Understanding QlikView’s Associative Architecture

May 12, 2010
Author: Dan English

QlikTech’s product literature makes references to “associative technology” and “analytics the way your mind works.” Are these real technical innovations that bring new business insights or just marketing terms? The goal of this paper is to shed some light on QlikView’s associative architecture and explain how it fundamentally changes the way business analysis is done. In doing so, this paper explores the technical essence of what associative architecture means, why it’s the best technology to empower business decisions, and why no other tool comes close to offering the benefits of QlikView’s associative architecture.

**Associative Analysis – The Vision**

Way back in 1993, the founders of QlikTech had a vision. They believed that anyone making complex business decisions should have full access to their decision support data, and not as isolated and discrete queries without context between one query and the next. Instead, decision makers should be able to interact with their data as an organic whole with all of the associations that make up the living web of data intact. *In fact, every data point should have a valid association with every other data point at all times throughout the entire analysis process.* Only by having this holistic and multifaceted view of the data and by retaining those associations would decision makers be armed with the information needed to make the most informed business decision possible.

The desire to associate data does not originate with QlikView. The concept of association has long been seen as the holy grail of business analytics. The real genius of QlikTech’s founders can be seen first in the extent of association, stating that “every data point anywhere in the entire dataset to be analyzed, regardless of how many data fields there are or how complex the underlying schema may be, should always be associated with all other data points at all times.” Second, starting with that premise, they invented a revolutionary new technical architecture to make this vision a reality.

**The QlikView Associative Data Engine**
In order to support their vision of associative analysis, the founders needed to build an associative data engine. In order to ensure that every data point maintained its proper association with every other data point, the founders of QlikTech were forced to invent the world’s first in-memory analytic architecture. Building the associative engine entirely in memory means that every association with every data point is processed dynamically with every click.

**In-Memory Architecture vs. Associative Architecture**

QlikTech’s founders invented the world’s first in-memory analytic architecture to achieve their vision of associative analysis. In-memory architecture was not an end in itself; it was just the means for achieving associative analysis.

It is interesting to note that primarily because of QlikTech’s success in the market, today many vendors are providing in-memory analytics. Because QlikView pioneered the use of in-memory architecture to support analytics a decade earlier than any other vendor, these vendors assume that in-memory architecture represents QlikTech’s competitive advantage. Based on this assumption, they attempt to make inroads against QlikTech’s competitive position in the marketplace by rushing their own “in-memory” analytic tools to market.

By and large, these competitors misunderstand the market appeal of QlikView. Simply taking an existing structured query language (SQL) or cube-based query technology and caching it in RAM may produce faster results and certainly does make their offering an “in-memory analytic platform,” but these in-memory offerings are still query-based and provide none of the unique benefits of QlikView’s associative architecture.

**Query-Based Architectures—The Status Quo from 1960 to the Present**

For the last 50 years, query-based architectures represented the status quo for decision support. These query-based architectures extract a very small subset of data from the main dataset. They aggregate the extracted data and return it in the form of a query result set. This result set is completely divorced from any other data not contained in the individual query. The act of extracting the subset of data from the main dataset breaks all associations.

For most of these years, SQL queries have been the overriding paradigm for providing decision support. SQL has a venerable track record, and for many years, SQL queries were the best tool for the job. However, SQL’s ubiquity creates a blind spot to the shortcomings of using queries—whether SQL, multidimensional query expressions (MDX), or otherwise—as the fundamental component of a decision support engine.
Online analytical processing (OLAP) uses aggregated data for decision support. Although today there are many variations of OLAP, they are all fundamentally query-based.

The oldest form of OLAP decision support is ROLAP (Relational Online Analytical Processing) and it is still quite prevalent today. ROLAP simply uses SQL or another query technology to extract and calculate aggregates in real time as the user needs them. Once thought of as slow and unresponsive, today ROLAP is enjoying something of a renaissance with the advent of highly scalable, extremely fast decision support database architectures. Despite this, ROLAP is still a query-based architecture. The data in any given query has no intrinsic association with data in any other query.

The next generation of technology for decision support came in the form of Multidimensional Online Analytical Processing or MOLAP, also known as cube-based OLAP. With MOLAP, the data is preaggregated for every permutation of data points along several preselected dimensions. This provides near instantaneous access to aggregates as long as the question lies within the predefined dimensionality. MOLAP is also query-based. The only real difference between ROLAP and MOLAP is when the query results are aggregated (in advance for MOLAP; as needed for ROLAP). Aggregates are returned in discrete result sets with no associations between query result sets.

ROLAP and MOLAP both have merits and drawbacks. In general, ROLAP is thought to be more flexible (no predefined dimensionality) but more computationally expensive (and therefore slower). In addition, ROLAP is generally considered a better choice when high cardinality exists in the dimensions to be analyzed. In contrast, MOLAP is generally thought to be faster (near instantaneous access to aggregates because they are already precalculated) but less flexible (predetermined dimensionality). MOLAP is generally better when there is low cardinality in the dimensions to be analyzed. This is because high dimensionality in multiple dimensions will lead to an exponential explosion of data.

The relative strengths and weaknesses of ROLAP and MOLAP led to the creation of a third technology, HOLAP (Hybrid Online Analytical Processing). HOLAP is quite simply any architecture that leverages both ROLAP and MOLAP in an attempt to offset the relative weaknesses of each. As you might expect, since HOLAP is the product of the marriage of two query-based technologies, it is also a fundamentally a query-based technology.

**Why Customers Are Passionate about QlikView**

Many customers choose QlikView because it is both fast and flexible. QlikView offers the same flexibility found in ROLAP (no predefined dimensionality) along with the speed of MOLAP (near instantaneous access to aggregates). This is reason enough for designating QlikView the tool of choice for decision support, but nonetheless it shortchanges the value of the QlikView platform by ignoring the product’s biggest competitive differentiator.

In looking at feedback from the customer base, what immediately
QlikView jumps out is passion for QlikView. What fuels this passion is much more than speed and flexibility.

Users are passionate about QlikView because it is associative. When users look at two different data points, in QlikView they know precisely how they relate to each other. If they want to narrow down their data to a single product (or country or year), they can see how every other data point in their analytic dataset reacts to that selection (instead of just affecting the individual query results they are looking at).

**QlikView’s Associative Architecture vs. Query-Based Architectures**

Each individual query represents a single discrete chunk of information. This information is extracted from the underlying database or cube and passed on to the calling tool or user as a discrete set of data. The process of extracting data divorces it from its context. If you want to know more about how a piece of data contained in your query relates to another piece of data outside your query, there is no choice but to formulate a new query that incorporates both data points and resubmit it.

QlikView works quite differently. Every data point in every field is associated with every other data point in the table. That means that through the key fields that join the tables together, every data point in the analytic dataset is associated with every other data point in the dataset. In QlikView, datasets can be hundreds of tables with thousands of fields. When you click on a data point in a field in a QlikView document, no queries are fired. Instead, all the other fields instantaneously filter themselves based on that selection and any and all aggregates are recalculated in real time, regardless of the source fields for those aggregates in the schema.

**An Analogy – The Internal Combustion Engine**

As an analogy, let’s say the goal is to understand how an internal combustion engine works. With the query-based paradigm, we would look at individual parts in isolation. We would be presented with one part at a time. Something like this...

![Query #1](image1)

and...

![Query #2](image2)
Using QlikView’s associative technology, we have access to a complete working engine with each part in its correct relationship (or association) with all of the other moving parts.

We can tweak the throttle (execute a selection) and see how that affects the fuel intake, the carburetor, and the exhaust. We can watch the pistons pump and turn the crankshaft. We can decompose the engine at our leisure and look at each part in context with the parts next to it. Quite simply, this is the power of QlikView’s associative architecture.
Associations in Query-Based Tools

More recently, cube-based OLAP has become a popular alternative to SQL for aggregation. But since cube-based OLAP is simply another example of a query-based architecture, the same limitations apply.

Many query-based tools make no attempt to associate data returned in different query result sets (with these tools, if you want associations at all you must cram every field you want associated into a single query). Those tools that attempt to recreate associations between query result sets face a near impossible task.

Of course, in 1993, there was no computing architecture that could support QlikTech’s founders’ vision of associative analysis.

Associations in the Real World

QlikTech did not invent the idea that one data point can be associated with another. Likewise, there is nothing magical about creating an association. In fact, most BI tools on the market strive to present data in context and with the correct business relationships (or associations). Why then does QlikView have a sustainable competitive advantage over all of these tools? Because QlikView manages these associations at the engine level and not at the application level. For example, a conceptual view of a typical query-based BI tool might look like this:

The stars represent individual SQL (or MDX) queries. The least sophisticated tools don’t even attempt to provide context by tracking associations between one query and another. More sophisticated tools attempt to provide context between queries, but this is a huge task because each query may be
different and most are not even defined until the end user sits down to use the tool. No one knows what query the end user will formulate next. How then to provide associations between one query and the next? In this case, the job of creating, assigning, and maintaining associations among individual queries belongs to the application layer. This leads to long deployments where the application layer is customized to manage the specific associations required to answer a particular business question. When you need to answer a slightly different business question, the application layer must be altered to answer the next business question. This process of altering the application layer to provide new associations is extremely time-consuming and expensive.

In contrast, a conceptual view of QlikView looks like this:

The scrolls indicate individual tables stored in QlikView’s in-memory associative engine and the lines indicate all the associations stored generically against the entire dataset, ready to answer any business question as it comes up without requiring customization. QlikView does not use queries. All data from all tables is always available in context and ready to answer the next business question whatever that may be.
Associative architecture creates a living fabric of data where all the fibers are connected. As you find out the answer to one question, you think of another question. You blaze a trail through the data, bringing some information into focus, examining it in a new way, and taking the analysis in a fresh direction. Query-based technologies can never offer this type of dynamic and holistic data exploration.

The Benefits of QlikView’s Associative Architecture

Beyond the ability to see things in their data that they’ve never seen before, customers give two additional reasons for their passion for QlikView: it’s quick to deploy as well as easy to use. Some competitors attempt to characterize QlikView as quick to deploy and easy to use because it is a toy and the technology is trivial and not robust. They are either intentionally or unintentionally missing the point. QlikView is quick to deploy and easy to use precisely because of its sophisticated next generation associative architecture. The fact that QlikView’s associative architecture does not need to be reconfigured to answer new business questions makes QlikView quick to deploy. Because QlikView’s associative architecture preserves all of the associations between all fields being analyzed, end-users find QlikView easy to use and intuitive. Of course, being quick to deploy, intuitive, and easy to use leads to fast ROI and lower TCO, but the ultimate value in QlikView is enabling decision makers to interact with data holistically, leading to better business decisions that ultimately help the bottom line. See the QlikView web site to review the many case studies that describe how QlikView enables customers to benefit from better business decisions.

The Competitive Landscape

To date the market buzz has been around “in-memory analytics” with little or no attention given to QlikView’s superior associative architecture. QlikTech has benefited because competitors have disrupted production schedules to rush in-memory analytic products to market. Because the vast majority of these new in-memory products are built on a foundation of query-based tools, they still cannot compete with the benefits provided by QlikView’s associative architecture. We do however recognize that this will not last forever. We must anticipate that as the market is saturated with in-memory tools, our more savvy competitors will catch on (or are already catching on) to the benefits of an associative architecture. When looking to the future where we may have to compete against other associative tools, we see two primary sustainable competitive advantages.

When competing against the large stack vendors, our advantage lies in their ongoing reliance on the sales and distribution of their older query-based tools. If and when they ultimately do launch successful tools which take advantage of an associative architecture, they will cannibalize their own sales. To make matters worse, their new associative tool will be easier to maintain and require less consulting so they will also lose a tremendous amount of services revenue. We believe these factors will make it very difficult for these stack vendors to mount a serious challenge to QlikTech’s market position through the use of associatively architected business intelligence tools.
Over the foreseeable future we expect our more serious competition to come from an as yet unidentified small vendor that adds some innovation on top of our associative architecture. Our sustainable competitive advantage in this case is the 15 years experience we have in refining and optimizing our associative engine and all the other components of QlikView as well as our growing experience in servicing enterprise markets.