TPC-D: Benchmarking for Decision Support

Carrie Ballinger, NCR Parallel Systems

I. Introduction: A Child of the Nineties

In the 1990 world that the TPC-D development effort was born into, decision support was a minor league game. The term "data warehouse" had not yet been coined, and many of the parallel technologies taking aim at decision support were still on the drawing board. Global competition had not yet forced corporations into serious self-examination, nor had it taught them to respect and cherish their data. So the knowledge base, the experience level, and the interest in decision support, both in and out of the TPC Council, was in its infancy.

Subsequently, the initial group of individuals from within the TPC Council that stepped forward to build TPC-D relied heavily on transaction processing experiences and ideas. But in looking around the TPC-D committee meeting room five years later, as the ink was drying on the final specification document in April of 1995, the faces of the contributers, as well as their backgrounds, were completely different from those of the original group. SQL query and parallel processing expertise was well represented from almost every vendor. The TPC-D committee membership, as well as its benchmark, had transformed during that five year period, at the same tempo and in the same direction as vendor products and industry need, making TPC-D a true child of the nineties.

Snapshot of TPC-D Version 1

- The application: 17 complex queries and two update functions.
- Update functions insert or delete .1% rows into the two largest tables.
- Modeled after an ad hoc workload on a continuously available database.
- Designed with business questions built first, followed by SQL.
- No think time: A constant flow of SQL requests within each stream.
- Primary metrics: Two for performance (Power for single-user, Throughput for multi-user); one for price-performance.
- Volume: May be published at one of eight distinct scale factors.
- Secondary metrics: 1) Reported load time, and 2) a disk storage ratio (total disk space in the system divided by the scale factor).
- ACID tests: Concurrent update and query capability required.
- Version 1 completed in April 1995, Version 2 is under development.

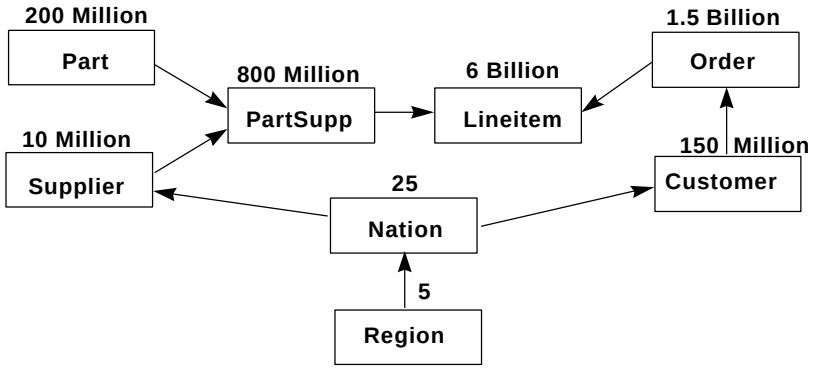
II. Benchmark Evolution Overview

In May of 1990, at the same TPC Council meeting where Digital (DEC) proposed what was to become the TPC-C benchmark, Teradata, a small parallel processing company based in Los Angeles, put a decision support benchmark on the table. The original Teradata benchmark, while triggering the TPC-D development process, served its purpose and died a quick death in the first committee meeting, for two opportune reasons: First, the development committee wanted a convincing real-world story behind the benchmark, while the original proposal

lacked a business model; and second, vendor-neutral, non-biased queries were considered critical, while the decision support in the Teradata proposal reflected only one vendor's experiences. These two motives for starting with a clean slate foreshadowed two key qualities that gave the final TPC-D integrity and staying power five years later: Being a real-world application, and vendor neutrality.

Early on the committee sent out a survey of DS characteristics to the participating vendors' customer base, asking about database volumes, tables-per-query as well as SQL functionality. Integrating these diverse DS user experiences was key to establishing the foundation of the benchmark, as was the decision to use customer-like business questions that users could identify with, marking a clear distinction from the SQL-based statements popular in previous query benchmarks. As a result, the emerging TPC-D workload was driven from two directions: First by the formulation of believable business questions, and second, by using results of the customer survey as a guidepost for functionality.

A cooperative effort by Red Brick and IBM established the TPC-D Parts-Supplier database, and beginning to break away from it's OLTP heritage, the committee voted to accept a large detail data table into this schema, the Lineitem table. Inclusion of a detail data table, which comprised about 67% of the total space, set the stage for many of the DS complexities illustrated in the final benchmark. The schema models a continuously available system, where Lineitems ordered are the key business entity.



Number of Rows per Table at Scale Factor 1000 (1 TB)

Twenty-one candidate business questions originating from five different vendors were put through a grueling "business-reality" test by the committee, and those that passed were given a rigorous SQL code review. All queries went through some change, but the vast majority, 17 out of 21, are present in the current TPC-D query set, a much higher number than the original goal of 8 to 10. To be given equal weight with the queries, two update functions, a mass insert (UF1) and a mass delete (UF2), were designed, simulating periodic or trickle updates. Coincident with query development, Informix stepped forward to produce a TPC-D data generation utility, allowing testing of the SQL to begin across several vendor test systems, and providing a sounding board for further expansion of the schema.

Constructing a fair performance metric was the largest challenge faced by the committee, who eventually voted to split the problem in two. Agreeing on the use of two performance metrics

resolved this novel quandry: Should a product be rewarded more for being able to deliver optimal query times to a single user, or for being able to establish high throughput with many users on the system? Both qualities of a decision support product were deemed important, and two metrics, known as Power and Throughput, were a way to reward and recognize each aspect.

At the same time as the TPC-D application was being built up in the '93 to '94 timeframe, decision support applications outside the Council began catching the attention of the industry press, and data warehouses began basking in early popularity. By the time IBM published the first TPC-D benchmark ever with their DB2 PE database in December of 1995, decision support was mainstream.

III. Significance of TPC-D's Database Focus

TPC-D is first and foremost a database benchmark, whose intent is to simulate ad hoc queries that provide answers to real-world business questions. [Transaction Processing Performance Council, 1997] Performance in TPC-D is especially influenced by the intelligence of the query optimizer, table partitioning schemes, SQL functionality and advanced indexing techniques.

TPC-D requires a full-function DBMS, SQL-based and able to support concurrent query and update. This prerequisite is only one of the high standards consciously set by the development committee, both for the sake of comparability and to encourage database product evolution. TPC-D has become a major carrot in vendor product planning organizations. Every database vendor that has published TPC-D, and many who have not, have had to look long and hard into the decision support mirror provided by these 17 queries and two update functions.

TPC-C vs TPC-D Performance Leaps

In the first two years of published results, the TPC-D 100GB Power performance metric grew by a factor of 15, from the low 200s into the 3000s. While improved hardware or larger configurations could account for a several times increase, the majority of this much larger performance rise is due to both software enhancements, and application tuning made possible by these improvements—the database.

Compare this 15-fold increase in the Power metric to the increase in TPC-C tpms during the same two-year interval. In December of '95 Tandem held the highest TPC-C performance at 20,918 tpm, and by December of '97 that position was held by Sun with a tpm of 51,871—a 2.5-fold growth, comparable to a hardware-only improvement rate. The database and application improvements TPC-D is experiencing today, that contribute to great surges in benchmark performance, these types of changes took place in the transaction world during the life-span of TPC-A and the early years of TPC-C. For this first version of TPC-D, having grown up at the same time as customer decision support applications, the database-tuning phase is still dominating performance change.

Optimizer-Specific Challenges

The biggest challenge facing database vendors when they first experiment with TPC-D is the query plans that are produced. TPC-D has a few straightforward queries, such as Query 1, a full table scan of the Lineitem table, performing aggregation on 8 columns and producing 4 rows in the answer set. But for the most part, the queries involve joins of from 3 up to 8

tables, and the sequence in which these joins are performed, as well as the operational activity behind the join, will be critical to performance.

The severity of the optimizer challenge can be observed in Query 9. None of the 6 tables to be joined in Query 9 have selection constraints except Part, a moderately-sized table, so it is critical that the Part table drive the join activity, and thereby reduce the impact of the larger tables' fully-participating joins. If the optimizer is inaccurate in making row count estimates in its join costing process, Query 9's plan could join all the rows of all the largest tables in the database first, and then subsequently perform a join to the Part table. If this happens, then the earlier joins will be doing 20 times or more work than is actually required to build the answer set.

Query 9 Business Question: The Product Type Profit Measure Query finds, for each nation and each year, the profit for all parts ordered in that year which contain a specified substring in their names and which were filled by a supplier in that nation. The profit is defined as the sum of [(L_EXTENDEDPRICE*(1-L_DISCOUNT)) - (PS_SUPPLYCOST * L_QUANTITY)] for all lineitems describing parts in the specified line. The query lists the nations in ascending alphabetical order and, for each nation, the year and profit in descending order by year.

```
SELECT
 NATION, YEAR, SUM(AMOUNT) AS SUM_PROFIT
 FROM (SELECT
 N_NAME AS NATION, EXTRACT (YEAR FROM O_ORDERDATE) AS YEAR,
 L_EXTENDEDPRICE*(1-L_DISCOUNT)-PS_SUPPLYCOST*L_QUANTITY
 AS AMOUNT
  FROM PART, SUPPLIER, LINEITEM, PARTSUPP, ORDER, NATION
  WHERE
  S_SUPPKEY = L_SUPPKEY
  AND PS_SUPPKEY = L_SUPPKEY
  AND PS_PARTKEY = L_PARTKEY
  AND P_PARTKEY = L_PARTKEY
  AND O_ORDERKEY = L_ORDERKEY
  AND S_NATIONKEY = N_NATIONKEY
  AND P_NAME LIKE '%green%'
  GROUP BY NATION, YEAR
  ORDER BY NATION, YEAR DESC;
```

Many of the database enrichments are in-code improvements you can't see, such as better optimizer costing algorithms, more efficient join techniques, faster sorts or more intelligent methods of applying selection criteria. Product improvements that can be observed in the published Full Disclosure Reports show an introduction of ANSI-compliant SQL functionality driving better performance, as well as new table partitioning schemes, or expanded indexing capabilities.

The Physical vs Logical Database Design

Understanding the temptation to violate the spirit of a benchmark with excessive tuning, the committee clearly defined the limits of tuning efforts by holding the logical definitions to the specification model and taking a flexible, open attitude toward physical tuning. On the logical side, the database objects must exist as defined, and be discreet and independent. But at the physical level, performance may may be enhanced by a variety of secondary structures, may include clustering or replication, as long as any additional operations and objects are

transparent to the queries. The periodic update requirement and strict ACID property testing are a guarantee that all secondary structures will support data integrity at the logical level.

Costing Index Use

While benchmarks such as Set Query mandate index choices [O'Neil, 1993], increasing comparability by standardizing on access methods, the TPC-D rules give complete freedom to vendors on which indexes to build. As a method of inhibiting overly-aggressive index use, TPC-D put a cost on the use of index structures, by three methods:

1) Load time, which includes the time to load data, to build indexes and to collect statistics, is a secondary metric. The more indexes that need to be built, the longer this measurement interval becomes.
2) The primary performance metrics include the time for performing the two update functions, UF1 and UF2. Elapsed time for both will increase due to index maintenance overhead for any indexes carried on the affected tables.
3) The disk storage ratio, a secondary metric that shows the ratio of total disk in the system to the scale factor (raw data volume), will increase, due to the additional space requirements of the secondary structures, as will the total system cost.

IV. Benchmark Compromises: Balancing the Real and the Executable

Drawing up a list of what to include in a benchmark is the easy part, it's what to leave out and what to bargain over that is difficult. In TPC-D, concious benchmark compromises were introduced to allow simulation of complex activities in a manner that are both executable and understandable, but also to find a middle ground among mutually-exclusive viewpoints. All the compromises were not successful in terms of contributing to a stronger benchmark, but every one of them played an instrumental role in achieving consensus. The compromises made within TPC-D are at the core of its democratic and vendor-neutral nature.

Benchmark Data

The heart and soul of a decision support is its data. So much of this data must be examined when satisfying day-to-day queries that characteristics such as column cardinality, evenness of primary key-foreign key associations, magnitude of duplicate values can all influence query response time. In order to achieve comparability across benchmark executions, a degree of predictability in the data was necessary. In TPC-D this takes two forms: Identical (as opposed to similar) data, and uniformly distributed attribute values.

The first, identical data, adds integrity to comparisons, as well as overcoming a recognized weakness of client benchmarking, the absence of control over the content of table's rows. The second, uniformly distributed values, while completely out of step with the diversity found within customer databases, exists solely to enhance benchmark reproducibility. This enforced evenness is illustrated in the TPC-D database by observing that Peru has the same number of customers as the United States, or looking at it from another direction, that each part has a transactional history with the same number of lineitems, about 30. In actual applications, the number of customers will tend to vary widely from one geographic area to another, whether city, state, or nation, and some parts will be more popular, other parts less popular.

Input Variables Combine With Even Distributions

The challenge facing the committee at this point was how to build an ad hoc workload, which tends to be fluctuating and unpredictable (and therefore unbenchmarkable), and still have a reproducible and comparable test. Placing input variables within each query, combined with the absence of skew was an effort to resolve this issue while sustaining an ad hoc flavor. With the natural peaks and valleys flattened out of TPC-D's data, input variables allow iterative executions of the same query, using differing input variable values, to touch different groupings of data and yet do the equivalent work. Two critical benefits gained by this decision were 1) Reduction of any advantage from data caching or pre-aggregation of the data; and 2) A more effective simulation of an ad hoc workload, as queries will read unbuffered data on each iteration.

As an example of this, Query 16 below has 3 columns that are associated with input variables. With the column names and the operators being consistent, these variables are intended to be randomly selected at execution time, changing each time the query is run. The variables are for one part brand (of which there are 25 distinct values), one part type (150 distinct values) and for 8 different part sizes (out of 50 distinct values):

Query 16 Business Question: The Parts/Supplier Relationship Query counts the number of suppliers who can supply parts that satisfy a particular customer's requirements. The customer is interested in parts of eight different sizes as long as they are not of a given type, not of a given brand, and not from a supplier who has had complaints registered at the Better Business Bureau. Results must be presented in descending count and ascending brand, type, and size.

```
SELECT
 P_BRAND, P_TYPE, P_SIZE, COUNT(DISTINCT PS_SUPPKEY) AS SUPPLIER_CNT
 FROM PARTSUPP, PART
 WHERE P_PARTKEY = PS_PARTKEY
 AND P_BRAND <> 'Brand#45'
 AND P_TYPE NOT LIKE 'MEDIUM POLISHED%'
 AND P_SIZE IN (49, 14, 23, 45, 19, 3, 36, 9)
 AND PS_SUPPKEY NOT IN
 ( SELECT S_SUPPKEY FROM SUPPLIER
  WHERE S_COMMENT LIKE '%Better Business Bureau%Complaints%')
  GROUP BY P_BRAND, P_TYPE, P_SIZE
  ORDER BY SUPPLIER_CNT DESC, P_BRAND, P_TYPE, P_SIZE;
```

Conformity of Data Volumes

An important contribution made by TPC-D is standardization of the definition of "volume" to mean the size of the raw data before it is placed in the data base. In order to execute a TPC-D benchmark, additional disk space will be required for such things as row overhead, index strucutures, temporary work space and the DBMS. The secondary metric, disk storage ratio, is intended to express the extent of this overhead, and has ranged in published benchmarks from a low of 2.9 (290 GB for a 100 GB scale factor) to a high of 16.8 (504 GB for a 30 GB scale factor).

Due to decision support's high sensitivity to volume, setting fixed volume points, or scale factors, became an inevitable compromise for TPC-D. By holding volume constant, comparability of results became more likely. And because some operational steps being done within the query become disproportionately more difficult as volume grows, a range of

increasing scale factors made for a more useful and relevant benchmark. Today the eight legal scale factors range from 1 GB up to 10 TB, with comparisons between platforms only permitted at the same volume.

A second, less natural compromise related to volume was voted in by the committee, having to do with how tables are populated at the higher volumes. In TPC-D all tables are grown at the same rate, with the exception of the two static lookup tables, Nation and Region. This compromise allows theorectical comparisons to be made between queries executing at different volumes, because although proportionally larger, the same balance of work is being done. Combined with the absence of skew, this uniform table growth means a given query will theoretically do 10 times more work at the 1 TB scale factor than it does at 100 GB scale factor, allowing sizeup analysis, in which a workload is increased across a constant configuration, to be performed.

Application Compromises

Several agreements were reached around design points of the application that speak specifically to increasing executability and comparability.

SQL-as-written was adopted both as a cornerstone for application comparability, and because it furthers the benchmark's ad hoc premise. Insisting that all benchmark executions use the same standard SQL, or approved variants of it, this explicitly disallows crutches such as optimizer hints or arbitrary query rewrites, and has sped up vendor compliance with ANSI-92 SQL standards. TPC-D's SQL-as-written requirement is an example of the best in industry benchmarking: Setting high standards drives product improvements.

In order to reconcile answer set size with the ad hoc model, and to prevent the formatting and return of large answer sets from dominating the reported query times, aggregations were placed into almost every TPC-D query. Because the schema was defined before this decision was made, some repetition of the same aggregate functions exist, as in the case of lineitem revenue, defined as SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT)), an aggregation that appears in 11 of the 17 queries. For variety's sake, the committee did relax control over returned rows in two cases: Query 3 asks for the first ten rows, but has a large final answer set; and Query 10 consistently returns millions of rows. The growth of rows in the final answer set for each of these queries is expressed below. Interesting to note is the close to perfect correlation between volume increase and answer set growth, proving that the absence of skew and the proportional growth of all tables supports comparability across volumes.

	Number of Rows in Final Answer Set		
	100 GB	**300 GB**	**1TB**
Query 3, the Shipping Priority Query	1,132,769	3,399,328	11,330,325
Query 10, Returned Item Reporting Query	3,892,496	11,718,271	38,615,418

A less intuitive application compromise ties the amount of updating required in the Throughput Test directly to the number of streams, or concurrent users, the benchmarker selects. For the multi-user Throughput Test the tester selects from one to any number of streams. For each stream in the Throughput Test, a pair of update functions (a mass insert and a mass delete) are required, either executed at the same time as the queries are running, or at the completion of the query streams. The motive behind increasing the update effort with

query concurrency was so that the update activity would always carry the same relative weight as the queries do during the Throughput Test, inhibiting a benchmarker from trivializing the update work by raising the level of streams.

V. TPC-D Performance Metrics

The committee's biggest challenge was determining how to associate dissimilar performance facets into one meaningful metric. The ideal decision support metric would reward shorter response times, higher volumes of data, lower load and data preparation times, and reasonable update intervals, as well as shield benchmarkers from unexpected runaway queries that could dominate the measurement interval, possibly reward higher numbers of users, and yet be totally comparable across all executions while still providing meaning, in and of itself, to a user. While the two TPC-D performance metrics only partially reach this ideal, they do break new ground by succesfully marrying some number of unlike aspects of performance in unique yet useful ways.

The Calculations

Following TPC Council "event-per-time" metric tradition popularized with transaction benchmarks, both performance metrics simulate query-per-hour rates. The most well-known of the two performance metrics, Query Processing Power D (QppD), is based on the geometric mean of the 17 query times and the two update functions. The formula automatically converts seconds into hours by factoring 3600 in the numerator:

$$\text{QppD@Size} = \frac{3600 * \text{Scale Factor}}{\sqrt[19]{q1 \times q2 \ldots \times q17 \times UF1 \times UF2}}$$

What attracted the committee to the geometric mean initially was it's equalizing and forgiving nature. Because the geometric mean is based on multiplying and taking the "n"th-root, it tends to reduce the importance of the long-running queries that is seen with the arithmetic average. The geometric mean treats equally all percent improvements in query times, regardless of the actual amount of time involved, and equally rewards similar-percentage reductions for long and short queries alike. This assured the committee that each query in the set would carry equal weight, leaving it up to the observer to determine which of the 17 were the most interesting to them.

	Original Query Time	**Halve the Shortest Query**	**Halve the Longest Query**
Query 1	9	9	4.5
Query 2	2	2	2
Query 3	5	5	5
Query 4	3	3	3
Query 5	1	0.5	1
Sum	20	19.5	15.5
Arithmetic Avg	4	3.9	3.1
Geometric Mean	3.1	**2.7** ←	**2.7** ←

The Geometric Mean Rewards Percent Improvements

Tuning a TPC-D application is like squeezing a balloon-- reducing response time for one query often results in the increase of another. Consider the case where you've cut the longest running query time in half, but a the expense of doubling the shortest one. The run times are (4.5,2 ,5, 3, 2), a combination of the second and third columns above. The arithmetic average then becomes 3.3, a strong 17.5% percent improvement, but the geometric mean is still 3.1, no improvement.

Several more obscure side-effects of the geometric mean came to the attention of the committee. One was that this average has the potential for being unnaturally influenced by very short queries when there is a wide range among query times. The closer a query moves toward a zero-second response time, especially if there is only one very short query, the larger is its effect on the geometric mean.

Because of this, the committee put a safety net into the TPC-D specification requiring adjustment of very short query times if the spread of times across all queries is very wide: If the ratio between the longest query timing interval and the shortest is greater than 1000, then all query timings which are smaller than $1/1000^{th}$ of the longest query timing must be increased. In a recent reported TPC-D Power Test, for example, Query 13 executed in 1.6 seconds. But since, in the same Power Test, Query 9's timing interval was 4530.9 seconds, over 1000 times longer, Query 13's reported time in the Power Test was increased to 4.5 seconds (1/1000th of the longest query's time).

A second side-effect was a loop-hole opened by the geometric mean when used with multiple streams. This led to the decision to limit the geometric mean to the single-user test, and re-inforced in the decision to have two performance metrics. The concern was that if the geometric mean were applied to multiple streams, because it rewards small values disproportionately, the possibility of achieving a higher score exists if the benchmarker is able to start all queries concurrently, but serialize their actual execution, ending up with some very long and some very short query times.

The second performance metric, known as Query Throughput D (QthD), is based on the arithmetic average, as shown below:

$$\text{QthD@Size} = \frac{(\#\text{Streams} \times 17 \times 3600)}{\text{Throughput Interval}} \times \text{Scale Factor}$$

Both performance metrics were put on equal footing by combining them to feed into price-performance. This geometric mean of QppD and QthD, as it appears in the denominator below, is also known as the QphD, or the Composite Query per Hour rate.

$$\text{Price-Performance} = \frac{\text{Total 5-year Price}}{\sqrt{(\text{QppD@Size} \times \text{QthD@Size})}}$$

Geometric Mean Revisited

Use of the geometric mean has received a mixed review. On the one hand, SPEC (Standard Performance Evaluation Corporation) had previously selected the geometric mean as a

composite benchmark metric for SPEC benchmarks [Dixit, 1993], so it was in the realm of current practice.

However, industry analysts such as Stephen Brobst have suggested that the geometric mean leads to "Some perverse dynamics in optimizing benchmark results...In production environments, customers are more likely to focus on reducing total execution time by optimizing longer-running queries than shaving seconds off short-running ones." [Brobst, 1997] In addition, marketing organizations have over-emphasized the Power metric, both because it is the first metric listed, and because TPC-D Power always produces a larger number than Throughput. Power comes out as a larger number in every vendor's benchmark because the geometric mean is mathematically guaranteed to be a smaller number than the arithmetic average, which is more influenced by long queries. This translates to higher query-per-hour rates in Power than in Throughput.

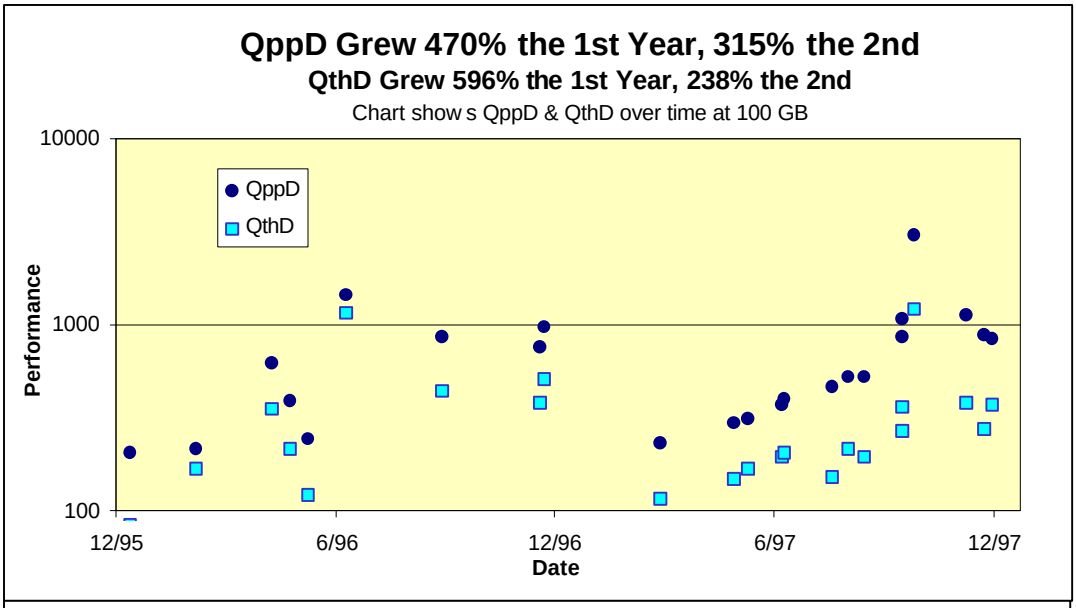
Consolidating Performance Characteristics

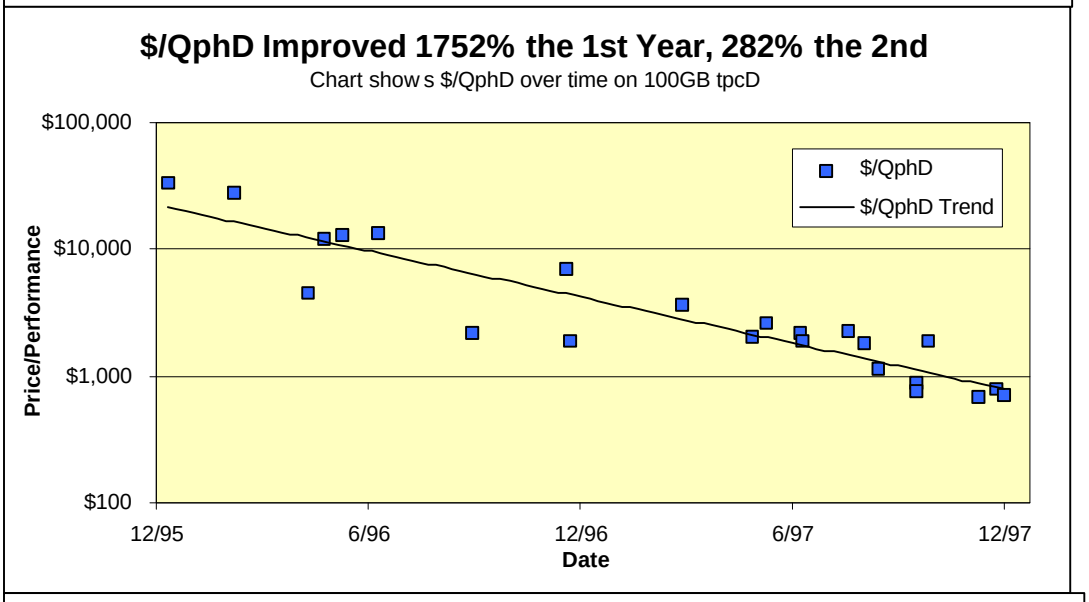
In spite of these controversies, these two metrics have successfully combined disparate aspects of performance in fresh ways. By including scale factor in the formula, both metric calculations normalize for volume, making it possible to cross-compare different volumes on the same platform. Cross-volume comparisons are discouraged by TPC Council, and are only useful when applied against consistent hardware and software, everything remaining constant except volume. In one such case in 1996, a sequence of TPC-D results at 100 GB and at 300 GB on the same platform produced QppD results less than 5% apart, validating this aspect of the metric. This is explainable because the slower query times at the higher volumes are compensated by the formula carrying a proportionally larger scale factor. Combined with the absence of skew in the data and the even table growth, the metric makeup supports both scale-up and speed-up observations.

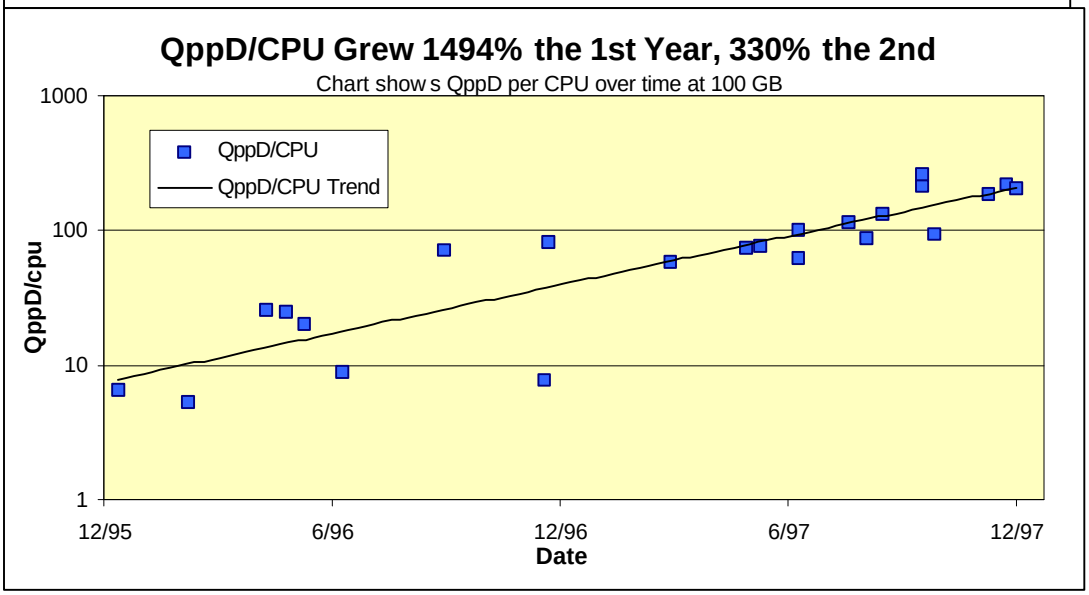
The Throughput formula shows another novel consolidation. By factoring in the level of query concurrency, the Throughput metric provides an opportunity for vendors to show either a constant or improved result with the addition of more streams, depending on the degree of benefit the product derives from overlapping queries. Vendors such as IBM, NCR, and Sun who have published TPC-D benchmarks with multiple streams, have shown that the Throughput metric can be improved as streams are added. In translating the level of concurrency used in a TPC-D benchmark to the real world, it is important to keep in mind that a query stream in TPC-D does not include think time. Each TPC-D stream executed reflects a higher degree of effort, perhaps 5 or 10 times more, than one decision support user would generate in the real world.

Version 1 Growth of Performance and Price-Performance

The following charts illustrate the marked improvements in performance, price-performance and Power (QppD) per CPU for the 100 GB scale factor only. Results are charted over the first two years of TPC-D benchmark publications, December 1995 to December 1997.


QppD Grew 470% the 1st Year, 315% the 2nd
QthD Grew 596% the 1st Year, 238% the 2nd
Chart shows QppD & QthD over time at 100 GB
10000
1000
100
Performance
QppD
QthD
12/95
6/96
12/96
6/97
12/97
Date


$/QphD Improved 1752% the 1st Year, 282% the 2nd
Chart shows $/QphD over time on 100GB tpcD
$100,000
$10,000
$1,000
$100
Price/Performance
$/QphD
$/QphD Trend
12/95
6/96
12/96
6/97
12/97
Date


QppD/CPU Grew 1494% the 1st Year, 330% the 2nd
Chart shows QppD per CPU over time at 100 GB
1000
100
10
1
QppD/cpu
QppD/CPU
QppD/CPU Trend
12/95
6/96
12/96
6/97
12/97
Date

VI. Benchmark Options Both Widen and Weaken TPC-D

Choices were placed in the benchmark to broaden the usefulness of results and encourage more vendors to participate. In almost all cases, comparability of results was either lost or reduced, and in some cases an easy way out of the tough or undesirable portions of the test was unintentionally provided.

A Choice in the Number of Streams in the Throughput Test

After establishing the Throughput metric with the number of streams in the calculation, it became clear to the development committee that this formula would apply to all numbers of streams equally, and would allow fair comparisons between benchmarks using different numbers of streams. Making the number of streams an option resolved a tough open issue: How to mandate concurrency without making the benchmark too difficult to run. Expanding this choice to include just one user in the Throughput Test was a practical decision, seen as an alternative that could reduce the cost and time of benchmarking, particularly at the higher volumes. According to benchmark rules, if one stream is chosen, a vendor is not required to perform the Throughput Test, but can feed the results achieved in the Power Test into the Throughput formula. A vendor who bypasses the second test in this manner reports "zero streams" in the benchmark documentation.

What was a surprise to the committee was how popular the zero stream TPC-D executions have become, and how few vendors have attempted the Throughput Test. Out of 15 published vendors, only 3 have published multi-user benchmarks to date. Allowing single-user tests to be compared against multi-user executions muddies the waters of product comparison and also creates confusion, especially when a vendor reports the metric but doesn't actually perform the test.

Database Maintenance Operation: Evolve or Reset

In order to prove that benchmark performance is repeatable, a complete re-run of the entire benchmark is required, and performance must be within 5% between two consecutive runs, or 10% with 3 consecutive runs. Because some number of update functions have taken place in Run 1, Run 2 may execute against data that has been disorganized to some degree. By choosing a database maintenance operation of "reset" a tester may reverse all updates between the two runs. On the other hand, selecting the "evolve" option means doing nothing, and preceeding to the second run as is. Both choices result in the Run 2 facing some level of data disorganization, as no reorganization of data blocks or index structures between runs is permitted. Because you are required to publish TPC-D using the lower of the two sets of metrics, results from one vendor may be against mature data that has experienced several or many update functions, and those results could be compared to another vendor who chose to back off updates by resetting the database.

Update Functions With or Without the Queries

Benchmarkers who choose to perform the Throughput Test have to make this decision: Execute the updates concurrently with the queries, or hold the updates back until the query streams have completed and execute them serially at the end. Using the example of a 10-stream TPC-D benchmark, 20 update functions will be required in the Throughput Test.

These update functions may be run at the same time as the query streams, or may execute end-to-end after the queries have completed. This choice came into being because the committee had a view of the future in which concurrent update and query processing would be the norm in most customer systems. Vendors publishing TPC-D benchmarks, they believed at that time, would want to demonstrate their concurrent update and queries capabilities within TPC-D benchmarks, and would tend to select the first choice. However, results to date have not shown that tendency. There are no results today with Throughput Test queries and updates running at the same time that might allow a useful assessment of this option, and its effect on comparability.

Scale Factor

Unlike the other options, giving the benchmarker a choice on the volume of data is the flexibility point within TPC-D that has most helped to make the benchmark more comparable and results easier to understand. Because of the upsurge of reported results, clustering of results, particularily at the 100 GB and 300 GB levels, has brought increased focus and understanding using the primary metrics. The existence of multiple scale factors actually has created many TPC-D benchmarks, each with their own winners and patterns of results, adding both depth and clarity to accumulated results.

VII. Where TPC-D Missed the Mark

While the vast majority of decisions that formed TPC-D were based on solid theory with facts behind them, some ended up as best guesses. While this hybrid decision-making was unavoidable, some of those conjectures missed the mark. Looking backward to identify the ones that could have been better is a way to influence stronger decisions in the future.

Lack of Compulsory Multi-User Testing

According to industry analyst Richard Winter, whose writings have been very supportive of TPC-D, "The current TPC-D specification has a loophole in regard to multiuser processing that is widely exploited. As a result, most of those magnificently posted results that we see literally say *nothing* about the ability of the database engine to deal efficiently with multiple concurrent users. This flaw is a serious problem." [Winter, 1997]. The vast majority of data warehouses support many users. A one-user benchmark can easily become a greenhouse environment where opportunities exist for artificially elevating performance. This is a step backwards from the real world that has been a mantra within the TPC-D committee.

All high-volume TPC-D results to date have executed using parallel databases, and this technology is particularly vulnerable to the challenges of multiple users. Designed to generously apply parallelism to the needs of just one query, parallel functions can easily saturate the system, and in so doing, can become difficult to reign in and coordinate, producing their own severe type of conflict. Single-user-only benchmarks in general can mask a multitude of product weaknesses, such as poor flow control, unevenness of execution, or inability to manage contention and prioritization.

Allowing a Reset of Updates Between Benchmark Runs

Good solid thinking went into designing the UF1 and UF2 activity, where updates are spread randomly across the database taking advantage of intentional gaps in the key values left by the data generation utility. Deletes are applied to different rows than were inserted, foiling

attempts to bunch up inserts so as to delete them as a partition drop. But the database maintenance operation was not equally well thought out.

Allowing benchmarkers to reverse updates between repeatability runs dilutes the role of the periodic updates, and works against the 7 x 24 data model, where a live database would require users to forge ahead in spite of data maturity. Neither does this option make the benchmark easier to understand or compare, and it can add complexity. Since the rules of the benchmark do not support re-packing the data blocks, or anything beyond a simple delete and insert reversal, only a portion of the disorganization effect is actually reversed using reset.

Tying the Amount of Updating to the Number of Streams

Although a necessary benchmark artifact, as shown earlier, increasing update activity at the same rate as the number of streams are added in the Throughput Test defies the typical data warehouse today, where the amount of data that needs to be updated will be independent of the number of users using the system. By strictly associating these two factors in TPC-D, it tends to distort the value of the Throughput Test, rendering it less useful outside the benchmark.

Second, being faced with increasing numbers of UF1 and UF2 pairs may discourage higher numbers of streams, by making repeatability more of a challenge. Even if the database maintenance option of reset is chosen and updates are reversed between the repeatability runs, data fragmentation and block splits will remain. The more updating that is done, the more disorganized the database will be for the second and subsequent runs.

And third, for some platforms updates will have an increasing effect on the Throughput metric as more streams are executed. Because query streams are run concurrently, it is possible there will be some reduction in query times as streams increase because of potential overlap or sharing in resource usage. An example for one system can be observed in the NCR 10-stream 100 GB TPC-D published in September, 1997, where the timing interval for one stream of queries in the Power Test was 3181 seconds, and for 10 streams running concurrently in the Throughput Test was 24,651 seconds. Because there was additional capacity that could be utilized, the query portion of the Throughput Test took 7.75 times longer (24651 / 3181 = 7.75) but did 10 times the work. The updates must be executed serially, one UF1 or UF2 event at a time, with strict rules against batching them up. If query times can be condensed, and update times cannot, this opens the door for update response times to carry more weight as streams are added.

It was envisioned that technology and demands on the industry would push most vendors into executing TPC-D while demonstrating concurrent updates and queries on the Throughput Test. If this had materialized, then the motive of balancing the update I/O with the query I/O as streams are added would have been legitimate. But this has not happened. This is a case of a benchmark option (update with queries vs update separately) defeating the purpose of what might have been a reasonable compromise (tying updates to the number streams).

TPC-D Falls Short of Ad Hoc

The TPC-D benchmark specification is overly-ambitious when it labels the 17 queries ad hoc. In fact, as the benchmark life-cycle progresses, and TPC-D benchmarks exhaust the database and application tuning opportunities, benchmark iterations will become even less natural, and

less ad hoc. This move away from ad hoc is aggravated because there are so few queries in TPC-D compared to a healthy decision support application. Index structures and partitioning schemes used in published benchmarks today are tightly designed around the TPC-D query set, with a trend towards precision-tuning the overly-familiar.

When TPC-D application and query tuning has intensified to the point that all the fat has been removed from each query plan and the only rows touched are the rows actually required for the answer set, TPC-D performance growth will slow down to the rhythm of hardware-driven change, as TPC-C has already demonstrated.

Query Plans are not Disclosed

When the committee voted not to disclose query plans, they further cut off TPC-D performance from the real world and caused a disassociation of query times from the underlying hardware platform. Without seeing the sequence of steps the optimizer has chosen, it is not possible to understand why or where a query has slowed down, or sped up. For example, only by substantiating that two vendors executed Query 6 as a full table scan of the Lineitem table, can interesting comparisons of disk throughput between both platforms can be made. Understanding variances in resource use is not possible unless plans are available.

Including query plans in the benchmark documentation would be invaluable for making these types of observations:

- How plans vary between an SMP and an MPP platform
- The extent that query plans scale (remain consistent) as volume grows
- How closely matched these plans are with plans a client is experiencing
- The variety and usage of optimizer choices

VIII. Concluding Remarks

TPC-D has done more than provide an instrument for decision support comparisons. The benchmark has been an educational tool within the industry, and more important, an impetus to many serious product improvements. Clients have migrated to the TPC-D application for their own in-house benchmarks, both because of its convincing real-life queries and its vendor neutrality. Vendors have latched on to the TPC-D application for demos at trade fairs and for proof-of-concept demonstrations at client sites.

As TPC-D was born into and influenced a world of change from which decision support emerged, it must now continue to drive change, within the vendor world and within its own self-definition. Because decision support is grounded in the ad hoc, the spontaneous, the life cycle of a decision support benchmark is by necessity short, and requires frequent rejuvenation. After two years of use and analysis, clear roadsigns for TPC-D Version 2 are in place. The TPC Council now faces the challenge and has the exciting opportunity of building the second version of TPC-D into a stronger and more relevant benchmark.

The TPC-D committee has already made some steps in that direction, with decisions due to take effect in 1998 aimed at strengthening the current benchmark while work on a more substantial rewrite is underway. Some of the short-term improvements expected in TPC-D

Version 2 include a mandatory multi-user Throughput Test, and replacement of the two primary performance metrics (QppD and QthD) with a single performance metric (QphD). Both QppD (Power) and QthD (Throughput) will have an equal impact on the calculation of the QphD, and will be reported, but will be relegated to secondary metric status.

IX. Acknowledgements

Many people, including over 50 contributers from 14 vendor companies, derserve credit for their contribution to TPC-D Version 1 over the five years of its developement. It was truly a group effort. But special appreciation is due longterm contributers Alain Crolotte (representing NCR), Suzanne Englert (representing Tandem), Jean-Jacques Gillemaud (representing Bull), Brian LaPlante (representing Unisys), Indira Patel (representing Pyramid), Bernie Schiefer (representing IBM), Frank Stephens (representing Unisys), and Jack Stephens (representing Informix), without whose extra efforts and dedication there would be no benchmark today.

A special word of thanks goes to Francois Raab for his work in creating the TPC-D Specification document and helping us to organize our thoughts on paper, and to Kim Shanley, Chief Operating Officer of the TPC Council, for his support and unwavering conviction that TPC-D would be important.

And finally, my sincere thanks go to Jack Stephens for his insightful review of this chapter, and for helping me think through the opinions I have expressed in this article.

X. References

Transaction Processing Performance Council, (1997). *TPC Benchmark D Standard Specification, Revision 1.2.3.*

O'Neil, P.E.(1993). The Set Query Benchmark, In J. Gray (Ed.), *The Benchmark Handbook, Second Edition*. Morgan Kaufmann.

Dixit, K.M. (1993). Overview of SPEC Benchmarks, In J. Gray (Ed.), *The Benchmark Handbook, Second Edition*. Morgan Kaufmann.

Brobst, S. & Monoghan, M. (1997). TPC-D Benchmarking: The Good, The Bad, and The Ugly. *Database Programming & Design, Vol 10 No 11.* San Mateo, CA: Miller Freeman Inc.

Winter, R. (1997). Diagnosis for TPC-D. *Teradata Review, Winter 1997.* San Mateo, CA: Miller Freeman Inc.