*eXtreme*DB®
HPC

*eXtreme*DB "reduced database transaction times by more than an order of magnitude compared to traditional DBMSs."

-- *TradeStation*

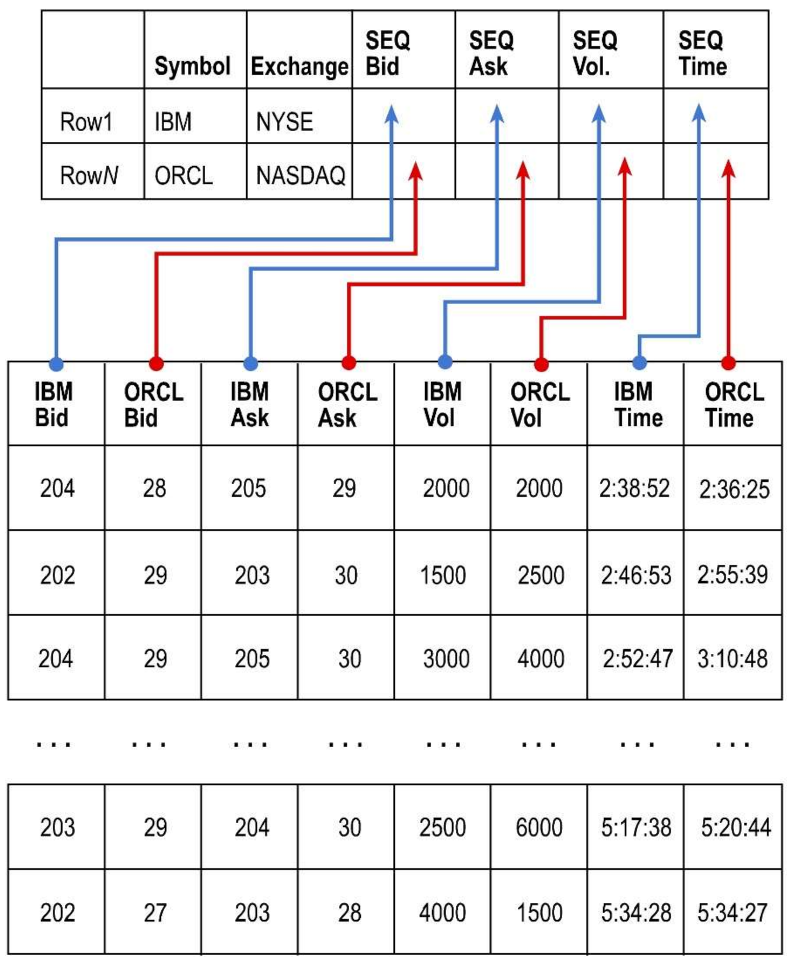
*eXtreme*DB, the low-latency DBMS for high-performance computing.

Profitability in today's markets hinges on acting instantly on growing volumes of Big Data. McObject's *eXtreme*DB HPC database system achieves breakthrough speed by optimizing handling of time series data within the CPU: maximizing the flow of relevant data into CPU cache and keeping it near processing cores even when manipulating big data with complex operations. Better still, *eXtreme*DB HPC achieves this while also delivering reliability, developer flexibility – including access to its most powerful features via declarative (SQL) and scripting in Python and LUA languages – and low total cost-of-ownership.

"In-chip" Analytics

Two features and one powerful programming technique combine to create "In-chip Analytics", which is central to *eXtreme*DB HPC's efficient handling of time series data. The first feature, a "sequence" data type, implements columnar layout for data elements. Sequences are ideal for representing time series such as tick streams, historical trades and other sequential data such as commonly generated by Internet of Things edge devices. The second feature, a rich library of vector-based statistical functions, is provided to accelerate analysis of such time series data.

The programming technique is pipelining, which combines the vector-based statistical functions into assembly lines of processing for time series data, with the output of one function becoming input for the next, all within CPU cache. In contrast, with vector-based languages or data analytics function libraries that do not employ pipelining, the results of each operation or function are transferred outside CPU cache, into temporary tables in main memory, then back into cache for processing by the next function. This back-and-forth imposes significant latency, since CPU cache speed is 2x to 4x faster than data can move between main memory and cache. Pipelining eliminates these transfers, keeping interim results of processing in CPU cache.



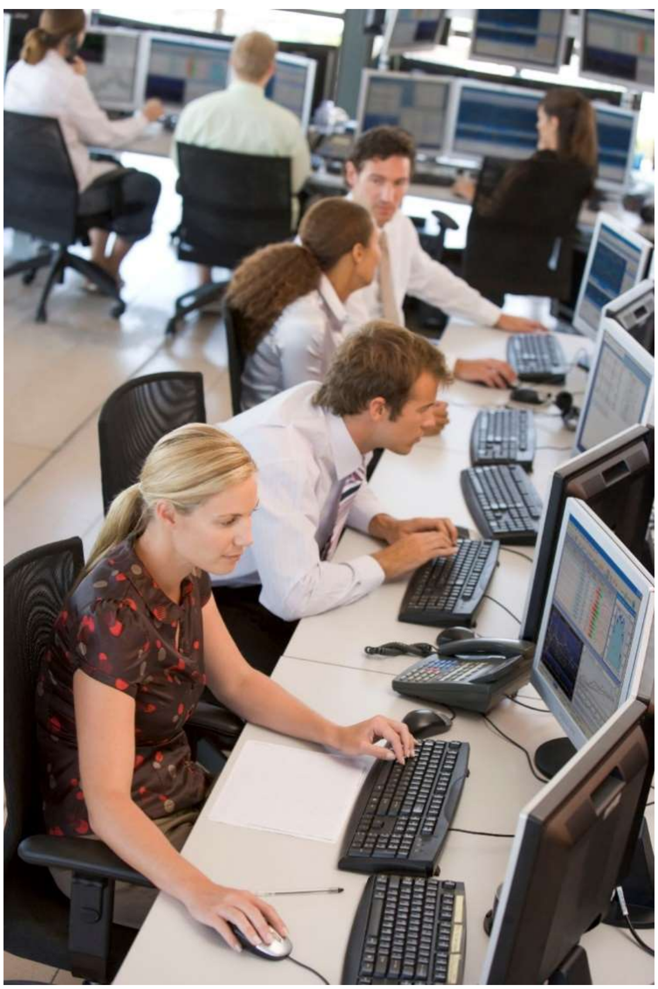
Traditional DBMSs bring rows of data into CPU cache for processing. But time series data – such as IoT, ticks, trades and quotes – are better handled by a column-based layout that maximizes efficiency in fetching needed information.

Pipelining Example

Here's how pipelining works. Let's say the application needs to calculate 5-day and 21-day moving averages for a stock and detect the points where the faster moving average (5-day) crosses over or under the slower one (21-day).

This is accomplished using *eXtreme*DB HPC's vector-based statistical functions as arguments to a SELECT statement:

```
SELECT seq_map(Close,
       seq_cross(seq_sub(
       seq_window_agg_avg(Close, 5),
       seq_window_agg_avg(Close, 21)), 1))
FROM Security
WHERE symbol = 'IBM';
```



1. Two invocations of 'seq_window_agg_avg' execute over the closing price sequence (Close) to produce two temporary sequences containing the 5-day and 21-day moving averages;

2. The two temporary sequences become inputs to the function 'seq_sub' that subtracts 21- from 5-day moving averages and produces a third temporary sequence;

3. This "feeds" a fourth function, 'seq_cross', that generates a final temporary sequence identifying the position within the 'Close' time series that the 5- and 21-day moving averages cross;

4. Finally, the function 'seq_map' maps the crossovers to the original 'Close' sequence, returning closing prices where the moving averages crossed.

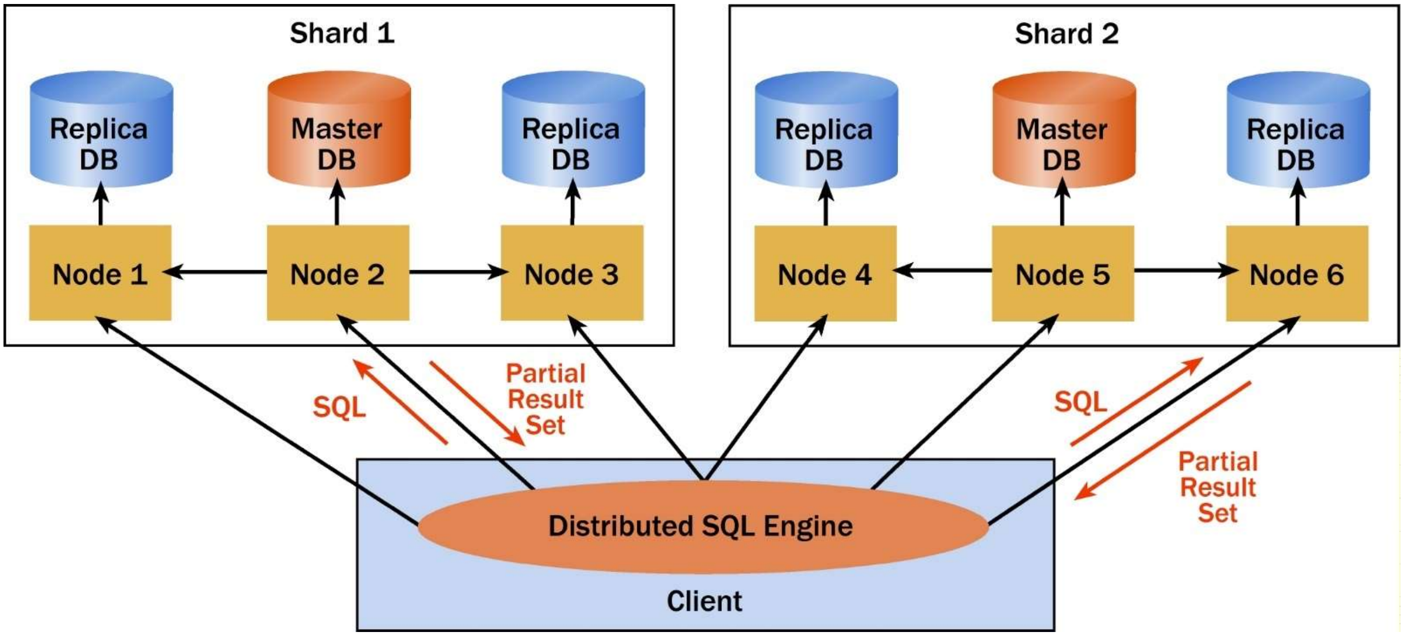
Columnar handling results in faster performance because only the column(s) of interest (closing prices, in our example) are brought into CPU cache at the start of the operation. In contrast, conventional row-wise handling would bring database pages consisting of entire rows with all their columns into CPU cache, flooding the cache with irrelevant data.

Even more significantly, this approach eliminates the need to create, populate and query temporary tables outside CPU cache (i.e. in main memory), as would be required by other database systems and vector-based programming languages to manage interim results of processing (in our example, no temporary table is needed to contain 5-day moving averages, 21-day moving averages, etc.).

Architecture for Minimizing Latency

In-chip Analytics combines with *eXtreme*DB HPC's rich core features to provide the fastest and most complete data management solution for high performance computing needs. Key strengths include:

Flexible storage. *eXtreme*DB HPC is based on a core in-memory database system (IMDS) design that eliminates performance-draining I/O, cache management, data transfer, and other sources of latency that are hardwired into traditional relational database management systems (RDBMSs). Persistent (on-disk) storage is optional and can be implemented selectively.



Sharding, with distributed query processing, leverages the processing power, memory and bandwidth of multiple hardware nodes. Each database shard can have one or more backup (replica) copies to ensure continuous availability.

Elastic scalability. With its support for distributed query processing, *e*X*treme*DB HPC databases can be partitioned ("sharded") with each partition/shard managed by an instance of the DBMS server. Database shards can be distributed on a storage array (which may be a SAN) – with each database server keeping a CPU core busy – or database servers can be distributed across two or more physical servers with their own storage systems, or any combination of storage and servers. Each shard can have one or more backup (replica) copies, which in addition to delivering high availability via failover can also be utilized to further exploit distributed query processing. Distributed query processing across multiple servers, CPUs and/or CPU cores accelerates performance – dramatically, in some cases – via parallel execution of database operations and by harnessing the combined processing power, memory and I/O bandwidth of many nodes rather than just one.

Programming interfaces. *e*X*treme*DB HPC supports development using industry-standard SQL/ODBC/JDBC through Java, C#, and C/C++. It also supports LUA and Python, high-level languages favored in many markets for fast time-to-deployment. Using Python and *e*X*treme*DB HPC's dynamic database definition language (DDL) capability, developers can implement their ideas quickly, and optimize rapidly by testing changes to code, database tables and indexes, while exploiting LUA for stored procedures.

Record-Setting STAC-M3 Benchmarks

How fast is *e*X*treme*DB HPC? In the STAC-M3 – an independent, audited benchmark suite considered the gold standard for assessing time series data management solutions (specifically, tick databases) –*e*X*treme*DB has repeatedly set records for lowest overall latency and highest consistency of results (low jitter). *e*X*treme*DB HPC achieved these results with highly productive Python and SQL as the database programming language, beating other DBMSs even when their STAC-M3 implementations relied on more labor-intensive low-level programming languages.

xPanel Eases Configuration and Optimization

With *e*X*treme*DB HPC's integrated *x*P*anel* dashboard, system architects can easily configure database parameters, high availability, shards, optimize and monitor their applications, and more. *x*P*anel* operates with *e*X*treme*DB through a RESTful API, so developers can offer identical capability under their own product's branding.

Superior Scalability

*eXtreme*DB HPC provides features to maximize throughput and fully leverage multi-core CPUs:

- In-memory databases scale to terabyte-plus sizes while on-disk database size can grow to exabytes. The developer can specify the storage for any record type, which is ideal for handling real-time data (in-memory) and historical data (persistent) within a single database architecture.
- Multi-version concurrency control (MVCC) transaction manager eliminates "pessimistic" locking to accelerate multi-threaded applications running on multi-core hardware.
- Sharding and distributed query processing leverage the power of many drives, servers, and/or CPUs for nearly linear elastic scalability.
- Run-length encoding (RLE compression) of time series data. In McObject's tests, this feature reduced Chicago Board Options Exchange Market Volatility Index (VIX) tick data to one-quarter its pre-compression size, and increased speed in reading the database by 21%.

Application Areas

*eXtreme*DB HPC was designed to excel in high-performance environments such as the Industrial and Consumer Internet of Things, algorithmic trading, ticker plants, order matching, risk and compliance, quantitative analysis, back-testing and others.

Additional Features & Benefits

- *eXtreme*DB is transactional and supports the ACID (Atomic, Consistent, Isolated and Durable) properties to ensure data integrity.
- Open replication: *eXtreme*DB Data Relay technology facilitates full or selective data sharing between applications based on *eXtreme*DB, and external systems such as enterprise RDBMSs.
- Event notifications inform the application when something of interest in the database changes.
- A short DBMS code path shaves micro-seconds from latency by eliminating CPU cycles required to execute an operation.

"Our evaluation determined that eXtremeDB *out performs other in-memory database systems, meeting [our] current needs and, just as importantly, accommodating future growth."*

--Transaction Network Services, Inc.

McObject LLC
33309 1st Way South, SuiteA-208
Federal Way, WA98003

Phone: +1-425-888-8505
Fax: +1-253-878-5481
Web: http://www.mcobject.com
E-mail: info@mcobject.com