EasyChair Preprint

№ 3635

Quantifying Scalability Using Extended TPC-DS Performance Metric

Guoheng Chen, Miso Cilimdzic and Timothy Johnson

June 17, 2020

Quantifying Scalability Using Extended TPC-DS Performance Metric

Guoheng Chen, Miso Cilimdzic, Timothy Johnson

Microsoft Corporation

One Microsoft Way

Redmond, WA 98052 USA

{guche, misoc, tijohnso}@microsoft.com

Abstract. The TPC Benchmark™DS (TPC-DS) is a decision support benchmark that models several generally applicable aspects of a decision support system, including data loading, queries and data maintenance. The benchmark provides a representative evaluation of the System Under Test's (SUT) performance as a general-purpose decision support system. TPC-DS defines three primary metrics. The most important is the Performance Metric, QphDS@SF, reflecting the TPC-DS query throughput at various scale factors. Performance metrics at different scale factors are not comparable, due to the substantially different computational challenges found at different data volumes. Data analytics platforms have two main components, Compute and Storage. In the last decade, many cloud data analytics platforms have begun to separate compute and storage. With this separation, data analytics platforms now can scale compute in and out independently on the same dataset with the same storage. The performance metric QphDS@SF at different compute levels demonstrates how well the system performance scales. This article shows some scalability analysis in the TPC-DS workload on a cloud data analytics platform and proposes a benchmark as an extension to TPC-DS.

1. Introduction

TPC-DS Benchmark

The TPC Benchmark™DS (TPC-DS) (Gunther, 2007) (TPC, n.d.)is a decision support benchmark that models several generally applicable aspects of a decision support system, including queries and data maintenance. The benchmark provides a representative evaluation of the System Under Test's (SUT) performance as a general-purpose decision support system.

TPC-DS (TPC, n.d.)defines the following scale factors: 1TB, 3TB, 10TB, 30TB, 100TB and 1GB (for qualification database only).

TPC-DS defines a workload with following sequence of tests:
a) Database Load Test
b) Power Test
c) Throughput Test 1
d) Data Maintenance Test 1
e) Throughput Test 2
f) Data Maintenance Test 2
The primary performance metric of the benchmark is QphDS@SF, defined as:

$$QphDS@SF = \left| \frac{SF \times Q}{\sqrt[4]{T_{PT} \times T_{TT} \times T_{DM} \times T_{LD}}} \right|$$

Where:

- SF is defined in Clause 3.1.3, and is based on the scale factor used in the benchmark.
- Q is the total number of weighted queries: Q=Sq*99, with Sq being the number of streams executed in a Throughput Test.
- T_{PT}=T_{Power}*Sq, where T_{Power} is the total elapsed time to complete the Power Test, as defined in Clause 7.4.4, and Sq is the number of streams executed in a Throughput Test.
- T_{TT}= T_{TT1}+T_{TT2}, where T_{TT1} is the total elapsed time of Throughput Test 1 and T_{TT2} is the total elapsed time of Throughput Test 2, as defined in Clause 7.4.6.
- T_{DM}= T_{DM1}+T_{DM2}, where T_{DM1} is the total elapsed time of Data Maintenance Test 1 and T_{DM2} is the total elapsed time of Data Maintenance Test 2, as defined in Clause 7.4.9.
- T_{LD} is the load factor computed as T_{LD}=0.01*Sq*TLoad, where Sq is the number of streams executed in a Throughput Test and TLoad is the time to finish the load, as defined in Clause 7.1.2.
- T_{PT}, T_{TT}, T_{DM} and T_{LD} quantities are in units of decimal hours with a resolution of at least 1/3600th of an hour.

Results at the different scale factors are not comparable, due to the substantially different computational challenges found at different data volumes. Similarly, the system price/performance may not scale down linearly with a decrease in database size due to configuration changes required by changes in database size.

Vendor Competition of TPC-DS

Cloud data analytic platform vendors have been using TPC-DS queries to measure and compare query performance. (Gigaom, n.d.) Most of the performance test is on power run. It measures the ability of the system to process a sequence of queries in the least amount of time in a single stream fashion. But TPC-DS performance gives equal weight to Load, Power Run, Throughput Run and Data Maintenance, taking the number of streams into consideration. It models the challenges of business intelligence systems where operational data is used both to support the making of sound business decisions in near real time and to direct long-range planning and exploration. It measures more completely the overall performance of a data analytic platform system. Generally, vendors compete at certain scale factor level, 1TB, 10TB, 30TB, or 100TB, etc. Recently, we see more audit reports on TPC-DS. (TPC, n.d.)

Separation of Storage and Compute in Cloud Data Analytic Platform

Data analytics platforms have two main components, Compute and Storage. In the last decade, many cloud data analytics platforms have begun to separate compute and storage. (Snowflake Technology, n.d.) (Microsoft, n.d.) With this separation, data analytics platforms now can scale compute in and out independently on the same dataset with the same storage. Vendors provide this scalability by changing the slice of a node, the number of nodes, and or types of nodes, with different CPUs, number of cores, memory, local disk, network bandwidth, etc.

Although the TPC-DS benchmark states that performance metric results at the different scale factors are not comparable, the performance metric of the same dataset running on different hardware

configurations can be very interesting. Further analysis for each test in the TPC-DS workload can also be interesting. It measures not only how well the system performs at each hardware configuration, but also how well the system scales in/out between different hardware configuration levels. Furthermore, if we can design a systematic way to compare the performance and scalability, that would be even more interesting. That is what we try to do in this paper.

2. Related Work in Measuring Scalability

Quantifying scalability can be a challenge. In computer architecture, Amdahl's law is a formula which gives the theoretical speedup in latency of the execution of a task at fixed workload that can be expected of a system whose resources are improved. It is named after computer scientist Gene Amdahl, and was presented at the AFIPS Spring Joint Computer Conference in 1967.

In Amdahl's law (Wikipedia, n.d.), most workloads have two portions, one that can be executed in parallel, and one that must be serialized. Let the total workload execution for single processor is $T_{1.\,If}$ the serial faction is σ, 0 < σ < 1, then the parallelized portion is 1- σ. The total execution time with p processors T_p

$$T_p = \sigma T_1 + \frac{(1-\sigma)T_1}{p}$$

So p-way scale up is:

$$C(p) = \frac{p}{1+\sigma(p-1)}$$

In Universal Scalability Law (USL), Neil Gunther extended Amdahl's law to account for the additional overhead due to interprocess communication. This can be explained using a typical database application, where multiple server processes need to communicate with a single database. Even though each process can progress without explicit communication with other processes when reading elements from the database, they should explicitly communicate with other processes when updating the database in order to maintain the ACID properties of transactions. If there are p processors running in parallel, then each processor should communicate with p-1 number of other processors. Hence on average, p(p-1) number of interactions take place. In Neil Gunther's book, Guerrilla Capacity Planning, (Gunther, 2007) he defines relative capacity

$$C(N) = \frac{N}{1+\alpha(N-1)+\beta N(N-1)}$$

So in USL

$$C(p) = \frac{p}{1 + \sigma(p-1) + \kappa p(p-1)}$$

Neil Gunther later further normalized the formula to

$$C(p) = \frac{\gamma N}{1 + \sigma(p-1) + \kappa p(p-1)}$$

The three coefficients, α, β, γ, in this equation can be identified respectively with the three Cs [Gunther 2018,SF ACM 2018]:

- CONCURRENCY or ideal parallelism (with proportionality γ), which can also be interpreted as either:
    a. the slope associated with linear-rising scalability
    b. the maximum throughput attainable with a single load generator, i.e., X(1) = γ
- CONTENTION (with proportionality α) due to waiting or queueing for shared resources
- COHERENCY or data consistency (with proportionality β) due to the delay for data to become consistent, or cache coherent, by virtue of point-to-point exchange of data between resources that are distributed

Amdahl's law is a special case of USL when β = 0 and γ = 1

Both Amdahl's law and USL provide a quantitative approach to measure system scalability. We leverage some of the ideas in these two studies to analyze TPC-DS performance metric on a cloud Data Analytics Platform.

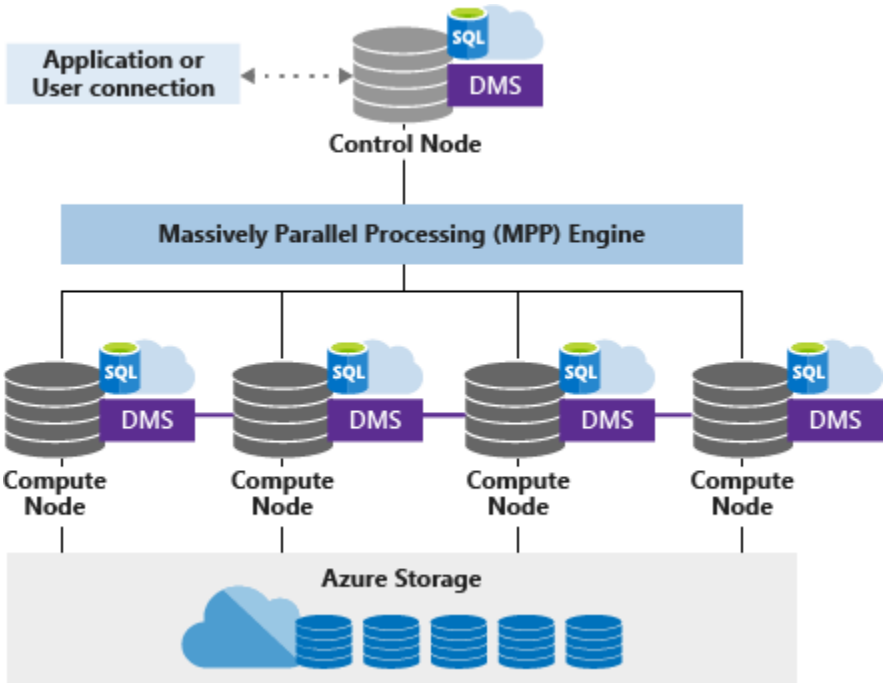
3. An experiment for measuring scalability on Azure Synapse Analytics

Architecture Overview

Synapse SQL leverages a scale-out architecture to distribute computational processing of data across multiple nodes. The unit of scale is an abstraction of compute power that is known as a data warehouse unit. Compute is separate from storage, which enables you to scale compute independently of the data in your system. The current offering in Gen2 architecture. (Microsoft, n.d.)

Synapse SQL uses a node-based architecture. Applications connect and issue T-SQL commands to a Control node, which is the single point of entry for Synapse SQL. The Control node runs the MPP engine, which optimizes queries for parallel processing, and then passes operations to Compute nodes to do their work in parallel.

The Compute nodes store all user data in Azure Storage and run the parallel queries. The Data Movement Service (DMS) is a system-level internal service that moves data across the nodes as necessary to run queries in parallel and return accurate results.



Synapse SQL leverages Azure Storage for local and remote storage management. Permanent data is sharded into distributions in Azure Storage to optimize the performance of the system.

The Compute nodes provide the computational power. Distributions map to Compute nodes for processing. The number of compute nodes ranges from 1 to 60 and is determined by the service level objective for Synapse SQL. Currently, it supports the following service levels: DW100c, DW200c, DW300c, DW400c, DW500c, DW1000c, DW1500c, DW2000c, DW2500c, DW3000c, DW5000c, DW6000c, DW7500c, DW10000c, DW15000c, DW30000c. (Microsoft, n.d.)

Experiment Setup

Workload: Modified TPC-DS workload. In this workload, one extra PowerRun has been added after load. This is to capture the difference between cold data and warm data in power run. Only cold data is used to calculate performance metric QphDS@SF. Adding extra test here is not harmful as this test is mainly focusing on scalability, not an official benchmark run. So the workload sequence is

a) Database Load Test
b) Power Test cold
c) Power Test warm

d) Throughput Test 1
e) Data Maintenance Test 1
f) Throughput Test 2
g) Data Maintenance Test 2

Scale factors: 10000 (10TB)

Dataset: Generated with dsdgen and storage in Azure Storage. Number of blob files for each table ranges from 1 to 1440.

Number of throughput test query stream: 4

Scalability Test: Use N of the supported service level objective and run the modified TPC-DS workload.

Test Result

All the operations in TPC-DS runs are measured in unit of second. The test result is not presented here. Since this experiment is to measure the scalability of the system, so the test result has been normalized based on some formula. The normalization process does not affect the scalability characteristics of the system. From the normalized test result, we have come up with an extended performance metric.

4. Extended TPC-DS Performance Metric

In Azure Synapse Analytic, service level objective represents the capacity of the system approximatively.

In this experiment, we have multiple different service level objective(SLO), SLO_1, SLO_2, SLO_2, … , SLO_n

Normalize capacity to (p_1, p_2, …, p_n) such as (1, 1.5, 2, …., 30).

Total capacity of these tests is defined as $P = \sum_{i=1}^{n} p_i$

For each test in TPC-DS workload sequence, use load as an example, for each SLO, measure of the load test performance in seconds T_1, T_2, …, T_n

Normalized measured performance is X(1), X(2), …, X(n)

Define relative capacity as $C_n = \frac{X(n)}{X(1)}$, capacity used in the tests becomes C_1, C_2, …, C_3

Define efficiency as $e_i = \frac{C_i}{p_i}$

Define weighted scalability at scale as $w_i = \frac{p_i}{P} \times e_i$

Define scalability of load operation in TPC-DS workload overall system as $S_{load} = \sum_{i=1}^{n} w_i$

Similarly, define scalability of PowerRun, ThroughutRun and Data Maintenance operations in TPC-DS as $S_{PowerRun}, S_{ThroughutRun}, S_{Maintenance}$. We further simplify them as $S_{LD}, S_{PT}, S_{TT}, S_{DM}$

Finally, we propose an extended benchmark performance metric, scaled TPC-DS performance metric as

$$ScaledQphDS@SF = \left| \frac{SF \times Q}{\sqrt[4]{\frac{T_{PT}}{S_{PT}} \times \frac{T_{TT}}{S_{TT}} \times \frac{T_{DM}}{S_{DM}} \times \frac{T_{LD}}{S_{LD}}}} \right|$$

Note: It is assumed S_{LD}, S_{PT}, S_{TT}, S_{DM}, T_{LD}, T_{PT}, T_{TT}, T_{DM} are not 0. This formula can be rewritten as

$$ScaledQphDS@SF = \left| \frac{SF \times Q}{\sqrt[4]{T_{PT} \times T_{TT} \times T_{DM} \times T_{LD}}} \right| \times \sqrt[4]{S_{PT} \times S_{TT} \times S_{DM} \times S_{LD}}$$

Which can be simplified as

$$ScaledQphDS@SF = QphDS@SF \times \sqrt[4]{S_{PT} \times S_{TT} \times S_{DM} \times S_{LD}}$$

Property of the extended performance metric ScaledQphDS@SF:

- If a system is a linear scale system, S_{LD} x S_{PT} x S_{TT} x S_{DM} = 1, ScaledQphDS@SF is simplified as SCaledQphDS@SF.
- Scalability of each operation in TPC-DS, S_{LD} x S_{PT} x S_{TT} x $S_{DM,}$ has the same weight, this is similar to T_{LD} x T_{PT} x T_{TT} x $T_{DM.}$ Any suboptimal scalability can cause the overall scalability down.

Scalability Measurement Result

Different operations in TPC-DS workload has different scalability characteristics. Below is portion of the measure in normalized format:

Load operation scalability:

Load	Normalized Capacity(p)	Relative Capacity(C)	Efficiency(e)	Weighted Scalability(wi)
SLO_1	1	1	1	0.0122
SLO_2	2	2.5003	1.2502	0.0305
SLO_3	2.5	3.5322	1.4129	0.0431
SLO_4	3	4.3272	1.4424	0.0528

SLO_5	5	6.7421	1.3484	0.0822
… …	… …	… …	… …	… …
SLO_N	… …	… …	… …	… …
	Total Capacity(P)			**System Scalability**
	82			0.9154

PowerRun operation scalability:

PowerRun	Normalized Capacity(p)	Relative Capacity(C)	Efficiency(e)	Weighted Scalability(wi)
SLO_1	1	1	1	0.012
SLO_2	2	2.37	1.18	0.029
SLO_3	2.5	2.84	1.14	0.035
SLO_4	3	3.22	1.07	0.039
SLO_5	5	4.57	0.91	0.056
… …	… …	… …	… …	… …
SLO_N	… …	… …	… …	… …
	Total Capacity(P)			**System Scalability**
	82			0.424

Throughput Run operation scalability:

ThroughputRun1	Normalized Capacity(p)	Relative Capacity(C)	Efficiency(e)	Weighted Scalability(wi)
SLO_1	1	1	1	0.0122
SLO_2	2	2.9069	1.4535	0.0355
SLO_3	2.5	3.9679	1.5871	0.0484
SLO_4	3	4.756	1.5853	0.058
SLO_5	5	8.0376	1.6075	0.098
… …	… …	… …	… …	… …
SLO_N	… …	… …	… …	… …
	Total Capacity(P)			**System Scalability**
	82			0.926

Data Maintenance operation scalability:

DataMaintenance1	Normalized Capacity(p)	Relative Capacity(C)	Efficiency(e)	Weighted Scalability(wi)

SLO_1	1	1	1	0.0122
SLO_2	2	1.1172	0.5586	0.0136
SLO_3	2.5	1.1523	0.4609	0.0141
SLO_4	3	1.2328	0.4109	0.015
SLO_5	5	1.4732	0.2946	0.018
… …	… …	… …	… …	… …
SLO_N	… …	… …	… …	… …
	Total Capacity(P)			**System Scalability**
	82			0.1612

$$\mathbf{ScaledQphDS@SF} = \sqrt[4]{S_{PT} \times S_{TT} \times S_{DM} \times S_{LD}} \times \mathbf{QphDS@SF} = 0.490619 \times \mathbf{QphDS@SF}$$

While TPC-DS performance metric QphDS@SF measure the overall performance of the system, extended performance metric ScaledQphDS@SF also measures the scalability of the system.

5. Future work

- Quantifying scalability is a changeling problem. In this paper, we explored one dimension of the problem, that is a sparse compute configuration on one fixed data set. Based on that, we proposed a modified TPC-DS performance metric, ScaledQphDS@SF. TPC-DS has another important primary metric, the price performance metric.

$$\$ / QphDS@SF = \frac{P}{QphDS@SF}$$

- Price performance metric has been purposely ignored in this paper as we just focus on system scalability. But data analytic platform vendors eventually compete at price performance. TPC-DS Price performance metric at different scale factor level is not comparable, but it should be comparable at different compute power level. Some research is needed to investigate whether a similar scaled price performance metric is needed.
- In this paper, we focus on scale out architecture. Even scale up architecture can use this extended performance metric to measure its scalability. Scale up architecture might have different factors affecting scalability. The 3C's in USL might be different from scale out architecture.
- Most of the research on scalability in Data Analytic Platform has been with different hardware configurations against the same dataset. There is another dimension to the problem. The dataset can change, especially growing significantly, with the same or difference hardware configuration. For further research in these combined scenarios, not only we need compute to scale out continuously, but we also need data set scale out

continuously. Dsdgen currently officially only supports 1TB, 3TB, 10TB, 30TB, 100TB. This can be a limiting factor for this type of experiment.

- With both compute dataset scaling out continuously, the test matrix grow exponentially, how to choose a right set of test to access scalability of the whole system also need further research.

6. Acknowledgement

We should like to thank the Azure Synapse Analytics engineering team for their insights in many areas of system performance evaluation.

7. References

Gigaom. (n.d.). *Cloud Data Warehouse Performance Testing*. Retrieved from Gigaom: https://gigaom.com/report/cloud-data-warehouse-performance-testing/

Gunther, N. J. (2007). *Guerrilla Capacity Planning.* Springer; 2007 edition (November 14, 2006).

Microsoft. (n.d.). *Azure Synapse Analytics (formerly SQL DW) architecture*. Retrieved from https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/massively-parallel-processing-mpp-architecture

Microsoft. (n.d.). *Azure Synapse Analytics pricing* . Retrieved from https://azure.microsoft.com/en-us/pricing/details/synapse-analytics/

Microsoft. (n.d.). *Data warehousing*. Retrieved from https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/data-warehousing

Snowflake Technology. (n.d.). Retrieved from https://www.snowflake.com/blog/elasticitycomputeandstorageseparation/

TPC. (n.d.). *TPC BENCHMARK ™ DS*. Retrieved from TPC: http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-ds_v2.13.0.pdf

TPC. (n.d.). *TPC-DS - Most Recently Published Results*. Retrieved from TPC: http://www.tpc.org/tpcds/results/tpcds_last_ten_results5.asp

Wikipedia. (n.d.). *Amdahl's law*. Retrieved from https://en.wikipedia.org/wiki/Amdahl%27s_law