Postgres Plus® Advanced Server

An Updated Performance Benchmark

An EnterpriseDB White Paper

For DBAs, Application Developers & Enterprise Architects

June 2013

EDB ENTERPRISEDB

Table of Contents

Disclaimer

The following is intended as an outline of EnterpriseDB's general product direction. It is intended for informational purposes only, and it should not be relied upon in making purchasing decisions. This information may not be incorporated into any contract. It is not a commitment or obligation on the part of EnterpriseDB to release, launch or deliver any updates, modifications, material, code or functional improvements and may change at EnterpriseDB's sole discretion.

Executive Summary

A key requirement of any enterprise-caliber database in the scalable world of the Internet serving large user populations is high performance when creating and updating data as well as reading existing data. Prior to selecting a database for the first time or switching an existing database to a new one, it is always prudent to seek out and investigate a database's performance characteristics to obtain a level of assurance it will meet your needs both now and in the future.

This paper is intended to assist organizations in this regard by evaluating some basic performance metrics of Postgres Plus Advanced Server from EnterpriseDB Corporation.

EnterpriseDB has been an active member in, and sponsor of, the PostgreSQL community since the company was founded in March 2004 and has worked closely with the community to improve the performance of the PostgreSQL database.

As a result of these efforts and the efforts of the PostgreSQL community, native PostgreSQL performance has, and continues to steadily improve with each release. A representative list of improvements in Postgres Plus Advanced Server including improvements from community contributions is presented at the end of this paper.

Postgres Plus Advanced Server is based upon and kept in synch with community PostgreSQL and contains all the features, fixes and improvements that are added with each release. In addition, EnterpriseDB continues to add advanced performance features to Postgres Plus Advanced Server to further enhance the product's performance. As part of this process, EnterpriseDB engineers measure the performance of the Advanced Server product.

The results of the latest engineering study on performance are presented in this paper showing data points that should prove useful to organizations with existing knowledge of the performance metrics they are looking to achieve in their database applications. 20131010

BenchmarkSQL was utilized for the performance testing in this study because of its ability to run across multiple operating system/hardware platforms and multiple database technologies. In addition, BenchmarkSQL can be run with varying test lengths to simulate real-world loads.

This test is built on the open source JTPCC project. Links for both tests can be found in the next section.

An in-depth discussion on performance targeted specifically to your organization's requirements can be scheduled with an EnterpriseDB domain expert by sending an email to sales@enterprisedb.com.

Benchmark Application and Methods

Benchmarking database performance is a difficult endeavor because of the number of tuning characteristics associated with each database. In addition, the types of performance will vary greatly depending on the type of application, the architecture of the database and an almost infinite number of other parameters.

In an effort to create an objective comparison in performance benchmarking, several companies formed a benchmarking association, which produced scenario specifications to test different types of database application use cases.

Completing a formal public benchmark takes significant time and resources and includes a test driver application written from the ground up specifically for the hardware and database software being tested. A cross-database benchmark requires something different - a test driver application that is database vendor and hardware platform neutral.

The benchmarking suite used by EnterpriseDB engineers in this test is modeled on an industry benchmarking association scenario. However, it is not a formal implementation nor should this study be construed to be a formal public report of results.

For this test EnterpriseDB selected the open source database benchmark driver application JTPCC and enhanced it to push prepared SQL statements to the tested database. This enhanced version of JTPCC has been released as an open source project called BenchmarkSQL.

BenchmarkSQL allows tests to be executed against many different databases. It is a Java application, which is operating system and hardware-neutral, and uses database-neutral drivers – in this case JDBC – to communicate with the database. This effectively eliminates outside performance-influencing factors such as proprietary interfaces so that the end-result comparison is more on the core SQL processing and transaction handling capabilities of the tested databases.

The open source BenchmarkSQL project is available at http://sourceforge.net/projects/benchmarksql.

The JTPCC benchmark, upon which BenchmarkSQL is based, is also available on SourceForge at http://sourceforge.net/projects/jtpcc.

The BenchmarkSQL OLTP (Online Transaction Processing) scenario models a wholesale supplier managing orders. The test is designed to impose a transaction load on a database and to then count how many new orders can be placed and completed under this load.

In addition to transaction processing, the benchmarking suite strings operations together into large transactions. A transaction history is maintained during the execution of the test and this history is compared with actual results to ensure that transactional and referential integrity is maintained throughout the term of the test. Non-transactional database engines will fail this test outright.

20131010

Benchmark Configuration

Setting the testing parameters

The test models a set of five transactions that are being driven by a group of simulated operators.

The transactions modeled are:

- New order
- Payment
- Order Status
- Delivery
- Stock Level

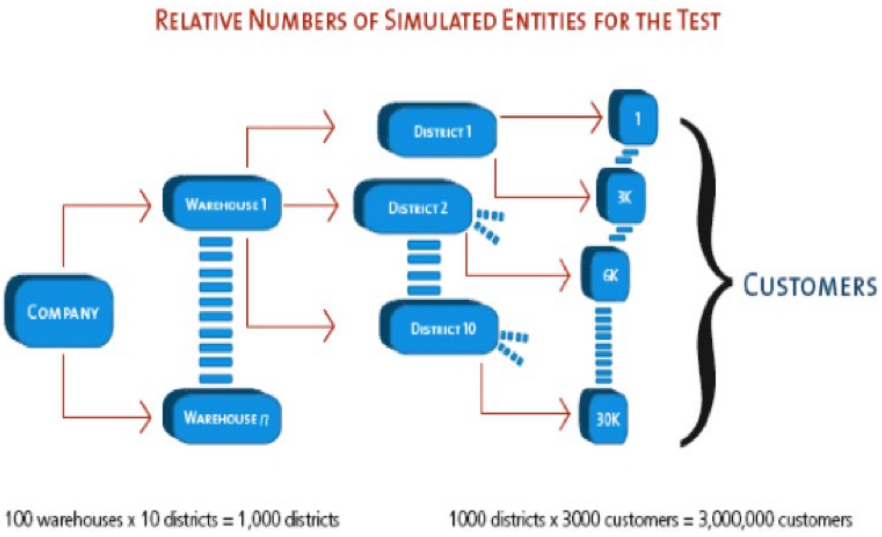
The data set exercised by these operators emulates the structural data requirements of a real business. In this example, a fictitious company has multiple warehouses and each of these warehouses has 10 districts.

Each district has its own sequential system for numbering order transactions. Each district has its own operator who creates new orders, books payments, checks status of existing orders, issues delivery tickets, and checks stock level. Each district has 3000 customers.

Each warehouse has inventory from a list of 100,000 parts so a stock-level of up to 100,000 parts must be maintained for each warehouse. The amount of information grows very quickly with the addition of every warehouse.

For example, in a test run on 100 warehouses there will be 1000 districts, each with an operator (meaning there will be 1000 terminal connections pushing transactions), and with 3000 customers for a total of 3,000,000 customers to track along with the status of any orders generated for each customer.

As a result, a benchmark simulating 10,000 warehouses requires significantly more underlying hardware than a test simulating 100 warehouses.



The test is designed to measure not just the raw throughput of a database but the throughput of New-Order transactions while under a heavy load from the other four transactions listed above. These transactions not only place load but also exercise the ability of the database to effectively and efficiently maintain the integrity of information as it is being accessed and changed from multiple points.

The database is responsible for processing concurrent transactions on the same information and giving results that are accurate for the specific point in time in which they are relevant. Checking status of an order, for example, tests the multi-version concurrency control of a database – the value that is returned from a query on an order's status should reflect the state at the exact time of the request. This is true even if milliseconds after the query was issued an update is performed that would change the state of that order.

A minimum ratio of the other four transaction types is maintained to ensure a healthy load is placed on the database at the same time New-Order transactions are being processed. This ratio is shown by the following table.

Ratio of Transactions for the Test	
Transaction Type	% of mix
New-Order	Up to 30.0
Payment	43.0 Minimum
Order-Status	19.0 Minimum
Delivery	4.0 Minimum
Stock-Level	4.0 Minimum

The measurement interval was 50 minutes. This measurement interval is the time during which transactions per minute are tracked.

The test driver application, BenchmarkSQL, was configured to push as many transactions as possible at each of the tested databases. Further, in this scenario it was configured to skip the wait times of a standard benchmarking association-style test in order to create the most intense load and update contention possible.

BenchmarkSQL Parameters for the Test	
Parameter	Value
Warehouses	1,000
Entry Terminals	100
Districts	10,000
Customers	30,000,000
Test Length	50 minutes

Hardware/Software Configuration and Results

Two machines were used in a client/server configuration where the client hosted the BenchmarkSQL client application and the server ran Postgres Plus Advanced Server 9.2.

The following table shows the hardware/software configuration used for the test and the resulting transactions per minute rate of the test.

EnterpriseDB Advanced Server 9.2		April 2013
System Components	Client	Server
Model	Dell™ PowerEdge™ 2950 Dual Processor Quad Core	Dell™ PowerEdge™ R900 Quad Processor Quad Core
Processor	2 x Intel® Xeon® Quad Core E5345 @ 2.33 GHz	4 x Intel® Xeon® Quad Core E7330 @ 2.40 GHz
Memory	32 GB	64 GB
Storage	115 GB SCSI Internal Storage	570 GB SCSI Storage – PowerVault MD
Operating System	CentOS 6.2	CentOS 5.9
Transactions Per Minute		13,591

The following table shows the approximate transaction rate for each type of transaction.

Approximate Transaction Rate by Transaction Type	
Transaction Type	Transactions Per Minute
New-Order	4,100
Payment	5,800
Order-Status	2,600
Delivery	500
Stock-Level	500

The next table shows a BenchmarkSQL test performed in 2008 using EnterpriseDB Postgres Plus Advanced Server 8.2.

EnterpriseDB Advanced Server 8.2		February 2008
System Components	Client	Server
Model	Dell™ Dimension E521	Dell™ PowerEdge™ 6850
Processor	1 x AMD Athlon™ 64 3200+	4 x Intel® Xeon® Dual Core @ 2.66 GHz
Memory	1 GB	4 GB
Storage	80 GB SATA 7200RPM	576 GB PowerVault MD1000
Operating System	Red Hat Enterprise 4 Linux Workstation	Novell SUSE Linux Enterprise Server 10
Transactions Per Minute		3,634

20131010

The following table shows the approximate transaction rate for each type of transaction.

Approximate Transaction Rate by Transaction Type	
Transaction Type	Transactions Per Minute
New-Order	1,640
Payment	1,560
Order-Status	150
Delivery	150
Stock-Level	150

Note: This is not meant to be a direct comparison since the hardware and operating system environment are completely different, as well as some of the BenchmarkSQL performance test parameters.

Conclusion

Enterprise class databases must prove their mettle in performance benchmark testing.

One measure of EnterpriseDB's seriousness as the open source leader for enterprise class databases is its dedication to performance improvements and testing.

EnterpriseDB has contributed to efforts that have improved the performance of PostgreSQL and continues to augment Postgres Plus Advanced Server with even more performance enhancements.

The mature OLTP database market is poised for disruption and Postgres Plus Advanced Server is a viable alternative.

NOTE: The findings of this paper are based on EnterpriseDB studies and every effort has been made to optimize the performance of the databases tested in this study. However, these results are for illustrative purposes only and different tuning parameters may result in different findings.

For more information regarding Postgres Plus Advanced Server in your environment, please contact us at: https://www.enterprisedb.com/general-inquiry-form or contact the Sales department at: sales-us@enterprisedb.com (US), sales-intl@enterprisedb.com (Intl), or call +1-781-357-3390, 1-877-377-4352 (US Only).

20131010

Advanced Server Performance Improvements

Listed below are some of the most notable performance improvements made to Postgres Plus Advanced Server by version demonstrating EnterpriseDB's and the PostgreSQL community's commitment to performance.

Version 9.3

- Partition fast pruning
- Ability to scale to larger number of partitions
- Materialized Views
- Parallel pg_dump
- Parallel pg_upgrade

Version 9.2

- Index-Only Scans a.k.a. Covering Indexes
- Faster in-memory sorting with up to 20% improvement
- Pgdump options for object handling and faster restores
- INSERT Append Optimizer hint

Version 9.1

- Unlogged tables where no WAL records are kept to offer increased performance
- K-Nearest-Neighbor Indexing for GIST indexes
- Improved Read Lock Management significantly improve performance, especially on read-only workloads and particularly on machines with more CPUs
- Improved Write Lock Management improves write scalability, with improvements more noticeable on large core count machines
- Index Advisor support for Composite Indexes
- ORDERED Optimizer hint support

Version 9.0

- High speed streaming replication
- Faster Vacuum utility for recovering table space
- Deferrable Unique Constraints
- Optimizations for some automatically generated queries, such as those produced by object-relational mappers (ORMs)
- Pgupgrade module eliminates long running dump and restore operations for version upgrades
- PL/pgSQL Code Profiler
- Index Advisor
- EDB*Loader (Oracle like bulk data loader) support for "DIRECT" and "ERROR" in control file
- EDB*Loader support for parallel data load

Version 8.4

- Windowing Functions
- Common Table Expressions and Recursive Queries
- Hash Methods for DISTINCT/UNION/INTERSECT/EXCEPTION queries
- Semi-Joins and Anti-Joins accelerate existing complex reporting queries by executing them more intelligently
- Parallel Restore

Version 8.3

- Heap-Only Tuples (HOT) to reduce overhead of updates
- "Distributed" checkpoints to spread out the I/O load of a checkpoint
- Reduction of on-disk data size through reducing both per-tuple and per-field space overhead
- Bulk execution functions to allow a client application to perform bulk inserts and updates through array binding
- Bulk Collect/Bulk Bind to reduce the SQL processing overhead by efficient use of collections in SQL statements
- Optimizer Hints to allow embedded hints in queries to alter execution plans
- Dynamic Runtime Instrumentation to expose SQL wait events
- Bundling Memcached & Pgmemcache a high-performance

distributed memory object caching system
- Bundling pgBouncer for connection pooling
- Asynchronous Pre-Fetch support for better RAID optimization
- InfiniteCache a horizontal scaling solution characterized by a commodity cache blade architecture

Version 8.2

- Index creation without blocking concurrent insert, update, or delete operations
- Query optimization improvements including support for optimal reordering of outer joins
- Improved sorting performance with lower memory usage
- More efficient locking with better concurrency
- More efficient vacuuming
- New FILLFACTOR support for tables and indexes
- DynaTune to automatically configure the database after hardware memory upgrades
- GridSQL introduced

About EnterpriseDB

EnterpriseDB is the leading worldwide provider of software and services for PostgreSQL deployments in the enterprise. EnterpriseDB's comprehensive ecosystem of performance and security software enhancements for PostgreSQL, Oracle compatibility software and migration guidance, sophisticated management tools for global deployments, enterprise-class support and training help ensure successful enterprise deployments of PostgreSQL. Building upon 25-plus years of open source community PostgreSQL development, EnterpriseDB provides companies a proven, secure and feature-rich alternative to costly traditional databases. EnterpriseDB, based in Bedford, MA, is backed by Charles River Ventures, Volition Capital

(formerly Fidelity Ventures), Valhalla Partners and strategic investors including Red Hat and IBM. For more information, please visit http://www.enterprisedb.com.

20131010