Introduction to Fair-Use TPC® Benchmarking Kits

Mark Wong
mark.wong@enterprisedb.com
markwkm@postgresql.org

PostgreSQL Conference Europe

24 October 2024

Goal

Understand how TPC® benchmarks are intended to be run so they can be applied to performance testing:

1. **TPC Benchmark™**
   - What is it?
   - Lightly review 4 of them: C, E, H, DS
2. What is fair-use?
   - Not for competitive marketing
   - For system characterization research
3. Open-Source Benchmarking Kits
   - Benchmarkers must develop an implementation
   - Or use an available kit
     - Examples with OSDL's Database Test (DBT) kits
     - Concepts should apply to other available kits: Benchbase, HammerDB, etc.

A brief note of the TPC®

The TPC® (Transaction Processing Performance Council) formed to create good benchmarks for fair competition:

- Members: `https://www.tpc.org/information/who/whoweare5.asp`
- Benchmark specifications and results: `https://www.tpc.org/information/benchmarks5.asp`

TPC Benchmark™ results are more than a rate

- Each benchmark has a *primary metric*
  - Transactions per minute or second
  - Queries run per hour
  - Not all database transaction
- Each result prices the system under test
- 3 years service and support
  - Hardware, must be available at time of publication
  - Software, must have commercial support

TPC Benchmark™ C (TPC-C) *1,000,000 tpmC* result

Many facts can be derived from the primary metric:

- 1,000,000 New Orders processed per minute
- *Scale factor* of at least ~77,000 warehouses
  - ~7 terabytes of raw data
  - ~770,000 users emulated
- Over 2,200,000 database transactions per minute
- Over 36,000 database transaction per second

Total system cost: $8,000,000

- Price / Performance for a 1,000,000 tpmC result
  - $8.00 / tpmC
  - $8 per order per minute
- Enough storage for 60 days of growth

Fair-use

Unless publishing an official result, do **not**:

- Compare to official TPC® publications
- Market against competitors

Instead one may:

- Do system characterization research:
  - Test patches
  - Tune the operating system or database management system
- Ignore parts of the TPC Benchmark™:
  - Auditing
  - Pricing
  - Commercially supported or available hardware or software
  - Rules that make the workload harder to run

Some of the benchmarks

Covering just these 4 current specifications:

- On-Line Transaction Processing (OLTP) - mixtures of read-only and update intensive transactions
  - TPC Benchmark™ C (TPC-C)
  - TPC Benchmark™ E (TPC-E)
- Decision Support (DSS) - queries and data maintenance
  - TPC Benchmark H™ (TPC-H)
  - TPC Benchmark DS™ (TPC-DS)

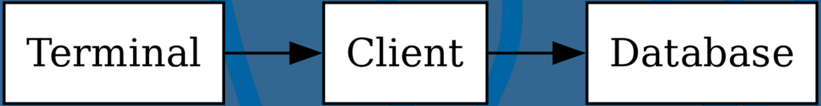
TPC Benchmark™ C (TPC-C)

TPC-C Description

Quoting the specification (1992):

- wholesale supplier with a number of geographically distributed sales districts and associated warehouses
- warehouses maintain stocks for the 100,000 items
- Customers call the Company to
  - place a new order
  - request the status of an existing order

TPC-C System Model



TPC-C rules to note

How you follow them depends on what you want to achieve:

- Database size (scale factor) is number of warehouses
- 100 warehouses is roughly 10 MB of raw data
- Emulate 10 terminals per warehouse
- Implement 5 database transactions run at different rates
- Primary metric: New Order transactions per minute
- Physically limit metric by emulating a person's thinking and keying time

$$\mathit{throughput} = \mathit{warehouses} \times 12.86 \tag{1}$$

- Checkpoint at least every 30 minutes

OSDL Database Test 2 (DBT-2)

Using DBT-2: 1-tier client-server configuration

Two basic tasks to learn[1]:

1. Build the database

```
dbt2 build --warehouses=100 pgsql
```

2. Run tests

```
dbt2 run --stats            \ # collect system stats
         --warehouses=100 \
         --duration=7200  \ # in seconds
         pgsql \
         ./results
```

[1] https://osdldbt.github.io/dbt2/

DBT-2 Results Summary: 100 Warehosues

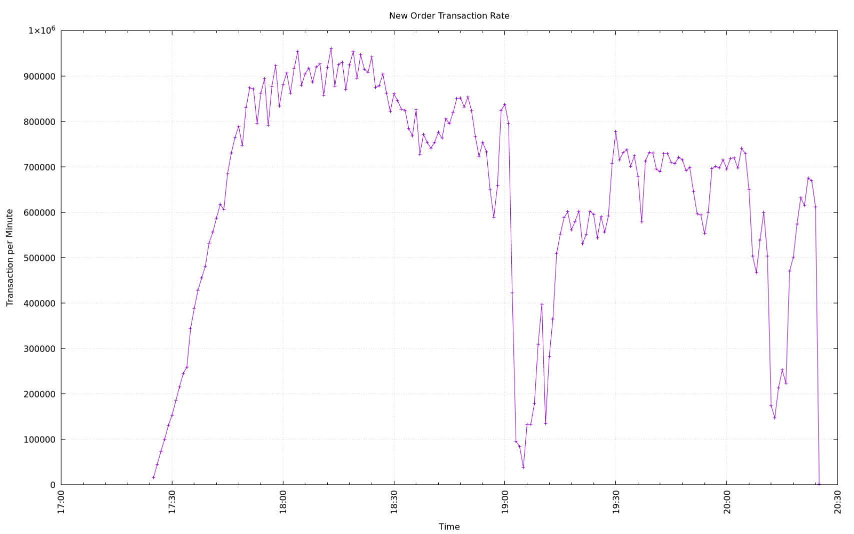
```
============  ======  =========  =========  ===========  ===========  ======
          ..      ..    Response Time (s)            ..           ..      ..
------------  ------  --------------------  -----------  -----------  ------
 Transaction       %    Average     90th %        Total    Rollbacks       %
============  ======  =========  =========  ===========  ===========  ======
    Delivery    4.00      0.003      0.004      7580694            0    0.00
   New Order   44.98      0.003      0.003     85186839       846034    0.99
Order Status    4.00      0.001      0.001      7570877            0    0.00
     Payment   43.02      0.001      0.001     81473933            0    0.00
 Stock Level    4.01      0.001      0.002      7595289            0    0.00
============  ======  =========  =========  ===========  ===========  ======

* Throughput: 709890.32 new-order transactions per minute (NOTPM)
* Duration: 120.0 minute(s)
* Unknown Errors: 0
* Ramp Up Time: 0.0 minute(s)
```

DBT-2: What is a good result?

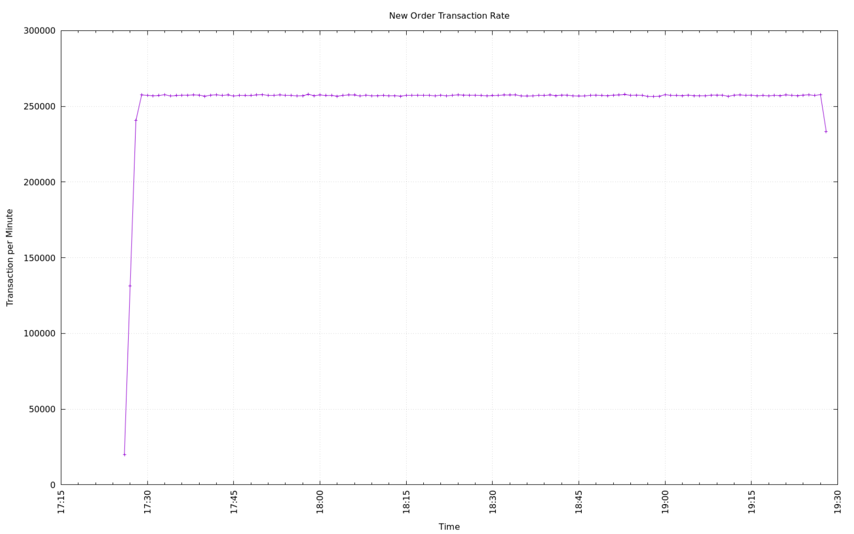
In the spirit of system characterization:

- The primry metric doesn't always tell you everything
- Should also review
  - Transaction rate over time
  - Processor utilization
  - Storage utilization

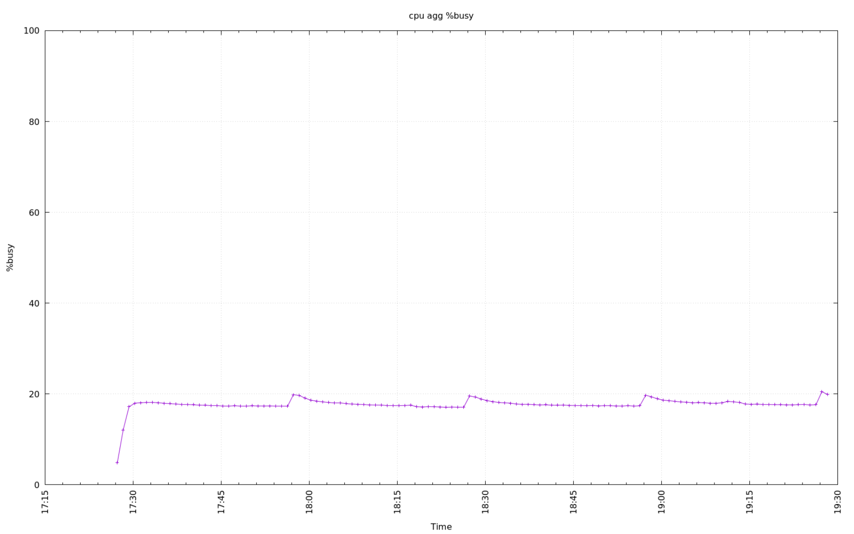
DBT-2: Bad transaction rate



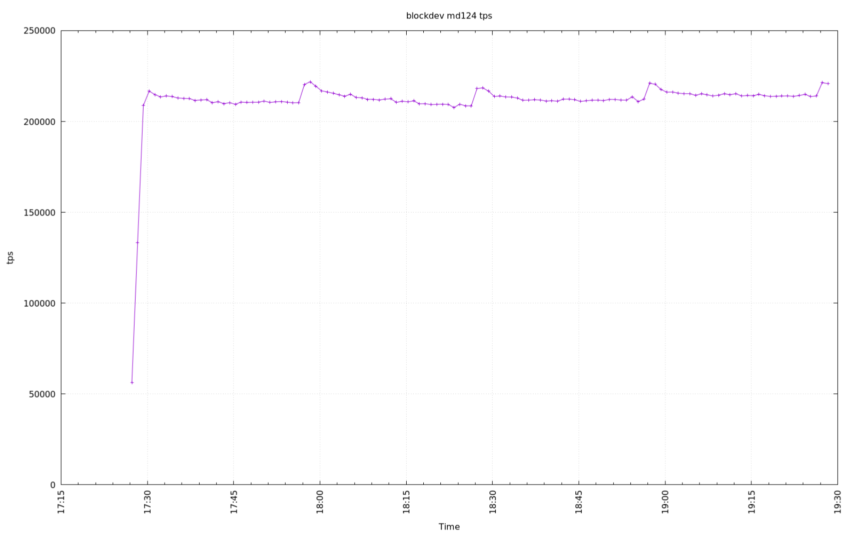
DBT-2: Good Transaction Rate



DBT-2: Processor Utilization



DBT-2: Storage IOPS



DBT-2: One Definition of a good result

Benchmark workload is designed for simulating peak loads:

- Smooth and flat transaction rate of time
- Processors mostly utilized
- Storage throughput mostly utilized

TPC-C-like Advanced Usage

Other ways TPC-C-like kits can be used:

- Client- vs Server-side application logic
- Adjust transaction mix
- Adjust thinking/keying times
- Partition the database
- Partition the driver

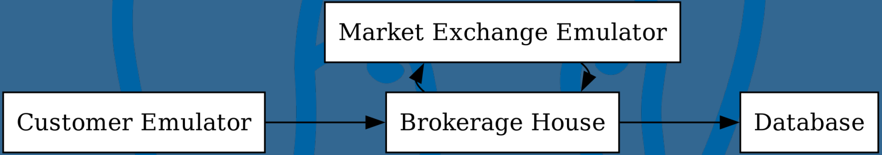
TPC Benchmark™ E (TPC-E)

TPC-E Description

Quoting the specification (2006):

- brokerage firm
- executes transactions related to the firm's customer accounts

TPC-E System Model



- **Customer Emulator** - Workstations, laptops, cell phones
- **Brokerage House** - Performance reports on brokers, customers' position, security research, market analysis, buying and selling securities, review trade status, updating trade requests
- **Market Exchange Server** - Trade confirmations, tracking market activity (i.e. processing *ticket tape*)

TPC-E rules to note

Whether you follow them depends on what you want to achieve:

- The database size is determined by the number of customers
- Implement 11 transactions run at different rates
- Primary metric is the number of **Trade Result** transactions per second
- The metric is constrained around $TotalCustomers/500$:
- Checkpoint at least every 30 minutes

OSDL Database Test 5 (DBT-5)

Using DBT-5 in three basic steps[3] on a single system

1. Register, download, extract to /opt/egen and build TPC-E Tools[2]

```
dbt5 build-egen /opt/egen
```

2. Build the database:

```
dbt5 build -t 5000 pgsql
```

3. Run tests

```
dbt5 run --tpcetools=/opt/egen \
         --stats \ # collect system stats
         -d 120  \ # test duration in seconds
         -t 5000 \ # customers to build
         -u 1    \ # number of users
         pgsql ./results
```

[2] https://www.tpc.org/tpc_documents_current_versions/current_specifications5.asp

[3] https://osdldbt.github.io/dbt5/

DBT-5 Results Summary

```
Reported Throughput:           14.47 trtps  Configured Customers:         5000
==========================================  ==================================


Response Times (s)     Minimum     Average  90th %tile     Maximum
==================  ==========  ==========  ==========  ==========
      Trade Result        0.00        0.01        0.01        0.02
...

   Transaction Mix   Txn Count   Mix %tile  Rollbacks     Warnings     Invalid
==================  ==========  ==========  ==========  ==========  ==========
      Trade Result        1737       8.432           0           0           0
...

Test Duration and Timings
==================================================================  ==========
                                            Ramp-up Time (minutes)         0.0
                                    Measurement Interval (minutes)         2.0
    Total Number of Transactions Completed in Measurement Interval       20601
```

DBT-5: What is a good result?

Same as DBT-2:

- Benchmark designed for simulating peak loads
- Transaction rates smooth and flat
- Processors well utilization
- Storage well utilized

TPC-E-like Advanced Usage

Just to mention some items:

- Use pacing delays
- Customize stored procedures, but may still need to return pass-able results

TPC-C vs TPC-E

How to choose?

- TPC-C is less complex than the TPC-E in almost all aspects
- TPC-E is considered a more modern workload
- TPC-E requires less storage per processor than the TPC-C, when run to specification

TPC Benchmark™ H (TPC-H)

TPC-H Description

Quoting the specification (1999):

- any industry which must manage, sell or distribute a product worldwide
- queries … are of an ad hoc nature
- queries provide answers to the following classes of business analysis
  - pricing and promotions
  - supply and demand management
  - profit and revenue management
  - customer satisfaction study
  - market share study
  - shipping management

TPC-H Workload

The benchmark is composed of three parts, broken up into two categories:

- Load test - loading, indexes, analyzing, sorting, etc.
- Performance test
  - Power Test - run 22 queries and 2 data refresh operations
  - Throughput Test - run multiple simultaneous Power Tests

TPC-H rules of interest

- The benchmark is running both the load and performance test
- The **Scale Factor** is roughly equivalent to gigabytes of raw data
- There is a fixed set of valid **Scale Factors**: 1, 10, 30, 100, 300, 1000, 3000, 10,000, 30,000, 100,000
- The number of streams for the **Throughput Test** is based on the **Scale Factor**
- Indexes are only allowed on specific columns
- Cannot rewrite any of the queries (although some permit using an alternate syntax)
- The primary metric is a Query-per-hour score based on the results of the Power and Throughput Tests.

OSDL Database Test 3 (DBT-3)

Using DBT-3

Two basic steps[4]:

1. Register, download, extract to /opt/dbgen and build TPC-H Tools[5]

```
dbt3 build-dbgen pgsql /opt/dbgen
```

2. Run tests

```
dbt3 run --tpchtools=/opt/dbgen \
         --stats                 \ # collect system stats
         --explain               \ # EXPLAIN ANALYZE
         --scale-factor 1 \
         pgsql \
         ./results
```

[4] https://osdldbt.github.io/dbt3/

[5] https://www.tpc.org/tpc_documents_current_versions/current_specifications5.asp

DBT-3 Results Summary: Scale Factor 100

```
       Composite Score:    12211.59
Load Test Time (hours):         .44
      Power Test Score:    11701.87
 Throughput Test Score:    12743.53
```

- **Composite Score** represents a Query-per-hour metric
- Calculated from:

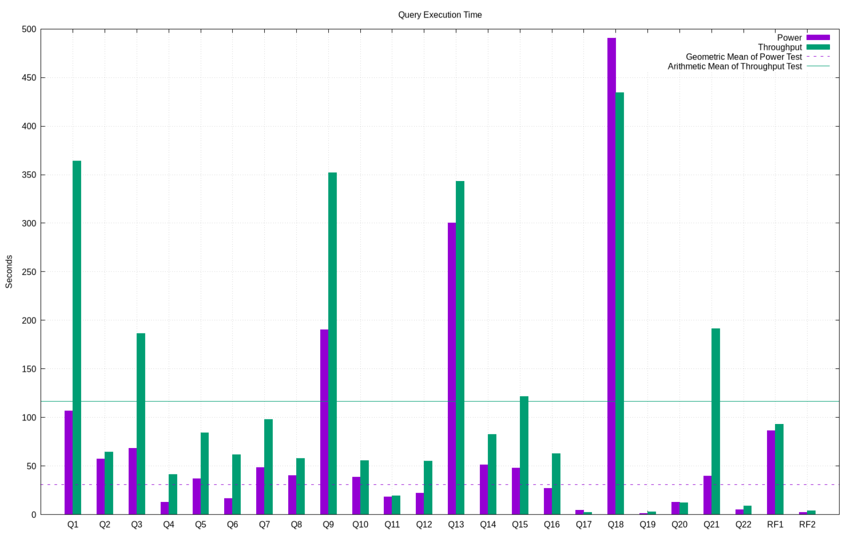
$$\sqrt{PowerTestScore \times ThroughputTestScore} \tag{2}$$

- **Power Test Score** is weighted geometric mean of individual query and refresh stream execution times
- **Throughput Test Score** is a weighted measurement of the time taken to run this portion of the test

DBT-3 What is a good result?

- **Composite Score** goes up whenever query execution times goes down.

DBT-3 Performance Test Bar Plot



DBT-3: Testing individual queries

1. Load data

```
dbt3 run --tpchtools=/opt/dbgen \
         --load \ # just run load test
         --scale-factor 1 \
         pgsql \
         ./results
```

2. Run a specific query

```
dbt3 run-query 9
        --tpchtools=/opt/dbgen \
        --explain              \ # EXPLAIN ANALYZE
        --scale-factor 1 \
        pgsql
```

TPC-H-like Advanced Usage

Just to mention some items:

- Use non-confirming scale factor
- Customize number of throughput streams
- Rewrite queries
- Use non-specification compliant indexes

TPC Benchmark™ DS (TPC-DS)

TPC-DS Description

Quoting the specification (2015):

- models the decision support functions of a retail product supplier
- snowflake schema
- query classes
  - reporting
  - ah hoc
  - iterative OLAP
  - data mining

TPC-DS Workload

Similar to the TPC-H, the TPC-DS is also is composed of the same kind of parts, broken up into two categories:

- Load test - loading, indexes, analyzing, etc.
- Performance test
  - Power Test - run 99 queries
  - Throughput Test - run multiple simultaneous Power Tests
  - Data Maintenance Test - similar to TPC-H refresh streams
  - Repeat Throughput Test
  - Repeat Data Maintenance Test

TPC-DS rules of interest

- The **Scale Factor** is roughly equivalent to gigabytes of raw data.
- There is a fixed set of valid **Scale Factors**: 1000, 3000, 10,000, 30,000, 100,000
- The number of refresh streams for the **Throughput Test** is based on the **Scale Factor**
- Primary metric is Query-per-hour calculated from all tests

$$\left\lfloor \frac{SF \times S_q \times 99}{\sqrt[4]{T_{Power} \times S_q \times (T_{TT1} + T_{TT2}) \times (T_{DM1} + T_{DM2}) \times 0.01 \times S_q \times T_{Load}}} \right\rfloor \tag{3}$$

OSDL Database Test 7 (DBT-7)

DBT-7 Results Summary: Scale Factor 1

```
Queries per Hour: 1340

...

Test                 Start Timestamp        End Timestamp          Elapsed Time
=================    ====================   ====================   ================
Database Load        2024-06-27 21:18:06    2024-06-27 21:18:41    00:00:34.3884
Power Test           2024-06-27 21:18:46    2024-06-27 21:25:44    00:06:57.895323
Throughput Run 1     2024-06-27 21:25:47    2024-06-27 21:33:45    00:07:57.972586
Refresh Run 1        2024-06-27 21:33:45    2024-06-27 21:35:04    00:01:19.113499
Throughput Run 2     2024-06-27 21:35:04    2024-06-27 21:43:12    00:08:07.926494
Refresh Run 2        2024-06-27 21:43:12    2024-06-27 21:44:33    00:01:20.907339
=================    ====================   ====================   ================

 Q        Minimum        25th Percentile        Median        75th Percentile        Maximum
--  ----------------  -----------------  ----------------  -----------------  ----------------
 #   Run1     Run2     Run1      Run2     Run1     Run2     Run1      Run2     Run1     Run2
==  =======  =======  =======  ========  =======  =======  =======  ========  =======  =======
 1     68.2    201.3     68.7     208.4    205.6    208.9    205.6     212.1    205.6    212.1
...
```

Using DBT-7

Two basic steps[6]:

1. Register, download, extract to /opt/dsgen and build TPC-DS Tools[7]

```
dbt7 build-dsgen pgsql /opt/dsgen
```

2. Run tests

```
dbt7 run --tpcdstools=/opt/dsgen \
         --stats             \ # collect system stats
         -d postgresqlea     \ # EXPLAIN ANALYZE
         --scale-factor 1 \
         pgsql \
         ./results
```

[6] https://osdldbt.github.io/dbt7/

[7] https://www.tpc.org/tpc_documents_current_versions/current_specifications5.asp

DBT-7 What is a good result?

- Same consideration as DBT-3 above.
- Reducing query execution times should improve scores
- Reducing load time should also improve composite score

DBT-7: Testing individual queries

1. Run tests

```
dbt7 run --tpcdstools=/opt/dsgen \
         --scale-factor 1 \
          pgsql \
          ./results
```

2. Run a query

```
dbt7 run-query 1
        --tpcdstools=/opt/dsgen \
        -d postgresqlea  \ # EXPLAIN ANALYZE
        --scale-factor 1 \
         pgsql
```

TPC-DS-like Advanced Usage

Just to mention some items:

- Use non-confirming scale factor
- Customize number of throughput streams
- Rewrite queries

TPC-H vs. TPC-DS

Which one should you run?

- TPC-H 22 vs TPC-DS 99 queries
- TPC-DS is a denormalized star-schema/snowflake schema

Hayley Jane Wakenshaw

```
          __      __
         /  \~~~/  \  . o O ( Thank you! )
   ,----(     oo    )
  /      \__     __/
 /|         (\  |(
^ \   /___\  /\ |
   |__|   |__|-"
```