AVEVA™
Historian Database Reference
formerly Wonderware

Publication date: Thursday, December 3, 2020

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

This guide describes the database model of the AVEVA Historian system. Each database entity is described, and the relationships between the entities are defined. It is very important that you understand these data structures and relationships to effectively query AVEVA Historian and build productive client applications that interact with it.

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.

CHAPTER 1

Table Categories

There are eight table categories within the AVEVA Historian Runtime database. Tables in a category together facilitate a particular functionality in the historian.

Note: Additional tables and views are provided for backward compatibility support. For more information, see *Backward Compatibility Entities* on page 293.

History Tables and Views

Because normal Microsoft SQL Server functionality cannot handle the storage and retrieval of huge quantities of rapidly changing data, plant data storage and retrieval are made possible by the AVEVA Historian storage subsystem, the history tables, and the retrieval system.

Some of the history tables are implemented as normal SQL Server tables, and the information contained in them is stored in the Runtime database file (Run110Dat.mdf). Others are implemented as a special type of table called a remote table, or extension table. Extension tables do not actually exist in the database, but rather expose data that is stored in special history files (history blocks) on disk using OLE DB technology.

For more information, see the About Data Retrieval in the *AVEVA Historian Retrieval Guide.*

Acquired tag data can be presented in the history tables in four different formats:

- Normal historical format
- "Wide" format
- "Live" format
- Analog/state summary history format

Information about the history blocks is stored in the special *HistoryBlock (INSQL.Runtime.dbo.HistoryBlock)* on page 84 extension table.

AVEVA Historian also includes several views to make querying from the history tables easier. Instead of specifying the table name using the required four-part syntax (INSQL.Runtime.dbo.<tablename>), you can simply use the view name instead. The history tables and associated views are listed in the following table. (Backward compatibility tables and views are not included.)

Table Name (OLE DB Provider Syntax)	Associated View
History (INSQL.Runtime.dbo.History) on page 78	History
WideHistory (INSQL.Runtime.dbo.WideHistory) on page 138	(none)
StateWideHistory (INSQL.Runtime.dbo.StateWideHistory) on page 116	(none)
Live (INSQL.Runtime.dbo.Live) on page 92	Live

Table Name (OLE DB Provider Syntax)	Associated View
HistoryBlock (INSQL.Runtime.dbo.HistoryBlock) on page 84	HistoryBlock
AnalogSummaryHistory (INSQL.Runtime.dbo.AnalogSummaryHistory)	AnalogSummaryHistory
StateSummaryHistory (INSQL.Runtime.dbo.StateSummaryHistory) on page 113	StateSummaryHistory

The History and Live tables can accommodate a mixture of tag types and should be used for all queries. The vValue column returns a sql_variant for all tag types. The Value column returns a float value for analog and discrete tags and a NULL for string tags. The Value column is included to allow for aggregation and other operations that are not permitted on a sql_variant column.

You can relate these tables to other tables in the AVEVA Historian database.

For more information on each of these tables, see the corresponding table description in this documentation.

Note: The AnalogHistory, DiscreteHistory, StringHistory, AnalogLive, DiscreteLive, and StringLive tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you cannot mix the tag types in the query without doing a UNION.

In SQL Server Management Studio, the extension tables are listed under the INSQL or INSQLD linked servers under the **Server objects** tree item.

History Table Format

The History table presents acquired plant data in a historical format, which is shown as follows:

DateTime	TagName	Value	vValue	Quality	QualityDetail	(continued...)
2017-02-17 15:40:01.0000000	Temp1	78	78	0	192	(continued...)
2017-02-17 15:40:31.0000000	Temp2	79	79	0	192	(continued...)
2017-02-17 15:41:01.0000000	Temp3	77	77	0	192	(continued...)
2017-02-17 15:41:31.0000000	Temp4	80	80	0	192	(continued...)
2017-02-17 15:42:01.0000000	Temp1	77	77	0	192	(continued...)
2017-02-17 15:42:31.0000000	Temp2	78	78	0	192	(continued...)
2017-02-17 15:43:01.0000000	Temp3	76	76	0	192	(continued...)
2017-02-17 15:43:31.0000000	Temp4	79	79	0	192	(continued...)
2017-02-17 15:44:01.0000000	Temp1	76	76	0	192	(continued...)
2017-02-17 15:44:31.0000000	Temp2	77	77	0	192	(continued...)
2017-02-17 15:45:01.0000000	Temp3	78	78	0	192	(continued...)
2017-02-17 15:45:31.0000000	Temp4	80	80	0	192	(continued...)

There is one row for a single tag's value for a particular timestamp.

Note: The AnalogHistory, DiscreteHistory, and StringHistory tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you cannot mix the tag types in the query without doing a UNION. The History table, however, can accommodate a mixture of tag types and should be used instead of the AnalogHistory, DiscreteHistory, or StringHistory tables. The Value column returns a float value for analog and discrete tags, a NULL for string tags. The vValue column returns a sql_variant for all tag types.

"Wide" History Table Format

The WideHistory table contains the same data as the History table, but in a different format. The WideHistory table presents data for one or more tag values for a single timestamp, thus providing a "wide" view of the data. To query for values in the WideHistory table, you must specify the timestamp and one or more tagnames as the column names in the query syntax. The results will contain a column for the timestamp and columns for the value of each specified tag at that timestamp. In the following example, Temp1, Temp2, Temp3, and Temp4 are tagnames:

DateTime	Temp1	Temp2	Temp3	Temp4
02:17:01:03	78	79	77	80
02:17:01:04	77	78	76	79
02:17:01:05	77	78	76	79

Using the History table to perform the same task is much more difficult.

You can also specify search criteria for the values you want to return (for example, where Temp1 > 75). The WideHistory table can only be related to other tables based on the timestamp.

Note: The AnalogWideHistory, DiscreteWideHistory, and StringWideHistory tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you can't mix the tag types in the query. The WideHistory table, however, can accommodate a mixture of tag types and should be used instead of the AnalogWideHistory, DiscreteWideHistory, or StringWideHistory tables.

The WideHistory table column type returns a SQL Server type float for analog, a SQL Server type int for discrete tags, and an nvarchar(512) for string tags. The schema of the definition table, WideHistory_OLEDB, indicates a sql_variant type. This is simply a shorthand notation; it does not represent the type actually returned.

There is no Quality column for the WideHistory table because there is more than one tag value for each row returned. However, a value returned for a specified tag will be set to NULL if the quality of the value is invalid, inhibited, or unavailable.

The following restrictions apply when performing a query against the WideHistory table:

- Column names must be specified.
- The table is only accessible using the OPENQUERY statement.

Because tagnames are used for column names, the tagname can include any characters as defined by the rules for Microsoft SQL Server identifiers. An identifier that does not comply with the rules for the format of regular identifiers must always be delimited using brackets ([ ]). For more information on identifiers and delimiters, see your Microsoft SQL Server documentation.

If you include an illegal column name in your query and do not use delimiters, no data will be returned.

The StateWideHistory table is similar to the WideHistory table, except that it allows for retrieval of calculated "time in state" values for multiple tags, instead of actual history values. This table includes a vValue column, and the tag columns contain the time in state for the corresponding value. For more information on this table, see *StateWideHistory (INSQL.Runtime.dbo.StateWideHistory)* on page 116. For information on how to query this table, see -OLD-ValueState Retrieval in the *AVEVA Historian Concepts Guide.*

"Live" Table Format

The Live table presents the current value of the specified tag(s).

Note: In certain situations, data can bypass the Live table. These situations include:
- Receiving non-streamed original data (store/forward or CSV);
- Receiving revision data for a Latest value;
- Receiving no new streamed values after Historian was shut down and disabled, or after the computer was rebooted.

The format of the Live table is as follows. The DateTime column will indicate the time the value was received.

DateTime	TagName	Value	vValue	Quality	QualityDetail	(continued...)
02:17:01:05	Temp1	77	77	0	192	(continued...)
02:17:01:05	Temp2	78	78	0	192	(continued...)
02:17:01:05	Temp3	76	76	0	192	(continued...)
02:17:01:05	Temp4	79	79	0	192	(continued...)

Note: The AnalogLive, DiscreteLive, and StringLive tables are provided for backward compatibility and can only accept tagnames in the SELECT statement that are of the same type; that is, you can't mix the tag types in the query. The Live table, however, can accommodate a mixture of tag types and should be used instead of the AnalogLive, DiscreteLive, or StringLive tables.

Auto-Summary Replication Tables

AVEVA Historian performs two types of replication -- *Standard Replication Tables* on page 22 and auto-summary replication.

With auto-summary replication, Historian automatically computes and records an hourly summary as its corresponding real-time data is acquired. This allows Historian to quickly and efficiently retrieve large-volume data for a long duration, even months or years.

The Auto-Summary Replication tables are:

- *_AutoReplicationGroup* on page 25 *
- *_AutoReplicationRule* on page 26 *
- *_AutoReplicationTagEntity* on page 29 *
- *_AutoReplicationServer* on page 27 *
- *_AutoTag* on page 31 *
- *_AutoTagHistory* on page 36 *

* System-level table. Do not edit.

Note: The auto-summary feature was available beginning with AVEVA Historian 2017. From the time you installed or upgraded to AVEVA Historian 2017, the system has been creating auto-summary values for your analog tags. To backfill values for time before that instillation or upgrade, you can use the Replication Backfill Manager. For more information, see Adding Auto-Summary Values for a Defined Timeframe in the *AVEVA Historian Administration Guide*.

Tag Definition Tables

Types of tags that can be defined in the AVEVA Historian are analog, discrete, event, and string. The Tag Definition tables describe the qualities of the tags in your system.

The Tag Definition tables are:

- *_Tag* on page 49
- *ChannelStatus* on page 63
- *CurrentEditor* on page 69
- *EngineeringUnit* on page 73
- *Message* on page 96
- *TagHistory* on page 128
- *TagType* on page 135

Note: The _Tag table replace the previous Tag table. A new view named Tag now provide backward-compatibly.

Tag Definition Specialty Tables

The Tag Definition Specialty tables contain information about tag definitions that are imported into the Runtime database from the InTouch tagname.x Importer.

The Tag Definition Specialty tables are:

- *Context* on page 69
- *Deviation* on page 72
- *Limit* on page 90
- *LimitName* on page 91
- *RateOfChange* on page 101

I/O Data Acquisition Tables

I/O Data Acquisition tables contain information about tag definitions that are imported into the Runtime database from an IDAS.

The I/O Data Acquisition tables are:

- *_IODriver* on page 43
- *_IOServer* on page 47
- *_Topic* on page 54
- *IOServerType* on page 90

Note: The _IODriver, _IOServer, and _Topic tables replace the previous IODriver, IOServer, and Topic tables. New view named IODriver, IOServer, and Topic now provide backward-compatibly.

Insight Client Content Tables

Insight Client Content tables contain information needed to create content in the AVEVA Insight or Historian Insight tool.

The Insight Client Content tables are:

- *ChartConfiguration* on page 64
- *ChartConfigurationAuditLog* on page 65
- *ChartConfigurationKeyword* on page 65
- *ChartConfigurationProperty* on page 65
- *ChartConfigurationStatistics* on page 66
- *ChartConfigurationTag* on page 66
- *DashboardConfiguration* on page 70

InTouch Node Detail Tables

InTouch Node Detail tables contain information about tag definitions that are imported into the Runtime database from an InTouch application.

The InTouch Node Detail tables are:

- *InTouchNode* on page 88
- *InTouchSpecific* on page 89
- *TopicImportInfo* on page 137

Legacy Event and Summary Tables

Legacy Event and Summary tables contain definitions for events, including tags associated with events, detectors for events, and actions for events. The event system tables can also store "snapshots" of tag values at the time of an event, as well as details about the event itself.

A special type of event action is a summarization of tag values. A subset of the event tables provide the supporting framework for fully automated summary generation for analog, discrete and string tags.

Legacy Event and Summary tables are:

- *_EventTag* on page 41
- *ActionType*
- *AnalogSnapshot*
- *CalcType* on page 63
- *DetectorType* on page 71
- *DiscreteSnapshot* on page 72
- *EventHistory* on page 76
- *EventTagPendingDelete* on page 77*
- *Frequency* on page 77
- *SnapshotTag* on page 112
- *StringSnapshot* on page 122
- *SummaryData* on page 123

- *SummaryHistory* on page 124
- *SummaryOperation* on page 125
- *SummaryTagList* on page 126
- *TimeDetectorDetail* on page 135
- *TimeDetectorDetailPendingDelete* on page 136*

* System-level table. Do not edit.

Note: The _EventTag table replaces the previous EventTag table. New views named EventTag now provide backward-compatibly.

Manual Data Tables

Manual Data tables contain information about tag that were created manually.

The Manual Data tables are:

- *ManualAnalogHistory* on page 310
- *ManualDiscreteHistory* on page 310
- *ManualStringHistory* on page 311

Modification Tracking Tables

The modification tracking tables contain information about changes that are made to columns in the database.

The modification tracking tables are:

- *ModLogTable* on page 97
- *ModLogColumn* on page 96
- *UserDetail* on page 138

Namespace and User-SpecificTables

The namespaces and grouping tables contain information that defines how sets of tags can be grouped together for alarming, displays, event management, and batch management. These tables also define hierarchies for items in the system, public, or private namespaces.

The namespaces and grouping tables are:

- *Annotation*
- *GroupTagList* on page 309
- *PrivateGroupTag* on page 99
- *PrivateNameSpace* on page 99
- *PublicGroupTag* on page 100
- *PublicNameSpace* on page 100
- *ServerList* on page 110
- *TagRef* on page 134
- *UserDetail* on page 138

- *WideTableDictionary* on page 315

Standard Replication Tables

AVEVA Historian performs two types of replication -- standard replication and *auto-summary replication* (see "*Auto-Summary Replication Tables*" on page 18).

With standard replication, tag information can be replicated from source, or tier 1, servers to replication, or tier 2, servers. Standard replication lets you consolidate and summarize information from separate servers to a single replication server so you can then perform analyses and run reports from the replication server on the consolidated data. You can summarize tags to capture analog or state values. You can also do a simple replication, which copies tag information directly without summarizing it. For more information, see Managing and Configuring Replication in the *AVEVA Historian Administration Guide.*

The Standard Replication tables are:

- *CustomReplicationSchedule* on page 69
- *IntervalReplicationSchedule* on page 88
- *ReplicationGroup* on page 102
- *ReplicationRule* on page 103
- *ReplicationScheduleType* on page 104
- *ReplicationSchedule* on page 104
- *ReplicationSyncRequest* on page 107
- *ReplicationTagEntity* on page 108
- *ReplicationType* on page 109

Storage Tables

Storage tables describe the storage partitions (shards) used by AVEVA Historian to house your data.

The Storage tables are:

- *AttributeType* on page 62
- *ReplicationShard* on page 107
- *ShardAssignmentRule* on page 111
- *StorageLocation* on page 120
- *StorageShard* on page 121
- *StructureAttributes* on page 123
- *StructureType* on page 123

System Configuration Tables

All AVEVA Historian parameters are stored in system configuration tables. Parameters include information regarding the historian's physical nodes, site-specific configuration parameters, and parameters pertaining to the physical I/O equipment to which the system is connected.

The system configuration tables are:

- *_IODriver* on page 43
- *_IOServer* on page 47

- *_Tag* on page 49
- *_Topic* on page 54
- *ConfigStatusPending* on page 68 *
- *ConfigStatusSnapshot* on page 68*
- *ErrorLog* on page 76
- *IOServerType* on page 90
- *LocalizedText* on page 94*
- *OPCQualityMap* on page 98
- *SnapshotDetail* on page 112
- *StorageLocation* on page 120
- *StorageShard* on page 121
- *SystemParameter* on page 127
- *TimeZone* on page 136*
- *UserDetail* on page 138

* System-level table. Do not edit.

Note: The _IODriver, _IOServer, _Tag, and _Topic tables replace the previous IODriver, IOServer, Tag, and Topic tables. New views named IODriver, IOServer, Tag, and Topic now provide backward-compatibly.

ArchestrA Browsing Tables

The ArchestrA browsing tables store information required to support the browsing of the ArchestrA model view hierarchy by AVEVA Historian clients.

The ArchestrA browsing tables are:

- *aaAreaData* on page 55 *
- *aaAttributeData* *
- *aaAreaXML* *
- *aaObjectDataPending* *
- *aaObjectData* *

* These tables are for internal use only.

CHAPTER 2

Tables

All information regarding how the system is configured is stored in tables in the Runtime database. Event history, summary history, and summary data are also stored in SQL Server tables. You can view the details of all tables by using the Microsoft SQL Server Management Studio.

_AutoIntervalReplicationSchedule

Contains one row for each autosummarization schedule interval.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule.
Period	small int NOT NULL	The number of units that make up the interval.
Unit	nvarchar(32) NOT NULL	The unit of measure for this interval.

_AutoReplicationGroup

Contains one row for each tag group that is summarized for your system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) ReplicationGroupKey	int NOT NULL	The unique identifier for the replication group.
ReplicationGroupName	nvarchar(255) NOT NULL	The name of the replication group.
(PK, FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the ReplicationServer table.

Column	Data Type	Description
(FK) ReplicationTypeKey	tinyint NOT NULL	Can be 1, 2, or 3. (1 = Simple Replication, 2 = Analog Summary Replication, 3 = State Summary Replication.) ReplicationTypeKey is a foreign key from the ReplicationType table.
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the replication schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
SummaryReplicationNamingScheme	nvarchar(512) NULL	The naming scheme for the replication tags belonging to this replication group. If the summary replication naming scheme is NULL, the summary replication naming scheme from the replication server is used as the default naming scheme for summary tags.
GroupAbbreviation	nvarchar(32) NULL	The abbreviation for the replication group. If GroupAbbreviation is NULL, ScheduleAbbreviation is used as the default group abbreviation.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoReplicationRule

Contains one row for each rule that applies to autosummarization for your system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
Name	nvarchar(255), NOT NULL	The name of the replication rule.
Priority	int, NOT NULL	The priority for the rule.
TagFilter	nvarchar(4000), NOT NULL	Do not edit. This shows the OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
(FK) ReplicationGroupKey	int, NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.

Column	Data Type	Description
(FK) ReplicationServerKey	int, NOT NULL	The unique identifier for the replication server.
Enabled	bit, NOT NULL	Used to indicate whether the replication rule is enabled. 0 - not enabled; 1- enabled
ApplyOtherRules	bit, NOT NULL	Used to indicate whether other rules apply. 0 - other rules do not apply; 1- other rules appy.
Id	int, NOT NULL	The unique identifier for the object.
ChangeVersion	timestamp, NOT NULL	Internal use only.

_AutoReplicationSchedule

Contains one row for each autosummarization schedule.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule.
ReplicationScheduleName	nvarchar(255) NOT NULL	The name of the replication schedule.
(FK) ReplicationScheduleTypeKey	int NOT NULL	The type of autosummarization schedule. ReplicationScheduleType is a foreign key from the ReplicationScheduleType table.
ReplicationScheduleAbbreviation	nvarchar(32) NOT NULL	The abbreviation for the autosummarization schedule.
CreateGroup	bit NOT NULL	If TRUE, this autosummarization schedule is automatically added to new autosummarization groups.

_AutoReplicationServer

Contains one row for each replication server used for autosummarization.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server.
ReplicationServerName	nvarchar(255) NOT NULL	The name of the replication server.
Description	nvarchar(512) NULL	The description of the replication server.
SFPath	nvarchar(260) NULL	The local store-and-forward path associated with the replication server for this instance of AVEVA Historian.
SFFreeSpace	int NOT NULL	The free space for the store-and-forward path in MB.
CompressionEnabled	bit NULL	Used to specify whether compression should be enabled for the tag. 0 = No compression; 1= Compression.
UserName	nvarchar(255) NULL	The user name for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
Password	nvarchar(512) NULL	The encrypted password for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
TCPPort	int NOT NULL	The TCP port to use to log in to the replication server.
SummaryReplicationNamingScheme	nvarchar(512) NULL	The naming rule for summary replication tags. If ReplicationGroupKey is NULL, the naming rule is used from the ReplicationServerName scheme. If ReplicationServerName is NULL, the naming rule is used from the SummaryReplicationNamingScheme system parameter.
SimpleReplicationNamingScheme	nvarchar(512) NULL	Naming rule for simple replication tags. If NULL the naming rule specified in the simple replication naming scheme system parameter is used.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
Bandwidth	int NOT NULL	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited.

Column	Data Type	Description
MinSFDuration	int NOT NULL	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
ConnectionDetails	nvarchar(4000) NULL	Internal use only.
IntegratedSecurity	bit, NULL	Indicates whether this will be used for local replication connection and not for remote. (For remote replication, users are expected to provide username and password.)
ReplicationEvents	bit, NOT NULL	Specifies whether events are to be replicated.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoReplicationTagEntity

Contains one row for each tag replicated by the system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
(PK) (FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the Replication Server table.
(PK) DestinationTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
DestinationTagID	uniqueidentifier NOT NULL	The unique identifier for the destination tag.

Column	Data Type	Description
(FK) SourceTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the source tag. SourceTagName is a foreign key from the Tag table.
(FK) ReplicationGroupKey	int NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
MaximumStates	tinyint NOT NULL	Maximum number of states to track for state summary tags. Discrete summary tags have a limit of 3 states. Analog summary tags of a limit of 100 states. The default is 10 states.
(FK) CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
ReplicationTagEntityId	uniqueidentifier, NOT NULL	This plays a role in SyncQueue to identify the entity.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoTag

Contains one row for each tag defined in the system.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
TagId	uniqueidentifier, NOT NULL	The unique identifier for the tag.
(PK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
(FK) IOServerKey	int, NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
(FK) TopicKey	int, NULL	The unique numerical identifier for the topic. TopicKey is a foreign key from the Topic table.
Description	nvarchar(512), NULL	The description of the tag.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver.
StorageType	smallint, NOTNULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored."
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds.
ItemName	nvarchar(256), NULL	The address string of the tag.
(FK) TagType	int, NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table.

Column	Data Type	Description
DeadbandType	smallint, NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband.
TimeDeadband	int, NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.
ServerTimeStamp	bit, NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage.
(FK) ChannelStatus	tinyint, NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table.
(FK) MessageKey	int, NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.
(FK) EUKey	int, NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float, NULL	The minimum value of the tag, measured in engineering units.

Column	Data Type	Description
MaxEU	float, NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float, NULL	The minimum value of the raw acquired value.
MaxRaw	float, NULL	The maximum value of the raw acquired value.
Scaling	int, NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use).
RawType	int, NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float, NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
IntegerSize	tinyint, NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit, NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.
RateDeadband	float, NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.

Column	Data Type	Description
InterpolationType	tinyint, NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.
RolloverValue	float, NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.
MaxLength	smallint, NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint, NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0.
(FK) StructureId	uniqueidentifer, NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.
SourceTag	nvarchar(256), NULL	The source (tier 1) tag for the summary tag
SourceServer	nvarchar(255), NULL	The source (tier 1) server for the summary tag.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.

Column	Data Type	Description
(FK) CurrentEditor	tinyint, NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian. wwTagKey is populated from the Tag table, but is not a foreign key.
AIHistory	bit, NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage.
DateCreated	datetime2(7), NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated. Internal use only.
CreatedBy	nvarchar(255), NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated. Internal use only.
ChangeVersion	timestamp, NOT NULL	Internal use only.
CEVersion	tinyint, NOT NULL	The version number used to track changes to the information in the Tag table. Any change to the data in a row will cause the version indicator to change. The Configuration Editor (and other client tools) can detect the changed version and reload the corresponding tag details. Changes to this column are not tracked by the modification tracking system.

Column	Data Type	Description
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_AutoTagHistory

Contains one row for each tag metadata instance uniquely identified by the TagId column.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) TagId	uniqueidentifier, NOT NULL	The unique identifier for the tag.
TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system. Internal use only.
Description	nvarchar(512), NULL	The description of the tag. Internal use only.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver. Internal use only.
StorageType	smallint, NOTNULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." Internal use only.
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. Internal use only.
TagType	int NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table. Internal use only.
TimeDeadband	int NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. Internal use only.

Column	Data Type	Description
DateCreated	datetime2(7) NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated. Internal use only.
CreatedBy	nvarchar(256) NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated. Internal use only.
CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server. CurrentEditor is a foreign key from CurrentEditor table. Internal use only.
ServerTimeStamp	bit NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. Internal use only.
DeadbandType	smallint NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. Internal use only.

Column	Data Type	Description
ChannelStatus	tinyint NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table. Internal use only.
AIHistory	bit NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage. Internal use only.
Message0	nvarchar(64) NULL	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. Internal use only.
Message1	nvarchar(64) NULL	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. Internal use only.
Unit	nvarchar(32) NULL	The unit of measure. Examples are mph, grams, and pounds. Internal use only.
DefaultTagRate	init NULL	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag. Internal use only.

Column	Data Type	Description
IntegralDivisor	float NULL	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only.
MinEU	float NULL	The minimum value of the tag, measured in engineering units. Internal use only.
MaxEU	float NULL	The maximum value of the tag, measured in engineering units.Internal use only.
MinRaw	float NULL	The minimum value of the raw acquired value. Internal use only.
MaxRaw	float NULL	The maximum value of the raw acquired value. Internal use only.
Scaling	int NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use). Internal use only.
RawType	int NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes). Internal use only.
ValueDeadband	float NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. Internal use only.
IntegerSize	tinyint NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use). Internal use only.

Column	Data Type	Description
SignedInteger	bit NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed. Internal use only.
RateDeadband	float NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. Internal use only.
InterpolationType	tinyint NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. Internal use only.
RolloverValue	float NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000. Internal use only.
MaxLength	smallint NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512. Internal use only.
DoubleByte	tinyint NULL	Used to specify whether or not to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0. Internal use only.
StructureId	uniqueidentifier NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table. Internal use only.
SourceTag	nvarchar(256) NULL	The name of the source tag used for the replication tag. Internal use only.
SourceServer	nvarchar(255) NULL	The name of the tier 1 server with the source tag. Internal use only.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.

Column	Data Type	Description
ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).

_DeletedReplicationTagExtendedProperty

Contains one row for each deleted replication tag extended property.

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(FK) ReplicationTagExtended PropertyKey	int, NOT NULL	A unique identifier for the replication tag extended property.
ChangeVersion	timestamp, NOT NULL	Internal use only.

_EventTag

Contains one row for each event definition. Configuration information specific to event tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) DetectorTypeKey	int, NULL	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6 = Time-based (schedule). DetectorTypeKey is a foreign key from the DetectorType table.
DetectorString	nvarchar(1500), NULL	The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian.

Column	Data Type	Description
(FK) ActionTypeKey	int, NULL	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 = Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary. ActionTypeKey is a foreign key from the ActionType table.
ActionString	nvarchar(1500), NULL	The script that specifies the event action. Action scripts run on the local AVEVA Historian.
UseThreadPool	bit, NOT NULL	Used to specify how system threads are used to process events. 1 = All events are handled by a single thread and a single logon to the SQL Server; 0 = Each event uses a separate system thread and logon. This will allow the event subsystem to manage the scan rates of each detector component concurrently. (Reserved for future use.)
ScanRate	int, NULL	The interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).
Logged	bit, NOT NULL	Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured. 0 = Not logged; 1 = Logged. The default is 1.
PostDetectorDelay	int, NOT NULL	The amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.

Column	Data Type	Description
Priority	tinyint, NOT NULL	The priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. 0 = Normal; 1 = Critical. The default is 0.
Edge	tinyint, NOT NULL	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector.
Status	tinyint, NOT NULL	The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.

_IODriver

Contains one row for each IDAS providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IODriverKey	int NOT NULL	The unique identifier for an IDAS. This value is automatically generated by the system when the IDAS is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
ComputerName	nvarchar(255) NOT NULL	The name of the computer on which the IDAS runs.

Column	Data Type	Description
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, redundant IDAS runs. You must use the fully qualified name of the computer. You could also use the IP address. This should be set to an empty string if no redundant IDAS is specified. Make sure that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online. By default, this column is an empty string.
StoreForwardMode	tinyint NOT NULL	Used to specify whether or not store-and-forward capability is enabled. If enabled, and the network connection between the IDAS and the storage node fails, data will be "buffered" to the location specified by the store-and-forward path. Valid values are: 0 = Disabled; 1 = Enabled; 2 = Autonomous. The Autonomous mode (2) is an extension of the normal store-and-forward mode (1). It allows the IDAS to start up using an IDAS configuration file and collect data in store-and-forward mode if the network connection to the AVEVA Historian is not available.

Column	Data Type	Description
StoreForwardPath	nvarchar(255) NULL	Used to specify the path for the IDAS data buffer on the local hard drive of the IDAS computer. The path should be absolute (for example, c:\IDASBuffer). Data is written to this path until the minimum threshold for the buffer is reached. Remote buffer paths are not supported. When the store-and-forward path specified for the IDAS is invalid, the default path picked by the system is: <public folder>\ArchestrA\Historian\IDAS\SF where the <public folder> is dependent on the operating system. For example, for the Windows 2008 operating system, the path is C:\ProgramData\ArchestrA\Historian\IDAS\SF. When the store-and-forward path specified for the IDAS is just a folder name (without any path characters like \ and :), the default path picked by the system is: <Windows system path>\<folder name specified by the user>. For example, for the Windows Server 2008 32-bit operating system, the path is C:\WINDOWS\system32\<folder name>.
MinMBThreshold	int NOT NULL	The minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer.
Status	tinyint NULL	Automatically updated by the system if a change is made to IDAS: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
Enabled	bit NOT NULL	Used to indicate whether the IDAS is enabled or not. 0 = Not enabled; 1 = enabled. Disabling the IDAS allows for the configuration to be retained in the database, even though the IDAS is removed from the system.
StoreForwardDuration	int NOT NULL	The minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.

Column	Data Type	Description
AutonomousStartupTimeout	int NOT NULL	The amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates.
FileChunkSize	int NOT NULL	The size, in bytes, of the data "chunks" that are sent to the historian when store-and-forward data is forwarded. The size of the chunks can be decreased to accommodate slower networks. Decrease this number only if the forwarding delay is greater than zero.
ForwardingDelay	int NOT NULL	The time interval, in milliseconds, at which "chunks" of store-and-forward data are forwarded to the historian. The length of the interval may need to be increased to accommodate slower networks.
ConnectionTimeout	int NOT NULL	The amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks.
CompressionEnabled	bit NOT NULL	Used for HCAL connections, this specifies whether compression is enabled.
TCPPort	int NOT NULL	Used for HCAL connections, this identifies the TCP port on remote IDAS server where HCAP is listening. Default is 32568
IntegratedSecurity	bit NOT NULL	Specifies whether to use Integrated security for communication. Applies to Domain environment only. In case of remote IDAS, the IDAS system will need to trust the server.
UserName	nvchar(255) NULL	Identifies the login username for the domain and workgroup. This is used for connecting to remote IDAS to push configuration.

Column	Data Type	Description
Password	nvchar(512) NULL	Identifies the login password for the domain and workgroup. This is used for connecting to remote IDAS to push configuration.
ConnectionDetails	nvchar(4000) NULL	Specifies the encrypted connection token for allowing the remote IDAS to connect to Historian. Generated by the system.
Classic	bit NOT NULL	Specifies whether this is used for classic IDAS. Used during migration. Once the system detects the new IDAS, this is set as false.
ChangeVersion	timestamp NOT NULL	For internal use only.

_IOServer

Contains one row for each I/O Server providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IOServerKey	int NOT NULL	The unique numerical identifier for the I/O Server. This value is automatically generated by the system when the I/O Server is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
(FK) IODriverKey	int NULL	The unique identifier for an IDAS. IODriverKey is a foreign key from the IODriver table.
(FK) ApplicationName	nvarchar(32) NULL	The application name of the I/O Server. This name is usually the same as the executable file name. ApplicationName is a foreign key from the IOServerType table.
Path	nvarchar(255) NULL	The full UNC path (including the filename) to locate the executable file for the I/O Server. If the I/O Server type key is specified, the filename may be omitted.
ComputerName	nvarchar(255) NULL	The name of the computer on which the I/O Server runs.
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made.
AutoStart	bit NOT NULL	Used to control how the I/O Server starts up. 0 = Automatic startup when the system starts. 1 = Manual startup required. Currently not used.

Column	Data Type	Description
ExeType	int NOT NULL	The type of executable for the I/O Server. Used by the Historian System Management Console to determine how to start the I/O Server. 0 = Service; 1 = Console application; 2 = Windows application.
InitializationStatus	tinyint NOT NULL	A control flag used to ensure that each I/O Server has been asked for the data type (integer or real) of each tag that it will send. Only needed after a database modification.
ProtocolType	tinyint NOT NULL	The protocol used by the AVEVA Historian server to communicate with the I/O Server. 1 = DDE; 2 = SuiteLink™; 3 = AVEVA Historian named pipe driver (for compatibility with IndustrialSQL Server 3.0 and previous versions). Of the operating systems currently supported by the AVEVA Historian, DDE is only supported on the Windows XP operating system.
Description	nvarchar(50) NULL	The description of the I/O Server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_ReplicationTagExtendedProperty

Contains one entry for each property for a replicated tag.

Column	Data Type	Description
ReplicationTag ExtendedProperty Key	int NOT NULL	A unique identifier for the replication tag extended property.
(FK) ReplicationServer Key	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the Replication Server table.
(FK) DestinationTag Name	nvarchar (256) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
(FK) PropertyNameKey	int NULL	A unique identifier for the extended property name. PropertyNameKey is a foreign key from the TagExtendedPropertyName table.

Column	Data Type	Description
PropertyValue	sql_variant NOT NULL	The value of this replication tag extended property.
ChangeVersion	timestamp NOT NULL	Internal use only.
Status	tinyint NOT NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_Tag

Contains one row for each tag in the system and includes the basic definition for the tag, such as the I/O Server that supplies the values.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
TagId	uniqueidentifier NOT NULL	The unique identifier for the tag.
(PK) TagName	TagNameType (nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
(FK) IOServerKey	int, NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
(FK) TopicKey	int, NULL	The unique numerical identifier for the topic. TopicKey is a foreign key from the Topic table.
Description	nvarchar(512), NULL	The description of the tag.
AcquisitionType	tinyint, NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver.

Column	Data Type	Description
StorageType	smallint, NOT NULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored."
StorageRate	int, NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds.
ItemName	nvarchar(256), NULL	The address string of the tag.
(FK) TagType	int, NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table.
DeadbandType	smallint, NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband.
TimeDeadband	int, NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.
ServerTimeStamp	bit, NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage.
(FK) ChannelStatus	tinyint, NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of

Column	Data Type	Description
		NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table.
(FK) MessageKey	int, NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.
(FK) EUKey	int, NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float, NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float, NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float, NULL	The minimum value of the raw acquired value.
MaxRaw	float, NULL	The maximum value of the raw acquired value.
Scaling	int, NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use).
RawType	int, NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float, NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
IntegerSize	tinyint,, NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit, NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.

Column	Data Type	Description
RateDeadband	float, NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.
InterpolationType	tinyint, NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.
RolloverValue	float, NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.
MaxLength	smallint, NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint, NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0.
(FK) StructureId	uniqueidentifer, NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.
SourceTag	nvarchar(256), NULL	The name of the source tag used for the replication tag.
SourceServer	nvarchar(255), NULL	The name of the tier 1 server with the source tag.
SourceTagId	uniqueidentifer, NULL	The unique identifier for the source tag.

Column	Data Type	Description
(FK) CurrentEditor	tinyint, NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.
AIHistory	bit, NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage.
DateCreated	datetime(2) 7, NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated.
CreatedBy	nvarchar(256), NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated.
ChangeVersion	timestamp, NOT NULL	Internal use only.
CEVersion	tinyint, NOT NULL	The version number used to track changes to the information in the Tag table. Any change to the data in a row will cause the version indicator to change. The Configuration Editor (and other client tools) can detect the changed version and reload the corresponding tag details. Changes to this column are not tracked by the modification tracking system.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

_TagExtendedProperty

Contains one entry for each property for a tag.

Column	Data Type	Description
(FK) TagName	nvarchar (256) NOT NULL	Specifies a tag name. TagName is a foreign key from the _Tag table.
(FK) PropertyNameKey	int NULL	A unique identifier for the extended property name. PropertyNameKey is a foreign key from the TagExtendedPropertyName table.
PropertyValue	sql_variant NOT NULL	The value of the tag extended property for this tag.
ChangeVersion	timestamp NOT NULL	Internal use only.

_Topic

Contains one row for each topic to be read from an I/O Server.

Column	Data Type	Description
(PK) TopicKey	int, NOT NULL	The unique numerical identifier for the topic. This value is automatically generated by the system when the topic is added.
(PK) (FK) IOServerKey	int, NOT NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
Name	nvarchar(180), NOT NULL	The name of the topic.
TimeOut	int NOT NULL	The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the topic: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
LateData	bit, NOT NULL	Used to enable acquisition of "late" data. 0 = Late data disabled; 1 = Late data enabled.

Column	Data Type	Description
IdleDuration	int, NOT NULL	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds.
ProcessingInterval	int, NOT NULL	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration.

aaAreaData

Contains one row for each area referenced by an object in the ArchestrA namespace

Column	Data Type	Description
(PK) AreaKey	int, NOTNULL	The unique identifier for the item in the Area data hierarchy.
Category	int, NOTNULL	The type of the item in the Area data hierarchy. 0 = Galaxy; 1 = WinPlatform; 3 = AppEngine; 13 = Area; 11 = DDESuiteLinkClient, OPCClient or InTouchProxy; 24 = RedundantDIObject. All other values are reserved for future use.
AreaName	nvarchar(255), NOT NULL	The name of the item in the Area data hierarchy.
ContainedName	nvarchar(255), NULL	The contained name (if relevant) of the item in the Area data hierarchy.
ParentKey	int, NOTNULL	The unique identifier for the parent item of this item. For the Galaxy item, this value is 0.

aaAreaXML

Contains a single row describing the latest Area data sent from ArchestrA.

Column	Data Type	Description
Version	bigint NULL	The version number of the latest ArchestrA Area data package.

Column	Data Type	Description
AreaXML	ntext NULL	Reserved for future use.

aaAttributeData

Contains one row for each attribute referenced by an object in the ArchestrA namespace.

Column	Data Type	Description
AttributeName	nvarchar(256) NOT NULL	The ArchestrA attribute name. This name corresponds to an AVEVA Historian tagname.
(FK) ObjectKey	int NOT NULL	ObjectKey is a foreign key from the aaObjectData table.
wwDomainTagKey	int NOT NULL	The unique numerical identifier for the ArchestrA attribute (historian tag) in a specific domain.
HierarchicalAttributeName	TagNameType (nvarachar(256) NOT NULL	The hierarchical attribute name for the tag.

aaAttributeDataPending

Contains one row for each attribute in the latest ArchestrA attribute data package.

Column	Data Type	Description
AttributeName	nvarchar(256) NOT NULL	The ArchestrA attribute name. This name corresponds to a AVEVA Historian tagname.
(FK) ObjectKey	int NOT NULL	ObjectKey is a foreign key from the aaObjectDataPending table.

aaObjectData

Contains one row for each object in the ArchestrA namespace.

Column	Data Type	Description
(PK) ObjectKey	int NOT NULL	The unique identifier for the object. This column does not have the same numeric value as ObjectKey column of the aaObjectDataPending table.
Type	int NOT NULL	The type of the object. 0 = Area; 1 = ApplicationObject (regular); 2 = Traceability object. All other values are reserved for future use.
aaTagName	TagNameType (nvarachar(256)) NULL	The ArchestrA tag name for the object.
ContainedName	nvarchar(256) NULL	The ArchestrA contained name for the object.
(FK) ParentKey	int NOT NULL	The unique identifier for the parent of this object.
Status	tinyint NOT NULL	Used to indicate whether a name change has occurred. 0 = No change; 1 = The tag name has changed; 2 = The contained name has changed. The default is 0.

aaObjectDataPending

Contains one row for each object in the latest ArchestrA object data package.

Column	Data Type	Description
(PK) ObjectKey	int NOT NULL	The unique identifier for the object. This identifier is unique only within an object data package and may be repeated in subsequent data packages.
Type	int NOT NULL	The type of the object. 0 = Area; 1 = ApplicationObject (regular); 2 = Traceability object. All other values are reserved for future use.
aaTagName	TagNameType (nvarachar(256)) NULL	The ArchestrA tag name for the object.
ContainedName	nvarchar(256) NULL	The ArchestrA contained name for the object.
(FK) ParentKey	int NOT NULL	The unique identifier for the parent of this object.

ActionType

Contains one row for each type of event action.

Column	Data Type	Description
(PK) ActionTypeKey	int NOT NULL	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 = Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary. This value is automatically generated when a new action is created.
Name	nvarchar(33) NOT NULL	The name given to the type of action.
Description	nvarchar(50) NULL	The description of the action.
EditorClassName	nvarchar(80) NULL	The name by which the component is referenced by a client application, such as the System Management Console, in order to provide a visual representation.
ActionClassName	nvarchar(80) NULL	The name by which the action component (COM object) is referenced in the system in order to perform the action.

AnalogSnapshot

Contains one row for each analog tag value that was configured to be stored when a defined event occurred. To view analog, discrete, and string snapshot values at the same time, use the v_SnapshotData view instead. For more information, see *v_SnapshotData* on page 164.

Column	Data Type	Description
(PK) (FK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. SnapshotTagKey is a foreign key from the SnapshotTag table.
(PK) (FK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. EventLogKey is a foreign key from the EventHistory table.

Column	Data Type	Description
Value	float NULL	The value of the tag at the time of the event occurrence. Measured in engineering units.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

AnalogSummaryHistory (INSQL.Runtime.dbo.AnalogSummaryHistory)

The AnalogSummaryHistory view returns results for analog summary points.

Column	Data Type	Description
TagName	nvarchar(256) NOT NULL	The name of the summary tag.
StartDateTime	datetime2(7) NOT NULL	Start time of the retrieval cycle for which this row is returned.
EndDateTime	datetime2(7) NOT NULL	End time of the retrieval cycle for which this row is returned.
SliceBy	int, Discrete, or String	Performs dynamic resolution/cycle computation by tag. Returns one Analog Summary value per tag per dynamic cycle with start and end date time.
SliceByValue	var	Specifies the filter criterion to get the summary values for SlicedBy, based on that filter value.
OPCQuality	int NULL	OPC quality. Normal OPC quality retrieval logic is applied if: • All the points found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. • If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. • For any other scenario with any mixture of GOOD and BAD points, a DOUBTFUL OPC quality (64) is returned.
PercentGood	float NULL	Time in seconds that the value was good for the retrieval cycle (pro-rated for partial cycles).

Column	Data Type	Description
First	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the first point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
FirstDateTime	datetime2(7) NULL	Timestamp associated with first value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle.
Last	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the last point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
LastDateTime	datetime2(7) NULL	Timestamp associated with last value. This might be earlier than StartDateTime if this is the initial value for the retrieval cycle.
Minimum	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the minimum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
MinDateTime	datetime2(7) NULL	Timestamp associated with Min value. NULL if Min is NULL.

Column	Data Type	Description
Maximum	float NULL	If at least one non-NULL point exists for the tag in question within the retrieval cycle, then the value returned is the maximum point stored with a time stamp within the retrieval cycle. If no points exist within the retrieval cycle, then the value returned is the current value at the cycle start time. If no non-NULL points can be found, then NULL is returned.
MaxDateTime	datetime2(7) NULL	Timestamp associated with Max value. NULL if Max is NULL.
Average	float NULL	Time weighted average value of retrieval cycle. This is calculated by using the individual summary averages. The calculation is "Sum(average * delta t) / Total time of average in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Average query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.)
StdDev	float NULL	Time weighted standard deviation value of the retrieval cycle. The value is calculated using time weighted sums (Integrals) and time weighted sums of squares (IntegralOfSquares) values, prorated for any partially contained storage cycles. For analog tags, similar StdDev values are produced for each cycle.
Integral	float NULL	Area under value curve of retrieval cycle. The calculation is "Sum(value * delta t) / Total time of integral in all cycles" - delta t is prorated for any partially contained storage cycles For analog tags, the calculation is "Sum(value * delta t) / Total time. (This is like the values returned by an Integral query against the History table for a cycle of the same length, where the History row DateTime is the same as the EndDateTime here.) For analog tags, similar Integral values are produced for each cycle.
ValueCount	int NULL	Number of values in a particular cycle.
SourceTag	nvarchar(256) null	The source (tier 1) tag for the summary tag.

Column	Data Type	Description
SourceServer	nvarchar(256) null	The source (tier 1) server for the summary tag.
wwCycleCount	int NULL	The number of cycles into which the entire query time range has been divided.
wwResolution	int NULL	Length of cycles in milliseconds. The default is 3600000 (equal to 1 hour).
wwTimeZone	nvarchar(50) NULL	Time zone to use for interpreting both input and output timestamp parameters. If none is specified, then the default is set to LOCAL.
wwVersion	nvarchar(30) NULL	Data version, ORIGINAL or LATEST. If none is specified, the default is LATEST.
wwTagKey	int NOT NULL	Tag key.
wwRetrievalMode	nvarchar(16) NOT NULL	Determines whether to use CYCLIC or DELTA retrieval. The default is DELTA.
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

AttributeType

Contains one row for each attribute type.

Column	Data Type	Description
(PK) AttributeTypeKey	int NOT NULL	The unique numerical identifier for the attribute. This value is automatically generated by the system when the attribute is added.
AttributeTypeName	nvarchar(255) NOT NULL	The name of the attribute type.
AttributeTypeValue	tinyint NOT NULL	The bit mask for the attribute type.

CalcType

Contains one row for each type of summary calculation that can be performed by the Event subsystem.

Column	Data Type	Description
(PK) CalcType	CalcTypes(char(3)) NOT NULL	The type of calculation to be performed: SUM, MAX, MIN, or AVG.
Description	nvarchar(50) NULL	The description of the calculation.

ChangeNotification

Contains one row for each configuration modification made for a tag.

Column	Data Type	Description
(PK) ChangeType	sysname (nvarchar(128)) NOT NULL	Internal use only.
ChangeTime	datetime2(7) NOT NULL	Internal use only.
ChangeVersion	timestamp NOT NULL	Internal use only.

ChannelStatus

Contains one row for each type of channel status.

Column	Data Type	Description
(PK) ChannelStatus	tinyint NOT NULL	Internal use only.
Description	nvarchar(255) NOT NULL	Internal use only.

ChartConfiguration

Defines configuration settings for a particular Insight content.

Column	Data Type	Description
(PK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
ChartConfigurationName	nvarchar(200), NOT NULL	The name of the Insight content.
ChartConfigurationUrl	nvarchar(100), NOT NULL	The web address for the Insight content.
ChartConfigurationType	tinyint, NOT NULL	Specifies what type of chart was saved. For example, single chart or dashboard
(FK) ChartConfigurationOwnerKey	int, NOT NULL	A unique identifier for the Insight content owner.
ChartConfigurationShareMode	tinyint, NOT NULL	Specifies whether the Insight content is shared.
LastSharedDateTimeUtc	datetime2(7), NULL	Specifies when the Insight content was last shared.
CreationDateTimeUtc	datetime2(7), NOT NULL	Specifies when the InSight content was created.
TimePreset	nvarchar(200), NULL	Specifies the selected time frame of the saved content. For example: Last 30 days, Last hour, or specific start and end times (for Custom).
TimeAggregate	tinyint, NULL	Specifies the aggregates used by the saved content. For example, Hour/Day for a Column chart.
ChartType	nvarchar(100), NULL	The type of chart used for this Insight content.
MobileShareMode	tinyint, NOT NULL	Specifies whether this Insight content is shared with mobile users.

Column	Data Type	Description
EmbedShareMode	tinyint, NOT NULL	Specifies whether this Insight content can be embedded into a web page or other object.

ChartConfigurationAuditLog

Contains one row for each chart configuration audit log entry.

Column	Data Type	Description
ChartConfigurationUrl	nvarchar(100), NOT NULL	The web address for this InSight content.
(PK) ChartConfigurationUserKey	int, NOT NULL	A unique identifier for the InSight content user.
AuditLogDescription	nvarchart(200), NOT NULL	A descriptive record for the log.
(PK) CreationDateTimeUtc	datetime2(7), NOT NULL	The creation date and time for the log entry.

ChartConfigurationKeyword

Contains one row for each keyword associated with a particular Insight content.

Column	Data Type	Description
(PK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
Keyword	nvarchar(50), NOT NULL	A list of keywords associated with the content.

ChartConfigurationProperty

Contains one row for configuration property used by Insight charts.

Column	Data Type	Description
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
(PK) ChartConfigurationPropertyKey	nvarchar(100), NOT NULL	The unique identifier for the configuration property.

Column	Data Type	Description
(PK) ChartConfigurationPropertyValue	nvarchar(200), NOT NULL	A value for the property.

ChartConfigurationStatistics

Contains statistics about chart configuration access.

Column	Data Type	Description
(PK, FK) ChartConfigurationUserKey	int, NOT NULL	The unique identifier for the InSight user.
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the InSight content.
LastAccessDateTimeUtc	datetime2(7), NULL	Specifies when the InSight content was last accessed.

ChartConfigurationTag

Contains one row for each tag configuration used in an Insight chart.

Column	Data Type	Description
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the Insight content.
(PK, FK) FQN	TagNameType(nvarchar(256)), NOT NULL	The fully qualified name for the tag. A fully qualified tagname uses the format: DataSourceName.TagName.
Selected	bit, NOT NULL	Indicates whether the tag is selected to display in the chart of saved content.
Color	nvarchar(10), NOT NULL	Indicates the color associated with the charted tag.
ActiveGroup	bit, NOT NULL	Specifies whether the tag is part of the active group.
(PK) LayoutIndex	smallint, NOT NULL	Indicates the index of the layout when the content is recreated in the browser.
SelectedOrder	smallint, NULL	Insight tracks the order in which tags for a chart are selected. This column indicates the position for this particular tag in that selection order.

Comments

Contains details of comments associated with a tag.

Column	Data Type	Description
Comments Key	int NOT NULL	The unique numerical identifier for the comment.
(FK) Comments TypeKey	tinyint NOT NULL	The unique numerical identifier for the comment type. CommentsTypeKey is a foreign key from the CommentsType table.
(FK) Comments OwnerKey	int NOT NULL	The unique numerical identifier for the owner of the comment.
(FK) Comments ShareMode	tinyint NOT NULL	Indicates whether or not the comment is shared.
(FK) FQN	nvarchar(256) NOT NULL	Fully qualified name of the associated tag. A fully qualified tagname uses the format: DataSourceName.TagName. FQN is a foreign key from the ChartConfigurationTag table.
DateTime	datetime2(7) NOT NULL	The timestamp for this comment.
DateTimeUtc	datetime2(7) NOT NULL	The UTC timestamp for this comment.
Content	nvarchar(1024) NOT NULL	The comment text.
CreationDateTime	datetime2(7) NOT NULL	The date and time that the comment was created
CreationDateTimeUtc	datetime2(7) NOT NULL	The UTC date and time that the comment was created
ModifiedDateTime	datetime2(7) NULL	The date and time that the comment was modified.
ModifiedDateTimeUtc	datetime2(7) NULL	The URC date and time that the comment was modified
LastSharedDateTime	datetime2(7) NULL	The date and time that the comment was last shared
LastSharedDateTimeUtc	datetime2(7) NULL	The UTC date and time that the comment was last shared

CommentsType

Contains comment type definitions.

Column	Data Type	Description
(PK) Comments TypeKey	tinyint NOT NULL	The unique numerical identifier for the comment type.
Comments TypeName	nvarchar(32) NULL	A name for this comment type.

ConfigStatusPending

Contains one row for each database modification that requires a reinitialization of the system.

Important: Do not edit this table.

Column	Data Type	Description
(PK) ID	int NOT NULL	The unique identifier for the database modification.
Type	tinyint NOT NULL	Used to indicate the type of object to which the modifications apply. 0 = IDAS; 1 = IOServer; 2 = Topic; 3 = Tag; 4 = StorageLocation; 5 = SnapshotDetail; 6 = NamedSystemParameter; 7 = EngineeringUnit.
ObjectKey	int NOT NULL	The unique identifier of the modified object. If the modified object is a system parameter, the value will be 0. For all other object types, the value is from one of the following tables and columns: IODriver.IODriverKey; IOServer.IOServerKey; Topic.TopicKey; Tag.wwTagKey; StorageLocation.StorageType; SnapshotDetail.StorageSize.
Status	tinyint NULL	Used to indicate the type of modification. 1 = Insert; 2 = Update; 3 = Delete; 6 = The tag's source has changed (that is, if the value of the IOServerKey or TopicKey column in the Tag table has changed).

ConfigStatusSnapshot

When changes to the historian system are committed, a snapshot of the contents of the ConfigStatusPending table are stored to this table. The internal configuration object then finishes processing the reinitialization based on the data in this table, while any new changes are being stored in the ConfigStatusPending table.

The columns in this table are identical to the columns in the ConfigStatusPending table.

Important: Do not edit this table.

Context

Contains one row for each context to which a group of limits, rates of change, or deviations can belong. Example contexts are "Normal Operation" and "Cold Shutdown."

Column	Data Type	Description
(PK) ContextKey	int NOT NULL	The unique numerical identifier for the context. This value is automatically generated when a new context is added.
Description	nvarchar(50) NOT NULL	The description of the context.

CurrentEditor

Contains one row for each editor.

Column	Data Type	Description
(PK) CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
EditorName	nvarchar(max) NOT NULL	The name of the editor.

CustomReplicationSchedule

Contains one row for each trigger time for a custom replication schedule of ScheduleType CUSTOM. (This is used exclusively for tiered historian installations.) Interval-based replication schedules are handled in the IntervalReplicationSchedule table.

Column	Data Type	Description
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
TimeOfDay	nvarchar(10) NOT NULL	The time of day (in the local time for the AVEVA Historian) for the trigger time in the custom replication schedule. This value is automatically populated based on the schedule. The format is <Hour:Minutes><AM/PM>. Time is displayed on a 12-hour clock.

DashboardConfiguration

Contains one row for each InSight dashboard configured.

Column	Data Type	Description
(PK, FK) DashboardConfigurationKey	int, NOT NULL	The unique identifier for the InSight dashboard.
(PK, FK) ChartConfigurationKey	int, NOT NULL	The unique identifier for the InSight content.
Positions	smallint, NOT NULL	Indicates the position/index of the chart when the chart is displayed in the Dashboard along with the other charts.

DeletedReplicationTagEntity

Contains one row for each attribute ...

Column	Data Type	Description
ReplicationTagEntityKey	int, NOT NULL	The unique identifier for the replication tag entity.
ChangeVersion	timestamp, NOT NULL	Internal use only.

DeletedTag

Contains one row for each deleted tag.

Column	Data Type	Description
TagId	uniqueidentifier NOT NULL	Internal use only.
ChangeVersion	timestamp NOT NULL	Internal use only.

DetectorType

Contains one row for each type of event detector.

Column	Data Type	Description
(PK) DetectorTypeKey	int NOT NULL	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6 = Time-based (schedule). This value is automatically generated when a new detector is created.
Name	nvarchar(33) NOT NULL	The name given to the type of detector.
Description	nvarchar(50) NULL	The description of the detector.
EditorClassName	nvarchar(80) NULL	The name by which the component is referenced by a client application, such as the System Management Console, in order to provide a visual representation.
DetectorClassName	nvarchar(80) NULL	The name by which the detector component (COM object) is referenced in the system in order to perform the detection.
ExecutionMode	tinyint NOT NULL	Used to specify the manner in which the detector executes. 0 = Executed cyclically by the event subsystem according to the event tag scan rate; 1 = Asynchronous and triggered by an external mechanism. The default is 0.

Deviation

Contains one row for each defined deviation for an analog tag. The deviation is the percentage of change in a tag's value from a fixed value, called the target. Each analog tag can have two defined deviations: major and minor. This table is populated when an InTouch application is imported and is not used by the AVEVA Historian.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(PK) (FK) ContextKey	int NOT NULL	The unique numerical identifier for the context. ContextKey is a foreign key from the Context table.
MinorDeviation	real NULL	The percentage that the tag can deviate from the target value before a minor deviation alarm condition is produced.
MinorChecked	bit NOT NULL	Used to determine the alarm state of the tag based on the minor deviation. 0 = Not in an alarm condition; 1 = In an alarm condition.
MinorPriority	int NULL	The priority level for the minor deviation. Valid values are numbers between 1 and 999, with 1 being the highest priority and 999 being the lowest priority.
MajorDeviation	real NULL	The percentage that the tag can deviate from the target value before a major deviation alarm condition is produced.
MajorChecked	bit NOT NULL	Used to determine the alarm state of the tag based on the major deviation. 0 = Not in an alarm condition; 1 = In an alarm condition.
MajorPriority	int NULL	The priority level for the major deviation. Valid values are numbers between 1 and 999, with 1 being the highest priority and 999 being the lowest priority.
Target	float NULL	The reference value of the tag from which minor and/or major deviation percentages are based.
Deadband	real NULL	The deviation percentage the tag value must drop below the target before the tag is taken out of alarm.

DiscreteSnapshot

Contains one row for each discrete tag value that was configured to be stored when a defined event occurred. To view analog, discrete, and string snapshot values at the same time, use the v_SnapshotData view instead. For more information, see *v_SnapshotData* on page 164.

Column	Data Type	Description
(PK) (FK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. SnapshotTagKey is a foreign key from the SnapshotTag table.
(PK) (FK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. EventLogKey is a foreign key from the EventHistory table.
Value	tinyint NULL	The state of the discrete tag at the time that the event occurred. 0 = FALSE; 1 = TRUE.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

EngineeringUnit

Contains one row for each defined engineering unit (unit of measure).

Column	Data Type	Description
(PK) EUKey	int NOT NULL	The unique numerical identifier of an engineering unit. This value is automatically generated by the system when the engineering unit is added.
Unit	nvarchar(32) NULL	The unit of measure. Examples are mph, grams, and pounds.
DefaultTagRate	int NULL	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag.
Status	tinyint NULL	Automatically updated by the system if a change is made to the engineering unit: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
(FK) EUCKey	int NOT NULL	The unique numerical identifier for the catalog unit. Foreign key to the EngineeringUnitCatalog table.

Column	Data Type	Description
EUChangeVersion	timestamp NOT NULL	For system use. Updated automatically to indicate the date and time of the last update.

EngineeringUnitCatalog

Contains one row for each defined engineering catalog unit.

Column	Data Type	Description
(PK) EUCKey	int NOT NULL	The unique numerical identifier of an engineering catalog unit. This value is automatically generated by the system when the engineering unit is added.
(FK) EUDKey	int NOT NULL	The unique numerical identifier of the dimension the unit belongs to. Foreign key to the EngineeringUnitDimension table.
(FK) EUSKey	int NOT NULL	The unique numerical identifier of the system of measure the unit belongs to. Foreign key to the EngineeringUnitSystem table.
Symbol	nvarchar(32) NOT NULL	Canonical (standard) symbol used by the system of measurement that the unit belongs to. May contain special characters, such as the degree symbol for temperature measurements (for example, 68°).
BasicSymbol	nvarchar(32) NOT NULL	Canonical (standard) symbol used by the system of measurement that the unit belongs to. May not contain special characters.
Description	nvarchar(80) NOT NULL	A more descriptive name for the unit.
IntegralDivisor	float NOT NULL	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000.
ToBaseOffset	float NOT NULL	The offset used in linear conversion with respect to the base unit (b in the following line equation: $y = m(x+b)$).
ToBaseScaleFactor	float NOT NULL	The scale factor used in linear conversion with respect to the base unit (m in the following line equation: $y = m(x+b)$).

Column	Data Type	Description
IsVisible	bit NOT NULL	Set to 1 if this unit is displayed when you are creating a new engineering unit, and trying to link it to an existing catalog unit. Default value = 1.
IsSystem	bit NOT NULL	Set to 1 if this unit is defined by the system. (For example, it was created by the RuntimePostData.sql script.) Default value = 0.

EngineeringUnitDimension

Contains one row for each defined engineering unit dimension.

Column	Data Type	Description
(PK) EUDKey	int NOT NULL	The unique numerical identifier of a unit of measure's dimension.
Dimension	nvarchar(50) NOT NULL	The unit of measure's dimension name. For example, temperature, flow rate, length. Case-sensitivity depends on the default database collation.
BaseEUCKey	int NOT NULL	The unique numerical identifier of the base catalog unit for the given dimension. Linear conversions will be performed via the base unit. Foreign key to the EngineeringUnitCatalog table. Default value is 0 when creating a new dimension.
(FK) IntegralEUDKey	int NULL	The unique numerical identifier for this dimension's integral dimension. If not null, it indicates that accumulating units over time in this dimension results in units in the integral dimension. For example, the Volumetric Flow dimension's integral dimension is Volume.
(FK) DerivativeEUDKey	int NULL	The unique numerical identifier for this dimension's derivative dimension. If not null, it indicates that derivation of units over time in this unit results in units in the derivative dimension. For example, the Length dimension's derivative dimension is Speed.

EngineeringUnitSystem

Contains one row for each defined engineering unit system.

Column	Data Type	Description
(PK) EUSKey	int NOT NULL	The unique numerical identifier of an engineering unit system.
System	nvarchar(50) NOT NULL	The name of the system of measure. For example, SI, US, British Imperial. Case-sensitivity depends on the default database collation.

ErrorLog

Contains one row for each system message (or error message). Typically, this table is not used. The actual message text is stored in the LocalizedText table, and can be retrieved by specifying the error code in the SQL query. Or, you can use the v_ErrorLog view to retrieve the data included in this table, plus the actual text.

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The date that the message was written to the system log, in the local time of the AVEVA Historian.
Type	nvarchar(10) NULL	The type of system message.
ErrorCode	int NULL	The unique identifier for the message.
Parameter	nvarchar(256) NULL	Optional details pertaining to the message text. For example, for the message "Disk space remaining on circular path" the parameter would contain the number of MB.
TotalCount	int NULL	Used to prevent "flooding" conditions in the log file. If a particular message is generated numerous times during a relatively short period of time, the message is written to the log file only once, and the total number of times that it occurred appears in this column.
ModuleID	int NULL	A unique number assigned to the AVEVA Historian subsystem that generated the message.
Host	nvarchar(32) NULL	The computer on which the AVEVA Historian subsystem runs.
FileName	nvarchar(64) NULL	Used to indicate the program file that contains the line of code that an error message comes from. Used for debugging.
Line	int NULL	Used to indicate the line of code that an error message comes from. Used for debugging.

EventHistory

Contains one row for each stored event, as labeled by the tagname. Event data must be configured to be logged into this table.

Column	Data Type	Description
(PK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. This value is automatically generated by the system when the event record is added.

Column	Data Type	Description
(FK) TagName	TagNameType (nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the EventTag table.
DateTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectDateTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the event system.
Edge	tinyint NULL	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector.

EventTagPendingDelete

Contains one row for each event tag that is pending deletion. This table is used internally by the system during the deletion process. The columns in this table are the same as in the *_EventTag* on page 41 table.

Frequency

Contains one row for each available frequency for summary operations.

Column	Data Type	Description
(PK) FrequencyID	int NOT NULL	The unique numerical identifier for the frequency. Used to link a frequency with a time-based detector. 1= Hourly; 2 = Daily; 3 = Weekly; 4 = Monthly; 5 = Periodic; 6 = Other (Reserved for future use). This value is automatically generated by the system when the summarized tag is added.
Frequency	nvarchar(12) NOT NULL	The name for the frequency.

GroupTagList

Contains one row for each identified group of tags.

Column	Data Type	Description
(PK, FK) GroupID	int, NOT NULL	Globally unique identifier for the tag group.
(PK, FK) wwDomainTagKey	int, NOT NULL	The unique numerical identifier for a tag in a specific domain.
Triggerval	float, NULL	A value that can be read by an application as a trigger value.

History (INSQL.Runtime.dbo.History)

Contains one row for each stored tag value.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp of the returned value. For delta retrieval, this is typically the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
TagName	(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float NULL	The value of the tag at the timestamp. The value is always NULL for string tags.
vValue	nvarchar(4000) NULL	The value of the analog, discrete, or string tag stored as a sql_variant. Using this column in a query allows you to have values with mixed datatypes as a result.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
OPCQuality	int NULL	The quality value received from the data source.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

Column	Data Type	Description
wwRowCount	int NULL	The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query. This parameter has been deprecated; do not use. Use the wwCycleCount parameter instead.
wwResolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
wwEdgeDetection	nvarchar(16) NULL	The type of edge detection result set that the query will return. Valid values are NONE, LEADING, TRAILING, and BOTH.
wwOption	nvarchar(512) NULL	Specifies whether to return information about original data, summary data, or gaps in the storage blocks. Valid values are: • AutoSummaryData • PrimaryData • BlockGaps

Column	Data Type	Description
wwRetrievalMode	nvarchar(16) NULL	Used to specify how retrieved data is processed before it is returned to the client. Valid values are: CYCLIC, DELTA, FULL, INTERPOLATED, BESTFIT, AVERAGE, MINIMUM, MAXIMUM, INTEGRAL, SLOPE, COUNTER, VALUESTATE, and ROUNDTRIP. • FULL = All stored values are returned • CYCLIC = All stored data for tags during the specified time interval are returned for the number of retrieval cycles or resolution specified • DELTA = Only values that changed during the specified time interval are returned. For all other modes, a calculation is performed by the system on the data and the value(s) are returned. The default is CYCLIC for retrieval from analog tables, DELTA for retrieval from discrete and string tables, and default is DELTA for retrieval from the History table, unless the specific retrieval mode implies otherwise. For example, SLOPE always has DELTA characteristics.
wwTimeDeadband	int NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
wwValueDeadband	float NULL	The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval. The default is 0.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.
wwVersion	nvarchar(30) NULL	If the original data values have been modified in the database, use this column to specify which version of the stored data is to be retrieved. Valid values are: ORIGINAL or LATEST. If no parameter is specified, the latest version of the data is retrieved by default. Modification is indicated by the QualityDetail.

Column	Data Type	Description
wwCycleCount	int NULL	The number of retrieval cycles (sub-intervals) for the specified time period. The cycles will be spaced evenly across the time period. For example, if you specify a cycle count of four, the time period will be divided into four even cycles, and one or more values (depending on the retrieval mode) will be returned per cycle.
wwTimeStampRule	nvarchar(20) NULL	Used to specify whether cyclic results are timestamped at the beginning of the cycle or at the end of the cycle. Valid values are START and END. If no timestamp rule is specified in the query, then retrieval uses the setting of the TimeStampRule system parameter.
wwInterpolationType	nvarchar(20) NULL	Used to determine which analog value to return at a given cycle boundary. Valid values are STAIRSTEP and LINEAR. If STAIRSTEP is specified, no interpolation occurs. The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned. If LINEAR is specified, the system calculates a new value at the given cycle time by interpolating between the last known value prior to the cycle time and the first value after the cycle time.

Column	Data Type	Description
wwQualityRule	nvarchar(20) NULL	Used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC. A quality rule of GOOD means that data values with doubtful (64) OPC quality will not be used in the retrieval calculations and will be ignored. Values with bad QualityDetail indicate gaps in the data. A quality rule of EXTENDED means that data values with both good and doubtful OPC quality will be used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data. A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values will not cause the overall calculations to return NULL. You can apply wwQualityRule to all retrieval modes.
wwStateCalc	nvarchar(20) NULL	Used to indicate the type of calculation to return in the StateTime column for the "value state" retrieval mode. Valid values are: MINIMUM, MAXIMUM, AVERAGE, TOTAL, or PERCENT. You can also use the shortened versions: MIN, MAX, AVG, or SUM. The default for this column is TOTAL.
StateTime	float NULL	The amount of time in the state, expressed as a float (64-bit) number of milliseconds, for all time-in-state modes except for "Percent." For a time-in-state percentage calculation, this value is the percentage of the total time interval, in the range 0.0 to 100.0, that the value was in the state.
PercentGood	float NULL	The ratio of the number of rows that have "good" quality to the total number of rows in the retrieval cycle, expressed as a percentage in the range 0 to 100.
wwParameters	nvarchar(128) NULL	Contains the "stream index" (used for informational purposes only) and the special index value to indicate

Column	Data Type	Description
		that the value was calculated by the "SLR()" filter. SLR stands for "simple linear regression," the algorithm used for predictive retrieval. By default, the value of this parameter is an empty string.
StartDateTime	datetime2 NOT NULL	Start time of the retrieval cycle for which this row is returned.
SourceTag	nvarchar(256) NULL	Returns the name of the source tag for a replicated tag at the time this point was stored. With the SourceServer, this column uniquely identifies the tag from which this replicated point is coming.
SourceServer	nvarchar(256) NULL	Returns the name of the server from which replication occurred for this replicated tag at the time this point was stored.
wwFilter	nvarchar(512) NULL	Gives the name of the filter. Filters are specified as C-like functions and parentheses are always required, even when the filter does not override the default parameters (no parameters are passed). Filter values are NoFilter, ToDiscrete(), SigmaLimit(), SnapTo(), and SLR(). The default value is NoFilter. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.
wwValueSelector	nvarchar(128) NOT NULL	Used to specify which column to return for specified analog summary tags in the four basic retrieval modes: DELTA, FULL, CYCLIC, and INTERPOLATED. The defined set of selectors are AUTO (the default in all modes if not overridden), MINIMUM or MIN, MAXIMUM or MAX, FIRST, LAST, AVERAGE or AVG, INTEGRAL, and STDDEV or STANDDEVIATION. The default AUTO setting returns the Last attribute in the Value column (which makes it accessible in the WideHistory table). You can only override the selector for the basic retrieval modes. FIRST, LAST, MIN, and MAX each have their own timestamp that will be used for the time part of the VTQ. AVG, INTEGRAL and STDDEV represent values that hold for the entire cycle so the summary period start time will be used for the time part of a VTQ.
wwMaxStates	int NULL	For internal use only.

Column	Data Type	Description
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

HistoryBlock (INSQL.Runtime.dbo.HistoryBlock)

Contains one row for each block of history data stored in the main storage partition's timeline.

Column	Data Type	Description
FromDate	datetime2(7) NOT NULL	The starting timestamp for the history block.
ToDate	datetime2(7) NOT NULL	The ending timestamp for the history block.
StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node.
Description	nvarchar(50) NULL	The description of the history block.
OnLine	tinyint NOT NULL	Deprecated.
HistoryArchived	int NOT NULL	Used to indicate whether or not the history block has been archived (backed up). 1 = No status; 2 = Archived; 3 = Restored; 4 = Deleted. Reserved for future use.

Column	Data Type	Description
SummaryArchived	int NOT NULL	Used to indicate whether or not the tag summary has been archived (backed up). 1 = No status; 2 = Archived; 3 = Restored; 4 = Deleted. Reserved for future use.
EventArchived	int NOT NULL	Used to indicate whether or not the event has been archived (backed up). 1 = No status; 2 = Archived; 3 = Restored; 4 = Deleted. Reserved for future use.
StorageAreaType	int NOT NULL	The paradigm used for storage. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent. Reserved for future use.
ArchiveDate	datetime2(7) NULL	The date at which the history block was archived. Reserved for future use.
ArchiveLocation	nvarchar(50) NULL	The location to which the history block was archived. Reserved for future use.
Version	int NULL	The version number for the history block. 1 = Block format used until release 3.0; 2 = Block format used for releases 3.0 and later. Reserved for future use.
Compression	int NULL	The version number for cyclic compression. 1 = No compression; 2 = Huffman encoding. Reserved for future use.
Sequence	int NOT NULL	The sequence number for the data stream. (1...n) Reserved for future use.
TimeZoneOffset	int NULL	The UTC offset, in minutes, from the local timestamp for when the history block was created. For example, a value of 480 would indicate an 8-hour offset from UTC, which would be Pacific Standard Time.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.

HistorianSysObjects

Contains one row for each object in the database for which changes can be tracked.

Column	Data Type	Description
(PK) id	int NOT NULL	The unique identifier for the object.
Type	char(2) NULL	The type of object. C = CHECK constraint; D = Default or DEFAULT constraint; F = FOREIGN KEY constraint; K = PRIMARY KEY or UNIQUE constraint; L = Log; P = Stored procedure; R = Rule; RF = Stored procedure for replication; S = System table; TR = Trigger; U = User table; V = View; X = Extended stored procedure. Currently, only changes for the user tables (object type U) are tracked.
Name	varchar(50) NULL	The name of the modified object.

aaHistClientReport

Contains one row for each Historian Client report sent from ArchestrA.

Column	Data Type	Description
(PK) ReportKey	int, NOT NULL	The unique numerical identifier for the Historian Client report.
(FK) ReportSiteKey	int, NOT NULL	The unique numerical identifier for the report site.
(FK) ReportFolderKey	int, NOT NULL	The unique numerical identifier for the report folder.
Name	nvarchar(255), NOT NULL	The name of the Historian Client report.
ReportType	tinyint, NOT NULL	Specifies the report type for the History Client report.
Description	nvarchar(255), NULL	A description of the Historian Client report.
ApplicationType	int, NULL	Provides the application type for ___
LastRun	datetime, NOT NULL	

Column	Data Type	Description
ReportData	image, NOT NULL	
Published	int, NOT NULL	
LockdownOptions	nvarchar(1024), NULL	

aahHistClientReportsFolder

Contains unique identifiers for Historian Client reports folders.

Column	Data Type	Description
(FK) ReportSiteKey	int, NOT NULL	The unique numerical identifier.
(PK) ReportFolderKey	int, NOT NULL	The unique numerical identifier.
ReportFolderName	nvarchar(255), NULL	Provides the name of the History Client Report folder.
Description	nvarchar(255), NULL	A description of the report folder.
ReportType	tinyint, NOT NULL	Indicates the report type.
Disabled	tinyint, NOT NULL	Specifies whether the folder is disabled.
DirectoryName	nvarchar(255), NULL	Provides the directory path for the History Client Report folder.
FileNameFormat	nvarchar(255), NULL	Indicates the filename format.
JobType	int, NULL	Indicates the job type.

aaHistClientReportSite

Contains unique identifiers for Historian Client report sites.

Column	Data Type	Description
(PK) ReportSiteKey	int, NOT NULL	The unique numerical identifier for the Historian Client report site.
SiteURL	nvarchar(255), NOT NULL	The URL for the Historian Client report site.
SitePath	nvarchar(255), NOT NULL	The path for the Historian Client report site.
Description	nvarchar(255), NULL	A description of the report site.
SiteVersion	nvarchar(10), NULL	The version of the site.
WISSiteName	nvarchar(50), NULL	The associated WIS site.

IntervalReplicationSchedule

Contains one row for each replication schedule of ScheduleType INTERVAL. (This is used exclusively for tiered historian installations.) Custom replication schedules are handled in the CustomReplicationSchedule table.

Column	Data Type	Description
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.
Period	smallint NOT NULL	The period value.
Unit	nvarchar(32) NOT NULL	The name of the unit.

InTouchNode

Contains one row for each InTouch node from which a tagname data dictionary (Tagname.x) is imported into the AVEVA Historian.

Column	Data Type	Description
(PK) NodeKey	int NOT NULL	The unique numerical identifier of the named InTouch node. A node key is automatically generated by the system when a node is added.
MachineName	nvarchar(255) NOT NULL	The name of the computer on which the InTouch application resides.
ApplicationName	nvarchar(32) NULL	The name of the InTouch application (VIEW).
Path	nvarchar(250) NULL	The UNC path to the InTouch Tagname.X file.
Description	nvarchar(50) NULL	The description of the InTouch node.
DuplicateChar	nvarchar(12) NOT NULL	The string that was added to a tag name as a prefix or suffix to make it unique.
PrefixOrSuffix	bit NOT NULL	Used to indicate whether unique tags were created by prefixing or suffixing the unique string for the node. 0 = Suffix; 1 = Prefix. Internal use only.
AlwaysModifyName	bit NOT NULL	Used to indicate whether a uniqueness string was added to every tag for the node. Internal use only.
ImportPlantTags	tinyint NOT NULL	Used to indicate whether plant tags were imported. (In InTouch, plant tags are called I/O tags.) Internal use only.
ImportSystemTags	tinyint NOT NULL	Used to indicate whether system tags were imported. Internal use only.
ImportMemoryTags	tinyint NOT NULL	Used to indicate whether memory tags were imported. Internal use only.
ImportAllTags	int NOT NULL	Used to indicate whether all tags were imported. Internal use only.
FixedStorageRate	tinyint NOT NULL	The cyclic storage rate, in seconds, for imported tags. Internal use only.
ImportRoute	tinyint NOT NULL	Used to indicate the type of import that was last performed for the node. Internal use only.

InTouchSpecific

Contains one row of import-related information for each data dictionary (Tagname.x) imported from InTouch HMI software.

Column	Data Type	Description
(PK) (FK) NodeKey	int NOT NULL	The unique numerical identifier of the named InTouch node. NodeKey is a foreign key from the InTouchNode table.
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
OriginalName	nvarchar(32) NOT NULL	The original tag name in an InTouch application. The tag name may be different than the AVEVA Historian tag name if a new name was generated to ensure uniqueness.
TypeInfo	int NOT NULL	The type of tag in an InTouch application. For more information about InTouch tag types, see your InTouch documentation. Internal use only.
InInSQL	bit NOT NULL	Used to specify whether or not the tag information has been imported from InTouch into the AVEVA Historian database. Internal use only.
Comment	nvarchar(50) NULL	The original description for the tag that was imported from InTouch.

IOServerType

Contains one row for every known I/O Server type. Information about a new I/O Server is added to this table when a server is installed. This table is populated with the latest information about AVEVA I/O Servers at the time of shipping.

Column	Data Type	Description
(PK) ApplicationName	nvarchar(32) NOT NULL	The application name of the I/O Server. This name is usually the same as the executable file name.
Description	nvarchar(100) NULL	The description of the I/O Server type.
ExeName	nvarchar(255) NULL	The name of the I/O Server's executable file.
Revision	nchar(20) NULL	The revision number for the I/O Server.

Limit

Contains one row for each monitored limit for a specified tag. A limit can be associated with one or more tags and/or contexts.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the AnalogTag table.
(PK) (FK) ContextKey	int NOT NULL	The unique numerical identifier for the context. ContextKey is a foreign key from the Context table.
(PK) LimitType	tinyint NOT NULL	The type of limit; that is, whether it is a rising (up) or falling (down) limit. 0 = Rising; 1 = Falling.
(PK) ,Value	float NOT NULL	The value that is used as a specific limit for a tag. In theory, a tag can have an infinite number of limits defined.
(PK) (FK) LimitNameKey	int NOT NULL	The unique numerical identifier associated with a limit name. LimitNameKey is a foreign key from the LimitName table.
Priority	int NOT NULL	The priority for the limit. Priorities can range from 1 to over 2 billion, with 1 being the highest priority.
Checked	bit NOT NULL	Used to specify whether a tag imported from InTouch is configured for automatic limit checking. Only checked limits are imported. 0 = Checking disabled; 1 = Checking enabled.
Description	nvarchar(50) NULL	The description of the limit.

LimitName

Contains one row for each name that is associated with a defined limit. Examples are "high," "low," and "maintenance."

Column	Data Type	Description
(PK) LimitNameKey	int NOT NULL	The unique numerical identifier associated with a limit name. This value is automatically generated by the system when a limit is added.
Name	nvarchar(20) NULL	The name for the limit.

Live (INSQL.Runtime.dbo.Live)

Contains one row for each analog, discrete, or string tag. The value of each tag in this table is updated every time a new value is received.

Note: In certain situations, data can bypass the Live table. These situations include:
- Receiving non-streamed original data (store/forward or CSV);
- Receiving revision data for a Latest value;
- Receiving no new streamed values after Historian was shut down and disabled, or after the computer was rebooted.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data last changed.
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float NULL	The value of the tag at date/time. This value is always NULL for string tags.
vValue	nvarchar(256) NULL	The value of the analog, discrete, or string tag stored as a sql_variant. Using this column in a query allows you to have values with mixed datatypes as a result.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
OPCQuality	int NULL	The quality value received from the data source.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.
wwRetrievalMode	nvarchar(16) NULL	For queries against this table, the value of this column is ignored.
wwTimeDeadband	int NULL	For queries against this table, the value of this column is ignored.
wwValueDeadband	float NULL	For queries against this table, the value of this column is ignored.

Column	Data Type	Description
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.
wwParameters	nvarchar(128) NULL	Used for additional parameters that can be specified. By default, the value of this parameter is an empty string.
SourceTag	nvarchar(256) NULL	Returns the name of the source tag for a replicated tag at the time this point was stored. With the SourceServer, this column uniquely identifies the tag from which this replicated point is coming.
SourceServer	nvarchar(256) NULL	Returns the name of the server from which replication occurred for this replicated tag at the time this point was stored.
wwValueSelector	nvarchar(128) NOT NULL	Used to specify which column to return for specified analog summary tags in the four basic retrieval modes, DELTA, FULL, CYCLIC, and INTERPOLATED. The defined set of selectors are AUTO (the default in all modes if not overridden), MINIMUM or MIN, MAXIMUM or MAX, FIRST, LAST, AVERAGE or AVG, and INTEGRAL. The default AUTO setting returns the Last attribute in the Value column (which makes it accessible in the WideHistory table). You can only override the selector for the basic retrieval modes.

Column	Data Type	Description
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

LocalizedText

Contains one row for each string of text that can be returned to a client from AVEVA Historian (for example, error messages and status messages).

If you add new text to the LocalizedText table, you must stop and restart AVEVA Historian for the changes to go into effect.

Column	Data Type	Description
(PK) TextKey	int NOT NULL	The unique identifier for the message.
(PK) LanguageID	int NOT NULL	The locale ID for the language used. This ID is also used in the SQL Server syslanguages table.
LocalizedText	nvarchar(max) NULL	The content of the message.

ManualAnalogHistory

Contains one row for each tag history that was manually defined for analog tags.

Column	Data Type	Description
(PK) DateTime	datetime2(7), NOT NULL	The date and time that the tag history was defined.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	tinyint, NULL	The value of the tag at date/time.
Quality	tinyint, NOT NULL	An internal representation of data quality.
QualityDetail	int, NULL	An internal representation of data quality.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualDiscreteHistory

Contains one row for each tag history that was manually defined for discrete tags

Column	Data Type	Description
(PK) DateTime	datetime2(7), NOT NULL	The date and time that the tag history was defined.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	tinyint, NULL	The value of the tag at date/time.
Quality	tinyint, NOT NULL	An internal representation of data quality.
QualityDetail	int, NULL	An internal representation of data quality.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualStringHistory

Contains one row for each tag history that was manually defined for string tags.

Column	Data Type	Description
(PK) DateTime	datetime2(7), NOT NULL	The date and time that the tag history was defined.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512), NULL	The value of the tag at date/time. This value is always NULL for string tags.
Quality	tinyint, NOT NULL	An internal representation of data quality.
QualityDetail	int, NULL	An internal representation of data quality.
wwTagKey	int, NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

Message

Contains one row for each on/off message pair that can be associated with a discrete tag. For example, a message pair may be "Open" and "Closed" and could be associated with valve and switch positions.

Column	Data Type	Description
(PK) MessageKey	int NOT NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is automatically generated by the system when the message pair is added.
Message0	nvarchar(64) NULL	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state.
Message1	nvarchar(64) NULL	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state.

ModLogColumn

Contains one row for each database column on which an INSERT, UPDATE, or DELETE has been performed.

Column	Data Type	Description
(FK) ModTableKey	int NOT NULL	The unique numerical identifier for the modification. ModTableKey is a foreign key from the ModLogTable table.
ColumnName	nvarchar(30) NOT NULL	The name of the modified column.
OldValue	sql_variant NULL	The value stored in the column before the modification was made, if the modification was to a configuration table. For modifications to history data using SQL INSERT and UPDATE statements, this column contains the timestamp of the earliest data affected by the INSERT or UPDATE operation. If multiple changes are made to the same data, then only the most recent change will be contained in this column. This column is not used for modifications made to history data using a CSV file.
NewValue	sql_variant NULL	The new value stored in the column, if the modification was to a configuration table. For modifications to history data, this column contains the total count of consecutive value updates attempted.

ModLogTable

Contains one row for each database table on which an INSERT, UPDATE, or DELETE has been performed.

Column	Data Type	Description
(PK) ModTableKey	int NOT NULL	The unique numerical identifier for the modification. This value is automatically generated by the system when a new modification record is added.
(FK) id	int NOT NULL	The unique identifier for the object that was modified. id is a foreign key from the HistorianSysObjects table.
ModType	char(1) NOT NULL	The type of modification. U = Update; I = Insert; D = Delete; 1 = SQL insert; 2 = SQL original insert; 3 = SQL update; 4 = CSV insert; 5 = CSV original insert; 6 = CSV update; 7 = CSV multi-point update; 8 = CSV "fast load" insert.
RowKey	sql_variant NOT NULL	The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.

Column	Data Type	Description
UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. This value is from the UserDetail table. Currently not used.
DateTime	datetime2(7) NOT NULL	The timestamp of when the modification occurred.
UserName	nvarchar(256) NOT NULL	The name of the database user that made the modification. The value of this column reflects the Windows authentication user name (for example, DOMAIN\user_login_name) or the SQL Server authentication user name (for example, dbo), depending on how the user is logged into the SQL Server when the modification is made. In the case of a CSV file import, this column contains the user name as it appears in the CSV file.

NameSpaceIcons

Contains one row for each icon.

Column	Data Type	Description
(PK) Type	int, NOT NULL	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups. This value is of data type int, with no default.
Icon	image, NULL	Includes the image of the icon.
Name	nvarchar(30), NOT NULL	The name of the object in the hierarchy.
Description	nvarchar(50), NULL	A description of the namespace icon.

OPCQualityMap

Contains one row for each defined OPC quality.

Column	Data Type	Description
(PK) OPCQuality	tinyint NOT NULL	The quality value received from the data source.
Description	nvarchar(100) NULL	The text that describes what the OPC quality value means. Do not modify this description.

PrivateGroupTag

Contains one row for each instance of a tag in a user's private namespace.

Column	Data Type	Description
(PK) (FK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. NameKey is a foreign key from the PrivateNameSpace table.
(PK) (FK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(PK) (FK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain. wwDomainTagKey is a foreign key from the TagRef table.

PrivateNameSpace

Contains one row for each object in the private namespace. Objects in the private namespace can include plant machines, areas, tags, and so on, and are organized in a hierarchy. Allows for more than one name to map to a single tag.

Column	Data Type	Description
(PK) (FK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(PK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. This value is automatically generated by the system when the object is added.

Column	Data Type	Description
Type	int NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
Name	nvarchar(255) NULL	The name of this object in the hierarchy.
ConfigStor	ntext(16) NULL	If the namespace object has configuration information associated with it (for example, configuration information for a set of trend curves, the name of the file that contains the configuration information).
ParentKey	int NOT NULL	The unique identifier for a named object in this namespace.

PublicGroupTag

Contains one row for each instance of a tag in the public namespace.

Column	Data Type	Description
(PK) (FK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. NameKey is a foreign key from the PublicNameSpace table.
(PK) (FK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain. wwDomainTagKey is a foreign key from the TagRef table.

PublicNameSpace

Contains one row for each object in the public namespace. Objects in the public namespace can include plant machines, areas, and so on, and are organized in a hierarchy. Allows more than one name to map to a single tag.

Column	Data Type	Description
(PK) NameKey	int NOT NULL	The unique identifier for the object in the namespace. This value is automatically generated by the system when the object is added.
Type	int NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
Name	nvarchar(255) NULL	The name of this object in the hierarchy.
ConfigStor	ntext NULL	If the namespace object has configuration information associated with it (for example, configuration information for a set of trend curves, the name of the file that contains the configuration information).
ParentKey	int NOT NULL	The unique identifier for a named object in this namespace.
OriginalName	nvarchar(255) NOT NULL	Internal use only.

QualityMap

Contains one row for every permutation of quality detail for a tag value.

Column	Data Type	Description
(PK) QualityDetail	int NOT NULL	An internal representation of data quality.
QualityString	nvarchar(max) NULL	The text string that describes what the quality detail value means.

RateOfChange

Contains one row for each monitored rate of change for a tag.

Column	Data Type	Description
(PK) (FK) TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(PK) (FK) ContextKey	int NOT NULL	The unique numerical identifier for the context. ContextKey is a foreign key from the Context table.
Value	float NOT NULL	The percentage of change for a tag during the amount of time specified by the time base.
TimeBase	int NOT NULL	The unit of time against which the rate of change will be measured.
Priority	int NOT NULL	The priority for the rate of change. Priorities can range from 1 to over 2 billion, with 1 being the highest priority.
Checked	bit NOT NULL	Used to specify whether a tag imported from InTouch was configured for automatic rate of change checking. 0 = Checking disabled; 1 = Checking enabled.

ReplicationGroup

Contains one row for each replication group. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationGroupKey	int NOT NULL	The unique identifier for the replication group.
ReplicationGroupName	nvarchar (255) NOT NULL	The name of the replication group.
(PK) (FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the ReplicationServer table.
(FK) ReplicationTypeKey	tinyint NOT NULL	Can be 1, 2, or 3. (1 = Simple Replication, 2 = Analog Summary Replication, 3 = State Summary Replication.) ReplicationTypeKey is a foreign key from the ReplicationType table.
(FK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the replication schedule. ReplicationScheduleKey is a foreign key from the ReplicationSchedule table.

Column	Data Type	Description
SummaryReplicationNamingScheme	nvarchar(512) NULL	The naming scheme for the replication tags belonging to this replication group. If the summary replication naming scheme is NULL, the summary replication naming scheme from the replication server is used as the default naming scheme for summary tags.
GroupAbbreviation	nvarchar(32) NULL	The abbreviation for the replication group. If GroupAbbreviation is NULL, ScheduleAbbreviation is used as the default group abbreviation.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationRule

Contains one row for each replication rule for your system.

Column	Data Type	Description
Name	nvarchar(255), NOT NULL	The name of the replication rule.
Priority	int, NOT NULL	The priority for the rule.
TagFilter	nvarchar(4000), NOT NULL	Do not edit. This shows the OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
(FK) ReplicationGroupKey	int, NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
(FK) ReplicationServerKey	int, NOT NULL	The unique identifier for the replication server.
Enabled	bit, NOT NULL	Used to indicate whether the replication rule is enabled. 0 - not enabled; 1- enabled
ApplyOtherRules	bit, NOT NULL	Used to indicate whether other rules apply. 0 - other rules do not apply; 1- other rules appy.

Column	Data Type	Description
Id	int, NOT NULL	
ChangeVersion	timestamp, NOT NULL	Internal use only.

ReplicationSchedule

Contains one row for each replication schedule. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationScheduleKey	int NOT NULL	The unique identifier for the schedule.
ReplicationScheduleName	nvarchar(255) NOT NULL	The name of the replication schedule.
(FK) ReplicationScheduleTypeKey	int NOT NULL	The type of replication schedule. ReplicationScheduleType is a foreign key from the ReplicationScheduleType table.
ReplicationScheduleAbbreviation	nvarchar(32) NOT NULL	The abbreviation for the replication schedule.
CreateGroup	bit NOT NULL	If TRUE, this replication schedule is automatically added to new replication groups.

ReplicationScheduleType

Contains one row for each type of replication schedule. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationScheduleTypeKey	int NOT NULL	The unique identifier for the schedule type.
ReplicationScheduleTypeName	nvarchar(32) NOT NULL	The name of the replication schedule type, either INTERVAL or CUSTOM. The default is INTERVAL.

ReplicationServer

Contains one row for each replication server. (This is used exclusively for tiered historian installations.) The password is encrypted by an internal routine before storing in this table.

Column	Data Type	Description
(PK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server.
ReplicationServerName	nvarchar(255) NOT NULL	The name of the replication server.
Description	nvarchar(512) NULL	The description of the replication server.
SFPath	nvarchar(260) NULL	The local store-and-forward path associated with the replication server for this instance of AVEVA Historian.
SFFreeSpace	int NOT NULL	The free space for the store-and-forward path in MB.
CompressionEnabled	bit NULL	Used to specify whether compression should be enabled for the tag. 0 = No compression; 1= Compression.
UserName	nvarchar(255) NULL	The user name for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
Password	nvarchar(512) NULL	The encrypted password for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
TCPPort	int NOT NULL	The TCP port to use to log in to the replication server.
SummaryReplicationNamingScheme	nvarchar(512) NULL	The naming rule for summary replication tags. If ReplicationGroupKey is NULL, the naming rule is used from the ReplicationServerName scheme. If ReplicationServerName is NULL, the naming rule is used from the SummaryReplicationNamingScheme system parameter.

Column	Data Type	Description
SimpleReplicationNamingScheme	nvarchar(512) NULL	Naming rule for simple replication tags. If NULL the naming rule specified in the simple replication naming scheme system parameter is used.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
Bandwidth	int NOT NULL	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited.
MinSFDuration	int NOT NULL	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
ConnectionDetails	nvarchar(1024) NULL	Internal use only.
IntegratedSecurity	bit, NULL	Indicates whether this will be used for local replication connection and not for remote. (For remote replication, users are expected to provide username and password.)
ReplicationEvents	bit, NOT NULL	Specifies whether events are to be replicated. This applies only to remote servers.
ChangeVersion	timestamp, NOT NULL	Internal use only.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication server: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationShard

Contains one row for each partition (shard) used for replication.

Column	Data Type	Description
(PK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
ShardName	nvarchar (255), NOT NULL	The name of the partition.
Description	nvarchar (512), NULL	The description of the partition.
ComputerName	nvarchar (255), NULL	The name of the computer on which the partition resides.
CmdArgs	nvarchar (4000), NULL	Do not edit. These are command line parameters for customizing replication and storage execution.
CmdExtArgs	nvarchar (4000), NULL	Do not edit. These are command line parameters for customizing replication and storage execution.
Id	int, NOT NULL	The unique identifier for the object.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationSyncRequest

Contains one row for each replication synchronization request. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationSyncRequestKey	bigint NOT NULL	The unique identifier for the replication synchronization request.
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
RequestVersion	smallint NOT NULL	The version type. 0 = Initial version; 1 = Latest version.
ModStartDateTimeUtc	datetime2(7) NOT NULL	The start time (in UTC) for the replication synchronization request.

Column	Data Type	Description
ModEndDateTimeUtc	datetime2(7) NOT NULL	The end time (in UTC) for the replication synchronization request.
EarliestExecutionDateTimeUtc	datetime2(7) NULL	The earliest execution date (in UTC) for the replication synchronization request.
ExecuteState	tinyint NOT NULL	Value automatically changes as the rep service processes the sync queue. 0 = ready to process; 1 = currently being processed; 2 = rows needs merging/unmerging.

ReplicationSyncRequestPending

Contains one row for each pending replication synchronization request. This table is used internally by the system.

The columns in this table are the same as in the *ReplicationSyncRequest* on page 107 table.

ReplicationTagEntity

Contains one row for each replication tag entity. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
(PK) (FK) ReplicationServerKey	int NOT NULL	The unique identifier for the replication server. ReplicationServerKey is a foreign key from the Replication Server table.
(PK) DestinationTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
DestinationTagID	uniqueidentifier NOT NULL	The unique identifier for the destination tag.
(FK) SourceTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the source tag. SourceTagName is a foreign key from the Tag table.
(FK) ReplicationGroupKey	int NOT NULL	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.

Column	Data Type	Description
MaximumStates	tinyint NOT NULL	Maximum number of states to track for state summary tags. Discrete summary tags have a limit of 3 states. Analog summary tags of a limit of 100 states. The default is 10 states.
(FK) CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication tag: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationType

Contains one row for each replication type. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
(PK) ReplicationTypeKey	tinyint NOT NULL	Can be 1, 2, or 3.
ReplicationTypeName	nvarchar(255) NOT NULL	Value is determined by the ReplicationTypeKey. 1 = Simple Replication, 2 = Analogy Summary Replication, 3 = State Summary Replication.

SearchMessageSyncRequest

Contains one row for each search message synchronization request. (This is used exclusively for tiered historian installations.)

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
(PK) Id	int, NOT NULL	The unique numerical identifier for a search message synchronization request.
JobId	nvarchar(255), NOT NULL	The unique numerical identifier for a search message synchronization job.
Indexing Message	nvarchar(MAX), NOT NULL	Details from the JSON file used for search indexing to make the associated content searchable. This is an example: `{"_messageid":"test_636571577850365608","_tenantid":"","_body":[{"_keywords":[],"_fields":[{"ContentName":"test","_analyzers":"nGram"}],"_id":"afwwGwD30FUMl9UpE`
DocumentType	nvarchar(50), NOT NULL	Specifies the type of index document; for example SavedContent or Tag.
DeQueue Count	smallint, NULL	The number of times this message has been dequeued.
JobSubmisionTimeUtc	datetime2(7), NOT NULL	The time the job was submitted in UTC time.
LastModified TimeUtc	datetime2(7), NULL	The time the job was last modified in UTC time.
Status	smallint, NULL	The status of the search message synchronization job.

ServerList

Contains one row for each server used in an enterprise system. Allows for the creation of the system namespace, which contains a list of servers, and a flat namespace of tags per server.

Column	Data Type	Description
(PK) ServerKey	int NOT NULL	The unique numerical identifier of am AVEVA Historian server. This value is automatically generated by the system when a server is added.
ComputerName	nvarchar(50) NOT NULL	The Microsoft network name of the server computer.

Column	Data Type	Description
Description	nvarchar(50) NULL	The description of the server.

ShardAssignmentRule

Contains one row for each rule used for assigning tags to particular partitions (shards).

Note: This table is for internal use only. Do not edit this table.

Column	Data Type	Description
Name	nvarchar(255), NOT NULL	The name of the assignment rule.
Priority	int, NOT NULL	The priority assigned to this assignment rule.
TagFilter	nvarchar(4000), NOT NULL	Do not edit. This shows the OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
Enabled	bit, NOT NULL	Indicates whether this rule is enabled. 0 - not enabled; 1 - enabled.
Id	int, NOT NULL	The unique identifier for the object.
ChangeVersion	timestamp, NOT NULL	Internal use only.

ShareMode

Contains one row for each share mode used for InSight content.

Column	Data Type	Description
(FK) ShareModeKey	int, NOT NULL	A unique identifier for the share mode.
ShareModeName	timestamp, NOT NULL	The name of the InSight content share mode.

SnapshotDetail

Contains one row for each storage size configuration for tags. This table is used by the Classic Storage subsystem to manage the snapshot files.

Column	Data Type	Description
(PK) StorageSize	int NOT NULL	The storage size, in bytes, of the tag value: -1 = Blob; 0 = Variable length string; 1 = 1 byte; 2 = 2 byte; 4 = 4 byte; 8 = 8 byte.
SnapshotSize	int NOT NULL	The maximum size of the snapshot, in bytes. If this limit is reached, a new snapshot is created. The default is 2,097,152 bytes (2 MB).
ImageTime	int NOT NULL	The interval, in seconds, between updates to the snapshot file. The snapshot file is updated with tag value information from the snapshot buffer, which resides in memory. The default is 30 seconds, and the maximum value is 60 seconds.
ThresholdTime	int NOT NULL	The maximum amount of time, in seconds, that can elapse before a new snapshot is automatically created, provided that the value for the snapshot size has not been reached. The default is 3600 seconds (1 hour).
Status	tinyint NULL	Automatically updated by the system if a change is made to the snapshot: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

SnapshotTag

Contains one row for each tag that is included in the snapshot action associated with a given event tag.

Column	Data Type	Description
(PK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. This value is automatically generated by the system when the snapshot is added.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. This tag is the snapshot tag. TagName is a foreign key from the Tag table.
(FK) EventTagName	TagNameType (nvarchar(256)) NOT NULL	The name of the event tag to which the snapshot tag is related. EventTagName is a foreign key from the EventTag table.
(FK) TagType	int NOT NULL	Used to indicate the type of tag. 1 = Analog; 2 = Discrete; 3 = String. The default is 1. TagType is a foreign key from the TagRef table.

SQLTemplate

Contains one row for each pre-defined SQL script, which can be copied and used as a basis for an event detection or action script.

Column	Data Type	Description
(PK) TemplateKey	int NOT NULL	The unique numerical identifier for a SQL template. This value is automatically generated when a new SQL template is created.
Description	nvarchar(50) NULL	The description of the SQL template.
Type	int NULL	The type of SQL template. 0 = Detector; 1 = Action.
Script	ntext NULL	A pre-defined SQL script. This script can be copied and used as an event detection or action script.

StateSummaryHistory (INSQL.Runtime.dbo.StateSummaryHistory)

The StateSummaryHistory extension table returns results for state summary points.

Column	Data Type	Description
TagName	nvarchar(256) NOT NULL	The tag name.
StartDateTime	datetime2(7) NOT NULL	Start time of retrieval cycle.
EndDateTime	datetime2(7) NOT NULL	End time of retrieval cycle.
Value	float NULL	Numeric state.
vValue	nvarchar(4000) NULL	Non-numeric state.

Column	Data Type	Description
OPCQuality	int NULL	OPC quality. Normal OPC quality retrieval logic is applied if: • All the points found and processed for this row have GOOD quality. If they all have the same GOOD quality, then that quality is returned. • If there is a gap in the entire calculation cycle, then BAD quality is returned for the tag. • If the retrieval cycle contains a tag datatype transition (for example, the tag was configured to store integer values initially, then changed to store floating-point values) a DOUBTFUL OPC quality (64) is returned.
StateCount	int NULL	The number of times the state occurred within the retrieval cycle, including states that only partially occur in the cycle.
ContainedStateCount	int NULL	The number of times the state occurred fully contained within the retrieval cycle. States that only partially occur in the cycle are not counted.
StateTimeMin	float NULL	Minimum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeMinContained	float NULL	The minimum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.

Column	Data Type	Description
StateTimeMax	float NULL	Maximum time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeMaxContained	float NULL	The maximum of the contained times in this state among all occurrences of this state during the entire retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeAvg	float NULL	Average time in this state among all occurrences of this state during this retrieval cycle, including state occurrences that fall only partially within the period.
StateTimeAvgContained	float NULL	Average time in this state among all occurrences of this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimeTotal	float NULL	Total time in this state during this retrieval cycle, including state occurrences that fall only partially within the period.
StateTimeTotalContained	float NULL	Total time in this state during this retrieval cycle, excluding state occurrences that fall only partially within the period. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
StateTimePercent	float NULL	Percent of the time during this retrieval cycle that the tag was in this state, including state occurrences that fall only partially within the period.

Column	Data Type	Description
StateTimePercentContained	float NULL	The precentage of the entire retrieval cycle time that the tag was in this state, excluding state occurrences that fall only partially within the period. This is a ratio between StateTimeTotalContained and StateTimeTotal expressed as a percentage in the range 0 to 100. An occurrence that was partially contained in two or more consecutive storage cycles is converted to a contained state within the retrieval cycle if possible.
SourceTag	nvarchar(256) NULL	The source (tier 1) tag for the summary tag.
SourceServer	nvarchar(256) NULL	The source (tier 1) server for the summary tag.
wwCycleCount	int NULL	The number of cycles into which the entire query time range has been divided.
wwResolution	int NULL	Length of cycles in milliseconds. The default is 3600000 (equal to 1 hour).
wwTimeZone	nvarchar(50) NOT NULL	Time zone to use for interpreting both input and output timestamp parameters. If none is specified, then the default is set to LOCAL.
wwVersion	nvarchar(30) NOT NULL	Data version, ORIGINAL or LATEST. If none is specified, the default is LATEST.
wwTagKey	int NOT NULL	Tag key.
wwRetrievalMode	nvarchar(16) NOT NULL	Determines whether to use CYCLIC or DELTA retrieval. The default is DELTA.
wwMaxStates	int NULL	The maximum number of states (for state summaries) that are stored. The first N states will have summary values. For internal use only.

StateWideHistory (INSQL.Runtime.dbo.StateWideHistory)

Contains one row for the amount of time one or more analog, discrete, or string tags have been in a particular state, thus providing a "wide" view of the data.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp for the start of the time-in-state period.
vValue	sql_variant NULL	The string representation of the state, the ordinal for state types that do not have a string representation, or NULL for a gap or "bad" value.
Tag1	float NULL	The name of a tag to query.
Tag2	float NULL	The name of a tag to query.
ManyOtherTags	float NULL	A "placeholder" column for one or more tags in the wide table format. In the wide table format, tagnames are used as column names. The ManyOtherTags column is "duplicated" for as many tags as are specified in the database query.
wwRowCount	int NULL	The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query. This parameter has been deprecated; do not use. Use the wwCycleCount parameter instead.
wwResolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
wwRetrievalMode	nvarchar(16) NULL	Used to specify the time-in-state retrieval mode. The valid values are VALUESTATE and ROUNDTRIP. The default wwRetrievalMode is VALUESTATE.
wwTimeDeadband	int NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
wwValueDeadband	real NULL	The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval. The default is 0.

Column	Data Type	Description
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.
wwVersion	nvarchar(30) NULL	If the original data values have been modified in the database, use this column to specify which version of the stored data is to be retrieved. Valid values are: ORIGINAL or LATEST. If no parameter is specified, the latest version of the data is retrieved by default. Modification is indicated by the QualityDetail.
wwCycleCount	int NULL	The number of retrieval cycles (sub-intervals) for the specified time period. The cycles will be spaced evenly across the time period. For example, if you specify a cycle count of four, the time period will be divided into four even cycles, and one or more values (depending on the retrieval mode) will be returned per cycle.
wwTimeStampRule	nvarchar(20) NULL	Used to specify whether cyclic results are timestamped at the beginning of the cycle or at the end of the cycle. Valid values are START and END. If no timestamp rule is specified in the query, then retrieval uses the setting of the TimeStampRule system parameter.

Column	Data Type	Description
wwQualityRule	nvarchar(20) NULL	Used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC. A quality rule of GOOD means that data values with doubtful (64) OPC quality will not be used in the retrieval calculations and will be ignored. Values with bad QualityDetail indicate gaps in the data. A quality rule of EXTENDED means that data values with both good and doubtful OPC quality will be used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data. A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values will not cause the overall calculations to return NULL. You can apply wwQualityRule to all retrieval modes.
wwStateCalc	nvarchar(20) NULL	Used to indicate the type of calculation to return in the StateTime column for the "value state" retrieval mode. Valid values are: MINIMUM, MAXIMUM, AVERAGE, TOTAL, CONTAINED, or PERCENT. You can also use the shortened versions: MIN, MAX, AVG, or SUM. The default for this column is TOTAL.
wwParameters	nvarchar(128) NULL	Used for additional parameters that can be specified. By default, the value of this parameter is an empty string.
StartDateTime	datetime2(7) NOT NULL	Start time of the retrieval cycle for which this row is returned.
wwFilter	nvarchar(512) NOT NULL	The name of the filter. Filters are specified as C-like functions and parentheses are always required, even when the filter does not override the default parameters (no parameters are passed). Filter values are NoFilter, ToDiscrete(), SigmaLimit(), and SnapTo(). The default value is NoFilter. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.
wwMaxStates	int NULL	For internal use only.

StorageLocation

Contains one row for each defined storage location on a specific storage partition (shard).

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	A unique identifier for the partition.
(FK) StorageType	int, NOT NULL	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent. There can be only one storage location of each type.
Path	nvarchar(255), NOT NULL	The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different internal hard drive. If you use a network location, then the ArchestrA user must have full access to the network location. The locations must be specified using UNC notation. Mapped drives are not supported. If empty, the default <SystemDataPath>\Wonderware\Data\Circular is used.
MaxMBSize	int, NOT NULL	The limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used.
MinMBThreshold	int, NOT NULL	The minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block (before any compression) by 1.5 to determine the minimum threshold.
MaxAgeThreshold	int, NOT NULL	The age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied.

Column	Data Type	Description
(PK) Id	int, NOT NULL	The unique identifier for the object.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the storage location: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

StorageShard

Contains one row for each storage partition used by the system.

Column	Data Type	Description
(FK) ShardId	uniqueidentifier, NOT NULL	The unique identifier for the partition (shard).
ShardName	nvarchar(255), NOT NULL	The name of the partition.
Description	nvarchar (512), NULL	The description of the partition.
ComputerName	nvarchar (255), NULL	The network name of the computer on which the storage partition resides.
BlockDuration	int, NOT NULL	Duration, in hours, for history blocks. Valid values are: 1, 2, 3, 4, 6, 8, 12, 24. The default is 24 hours. The history block size must always be greater than the highest scan rate. For more information, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.
TimeUnitId	tinyint, NOT NULL	Foreign key to TimeUnit. Indicates whether the block duration is in hours/days.
TimeZoneId	smallint, NULL	The time zone associated with this storage partition.
AdjustToDST	tinyint, NOT NULL	Internal use only.
MaxSnapshotSize	int, NOT NULL	Maximum size, in MB, for data storage snapshots in memory. Bigger snapshots allow for faster retrieval. You might need to increase this size for systems with very high data rates. For example, if retrieval is slow from data in the current history block, try increasing this rate. Also be sure that you have enough RAM, up to 1 GB.

Column	Data Type	Description
Id	int, NOT NULL	The unique identifier for the object.
Status	tinyint, NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

StorageType

Contains one row for each type of storage used by the system.

Column	Data Type	Description
(PK) StorageTypeId	int, NOT NULL	A unique identifier for this storage type.
(FK) StorageType	int, NOT NULL	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent. There can be only one storage location of each type.
Description	nvarchar(255), NULL	A description of the storage type.

StringSnapshot

Contains one row for each string tag value that was configured to be stored when a defined event occurred. To view analog, discrete, and string snapshot values at the same time, use the v_SnapshotData view instead. For more information, see *v_SnapshotData* on page 164.

Column	Data Type	Description
(PK) (FK) SnapshotTagKey	int NOT NULL	The unique numerical identifier of the tag included in the snapshot. SnapshotTagKey is a foreign key from the SnapshotTag table.
(PK) (FK) EventLogKey	int NOT NULL	The unique numerical identifier of an event occurrence. EventLogKey is a foreign key from the EventHistory table.
Value	nvarchar(512) NULL	The value of the string tag at the event timestamp.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.

Column	Data Type	Description
QualityDetail	int NULL	An internal representation of data quality.

StructureAttributes

Contains one row for each attribute definition for the StructureType read-only table.

Column	Data Type	Description
(PK) (FK) StructureId	uniqueidentifier NOT NULL	The unique identifier for the structure. StructureID is a foreign key from the StructureType table.
(PK) AttributeName	nvarchar(255) NOT NULL	The name of the structure attribute.
(FK) AttributeTypeKey	int NOT NULL	The unique identifier for the structure attribute. AttributeTypeKey is a foreign key from the AttributeType table.
AttributeOrder	tinyint NOT NULL	The order of the attribute within the structure.

StructureType

Contains one row for each structure type. Read-only table.

Column	Data Type	Description
StructureId	uniqueidentifier NOT NULL	The unique identifier for the structure.
StructureTypeName	nvarchar(255) NOT NULL	The name of the structure type.
Description	nvarchar(512) NOT NULL	The description of the structure type.

SummaryData

Contains one row for each summarized value, or result, for a tag. This table is used by the event subsystem; it is not used by the replication subsystem. The Quality column contains the highest quality value of the raw data from which the result is calculated.

Column	Data Type	Description
(PK) (FK) LogKey	int NOT NULL	The unique numerical identifier of the summary's historical log. LogKey is a foreign key from the SummaryHistory table.
(PK) (FK) SumVarKey	int NOT NULL	The unique numerical identifier for a summarized tag. SumVarKey is a foreign key from the SummaryTagList table.
Value	float NULL	The value of the summary.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
Modified	int NOT NULL	Used to specify whether or not the data has been modified. This value is optional. 1 = Modified; 0 = Not modified.

SummaryHistory

Contains one row for each occurrence of a summary operation. This table is used by the Event subsystem; it is not used by the replication subsystem. Rows are inserted even if the operation did not return data.

Column	Data Type	Description
(PK) LogKey	int NOT NULL	The unique numerical identifier of the summary's historical log. This value is automatically generated by the system when the record is added.
(FK) OperationKey	int NOT NULL	The unique numerical identifier for the summary operation. OperationKey is a foreign key from the SummaryOperation table.
SummaryDate	datetime2(7) NOT NULL	The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.
SumDateTimeStamp	tinyint NULL	Duplication of the TimeStamp column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation timestamp setting performed, in case of a later modification of the summary operation definition.

Column	Data Type	Description
SumDateCalcType	varchar(3) NULL	Duplication of the CalcType column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation type performed, in case of a later modification of the summary operation definition.
SumDateDuration	real NULL	Duplication of the Duration column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation duration used in case of a later modification of the summary operation definition.
SumDateResolution	int NULL	Duplication of the Resolution column of the SummaryOperation table at the SummaryDate. This column allows you to keep the original calculation resolution used, in case of a later modification of the summary operation definition.
Status	tinyint NOT NULL	The flag indicating the status of the operation. 0 = Operation completed successfully; Not 0 = Operation is in progress or has failed. Reserved for future use.
OperationStart	datetime2(7) NULL	The timestamp when the calculation started for the operation.
OperationEnd	datetime2(7) NULL	The timestamp when the calculation completed for the operation.

SummaryOperation

Contains one row for each defined summary operation that is associated with the event tag specified in the TagName column. This table is used by the Event subsystem; it is not used by the Replication subsystem.

Column	Data Type	Description
(PK) OperationKey	int NOT NULL	The unique numerical identifier for the summary operation. This value is automatically generated by the system when the operation is added.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the EventTag table.
(FK) CalcType	CalcTypes (char(3)) NOT NULL	The type of calculation to be performed: SUM, MAX, MIN, or AVG. CalcType is a foreign key from the CalcType table.

Column	Data Type	Description
Description	nvarchar(50) NULL	The description of the summary operation.
Duration	real NOT NULL	The period, in seconds, for which the calculation is performed.
Resolution	int NOT NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
TimeStamp	tinyint NOT NULL	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends. 0 = Beginning of the calculation period; 1 = End of the calculation period.
Frequency	nvarchar(12) NULL	The name for the frequency.
SourceType	varchar(3) NULL	The type of summary, set to 'DYN' (for "dynamic" data). Used for backward compatibility with Industrial Workbook.

SummaryTagList

Contains one row for each combination of a summarized tag and a specific summary operation. This table is used by the Event subsystem; it is not used by the Replication subsystem. This table is a linking table that allows tags to be associated with a type of operation.

Column	Data Type	Description
(PK) SumVarKey	int NOT NULL	The unique numerical identifier for a summarized tag. This value is automatically generated by the system when the summarized tag is added.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) OperationKey	int NOT NULL	The unique numerical identifier for the summary operation. OperationKey is a foreign key from the SummaryOperation table.

Column	Data Type	Description
LowerLimit	float NULL	The lower limit of validity for the tag's value. Values lower than this limit are not used in the calculation. By default, this value is set to -1000000000.
UpperLimit	float NULL	The upper limit of validity for the tag's value. Values higher than this limit are not used in the calculation. By default, this value is set to 1000000000.
Description	nvarchar(50) NULL	The description of the summarized tag. This normally describes the result of the operation, although this description can be the same as that of the tag on which the operation is performed.

SystemParameter

Contains one row for each system parameter.

Column	Data Type	Description
(PK) Name	nvarchar(50) NOT NULL	The unique name for the system parameter.
Value	sql_variant NULL	The value of the system parameter.
Editable	bit NOT NULL	Used to determine if the value of the named system parameter can be changed using the InSQL Console. 1 = Editable; 0 = Not editable.
Description	nvarchar(255) NULL	The description of the system parameter.
Status	tinyint NULL	Automatically updated by the system if a change is made to the named system parameter: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

TagExtendedPropertyName

Contains the name of each extended tag property stored by AVEVA Historian, namely HierarchicalName and Alias.

Column	Data Type	Description
(PK) PropertyNameKey	int NOT NULL	A unique identifier for the extended tag property name.

Column	Data Type	Description
PropertyName	nvarchar (256) NOT NULL	The extended tag property name.
(FK) PropertyType	int NOT NULL	Specifies a type for this extended tag property.
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
Substring Searchable	bit NOT NULL	Specifies whether the name can be located with a substring search.

TagGroup

Contains one row for each tag group used by the system.

Column	Data Type	Description
(PK) GroupID	int, NOT NULL	A unique identifier for the tag group.
Description	nvarchar(50), NULL	A description of the tag group.
CreatedDate	datetime2(7), NULL	The date the tag group was created.
CreatedBy	nvarchar(18), NULL	The person or application that created the tag group.
Type	int, NULL	The type of tags in this group.

TagHistory

Historian needs some essential information about how time-series data is stored ("metadata") in order to correctly interpret that data when stored in the history blocks. Since this metadata can change over the life of a tag, the system must preserve the complete metadata record for all history blocks. The ability to store tag metadata to an alternate file location is also possible.

When a tag record gets modified or deleted in the Tag table, the system automatically preserves the previous tag version in the TagHistory table. The TagHistory table should not be modified by the user, otherwise the data stored in the history block may become inaccessible.

Contains one row for each tag metadata instance uniquely identified by the TagId column.

Important: This table is for internal use only. Do not edit this table as it may result in unpredictable behavior. Additionally, Invensys reserves the right to make modifications to the structure/schema of the table as needed.

Column	Data Type	Description
(PK) TagId	uniqueidentifier NOT NULL	The unique identifier for the tag. Internal use only.
TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. Internal use only.
Description	nvarchar(512) NULL	The description of the tag. Internal use only.
AcquisitionType	tinyint NOT NULL	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired; 1 = Acquired via an I/O Server; 2 = Acquired via HCAL or MDAS or a manual update; 3 = System driver. Internal use only.
StorageType	smallint NOT NULL	The type of storage defined for the tag. 0 = Not stored; 1 = Cyclic; 2 = Delta; 3 = Forced storage; 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." Internal use only.
StorageRate	int NOT NULL	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. Internal use only.
TagType	int NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 5 = Event, 7 = Summary tag (analog or state). TagType is a foreign key from the TagRef table. Internal use only.

Column	Data Type	Description
TimeDeadband	int NULL	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. Internal use only.
DateCreated	datetime2(7) NOT NULL	The date that the tag was created. If not specified, this date will be automatically generated. Internal use only.
CreatedBy	nvarchar(256) NOT NULL	The name of the user or application that created the tag. If not specified, this name will be automatically generated. Internal use only.
CurrentEditor	tinyint NOT NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server. CurrentEditor is a foreign key from CurrentEditor table. Internal use only.

Column	Data Type	Description
ServerTimeStamp	bit NOT NULL	Used to indicate whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used; 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. Internal use only.
DeadbandType	smallint NOT NULL	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. Internal use only.
ChannelStatus	tinyint NOT NULL	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. ChannelStatus is a foreign key from ChannelStatus table. Internal use only.
AIHistory	bit NOT NULL	Used to indicate whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage; 1 = The tag may have data previously collected by classic storage. Internal use only.
Message0	nvarchar(64) NULL	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. Internal use only.

Column	Data Type	Description
Message1	nvarchar(64) NULL	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. Internal use only.
Unit	nvarchar(32) NULL	The unit of measure. Examples are mph, grams, and pounds. Internal use only.
DefaultTagRate	init NULL	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag. Internal use only.
IntegralDivisor	float NULL	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. Internal use only.
MinEU	float NULL	The minimum value of the tag, measured in engineering units. Internal use only.
MaxEU	float NULL	The maximum value of the tag, measured in engineering units.Internal use only.
MinRaw	float NULL	The minimum value of the raw acquired value. Internal use only.
MaxRaw	float NULL	The maximum value of the raw acquired value. Internal use only.
Scaling	int NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use). Internal use only.

Column	Data Type	Description
RawType	int NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes). Internal use only.
ValueDeadband	float NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. Internal use only.
IntegerSize	tinyint NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use). Internal use only.
SignedInteger	bit NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed. Internal use only.
RateDeadband	float NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. Internal use only.
InterpolationType	tinyint NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. Internal use only.
RolloverValue	float NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000. Internal use only.

Column	Data Type	Description
MaxLength	smallint NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512. Internal use only.
DoubleByte	tinyint NULL	Used to specify whether or not to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0. Internal use only.
StructureId	uniqueidentifier NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table. Internal use only.
SourceTag	nvarchar(256) NULL	The name of the source tag used for the replication tag. Internal use only.
SourceServer	nvarchar(255) NULL	The name of the tier 1 server with the source tag. Internal use only.

TagRef

Contains one row for each tag in the system. This table is used as a reference table for the Tag table, so that the TagName column is not propagated as the primary key of child tables.

Column	Data Type	Description
(PK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain. This value is automatically generated by the system when the tag is added.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian. wwTagKey is populated from the Tag table, but is not a foreign key.
(FK) ServerKey	int NOT NULL	The unique numerical identifier of an AVEVA Historian server. ServerKey is a foreign key from the ServerList table.
(FK) TagName	TagNameType (nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
TagType	int NOT NULL	The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 4 = Complex; 5 = Event, 7 = Summary tag (analog or state).

TagType

Contains one row for each tag type.

Column	Data Type	Description
(PK) TagTypeKey	int NOT NULL	The unique identifier for the tag type.
TagTypeName	nvarchar(32) NULL	The name of the tag type.

TimeDetectorDetail

Contains at least one row for each event tag associated with a time detector.

Column	Data Type	Description
(PK) TimeDetectorDetailKey	int NOT NULL	The unique numerical identifier for each time-based event tag. This value is automatically generated by the system when a time-based event tag is created.
(FK) FrequencyID	int NOT NULL	The unique numerical identifier for the frequency. Used to link a frequency with a time-based detector. 1= Hourly; 2 = Daily; 3 = Weekly; 4 = Monthly; 5 = Periodic; 6 = Other (Reserved for future use). FrequencyID is a foreign key from the Frequency table.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
Periodicity	int NULL	The interval period in minutes between detector events. Only used for a periodic detection.
StartDateTime	datetime2(7) NULL	The timestamp from which the time detector starts. Only used for a periodic detection.
RunTimeDay	tinyint NULL	In the context of a weekly detector, RunTimeDay maps the week day number (0 = Sunday – 6 = Saturday). In the context of a monthly detector, RunTimeDay maps to the day of the month. Not used for periodic detections.
RunTimeHour	tinyint NULL	The hour of the day at which the time detector triggers. Not used for periodic detections.

Column	Data Type	Description
RunTimeMin	tinyint NULL	The minute of the hour at which the time detector triggers. Not used for periodic detections.

TimeDetectorDetailPendingDelete

Contains one row for each time detector that is pending deletion. This table is used internally by the system during the deletion process.

The columns in this table are the same as in the *TimeDetectorDetail* on page 135 table.

TimeUnit

Contains one row for each unit of time used by the system.

Column	Data Type	Description
Id	tinyint, NOT NULL	The unique identifier for the object.
(PK) Name	nvarchar(32), NOT NULL	The name of the time unit.

TimeZone

Contains one row for each time zone as defined by the Windows operating system. This table is automatically populated by the system.

Column	Data Type	Description
(PK) TimeZoneID	smallint NULL	The unique numerical identifier for the time zone.
TimeZone	nvarchar(100) NULL	The name of the time zone.
Description	nvarchar(100) NULL	The description of the time zone that includes the hour offset from UTC (GMT) and major cities or regions in the time zone.
Offset	smallint NOT NULL	The offset, in minutes, for daylight savings time, when in effect.
RegistryName	nvarchar(100) NULL	The Windows registry name of the time zone, which is always in English.

TopicImportInfo

Contains one row for each topic definition imported from an InTouch node.

Column	Data Type	Description
(PK) (FK) NodeKey	int NOT NULL	The unique numerical identifier of the named InTouch node. This value is automatically generated by the system when the node is added.
(PK) DdeSourceKey	int NOT NULL	The unique identifier for the DDE source. Assigned by the AVEVA Historian system when data is imported.
SourceName	nchar(50) NOT NULL	The DDE Access Name from InTouch.
ApplicationName	nchar(50) NULL	The name of the InTouch application from which the topic definition is imported.
TopicName	nchar(50) NOT NULL	The name of the topic definition that is imported.
RequestInitialData	bit NOT NULL	Used to determine if the topic was configured to request initial data. See the InTouch documentation for more information. Internal use only.
AlwaysAdvise	bit NOT NULL	Used to determine if the topic was configured to poll all items for data. See the InTouch documentation for more information. Internal use only.
DefaultStorageRate	int NOT NULL	The cyclic storage rate for the topic. Internal use only.
DefaultStorageType	int NOT NULL	The default storage type for the topic. Internal use only.
TimeDeadband	int NOT NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
ValueDeadband	float NOT NULL	Either the InTouch log deadband or the AVEVA Historian deadband, as specified by the DeadbandType column. Internal use only.
DeadbandType	tinyint NULL	The type of deadband used. Internal use only.
Import	bit NOT NULL	Used to indicate whether the topic has previously been imported from InTouch into AVEVA Historian. Internal use only.

Column	Data Type	Description
ProtocolType	tinyint NOT NULL	The protocol used by the AVEVA Historian server to communicate with the I/O Server. Internal use only.
IODriverKey	int NULL	The unique identifier for an IDAS.
RateDeadband	float NOT NULL	The rate deadband that was specified during the InTouch topic import. Internal use only. This rate deadband is not used for swinging door storage. For more information on the rate deadband for swinging door storage, see *AnalogSummaryTag*.

UserDetail

Contains one row for each AVEVA Historian user. Used to store additional user information that is not stored in the Microsoft SQL Server Runtime.sysusers table. Applicable for both users and groups of users.

When AVEVA Historian is installed, a SQL job is created on the Microsoft SQL Server that automatically updates this table every hour. In order for this job to run, the SQL Server Agent must be running. For more information about jobs, see your Microsoft Online Books.

Column	Data Type	Description
(PK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table.
UserName	nvarchar(128) NOT NULL	The name of the database user.
AccessLevel	int NOT NULL	The security level for the user. 1 is the lowest level and 9999 is the highest. Used to limit access of certain users.
uid	int NOT NULL	The identifier for the user. This ID is referenced from the Microsoft SQL Server sysusers table.
gid	int NOT NULL	The identifier for the group in which a user belongs. This ID is referenced from the Microsoft SQL Server sysusers table.

WideHistory (INSQL.Runtime.dbo.WideHistory)

Contains one row of values for multiple analog, discrete, or string tags for a single timestamp, thus providing a "wide" view of the data.

Because tagnames are used as column names for the returned data (indicated by Tag1, Tag2, and ManyOtherTags), the value data types will be of the appropriate type for associated tags.

Column	Data Type	Description
DateTime	datetime2(7) NOT NULL	The timestamp for the returned value. For delta retrieval, this is typically the time at which the value was acquired by the AVEVA Historian. For cyclic retrieval, this is the specific time requested or calculated (using a SQL function).
Tag1	(as per the tag type) NULL	The name of a tag to query.
Tag2	(as per the tag type) NULL	The name of a tag to query.
ManyOtherTags	(as per the tag type) NULL	A "placeholder" column for one or more tags in the wide table format. In the wide table format, tagnames are used as column names. The ManyOtherTags column is "duplicated" for as many tags as are specified in the database query.
wwRowCount	int NULL	The number of rows to be returned for a specified time period. For cyclic retrieval, the rows are spaced evenly across the time period, and the default row count is 100 rows. For cyclic retrieval, the row count is applied for each tag in a query. This parameter has been deprecated; do not use. Use the wwCycleCount parameter instead.
wwResolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
wwEdgeDetection	nvarchar(16) NULL	The type of edge detection result set that the query will return. Valid values are NONE, LEADING, TRAILING, and BOTH.

Column	Data Type	Description
wwRetrievalMode	nvarchar(16) NULL	Used to specify how retrieved data is processed before it is returned to the client. Valid values are: CYCLIC, DELTA, FULL, INTERPOLATED, BESTFIT, AVERAGE, MINIMUM, MAXIMUM, INTEGRAL, SLOPE, COUNTER, VALUESTATE, and ROUNDTRIP. FULL = All stored values are returned; CYCLIC = All stored data for tags during the specified time interval are returned for the number of retrieval cycles or resolution specified; DELTA = Only values that changed during the specified time interval are returned. For all other modes, a calculation is performed by the system on the data and the value(s) are returned. The default is CYCLIC for retrieval from analog tables, DELTA for retrieval from discrete and string tables, and default is DELTA for retrieval from the History table, unless the specific retrieval mode implies otherwise. For example, SLOPE always has DELTA characteristics. The default value for wwRetrievalMode is DELTA.
wwTimeDeadband	int NULL	The minimum time, in milliseconds, between returned values for a single tag. Applies only to delta retrieval.
wwValueDeadband	real NULL	The percentage of full scale (range), in engineering units. Any value changes that are less than this percentage are not returned. Applies only to delta retrieval. The default is 0.
wwTimeZone	nvarchar(50) NULL	Used to specify the time zone for retrieval. By default, the retrieval subsystem converts the UTC (Universal Time Coordinated) timestamps for the stored data to the local time of the AVEVA Historian computer, including adjustments for daylight savings time. To keep the timestamps in UTC, set the value of wwTimeZone to UTC. To convert the timestamps to a client computer's time zone, set this parameter to the appropriate time zone text key from the TimeZone table.
wwVersion	nvarchar(30) NULL	If the original data values have been modified in the database, use this column to specify which version of the stored data is to be retrieved. Valid values are: ORIGINAL or LATEST. If no parameter is specified, the latest version of the data is retrieved by default. Modification is indicated by the QualityDetail.

Column	Data Type	Description
wwCycleCount	int NULL	The number of retrieval cycles (sub-intervals) for the specified time period. The cycles will be spaced evenly across the time period. For example, if you specify a cycle count of four, the time period will be divided into four even cycles, and one or more values (depending on the retrieval mode) will be returned per cycle.
wwTimeStampRule	nvarchar(20) NULL	Used to specify whether cyclic results are timestamped at the beginning of the cycle or at the end of the cycle. Valid values are START and END. If no timestamp rule is specified in the query, then retrieval uses the setting of the TimeStampRule system parameter.
wwInterpolationType	nvarchar(20) NULL	Used to determine which analog value to return at a given cycle boundary. Valid values are STAIRSTEP and LINEAR. If STAIRSTEP is specified, no interpolation occurs. The last known point is returned with the given cycle time. If no valid value can be found, a NULL is returned. If LINEAR is specified, the system calculates a new value at the given cycle time by interpolating between the last known value prior to the cycle time and the first value after the cycle time.

Column	Data Type	Description
wwQualityRule	nvarchar(20) NULL	Used to specify whether values with certain characteristics are explicitly excluded from consideration by data retrieval. This parameter will override the setting of the QualityRule system parameter. Valid values are GOOD, EXTENDED, or OPTIMISTIC. A quality rule of GOOD means that data values with doubtful (64) OPC quality will not be used in the retrieval calculations and will be ignored. Values with bad QualityDetail indicate gaps in the data. A quality rule of EXTENDED means that data values with both good and doubtful OPC quality will be used in the retrieval calculations. Values with bad QualityDetail indicate gaps in the data. A quality rule of OPTIMISTIC means that calculations that include some good and some NULL values will not cause the overall calculations to return NULL. You can apply wwQualityRule to all retrieval modes.
wwValueSelector	nvarchar(20) NULL	Used to specify which column to return for specified analog summary tags in the four basic retrieval modes, DELTA, FULL, CYCLIC, and INTERPOLATED. The defined set of selectors are AUTO (the default in all modes if not overridden), MINIMUM or MIN, MAXIMUM or MAX, FIRST, LAST, AVERAGE or AVG, and INTEGRAL. The default AUTO setting returns the Last attribute in the Value column (which makes it accessible in the WideHistory table). You can only override the selector for the basic retrieval modes.
wwFilter	nvarchar(512) NOT NULL	Gives the name of the filter. Filters are specified as C-like functions and parentheses are always required, even when the filter does not override the default parameters (no parameters are passed). Filter values are NoFilter, ToDiscrete(), SigmaLimit(), SnapTo(), and SLR(). The default value is NoFilter. If the query does not specify the wwFilter element at all, or if its default value is not overridden, then no filter is applied.
wwParameters	nvarchar(128) NULL	Contains the "stream index" (used for informational purposes only) and the special index value to indicate that the value was calculated by the "SLR()" filter. SLR stands for "simple linear regression," the algorithm used for

Column	Data Type	Description
		predictive retrieval. By default, the value of this parameter is an empty string.
StartDateTime	datetime2(7) NOT NULL	Start time of the retrieval cycle for which this row is returned.
wwExpression	nvarchar(4000) NULL	Used to specify an expression for unit of measure conversion, specified in the following format using tag/unit pairs: `UOM(TAG1,UNIT1;TAG2,UNIT2;...)` For example, the expression `UOM(DistanceTag,m;TempTag,F;DurationTag,Minute)` returns the values for the tag named DistanceTag measured in meters, the values for TempTag measured in degrees Farenheit, and the values for DurationTag measured in minutes. The following rules apply: 1. If any of the unit conversions specified are invalid and fail (for example, trying to convert a tag measured in meters to a unit of hours) then no unit conversions are performed. 2. If any tags specified in the expression for unit conversion are not part of the query, those tags will be ignored for the purpose of unit conversion.
wwUnit	nvarchar(512) NULL	Returns the currently used unit of measure.

WideTableDictionary

Contains unique identifiers for tags and users.

Column	Data Type	Description
(PK, FK) UserKey	int, NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table.
(PK, FK) TagName	TagNameType(nvarchar(256)), NOT NULL	The unique name of the tag within the AVEVA Historian system.

CHAPTER 3

Views

A view is a logical way of looking at data from one or more tables in the database. A view is a "virtual" table that does not actually exist in the database. A view contains pointers to the actual tables in the database. Views can be used to include a subset of information stored in one or more tables, while leaving out other information. Views are part of normal SQL Server functionality.

To make it easier to query data from some of the AVEVA Historian tables, a number of views onto the data are provided. Queries are performed on these views as if they were normal physical tables.

Note: Some views are included in the database for backward compatibility support only. It is recommended that you discontinue the use of these views, as they will be dropped in a future release. For details about views retained for backward compatibility, see *Backward Compatibility Views* (see "*Backward Compatibility Views*" on page 293).

History Table Views

Views have been created for the extension tables to make querying these tables easier. Normally, you must use the full reference for an extension table in the query, which is **linked_server.catalog.schema.objectname**. An extension table view allows you to simply use the view name instead, eliminating the need to provide the long reference.

All of the following views reflect the same table structure as the extension tables after which they are named.

This view	References this extension table
History	INSQL.Runtime.dbo.History
HistoryBlock	INSQL.Runtime.dbo.HistoryBlock
Live	INSQL.Runtime.dbo.Live
AnalogSummaryHistory	INSQL.Runtime.dbo.AnalogSummaryHistory
StateSummaryHistory	INSQL.Runtime.dbo.StateSummaryHistory

Note: Some History table views are included in the database for backward compatibility support only. It is recommended that you discontinue the use of these views, as they will be dropped in a future release. For details about History table views retained for backward compatibility, see *History Table Views (Backward Compatible)* on page 293.

Annotation

Contains one row for each user annotation about a tag value. Users can make personal (or public) notes about a tag value. This information is stored with the tag value and timestamp to which the annotation applies. Each annotation in this view is linked to a database user.

Column	Data Type	Description
(PK) AnnotationKey	int NOT NULL	The unique numerical identifier for the annotation. This value is automatically generated by the system when the annotation is added.
(FK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(FK) TagName	TagNameType (nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
DateCreated	datetime(2)7 NULL	The date that the annotation was created.
Content	nvarchar(1000) NOT NULL	The annotation text.
DateTime	datetime2(7) NOT NULL	The timestamp of the tag value for which the user has made an annotation.
Value	float NULL	The value of the tag at the time of the annotation.

Events

Returns one row for each alarm and event.

Column	Data Type	Description
ID	GUID	Unique identifier for the event
EventTime	DateTime2	Time of the event, returned by default in the local time for the Historian server. **Note:** EventTime is modified by wwTimeZone criteria in the query, similar to the way DateTime is in the History table.
EventTimeUtc	DateTime2	UTC time stamp showing when the event occurred.
EventTimeUTCOffsetMIns	Integer	Offset, in minutes, between the Historian server time and UTC time.
Type	String	Main categorization of the event: Alarm.Set, Alarm.Clear, Alarm.Acknowledged, Alarm.Silenced.Set, Alarm.Silenced.Clear, Alarm.Suppressed.Set, Alarm.Suppressed.Clear, Alarm.Disabled.Set, Alarm.Disabled.Clear, User.Write, User.Write.Secured, User.Write.Verified, System.Deploy, System.Undeploy,

Column	Data Type	Description
		Engine.Start, Engine.Stop, Engine.OnScan, Engine.OffScan, Engine.Active, Engine.Standby, or Engine.Terminated.
Priority	Int32	Event importance. Values range from 1 to 999, with lower numbers indicating higher importance.
Severity	Int32	Urgency categorization of alarms or events: 1=Critical, 2=High, 3=Medium, 4=Low.
IsAlarm	Bool	Indicator telling whether the message is an alarm type message.
Source_ProcessVariable	String	Process variable to which the event is related. For example, if "TI101" has a field attribute "PV" and this is a "Hi" alarm, this is TI101.PV". This will normally match the name of the associated Historian tagname. In a User.Write or similar event, this is the attribute being written to. Also see the Source_ConditionVariable column.
Source_Units	String	Engineering units used for process variable. For example: feet, pounds, or N/m^2.
Alarm_Acknowledged	Boolean	Indicates whether the alarm was acknowledged (true or false).
Alarm_Class	String	(Legacy) InTouch alarm class..
Alarm_Condition	String	Condition being alarmed. It should be one of these predefined values, if appropriate: Limit.HiHi, Limit.Hi, Limit.Lo, Limit.LoLo, ROC.Hi, ROC.Lo, System, Discrete.True, Discrete.False, Deviation.Minor, Deviation.Major, SPC, Batch, Process, Software, System.
Alarm_DurationMs	Int32	Time, in milliseconds, between when the alarm is raised and when it is cleared.
Alarm_ID	String	Unique identifier for an instance of an alarm. This is used to link different alarm events together from the time the alarm is raised until the time it is no longered considered an alarm.
Alarm_InAlarm	Boolean	State of the alarm after the transition (true or false).
Alarm_IsShelved	Boolean	Indicator showing whether alarm was shelved (true or false).
Alarm_IsSilenced	Boolean	Indicator showing whether alarm is silenced (true or false).

Column	Data Type	Description
Alarm_LimitString	String	Limit being alarmed.
Alarm_OriginationTime	DateTime2	Date/Time stamp for when the initial alarm condition originated.
Alarm_ShelveDurationMs	Int32	The duration, in milliseconds, for which the alarm was shelved. Used only if alarm is currently shelved.
Alarm_ShelveReason	String	Reason for shelving. Used only if alarm is currently shelved.
Alarm_ShelveStartTime	DateTime2	Date/time stamp showing scheduled start of shelve time, if the alarm is shelved. Returned by default in local time for the Historian server. Used only if alarm is currently shelved.
Alarm_State	String	State of the alarm (UNACK_ALM, UNACK_RTN, ACK_ALM, or ACK_RTN).
Alarm_TagType	String	Types of tags associated with the alarm (S, I, or F).
Alarm_Type	String	(Legacy) InTouch alarm type.
Alarm_UnAckDurationMs	Int32	Time, in milliseconds, between when the alarm is raised and when it is acknowledged. This property should not be included for an alarm until it is acknowledged.
Comment	String	Additional comments or descriptions for the event.
IntouchType	String	Legacy InTouch event type. Valid values are: ALM, RTN, ACK, and SYS.
Provider_ApplicationName	String	Name of the application that generated the event. For Application Server, this is the galaxy name. For InTouch, this is the InTouch application name.
Provider_InstanceName	String	Provider-specific string that uniquely identifies the instance on a given node name for the given application.
Provider_NodeName	String	Name of the node that generated the event. For example, "AOS-N-01".
Provider_System	String	Software system that generated the event. For example, Application Server, InTouch, or InBatch.
Provider_SystemVersion	String	Software version for the component identified by Provider_ApplicationName.
ReceivedTime	DateTime2	Time at which the the Historian server received the event. Returned by default in the local time for the Historian server.

Column	Data Type	Description
Source_Area	String	Non-hierarchical area name. For example, "Area_001".
Source_ConditionVariable	String	Related condition variable. For example, if "TI101" has a field attribute "PV" and this is a "Hi" alarm, this value would be "TI101.PV.Hi". Also see the Source_ProcessVariable column.
Source_Engine	String	Non-hierarchical engine name. For example, "AppEngine_001".
Source_HierarchicalArea	String	Hierarchical area name. For example, "Plant_001.Building_002.Area_001".
Source_HierarchicalObject	String	Hierarchical name of source object. For example, For example, "Reactor_001.TIC".
Source_Object	String	Non-hierarchical name of source object. For example, "TIC101".
Source_Platform	String	Non-hierarchical platform name. For example, "WinPlatform_001".
User_Account	String	Login name for the operator for the specified application. Depending on the security model being used, this can be <Domain>\<Alias>, email address, or user profile name. For Application Server/GalaxySecurity Model: User Profile Name For Application Server/OS Model: OperatorDoman\Login For InTouch: Operator Login For Windows Live Login: TBD
User_Agent	String	Name of the application that the user was running when the event was generated.
User_Email	String	User email address. Note: This property is not currently in the OData metadata for events, but may be added in the future.
User_Name	String	Complete first name and last name. For example, "Chris Walton".
User_NodeName	String	Computer name from which the user performed the action. For example, "ww112.baytown.amci.com".
User_Phone	String	User's phone number. For example, "1-800-555-1212". Note: This property is not currently in the OData metadata for events, but may be added in the future.

Column	Data Type	Description
ValueString	String	The value that triggered the alarm, represented as a string. The maximum length of this string is 329 characters. **Note:** We recommend that you do not use this property except for legacy purposes, as extensive use of this property can negatively affect performance.
Verifier_Account	String	Login name for the verifier. Depending on the security model being used, this can be <Domain>\<Alias>, email address, or user profile name. For Application Server/GalaxySecurity Model: User Profile Name For Application Server/OS Model: OperatorDoman\Login For InTouch: Operator Login For Windows Live Login: TBD
Verifier _Agent	String	Name of the application that the verifier was running.
Verifier _Email	String	Verifier email address. Note: This property is not currently in the OData metadata for events, but may be added in the future.
Verifier _Name	String	Complete first name and last name
Verifier _NodeName	String	Computer name from where the verification occurred. For example, "ww112.baytown.amci.com".
Verifier _Phone	String	Verifier's phone number. For example, "1-800-555-1212". Note: This property is not currently in the OData metadata for events, but may be added in the future.
wwTimeZone	String	Specifies the time zone for this record.

IODriver

Contains one row for each IDAS providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IODriverKey	int NOT NULL	The unique identifier for an IDAS. This value is automatically generated by the system when the IDAS is added.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
ComputerName	nvarchar(255) NOT NULL	The name of the computer on which the IDAS runs.
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, redundant IDAS runs. You must use the fully qualified name of the computer. You could also use the IP address. This should be set to an empty string if no redundant IDAS is specified. Make sure that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online. By default, this column is an empty string.
StoreForwardMode	tinyint NOT NULL	Used to specify whether or not store-and-forward capability is enabled. If enabled, and the network connection between the IDAS and the storage node fails, data will be "buffered" to the location specified by the store-and-forward path. Valid values are: 0 = Disabled; 1 = Enabled; 2 = Autonomous. The Autonomous mode (2) is an extension of the normal store-and-forward mode (1). It allows the IDAS to start up using an IDAS configuration file and collect data in store-and-forward mode if the network connection to the AVEVA Historian is not available.

Column	Data Type	Description
StoreForwardPath	nvarchar(255) NULL	Used to specify the path for the IDAS data buffer on the local hard drive of the IDAS computer. The path should be absolute (for example, c:\IDASBuffer). Data is written to this path until the minimum threshold for the buffer is reached. Remote buffer paths are not supported. When the store-and-forward path specified for the IDAS is invalid, the default path picked by the system is: <public folder>\ArchestrA\Historian\IDAS\SF where the <public folder> is dependent on the operating system. For example, for the Windows 2008 operating system, the path is C:\ProgramData\ArchestrA\Historian\IDAS\SF. When the store-and-forward path specified for the IDAS is just a folder name (without any path characters like \ and :), the default path picked by the system is: <Windows system path>\<folder name specified by the user>. For example, for the Windows Server 2008 32-bit operating system, the path is C:\WINDOWS\system32\<folder name>.
MinMBThreshold	int NOT NULL	The minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer.
Status	tinyint NULL	Automatically updated by the system if a change is made to IDAS: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
Enabled	bit NOT NULL	Used to indicate whether the IDAS is enabled or not. 0 = Not enabled; 1 = enabled. Disabling the IDAS allows for the configuration to be retained in the database, even though the IDAS is removed from the system.
StoreForwardDuration	int NOT NULL	The minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.

Column	Data Type	Description
AutonomousStartupTimeout	int NOT NULL	The amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network.
BufferCount	int NOT NULL	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates.
FileChunkSize	int NOT NULL	The size, in bytes, of the data "chunks" that are sent to the historian when store-and-forward data is forwarded. The size of the chunks can be decreased to accommodate slower networks. Decrease this number only if the forwarding delay is greater than zero.
ForwardingDelay	int NOT NULL	The time interval, in milliseconds, at which "chunks" of store-and-forward data are forwarded to the historian. The length of the interval may need to be increased to accommodate slower networks.
ConnectionTimeout	int NOT NULL	The amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks.

IOServer

Contains one row for each I/O Server providing data to the AVEVA Historian.

Column	Data Type	Description
(PK) IOServerKey	int NOT NULL	The unique numerical identifier for the I/O Server. This value is automatically generated by the system when the I/O Server is added.

Column	Data Type	Description
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
(FK) IODriverKey	int NULL	The unique identifier for an IDAS. IODriverKey is a foreign key from the IODriver table.
(FK) ApplicationName	nvarchar(32) NULL	The application name of the I/O Server. This name is usually the same as the executable file name. ApplicationName is a foreign key from the IOServerType table.
Path	nvarchar(255) NULL	The full UNC path (including the filename) to locate the executable file for the I/O Server. If the I/O Server type key is specified, the filename may be omitted.
ComputerName	nvarchar(255) NULL	The name of the computer on which the I/O Server runs.
AltComputerName	nvarchar(255) NULL	The name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made.
AutoStart	bit NOT NULL	Used to control how the I/O Server starts up. 0 = Automatic startup when the system starts. 1 = Manual startup required. Currently not used.
ExeType	int NOT NULL	The type of executable for the I/O Server. Used by the Historian System Management Console to determine how to start the I/O Server. 0 = Service; 1 = Console application; 2 = Windows application.
InitializationStatus	tinyint NOT NULL	A control flag used to ensure that each I/O Server has been asked for the data type (integer or real) of each tag that it will send. Only needed after a database modification.
ProtocolType	tinyint NOT NULL	The protocol used by the AVEVA Historian server to communicate with the I/O Server. 1 = DDE; 2 = SuiteLink™; 3 = AVEVA Historian named pipe driver (for compatibility with IndustrialSQL Server 3.0 and previous versions). Of the operating systems currently supported by the AVEVA Historian, DDE is only supported on the Windows XP operating system.
Description	nvarchar(50) NULL	The description of the I/O Server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the I/O Server: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

ReplicationSyncRequestInfo

Contains one row for each replication synchronization request. (This is used exclusively for tiered historian installations.)

Column	Data Type	Description
SourceTagName	TagNameType (nvarchar(256)) NULL	The name of the source tag used for the replication tag.
ReplicationServerName	nvarchar(255) NULL	The name of the replication server.
DestinationTagName	TagNameType (nvarchar(256)) NULL	The name of the destination tag.
EarliestExecutionDateTimeUtc	datetime2(7) NULL	The earliest execution date (in UTC) for the replication synchronization request.
ModStartDateTimeUtc	datetime2(7) NOT NULL	The start time (in UTC) for the replication synchronization request.
ModEndDateTimeUtc	datetime2(7) NOT NULL	The end time (in UTC) for the replication synchronization request.
ReplicationSyncRequestKey	bigint NOT NULL	The unique identifier for the replication synchronization request.
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.
RequestVersion	smallint NOT NULL	The version type. 0 = Initial version; 1 = Latest version.
ExecuteState	tinyint NOT NULL	Value automatically changes as the rep service processes the sync queue. 0 = ready to process; 1 = currently being processed; 2 = rows needs merging/unmerging.

Column	Data Type	Description
CurrentEditor	tinyint NULL	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server.
DestinationTagID	uniqueidentifier NULL	The unique identifier for the destination tag.
MaximumStates	tinyint NULL	Maximum number of states to track for state summary tags.
ReplicationGroupKey	int NULL	The unique identifier for the replication group.
ReplicationServerKey	int NULL	The unique identifier for the replication server.
Status	tinyint NULL	Automatically updated by the system if a change is made to the replication server: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

Column	Data Type	Description
AuthenticateWithAAUser	bit NULL	1 if the login should be authenticated using the ArchestrA user name; otherwise, 0 to authenticate with the UserName and Password.
Bandwidth	int NULL	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited.
BufferCount	int NULL	The number of buffers.
Description	nvarchar(512) NULL	The description of the replication server.
MinSFDuration	int NULL	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds.
Password	nvarchar(512) NULL	The password for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
SFFreeSpace	int NULL	The free space for the store-and-forward path in MB.
SFPath	nvarchar(260) NULL	The local store-and-forward path associated with the replication server for this instance of AVEVA Historian.
ServerDefaultSimpleReplicationNamingScheme	nvarchar(512) NULL	Naming rule for simple replication tags. If NULL the naming rule specified in the SimpleReplicationNamingScheme system parameters is used.
ServerDefaultSummaryReplicationNamingScheme	nvarchar(512) NULL	The default naming rule for summary replication tags. If NULL, the naming rule specified in the SummaryReplicationNamingScheme system parameter is used.
TCPPort	int NULL	The TCP port to use to log in to the replication server.

Column	Data Type	Description
UserName	nvarchar(255) NULL	The user name for logging in to the replication server. (AuthenticateWithAAUser must be 0.)
GroupAbbreviation	nvarchar(32) NULL	The abbreviation for the replication group.
ReplicationGroupName	nvarchar(255) NULL	The unique identifier for the replication group.
ReplicationScheduleKey	int NULL	The unique identifier for the schedule.
ReplicationTypeKey	tinyint NULL	Can be 1, 2, or 3. (1 = Simple Replication, 2 = Analog Summary Replication, 3 = State Summary Replication.)
GroupDefaultSummaryReplicationNamingScheme	nvarchar(512) NULL	The group default naming rule for summary replication tags.
CompressionEnabled	bit NULL	0 = Compression off. 1 = Enable compression for the packets sent to the replication server.
ConnectionDetails	nvarchar(1024) NULL	Internal use only.

ReplicationTagExtendedPropertyInfo

Contains one entry for each replicated tag extended property.

Column	Data Type	Description
ReplicationTagExtendedPropertyKey	int NOT NULL	A unique identifier for the replication tag extended property.
ReplicationServerKey	int NOT NULL	The unique identifier for the replication server.
DestinationTagName	nvarchar (256) NOT NULL	The name of the destination tag. If the destination tag name is not specified, it is generated based on the naming convention for the replication tag and stored in the database.
ReplicationTagEntityKey	int NOT NULL	The unique identifier for the replication tag entity.

Column	Data Type	Description
SourceTagName	nvarchar (256) NOT NULL	The unique name for the source tag.
PropertyNameKey	int NULL	A unique identifier for the extended tag property name.
PropertyName	nvarchar (256) NOT NULL	The extended tag property name.
PropertyType	int NOT NULL	Specifies a type for this extended tag property.
PropertyValue	sql_variant NOT NULL	The value of this replication tag extended property.
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
SubstringSearchable	bit NOT NULL	Specifies whether the name can be located with a substring search.
AttributeTypeName	nvarchar (255) NOT NULL	The name of the attribute type.
AttributeTypeValue	tinyint NOT NULL	The bit mask for the attribute type
ChangeVersion	timestamp NOT NULL	Internal use only.
Status	tinyint NOT NULL	Automatically updated by the system if a change is made to the replication group: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.

TagExtendedPropertyInfo

Contains one entry for each extended property for a tag.

Column	Data Type	Description
TagName	nvarchar (256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Property NameKey	int NULL	A unique identifier for the extended tag property name.
Property Name	nvarchar (256) NOT NULL	The extended tag property name.
PropertyType	int NOT NULL	Specifies a type for this extended tag property.
Property Value	sql_variant NOT NULL	The value of this replication tag extended property.
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
Substring Searchable	bit NOT NULL	Specifies whether the name can be located with a substring search.
AttributeType Name	nvarchar (255) NOT NULL	The name of the attribute type.
AttributeType Value	tinyint NOT NULL	The bit mask for the attribute type
Change Version	timestamp NOT NULL	Internal use only.

TagExtendedPropertyNameInfo

Contains the name of the extended properties currently stored by AVEVA Historian, namely HierarchicalName and Alias.

Column	Data Type	Description
Property Name	nvarchar (256) NOT NULL	The extended tag property name.
PropertyType	int NOT NULL	Specifies a type for this extended tag property.

Column	Data Type	Description
Facetable	bit NOT NULL	Specifies whether the extended property can be included in grouped search results. For example, if a user searches for all items containing the string "temp", the search engine could display a list of multiple results.
Searchable	bit NOT NULL	Specifies whether the extended property is searchable.
Substring Searchable	bit NOT NULL	Specifies whether the name can be located with a substring search.
AttributeType Name	nvarchar (255) NOT NULL	The name of the attribute type.
AttributeType Value	tinyint NOT NULL	The bit mask for the attribute type.

Topic

Contains one row for each topic to be read from an I/O Server.

Column	Data Type	Description
(PK) TopicKey	int NOT NULL	The unique numerical identifier for the topic. This value is automatically generated by the system when the topic is added.
(PK) (FK) IOServerKey	int NOT NULL	The unique numerical identifier for the I/O Server. IOServerKey is a foreign key from the IOServer table.
(PK) (FK) StorageNodeKey	int NOT NULL	The unique numerical identifier for the storage node. StorageNodeKey is a foreign key from the StorageNode table.
Name	nvarchar(180) NOT NULL	The name of the topic.
TimeOut	int NOT NULL	The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic.
Status	tinyint NULL	Automatically updated by the system if a change is made to the topic: 0 = No change; 1 = Insertion; 2 = Modification; 3 = Deletion.
LateData	bit NOT NULL	Used to enable acquisition of "late" data. 0 = Late data disabled; 1 = Late data enabled.

Column	Data Type	Description
IdleDuration	int NOT NULL	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds.
ProcessingInterval	int NOT NULL	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration.

v_EventSnapshot

Returns one row for each snapshot value for an analog and/or discrete tag (specified by the TagName column) associated with a particular snapshot event (specified by the Event column).

Column	Data type	Description
Event	nvarchar(256) NULL	The unique name of the tag within the AVEVA Historian system.
EventTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectionTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the event system.
Edge	nvarchar(8) NOT NULL	The "edge" for the event detection. For more information on edge detection, see Edge Detection for Events (wwEdgeDetection) in the *AVEVA Historian Retrieval Guide*.
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float(8) NULL	The value of the tag at the time of the event occurrence. Measured in engineering units.
Quality	tinyint NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

Note: When an event is not linked to a snapshot action, the TagName column is set to '-' and the Value, Quality, and QualityDetail columns are set to NULL.

v_EventStringSnapshot

Returns one row for each snapshot value for a string tag (specified by the TagName column) associated with a particular snapshot event (specified by the Event column).

Column	data type	Description
Event	TagNameType(nvarchar(256)) NULL	The unique name of the tag within the AVEVA Historian system.
EventTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectionTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the event system.
Edge	nvarchar(8) NOT NULL	The "edge" for the event detection.
TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512) NULL	The value of the string tag at the event timestamp.
Quality	tinyint NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

v_ModTracking

Returns one row for each database modification made. For more information, see Viewing Database Modifications in the *AVEVA Historian Administration Guide.*

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The timestamp of when the modification occurred.
Table	varchar(50) NULL	The name of the modified object.

Column	Data type	Description
Column	nvarchar(128) NOT NULL	The name of the modified column.
ModType	char(1) NOT NULL	The type of modification. U = Update; I = Insert; D = Delete; 1 = SQL insert; 2 = SQL original insert; 3 = SQL update; 4 = CSV insert; 5 = CSV original insert; 6 = CSV update; 7 = CSV multi-point update; 8 = CSV "fast load" insert.
RowKey	sql_variant NOT NULL	The key identifier for the column modified in the table. For example, TagName for the Tag table, Name for the Topic table, and so on.
NewValue	sql_variant NULL	The new value stored in the column, if the modification was to a configuration table. For modifications to history data, this column contains the total count of consecutive value updates attempted.
OldValue	sql_variant NULL	The value stored in the column before the modification was made, if the modification was to a configuration table. For modifications to history data using SQL INSERT and UPDATE statements, this column contains the timestamp of the earliest data affected by the INSERT or UPDATE operation. If multiple changes are made to the same data, then only the most recent change will be contained in this column. This column is not used for modifications made to history data using a CSV file.
User	nvarchar(256) NOT NULL	The name of the database user that made the modification. The value of this column reflects the Windows authentication user name (for example, DOMAIN\user_login_name) or the SQL Server authentication user name (for example, dbo), depending on how the user is logged into the SQL Server when the modification is made. In the case of a CSV file import, this column contains the user name as it appears in the CSV file.

v_SnapshotData

Returns one row for each snapshot value for an analog, discrete, and/or string tag (specified by the TagName column) associated with a particular snapshot event (specified by the Event column).

Column	Data type	Description
Event	TagNameType(nvarchar(256)) NULL	The name of the event tag to which the snapshot tag is related.
EventTime	datetime2(7) NOT NULL	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian.
DetectionTime	datetime2(7) NOT NULL	The timestamp reflecting when the event was detected by the Event system.
Edge	nvarchar(8) NOT NULL	The "edge" for the event detection.
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512) NULL	The value of the snapshot tag at the event timestamp.
Quality	tinyint NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.

CHAPTER 4

Stored Procedures

Some stored procedures are useful when performing database queries to return information about specific tags in the system. These stored procedures allow you to return information on a tag's definition or to narrow the scope of a query on a data storage table. You can use these stored procedures when querying the database using ad-hoc query tools, such as SQL Server Management Studio.

Other stored procedures are used to configure AVEVA Historian. System stored procedures are normally run during startup and during changes to the system configuration. These stored procedures are used mainly by the historian setup program, the Event subsystem, the System Management Console, and client applications.

Note: Stored procedures prefixed with "ww_" are provided only for backward compatibility and are deprecated. For more information, see *Renamed Stored Procedures* on page 319.

Stored Procedures

aaActionStringSelect

Selects the action string for a specified event tag.

Syntax

aaActionStringSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddAnalogSummaryTag

Add an analog summary tag.

Syntax

aaAddAnalogSummaryTag *TagName, TagId, Description, SourceTag, SourceServer, SourceTagScaling, SourceTagRawType, SourceTagIntegerSize, SourceTagSignedInteger, CreatedBy, DateCreated, StructureId, AcquisitionType, StorageNodeKey, IOServerKey, TopicKey, StorageType, EngineeringUnit, IntegralDivisor, MinEU, MaxEU, MinRaw, MaxRaw, DeadbandType, TimeDeadband, CurrentEditor, wwTagKey, ChannelStatus*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
TagId	The unique tag ID of the tag within the AVEVA Historian system. The value is of data type uniqueidentifier, with a default of NULL.
Description	The description of the analog summary tag. This value is of data type nvarchar(512), with a default of an empty string.
SourceTag	The name of the source tag to create the analog summary tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
SourceTagScaling	Used to specify whether the value is scaled. 0 = Not scaled. 1 = scaled. This value is of data type int, with a default of NULL.
SourceTagRawType	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes) 2 = MS Float (4 bytes) 3 = Integer (2 or 4 bytes) 4 = MS Double (reserved for future use) (8 bytes) This value is of data type int, with a default of 3.
SourceTagIntegerSize	The bit size of the analog tag. 12 = 12-bit 15 = 15-bit 16 = 16-bit 32 = 32-bit 64 = 64-bit (reserved for future use) This value is of data type tinyint, with a default of 16.
SourceTagSignedInteger	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned 1 = Signed This value is of data type bit, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
StructureId	The unique identifier for the structure. The value is of data type uniqueidentifier, with a default of NULL.

Argument	Description
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
EngineeringUnit	The unit of measure. Examples are mph, grams, and pounds. This value is of data type nvarchar(32), with a default of an empty string.
IntegralDivisor	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. This value is of data type float(25), with a default of 1.
MinEU	The minimum value of the tag, measured in engineering units. This value is of data type float, with a default of 0.

Argument	Description
MaxEU	The maximum value of the tag, measured in engineering units. This value is of data type float, with a default of 100.
MinRaw	The minimum value of the raw acquired value. This value is of data type float, with a default of 0.
MaxRaw	The maximum value of the raw acquired value. This value is of data type float, with a default of 4095.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int.

Argument	Description
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationGroup

Add or modify replication groups.

Syntax

aaAddReplicationGroup *ReplicationGroupName, ReplicationServerName, ReplicationTypeKey, ReplicationScheduleName, SummaryReplicationNamingScheme, GroupAbbreviation, ReplicationGroupKey*

where:

Argument	Description
ReplicationGroupName	The name of the replication group. This parameter has to be specified, else will return an error. This value is of data type nvarchar(255), with no default.
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 3.
ReplicationScheduleName	The name of the schedule. This value is of data type nvarchar(255), with no default.

Argument	Description
SummaryReplicationNamingScheme	The naming scheme for summary replication tags. If not specified, the one specified in the ReplicationServer will be used. This value is of data type nvarchar(512), with a default of NULL.
GroupAbbreviation	Used as part of naming. If not specified, the one specified in the Schedule will be chosen as group abbreviation. This value is of data type nvarchar(32), with a default of NULL.
ReplicationGroupKey	Unique identifier for the replication group. If specified, this will overwrite the properties of the replication group. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationRule

Add or modify a replication rule.

Syntax

aaAddReplicationRule *Name, Priority, TagFilter, ReplicationGroupKey, ReplicationServerKey, Enable, ApplyOtherRules, AutoReplication*

where:

Argument	Description
Name	The name of the replication rule.
Priority	The priority for the rule.
TagFilter	The OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
ReplicationGroupKey	The unique identification for the replication group. ReplicationGroupKey is a foreign key from the Replication Group table.
ReplicationServerKey	The unique identifier for the replication server.
Enable	Used to indicate whether the replication rule is enabled. 0 - not enabled; 1- enabled
ApplyOtherRules	Used to indicate whether other rules apply. 0 - other rules do not apply; 1- other rules apply.
AutoReplication	Used to indicate whether autosummary is enabled. 0 - disabled; 1- enabled.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddReplicationSchedule

Add or modify the schedules for replication.

Syntax

aaAddReplicationSchedule *ReplicationScheduleName, ReplicationScheduleTypeName, ReplicationScheduleAbbreviation, CreateGroup, Period, Unit, TimesOfDay, ReplicationScheduleKey*

where:

Argument	Description
ReplicationScheduleName	The name of the schedule. This parameter is required. This value is of data type nvarchar(255), with no default.
ReplicationScheduleTypeName	The name of the schedule type. Can be either INTERVAL or CUSTOM. This value is of data type nvarchar(32), with a default of INTERVAL.
ReplicationScheduleAbbreviation	Will be used when creating groups as group abbreviation if not specified. This value is of data type nvarchar(32), with a default of the schedule abbreviation.
CreateGroup	If set to TRUE, groups will be created automatically when the replication server is created. This value is of data type bit, with a default of 1.
Period	The period value. This parameter is required when the schedule type is INTERVAL. This value is of data type smallint, with a default of 0.
Unit	The name of the unit. This parameter is required when the schedule type is INTERVAL. This value is of data type nvarchar(32), with a default of an empty string.
TimesOfDay	A semicolon-separated list of the times of day. This parameter is required when the schedule type is CUSTOM. This value is of data type nvarchar(max), with a default of an empty string.
ReplicationScheduleKey	The unique identifier for the schedule. If specified, this will overwrite the properties of the identified schedule. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationServer

Add or modify replication servers.

Syntax

aaAddReplicationServer *ReplicationServerName, Description, SFPath, SFFreeSpace, AuthenticateWithAAUser, UserName, Password, TCPPort, SummaryReplicationNamingScheme, SimpleReplicationNamingScheme, BufferCount, Bandwidth, MinSFDuration, ReplicationServerKey, CompressionEnabled, ConnectionDetails*

where:

Argument	Description
ReplicationServerName	Name or IP address of the tier 2 server. This value is of data type nvarchar(255), with a default of an empty string.
Description	Description of the replication server. This value is of data type nvarchar(512), with a default of an empty string.
SFPath	Store forward path for the replication server. The default is an empty string. This value is of data type nvarchar(260), with a default of an empty string.
SFFreeSpace	Free space for the store forward path in MB. This value is of data type int, with a default of 125.
AuthenticateWithAAUser	Set to 1 to authenticate with ArchestrA user. This value is of data type bit, with a default of 1.
UserName	User name for authenticating with tier 2 server. This value is nvarchar(255), with a default of NULL.
Password	Password for authenticating with tier 2 server. This value is of data type nvarchar(512), with a default of NULL.
TCPPort	TCP Port for communicating with tier 2 server. This value is of data type int, with a default of 32568.
SummaryReplicationNamingScheme	Naming rule for summary replication tags. If this is NULL, the naming rule specified in system parameters will be used. This value is of data type nvarchar(512), with a default of <*ReplicationDefaultPrefix*>.<*SourceTagName*>.<*TypeAbbreviation*><*GroupAbbreviation*>.
SimpleReplicationNamingScheme	Naming rule for simple replication tags. If this is NULL, the naming rule specified in System parameters will be used. This value is of data type nvarchar(512), with a default of <*ReplicationDefaultPrefix*>.<*SourceTagName*>.
BufferCount	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
Bandwidth	The bandwidth in kbps used between tier-1 and tier-2. -1 = unlimited. This value is of data type int.
MinSFDuration	The minimum duration, in seconds, for the replication service server node to function in store-and-forward mode. The replication service server node functions in store-and-forward mode for this length of time even if the condition that caused replication service server node to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds. This value is of data type int, with a default of 180.
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with a default of NULL.

Argument	Description
CompressionEnabled	0 = Compression off. 1 = Enable compression for the packets sent to the replication server.
ConnectionDetails	Internal use only.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationTagEntity

Add or modify a replication tag entity.

Syntax

aaAddReplicationTagEntity *SourceTagName, ReplicationGroupName, ReplicationServerName, ReplicationTypeKey, MaximumStates, CurrentEditor, DestinationTagId, DestinationTagName*

where:

Argument	Description
SourceTagName	The source tag name. This value is of data type nvarchar(256), with no default.
ReplicationGroupName	The name of the replication group. If this is NULL, the replication type is set to simple replication. This value is nvarchar(255), with a default of NULL.
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
ReplicationTypeKey	The type of replication. Valid values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 2.
MaximumStates	Maximum number of states to track for state summary tags. This value is of data type tinyint, with a default of 10 and a maximum of 100.
CurrentEditor	0 - Historian 2 - AVEVA Application Server This value is of data type tinyint, with a default of 0.
DestinationTagID	Unique ID of the destination tag. If NULL, the destination tag name is generated based on the naming rule. This value is of data type uniqueidentifier, with a default of NULL.
DestinationTagName	Name of the destination tag. If NULL, the destination tag name is generated based on the naming rule. This value is of data type nvarchar(256), with a default of NULL.

Argument	Description
ReplicationTagEntityKey	The unique identifier for the replication tag entity. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddReplicationTagExtendedProperty

Add or modify a replication tag extended property.

Syntax

aaAddReplicationTagExtendedProperty, *ReplicationServerKey, DestinationTagName, PropertyName, PropertyValue*

where:

Argument	Description
ReplicationServerKey	The unique identifier for the replication server.
DestinationTagName	Name of the destination tag. If NULL, the destination tag name is generated based on the naming rule. This value is of data type nvarchar(256), with a default of NULL.
PropertyName	The extended tag property name.
PropertyValue	The value of this replication tag extended property.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddStateSummaryTag

Add or modify a state summary tag.

Syntax

aaStateSummaryTag *TagName, TagId, Description, SourceTag, SourceServer, CreatedBy, DateCreated, AcquisitionType, StorageNodeKey, IOServerKey, TopicKey, StorageType, DeadbandType, TimeDeadband, CurrentEditor, wwTagKey, ChannelStatus*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
TagId	The unique tag ID of the tag within the AVEVA Historian system. The value is of data type uniqueidentifier, with a default of NULL.
Description	The description of the analog summary tag. This value is of data type nvarchar(512), with a default of an empty string.

Argument	Description
SourceTag	The name of the source tag to create the analog summary tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver. This value is of data type tinyint, with a default of 1.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. This value is of data type smallint, with a default of 1.

Argument	Description
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStorageLocation

Add or modify a storage location

Syntax

aaAddStorageLocation *ShardId, StorageType, Path, MaxMBSize, MinMBThreshold, MaxAgeThreshold*

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).
StorageType	The type of storage used for the specified location. 1 = Circular; 2 = Alternate; 3 = Buffer; 4 = Permanent. There can be only one storage location of each type.
Path	The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different internal hard drive. If you use a network location, then the ArchestrA user must have full access to the network location. The locations must be specified using UNC notation. Mapped drives are not supported. If empty, the default <SystemDataPath>\Wonderware\Data\Circular is used.
MaxMBSize	The limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used.
MinMBThreshold	The minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block (before any compression) by 1.5 to determine the minimum threshold.
MaxAgeThreshold	The age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStorageShard

Returns definition information for specified storage shard (partition).

Syntax

aaAddStorageShard *ShardId, ShardName, Description, ComputerName, BlockDuration, TimeUnitId, TimeZoneId, AdjustToDST, MaxSnapshotSize, CmdArgs, CmdExtArgs*

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).
ShardName	The name of the partition.
Description	The description of the partition.
ComputerName	The network name of the computer on which the storage partition resides.
BlockDuration	Duration, in hours, for history blocks. Valid values are: 1, 2, 3, 4, 6, 8, 12, 24. The default is 24 hours. The history block size must always be greater than the highest scan rate. For more information, see Managing Partitions and History Blocks in the *AVEVA Historian Administration Guide*.
TimeUnitId	Foreign key to TimeUnit. Indicates whether the block duration is in hours/days.
TimeZoneId	The time zone associated with this storage partition.
AdjustToDST	Internal use only.
MaxSnapshotSize	Maximum size, in MB, for data storage snapshots in memory. Bigger snapshots allow for faster retrieval. You might need to increase this size for systems with very high data rates. For example, if retrieval is slow from data in the current history block, try increasing this rate. Also be sure that you have enough RAM, up to 1 GB.
CmdArgs	Command line parameters for customizing replication and storage execution.
CmdExtArgs	Command line parameters for customizing replication and storage execution.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStorageShardAssignmentRule

Add or modify an assignment rule for a storage partition (shard).

Syntax

aaAddStorageShardAssignmentRule *Name, Priority, TagFilter, ShardId, Enabled*

where:

Argument	Description
Name	The name of the rule.
Priority	The priority for the rule.
TagFilter	The OData filters that will play a role in how the tags are assigned to partitions or how a tag is set for auto-summary.
ShardId	The unique identifier for the partition (shard).

Argument	Description
Enabled	Specifies whether the rule is enabled.

Permission

Execute permission defaults to the aaAdministrators group.

aaAddStructureTag

Add or modify a structure tag.

Syntax

aaStructureTag *TagName, TagId, Description, SourceTag, SourceServer, CreatedBy, DateCreated, StructureId, AcquisitionType, StorageNodeKey, IOServerKey, TopicKey, StorageType, EngineeringUnit, IntegralDivisor, MinEU, MaxEU, MinRaw, MaxRaw, DeadbandType, TimeDeadband, CurrentEditor, wwTagKey, ChannelStatus*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
TagId	The unique tag ID of the tag within the AVEVA Historian system. The value is of data type uniqueidentifier, with a default of NULL.
Description	The description of the analog summary tag. This value is of data type nvarchar(512), with a default of an empty string.
SourceTag	The name of the source tag to create the analog summary tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
StructureId	The ID for the structure. The value is of data type uniqueidentifier, with a default of NULL.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
EngineeringUnit	The unit of measure. Examples are mph, grams, and pounds. This value is of data type nvarchar(32), with a default of an empty string.
IntegralDivisor	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. This value is of data type float(25), with a default of 1.
MinEU	The minimum value of the tag, measured in engineering units. This value is of data type float, with a default of 0.
MaxEU	The maximum value of the tag, measured in engineering units. This value is of data type float, with a default of 100.
MinRaw	The minimum value of the raw acquired value. This value is of data type float, with a default of 0.
MaxRaw	The maximum value of the raw acquired value. This value is of data type float, with a default of 4095.

Argument	Description
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband; 2 = Rate (swinging door) deadband. This value is of data type smallint, with a default of 1.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAddTagExtendedProperty

Adds an extended property to a tag.

Syntax

aaAddTagExtendedProperty *TagName, PropertyName, PropertyValue*

where:

Argument	Description
TagName	The name of the tag. This value is of data type nvarchar(255), with no default.
PropertyName	The extended tag property name. This value is of data type nvarchar(255), with no default.
PropertyValue	The value to assign to the property. This value is of data type sql_variant, with no default.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers group.

aaAnalogDetail

Returns information about one or more specified analog tags, including the name of the tag, a description, the acquisition rate, the engineering unit, and the minimum and maximum values in engineering units.

Syntax

aaAnalogDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaAnalogTagDelete

Deletes an analog tag.

Syntax

aaAnalogTagDelete *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and **aaAdministrators** groups.

aaAnalogTagInsert

Inserts an analog tag.

Syntax

aaAnalogTagInsert *TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, EUKey, MinEU, MaxEU, MinRaw, MaxRaw, Scaling, RawType, ValueDeadband, InitialValue, IntegerSize, SignedInteger, TopicKey, IOServerKey, StorageNodeKey, AIRetrievalMode, SamplesInActiveImage, RateDeadband, InterpolationType, RolloverValue, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHistory*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS, or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.

Argument	Description
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. This value is of data type int, with a default of 10000.
ItemName	The address string of the tag. This value is of data type nvarchar(256), with a default of an empty string.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.

Argument	Description
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with a default of 1.
MinEU	The minimum value of the tag, measured in engineering units. This value is of data type float, with a default of 0.
MaxEU	The maximum value of the tag, measured in engineering units. This value is of data type float, with a default of 100.
MinRaw	The minimum value of the raw acquired value. This value is of data type float, with a default of 0.
MaxRaw	The maximum value of the raw acquired value. This value is of data type float, with a default of 4095.
Scaling	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None 1 = Linear 2 = Square Root (reserved for future use) This value is of data type int, with a default of 1.
RawType	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes) 2 = MS Float (4 bytes) 3 = Integer (2 or 4 bytes) 4 = MS Double (8 bytes; reserved for future use) This value is of data type int, with a default of 3.

Argument	Description
ValueDeadband	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. This value is of data type float, with a default of 0.
InitialValue	The initial value as imported from an external source (for example, from InTouch). This value is of data type float, with a default of 0.
IntegerSize	The bit size of the analog tag. 12 = 12-bit 15 = 15-bit 16 = 16-bit 32 = 32-bit 64 = 64-bit (reserved for future use) This value is of data type tinyint, with a default of 16.
SignedInteger	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned 1 = Signed This value is of data type bit, with a default of 0.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
AIRetrievalMode	Used to specify the behavior of retrieval for data in active image. You can either retrieve from all acquired data values that are currently in the active image, or only the data values that are configured to be stored on disk. Data on disk may be a subset of that in the active image, depending on the storage rate for the tag.Valid values are: 0 = All of the values received into the active image will be included in the returned data (default); 1 = Only the values that will be moved into storage will be included in the returned data. This value is of data type tinyint, with a default of 0.
SamplesInActiveImage	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.

Argument	Description
RateDeadband	Used to percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. This value is of data type float, with a default of 0.
InterpolationType	The interpolation type for retrieval. 0 = Stair-stepped interpolation 1 = Linear interpolation (if applicable, based on the tag type) 254 = System default interpolation mode The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes. This value is of data type tinyint, with a default of 254.
RolloverValue	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000. This value is of data type int, with a default of 0.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the Storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
SourceTag	The name of the source tag to create the tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.

Argument	Description
AITag	Used to specify whether the tag's values are stored by the Classic Storage subsystem. 0 = Not stored by the Classic Storage subsystem 1 = Stored by the Classic Storage subsystem This value is of data type bit, with a default of 1.
TagId	The unique identifier for the tag. The value is of data type uniqueidentifier, with a default of NULL.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.
AIHistory	Used to specify whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage. 1 = The tag may have data previously collected by classic storage. This value is of data type bit, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAnalogTagSelect

Selects an analog tag.

Syntax

aaAnalogTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaAnalogTagUpdate

Updates an analog tag.

Syntax

aaAnalogTagUpdate *wwTagKey, TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, EUKey, MinEU, MaxEU, MinRaw, MaxRaw, Scaling, RawType, ValueDeadband, InitialValue, IntegerSize, SignedInteger, TopicKey, IOServerKey, StorageNodeKey, AIRetrievalMode, SamplesInActiveImage, RateDeadband, InterpolationType, RolloverValue, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHistory*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as the *aaAnalogTagInsert* on page 185 stored procedure. However, only these arguments have defaults:

- *Description*
- *AcquisitionType*
- *ItemName*
- *CreatedBy*
- *DateCreated*
- *CurrentEditor*
- *SamplesInActiveImage*
- *RateDeadband*
- *InterpolationType*
- *RolloverValue*
- *ServerTimeStamp*
- *DeadbandType*
- *SourceTag*
- *SourceServer*
- *AITag*
- *TagId*
- *ChannelStatus*
- *AIHistory*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaAnnotationDelete

Deletes an annotation.

Syntax

aaAnnotationDelete *AnnotationKey*

where:

Argument	Description
AnnotationKey	The unique numerical identifier for the annotation. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaUsers, aaPowerUsers, and aaAdministrators groups.

aaAnnotationInsert

Inserts an annotation.

Syntax

aaAnnotationInsert *TagName, UserKey, DateTime, DateCreated, Content, Value*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
UserKey	The unique numerical identifier for a database user as defined in the UserDetail table. This value is of data type int, with a default of NULL.
DateTime	The timestamp of the tag value for which the user has made an annotation. This value is of data type datetime2(7), with a default of NULL.
DateCreated	The date that the annotation was created. This value is of data type datetime2(7), with a default of NULL.
Content	The annotation text. This value is of data type nvarchar(1000), with a default of "Annotation."
Value	The value of the tag at the time of the annotation. This value is of data type real, with a default of 0.0.

Permission

Execute permission defaults to the aaUsers, aaPowerUsers, and aaAdministrators groups.

aaAnnotationRetrieve

Retrieves one or more annotations.

Syntax

aaAnnotationRetrieve *TagList, StartTime, EndTime*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the data to query. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the data to query. This value is of data type nvarchar(50), with no default.

Permission

Execute permission defaults to the public group.

aaAnnotationSelect

Selects an annotation.

Syntax

aaAnnotationSelect *AnnotationKey*

where:

Argument	Description
AnnotationKey	The unique numerical identifier for the annotation. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaAnnotationUpdate

Updates an annotation.

Syntax

aaAnnotationUpdate *AnnotationKey, TagName, UserKey, DateTime, Content, Value*

where:

Argument	Description
AnnotationKey	The unique numerical identifier for the annotation. This value is of data type int, with no default.

The remaining arguments are similar to those for the *aaAnnotationInsert* on page 192 stored procedure.

Permission

Execute permission defaults to the aaUsers, aaPowerUsers, and aaAdministrators groups.

aaArchestrANSClear

Removes all ArchestrA entries from the public namespace.

Syntax

aaArchestrANSClear

Remarks

In general, using this stored procedure is not recommended. If you need to remove the ArchestrA entries because of a namespace corruption, contact Technical Support for guidance.

Permission

Execute permission defaults to the aaAdministrators group.

aaCheckChartConfigurationNameExists

Checks to see if a configuration name exists for some InSight content.

Syntax

aaCheckChartConfigurationNameExists *ChartConfigurationName*

where:

Argument	Description
ChartConfigurationName	The name of the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaCleanupAfterCommit

Runs once after reinitialization or system startup is complete.

Syntax

aaCleanupAfterCommit

Remarks

This stored procedure does two things:

1. Sets the DbStatus column of the StorageNode table to 0.
2. Deletes the contents of the ConfigStatusSnapshot table.

Permission

Execute permission defaults to the aaAdministrators group.

aaCleanupSystemNotRunning

Runs once whether or not reinitialization or system startup is complete. For internal use only.

Syntax

aaCleanupSystemNotRunning

Remarks

This stored procedure does two things:

1. Sets the DbStatus column of the StorageNode table to 0.
2. Deletes information from the ConfigStatusSnapshot table.

Permission

Execute permission defaults to the aaAdministrators group.

aaClearDeletedTags

Used by storage for handling deleted tags. Internal use only.

Syntax

aaClearDeletedTags *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaClearDeletedReplicationTagEntities

Used by storage for handling deleted replication tag entities. Internal use only.

Syntax

aaClearDeleteReplicationTagEntities *TagList*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaCommitAllowed

Used to allow a reinitialization of the system.

Syntax

aaCommitAllowed allowCommit

where:

Argument	Description
allowCommit	Specifies the change. *<Is this an ID or name? Which table does it come from?>*

Permission

Execute permission defaults to the public group.

aaCommitChanges

Used to trigger a reinitialization of the system.

Syntax

aaCommitChanges

Remarks

This stored procedure performs the following if a change is made:

1. Copies the contents of the ConfigStatusPending table to the ConfigStatusSnapshot table.
2. Resets the Status column in the applicable database table (Tag, Topic, IOServer, IODriver, StorageLocation, SnapshotDetail, or SystemParameter) to 0.
3. Deletes the contents of the ConfigStatusPending table.

Permission

Execute permission defaults to the aaAdministrators group.

aaCommitChangesAtStartup

Used to specify a reinitialization of the system at startup.

Syntax

aaCommitChangesAtStartup

Remarks

This stored procedure is used only when a modification to a storage location has been made. The functionality of this stored procedure is similar to that of **aaCommitChanges**.

Permission

Execute permission defaults to the aaAdministrators group.

aaContextDelete

Deletes a context.

Syntax

aaContextDelete *ContextKey*

where:

Argument	Description
ContextKey	The unique numerical identifier for the context. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers, and aaAdministrators groups.

aaContextInsert

Inserts a context.

Syntax

aaContextInsert *Description*

where:

Argument	Description
Description	The description of the context. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers, and aaAdministrators groups.

aaContextSelect

Selects a context.

Syntax

aaContextSelect *ContextKey*

where:

Argument	Description
ContextKey	The unique numerical identifier for the context. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaContextUpdate

Updates a context.

Syntax

aaContextUpdate *ContextKey, Description*

where:

Argument	Description
ContextKey	The unique numerical identifier for the context. This value is of data type int, with no default.
Description	The description of the context. This value is of data type nvarchar(50), with no default.

Permission

Execute permission defaults to the aaPowerUsers, and aaAdministrators groups.

CreateReplicationServerDefaultGroups

Used to create default replication server groups.

Syntax

CreateReplicationServerDefaultGroups *ReplicationServerKey*

where:

Argument	Description
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and aaAdministrators groups.

CreateReplicationServerSystemTags

Creates replication server default groups if the CreateGroups setting is set to true. Internal use only.

Syntax

CreateReplicationServerSystemTags *ReplicationServerKey*

where:

Argument	Description
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and aaAdministrators groups.

aaDBChangesPending

Returns a list of modifications pending, from the ConfigStatusPending table, in a readable format.

Syntax

aaDBChangesPending

Permission

Execute permission defaults to the public group.

aaDBConfig

Returns a summary of the current database configuration, such as number of tags, number of tags per type, storage configuration, event tags, summary configuration, and so on.

Syntax

aaDBConfig

Permission

Execute permission defaults to the public group.

aaDeleteChartConfiguration

Deletes a chart configuration.

Syntax

aaDeleteChartConfiguration *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaDeleteComment

Deletes a comment (annotation).

Syntax

aaDeleteComment *CommentsKey*

where:

Argument	Description
CommentsKey	The unique identifier for the comment.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaDeleteOlderEvents

Deletes old events from event storage.

Syntax

aaDeleteOlderEvents

Remarks

This stored procedure is executed by the *aaSpaceManager* on page 262 stored procedure every ten minutes. The duration for which events are kept is stored in the *SystemParameter* on page 127 table. Events will be deleted from the *EventHistory* on page 76 table.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteOlderSummaries

Deletes old summaries from summary storage.

Syntax

aaDeleteOlderSummaries

Remarks

This stored procedure is executed by the *aaSpaceManager* on page 262 stored procedure every ten minutes. The duration for which summaries are kept is stored in the *SystemParameter* on page 127 table. Summaries will be deleted from the *SummaryHistory* on page 124 table.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteReplicationGroup

Deletes an existing replication group. If the group being deleted is referenced by a replication tag entity, the procedure returns an error message.

Syntax

```
aaDeleteReplicationGroup ReplicationGroupName, ReplicationServerName,
ReplicationTypeKey
```

where:

Argument	Description
ReplicationGroupName	The name of the group. This parameter is required. This value is of data type nvarchar(255), with no default.
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.

Argument	Description
ReplicationTypeKey	The type of replication. Valid values are: 1 = Simple Replication 2 = Analog Summary Replication 3 = State Summary Replication This value is of data type tinyint, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteReplicationSchedule

Deletes a replication schedule.

Syntax

aaDeleteReplicationSchedule *ReplicationScheduleName*

where:

Argument	Description
ReplicationScheduleName	The name of the schedule. This parameter is required. This value is of data type nvarchar(255), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteReplicationServer

Deletes an existing replication server. If the server being deleted is referenced by a replication group, the procedure returns an error message.

Syntax

aaDeleteReplicationServer *ReplicationServerName*

where:

Argument	Description
ReplicationServerName	The name of the replication server. This parameter is required. This value is of data type nvarchar(255), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

DeleteReplicationServerSystemTags

Deletes replication server default groups if the DeleteGroups setting is set to true. Internal use only.

Syntax

DeleteReplicationServerSystemTags *ReplicationServerKey*

where:

Argument	Description
ReplicationServerKey	Unique identifier for the replication server. If specified, this will overwrite the properties of the server identified by the key. This value is of data type int, with no default.

Permission

Execute permission defaults to the **aaPowerUsers** and aaAdministrators groups.

aaDeleteReplicationTagEntity

Deletes an existing replication entity from a tier 1 server.

Syntax

aaDeleteReplicationTagEntity *ReplicationServerName, DestinationTagName*

where:

Argument	Description
ReplicationServerName	The name of the replication server. This value is of data type nvarchar(255), with no default.
DestinationTagName	The name of the destination tag. This value is of data type nvarchar(255), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDeleteTag

Deletes a tag from the database.

Syntax

aaDeleteTag *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaDeleteTagExtendedProperty

Deletes a tag extended property.

Syntax

aaDeleteTagExtendedProperty *TagName, PropertyName*

where:

Argument	Description
TagName	The name of the tag.
PropertyName	The extended tag property name.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers group.

aaDetectorStringSelect

Selects the detector string for a specified event tag.

Syntax

aaDetectorStringSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaDiscreteDetail

Returns information about one or more specified discrete tags, including the name of the tag, a description, the message for the TRUE (1) state of the tag, and the message for the FALSE (0) state of the tag.

Syntax

aaDescreteDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaDiscreteTagDelete

Deletes a discrete tag.

Syntax

aaDiscreteTagDelete *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaDiscreteTagInsert

Inserts a discrete tag.

Syntax

aaDiscreteTagInsert *TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, MessageKey, InitialValue, TopicKey, IOServerKey, AIRetrievalMode, SamplesInActiveImage, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.

Argument	Description
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. This value is of data type int, with a default of 0.
ItemName	The address string of the tag. This value is of data type nvarchar(256), with a default of an empty string.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.

Argument	Description
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with a default of 1.
InitialValue	The initial value as imported from an external source (for example, from InTouch). This value is of data type tinyint, with a default of 0.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
AIRetrievalMode	Used to specify the behavior of retrieval for data in active image. You can either retrieve from all acquired data values that are currently in the active image, or only the data values that are configured to be stored on disk. Data on disk may be a subset of that in the active image, depending on the storage rate for the tag.Valid values are: 0 = All of the values received into the active image will be included in the returned data (default). 1 = Only the values that will be moved into storage will be included in the returned data. This value is of data type tinyint, with a default of 0.The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.

Argument	Description
SamplesInActiveImage	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.
SourceTag	The name of the source tag to create the tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
AITag	Used to specify whether the tag's values are stored by the classic storage subsystem. 0 = Not stored by the Classic Storage subsystem; 1 = Stored by the Classic Storage subsystem. This value is of data type bit, with a default of 1.
TagId	The unique identifier for the tag. The value is of data type uniqueidentifier, with a default of NULL.

Argument	Description
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.
AIHistory	Used to specify whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage. 1 = The tag may have data previously collected by classic storage. This value is of data type bit, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaDiscreteTagSelect

Selects a discrete tag.

Syntax

aaDiscreteTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaDiscreteTagUpdate

Updates a discrete tag.

Syntax

aaDiscreteTagUpdate *wwTagKey, TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, CurrentEditor, MessageKey, InitialValue, TopicKey, IOServerKey, AIRetrievalMode, SamplesInActiveImage, ServerTimeStamp, DeadbandType, ServerTimeStamp, DeadbandType, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaDiscreteTagInsert* on page 204 stored procedure. However, only these arguments have defaults:

- *Description*
- *AcquisitionType*
- *ItemName*
- *CreatedBy*
- *DateCreated*
- *SamplesInActiveImage*
- *ServerTimeStamp*
- *DeadbandType*
- *SourceTag*
- *SourceServer*
- *AITag*
- *TagId*
- *ChannelStatus*
- *AIHIstory*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEngineeringUnitDelete

Deletes an engineering unit.

Syntax

aaEngineeringUnitDelete *EUKey*

where:

Argument	Description
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEngineeringUnitInsert

Inserts an engineering unit.

Syntax

aaEngineeringUnitInsert *Unit, DefaultTagRate, IntegralDivisor*

where:

Argument	Description
Unit	The unit of measure. Examples are mph, grams, and pounds. This value is of data type nvarchar(32), with no default.
DefaultTagRate	The default rate, in milliseconds, at which tags are cyclically stored, based on engineering units. Although the system does not make use of this engineering unit based tag rate, you can reference this value in custom SQL scripts. The value you enter for this tag rate does not affect the default storage rate set for the tag. This value is of data type int, with a default of 10000.
IntegralDivisor	The factor to be applied when integrating a rate with the units [EngUnits/TimeUnit] to a quantity with units [EngUnits]. This factor is called the integral divisor. The default value of 1 assumes a time unit of seconds and ensures that a rate of [Unit/second] is correctly integrated to [Unit]. For a time unit of minutes, set the integral divisor value to 60; for a unit of hours, set the integral divisor value to 3600. The integral divisor is applied similarly to rates or quantities that are not expressed in terms of a time unit. For example, to convert watts to watt-hours, the integral divisor is 1/3600. To convert watts to kilowatt-hours, the integral divisor is 1/3600000. This value is of data type float(25), with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEngineeringUnitSelect

Selects an engineering unit.

Syntax

aaEngineeringUnitSelect *EUKey*

where:

Argument	Description
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaEngineeringUnitUpdate

Updates an engineering unit.

Syntax

aaEngineeringUnitUpdate *EUKey, Unit, DefaultTagRate, IntegralDivisor*

where:

Argument	Description
EUKey	The unique numerical identifier of an engineering unit. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaEngineeringUnitInsert* on page 209 stored procedure. However, only the *IntegralDivisor* argument has a default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEventDetection

Detects the number of events in history in which the data value for the specified tag matched the criteria defined by the remaining arguments. This stored procedure is used by the event subsystem and should not be modified.

Syntax

aaEventDetection *TagName, Operator, DetectValue, Edge, Resolution, StartTime, EndTime*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Operator	The comparison operator. Valid values are: • >= • < • <= • = • <> This value is of data type char(2), with no default.
DetectValue	The value against which the stored values for the tag are compared to determine if the event occurred. This value is of data type float(25), with a default of none.

Argument	Description
Edge	The type of edge detection result set that the query will return. Valid values are: • NONE • LEADING • TRAILING • BOTH This value is of data type char(8), with no default.
Resolution	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date. This value is of data type int, with no default.
StartTime	The starting timestamp for the data to query. This value is of datatype varchar(30), with no default.
EndTime	The ending timestamp for the data to query. This value is of datatype varchar(30), with no default.

Remarks

You can apply a resolution only if you set the value of the Edge argument to NONE.

Permission

Execute permission defaults to the aaAdministrators group.

aaEventHistoryInsert

Inserts a row into the EventHistory table for each occurrence of an event for a specified event tag. This stored procedure is used by the event subsystem and should not be modified.

Syntax

aaEventHistoryInsert *TagName, DateTime, DetectDateTime, Edge*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
DateTime	The timestamp reflecting when event history data was acquired. This is the time when an event actually occurred. This time reflects the time zone of the AVEVA Historian. This value is of data type datetime2(7), with no default.
DetectDateTime	The timestamp reflecting when the event was detected by the event system. This value is of data type datetime2(7), with no default.

Argument	Description
Edge	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaEventHistorySelect

Returns information stored in the EventHistory table for each specified event tag.

Syntax

aaEventHistorySelect *TagList, StartTime, EndTime*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the data to query. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the data to query. This value is of data type nvarchar(50), with no default.

Remarks

This stored procedure will return information for all events that occurred between the starting time and the ending time.

Permission

Execute permission defaults to the public group.

aaEventSnapshotInsert

Inserts snapshot values into the AnalogSnapshot, DiscreteSnapshot, and StringSnapshot tables. This stored procedure is used by the event subsystem and should not be modified.

Syntax

```
aaEventSnapshotInsert EventLogKey, EventTime, EventTagName
```

Arguments

EventLogKey
The unique numerical identifier of an event occurrence. This value is of data type int, with no default.

EventTime
The timestamp reflecting when the event history data was acquired. This is the time for when the event actually occurred. This value is of data type datetime2(7), with no default.

EventTagName
The name of the event tag to which the snapshot tag is related. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaEventSnapshotSelect

Returns the snapshot tag values for each of the event tags specified by the tag list.

Syntax

aaEventSnapshot *TagList, StartTime, EndTime, OrderBy*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the data to query. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the data to query. This value is of data type nvarchar(50), with no default.
OrderBy	The name of the column in the v_EventSnapshot view used to order the rows in the result set. The value is of data type nvarchar(2000), with a default of 'Event'.

Remarks

This stored procedure will return information for all events that occurred between the starting time and the ending time.

This stored procedure does not work with string snapshots.

Permission

Execute permission defaults to the public group.

aaEventTagDelete

Deletes an event tag.

Syntax

aaEventTagDelete *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEventTagDetail

Returns the details for a specified event tag, including time detector information, if applicable.

Syntax

aaEventTagDetail *TagList*

where:

Argument	Description
TagList	The list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with a default of '%'.

Permission

Execute permission defaults to the public group.

aaEventTagInsert

Inserts an event tag.

Syntax

aaEventTagInsert *TagName, Description, CreatedBy, DateCreated, CurrentEditor, ScanRate, TimeDeadband, Logged, Status, PostDetectorDelay, UseThreadPool, DetectorTypeKey, DetectorString, ActionTypeKey, ActionString, Priority, Edge*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.

Argument	Description
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian 1 = InTouch 2 = AVEVA Application Server This value is of data type int, with a default of 0.
ScanRate	The interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms). This value is of data type int, with a default of 0.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
Logged	Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured. 0 = Not logged 1 = Logged This value is of data type bit, with a default of 0.

Argument	Description
Status	The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system. This value is of data type tinyint, with a default of 0.
PostDetectorDelay	The amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed. This value is of data type int, with a default of 0.
UseThreadPool	To specify how system threads are used to process events. 1 = All events are handled by a single thread and a single logon to the SQL Server; 0 = Each event uses a separate system thread and logon. This will allow the Event subsystem to manage the scan rates of each detector component concurrently. (Reserved for future use.) This value is of data type bit, with a default of 1.
DetectorTypeKey	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System 2 = External event 3 = Generic SQL 4 = Analog specific value 5 = Discrete specific value 6 = Time-based (schedule) This value is of data type int, with a default of 0.
DetectorString	The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian. This value is of data type nvarchar(1500), with a default of NULL.

Argument	Description
ActionTypeKey	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action 2 = Generic SQL 3 = Snapshot 4 = E-mail 5 = Deadband 6 = Summary This value is of data type int, with a default of 0.
ActionString	The script that specifies the event action. Action scripts run on the local AVEVA Historian. This value is of data type nvarchar(1500), with a default of NULL.
Priority	The priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. This value is of data type tinyint, with a default of 0.
Edge	The "edge" for the event detection. 0 = Trailing 1 = Leading 2 = Both 3 = None 4 = Time Detector 5 = External Detector This value is of data type tinyint, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaEventTagSelect

Selects an event tag.

Syntax

aaEventTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaEventTagSelectAll

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectAll

Permission

Execute permission defaults to the public group.

aaEventTagSelectDeleted

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectDeleted

Permission

Execute permission defaults to the public group.

aaEventTagSelectDisabled

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectDisabled

Permission

Execute permission defaults to the public group.

aaEventTagSelectInserted

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectInserted

Permission

Execute permission defaults to the public group.

aaEventTagSelectUpdated

Used by the event system to determine changes for dynamic reinitialization.

Syntax

aaEventTagSelectUpdated

Permission

Execute permission defaults to the public group.

aaEventTagUpdate

Updates an event tag.

Syntax

aaEventTagUpdate *wwTagKey, TagName, Description, CreatedBy, DateCreated, CurrentEditor, ScanRate, TimeDeadband, Logged, Status, PostDetectorDelay, UseThreadPool, DetectorTypeKey, DetectorString, ActionTypeKey, ActionString, Priority, Edge*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaEventTagInsert* on page 215 stored procedure. However, these arguments do not have defaults:

- wwTagKey
- TagName
- Description
- DateCreated
- DetectorString
- ActionString

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaGetAnalogSummaryTags

Returns all the properties for the specified analog summary tag or if you don't specify a tag, returns this info for them all.

Syntax

aaGetAnalogSummaryTags *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the public group.

aaGetChangeNotification

Used internally to manage configuration changes for a tag. Internal use only.

Syntax

aaGetChangeNotification *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaGetChartConfigurationLayout

Returns chart layout information for specific InSight content.

Syntax

aaGetChartConfigurationLayout *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetChartConfigurations

Returns chart configuration information for specific InSight content.

Syntax

aaGetChartConfigurations *ChartConfigurationName, ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationName	The name of the InSight content.
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetChartConfigurationsForDashboard

Returns definition information for each specified tag.

Syntax

aaGetChartConfigurationsForDashboard *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The web address for the InSight content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetChartConfigurationsForKeywords

Returns keyword associated with specific InSight content.

Syntax

aaTagInfo *TagList*

where:

Argument	Description
Keyword	A list of keywords associated with the content.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetDbRevision

Used to determine the current revision number of the database.

Syntax

aaGetDbRevision

Permission

Execute permission defaults to the public group.

aaGetDeletedReplicationTagEntities

Returns a list of deleted replication tag entities.

Syntax

aaGetDeletedReplicationTagEntities *ChangeVersion*

where:

Argument	Description
ChangeVersion	Specifies the change version. Internal use only.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaGetDeletedTags

Used by storage to handle deleted tags. Internal use only.

Syntax

aaGetDeletedTags *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaGetHistorianPartners

Returns the name of the partner historian. Internal use only.

Syntax

aaGetHistorianPartners

Permission

Execute permission defaults to the aaAdministrators group.

aaGetLastTagKey

Returns the details for the last inserted tag.

Syntax

aaGetLastTagKey *TagType*

where:

Argument	Description
TagType	The type of tag. 1 = Analog 2 = Discrete

Argument	Description
	3 = String 4 = Complex 5 = Event 7 = Summary tag (analog or state) This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaGetReplicationGroups

Returns the groups configured in the Historian database for a given replication server and type. If you specify all the parameters, then the specific group identified by the parameters is returned.

Syntax

aaGetReplicationGroups *ConfigurationToReturn, ReplicationServerName, ReplicationTypeKey, ReplicationGroupName, IncludeEmptyGroups, ReplicationGroupKey, RowsToReturn*

where:

Argument	Description
ConfigurationToReturn	The return configuration for the replication service.This value is of data type tinyint, with a default of 1.
ReplicationServerName	The name of the replication server. This value is nvarchar(255), with a default of NULL.
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 2.
ReplicationGroupName	The name of the replication group. This value is nvarchar(255), with a default of NULL.
IncludeEmptyGroups	Bit that specifies whether to include empty groups in the return. This value is of data type bit, with a default of 0.
ReplicationGroupKey	Unique identifier for the replication group. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationNamingParameters

Returns the naming parameters for the specified replication type in the specified replication group.

Syntax

aaGetReplicationNamingParameters *ReplicationTypeKey, ReplicationGroupKey*

where:

Argument	Description
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 3.
ReplicationGroupKey	Unique identifier for the replication group. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaGetReplicationRule

Returns details about a replication rule

Syntax

aaGetReplicationRule *Name, AutoReplication*

where:

Argument	Description
Name	The name of the replication rule.
AutoReplication	Specifies whether this rule is used for autosummary replication. 0 - not used for autosummary replication; 1 - used for autosummary replication.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetReplicationSchedules

Returns the schedules configured in the Historian database.

Syntax

aaGetReplicationSchedules *ConfigurationToReturn, ReplicationScheduleName, ReplicationScheduleKey, RowsToReturn*

where:

Argument	Description
ConfigurationToReturn	The return configuration for the replication schedule. This value is of data type tinyint, with a default of 0.
ReplicationScheduleName	The name of the schedule. This value is nvarchar(255), with a default of NULL.
ReplicationScheduleKey	The unique identifier for the schedule. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationServers

Returns the configured replication servers in the database. If the server name is specified, then it will return only the properties of the server identified by the name.

Syntax

aaGetReplicationServers *ConfigurationToReturn, ReplicationServerName, ReplicationServerKey, RowsToReturn*

where:

Argument	Description
ConfigurationToReturn	Returns the configuration for the replication service. This value is of data type tinyint, with a default of 0.
ReplicationServerName	The name of the server. This value is nvarchar(255), with a default of NULL.
ReplicationServerKey	The unique identifier for the server. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationShard

Returns details about a replication shard.

Syntax

aaGetReplicationShard *ShardId*

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).

Permission

Execute permission defaults to the aaAdministrators group.

aaGetReplicationTagEntities

Returns the replication entities configured in the Historian database. This procedure will return the properties of the replication tag entity based on the following parameter order:

- If a ReplicationTagEntityKey is specified, then the specific entity properties are returned.
- If a SourceTagName is specified, then all the entities with the specific SourceTagName are returned.
- If the Replication Server and GroupName are not specified, then all the entities belonging to the specific replication type are returned.
- If the Replication Server and GroupName and type are specified, then all the entities belonging to the specific group and type are returned.

Syntax

aaGetReplicationTagEntities *ConfigurationToReturn, ReplicationServerName, ReplicationGroupName, ReplicationTypeKey, SourceTagName, ReplicationTagEntityKey, RowsToReturn, FetchModified*

where:

Argument	Description
ConfigurationToReturn	The return configuration for the replication entities. This value is of data type tinyint, with a default of 1.
ReplicationServerName	The name of the server. This value is nvarchar(255), with a default of NULL.
ReplicationGroupName	The name of the replication group. This value is nvarchar(255), with a default of NULL.
ReplicationTypeKey	The type of replication. Value values are: 1 - Simple Replication 2 - Analog Summary Replication 3 - State Summary Replication This value is of data type tinyint, with a default of 2.

Argument	Description
SourceTagName	The name of the source tag. This value is of data type nvarchar(256), with a default of NULL.
ReplicationTagEntityKey	The unique identifier for the replication tag entity. This value is of data type int, with a default of NULL.
RowsToReturn	The number of rows to return. This value is of data type int, with a default of 3.
FetchModified	Returns requested entities. This value is of data type bit. Valid values are: 1 = Fetch only modified entities 0 = Fetch all

Permission

Execute permission defaults to the **aaUsers**, **aaPowerUsers**, and aaAdministrators groups.

aaGetReplicationTags

Returns the specified replication tag.

Syntax

aaGetReplicationTags *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaGetRowCount

Internal use only.

Syntax

aaGetRowCount *TableName*

where:

Argument	Description
TableName	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaGetStateSummaryTags

Returns the specified state summary tag.

Syntax

aaGetStateSummaryTags *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.

Permission

Execute permission defaults to the public group.

aaGetStorageShard

Returns a storage shard configuration.

Syntax

aaGetStorageShard *ShardId, ConfigurationToReturn*

where:

Argument	Description
ShardId	The unique identifier for the partition (shard).
ConfigurationToReturn	Specifies which storage shard configuration to return.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetStorageShardAssignmentRule

Returns a storage shard assignment rule.

Syntax

aaGetStorageShardAssignmentRule *Name*

where:

Argument	Description
Name	The name of the assignment rule.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetTagExtendedProperties

Retrieves all tag extended properties.

Syntax

aaGetTagExtendedProperties *ChangeVersion*

where:

Argument	Description
ChangeVersion	Internal use only.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaGetUserKey

Internal use only.

Syntax

aaGetUserKey

aaHistorianConfigNSExpand

Expands the tree view under an AVEVA Historian in the namespace. This stored procedure is used by the Configuration Editor component of the System Management Console and should not be modified.

Syntax

aaHistorianConfigNSExpand *PKey*

where:

Argument	Description
PKey	A local variable used to identify the AVEVA Historian in the namespace. This value is of data type int, with no default.

Remarks

An object can have one or more objects below it in the namespace hierarchy.

Permission

Execute permission defaults to the public group.

aaHistorianNSExpand

Expands the tree view under an AVEVA Historian in the namespace.

Syntax

aaHistorianNSExpand *PKey*

where:

Argument	Description
PKey	A local variable used to identify the AVEVA Historian in the namespace. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaHistorianStatusSelect

Used to select the value of the database status flag, DbStatus.

Syntax

aaHistorianStatusSelect

Remarks

This stored procedure is used by the System Management Console to determine the state of a database modification.

Permission

Execute permission defaults to the public group.

aaHistorianStatusSet

Sets the value of the status flag, DbStatus, to a value greater than 0 when a database modification needs to be processed by the server (back end). Sets the value of DbStatus back to 0 when a database modification is complete.

Syntax

aaHistorianStatusSet *DbStatus, Acquisition, Storage, DBServer*

where:

Argument	Description
DbStatus	For releases prior to 8.0, used to store the status of server reinitializations. 2 = Certain columns in the Tag, AnalogTag, DiscreteTag, StringTag, Topic, and IOServer tables were changed 3 = Reinitialization needed. 4 = Commit phase of a database update is in progress. 0 = Reinitialization complete. A negative value indicates that an error was encountered during reinitialization.

Argument	Description
	This value is of data type int, with no default.
Acquisition	Used with DbStatus to indicate to the back end whether the acquisition subsystem needs to be restarted. 0 = Restart not needed. 1 = Restart needed. Currently not used. This value is of data type int, with a default of 0.
Storage	Used with DbStatus to indicate to the back end whether the Storage subsystem needs to be restarted. 0 = Restart not needed. 1 = Restart needed. Currently not used. This value is of data type int, with a default of 0.
DBServer	Used with DbStatus to indicate to the back end whether the database server needs to be restarted. 0 = Restart not needed. 1 = Restart needed. Currently not used. This value is of data type int, with a default of 0.

Note: Only the first argument is required; the others are used to indicate that a specific subsystem needs to be initialized.

Remarks

When a change is made to the Runtime database configuration using the System Management Console, the value of the DbStatus attribute in the StorageNode table is set to a value greater than 0, meaning that modifications are outstanding and a reinitialization has yet to occur. The System Management Console, after detecting that a change is outstanding based on the value of DbStatus, makes the required changes, reinitializes the AVEVA Historian, if necessary, and then sets the value of DbStatus back to 0, meaning that reinitialization has been completed.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaHistoryBlockSelect

Returns the list of history blocks for the selected time period. If no arguments are passed, the complete list is returned.

Syntax

aaHistoryBlockSelect *FromDate, ToDate*

where:

Argument	Description
FromDate	The starting timestamp for the history block. This value is of data type datetime2(7), with a default of NULL.

Argument	Description
ToDate	The ending timestamp for the history block. This value is of data type datetime2(7), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaInTouchNodeTagList

Used by the System Management Console to display a list of imported tags for an InTouch node.

Syntax

aaInTouchNodeTagList *NodeKey, FilterStr*

where:

Argument	Description
NodeKey	The unique numerical identifier of the named InTouch node. This value is of data type int, with a default of NULL.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Remarks

This stored procedure returns the AVEVA Historian tagname, the original InTouch tagname, and the InTouch tag type (for example, memory integer).

Permission

Execute permission defaults to the public group.

aaIODriverDelete

Deletes an IDAS.

Syntax

aaIODriverDelete *IODriverKey*

where:

Argument	Description
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaIODriverInsert

Inserts an IDAS.

Syntax

aaIODriverInsert *StorageNodeKey, ComputerName, StoreForwardMode, StoreForwardPath, MinMBThreshold, AltComputerName, Enabled, StoreForwardDuration, AutonomousStartupTimeout, BufferCount, FileChunkSize, ForwardingDelay, ConnectionTimeout*

where:

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
ComputerName	The name of the computer on which the IDAS runs. This value is of data type nvarchar(255), with a default of the name of the local server running Microsoft SQL Server.
StoreForwardMode	Used to specify whether or not store-and-forward capability is enabled. If enabled, and the network connection between the IDAS and the storage node fails, data will be "buffered" to the location specified by the store-and-forward path. Valid values are: = Disabled 1 = Enabled 2 = Autonomous The Autonomous mode (2) is an extension of the normal store-and-forward mode (1). It allows the IDAS to start up using an IDAS configuration file and collect data in store-and-forward mode if the network connection to the AVEVA Historian is not available. This value is of data type tinyint, with a default of 0.
StoreForwardPath	Used to specify the path for the IDAS data buffer on the local hard drive of the IDAS computer. The path should be absolute (for example, c:\IDASBuffer). Data is written to this path until the minimum threshold for the buffer is reached. Remote buffer paths are not supported. When the store-and-forward path specified for the IDAS is invalid, the default path picked by the system is: <public folder>\ArchestrA\Historian\IDAS\SF where the <public folder> is dependent on the operating system. For example, for the Windows 2008 operating system, the path is C:\ProgramData\ArchestrA\Historian\IDAS\SF. When the store-and-forward path specified for the IDAS is just a folder name (without any path characters like \ and :), the default path picked by the system is: <Windows system path>\<folder name specified by the user>. For example, for the Windows Server 2008 32-bit operating system, the path is C:\WINDOWS\system32\<folder name>. This value is of data type nvarchar(255), with a default of an empty string.
MinMBThreshold	The minimum amount of free disk space, in megabytes, at which IDAS stops collecting data in the store-and-forward buffer. This value is of data type int, with a default of 16.

Argument	Description
AltComputerName	The name of the computer on which an optional, redundant IDAS runs. You must use the fully qualified name of the computer. You could also use the IP address. This should be set to an empty string if no redundant IDAS is specified. Make sure that the IDAS software is installed on the target failover computer. If the failure of the primary IDAS is detected by the system, the failover IDAS is automatically started. The failover IDAS is shut down after the primary IDAS is back online. This value is of data type nvarchar(255), with a default of an empty string.
Enabled	Used to specify whether the IDAS is enabled or not. 0 = Not enabled 1 = Enabled Disabling the IDAS allows for the configuration to be retained in the database, even though the IDAS is removed from the system. This value is of data type bit, with a default of 1.
StoreForwardDuration	The minimum duration, in seconds, for the IDAS to function in store-and-forward mode. The IDAS functions in store-and-forward mode for this length of time even if the condition that caused IDAS to function in store-and-forward mode no longer exists. The maximum duration is 3600 seconds, and the minimum is 0 seconds. This value is of data type int, with a default of 180.
AutonomousStartupTimeout	The amount of time, in seconds, that the autonomous IDAS should wait for configuration commands when started by the Configuration service before going to the autonomous mode. This timeout may need to be increased only if you have a large number of IDASs configured as autonomous on a slow network. This value is of data type int, with a default of 60.
BufferCount	The number of 64 KB buffers pre-allocated for buffering data. This number may need to be increased to accommodate high data rates. This value is of data type int, with a default of 128.
FileChunkSize	The size, in bytes, of the data "chunks" that are sent to the historian when store-and-forward data is forwarded. The size of the chunks can be decreased to accommodate slower networks. Decrease this number only if the forwarding delay is greater than zero. This value is of data type int, with a default of 65536.
ForwardingDelay	The interval, in milliseconds, at which "chunks" of store-and-forward data are forwarded to the historian. The length of the interval may need to be increased to accommodate slower networks. This value is of data type int, with a default of 0.
ConnectionTimeout	The amount of time, in seconds, that the Configuration service attempts to communicate with an IDAS for configuration/reconfiguration. If this timeout elapses, the Configuration service assumes that the IDAS connection has been dropped. This number may need to be increased to accommodate slower networks. This value is of data type int, with a default of 30.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIODriverSelect

Selects an IDAS.

Syntax

aaIODriverSelect *IODriverKey*

where:

Argument	Description
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaIODriverUpdate

Updates an IDAS.

Syntax

aaIODriverUpdate *IODriverKey, StorageNodeKey, ComputerName, StoreForwardMode, StoreForwardPath, MinMBThreshold, AltComputerName, Enabled, StoreForwardDuration, AutonomousStartupTimeout, BufferCount, FileChunkSize, ForwardingDelay, ConnectionTimeout*

where:

Argument	Description
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaIODriverInsert* on page 233 stored procedure. However, only these have defaults:

- StorageNodeKey
- MinMBThreshold
- Enabled
- StoreForwardDuration
- AutonomousStartupTimeout
- BufferCount
- FileChunkSize
- ForwardingDelay
- ConnectionTimeout

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerDelete

Deletes an I/O Server from the system configuration.

Syntax

aaIOServerDelete *IOServerKey*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerInsert

Inserts an I/O Server into the system configuration.

Syntax

aaIOServerInsert *StorageNodeKey, ApplicationName, Description, Path, ComputerName, AutoStart, ExeType, InitializationStatus, ProtocolType, AltComputerName, IODriverKey*

where:

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with no default.
Description	The description of the I/O Server. This value is of data type nvarchar(50), with a default of NULL.
Path	The full UNC path (including the filename) to locate the executable file for the I/O Server. If the I/O Server type key is specified, the filename may be omitted. This value is nvarchar(255), with a default of NULL.
ComputerName	The name of the computer on which the I/O Server runs.This value is of data type nvarchar(255), with no default.
AutoStart	Used to control how the I/O Server starts up. 0 = Automatic startup when the system starts. 1 = Manual startup required. Currently not used. This value is of data type bit, with a default of 0.

Argument	Description
ExeType	The type of executable for the I/O Server. Used by the Historian System Management Console to determine how to start the I/O Server. 0 = Service 1 = Console application 2 = Windows application This value is of data type int, with a default of 0.
InitializationStatus	The control flag used to ensure that each I/O Server has been asked for the data type (integer or real) of each tag that it will send. Only needed after a database modification. This value is of data type tinyint, with no default.
ProtocolType	The protocol used by the AVEVA Historian server to communicate with the I/O Server. 1 = DDE (supported only on Windows XP operating system) 2 = SuiteLink 3 = AVEVA Historian named pipe driver (for compatibility with IndustrialSQL Server 3.0 and previous versions) This value is of data type int, with a default of 1.
AltComputerName	The name of the computer on which an optional, failover I/O Server runs. The failover I/O Server must be running in order for the switch to be made. This value is nvarchar(255), with a default of NULL.
IODriverKey	The unique identifier for an IDAS. This value is of data type int, with a default of 2.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerSelect

Selects an I/O Server from the system configuration.

Syntax

aaIOServerSelect *IOServerKey*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaIOServerTypeDelete

Deletes an I/O Server type from the system configuration.

Syntax

aaIOServerTypeDelete *ApplicationName*

where:

Argument	Description
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerTypeInsert

Inserts an I/O Server type into the system configuration.

Syntax

aaIOServerTypeInsert *ApplicationName, Description, ExeName, Revision, Platform*

where:

Argument	Description
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with no default.
Description	The description of the I/O Server type. This value is of data type nvarchar(50), with a default of NULL.
ExeName	The name of the I/O Server's executable file. This value is nvarchar(255), with a default of NULL.
Revision	The revision number for the I/O Server. This value is of data type nchar(20), with a default of NULL.

Argument	Description
Platform	The operating system required by the I/O Server. Valid operating systems are: • WINDOWS NT • WINDOWS 95 • WINDOWS 98 • WINDOWS XP • WINDOWS 2000 • WINDOWS 2003 • WINDOWS XP • WINDOWS VISTA This value is of data type nchar(20), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerTypeSelect

Selects an I/O Server type from the system configuration.

Syntax

aaIOServerTypeSelect *ApplicationName*

where:

Argument	Description
ApplicationName	The application name of the I/O Server. This name is usually the same as the executable file name. This value is of data type nvarchar(32), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaIOServerTypeUpdate

Updates an I/O Server type in the system configuration.

Syntax

aaIOServerTypeUpdate *ApplicationName, Description, ExeName, Revision, Platform*

Arguments

All arguments are the same as for the *aaIOServerTypeInsert* on page 239 stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaIOServerUpdate

Updates an I/O Server in the system configuration.

Syntax

aaIOServerUpdate *IOServerKey, StorageNodeKey, IODriverKey, ApplicationName, Description, Path, ComputerName, AutoStart, ExeType, InitializationStatus, ProtocolType, AltComputerName*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaIOServerInsert* on page 237 stored procedure. However, only the *AltComputerName* argument has a default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitDelete

Deletes a limit.

Syntax

aaLimitDelete *TagName, ContextKey, LimitNameKey*

Arguments

All arguments are the same as for the *aaLimitInsert* on page 242 stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitInsert

Inserts a limit.

Syntax

aaLimitInsert *TagName, ContextKey, LimitType, Value, LimitNameKey, Priority, Checked, Description*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.
ContextKey	The unique numerical identifier for the context. This value is of data type int, with a default of 1.
LimitType	The type of limit; that is, whether it is a rising (up) or falling (down) limit. 0 = Rising; 1 = Falling. This value is of data type int, with a default of 1.
Value	The value that is used as a specific limit for a tag. In theory, a tag can have an infinite number of limits defined. This value is of data type real, with no default.
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with no default.
Priority	The priority for the limit. Priorities can range from 1 to over 2 billion, with 1 being the highest priority. This value is of data type int, with a default of 1.
Checked	Used to specify whether a tag imported from InTouch is configured for automatic limit checking. Only checked limits are imported. 0 = Checking disabled; 1 = Checking enabled. This value is of data type bit, with a default of 1.
Description	The description of the limit. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitNameDelete

Deletes a limit name.

Syntax

aaLimitNameDelete *LimitNameKey*

where:

Argument	Description
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitNameInsert

Inserts a limit name.

Syntax

aaLimitNameInsert *Name*

where:

Argument	Description
Name	The name for the limit. This value is of data type nvarchar(20), with a default of an empty string.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitNameSelect

Selects a limit name.

Syntax

aaLimitNameSelect *LimitNameKey*

where:

Argument	Description
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaLimitNameUpdate

Updates a limit name.

Syntax

aaLimitNameUpdate *LimitNameKey, Name*

where:

Argument	Description
LimitNameKey	The unique numerical identifier associated with a limit name. This value is of data type int, with no default.
Name	The name for the limit. This value is of data type nvarchar(20), with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaLimitSelect

Selects a limit.

Syntax

aaLimitSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. The limit will be selected for the specified tag. This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaLimitUpdate

Updates a limit.

Syntax

aaLimitUpdate *TagName, ContextKey, LimitType, Value, LimitNameKey, Priority, Checked, Description*

Arguments

All arguments are the same as for the *aaLimitInsert* on page 242 stored procedure. However, only the *Description* argument has a default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaMessageDelete

Deletes a message for a discrete tag.

Syntax

aaMessageDelete *MessageKey*

where:

Argument	Description
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaMessageInsert

Inserts a message for a discrete tag.

Syntax

aaMessageInsert *Message0, Message1*

where:

Argument	Description
Message0	The message associated with the FALSE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 0 is in the FALSE state. This value is of data type nvarchar(64), with a default of NULL.
Message1	The message associated with the TRUE state of the discrete tag. The maximum number of characters is 64. A discrete tag set to 1 is in the TRUE state. This value is of data type nvarchar(64), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaMessageSelect

Selects a message for a discrete tag.

Syntax

aaMessageSelect *MessageKey*

where:

Argument	Description
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaMessageUpdate

Updates a message for a discrete tag.

Syntax

aaMessageUpdate *MessageKey, Message0, Message1*

where:

Argument	Description
MessageKey	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaMessageInsert* on page 245 stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaModLogStatus

Used to retrieve the status of modification tracking.

Syntax

aaModLogStatus

Remarks

This stored procedure is used by the System Management Console. Modification tracking is controlled by the value of the ModLogTrackingStatus system parameter, which is stored in the Value column of the *SystemParameter* on page 127 table. If the value of this column is set to a value from 1 to 7, then modification tracking is on (0 = off).

Permission

Execute permission defaults to the public group.

aaNotifyChange

Used internally to track configuration changes for tags. Internal use only.

Syntax

aaNotifyChange *ChangeType*

where:

Argument	Description
ChangeType	Internal use only.

Permission

Execute permission defaults to the aaAdministrators group.

aaPrivateNSAddGroup

Adds a group object in the private namespace under the specified parent object in the namespace hierarchy.

Syntax

aaPrivateNSAddGroup *Name, ParentKey, Type*

where:

Argument	Description
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.
ParentKey	The unique identifier for a named object in this namespace. This value is of data type int, with no default.
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups This value is of data type int, with a default of 1000000.

Permission

Execute permission defaults to the public group.

aaPrivateNSAddLeaf

Adds a single object in the private namespace under the currently selected object in the namespace hierarchy.

Syntax

aaPrivateNSAddLeaf *wwTagKey, NameKey, ServerKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
ServerKey	The unique numerical identifier of an AVEVA Historian server. This value is of data type int, with a default of 1.

Permission

Execute permission defaults to the public group.

aaPrivateNSDeleteGroup

Deletes a group object, as well as any objects under it, in the private namespace.

Syntax

aaPrivateNSDeleteGroup *NameKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaPrivateNSDeleteLeaf

Deletes a single object in the private namespace.

Syntax

aaPrivateNSDeleteLeaf *NameKey, wwTagKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaPrivateNSExpand

Expands the tree view one level under a single parent object in the private namespace.

Syntax

aaPrivateNSExpand *PKey, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Remarks

A parent object can have one or more objects below it in the namespace hierarchy.

Permission

Execute permission defaults to the public group.

aaPrivateNSSelect

Selects all valid group objects (items) for the current user in the private namespace.

Syntax

aaPrivateNSSelect

Permission

Execute permission defaults to the public group.

aaPrivateNSUpdateGroup

Updates a group object in the private namespace.

Syntax

aaPrivateNSUpdateGroup *NameKey, Name, Type*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.

Argument	Description
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

aaPublicNSAddGroup

Adds a group object in the public namespace under the specified parent object in the namespace hierarchy.

Syntax

aaPublicNSAddGroup *Name, ParentKey, Type*

where:

Argument	Description
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.
ParentKey	The unique identifier for a named object in this namespace. This value is of data type int, with no default.

Argument	Description
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy 1999001 = WinPlatform object 1999003 = AppEngine object 1999013 = Area object 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects 1999024 = RedundantDIObject object 1999033 = Undeployed object represented by a generic name 1999901 = ApplicationObject 1999902 = Traceability object This value is of data type int, with a default of 1000000.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSAddLeaf

Adds a single object in the public namespace under the currently selected object in the namespace hierarchy.

Syntax

aaPublicNSAddLeaf *wwTagKey, NameKey, ServerKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
ServerKey	The unique numerical identifier of an AVEVA Historian server. This value is of data type int, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSDeleteGroup

Deletes a group object, as well as any objects under it, in the public namespace.

Syntax

aaPublicNSDeleteGroup *NameKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSDeleteLeaf

Deletes a single object in the public namespace.

Syntax

aaPublicNSDeleteLeaf *NameKey, wwTagKey*

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaPublicNSExpand

Expands the tree view one level under a single parent object in the public namespace.

Syntax

aaPublicNSExpand *PKey, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Remarks

A parent object can have one or more objects below it in the namespace hierarchy.

Permission

Execute permission defaults to the public group.

aaPublicNSSelect

Selects all valid group objects (items) in the public namespace.

Syntax

aaPublicNSSelect

Permission

Execute permission defaults to the public group.

aaPublicNSUpdateGroup

Updates a group object in the public namespace.

Syntax

```
aaPublicNSUpdateGroup NameKey, Name, Type
```

where:

Argument	Description
NameKey	The unique identifier for the object in the namespace. This value is of data type int, with no default.
Name	The name of this object in the hierarchy. This value is of data type nvarchar(255), with no default.

Argument	Description
Type	The value that specifies the type of namespace. 1 to 6 = Tag 1 to 2 million = System 2+ million = Groups Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy 1999001 = WinPlatform object 1999003 = AppEngine object 1999013 = Area object 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects 1999024 = RedundantDIObject object 1999033 = Undeployed object represented by a generic name 1999901 = ApplicationObject 1999902 = Traceability object This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaRedirectToInTouch

Redirects the tag address (item name) to the InTouch node, rather than to the original I/O Server.

Syntax

aaRedirectToInTouch *IOServerKey, InTouchNodeKey*

where:

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.
InTouchNodeKey	The unique numerical identifier of the named InTouch node. This value is of data type int, with no default.

Remarks

When you redirect to InTouch HMI software, all tag values will come from the HMI, not directly from the I/O Server. If you redirect an I/O Server, all topics and tags for that particular I/O Server are affected.

Permission

Execute permission defaults to the aaAdministrators group.

aaSaveChartConfiguration

Returns configuration settings for a particular InSight chart.

Syntax

aaSaveChartConfiguration *ChartConfigurationName, ChartConfigurationUrl, ChartConfigurationType, ChartConfigurationShareMode, LastSharedDateTimeUtc, TimeAggregate, TimePreset, ChartType, MobileShareMode, EmbedShareMode, LastAccessDateTimeUtc, ChartConfiguationKeyword, ChartConfigurationTag, ChartConfigurationProperty, DashboardConfigurationDetail*

where:

Argument	Description
ChartConfigurationName	The name of the InSight content.
ChartConfigurationUrl	The web address for the InSight content.
ChartConfigurationType	Specifies what type of chart was savedFor example, single chart or dashboard.
ChartConfigurationShareMode	Specifies whether the InSight content is shared.
LastSharedDateTimeUtc	Specifies when the InSight content was last shared.
TimeAggregate	Specifies the aggregates used by the saved conten. For example, Hour/Day for a Column chart.
TimePreset	Specifies the selected time frame of the saved content. For example: Last 30 days, Last hour, or specific start and end times (for Custom).
ChartType	The type of chart used for this InSight content.
MobileShareMode	Specifies whether this InSight content is shared with mobile users.
EmbedShareMode	Specifies whether this InSight content can be embedded into a web page or other object.
LastAccessDateTimeUtc	Specifies when the InSight content was last accessed.
ChartConfiguationKeyword	Specifies keywords for the chart.
ChartConfigurationTag	Specifies tags used in the chart.
ChartConfigurationProperty	Specifies a property used by the chart.
DashboardConfigurationDetail	Specifies the position/index of the chart when it is displayed in a dashboard with other charts.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaSearchMessageInsert

Inserts a document to the SearchMessageSyncRequest table so that the search indexer service can dequeue and index that document.

Syntax

aaSearchMessageInsert *JobId, IndexingMessage, DocumentType*

where:

Argument	Description
JobId	The unique numerical identifier for a search message synchronization job.
IndexingMessage	Details from the JSON file used for search indexing to make the associated content searchable. This is an example: `{"_messageid":"test_636571577850365608","_tenantid":"","_body":[{"_keywords":[],"_fields":[{"ContentName":"test","_analyzers":"nGram"}],"_id":"afwwGwD30FUMl9UpE`
DocumentType	Specifies the type of index document; for example SavedContent or Tag.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaSetCalculatedAISamples

Used to _____

Syntax

aaSetCalculatedAISamples

Permission

Execute permission defaults to_____.

aaSetServerTimeStamp

Specifies whether or not incoming data values will be timestamped using the time of the local AVEVA Historian.

Syntax

aaSetServerTimeStamp *TopicName, ServerTimeStamp*

where:

Argument	Description
TopicName	The name of the topic. This value is of data type nvarchar(80), with no default.

Argument	Description
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the Storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaSetStorageRule

Sets storage rules at various levels of the tag definition.

Syntax

aaSetStorageRule *Type, Key, StorageType, StorageRate, TimeDB, ValueDB, AcqType, DBType, RateDB, ServerTimeStamp, LateData, IdleDuration, ProcInterval*

where:

Argument	Description
Type	The level at which the new storage rule will be set for the tag definition. 1 = I/O Server 2 = Topic This value is of data type tinyint, with no default.
Key	The database key value for the relevant type, either the I/O Server key or the topic key. This value is of data type int, with no default.

Argument	Description
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type tinyint, with no default.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in seconds. This value is of data type int, with a default of 0.
TimeDB	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of -1.
ValueDB	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied. This value is of data type float, with a default of -1.
AcqType	Used to turn acquisition on or off. 0 = Acquisition off 1 = Acquisition on This value is of data type smallint, with a default of -1.
DBType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of -1.

Argument	Description
RateDB	Used to percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied. This value is of data type float, with a default of -1.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type smallint, with a default of -1.
LateData	Used to enable acquisition of "late" data. 0 = Late data disabled 1 = Late data enabled This value is of data type smallint, with a default of -1.
IdleDuration	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds. This value is of data type int, with a default of 60.
ProcInterval	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration. This value is of data type int, with a default of 120.

Remarks

To ignore an argument, set the value to -1.

Permission

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

aaSetTagStorage

Sets storage on or off from various level of the tag definition.

Syntax

```
aaSetTagStorage Type, List, Set
```

where:

Argument	Description
Type	The level at which the new storage rule will be set for the tag definition. 1 = I/O Server 2 = Topic This value is of data type tinyint, with no default.
List	If the type is an I/O Server, topic or public group, the IDENTITY key(s) of the relevant type. If the type is a tag, a list of tagnames separated by commas. This value is of data type nvarchar(4000), with no default.
Set	Used to set storage on or off. Valid values are: • ON • OFF This value is of datatype varchar(3), with no default.

Remarks

This stored procedure applies to analog, discrete, string and complex tag types.

Permissions

Execute permission defaults to the aaAdministrators and aaPowerUsers groups.

Examples

The following example turns data storage off for all tags associated with I/O Servers that are identified by the IOServerKeys 2 and 3.

```
aaSetTagStorage 1, '2,3', 'OFF'
```

The following example turns data storage on for the listed tags.

```
aaSetTagStorage 3, 'Tag1, Tag2, Tag3', 'ON'
```

aaSnapshotDetailSelect

Returns snapshot information from the columns of the SnapshotDetail table, based on the storage size.

Syntax

aaSnapshotDetailSelect *StorageSize*

where:

Argument	Description
StorageSize	The storage size, in bytes, of the tag value: -1 = Blob; 0 = Variable length string; 1 = 1 byte; 2 = 2 byte; 4 = 4 byte; 8 = 8 byte. This value is of data type int, with a default of NULL.

Remarks

If you do not pass an argument for the storage size, information for all storage sizes in the table will be returned.

Permission

Execute permission defaults to the public group.

aaSnapshotDetailUpdate

pdates the SnapshotDetail table.

Syntax

aaSnapshotDetailUpdate *StorageSize, SnapshotSize, ImageTime, ThresholdTime*

where:

Argument	Description
StorageSize	The storage size, in bytes, of the tag value: -1 = Blob 0 = Variable length string 1 = 1 byte 2 = 2 byte 4 = 4 byte 8 = 8 byte This value is of data type int, with no default.
SnapshotSize	The maximum size of the snapshot, in bytes. If this limit is reached, a new snapshot is created. This value is of data type int, with no default.
ImageTime	The interval, in seconds, between updates to the snapshot file. The snapshot file is updated with tag value information from the snapshot buffer, which resides in memory. This value is of data type int, with no default.
ThresholdTime	The maximum amount of time, in seconds, that can elapse before a new snapshot is automatically created, provided that the value for the snapshot size has not been reached. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSnapToSummary

Used by the event system when configuring summary operations.

Syntax

aaSnapToSummary *OpKey, Start, End, DateStamp*

where:

Argument	Description
OpKey	An internal variable that identifies the summary operation to perform. This value is of data type int, with no default.
Start	The starting timestamp for the calculation. This value is of data type datetime2(7), with no default.
End	The ending timestamp for the calculation. This value is of data type datetime2(7), with no default.
DateStamp	The time the summary operation was performed. This value is of data type smalldatetime, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaSpaceManager

Used by the system to manage the amount of disk space used to store historical data for summaries and events.

Syntax

aaSpaceManager

Remarks

This stored procedure is automatically run by the system every ten minutes. This stored procedure executes the *aaDeleteOlderEvents* on page 200 and *aaDeleteOlderSummaries* on page 200 stored procedures to clear out old historical data. The duration for which event and summary history is kept is based on system parameters stored in the *SystemParameter* on page 127 table.

Permission

Execute permission defaults to the aaAdministrators group.

aaStorageLocationSelect

Selects a storage location.

Syntax

aaStorageLocationSelect *StorageType, StorageNodeKey*

where:

Argument	Description
StorageType	The type of storage used for the specified location. 1 = Circular 2 = Alternate 3 = Buffer 4 = Permanent There can be only one storage location of each type. This value is of data type int, with a default of NULL.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with a default of 1.

Permission

Execute permission defaults to the public group.

aaStorageLocationUpdate

Updates the storage location.

Syntax

aaStorageLocationUpdate *StorageType, StorageNodeKey, SortOrder, Path, MaxMBSize, MinMBThreshold*

where:

Argument	Description
StorageType	The type of storage used for the specified location. 1 = Circular 2 = Alternate 3 = Buffer 4 = Permanent There can be only one storage location of each type. This value is of data type int, with no default.
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with no default.
SortOrder	Applies only to the alternate area. If more than one location is defined, the sort order determines the order in which the alternate areas are used. Reserved for future use. This value is of data type int, with no default.

Argument	Description
Path	The path to the storage location. The circular storage location must be a local drive on the server machine, and the path must be specified using normal drive letter notation (for example, c:\Historian\Data\Circular). While the alternate, buffer, and permanent storage locations can be anywhere on the network, it is strongly recommended to have the alternate storage location configured on a dedicated physical drive locally attached by a high-speed interface to the Historian server or configured to be on a different internal hard drive. If you use a network location, then the ArchestrA user must have full access to the network location. The locations must be specified using UNC notation. Mapped drives are not supported. This value is of data type nvarchar(255), with no default.
MaxMBSize	The limit, in megabytes, for the amount of data to be stored to the specified location. The maximum size applies to circular and alternate storage only. If the maximum size is set to 0, all available space at the storage location is used. This value is of data type int, with no default.
MinMBThreshold	The minimum amount of disk space, in megabytes, at which the system attempts to start freeing up space. The threshold applies to circular and alternate storage only. Typically, you should multiply the size of the average history block (before any compression) by 1.5 to determine the minimum threshold. This value is of data type int, with no default.
MaxAgeThreshold	The age, in days, of data that will be deleted by system to free up disk space. The threshold applies to circular and alternate storage only. The minimum age is 2 days. A value of 0 indicates that no age threshold is applied. This value is of data type int, with a default of 0.

Permission

Execute permission defaults to the aaAdministrators group.

aaStringDetail

Returns a description for one or more specified tags.

Syntax

aaStringDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaStringTagDelete

Deletes a string tag.

Syntax

```
aaStringTagDelete wwTagKey
```

Arguments

wwTagKey
The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaStringTagInsert

Inserts a string tag.

Syntax

aaStringTagInsert *TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, MaxLength, InitialValue, TopicKey, IOServerKey, CurrentEditor, DoubleByte, SamplesInActiveImage, ServerTimeStamp, DeadbandType, AIRetrievalMode, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Description	The description of the tag. This value is of data type nvarchar(512), with a default of an empty string.
AcquisitionType	The method by which the tag's value is acquired. If the tag value is acquired from an I/O Server, specify the name of the I/O Server, topic, and item. 0 = Not acquired 1 = Acquired via an I/O Server 2 = Acquired via HCAL or MDAS or a manual update 3 = System driver This value is of data type tinyint, with a default of 1.

Argument	Description
StorageType	The type of storage defined for the tag. 0 = Not stored. 1 = Cyclic. 2 = Delta. 3 = Forced storage. 17 = The storage type has been changed from cyclic to "not stored." 18 = The storage type has been changed from delta to "not stored." 19 = The storage type has been changed from "forced" to "not stored." This value is of data type smallint, with a default of 2.
StorageRate	The rate at which the tag is stored if the storage type is cyclic. The rate is in milliseconds. This value is of data type int, with a default of 1000.
ItemName	The address string of the tag. This value is of data type nvarchar(256), with a default of an empty string.
TimeDeadband	The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes. This value is of data type int, with a default of 0.
CreatedBy	The name of the user or application that created the tag. This value is of data type nvarchar(256), with a default of an empty string.
DateCreated	The date that the tag was created. This value is of data type datetime2(7), with a default of NULL.
MaxLength	The maximum number of characters for the string. This value is of data type smallint, with a default of 131. Valid values are: • 8 • 64 • 16 • 128 • 24 • 131 • 32 • 256 • 48 • 512 •
InitialValue	The initial value as imported from an external source (for example, from InTouch). This value is of data type nvarchar(512), with a default of an empty string.
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.

Argument	Description
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with a default of NULL.
CurrentEditor	Used to specify which application or editing environment controls the tag definition. Tags imported from the InTouch HMI software use InTouch as the current editor. If modifications are made to an imported tag in the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. If a re-import is performed, any modifications made using the Configuration Editor are preserved. You can manually maintain InTouch as the current editor for re-importing; however, all changes made to the tag using the Configuration Editor are lost during the re-import. Tags (attributes) that are initially configured using AVEVA Application Server use the ArchestrA Integrated Development Environment (IDE) as the current editor. If you modify an Application Server tag using the historian Configuration Editor, then the current editor for the tag is changed to AVEVA Historian. However, the next time you redeploy the engine, the changes are not preserved. 0 = AVEVA Historian; 1 = InTouch; 2 = AVEVA Application Server. This value is of data type int, with a default of 0.
DoubleByte	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. This value is of data type tinyint, with a default of 0.
SamplesInActiveImage	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with a default of 0.
ServerTimeStamp	Used to specify whether local timestamping by the AVEVA Historian is used. 0 = The IDAS timestamp is used. 1 = The AVEVA Historian time is used for the timestamp. If a fast-changing tag is configured to use server timestamping, the packet of data that is sent to the storage subsystem may contain multiple data values with the same timestamp, which may affect data calculations, such as for swinging door storage. This value is of data type bit, with a default of 0.
DeadbandType	The type of delta storage deadband to be applied for the tag. This setting is only in effect if delta storage is configured for the tag. 1= Time and/or value deadband 2 = Rate (swinging door) deadband This value is of data type smallint, with a default of 1.

Argument	Description
AIRetrievalMode	Used to specify the behavior of retrieval for data in active image. You can either retrieve from all acquired data values that are currently in the active image, or only the data values that are configured to be stored on disk. Data on disk may be a subset of that in the active image, depending on the storage rate for the tag. This value is of datatype tinyint. Valid values are: 0 = All of the values received into the active image will be included in the returned data (default). 1 = Only the values that will be moved into storage will be included in the returned data.
SourceTag	The name of the source tag to create the tag from. This value is of data type nvarchar(256), with a default of an empty string.
SourceServer	The name of the source server for the source tag. This value is of data type nvarchar(256), with a default of an empty string.
AITag	Used to specify whether the tag's values are stored by the Classic Storage subsystem. 0 = Not stored by the Classic Storage subsystem; 1 = Stored by the Classic Storage subsystem. This value is of data type bit, with a default of 1.
TagId	The unique identifier for the tag. The value is of data type uniqueidentifier, with a default of NULL.
ChannelStatus	Used for tags from AVEVA Application Server 2012 R2 or later or the AVEVA Historian SDK 2012 R2 or later. Used to specify how disconnects between these sources and the AVEVA Historian are reflected in the data until the disconnect period can be backfilled with store-and-forward data, if store-and-forward is enabled. 1 = Enabled. NULL values are injected into the data stream for the disconnect period. For a trend, this means that a line gap appears during the period of NULL values. The tag remains in store-and-forward mode until the timestamps become greater than the startup time of the server or the time that the connection was restored. 0 = Disabled. NULL values are not injected and no gap is shown in client-side trends. The channel status value is ignored for tags that use classic storage. This value is of data type tinyint, with a default of 1.
AIHistory	Used to specify whether data exists for a tag in both storage and classic storage. 0 = No data was previously collected by classic storage. 1 = The tag may have data previously collected by classic storage. This value is of data type bit, with a default of 1.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaStringTagSelect

Selects a string tag.

Syntax

aaStringTagSelect *wwTagKey*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaStringTagUpdate

Updates a string tag.

Syntax

aaStringTagUpdate *wwTagKey, TagName, Description, AcquisitionType, StorageType, StorageRate, ItemName, TimeDeadband, CreatedBy, DateCreated, MaxLength, InitialValue, TopicKey, IOServerKey, CurrentEditor, DoubleByte, SamplesInActiveImage, ServerTimeStamp, DeadbandType, AIRetrievalMode, SourceTag, SourceServer, AITag, TagId, ChannelStatus, AIHIstory*

where:

Argument	Description
wwTagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaStringTagInsert* on page 265 stored procedure. However, only these have defaults:

- *AcquisitionType*
- *StorageType*
- *CreatedBy*
- *DateCreated*
- *MaxLength*
- *DoubleByte*
- *SamplesInActiveImage*
- *ServerTimeStamp*
- *DeadbandType*
- *AIRetrievalMode*
- *SourceTag*
- *SourceServer*
- *AITag*
- *TagId*
- *ChannelStatus*
- *AIHIstory*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaSummaryActionInsert

Used by the event subsystem to perform a summary operation for the specified tag.

Syntax

aaSummaryActionInsert *EventTagName, SumDateTime*

where:

Argument	Description
EventTagName	The name of the event tag with which the summary operation is associated. This value is of data type nvarchar(256), with no default.
SumDateTime	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends. This value is of data type datetime2(7), with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryDetail

Returns summary details for one or more tags that are associated with a particular summary operation. The type of aggregation for the tag can optionally be included for each tag that you list.

The starting and ending times are used to specify the time at which the calculation started/ended for the operation.

Syntax

aaSummaryDetail *TagList, StartTime, EndTime, OrderBy*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.
StartTime	The starting timestamp for the calculation. This value is of data type nvarchar(50), with no default.
EndTime	The ending timestamp for the calculation. This value is of data type nvarchar(50), with no default.
OrderBy	The column by which the results will be ordered. By default, the TagName column is used. This value is of data type nvarchar(500).

Permission

Execute permission defaults to the public group.

Examples

This example returns the average and minimum values for 'ReactTemp' and the maximum value for 'ReactLevel' between 12:12 p.m. and 2:14 p.m. on May 12, 2001. The returned rows are ordered by the date of the summary.

```
aaSummaryDetail "ReactTemp('AVG','MIN'), ReactLevel('MAX')", "2001-05-12
12:12:00.000", "2001-05-12 12:14:00.000", "SummaryDate"
```

This example returns all aggregate values for 'ReactTemp' and 'ReactLevel' between 12:12 p.m. and 2:14 p.m. on May 12, 2001.

```
aaSummaryDetail "ReactTemp, ReactLevel", "2001-05-12 12:12:00.000",
"2001-05-12 12:14:00.000"
```

aaSummaryOperationDelete

Deletes a summary operation.

Syntax

aaSummaryOperationDelete *OperationKey*

where:

Argument	Description
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryOperationInsert

Inserts a summary operation that will be associated with the specified event tag.

Syntax

aaSummaryOperationInsert *TagName, CalcType, Duration, Resolution, TimeStamp, Description*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with no default.
CalcType	The type of calculation to be performed: SUM, MAX, MIN, or AVG. This value is of datatype char(3), with no default.
Duration	The period, in seconds, for which the calculation is performed. This value is of data type real, with no default.

Argument	Description
Resolution	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date. This value is of data type int, with no default.
TimeStamp	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends. 0 = Beginning of the calculation period 1 = End of the calculation period This value is of data type tinyint, with no default.
Description	The description of the summary operation. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryOperationSelect

Selects a summary operation.

Syntax

```
aaSummaryOperationSelect EventTagName, CalcType, Duration, Resolution,
TimeStamp
```

Arguments

EventTagName
The name of the event tag with which the summary operation is associated. This value is of data type nvarchar(256), with a default of NULL.

The remaining arguments are the same as for the aaSummaryOperationInsert stored procedure. However, all of the arguments have a default of NULL.

Remarks

The arguments of this stored procedure are used in three ways: (1) if no arguments are specified, all summary operations will be returned; (2) if the EventTagName argument is specified, all summary operations for that event tag will be returned; (3) if all arguments are specified, only the summary operation that matches the criteria will be returned.

Permission

Execute permission defaults to the public group.

aaSummaryOperationUpdate

Updates the summary operation that is associated with a specified event tag.

Syntax

aaSummaryOperationUpdate *OperationKey, TagName, CalcType, Duration, Resolution, TimeStamp, Description*

where:

Argument	Description
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.

The remaining arguments are the same as for the *aaSummaryOperationInsert* on page 271 stored procedure. However, only the *Description* argument has a default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryTagListDelete

Deletes summary information for a tag.

Syntax

aaSummaryTagListDelete *SumVarKey*

where:

Argument	Description
SumVarKey	The unique numerical identifier for a summarized tag. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryTagListInsert

Inserts summary information for a specified tag.

Syntax

aaSummaryTagListInsert *TagName, OperationKey, LowerLimit, UpperLimit, Description*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.

Argument	Description
LowerLimit	The lower limit of validity for the tag's value. Values lower than this limit are not used in the calculation. By default, this value is set to -1000000000. This value is of data type real.
UpperLimit	The upper limit of validity for the tag's value. Values higher than this limit are not used in the calculation. By default, this value is set to 1000000000. This value is of data type real.
Description	The description of the summarized tag. This normally describes the result of the operation, although this description can be the same as that of the tag on which the operation is performed. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the aaAdministrators group.

aaSummaryTagListSelect

Selects summary information for a tag.

Syntax

aaSummaryTagListSelect *OperationKey, TagName*

where:

Argument	Description
OperationKey	The unique numerical identifier for the summary operation. This value is of data type int, with no default.
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSummaryTagListUpdate

Updates summary information for a specified tag.

Syntax

aaSummaryTagListUpdate *SumVarKey, TagName, OperationKey, LowerLimit, UpperLimit, Description*

where:

Argument	Description
SumVarKey	The unique numerical identifier for a summarized tag. This value is of data type int, with no default.

The remaining arguments are the same as for the aaSummaryTagListInsert stored procedure.

Permission

Execute permission defaults to the aaAdministrators group.

aaSystemConfigNSExpand

Expands the tree view under a single object in the system namespace. This stored procedure is used by the System Management Console.

Syntax

aaSystemNSExpand *PKey, FKey1, FKey2, FKey3, TokenType, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FKey1-FKey3	A local variable used to determine the position of the object in the tree view. This value is of data type int, with no default.
TokenType	The type of system namespace group. 1000010 = Data Acquisition 1000017 = System Driver 1000018 = IDASs 1000019 = I/O Servers This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemNSExpand

Expands the tree view under a single object in the system namespace.

Syntax

aaSystemNSExpand *PKey, FKey1, FKey2, FKey3, TokenType, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.

Argument	Description
FKey1-FKey3	A local variable used to determine the position of the object in the tree view. This value is of data type int, with no default.
TokenType	The type of system namespace group. 1000010 = Data Acquisition 1000017 = System Driver 1000018 = IDASs 1000019 = I/O Servers This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemNSExpand2

Expands the tree view under a single object in the system namespace.

Note: This stored procedure is a simpler version of the aaSystemNSExpand stored procedure.

Syntax

aaSystemNSExpand2 *PKey, FilterStr*

where:

Argument	Description
PKey	A local variable used to identify the object in the namespace. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemParameterSelect

Returns details for a specified system parameter, such as a description of the parameter, the current value, and so on. If you do not specify a name, the stored procedure returns details for all defined system parameters.

Syntax

aaSystemParameterSelect *name*

where:

Argument	Description
Name	The unique name for the system parameter. This value is of data type nvarchar(50), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaSystemParameterUpdate

Updates the value and description for a specified system parameter. If you do not provide a description, the previous description is used.

Syntax

aaSystemParameterUpdate *Name, Value, Description*

where:

Argument	Description
Name	The unique name for the system parameter. This value is of data type nvarchar(50), with no default.
Value	The value of the system parameter. This value is of data type sql_variant, with no default.
Description	The description of the system parameter. This value is nvarchar(255), with a default of NULL.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTagConfig

Used by the internal Configuration Manager when the AVEVA Historian starts.

Syntax

aaTagConfig

Remarks

This stored procedure takes a snapshot of the system configuration at the tag level.

Permission

Execute permission defaults to the public group.

aaTagConfigModified

Used by the internal configuration object.

Syntax

aaTagConfigModified

Remarks

This stored procedure has the same functionality as the *aaTagConfig* on page 277 stored procedure, but only retrieves the database modifications pending when a commit of changes is performed.

Permission

Execute permission defaults to the public group.

aaTagConfigSelect

Used by the System Management Console to return a list of tags associated with a particular engineering unit (for analog tags) or message (for discrete tags).

Syntax

aaTagConfigSelect, *TagType, Key, FilterStr*

where:

Argument	Description
TagType	The type of tag to retrieve. 1 = Analog 2 = Discrete This value is of data type int, with no default.
Key	The database key value for the relevant type, either the message key or the engineering unit key. This value is of data type int, with no default.
FilterStr	Used to order the tagnames in the namespace. This value is of data type nvarchar(100), with a default of N%.

Permission

Execute permission defaults to public group.

aaTagInfo

Returns definition information for each specified tag.

Syntax

aaTagInfo *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaTagType

Returns the tag type for each specified tag.

Syntax

aaTagType *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaTimeDetectorDetailInsert

Inserts time detector details that are associated with a specified event tag.

Syntax

aaTimeDetectorDetailInsert *FrequencyID, TagName, Periodicity, StartDateTime, RunTimeDay, RunTimeHour, RunTimeMin*

where:

Argument	Description
FrequencyID	The unique numerical identifier for the frequency. Used to link a frequency with a time-based detector. 1= Hourl 2 = Daily 3 = Weekly 4 = Monthly 5 = Periodi 6 = Other (Reserved for future use) This value is of data type int, with no default.
TagName	The unique name of the tag within the AVEVA Historian system. This value is of data type nvarchar(256), with no default.
Periodicity	The interval period in minutes between detector events. Only used for a periodic detection. This value is of data type int, with no default.
StartDateTime	The timestamp from which the time detector starts. Only used for a periodic detection. This value is of data type datetime2(7), with no default.

Argument	Description
RunTimeDay	In the context of a weekly detector, RunTimeDay maps the week day number (0 = Sunday – 6 = Saturday). In the context of a monthly detector, RunTimeDay maps to the day of the month. Not used for periodic detections. This value is of data type tinyint, with no default.
RunTimeHour	The hour of the day at which the time detector triggers. Not used for periodic detections. This value is of data type tinyint, with no default.
RunTimeMin	The minute of the hour at which the time detector triggers. Not used for periodic detections. This value is of data type tinyint, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTimeDetectorDetailSelect

Selects the time detector from the TimeDetectorDetail table that is associated with the specified event tag.

Syntax

aaTimeDetectorDetailSelect *TagName*

where:

Argument	Description
TagName	The unique name of the tag within the AVEVA Historian system.This value is of data type nvarchar(256), with a default of NULL.

Permission

Execute permission defaults to the public group.

aaTimeDetectorDetailUpdate

Updates the time detector in the TimeDetectorDetail table that is associated with the specified event tag.

Syntax

aaTimeDetectorDetailUpdate *FrequencyID, TagName, Periodicity, StartDateTime, RunTimeDay, RunTimeHour, RunTimeMin*

Arguments

The arguments are the same as for the *aaTimeDetectorDetailUpdate* on page 280 stored procedure. However, none of the arguments have defaults.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTopicDelete

Deletes an I/O topic.

Syntax

aaTopicDelete *TopicKey*

where:

Argument	Description
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with no default.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTopicInsert

Inserts an I/O topic.

Syntax

aaTopicInsert *StorageNodeKey, IOServerKey, Name, TimeOut, LateData, IdleDuration, ProcessingInterval*

where:

Argument	Description
StorageNodeKey	The unique numerical identifier for the storage node. This value is of data type int, with no default.
IOServerKey	The unique numerical identifier for the I/O Server. This value is of data type int, with no default.
Name	The name of the topic. This value is of data type nvarchar(80), with no default.
TimeOut	The time span, in milliseconds, in which a data point must be received on the topic. If no data point is received in this time span, the topic is considered "dead." The historian disconnects and then attempts to reconnect to the topic. This value is of data type int, with a default of 60000.
LateData	Used to enable acquisition of "late" data. 0 = Late data disabled 1 = Late data enabled This value is of data type bit, with a default of 0.

Argument	Description
IdleDuration	The amount of time, in seconds, before data is processed from the I/O Server. For example, if you set this value to 60 seconds, data from this I/O Server is cached and only processed by the storage engine after no more data has been received from the I/O Server for at least 60 seconds. This value is of data type int, with a default of 60.
ProcessingInterval	The amount of time, in seconds, after which late data from the I/O Server is processed, regardless of the idle duration. If the nature of the data is such that the idle duration is never satisfied, the historian storage engine processes data from the topic at least one time every processing interval. The processing interval defaults to twice the idle duration and cannot be set to a value less than the idle duration. This value is of data type int, with a default of 120.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaTopicSelect

Selects an I/O topic.

Syntax

aaTopicSelect *TopicKey*

where:

Argument	Description
TopicKey	The unique numerical identifier for the topic. This value is of data type int, with a default of NULL.

Permission

Execute permission defaults to the public group.

aaTopicUpdate

Updates an I/O topic.

Syntax

aaTopicUpdate *TopicKey, StorageNodeKey, IOServerKey, Name, TimeOut, LateData, IdleDuration, ProcessingInterval*

where:

Argument	Description
TopicKey	The unique numerical identifier for the topic. This value is of data

Argument	Description
	type int, with no default.

The remaining arguments are the same as for the *aaTopicInsert* on page 281 stored procedure. However, only these have defaults:

- *TimeOut*
- *LateData*
- *IdleDuration*
- *ProcessingInterval*

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaUpdateCalculatedAISamples

Used by the system to write the optimum number of samples in the active image to the CalculatedAISamples column in the Tag table. This stored procedure is used by the AVEVA Historian and should not be executed by users.

Syntax

aaSetCalculatedAISamples *TagKey, Samples*

where:

Argument	Description
TagKey	The unique numerical identifier of a tag within a single AVEVA Historian. This value is of data type int, with no default.
Samples	The number of samples that the active image holds for the tag. 0 indicates that the active image is using the default of 65 values. The higher the number of samples, the higher the load on memory resources. This value is of data type int, with no default.

Remarks

If the AIAutoResize system parameter is set to 1, the system continuously recalculates the optimum number of samples for each tag based on the data rates received. If the calculated value exceeds the current value in the database, then the system updates the CalculatedAISamples column in the *_Tag* on page 49 table.

Permission

Execute permission defaults to the aaPowerUsers and aaAdministrators groups.

aaUpdateChartConfigurationStatistics

Returns statistics about chart configuration access.

Syntax

aaUpdateChartConfigurationStatistics *ChartConfigurationUrl*

where:

Argument	Description
ChartConfigurationUrl	The URL for the chart configuration.

Permission

Execute permission defaults to the aaAdministrators, aaPowerUsers, and aaUsers groups.

aaUserAccessLevelSelect

Returns the access level associated with the currently logged on user.

Access levels are:

1 = Undefined AVEVA Historian user (for example, public)

2 = User (hUser permissions)

3 = PowerUser (hPowerUser permission)

3 = Admin (hAdmin permissions)

9999 = dbo

Syntax

aaUserAccessLevelSelect

Remarks

The access level values correspond to values in the *UserDetail* on page 138 table, which is populated during installation.

Permission

Execute permission defaults to the public group.

aaUserDetailUpdate

Allows the UserDetail table to be populated from information contained in the sysusers table.

Syntax

aaUserDetailUpdate

Permission

Execute permission defaults to the aaAdministrators group.

Stored Procedures for Internal Use

Stored procedures that are used internally by the system are prefixed with "aaInternal". For example, aaInternalAnalogTagExport. Do not use these stored procedures or change them in any way. Internal stored procedures may change from release to release, and no legacy support will be provided.

Creating Stored Procedures

You can create your own stored procedures for use with the AVEVA Historian. All procedure names will be stored in the Runtime database. The stored procedure text will be stored in the Microsoft SQL Server and retrieved at startup (from procedures created in an older session) as well as at creation time (from procedures created in the current session). Temporary procedures will not be supported. No arguments are allowed.

As with Microsoft SQL Server support, support for dynamic stored procedures for the historian will be such that when defining a stored procedure, you can create a stored procedure only in the current database, and the CREATE PROCEDURE statement cannot be combined with other SQL statements in a single batch.

Creating your own stored procedures is useful when you want to execute certain types of queries through a typical ODBC connection. The historian requires a specific ODBC configuration unless you create a stored procedure to execute the query.

For example, the following query creates a stored procedure that returns the timestamp and value for the tag 'ReactLevel" for the last 15 minutes.

```
CREATE PROCedure MyProc
AS
SELECT DateTime, TagName, Value
   FROM History
      WHERE TagName = 'ReactLevel'
         AND DateTime >= DATEADD(mi, -15, GETDATE())
         AND DateTime <= GETDATE()
```

CHAPTER 5

User-Defined Functions

A SQL Server function is a subroutine containing one or more Transact-SQL statements. Functions can be used to encapsulate code for reuse.

faaCheckLicenseViolation

Checks to see if the total number of tags in the AVEVA Historian is less than or equal to the number allowed by the current license.

Syntax

```
SELECT dbo.faaCheckLicenseViolation()
```

Return Type

Integer.

Remarks

If the total number of tags in the system is below the amount allowed, the result of this function will be 0. If not, the number of tags that exceed the allocated amount will be returned. For example, if a system has 100 tags, but the license only allows for 60, a value of 40 will be returned.

faaContainedName

Given a string in the form of "TagName [ContainedName]," returns the ContainedName.

Syntax

```
SELECT dbo.faaContainedName(DisplayName)
```

Arguments

DisplayName
The name as it appears in the model view hierarchy. The display name format is: TagName [ContainedName].

Return Type

Nvarchar(255).

Remarks

The maximum number of characters for both the display name and the returned contained name is 255.

faaGetHierarchicalAttributeNames

Returns the ArchestrA hierarchical name plus the attribute name, when provided a historian tagname.

Syntax

```
SELECT dbo.faaGetHierarchicalAttributeNames(HistorianTagname)
```

Arguments

HistorianTagname
Tagname within the historian for which you want to return the hierarchical name. This value is of data type nvarchar(256).

faaGetHistorianTagNames

Returns the historian tagname, when provided an ArchestrA hierarchical attribute name starting with tagname as the input.

Syntax

```
SELECT dbo.faaGetHistorianTagNames(HierarchicalAttributeName)
```

Arguments

HierarchicalAttributeName
An ArchestrA hierarchical attribute name starting with tagname as the input. This value is of data type nvarchar(256).

faaLicensedTagDetails

Returns the total number of tags and the number of licensed tags in the system, as well as for each tag type.

Syntax

```
SELECT * FROM dbo.faaLicensedTagDetails()
```

Table Returned

The result is returned in a table format. For example:

Tag Type	Tag Count	Licensed Tags
Analog	213	121
Discrete	68	60
String	27	26
Event	3	0
Total	311	207

Remarks

System tags and event tags are not included in the total tag count for licensing purposes.

faaLicensedTagTotal

Returns the total number of tags in the system for the specified tag type or for all tags.

Syntax

```
SELECT dbo.faaLicensedTagTotal(TagType)
```

Arguments

TagType
The type of tag. 1 = Analog; 2 = Discrete; 3 = String; 4 = Complex; 0 = All tags.

Return Type

Integer.

Remarks

System tags and event tags are not included in the total tag count for licensing purposes.

faaObjectTagName

Given a string in the form of "TagName [ContainedName]," returns the Tagname.

Syntax

```
SELECT dbo.faaObjectTagName(DisplayName)
```

Arguments

DisplayName
The name as it appears in the model view hierarchy. The display name format is: TagName [ContainedName].

Return Type

Nvarchar(255).

Remarks

The maximum number of characters for both the display name and the returned tagname is 255.

faaTagsInLicenseViolation

Returns information about tags that have been disabled due to violation of the licensed tag count. The information is returned in a table format.

Syntax

```
SELECT * FROM dbo.faaTagsInLicenseViolation()
```

Table Returned

Column	Data type	Description
TagName	nvarchar(256)	The unique name of the tag within the AVEVA Historian system.
wwTagKey	int	The unique numerical identifier of a tag within a single AVEVA Historian.
Description	nvarchar(512)	The description of the tag.

Column	Data type	Description
Address	nvarchar(512)	The address information for the tag, which consists of the computer name, application name, topic, and item name. For example, \\kc1\VIEW!Tagname:ReactLevel.

Remarks

If the licensed tag count has been exceeded, the historian will disable enough tags to run with the allowed licensed tag count. To determine the tags that are in violation of the license, the system first generates the total number of analog, discrete, complex, and string tags. (System and event tags are not included in the total tag count for licensing.) If the total number of tags exceeds the number allowed by the license, the most recent tag additions to the system are disabled. The most recent additions are indicated by the wwTagKey column of the Tag table; the higher the number, the more recent the addition.

faaTZgetdate

Returns the date/time, in the appropriate time zone.

Syntax

```
SELECT dbo.faaTZgetdate(TimeZone)
```

Arguments

TimeZone
The name of the time zone.

Return Type

nvarchar(100).

Remarks

Use this function instead of the SQL **GetDate()** function to specify a time zone other than the server time zone in a query. To retrieve data in the time zone of the server, just use the SQL **GetDate()** function.

Example

```
DECLARE @starttime datetime
SET @starttime = dbo.faaTZgetdate('eastern daylight time')
SELECT DateTime, TagName, Value FROM History
   WHERE TagName IN ('SysTimeHour', 'SysTimeMin', 'SysTimeSec')
      AND DateTime > DateAdd(mi, -30, @starttime)
      AND DateTime < DateAdd(mi, -5, @starttime)
      AND wwTimeZone = 'eastern daylight time'
```

For more information on using date/time functions in a query, see Using DateTime Functions in the *AVEVA Historian Concepts Guide.*

faaUser_ID

Returns the database user ID (in the Runtime database) for the current user, if the user has an individual login. Returns the database ID of the appropriate Windows security group, if the current user is a group member and does have an individual login (that is, the current user logs in by virtue of being a member of the Windows group).

Syntax

```
SELECT dbo.faaUser_ID()
```

Return Type

Integer.

Remarks

This function is used for processing annotations and for support of private namespaces.

- If a user has their own database login, the user has a completely private namespace and private annotations.
- If the user is part of a Windows security group, and logs in only by virtue of being part of the group, the private namespace and annotations are shared with all members of that group.

This stored procedure assumes that Windows users that are logged in are only members of a single Windows group (configured in SQL Server). If a user is found in more than one group, the ID of the last group found is used. This could be a problem if you are expecting a given user to have access to a particular private group.

Also, it is possible that annotations and namespace entries are created under names that you might not expect. For example, a user is a local administrator on a computer, and the user's login has also been added to the aaUsers local group. When logging in to SQL Server, the user will be mapped to the sysadmin fixed server role, by virtue of the user's membership in the BUILTIN\Administrators group. (This assumes that the BUILTIN\Administrators login has not been modified or disabled for security reasons). If this user creates annotations or private namespace entries, these appear as if they had been created by "dbo," rather than by a member of the local aaUsers group.

fww_GetLocalizedText

Returns the strings from the LocalizedText table for the requested language. If the specified translation is not found, English strings are returned.

Syntax

```
SELECT * FROM dbo.fww_GetLocalizedText(LangID)
```

Arguments

LangID
The locale ID for the language used. This ID is also used in the SQL Server syslanguages table. This value is of data type int.

Table Returned

The results are returned as a table that has the same columns as the LocalizedText table. However, the returned table will only include those rows containing strings in the specified language.

CHAPTER 6

Backward Compatibility Entities

Some entities are included in the database for backward compatibility support only. It is recommended that you discontinue the use of these entities, as they will be dropped in a future release.

Note: Utility extended stored procedures are no longer supported by Microsoft SQL Server nor by AVEVA Historian.

Backward Compatibility Views

Backward compatibility views include:

- *History Table Views (Backward Compatible)* on page 293
- *Summary Views* on page 300
- *NamedSystemParameter* on page 302
- *SystemNameSpace* on page 302
- *InSQLSysObjects* on page 303
- *v_ErrorLog* on page 304

History Table Views (Backward Compatible)

The following views reflect the same table structure as the extension tables for which they are named.

These views	Reference this extension table
AnalogHistory, v_AnalogHistory	INSQL.Runtime.dbo.AnalogHistory
AnalogLive, v_AnalogLive	INSQL.Runtime.dbo.AnalogLive
DiscreteHistory, v_DiscreteHistory	INSQLD.Runtime.dbo.DiscreteHistory
DiscreteLive v_DiscreteLive	INSQLD.Runtime.dbo.DiscreteLive
v_History	INSQL.Runtime.dbo.History
v_HistoryBlock	INSQL.Runtime.dbo.HistoryBlock
v_Live	INSQL.Runtime.dbo.Live
StringHistory, v_StringHistory	INSQL.Runtime.dbo.StringHistory
StringLive, v_StringLive	INSQL.Runtime.dbo.StringLive

To allow joins between the analog, string, and discrete tables, the analog and string views reference the OLE DB linked server "INSQL," while the discrete views reference the OLE DB linked server "INSQLD."

Note: In SQL Server Management Studio, the extension tables are listed under the INSQL or INSQLD linked servers under the Server objects tree item.

Tag Table Views

The following views are included for backward compatibility. They have the same names and structures as tables that were included in AVEVA Historian before version 2017. These views all reference the _Tag table:

- *AnalogSummaryTag*
- *AnalogTag*
- *DiscreteTag* on page 296
- *ReplicationTag* on page 298
- *StringTag* on page 299
- *StructureTag* on page 299

AnalogSummaryTag

Contains one row for each defined analog summary tag. (This is used exclusively for tiered historian installations.) Configuration information specific to analog summary tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) EUKey	int NOT NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float NOT NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float NOT NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float NULL	The minimum value of the raw acquired value.
MaxRaw	float NULL	The maximum value of the raw acquired value.

AnalogTag

Contains one row for each defined analog tag. Configuration information specific to analog tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) EUKey	int NOT NULL	The unique numerical identifier of an engineering unit. EUKey is a foreign key from the EngineeringUnit table.
MinEU	float NOT NULL	The minimum value of the tag, measured in engineering units.
MaxEU	float NOT NULL	The maximum value of the tag, measured in engineering units.
MinRaw	float NULL	The minimum value of the raw acquired value.
MaxRaw	float NULL	The maximum value of the raw acquired value.
Scaling	int NOT NULL	The type of algorithm used to scale raw values to engineering units. For linear scaling, the result is calculated using linear interpolation between the end points. 0 = None; 1 = Linear; 2 = Square Root. (Square root is reserved for future use).
RawType	int NOT NULL	The numeric type for the raw value. 1 = Euro Float, an outdated data type (4 bytes); 2 = MS Float (4 bytes); 3 = Integer (2 or 4 bytes); 4 = MS Double (reserved for future use) (8 bytes).
ValueDeadband	float NOT NULL	The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.
IntegerSize	tinyint NOT NULL	The bit size of the analog tag. 12 = 12-bit; 15 = 15-bit; 16 = 16-bit; 32 = 32-bit; 64 = 64-bit (reserved for future use).
SignedInteger	bit NOT NULL	Used to specify whether an integer is a signed number (positive or negative) or an unsigned number (positive only). 0 = Unsigned; 1 = Signed.
RateDeadband	float NOT NULL	The percentage of deviation in the full-scale value range for an analog tag. The swinging door (rate) deadband applies to delta storage only. Time and/or value deadbands can be used in addition to the swinging door deadband. Any value greater than 0 can be used for the deadband. A value of 0 indicates that a swinging door deadband will not be applied.

Column	Data Type	Description
InterpolationType	tinyint NOT NULL	The interpolation type for retrieval. 0 = Stair-stepped interpolation; 1 = Linear interpolation (if applicable, based on the tag type); 254 = System default interpolation mode. The system default interpolation type is to use the system default for the analog type, either integer or real. The system default interpolation type for an analog type is determined by the setting of the InterpolationTypeInteger and InterpolationTypeReal system parameters. This setting impacts Interpolated, Average, and Integral retrieval modes.
RolloverValue	float NOT NULL	The first value that causes the counter to "roll over." This rollover value is used by the "counter" retrieval mode. For example, a counter that counts from 0 to 9999, the counter rolls over back to 0 for the 10,000th value it receives. Therefore, set the rollover value to 10,000.

DiscreteTag

Contains one row for each defined discrete tag. Configuration information specific to discrete tags is stored in this table, while general information for all tag types is stored in the Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
(FK) MessageKey	int NOT NULL	The unique numerical identifier of a TRUE/FALSE message pair that can be associated with a discrete tag. MessageKey is a foreign key from the Message table.

EventTag

Contains one row for each event definition. Configuration information specific to event tags is stored in the _Tag table.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.

Column	Data Type	Description
(FK) DetectorTypeKey	int NULL	The unique identifier of a particular type of detector. Event tags and detectors are linked by means of this key. The event system relies on the following values, which are added during installation: 1 = System; 2 = External event; 3 = Generic SQL; 4 = Analog specific value; 5 = Discrete specific value; 6 = Time-based (schedule). DetectorTypeKey is a foreign key from the DetectorType table.
(FK) ActionTypeKey	int NULL	The unique identifier for a particular type of action. Event tags and actions are linked by this key. The event subsystem relies on the following values, which are added during installation: 1 = No action; 2 = Generic SQL; 3 = Snapshot; 4 = E-mail; 5 = Deadband; 6 = Summary. ActionTypeKey is a foreign key from the ActionType table.
ScanRate	int NULL	The interval, in milliseconds, at which the system checks to see if the event conditions specified by the detector occurred. This value must be greater than or equal to 500 milliseconds, and less than or equal to 1 hour (3600000 ms).
TimeDeadband	int NOT NULL	The minimum time, in milliseconds, between stored events. If more than one event occurs during the deadband, only the most recent will be stored. The system will not store another event until the specified time has elapsed. A time deadband of 0 indicates that the system will store all events. Reserved for future use.
Logged	bit NOT NULL	Used to specify whether or not to log events for this tag into the EventHistory table. Event logging can only be turned off if no associated actions are configured. 0 = Not logged; 1 = Logged. The default is 1.
Status	tinyint NOT NULL	The flag used by the event system at system startup and during runtime to determine if the event tag has been modified. 0 = Posted. Any changes have been detected and effected by the system. 1 = New. An event tag has been inserted, but is not yet executing. 2 = Modification. An event tag has been updated, but the older one is already executing. 98 = Disabled. 99 = Disabling requested. The event tag does not execute, even though the definition still exists in the schema. Note that there may be a delay of up to 30 seconds before a change in an event tag is seen by the running system.

Column	Data Type	Description
PostDetectorDelay	int NOT NULL	The amount of time, in milliseconds, that must elapse after an event is detected before the event action can be executed.
UseThreadPool	bit NOT NULL	Used to specify how system threads are used to process events. 1 = All events are handled by a single thread and a single logon to the SQL Server; 0 = Each event uses a separate system thread and logon. This will allow the event subsystem to manage the scan rates of each detector component concurrently. (Reserved for future use.)
DetectorString	nvarchar(1500) NULL	The script that contains the criteria for event detection. Detector scripts are executed on the local AVEVA Historian.
ActionString	nvarchar(1500) NULL	The script that specifies the event action. Action scripts run on the local AVEVA Historian.
Priority	tinyint NOT NULL	The priority level for the action, either critical or normal. The priority level determines the sorting queue to which the action will be sent. The critical queue is used for highly important events. If a system overload condition occurs, events that are given a critical priority will always be processed first. Events that are given a normal priority will be processed after any critical events and may possibly be dropped (that is, not performed) on an overloaded system. 0 = Normal; 1 = Critical. The default is 0.
Edge	tinyint NOT NULL	The "edge" for the event detection. 0 = Trailing; 1 = Leading; 2 = Both; 3 = None; 4 = Time Detector; 5 = External Detector.

ReplicationTag

Contains one row for each replication tag. (This is used exclusively for tiered historian installations.) Replication tags follow the same naming convention as regular tags.

Column	Data Type	Description
(PK) (FK) TagName	TagName Type (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
SourceTag	nvarchar(256) NOT NULL	The name of the source tag used for the replication tag.

Column	Data Type	Description
SourceServer	nvarchar(255) NOT NULL	The name of the tier 1 server with the source tag.

StringTag

Contains one row for each defined string tag. Configuration information specific to string tags is stored in this table, while general information for all tag types is stored in the Tag table.

Element	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.
MaxLength	smallint NOT NULL	The maximum number of characters for the string. Valid values are: 8, 16, 24, 32, 48, 64, 128, 131, 256, 512.
DoubleByte	tinyint NOT NULL	Used to store the string as a double-byte string. 0 = Not stored as double-byte; 1 = Stored as double-byte. The default is 0.

StructureTag

Contains one row for each summary tag.

Column	Data Type	Description
(PK) (FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique numerical identifier for a SQL template. TagName is a foreign key from the Tag table.
(FK) StructureId	uniqueidentifier NOT NULL	The unique identifier for the structure. StructureId is a foreign key from the StructureType table.

Alarm and Event Views (Backward Compatible)

The following Alarm and Event views are included for backward compatibility:

- Events *
- v_EventHistory *
- v_AlarmHistory *
- v_AlarmHistory2 *
- v_AlarmEventHistory2 *

- v_AlarmEventHistoryInternal2 *

* These views reflect the tables within the WWALMDB and A2ALMDB databases, or history blocks. For details about the related and tables, see "Recording Alarms into an Alarm Database" in the *AVEVA InTouch HMI Alarms and Events Guide*.

The columns of these views are generally compatible between history blocks, and the WWALMDB and A2ALMDB databases, except for the following exceptions in history blocks:

- User1 for alarms: Alarm Severity
- User2: Not populated
- User3: Not populated
- UnAckDuration: Format is in milliseconds.

Note: Earlier versions of AVEVA Historian stored alarm and event in history blocks. They could alternatively be stored in the A2ALMDB database. Earlier versions of AVEVA System Platform used WWALMDB database to store alarms and events. For more information, see A2ALMDB Database in the *AVEVA Historian Administration Guide*.

Summary Views

The summary views allow you to query for data that was summarized by the event subsystem. Each of the views contains data for a specific source, frequency, and operation.

View	Contains One Row For Each
DynDailyAvg	Daily average value for a tag.
DynDailySum	Daily summary value for a tag.
DynHourlyAvg	Hourly average value for a tag.
DynHourlyMax	Hourly maximum value for a tag.
DynHourlyMin	Hourly minimum value for a tag.
DynHourlySum	Hourly summary value for a tag.
v_SummaryData	Returns one row for each summarization of a tag for an associated summary event tag.

Each table view contains the following columns:

Column	Data type	Description
TagName	nvarchar(256) NOT NULL	The unique name of the tag within the AVEVA Historian system.
SummaryDate	datetime2(7) NOT NULL	The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.
Value	float NULL	The value of the summary.

Column	Data type	Description
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.

v_SummaryData

Returns one row for each summarization of a tag (specified by the TagName column) for an associated summary event tag (specified by the EventTag column). The resolution is applied to data before the calculation is performed. The quality value returned is the highest quality value of the raw data from which the result is calculated.

Column	Data type	Description
TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
CalcType	varchar(3) NULL	The type of calculation to be performed: SUM, MAX, MIN, or AVG.
SummaryDate	datetime2(7) NOT NULL	The date applicable to the results of the calculation. It is either the time of the beginning or end of the calculation period, as specified by the summary operation definition.
Value	float NULL	The value of the summary.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
Duration	real NULL	The period, in seconds, for which the calculation is performed.
Resolution	int NULL	The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.
TimeStamp	tinyint NULL	The timestamp to use when storing the result of the calculation. The timestamp can be either the time when the calculation period starts or ends.

Column	Data type	Description
EventTag	TagNameType(nvarchar(256)) NOT NULL	The name of the event tag to which the snapshot tag is related.

NamedSystemParameter

Contains one row for each system parameter. This view provides backward compatibility support for the NamedSystemParameter table, which has been replaced by the SystemParameter table.

Column	Data type	Description
Name	nvarchar(50) NOT NULL	The unique name for the system parameter.
Type	varchar(7) NULL	Used to specify the datatype for the system parameter value. Valid values are: NUMERIC, STRING.
StringValue	varchar(255) NULL	The value of the system parameter. This column only contains values of type STRING.
NumericValue	real NULL	The value of the system parameter. This column only contains values of type NUMERIC.
Editable	bit NULL	Used to determine if the value of the named system parameter can be changed using the InSQL Console. 1 = Editable; 0 = Not editable.
Description	nvarchar(255) NULL	The description of the system parameter.

SystemNameSpace

Contains one row for each item in a single system namespace. Items in the system namespace include servers, topics, and users. The items are organized in a hierarchy. This view provides backward compatibility support for the SystemNameSpace table, which has been deleted.

Column	Data type	Description
NameKey	int NULL	The unique identifier for the object in the namespace.

Column	Data type	Description
Type	int NOT NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
ParentKey	int NULL	The unique identifier for a named object in this namespace.
Name	nvarchar(290) NULL	The name of this object in the hierarchy.
PKey	int NULL	The primary key reference for other tables.

InSQLSysObjects

Contains one row for each object in the database for which changes can be tracked. This view provides backward compatibility support for the InSQLSysObjects table, which has been renamed to HistorianSysObjects.

Column	Data Type	Description
id	int NOT NULL	The unique identifier for the object.
Type	char(2) NULL	The type of object. C = CHECK constraint; D = Default or DEFAULT constraint; F = FOREIGN KEY constraint; K = PRIMARY KEY or UNIQUE constraint; L = Log; P = Stored procedure; R = Rule; RF = Stored procedure for replication; S = System table; TR = Trigger; U = User table; V = View; X = Extended stored procedure. Currently, only changes for the user tables (object type U) are tracked.
Name	varchar(50) NULL	The name of the modified object.

v_ErrorLog

Contains one row for each system message (error message), if this functionality was enabled. By default, this table is not used.

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The date that the message was written to the system log, in the local time of the AVEVA Historian.
Type	nvarchar(10) NULL	The type of system message.
LocalizedText	nvarchar(256) NULL	The content of the message.
Parameter	nvarchar(256) NULL	Optional details pertaining to the message text. For example, for the message "Disk space remaining on circular path" the parameter would contain the number of MB.
TotalCount	int NULL	Used to prevent "flooding" conditions in the log file. If a particular message is generated numerous times during a relatively short period of time, the message is written to the log file only once, and the total number of times that it occurred appears in this column.
ModuleID	int NULL	A unique number assigned to the AVEVA Historian subsystem that generated the message.
Host	nvarchar(32) NULL	The computer on which the AVEVA Historian subsystem runs.
FileName	nvarchar(64) NULL	Used to indicate the program file that contains the line of code that an error message comes from. Used for debugging.
Line	int NULL	Used to indicate the line of code that an error message comes from. Used for debugging.

Backward Compatibility Tables

The backward compatibility tables include:

- *AnalogHistory (INSQL.Runtime.dbo.AnalogHistory)*
- *AnalogLive (INSQL.Runtime.dbo.AnalogLive)*
- *AnalogWideHistory*
- *DiscreteHistory (INSQL.Runtime.dbo.DiscreteHistory)* on page 307

AnalogHistory (INSQL.Runtime.dbo.AnalogHistory)

This table has been superceded by the *History (INSQL.Runtime.dbo.History)* on page 78 table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(16) NULL
wwTimeDeadband	int NULL
wwValueDeadband	float NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL

Column	Data type
wwTimeStampRule	nvarchar(20) NULL
wwInterpolationType	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

AnalogLive (INSQL.Runtime.dbo.AnalogLive)

This table has been superceded by the *Live (INSQL.Runtime.dbo.Live)* on page 92 table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRetrievalMode	nvarchar(16) NULL
wwTimeDeadband	int NULL
wwValueDeadband	float NULL
wwTimeZone	nvarchar(50) NULL
wwParameters	nvarchar(128) NULL

AnalogWideHistory

This table (INSQL.Runtime.dbo.AnalogWideHistory) has been superceded by the *WideHistory (INSQL.Runtime.dbo.WideHistory)* on page 138 table. AnalogWideHistory is the wide version of AnalogHistory. In a query, this table must be referenced using an OPENQUERY statement.

Column	Data type
DateTime	datetime NOT NULL

Column	Data type
TagA1	float NULL
TagA2	float NULL
ManyOtherTags	float NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(20) NULL (default wwRetrievalMode is CYCLIC)
wwTimeDeadband	int NULL
wwValueDeadband	real NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwInterpolationType	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

DiscreteHistory (INSQL.Runtime.dbo.DiscreteHistory)

This table has been superceded by the *History (INSQL.Runtime.dbo.History)* on page 78 table.

Column	Data type
DateTime	datetime NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL

Column	Data type
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(16) NULL
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

DiscreteLive (INSQL.Runtime.dbo.DiscreteLive)

This table has been superceded by the *Live (INSQL.Runtime.dbo.Live)* on page 92 table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	float NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRetrievalMode	nvarchar(16) NULL The default mode is DELTA. No other retrieval mode is allowed.
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwParameters	nvarchar(128) NULL

DiscreteWideHistory

This table (INSQL.Runtime.dbo.DiscreteWideHistory) has been superceded by the WideHistory_OLEDB table. DiscreteWideHistory is the wide version of DiscreteHistory, where only discrete tags are considered. It is the same as WideHistory applied to discrete tags. In a query, this table must be referenced using an OPENQUERY statement.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagD1	tinyint NULL
TagD2	tinyint NULL
ManyOtherTags	tinyint NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(20) NULL The default is DELTA.
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

GroupTagList

Contains one row for each defined group of tags.

Column	Data type	Description
(PK) GroupID	int NOT NULL	The identifier for a group of tags.
(PK) wwDomainTagKey	int NOT NULL	The unique numerical identifier for a tag in a specific domain.

Column	Data type	Description
Triggerval	float NULL	A value that can be read by an application as a trigger value.

ManualAnalogHistory

This table can be used by custom client applications to store values for analog tags. By default, this table is empty. If written to by a client application, this table will contain one row for each defined analog tag per sample period. ManualAnalogHistory is a normal SQL Server table and does not support any of the AVEVA Historian extensions for handling data.

Column	Data type	Description
(PK) DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data was acquired or stored.
(FK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	float NULL	The value of the tag at the timestamp. Measured in engineering units.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualDiscreteHistory

This table can be used by custom client applications to store values for discrete tags. By default, this table is empty. If written to by a client application, this table will contain one row for each defined discrete tag per sample period. ManualDiscreteHistory is a normal SQL Server table and does not support any of the AVEVA Historian extensions for handling data.

Column	Data type	Description
(PK) DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data was acquired or stored.
(PK) TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	tinyint NULL	The value of the discrete tag at timestamp. 0 = FALSE; 1 = TRUE; NULL = No data.

Column	Data type	Description
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

ManualStringHistory

This table can be used by custom client applications to store values for string tags. By default, this table is empty. If written to by a client application, this table will contain one row for each defined string tag per sample period. ManualStringHistory is a normal SQL Server table and does not support any of the AVEVA Historian extensions for handling data.

Column	Data type	Description
DateTime	datetime2(7) NOT NULL	The timestamp reflecting when the data was acquired or stored.
(PK) TagName	TagNameType(nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system.
Value	nvarchar(512) NULL	The value of the string tag at the timestamp.
Quality	tinyint NOT NULL	The basic data quality indicator associated with the data value.
QualityDetail	int NULL	An internal representation of data quality.
wwTagKey	int NOT NULL	The unique numerical identifier of a tag within a single AVEVA Historian.

NameSpaceIcons

Contains one row for each defined namespace icon. Namespace icons can be shown in an application browser for each level of the namespace (system, public, and private).

Column	Data type	Description
(PK) Type	int NOT NULL	The value that specifies the type of namespace. 1 to 6 = Tag; 1 to 2 million = System; 2+ million = Groups. Within the system range, the following values designate ArchestrA object types: 1999023 = Galaxy; 1999001 = WinPlatform object; 1999003 = AppEngine object; 1999013 = Area object; 1999011 = DDESuiteLinkClient, OPCClient, and InTouchProxy objects; 1999024 = RedundantDIObject object; 1999033 = Undeployed object represented by a generic name; 1999901 = ApplicationObject; 1999902 = Traceability object.
Icon	image NULL	The icon for the namespace.
Name	nvarchar(30) NOT NULL	The name of the icon.
Description	nvarchar(50) NULL	The description of the icon.

StringHistory (INSQL.Runtime.dbo.StringHistory)

This table has been superceded by the History table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	nvarchar(512) NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(16) NULL

Column	Data type
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwQualityRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

StringLive (INSQL.Runtime.dbo.StringLive)

This table has been superceded by the *Live (INSQL.Runtime.dbo.Live)* on page 92 table.

Column	Data type
DateTime	datetime2(7) NOT NULL
TagName	nvarchar(256) NOT NULL
Value	nvarchar(512) NULL
Quality	tinyint NOT NULL
QualityDetail	int NULL
OPCQuality	int NULL
wwTagKey	int NOT NULL
wwRetrievalMode	nvarchar(16) NULL The default mode is DELTA. No other retrieval mode is allowed.
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwParameters	nvarchar(128) NULL

StringWideHistory

This table (INSQL.Runtime.dbo.StringWideHistory) was implemented for completeness. However, you should use the WideHistory table instead of this table. In a query, this table must be referenced using an OPENQUERY statement.

Column	Data type
DateTime	datetime NOT NULL
TagS1	nvarchar(512) NULL
TagS2	nvarchar(512) NULL
ManyOtherTags	nvarchar(512) NULL
wwRowCount	int NULL
wwResolution	int NULL
wwEdgeDetection	nvarchar(16) NULL
wwRetrievalMode	nvarchar(20) NULL (default wwRetrievalMode is DELTA)
wwTimeDeadband	int NULL
wwTimeZone	nvarchar(50) NULL
wwVersion	nvarchar(30) NULL
wwCycleCount	int NULL
wwTimeStampRule	nvarchar(20) NULL
wwParameters	nvarchar(128) NULL

TagGroup

Contains one row for each defined tag group. A tag group is a simple, non-hierarchical grouping of tags that can be used by the system.

Column	Data type	Description
(PK) GroupID	int NOT NULL	The identifier for a group of tags.
Description	nvarchar(50) NULL	The description for the group of tags.
CreatedDate	datetime2(7) NULL	The creation date for the tag grouping.
CreatedBy	nchar(18) NULL	The name of the user or application that created the group of tags.

Column	Data type	Description
Type	int NULL	The type of tag group. 1 to 100 = System use. 100 = Users and third-party client applications.

WideTableDictionary

Contains one row of values for up to 249 tags. These 249 tags appear as columns that will be visible in the data dictionary for each user. Mainly used by ad-hoc query tools. Does not affect the ability to access the values stored for a tag.

Column	Data type	Description
(PK) UserKey	int NOT NULL	The unique numerical identifier for a database user as defined in the UserDetail table. UserKey is a foreign key from the UserDetail table.
(PK) TagName	TagNameType (nvarchar(256)) NOT NULL	The unique name of the tag within the AVEVA Historian system. TagName is a foreign key from the Tag table.

Renamed Tables

The following table has been renamed. A view named *InSQLSysObjects* on page 303 has been created for backward compatibility.

Old Name	New Name
InSQLSysObjects	HistorianSysObjects

Backward Compatibility Stored Procedures

Stored procedures that have been retained for backward compatibility are:

- *aaAnalogDetail*
- *aaDiscreteDetail*
- *aaStringDetail*
- *ww_CheckClientVersion* on page 317
- *ww_CheckWhichDb* on page 317
- *ww_dbCheck* on page 318

- *ww_LoadInSQLProcedureBody* on page 318
- *ww_MDASAnalogTagInsert* on page 319
- *ww_MDASAnalogTagUpdate* on page 319
- *ww_MDASDiscreteTagInsert* on page 319
- *ww_MDASDiscreteTagUpdate* on page 319
- *ww_MDASStringTagInsert* on page 319
- *ww_MDASStringTagUpdate* on page 319

Backward compatibility extended stored procedures are discussed later in this chapter.

aaAnalogDetail

Returns information about one or more specified analog tags, including the name of the tag, a description, the acquisition rate, the engineering unit, and the minimum and maximum values in engineering units.

Syntax

aaAnalogDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaDiscreteDetail

Returns information about one or more specified discrete tags, including the name of the tag, a description, the message for the TRUE (1) state of the tag, and the message for the FALSE (0) state of the tag.

Syntax

aaDescreteDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

aaStringDetail

Returns a description for one or more specified tags.

Syntax

aaStringDetail *TagList*

where:

Argument	Description
TagList	A list of tags delimited by a comma (,). This value is of data type nvarchar(4000), with no default.

Permission

Execute permission defaults to the public group.

ww_CheckClientVersion

Checks which version of the client application is running.

Syntax

```
ww_CheckClientVersion AppName, AppVersion
```

Arguments

AppName
The name of the application. This value is of data type nvarchar(255), with a default of an empty string.

AppVersion
The version number of the application. This value is of data type nvarchar(255), with a default of an empty string.

Remarks

This stored procedure is used by the AVEVA Historian to ensure that a version of an AVEVA client application will run against the database. A client application will not be allowed to run against a database version that does not support that client.

Important: This stored procedure is for AVEVA use only. Do not attempt to use this stored procedure for any third-party client application.

Permission

Execute permission defaults to the public group.

ww_CheckWhichDb

Used to determine if querying the correct database.

Syntax

```
ww_CheckWhichDb dbType
```

Arguments

dbType
The identifier for the database. 1 = Runtime; 2 = Holding; 3 = Development. This value is of data type int, with no default.

Permission

Execute permission defaults to the public group.

ww_dbCheck

Used to invalidate FactorySuite 1000 clients.

Syntax

```
ww_dbCheck dbType
```

Arguments

dbType

The identifier for the database. 1 = Runtime; 2 = Holding; 3 = Development.

This value is of data type int, with no default.

Remarks

This stored procedure is only used by client applications released prior to FactorySuite 2000.

Permission

Execute permission defaults to the public group.

ww_DBConfig

Returns a summary of the current database configuration, such as number of tags, number of tags per type, storage configuration, event tags, and summary configuration.

Syntax

```
ww_DBConfig
```

Permission

Execute permission defaults to the public group.

ww_LoadInSQLProcedureBody

Used internally to track which stored procedures reference the extension tables.

Syntax

```
ww_LoadSQLProcedureBody ObjName
```

Arguments

ObjName
The name of the stored procedure to load. This value is of data type varchar(92), with no default.

Permission

Execute permission defaults to the public group.

ww_MDASAnalogTagInsert

Used by the Manual Data Acquisition Service to add an analog tag.

This stored procedure calls the aaInternalMDASAnalogTagInsert stored procedure, which is for internal use only.

ww_MDASAnalogTagUpdate

Used by the Manual Data Acquisition Service to update an analog tag.

This stored procedure calls the aaInternalMDASAnalogTagUpdate stored procedure, which is for internal use only.

ww_MDASDiscreteTagInsert

Used by the Manual Data Acquisition Service to add a discrete tag.

This stored procedure calls the aaInternalMDASDiscreteTagInsert stored procedure, which is for internal use only.

ww_MDASDiscreteTagUpdate

Used by the Manual Data Acquisition Service to update a discrete tag.

This stored procedure calls the aaInternalMDASDiscreteTagUpdate stored procedure, which is for internal use only.

ww_MDASStringTagInsert

Used by the Manual Data Acquisition Service to add a string tag.

This stored procedure calls the aaInternalMDASStringTagInsert stored procedure, which is for internal use only.

ww_MDASStringTagUpdate

Used by the Manual Data Acquisition Service to update a string tag.

This stored procedure calls the aaInternalMDASStringTagUpdate stored procedure, which is for internal use only.

Renamed Stored Procedures

The following stored procedures have been renamed. The old stored procedures have been retained in the system for backward compatibility.

Old Name	New Name
ww_ActionStringSelect	aaAddAnalogSummaryTag
ww_AddTag	aaAddStructureTag
ww_AnalogDetail	aaAnalogDetail
ww_AnalogTagDelete	aaAnalogTagDelete
ww_AnalogTagInsert	aaAnalogTagInsert
ww_AnalogTagSelect	aaAnalogTagSelect
ww_AnalogTagUpdate	aaAnalogTagUpdate
ww_Annotation	aaAnnotationRetrieve
ww_AnnotationDelete	aaAnnotationDelete
ww_AnnotationInsert	aaAnnotationInsert
ww_AnnotationSelect	aaAnnotationSelect
ww_AnnotationUpdate	aaAnnotationUpdate
ww_CheckClientVersion	--
ww_CheckWhichDb	--
ww_CleanupAfterCommit	aaCleanupAfterCommit
ww_CommitChanges	aaCommitChanges
ww_CommitChangesAtStartup	aaCommitChangesAtStartup
ww_ContextDelete	aaContextDelete
ww_ContextInsert	aaContextInsert
ww_ContextSelect	aaContextSelect
ww_ContextUpdate	aaContextUpdate
ww_DBChangesPending	aaDBChangesPending
ww_dbCheck	--
ww_DBConfig	aaDBConfig
ww_DeleteOlderEvents	aaDeleteOlderEvents
ww_DeleteOlderSummaries	aaDeleteOlderSummaries
ww_DeleteTag	aaDeleteTag
ww_DetectorStringSelect	aaDetectorStringSelect
ww_DiscreteDetail	aaDiscreteDetail

Old Name	New Name
ww_DiscreteTagDelete	aaDiscreteTagDelete
ww_DiscreteTagInsert	aaDiscreteTagInsert
ww_DiscreteTagSelect	aaDiscreteTagSelect
ww_DiscreteTagUpdate	aaDiscreteTagUpdate
ww_EngineeringUnitDelete	aaEngineeringUnitDelete
ww_EngineeringUnitInsert	aaEngineeringUnitInsert
ww_EngineeringUnitSelect	aaEngineeringUnitSelect
ww_EngineeringUnitUpdate	aaEngineeringUnitUpdate
ww_EventDetection	aaEventDetection
ww_EventHistory	aaEventHistorySelect
ww_EventHistoryInsert	aaEventHistoryInsert
ww_EventSnapshot	aaEventSnapshotSelect
ww_EventSnapshotInsert	aaEventSnapshotInsert
ww_EventTagDelete	aaEventTagDelete
ww_EventTagDetail	aaEventTagDetail
ww_EventTagInsert	aaEventTagInsert
ww_EventTagSelect	aaEventTagSelect
ww_EventTagSelectAll	aaEventTagSelectAll
ww_EventTagSelectDeleted	aaEventTagSelectDeleted
ww_EventTagSelectDisabled	aaEventTagSelectDisabled
ww_EventTagSelectInserted	aaEventTagSelectInserted
ww_EventTagSelectUpdated	aaEventTagSelectUpdated
ww_EventTagUpdate	aaEventTagUpdate
ww_GetDbRevision	aaGetDbRevision
ww_GetLastTagKey	aaGetLastTagKey
ww_HistoryBlockSelect	aaHistoryBlockSelect
ww_InSQLConfigNSExpand	aaHistorianConfigNSExpand
ww_InSQLNSExpand	aaHistorianNSExpand
ww_InSQLStatusSelect	aaHistorianStatusSelect

Old Name	New Name
ww_InSQLStatusSet	aaHistorianStatusSet
ww_InTouchNodeTagList	aaInTouchNodeTagList
ww_IODriverDelete	aaIODriverDelete
ww_IODriverInsert	aaIODriverInsert
ww_IODriverSelect	aaIODriverSelect
ww_IODriverUpdate	aaIODriverUpdate
ww_IOServerDelete	aaIOServerDelete
ww_IOServerInsert	aaIOServerInsert
ww_IOServerSelect	aaIOServerSelect
ww_IOServerTypeDelete	aaIOServerTypeDelete
ww_IOServerTypeInsert	aaIOServerTypeInsert
ww_IOServerTypeSelect	aaIOServerTypeSelect
ww_IOServerTypeUpdate	aaIOServerTypeUpdate
ww_IOServerUpdate	aaIOServerUpdate
ww_LimitDelete	aaLimitDelete
ww_LimitInsert	aaLimitInsert
ww_LimitNameDelete	aaLimitNameDelete
ww_LimitNameInsert	aaLimitNameInsert
ww_LimitNameSelect	aaLimitNameSelect
ww_LimitNameUpdate	aaLimitNameUpdate
ww_LimitSelect	aaLimitSelect
ww_LimitUpdate	aaLimitUpdate
ww_LoadInSQLProcedureBody	--
ww_MessageDelete	aaMessageDelete
ww_MessageInsert	aaMessageInsert
ww_MessageSelect	aaMessageSelect
ww_MessageUpdate	aaMessageUpdate
ww_ModLogStatus	aaModLogStatus
ww_PrivateNSAddGroup	aaPrivateNSAddGroup

Old Name	New Name
ww_PrivateNSAddLeaf	aaPrivateNSAddLeaf
ww_PrivateNSDeleteGroup	aaPrivateNSDeleteGroup
ww_PrivateNSDeleteLeaf	aaPrivateNSDeleteLeaf
ww_PrivateNSExpand	aaPrivateNSExpand
ww_PrivateNSSelect	aaPrivateNSSelect
ww_PrivateNSUpdateGroup	aaPrivateNSUpdateGroup
ww_PublicNSAddGroup	aaPublicNSAddGroup
ww_PublicNSAddLeaf	aaPublicNSAddLeaf
ww_PublicNSDeleteGroup	aaPublicNSDeleteGroup
ww_PublicNSDeleteLeaf	aaPublicNSDeleteLeaf
ww_PublicNSExpand	aaPublicNSExpand
ww_PublicNSSelect	aaPublicNSSelect
ww_PublicNSUpdateGroup	aaPublicNSUpdateGroup
ww_RedirectToInTouch	aaRedirectToInTouch
ww_SetStorageRule	aaSetStorageRule
ww_SetTagStorage	aaSetTagStorage
ww_SnapshotDetailSelect	aaSnapshotDetailSelect
ww_SnapshotDetailUpdate	aaSnapshotDetailUpdate
ww_SnapToSummary	aaSnapToSummary
ww_SpaceManager	aaSpaceManager
ww_StorageLocationSelect	aaStorageLocationSelect
ww_StorageLocationUpdate	aaStorageLocationUpdate
ww_StringDetail	aaStringDetail
ww_StringTagDelete	aaStringTagDelete
ww_StringTagInsert	aaStringTagInsert
ww_StringTagSelect	aaStringTagSelect
ww_StringTagUpdate	aaStringTagUpdate
ww_SummaryActionInsert	aaSummaryActionInsert
ww_SummaryDetail	aaSummaryDetail

Old Name	New Name
ww_SummaryOperationDelete	aaSummaryOperationDelete
ww_SummaryOperationInsert	aaSummaryOperationInsert
ww_SummaryOperationSelect	aaSummaryOperationSelect
ww_SummaryOperationUpdate	aaSummaryOperationUpdate
ww_SummaryTagListDelete	aaSummaryTagListDelete
ww_SummaryTagListInsert	aaSummaryTagListInsert
ww_SummaryTagListSelect	aaSummaryTagListSelect
ww_SummaryTagListUpdate	aaSummaryTagListUpdate
ww_SystemConfigNSExpand	aaSystemConfigNSExpand
ww_SystemNSExpand	aaSystemNSExpand
ww_SystemNSExpand2	aaSystemNSExpand2
ww_SystemParameterSelect	aaSystemParameterSelect
ww_SystemParameterUpdate	aaSystemParameterUpdate
ww_TagConfig	aaTagConfig
ww_TagConfigModified	aaTagConfigModified
ww_TagConfigSelect	aaTagConfigSelect
ww_TagInfo	aaTagInfo
ww_TagType	aaTagType
ww_TimeDetectorDetailInsert	aaTimeDetectorDetailInsert
ww_TimeDetectorDetailSelect	aaTimeDetectorDetailSelect
ww_TimeDetectorDetailUpdate	aaTimeDetectorDetailUpdate
ww_TopicDelete	aaTopicDelete
ww_TopicInsert	aaTopicInsert
ww_TopicSelect	aaTopicSelect
ww_TopicUpdate	aaTopicUpdate
ww_UpdateCalculatedAISamples	aaUpdateCalculatedAISamples
ww_UserAccessLevelSelect	aaUserAccessLevelSelect
ww_UserDetailUpdate	aaUserDetailUpdate

Extended Stored Procedure Arguments

Note: Extended stored procedures are no longer supported.

Most of the extended stored procedures for the AVEVA Historian use one or more of the following arguments:

StartTime, EndTime
The StartTime string value represents the starting timestamp for the data to query. The EndTime string value represents the ending timestamp for the data to query. The date/time value can be any valid SQL Server date/time string.
The notion of specifying a time zone is not supported. All date/time strings passed as parameters to an extended stored procedure are considered as local server time.
For start and end times, the GetDate() and DateAdd(...) functions are supported, as well as literal dates. For more information, see *Literal Date Expressions* on page 326, *GetDate() Expressions* on page 326, and *DateAdd(...) Expressions* on page 326.
The extended stored procedures round timestamps up or down to the next supported millisecond value: 0, 3, or 7. The standard four-part query and open query do not round timestamps, so if you retrieve data with the extended stored procedure and the four-query, you can have different timestamps for the same data value.

Resolution
The sampling rate, in milliseconds, for retrieving the data in cyclic mode. The system returns values stored over the requested time period at the interval specified by the resolution. For example, if you specify a 5000 ms resolution, the system queries for all data during the time period and then only returns those values that occur at each 5000 ms interval, starting with the start date and ending with the end date.

MaxRowCount
The maximum number of rows to be returned for a specified time period.

ValueDeadBand
The percentage of the difference between the minimum and maximum engineering units for the tag. Any data values that change less than the specified deadband are not stored. The value deadband applies to delta storage only. A value of 0 indicates that a value deadband will not be applied.

TimeDeadband
The minimum time, in milliseconds, between stored values for a single tag. Any value changes that occur within the time deadband are not stored. The time deadband applies to delta storage only. A time deadband of 0 indicates that the system will store the value of the tag each time it changes.

Description
The description of the history data that will be copied.

TagN
Tag1, Tag2… TagN are tagname values. Each tagname must be comma delimited and can optionally be surrounded with quotes. For example:

```
exec xp_AnalogHistory "DateAdd(HOUR, -1, GetDate())", "GetDate()", 1000,
SysTimeSec, SysTimeMin

exec xp_DiscreteHistory "DateAdd(HOUR, -1, GetDate())", "GetDate()", 1000,
"SysPulse"
```

If a tagname is not of the same type as expected for the named stored procedure then it is ignored. For example, if you pass an analog tagname as a parameter to xp_DiscreteHistory, it will be ignored. For information regarding valid tagnames, see "Naming Conventions for Tagnames" in Chapter 2, "System-Level Concepts and Functionality" in the AVEVA Historian Concepts Guide.

Literal Date Expressions

Note: Extended stored procedures are no longer supported.

Date expressions can be any valid SQL Server date expression. Here are some examples

```
"4/2/2001 13:00:00:00"
"4/2/2001 12:00 PM"
"2001-4-2 1:00 AM"
```

Years expressed as two digits are interpreted as years in the 1900s. The SQL Server configuration option that supports a two-digit year cutoff is not used.

GetDate() Expressions

Note: Extended stored procedures are no longer supported.

Date/time values can have a string value expression containing the string "GetDate". This is not the same as the SQL Server GetDate() function, although the effect is the same. Example expressions are:

```
GetDate
"GetDate"
"GetDate()"
```

For example:

```
exec xp_AnalogHistory GetDate, "GetDate()", 1000, 'SysTimeSec'
```

DateAdd(...) Expressions

Note: Extended stored procedures are no longer supported.

Date/time values can have a string value expression containing the string "DateAdd(...)". This is not the same as the SQL Server DateAdd() function, although the effect is very similar.

Syntax

```
"DATEADD (datepart, number, date)"
```

Parameters

DatePart
Specifies on which part of the date to return a new value. The following table lists the date parts and abbreviations recognized by the extended stored procedures for the AVEVA Historian.

MILLISECOND	MS
SECOND	SS
MINUTE	MI
HOUR	HH
DAY	DD
WEEKDAY	DW
WEEK	WK

DAYOFYEAR	DY
MONTH	MM
QUARTER	QQ
YEAR	YY

Number
The value used to increment datepart. If you specify a value that is not an integer, the fractional part of the value is discarded. For example, if you specify day for datepart and 1.75 for number, date is incremented by 1.

Date
Either a literal date value without quotes (see *Literal Date Expressions* on page 326) or a GetDate() expression also without quotes (see *GetDate() Expressions* on page 326).

Example Expressions

```
"DateAdd(HOUR, -1, GetDate())"
"DateAdd(MINUTE, -30, 4/2/2001 13:00:00:00)"
```

Extended Stored Procedure Date Expression Examples

```
xp_DiscreteHistory "DateAdd(HOUR, -1, GetDate())", "GetDate()", 1000,
'SysPulse'
xp_DiscreteHistoryDelta "DateAdd(DAY, -1, 4/2/2001)", "GetDate()", 100,
'SysPulse'
```

Backward Compatibility Functions

The following functions have been renamed. The old functions have been retained in the system for backward compatibility.

Old Name	New Name
fww_CheckLicenseViolation	faaCheckLicenseViolation
fww_GetLocalizedText	faaGetLocalizedText
fww_InSQLgetdate	faaTZgetdate
fww_LicensedTagDetails	faaLicensedTagDetails
fww_LicensedTagTotal	faaLicensedTagTotal
fww_TagsInLicenseViolation	faaTagsInLicenseViolation