The Vertica Database

A DBMS Architecture Optimized for Call Detail Record Analysis

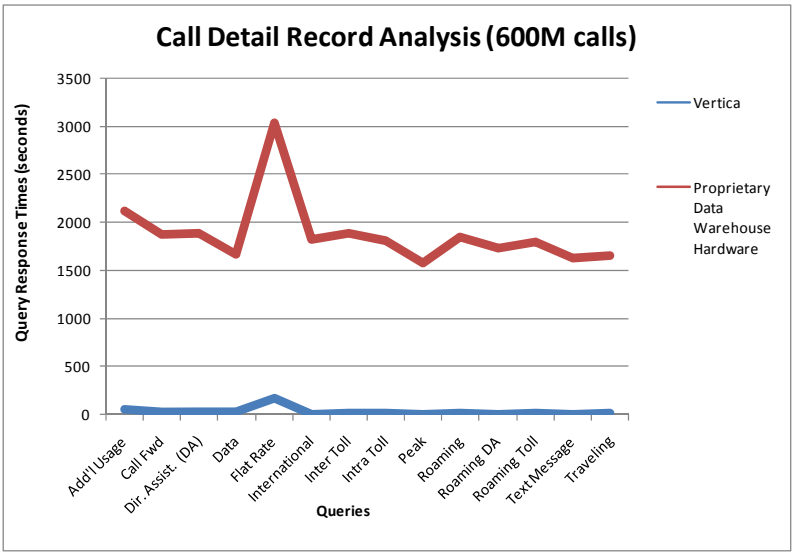
The Scenario: "Where is the money leaking from the system?"

That is one of the questions Revenue Assurance analysts for communications service providers try to answer every day. Primarily focused on detecting fraud, revenue sharing contract violations and incomplete revenue collections, they query and analyze call detail record (CDR) databases that grow by millions of new CDRs every day. The business angles they examine include:

Customer Analysis	Usage profiles Aggregate reports for customers with multiple accounts/services Customer segmentation reports Customer retention and acquisition Selling optional services
Product Management	Profit margin reports Analysis of bundle profits Competitive analysis Reports of promotion ROI
Revenue Reporting	Lost revenue reporting (under-billing, etc.) Verifying billing data with call data from switches Verifying customer orders and billing Sarbanes-Oxley verification/audits
Network Cost Analysis	Support data for negotiating access to competitors' networks Calculating most cost-effective network routes Network utilization versus capacity reports

Led by database research pioneer Dr. Michael Stonebraker, Vertica has developed a break-through SQL database that offers communications companies a competitively advantageous and cost-effective new way to analyze terabytes of CDRs. In one benchmark based on 600 million CDRs, here's how Vertica, running on a cluster of four, $5,000 off-the-shelf servers, compared to a popular proprietary data warehouse server (embedding 4 CPU cores, 48 disks and a derivative of the Postgres database enhanced with MPP support). In the benchmark, Vertica:

- Answered queries **214x faster** on average
- **Stores 1.5 years of CDR data** in the same space the other system took to store 90 days' worth
- Ran on hardware **costing 50% less** than the proprietary data warehouse hardware



This white paper describes the innovative architecture of the Vertica Database and how it is able to provide such remarkable performance compared to other data management solutions.

THE TIME FOR DATABASE INNOVATION IS NOW

The number of subscribers to mobile, fixed-line and cable communications services is growing by millions of people every year, and the volume of CDR data that communications companies must store and analyze is exploding, reaching into the terabytes of new CDR data per year.

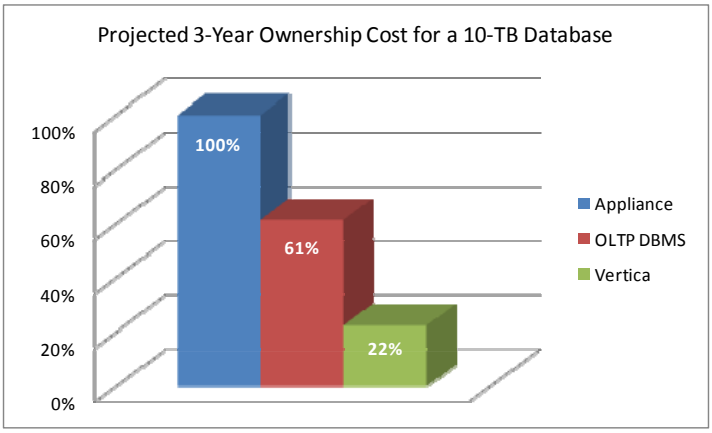
Yet, during the last 30 years, there has been little database management system (DBMS) innovation to keep pace. Performing ad hoc queries on such large data volumes does not come naturally for existing DBMSs, which use a row-oriented design optimized for write-intensive transaction processing workloads rather than for read-intensive analytical workloads. Desperate for better performance, row-oriented DBMS customers spend millions of dollars annually on stop-gap measures such as adding DBA resources, creating and maintaining OLAP cubes or replacing their DBMS with expensive, proprietary data warehouse hardware.

THE VERTICA DATABASE ADVANTAGE

Vertica believes it's time for significant innovation in the database industry. Led by database research pioneer Michael Stonebraker, Vertica has invented a brand-new SQL database that provides blindingly fast query performance for databases scaling from hundreds of gigabytes to hundreds of terabytes, and is optimized for environments where end-user requirements change rapidly. Here is what sets Vertica apart:

RADICALLY IMPROVED DATABASE PRICE-PERFORMANCE

CDR analysis performance benchmarks by communications companies show Vertica out-performs traditional RDBMSs and proprietary data warehouse hardware by 40x-215x. Vertica runs on commodity hardware and compresses data to reduce storage requirements by 20x enabling you to deploy large-scale, query-intensive databases for a fraction of the cost of other solutions.



PAINLESS SCALABILITY

Vertica runs on shared-nothing clusters of off-the-shelf Linux hardware. Vertica scales simply by adding additional servers to the cluster (bonus: Vertica is licensed based on the volume of data you store—not how many CPUs you deploy, so configure your Vertica cluster in any way you want to get the performance you need without worrying about per-CPU fees).

DBA LIBERATION

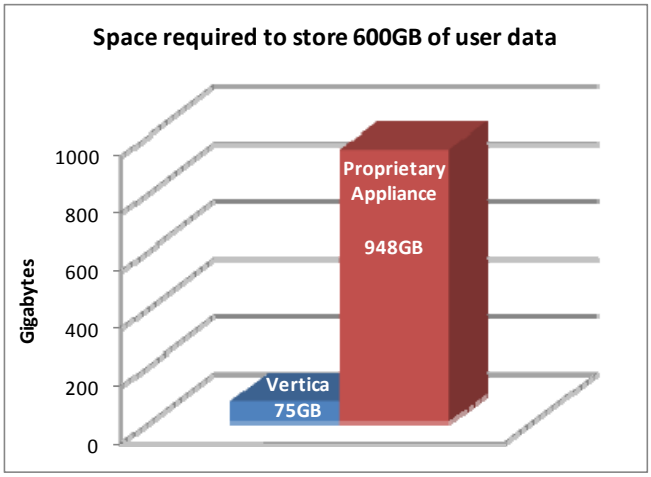
Vertica includes a lot of built-in DBA "know-how" to keep it running efficiently without much administrative overhead. High availability, disaster recovery, schema design and physical optimization are performed automatically, freeing real-world DBAs to focus on higher-value-added activities.

WHAT MAKES VERTICA DATABASE UNIQUE – KEY INNOVATIONS

From a database developer perspective, the Vertica Database looks very standard; it supports SQL, ACID transactions, JDBC, ODBC and works with popular ETL and BI reporting products. Underneath the covers, it's a different story. Vertica is designed to aggressively economize disk I/O and is written natively to support grid computing. Vertica is a 21st-century solution for today's large-scale, read-intensive database applications, featuring ground-breaking architectural features such as:

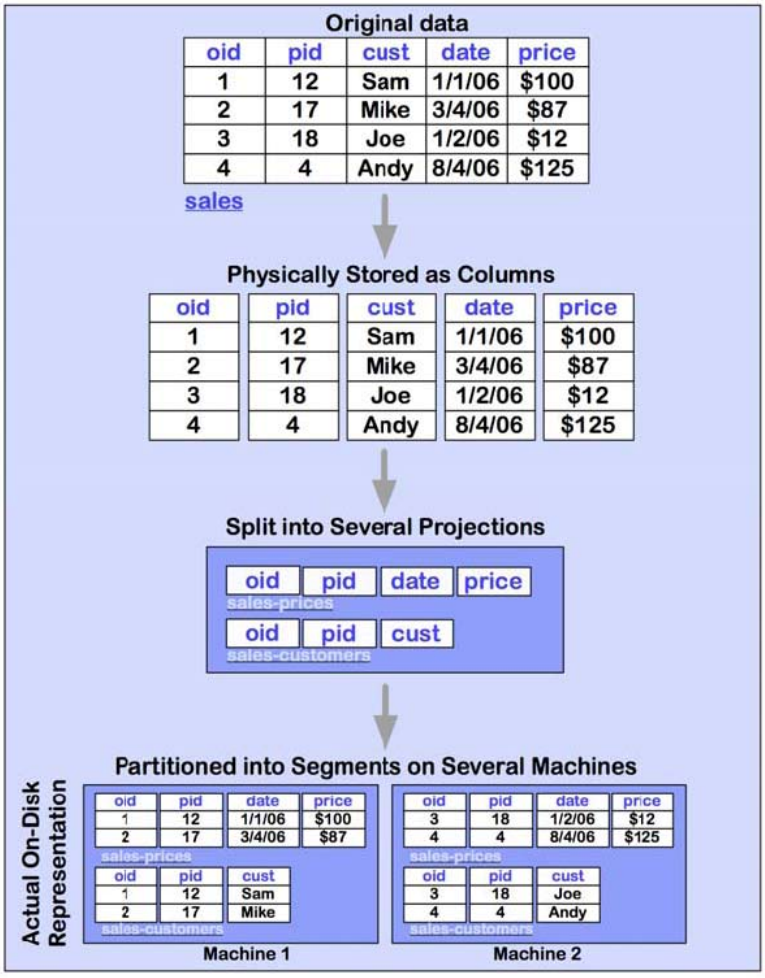
Column Store Architecture. In the Vertica Database, data for each column is independently stored in contiguous blocks on disk. Column values are linked into rows implicitly, based on the relative position of values in each column. Unlike most databases where all columns for each row are stored together, Vertica only needs to retrieve those columns needed for a specific query, rather than all columns in the selected rows (see Figure 4 below). Vertica's vertical partitioning approach produces dramatic I/O savings for the large majority of CDR queries that only retrieve a subset of columns. For example, consider a telecommunications company that has call detail records with 230 columns. Most queries touch fewer than 10 of these, reducing disk I/O time by over 50% compared to row-oriented databases and proprietary data warehouse hardware.

Aggressive Compression. Vertica employs multiple compression algorithms, depending on data type, cardinality, and sort order, to minimize the space occupied by a column. These include run length encoding, delta value encoding and integer packing for integer data, block-based dictionary encoding for character data, and Lempel-Ziv compression. Vertica automatically chooses a good algorithm for compressing data in each column, based on a sample of the data. Compressing data by column often improves compression ratios because the data shares a common data type and value range. Run-length encoding (RLE), for example works best for columns of ordered data, or data with few distinct values compared to the number of rows. This ensures long runs of identical values, which RLE compresses quite well. Vertica demonstrates overall compression ratios ranging from 4x to 10x relative to the ASCII input data. The Vertica query engine processes data in compressed form.



Multiple Projections Stored. Instead of storing data in tables as defined in the logical schema, Vertica physically stores views of the table data, called projections. Each projection contains a subset of the columns of a table in a particular sort order. Rows in a projection consist of the value at the same position in each of the column stores comprising the projection (see Figure 4). Projections can also contain columns from multiple tables, thus materializing joins. To support ad hoc queries, every data element is guaranteed to appear in at least one projection. Vertica automatically selects appropriate projections to optimize query performance for the expected workload. Benefiting from large storage savings due to its extensive use of compression, the Vertica Database can maintain multiple projections with different and often overlapping sets of columns, in several different sort orders, to improve performance for a wide range of expected queries.

Figure 4: Vertica Database Architecture



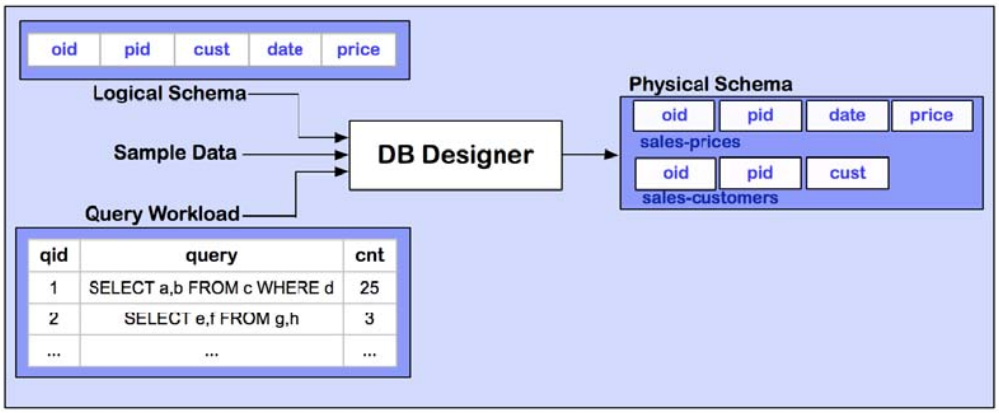
Shared-nothing Parallelism. The Vertica Database is a shared nothing system, designed to run on a collection of homogeneous nodes of a Linux cluster or grid connected by a TCP/IP network. In Vertica parlance, each node is called a site. Nodes contain commodity, multi-core processors with 2 to 4 GB of RAM per core. Storage can be directly attached to each node, or can be SAN-based. In the Vertica Database, parallelism is optimized for star or snowflake data models. Fact tables are range partitioned across the nodes of the cluster (see Figure 4). Dimension tables are typically replicated on each site of the cluster. Very large dimensions are partitioned on the same key as the fact table. This limits the need to share data among sites during query execution. Queries can be initiated on any site. The query planner determines what work needs to be done to answer the query, distributes it to participating sites, collects each site's partial result and prepares the final answer to be sent to the requestor.

K-Safe based Availability. The Vertica Database maintains multiple stored projections, which can also serve as redundant copies of the data for purposes of high availability. By imposing an additional constraint, namely, that the system guarantees that projections are partitioned such that each data element exists on multiple sites, Vertica implements intelligent data mirroring as an integral part of the database. Vertica calls this K-Safety, where *k* is the number of site failures that a given set of Vertica projections will tolerate. Vertica guarantees K-Safety by building *k*+1 replicas of all segmented projections - where each replica has the same columns and partitioning key, though the sort order may differ – and offsetting the distribution of partitions across sites for each replica. K-Safety allows requests for data owned by failed nodes to be satisfied by existing projections on surviving nodes, even though the optimal projection may no longer be available. Once the failed site is restored, Vertica uses the projections on the other sites to automatically re-populate data on the previously failed site.

Automatic Physical Database Design. With the Vertica Database, users need only specify a logical database schema. Today, Vertica targets star or snowflake schemas, so the logical model must satisfy this constraint. Given a logical schema, the Vertica DB Designer automatically generates an appropriate physical database design based on that schema, a sample of representative data and queries, and a space budget for the database (see figure 5). The DB

Designer guarantees that any valid query on the schema can be answered by insuring that all data appears in at least one projection. The DB Designer chooses compression technique to use for each column. It determines which projections to build in order to optimize performance of the sample workload, which columns and joins they will contain, and how each projection should be sorted. It selects the appropriate partitioning key for each projection and guarantees that the projections satisfy the specified K-Safety level. By taking over the details of the physical design, the Vertica Database simplifies database implementation and allows database designers and administrators to focus on the best logical data model to meet their business needs.

Figure 5: Vertica DB Designer Process

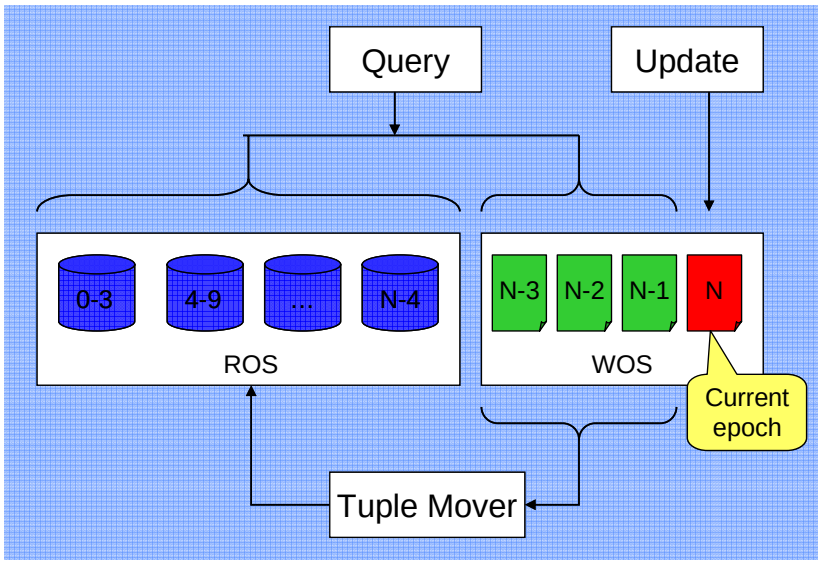


Hybrid Storage Model. The Vertica Database caches all database updates to a queryable main memory cache called the Write-optimized Store (WOS) (see figure 6). The WOS organizes data into projections that are stored as collections of uncompressed, unsorted column arrays, maintained in update order. Periodically, an asynchronous background process on each site, called the Tuple Mover, migrates recent updates to permanent disk storage in the Read-optimized Store (ROS). Data in the ROS is sorted, compressed, and densely packed into variable length disk blocks, optimized for query performance. Appended data is simply added to the end of the appropriate column stores in the ROS. Data inserted into the middle of sorted projection causes the Tuple Mover to rebuild and rewrite disk blocks to maintain the sort order and dense data packing in the ROS. SQL queries can be issued to execute against data in the ROS only, or the ROS+WOS if real-time results are required.

The Vertica Database supports snapshot isolation for query processing, which means that queries and updates do not interfere with one another, and that read-only queries do not require locking. Updates are collected in time-based buckets of fixed duration called epochs. At fixed intervals, Vertica closes the current epoch and begins a new one. New updates are grouped in the new current epoch. Data in older epochs is available for query processing and eventual migration by the Tuple Mover to the ROS (see figure 4).

Vertica's hybrid storage model supports both bulk loads and trickle-feed updates. The Vertica Tuple Mover is regularly working in the background to drain the WOS and merge updates into the ROS to keep it current. While clearly designed for read-mostly application, this approach also works for near-real time data warehouses with high append data volumes, as long as the data latency requirement exceeds the epoch period.

Figure 6: Vertica's Hybrid Storage Model



IN SUMMARY: A DBMS ARCHITECTURE BUILT FOR CDR ANALYSIS

Together, the key features of the Vertica Database create an elegant architecture for large-volume, high-speed CDR analytics applications. Vertica's vertical partitioning by column, extensive use of compression, and hybrid storage model reduce the I/O required to execute queries. CDRs contain many columns per customer, but individual reports cull only a few of these. The types of analytic applications that use a small, arbitrary subset of columns per query are both common, and ideally suited for the vertical partitioning provided by the Vertica Database. At the same time, Vertica's data partitioning, which divides work across multiple nodes in a computer cluster, supports the very large data volumes to which these applications often grow. These analytic applications also typically support users from many disciplines, interrogating the database from multiple individual perspectives. Here, Vertica's ability to keep multiple physical projections of the data is a natural fit for such usage patterns. In summary, the architecture of the Vertica Database was designed specifically to handle the common characteristics of CDR analytics applications.

BENCHMARK THE VERTICA DATABASE YOURSELF

Getting started with the Vertica Database is easy. It supports SQL, and integrates with ETL, analytical and reporting tools, and business intelligence applications via JDBC, ODBC and specific language bindings.

If you would like to learn more about how the Vertica Database can help your company more effectively perform CDR analysis or if you would like to run your own benchmark test, please visit and register at www.vertica.com to find out more.

Vertica Database System Requirements:
A cluster of 1 or more (at least 3 for production use) shared-nothing computers, each featuring:

- Dual-core CPU (at least 2.4GHz), 800MHz (or more) Front-side Bus
- At least 2GB of RAM per CPU core
- 500GB SATA [or any 150GB – 500GB drive, 7.2K RPM, SATA or SCSI]
- Red Hat Enterprise Linux 4, SuSE 10 or Fedora Core 6