BLOG@CACM

The *Communications* Web site, http://cacm.acm.org, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we publish selected posts or excerpts.

twitter
Follow us on Twitter at http://twitter.com/blogCACM

DOI:10.1145/2366316.2366319 http://cacm.acm.org/blogs/blog-cacm

New Opportunities for New SQL

Michael Stonebraker expects a substantial increase in the number of New SQL engines using a variety of architectures in the near future.

Michael Stonebraker
"New SQL: An Alternative to NoSQL and Old SQL for New OLTP Apps"
http://cacm.acm.org/blogs/blog-cacm/109710
June 16, 2011

Historically, Online Transaction Processing (OLTP) was performed by customers submitting traditional transactions (order something, withdraw money, cash a check, etc.) to a relational DBMS. Large enterprises might have dozens to hundreds of these systems. Invariably, enterprises wanted to consolidate the information in these OLTP systems for business analysis, cross selling, or some other purpose. Hence, Extract-Transform-and-Load (ETL) products were used to convert OLTP data to a common format and load it into a data warehouse. Data warehouse activity rarely shared machine resources with OLTP because of lock contention in the DBMS and because business intelligence (BI) queries were so resource-heavy that they got in the way of timely responses to transactions.

This combination of a collection of OLTP systems, connected to ETL, and connected to one or more data warehouses is the gold standard in enterprise computing. I will term it "Old OLTP." By and large, this activity was supported by the traditional RDBMS vendors. In the past I have affectionately called them "the elephants"; in this posting I refer to them as "Old SQL."

As noted by most pundits, "the Web changes everything," and I have noticed a very different collection of OLTP requirements that are emerging for Web properties, which I will term "New OLTP." These sites seem to be driven by two customer requirements:

The need for far more OLTP throughput. Consider new Web-based applications such as multiplayer games, social networking sites, and online gambling networks. The aggregate number of interactions per second is skyrocketing for the successful Web properties in this category. In addition, the explosive growth of smartphones has created a market for applications that use the phone as a geographic sensor and provide location-based services. Again, successful applications are seeing explosive growth in transaction requirements. Hence, the Web and smartphones are driving the volume of interactions with a DBMS through the roof, and New OLTP developers need vastly better DBMS performance and enhanced scalability.

The need for real-time analytics. Intermixed with a tidal wave of updates is the need for a query capability. For example, a Web property wants to know the number of current users playing its game, or a smartphone user wants to know "What is around me?" These are not the typical BI requests to consolidated data, but rather real-time inquiries to current data. Hence, New OLTP requires a real-time query capability.

In my opinion, these two characteristics are shared by quite a number of enterprise non-Web applications. For example, electronic trading firms often trade securities in several locations around the world. The enterprise wants to keep track of the global position for each security. To do so, all trading actions must be recorded, creating a fire hose of updates. Furthermore, there are occasional real-time queries. Some of these are triggered by risk exposure—i.e., alert the CEO if the aggregate risk for or against a particular security exceeds a certain monetary threshold. Others come from humans, e.g., "What is the current position of the firm with respect to security X?"

Hence, we expect New OLTP to be a substantial application area, driven by Web applications as the early adopters. These applications will be followed by more traditional enterprise systems. Let's look at the deployment options.

1. Traditional OLTP. This architec-

ture is not ideal for New OLTP for two reasons. First, the OLTP workload experienced by New OLTP may exceed the capabilities of Old SQL solutions. In addition, data warehouses are typically stale by tens of minutes to hours. Hence, this technology is incapable of providing real-time analytics.

2. NoSQL. There have been a variety of startups in the past few years that call themselves NoSQL vendors. Most claim extreme scalability and high performance, achieved through relaxing or eliminating transaction support and moving back to a low-level DBMS interface, thereby eliminating SQL.

In my opinion, these vendors have a couple of issues when presented with New OLTP. First, most New OLTP applications want real ACID. Replacing real ACID with either no ACID or "ACID lite" just pushes consistency problems into the applications where they are far harder to solve. Second, the absence of SQL makes queries a lot of work. In summary, NoSQL will translate into "lots of work for the application"—i.e., this will be the full employment act for programmers for the indefinite future.

3. New SQL. Systems are starting to appear that preserve SQL and offer high performance and scalability, while preserving the traditional ACID notion for transactions. To distinguish these solutions from the traditional vendors, we term this class of systems "New SQL." Such systems should be equally capable of high throughput as the NoSQL solutions, without the need for application-level consistency code. Moreover, they preserve the high-level language query capabilities of SQL. Such systems include Clustrix, NuoDB, SQLFire, MemSQL, and VoltDB. (Disclosure: I am a founder of VoltDB.)

Hence, New SQL should be considered as an alternative to NoSQL or Old SQL for New OLTP applications. If New OLTP is as big a market as I foresee, I expect we will see many more New SQL engines employing a variety of architectures in the near future.

Disclosure: Michael Stonebraker is associated with four startups that are either producers or consumers of database technology.

Readers' Comments

RavenDB, a document database of the NoSQL genre, has great ETL, OLTP support, and is backed by ACID storage. See http://t.co/fVxvmSV (and it has evolved a lot since then).

—Anonymous

I was hoping to know the characteristics of New SQL. I am actually not convinced with your arguments about NoSQL databases as you have presented no arguments. Problems with Old SQL are known, promises of NoSQL are also known. What is New SQL?

—Anonymous

In the context of transaction processing, I would define a New SQL DBMS as one having the following five characteristics: 1) SQL as the primary mechanism for application interaction; 2) ACID support for transactions; 3) A nonlocking concurrency control mechanism so real-time reads will not conflict with writes and thereby cause them to stall; 4) An architecture providing much higher per-node performance than available from the traditional "elephants"; 5) A scale-out, shared-nothing architecture capable of running on a large number of nodes without bottlenecking.

—Michael Stonebraker

About point 3, isn't this achieved in most traditional databases by MVCC?

About point 4, most databases (Clustrix, Akiban, NimbusDB) that are New SQL candidates talk only about better query performance using distributed query or a kind of object storage. I am not sure if they have anything in better for DML performance.

VoltDB is an exception; I am not sure if it is much better than TeraData or Greenplum, which are based on Old RDBMS architecture.

About point 5, yes, this is a new feature. If I understand correctly, it means scaling the performance by adding new nodes without interrupting existing users.

—Anonymous

My previous comment suggested five criteria that defined a New SQL DBMS. I would like to stress three points that I made previously. First, my posting focused on DBMSs for new OLTP applications. Data warehouse vendors, such as TeraData and Greenplum, are focused on a completely different market, and are not designed to perform high-velocity transactions. Hence, they are not considered as New SQL vendors.

Second, most of the Old SQL vendors use standard two-phase locking, although there are exceptions. Hence, there are Old SQL engines that satisfy some of my five criteria.

Third, one of the big problems with Old SQL engines is their mediocre per-node performance. One of my criteria for New SQL vendors is much better per-node performance. The proof of this is via performance on standard benchmarks. Hence, whether any particular vendor satisfies this criteria would have to be determined experimentally. As such, the list of vendors who satisfy the five criteria may well change over time.

—Michael Stonebraker

In case any database has scale-out architecture, why is it most necessary that per-node performance also should be very high? Anyway, if performance is getting better by adding more nodes, it will be achieved.

My main focus is to understand why can't some existing database like PostgreSQL be considered an option similar to New SQL if we enhance it to support scale-out architecture in it. Saying this doesn't mean I have a clear idea about how to achieve it.

It will have benefits to existing customers as well even though the performance is somewhat less than New SQL engines. It can save a lot of effort to change applications to suit to new engines.

—Anonymous

In round numbers, New SQL engines are a factor of 50 or more faster on New OLTP than Old SQL. That means that an Old SQL engine would require 500 nodes to do a task that can be accomplished by a New SQL engine with 10 nodes.

The downside of 500 nodes is increased hardware cost, increased power consumption, increased system administration complexity, increased database administration complexity, high availability complexity (if a node fails once a month, then New SQL fails every third day), while Old SQL fails once an hour, and less flexibility (if load doubles, have to add 500 more nodes, rather than 10).

—Michael Stonebraker

Michael Stonebraker is an adjunct professor at the Massachusetts Institute of Technology.

© 2012 ACM 0001-0782/12/11 $15.00