TPC Benchmark™ H
Full Disclosure Report

Kickfire™ Appliance 2400
Using MySQL Database

Submitted for Review
Report Date: May 5, 2008
TPCH Benchmark Full Disclosure Report

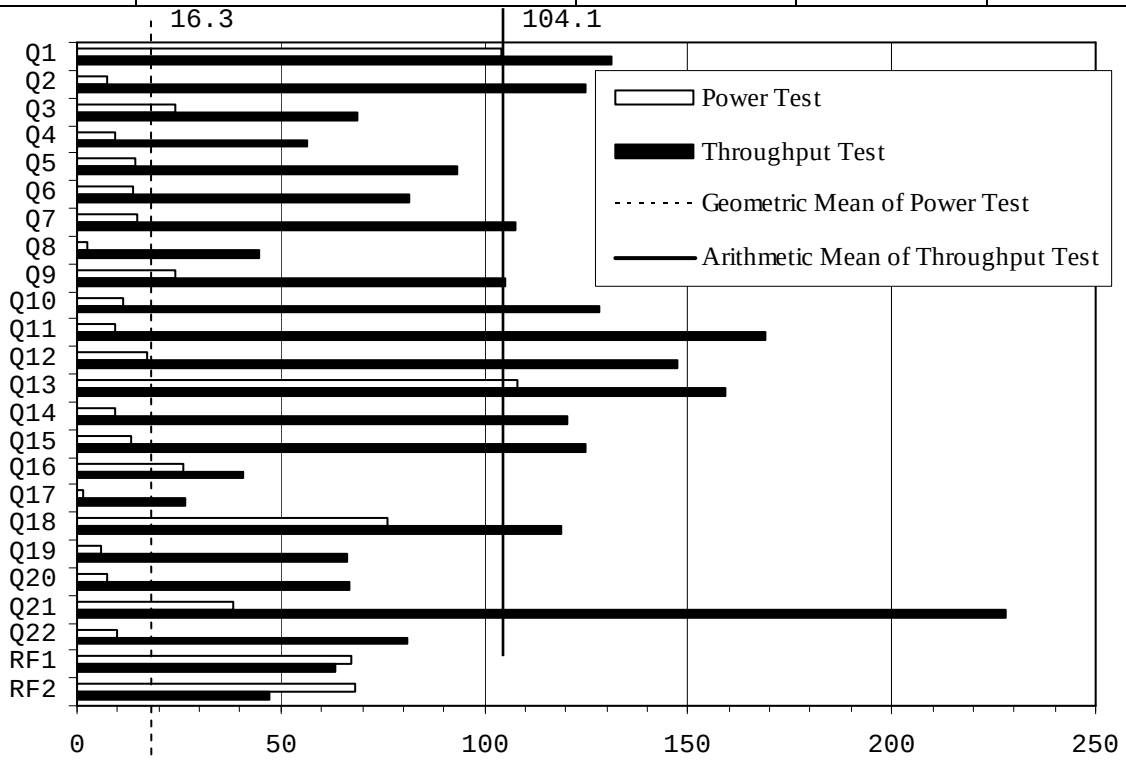
Added discount explanation note (June 30, 2008)

2055 Laurelwood Rd, Suite 150
Santa Clara, CA 95054

KICKFIRE	**Kickfire™ Appliance 2400 with MySQL Database**	**TPC-H Rev. 2.6.2** **Report Date:** May 5, 2008

Total System Cost	Composite Query Per Hour Metric	Price/Performance
$48,790	**54,895.1**	**$0.89**

Database Size	Database Manager	Operating System	Other Software	Availability Date
300GB	**MySQL5.1 with Kickfire database**	**CentOS 5.0**	**None**	**October 14, 2008**



Database Load Time 02:40:54	Load Included Backup: N	Total Data Storage / Database Size = 1.81

RAID (Base tables only): N	RAID (Base tables and auxiliary data structures): N	RAID (All): Y

System Configuration:

1 x Kickfire Base Server Module, with:

2 Intel Xeon E5440 2.83GHz processors (each is 1 chip, 4 cores, 4 threads)

16 GB RAM

8 x 73 GB (15K RPM) internal SAS disks

1 x Kickfire Query Processing Module, with 128 GB RAM

Total Storage: 544 GB

(1 GB = 2^{30} bytes)

	Kickfire™ Appliance 2400 with MySQL Database	TPC-H Rev. 2.6.2 **Report Date:** May 5, 2008

Description	Part No.	Src	Unit Price	Qty	Ext. Price	3 Yr. Main.
Kickfire Database Appliance Series 2400 Includes: BSM: 2-CPUs, 16 GB RAM, 8 X 73GB 15K RPM SAS Drives QPM: 128GB RAM, SQL Chip MySQLEnterprise v5.1, KickfireDB v1.0, KickFire Utilities	K2400-0001	1	$57,400	1	$57,400	
Kickfire Discount (18%) discount K2400-0001 (includes Net 30)					($10,332)	
Kickfire 3 Year, 24X7, 4Hr. H/W Basic Support Includes: Kickfire and MySQL S/W support	KS0000-1001	1	$1,722	1		$1,722
Total ***3 Yr Cost in USD***					$47,068 $48,790	$1,722
QphH @ 300GB ***USD$/QphH @ 300GB***					54,895.1 0.89	
Source: 1 – Kickfire Inc. Note: Discount is computed based on purchase price and volume; similarly sized models will receive similar discount as quoted here. Price available by contacting Kickfire Sales at sales@kickfire.com or (408) 450-5400						

Audited by: **Lorna Livingtree of Performance Metrics Inc.**

Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specifications.

KICKFIRE	**Kickfire™ Appliance 2400 with MySQL Database**	**TPC-H Rev. 2.6.2** **Report Date:** May 5, 2008

Numerical Quantities

Measurement Results

Database Scale Factor	= 300GB
Total Data Storage/Database Size	= 1.81
Start of database load time	= 00:14:26 04/19/08
End of database load time	= 02:55:20 04/19/08
Database load time	= 02:40:54
Query streams for throughput test	= 6
TPC-H Power	= 66183.9
TPC-H Throughput	= 45531.8
TPC-H Composite Query-per-hour rating (QphH@300GB)	= 54895.1
Total System price over 3 years	= $48,790
TPC-H Price/Performance metric ($/QphH@300GB)	= $0.89 USD

Measurement Intervals

Measurement interval in Throughput test (Ts) = 3131.0 seconds

Duration of Stream Execution

Stream ID	Seed	Start Date	Start Time	End Date	End Time	Duration
Stream 0	419025520	04/19/08	07:46:04	04/19/08	07:57:25	11min:21sec
Stream 1	419025521	04/19/08	07:57:25	04/19/08	08:38:34	41min:09sec
Stream 2	419025522	04/19/08	07:57:25	04/19/08	08:36:48	39min:23sec
Stream 3	419025523	04/19/08	07:57:25	04/19/08	08:35:04	37min:39sec
Stream 4	419025524	04/19/08	07:57:25	04/19/08	08:30:26	33min:01sec
Stream 5	419025525	04/19/08	07:57:25	04/19/08	08:35:25	38min:00sec
Stream 6	419025526	04/19/08	07:57:25	04/19/08	08:37:17	39min:52sec
Refresh		04/19/08	08:38:34	04/19/08	08:49:36	11min:02sec

TPC-H Timing Intervals (in seconds)

Stream ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Stream 00	104.1	7.3	23.9	9.1	14.1	13.6	14.8	2.3	24.2	11.1	9.1	17.3
Stream 01	138.6	239	62.5	8.8	46.1	36.4	116	14.6	72.8	272.6	420.8	113.8
Stream 02	125.8	103.1	27.6	110.9	19.1	306.0	244.9	17.6	107.6	100.2	156.8	45.0
Stream 03	122.1	8.0	40.0	37.8	133.1	19.5	23.9	135.8	92.4	129.5	69.7	91.6
Stream 04	131.6	21.7	136.1	52.2	89.6	77.7	76.9	13.6	31.6	103.1	49.9	32.1
Stream 05	134.2	366.9	99.9	12.4	143.4	28.2	91.5	67.7	85.9	45.4	271.9	126.8
Stream 06	135.0	9.8	46.7	115.7	129.1	22.1	93.0	19.8	240.9	118.5	45.3	475.8
Minimum	122.1	8.0	27.6	8.8	19.1	19.5	23.9	13.6	31.6	45.4	45.3	32.1
Average	131.2	124.8	68.8	56.3	93.4	81.7	107.7	44.9	105.2	128.2	169.1	147.5
Maximum	138.6	366.9	136.1	115.7	143.4	306.0	244.9	135.8	240.9	272.6	420.8	475.8

Stream ID	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	RF1	RF2
Stream 00	108.2	9.2	13.3	25.9	1.5	76.0	5.8	7.2	38.5	9.7	67.1	68.1
Stream 01	157.7	223.3	104.0	37.3	44.9	13.2	21.6	115.7	69.3	19.7	64.8	64.5
Stream 02	189.6	180.2	32.6	44.5	6.8	75.6	69.9	33.3	198.4	167.5	64.9	42.4
Stream 03	142.8	18.3	25.9	27.8	28.8	94.1	17.5	14.9	896.1	89.9	62.6	45.3
Stream 04	199.8	217.6	84.5	45.3	33.8	154.9	197.4	115.2	96.2	20.3	61.1	41.8
Stream 05	128.4	29.3	312.2	41.1	41.8	82.8	77.6	27.6	39.6	26.1	62.9	44.7
Stream 06	137.7	52.5	189.6	49.6	1.6	172.9	15.1	93.2	66.5	161.8	62.9	44.5
Minimum	128.4	18.3	25.9	27.8	1.6	75.6	15.1	14.9	39.6	19.7	61.1	41.8
Average	159.3	120.2	124.8	40.9	26.3	118.9	66.5	66.7	227.7	80.9	63.2	47.2
Maximum	199.8	223.3	312.2	49.6	44.9	172.9	197.4	115.7	896.1	167.5	64.9	64.5

PERFORMANCE METRICS INC.
TPC Certified Auditors

April 28, 2008

Mr. Joseph Chamdani
Chief Technical Officer
Kickfire, Inc.
2055 Laurelwood Road, Suite150
Santa Clara, CA 95054

I have verified both on site and by remote the TPC Benchmark™ H for the following configuration:

Platform: KickFire Appliance 2400
Database Manager: MySQL5.1 with KickFire Database
Operating System: CentOS 5.0

CPU's	Memory	Total Disks	Qpph@ 300GB	QthH @ 300GB	QphH@300GB
2 Intel Xeon quad core @ 2.83 Ghz	16 GB	8 @ 73 GB	**66,183.9**	**45,531.8**	**54,895.1**

In my opinion, these performance results were produced in compliance with the TPC requirements for the benchmark. The following attributes of the benchmark were given special attention:

- The database tables were defined with the proper columns, layout and sizes.
- The tested database was correctly scaled and populated for 300GB using DBGEN. The version of DBGEN was 2.6.0.
- The data generated by DBGEN was successfully compared to reference data.
- The qualification database layout was identical to the tested database except for the size of the files.
- The query text was verified to use only compliant variants and minor modifications.

- The executable query text was generated by QGEN and submitted through a standard interactive interface. The version of QGEN was 2.6.0.
- The validation of the query text against the qualification database produced compliant results.
- The refresh functions were properly implemented and executed the correct number of inserts and deletes.
- The load timing was properly measured and reported.
- The execution times were correctly measured and reported.
- The performance metrics were correctly computed and reported.
- The repeatability of the measurement was verified.
- The ACID properties were successfully demonstrated and verified.
- The system pricing was checked for major components and maintenance.
- The executive summary pages of the FDR were verified for accuracy.

Auditor's Notes: None

Sincerely,

Lorna Livingtree
Auditor

Table of Contents

1. General Items

1.1. Benchmark Sponsor

A statement identifying the benchmark sponsor(s) and other participating companies must be provided.

Kickfire, Inc. is a sponsor of this TPC-H benchmark.

1.2. Parameter Settings

Settings must be provided for all customer-tunable parameters and options that have been changed from the defaults found in actual products, including but not limited to:

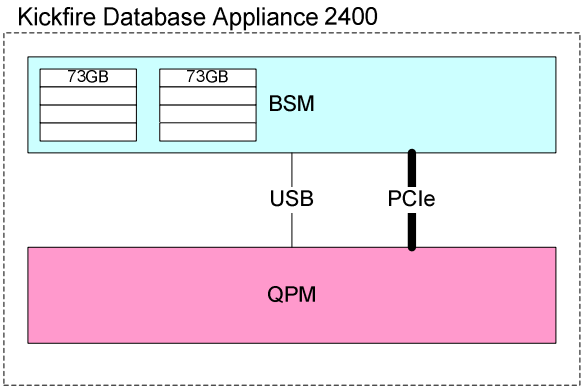
- Database Tuning Options
- Optimizer/Query execution options
- Query processing tool/language configuration parameters
- Recovery/commit options
- Consistency/locking options
- Operating system and configuration parameters
- Configuration parameters and options for any other software component incorporated into the pricing structure
- Compiler optimization options

Appendix A contains the Kickfire database and Linux parameters used in this benchmark.

1.3. Configuration Diagram

The measured and priced configuration is as follows:

1 x Kickfire Base Server Module, with:
    2 Intel Xeon E5440 2.83GHz processors (each is 1 chip, 4 cores, 4 threads)
    16 GB RAM
    8 x 73 GB (15K RPM) internal SAS disks
1 x Kickfire Query Processing Module, with 128 GB RAM



2. Clause 1 Logical Database Design

2.1. Database Definition Statements

Listings must be provided for all table definition statements and all other statements used to set up the test and qualification databases.

Appendix B contains the programs and scripts that create and analyze the tables for the TPC-H database.

2.2. Physical Organization

The physical organization of tables and indices within the test and qualification databases must be disclosed. If the column ordering of any table is different from that specified in Clause 1.4, it must be noted.

No record clustering or index clustering was used. Column ordering of the tables differs from that specified in clause 1.4. Appendix B contains the DDLs for the table definitions.

2.3. Horizontal Partitioning

Horizontal partitioning of tables and rows in the test and qualification databases (see Clause 1.5.4) must be disclosed.

No horizontal partitioning was used.

2.4. Replication

Any replication of physical objects must be disclosed and must conform to the requirements of Clause 1.5.6.

No form of data replication was used.

3. Clause 2 Queries and Refresh Functions

3.1. Query Language

The query language used to implement the queries must be identified.

SQL as implemented by MySQL was the query language used to implement all queries.

3.2. Verifying Method for Random Number Generation

The method of verification for the random number generation must be described unless the supplied DBGEN and QGEN were used.

TPC supplied versions 2.6.0 of DBGEN and QGEN were used for this TPC-H benchmark.

3.3. Generating Values for Substitution Parameters

The method used to generate values for substitution parameters must be disclosed. If QGEN is not used for this purpose, then the source code of any non-commercial tool used must be disclosed. If QGEN is used, the version number, release number, modification number, and patch level of QGEN must be disclosed.

The supplied QGEN version 2.6.0 was used to generate the substitution parameters.

3.4. Query Text and Output Data from Qualification Database

The executable query text used for query validation must be disclosed along with the corresponding output data generated during the execution of the query text against the qualification database. If minor modifications (see Clause 2.2.3) have been applied to any functional query definitions or approved variants in order to obtain executable query text, these modifications must be disclosed and justified. The justification for a particular minor query modification can apply collectively to all queries for which it has been used. The output data for the power and throughput tests must be made available electronically upon request.

Appendix C contains the query text and query output. The standard queries were used throughout with the following modifications:

- Q1, Remove interval precision "day (3)" to "day"

```
select
        l_returnflag,
        l_linestatus,
        sum(l_quantity) as sum_qty,
        sum(l_extendedprice) as sum_base_price,
        sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
        sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
        avg(l_quantity) as avg_qty,
        avg(l_extendedprice) as avg_price,
        avg(l_discount) as avg_disc,
        count(*) as count_order
from
        lineitem
where
        l_shipdate <= date '1998-12-01' - interval ':1' day (3)
group by
        l_returnflag,
```

```
        l_linestatus
order by
        l_returnflag,
        l_linestatus;
```

- Q13, move column alias to column projection

```
select
        c_count,
        count(*) as custdist
from
        (
                select
                        c_custkey as c_custkey
                        count(o_orderkey) as c_count
                from
                        customer left outer join orders on
                                c_custkey = o_custkey
                                and o_comment not like '%:1%:2%'
                group by
                        c_custkey
        ) as c_orders (c_custkey, c_count)
group by
        c_count
order by
        custdist desc,
 c_count desc;
```

- Use LIMIT where for the below queries which has limit in number of rows returned
  - 10.sql::n 20
  - 18.sql::n 100
  - 21.sql::n 100
  - 2.sql::n 100
  - 3.sql::n 10

3.5. Query Substitution Parameters and Seeds Used

The query substitution parameters used for all performance tests must be disclosed in tabular format, along with the seeds used to generate these parameters.

Appendix D contains the seed and query substitution parameters.

3.6. Query Isolation Level

The isolation level used to run the queries must be disclosed. If the isolation level does not map closely to the levels defined in Clause 3.4, additional descriptive detail must be provided.

The queries and transactions were run with isolation level 3.

3.7. Source Code of Refresh Functions

The details of how the refresh functions were implemented must be disclosed (including source code of any non-commercial program used).

Appendix B contains the source code for the refresh functions.

4. Clause 3 Database System Properties

4.1. ACID Properties

The ACID (Atomicity, Consistency, Isolation and Durability) properties of transaction processing systems must be supported by the system under test during the timed portion of this benchmark. Since TPC-H is not a transaction processing benchmark, the ACID properties must be evaluated outside the timed portion of the test.

Source code for the ACID tests is included in Appendix B.

4.2. Atomicity

The system under test must guarantee that transactions are atomic; the system will either perform all individual operations on the data, or will assure that no partially-completed operations leave any effects on the data.

4.2.1. Completed Transaction

Perform the ACID Transaction for a randomly selected set of input data and verify that the appropriate rows have been changed in the ORDERS, LINEITEM, and HISTORY tables.

1. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the order key from step 1.
3. The ACID Transaction committed.
4. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for the same order key. It was verified that the appropriate rows had been changed.

4.2.2. Aborted Transaction

Perform the ACID Transaction for a randomly selected set of input data, substituting a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the appropriate rows have not been changed in the ORDERS, LINEITEM, and HISTORY tables.

1. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for a randomly selected order key.
2. The ACID Transaction was performed using the order key from step 1. The transaction was stopped prior to the commit.
3. The ACID Transaction was ROLLED BACK.
4. The total price from the ORDERS table and the extended price from the LINEITEM table were retrieved for the same order key. It was verified that the appropriate rows had not been changed.

4.3. Consistency

Consistency is the property of the application that requires any execution of transactions to take the database from one consistent state to another.

4.3.1. Consistency Test

Verify that ORDERS and LINEITEM tables are initially consistent, submit the prescribed number of ACID Transactions with randomly selected input parameters, and re-verify the consistency of the ORDERS and LINEITEM.

1. The consistency of the ORDERS and LINEITEM tables was verified based on a sample of order keys.
2. 100 ACID Transactions were submitted by each of 7 execution streams.
3. The consistency of the ORDERS and LINEITEM tables was re-verified.

4.4. Isolation

Operations of concurrent transactions must yield results which are indistinguishable from the results which would be obtained by forcing each transaction to be serially executed to completion in the proper order.

4.4.1. Read-Write Conflict with Commit

Demonstrate isolation for the read-write conflict of a read-write transaction and a read-only transaction when the read-write transaction is committed.

1. An ACID Transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID Transaction was suspended prior to COMMIT.
2. An ACID Query was started for the same O_KEY used in step 1. After a thirty-second wait, it had not completed, implying that it was blocked by the suspended ACID Transaction.
3. The ACID Transaction was resumed and COMMITTED.
4. The ACID Query completed. It returned the data as committed by the ACID Transaction.

4.4.2. Read-Write Conflict with Rollback

Demonstrate isolation for the read-write conflict of a read-write transaction and a read-only transaction when the read-write transaction is rolled back.

1. An ACID Transaction was started for a randomly selected O_KEY, L_KEY, and DELTA. The ACID Transaction was suspended prior to ROLLBACK.
2. An ACID Query was started for the same O_KEY used in step 1. After a thirty-second wait, it had not completed, implying that it was blocked by the suspended ACID Transaction..
3. The ACID Transaction was ROLLED BACK.
4. The ACID Query completed, and did not see the uncommitted changes made by the ACID Transaction.

4.4.3. Write-Write Conflict with Commit

Demonstrate isolation for the write-write conflict of two update transactions when the first transaction is committed.

1. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. T1 was suspended prior to COMMIT.
2. Another ACID Transaction, T2, was started using the same O_KEY and L_KEY and a randomly selected DELTA.
3. T2 waited thirty seconds.
4. T1 was allowed to COMMIT and T2 completed.
5. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE +

   (DELTA1*(T1.L_EXTENDEDPRICE/T1.L_QUANTITY))

4.4.4. Write-Write Conflict with Rollback

Demonstrate isolation for the write-write conflict of two update transactions when the first transaction is rolled back.

1. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. T1 was suspended prior to COMMIT.
2. Another ACID Transaction, T2, was started using the same O_KEY and L_KEY and a randomly selected DELTA.
3. T2 waited thirty seconds.
4. T1 was allowed to ROLLBACK and T2 completed.
5. It was verified that T2.L_EXTENDEDPRICE = T1.L_EXTENDEDPRICE.

4.4.5. Concurrent Progress of Read and Write Transactions

Demonstrate the ability of read and write transactions affecting different database tables to make progress concurrently.

1. An ACID Transaction, T1, was started for a randomly selected O_KEY, L_KEY, and DELTA. T1 was suspended prior to ROLLBACK.
2. Another Transaction, T2, was started which did the following:
3. For random values of PS_PARTKEY and PS_SUPPKEY, all columns of the PARTSUPP table for which PS_PARTKEY and PS_SUPPKEY are equal, are returned.
4. T2 completed.
5. T1 was allowed to COMMIT.
6. It was verified that appropriate rows in ORDERS, LINEITEM and HISTORY tables were changed.

4.4.6. Read-Only Query Conflict with Update Transaction

Demonstrate that the continuous submission of arbitrary (read-only) queries against one or more tables of the database does not indefinitely delay update transactions affecting those tables from making progress.

1. A transaction, T1, executing Query 1against the qualification database, was started with its [delta] substitution parameter set to 0.
2. An ACID transaction T2, was started for a randomly selected O_KEY, L_KEY and DELTA.
3. Once T2 had finished its initial two queries, a transaction T3, identical to T1 except that its [delta] substitution parameter was set to 1, was started.
4. It was verified that T1 had not yet completed.
5. Transaction T1 completed executing iso6-Query.
6. T2 was suspended for thirty seconds prior to committing. At the end of this time, T3 had not yet completed.
7. T2 was allowed to commit and T3 completed.
8. It was verified that T2 had changed the appropriate rows in ORDERS, LINEITEM and HISTORY.

4.5. Durability

The SUT must guarantee durability: the ability to preserve the effects of committed transactions and insure database consistency after recovery from any one of the failures listed in Clause 3.5.3.

4.5.1. Failure of a Durable Medium

Guarantee the database and committed updates are preserved across a permanent irrecoverable failure of any single durable medium containing TPC-H database tables or recovery log tables.

All disks containing TPC-H tables are housed on a RAID5 volume striped across multiple physical disks. A permanent irrecoverable failure of a single durable medium was simulated by removing one of these disks from the server. The system sent an alert e-mail to inform the administrator of the failure, and continued running without interruption, as is to be expected in the failure of one disk in a RAID5 volume.

4.5.2. System Crash

Guarantee the database and committed updates are preserved across an instantaneous interruption (system crash/system hang) in processing which requires the system to reboot to recover.

A system crash was simulated by cutting off power to both the [QPM? Fill in terminology] and the Intel server simultaneously. When power was restored, the Intel server automatically rebooted. The [QPM?] was manually re-initialized and the database server was restarted. The database was automatically recovered. The durability success file and the HISTORY table were compared successfully, and the database was confirmed to be consistent as in 4.3.1.

4.5.3. Memory Failure

Guarantee the database and committed updates are preserved across failure of all or part of memory (loss of contents).

The failure was tested as described in 4.5.2. As power failure causes a loss of memory contents, this test was sufficient to test failure of both QPM and base server memory.

5. Clause 4 Scaling and Database Population

5.1. Cardinality of Tables

The cardinality (i.e., the number of rows) of each table of the test database, as it existed at the completion of the database load (see clause 4.2.5) must be disclosed.

Table	Rows
Lineitem	1799989091
Orders	450000000
Partsupp	240000000
Part	60000000
Customer	45000000
Supplier	3000000
Nation	25
Region	5

5.2. Distribution of Tables and Logs Across Media

The distribution of tables and logs across all media must be explicitly described.

- All tables were stored on 1 RAID 5 volume per server. The volume was constructed using the system RAID Controller.

The following table shows all the disk slices.

Partition	Use	Size
/dev/sda1	Boot partition	99M
/dev/sda2	Linux OS + swap space	22G
/dev/sda3	Data (raw partition)	411G
/dev/sda4	Log	36G

5.3. Database partition/replication mapping

The mapping of database partitions/replications must be explicitly described.

No data partitioning was used.

No data replication was used.

5.4. RAID Feature

Implementations may use some form of RAID to ensure high availability. If used for data, auxiliary storage (e.g. indexes) or temporary space, the level of RAID must be disclosed for each device.

RAID 5 was used for all base tables, auxiliary data structures and the recovery logs.

5.5. Modifications to the DBGEN

Any modifications to the DBGEN (see Clause 4.2.1) source code must be disclosed. In the event that a program other than DBGEN was used to populate the database, it must be disclosed in its entirety.

DBGEN version 2.6.0 was used to generate the database population for this benchmark. DBGEN was altered to change the table column order and generate data in fixed width format. Changes made to DBGEN are listed in appendix F.

5.6. Database Load Time

The database load time for the test database (see clause 4.3) must be disclosed.

The database load time was 02:40:54

5.7. Data Storage Ratio

The data storage ratio must be disclosed. It is computed as the ratio between the total amount of priced disk space, and the chosen test database size as defined in Clause 4.1.3.

The data storage ratio is computed from the following information:

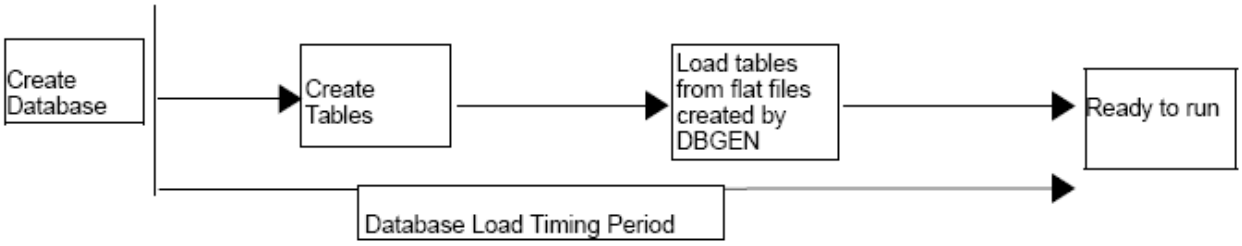
Disk Type	**# of Disks**	**GB* per disk**	**Total Disk Space in GB****
Internal	8	73	544
		Total Space	544
		Data Storage Ratio	1.81

** Disk manufacturer definition of one GB is 10^9 bytes*

***In this calculation one GB is defined as 2^30 bytes*

5.8. Database Load Mechanism Details and Illustration

The details of the database load must be described, including a block diagram illustrating the overall process.



The test database was loaded using flat files. All load scripts are included in Appendix B.

Any differences between the configuration of the qualification database and the test database must be disclosed.

The qualification database used identical scripts to create and load the data with only the necessary adjustments for size differences.

6. Clause 5 Performance Metrics and Execution Rules

6.1. System Activity Between Load and Performance Tests

Any system activity on the SUT that takes place between the conclusion of the load test and the beginning of the performance test must be fully disclosed.

1. The SUT was restarted at the conclusion of the load test, before the beginning of the performance test.

All scripts and queries used are included in Appendix F.

6.2. Steps in the Power Test

The details of the steps followed to implement the power test (.e.g., system boot, database restart, etc.) must be disclosed.

The following steps were used to implement the power test:

1. RF1 Refresh Transaction
2. Stream 00 Execution
3. RF2 Refresh Transaction

6.3. Timing Intervals for Each Query and Refresh Functions

The timing intervals for each query and for both refresh functions must be reported for the power test.

The timing intervals for each query and both update functions are reported on the page titled “Numerical Quantities”, contained in the beginning of this document and replicated in the Executive Summary document.

6.4. Number of Streams for the Throughput Test

The number of execution streams used for the throughput test must be disclosed.

6 query streams and one refresh stream were used for the throughput test.

6.5. Start and End Date/Times for Each Query Stream

The start time and finish time for each query stream must be reported for the throughput test.

The start times and finish times for each query stream in the throughput test are reported on the page titled "Numerical Quantities", contained in the beginning of this document and replicated in the Executive Summary document.

6.6. Total Elapsed Time of the Measurement Interval

The total elapsed time of the measurement interval must be reported for the throughput test.

The total elapsed time of the throughput test is reported on the page titled "Numerical Quantities", contained in the beginning of this document and replicated in the Executive Summary document.

6.7. Refresh Function Start Date/Time and Finish Date/Time

Start and finish time for each refresh function in the refresh stream must be reported for the throughput test.

The start and finish times for each refresh function:

Stream ID	Refresh Function	Start Date	Start Time	End Date	End Time
Stream1	RF1	04/19/08	08:38:34	04/19/08	08:39:38
Stream1	RF2	04/19/08	08:39:38	04/19/08	08:40:43
Stream2	RF1	04/19/08	08:40:43	04/19/08	08:41:48
Stream2	RF2	04/19/08	08:41:48	04/19/08	08:42:30
Stream3	RF1	04/19/08	08:42:30	04/19/08	08:43:33
Stream3	RF2	04/19/08	08:43:33	04/19/08	08:44:18
Stream4	RF1	04/19/08	08:44:18	04/19/08	08:45:19
Stream4	RF2	04/19/08	08:45:19	04/19/08	08:46:01
Stream5	RF1	04/19/08	08:46:01	04/19/08	08:47:04
Stream5	RF2	04/19/08	08:47:04	04/19/08	08:47:49
Stream6	RF1	04/19/08	08:47:49	04/19/08	08:48:51
Stream6	RF2	04/19/08	08:48:51	04/19/08	08:49:36

6.8. Timing Intervals for Each Query and Each Refresh Function for Each Stream

The timing intervals for each query of each stream and each refresh function must be reported for the throughput test.

The timing intervals for each query and each refresh function for the throughput test are reported on the page titled "Numerical Quantities", contained in the beginning of this document and replicated in the Executive Summary document.

6.9. Performance Metrics

The computed performance metric, related numerical quantities and price performance metric must be reported.

The performance metrics, and the numbers on which they are based, are reported on the page titled "Numerical Quantities", contained in the beginning of this document and replicated in the Executive Summary document.

6.10. Performance Metric and Numerical Quantities from Both Runs

The performance metric and numerical quantities from both runs must be disclosed.

Performance results from the first two executions of the TPC-H benchmark indicated the following percent difference for the three metrics:

Run ID	QppH@ 300GB	QthH@ 300GB	QphH@ 300GB
1	66183.9	45531.8	54895.1
2	80639.4	46061.4	60945.6

6.11. System Activity Between Performance Tests

Any activity on the SUT that takes place between the conclusion of Run1 and the beginning of Run2 must be disclosed.

No activity occurred between Run 1 and Run 2.

7. Clause 6 SUT and Driver Implementation

7.1. Driver

A detailed description of how the driver performs its functions must be supplied, including any related source code or scripts. This description should allow an independent reconstruction of the driver.

The entire test is run by executing a python script tpch_run.py.

The text of tpch_run.py is reproduced in Appendix E and the texts of the subscripts invoked by tpch_run.py are reproduced in Appendix B.

Tpch_run.py will invokes QGEN to generate the actual query stream files before performing the power and throughput tests.

The Power Test is performed when tpch_run.py is executed with option –p –u. It generates and invokes refresh functions files (rf_1_0_[1-2].txt) and then the power stream queries (query_1_s0_q[1-22].sql).

The Throughput Test is performed when tpch_run.py is executed with –p –u option. It concurrently executes the 6 query stream scripts (query_1_s[1-6]_q[1-22].txt), together with the refresh function stream (rf_1_[1-6]_[1-2].sql].

7.2. Implementation-Specific Layer

If an implementation-specific layer is used, then a detailed description of how it performs its functions must be supplied, including any related source code or scripts. This description should allow an independent reconstruction of the implementation-specific layer.

All database configuration was done through scripts disclosed in Appendix B.

The ACID tests are performed using MySQL. All the ACID test scripts are reproduced in Appendix B.

7.3. Profile-Directed Optimization

If profile-directed optimization as described in Clause 5.2.9 is used, such use must be disclosed.

Profile-directed optimization was not used.

8. Clause 7 Pricing

8.1. Hardware and Software Used

A detailed list of hardware and software used in the priced system must be reported. Each item must have vendor part number, description, and release/revision level, and either general availability status or committed delivery date. If package-pricing is used, contents of the package must be disclosed. Pricing source(s) and effective date(s) of price(s) must also be reported.

Refer to the Executive Summary for the pricing spreadsheet and Appendix G for the actual price quotes used to create the spreadsheet.

8.2. Total Three Year Price

The total 3-year price of the entire configuration must be reported, including hardware, software, and maintenance charges. Separate component pricing is recommended. The basis of all discounts used must be disclosed.

The total 3-year price of the configuration is $48,790.

Discount is computed based on purchase price and volume; similarly sized models will receive similar discount as quoted here.

For details of pricing, see the second page of the Executive Summary.

8.3. Availability Date

The committed delivery date for general availability of products used in the price calculations must be reported. When the priced system includes products with different availability dates, the reported availability date for the priced system must be the date at which all components are committed to be available.

All hardware and software components used in the measured configuration are generally available as of October 14, 2008.

8.4. Auditor's Information and Attestation Letter

The auditor's agency name, address, phone number, and Attestation letter with a brief audit summary report indicating compliance must be included in the full disclosure report. A statement should be included specifying who to contact in order to obtain further information regarding the audit process.

The auditor's attestation letter precedes the table of contents.

A. Kickfire Database and Linux Parameters

This Appendix contains Linux kernel parameters and environment variables and database system parameters.

- KickFire Database Server parameters (altered from default)
    - Not applicable.
- Linux parameters (altered from default)
    - Not Applicable

B. Programs and Scripts

a. acidt.py [DB_NAME] [PORT#]

```
#!/usr/bin/python
import socket
import random
import MySQLdb
import threading
import Queue
import sys
import time
import datetime
import os
import fixedpoint

def canonical_trunc(n, p): return float(long(n* (10**p)))/(10**p)
def canonical_trunc_decimal(n, p): return fixedpoint.FixedPoint(long(n*
(10**p)),n.precision)/(10**p)
def decimal(number):
      a = fixedpoint.FixedPoint(number,2)
      a.set_precision(12)
      return a
#

#psize = 0.001
#phost = "fred.c2app.com"
phost = socket.gethostname()
if os.environ.has_key("TPCH_HOST"):
      phost = os.environ["TPCH_HOST"]
psize = 1
if len(sys.argv) > 4:
      psize = float(sys.argv[4])

global durability_over
durability_over=0

def pity_log(q):
      a = open("attempt.log","a")
      a.write("%s\n" % (q,))
      a.close()

def get_date_string():
      if len(sys.argv) > 3:
            return "LOG"
      else:
            return datetime.datetime.now().time()

def log_transaction(history_tuple, return_tuple):
      logf = open("dura.log","a")
      logf.write(",".join(map(str,history_tuple)))
      logf.write("|")
      logf.write(",".join(map(str,return_tuple)))
      logf.write("\n")
      logf.close()
```

```
def read_log():
      logf = open("dura.log","r")
      for line in logf:
            yield map((lambda x: x.split(",")),line.split("|"))
      logf.close()

def run_descriptive_acid_query(o_key, cursor):
      #print o_key
      cursor.execute("SELECT truncate(truncate(L_EXTENDEDPRICE * (1 -
L_DISCOUNT),2) * (1 + L_TAX),2), l_extendedprice, l_discount, l_tax FROM
lineitem WHERE L_ORDERKEY =" + str(o_key))
      val = cursor.fetchall()
      for row in val:
            print row

def run_acid_query(o_key, cursor):
      #print o_key
      cursor.execute("set @c2_skip_intercept:='true';")
      cursor.execute("SELECT SUM(truncate(truncate(L_EXTENDEDPRICE * (1 -
L_DISCOUNT),2) * (1 + L_TAX),2)) FROM lineitem WHERE L_ORDERKEY =" +
str(o_key))
      val = cursor.fetchone()
      cursor.execute("set @c2_skip_intercept:='false';")
      try:
            rval=val[0]
            return rval
      except TypeError, te:
            print te
            return val

def get_order_state(values, cursor):
      o_orderkey, l_linenumber, delta = values
      cursor.execute("select o_totalprice from orders where o_orderkey = " +
str(o_orderkey) + ";")
      record = cursor.fetchone()
      try:
            ototal = record[0]
      except TypeError:
            ototal = record
      global durability_over
      if durability_over:
            print durability_over
            sys.exit(0) #This will only be set during Dura, and then only if
the test is over.
      cursor.execute("select L_QUANTITY, L_EXTENDEDPRICE, L_PARTKEY,
L_SUPPKEY, L_TAX, L_DISCOUNT from lineitem where l_orderkey = " +
str(o_orderkey) + " and l_linenumber = " + str(l_linenumber) + ";")
      record = cursor.fetchone()
      if record == None:
            print "FATAL ERROR NO RID"
            print ototal, values
            print "WTF"
            sys.exit(1)
      quantity = int(decimal(record[0]))
      extprice = decimal(record[1])
      pkey = record[2]
      skey = record[3]
      tax = decimal(record[4])
      disc = decimal(record[5])
      return (decimal(ototal),quantity,extprice,pkey,skey,tax,disc)
```

```
def is_consistent(cursor, buffer, sf):
      for o_orderkey, l_linenumber, delta in buffer:
            cursor.execute("select o_totalprice from orders where o_orderkey="
+ str(o_orderkey))
            o_val = (cursor.fetchone()[0])
            l_val = (run_acid_query(o_orderkey, cursor))
            if o_val != l_val:
                  print "\tMISMATCH on order #%d: recorded total price is %s
but calculated price is %s!" % (o_orderkey, o_val, l_val)
                  return o_orderkey
            else:
                  print "\tMatch on order #%d: Consistent total price %s." %
(o_orderkey, o_val)
      return -1

def run_acid_transaction(values, cursor, ending, wait=0, report=None, resub=0,
readysig=None):
      try:
            o_orderkey, l_linenumber, delta = values
            cursor.execute("start transaction;")
            if (report != None):
                  print "\t[%s] ACID Transaction"%get_date_string(),report,"on
values",values,("IN FLIGHT","RESUBMIT")[resub]
                  sys.stdout.flush()
            a = open("attempt.log","a")
            a.write("%s" % (values,))
            a.close()
            ototal,quantity,extprice,pkey,skey,tax,disc =
get_order_state(values,cursor)
            pity_log( ("ototal", ototal, "quantity", quantity, "extprice",
extprice))
            if report != None:
                  print "\t\tOld values as seen by Txn %s: o_totalprice=%f,
l_extendedprice=%f, l_quantity=%d" % (report,ototal,extprice,quantity)
            ototal = ototal -
canonical_trunc_decimal(canonical_trunc_decimal(extprice * (1 - disc),2) *
(1+tax),2)
            rprice = canonical_trunc_decimal((extprice)/quantity,2)
            cost = canonical_trunc_decimal((rprice)*delta,2)
            new_extprice = (extprice + cost)
            new_ototal = canonical_trunc_decimal(new_extprice * (1.0-disc), 2)
            new_ototal = canonical_trunc_decimal(new_ototal * (1.0+tax), 2)
            new_ototal += ototal
            if readysig: #We need to signal back that we're starting
                  readysig.put("Ready")
            pity_log(("changing extprice to", str(new_extprice), "quantity
to", str(quantity+delta), "totalprice to", new_ototal))
#           print "update orders set o_totalprice=" + str(new_ototal) + "
where o_orderkey=" + str(o_orderkey) + ";"
            if report != None:
                  print "\t\tNew values as seen by Txn %s: o_totalprice=%s,
l_extendedprice=%s, l_quantity=%d" %
(report,new_ototal,new_extprice,quantity+delta)
            try:
                  try:
                        cursor.execute("update orders set o_totalprice=" +
str(new_ototal) + " where o_orderkey=" + str(o_orderkey) + ";")
                  except Warning, oe:
                        a,b = oe
                        print oe, a, b
```

```
                        pass
            except Exception, reale:
                  #a,b = reale
                  #print reale, a,b
                  if str(reale).find("Warnings") > -1:
                        pass #print reale
                  else:
                        raise reale
#           print "update lineitem set l_extendedprice=" + str(new_extprice) +
", l_quantity=" + str(quantity+delta) + " where l_orderkey = " +
str(o_orderkey) + " and l_linenumber = " + str(l_linenumber) + ";"
            try:
                  try:
                        cursor.execute("update lineitem set l_extendedprice="
+ str(new_extprice) + ", l_quantity=" + str(quantity+delta) + " where
l_orderkey = " + str(o_orderkey) + " and l_linenumber = " + str(l_linenumber)
+ ";")
                  except Warning:
                        pass
            except Exception, reale:
                  #print type(reale)
                  if str(reale).find("Warnings") > -1:
                        pass#print reale
                        #cursor.execute("show warnings;")
                        #print cursor.fetchall()
                  else:
                        raise reale
            try:
                  print "insert into history values(" + str(pkey) + "," +
str(skey) + "," + str(o_orderkey) + "," + str(l_linenumber) + "," + str(delta)
+ ", now());"
                  cursor.execute("insert into history values(" + str(pkey) +
"," + str(skey) + "," + str(o_orderkey) + "," + str(l_linenumber) + "," +
str(delta) + ", now());")
            except Warning:
                  pass
            if wait:
                  print "\tACID Transaction",report,"WAITING in mid-flight -
fail the S.U.T now."
                  sys.stdout.flush()
                  atoc = "Y"
                  if len(sys.argv) > 3:
                        #Database kill and restart occur right here, right
now.
                        condemned_pid =
int(open(os.environ["MYSQLTEST_VARDIR"]+"/run/master.pid").read())
                        os.kill(condemned_pid,9)
                        print "\tWaiting for the process to die..."
                        while condemned_pid:
                              try:
                                    os.getpgid(condemned_pid)
                                    time.sleep(.25)
                              except OSError, e:
                                    if e[0]==3:
                                          condemned_pid=0
                        print "\tKilled with signal 9."
                        atoc = "Y"
                        if os.environ.has_key("WRAPPER_RUN"):
                              os.system("$VIEW_ROOT/mysql-5.1/sql/mysqld --
pid-file=$VIEW_ROOT/mysql-5.1/mysql-test/var/run/master.pid --port=12361 --
core-file --basedir=$VIEW_ROOT/mysql-5.1 --datadir=$VIEW_ROOT/mysql-5.1/mysql-
```

```
test/var/master-data --language=$VIEW_ROOT/mysql-5.1/sql/share/english --
socket=$VIEW_ROOT/mysql-5.1/mysql-test/var/tmp/master.sock --
plugin_dir=$VIEW_ROOT/c2-test/unittests/c2-shared-objects-udf &> mysql.err &")
                        else:
                              os.system(os.environ["EXE_MYSQLD"]+" --no-
defaults --console --socket="+os.environ["MASTER_MYSOCK"]+" --pid-
file="+os.environ["MYSQLTEST_VARDIR"]
                                    +"/run/master.pid --
port="+os.environ["MASTER_MYPORT"]+"  --basedir="+os.environ["BASEDIR"]+" --
datadir="+os.environ["MYSQLTEST_VARDIR"]
                                    +"/master-data --skip-innodb --skip-
ndbcluster --tmpdir=. --core-file --wait_timeout=300 --
language="+os.environ["LANGUAGEDIR"]
                                    +" --character-sets-
dir="+os.environ["CHARSETSDIR"]+"  --allow-suspicious-udfs  >&
"+os.environ["MYSQLTEST_VARDIR"]+"/log/master_2.err &")
                        time.sleep(10)
                  else:
                        atoc = raw_input("\tDo we have to reconnect
(Yes/No/Quit)?")
                  global durability_over
                  durability_over=1
                  if (atoc[0].lower() == "y") or (atoc[0].lower() == "n"):
                        if (atoc[0].lower() == "y"):
                              print "\tAttempting to reconnect..."
                              innercursor = None
                              while innercursor == None:
                                    try:
                                          rdbconn =
MySQLdb.connect(host=phost, user="root",db=sys.argv[1], port=int(sys.argv[2]))
                                          innercursor = rdbconn.cursor()
                                    except MySQLdb.OperationalError, oe:
                                          if oe[0] == 2003:
                                                innercursor = None
                                          else:
                                                raise oe
                        else:
                              innercursor = cursor
                              innercursor.execute("rollback;")
                        print "\tServer found, connection established.
Beginning validation of data..."
                        logct = 0
                        youlose = False
                        for history_tuple, status_tuple in read_log():
                              innercursor.execute("select count(*) from
history where h_p_key=%s and h_s_key=%s and h_o_key=%s and h_l_key=%s and
h_delta=%s;"%tuple(history_tuple[:5]))
                              zig = innercursor.fetchone()
                              if (zig == None) or zig[0] != 1:
                                    print "\tMissing or duplicated
transaction:",history_tuple
                                    print "\tFound total",zig,"matches in
History!"
                                    youlose=True
                              logct += 1
                        innercursor.execute("select count(*) from history;")
                        histct = innercursor.fetchone()[0]
                        if int(histct) != logct:
                              print "\tHistory table is %d rows, but duralog
is %d rows."%(histct,logct)
                              youlose = True
```

```
                        if not youlose:
                              print "\tAll",
                              if len(sys.argv)<=3:
                                    print logct
                              print "rows in duralog have matching rows in
history table."
                        innercursor.execute("select * from history where
h_p_key=%d and h_s_key=%d and h_o_key=%d and h_l_key=%d and
h_delta=%d;"%(pkey,skey,o_orderkey,l_linenumber,delta))
                        if innercursor.fetchone():
                              print "\tThe transaction that was in flight
during the crash seems to have committed."
                              youlose = True
                        sresult =
is_consistent(innercursor,d_buffers[0][:10],psize)
                        innercursor.execute("select * from history;")
                        fhisto = open("history.tbl","a")
                        for row in innercursor.fetchall():
                              fhisto.write("\t".join(map(str,row))+"\n")
                        fhisto.close()
                        if sresult > -1:
                              print "} FAILED due to inconsistency at",sresult
                              return -2
                        if youlose:
                              print "} FAILED"
                              return -2
                        print "} PASSED"
                        return -1
                  else:
                        print "Atoc",atoc,"aborting"
                        sys.exit(113)
            if report and (ending.find("select") == -1):
                  print "\t[%s] ACID
Transaction"%get_date_string(),report,ending.upper()[:-1]
                  sys.stdout.flush()
            cursor.execute(ending)
            rettuple = (rprice,quantity,tax,disc,extprice,ototal)
            if (ending.lower().find("commit") > -1) or (ending.find("2") > -
1):

      log_transaction((pkey,skey,o_orderkey,l_linenumber,delta,datetime.dateti
me.now()),rettuple)
            sys.stdout.flush()
            return rettuple
      except MySQLdb.OperationalError, oe:
            a, b = oe
            if a == 2013:
                  print "\tConnection lost during ACID transaction."
                  raise SystemExit
            elif a != 1213:
                  raise
            else:
                  time.sleep(1)
                  return run_acid_transaction(values, cursor, ending, wait,
report,1)

def verify_change(state, values, shouldchanged):
      o_orderkey, l_linenumber, delta = values
      srprice, squantity, stax, sdisc, sextprice, sototal = state
      ototal,quantity,extprice,pkey,skey,tax,disc =
get_order_state(values,cursor)
```

```
      cursor.execute("select count(*) from history where h_o_key = " +
str(o_orderkey) + " and h_l_key = " + str(l_linenumber) + ";")
      result = cursor.fetchone()
      if shouldchanged:
            scost = canonical_trunc_decimal((srprice)*delta,2)
            new_extprice = (sextprice + scost)
            new_ototal = canonical_trunc_decimal(new_extprice * (1.0-sdisc),
2)
            new_ototal = canonical_trunc_decimal(new_ototal * (1.0+stax), 2)
            new_ototal += sototal
            if canonical_trunc_decimal(extprice, 2) !=
canonical_trunc_decimal(new_extprice, 2):
                  if abs(canonical_trunc_decimal(extprice, 2) -
canonical_trunc_decimal(new_extprice, 2)) <= new_extprice/10000:
                        print "(fudge)"
                  else:
                        print "extprice", extprice, new_extprice
                        return 0
            if quantity != squantity + delta:
                  if abs(quantity - (squantity+delta)) <= quantity/10000:
                        print "(fudge)"
                  print "quantity", quantity, squantity+delta
                  return 0
            if ototal != new_ototal:
                  if str(ototal) != str(new_ototal):
                        if abs(ototal - new_ototal) <= ototal/10000:
                              print "ototalfudge", ototal,new_ototal
                        else:
                              print "ototal", ototal, new_ototal
                              return 0
            if result == 0: return 0
      else:
            if extprice != sextprice:
                  if abs(extprice - sextprice) <= sextprice/10000:
                        print ("(fudge)")
                  else:
                        print "extprice",extprice,sextprice
                        return 0
            if quantity != squantity:
                  print "quantity", quantity, squantity
                  return 0
            if ototal - canonical_trunc_decimal(canonical_trunc(extprice * (1
- disc),2) * (1+tax),2) != sototal:
                  if abs(ototal -
canonical_trunc_decimal(canonical_trunc(extprice * (1 - disc),2) * (1+tax),2)
- sototal) <= sototal/10000:
                        print "(fudge)"
                  else:
                        print "ototal",ototal,"does not update to",sototal
                        return 0
            if result > 0:
                  try:
                        if result[0] > 0:
                              print result,"matching row in HISTORY!!"
                              return 0
                  except:
                        print result,"matching row in HISTORY!!"
                        return 0
      return 1
```

```
def run_acid_query_apart(o_key, cursor, resultq):
      print "\tACID Query on",o_key,"launched; this should block."
      rrr = run_acid_query(o_key,cursor)
      print "\tACID Query on",o_key,"has returned a result."
      resultq.put(rrr)

def run_acid_transaction_apart(buffer, cursor,
resultq,sign=False,report=None,commitmsg="commit;"):
      print "\tACID transaction on",buffer,"launched; this should block."
      if sign:
            rrr =
run_acid_transaction(buffer,cursor,commitmsg,0,report,0,resultq)
      else:
            rrr = run_acid_transaction(buffer,cursor,commitmsg,0,report)
      print "\tACID transaction on",buffer,"has returned a result."
      resultq.put(rrr)


class Signal:
      def __init__(self,waitcount):
            self.counter=waitcount
            self.lock=threading.Lock()
      def __call__(self,decrement):
            self.lock.acquire()
            self.counter -= decrement
            rv=self.counter
            self.lock.release()
            return rv

def run_transaction_stream(buffer, streamconn, streamlen, do_pause,
id,sigobj):
      global durability_over
      stream = streamconn.cursor()
      for iter in map(int,range(streamlen/2)):
            run_acid_transaction(buffer[iter],stream,"commit;",0,("%d (%.4d)"
% (id,iter),None)[len(sys.argv) > 3])
      if do_pause:
            sigobj(1)
            print "\tStream",(id,"is")[len(sys.argv) > 3],"ready"
            for iter in map(int,range(streamlen/2,streamlen)):
                  #When we've done the crash, no need to continue.
                  if
run_acid_transaction(buffer[iter],stream,"commit;",(0,5)[sigobj(0) <
1],(None,None)[sigobj(0) < 1]  ) == -1:
                        return None
                  if durability_over:
                        return None
            if len(sys.argv) <= 3:
                  print "\tStream",id,"ran to the end before failure could be
induced."
                  print "} TEST SPOILED"
                  durability_over=1
            return None
      else:
            for iter in map(int,range(streamlen/2,streamlen)):
                  run_acid_transaction(buffer[iter],stream,"commit;",0,("%d
(%.4d)" % (id,iter),None)[len(sys.argv) > 3])

def run_transaction_streams(streams, streambuffers, do_pause, sf):
      threads = []
```

```
      sid=0;
      sig=Signal(len(streams))
      sblist=streambuffers[:] #100% sure of non-destructivity
      for stream in map(int,range(len(streams))):
            buffer = sblist[0]
            sblist = sblist[1:]
            threads +=
[threading.Thread(target=run_transaction_stream,args=(buffer,streams[stream],l
en(buffer),do_pause,sid,sig) )]
            sid += 1
      for thread in threads: thread.start()
      for thread in threads: thread.join()

def expected_acid_query_result(buffer,transactionresult):
      o_orderkey, l_linenumber, delta = buffer
      rprice,quantity,tax,disc,extprice,ototal = transactionresult
      return canonical_trunc_decimal(
            canonical_trunc_decimal(
                  (extprice + canonical_trunc_decimal((rprice)*delta,2))
             * (1.0-disc), 2)
      * (1.0+tax), 2)+ototal

def run_refresh1(cursor, orders_stream_filename, lineitem_stream_filename):
      cursor.execute("load data infile " + orders_stream_filename + " into
table orders fields terminated by '|';")
      cursor.execute("load data infile " + lineitem_stream_filename + " into
table lineitem fields terminated by '|';")

def run_refresh2(cursor, deletion_stream_filename):
      delfile = open(deletion_stream_filename,"r")
      for line in delfile:
            cursor.execute("delete from lineitem where l_orderkey=" +
line.split("|")[0]+";")
            cursor.execute("delete from orders where o_orderkey=" +
line.split("|")[0]+";")
      delfile.close()

global canonical_lcnt_curvals
canonical_lcnt_curvals = [1434868289L]
def get_canonical_lcnt(o_orderkey):
      global canonical_lcnt_curvals
      rowid = ((o_orderkey / 32) * 8) + (o_orderkey %16)
      if len(canonical_lcnt_curvals) == 1:
            curval = 1434868289L # Thank you TPC for this magic number that I
am about to eat.
            dRange = 7.0
            nRange = 7
            for indx in range(int(psize*1500000)):
                  canonical_lcnt_curvals += [(canonical_lcnt_curvals[-1] *
16807) % 2147483647]
      return 1 + long((  float(canonical_lcnt_curvals[rowid]) / 2147483647.0 )
* 7.0)
get_canonical_lcnt(0)

def gen_acid_buffer(count, sf):
      acid_buffer = []
      for indx in range(count):
            o_orderkey = (random.randint(0,int(187500*sf)-1)*32) +
random.randint(0,7)
            if o_orderkey == 0:
                  o_orderkey = int(187500*sf)*32
```

```
            l_linenumber = random.randint(1,get_canonical_lcnt(o_orderkey))
            delta = random.randint(1,100)
            acid_buffer += [(o_orderkey,l_linenumber,delta)]
      return acid_buffer


sleeptime = 30 # We can make this configurable, too.

transcount = 100
if len(sys.argv) > 5:
      transcount = int(sys.argv[5])

dbconn = MySQLdb.connect(host=phost, user="root",db=sys.argv[1],
port=int(sys.argv[2]))

if len(sys.argv) > 6:
      streams = []
      for i in range(int(sys.argv[6])):
            streams += [MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2]))]
else:
      streams = [MySQLdb.connect(host=phost, user="root",db=sys.argv[1],
port=int(sys.argv[2])),\
      MySQLdb.connect(host=phost, user="root",db=sys.argv[1],
port=int(sys.argv[2])),\
      MySQLdb.connect(host=phost, user="root",db=sys.argv[1],
port=int(sys.argv[2])),\
      MySQLdb.connect(host=phost, user="root",db=sys.argv[1],
port=int(sys.argv[2])),\
      MySQLdb.connect(host=phost, user="root",db=sys.argv[1],
port=int(sys.argv[2]))]


cursor = dbconn.cursor()


if len(sys.argv) > 3:   # For individual tests, we want reproducible results.
      random.seed(sys.argv[3])

create_acid_testtable = """create table history(
      h_p_key integer not null,
      h_s_key integer not null,
      h_o_key integer not null,
      h_l_key integer,
      h_delta integer,
      h_date_t datetime,
      foreign key h_p_fk(h_p_key) references part(p_partkey),
      foreign key h_s_fk(h_s_key) references supplier(s_suppkey),
      foreign key h_o_fk(h_o_key) references orders(o_orderkey)

      )engine=c2db;"""

query_one = """select
      l_returnflag,
      l_linestatus,
      sum(l_quantity) as sum_qty,
      sum(l_extendedprice) as sum_base_price,
      sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
      sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
```

```
      avg(l_quantity) as avg_qty,
      avg(l_extendedprice) as avg_price,
      avg(l_discount) as avg_disc,
      count(*) as count_order
from
      lineitem
where
      l_shipdate <= date '1998-12-01' - interval '%d' day
group by
      l_returnflag,
      l_linestatus
order by
      l_returnflag,
      l_linestatus;"""


buffer = gen_acid_buffer(12,psize)

open("dura.log","w").close() #Clean HISTORY requires clean log.
open("attempt.log","w").close() #Clean HISTORY requires clean log.

cursor.execute("drop table if exists history;")
print "...",
cursor.execute(create_acid_testtable)
print "ready"

if (len(sys.argv) <= 3) or (sys.argv[3].find("a=1") > -1):
      print "/tpch/acid/atomicity/commit",
      state = run_acid_transaction(buffer[0],cursor,"commit;")
      if not verify_change(state,buffer[0],1):
            print "FAILED"
            print state
            print buffer[0]
            raw_input("Press Enter to continue.")
      else:
            print "PASSED"


if (len(sys.argv) <= 3) or (sys.argv[3].find("a=2") > -1):
      print "/tpch/acid/atomicity/rollback",
      state = run_acid_transaction(buffer[1],cursor,"rollback;")
      if not verify_change(state,buffer[1],0):
            print "FAILED"
            print state
            print buffer[1]
            raw_input("Press Enter to continue.")
      else:
            print "PASSED"

c_buffers = map(lambda x: gen_acid_buffer(transcount, psize),streams)
if (len(sys.argv) <= 3) or (sys.argv[3].find("c=1") > -1):
      print "/tpch/acid/consistency/prologue {"
      sresult = is_consistent(cursor,c_buffers[0][:10],psize)
      if sresult > 0:
            print "} FAILED due to inconsistent order at orderkey", sresult
      else:
            print "} PASSED"

if (len(sys.argv) <= 3) or (sys.argv[3].find("c=2") > -1):
      print "/tpch/acid/consistency/stream-run {"
      run_transaction_streams(streams,c_buffers,0,psize)
```

```
	print "}"
	print "/tpch/acid/consistency/epilogue {"
	sresult = is_consistent(cursor,c_buffers[0][:10],psize)
	if sresult > 0:
		print "} FAILED due to inconsistent order at orderkey", sresult
	else:
		print "} PASSED"


resultq = Queue.Queue(2) #Used for passing back second-stream results.

if (len(sys.argv) <= 3) or (sys.argv[3].find("i=1") > -1):
	print "/tpch/acid/isolation/read-write-commit {"
	state = run_acid_transaction(buffer[2],cursor,"select 2;",0,"#1") #
Select 2 does nothing.
	print "\t[%s] ACID transaction #1 suspended before
commit."%(get_date_string())
	isothread =
threading.Thread(target=run_acid_query_apart,args=(buffer[2][0],
streams[0].cursor() , resultq))
	isothread.start()
	print "\t[%s] ACID query launched; this should
block."%(get_date_string())
	print "\tACID transaction will now sleep",sleeptime,"seconds to prove
that this is a 'real' block."
	time.sleep(sleeptime)

	try:
		resultq.get(False)
		print "\tError - the ACID query ran through the ACID transaction's
lock!"
		print "} FAILED"
	except Queue.Empty:
		print "\t[%s] ACID transaction #1 may now wake up and
commit."%get_date_string()
		cursor.execute("commit;")
		print "\t[%s] ACID transaction #1 has
committed."%(get_date_string())
		try:
			returnedval = resultq.get(True,sleeptime)
			print "\t[%s] ACID query has returned calculated total
price:"%(get_date_string()),returnedval
			expectedval = expected_acid_query_result(buffer[2],state)
			print "\tExpected value by our calculations is:",expectedval
			if (expectedval == returnedval):
				print "} PASSED"
			else:
				print "} FAILED due to mismatch."
		except Queue.Empty:
			print "\tError - the ACID query didn't wake up after the
commit!"
			print "} FAILED"

if (len(sys.argv) <= 3) or (sys.argv[3].find("i=2") > -1):
	print "/tpch/acid/isolation/read-write-rollback {"
	state = run_acid_transaction(buffer[3],cursor,"select 1;",0,"#1") #
Select 1 does nothing.
	print "\t[%s] ACID transaction #1 suspended before
rollback."%(get_date_string())
```

```
      isothread =
threading.Thread(target=run_acid_query_apart,args=(buffer[3][0],
streams[0].cursor() , resultq))
      isothread.start()
      print "\t[%s] ACID query launched; this should
block."%(get_date_string())
      print "\tACID transaction will now sleep",sleeptime,"seconds to prove
that this is a 'real' block."
      time.sleep(sleeptime)
      try:
            resultq.get(False)
            print "\tError - the ACID query ran through the ACID transaction's
lock!"
            print "} FAILED"
      except Queue.Empty:
            print "\t[%s] ACID transaction #1 may now wake up and roll
back."%get_date_string()
            cursor.execute("rollback;")
            print "\t[%s] ACID transaction #1 has rolled
back."%(get_date_string())
            try:
                  returnedval = resultq.get(True,sleeptime)
                  expectedval = expected_acid_query_result(buffer[3],state)
                  print "\t[%s] ACID query has returned calculated total
price:"%(get_date_string()),returnedval
                  if expectedval != returnedval:
                        print "} PASSED"
                  else:
                        print "Mismatch - ACID query returned",
returnedval,"but this change should have been rolled back!"
                        print "} FAILED"
            except Queue.Empty:
                  print "\tError - the ACID query didn't wake up after the
rollback!"
                  print "} FAILED"



if (len(sys.argv) <= 3) or (sys.argv[3].find("i=3") > -1):
      print "/tpch/acid/isolation/write-write-commit {"
      buffer[5] = buffer[4][:2] + buffer[5][2:]
      #buffer[5][0] = buffer[4][0]
      #buffer[5][1] = buffer[4][1]
      state = run_acid_transaction(buffer[4],cursor,"select 2;",0,"#1") #
Select 2 does nothing.
      print "\t[%s] ACID transaction #1 suspended before
commit."%(get_date_string())
      isothread =
threading.Thread(target=run_acid_transaction_apart,args=(buffer[5],
streams[0].cursor() , resultq,False,"#2"))
      isothread.start()
      print "\t[%s] ACID transaction #2 launched; this should
block."%(get_date_string())
      print "\tACID transaction #1 will now sleep",sleeptime,"seconds to prove
that this is a 'real' block."
      time.sleep(sleeptime)
      try:
            resultq.get(False)
            print "\tError - Txn2 ran through Txn1's lock!"
            print "} FAILED"
```

```
      except Queue.Empty:
            print "\t[%s] ACID transaction #1 may now wake up and
commit."%get_date_string()
            cursor.execute("commit;")
            print "\t[%s] ACID transaction #1 has
committed."%(get_date_string())
            try:
                  returnedval = resultq.get(True,sleeptime)
                  expectedval =
canonical_trunc_decimal(canonical_trunc_decimal((state[4])/state[1],2)
*(buffer[4][2]),2) + state[4]
                  print "\t[%s] Transaction #2 committed, setting
l_extendedprice to:"%(get_date_string()),returnedval[4]
                  print "\tExpected value for extprice (based on txn 1)
is",expectedval
                  if (expectedval == returnedval[4]) or ("%.4f"%(expectedval)
== "%.4f"%(returnedval[4])):
                        print "} PASSED"
                  else:
                        print "} FAILED due to mismatch"
            except Queue.Empty:
                  print "\tError - Txn2 didn't wake up after the commit!"
                  print "} FAILED"


if (len(sys.argv) <= 3) or (sys.argv[3].find("i=4") > -1):
      print "/tpch/acid/isolation/write-write-rollback {"
      buffer[7] = buffer[6][:2] + buffer[7][2:]
      state = run_acid_transaction(buffer[6],cursor,"select 1;",0,"#1") #
Select 1 does nothing.
      print "\t[%s] ACID transaction #1 suspended before
rollback."%(get_date_string())
      isothread =
threading.Thread(target=run_acid_transaction_apart,args=(buffer[7],
streams[0].cursor() , resultq,))
      isothread.start()
      print "\tACID transaction #1 will now sleep",sleeptime,"seconds to prove
that this is a 'real' block."
      time.sleep(sleeptime)
      try:
            resultq.get(False)
            print "\tError - Txn2 ran through Txn1's lock!"
            print "} FAILED"
      except Queue.Empty:
            print "\t[%s] ACID transaction #1 may now wake up and roll
back."%get_date_string()
            cursor.execute("rollback;")
            print "\t[%s] ACID transaction #1 has rolled
back."%(get_date_string())
            try:
                  returnedval = resultq.get(True,sleeptime)
                  print "\t[%s] Transaction #2 committed, setting
l_extendedprice to:"%(get_date_string()),returnedval
                  if state[4] == returnedval[4]:
                        print "} PASSED"
                  else:
                        print "Mismatch - Txn2 saw an l_extendedprice of",
returnedval[4],"but that doesn't match Txn1's observed price of",expectedval
                        print "} FAILED"
            except Queue.Empty:
                  print "\tError - Txn2 didn't wake up after the rollback!"
```

```
                print "} FAILED"


if (len(sys.argv) <= 3) or (sys.argv[3].find("i=5") > -1):
      print "/tpch/acid/isolation/read-write-noninterfere {"
      state = run_acid_transaction(buffer[8],cursor,"select 2;",0,"#1") #
Select 1 does nothing.
      print "\t[%s] ACID transaction #1 suspended before
commit."%(get_date_string())
      partsupp_selectset = (random.randint(1,10000*psize),
random.randint(1,200000*psize))
      isothread =
threading.Thread(target=streams[0].cursor().execute,args=("select * from
partsupp where ps_partkey=%d and ps_suppkey=%d" % partsupp_selectset,))
      isothread.start()
      print "\t[%s] Partsupp query"%(get_date_string()),"dispatched. We will
wait",sleeptime,"for it to complete."
      isothread.join(sleeptime)
      if isothread.isAlive():
            print "\tQuery did not run. It was probably blocked by the
transaction."
            print "} FAILED"
      else:
            print "\t[%s] Partsupp query has returned; we can commit
now."%(get_date_string())
            cursor.execute("commit;");
            print "\t[%s] ACID transaction #1 has
committed."%(get_date_string())
            if not verify_change(state,buffer[8],1):
                  print "\tMismatch - ACID transaction did not update
properly. :("
                  print "} FAILED"
            else:
                  print "} PASSED"


def run_query_delayed(qstr,c):
      #print "\t1\tBegin"
      c.execute("start transaction;")
      #print "\t1\tExec"
      c.execute(qstr)
      #print "\t1\tSleep"
#     c.execute("select sleep(1)")
      time.sleep(1)
      #print "\t1\tCommit"
      c.execute("commit;")

if (len(sys.argv) <= 3) or (sys.argv[3].find("i=6") > -1):
      print "/tpch/acid/isolation/three-transaction-lock {"
      txnone = threading.Thread(target=run_query_delayed,args=(query_one %
(0,),streams[0].cursor()));
      txntwo =
threading.Thread(target=run_acid_transaction_apart,args=(buffer[9],cursor ,
resultq,True,"#1","select 2;"))
      txnthree =
threading.Thread(target=streams[1].cursor().execute,args=(query_one % (1,),))
      txnone.start()
      print "\t[%s] Query (#1) dispatched"%(get_date_string())
      txntwo.start()
      #It's told to report; it WILL report.
```

```
      resultq.get() # Wait for the transaction to signal that it's started to
do real work.
      time.sleep(1)
      txnthree.start()
      print "\t[%s] Query (#3) dispatched"%(get_date_string())
      if not txnone.isAlive():
            print "} Query #1 finished too quickly; test spoiled and must be
re-run."
            txnone.join()
            txntwo.join()
            txnthree.join()
      else:
            txnone.join()
            print "\t[%s] Query (#1) returned. Transaction (#2) should return
next."%get_date_string()
            txntwo.join()
            print "\t[%s] Transaction (#2) is ready to commit. We will sleep
here to prove that the query is legitimately blocked."%get_date_string()
            time.sleep(sleeptime - 1)
            if not txnthree.isAlive():
                  print "\tQuery (#3) has returned before transaction commit!"
                  print "} FAILED"
            else:
                  print "\t[%s] Transaction (#2) will now
commit."%get_date_string()
                  cursor.execute("commit;")
                  print "\tWaiting for Query (#3) to return..."
                  txnthree.join()
                  print "\t[%s] Query (#3) returned."%get_date_string()
                  try:
                        if not
verify_change(resultq.get(True,sleeptime),buffer[9],1):
                              print "\tMismatch - Transaction did not update
properly. :("
                              print "} FAILED"
                        else:
                              print "} PASSED"
                     except Queue.Empty:
                              print "\tError - Transaction never returned a
value - must have crashed."
                              print "} FAILED"

#sys.exit(0)


if (len(sys.argv) <= 3) or (sys.argv[3].find("d=") > -1):
      failures = ["killnine"] #for example
      for failure in failures:
            print "/tpch/acid/durability/" + str(failure) + "{"
            d_buffers = map(lambda x: gen_acid_buffer(2*transcount,
psize),streams)
            sresult =
is_consistent(streams[0].cursor(),d_buffers[0][:10],psize)
            durability_over=0
            if sresult > 0:
                  print "} FAILED due to inconsistent database."
            else:

      run_transaction_streams(streams,d_buffers,2*transcount,psize)
                  if failure != failures[-1]:
```

```
                        streams = [MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2])),\
                                   MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2])),\
                                   MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2])),\
                                   MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2])),\
                                   MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2])),\
                                   MySQLdb.connect(host=phost,
user="root",db=sys.argv[1], port=int(sys.argv[2]))]

if (len(sys.argv) > 3) and (sys.argv[3].find("d=0") == -1):
      os.system("mysqladmin --socket="+os.environ["MASTER_MYSOCK"]+" -u root
shutdown")
      sys.exit(0)


# Don't blink.
```

b. tpch_run.py

```
#!/usr/bin/env python

"""
tpch audit run

Options:
  -h, --help      show this help
  --db=           database name
  -s, --scale=    scale factor (default 1, 1GB)
  -l, --loader    call loader
  -p, --power     power test only
  -t, --throughput  throughput test only
  --num_thread     number of thread in the througput run, will override
default if given
  -u, --update    do update or rf1, rf2

  --test_only     only test
  -d              debug mode

  -e, --rand_seed  random seed for qgen, if use loader, will use timestamp
after loading, -1 default
  --run_id=       The id of the run, default 1 -> results/r1,  use different
one to avoid overwrite earlier results
  --no_cleanup    If loader is called, it will cleanup the db directory, call
this if loader will use the current dir and load to a different db
  --leave_server_up  set this if want the server to be up after run, default
is to shut it down

  -q, --query_program=  program for query streams, default to use mysql
  -r, --root=     root directoy, default to c2-tests/tpch_audit
  g
"""

import sys
import getopt
```

```
import re
import random
import copy
from threading import Thread
import difflib
import os
from sets import Set
import pdb
import time
import shutil
import inspect
import math

# global variables
_debug = 0
mysock=""
env_view_root=""
mysql_test_dir=""
mysql_cmd = ""
mysql_cmd_loader = ""
load_cmd=""


# global variables
script_log_file=""
run_id=1
scale_factor=1
rand_seed=1
audit_dir=""
dbgen_dir=""
qgen=""
output_dir=""
data_dir=""
num_streams_scale ={1:5,10:5,30:4,100:5,300:6,1000:7,3000:8,10000:9}
loader_input_dir={1:'/nfs/dumpster/tpch_fixed_width/1G_col_order',

10:['/data2/testdb/tpch_fixed_width/10G_col_order','/nfs/dumpster/tpch_fixed_w
idth/10G_col_order'],
                  30:'/nfs/dumpster/tpch_fixed_width/30G_col_order',
                  100:['/data2/testdb/tpch_fixed_width/100G_col_order',
'/nfs/dumpster/tpch_fixed_width/100G_col_order'],
                  300:['/data2/testdb/tpch_fixed_width/300G_col_order',
'/nfs/dumpster/tpch_fixed_width/300G_col_order']}

gen_new_rfs = False             # if set, gen RF data using dbgen on the fly,
inc_loader_input will NOT has effect
inc_loader_input_dir={1:'/nfs/dumpster/tpch_fixed_width/1G_col_order_IL',

10:['/data2/testdb/tpch_fixed_width/10G_col_order_IL','/nfs/dumpster/tpch_fixe
d_width/10G_col_order_IL'],
                  30:'/nfs/dumpster/tpch_fixed_width/30G_col_order_IL',
                  100:['/data2/testdb/tpch_fixed_width/100G_col_order_IL',
'/nfs/dumpster/tpch_fixed_width/100G_col_order_IL'],
                  300:['/data2/testdb/tpch_fixed_width/300G_col_order_IL',
'/nfs/dumpster/tpch_fixed_width/300G_col_order_IL']}

do_power=False
do_throughput=False
do_update=False
do_loader=False
rf_tmp_file_name_template = "%s/rf%s_%s_s%s.sql"
```

```
query_file_template="%s/query_%s_s%s.sql"
use_mysql_client = True
tpch_conn = None
test_only=False
rf_throughput_concurrent=False
num_seqs=1                        # how many iterations
num_rf_streames=1
tpch_db_name = "tpcd_1G"
no_cleanup = False               # do not cleanup
leave_server_up = False          # leave the server up after the run
restart_after_load = False       # restart after load

qtime=[]   # query time (seq, stream_id, qnum, start,end, time)
rftime=[]  # refresh time (seq, stream_id, rf_id, start, end, time)
streams_time=[]  # streams time  (seq, stream_id, start, end, duration)
tpch_power_numbers=[]
tpch_throughput_numbers=[]
all_queries={}

# for reporting
master_err_file_hanle=None
master_err_file=None

#====Utilties============
def sys_call(cmd):
    dbg_print("%s:%s" % (os.getcwd(),cmd))
    os.system(cmd)

def sys_pipe_call(cmd):
    dbg_print("%s:%s" % (os.getcwd(),cmd))
    return os.popen(cmd).read()

def sys_pipe_call_1(cmd):
    ''' also return the errors '''
    dbg_print("%s:%s" % (os.getcwd(),cmd))
    return os.popen4(cmd)[1].read()

def sys_call_env(cmd):
    cmds=cmd.split()
    dbg_print("cmds= %s " % cmds)
    os.spawnv( os.P_WAIT, cmds[0], cmds[1:])

def usage():
    caller_name = sys._getframe(1).f_code.co_name
    if (caller_name=='main'):
        print __doc__
    else:
        print globals()[caller_name].__doc__
    sys.exit(0)

def dbg_print(in_str):
    if _debug:
        print ("== " + sys._getframe(1).f_code.co_name + " == " + str(in_str))

def whoami():
    return sys._getframe(1).f_code.co_name

def tpch_error(s):
        print ("TPCH Error:" + s)
        sys.exit(1)
```

```
def get_time():
    return float("%0.2f" % time.time())   # keep 2 digits

def write_log(s,log_file_name=None,timestamp=True,pstdout=False):
    if not log_file_name: pstdout = True  # print the master
    indent_start=4
    if not log_file_name: log_file_name = script_log_file
    #dbg_print("log_file_name=%s" % log_file_name)
    #dbg_print("script_log_file=%s" % script_log_file)
    #indent = "".join(["  " for i in range(len(inspect.stack()))])
    indent = "".rjust((len(inspect.stack())-indent_start)*3)
    ts = timestamp and "%s :" % time.strftime('%x %X') or ""
    line = ts + indent + s + "\n"
    open(log_file_name,"a+").write(line)
    if pstdout: sys.stdout.write(line)
    if not log_file_name: dbg_print(line)

def setup_mysql_udfs():
    udfs = '''
    # create the UDF
drop function if exists c2loader;
create function c2loader returns string soname "c2loader.so";
'''
    in_file = "%s/mysql_init.sql" %  (output_dir)
    open(in_file,"w").write(udfs)
    output = sys_pipe_call_1(mysql_cmd % in_file)
    out_file = "%s/mysql_init.log" %  (output_dir)
    open(out_file,"w").write(output)

import distutils.dir_util
def setup_mysql_copy_seed(force=False):
    db_dir = "%s/var/master-data" % mysql_test_dir
    if (force):
        write_log("Remove database and copy seed",pstdout=True)
        shutil.rmtree(db_dir)
        os.mkdir(db_dir)
    if not os.path.isfile("%s/data.c2db" % db_dir):
        distutils.dir_util.mkpath(db_dir)
        sys_call("cp -rf  %s/std_data/c2db_seed_db/* %s" % (mysql_test_dir,
db_dir))
    distutils.dir_util.mkpath("%s/var/log" % mysql_test_dir)

def setup_mysql():
    global mysql_test_dir, mysock, env_view_root
    mysql_test_dir=os.getenv('MYSQL_TEST_DIR')
    mysock=os.getenv('MASTER_MYSOCK')
    env_view_root=os.getenv('VIEW_ROOT')
    if not mysql_test_dir:
        if (env_view_root):
            mysql_test_dir="%s/mysql-5.1/mysql-test" % env_view_root
            mysock='%s/mysql-5.1/mysql-test/var/tmp/master.sock' %
env_view_root
            if not server_is_up(): mysock='%s/master.sock' % audit_dir
    if not mysql_test_dir: raise AssertionError, "MYSQL_TEST_DIR is not
defined\n"
    setup_mysql_copy_seed(do_loader and not no_cleanup)  # cleanup

    os.chdir("%s/var/log" % mysql_test_dir)
    if not mysock: raise AssertionError, "no Master Sock\n"
    global mysql_cmd, mysql_cmd_loader
    mysql_cmd = " cat %s | mysql --sock=" + mysock + " --database=%s --force
```

```
-u root -v -v -v" % tpch_db_name # disregard error
    mysql_cmd_loader = " cat %s | mysql --sock=" + mysock + " --force  -u root
-v -v -v"
    dbg_print("mysql_cmd = %s" % mysql_cmd)
    #if server_is_up(): shutdown_server()
    start_up_server_if_needed()
    #setup_mysql_udfs()

def setup_tpch():
    global audit_dir, dbgen_dir, qgen, output_dir, script_log_file
    audit_dir=os.getenv('AUDIT_DIR')
    global env_view_root
    env_view_root=os.getenv('VIEW_ROOT')
    mysql_test_dir=os.getenv('MYSQL_TEST_DIR')
    if not audit_dir:
        if (env_view_root):
            audit_dir="%s/c2-test/tpch_audit" % env_view_root
        if (mysql_test_dir):
            audit_dir="%s/../../c2-test/tpch_audit" % mysql_test_dir
    if not audit_dir: raise AssertionError, "AUDIT_DIR is not defined and not
in a view\n"

    if not dbgen_dir: dbgen_dir="%s/dbgen" % audit_dir
    if not os.path.isfile("%s/dbgen" % dbgen_dir):  # compile if needed
        try:
            saved_cur_dir = os.getcwd()
        except OSError:
            saved_cur_dir=audit_dir
        os.chdir(dbgen_dir)
        os.environ['PATH'] = ".:%s" %os.environ['PATH']
        sys_call("make -f makefile")
        os.chdir(saved_cur_dir)
        if not os.path.isfile("%s/dbgen" % dbgen_dir):
            tpch_error("%s/dbgen does not exist" % dbgen_dir)

    try:
        os.mkdir("%s/results" % audit_dir)
    except: pass
    output_dir = "%s/results/r%s" % (audit_dir, run_id)
    try:
        shutil.rmtree(output_dir)
    except: pass
    os.mkdir(output_dir)

    qgen = "%s/qgen" % dbgen_dir
    # set this so qgen can find the query template
    os.environ['DSS_QUERY'] = '%s/queries' % (dbgen_dir)
    os.environ['DSS_PATH'] = output_dir    # this is where the update file is
stored
    os.environ['DSS_CONFIG'] = dbgen_dir

    script_log_file = "%s/tpch.log" % (output_dir)
    setup_mysql()   # setup mysql related

def gen_load_tpch_ddl(ddl_file):
    cmd = '''
    drop database if exists %s;
    create database %s;
    use %s;

    drop function if exists c2loader;
```

```
    create function c2loader returns string soname "c2loader.so";

    set @cur_db='%s';
    set storage_engine=c2db;
    source %s/scripts/c2loader_table_creation_with_index.inc;
    use %s;
    show tables;
    ''' % (tpch_db_name, tpch_db_name, tpch_db_name, tpch_db_name, audit_dir,
tpch_db_name)
    open(ddl_file,"w").write(cmd)

def location_exists(input_location, error_out=True):
    if not isinstance(input_location, list): locations = [input_location]
    else: locations = input_location
    location = None
    for d in locations:
        if os.path.exists(d) and os.path.isdir(d):
            location = d
            break
    if not location and error_out:
        tpch_error("Input directory '%s' does not exist" % input_location)
    return location

def gen_load_tpch_file(in_file):
    input_dir = location_exists(loader_input_dir[scale_factor])

    loader_options = ",'-flags', 'on_window_size=-1'"
    cmd = '''
    use %s;

    select c2loader('INITIAL',
        '%s.region',
        '%s/region.tbl',
        'FIXED_WIDTH_ROW',
        'r_name:25#:r_comment:116#',

        '%s.nation',
        '%s/nation.tbl',
        'FIXED_WIDTH_ROW',
        'n_name:25#:n_comment:116#',

        '%s.customer',
        '%s/customer.tbl',
        'FIXED_WIDTH_ROW',

'c_name:25#:c_address:40#:c_phone:15#:c_mktsegment:10#:c_comment:117#',

        '%s.orders',
        '%s/orders.tbl',
        'FIXED_WIDTH_ROW',
        'o_orderstatus:1#:o_orderpriority:15#:o_clerk:15#:o_comment:79#',

        '%s.part',
        '%s/part.tbl',
        'FIXED_WIDTH_ROW',

'p_name:55#:p_mfgr:25#:p_brand:10#:p_type:25#:p_container:10#:p_comment:23#',

        '%s.supplier',
        '%s/supplier.tbl',
        'FIXED_WIDTH_ROW',
```

```
        's_name:25#:s_address:40#:s_phone:15#:s_comment:101#',

        '%s.partsupp',
        '%s/partsupp.tbl',
        'FIXED_WIDTH_ROW',
        'ps_comment:199#',

        '%s.lineitem',
        '%s/lineitem.tbl',
        'FIXED_WIDTH_ROW',

'l_quantity:011.00:l_returnflag:1#:l_linestatus:1#:l_shipinstruct:25#:l_shipmo
de:10#:l_comment:44#'

        %s
        );

    ''' % (tpch_db_name,
    tpch_db_name, input_dir,   # region
    tpch_db_name, input_dir,   # nation
    tpch_db_name, input_dir,   # customer
    tpch_db_name, input_dir,   # orders
    tpch_db_name, input_dir,   # part
    tpch_db_name, input_dir,   # supplier
    tpch_db_name, input_dir,   # partsupp
    tpch_db_name, input_dir, loader_options)   # lineitem, avoid confusing
emacs, add '
    open(in_file,"w").write(cmd)

def gen_cnt_ri_file(cnt_file):
    cnt_cmd='''

select 'Count after loading', now();

use %s
select count(*) from region;
select count(*) from nation;
select count(*) from customer;
select count(*) from orders;
select count(*) from supplier;
select count(*) from part;
select count(*) from partsupp;
select count(*) from lineitem;

/*********************************************************************/
select 'RI verification starting', now();
/*********************************************************************/

/*********************************************************************/
/* this should failed because cust key is invalid                    */
/*********************************************************************/
insert into orders
      (
       O_ORDERKEY ,
       O_CUSTKEY  ,
       O_ORDERSTATUS    ,
       O_TOTALPRICE     ,
       O_ORDERDATE      ,
       O_ORDERPRIORITY,
       O_CLERK    ,
       O_SHIPPRIORITY   ,
```

```
       O_COMMENT
      )
values (
      8,
      1500000001,
      'O',
      112631.22,
      '08/12/1995',
      '3-MEDIUM',
      'Clerk#002676842',
      1,
      'variable text size 79'
      )
;
/*********************************************************************/
/* this should failed because nation key is invalid                  */
/*********************************************************************/
insert into customer
      (
       C_CUSTKEY,
       C_NAME     ,
       C_ADDRESS,
       C_NATIONKEY,
       C_PHONE,
       C_ACCTBAL,
       C_MKTSEGMENT,
       C_COMMENT
      )
values (
      150000002,
      'Lorna Livingtree',
      '140 Redwood Hiway',
      29,
      '25-989-741-2988',
      711.56,
      'BUILDING',
      'variable text size 117'
      )
;

/*********************************************************************/
/* nation has a foreign key of N_REGIONKEY                           */
/* region key values are [01234]
*/
/* this should failed because region key is still valid              */
/*********************************************************************/

delete from nation
      where N_REGIONKEY = 4
;

/*********************************************************************/
/* customer has a foreign key of C_NATIONKEY                         */
/* nation key values are 1 thru 24                                   */
/* this should failed because nation key is still valid              */
/*********************************************************************/

delete from customer
      where       C_NATIONKEY = 17
;
```

```
/*************************************************************************/
select 'RI verification DONE', now()
;
/*************************************************************************/
    ''' % tpch_db_name # lineitem, avoid confusing emacs, add '
    # good for 1TB, customer 150 million, orders 1500 million
    open(cnt_file,"w+").write(cnt_cmd)

def load_tpch():
    line_header="Loading"
    write_log("Start: load database")
    # generation
    cnt_file = "%s/cnt_ri.sql" % (output_dir)
    gen_cnt_ri_file(cnt_file)

    ddl_file = "%s/ddl.sql" %  (output_dir)
    ddl_log = "%s/ddl.log" %  (output_dir)
    gen_load_tpch_ddl(ddl_file)

    cnt_log = "%s/cnt_ri.log" % (output_dir)
    in_file = "%s/loader.sql" %  (output_dir)
    gen_load_tpch_file(in_file)

    if (do_loader):
        if not test_only:
            write_log("Begin DDL")
            output = sys_pipe_call_1(mysql_cmd_loader % ddl_file)
            open(ddl_log,"w+").write(output)
            write_log("End DDL")
        local_log = "%s/loader.log" %  (output_dir)
        write_log("%s begin" % (line_header))
        write_log("%s begin" % (line_header), local_log)
        start_time=get_time()
        write_log("%s start time %s" % (line_header, start_time))
        write_log("%s start time %s" % (line_header, start_time), local_log)
        write_log("%s start time %s" % (line_header,
conv_to_date(start_time)))
        write_log("%s start time %s" % (line_header,
conv_to_date(start_time)), local_log)
        if not test_only: output = sys_pipe_call_1(mysql_cmd_loader % in_file)
        else: output = "test only\n"
        end_time = get_time()
        duration = end_time - start_time
        open(local_log,"a+").write(output)
        write_log("%s end time %s" % (line_header, end_time))
        write_log("%s end time %s" % (line_header, end_time), local_log)
        write_log("%s end time %s" % (line_header, conv_to_date(end_time)))
        write_log("%s end time %s" % (line_header, conv_to_date(end_time)),
local_log)
        write_log("%s duration %s" % (line_header, duration))
        write_log("%s duration %s" % (line_header, duration), local_log)
        global rand_seed
        if rand_seed >=0:
            rand_seed = long("".join(["%02d" % x for x in
time.localtime(end_time)[1:6]]))
            write_log("Seed based on loading end time = %s" % rand_seed)
        else:
            rand_seed = None
    write_log("End: load database")
    write_log("Begin RI and count")
```

```
    if not test_only and do_loader:
        output = sys_pipe_call_1(mysql_cmd_loader % cnt_file)
        open(cnt_log,"w+").write(output)
    write_log("End RI and count")

    if do_loader and restart_after_load:
        write_log("Restarting after load")
        if server_is_up(): shutdown_server()
        start_up_server_if_needed()

#    raw_input("End of loading #1 . Press any key to continue!!!")
#    raw_input("End of loading #2. Press any key to continue!!!")
#    raw_input("End of loading #3. Press any key to continue!!!")

def compute_result(power_run,seq,num_streams,duration,result):
    if power_run:
        run_time = [x[-2] for x in qtime if x[0] == seq and x[1] == 0 ]
        run_time.extend([x[-1] for x in rftime if x[0] == seq and x[1] == 0 ])
    else:
        run_time = [x[-2] for x in qtime if x[0] == seq and x[1] != 0 ]
        run_time.extend([x[-1] for x in rftime if x[0] == seq and x[1] != 0 ])

    num_queries_and_rfs = len(run_time)
    write_log("%s Seq %s num queries and rfs = %s" % (power_run and "Power" or
"Throughput", seq,num_queries_and_rfs), pstdout=True)
    geo_mean = math.exp(sum([math.log(t) for t in
run_time])*1.0/float(num_queries_and_rfs))
    arith_mean = sum([t for t in run_time])/float(num_queries_and_rfs)
    if power_run: result_number = 3600.0*scale_factor/geo_mean
    else: result_number = (num_streams*22.0*3600)/duration*scale_factor

    result.append((result_number, geo_mean, arith_mean, duration))

def print_detail_results(order_seq, result_seq):

    if order_seq == 'Run Order':
        qtime.sort(lambda x,y: int(float(x[0]*10000+x[1]*100)-
float(y[0]*10000+y[1]*100)))
    else:
        qtime.sort(lambda x,y: int(float(x[0]*10000+x[1]*100+x[2])-
float(y[0]*10000+y[1]*100+y[2])))

    write_log("TPCH Detailed Results %s" % order_seq )
    for i in range(result_seq):   # seq number
        tmp_qtime = [list(x) for x in qtime if x[0]==i+1]
        for q in tmp_qtime:
            write_log("Seq %s, Streams %s, Query %s: start_time= %s, end_time
= %s, duration = %s " % tuple(q[:-1]))
            (seq, stream_id, qnum) = tuple(q[:3])
            out_file = "%s/query_%s_s%s_q%s.log" % (output_dir,seq,
stream_id,qnum)
            open(out_file,"w+").write(q[-1])
        tmp_qtime=([list(x) for x in rftime if x[0] ==i+1])
        for q in tmp_qtime:
            write_log("Seq %s, Streams %s, RF %s: start_time = %s,
end_time=%s, duration = %s " % (tuple(q)))

def print_results(result_seq=None):
    ''' print all the results '''

    if not result_seq: result_seq = num_seqs
```

```
    print_detail_results('Run Order', result_seq)
    print_detail_results('Sorted', result_seq)

    write_log("TPCH Streams Results")
    streams_time.sort(lambda x,y: int(float(x[0]*100+x[1])-
float(y[0]*100+y[1])))
    for x in streams_time:
#        write_log("Seq %s, Streams %s: start_time = %s , end_time = %s,
duration = %s " % x)
        (seq_id, stream_id, start_time, end_time, duration) = x
        stream_rand_seed = rand_seed and long(rand_seed) + stream_id or
'default'
        if stream_id<100: tmp_x = [seq_id, "Query Streams %s" % stream_id,
"rand_seed = %s" % stream_rand_seed]
        else: tmp_x = [seq_id, 'RF throughput streams','']   # RF is 1001 for
sorting purpose
        tmp_x.append(conv_to_date(start_time))
        tmp_x.append(conv_to_date(end_time))
        tmp_x.append(duration)
        write_log("Seq %s, %s: %s, start_time = %s , end_time = %s, duration =
%0.1f " % tuple(tmp_x))

    # aggregated results
    if do_power:
        write_log("TPCH Power Tests Results", pstdout=True)
        for i in range(result_seq):
            write_log("  Power  Seq %s: (Power Metric, GeoMean, ArithMean) =
%s  " % (i+1, "(%0.1f, %0.1f, %0.1f)" % tpch_power_numbers[i][:-1]),
pstdout=True)
    if do_throughput:
        write_log("TPCH Throughput Tests Results",pstdout=True)
        for i in range(result_seq):
            write_log("  ThroughPut Seq %s: (Throughput Metric, GeoMean,
ArithMean, Duration) = %s " % (i+1, "(%0.1f, %0.1f, %0.1f, %0.1f)" %
tpch_throughput_numbers[i]), pstdout=True)
    if do_power and do_throughput:
        write_log("TPCH QphH Results",pstdout=True)
        for i in range(result_seq):
            qph = math.exp((math.log(tpch_power_numbers[i][0]) +
math.log(tpch_throughput_numbers[i][0]))*0.5)
            write_log("  TPCH QphH for Seq %s  = %0.1f " % (i+1, qph),
pstdout=True)


def run_tpch_all():
    load_tpch()   # load the data
    write_log("Start TPC-H Benchmark for Run id: %s " % run_id)
    if do_update:
        gen_rfs()
    for i in range(num_seqs):
        if do_power: run_tpch_power_test(i+1)
        if do_throughput: run_tpch_throughput_test(i+1)
        print_results(i+1)
    if server_is_up():
        if leave_server_up: write_log("  Leave MySQL server up", pstdout=True)
        else: shutdown_server()
    else:
        write_log("  MySQL Server is down during the Run", pstdout=True)
        sys.exit(9);
    write_log("End TPC-H Benchmark for Run id: %s " % run_id)
```

```
def parse_query_file(query_file):
    lines = [l.rstrip() for l in open(query_file,"r").readlines()]
    query_begin_re = re.compile("TPC-H/TPC-R .* Query \(Q([0-9]+)\)")
    query_end_re = re.compile("^set rowcount ([\-0-9]+)")
    dbg_print("query_file = %s " % query_file)
    first_l=lines[0]
    query_begin = False
    queries=[]
    for l in lines:
        m = query_begin_re.search(l)
        if m:
            #dbg_print("begin l= %s " % l)
            query_begin = True
            cur_qry=[]
            qry_name=m.group(1)
            cur_qry.append(first_l)
        if query_begin:
            m1 = query_end_re.search(l)
            if m1:
                query_begin = False
                row_count = int(m1.group(1))
                if row_count > 0:
                    last_line = cur_qry[-1]
                    cur_qry[-1] =  last_line.split(";")[0] + " limit %s;" %
row_count
                queries.append((int(qry_name),"\n".join(cur_qry)))
            else:  cur_qry.append(l)

    if len(queries) != 22:
        print "queries = %s " % queries
        raise AssertionError, "There should be 22 queries"
    return queries

def conv_to_date(t):
    return time.strftime("%X %x",time.localtime(t))

def open_connection():
    if (use_mysql_client): return None
    global tpch_conn
    import MySQLdb
    tpch_conn = MySQLdb.connect (host="localhost", user="root",
passwd="",db=tpch_db_name,unix_socket=mysock)
    if not tpch_conn:  raise AssertionError, "Connection failure"
    else: return tpch_conn

def close_connection():
    if (use_mysql_client): return
    tpch_conn.close()

def execute_one_query(seq, stream_id, qnum, q):
    if (test_only):
        time.sleep(0.1)
        return "Fake results\n"
    #out_file = "%s/query_%s_s%s_q%s.log" % (output_dir,seq,stream_id,qnum)
    if (use_mysql_client):
        in_file = "%s/query_%s_s%s_q%s.sql" % (output_dir,seq,stream_id,qnum)
        #open(in_file,"w").write(q)
        output = sys_pipe_call_1(mysql_cmd % in_file)
        #open(out_file,"w").write(output)
    else:
        cursor = tpch_conn.cursor()
```

```
        cursor.execute(q)
        result = cursor.fetchall()
        cursor.close()
        output = "\n".join(["|".join([str(f) for f in r]) for r in result])
        #open(out_file,"w").write(output)
    return output

def run_query(seq, stream_id):
    #local_log = "%s/query_%s_s%s.log" % (output_dir, seq, stream_id)
    line_header="Query Stream"
    queries=all_queries["%s_%s" % (seq,stream_id)]
    global qtime
    #pdb.set_trace()
    open_connection()
    pre_start_time = None
    for i in range(len(queries)):
        (qnum, q) = queries[i]

        start_time=get_time()
        if pre_start_time:    #  previous interval is pre_start_time
            pre_duration = round_to_point_one(start_time - pre_start_time)
            (pre_seq, pre_stream_id, pre_qnum) = tuple(pre_qtime[:3])
            write_log("%s %s seq %s qnum %s duration %s" % (line_header,
pre_stream_id, pre_seq, pre_qnum, pre_duration))
        write_log("%s %s seq %s qnum %s begin" % (line_header, stream_id, seq,
qnum))
        output = execute_one_query(seq, stream_id, qnum, q)
        end_time = get_time()

        if pre_start_time:    #  previous interval is pre_start_time
            pre_qtime[5] = pre_duration
            qtime.append(tuple(pre_qtime))

        write_log("%s %s seq %s qnum %s start time %s" % (line_header,
stream_id, seq, qnum, start_time))
        write_log("%s %s seq %s qnum %s end time %s" % (line_header,
stream_id, seq, qnum, end_time))

        if pre_start_time and i==21:    #  previous interval is pre_start_time
                duration = round_to_point_one(end_time - start_time)
                duration = math.ceil(duration * 100) / 100
                qtime.append((seq, stream_id, qnum, start_time, end_time,
duration, output))
                pre_start_time = start_time
                write_log("%s %s seq %s qnum %s duration %s" % (line_header,
stream_id, seq, qnum, duration))
        pre_start_time = start_time
        pre_qtime = [seq, stream_id, qnum, start_time, end_time, 0, output]

    close_connection()

def gen_rfs():
    if not do_update: return
    num_streams = num_streams_scale[scale_factor]
    if do_power and not do_throughput:
        num_rf_streams = 1
    elif not do_power and do_throughput:
        num_rf_streams = num_streams
    elif do_power and do_throughput:
        num_rf_streams = num_streams + 1
    dbg_print("gen_rfs begin, num_rf_streams=%s"%num_rf_streams )
```

```
    try:
        saved_cur_dir = os.getcwd()
    except OSError:
        saved_cur_dir=audit_dir
    os.chdir(output_dir)
    global gen_new_rfs
    if not gen_new_rfs:
        input_dir = location_exists(inc_loader_input_dir[scale_factor], False)
        if not input_dir: gen_new_rfs = True

    if gen_new_rfs:
        input_dir = output_dir
        dbgen_file = "%s/dbgen.log" % output_dir
        if not test_only:
            output = sys_pipe_call_1("%s/dbgen -f -O m -s %s -U %s" %
(dbgen_dir, scale_factor, num_rf_streams*num_seqs))
            output = sys_pipe_call_1("%s/dbgen -h" % dbgen_dir)
        else:
            print("%s/dbgen -f -O m -s %s -U %s" % (dbgen_dir, scale_factor,
num_rf_streams*num_seqs))
            output = sys_pipe_call_1("%s/dbgen -h" % dbgen_dir)
        open(dbgen_file,"w").write(output)

    rf_files=["orders.tbl.u%s","lineitem.tbl.u%s","delete.%s"]
    for seq in range(num_seqs):
        for stream_id in range(num_rf_streams):
            rf_seq_no = (stream_id + 1 + num_rf_streams * seq)
            for f in [x % rf_seq_no for x in rf_files]:
                f_full = "%s/%s" % (input_dir,f)
                dbg_print("inc input file = %s" % f_full)
                if not test_only:
                    if not os.path.isfile(f_full): tpch_error("RF file %s does
not exist" % f_full)
                    sys_call("ln -s %s rf_%s_%s_%s" % (f_full, seq+1,
stream_id, f.split(".")[0]))
    dbg_print("gen_rfs done")
    os.chdir(saved_cur_dir)
    for seq in range(num_seqs):
        for stream_id in range(num_rf_streams):
            for i in range(2):  # RF1, RF2
                gen_tmp_rf_file(i+1, seq+1, stream_id)

def gen_tmp_rf_file(rf_id, seq, stream_id):
    file_name = rf_tmp_file_name_template % (output_dir, rf_id, seq,stream_id)
    dbg_print("in gen_tmp_rf_file rf_id= %s " % rf_id)
    if rf_id==1:  # update
        orders_file="%s/rf_%s_%s_%s" % (output_dir,  seq, stream_id, 'orders')
        lineitem_file="%s/rf_%s_%s_%s" % (output_dir, seq,stream_id,
'lineitem')
        f = open(file_name,"w")
        cmd = '''
        drop function if exists c2loader;
        create function c2loader returns string soname "c2loader.so";

        select c2loader('INCREMENTAL', 'INSERT', 'YES',
                '%s.orders',
                '%s',
                'FIXED_WIDTH_ROW',

'o_orderstatus:1#:o_orderpriority:15#:o_clerk:15#:o_comment:79#',
```

```
                '%s.lineitem',
                '%s',
                'FIXED_WIDTH_ROW',

'l_quantity:011.00:l_returnflag:1#:l_linestatus:1#:l_shipinstruct:25#:l_shipmo
de:10#:l_comment:44#');
        ''' % (tpch_db_name,orders_file, tpch_db_name,lineitem_file)   # avoid
confusing emacs, add '

        f.write('set SQL_LOG_BIN=0;\n')  # disable the bin log
        f.write(cmd)
        f.close()

    elif rf_id==2:  # delete
        orders_delete_file =  "%s/rf_%s_%s_delete" % (output_dir, seq,
stream_id)
        f = open(file_name,"w")
        cmd='''
        select c2loader('INCREMENTAL', 'DELETE', 'YES',
                '%s.orders',
                '%s',
                'FIXED_WIDTH_ROW',
                '');
        ''' % (tpch_db_name,orders_delete_file)      # avoid confusing emacs,
add '
        f.write('set SQL_LOG_BIN=0;\n')  # disable the bin log
        f.write(cmd)
        f.close()

def run_rf(rf_id, seq, stream_id):
    global rftime
    if test_only:
        rftime.append((seq, stream_id, rf_id, 1, 2, 3))
        time.sleep(0.1)
    if not do_update:
        #rftime.append((seq, stream_id, rf_id, 101, 10))  # fake it
        return
    dbg_print("begin run_rf %s " % rf_id)
    tmp_file = rf_tmp_file_name_template % (output_dir, rf_id, seq,stream_id)
    local_log = "%s/rf%s_%s_s%s.log" % (output_dir, rf_id, seq, stream_id)
    line_header="RF %s" % rf_id
    write_log("%s seq %s stream %s begin" % (line_header, seq, stream_id))
    write_log("%s seq %s stream %s begin" % (line_header, seq, stream_id),
local_log)
    start_time=get_time()
    if not test_only: output = sys_pipe_call_1(mysql_cmd % tmp_file)
    else: output = "test only"
    end_time = get_time()
    duration = round_to_point_one(end_time - start_time)
    write_log("%s seq %s stream %s start time %s" % (line_header, seq,
stream_id, start_time), local_log)
    write_log("%s seq %s stream %s start time %s" % (line_header, seq,
stream_id, start_time))
    open(local_log,"a+").write(output)
    write_log("%s seq %s stream %s end time %s" % (line_header, seq,
stream_id, end_time))
    write_log("%s seq %s stream %s end time %s" % (line_header, seq,
stream_id, end_time), local_log)
    write_log("%s seq %s stream %s duration %s" % (line_header, seq,
stream_id, duration))
    write_log("%s seq %s stream %s duration %s" % (line_header, seq,
```

```
stream_id, duration), local_log)
    rftime.append((seq, stream_id, rf_id, start_time, end_time, duration))

def run_qgen(stream_id, seq):
    query_parameter_file="%s/qp_%s_s%s.log" % (output_dir, seq, stream_id)
    query_file= query_file_template % (output_dir, seq, stream_id)
    dbg_print("run_qgen begin")
    try:
        saved_cur_dir = os.getcwd()
    except OSError:
        saved_cur_dir=audit_dir
    os.chdir(dbgen_dir)
    if rand_seed:
        stream_rand_seed = long(rand_seed) + stream_id
        write_log("Generates query template file for streams %s with seed: %s
" % (stream_id, stream_rand_seed))
        sys_call("%s -c -r %s -p %s -s %s -l %s > %s" % (qgen,
stream_rand_seed, stream_id, scale_factor, query_parameter_file, query_file))
    else:
        write_log("Generates query template file for streams %s with seed: %s
" % (stream_id, 'default'))
        sys_call("%s -c -d -p %s -s %s -l %s > %s" % (qgen, stream_id,
scale_factor, query_parameter_file, query_file))
    line_header="Query Stream"
    queries = parse_query_file(query_file)
    global all_queries
    all_queries["%s_%s" % (seq,stream_id)] = queries
    for (qnum, q) in queries:
        in_file = "%s/query_%s_s%s_q%s.sql" % (output_dir,seq,stream_id,qnum)
        open(in_file,"w").write(q)
    # result files are in dbgen_dir/orders.tbl.u1 lineitem.tbl.u1. delete.1
    os.chdir(saved_cur_dir)
    dbg_print("run_qgen end")

def run_tpch_power_test(seq=1):
    stream_id = 0   # power_test is stream 0
    global streams_time
    write_log("Start TPC-H Power test Run id: %s Seq: %s" % (run_id, seq))
    run_qgen(stream_id, seq)
    stream_start_time =get_time()
    run_rf(1,seq, stream_id)
    run_query(seq, stream_id)
    run_rf(2, seq, stream_id)
    stream_end_time = get_time()
    stream_duration = round_to_point_one(stream_end_time - stream_start_time)
    streams_time.append((seq, stream_id, stream_start_time, stream_end_time,
stream_duration))
    write_log("End TPC-H Power test Run id: %s Seq: %s" % (run_id, seq))
    global tpch_power_numbers
    compute_result(True, seq, 1, 0, tpch_power_numbers)

class throughput_stream(Thread):
    (rp, wp) = os.pipe()
    def __init__ (self, seq, stream_id, stream_type='query'):
        Thread.__init__(self)
        self.seq = seq
        self.stream_id = stream_id
        self.stream_type=stream_type
    def run(self):
        global streams_time
        if self.stream_type == 'query':
```

```
            stream_start_time =get_time()
            run_query(self.seq, self.stream_id)
            stream_end_time = get_time()
            stream_duration = round_to_point_one(stream_end_time -
stream_start_time)
            streams_time.append((self.seq, self.stream_id, stream_start_time,
stream_end_time, stream_duration))
            os.write(self.wp,"1")
        else:
            for i in range(num_streams_scale[scale_factor]):
                os.read(self.rp,1)
            write_log("RF unblocking")
            stream_start_time =get_time()
            for i in range(num_streams_scale[scale_factor]):
                run_rf(1,self.seq, i+1)
                run_rf(2, self.seq, i+1)
            stream_end_time = get_time()
            stream_duration = stream_end_time - stream_start_time
            streams_time.append((self.seq, self.stream_id, stream_start_time,
stream_end_time, stream_duration))

def round_to_point_one(x):
    return max(math.floor(x * 10 + 0.5) / 10, 0.1)  # minmum 0.1

def run_tpch_throughput_test(seq=1):
    write_log("Start TPC-H Through Put test Run id: %s Seq: %s" % (run_id,
seq))
    num_streams = num_streams_scale[scale_factor]
    t_streams=[]
    for s in range(num_streams):
        run_qgen(s+1, seq)
        t_streams.append(throughput_stream(seq, s+1, 'query'))
    t_streams.append(throughput_stream(seq, 1001, 'rf'))
    #c = raw_input("continue? (Y/N)").upper()
    throughput_start_time=get_time()
    if (rf_throughput_concurrent):
        for t in t_streams: t.start()
        for t in t_streams: t.join()
    else:
        global streams_time
        stream_start_time =get_time()
        for t in t_streams: t.start()
        for t in t_streams: t.join()
    throughput_end_time=get_time()
    throughput_duration = round_to_point_one(throughput_end_time -
throughput_start_time)
    global tpch_throughput_numbers
    compute_result(False, seq, num_streams,
throughput_duration,tpch_throughput_numbers)
    write_log("End TPC-H Through Put test Run id: %s, Seq: %s, Scale: %s,
Number of Streams: %s" % (run_id, seq, scale_factor, num_streams))
    write_log("Through Put duration  = %s" % (throughput_duration))

def server_is_up():
    ret = sys_pipe_call_1("echo exit | mysql --sock=%s" % mysock)
    return not re.compile("Can't connect to local MySQL server through
socket").search(ret)

def start_up_server_if_needed():
    if not server_is_up():
        write_log ("Cannot connect to MySQL server. Trying to start mysqld!",
```

```
pstdout=True)
        global master_err_file_handle, master_err_file
        #master_err_file = "%s/var/log/master.err" % mysql_test_dir
        master_err_file = "%s/master.err" % output_dir
        c2_config_file = "%s/c2_config.cnf" % audit_dir   # enable this in
real audit
        sys_pipe_call_1("rm /log/databases/tpch/*")
        if not os.path.isfile(c2_config_file):
            c2_config_file = "%s/c2_config.cnf" % mysql_test_dir
        import random
        sys_call ("%s/../sql/mysqld --port=%s --core-file --basedir=%s/.. --
datadir=%s/var/master-data --language=%s/../sql/share/english --
c2db_event_log_level=%s --socket=%s --c2db-config-file=%s>& %s &"
              % (mysql_test_dir,9534, mysql_test_dir,mysql_test_dir,
mysql_test_dir, ("ERROR","TRACE")[os.environ.has_key("TPCH_TRACE_EVENTS")],
mysock, c2_config_file, master_err_file ))
        max_wait_sec = 600
        for i in range(max_wait_sec+1):
            if server_is_up():
                write_log("Mysqld started successfully on sock %s" %
mysock,pstdout = True)
                break
            time.sleep(2)
        if (i == max_wait_sec): raise AssertionError, "Time out waiting for
mysql server to start\n"
        master_err_file_handle = open(master_err_file)
        if not master_err_file_handle: raise AssertionError, "master.err is
not created\n"
    else:
        write_log("MySQL server is up. Use the existing one and continue",
pstdout=True)

def shutdown_server():
    if not server_is_up(): return
    dbg_print("Begin shutdown sever")
    sys_call("mysqladmin -u root -S %s shutdown" % mysock)
    max_wait_sec = 20
    for i in range(max_wait_sec+1):
        if not server_is_up():
            write_log("Mysqld is shutdown successfully on sock %s" % mysock,
pstdout=True)
            break
        time.sleep(2)
        if (i == max_wait_sec): raise AssertionError, "Time out waiting for
mysql server to shutdown\n"

def main(argv):
    # default
    global _debug, scale_factor, do_power, do_throughput, do_update,
tpch_db_name, test_only, do_loader, run_id, rand_seed, num_seqs, no_cleanup,
leave_server_up, restart_after_load
    os.environ['TZ'] = 'US/Pacific'
    time.tzset()
    num_thread = None

    try:
       opts, args = getopt.gnu_getopt(argv, "hdq:s:r:ptule:", ["help",
"debug","query_program=", "scale=", "root=", "power", "throughput",
"update","db=","test_only","loader","run_id=","rand_seed=","iter=","no_cleanup
","leave_server_up","num_thread=","restart_after_load"])
    except getopt.GetoptError:
```

```
        usage()
        sys.exit(2)
    for opt, arg in opts:
        if opt in ("-h", "--help"):
            usage()
            sys.exit(0)
        elif opt == '-d':
            _debug = 1
        elif opt in ("-q", "--query_program"):
            query_exec = arg
        elif opt in ("-s", "--scale"):
            scale_factor = int(arg)
        elif opt in ("-r", "--root"):
            audit_dir= arg
        elif opt in ("-p", "--power"):
            do_power = True
        elif opt in ("-t", "--throughput"):
            do_throughput = True
        elif opt in ("-u", "--update"):
            do_update = True
        elif opt in ("-l", "--loader"):
            do_loader = True
            #print "do update"
        elif opt in ("-e", "--rand_seed"):
            rand_seed = int(arg)
        elif opt in ("--test_only"):
            test_only = True
        elif opt in ("--db"):
            tpch_db_name = arg
        elif opt in ("--run_id"):
            run_id = arg
        elif opt in ("--iter"):
            num_seqs = int(arg)
        elif opt in ("--no_cleanup"):
            no_cleanup = True
        elif opt in ("--leave_server_up"):
            leave_server_up = True
        elif opt in ("--restart_after_load"):
            restart_after_load = True
        elif opt in ("--num_thread="):   # comma separated
            num_thread = int(arg)
    if num_thread: num_streams_scale[scale_factor] = num_thread
    setup_tpch()
    run_tpch_all()

if __name__ == "__main__":
    main(sys.argv[1:])
```

c. Change of dbgen 2.6.0 including query template

```
diff -c -r ./config.h /dbgen/config.h
*** ./config.h    2007-08-01 14:45:56.579223000 -0700
--- /dbgen/config.h     2008-04-05 10:41:45.494556000 -0700
```

```
***************
*** 136,142 ****
  #define STDLIB_HAS_GETOPT
  #define SUPPORT_64BITS
  #define DSS_HUGE long long int
- /* #define HUGE_FORMAT      "%09lld"  */
  #define HUGE_FORMAT   "%lld"
  #define HUGE_DATE_FORMAT    "%02lld"
  #define RNG_A   6364136223846793005ull
--- 136,141 ----
diff -c -r ./dss.h /dbgen/dss.h
*** ./dss.h 2006-07-31 09:56:54.000000000 -0700
--- /dbgen/dss.h  2008-04-05 10:41:45.421557000 -0700
***************
*** 476,487 ****
--- 476,491 ----

  int dbg_print(int dt, FILE *tgt, void *data, int len, int eol);
  #define PR_STR(f, str, len)       dbg_print(DT_STR, f, (void *)str, len, 1)
+ #define PR_STR_LAST(f, str, len)        dbg_print(DT_STR, f, (void *)str,
len, 0)
  #define PR_VSTR(f, str, len)      dbg_print(DT_VSTR, f, (void *)str, len, 1)
  #define PR_VSTR_LAST(f, str, len)       dbg_print(DT_VSTR, f, (void *)str,
len, 0)
  #define PR_INT(f, str)                  dbg_print(DT_INT, f, (void *)str, 0,
1)
+ #define PR_INT_LAST(f, str)                   dbg_print(DT_INT, f, (void
*)str, 0, 0)
+
  #define PR_HUGE(f, str)           dbg_print(DT_HUGE, f, (void *)str, 0, 1)
  #define PR_KEY(f, str)                  dbg_print(DT_KEY, f, (void *)str, 0,
-1)
  #define PR_MONEY(f, str)          dbg_print(DT_MONEY, f, (void *)str, 0, 1)
+ #define PR_MONEY_LAST(f, str)           dbg_print(DT_MONEY, f, (void *)str,
0, 0)
  #define PR_CHR(f, str)                  dbg_print(DT_CHR, f, (void *)str, 0,
1)
  #define  PR_STRT(fp)   /* any line prep for a record goes here */
  #define  PR_END(fp)    fprintf(fp, "\n")   /* finish the record here */
diff -c -r ./makefile.suite /dbgen/makefile.suite
*** ./makefile.suite    2007-08-01 18:46:43.891093000 -0700
--- /dbgen/makefile.suite     2008-04-05 10:41:46.345407000 -0700
***************
*** 73,92 ****
  #################
  ## CHANGE NAME OF ANSI COMPILER HERE
  #################
! CC      = gcc
  # Current values for DATABASE are: INFORMIX, DB2, TDAT (Teradata)
  #                                  SQLSERVER, SYBASE
  # Current values for MACHINE are:  ATT, DOS, HP, IBM, ICL, MVS,
  #                                  SGI, SUN, U2200, VMS, LINUX, WIN32
  # Current values for WORKLOAD are:  TPCH
! DATABASE= SYBASE
! MACHINE = LINUX
! WORKLOAD = TPCH
  #
  # add -EDTERABYTE if orderkey will execeed 32 bits (SF >= 300)
  # and make the appropriate change in gen_schema() of runit.sh
  CFLAGS    = -O -DDBNAME=\"dss\" -D$(MACHINE) -D$(DATABASE) -D$(WORKLOAD)
- #CFLAGS   = -g -DDBNAME=\"dss\" -D$(MACHINE) -D$(DATABASE) -D$(WORKLOAD)
```

```
  LDFLAGS = -g
  # The OBJ,EXE and LIB macros will need to be changed for compilation under
  #  Windows NT
--- 73,91 ----
  ################
  ## CHANGE NAME OF ANSI COMPILER HERE
  ################
! CC      =
  # Current values for DATABASE are: INFORMIX, DB2, TDAT (Teradata)
  #                                  SQLSERVER, SYBASE
  # Current values for MACHINE are:  ATT, DOS, HP, IBM, ICL, MVS,
  #                                  SGI, SUN, U2200, VMS, LINUX, WIN32
  # Current values for WORKLOAD are:  TPCH
! DATABASE=
! MACHINE =
! WORKLOAD =
  #
  # add -EDTERABYTE if orderkey will execeed 32 bits (SF >= 300)
  # and make the appropriate change in gen_schema() of runit.sh
  CFLAGS    = -O -DDBNAME=\"dss\" -D$(MACHINE) -D$(DATABASE) -D$(WORKLOAD)
  LDFLAGS = -g
  # The OBJ,EXE and LIB macros will need to be changed for compilation under
  #  Windows NT
diff -c -r ./print.c /dbgen/print.c
*** ./print.c     2007-08-01 15:18:44.108754000 -0700
--- /dbgen/print.c      2008-04-05 10:41:46.418409000 -0700
***************
*** 102,108 ****
  int
  dbg_print(int format, FILE *target, void *data, int len, int sep)
  {
!     int dollars,
            cents;

      switch(format)
--- 102,108 ----
  int
  dbg_print(int format, FILE *target, void *data, int len, int sep)
  {
!     DSS_HUGE dollars,
            cents;

      switch(format)
***************
*** 122,134 ****
  #endif /* MVS */
      case DT_INT:
            if (columnar)
!                 fprintf(target, "%12ld", (long)data);
            else
                  fprintf(target, "%ld", (long)data);
            break;
      case DT_HUGE:
!           /* fprintf(target, HUGE_FORMAT, *(DSS_HUGE *)data); */
!           fprintf(target, "%12ld", *(DSS_HUGE *)data);
            break;
      case DT_KEY:
            fprintf(target, "%ld", (long)data);
--- 122,136 ----
  #endif /* MVS */
      case DT_INT:
```

```
            if (columnar)
!                 fprintf(target, "%011ld", (long)data);
            else
                  fprintf(target, "%ld", (long)data);
            break;
      case DT_HUGE:
!           if (columnar)
!       fprintf(target, "%011lld", *(DSS_HUGE *)data);
!     else
!       fprintf(target, HUGE_FORMAT, *(DSS_HUGE *)data);
            break;
      case DT_KEY:
            fprintf(target, "%ld", (long)data);
***************
*** 140,157 ****
                  fprintf(target, "-");
                  cents = -cents;
                  }
!                 else
!                 fprintf(target, " ");
            dollars = cents / 100;
            cents %= 100;
            if (columnar)
!                 fprintf(target, "%12ld.%02ld", dollars, cents);
            else
                  fprintf(target, "%ld.%02ld", dollars, cents);
            break;
      case DT_CHR:
            if (columnar)
!                 fprintf(target, "%c ", *(char *)data);
            else
                  fprintf(target, "%c", *(char *)data);
            break;
--- 142,161 ----
                  fprintf(target, "-");
                  cents = -cents;
                  }
!         else
!                 {
!                 if (columnar) { fprintf(target, "+"); }
!                 }
            dollars = cents / 100;
            cents %= 100;
            if (columnar)
!                 fprintf(target, "%013ld.%02ld", dollars, cents);
            else
                  fprintf(target, "%ld.%02ld", dollars, cents);
            break;
      case DT_CHR:
            if (columnar)
!                 fprintf(target, "%c", *(char *)data);
            else
                  fprintf(target, "%c", *(char *)data);
            break;
***************
*** 160,166 ****
  #ifdef EOL_HANDLING
      if (sep)
  #endif /* EOL_HANDLING */
!     /*if (!columnar) */
            fprintf(target, "%c", SEPARATOR);
```

```
      return(0);
--- 164,170 ----
  #ifdef EOL_HANDLING
      if (sep)
  #endif /* EOL_HANDLING */
!     if (!columnar)
            fprintf(target, "%c", SEPARATOR);

      return(0);
***************
*** 184,193 ****
          (columnar)?(long)(ceil(C_ADDR_LEN * V_STR_HGH)):c->alen);
     PR_HUGE(fp, &c->nation_code);
     PR_STR(fp, c->phone, PHONE_LEN);
!    PR_MONEY(fp, &c->acctbal);
!    PR_STR(fp, c->mktsegment, C_MSEG_LEN);
!    PR_VSTR_LAST(fp, c->comment,
          (columnar)?(long)(ceil(C_CMNT_LEN * V_STR_HGH)):c->clen);
     PR_END(fp);

     return(0);
--- 188,197 ----
          (columnar)?(long)(ceil(C_ADDR_LEN * V_STR_HGH)):c->alen);
     PR_HUGE(fp, &c->nation_code);
     PR_STR(fp, c->phone, PHONE_LEN);
!    PR_VSTR(fp, c->comment,
          (columnar)?(long)(ceil(C_CMNT_LEN * V_STR_HGH)):c->clen);
+    PR_MONEY(fp, &c->acctbal);
+    PR_STR_LAST(fp, c->mktsegment, C_MSEG_LEN);
     PR_END(fp);

     return(0);
***************
*** 210,220 ****
           last_mode = mode;
           }
      PR_STRT(fp_o);
      PR_HUGE(fp_o, &o->okey);
      PR_HUGE(fp_o, &o->custkey);
      PR_CHR(fp_o, &o->orderstatus);
      PR_MONEY(fp_o, &o->totalprice);
-     PR_STR(fp_o, o->odate, DATE_LEN);
      PR_STR(fp_o, o->opriority, O_OPRIO_LEN);
      PR_STR(fp_o, o->clerk, O_CLRK_LEN);
      PR_INT(fp_o, o->spriority);
--- 214,224 ----
           last_mode = mode;
           }
      PR_STRT(fp_o);
+     PR_STR(fp_o, o->odate, PRINT_DATE_LEN);
      PR_HUGE(fp_o, &o->okey);
      PR_HUGE(fp_o, &o->custkey);
      PR_CHR(fp_o, &o->orderstatus);
      PR_MONEY(fp_o, &o->totalprice);
      PR_STR(fp_o, o->opriority, O_OPRIO_LEN);
      PR_STR(fp_o, o->clerk, O_CLRK_LEN);
      PR_INT(fp_o, o->spriority);
***************
*** 246,251 ****
--- 250,256 ----
```

```
      for (i = 0; i < o->lines; i++)
          {
          PR_STRT(fp_l);
+         PR_STR(fp_l, o->l[i].sdate, PRINT_DATE_LEN);
          PR_HUGE(fp_l, &o->l[i].okey);
          PR_HUGE(fp_l, &o->l[i].partkey);
          PR_HUGE(fp_l, &o->l[i].suppkey);
***************
*** 256,268 ****
          PR_MONEY(fp_l, &o->l[i].tax);
          PR_CHR(fp_l, &o->l[i].rflag[0]);
          PR_CHR(fp_l, &o->l[i].lstatus[0]);
!         PR_STR(fp_l, o->l[i].sdate, DATE_LEN);
!         PR_STR(fp_l, o->l[i].cdate, DATE_LEN);
!         PR_STR(fp_l, o->l[i].rdate, DATE_LEN);
          PR_STR(fp_l, o->l[i].shipinstruct, L_INST_LEN);
!         PR_STR(fp_l, o->l[i].shipmode, L_SMODE_LEN);
!         PR_VSTR_LAST(fp_l, o->l[i].comment,
              (columnar)?(long)(ceil(L_CMNT_LEN * V_STR_HGH)):o->l[i].clen);
          PR_END(fp_l);
          }

--- 261,272 ----
          PR_MONEY(fp_l, &o->l[i].tax);
          PR_CHR(fp_l, &o->l[i].rflag[0]);
          PR_CHR(fp_l, &o->l[i].lstatus[0]);
!         PR_STR(fp_l, o->l[i].cdate, PRINT_DATE_LEN);
!         PR_STR(fp_l, o->l[i].rdate, PRINT_DATE_LEN);
          PR_STR(fp_l, o->l[i].shipinstruct, L_INST_LEN);
!         PR_VSTR(fp_l, o->l[i].comment,
              (columnar)?(long)(ceil(L_CMNT_LEN * V_STR_HGH)):o->l[i].clen);
+         PR_STR_LAST(fp_l, o->l[i].shipmode, L_SMODE_LEN);
          PR_END(fp_l);
          }

***************
*** 295,311 ****

     PR_STRT(p_fp);
     PR_HUGE(p_fp, &part->partkey);
-    PR_VSTR(p_fp, part->name,
-        (columnar)?(long)P_NAME_LEN:part->nlen);
     PR_STR(p_fp, part->mfgr, P_MFG_LEN);
     PR_STR(p_fp, part->brand, P_BRND_LEN);
     PR_VSTR(p_fp, part->type,
         (columnar)?(long)P_TYPE_LEN:part->tlen);
     PR_HUGE(p_fp, &part->size);
     PR_STR(p_fp, part->container, P_CNTR_LEN);
!    PR_MONEY(p_fp, &part->retailprice);
!    PR_VSTR_LAST(p_fp, part->comment,
         (columnar)?(long)(ceil(P_CMNT_LEN * V_STR_HGH)):part->clen);
     PR_END(p_fp);

     return(0);
--- 299,315 ----

     PR_STRT(p_fp);
     PR_HUGE(p_fp, &part->partkey);
     PR_STR(p_fp, part->mfgr, P_MFG_LEN);
     PR_STR(p_fp, part->brand, P_BRND_LEN);
     PR_VSTR(p_fp, part->type,
```

```
         (columnar)?(long)P_TYPE_LEN:part->tlen);
      PR_HUGE(p_fp, &part->size);
      PR_STR(p_fp, part->container, P_CNTR_LEN);
!     PR_VSTR(p_fp, part->comment,
         (columnar)?(long)(ceil(P_CMNT_LEN * V_STR_HGH)):part->clen);
+     PR_MONEY(p_fp, &part->retailprice);
+     PR_VSTR_LAST(p_fp, part->name,
+        (columnar)?(long)P_NAME_LEN:part->nlen);
      PR_END(p_fp);

      return(0);
***************
*** 328,337 ****
         PR_STRT(ps_fp);
         PR_HUGE(ps_fp, &part->s[i].partkey);
         PR_HUGE(ps_fp, &part->s[i].suppkey);
!        PR_HUGE(ps_fp, &part->s[i].qty);
!        PR_MONEY(ps_fp, &part->s[i].scost);
!        PR_VSTR_LAST(ps_fp, part->s[i].comment,
          (columnar)?(long)(ceil(PS_CMNT_LEN * V_STR_HGH)):part->s[i].clen);
         PR_END(ps_fp);
         }

--- 332,341 ----
         PR_STRT(ps_fp);
         PR_HUGE(ps_fp, &part->s[i].partkey);
         PR_HUGE(ps_fp, &part->s[i].suppkey);
!        PR_MONEY_LAST(ps_fp, &part->s[i].scost);
!        PR_VSTR(ps_fp, part->s[i].comment,
          (columnar)?(long)(ceil(PS_CMNT_LEN * V_STR_HGH)):part->s[i].clen);
+        PR_HUGE(ps_fp, &part->s[i].qty);
         PR_END(ps_fp);
         }

***************
*** 385,393 ****
      PR_STRT(fp);
      PR_HUGE(fp, &c->code);
      PR_STR(fp, c->text, NATION_LEN);
!     PR_INT(fp, c->join);
!     PR_VSTR_LAST(fp, c->comment,
         (columnar)?(long)(ceil(N_CMNT_LEN * V_STR_HGH)):c->clen);
      PR_END(fp);

      return(0);
--- 389,397 ----
      PR_STRT(fp);
      PR_HUGE(fp, &c->code);
      PR_STR(fp, c->text, NATION_LEN);
!     PR_VSTR(fp, c->comment,
         (columnar)?(long)(ceil(N_CMNT_LEN * V_STR_HGH)):c->clen);
+     PR_INT_LAST(fp, c->join);
      PR_END(fp);

      return(0);
***************
*** 403,411 ****

      PR_STRT(fp);
      PR_HUGE(fp, &c->code);
!     PR_STR(fp, c->text, REGION_LEN);
```

```
!    PR_VSTR_LAST(fp, c->comment,
         (columnar)?(long)(ceil(R_CMNT_LEN * V_STR_HGH)):c->clen);
     PR_END(fp);

     return(0);
--- 407,415 ----

     PR_STRT(fp);
     PR_HUGE(fp, &c->code);
!    PR_VSTR(fp, c->comment,
         (columnar)?(long)(ceil(R_CMNT_LEN * V_STR_HGH)):c->clen);
+    PR_STR_LAST(fp, c->text, REGION_LEN);
     PR_END(fp);

     return(0);
diff -c -r ./queries/13.sql /dbgen/queries/13.sql
*** ./queries/13.sql    2004-11-24 15:31:52.000000000 -0800
--- /dbgen/queries/13.sql     2008-04-05 10:41:45.120556000 -0700
***************
*** 2,7 ****
--- 2,8 ----
  -- TPC-H/TPC-R Customer Distribution Query (Q13)

  -- Functional Query Definition

  -- Approved February 1998

+ -- dzhang 9/4/07. c_orders column list "c_orders (c_custkey, c_count)" moved
to subselect
  :x

  :o

  select

***************
*** 10,24 ****
  from

      (

            select

!                 c_custkey,

!                 count(o_orderkey)

            from

                  customer left outer join orders on

                        c_custkey = o_custkey

                        and o_comment not like '%:1%:2%'

            group by

                  c_custkey

!     ) as c_orders (c_custkey, c_count)
```

```
  group by

      c_count

  order by

--- 11,25 ----
  from

      (

            select

!                 c_custkey as c_custkey,

!                 count(o_orderkey) as c_count

            from

                  customer left outer join orders on

                        c_custkey = o_custkey

                        and o_comment not like '%:1%:2%'

            group by

                  c_custkey

!     ) as c_orders
  group by

      c_count

  order by

diff -c -r ./queries/1.sql /dbgen/queries/1.sql
*** ./queries/1.sql     2004-11-24 15:31:52.000000000 -0800
--- /dbgen/queries/1.sql      2008-04-05 10:41:45.202559000 -0700
***************
*** 2,7 ****
--- 2,8 ----
  -- TPC-H/TPC-R Pricing Summary Report Query (Q1)

  -- Functional Query Definition

  -- Approved February 1998

+ -- dzhang 9/4/07. Remove interval precision "day (3)" to "day"
  :x

  :o

  select

***************
*** 18,24 ****
  from

      lineitem
```

```
  where

!     l_shipdate <= date '1998-12-01' - interval ':1' day (3)

  group by

      l_returnflag,

      l_linestatus

--- 19,25 ----
  from

      lineitem

  where

!     l_shipdate <= date '1998-12-01' - interval ':1' day

  group by

      l_returnflag,

      l_linestatus

diff -c -r ./shared.h /dbgen/shared.h
*** ./shared.h    2005-01-03 12:08:59.000000000 -0800
--- /dbgen/shared.h     2008-04-05 10:41:46.256411000 -0700
***************
*** 38,44 ****
  #define  S_CMNT_MAX   101
  #define  PS_CMNT_LEN  124
  #define  PS_CMNT_MAX  199
! #define  C_NAME_LEN    18
  #define  C_ADDR_LEN    25
  #define  C_ADDR_MAX    40
  #define  C_MSEG_LEN    10
--- 38,44 ----
  #define  S_CMNT_MAX   101
  #define  PS_CMNT_LEN  124
  #define  PS_CMNT_MAX  199
! #define  C_NAME_LEN    25
  #define  C_ADDR_LEN    25
  #define  C_ADDR_MAX    40
  #define  C_MSEG_LEN    10
***************
*** 54,59 ****
--- 54,60 ----
  #define  L_SMODE_LEN   10
  #define  T_ALPHA_LEN   10
  #define  DATE_LEN      13  /* long enough to hold either date format */
+ #define  PRINT_DATE_LEN      10
  #define  NATION_LEN    25
  #define  REGION_LEN    25
  #define  PHONE_LEN     15
```

d. c2loader_table_creation_with_index.inc

```
#-------------------------------------------------------------
# CREATE region
create table region (r_regionkey integer not null,
                     r_comment char(152) not null,
                     r_name char(25) not null,
                     primary key (r_regionkey)) engine=c2db;
#-------------------------------------------------------------
# CREATE nation
create table nation (n_nationkey integer not null,
                     n_name char(25) not null,
                     n_comment char(152) not null,
                     n_regionkey integer not null,
                     primary key (n_nationkey),
                     foreign key nation_fk1(n_regionkey)
                     references region(r_regionkey)) engine=c2db;
#-------------------------------------------------------------
# CREATE customer
create table customer (c_custkey integer not null,
                       c_name char(25) not null,
                       c_address char(40) not null,
                       c_nationkey integer not null,
                       c_phone char(15) not null,
                       c_comment char(117) not null,
                       c_acctbal decimal(15,2) not null,
                       c_mktsegment char(10) not null,
                       primary key (c_custkey),
                       foreign key customer_fk1(c_nationkey)
                       references nation(n_nationkey)) engine=c2db;
#-------------------------------------------------------------
# CREATE orders
create table orders (o_orderdate date not null,
                     o_orderkey integer not null,
                     o_custkey integer not null,
                     o_orderstatus char(1) not null,
                     o_totalprice decimal(15,2) not null,
                     o_orderpriority char(15) not null,
                     o_clerk char(15) not null,
                     o_shippriority integer not null,
                     o_comment char(79) not null,
                     primary key (o_orderkey),
                     foreign key orders_fk1(o_custkey)
                     references customer(c_custkey)
                     ,index orders_dt_idx (o_orderdate)
                     ) engine=c2db;
#-------------------------------------------------------------
# CREATE part
create table part (p_partkey integer not null,
                   p_mfgr char(25) not null,
                   p_brand char(10) not null,
                   p_type char(25) not null,
                   p_size integer not null,
                   p_container char(10) not null,
                   p_comment char(23) not null,
                   p_retailprice decimal(15,2) not null,
                   p_name char(55) not null,
                   primary key (p_partkey)
) engine=c2db;
#-------------------------------------------------------------
# CREATE supplier
create table supplier (s_suppkey integer not null,
                       s_name char(25) not null,
```

```
                        s_address char(40) not null,
                        s_nationkey integer not null,
                        s_phone char(15) not null,
                        s_acctbal decimal(15,2) not null,
                        s_comment char(101) not null,
                        primary key (s_suppkey),
                        foreign key supplier_fk1(s_nationkey)
                        references nation(n_nationkey)
) engine=c2db;
#-------------------------------------------------------------------
# CREATE partsupp
create table partsupp (ps_partkey integer not null,
                        ps_suppkey integer not null,
                        ps_supplycost decimal(15,2) not null,
                        ps_comment char(199) not null,
                        ps_availqty integer not null,
                        primary key (ps_partkey, ps_suppkey),
                        foreign key partsupp_fk1(ps_partkey)
                        references part(p_partkey),
                        foreign key partsupp_fk2(ps_suppkey)
                        references supplier(s_suppkey)) engine=c2db;
#-------------------------------------------------------------------
# CREATE lineitem
create table lineitem (l_shipdate date not null,
                        l_orderkey integer not null,
                        l_partkey  integer not null,
                        l_suppkey integer not null,
                        l_linenumber integer not null,
                        l_quantity decimal(15,2) not null,
                        l_extendedprice decimal(15,2) not null,
                        l_discount decimal(15,2) not null,
                        l_tax decimal(15,2) not null,
                        l_returnflag char(1) not null,
                        l_linestatus char(1) not null,
                        l_commitdate date not null,
                        l_receiptdate date not null,
                        l_shipinstruct   char(25) not null,
                        l_comment char(44) not null,
                        l_shipmode char(10) not null,
                        primary key (l_orderkey, l_linenumber),
                        foreign key lineitem_fk1(l_orderkey)
                        references orders(o_orderkey),
                        foreign key lineitem_fk2(l_suppkey)
                        references supplier(s_suppkey),
                        foreign key lineitem_fk3(l_partkey, l_suppkey)
                        references partsupp(ps_partkey, ps_suppkey),
                        foreign key lineitem_fk4(l_partkey)
                        references part(p_partkey)
                        ,index li_shp_dt_idx (l_shipdate)
                        ,index li_com_dt_idx (l_commitdate)
                        ,index li_rcpt_dt_idx (l_receiptdate)
) engine=c2db;

show tables;
use mysql;
#
set @fk_name=concat(@cur_db, ".lineitem.lineitem_fk4");
call c2_drop_constraint(@cur_db, "lineitem", @fk_name);
#
set @fk_name=concat(@cur_db, ".lineitem.lineitem_fk2");
call c2_drop_constraint(@cur_db, "lineitem", @fk_name);
```

C. Query Text and Query Output

```
Qualification query 1
--------------
select
      l_returnflag,
      l_linestatus,
      sum(l_quantity) as sum_qty,
      sum(l_extendedprice) as sum_base_price,
      sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
      sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
      avg(l_quantity) as avg_qty,
      avg(l_extendedprice) as avg_price,
      avg(l_discount) as avg_disc,
      count(*) as count_order
from
      lineitem
where
      l_shipdate <= date '1998-12-01' - interval '90' day
group by
      l_returnflag,
      l_linestatus
order by
      l_returnflag,
      l_linestatus
--------------

+--------------+--------------+-------------+-----------------+---------------
----+---------------------+-----------+--------------+----------+-------------
+
| l_returnflag | l_linestatus | sum_qty     | sum_base_price  | sum_disc_price
| sum_charge          | avg_qty   | avg_price    | avg_disc | count_order |
+--------------+--------------+-------------+-----------------+---------------
----+---------------------+-----------+--------------+----------+-------------
+
| A            | F            | 37734107.00 |  56586554400.73 |
53758257134.8700 |  55909065222.827690 | 25.522006 | 38273.129735 | 0.049985 |
1478493 |
| N            | F            |   991417.00 |   1487504710.38 |
1413082168.0541 |   1469649223.194375 | 25.516472 | 38284.467761 | 0.050093 |
38854 |
| N            | O            | 74476040.00 | 111701729697.74 |
106118230307.6056 | 110367043872.496994 | 25.502227 | 38249.117989 | 0.049997
|     2920374 |
| R            | F            | 37719753.00 |  56568041380.90 |
53741292684.6040 |  55889619119.831924 | 25.505794 | 38250.854626 | 0.050009 |
1478870 |
+--------------+--------------+-------------+-----------------+---------------
----+---------------------+-----------+--------------+----------+-------------
+
4 rows in set (1.31 sec)

Bye
Qualification query 2
--------------
select
```

```
      s_acctbal,
      s_name,
      n_name,
      p_partkey,
      p_mfgr,
      s_address,
      s_phone,
      s_comment
from
      part,
      supplier,
      partsupp,
      nation,
      region
where
      p_partkey = ps_partkey
      and s_suppkey = ps_suppkey
      and p_size = 15
      and p_type like '%BRASS'
      and s_nationkey = n_nationkey
      and n_regionkey = r_regionkey
      and r_name = 'EUROPE'
      and ps_supplycost = (
            select
                  min(ps_supplycost)
            from
                  partsupp,
                  supplier,
                  nation,
                  region
            where
                  p_partkey = ps_partkey
                  and s_suppkey = ps_suppkey
                  and s_nationkey = n_nationkey
                  and n_regionkey = r_regionkey
                  and r_name = 'EUROPE'
      )
order by
      s_acctbal desc,
      n_name,
      s_name,
      p_partkey limit 100
--------------

+-----------+--------------------+----------------+-----------+---------------
-+-----------------------------------------+-----------------+---------------
-------------------------------------------------------------------------------
---------+
| s_acctbal | s_name             | n_name         | p_partkey | p_mfgr
| s_address                               | s_phone         | s_comment
|
+-----------+--------------------+----------------+-----------+---------------
-+-----------------------------------------+-----------------+---------------
-------------------------------------------------------------------------------
---------+
|   9938.53 | Supplier#000005359 | UNITED KINGDOM |    185358 | Manufacturer#4
| QKuHYh,vZGiwu2FWEJoLDx04                | 33-429-790-6131 | uriously
regular requests hag
|
|   9937.84 | Supplier#000005969 | ROMANIA        |    108438 | Manufacturer#1
| ANDENSOSmk,miq23Xfb5RWt6dvUcvt6Qa       | 29-520-692-3537 | efully express
```

```
instructions. regular requests against the slyly fin
|
|   9936.22 | Supplier#000005250 | UNITED KINGDOM |       249 | Manufacturer#4
| B3rqp0xbSEim4Mpy2RH J                    | 33-320-228-2957 | etect about the
furiously final accounts. slyly ironic pinto beans sleep inside the furiously
|
|   9923.77 | Supplier#000002324 | GERMANY        |     29821 | Manufacturer#4
| y3OD9UywSTOk                             | 17-779-299-1839 | ackages boost
blithely. blithely regular deposits c
|
|   9871.22 | Supplier#000006373 | GERMANY        |     43868 | Manufacturer#5
| J8fcXWsTqM                               | 17-813-485-8637 | etect blithely
bold asymptotes. fluffily ironic platelets wake furiously; blit
|
|   9870.78 | Supplier#000001286 | GERMANY        |     81285 | Manufacturer#2
| YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH    | 17-516-924-4574 |  regular
accounts. furiously unusual courts above the fi
|
|   9870.78 | Supplier#000001286 | GERMANY        |    181285 | Manufacturer#4
| YKA,E2fjiVd7eUrzp2Ef8j1QxGo2DFnosaTEH    | 17-516-924-4574 |  regular
accounts. furiously unusual courts above the fi
|
|   9852.52 | Supplier#000008973 | RUSSIA         |     18972 | Manufacturer#2
| t5L67YdBYYH6o,Vz24jpDyQ9                 | 32-188-594-7038 | rns wake final
foxes. carefully unusual depende
|
|   9847.83 | Supplier#000008097 | RUSSIA         |    130557 | Manufacturer#2
| xMe97bpE69NzdwLoX                        | 32-375-640-3593 |  the special
excuses. silent sentiments serve carefully final ac
|
|   9847.57 | Supplier#000006345 | FRANCE         |     86344 | Manufacturer#1
| VSt3rzk3qG698u6ld8HhOByvrTcSTSvQlDQDag   | 16-886-766-7945 | ges. slyly
regular requests are. ruthless, express excuses cajole blithely across the unu
|
|   9847.57 | Supplier#000006345 | FRANCE         |    173827 | Manufacturer#2
| VSt3rzk3qG698u6ld8HhOByvrTcSTSvQlDQDag   | 16-886-766-7945 | ges. slyly
regular requests are. ruthless, express excuses cajole blithely across the unu
|
|   9836.93 | Supplier#000007342 | RUSSIA         |      4841 | Manufacturer#4
| JOlK7C1,7xrEZSSOw                        | 32-399-414-5385 | blithely
carefully bold theodolites. fur
|
|   9817.10 | Supplier#000002352 | RUSSIA         |    124815 | Manufacturer#2
| 4LfoHUZjgjEbAKw TgdKcgOc4D4uCYw          | 32-551-831-1437 | wake carefully
alongside of the carefully final ex
|
|   9817.10 | Supplier#000002352 | RUSSIA         |    152351 | Manufacturer#3
| 4LfoHUZjgjEbAKw TgdKcgOc4D4uCYw          | 32-551-831-1437 | wake carefully
alongside of the carefully final ex
|
|   9739.86 | Supplier#000003384 | FRANCE         |    138357 | Manufacturer#2
| o,Z3v4POifevE k9U1b 6J1ucX,I             | 16-494-913-5925 | s after the
furiously bold packages sleep fluffily idly final requests: quickly final
|
|   9721.95 | Supplier#000008757 | UNITED KINGDOM |    156241 | Manufacturer#3
| Atg6GnM4dT2                              | 33-821-407-2995 | eep furiously
sauternes; quickl
|
|   9681.33 | Supplier#000008406 | RUSSIA         |     78405 | Manufacturer#1
| ,qUuXcftUl                               | 32-139-873-8571 | haggle slyly
regular excuses. quic
```

|
| 9643.55 | Supplier#000005148 | ROMANIA | 107617 | Manufacturer#1
| kT4ciVFslx9z4s79p Js825 | 29-252-617-4850 | final excuses.
final ideas boost quickly furiously speci
|
| 9624.82 | Supplier#000001816 | FRANCE | 34306 | Manufacturer#3
| e7vab91vLJPWxxZnewmnDBpDmxYHrb | 16-392-237-6726 | e packages are
around the special ideas. special, pending foxes us
|
| 9624.78 | Supplier#000009658 | ROMANIA | 189657 | Manufacturer#1
| oE9uBgEfSS4opIcepXyAYM,x | 29-748-876-2014 | ronic
asymptotes wake bravely final
|
| 9612.94 | Supplier#000003228 | ROMANIA | 120715 | Manufacturer#2
| KDdpNKN3cWu7ZSrbdqp7AfSLxx,qWB | 29-325-784-8187 | warhorses.
quickly even deposits sublate daringly ironic instructions. slyly blithe t
|
| 9612.94 | Supplier#000003228 | ROMANIA | 198189 | Manufacturer#4
| KDdpNKN3cWu7ZSrbdqp7AfSLxx,qWB | 29-325-784-8187 | warhorses.
quickly even deposits sublate daringly ironic instructions. slyly blithe t
|
| 9571.83 | Supplier#000004305 | ROMANIA | 179270 | Manufacturer#2
| qNHZ7WmCzygwMPRDO9Ps | 29-973-481-1831 | kly carefully
express asymptotes. furiou
|
| 9558.10 | Supplier#000003532 | UNITED KINGDOM | 88515 | Manufacturer#4
| EOeuiiOn21OVpTlGguufFDFsbN1p0lhpxHp | 33-152-301-2164 | foxes. quickly
even excuses use. slyly special foxes nag bl
|
| 9492.79 | Supplier#000005975 | GERMANY | 25974 | Manufacturer#5
| S6mIiCTx82z7lV | 17-992-579-4839 | arefully
pending accounts. blithely regular excuses boost carefully carefully ironic p
|
| 9461.05 | Supplier#000002536 | UNITED KINGDOM | 20033 | Manufacturer#1
| 8mmGbyzaU 7ZS2wJumTibypncu9pNkDc4FYA | 33-556-973-5522 | . slyly regular
deposits wake slyly. furiously regular warthogs are.
|
| 9453.01 | Supplier#000000802 | ROMANIA | 175767 | Manufacturer#1
| ,6HYXb4uaHITmtMBj4Ak57Pd | 29-342-882-6463 | gular frets.
permanently special multipliers believe blithely alongs
|
| 9408.65 | Supplier#000007772 | UNITED KINGDOM | 117771 | Manufacturer#4
| AiC5YAH,gdu0i7 | 33-152-491-1126 | nag against the
final requests. furiously unusual packages cajole blit
|
| 9359.61 | Supplier#000004856 | ROMANIA | 62349 | Manufacturer#5
| HYogcF3Jb yh1 | 29-334-870-9731 | y ironic
theodolites. blithely sile
|
| 9357.45 | Supplier#000006188 | UNITED KINGDOM | 138648 | Manufacturer#1
| g801,ssP8wpTk4Hm | 33-583-607-1633 | ously always
regular packages. fluffily even accounts beneath the furiously final pack
|
| 9352.04 | Supplier#000003439 | GERMANY | 170921 | Manufacturer#4
| qYPDgoiBGhCYxjgC | 17-128-996-4650 | according to
the carefully bold ideas
|
| 9312.97 | Supplier#000007807 | RUSSIA | 90279 | Manufacturer#5
| oGYMPCk9XHGB2PBfKRnHA | 32-673-872-5854 | ecial packages
among the pending, even requests use regula
|

```
|   9312.97 | Supplier#000007807 | RUSSIA         |    100276 | Manufacturer#5
| oGYMPCk9XHGB2PBfKRnHA                    | 32-673-872-5854 | ecial packages
among the pending, even requests use regula
|
|   9280.27 | Supplier#000007194 | ROMANIA        |     47193 | Manufacturer#3
| zhRUQkBSrFYxIAXTfInj vyGRQjeK            | 29-318-454-2133 | o beans haggle
after the furiously unusual deposits. carefully silent dolphins cajole
carefully      |
|   9274.80 | Supplier#000008854 | RUSSIA         |     76346 | Manufacturer#3
| 1xhLoOUM7I3mZ1mKnerw OSqdbb4QbGa         | 32-524-148-5221 | y. courts do
wake slyly. carefully ironic platelets haggle above the slyly regular the
|
|   9249.35 | Supplier#000003973 | FRANCE         |     26466 | Manufacturer#1
| d18GiDsL6Wm2IsGXM,RZf1jCsgZAOjNYVThTRP4  | 16-722-866-1658 | uests are
furiously. regular tithes through the regular, final accounts cajole furiously
above the q |
|   9249.35 | Supplier#000003973 | FRANCE         |     33972 | Manufacturer#1
| d18GiDsL6Wm2IsGXM,RZf1jCsgZAOjNYVThTRP4  | 16-722-866-1658 | uests are
furiously. regular tithes through the regular, final accounts cajole furiously
above the q |
|   9208.70 | Supplier#000007769 | ROMANIA        |     40256 | Manufacturer#5
| rsimdze 5o9P Ht7xS                       | 29-964-424-9649 | lites was
quickly above the furiously ironic requests. slyly even foxes against the
blithely bold    |
|   9201.47 | Supplier#000009690 | UNITED KINGDOM |     67183 | Manufacturer#5
| CB BnUTlmi5zdeEl7R7                      | 33-121-267-9529 | e even, even
foxes. blithely ironic packages cajole regular packages. slyly final ide
|
|   9192.10 | Supplier#000000115 | UNITED KINGDOM |     85098 | Manufacturer#3
| nJ 2t0f7Ve,wL1,6WzGBJLNBUCKlsV           | 33-597-248-1220 | es across the
carefully express accounts boost caref
|
|   9189.98 | Supplier#000001226 | GERMANY        |     21225 | Manufacturer#4
| qsLCqSvLyZfuXIpjz                        | 17-725-903-1381 |  deposits.
blithely bold excuses about the slyly bold forges wake
|
|   9128.97 | Supplier#000004311 | RUSSIA         |    146768 | Manufacturer#5
| I8IjnXd7NSJRs594RxsRR0                   | 32-155-440-7120 | refully.
blithely unusual asymptotes haggle
|
|   9104.83 | Supplier#000008520 | GERMANY        |    150974 | Manufacturer#4
| RqRVDgD0ER J9 b41vR2,3                   | 17-728-804-1793 | ly about the
blithely ironic depths. slyly final theodolites among the fluffily bold ideas
print     |
|   9101.00 | Supplier#000005791 | ROMANIA        |    128254 | Manufacturer#5
| zub2zCV,jhHPPQqi,P2INAjE1zI n66cOEoXFG   | 29-549-251-5384 | ts. notornis
detect blithely above the carefully bold requests. blithely even package
|
|   9094.57 | Supplier#000004582 | RUSSIA         |     39575 | Manufacturer#1
| WB0XkCSG3r,mnQ n,h9VIxjjr9ARHFvKgMDf     | 32-587-577-1351 | jole. regular
accounts sleep blithely frets. final pinto beans play furiously past the
|
|   8996.87 | Supplier#000004702 | FRANCE         |    102191 | Manufacturer#5
| 8XVcQK23akp                              | 16-811-269-8946 | ickly final
packages along the express plat
|
|   8996.14 | Supplier#000009814 | ROMANIA        |    139813 | Manufacturer#2
| af0O5pg83lPU4IDVmEylXZVqYZQzSDlYLAmR     | 29-995-571-8781 |  dependencies
boost quickly across the furiously pending requests! unusual dolphins play sl
|
|   8968.42 | Supplier#000010000 | ROMANIA        |    119999 | Manufacturer#5
```

aTGLEusCiL4F PDBdv665XBJhPyCOB0i	29-578-432-2146	ly regular foxes boost slyly. quickly special waters boost carefully ironi
8936.82	Supplier#000007043	UNITED KINGDOM	109512	Manufacturer#1	FVajceZInZdbJE6Z9XsRUxrUEpiwHDrOXi,1Rz	33-784-177-8208	efully regular courts. furiousl
8929.42	Supplier#000008770	FRANCE	173735	Manufacturer#4	R7cG26TtXrHAP9 HckhfRi	16-242-746-9248	cajole furiously unusual requests. quickly stealthy requests are.
8920.59	Supplier#000003967	ROMANIA	26460	Manufacturer#1	eHoAXe62SY9	29-194-731-3944	aters. express, pending instructions sleep. brave, r
8920.59	Supplier#000003967	ROMANIA	173966	Manufacturer#2	eHoAXe62SY9	29-194-731-3944	aters. express, pending instructions sleep. brave, r
8913.96	Supplier#000004603	UNITED KINGDOM	137063	Manufacturer#2	OUzlvMUr7n,utLxmPNeYKSf3T24OXskxB5	33-789-255-7342	haggle slyly above the furiously regular pinto beans. even
8877.82	Supplier#000007967	FRANCE	167966	Manufacturer#5	A3pi1BARM4nx6R,qrwFoRPU	16-442-147-9345	ously foxes. express, ironic requests im
8862.24	Supplier#000003323	ROMANIA	73322	Manufacturer#3	W9 lYcsC9FwBqk3ItL	29-736-951-3710	ly pending ideas sleep about the furiously unu
8841.59	Supplier#000005750	ROMANIA	100729	Manufacturer#5	Erx3lAgu0g62iaHF9x50uMH4EgeN9hEG	29-344-502-5481	gainst the pinto beans. fluffily unusual dependencies affix slyly even deposits.
8781.71	Supplier#000003121	ROMANIA	13120	Manufacturer#5	wNqTogx238ZYCamFb,50v,bj 4IbNFW9Bvw1xP	29-707-291-5144	s wake quickly ironic ideas
8754.24	Supplier#000009407	UNITED KINGDOM	179406	Manufacturer#4	CHRCbkaWcf5B	33-903-970-9604	e ironic requests. carefully even foxes above the furious
8691.06	Supplier#000004429	UNITED KINGDOM	126892	Manufacturer#2	k,BQms5UhoAF1B2Asi,fLib	33-964-337-5038	efully express deposits kindle after the deposits. final
8655.99	Supplier#000006330	RUSSIA	193810	Manufacturer#2	UozlaENr0ytKe2w6CeIEWFWn iO3S8Rae7Ou	32-561-198-3705	symptotes use about the express dolphins. requests use after the express platelets. final, ex
8638.36	Supplier#000002920	RUSSIA	75398	Manufacturer#1	Je2a8bszf3L	32-122-621-7549	ly quickly ironic requests. even requests whithout t
8638.36	Supplier#000002920	RUSSIA	170402	Manufacturer#3	Je2a8bszf3L	32-122-621-7549	ly quickly ironic requests. even requests whithout t
8607.69	Supplier#000006003	UNITED KINGDOM	76002	Manufacturer#2	EH9wADcEiuenM0NR08zDwMidw,52Y2RyILEiA	33-416-807-5206	ar, pending

```
accounts. pending depende
|
|   8569.52 | Supplier#000005936 | RUSSIA         |      5935 | Manufacturer#5
| jXaNZ6vwnEWJ2ksLZJpjtgt0bY2a3AU          | 32-644-251-7916 | . regular foxes
nag carefully atop the regular, silent deposits. quickly regular packages
|
|   8564.12 | Supplier#000000033 | GERMANY        |    110032 | Manufacturer#1
| gfeKpYw3400L0SDywXA6Ya1Qmq1w6YB9f3R      | 17-138-897-9374 | n sauternes
along the regular asymptotes are regularly along the
|
|   8553.82 | Supplier#000003979 | ROMANIA        |    143978 | Manufacturer#4
| BfmVhCAnCMY3jzpjUMy4CNWs9 HzpdQR7INJU    | 29-124-646-4897 | ic requests
wake against the blithely unusual accounts. fluffily r
|
|   8517.23 | Supplier#000009529 | RUSSIA         |     37025 | Manufacturer#5
| e44R8o7JAIS9iMcr                         | 32-565-297-8775 | ove the even
courts. furiously special platelets
|
|   8517.23 | Supplier#000009529 | RUSSIA         |     59528 | Manufacturer#2
| e44R8o7JAIS9iMcr                         | 32-565-297-8775 | ove the even
courts. furiously special platelets
|
|   8503.70 | Supplier#000006830 | RUSSIA         |     44325 | Manufacturer#4
| BC4WFCYRUZyaIgchU 4S                     | 32-147-878-5069 | pades cajole.
furious packages among the carefully express excuses boost furiously across th
|
|   8457.09 | Supplier#000009456 | UNITED KINGDOM |     19455 | Manufacturer#1
| 7SBhZs8gP1cJjT0Qf433YBk                  | 33-858-440-4349 | cing requests
along the furiously unusual deposits promise among the furiously unus
|
|   8441.40 | Supplier#000003817 | FRANCE         |    141302 | Manufacturer#2
| hU3fz3xL78                               | 16-339-356-5115 | ely even ideas.
ideas wake slyly furiously unusual instructions. pinto beans sleep ag
|
|   8432.89 | Supplier#000003990 | RUSSIA         |    191470 | Manufacturer#1
| wehBBp1RQbfxAYDASS75MsywmsKHRVdkrvNe6m   | 32-839-509-9301 | ep furiously.
packages should have to haggle slyly across the deposits. furiously regu
|
|   8431.40 | Supplier#000002675 | ROMANIA        |      5174 | Manufacturer#1
| HJFStOu9R5NGPOegKhgbzBdyvrG2yh8w         | 29-474-643-1443 | ithely express
pinto beans. blithely even foxes haggle. furiously regular theodol
|
|   8407.04 | Supplier#000005406 | RUSSIA         |    162889 | Manufacturer#4
| j7 gYF5RW8DC5UrjKC                       | 32-626-152-4621 | r the blithely
regular packages. slyly ironic theodoli
|
|   8386.08 | Supplier#000008518 | FRANCE         |     36014 | Manufacturer#3
| 2jqzqqAVe9crMVGP,n9nTsQXulNLTUYoJjEDcqWV | 16-618-780-7481 | blithely bold
pains are carefully platelets. finally regular pinto beans sleep carefully
special     |
|   8376.52 | Supplier#000005306 | UNITED KINGDOM |    190267 | Manufacturer#5
| 9t8Y8 QqSIsoADPt6NLdk,TP5zyRx41oBUlgoGc9 | 33-632-514-7931 | ly final
accounts sleep special, regular requests. furiously regular
|
|   8348.74 | Supplier#000008851 | FRANCE         |     66344 | Manufacturer#4
| nWxi7GwEbjhw1                            | 16-796-240-2472 |  boldly final
deposits. regular, even instructions detect slyly. fluffily unusual pinto bea
|
|   8338.58 | Supplier#000007269 | FRANCE         |     17268 | Manufacturer#4
| ZwhJSwABUoiB04,3                         | 16-267-277-4365 | iously final
accounts. even pinto beans cajole slyly regular
```

```
|
|   8328.46 | Supplier#000001744 | ROMANIA        |     69237 | Manufacturer#5
| oLo3fV64q2,FKHa3p,qHnS7Yzv,ps8           | 29-330-728-5873 | ep carefully--
even, careful packages are slyly along t
|
|   8307.93 | Supplier#000003142 | GERMANY        |     18139 | Manufacturer#1
| dqblvV8dCNAorGlJ                         | 17-595-447-6026 | olites wake
furiously regular decoys. final requests nod
|
|   8231.61 | Supplier#000009558 | RUSSIA         |    192000 | Manufacturer#2
| mcdgen,yT1iJDHDS5fV                      | 32-762-137-5858 |  foxes
according to the furi
|
|   8152.61 | Supplier#000002731 | ROMANIA        |     15227 | Manufacturer#4
|  nluXJCuY1tu                             | 29-805-463-2030 |  special
requests. even, regular warhorses affix among the final gr
|
|   8109.09 | Supplier#000009186 | FRANCE         |     99185 | Manufacturer#1
| wgfosrVPexl9pEXWywaqlBMDYYf              | 16-668-570-1402 | tions haggle
slyly about the sil
|
|   8102.62 | Supplier#000003347 | UNITED KINGDOM |     18344 | Manufacturer#5
| m CtXS2S16i                              | 33-454-274-8532 | egrate with the
slyly bold instructions. special foxes haggle silently among the
|
|   8046.07 | Supplier#000008780 | FRANCE         |    191222 | Manufacturer#3
| AczzuE0UK9osj ,Lx0Jmh                    | 16-473-215-6395 | onic platelets
cajole after the regular instructions. permanently bold excuses
|
|   8042.09 | Supplier#000003245 | RUSSIA         |    135705 | Manufacturer#4
| Dh8Ikg39onrbOL4DyTfGw8a9oKUX3d9Y         | 32-836-132-8872 | osits. packages
cajole slyly. furiously regular deposits cajole slyly. q
|
|   8042.09 | Supplier#000003245 | RUSSIA         |    150729 | Manufacturer#1
| Dh8Ikg39onrbOL4DyTfGw8a9oKUX3d9Y         | 32-836-132-8872 | osits. packages
cajole slyly. furiously regular deposits cajole slyly. q
|
|   7992.40 | Supplier#000006108 | FRANCE         |    118574 | Manufacturer#1
| 8tBydnTDwUqfBfFV4l3                      | 16-974-998-8937 |  ironic ideas?
fluffily even instructions wake. blithel
|
|   7980.65 | Supplier#000001288 | FRANCE         |     13784 | Manufacturer#4
| zE,7HgVPrCn                              | 16-646-464-8247 | ully bold
courts. escapades nag slyly. furiously fluffy theodo
|
|   7950.37 | Supplier#000008101 | GERMANY        |     33094 | Manufacturer#5
| kkYvL6IuvojJgTNG IKkaXQDYgx8ILohj        | 17-627-663-8014 | arefully
unusual requests x-ray above the quickly final deposits.
|
|   7937.93 | Supplier#000009012 | ROMANIA        |     83995 | Manufacturer#2
| iUiTziH,Ek3i4lwSgunXMgrcTzwdb            | 29-250-925-9690 | to the blithely
ironic deposits nag sly
|
|   7914.45 | Supplier#000001013 | RUSSIA         |    125988 | Manufacturer#2
| riRcntps4KEDtYScjpMIWeYF6mNnR            | 32-194-698-3365 |  busily bold
packages are dolphi
|
|   7912.91 | Supplier#000004211 | GERMANY        |    159180 | Manufacturer#5
| 2wQRVovHrm3,v03IKzfTd,1PYsFXQFFOG        | 17-266-947-7315 | ay furiously
regular platelets. cou
|
```

```
|   7912.91 | Supplier#000004211 | GERMANY        |    184210 | Manufacturer#4
| 2wQRVovHrm3,v03IKzfTd,1PYsFXQFFOG        | 17-266-947-7315 | ay furiously
regular platelets. cou
|
|   7894.56 | Supplier#000007981 | GERMANY        |     85472 | Manufacturer#4
| NSJ96vMROAbeXP                           | 17-963-404-3760 | ic platelets
affix after the furiously
|
|   7887.08 | Supplier#000009792 | GERMANY        |    164759 | Manufacturer#3
| Y28ITVeYriT3kIGdV2K8fSZ V2UqT5H1Otz      | 17-988-938-4296 | ckly around the
carefully fluffy theodolites. slyly ironic pack
|
|   7871.50 | Supplier#000007206 | RUSSIA         |    104695 | Manufacturer#1
| 3w fNCnrVmvJjE95sgWZzvW                  | 32-432-452-7731 | ironic
requests. furiously final theodolites cajole. final, express packages sleep.
quickly reg      |
|   7852.45 | Supplier#000005864 | RUSSIA         |      8363 | Manufacturer#4
| WCNfBPZeSXh3h,c                          | 32-454-883-3821 | usly unusual
pinto beans. brave ideas sleep carefully quickly ironi
|
|   7850.66 | Supplier#000001518 | UNITED KINGDOM |     86501 | Manufacturer#1
| ONda3YJiHKJOC                            | 33-730-383-3892 | ifts haggle
fluffily pending pai
|
|   7843.52 | Supplier#000006683 | FRANCE         |     11680 | Manufacturer#4
| 2Z0JGkiv01Y00oCFwUGfviIbhzCdy            | 16-464-517-8943 |  express, final
pinto beans x-ray slyly asymptotes. unusual, unusual
|
+-----------+--------------------+----------------+-----------+---------------
-+-----------------------------------------+-----------------+---------------
--------------------------------------------------------------------------------
---------+
100 rows in set (0.84 sec)

Bye
Qualification query 3
--------------
select
      l_orderkey,
      sum(l_extendedprice * (1 - l_discount)) as revenue,
      o_orderdate,
      o_shippriority
from
      customer,
      orders,
      lineitem
where
      c_mktsegment = 'BUILDING'
      and c_custkey = o_custkey
      and l_orderkey = o_orderkey
      and o_orderdate < date '1995-03-15'
      and l_shipdate > date '1995-03-15'
group by
      l_orderkey,
      o_orderdate,
      o_shippriority
order by
      revenue desc,
      o_orderdate limit 10
--------------
```

```
+------------+-------------+-------------+----------------+
| l_orderkey | revenue     | o_orderdate | o_shippriority |
+------------+-------------+-------------+----------------+
|    2456423 | 406181.0111 | 1995-03-05  |              0 |
|    3459808 | 405838.6989 | 1995-03-04  |              0 |
|     492164 | 390324.0610 | 1995-02-19  |              0 |
|    1188320 | 384537.9359 | 1995-03-09  |              0 |
|    2435712 | 378673.0558 | 1995-02-26  |              0 |
|    4878020 | 378376.7952 | 1995-03-12  |              0 |
|    5521732 | 375153.9215 | 1995-03-13  |              0 |
|    2628192 | 373133.3094 | 1995-02-22  |              0 |
|     993600 | 371407.4595 | 1995-03-05  |              0 |
|    2300070 | 367371.1452 | 1995-03-13  |              0 |
+------------+-------------+-------------+----------------+
10 rows in set (0.14 sec)

Bye
Qualification query 4
--------------
select
      o_orderpriority,
      count(*) as order_count
from
      orders
where
      o_orderdate >= date '1993-07-01'
      and o_orderdate < date '1993-07-01' + interval '3' month
      and exists (
            select
                  *
            from
                  lineitem
            where
                  l_orderkey = o_orderkey
                  and l_commitdate < l_receiptdate
      )
group by
      o_orderpriority
order by
      o_orderpriority
--------------

+-----------------+-------------+
| o_orderpriority | order_count |
+-----------------+-------------+
| 1-URGENT        |       10594 |
| 2-HIGH          |       10476 |
| 3-MEDIUM        |       10410 |
| 4-NOT SPECIFIED |       10556 |
| 5-LOW           |       10487 |
+-----------------+-------------+
5 rows in set (0.16 sec)

Bye
Qualification query 5
--------------
select
      n_name,
      sum(l_extendedprice * (1 - l_discount)) as revenue
from
      customer,
```

```
      orders,
      lineitem,
      supplier,
      nation,
      region
where
      c_custkey = o_custkey
      and l_orderkey = o_orderkey
      and l_suppkey = s_suppkey
      and c_nationkey = s_nationkey
      and s_nationkey = n_nationkey
      and n_regionkey = r_regionkey
      and r_name = 'ASIA'
      and o_orderdate >= date '1994-01-01'
      and o_orderdate < date '1994-01-01' + interval '1' year
group by
      n_name
order by
      revenue desc
--------------

+-----------+---------------+
| n_name    | revenue       |
+-----------+---------------+
| INDONESIA | 55502041.1697 |
| VIETNAM   | 55295086.9967 |
| CHINA     | 53724494.2566 |
| INDIA     | 52035512.0002 |
| JAPAN     | 45410175.6954 |
+-----------+---------------+
5 rows in set (0.08 sec)

Bye
Qualification query 6
--------------
select
      sum(l_extendedprice * l_discount) as revenue
from
      lineitem
where
      l_shipdate >= date '1994-01-01'
      and l_shipdate < date '1994-01-01' + interval '1' year
      and l_discount between .06 - 0.01 and .06 + 0.01
      and l_quantity < 24
--------------

+----------------+
| revenue        |
+----------------+
| 123141078.2283 |
+----------------+
1 row in set (0.04 sec)

Bye
Qualification query 7
--------------
select
      supp_nation,
      cust_nation,
      l_year,
      sum(volume) as revenue
```

```
from
      (
            select
                  n1.n_name as supp_nation,
                  n2.n_name as cust_nation,
                  extract(year from l_shipdate) as l_year,
                  l_extendedprice * (1 - l_discount) as volume
            from
                  supplier,
                  lineitem,
                  orders,
                  customer,
                  nation n1,
                  nation n2
            where
                  s_suppkey = l_suppkey
                  and o_orderkey = l_orderkey
                  and c_custkey = o_custkey
                  and s_nationkey = n1.n_nationkey
                  and c_nationkey = n2.n_nationkey
                  and (
                        (n1.n_name = 'FRANCE' and n2.n_name = 'GERMANY')
                        or (n1.n_name = 'GERMANY' and n2.n_name = 'FRANCE')
                  )
                  and l_shipdate between date '1995-01-01' and date '1996-12-
31'
      ) as shipping
group by
      supp_nation,
      cust_nation,
      l_year
order by
      supp_nation,
      cust_nation,
      l_year
--------------

+-------------+-------------+--------+---------------+
| supp_nation | cust_nation | l_year | revenue       |
+-------------+-------------+--------+---------------+
| FRANCE      | GERMANY     |   1995 | 54639732.7336 |
| FRANCE      | GERMANY     |   1996 | 54633083.3076 |
| GERMANY     | FRANCE      |   1995 | 52531746.6697 |
| GERMANY     | FRANCE      |   1996 | 52520549.0224 |
+-------------+-------------+--------+---------------+
4 rows in set (0.16 sec)

Bye
Qualification query 8
--------------
select
      o_year,
      sum(case
            when nation = 'BRAZIL' then volume
            else 0
      end) / sum(volume) as mkt_share
from
      (
            select
                  extract(year from o_orderdate) as o_year,
                  l_extendedprice * (1 - l_discount) as volume,
```

```
                  n2.n_name as nation
            from
                  part,
                  supplier,
                  lineitem,
                  orders,
                  customer,
                  nation n1,
                  nation n2,
                  region
            where
                  p_partkey = l_partkey
                  and s_suppkey = l_suppkey
                  and l_orderkey = o_orderkey
                  and o_custkey = c_custkey
                  and c_nationkey = n1.n_nationkey
                  and n1.n_regionkey = r_regionkey
                  and r_name = 'AMERICA'
                  and s_nationkey = n2.n_nationkey
                  and o_orderdate between date '1995-01-01' and date '1996-12-
31'
                  and p_type = 'ECONOMY ANODIZED STEEL'
      ) as all_nations
group by
      o_year
order by
      o_year
--------------

+--------+------------+
| o_year | mkt_share  |
+--------+------------+
|   1995 | 0.03443589 |
|   1996 | 0.04148552 |
+--------+------------+
2 rows in set (0.31 sec)

Bye
Qualification query 9
--------------
select
      nation,
      o_year,
      sum(amount) as sum_profit
fro]m
      (
            select
                  n_name as nation,
                  extract(year from o_orderdate) as o_year,
                  l_extendedprice * (1 - l_discount) - ps_supplycost *
l_quantity as amount
            from
                  part,
                  supplier,
                  lineitem,
                  partsupp,
                  orders,
                  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%'
      ) as profit
group by
      nation,
      o_year
order by
      nation,
      o_year desc
--------------
```

nation	o_year	sum_profit
ALGERIA	1998	31342867.2345
ALGERIA	1997	57138193.0233
ALGERIA	1996	56140140.1330
ALGERIA	1995	53051469.6534
ALGERIA	1994	53867582.1286
ALGERIA	1993	54942718.1324
ALGERIA	1992	54628034.7127
ARGENTINA	1998	30211185.7081
ARGENTINA	1997	50805741.7523
ARGENTINA	1996	51923746.5755
ARGENTINA	1995	49298625.7666
ARGENTINA	1994	50835610.1095
ARGENTINA	1993	51646079.1775
ARGENTINA	1992	50410314.9948
BRAZIL	1998	27217924.3832
BRAZIL	1997	48378669.1989
BRAZIL	1996	50482870.3572
BRAZIL	1995	47623383.6349
BRAZIL	1994	47840165.7256
BRAZIL	1993	49054694.0351
BRAZIL	1992	48667639.0842
CANADA	1998	30379833.7685
CANADA	1997	50465052.3114
CANADA	1996	52560501.3904
CANADA	1995	52375332.8092
CANADA	1994	52600364.6587
CANADA	1993	52644504.0735
CANADA	1992	53932871.6970
CHINA	1998	31075466.1649
CHINA	1997	50551874.4499
CHINA	1996	51039293.8754
CHINA	1995	49287534.6169
CHINA	1994	50851090.0674
CHINA	1993	54229629.8330
CHINA	1992	52400529.3720
EGYPT	1998	29054433.3856
EGYPT	1997	50627611.4524
EGYPT	1996	49542212.8446
EGYPT	1995	48311550.3207
EGYPT	1994	49790644.7360
EGYPT	1993	48904292.9692
EGYPT	1992	49434932.6192
ETHIOPIA	1998	28040717.2674
ETHIOPIA	1997	47455009.8664

ETHIOPIA	1996	46491097.5729
ETHIOPIA	1995	46804449.3009
ETHIOPIA	1994	48516143.9170
ETHIOPIA	1993	46551891.5629
ETHIOPIA	1992	44934648.6428
FRANCE	1998	32226407.8392
FRANCE	1997	47121485.8602
FRANCE	1996	47263135.4960
FRANCE	1995	47275997.5707
FRANCE	1994	47067209.3315
FRANCE	1993	51163370.1062
FRANCE	1992	47846235.3313
GERMANY	1998	28624942.6598
GERMANY	1997	49309074.8773
GERMANY	1996	49918683.1684
GERMANY	1995	52650718.7242
GERMANY	1994	50346900.4226
GERMANY	1993	50991895.8059
GERMANY	1992	48274126.0991
INDIA	1998	29943144.3544
INDIA	1997	50665453.2312
INDIA	1996	50283092.2912
INDIA	1995	50006774.6446
INDIA	1994	48995190.7556
INDIA	1993	50286902.8525
INDIA	1992	50850329.4023
INDONESIA	1998	27672339.9965
INDONESIA	1997	50512145.7256
INDONESIA	1996	51653060.1166
INDONESIA	1995	51508779.5940
INDONESIA	1994	52817950.3218
INDONESIA	1993	47959994.9551
INDONESIA	1992	51776605.0323
IRAN	1998	29065736.2381
IRAN	1997	50042063.0545
IRAN	1996	50926653.1881
IRAN	1995	51249667.6485
IRAN	1994	50337085.8650
IRAN	1993	51730763.4902
IRAN	1992	49955856.5634
IRAQ	1998	31624551.0017
IRAQ	1997	55121749.0195
IRAQ	1996	55897663.7937
IRAQ	1995	54815472.5170
IRAQ	1994	54408516.1266
IRAQ	1993	53633167.9770
IRAQ	1992	55891939.3396
JAPAN	1998	27934179.6695
JAPAN	1997	44517162.5463
JAPAN	1996	42545606.1204
JAPAN	1995	43749356.3996
JAPAN	1994	44840243.0700
JAPAN	1993	44660015.5328
JAPAN	1992	45410249.1216
JORDAN	1998	26901488.5781
JORDAN	1997	45471878.4099
JORDAN	1996	46794325.7918
JORDAN	1995	45178828.5760
JORDAN	1994	45333636.5076
JORDAN	1993	47971496.0979
JORDAN	1992	44717239.1774

KENYA	1998	28597614.3370
KENYA	1997	47949733.7274
KENYA	1996	46886924.6229
KENYA	1995	46072338.7552
KENYA	1994	45772061.1711
KENYA	1993	46308728.2345
KENYA	1992	47257780.8406
MOROCCO	1998	26732115.5800
MOROCCO	1997	45637304.2499
MOROCCO	1996	45558221.7451
MOROCCO	1995	47851318.8874
MOROCCO	1994	46272172.9445
MOROCCO	1993	46764326.1818
MOROCCO	1992	48122783.5831
MOZAMBIQUE	1998	30712392.0112
MOZAMBIQUE	1997	50316528.7625
MOZAMBIQUE	1996	51640320.2509
MOZAMBIQUE	1995	50693774.5060
MOZAMBIQUE	1994	49253277.6260
MOZAMBIQUE	1993	49153016.5367
MOZAMBIQUE	1992	48247551.8503
PERU	1998	29326102.3196
PERU	1997	49753780.3951
PERU	1996	50935170.2927
PERU	1995	53309883.4072
PERU	1994	50643531.7968
PERU	1993	51584622.0020
PERU	1992	47523899.0547
ROMANIA	1998	30368667.3999
ROMANIA	1997	50365683.8526
ROMANIA	1996	49598999.0148
ROMANIA	1995	47537642.8697
ROMANIA	1994	51455283.0095
ROMANIA	1993	50407136.8922
ROMANIA	1992	48185385.1286
RUSSIA	1998	28322384.0266
RUSSIA	1997	50106685.1822
RUSSIA	1996	51753342.4302
RUSSIA	1995	49215820.3647
RUSSIA	1994	52205666.4407
RUSSIA	1993	51860230.0340
RUSSIA	1992	53251677.1530
SAUDI ARABIA	1998	31541259.8100
SAUDI ARABIA	1997	52438750.8076
SAUDI ARABIA	1996	52543737.8197
SAUDI ARABIA	1995	52938696.5331
SAUDI ARABIA	1994	51389601.9668
SAUDI ARABIA	1993	52937508.8818
SAUDI ARABIA	1992	54843459.6414
UNITED KINGDOM	1998	28494874.0040
UNITED KINGDOM	1997	49381810.8986
UNITED KINGDOM	1996	51386853.9604
UNITED KINGDOM	1995	51509586.7885
UNITED KINGDOM	1994	48086499.7115
UNITED KINGDOM	1993	49166827.2235
UNITED KINGDOM	1992	49349122.0825
UNITED STATES	1998	25126238.9461
UNITED STATES	1997	50077306.4186
UNITED STATES	1996	48048649.4703
UNITED STATES	1995	48809032.4226
UNITED STATES	1994	49296747.1827

```
| UNITED STATES  |   1993 | 48029946.8014 |
| UNITED STATES  |   1992 | 48671944.4983 |
| VIETNAM        |   1998 | 30442736.0594 |
| VIETNAM        |   1997 | 50309179.7942 |
| VIETNAM        |   1996 | 50488161.4100 |
| VIETNAM        |   1995 | 49658284.6125 |
| VIETNAM        |   1994 | 50596057.2607 |
| VIETNAM        |   1993 | 50953919.1519 |
| VIETNAM        |   1992 | 49613838.3151 |
+----------------+--------+---------------+
175 rows in set (0.38 sec)

Bye
```

D. Seed and Query Substitution Parameters

a. Seed values and parameters

```
Streams 0: rand_seed = 419025520
Streams 1: rand_seed = 419025521
Streams 2: rand_seed = 419025522
Streams 3: rand_seed = 419025523
Streams 4: rand_seed = 419025524
Streams 5: rand_seed = 419025525
Streams 6: rand_seed = 419025526

Streams 0: rand_seed = 419025520
14    1995-01-01
2     3     COPPER      EUROPE
9     rose
20    medium      1997-01-01  ARGENTINA
6     1996-01-01  0.09  25
17    Brand#33    LG PKG
18    313
8     PERU  AMERICA     ECONOMY PLATED COPPER
21    RUSSIA
13    unusual     requests
3     AUTOMOBILE  1995-03-22
22    13    11    17    20    14    34    30
16    Brand#43    MEDIUM POLISHED   17    7     3     15    27    16    5
      14
4     1996-06-01
11    IRAN  0.0000003333
15    1997-01-01
1     64
10    1993-12-01
19    Brand#23    Brand#51    Brand#12    1     11    22
5     MIDDLE EAST 1996-01-01
7     UNITED STATES     PERU
12    AIR   MAIL  1995-01-01

Streams 1: rand_seed = 419025521
21    MOROCCO
3     FURNITURE   1995-03-08
```

```
18    315
5     AFRICA      1997-01-01
11    UNITED KINGDOM    0.0000003333
7     MOZAMBIQUE  INDONESIA
6     1997-01-01  0.07  24
20    violet      1996-01-01  MOROCCO
17    Brand#35    MED CASE
12    REG AIR     MAIL  1995-01-01
16    Brand#23    ECONOMY BRUSHED   21    20    9     32    49    8     36
      39
15    1995-01-01
13    unusual     requests
10    1994-09-01
2     41    BRASS AMERICA
8     INDONESIA   ASIA  ECONOMY ANODIZED COPPER
14    1995-01-01
19    Brand#35    Brand#44    Brand#51    6     12    29
9     pink
22    25    34    13    17    22    32    19
1     72
4     1994-03-01

Streams 2: rand_seed = 419025522
6     1997-01-01  0.04  24
17    Brand#32    MED BAG
14    1995-01-01
16    Brand#13    SMALL BURNISHED   46    19    39    17    7     50    45
      8
19    Brand#32    Brand#22    Brand#55    1     13    25
10    1993-06-01
9     orange
2     28    NICKEL      EUROPE
15    1993-01-01
8     ARGENTINA   AMERICA     LARGE POLISHED COPPER
5     AMERICA     1997-01-01
22    17    10    13    24    14    27    26
12    SHIP  MAIL  1995-01-01
7     INDIA ARGENTINA
13    unusual     accounts
18    312
1     80
4     1996-10-01
20    grey  1994-01-01  ETHIOPIA
3     MACHINERY   1995-03-24
11    IRAQ  0.0000003333
21    GERMANY

Streams 3: rand_seed = 419025523
8     CHINA ASIA  LARGE BURNISHED COPPER
5     ASIA  1997-01-01
4     1994-07-01
6     1997-01-01  0.02  25
17    Brand#33    MED PKG
7     ALGERIA     CHINA
1     88
18    314
22    30    16    34    25    15    19    28
14    1996-01-01
9     mint
10    1994-04-01
15    1995-01-01
```

```
11    UNITED STATES     0.0000003333
20    royal 1997-01-01  SAUDI ARABIA
2     16    TIN   AMERICA
21    UNITED STATES
19    Brand#34    Brand#55    Brand#55    7     14    21
13    unusual     accounts
16    Brand#43    LARGE PLATED      46    45    49    10    25    26    31
      5
12    FOB   MAIL  1995-01-01
3     FURNITURE   1995-03-10

Streams 4: rand_seed = 419025524
5     EUROPE      1997-01-01
21    MOZAMBIQUE
14    1996-01-01
19    Brand#42    Brand#42    Brand#44    2     15    29
15    1993-01-01
17    Brand#35    JUMBO CASE
12    MAIL  FOB   1996-01-01
6     1997-01-01  0.07  24
4     1997-01-01
9     linen
8     IRAN  MIDDLE EAST LARGE ANODIZED TIN
16    Brand#23    PROMO BRUSHED     12    11    9     20    6     24    29
      27
11    IRAQ  0.0000003333
2     4     COPPER      MIDDLE EAST
10    1995-01-01
18    315
1     96
13    unusual     accounts
7     PERU  IRAN
22    18    28    15    13    12    25    17
3     MACHINERY   1995-03-26
20    cream 1996-01-01  INDONESIA

Streams 5: rand_seed = 419025525
21    INDIA
15    1996-01-01
4     1994-10-01
6     1993-01-01  0.05  24
7     INDONESIA   BRAZIL
16    Brand#13    MEDIUM ANODIZED   34    29    3     22    5     44    40
      41
19    Brand#44    Brand#25    Brand#43    7     16    25
18    313
14    1996-01-01
22    34    24    23    15    25    22    12
11    UNITED STATES     0.0000003333
13    express     accounts
3     BUILDING    1995-03-12
1     104
2     42    STEEL AMERICA
5     AFRICA      1993-01-01
8     BRAZIL      AMERICA     MEDIUM POLISHED TIN
20    olive 1994-01-01  UNITED STATES
12    RAIL  FOB   1996-01-01
17    Brand#32    JUMBO BAG
10    1993-10-01
9     lace
```

```
Streams 6: rand_seed = 419025526
10    1994-07-01
3     HOUSEHOLD   1995-03-28
15    1993-01-01
13    express     accounts
6     1993-01-01  0.02  25
8     ROMANIA     EUROPE      MEDIUM BURNISHED TIN
9     grey
7     ARGENTINA   ROMANIA
4     1997-05-01
11    JAPAN 0.0000003333
22    12    25    19    15    28    14    23
18    314
12    AIR   FOB   1996-01-01
1     112
5     AMERICA     1993-01-01
16    Brand#43    ECONOMY PLATED    4     40    3     29    31    33    48
      21
2     30    NICKEL      MIDDLE EAST
14    1996-01-01
19    Brand#41    Brand#13    Brand#42    2     17    21
20    beige 1993-01-01  KENYA
17    Brand#34    JUMBO PKG
21    ALGERIA
```

E. Implementation-specific Layer/Driver Code

All such code can be found in Appendix B.

F. Miscellaneous Database Scripts

```
--    File:    dbinserts.SQL
--
--    2005-10-24 Modified by Lorna Livingtree
---   Corrected for use with foreign key constraints
---
---   9:46 PM 10/25/2005 Modified by Mike Fitzner
---   Changed insert and delete order to reduce runtime
---


select now() 'Beginning timestamp for dbinserts:';

use tpch300g;

create temporary table temp_part as select * from part
      where P_partKEY = 1
;
update temp_part
      set P_partKEY = 2147483647
;
insert into part select * from temp_part
;

select * from part
      where P_partKEY = 2147483647
      or P_partKEY = 1
;
DROP TABLE temp_part
;


create temporary table temp_supplier as select * from supplier
      where S_suppKEY= 1
;


UPDATE temp_supplier set S_suppKEY= 2147483647
;


insert into supplier select * from temp_supplier
;
select * from supplier
      where S_suppKEY = 2147483647
      or S_suppKEY = 1
;

create temporary table temp_partsupp as select * from partsupp
      where PS_partKEY = 1
      and PS_suppKEY = 2
;
```

```
update temp_partsupp
      set PS_partKEY = 2147483647,
          PS_suppKEY = 2147483647
;


insert into partsupp select * from temp_partsupp
;
select * from partsupp
      where (PS_partKEY = 2147483647
      and    PS_suppKEY = 2147483647)
      or    (PS_partKEY = 1
      and    PS_suppKEY = 2)
;

create temporary table temp_customer as select * from customer
      where C_CUSTKEY = 1
;


update temp_customer set C_CUSTKEY = 2147483647
;


insert into customer select * from temp_customer
;
select * from customer
      where C_CUSTKEY = 2147483647
      or C_CUSTKEY = 1
;

create temporary table temp_orders as select * from orders
      where O_ORDERKEY = (select min(O_ORDERKEY) from orders)
;


update temp_orders set O_ORDERKEY = 2147483647

;

insert into orders select * from temp_orders

;
select * from orders
      where O_ORDERKEY = 2147483647
      or O_ORDERKEY = (select min(O_ORDERKEY) from orders)
;


##create temporary table temp_lineitem as select * from lineitem
##    where L_ORDERKEY = (select min(O_ORDERKEY) from orders) order by 1,2
limit 1
##/*  and   L_LINENUMBER = 1    */
##;

create temporary table temp_lineitem as select * from lineitem
      where L_ORDERKEY = (select min(O_ORDERKEY) from orders) and
      L_LINENUMBER = 1;
```

```
update temp_lineitem
      set L_ORDERKEY = 2147483647,
          L_partKEY = 2147483647,
          L_suppKEY = 2147483647,
          L_LINENUMBER = -2147483646
;

insert into lineitem select * from temp_lineitem
;
select * from lineitem
      where (L_ORDERKEY = 2147483647
      and    L_partKEY  = 2147483647
      and    L_suppKEY = 2147483647
      and    L_LINENUMBER = -2147483646)
      or    (L_ORDERKEY = (select min(O_ORDERKEY) from orders)
      and    L_LINENUMBER = 1)
;



create temporary table temp_nation as select * from nation
      where N_nationKEY = 1
;


update temp_nation
      set N_nationKEY = 2147483647
;


insert into nation select * from temp_nation
;
select * from nation
      where N_nationKEY = 2147483647
      or N_nationKEY = 1
;


create temporary table temp_region as select * from region
      where R_regionKEY = 1
;


update temp_region
      set R_regionKEY = 2147483647
;

insert into region select * from temp_region
;
select * from region
      where R_regionKEY = 2147483647
      or R_regionKEY = 1
;


--- ============================================================
--- Duplicates finished starting inserts for domain range
--- ============================================================

## select convert(char(30), getdate(),21)
```

```
delete from region where R_REGIONKEY = 2147483647  -- Prevent Primary Key
violation
;


insert into region
      (R_regionKEY, R_NAME, R_COMMENT)
      values
      (2147483647,
      'Ze ends of the earth....E',
      'A reasonable comment would ; herE')
;

select * from region where R_REGIONKEY = 2147483647
;



delete from nation where N_NATIONKEY = 2147483647;

  -- Prevent Primary Key violation



insert into nation
      (N_nationKEY, N_NAME, N_regionKEY, N_COMMENT)
      values
      (2147483647,
      'Ze Republic d MakebelievE',
      2147483647,
      'A nation comment for field size 152 no E')
;

select * from nation where N_NATIONKEY = 2147483647 and N_regionKEY =
2147483647
;


delete from lineitem                          -- prevent Foreign Key violation
      where L_ORDERKEY = 2147483647
      and   L_partKEY = 2147483647
      and   L_suppKEY = 2147483647
      and   L_LINENUMBER = -2147483646
;

delete from partsupp                          -- prevent Foreign Key violation
      where PS_partKEY = 2147483647
      and   PS_suppKEY = 2147483647
;

delete from supplier where S_suppKEY = 2147483647  -- prevent Foreign Key
violation
;


insert into supplier
      (S_suppKEY, S_NAME, S_ADDRESS, S_nationKEY, S_PHONE,
      S_ACCTBAL, S_COMMENT)
      values
      (2147483647, 'NAME text ............25E',
```

```
      'Address varchar ............30.......40E',
      2147483647,'This is phone E', 123456789012,
      'Supplier comment field is 101 long no E')
;
select * from supplier
      where S_suppKEY = 2147483647
;


delete from customer where C_CUSTKEY = 2147483647     -- prevent Primary Key
violation
;



insert into customer
      (C_CUSTKEY, C_NAME, C_ADDRESS, C_nationKEY,
      C_PHONE, C_ACCTBAL, C_MKTSEGMENT, C_COMMENT)
      values
      (2147483647, 'Customer Name ;es to 25E',
      'Customer Address ;es here..3.........4E',
      2147483647, 'This is phone E', 123456789012,
      'Markt segE', 'Customer comments fiels is 117 long no E')
;
select * from customer
      where C_CUSTKEY = 2147483647
;


delete from part where P_PARTKEY = 2147483647   -- prevent Primary Key
violation
;



insert into part
      (P_partKEY, P_NAME, P_MFGR, P_BRAND, P_TYPE,
      P_SIZE, P_CONTAINER, P_RETAILPRICE, P_COMMENT)
      values
      (2147483647, 'Pname text .......2.........3.........4....5E',
      'Pmfgr text........2....5E','Pbrand 10E',
      'Ptype varchar.....2....5E', -2147483646,
      'PcontainrE', 123456789012,
      'Part comment field  23E')
;

select * from part where P_PARTKEY = 2147483647
;




insert into partsupp
      (PS_partKEY, PS_suppKEY, PS_AVAILQTY, PS_SUPPLYCOST,
      PS_COMMENT)
      values
      (2147483647, 2147483647, -2147483646, 123456789012,
      'PS comment field is 199 long no E')
;
select * from partsupp
      where PS_partKEY = 2147483647
```

```
      and   PS_suppKEY = 2147483647
;


delete from orders where O_ORDERKEY = 2147483647      -- prevent Primary Key
violation
;


insert into orders
      (O_ORDERKEY, O_CUSTKEY, O_ordersTATUS, O_TOTALPRICE,
      O_ORDERDATE, O_ORDERPRIORITY, O_CLERK, O_SHIPPRIORITY,
      O_COMMENT)
      values
      (2147483647, 2147483647, 'X', 123456789012, '2008-03-27',
      'Order Priorty5E', 'Fixed text  15E', -2147483646,
      'Order comments field is 79 no E')
;
select * from orders
      where O_ORDERKEY = 2147483647
      and  O_CUSTKEY = 2147483647
;


insert into lineitem
      (L_ORDERKEY, L_partKEY, L_suppKEY, L_LINENUMBER,
      L_QUANTITY, L_EXTENDEDPRICE, L_DISCOUNT, L_TAX,
      L_RETURNFLAG, L_LINESTATUS, L_SHIPDATE, L_COMMITDATE,
      L_RECEIPTDATE, L_SHIPINSTRUCT, L_SHIPMODE, L_COMMENT)
      values
      (2147483647,
      2147483647,
      2147483647,
      -2147483646,
      -123456789012,
      -123456789012,
      -123456789012,
      -123456789012,
      'Q',
      'R',
      '2008-03-27',
      '2008-03-27',
      '2008-03-27',
      'Ship by camel .........5E',
      'Ship ASAPE',
      'Is this really what you wanted? 44 long....E')
;
select * from lineitem
      where L_ORDERKEY = 2147483647
      and L_partKEY = 2147483647
      and L_suppKEY = 2147483647
      and L_LINENUMBER = -2147483646
;


select now() 'End of inserts timestamp:';
```