



Myths about Historians

Author: Elliot Middleton, Historian Product Manager, Invensys Operations Management

What's Inside:

1. Introduction
2. Myth #1: Storage is so cheap that efficiency doesn't matter
3. Myth #2: Relational databases are fast enough
4. Myth #3: Managing time-series data in a relational database is trivial
5. Myth #4: Retrieving time-series data is no different from any other type of data
6. Myth #5: The only options are fully relational or fully proprietary historian solutions
7. Myth #6: There is nothing special about industrial applications
8. Myth #7: Relational database applications aren't historians
9. Myth #8: All data is equal in importance and validity

Myths about Historians

Collected from a number of blogs by Elliott Middleton and currently posted as blogs on WDN at:

<https://wdn.wonderware.com/sites/WDN/Lists/Historian%20%20Information%20Server%20Updates%20Blog/AllItems.aspx>

1. Introduction

Historian applications have been around a long time, long enough so that certain preconceived notions, unfounded impressions, or 'myths' have developed through the years that seem to perpetuate without substance, challenge, or evidence.

This white paper will cover the merits and shortfalls of relational databases as well as uncover and challenge these myths in the context of process historian applications without getting into the back end detailed technology. Review the basics of historian applications, discover what to look for and consider during an evaluation, and properly set your expectations on the value a historian can offer to fulfill your needs.

2. Myth #1: Storage is so cheap that efficiency doesn't matter

Just because the New York Stock Exchange trading systems and other high-throughput applications use a relational database doesn't mean they're right for everything. Take time-series data, for example: can SQL Server or Oracle store time-series data? Certainly. Are there some problems you should know about before you try it? Absolutely.

Be sure you really understand how much data a typical process actually generates. A modest 5,000 tag historian logging data every second generates 157 billion values per year. Stored efficiently in 8 bytes each, that's roughly a terabyte a year. In some tests comparing SQL Server storage requirements for time-series data with those of Wonderware® Historian® the difference was 50:1 when including the necessary indices. Even though storage prices are falling, 50 terabytes of data a year is still a lot. Also recognize that it isn't just a matter of having enough disk space to hold that much data, but most historian applications also need that data protected, multiplying the amount of storage for backups or disk mirroring. Some industries even have regulatory requirements for several years' worth of data, further amplifying the storage need.

3. Myth #2: Relational databases are fast enough

As hardware price-performance has improved relational databases have benefited. However, relational databases are designed to protect referential integrity around "transactions" that may update multiple table values in unison, which adds significant overhead. For example, on high-end hardware (running 64 Itanium processors) SQL Server 2008 established a world record 1126 transactions per second. Granted such transactions are not the same as those required for a historian, but even such high-end hardware would be taxed to store 5,000 values per second if each value was a transaction. This means a front end buffering application must collect the data and stream numerous values into the database as a single transaction. Other databases without full transactional support, such as MySQL's freeware MyISAM storage engine, can support higher throughputs, but still require a front end buffer to achieve adequate throughput for all but the tiniest historian applications.

Of course, the reason to store data in the first place is so that it is available to retrieve later. Naturally, that makes retrieval performance quite important, too. Particularly in general-purpose solutions like a relational database, it is possible to organize data so that it is either efficient to store (higher throughput) or efficient to retrieve (fast retrieval), but not both. Efficient retrieval of time-series data from a general purpose databases requires use of a "clustered index," something not available, for example, in the higher-throughput MyISAM storage engine.

In contrast, purpose-built storage engines designed specifically for time-series data leverage a knowledge of how data is collected and consumed to store it efficiently for both—this would not be possible if the data were more generalized.

4. Myth #3: Managing time-series data in a relational database is trivial

Relational databases are designed to accumulate massive amounts of data. However, as the amount of data grows, so do query execution times, the size of backups, and numerous other routine operations. To alleviate this performance problem of ever-growing tables database administrators must routinely purge data from the database. In any database that protects transactional integrity, this purge operation must suspend normal database updates—that's a problem for historian applications running 24/7/365. To even make the purge operation itself tolerable requires minimizing the amount of data maintained in the database.

Myths about Historians

In the event purged data is needed later (for example, in response to an audit or some regulatory demands), restoring the data is non-trivial. The generally recommended practice is to restore a full database backup that included the needed data either to a separate system dedicated for this purpose, or to take your production system offline and use it. This is even more problematic if the required data isn't available within a single database backup—for example, if you only maintain the last 30 days of data in the online database and the audit requires 90 days of data, you must either manually merge all the data into a single database, have three systems, each with an isolated 30-day window, or examine each backup serially.

True historians, on the other hand, are designed to both handle the rapid growth in data and to provide simple means of taking subsets of the data offline and online.

5. Myth #4: Retrieving time-series data is no different from any other type of data

With all the power of Structured Query Language (SQL) to query data, some may claim that relational databases are just as good at retrieving time-series data as they are transactional data. Its certainly true SQL gives great flexibility, but it is based on some fundamental assumptions that don't apply to time-series data: a) there is no inherent order in the data records [in fact, time-series data is ordered by time], b) all the data is explicitly stored [in fact, most historian data only represents samples from a continuum of the real data], all data is of equal significance.

These two differences are significant. For example, if an instrument reports a value timestamped at "7:59:58.603" and a user queries a relational database for the value at "8:00:00.000," no data will be returned since there is no records stored for precisely that time—the database does not recognize that time is a continuum. Similarly, if a temperature was "21.0° C" and two-minutes later was "23.0° C", it has no inherent ability to infer that halfway between these samples the temperature was approximately "22.0° C".

In historian applications, it is rarely steady-state operations that are most significant. If the only way for a client application to find exceptions is to query all of the data for a measurement, that can place a heavy load on the overall system: server, network and client. In contrast, historians generally have means of filtering out insignificant data (based on comparing sequential records) to radically reduce the volume of data that must be delivered to client applications.

6. Myth #5: The only options are fully relational or fully proprietary historian solutions

While its true that most historian solution either use fully proprietary technology to address the inherent limitations of relational database or fully leverage relational database to reduce their own engineering costs, Wonderware Historian actually delivers the best of both worlds. It relies on a solid relational schema for managing all the relatively static configuration data, but extends the native transactional storage engine and query processor of Microsoft SQL Server with proprietary extensions to address their limitations for time-series data.

Building on Microsoft SQL Server delivers a solution that is easier to secure and manage than fully proprietary solutions, but without compromising on the fundamental capabilities required in a historian.

7. Myth #6: There is nothing special about industrial applications

True historians provide facilities for dealing with the demanding, real-world realities of industrial applications that are outside the realm of pure relational databases. How do you intend to make use of the data? Do you need to convert rates into quantities for reporting? If so, that is quite complex with a SQL query. Is your instrumentation and data collection 100% reliable, or do you sometimes have to "make do" with data that includes instrument errors? While general-purpose databases can certainly store data, they aren't designed to incorporate notions of "data quality" into calculations and aren't able to simply perform routine time-series calculations such an integral calculation that are commonly needed.

8. Myth #7: Relational database applications aren't historians

A "historian" addresses several related functions: continuously collecting real-time data, storing noteworthy subsets of that data, and providing a means of extracting meaningful information from that data. Whereas "historian" describes an application, "relational database" names a technology. Though there are certainly some significant challenges in using a relational database technology for time-series data, just because an application uses one doesn't mean it isn't a historian. It can still be a historian, just a fairly basic one. Rather than focusing on the underlying technology choices (relational databases or proprietary files), focus on the functionality required—that involves much more than simply storing data.

9. Myth #8: All data is equal in importance and validity

To a relational database, a stored value is precisely that, a value, and is always assumed to be valid—if it isn't, it is up to someone to correct it. In collecting millions of samples from thousands of data points from around a process, it is inevitable that some information is incorrect. There may be issues with measurement equipment where values are out of range, communications was lost, or the data was simply erroneous.

In a plant historian, a stored data point not only has an associated value and time stamp, it also has an indication of the data quality. Storing a data point from an instrument, outside of the instrument's normal operating range, for example, will cause a specific series of quality indicators to be stored with the value. These indicators aren't simply separate columns in the database, but an inherent property of the sample. They can be retrieved, included in calculations and used to alert operations or engineering personnel to a potential anomaly.

When summarizing the values (for example, calculating an average temperature over the last hour), a historian must be able to reflect this data quality in calculation results, optionally filter out suspect data, and be able to extrapolate when data is missing or deemed invalid. If these real-world aberrations aren't handled correctly, resulting reports, business system integration, and decision making will be incorrectly skewed. Relational databases alone don't provide these capabilities.



Invensys Operations Management • 5601 Granite Parkway III, #1000, Plano, TX 75024 • Tel: (469) 365-6400 • Fax: (469) 365-6401 • iom.invensys.com

Invensys, the Invensys logo, ArchestrA, Avantis, Eurotherm, Foxboro, IMServ, InFusion, SimSci-Esscor, Skelta, Triconex, and Wonderware are trademarks of Invensys plc, its subsidiaries or affiliates. All other brands and product names may be the trademarks or service marks of their representative owners.

© 2011 Invensys Systems, Inc. All rights reserved. No part of the material protected by this copyright may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording, broadcasting, or by any information storage and retrieval system, without permission in writing from Invensys Systems, Inc.