OLTP Performance Benchmark Review

Jignesh Shah

Product Manager, vFabric Postgres

VMware, Inc
About Me

- Currently Product Manager of vFabric Postgres
- Performance Engineer, vFabric Data Director
- Previously with Sun Microsystems (2000-2010)
- Team Member that delivered the first published mainstream benchmark with PostgreSQL
- Blog at: http://jkshah.blogspot.com
Agenda

- Introduction
- pgbench
- Sysbench
- Dbt2
- BenchmarkSQL
- DVDStore
- A New benchmark
Introduction

- **Why do we need benchmarks?**
  - Reference data points
  - Stress Test for “Too Big to Fail” scenarios

- **Uses of Benchmark**
  - Improve Product Quality
    - Understand code path usage
    - Performance Characteristics
  - Baseline metrics (Reference points)
    - Release to release
    - Against other technologies to do same business operation

- **Abuses of Benchmark**
  - Benchmarketing
  - Fixated only on ones that are favorable
pgbench
PostgreSQL
Pgbench

- Based on TPC-B workload (circa 1990)
- Not an OLTP but stress benchmark for database
- Ratio is Branches: 10 Tellers: 100,000 Accounts
- Default TPC-B sort-of
  - Account transactions also impact teller and branch balances
  - Branch table becomes the biggest bottleneck
Pgbench

- Hints
  - PGSSLMODE disable (Unless you want to factor SSL communication overhead. Depending on your distribution)
  - -M prepared (unless you want to measure overhead of parsing)

- Various modes of benchmark

- Default TPC-B sort-of
  - Account transactions also impact teller and branch balances
  - Branch table becomes the biggest bottleneck

- -N Simple Update (with select, insert)
  - Account Update, Select balance. History insert
  - Account table update becomes the biggest bottleneck

- -S read only test
  - AccessShareLock on Accounts table and primary index becomes the bottleneck
  - Fixed in 9.2 (Thanks Robert Haas)
PGBench TPC-B Like Test

PGBench (TPC-B Like)

Transactions Per Second vs. Number of Clients

- PG9.1 TPC-B Like
- 9.2 TPC-B Like
Sysbench
MySQL
**sysbench**

- Originally developed to test systems
- Has an OLTP component which was based on MySQL
- Creates a table sbtest with a primary key.
- Various Modes of OLTP operation
  - Simple Read Only (web site primary key lookup)
  - Complex Read Only
  - Complex Read Write test
Sysbench – OLTP Simple Read

Sysbench Simple Read

Number of Clients

Transactions Per Second

OLTP Simple Read

9:2 OLTP Simple Read
Sysbench – OLTP Complex Read

![Graph showing Sysbench Complex Read](image)

- **Transactions Per Second**
- **Number of Clients**

**OLTP Complex Read**

- 9.2 OLTP Complex Read
Sysbench – OLTP Complex Read/Write

Sysbench OLTP Complex R/W

Transactions Per Second

Number of Clients

OLTP Complex Write

9.2 OLTP Complex Write
**Sysbench – Complex R/W Note**

- In 9.0 it was impossible to run sysbench complex r/w without hitting error - ERROR: duplicate key value violates unique constraint "sbtest_pkey"
- In 9.1 SSI was introduced and occurrence went down drastically
- In 9.2 haven't encountered the occurrence

```sql
Transaction A
BEGIN;
DELETE FROM sbtest WHERE id=500815;
BEGIN;
INSERT INTO sbtest values(500815,0,'','aaaaaaaaaaffffffffffffffffffffffffrrrrrrereeeeyyyyyyyyy');
BEGIN;
DELETE FROM sbtest WHERE id=500815;
END;
(COMMIT)
```

```sql
Transaction B
BEGIN;
DELETE FROM sbtest WHERE id=500815;< ------- hangs/waits
```

(rows . It doesn't roll back the transaction)

```sql
INSERT INTO sbtest values(500815,0,'','aaaaaaaaaaffffffffffffffffffffffffrrrrrrereeeeyyyyyyyyy');
ERROR: duplicate key value violates unique constraint "sbtest_pkey"
END;
(ROLLBACK)
```
dbt2
Dbt2 -

- **Fair Use implementation of TPC-C**
- **Implemented using C stored procedures using**
  **driver->client->database server architecture**
- **Nine Tables**
- **Five Transactions –**
  - New-Order (NOTPM) 45%
  - Payment 43%
  - Delivery 4%
  - Order Status 4%
  - Stock Level 4%
Dbt2 -

- Why is it not TPC-C compliant?
  - Not audited by TPC
  - No Terminal emulator
  - Official kit requires commercial Transaction Manager
  - Doesn’t’ cover ACID tests

- Two versions Available
  - Libpq
  - ODBC

- One potential problem is 3 network roundtrips per transaction which causes “Idle in Transaction” at high load
  - BEGIN, SELECT StoredProcedure() , END pattern of transactions
Dbt2 – Postgres 9.1

Cached Runs (data in bufferpool)
Short runs (limited checkpoint and vacuum impacts)
NOTPM = 45% of all DB Trans
DB Trans rate about 3000 TPS
BenchmarkSQL
BenchmarkSQL-

- Another implementation using TPC-C schema
- Implemented using JDBC
- Nine Tables
- Five Transactions –
  - New-Order (NOTPM) 45%
  - Payment 43%
  - Delivery 4%
  - Order Status 4%
  - Stock Level 4%
- Surprisingly can do better than dbt2 implementation but still has “idle in transactions” which means bottlenecked at network/client level
DVDStore
Implementation of Online DVD Store

Postgres support contributed by VMware

Implemented using various stacks

- JSP/Java/JDBC (supports Postgres)
- Linux/Apache/PHP/MySQL (supports Postgres)
- ASP.NET (not yet implemented for Postgres)
- Stored Procedures (supports Postgres via Npgsql)

Eight Tables

Main Transactions –

- New-Customers 0-10% (configurable)
- Customer Login
- DVD Browse (By category, by actor, by title)
- Purchase Order (Metric – Orders Per Minute)
- Stock ReOrder (via Triggers)
DVDStore

- JSP/Java JDBC Implementation
  - Tomcat may need tuning

- PHP-Postgres Implementation
  - Suffers from one connection per SQL command
  - Needs pg_bouncer (on same server as web server) and configure local connections to pg_bouncer which does connection caching to actual Postgres server

- Stored Procedure Implementation
  - Fastest Implementation (> 100,000 orders per minute)
  - Idle in transactions can still occur.

- Metric is Orders Per Minute
  - DB Transactions = (6-7 * OPM/60) ~ 10K – 11K TPS
TPC-E/V
Genesis of TPC-V

- Users are demanding benchmarks to measure performance of databases in a virtual environment
  - Existing virtualization benchmarks model consolidation:
    - Many VMs
    - Small VMs
    - Non-database workloads

- **TPC is developing a benchmark to satisfy that demand: TPC-V**
  - An OLTP workload typical of TPC benchmarks
  - Fewer, larger VMs
  - Cloud characteristics:
    - Variability: mix of small and large VMs
    - Elasticity: load driven to each VM varies by 10X
Benchmark requirements

- **Satisfies the industry need for a benchmark that:**
  - Has a database-centric workload
  - Stresses virtualization layer
  - Moderate # of VMs, exercising enterprise applications
  - Healthy storage and networking I/O content; emphasizes I/O in a virtualized environment
  - *NOT* many app environments in an app consolidation scenario

- **Timely development cycle (1-2 years)**
  - Based on the *TPC-E* benchmark and borrows a lot from it
What is **TPC-E**

- **TPC-E** is the TPC’s latest OLTP benchmark
  - More complex than TPC-C
  - Less I/O than TPC-C
  - A lot of the code is TPC-supplied

- Models a brokerage firm

![Diagram](image)

**Invoke the following transactions** ...

<table>
<thead>
<tr>
<th>READ-WRITE</th>
<th>READ-ONLY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Market-Feed</td>
<td>Broker-Volume</td>
</tr>
<tr>
<td>Trade-Order</td>
<td>Customer-Position</td>
</tr>
<tr>
<td>Trade-Result</td>
<td>Market-Watch</td>
</tr>
<tr>
<td>Trade-Update</td>
<td>Security-Detail</td>
</tr>
<tr>
<td></td>
<td>Trade-Lookup</td>
</tr>
<tr>
<td></td>
<td>Trade-Status</td>
</tr>
</tbody>
</table>

**... against the following data**

- Customer Data
- Brokerage Data
- Market Data
Abstraction of the Functional Components in an OLTP Environment

Legend
- Customer
- Sponsor Provided
- Stock Market
How does this all matter to the PostgreSQL community?

- **TPC is developing a benchmarking kit for TPC-V**
  - First time TPC has gone beyond publishing a functional specification
  - Full, end-to-end functionality
  - Publicly available kit
  - Produces the variability and load elasticity properties of the benchmark
    - Users need not worry about complexities of simulating cloud characteristics
  - Runs against an open source database
  - A “reference” kit; companies are allowed to develop their own kit

- **Anyone can install the kit and pound on the server with a cloud database workload**
  - Removes the high cost of entry typical to TPC benchmarks

- **The reference kit will run on PostgreSQL**
  - ODBC interface allows running the workload against other databases

- **Tentative plans to also release a TPC-E kit**
  - We started out with a kit to run TPC-E; now adding the TPC-V properties
Our dependence on PostgreSQL

- This reference kit will be a very successful new benchmark
  - But only if its performance on the open source database is at least decent compared to commercial databases

- PostgreSQL can benefit a lot from being the reference database for a major new benchmark
  - But only its performance is decent!

- Running the TPC-E prototype on PGSQL 8.4 on RHEL 6.1, we are at ~20% of published TPC-E results
  - Very early results
  - Current testbed is memory challenged
  - Good news: Query plans for the 10 queries implemented look good
  - Long, mostly-read queries => issue is the basic execution path, not redo log, latch contention, etc.
Benchmark Development Status

- **TPC-V Development Subcommittee**
  - 9 member companies
  - 3-4 engineers working actively on the reference kit
  - On version 0.12 of the draft spec
    - Worked through a lot of thorny issues
  - Betting the farm on the reference kit
    - But if we produce a good kit, TPC-V will be an immediate success

- **We expect to make a kit available to member companies in Q3 or Q4**

Bottom line: Cooperating to make the TPC-E/TPC-V reference kits run well on PostgreSQL will greatly benefit all of us
Acknowledgements!

- **VMware Performance Team**
  - Dong, Reza

- **VMware vFabric Postgres Team**
  - David Fetter, Dan, Alex, Nikhil, Scott, Yolanda
Thank You