Not only can SQL databases work effectively with SCADA, but utilizing SQL unlocks the full potential of what SCADA software can do.
How SQL is Redefining SCADA

If you get a group of control systems professionals in the same room, you are bound to start a heated conversation if you ask, “What do you think of blending SQL databases with supervisory control and data acquisition (SCADA) software?” Some may be indifferent to the subject, while others will be adamantly opposed to the use of SQL and SCADA, and a vocal few may even gush about how SQL is changing the way they use SCADA.

Whatever your knowledge level and feelings about SQL databases, one thing is clear: A general misunderstanding and mistrust of SQL databases exists in the controls system community. There are some valid reasons many feel SCADA and SQL don’t go together, but there are enormous benefits for those who overcome the fear of using SQL with SCADA.

SQL Database Basics

Before we get into the benefits of using SQL we need to cover some basics of SQL relational databases. Specifically, what relational databases and SQL are, and how they relate to each other.

What is a Relational Database?

A database is a central clearinghouse for information. It digitally stores any variety of information you can imagine, from secure financial records, or sensitive controls data, to information on who has late library books. A database is not defined by what kind of information it stores, but rather the structure in which it stores data.

Every true database requires a database management system (DBMS), and a relational DBMS is the most widely used. A relational database is one that stores information in the basic structure of tables made up of rows, columns, and cells. Think of it like a big spreadsheet, just way more flexible and powerful.

Since relational databases group data into tables, each individual data point is related to the data around it. Structuring information in this way makes it easier to organize and retrieve data rapidly, especially when large amounts of data are involved. Relational databases are very flexible because they are designed to handle both small and very large amounts of data effectively. As a result, relational databases are extremely popular, so much so that they pretty much define what the word “database” means. When you think of a database, you are more or less thinking of a relational database.

The real power of a relational database comes from its ability to search out data across multiple tables and relate data sets together by virtue of a shared field type. For example, if you had three different tables that contained contact information grouped together by social security numbers (SSN), then you could search all tables for the information associated with each SSN. A relational database would return all data associated with the SSN you searched, regardless of what tables or databases the information was stored in.
What is SQL?

SQL is an acronym that stands for Structured Query Language. SQL is a programming language designed for managing data in relational database management systems. Since its initial development, SQL has continued to grow in popularity and is the most widely used database language in the world.

SQL's popularity is due in large part to its simplicity. SQL only has a few basic operations that allow it to manipulate and select data. Through the combination of simple commands, SQL can perform a myriad of useful functions with data – making it an extremely powerful and flexible database language.

The most common operation in SQL is a “query”. A query is used to retrieve data based on a specific set of criteria. Queries can select any amount of data from the database in just about any combination you can imagine.

The term “SQL database” can be a little misleading. SQL is not a kind or brand of database. SQL has been adopted as the standardized programing language for relational databases, and as a result relational databases and SQL have become synonymous with each other. So the term “SQL database” means a relational database that understands SQL, which makes up a majority of the databases used worldwide.

The Rift Between SQL and SCADA Software

In the 1990s and early 2000s, SQL databases were firmly established as a new standard in the information technology industry. With the Dot-Com Age in full swing, SQL databases were being used in greater and greater frequency because of their ability to rapidly handle massive amounts of information. Their popularity has had a transformative effect on numerous industries and has helped to shape the fortunes of some of the biggest tech companies in the world, such as Oracle, IBM and Microsoft.

Although SQL databases are usually behind the scenes, they are there nonetheless. Most enterprise systems such as ERPs (enterprise resource planning), and CRMs (customer relationship management) have been driven by SQL databases for years. In today’s modern business world there is scarcely a day that goes by that the average person doesn’t interact with a database in some way.

Yet the industrial automation software industry has been noticeably absent from this trend – but why? Why has a well-developed and supported information technology so widely embraced by so many industries been seemingly ignored by those who develop SCADA (supervisory control and data acquisition) software?
SQL: The Next Big Thing in SCADA

Why SCADA Software Passed on SQL
It's difficult to say exactly why the rift between SCADA and SQL developed, but there are some clues to why their paths never met. Around the time that SQL databases where maturing, SCADA software was also going through a transitional period. One of the challenges it faced was continuing to provide real-time data as plants grew bigger, processes got more complex, and data became more and more plentiful.

Maybe SCADA software developers determined that, at the time, SQL databases couldn't handle the load fast enough. Maybe SCADA companies didn't want to develop software that involved the use of technology that was unfamiliar to their current customers. While all of these reasons likely contributed to SCADA software developers ignoring SQL, the most probable explanation is simple economics: Proprietary technologies make money while open technologies generally do not.

While it's hard to pin down the exact reason why SCADA software missed the boat in adopting SQL, it is easy to see that the companies that developed SCADA software rejected SQL databases and moved in another direction.

The Wall Separating IT and Controls
The industrial automation industry deals with huge amounts of data. Information from the plant floor is captured from PLCs (programmable logic controller) and passed to the SCADA system in the form of real-time data. Time-series data is produced as data points, measured at successive time intervals; they typically contain a value and a time. Even simple processes can potentially have hundreds, if not thousands, of data points to follow – such as tank temperatures, scale weights, and pressure readings.

Traditional SCADA applications have a lot of data to deal with and are primarily designed to display the data in real time. However, data that is not tracked cannot be analyzed, so SCADA software had to find some method of storing data. In the absence of using SQL databases, traditional SCADA resorted to a couple of different methods of tracking time-series data: PLC storage and process historians.

PLC Data Storing
One means of tracking time-series data is to store it in the PLC. This method has several problems, the most important being that PLCs simply aren't designed to store data. The simplicity and limited storage space of a PLC make it a very poor location to store time-series data. While using the PLC to store data that comes in high-speed bursts is okay, using a PLC as a long-term repository for time-series data is simply not efficient; it's just the wrong tool for the job.

Process Historians
A more robust solution is the use of a process historian. A process historian is specially designed to store time-series data. These historians come in all types and are optimized to handle large amounts of time-series data quickly. However, there are some major drawbacks to using a process historian – the most problematic being their proprietary nature.

Every historian is different, which means they save data in different formats that are often proprietary to the company that made the historian. There is no standardized language for historians, which can make working with them and supporting them difficult. Since historians are not standardized and often are structured in a proprietary format, they don't communicate well – if at all – with other databases. The lack of easy communication effectively walls off historians from the rest of the enterprise system and puts them on their own data island.
SQL: The Next Big Thing in SCADA

Breaking Down the Wall
Nothing kills a relationship quicker than lack of communication, and in a sense this is what has happened between industrial control and information technology (IT) professionals. Each gravitates to their own knowledge base when solving problems. So when SCADA problems arise, both professions see solutions from their own viewpoint; neither one sees each other eye to eye.

They just don’t speak the same language. The controls guys are fluent in PLC programming while IT guys are extremely comfortable with SQL. As a result, neither side fully understands the other, which can really affect teamwork and productivity.

IT is often hesitant to support SCADA systems built on proprietary technologies that don’t play nice with the rest of the enterprise. In response, many controls guys view IT as a roadblock to getting things done. This can lead to all kinds of problems, but what’s there to do about it? How can you break down the wall between controls and IT? You’ve got to get communication flowing again, and that must start with a common language.

Since most enterprise systems are already speaking in terms of SQL databases, it makes sense to get the SCADA system on the same page. If time-series data is stored in a SQL database it can easily be related to other data across an entire enterprise system. This can yield a variety of positive results, not the least of which is getting controls guys and IT talking again. Each side will see the other more and more as their greatest asset, and not a hindrance. This can improve communication and foster innovative problem-solving, benefiting each department and the company as a whole.

Dispelling SQL Myths
Although the majority of the IT world is already sold on the power of SQL databases, the controls community remains largely unconvinced. This is due mainly to some ideas about SQL databases that have stayed prevalent despite the maturity of the technology.

Myth #1: SQL Databases are Too Expensive
The idea that SQL databases are too expensive is likely a notion that was carried over from the early days of SQL. Like any new technology, databases were relatively expensive when they first hit the commercial market. However, times have changed and databases have become much more commonplace. Now, purchasing a SQL database is very affordable and there is some very good open-source DBMS software available for free. SQL databases are also easily scalable, so companies big and small can get started without a huge price tag attached.

Another cost misconception associated with using SQL databases is the expense of hiring and training staff to maintain the databases. But this isn’t necessarily a big expense because most companies already have IT personnel who are familiar with SQL, and since SQL database are so popular, it’s easy to find well-trained professionals with the necessary skills.

You don’t have to be a full-fledged database administrator in order to work with a SQL database.

Myth #2: SQL Databases are Too Complex
There is a common misconception that databases are very big, heavy, and complex. Some think that you have to be an IT specialist just to use a database. While there is a lot to databases, the truth is that they are simpler than most people think. You don’t have to be a full-fledged database administrator in order to work with a SQL database. Many
applications – such as MySQL – are designed to make it easy for beginners to get started using SQL databases.

SQL is a well-developed technology so there can be a lot to learn. This often dissuades many people from getting into it. The thought is that there is too much to learn, so it’s easier to just to stay away from it. While there is a lot of information out there about SQL, there are also numerous resources available to those who want to learn. Finding help with SQL can be as easy as going down to the local bookstore or doing a quick search online. There are abundant resources available to those willing to look.

**Myth #3: SQL Databases and SCADA Don’t Mix**

The roots of this myth go back several years and stem from the divergent development paths for SCADA and SQL technologies. SCADA developers decided against using SQL databases to store controls data and instead focused solely on displaying controls information. The result of this has been that many SCADA software packages perform controls functions, but have unsatisfactory data tracking and analysis tools. This has resulted in an array of expensive and unwieldy control system add-ons that try to do what SQL databases already do extremely well.

The perception that SCADA doesn’t function well with SQL databases is simply a false premise that has been perpetuated by SCADA software developers who are unwilling or unable to change. The truth is that not only can SQL databases work effectively with SCADA, but utilizing SQL unlocks the full potential of what SCADA software can do.

**How SQL is Changing SCADA**

Using a modern SCADA system that fully utilizes SQL databases offers such major advantages over traditional SCADA that it can literally redefine the way people think about SCADA. Some of the biggest effects that SQL can have on SCADA are how it can enrich time-series data, empower the ability to ask vital questions, and save huge amounts of time.

**SQL Enriches Time-Series Data**

One of the key differences between traditional SCADA and modern SCADA systems that utilize SQL databases is the concept of taking PLC data and putting it into a relational database.

Using a modern SCADA system that fully utilizes SQL databases can offer such major advantages over traditional SCADA that it can literally redefine the way people think about SCADA.

Time-series data in traditional SCADA is formatted in nonstandard ways; because of this, it is difficult for other enterprise systems to understand time-series data. SQL-driven SCADA software transforms time-series data into a more palatable format for the rest of your enterprise, making it easily readable by other database-driven systems. SQL databases also share a standardized connection, which makes it easy for controls information to be layered with information from other areas of the enterprise, such as ERP systems.

Combining of control and enterprise information is where the real advantages of SQL-driven SCADA systems come to light because it can put your time-series data into context with the rest of your enterprise. The ability to easily compare and relate time-series data with any other data across the entire business opens up a virtually unlimited number of new possibilities for using controls data to improve efficiency and profitability.

This can make time-series data infinitely more valuable to a company, enriching it from a cumbersome necessity to an invaluable asset.

**SQL Empowers You to Ask Questions of Your Data**

Relational databases were designed from the ground up to serve not just as repositories for data, but also as powerful engines to ask questions of data. The structure of a SQL database and SQL's simple, flexible commands make it extremely easy to ask just about any question that you can imagine.

The ability to ask questions of your data is extremely empowering, and it can be used endlessly to find important answers that can positively affect the profitability of a company.
Take a look at an example to see the benefits of asking questions (and being able to find the answers!). Let’s say a company wanted to see what the acid levels of their tanks are (PLC data) based upon what raw materials vendor they used (ERP data). They also wanted to see how that affected the quality of the product (quality data) and then in turn see how all those factors influenced the sale of the product (ERP data).

The answer to this question could reveal all kinds of interesting and important information. The company could discover which tanks were producing higher acid levels, or that the use of raw materials from a certain vendor was resulting in poorer sales of the product. There are any number of facts that can come to light when data is questioned – facts that can inform important decisions that affect a company’s bottom line. Having controls data easily accessible in a SQL database makes questions like this possible to ask, and easy to answer.

**SQL Results in Huge Time Savings**

In business time is money, and having controls data in a SQL database can save a large amount of time. To illustrate this, let’s take a look at some simple questions and how long it might take to answer them.

Continuing from the previous example, let’s say the company has a recall on some of their product because it was discovered to have high acid levels. Now they could recall their entire production, resulting in huge losses, or they could narrow down the cause of the problem and identify which product shipments were actually affected. To do this they would need to find the acid levels in all their tanks, and cross reference that with the product batches they produced, along with what shipment resulted from those batches and where they went.

If this company did not track controls data at all, it would simply be out of luck. If they did track it but could only keep it on spreadsheets, the company would be forced to manually go through all of their data to try to find the answer they needed. Depending on the size of the data set, this could take hours, days, or even weeks.

If another variable was entered into the equation, the search would have to begin all over again. This would take a lot of time, resulting in a loss of production and potentially costing the company huge profits.

If, however, the company had been properly tracking data in a SQL database that easily connected with the other enterprise systems, they would have a much easier – and faster – answer. They would be able to run the above query in the database and receive the results in milliseconds, giving the company the answer almost instantly.

**The New SCADA**

Simply put, data belongs in a SQL database and controls data is no exception to this. The speed and power of modern information technologies dictate the speed of business, and traditional SCADA software has been left behind. Doing today’s business with yesterday’s technology just doesn’t make sense in the modern fast-paced manufacturing industry.

The power of SQL databases is changing the definition of what a SCADA system can and should be. SCADA systems that take full advantage of SQL databases have flexibility, power and speed that traditional SCADA can’t offer. Companies that embrace the use of SQL with their SCADA systems stand to gain a huge advantage over their competition.