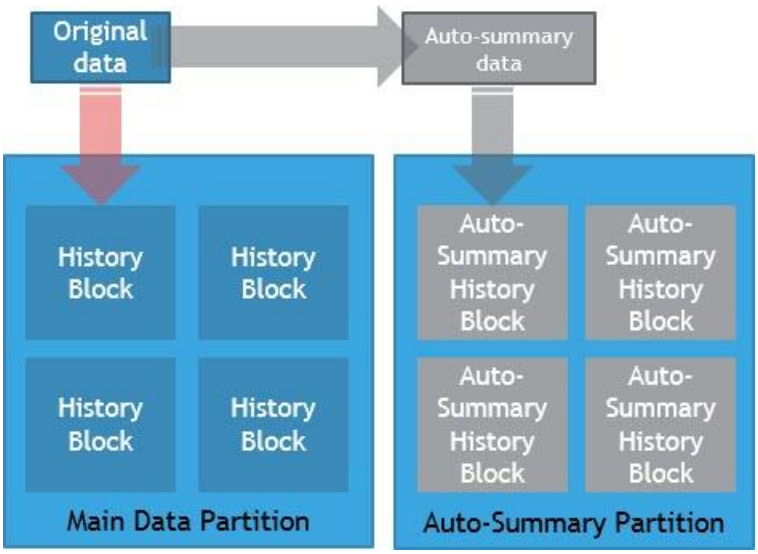
Historian organizes history blocks within partitions. As real-time data arrives, Historian stores it in history blocks located in the main data partition. At the same time, Historian automatically computes and records a corresponding hourly summary for each analog tag value received. The auto-summary values are stored in auto-summary history blocks within the auto-summary partition.

For more information on history blocks and partitions, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.

Auto-summarization

For every analog tag in the system, AVEVA Historian creates a local replication entity and a one-hour summary tag. As values arrive for an analog tag, Historian automatically computes and records a summary.

Auto-summary values are stored in their own history blocks within the auto-summary partition.



With auto-summarization, Historian can quickly and efficiently retrieve large-volume data for a long duration, even months or years.

Note: The auto-summarization feature is enabled by default, but can be disabled.

Data retrieval: Transforming data into information

AVEVA Historian can process SQL-based queries from any number of client applications, including: AVEVA Insight, AVEVA Historian Client, and ad hoc SQL query tools. It can also process queries via the OData interface and from SDK client applications.

When Historian receives a request for data, it performs the following steps:

1. Locates the requested data.
    - Historized process data is stored in history blocks.
    - Configuration data is stored in SQL Server database tables.
    - Replication data is stored in history blocks.
2. Apply a retrieval mode to the data.

    Because of the enormous amount of data potentially associated with a facility, Historian provides several retrieval modes that help interpret that data, turning the collected data into usable information.
3. Returns the results to the client application.

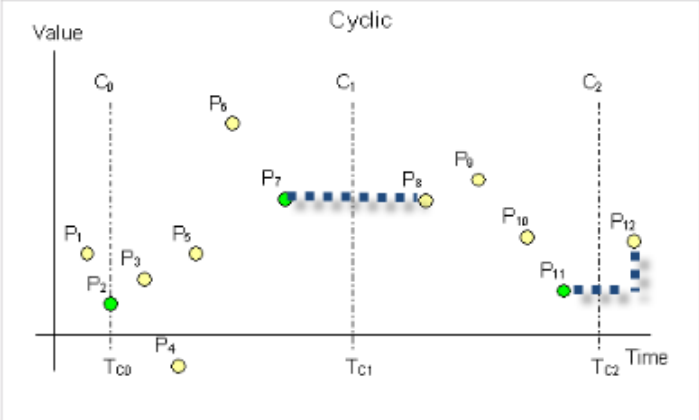
For more information about retrieving data from AVEVA Historian, see the *AVEVA Historian Retrieval Guide*.

Retrieval modes

Historian can acquire and store huge amounts of data and allows you to choose from among several retrieval modes to view and interpret the data you need.

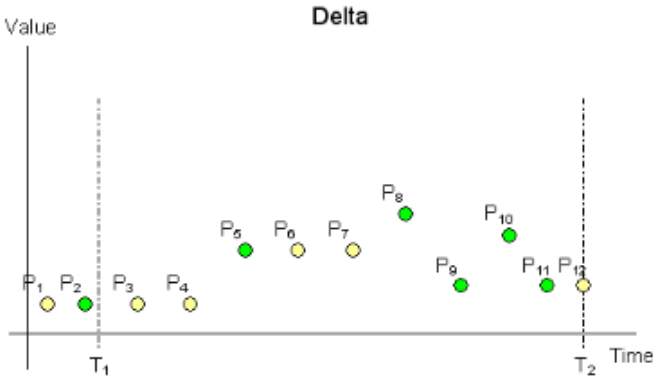
Cyclic

Retrieves one value per cycle. Whatever the value is when the cycle begins.



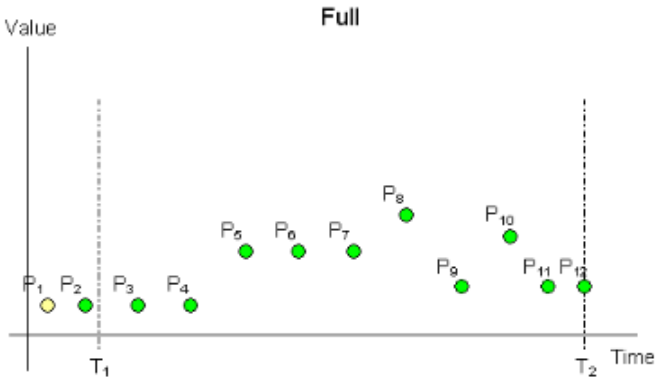
Delta

Retrieves a value each time the value changes from the previous value. For example, if the value of "4" followed an earlier value of "4", it would not be retrieved. But if "4’" followed "3", it would.



Full

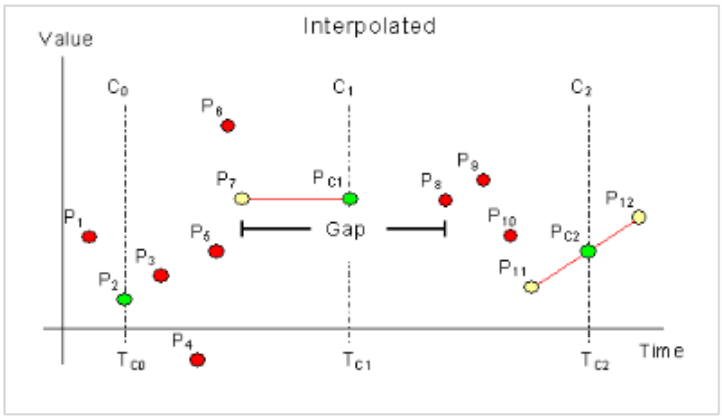
Every value within a time period is retrieved.



Interpolated

Based on values before and after a certain point in time, Historian estimates the value for that time.

In this example, P2 is located exactly at the query start time. Because of this, P2 is returned at that time without need for interpolation. At the following cycle boundary, point PC1 is returned, which is the NULL value represented by P7 shifted forward to time TC1. At the last cycle boundary, point PC2 is returned, which has been interpolated using points P11 and P12.

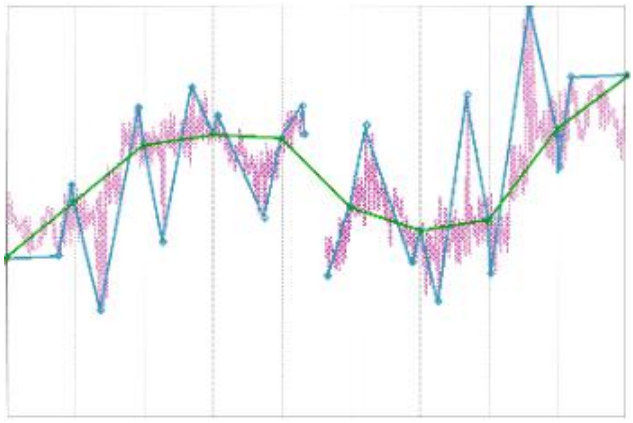


Best Fit

"Best fit" retrieval allows for a compromise between delta retrieval and cyclic retrieval.

Delta retrieval can accurately represent a process over a long period of time, but requires a large number of data values. Cyclic retrieval is much more efficient, but less accurate, because of fewer values.

Best fit provides faster retrieval, like cyclic retrieval, plus the better representation, like delta retrieval.

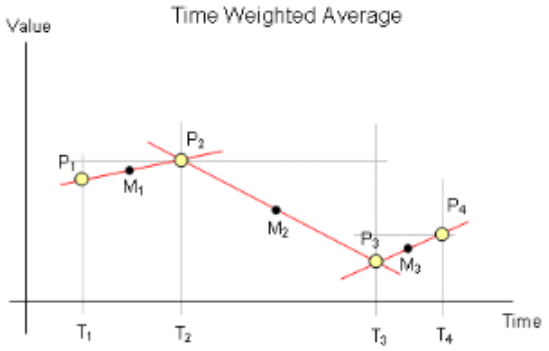


Average

Uses a time-weighted average algorithm to calculate the value for each retrieval cycle.

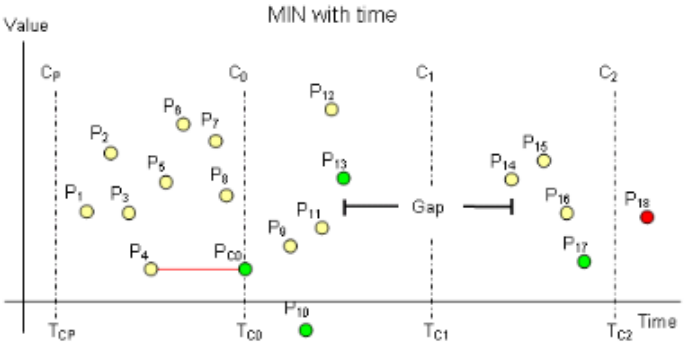
For the following data values of a tag that uses linear interpolation, the time-weighted average is computed as:

Average = (((P1 + P2) / 2) x (T2 - T1)) + (((P2 + P3) / 2) x (T3 - T2)) + (((P3 + P4) / 2) x (T4 - T3)) / (T4 - T1)



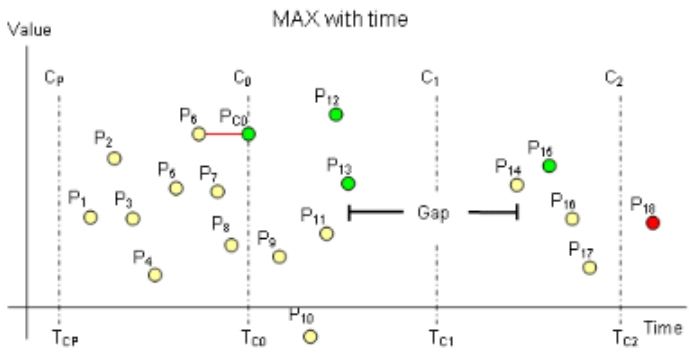
Minimum

Returns the minimum value from the actual data values within a retrieval cycle. If there are no actual data points stored on the historian for a given cycle, nothing is returned. If there are NULL values in the cycle, NULL is returned for that cycle.



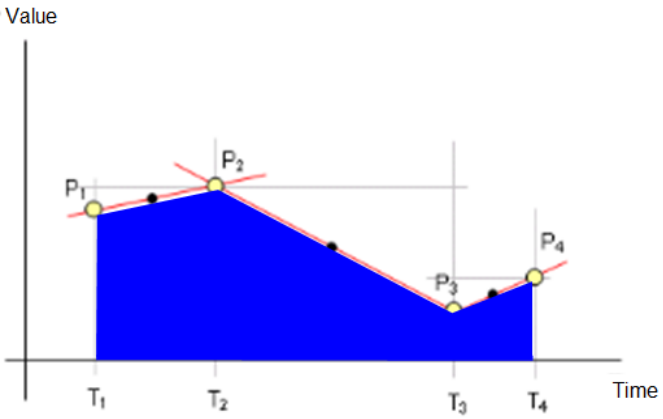
Maximum

Similarly, this mode returns the maximum value of actual data for the retrieval cycle.



Integral

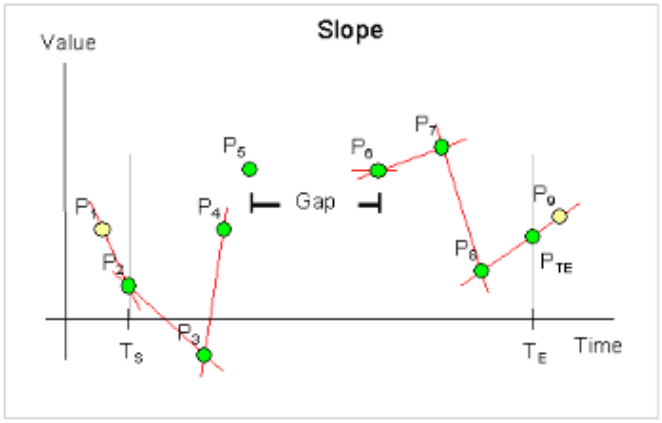
Calculates the values at retrieval cycle boundaries by integrating the graph described by the points stored for the tag. In other words, it works much like average retrieval, but it additionally applies a scaling factor. This retrieval mode is useful for calculating volume for a particular tag (for example, gallons of water flowing through a valve over a certain period).



Integral retrieval works with analog tags only. For all other tags, normal cyclic results are returned.

Slope

Returns the slope of a line drawn through a given point and the point immediately before it, thus expressing the rate at which values change.



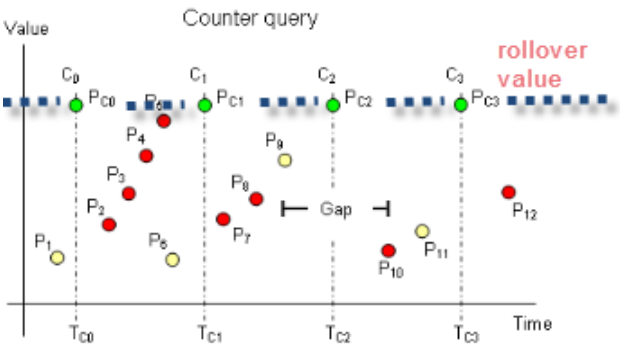
For example, two points P1 and P2 occur at times T1 and T2. The slope is calculated as:

(P2 - P1) / (T2 - T1)

The difference between T1 and T2 is measured in seconds, so the returned value represents the change in engineering units per second.

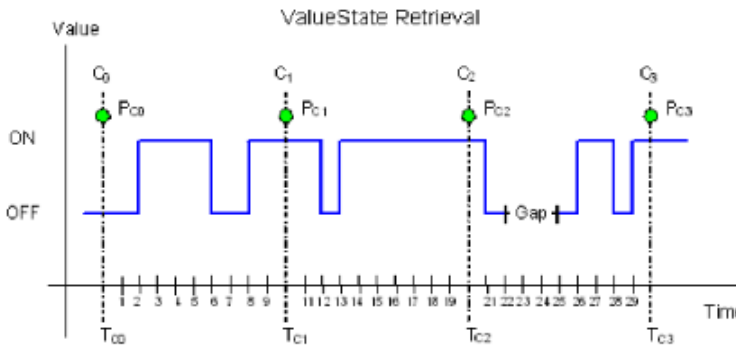
Counter

The change in a tag's value from the beginning to the end of the period, factoring in any rollover value for the counter. This retrieval mode is useful for determining how much of an item was produced during a particular time period.



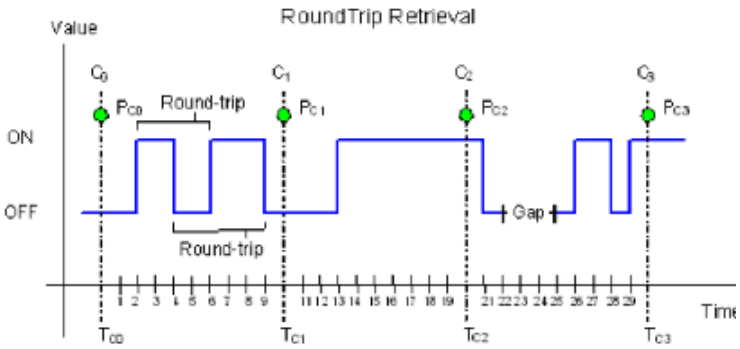
ValueState

Returns information on how long a tag has been in a particular value state during each retrieval cycle. That is, a time-in-state calculation is applied to the tag value.



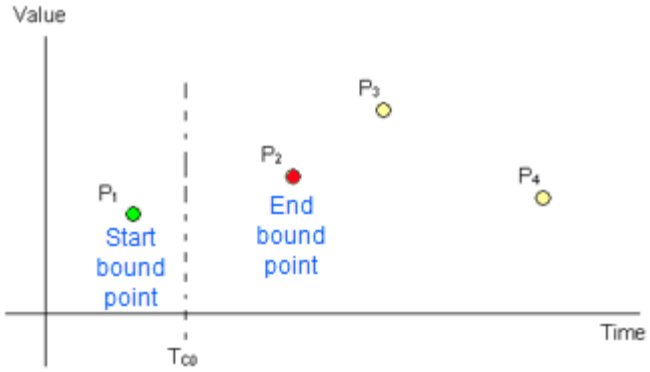
RoundTrip

Like ValueState retrieval, this mode uses state occurrences within a period for its calculations. RoundTrip retrieval calculates the time between consecutive leading edges of the same state.



Bound Value

Retrieves either the start bound point or the end bound point for a requested point in time. For a start bound point, Historian retrieves the first value on or before the requested date/time. For an end bound point, Historian retrieves the first value after the requested date/time.



Data query tools: Accessing your data

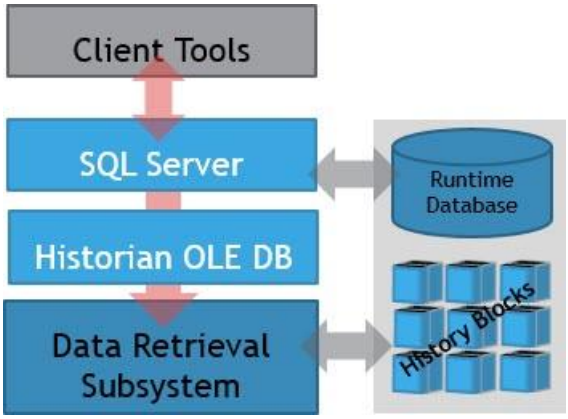
To report on the data you have stored in AVEVA Historian, you can use:

- AVEVA Insight, a tool included with AVEVA Historian
- Transact-SQL queries
- Any other query tool that can access SQL data sources

AVEVA Historian is part of a client/server architecture that supports desktop client applications, while ensuring the integrity and security of data on the server. This architecture provides common access to time-series data and associated configuration, event, and business data.

The computing power of both the client and the server is exploited by optimizing processor intensive operations on the server and minimizing data to be transmitted on the network to improve system performance.

Microsoft SQL Server acts as the gateway for accessing any type of information in the historian. Historian uses Microsoft linked server technology to plug in its own OLE DB provider. Because of this, any client application that can connect to Microsoft SQL Server can also connect to *AVEVA Historian*.



For users with client applications, it seems that queries are made to the Runtime database. That database is, in fact, the logical interface to the data.

When it receives a request for data, the Retrieval subsystem retrieves the historized data from history blocks.

Querying with AVEVA Insight

AVEVA Insight (included with AVEVA Historian) is a search-based tool that lets you quickly turn your data into easy-to-read charts.

With Insight, you can type the name -- or even a part of a name -- for the tags you want to analyze. Then you can choose the chart type and timeframe to report on. Insight also lets you save and share your data.

For more information on using Insight, see the online help.

Querying with Transact-SQL

Historian can handle traditional SQL queries. For example:

SQL Query

```
select Value as Step,
Label as Label,
StateTimeAvgContained as Avg,
StateTimeMinContained as Fastest,
StateTimeMax as Longest
from StateSummaryHistory
where TagName = 'Filler1.Step'
and StartDateTime >= '2016-05-01'
and EndDateTime <= '2016-05-02'
and wwCycleCount = 1
and wwRetrievalMode='cyclic'
```

Result

Step	Label	Avg	Fastest	Longest
1	Load	1029	1020	1038
2	Open	1009	1006	1012
3	Fill	5710	5612	5842
4	Settle	2022	2018	2036
5	Close	4558	1074	5816
6	Seal	1067	1058	1176
7	Release	1120	994	1140

Data replication: Delivering information to people who need it

AVEVA Historian allows you to replicate data. That is, a tag that was collected by one historian can be replicated and stored on another historian.

AVEVA Historian supports two types of replication:

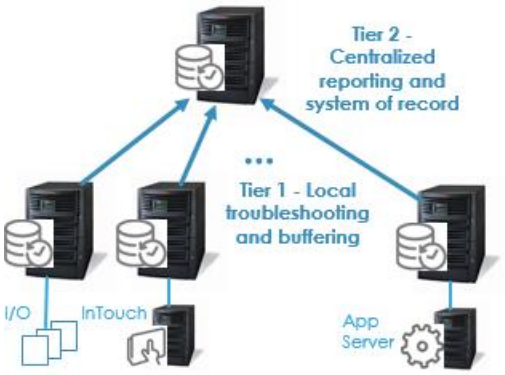
- **Simple replication**
  You can replicate tag data directly using simple replication, where the tag information is replicated directly from one historian to the other. For simple replication, every value for a tag is copied.
- **Summary replication**
  You can also set up summary tags that receive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

  Summary replication is useful to minimize bandwidth requirements. This is important, for example, when an application uses a wide-area network or connecting via satellite/cellular.

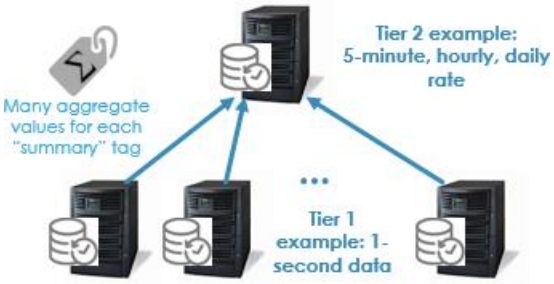
Note: Summary replication happens between tier 1 and tier 2 only. All data replication to tier 3 and beyond is simple replication.

Setting up replication servers is useful for several circumstances. For example:

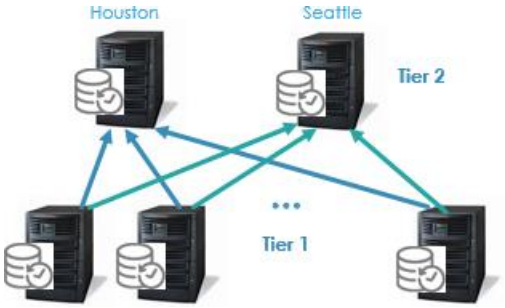
- **Centralized data**
  When you want to replicate data from multiple individual historians and sent it to a single centralized historian.



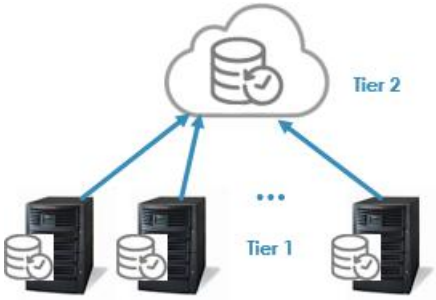
- **Replicated summaries**
  When you want summary data to be available from a centralized historian.



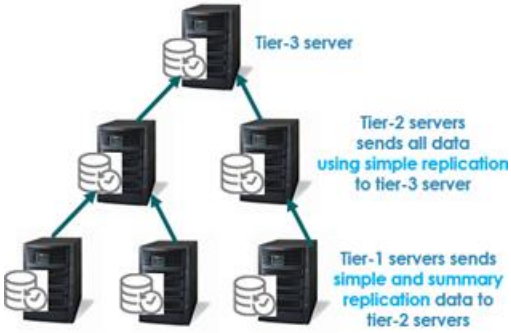
- **Many-to-many**
  When you want to set up a many-to-many relationship between tiers of historians.



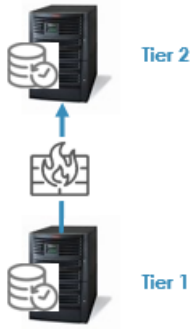
- **Cloud**
  When you want data available from the cloud.



- **Multiple levels**
  When you want to replicate data to multiple tiers of historians.



- **Across firewall**
  When your facility has a firewall and you want users on both sides of the firewall to have access to the same data.



For more information about setting up and using replication, see Managing and Configuring Replication in the *AVEVA Historian Administration Guide*.

Replication tiers

When you replicate data, it creates a tiered relationship between the historians.

That is, the tier-1 historian send its replicated data to a tier-2 historian.

AVEVA Historian can replicate process data as well as alarms and events.

A historian can act as a tier-1 and a tier-2 historian simultaneously.

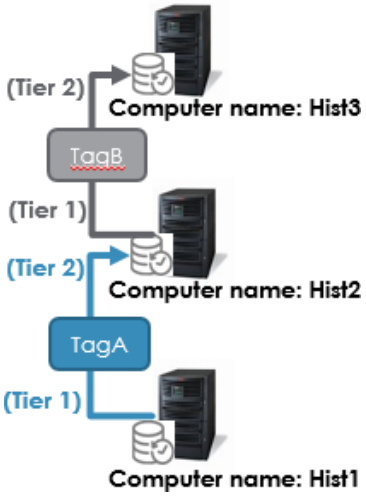
Here's a typical scenario for a tiered historian:

- TagA is collected by Hist1, a historian. It is replicated and stored on Hist2.

  In this relationship, Hist1 is a tier-1 historian and Hist2 is a tier-2 historian.

- Concurrently, TagB is collected by Hist2. It is replicated and stored on Hist3.

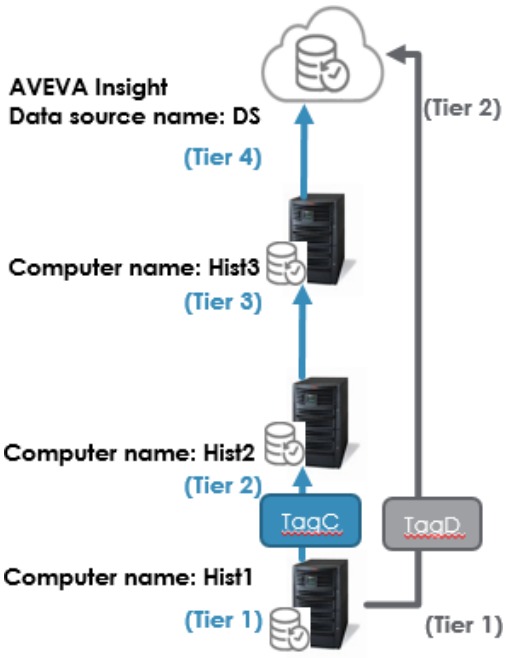
  In this relationship, Hist2 is a tier-1 historian and Hist3 is a tier-2 historian.



Historian supports multi-tier replication. Data originating at tier 1 can be replicated to tier 2, then again to tier 3, and so on.

This diagram illustrates two relationships between historians.

- TagC is replicated from Hist1, a tier-l historian, to Hist2, a tier-2 historian. Hist2 replicates the tag to Hist3, a tier-3 historian. Hist3 replicates the tag once again to AVEVA Insight (tier 4 in this scenario).
- TagD is also replicated from Hist1, but this time directly to AVEVA Insight. In this case, Hist1 is the tier-1 historian and Insight is the tier-2 historian.



The following tables show how the replicated data is named as it is replicated in these two scenarios. These examples are based on the default naming scheme.

TagC replicated across 4 tiers (Hist1, Hist2, Hist3, Insight)

Tier	Computer name	Tag name	Summary tag name
Tier 1	Hist1	TagC	TagC.1M
Tier 2	Hist2	Hist1.TagC	Hist1.TagC.1M
Tier 3	Hist3	Hist1.TagC	Hist1.TagC.1M (see note below)
Tier 4	Insight (and data source = DS)	DS.Hist1.TagC	DS.Hist1.TagC.1M

Note: Summary replication happens between tier 1 and tier 2 only. All data replication to tier 3 and beyond is simple replication.

TagD replicated from Hist1 to Insight

Tier	Computer name	Tag name	Summary tag name
Tier 1	Hist1	TagD	TagD.1M
Tier 2	Insight (and data source = DS)	DS.TagD	DS.TagD.1M

Note: Before version 17.3.100, replication to AVEVA Insight used the same default naming as any other tier 2 and still included the "DS" prefix (where "DS" is the name of the data source). For example, consider how "TagC" was replicated to Insight before and since version 17.3.100:

- Before 17.3.100: TagC was replicate to Insight as "DS.Hist1.TagC".
- Since 17.3.100: TagC is replicated to Insight as "DS.TagC".