Blistering Fast SQL Access to Hadoop using IBM BigInsights 3.0 with Big SQL 3.0

SQL-over-Hadoop implementations are ready to execute OLAP and complex query workloads at a fraction of the cost of traditional RDBMSs -- if you pick the right technology. Get beyond the hype and explore how IBM's BigInsights 3.0 with Big SQL 3.0, Cloudera's CDH 5.1.2 with Impala 1.4.1 and Hortonworks HDP 2.1 with Hive 0.13 executing on Tez stack up when running the Hadoop Decision Support benchmark. This benchmark typifies the kind of work associated with augmenting a relational data warehouse with a Hadoop-based platform and was modeled on the Transaction Processing Performance Council Decision Support (TPC-DS) benchmark and audited by a TPC-DS certified auditor.

Key Findings

■ **Query support matters**. All 99 TPC-DS queries could be expressed in Big SQL without the need to resort to extensive rewrites. Indeed, it took our SQL professionals less than one hour to port the whole workload to Big SQL. Due to limited SQL support, we were only able to express 88 of the 99 queries in Impala. For Hive, only 90 of the 99 queries could be successfully validated. The ports to both Hive and Impala took several weeks and SQL limitations forced us to extensively re-write many queries. See the "Query generation, re-write and validation" section for details.

■ **Scale matters.** Big SQL was the only implementation capable of executing all 99 queries at 10TB in both single-user and multi-user runs. By contrast, Hive executed only 70% of the workload at 10TB and Impala executed only 73%. Big SQL was also the only one able to execute the full workload at 30TB. See "Query Execution" section for details.

■ **Throughput matters.** When the workload was restricted to the 46 queries readily supported by all 3 implementations, ***Big SQL achieved 3.6 times more throughput than Impala and 5.4 times more throughput than Hive***.

Recommendations

Information management leaders should:

■ **Know your needs.** Consider the types of queries you might want to run initially, as well as those you may need to run in the future. How complex are your queries? What language constructs do you expect to use? How skilled is your staff in query rewrite technology?

■ **Look beyond the hype**. When vendors proffer performance claims, ask the tough questions: What workloads did they use? To what degree do these workloads comply with industry standards? What hardware configuration was used? Who audited the benchmark?

■ **Kick the tires.** If SQL support and/or performance are critical, there's no substitute for performing your own in-house comparative tests using your own workload. But if you don't have the skills, time, or resources to do so, examining comparative, audited benchmark results can be a good alternative.

Contents

Preface

The information contained in this document is distributed on an AS IS basis without any warranty either expressed or implied. The use of this information or the implementation of any of these techniques is the customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

Performance data contained in this document were determined in various controlled laboratory environments and are for reference purposes only. Customers should not adapt these performance numbers to their own environments and are for reference purposes only. Customers should not adapt these performance numbers to their own environments as system performance standards. The results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used.

This publication was produced in the United States. IBM may not offer the products, services, or features discussed in this document in other countries, and the information is subject to change without notice. Consult your local IBM representative for information on products and services available in your area.

Trademarks

IBM, the IBM logo, System x and System Storage are trademarks or registered trademarks of International Business Machines Corporation.

The following terms used in this publication are trademarks of other companies as follows: TPC Benchmark and TPC-DS are trademarks of Transaction Processing Performance Council; Intel and Xeon are trademarks or registered trademarks of Intel Corporation. Other company, product, or service names, which may be denoted by two asterisks (**), may be trademarks or service marks of others.

Cloudera, the Cloudera logo, Cloudera Impala are trademarks of Cloudera.

Hortonworks, the Hortonworks logo and other Hortonworks trademarks are trademarks of Hortonworks Inc. in the United States and other countries.

Apache, Apache Hadoop, Hadoop, Apache Hive, Hive, and Apache Tez are either registered trademarks or trademarks of the **Apache Software Foundation** in the United States and/or other countries.

Notes

[1] GHz and MHz only measures microprocessor internal clock speed, not application performance. Many factors affect application performance.

[2] When referring to hard disk capacity, GB, or gigabyte, means one thousand million bytes. Total user-accessible capacity may be less.

About the authors

Simon Harris: Simon is the Big SQL performance lead working in the IBM BigInsights development team. He has 20 years of experience working in information management including MPP RDBMS, federated database technology, tooling and big data. Simon now specializes in SQL over Hadoop technologies.

Abhayan Sundararajan: Abhayan is a Performance Analyst on IBM BigInsights with a focus on Big SQL. He has also held a variety of roles within the IBM DB2 team, including functional verification test and a brief foray into development before joining the performance team to work on DB2 BLU.

Edward G. (Gus) Branish: Gus is a Client Technical Professional on the Information and Analytics Group Team in the IBM Competitive Project Office. Gus has 20 years of experience on performance benchmarks of information management software. He has a special interest in query optimization in parallel processing environments.

Kenneth Chen: Ken is an Executive IT Specialist and Consulting Software Product Design Professional from the IBM Big SQL/DB2 Technical Alliance organization at the IBM Toronto Lab, specializing in business partner application enablement - application design, development, and performance. His areas of focus included distributed database platforms, as well as IBM DB2 Content Manager.

Special thanks to the following people for their contribution to the benchmark and content:

Berni Schiefer – Distinguish Engineer, Information Management Performance and Benchmarks, DB2 LUW, Big Data, MDM, Optim Data Studio Performance Tools; **Adriana Zubiri** – Program Director, Big Data Development; **John Poelman** – BigInsights Performance; **Mike Ahern** – BigInsights Performance; **Mi Shum** – Senior Performance Manager, Big Data; **Cindy Saracco** - Solution Architect, IM technologies - Big Data; **Avrilia Floratou** – IBM Research; **Fatma Özcan** – IBM Research; **Glen Sheffield** – Big Data Competitive Analyst; **Gord Sissons** – BigInsights Product Marketing Manager; **Matthew Emmerton** - DB2 OLTP Performance and Solutions Development; **Stewart Tate** – Senior Technical Staff Member, Information Management Performance Benchmarks and Solutions; **Jo A Ramos** - Executive Solutions Architect - Big Data and Analytics.

Introduction

Performance benchmarks are an integral part of software and systems development, as they can evaluate systems performance in an objective way. They have also become highly visible components of the exciting world of marketing SQL over Hadoop solutions.

Good benchmarks reflect, in a practical way, an abstraction of the essential elements of real customer workloads. In a typical Hadoop system this includes the task of loading the files onto HDFS as well as having both individual and concurrent users submitting SQL queries to the system.

IBM has constructed and used the Hadoop Decision Support (Hadoop-DS) benchmark, which was modelled on the industry standard TPC-DS[1] benchmark and validated by a TPC certified auditor. This benchmark typifies the kind of work associated with augmenting an existing relational data warehouse with a Hadoop-based platform. While adapting the workload for the nature of a Hadoop system we worked to ensure the essential attributes of both typical customer requirements and the benchmark were maintained.

In order to calibrate the Big SQL results we identified the common working subset of the full TPC-DS query set that could be measured and compared across three SQL over Hadoop implementations – IBM's BigInsights 3.0 with Big SQL 3.0 (hereafter referred to as Big SQL), Cloudera's CDH 5.1.2 with Impala 1.4.1 (hereafter referred to as Impala) and Hortonworks HDP 2.1 with Hive 0.13 executing on Tez (hereafter referred to as Hive 0.13) – these were the latest versions of the products available in September 2014. In a number of cases sophisticated SQL construction skills were required in order to translate the official reference SQL into a syntax executable on the target systems. The resulting subset of queries reflects a broad range of SQL functionality and query complexity. While we were surprised at just how difficult it was to get both Hive and Impala to run many of the queries, particularly in the multi-user concurrency test, we are proud to have clarified the validated TPC-DS query portfolio known to work on both Impala and Hive.

In this paper we will provide an overview of our methodology, details on query generation, selection and validation. We will also describe the system under test from a hardware and software perspective, and detail the process for creating the tables, populating the Hadoop files and gathering statistics. We also summarize and analyze the results of the performance measurements.

Methodology

The aim of this benchmark was to provide an accurate and fair comparison across all three SQL over Hadoop distributions in a scenario common to many organizations adopting the technology today. The most common scenario we see involves off-loading subsets of workloads from the traditional relational data warehouse to SQL over Hadoop solutions (a process commonly referred to as warehouse augmentation). For this reason our Hadoop-DS workload was modelled on the Transaction Processing Performance Council Benchmark DS (TPC-DS)[2]. The TPC-DS benchmark uses relational database management systems (RDBMSs) to model a decision support system that examines large volumes of data and gives answers to real-world business questions by executing queries of various types (such as ad-hoc, reporting, OLAP and data mining type queries). It is,

[1] TPC Benchmark and TPC-DS are trademarks of the Transaction Processing Performance Council (TPC).

[2] The Hadoop-DS benchmark is based on the latest revision of the TPC-DS specification. The specification can be found at http://www.tpc.org/tpcds/default.asp

therefore, an ideal fit to mimic the experience of an organization porting parts of their workload from a traditional warehouse housed on an RDBMS to a SQL over Hadoop technology. As highlighted in IBM's "Benchmarking SQL-on-Hadoop Systems: TPC or not TPC?"[3] Research paper, SQL over Hadoop solutions are in the "wild west" of benchmarking. Some vendors may use the data generators and queries of existing TPC benchmarks, but cherry pick the parts of the benchmark most likely to highlight their own strengths and fail to adhere to the specified rules – thus making comparison between results meaningless.

To make this a complete and fair comparison, we did not cherry-pick the parts of the TPC-DS workload that would highlight Big SQL's strengths. Instead, we included **all** parts of the TPC-DS workload that are appropriate for SQL over Hadoop solutions: data loading, single user performance and multi-user performance. Since TPC-DS is a benchmark designed for relational database engines, some aspects of the benchmark are not applicable to SQL over Hadoop solutions. Broadly speaking, those are the "Data Maintenance" and "Data Persistence" sections of the benchmark. Consequently these sections were omitted from our Hadoop-DS workload. The TPC-DS benchmark also defines restrictions related to real-life situations – such as preventing the vendor from changing the queries to include additional predicates based on a customized partitioning schema, employing query specific tuning mechanisms(such as optimizer hints), making configuration changes between the single and multi-user, tests etc.

To ensure the fairest possible comparison, we endeavoured to stay within the bounds of these restrictions for the Hadoop-DS workload and conducted the comparison with candor and due diligence. To validate our candor, we retained the services of Infosizing[4], an established and respected benchmark auditing firm with multiple TPC certified auditors, including one with TPC-DS certification, to review and audit all our benchmarking results. It is important to note that these are not official TPC-DS benchmark results since aspects of the standard benchmark that do not apply to SQL over Hadoop solutions were not implemented. However, the independent review of the environment and results by a certified auditor shows IBM commitment to openness and fair play in this arena. All deviations from the TPC-DS standard benchmark, for all products, are noted in the attached auditor's attestation letter in Appendix G. In addition, all the information required to reproduce the environment and the Hadoop-DS workload for all three vendors is published in the various Appendices of this document – thus allowing any vendor or third party the ability to execute the benchmark and verify the results independently.

There are 99 queries in the official TPC-DS benchmark along with a toolkit to generate the queries in various SQL dialects. Since there are many variations of SQL dialects, the specification also allows the sponsor to make pre-defined minor modifications to the queries so they can be successfully compiled and executed. The team attempted to port all 99 queries to all 3 platforms. For us, close to

[3] "Benchmarking SQL-on-Hadoop Systems: TPC or not TPC?" http://researcher.ibm.com/researcher/files/us-aflorat/BenchmarkingSQL-on-Hadoop.pdf

[4] Infosizing: www.**infosizing**.com/

a third of the queries ported to Impala needed extensive re-writes[5], while for Hive, close to a quarter of the queries required extensive re-writes. These extensive re-writes were necessary to avoid limitations in the SQL support for these vendors. Some were considered re-writes that are commonly employed by SQL programmers, but others were much more involved. The team assigned one SQL programmer to port the queries to Hive, one to port the queries to Impala and one to port the queries to Big SQL – each programmer had many years of SQL experience. As a consequence of the limited SQL dialect available in Hive and Impala, it took approximately four weeks of effort to port the queries to these products. Since Big SQL supports a rich set of SQL syntax, many queries worked *out of the box* and consequently the porting effort was less than 1 hour.

Although SQL dialect support was not the focus of this project, the teams experiences porting the queries show how important rich SQL support is for an SQL over Hadoop solution. This is even more important in an enterprise when the queries are being automatically generated by Business Intelligence tools such as Microstrategy or Cognos which restrict the ability to manually re-write the queries they generate. Parallels can certainly be drawn between the team's experience of porting the queries from TPC-DS (an RDBMS workload) to Hadoop-DS (an SQL over Hadoop workload) and organizations attempting to move their own workloads away from their data warehouses to SQL over Hadoop. **Unless an organization wants to spend considerable time and effort porting their queries, they should seriously evaluate the SQL capabilities and compliance of SQL over Hadoop vendors.**

Several queries that had been successfully ported to Hive and Impala and ran well in some circumstances, failed when executed on larger data sets or with multiple concurrent streams . The most common error for both vendors was running out of memory. The team spent several more weeks analysing the failed Hive and Impala queries and tuning the environment to get as many queries working as possible. All queries were successful in Big SQL without the need for additional tuning.

To provide the fairest apples to apples comparison the Hadoop-DS benchmark needed to use the same set of queries across all vendors – therefore the team executed a number of *query validation runs* to check which queries could successfully execute both the single and multi-user runs at the chosen scale factor. Any query which failed on either Hive or Impala was removed from the set of queries to be tested in the final performance run. In this way, we were left with a common sub-set of queries that all vendors could execute at the target scale factor. This query set was used to execute both the single-user and multi-user performance tests.

Cluster configuration & tuning

The benchmark was conducted on three identical 17 node clusters, each node being an IBM x3650 M4 BD server. A complete specification of the hardware used can be found in

[5] For Impala, we followed the guidelines in "Porting SQL from Other Database Systems to Impala" in the "Impala SQL Language Reference" http://www.cloudera.com/content/cloudera/en/documentation/cloudera-impala/v1/latest/Installing-and-Using-Impala/ciiu_porting.html?scroll=porting_statements_unique_1

Appendix B: Cluster topology and hardware configuration.

In our performance evaluation, a number of tuning best practices were adopted across all products. As with all benchmarks, the tuning process was iterative and continually refined across all three distributions as queries and workloads were executed. The clusters all started with the same set of OS and kernel tunings common to Hadoop clusters. The Hive cluster was tuned further using the recommendations found on Hortonworks latest benchmark (at http://hortonworks.com/blog/benchmarking-apache-hive-13-enterprise-hadoop/). The latest Impala benchmarks do not provide configurational information, so the team used the best practices from the Cloudera documentation. Big SQL was configured according to IBM's best practices. The detailed tuning and configuration used for each product can be found in Appendix C.

Tuning features that benefited performance of the overall workload were employed wherever reasonable. Of course, the team had a finite amount of time to complete the benchmark and it was not possible to evaluate all features and tuning knobs for all products. We focused on those features we thought would give the biggest payback in terms of performance, based on our understanding of each of the products.

Big SQL was configured to use unique and differentiating optimization features such as Informational Constraints and Statistical Views. Both these features provide additional information to the Big SQL optimizer allowing for better access strategy cost estimates and consequently, improved performance.

The TCP-DS specification recognizes the importance of constraints and statistics in the query optimization process. It is our belief, however, that some of the limitations imposed by the TPC-DS specification are less applicable to the read-only SQL over Hadoop environment than to the RDBMS environment for which they are targeted. Although these features may not meet all aspects of the TPC-DS specification, they are commonly used by customers in other IBM data management products, and therefore we strongly believe they provide value to our customers. Our statistics gathering commands, constraint definitions and statistical view definitions are disclosed in Appendix F.

In order to simulate a production system the Hadoop-DS benchmark was executed without individual query tuning and without configuration changes between the single and multi-user runs.

Database schema

The Hadoop-DS schema mirrors the TPC-DS schema and consists of 7 large fact tables and 17 smaller dimension tables. All products implemented the full 24 table schema.

Full schema definitions for each product can be found in Appendix D.

Query generation, re-write and validation

The team followed the documented process to generate the queries from the templates provided in the TPC-DS specification. Those queries which failed to compile or experienced run-time errors went onto the second phase which was to apply a set of minor query modifications in the spirit of what is allowed by the TPC-DS specification. Queries which still would not compile, execute successfully or provided incorrect results went onto a third phase in which more extensive and complex query re-writes were attempted. All queries were validated using the 1GB qualification database against pre-defined result sets provided by the TPC-DS specification – to help verify that the modified queries still answered the business questions being posed.

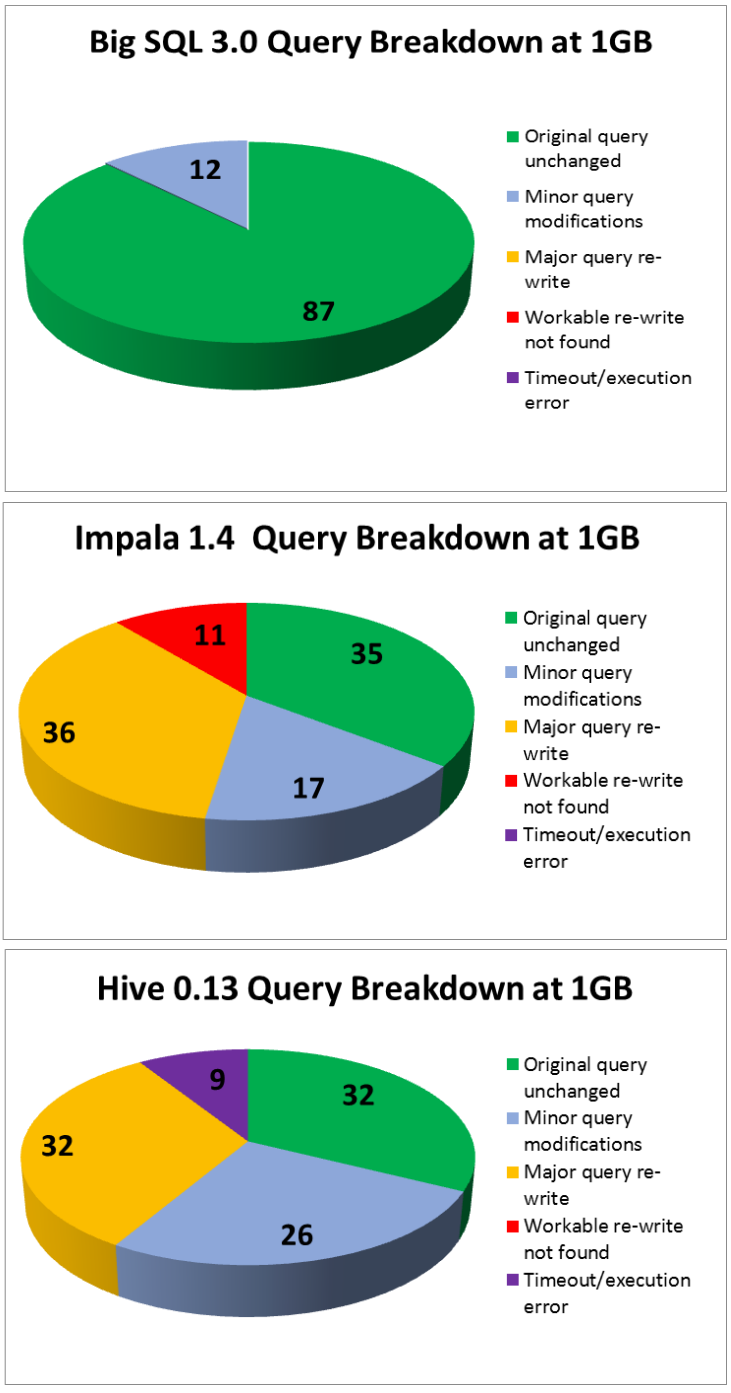


Chart 1: Query breakdown by vendor

The team dedicated significant time and effort to get as many queries as possible working. Chart 1 summarizes the breakdown of minor query modifications and major re-writes against the 1GB qualification database for each vendor.

A product's ability to execute SQL queries generated directly from the unchanged templates, or slightly altered with only minor query modifications is an excellent indicator of its support of the SQL language.

Following a substantial porting effort a valid re-write could not be found for 11 Impala queries - Impala 1.4.1 does not support SQL windowing functions. All 99 queries were ported to Hive syntax, but 9 had execution errors or timed-out (after 2 hours) on the 1GB qualification database.

For Big SQL, 87 of the 99 queries worked directly from the original query. The other 12 queries required only simple and minor modifications (mainly type casts) and took less than an hour to complete.

One of the recommendations from this experience is that organizations considering SQL over Hadoop solutions should carefully evaluate the level of SQL support in order to avoid a similar painful experience.

In summary, after the porting exercise there were 83 queries working across all three products against the 1GB qualification database.

Data load

The team started with a database scaled at 10TB with a plan to scale it up to 30TB later. This decision was motivated by the fact that Cloudera had published a TPC-DS like benchmark at 15TB, and HortonWorks similarly at 30TB – both using a smaller query subset that accessed just a fraction of the full set of TPC-DS tables

The data load phase of the workload includes everything required to ingest the data into the database and prepare the cluster to execute the single and multi-user runs. Briefly, this phase consists of hardware and software set-up, data generation and copy to HDFS, cluster tuning, table creation, ingest of the data into the storage format of choice, gathering of statistics and creation of any data structures to aid performance and scalability.

For each product the optimal storage format was chosen - for Hive 0.13 this is ORC with ZLib compression enabled. For Impala and Big SQL, the Parquet format was used with Snappy compression.

Table 1 shows the database ingest times for 10TB of raw data:

	Big SQL 3.0	Impala 1.4.1	Hive 0.13
Storage format	Parquet	Parquet	ORC
Compression	Snappy	Snappy	ZLib
Data ingest time (mins)	185	107	663

Table 1: Database load times for Hadoop-DS @10TB.

Query execution

Hadoop-DS uses the “Hadoop-DS Qph” metric to report query performance. The Hadoop-DS Qph metric is the effective query throughput, measured as the number of queries executed over a period of time. A primary factor in the Hadoop-DS metric is the scale factor (SF) -- size of data set -- which is used to scale the actual performance numbers. This means that results have a metric scaled to the database size which helps differentiate large databases from small database (since performance is typically a factor of database size).

A Hadoop-DS Qph metric is calculated for each of the single and multi-user runs using the following formula:

Hadoop-DS Qph @ SF = ((SF/100) * Q * S) / T

Where:

- SF is the scale factor used in GB (10,000 in our benchmark).
  SF is divided by 100 in order to normalize the results using 100GB as the baseline.
- Q is the total number of queries successfully executed
- S is the number of streams (1 for the single user run)
- T is the duration of the run measured in hours (with a resolution up to one second)

Hadoop-DS Qph metrics are reported at a specific scale factor. For example 'Hadoop-DS Qph@10TB' represents the effective throughput of the SQL over Hadoop solution against a 10TB database.

10TB scale factor using a common query set

The same 83 queries that successfully executed on all 3 products on the 1GB qualification database were taken and executed on a 10TB database. Both Hive and Impala struggled to execute all 83 common queries in the initial single user validation runs at 10TB. The most common error received came from running out of memory, followed by queries timing out (after 2 hours). There is a well-known restriction in Impala 1.4.1 that the internal state of the query has to fit into the aggregate memory on the cluster. However, to experience similar errors using Hive 0.13 on Tez was a surprising find. Hive also experienced *Null pointer* exceptions when particular (valid) predicates were substituted into some queries. Several more days were spent tuning Hive and Impala in order to increase the number of queries successfully completing at 10TB. Meanwhile, Big SQL was able to complete all 99 queries at 10TB without additional tuning.

Initially on the 4-stream concurrent run, there were many more out of memory failures in both Hive and Impala. For Impala, it appears the "coordinator node" for each Impala query requires more memory and CPU cycles than the other nodes that process the query[6]. Using the default configuration (of a single Impala coordinator) therefore led to additional out of memory errors when executing multiple concurrent streams. The Impala driver scripts were therefore modified to issue queries using round-robin scheduling, so that each connection used a different coordinator node. This load-balancing technique lets the Impala nodes share the additional coordinator work, rather than concentrating it on a single machine. In addition it was also necessary to configure *Dynamic Resource Pools* within the Cloudera cluster to manage resource allocations and prevent queries from failing. Full details can be found in Appendix C.

For Hive, the YARN memory configuration settings are particularly critical. The number of containers was increased by reducing the container size, as each YARN task is assigned a container. The Java heap (particularly the *mapreduce.reduce.java.opts* property) was increased to address many of the *out of Heap* issues in the multi-stream run. These changes were implemented to override default configurations in the *mapred-site, yarn-site, tez-site*, and *hive-site* configuration files. It was also noted that at times of high CPU usage some jobs in the multi-stream run timed-out before executing in the Tez session. Consequently the Tez time out property was set to a very large value to prevent

[6] http://www.cloudera.com/content/cloudera/en/documentation/cloudera-impala/v1/latest/Installing-and-Using-Impala/ciiu_concepts.html

jobs from timing-out and to allow the job to wait for the resource to be free to start execution. In order to provide the most available memory to the environment, 85% of total memory (of 128GB) was made available for configuration. In addition, HBase and Oozie were stopped as they were not required to run the benchmark. A number of other best practices across Hive and Tez were also adopted – see Appendix C for full details. It took the team a few weeks to resolve all these issues on Hive and Impala.

The benefits of Big SQL's rich SQL support and robust execution engine clearly shines through at this point and it is worth emphasizing the fact that Big SQL is the only product capable of executing all 99 queries at 10TB in both single and multi-stream validations runs. Chart 2 highlights the fact that Impala can only successfully execute 73% of the original workload, and Hive 70%.

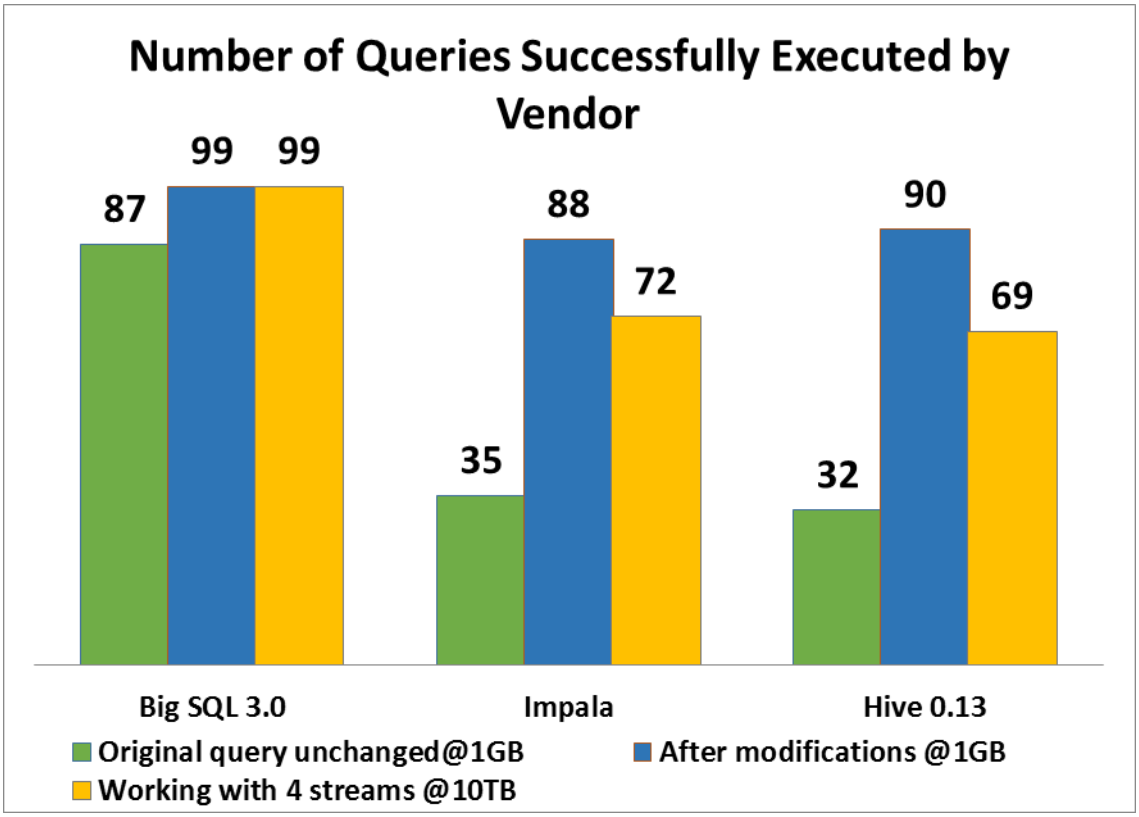


Chart 2: Number of queries successfully executed by product

Chart 2 also highlights how Impala and Hive struggle to scale. 18% of Impala queries that worked at 1GB failed to work at 10TB, and 23% of Hive queries that worked at 1GB failed to work at 10TB.

There were 46 common queries across the 72 Impala and 69 Hive queries working after the single and multi-user validation runs at 10TB. These were the queries used during the performance runs.

Having identified the common set of 46 working queries the team was now in a position to execute the performance run and thus provide a true apples to apples comparison across the three distributions.

In compliance with the TPC-DS specification, a single user performance run was executed, followed by a multi-user run[7]. In both runs, each stream executed the set of 46 common queries in the permutation defined by the specification. Due to the challenges discussed earlier of configuring multiple concurrent streams in Hive and Impala, it was decided to limit the multi-user test to only 4 streams. In our internal performance tests, Big SQL has successfully executed the same workload with 12 concurrent query streams.

Charts 3 and 4 summarise the elapsed times and effective query throughput of the single stream performance run on the 10TB dataset:

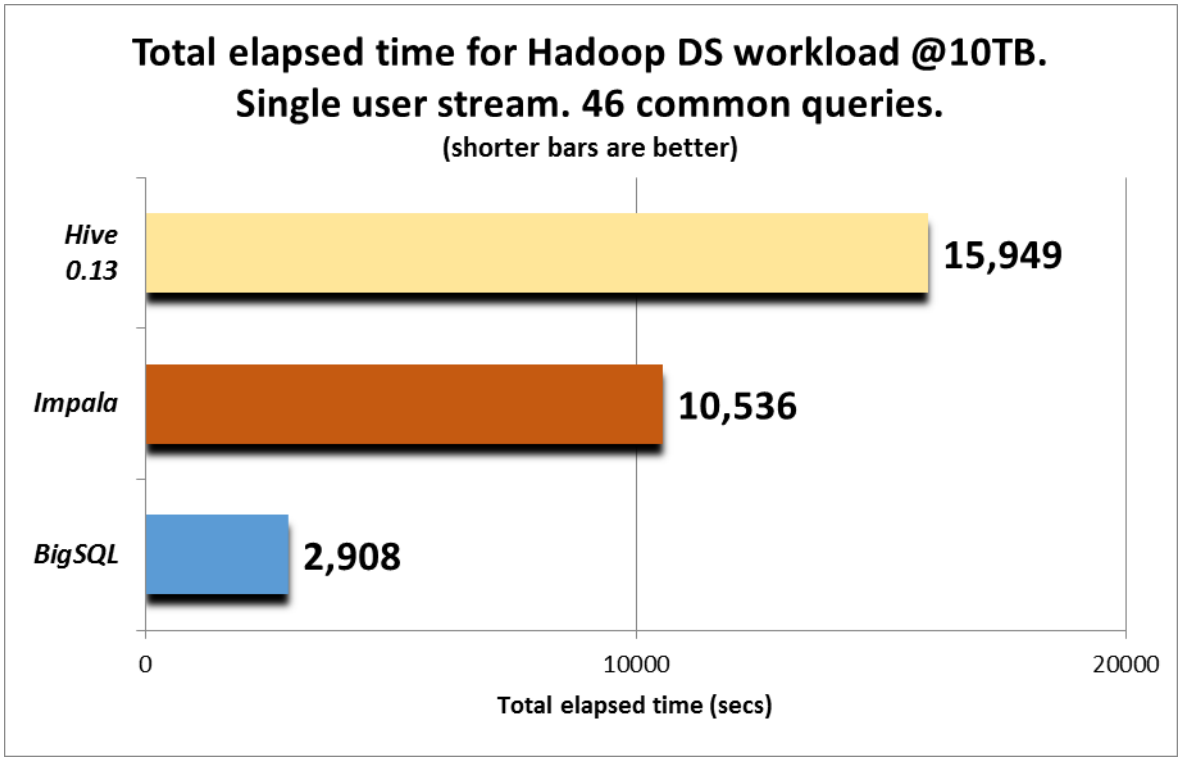


Chart 3: Hadoop-DS workload elapsed time @10TB across 46 common queries

[7] The TPC-DS specification calls for 2 multi-stream runs, separated by the Data Maintenance phase of the benchmark. Since the Hadoop-DS benchmark does not have a Data Maintenance phase, the second multi-stream run would work on identical data to the first, and consequently would not provide any additional useful insight. Therefore the 2nd multi-stream run was dropped from the Hadoop-DS benchmark.

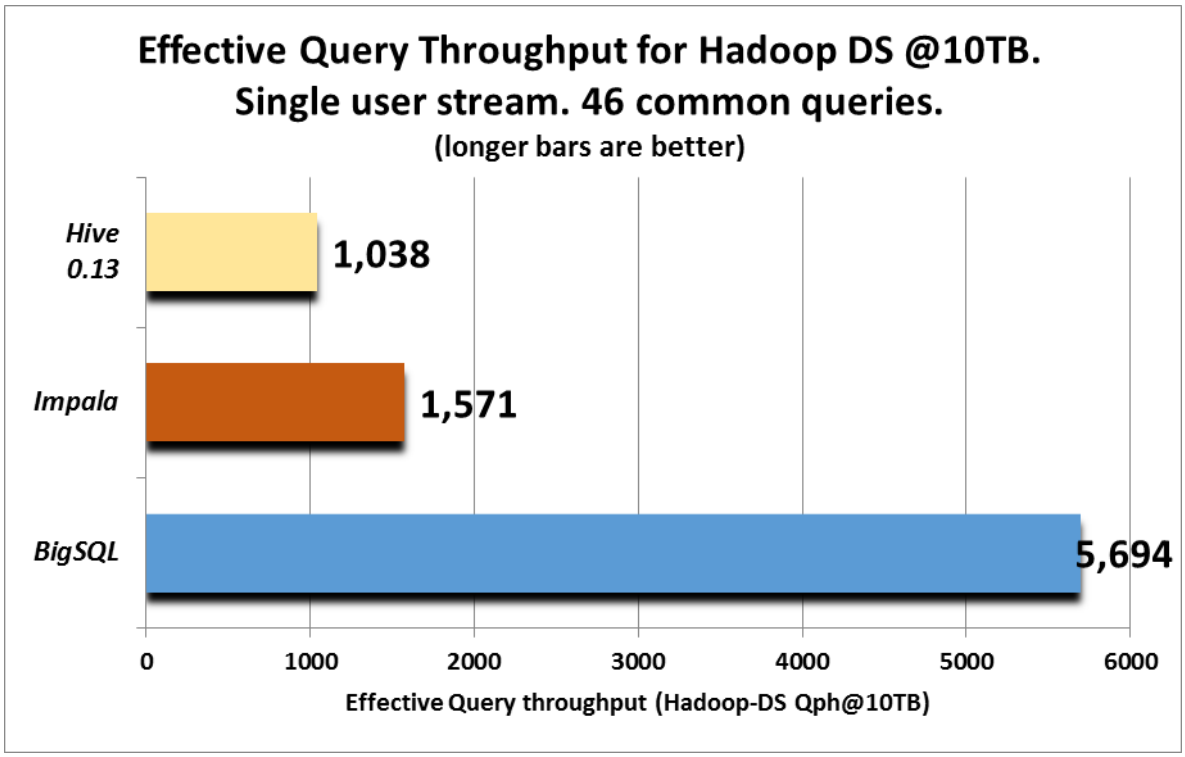


Chart 4: Hadoop-DS effective query throughput @10TB across 46 common queries

The results at 10TB show that Big SQL is 3.6x faster than Impala and 5.4x faster than Hive 0.13. This is an impressive result for Big SQL, especially given the fact that many of the long running queries (where Big SQL is strongest) were removed from the workload because Impala or Hive could not complete them.

Charts 5 and 6 shows the elapsed time and effective query throughput for the Hadoop-DS workload executing 4 concurrent query streams at 10TB:

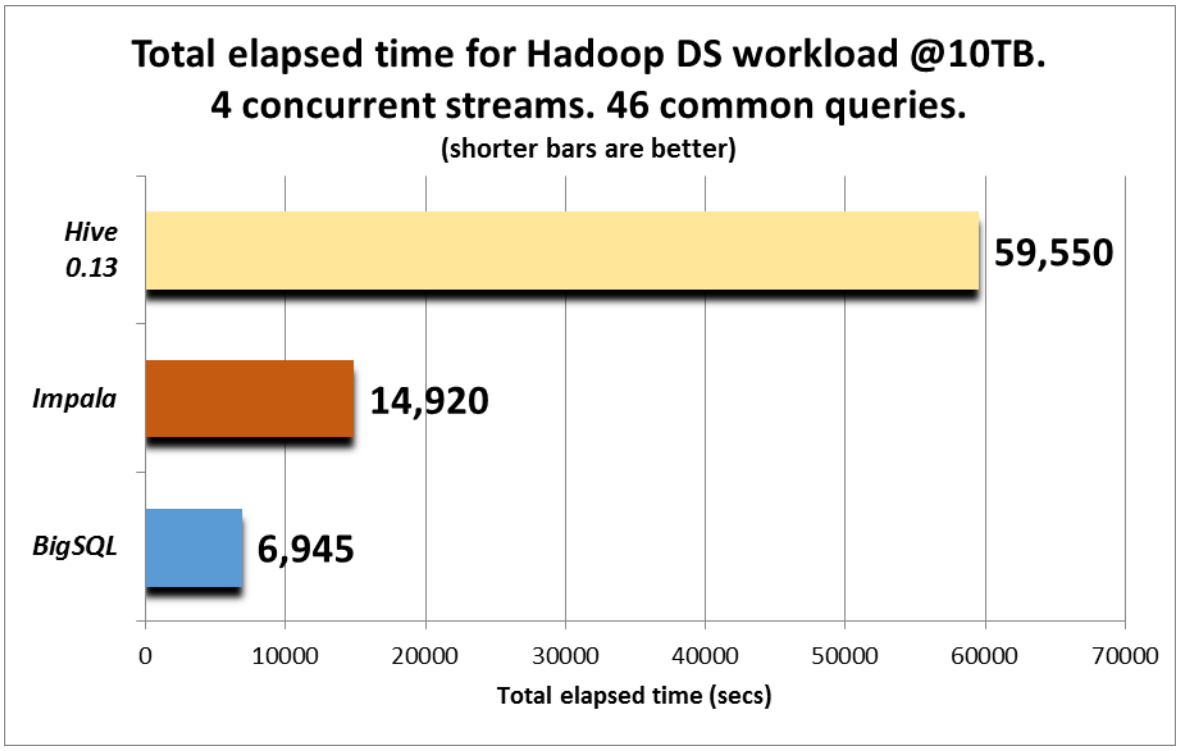


Chart 5: Workload elapsed time for 4 concurrent streams at 10TB across 46 common queries

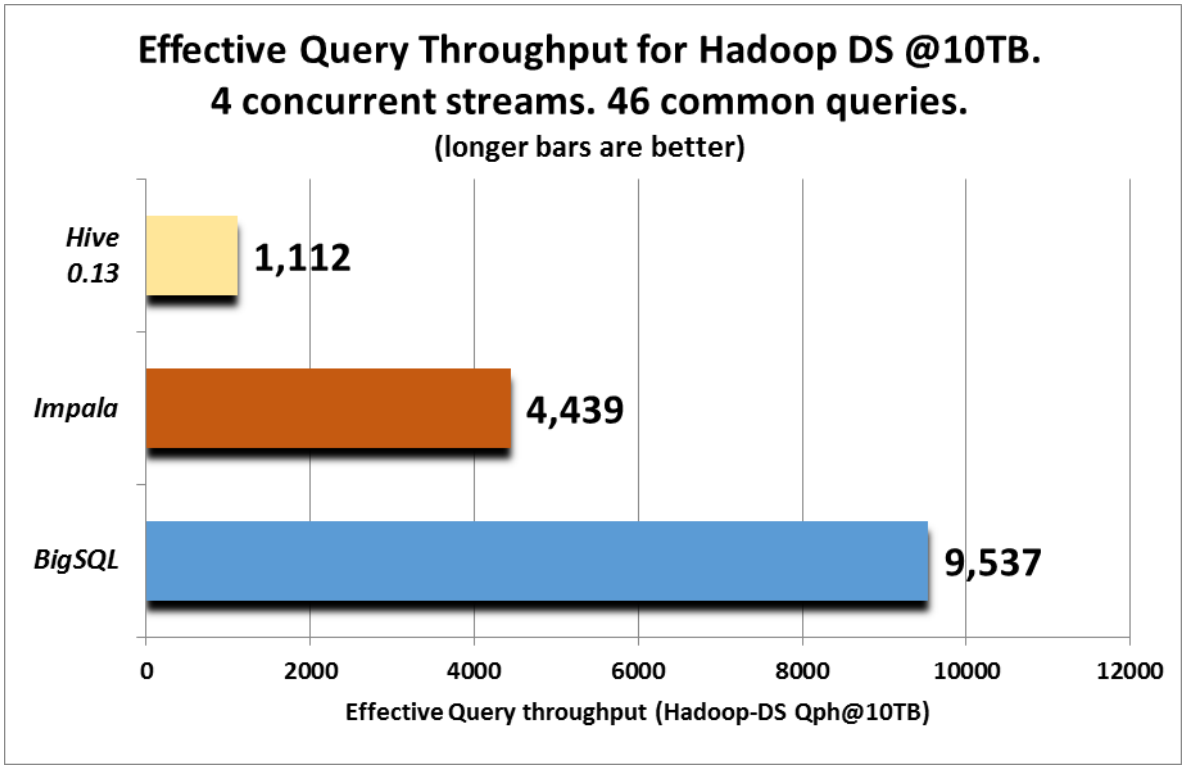


Chart 6: Effective query throughput for 4 concurrent streams at 10TB across 46 common queries

The results mirror those from the single stream run. **Big SQL achieves the highest throughput and has 2.1x greater throughput than Impala and 8.5x more throughput than Hive**.

10TB scale factor using all 99 queries

Since Big SQL supports a richer SQL syntax and was able to execute all 99 compliant queries in the workload, the team also executed a single stream and multi-stream run using the full set of 99 queries from the Hadoop-DS workload. For the single stream run, Big SQL completed execution of all 99 queries in approximately 8hrs and 59mins with an effective throughput rate of 1,101 Hadoop-DS Qph@10TB – as highlighted in chart 7:

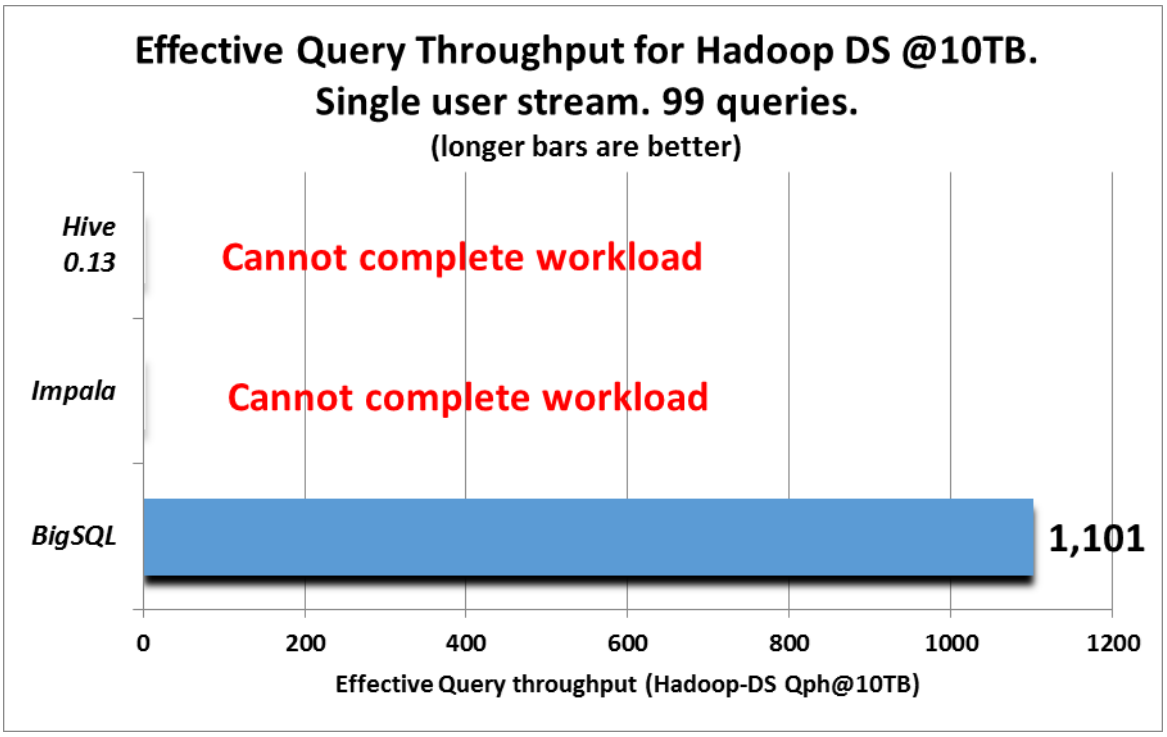


Chart 7: Hadoop-DS throughput for single query stream @10TB using all 99 queries

We used 6 concurrent query streams for the multi-stream run, in which Big SQL completed all 594 queries in just over 24hrs 39mins. This equates to an effective throughput rate of 2,409 Hadoop-DS Qph@10TB – as illustrated in Chart 8:

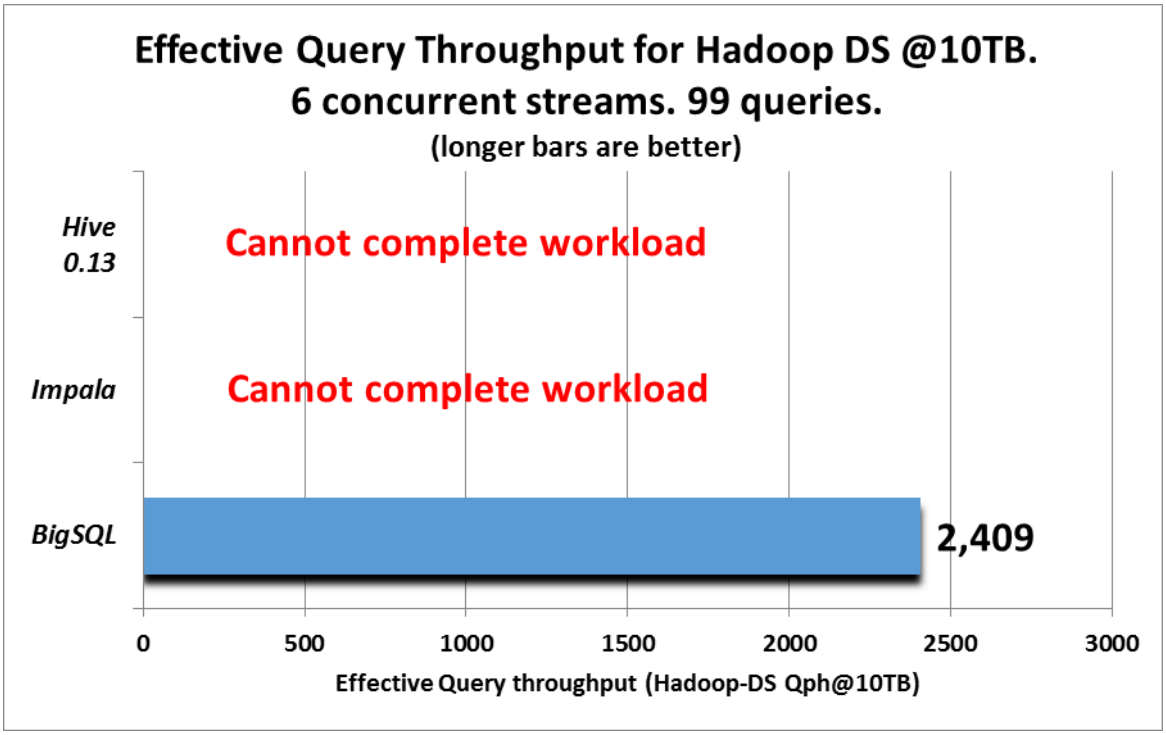


Chart 8: Hadoop DS throughput for 6 concurrent streams at 10TB using all 99 queries

Although the query throughput measured when running all 99 queries was lower than what was measured when running the subset of 46 queries, we need to consider that these are actually two different workloads. The entire 99 query workload included all of the complex, longer running queries which failed with Hive and/or Impala. This is highlighted by comparing the average elapsed times of the 46 queries in the common subset (64 sec. for Big SQL) with the average elapsed time of the remaining 53 queries from the full set (549 sec. for Big SQL). **So not only are these throughput numbers for Big SQL very impressive, they were achieved on a workload that includes queries that neither Impala nor Hive could run in our testing.**

Even more impressive is the scalability of Big SQL when running the complete workload. **Big SQL took just 2.7x longer to complete the 6 times volume of queries compared to the single query stream run** – as illustrated in chart 9:

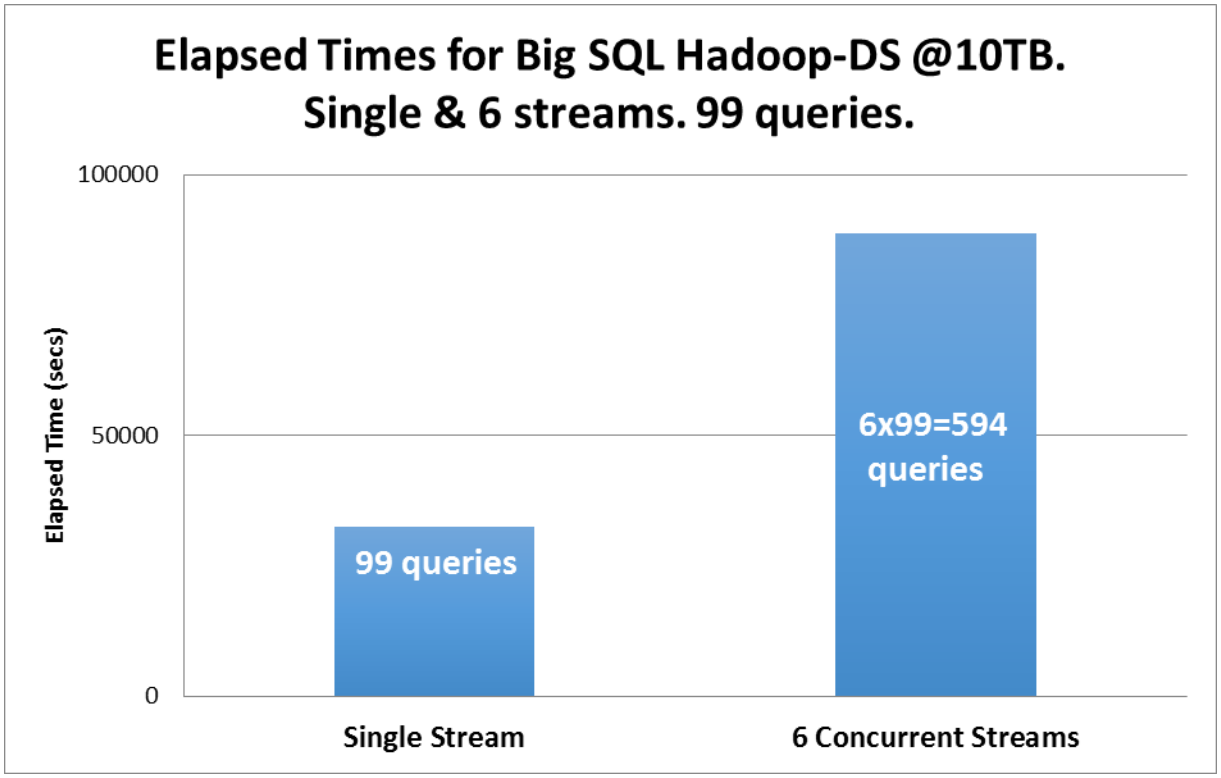


Chart 9: Big SQL multi-user scalability using 6 query streams @10TB

30TB scale factor using all 99 queries

Our original goal was to run a complete benchmark at 30TB, using all 99 queries, across all vendors. The 10TB dataset was to act as a stepping stone to achieve this. However, due to SQL limitations, query timeouts and errors that occurred when running Hive and Impala we had to compromise on the query set in order to get an apples-to-apples comparison at 10TB. It was the team's view that moving Hive and Impala to 30TB would reduce the common query set to such an extent as to make any comparison between the vendors meaningless. Consequently the team proceeded with a 30TB Hadoop-DS benchmark for Big SQL only.

At 30TB, (on the same 17-node cluster) Big SQL was again able to execute all 99 queries successfully in both the single and multi-user workloads. For the single user performance run, Big SQL's effective throughput rate was 1,023 Hadoop-DS Qph@30TB. For the multi-stream run using 4 concurrent streams Big SQL achieved a throughput of 2,274 Hadoop-DS Qph@30TB. Chart 10 summarises the elapsed times for the single and multi-user runs at 30TB:

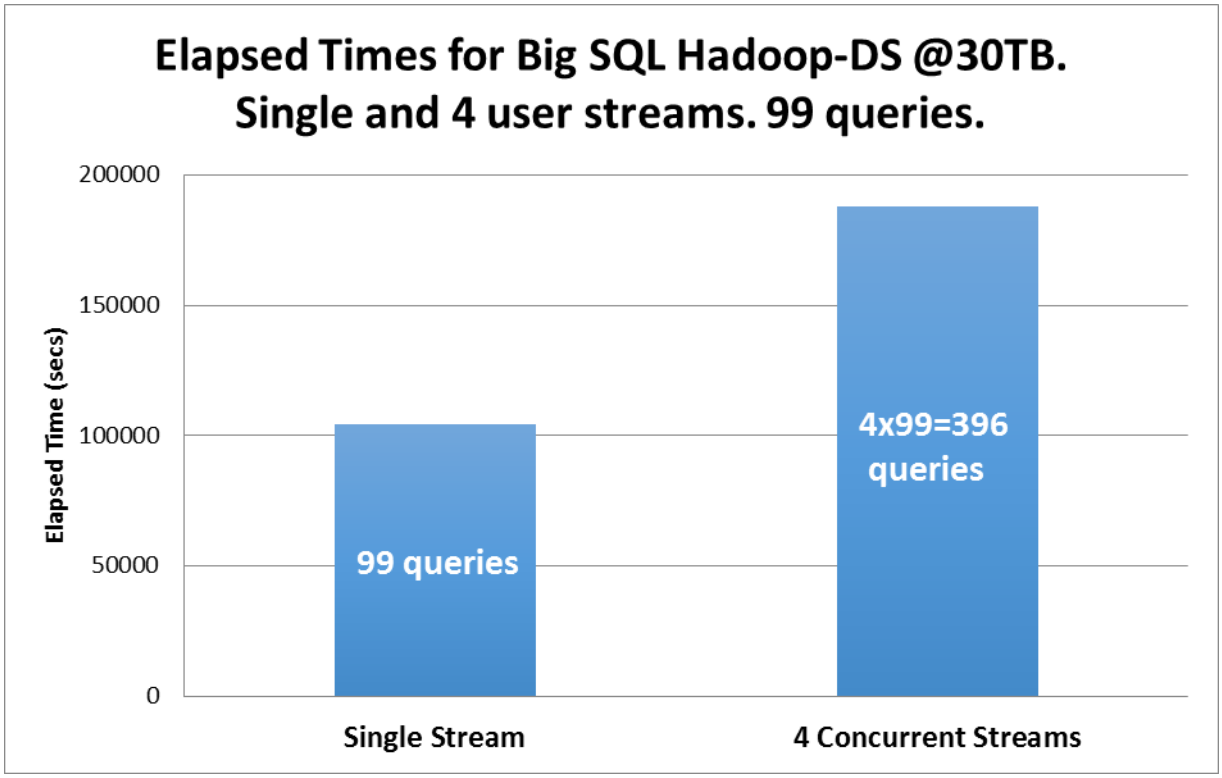


Chart 10: Big SQL multi-user scalability using 4 query streams @30TB

Of particular note is the fact that 4 concurrent query streams (and therefore 4 times more queries) only takes 1.8x longer than a single query stream at 30TB. Once again highlighting Big SQL's impressive multi-user scalability – this time at 30TB.

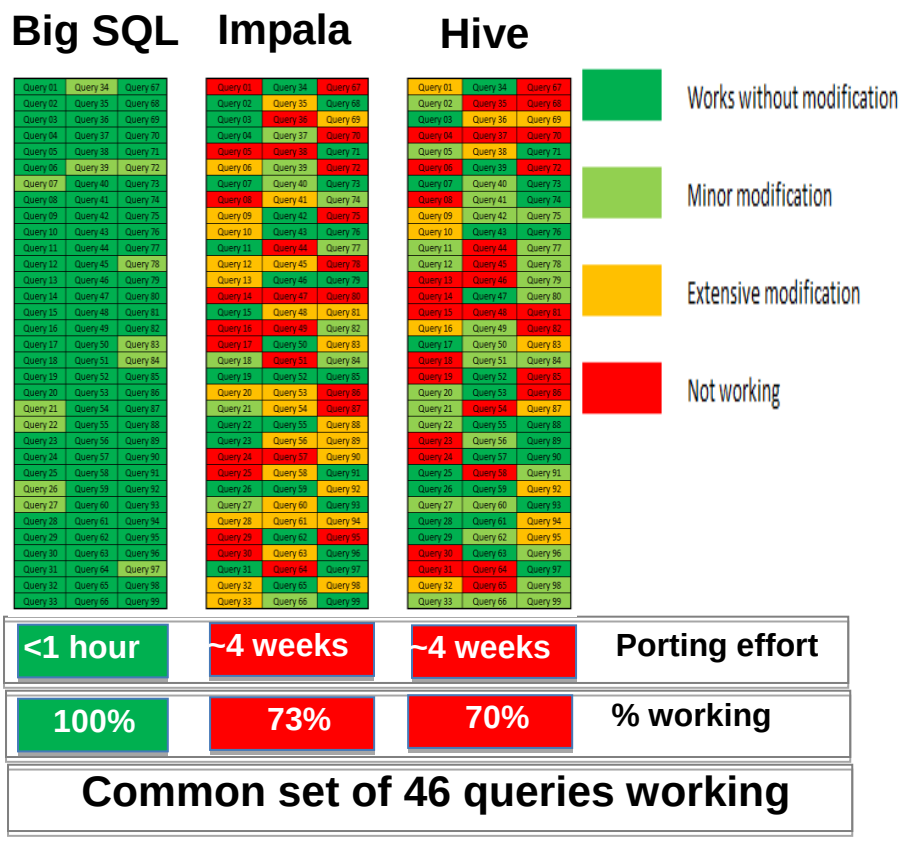
Summary

TPC-DS is the leading standardized analytic workload for RDBMs, which we adapted for Hadoop. We have named this derivative workload Hadoop-DS to avoid confusion and retained the services of a TPC certified auditor to validate fair comparisons.

As we have demonstrated in this white paper, IBM Big SQL was able to implement rich SQL with outstanding performance for both single and multiple concurrent users.

The heat map highlights the fact that Big SQL could run all 99 TPC-DS queries without major modification. Something we were not able to get the other offerings to do.

The heat map also shows the enormous variance in effort it took for us to rewrite the queries in the benchmark and tune the other systems to be able to run the benchmark at 10TB with 4 concurrent users.



We have also shown leadership performance results with Big SQL for single user and multi user tests at 10TB of raw data. In addition Big SQL was the only tested solution able to complete the full workload at 10 and 30TB.

			Elapsed time (s)		Hadoop-DS Qph@10TB		Big SQL Advantage (x times faster)[8]	
	# Queries		Single Stream	4-streams	Single Stream	4-streams	Single Stream	4-streams
Big SQL 3.0	46		2,908	6,945	5,694	9,537	-	-
Impala 1.4.1	46		10,536	14,920	1,571	4,439	3.6	2.1
Hive 0.13	46		15,949	59,550	1,038	1,112	5.4	8.5

			All 99 queries @ 10TB					
Big SQL 3.0	99		32,361	88,764	1,101	2,409		
Impala 1.4.1	99		Not Possible					
Hive 0.13	99		Not Possible					

			All 99 queries @ 30TB					
Big SQL 3.0	99		104,445	187,993	1,023	2,274		
Impala 1.4.1	99		Not Possible					
Hive 0.13	99		Not Possible					

These findings will be compelling to organizations augmenting data warehouse environments with Hadoop-based technologies. Not only was IBM's Big SQL the only Hadoop solution tested able to actually run the complete set of Hadoop-DS queries, but also we found it to be the fastest, the most scalable, and the most reliable as well.

Strict SQL compliance can translate into significant cost savings by allowing customers to leverage existing investments in databases, applications and skills and take advantage of SQL-over-Hadoop with minimal disruption to existing environments. Enterprise customers cannot afford to have different dialects of SQL across different data management platforms. In this testing, IBM's Big SQL demonstrated the highest degree of SQL language compatibility by a wide margin.

Not only was IBM Big SQL more compatible, but also it was significantly faster. This means that customers can realize business results faster, ask more complex questions, and realize great efficiencies per unit investment in infrastructure. All of these factors help provide a competitive advantage.

[8] X times faster calculation based on comparison of Qph values.

IBM Big SQL is the industry leading SQL over Hadoop solution available today. We believe the performance and language richness coupled with strict adherence to the TPC rules demonstrate this capability. With Big SQL we have set the bar an order of magnitude higher than previous SQL over Hadoop approaches.

Appendix A: Workload times:

Table A.1 provides a summary of the benchmark results for the 46 common queries:

	**IBM System x3650 M4 BD**		**Hadoop-DS**[9] **Nov 14, 2014**
Elapsed Times (secs)			
Test	**Big SQL 3.0**	**Impala 1.4.1**	**Hive 0.13**
Database Ingest	11,100	6,420	39,780
Run 1 – Single Stream	2,908	10,536	15,949
Run 2 – Multi-Stream	6,945	14,920	59,550
Effective Query Throughput (Hadoop-DS Qph@10TB)			
Run 1 – Single Stream	5,694	1,571	1,038
Run 2 – Multi-Stream	9,537	4,439	1,112
Query Numbers			
Queries working	99	72	69
Queries in common set	46		
Number Query Streams	4		

Table A.1: Summary of Hadoop-DS benchmark results @10TB

[9] The Hadoop-DS benchmark is derived from TPC Benchmark DS (TPC-DS) and is not comparable to published TPC-DS results. TPC Benchmark is a trademark of the Transaction Processing Performance Council.

Appendix B: Cluster topology and hardware configuration

IBM System x3650

Each cluster consisted of 17 IBM x3650 M4 BD servers with 1 master node and 16 data nodes. Each server was configured with:

- CPU: e5-2680@2.8GHz v2 2 sockets, 10 cores each, hyper threading enabled = 40 logical CPUs
- Memory: 128 GB RAM at 1866 MHz
- Storage: 10 x 2TB 3.5” Serial SATA, 7200RPM. One disk for OS, 9 for data
- Storage: 4 x 128GB SSD (not used during benchmark)
- Network: Dual port 10 Gb Ethernet
- OS: Red Hat Enterprise Linux Server release 6.4 (Santiago)

Appendix C: Tuning

C.1 Big SQL Configuration

Installation options:

During install, the following Big SQL properties were set. *Node resource percentage* was set to 90% in order to provide as much of the cluster resources as possible to Big SQL:

```
Big SQL administrator user:      bigsql
Big SQL FCM start port:          62000
Big SQL 1 server port:           7052
Scheduler service port:          7053
Scheduler administration port:   7054
Big SQL server port:             51000
Node resources percentage:       90%
```

The following are in-line with current BigInsights and Big SQL 3.0 best practices which recommend distributing all i/o for the Hadoop cluster across all disks:

```
BigSQL2 data directory:
/data1/db2/bigsql,/data2/db2/bigsql,/data3/db2/bigsql,/data4/db2/bigsql,/data5/db2/bigsql,/dat
a6/db2/bigsql,/data7/db2/bigsql,/data8/db2/bigsql,/data9/db2/bigsql

Cache directory:
/data1/hadoop/mapred/local,/data2/hadoop/mapred/local,/data3/hadoop/mapred/local,/data4/hadoop
/mapred/local,/data5/hadoop/mapred/local,/data6/hadoop/mapred/local,/data7/hadoop/mapred/local
,/data8/hadoop/mapred/local,/data9/hadoop/mapred/local

DataNode data directory:
/data1/hadoop/hdfs/data,/data2/hadoop/hdfs/data,/data3/hadoop/hdfs/data,/data4/hadoop/hdfs/dat
a,/data5/hadoop/hdfs/data,/data6/hadoop/hdfs/data,/data7/hadoop/hdfs/data,/data8/hadoop/hdfs/d
ata,/data9/hadoop/hdfs/data
```

Big SQL tuning options:

```
## Configured for 128 GB of memory per node
## 30 GB bufferpool
## 3.125 GB sortheap / 50 GB sheapthres_shr
## reader memory: 20% of total memory by default (user can raise it to 30%)
##
## other useful conf changes:
##   mapred-site.xml
##     mapred.tasktracker.map.tasks.maximum=20
##     mapred.tasktracker.reduce.tasks.maximum=6
##     mapreduce.map.java.opts="-Xmx3000m ..."
##     mapreduce.reduce.java.opts="-Xmx3000m ..."
##
##   bigsql-conf.xml
##     dfsio.num_scanner_threads=12
##     dfsio.read_size=4194304
##     dfsio.num_threads_per_disk=2
##     scheduler.client.request.timeout=600000
```

```
DBNAME=$1

db2 connect to ${DBNAME}

db2 -v "call syshadoop.big_sql_service_mode('on')"
db2 -v "alter bufferpool  IBMDEFAULTBP size 891520 "
## db2 -v "alter tablespace TEMPSPACE1 no file system caching"
db2 -v "update db cfg for ${DBNAME} using sortheap 819200 sheapthres_shr 13107200"
db2 -v "update db cfg for ${DBNAME} using dft_degree 8"
db2 -v "update dbm cfg using max_querydegree ANY"
db2 -v "update dbm cfg using aslheapsz 15"
db2 -v "update dbm cfg using cpuspeed 1.377671e-07"
db2 -v "update dbm cfg using INSTANCE_MEMORY 85"

## Added by Simon on 31/09. Useful for T'put run
db2 -v "update dbm cfg using CONN_ELAPSE 18"

## Added by Simon on 31/09. Disable auto maintenance
db2 -v "update db cfg for bigsql using AUTO_MAINT OFF AUTO_TBL_MAINT OFF AUTO_RUNSTATS OFF
AUTO_STMT_STATS OFF"

db2 terminate
```

BigInsights mapred-site.xml tuning:

The following changes (highlighted) were made to the Hadoop mapred-site.xml file to tune the number of map-reduce slots, and the maximum memory allocated to these slots. In Big SQL, Map-Reduce is used for the LOAD and ANALYZE commands only, not query execution. The properties were tuned in order to get the best possible performance from these commands.

```
  <property>
    <!-- The maximum number of map tasks that will be run simultaneously by a
         task tracker. Default: 2. Recommendations: set relevant to number of
         CPUs and amount of memory on each data node. -->
    <name>mapred.tasktracker.map.tasks.maximum</name>
    <!--value><%= Math.max(2, Math.ceil(0.66 * Math.min(numOfDisks, numOfCores, totalMem/1000)
* 1.75) - 2) %></value-->
    <value>20</value>
  </property>


  <property>
    <!-- The maximum number of reduce tasks that will be run simultaneously by
         a task tracker. Default: 2. Recommendations: set relevant to number of
         CPUs and amount of memory on each data node, note that reduces usually
         take more memory and do more I/O than maps. -->
    <name>mapred.tasktracker.reduce.tasks.maximum</name>
    <!--value><%= Math.max(2, Math.ceil(0.33 * Math.min(numOfDisks, numOfCores, totalMem/1000)
* 1.75) - 2)%></value-->
    <value>6</value>
```

```
  </property>


  <property>
    <!-- Max heap of child JVM spawned by tasktracker. Ideally as large as the
         task machine can afford. The default -Xmx200m is usually too small. -->
    <name>mapreduce.map.java.opts</name>
    <value>-Xmx3000m -Xms1000m -Xmn100m -Xtune:virtualized -
Xshareclasses:name=mrscc_%g,groupAccess,cacheDir=/var/ibm/biginsights/hadoop/tmp,nonFatal -
Xscmx20m -Xdump:java:file=/var/ibm/biginsights/hadoop/tmp/javacore.%Y%m%d.%H%M%S.%pid.%seq.txt
-Xdump:heap:file=/var/ibm/biginsights/hadoop/tmp/heapdump.%Y%m%d.%H%M%S.%pid.%seq.phd</value>
  </property>


  <property>
    <!-- Max heap of child JVM spawned by tasktracker. Ideally as large as the
         task machine can afford. The default -Xmx200m is usually too small. -->
    <name>mapreduce.reduce.java.opts</name>
    <value>-Xmx3000m -Xms1000m -Xmn100m -Xtune:virtualized -
Xshareclasses:name=mrscc_%g,groupAccess,cacheDir=/var/ibm/biginsights/hadoop/tmp,nonFatal -
Xscmx20m -Xdump:java:file=/var/ibm/biginsights/hadoop/tmp/javacore.%Y%m%d.%H%M%S.%pid.%seq.txt
-Xdump:heap:file=/var/ibm/biginsights/hadoop/tmp/heapdump.%Y%m%d.%H%M%S.%pid.%seq.phd</value>
  </property>
```

Big SQL dfs reader options:

The following properties were changed in the Big SQL bigsql-conf.xml file to tune dfs reader properties:

```
  <property>
    <name>scheduler.client.request.timeout</name>
    <value>600000</value>
    <description>
      Scheduler clients will wait for scheduler to respond for
      these many milli-seconds before timing out.
  </description>
  </property>
……
  <property>
    <!-- Number of threads reading from each disk.
         Set this to 0 to use default values. -->
    <name>dfsio.num_threads_per_disk</name>
    <value>2</value>
    <!--value>0</value-->
  </property>
  <property>
    <!-- Read Size (in bytes) - Size of the reads sent to Hdfs (i.e., also the max I/O read
buffer size).
         Default is 8*1024*1024 = 8388608 bytes -->
    <name>dfsio.read_size</name>
    <value>4194304</value>
    <!--value>8388608</value-->
  </property>
```

```
…..
  <property>
    <!-- (Advanced) Cap on the number of scanner threads that will be created.
                    If set to 0, the system decides. -->
    <name>dfsio.num_scanner_threads</name>
    <value>12</value>
  </property>
```

Big SQL dfs logging:

The minLogLevel property was changed in the Big SQL glog-dfsio.properties file to reduce the amount of logging by the dfs readers:

```
glog_enabled=true
log_dir=/var/ibm/biginsights/bigsql/logs
log_filename=bigsql-ndfsio.log

# 0 - INFO
# 1 - WARN
# 2 - ERROR
# 3 - FATAL
minloglevel=3
```

OS kernel changes:

```
echo 0 > /proc/sys/vm/swappiness

echo "net.ipv6.conf.all.disable_ipv6 = 1" >> /etc/sysctl.conf
```

Active Hadoop components:

In order to release valuable resources on the cluster only the following BigInsights components were started during the single and multi-stream runs: bigsql, Hadoop, hive, catalog, zookeeper and console.

C.2: Impala Configuration

Linux Configuration changes:

The following Linux kernel configuration changes were made to the Impala cluster:

```
echo 0 > /proc/sys/vm/swappiness

sysctl -w net.core.somaxconn=1024

echo "net.ipv6.conf.all.disable_ipv6 = 1" >> /etc/sysctl.conf
```

Cloudera Configuration:

The following table describes the Cloudera and Impala tuning undertaken during the benchmark.

Category	Property	Value	Description
Cloudera Manager/Hosts Configuration/Other	Memory Overcommit Validation Threshold	0.98	Threshold used when validating the allocation of RAM on a host. 0 means all of the memory is reserved for the system. 1 means none is reserved. Values can range from 0 to 1.
	Enable Cgroup-based Resource Management	true	Enables resource management using control groups (cgroups) for this host. Once toggled, roles on this host must be restarted for cgroups to be enabled or disabled. Per-resource controls can be found in the configuration pages of role configuration groups and individual roles. Cgroups are a feature of the Linux kernel, and as such, support varies by distribution; consult the Cloudera Manager documentation for details.
Dynamic Resource Pools	Name	tpcds	
	Scheduling Policy	FAIR	Schedules resources fairly based only on memory
	Max Running Queries	3	Maximum number of concurrently running queries in the pool (optional)

HDFS Configuration/Service-Wide	Zookeeper Service	Zookeeper	Name of the ZooKeeper service that this HDFS service depends on.
HDFS Configuration/Service-Wide/Advanced	HDFS Service Advanced Configuration Snippet (Safety Valve) for hdfs-site.xml	<property> <name>dfs.datanode.hdfs-blocks-metadata-enabled</name> <value>true</value> </property>	For advanced use only, a string to be inserted into hdfs-site.xml. Applies to configurations of all roles in this service except client configuration.
HDFS Configuration/Service-Wide/Performance	DataNode Local Path Access Users	impala	Comma separated list of users allowed to do short circuit read. A short circuit read allows a client co-located with the data to read HDFS file blocks directly from HDFS. If empty, will default to the DataNode process' user.
HDFS Configuration/Balancer Default Group	Rebalancing Policy	BlockPool	The policy that should be used to rebalance HDFS storage. The default DataNode policy balances the storage at the DataNode level. This is similar to the balancing policy from prior releases. The BlockPool policy balances the storage at the block pool level as well as at the Datanode level. The BlockPool policy is relevant only to a Federated HDFS

			service.
HDFS Configuration/DataNode Default Group	DataNode Data Directory	/data1/dfs/dn, /data2/dfs/dn, /data3/dfs/dn, /data4/dfs/dn, /data5/dfs/dn, /data6/dfs/dn, /data7/dfs/dn, /data8/dfs/dn, /data9/dfs/dn	Comma-delimited list of directories on the local file system where the DataNode stores HDFS block data. Typical values are /data/N/dfs/dn for N = 1, 2, 3… These directories should be mounted using the noatime option and the disks should be configured using JBOD. RAID is not recommended.
	DataNode Failed Volumes Tolerated	3	The number of volumes that are allowed to fail before a DataNode stops offering service. By default, any volume failure will cause a DataNode to shutdown.
HDFS Configuration/DataNode Default Group/Resource Management	Java Heap Size of DataNode in Bytes	512 MiB	Maximum size in bytes for the Java Process heap memory. Passed to Java –Xmx.
	Maximum Memory Used for Caching	1 GiB	The maximum amount of memory a DataNode may use to cache data blocks in memory. Setting it to 0 will disable caching.
HDFS Configuration/DataNode Default Group/Security	DataNode Data Directory Permissions	755	Permissions for the directories on the local file system where the DataNode stores its blocks. The permissions must be

			octal. 755 and 700 are typical values.
HDFS Configuration/DataNode Group 1	DataNode Data Directory	/data1/dfs/dn, /data2/dfs/dn, /data3/dfs/dn, /data4/dfs/dn, /data5/dfs/dn, /data6/dfs/dn, /data7/dfs/dn, /data8/dfs/dn, /data9/dfs/dn	Comma-delimited list of directories on the local file system where the DataNode stores HDFS block data. Typical values are /data/N/dfs/dn for N = 1, 2, 3… These directories should be mounted using the noatime option and the disks should be configured using JBOD. RAID is not recommended.
	DataNode Failed Volumes Tolerated	3	The number of volumes that are allowed to fail before a DataNode stops offering service. By default, any volume failure will cause a DataNode to shutdown.
HDFS Configuration/DataNode Group 1/Resource Management	Java Heap Size of DataNode in Bytes	920 MiB	Maximum size in bytes for the Java Process heap memory. Passed to Java –Xmx.
	Maximum Memory Used for Caching	3658 MiB	The maximum amount of memory a DataNode may use to cache data blocks in memory. Setting it to 0 will disable caching.
HDFS Configuration/DataNode Group 1/Security	DataNode Data Directory Permissions	755	Permissions for the directories on the local file system where the DataNode stores its blocks. The

			permissions must be octal. 755 and 700 are typical values.
HDFS Configuration/Gateway Default Group	Use Trash	true	Move deleted files to the trash so that they can be recovered if necessary. The client side configuration takes effect only if the HDFS service-wide trash is disabled (NameNode Filesystem Trash Interval set to 0) and is ignored otherwise. The trash is not automatically emptied when enabled with this configuration.
HDFS Configuration/NameNode Default Group	NameNode Data Directories	/data1/dfs/nn, /data2/dfs/nn	Determines where on the local file system the NameNode should store the name table (fsimage). For redundancy, enter a comma-delimited list of directories to replicate the name table in all of the directories. Typical values are /data/N/dfs/nn where N=1..3.
HDFS Configuration/NameNode Default Group/Performance	NameNode Handler Count	55	The number of server threads for the NameNode.
	NameNode Service Handler Count	55	The number of server threads for the NameNode used for service calls. Only used when NameNode Service RPC Port is

			configured.
HDFS Configuration/NameNode Default Group/Ports and Addresses	NameNode Service RPC Port	8022	Optional port for the service-rpc address which can be used by HDFS daemons instead of sharing the RPC address used by clients.
HDFS Configuration/NameNode Default Group/Resource Management	Java Heap Size of Namenode in Bytes	2111 MiB	Maximum size in bytes for the Java Process heap memory. Passed to Java -Xmx.
HDFS Configuration/SecondaryNameNode Default Group	HDFS Checkpoint Directory	/data1/dfs/snn	Determines where on the local file system the DFS SecondaryNameNode should store the temporary images to merge. For redundancy enter a comma-delimited list of directories to replicate the image in all of the directories. Typical values are /data/N/dfs/snn for N = 1, 2, 3…
HDFS Configuration/SecondaryNameNode Default Group/Resource Management	Java Heap Size of Secondary namenode in Bytes	2111 MiB	Maximum size in bytes for the Java Process heap memory. Passed to Java –Xmx.
Impala Configuration/Service-Wide	HDFS Service	HDFS	Name of the HDFS service that this Impala service instance depends on
	Hive Service	Hive	Name of the Hive service that this Impala service instance

			depends on
	Yarn Service for Resource Management	none	Name of YARN service to use for resource management integration between Impala and YARN. This service dependency and the existence of a Llama role is required for using said integration.
Impala Configuration/Service-Wide/Advanced	Maximum HBase Client Retries	30	Maximum number of HBase client retries for Impala. Used as a maximum for all operations such as fetching of the root region from the root RegionServer, getting a cell's value, and starting a row update. Overrides configuration in the HBase service.
	HBase RPC Timeout	30 seconds	Timeout in milliseconds for all HBase RPCs made by Impala. Overrides configuration in HBase service.
	Impala Command Line Argument Advanced Configuration Snippet (Safety Valve)	-llama_host=bigaperf180.svl.ibm.com -queue_wait_timeout_ms=1800000	For advanced use only, key-value pairs (one on each line) to be added (verbatim) to Impala Daemon command-line flags. Applies to all roles in this service. Key names should begin with a hyphen(-). **For example:** -log_filename=foo.log.

	Impala Service Environment Advanced Configuration Snippet (Safety Valve)	JAVA_TOOL_OPTIONS="-Xmx2G"	For advanced use only, key-value pairs (one on each line) to be inserted into a role's environment. Applies to configuration of all roles in this service except client configuration.
Impala Configuration/Service-Wide/Performance	StateStore Subscriber Timeout	4 hour(s)	Time in seconds before Impala Daemon or Catalog Server times out with the StateStore.
Impala Configuration/Impala Catalog Server Default Group/Advanced	Catalog Server Hive Metastore Connection Timeout	4 hour(s)	Timeout for requests to the Hive Metastore Server from Catalog Server. Consider increasing this if you have tables with a lot of metadata and see timeout errors.
Impala Configuration/Impala Catalog Server Default Group/Monitoring	Catalog Server Connectivity Tolerance at startup	30 minutes(s)	The amount of time to wait for the Catalog Server to fully start up and connect to the StateStore before enforcing the connectivity check.
Impala Configuration/Impala Daemon Default Group	Impala Daemon Scratch Directories	/data1/impala/impalad, /data2/impala/impalad, /data3/impala/impalad, /data4/impala/impalad, /data5/impala/impalad, /data6/impala/impalad, /data7/impala/impalad, /data8/impala/impalad	Directories where Impala Daemon will write data such as spilling information to disk to free up memory. This can potentially be large amounts of data.
Impala	Impala Daemon	RESERVATION_REQUEST_	A list of key-value pairs

Configuration/Impala Daemon Default Group	Query Options Advanced Configuration Snippet (Safety Valve)	TIMEOUT=1800000;	of additional query options to pass to the Impala Daemon command line, separated by ‘,’.
Impala Configuration/Impala Daemon Default Group/Advanced	Impala Daemon Hive Metastore Connection Timeout	4 hour(s)	Timeout for requests to the Hive Metastore Server from Impala. Consider increasing this if you have a lot of metadata and see timeout errors.
Impala Configuration/Impala Daemon Default Group/Monitoring	Impala Daemon Connectivity Tolerance at Startup	30 minute(s)	The amount of time to wait for the Impala Daemon to fully start up and connect to the StateStore before enforcing the connectivity check.
	Query Monitoring Timeout: 50 second(s) The timeout used by Cloudera Manager Agent’s query monitor when communicating with the Impala Daemon web server, specified in seconds.	Query Monitoring Timeout: 50 second(s) The timeout used by Cloudera Manager Agent’s query monitor when communicating with the Impala Daemon web server, specified in seconds.	Query Monitoring Timeout: 50 second(s) The timeout used by Cloudera Manager Agent’s query monitor when communicating with the Impala Daemon web server, specified in seconds.
Impala Configuration/Impala Daemon Default Group/Monitoring	Impala Configuration/Impala Daemon Default Group/Monitor	Impala Configuration/Impala Daemon Default Group/Monitoring	Impala Configuration/Impala Daemon Default Group/Monitoring

Query Monitoring Period: 1 second(s) The polling period of the Impala query Manager Agent, specified in seconds. If set to zero, query monitoring is disabled.	ing Query Monitoring Period: 1 second(s) The polling period of the Impala query Manager Agent, specified in seconds. If set to zero, query monitoring is disabled.	Query Monitoring Period: 1 second(s) The polling period of the Impala query Manager Agent, specified in seconds. If set to zero, query monitoring is disabled.	Query Monitoring Period: 1 second(s) The polling period of the Impala query Manager Agent, specified in seconds. If set to zero, query monitoring is disabled.
Impala Configuration/Impala Llama ApplicationMaster Default Group/Advanced	Resource Caching Idle Timeout	1 minute(s)	Timeout policy for resources being cached.
	Maximum Client Notification Retries	50	Maximum number of retries for a client notification. After the maximum number of client notification retries has been reached without success the client is considered lost ad all its reservations are released. A successful client notification resets.
Impala Configuration/Impala Llama ApplicationMaster Default Group/Performance	Thrift Transport Timeout	15 minute(s)	Socket time, in milliseconds, used Llama ApplicationMaster auxiliary service for all its server and client

			Thrift connections.
Hive Configuration/Service-Wide	MapReduce Service	YARN (MR2 Included)	MapReduce jobs are run against this service.
	Zookeeper Service	Zookeeper	Name of the ZooKeeper service hat this Hive service instance depends on.
Hive Configuration/Service-Wide/Hive Metastore Database	Hive Metastore Database Type	postgresql	Type of Hive Metastore database. Note that Derby is not recommended and Cloudera Impala does not support Derby.
	Hive Metastore Database Name	Hive	Name of Hive Metastore database.
	Hive Metastore Database Host	BigAPerf180.svl.ibm.com	Host name of Hive Metastore database.
	Hive Metastore Database Port	7432	Port number of Hive Metastore Database.
	Hive Metastore Database Password	XXXXXX	Password for Hive Metastore database.
ZooKeeper Configuration/Server Default Group/Resource Management	Java Heap Size of ZooKeeper Server in Bytes	920 MiB	Maximum size in bytes for the Java Process heap memory. Passed to Java –Xmx.

C.3: Hive Configuration

Tez was used as the Hive 0.13 execution engine.
The following list provides a summary of the configuration changes, after which the detailed tuning is documented:

1. The number of containers was set to 21
2. Size of containers was 5GB.
3. The available memory was set to 105GB, to satisfy 21 containers
4. The Java heap for Reduce tasks was set 2x to Java heap Map tasks

5. Tez session timeout (both client and submit) were set to 14400, or 4 hrs timeout, so they never timeout during submit. When the system is very busy, the job remains in submit mode, and waits until it gets enough resource to run. If the job waited for a long time,. It would timeout.
6. A number of memory parameters in mapred-site.xml, yarn-site.xml, tez-site.xml andd hive-site.xml were set based on calculation based on the documentation and through experimentation based on 5G container.
7. HBase & Ozzie were stopped to free-up memory
8. Bucket tables with appropriate split size were used

mapred-site:

mapreduce.map.memory.mb=5120
mapreduce.map.java.opts=4096
mapreduce.reduce.memory.mb=10240
mapreduce.reduce.java.opts=8192
mapreduce.task.io.sort.mb=1792

yarn-site.xml:

yarn.scheduler.minimum-allocation-mb=5120
yarn.scheduler.maximum-allocation-mb=108544
yarn.nodemanager.resource.memory-mb=108544
yarn.app.mapreduce.am.resource.mb=10240
yarn.app.mapreduce.am.command-opts=8192

tez-site.xml:

tez.am.resource.memory.mb=5120
tez.am.java.opts=4096
tez.session.am.dag.submit.timeout.secs=14400
tez.session.client.timeout.secs=14400

hive-site.xml:

hive.tez.container.size=5120
hive.tez.java.opts=4608
hive.auto.convert.join.noconditionaltask.size=1342177000

Execution settings:

These settings were passed for each query execution in Hive. Since they are the same for all queries they do not constitue per query tuning.

```
set ambari.hive.db.schema.name=hive;
set fs.file.impl.disable.cache=true;
set fs.hdfs.impl.disable.cache=true;
set hive.auto.convert.join.noconditionaltask=true;
set hive.auto.convert.join=true;
set hive.auto.convert.sortmerge.join.noconditionaltask=true;
set hive.auto.convert.sortmerge.join=true;
set hive.compactor.abortedtxn.threshold=1000;
set hive.compactor.check.interval=300;
set hive.compactor.delta.num.threshold=10;
set hive.compactor.delta.pct.threshold=0.1f;
set hive.compactor.initiator.on=false;
set hive.compactor.worker.threads=0;
set hive.compactor.worker.timeout=86400;
```

```
set hive.compute.query.using.stats=true;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.enforce.sortmergebucketmapjoin=true;
set hive.exec.failure.hooks=org.apache.hadoop.hive.ql.hooks.ATSHook;
set hive.exec.post.hooks=org.apache.hadoop.hive.ql.hooks.ATSHook;
set hive.exec.pre.hooks=org.apache.hadoop.hive.ql.hooks.ATSHook;
set hive.execution.engine=tez;
set hive.limit.pushdown.memory.usage=0.04;
set hive.map.aggr=true;
set hive.mapjoin.bucket.cache.size=10000;
set hive.mapred.reduce.tasks.speculative.execution=false;
set hive.metastore.cache.pinobjtypes=Table,Database,Type,FieldSchema,Order;
set hive.metastore.client.socket.timeout=60;
set hive.metastore.execute.setugi=true;
set hive.metastore.warehouse.dir=/apps/hive/warehouse;
set hive.optimize.bucketmapjoin.sortedmerge=false;
set hive.optimize.bucketmapjoin=true;
set hive.enforce.bucketmapjoin=true;
set hive.optimize.index.filter=true;
set hive.optimize.mapjoin.mapreduce=true;
set hive.optimize.reducededuplication.min.reducer=4;
set hive.optimize.reducededuplication=true;
set hive.orc.splits.include.file.footer=false;

set hive.security.authorization.enabled=false;
set
hive.security.metastore.authorization.manager=org.apache.hadoop.hive.ql.security.authorization.Stor
ageBasedAuthorizationProvider;
set hive.semantic.analyzer.factory.impl=org.apache.hivealog.cli.HCatSemanticAnalyzerFactory;
set hive.server2.enable.doAs=false;
set hive.server2.tez.default.queues=default;
set hive.server2.tez.initialize.default.sessions=false;
set hive.server2.tez.sessions.per.default.queue=1;
set hive.tez.input.format=org.apache.hadoop.hive.ql.io.HiveInputFormat;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;
set hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DummyTxnManager;
set hive.txn.max.open.batch=1000;
set hive.txn.timeout=600;
set hive.vectorized.execution.enabled=true;
set hive.vectorized.groupby.checkinterval=1024;
set hive.vectorized.groupby.flush.percent=1;
set hive.vectorized.groupby.maxentries=1024;
set hive.exec.parallel=true;
set hive.merge.mapfiles=true;
set mapred.output.compress=true;

set hive.optimize.tez=true;
set dfs.blocksize=1024000000;
```

C.4 OS Storage Configuration:

The following script was used to create ext4 filesystems on all disks (used to store data) on all nodes in the cluster (inc. the master) for each product.

All three vendors used the same storage layout.

Note that the SSDs were not used during the benchmark.

```
#!/bin/bash
# READ / WRITE Performance tests for EXT4 file systems
# Author - Stewart Tate, tates@us.ibm.com
# Copyright (C) 2013, IBM Corp. All rights reserved.:
#################################################################
# the follow is server unique and MUST be adjusted!             #
#################################################################

 drives=(b g h i j k l m n)
 SSDdrives=(c d e f)

 echo "Create EXT4 file systems, version 130213b"
 echo " "

pause()
{
  sleep 2
}

 # make ext4 file systems on HDDs
 echo "Create EXT4 file systems on HDDs"
 for dev_range in ${drives[@]}
 do
   echo "y" | mkfs.ext4 -b 4096 -O dir_index,extent /dev/sd$dev_range
 done

 for dev_range in ${drives[@]}
 do
  parted /dev/sd$dev_range print
 done

 pause


 # make ext4 file systems on SSDs
 echo "Create EXT4 file systems on SSDs"
 for dev_range in ${SSDdrives[@]}
 do
   echo "y" | mkfs.ext4 -b 4096 -O dir_index,extent /dev/sd$dev_range
 done

 for dev_range in ${SSDdrives[@]}
 do
  parted /dev/sd$dev_range print
  echo "Partitions aligned(important for performance) if following returns 0:"
  blockdev --getalignoff /dev/sd$dev_range
 done

exit
```

The filesystems are then mounted using the following script:

```
#!/bin/bash
# READ / WRITE Performance tests for EXT4 file systems
# Author - Stewart Tate, tates@us.ibm.com
# Copyright (C) 2013, IBM Corp. All rights reserved.:
#################################################################
# the follow is server unique and MUST be adjusted!             #
#################################################################

 drives=(b g h i j k l m n)
 SSDdrives=(c d e f)


 echo "Mount EXT4 file systems, version 130213b"
 echo " "

pause()
```

```
{
  sleep 2
}

 j=0
 echo "Create EXT4 mount points for HDDs"
 for i in ${drives[@]}
 do
   let j++
   mkdir /data$j
   mount -vs -t ext4 -o
nobarrier,noatime,nodiratime,nobh,nouser_xattr,data=writeback,commit=100 /dev/sd$i
/data$j
 done

 j=0
 echo "Create EXT4 mount points for SSDs"
 for i in ${SSDdrives[@]}
 do
   let j++
   mkdir /datassd$j
   mount -vs -t ext4 -o
nobarrier,noatime,nodiratime,discard,nobh,nouser_xattr,data=writeback,commit=100
/dev/sd$i /datassd$j
 done


echo "Done."

exit
```

Appendix D: Database Schema

D.1 Big SQL

Since the Parquet format does not support Date data type, VARCHAR(10) was used to store dates.

```
set schema $schema;

create hadoop table call_center
(
    cc_call_center_sk         bigint
not null,
    cc_call_center_id         varchar(16)
not null,
    cc_rec_start_date         varchar(10)
,
    cc_rec_end_date           varchar(10)
,
    cc_closed_date_sk         bigint
,
    cc_open_date_sk           bigint
,
    cc_name                   varchar(50)
,
    cc_class                  varchar(50)
,
    cc_employees              bigint
,
    cc_sq_ft                  bigint
,
    cc_hours                  varchar(20)
,
    cc_manager                varchar(40)
,
    cc_mkt_id                 bigint
,
    cc_mkt_class              varchar(50)
,
    cc_mkt_desc               varchar(100)
,
    cc_market_manager         varchar(40)
,
    cc_division               bigint
,
    cc_division_name          varchar(50)
,
    cc_company                bigint
,
    cc_company_name           varchar(50)
,
    cc_street_number          varchar(10)
,
    cc_street_name            varchar(60)
,
    cc_street_type            varchar(15)
,
    cc_suite_number           varchar(10)
,
    cc_city                   varchar(60)
,
    cc_county                 varchar(30)
,
    cc_state                  varchar(2)
,
    cc_zip                    varchar(10)
,
    cc_country                varchar(20)
,
    cc_gmt_offset             double
,
    cc_tax_percentage         double
)
STORED AS PARQUETFILE;

create hadoop table catalog_page
(
    cp_catalog_page_sk        bigint
not null,
    cp_catalog_page_id        varchar(16)
not null,
    cp_start_date_sk          bigint
,
    cp_end_date_sk            bigint
,
    cp_department             varchar(50)
,
    cp_catalog_number         bigint
,
    cp_catalog_page_number    bigint
,
    cp_description            varchar(100)
,
    cp_type                   varchar(100)
)
STORED AS PARQUETFILE;

create hadoop table catalog_returns
(
    cr_returned_date_sk       bigint
,
    cr_returned_time_sk       bigint
,
    cr_item_sk                bigint
not null,
    cr_refunded_customer_sk   bigint
,
    cr_refunded_cdemo_sk      bigint
,
    cr_refunded_hdemo_sk      bigint
,
    cr_refunded_addr_sk       bigint
,
    cr_returning_customer_sk  bigint
,
    cr_returning_cdemo_sk     bigint
,
    cr_returning_hdemo_sk     bigint
,
    cr_returning_addr_sk      bigint
,
    cr_call_center_sk         bigint
,
```

```
    cr_catalog_page_sk        bigint
,
    cr_ship_mode_sk           bigint
,
    cr_warehouse_sk           bigint
,
    cr_reason_sk              bigint
,
    cr_order_number           bigint
not null,
    cr_return_quantity        bigint
,
    cr_return_amount          double
,
    cr_return_tax             double
,
    cr_return_amt_inc_tax     double
,
    cr_fee                    double
,
    cr_return_ship_cost       double
,
    cr_refunded_cash          double
,
    cr_reversed_charge        double
,
    cr_store_credit           double
,
    cr_net_loss               double
)
STORED AS PARQUETFILE;

create hadoop table catalog_sales
(
    cs_sold_date_sk           bigint
,
    cs_sold_time_sk           bigint
,
    cs_ship_date_sk           bigint
,
    cs_bill_customer_sk       bigint
,
    cs_bill_cdemo_sk          bigint
,
    cs_bill_hdemo_sk          bigint
,
    cs_bill_addr_sk           bigint
,
    cs_ship_customer_sk       bigint
,
    cs_ship_cdemo_sk          bigint
,
    cs_ship_hdemo_sk          bigint
,
    cs_ship_addr_sk           bigint
,
    cs_call_center_sk         bigint
,
    cs_catalog_page_sk        bigint
,
    cs_ship_mode_sk           bigint
,
    cs_warehouse_sk           bigint
,
    cs_item_sk                bigint
not null,
    cs_promo_sk               bigint
,
    cs_order_number           bigint
not null,
    cs_quantity               bigint
,
    cs_wholesale_cost         double
,
    cs_list_price             double
,
    cs_sales_price            double
,
    cs_ext_discount_amt       double
,
    cs_ext_sales_price        double
,
    cs_ext_wholesale_cost     double
,
    cs_ext_list_price         double
,
    cs_ext_tax                double
,
    cs_coupon_amt             double
,
    cs_ext_ship_cost          double
,
    cs_net_paid               double
,
    cs_net_paid_inc_tax       double
,
    cs_net_paid_inc_ship      double
,
    cs_net_paid_inc_ship_tax  double
,
    cs_net_profit             double
)
STORED AS PARQUETFILE;

create hadoop table customer
(
    c_customer_sk             bigint
not null,
    c_customer_id             varchar(16)
not null,
    c_current_cdemo_sk        bigint
,
    c_current_hdemo_sk        bigint
,
    c_current_addr_sk         bigint
,
    c_first_shipto_date_sk    bigint
,
    c_first_sales_date_sk     bigint
,
    c_salutation              varchar(10)
,
    c_first_name              varchar(20)
,
    c_last_name               varchar(30)
,
    c_preferred_cust_flag     varchar(1)
,
    c_birth_day               bigint
,
    c_birth_month             bigint
,
```

```
    c_birth_year              bigint
,
    c_birth_country           varchar(20)
,
    c_login                   varchar(13)
,
    c_email_address           varchar(50)
,
    c_last_review_date        bigint
)
STORED AS PARQUETFILE;

create hadoop table customer_address
(
    ca_address_sk             bigint
not null,
    ca_address_id             varchar(16)
not null,
    ca_street_number          varchar(10)
,
    ca_street_name            varchar(60)
,
    ca_street_type            varchar(15)
,
    ca_suite_number           varchar(10)
,
    ca_city                   varchar(60)
,
    ca_county                 varchar(30)
,
    ca_state                  varchar(2)
,
    ca_zip                    varchar(10)
,
    ca_country                varchar(20)
,
    ca_gmt_offset             double
,
    ca_location_type          varchar(20)
)
STORED AS PARQUETFILE;

create hadoop table customer_demographics
(
    cd_demo_sk                bigint
not null,
    cd_gender                 varchar(1)
,
    cd_marital_status         varchar(1)
,
    cd_education_status       varchar(20)
,
    cd_purchase_estimate      bigint
,
    cd_credit_rating          varchar(10)
,
    cd_dep_count              bigint
,
    cd_dep_employed_count     bigint
,
    cd_dep_college_count      bigint
)
STORED AS PARQUETFILE;

create hadoop table date_dim
(
    d_date_sk                 bigint
not null,
    d_date_id                 varchar(16)
not null,
    d_date                    varchar(10)
,
    d_month_seq               bigint
,
    d_week_seq                bigint
,
    d_quarter_seq             bigint
,
    d_year                    bigint
,
    d_dow                     bigint
,
    d_moy                     bigint
,
    d_dom                     bigint
,
    d_qoy                     bigint
,
    d_fy_year                 bigint
,
    d_fy_quarter_seq          bigint
,
    d_fy_week_seq             bigint
,
    d_day_name                varchar(9)
,
    d_quarter_name            varchar(6)
,
    d_holiday                 varchar(1)
,
    d_weekend                 varchar(1)
,
    d_following_holiday       varchar(1)
,
    d_first_dom               bigint
,
    d_last_dom                bigint
,
    d_same_day_ly             bigint
,
    d_same_day_lq             bigint
,
    d_current_day             varchar(1)
,
    d_current_week            varchar(1)
,
    d_current_month           varchar(1)
,
    d_current_quarter         varchar(1)
,
    d_current_year            varchar(1)
)
STORED AS PARQUETFILE;

create hadoop table household_demographics
(
    hd_demo_sk                bigint
not null,
    hd_income_band_sk         bigint
,
    hd_buy_potential          varchar(15)
,
```

```
    hd_dep_count              bigint
,
    hd_vehicle_count          bigint
)
STORED AS PARQUETFILE;

create hadoop table income_band
(
    ib_income_band_sk         bigint
not null,
    ib_lower_bound            bigint
,
    ib_upper_bound            bigint
)
STORED AS PARQUETFILE;

create hadoop table inventory
(
    inv_date_sk               bigint
not null,
    inv_item_sk               bigint
not null,
    inv_warehouse_sk          bigint
not null,
    inv_quantity_on_hand      bigint
)
STORED AS PARQUETFILE;

create hadoop table item
(
    i_item_sk                 bigint
not null,
    i_item_id                 varchar(16)
not null,
    i_rec_start_date          varchar(10)
,
    i_rec_end_date            varchar(10)
,
    i_item_desc               varchar(200)
,
    i_current_price           double
,
    i_wholesale_cost          double
,
    i_brand_id                bigint
,
    i_brand                   varchar(50)
,
    i_class_id                bigint
,
    i_class                   varchar(50)
,
    i_category_id             bigint
,
    i_category                varchar(50)
,
    i_manufact_id             bigint
,
    i_manufact                varchar(50)
,
    i_size                    varchar(20)
,
    i_formulation             varchar(20)
,
    i_color                   varchar(20)
,
    i_units                   varchar(10)
,
    i_container               varchar(10)
,
    i_manager_id              bigint
,
    i_product_name            varchar(50)
)
STORED AS PARQUETFILE;

create hadoop table promotion
(
    p_promo_sk                bigint
not null,
    p_promo_id                varchar(16)
not null,
    p_start_date_sk           bigint
,
    p_end_date_sk             bigint
,
    p_item_sk                 bigint
,
    p_cost                    double
,
    p_response_target         bigint
,
    p_promo_name              varchar(50)
,
    p_channel_dmail           varchar(1)
,
    p_channel_email           varchar(1)
,
    p_channel_catalog         varchar(1)
,
    p_channel_tv              varchar(1)
,
    p_channel_radio           varchar(1)
,
    p_channel_press           varchar(1)
,
    p_channel_event           varchar(1)
,
    p_channel_demo            varchar(1)
,
    p_channel_details         varchar(100)
,
    p_purpose                 varchar(15)
,
    p_discount_active         varchar(1)
)
STORED AS PARQUETFILE;

create hadoop table reason
(
    r_reason_sk               bigint
not null,
    r_reason_id               varchar(16)
not null,
    r_reason_desc             varchar(100)
)
STORED AS PARQUETFILE;

create hadoop table ship_mode
(
    sm_ship_mode_sk           bigint
not null,
```

```
    sm_ship_mode_id           varchar(16)
not null,
    sm_type                   varchar(30)
,
    sm_code                   varchar(10)
,
    sm_carrier                varchar(20)
,
    sm_contract               varchar(20)
)
STORED AS PARQUETFILE;

create hadoop table store
(
    s_store_sk                bigint
not null,
    s_store_id                varchar(16)
not null,
    s_rec_start_date          varchar(10)
,
    s_rec_end_date            varchar(10)
,
    s_closed_date_sk          bigint
,
    s_store_name              varchar(50)
,
    s_number_employees        bigint
,
    s_floor_space             bigint
,
    s_hours                   varchar(20)
,
    s_manager                 varchar(40)
,
    s_market_id               bigint
,
    s_geography_class         varchar(100)
,
    s_market_desc             varchar(100)
,
    s_market_manager          varchar(40)
,
    s_division_id             bigint
,
    s_division_name           varchar(50)
,
    s_company_id              bigint
,
    s_company_name            varchar(50)
,
    s_street_number           varchar(10)
,
    s_street_name             varchar(60)
,
    s_street_type             varchar(15)
,
    s_suite_number            varchar(10)
,
    s_city                    varchar(60)
,
    s_county                  varchar(30)
,
    s_state                   varchar(2)
,
    s_zip                     varchar(10)
,
    s_country                 varchar(20)
,
    s_gmt_offset              double
,
    s_tax_precentage          double
)
STORED AS PARQUETFILE;

create hadoop table store_returns
(
    sr_returned_date_sk       bigint
,
    sr_return_time_sk         bigint
,
    sr_item_sk                bigint
not null,
    sr_customer_sk            bigint
,
    sr_cdemo_sk               bigint
,
    sr_hdemo_sk               bigint
,
    sr_addr_sk                bigint
,
    sr_store_sk               bigint
,
    sr_reason_sk              bigint
,
    sr_ticket_number          bigint
not null,
    sr_return_quantity        bigint
,
    sr_return_amt             double
,
    sr_return_tax             double
,
    sr_return_amt_inc_tax     double
,
    sr_fee                    double
,
    sr_return_ship_cost       double
,
    sr_refunded_cash          double
,
    sr_reversed_charge        double
,
    sr_store_credit           double
,
    sr_net_loss               double
)
STORED AS PARQUETFILE;

create hadoop table store_sales
(
    ss_sold_date_sk           bigint
,
    ss_sold_time_sk           bigint
,
    ss_item_sk                bigint
not null,
    ss_customer_sk            bigint
,
    ss_cdemo_sk               bigint
,
    ss_hdemo_sk               bigint
,
```

```
    ss_addr_sk                bigint
,
    ss_store_sk               bigint
,
    ss_promo_sk               bigint
,
    ss_ticket_number          bigint
not null,
    ss_quantity               bigint
,
    ss_wholesale_cost         double
,
    ss_list_price             double
,
    ss_sales_price            double
,
    ss_ext_discount_amt       double
,
    ss_ext_sales_price        double
,
    ss_ext_wholesale_cost     double
,
    ss_ext_list_price         double
,
    ss_ext_tax                double
,
    ss_coupon_amt             double
,
    ss_net_paid               double
,
    ss_net_paid_inc_tax       double
,
    ss_net_profit             double
)
STORED AS PARQUETFILE;

create hadoop table time_dim
(
    t_time_sk                 bigint
not null,
    t_time_id                 varchar(16)
not null,
    t_time                    bigint
,
    t_hour                    bigint
,
    t_minute                  bigint
,
    t_second                  bigint
,
    t_am_pm                   varchar(2)
,
    t_shift                   varchar(20)
,
    t_sub_shift               varchar(20)
,
    t_meal_time               varchar(20)
)
STORED AS PARQUETFILE;

create hadoop table warehouse
(
    w_warehouse_sk            bigint
not null,
    w_warehouse_id            varchar(16)
not null,
    w_warehouse_name          varchar(20)
,
    w_warehouse_sq_ft         bigint
,
    w_street_number           varchar(10)
,
    w_street_name             varchar(60)
,
    w_street_type             varchar(15)
,
    w_suite_number            varchar(10)
,
    w_city                    varchar(60)
,
    w_county                  varchar(30)
,
    w_state                   varchar(2)
,
    w_zip                     varchar(10)
,
    w_country                 varchar(20)
,
    w_gmt_offset              double
)
STORED AS PARQUETFILE;

create hadoop table web_page
(
    wp_web_page_sk            bigint
not null,
    wp_web_page_id            varchar(16)
not null,
    wp_rec_start_date         varchar(10)
,
    wp_rec_end_date           varchar(10)
,
    wp_creation_date_sk       bigint
,
    wp_access_date_sk         bigint
,
    wp_autogen_flag           varchar(1)
,
    wp_customer_sk            bigint
,
    wp_url                    varchar(100)
,
    wp_type                   varchar(50)
,
    wp_char_count             bigint
,
    wp_link_count             bigint
,
    wp_image_count            bigint
,
    wp_max_ad_count           bigint
)
STORED AS PARQUETFILE;

create hadoop table web_returns
(
    wr_returned_date_sk       bigint
,
    wr_returned_time_sk       bigint
,
    wr_item_sk                bigint
not null,
```

```
    wr_refunded_customer_sk   bigint
,
    wr_refunded_cdemo_sk      bigint
,
    wr_refunded_hdemo_sk      bigint
,
    wr_refunded_addr_sk       bigint
,
    wr_returning_customer_sk  bigint
,
    wr_returning_cdemo_sk     bigint
,
    wr_returning_hdemo_sk     bigint
,
    wr_returning_addr_sk      bigint
,
    wr_web_page_sk            bigint
,
    wr_reason_sk              bigint
,
    wr_order_number           bigint
not null,
    wr_return_quantity        bigint
,
    wr_return_amt             double
,
    wr_return_tax             double
,
    wr_return_amt_inc_tax     double
,
    wr_fee                    double
,
    wr_return_ship_cost       double
,
    wr_refunded_cash          double
,
    wr_reversed_charge        double
,
    wr_account_credit         double
,
    wr_net_loss               double
)
STORED AS PARQUETFILE;

create hadoop table web_sales
(
    ws_sold_date_sk           bigint
,
    ws_sold_time_sk           bigint
,
    ws_ship_date_sk           bigint
,
    ws_item_sk                bigint
not null,
    ws_bill_customer_sk       bigint
,
    ws_bill_cdemo_sk          bigint
,
    ws_bill_hdemo_sk          bigint
,
    ws_bill_addr_sk           bigint
,
    ws_ship_customer_sk       bigint
,
    ws_ship_cdemo_sk          bigint
,
    ws_ship_hdemo_sk          bigint
,
    ws_ship_addr_sk           bigint
,
    ws_web_page_sk            bigint
,
    ws_web_site_sk            bigint
,
    ws_ship_mode_sk           bigint
,
    ws_warehouse_sk           bigint
,
    ws_promo_sk               bigint
,
    ws_order_number           bigint
not null,
    ws_quantity               bigint
,
    ws_wholesale_cost         double
,
    ws_list_price             double
,
    ws_sales_price            double
,
    ws_ext_discount_amt       double
,
    ws_ext_sales_price        double
,
    ws_ext_wholesale_cost     double
,
    ws_ext_list_price         double
,
    ws_ext_tax                double
,
    ws_coupon_amt             double
,
    ws_ext_ship_cost          double
,
    ws_net_paid               double
,
    ws_net_paid_inc_tax       double
,
    ws_net_paid_inc_ship      double
,
    ws_net_paid_inc_ship_tax  double
,
    ws_net_profit             double
)
STORED AS PARQUETFILE;

create hadoop table web_site
(
    web_site_sk               bigint
not null,
    web_site_id               varchar(16)
not null,
    web_rec_start_date        varchar(10)
,
    web_rec_end_date          varchar(10)
,
    web_name                  varchar(50)
,
    web_open_date_sk          bigint
,
    web_close_date_sk         bigint
,
```

```
    web_class                 varchar(50)
,
    web_manager               varchar(40)
,
    web_mkt_id                bigint
,
    web_mkt_class             varchar(50)
,
    web_mkt_desc              varchar(100)
,
    web_market_manager        varchar(40)
,
    web_company_id            bigint
,
    web_company_name          varchar(50)
,
    web_street_number         varchar(10)
,
    web_street_name           varchar(60)
,
    web_street_type           varchar(15)
,
    web_suite_number          varchar(10)
,
    web_city                  varchar(60)
,
    web_county                varchar(30)
,
    web_state                 varchar(2)
,
    web_zip                   varchar(10)
,
    web_country               varchar(20)
,
    web_gmt_offset            double
,
    web_tax_percentage        double
)
STORED AS PARQUETFILE;

commit;
```

D.2 Impala

```
#!/bin/bash
impala-shell -d tpcds10000g <<EOF
create external table et_store_sales
(
  ss_sold_date_sk           int,
  ss_sold_time_sk           int,
  ss_item_sk                int,
  ss_customer_sk            int,
  ss_cdemo_sk               int,
  ss_hdemo_sk               smallint,
  ss_addr_sk                int,
  ss_store_sk               smallint,
  ss_promo_sk               smallint,
  ss_ticket_number          bigint,
  ss_quantity               bigint,
  ss_wholesale_cost
decimal(7,2),
  ss_list_price
decimal(7,2),
  ss_sales_price
decimal(7,2),
  ss_ext_discount_amt
decimal(7,2),
  ss_ext_sales_price
decimal(7,2),
  ss_ext_wholesale_cost
decimal(7,2),
  ss_ext_list_price
decimal(7,2),
  ss_ext_tax
decimal(7,2),
  ss_coupon_amt
decimal(7,2),
  ss_net_paid
decimal(7,2),
  ss_net_paid_inc_tax
decimal(7,2),
  ss_net_profit
decimal(7,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/store_sales'
tblproperties
('serialization.null.format'='')
;


create external table
et_customer_demographics
(
  cd_demo_sk                int,
  cd_gender                 string,
  cd_marital_status         string,
  cd_education_status       string,
  cd_purchase_estimate      bigint,
  cd_credit_rating          string,
  cd_dep_count              bigint,
  cd_dep_employed_count     bigint,
  cd_dep_college_count      bigint
)
row format delimited fields terminated
by '|'
location
'/tpcds10000g/customer_demographics'
tblproperties
('serialization.null.format'='')
;


create external table et_date_dim
```

```
(
  d_date_sk                 int,
  d_date_id                 string,
--  d_date                    string,
-- YYYY-MM-DD format
  d_date                    timestamp,
-- YYYY-MM-DD format
  d_month_seq               bigint,
  d_week_seq                bigint,
  d_quarter_seq             bigint,
  d_year                    bigint,
  d_dow                     bigint,
  d_moy                     bigint,
  d_dom                     bigint,
  d_qoy                     bigint,
  d_fy_year                 bigint,
  d_fy_quarter_seq          bigint,
  d_fy_week_seq             bigint,
  d_day_name                string,
  d_quarter_name            string,
  d_holiday                 string,
  d_weekend                 string,
  d_following_holiday       string,
  d_first_dom               bigint,
  d_last_dom                bigint,
  d_same_day_ly             bigint,
  d_same_day_lq             bigint,
  d_current_day             string,
  d_current_week            string,
  d_current_month           string,
  d_current_quarter         string,
  d_current_year            string
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/date_dim'
tblproperties
('serialization.null.format'='')
;


create external table et_time_dim
(
  t_time_sk                 int,
  t_time_id                 string,
  t_time                    bigint,
  t_hour                    bigint,
  t_minute                  bigint,
  t_second                  bigint,
  t_am_pm                   string,
  t_shift                   string,
  t_sub_shift               string,
  t_meal_time               string
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/time_dim'
tblproperties
('serialization.null.format'='')
;


create external table et_item
(
  i_item_sk                 int,
  i_item_id                 string,
  i_rec_start_date          timestamp,
  i_rec_end_date            timestamp,
  i_item_desc               string,
  i_current_price
decimal(7,2),
  i_wholesale_cost
decimal(7,2),
  i_brand_id                bigint,
  i_brand                   string,
  i_class_id                bigint,
  i_class                   string,
  i_category_id             bigint,
  i_category                string,
  i_manufact_id             bigint,
  i_manufact                string,
  i_size                    string,
  i_formulation             string,
  i_color                   string,
  i_units                   string,
  i_container               string,
  i_manager_id              bigint,
  i_product_name            string
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/item'
tblproperties
('serialization.null.format'='')
;


create external table et_store
(
  s_store_sk                smallint,
  s_store_id                string,
  s_rec_start_date          timestamp,
  s_rec_end_date            timestamp,
  s_closed_date_sk          int,
  s_store_name              string,
  s_number_employees        bigint,
  s_floor_space             bigint,
  s_hours                   string,
  s_manager                 string,
  s_market_id               bigint,
  s_geography_class         string,
  s_market_desc             string,
  s_market_manager          string,
  s_division_id             bigint,
  s_division_name           string,
  s_company_id              bigint,
  s_company_name            string,
  s_street_number           string,
```

```
  s_street_name             string,
  s_street_type             string,
  s_suite_number            string,
  s_city                    string,
  s_county                  string,
  s_state                   string,
  s_zip                     string,
  s_country                 string,
  s_gmt_offset
decimal(5,2),
  s_tax_precentage
decimal(5,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/store'
tblproperties
('serialization.null.format'='')
;


create external table et_customer
(
  c_customer_sk             int,
  c_customer_id             string,
  c_current_cdemo_sk        int,
  c_current_hdemo_sk        smallint,
  c_current_addr_sk         int,
  c_first_shipto_date_sk    int,
  c_first_sales_date_sk     int,
  c_salutation              string,
  c_first_name              string,
  c_last_name               string,
  c_preferred_cust_flag     string,
  c_birth_day               bigint,
  c_birth_month             bigint,
  c_birth_year              bigint,
  c_birth_country           string,
  c_login                   string,
  c_email_address           string,
  c_last_review_date        int
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/customer'
tblproperties
('serialization.null.format'='')
;


create external table et_promotion
(
  p_promo_sk                smallint,
  p_promo_id                string,
  p_start_date_sk           int,
  p_end_date_sk             int,
  p_item_sk                 int,
  p_cost                    double,
  p_response_target         bigint,
  p_promo_name              string,
  p_channel_dmail           string,
  p_channel_email           string,
  p_channel_catalog         string,
  p_channel_tv              string,
  p_channel_radio           string,
  p_channel_press           string,
  p_channel_event           string,
  p_channel_demo            string,
  p_channel_details         string,
  p_purpose                 string,
  p_discount_active         string
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/promotion'
tblproperties
('serialization.null.format'='')
;


create external table
et_household_demographics
(
  hd_demo_sk                smallint,
  hd_income_band_sk         tinyint,
  hd_buy_potential          string,
  hd_dep_count              bigint,
  hd_vehicle_count          bigint
)
row format delimited fields terminated
by '|'
location
'/tpcds10000g/household_demographics'
tblproperties
('serialization.null.format'='')
;


create external table
et_customer_address
(
  ca_address_sk             int,
  ca_address_id             string,
  ca_street_number          string,
  ca_street_name            string,
  ca_street_type            string,
  ca_suite_number           string,
  ca_city                   string,
  ca_county                 string,
  ca_state                  string,
  ca_zip                    string,
  ca_country                string,
  ca_gmt_offset
decimal(5,2),
  ca_location_type          string
)
row format delimited fields terminated
by '|'
```

```
location
'/tpcds10000g/customer_address'
tblproperties
('serialization.null.format'='')
;


create external table et_inventory
(
  inv_date_sk               int,
  inv_item_sk               int,
  inv_warehouse_sk          tinyint,
  inv_quantity_on_hand      bigint
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/inventory'
tblproperties
('serialization.null.format'='')
;


create external table et_call_center
(
  cc_call_center_sk         tinyint,
  cc_call_center_id         string,
  cc_rec_start_date         string,
  cc_rec_end_date           string,
  cc_closed_date_sk         int,
  cc_open_date_sk           int,
  cc_name                   string,
  cc_class                  string,
  cc_employees              bigint,
  cc_sq_ft                  bigint,
  cc_hours                  string,
  cc_manager                string,
  cc_mkt_id                 bigint,
  cc_mkt_class              string,
  cc_mkt_desc               string,
  cc_market_manager         string,
  cc_division               bigint,
  cc_division_name          string,
  cc_company                bigint,
  cc_company_name           string,
  cc_street_number          string,
  cc_street_name            string,
  cc_street_type            string,
  ca_suite_number           string,
  cc_city                   string,
  cc_county                 string,
  cc_state                  string,
  cc_zip                    string,
  cc_country                string,
  cc_gmt_offset
decimal(5,2),
  cc_tax_precentage
decimal(5,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/call_center'
tblproperties
('serialization.null.format'='')
;


create external table et_catalog_page
(
  cp_catalog_page_sk        int,
  cp_catalog_page_id        string,
  cp_start_date_sk          int,
  cp_end_date_sk            int,
  cp_department             string,
  cp_catalog_number         bigint,
  cp_catalog_page_number    bigint,
  cp_description            string,
  cp_type                   string
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/catalog_page'
tblproperties
('serialization.null.format'='')
;


create external table
et_catalog_returns
(
  cr_returned_date_sk       int,
  cr_returned_time_sk       int,
  cr_item_sk                int,
  cr_refunded_customer_sk   int,
  cr_refunded_cdemo_sk      int,
  cr_refunded_hdemo_sk      smallint,
  cr_refunded_addr_sk       int,
  cr_returning_customer_sk  int,
  cr_returning_cdemo_sk     int,
  cr_returning_hdemo_sk     smallint,
  cr_returning_addr_sk      int,
  cr_call_center_sk         tinyint,
  cr_catalog_page_sk        int,
  cr_ship_mode_sk           tinyint,
  cr_warehouse_sk           tinyint,
  cr_reason_sk              tinyint,
  cr_order_number           bigint,
  cr_return_quantity        bigint,
  cr_return_amount
decimal(7,2),
  cr_return_tax
decimal(7,2),
  cr_return_amt_inc_tax
decimal(7,2),
  cr_return_fee
decimal(7,2),
  cr_return_ship_cost
decimal(7,2),
```

```
  cr_refunded_cash
decimal(7,2),
  cr_reversed_charge
decimal(7,2),
  cr_store_credit
decimal(7,2),
  cr_net_loss
decimal(7,2)
)
row format delimited fields terminated
by '|'
location
'/tpcds10000g/catalog_returns'
tblproperties
('serialization.null.format'='')
;


create external table et_catalog_sales
(
  cs_sold_date_sk           int,
  cs_sold_time_sk           int,
  cs_ship_date_sk           int,
  cs_bill_customer_sk       int,
  cs_bill_cdemo_sk          int,
  cs_bill_hdemo_sk          smallint,
  cs_bill_addr_sk           int,
  cs_ship_customer_sk       int,
  cs_ship_cdemo_sk          int,
  cs_ship_hdemo_sk          smallint,
  cs_ship_addr_sk           int,
  cs_call_center_sk         tinyint,
  cs_catalog_page_sk        int,
  cs_ship_mode_sk           tinyint,
  cs_warehouse_sk           tinyint,
  cs_item_sk                int,
  cs_promo_sk               smallint,
  cs_order_number           bigint,
  cs_quantity               bigint,
  cs_wholesale_cost
decimal(7,2),
  cs_list_price
decimal(7,2),
  cs_sales_price
decimal(7,2),
  cs_ext_discount_amt
decimal(7,2),
  cs_ext_sales_price
decimal(7,2),
  cs_ext_wholesale_cost
decimal(7,2),
  cs_ext_list_price
decimal(7,2),
  cs_ext_tax
decimal(7,2),
  cs_coupon_amt
decimal(7,2),
  cs_ext_ship_cost
decimal(7,2),
  cs_net_paid
decimal(7,2),
  cs_net_paid_inc_tax
decimal(7,2),
  cs_net_paid_inc_ship
decimal(7,2),
  cs_net_paid_inc_ship_tax
decimal(7,2),
  cs_net_profit
decimal(7,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/catalog_sales'
tblproperties
('serialization.null.format'='')
;


create external table et_income_band
(
  ib_income_band_sk         tinyint,
  ib_lower_bound            bigint,
  ib_upper_bound            bigint
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/income_band'
tblproperties
('serialization.null.format'='')
;


create external table et_ship_mode
(
  sm_ship_mode_sk           tinyint,
  sm_ship_mode_id           string,
  sm_type                   string,
  sm_code                   string,
  sm_carrier                string,
  sm_contract               string
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/ship_mode'
tblproperties
('serialization.null.format'='')
;


create external table et_web_page
(
  wp_web_page_sk            smallint,
  wp_web_page_id            string,
  wp_rec_start_date         timestamp,
  wp_rec_end_date           timestamp,
  wp_creation_date_sk       int,
  wp_access_date_sk         int,
  wp_autogen_flag           string,
```

```
  wp_customer_sk            int,
  wp_url                    string,
  wp_type                   string,
  wp_char_count             bigint,
  wp_link_count             bigint,
  wp_image_count            bigint,
  wp_max_ad_count           bigint
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/web_page'
tblproperties
('serialization.null.format'='')
;


create external table et_store_returns
(
  sr_returned_date_sk       int,
  sr_return_time_sk         int,
  sr_item_sk                int,
  sr_customer_sk            int,
  sr_cdemo_sk               int,
  sr_hdemo_sk               smallint,
  sr_addr_sk                int,
  sr_store_sk               smallint,
  sr_reason_sk              tinyint,
  sr_ticket_number          bigint,
  sr_return_quantity        bigint,
  sr_return_amt
decimal(7,2),
  sr_return_tax
decimal(7,2),
  sr_return_amt_inc_tax
decimal(7,2),
  sr_return_fee
decimal(7,2),
  sr_return_ship_cost
decimal(7,2),
  sr_refunded_cash
decimal(7,2),
  sr_reversed_charge
decimal(7,2),
  sr_store_credit
decimal(7,2),
  sr_net_loss
decimal(7,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/store_returns'
tblproperties
('serialization.null.format'='')
;


create external table et_web_returns
(
  wr_returned_date_sk       int,
  wr_returned_time_sk       int,
  wr_item_sk                int,
  wr_refunded_customer_sk   int,
  wr_refunded_cdemo_sk      int,
  wr_refunded_hdemo_sk      smallint,
  wr_refunded_addr_sk       int,
  wr_returning_customer_sk  int,
  wr_returning_cdemo_sk     int,
  wr_returning_hdemo_sk     smallint,
  wr_returning_addr_sk      int,
  wr_web_page_sk            smallint,
  wr_reason_sk              tinyint,
  wr_order_number           bigint,
  wr_return_quantity        bigint,
  wr_return_amt
decimal(7,2),
  wr_return_tax
decimal(7,2),
  wr_return_amt_inc_tax
decimal(7,2),
  wr_fee
decimal(7,2),
  wr_return_ship_cost
decimal(7,2),
  wr_refunded_cash
decimal(7,2),
  wr_reversed_charge
decimal(7,2),
  wr_account_credit
decimal(7,2),
  wr_net_loss
decimal(7,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/web_returns'
tblproperties
('serialization.null.format'='')
;


create external table et_web_sales
(
  ws_sold_date_sk           int,
  ws_sold_time_sk           int,
  ws_ship_date_sk           int,
  ws_item_sk                int,
  ws_bill_customer_sk       int,
  ws_bill_cdemo_sk          int,
  ws_bill_hdemo_sk          smallint,
  ws_bill_addr_sk           int,
  ws_ship_customer_sk       int,
  ws_ship_cdemo_sk          int,
  ws_ship_hdemo_sk          smallint,
  ws_ship_addr_sk           int,
  ws_web_page_sk            smallint,
  ws_web_site_sk            tinyint,
  ws_ship_mode_sk           tinyint,
  ws_warehouse_sk           tinyint,
```

```
  ws_promo_sk               smallint,
  ws_order_number           bigint,
  ws_quantity               bigint,
  ws_wholesale_cost
decimal(7,2),
  ws_list_price
decimal(7,2),
  ws_sales_price
decimal(7,2),
  ws_ext_discount_amt
decimal(7,2),
  ws_ext_sales_price
decimal(7,2),
  ws_ext_wholesale_cost
decimal(7,2),
  ws_ext_list_price
decimal(7,2),
  ws_ext_tax
decimal(7,2),
  ws_coupon_amt
decimal(7,2),
  ws_ext_ship_cost
decimal(7,2),
  ws_net_paid
decimal(7,2),
  ws_net_paid_inc_tax
decimal(7,2),
  ws_net_paid_inc_ship
decimal(7,2),
  ws_net_paid_inc_ship_tax
decimal(7,2),
ring,
  web_street_number         string,
  web_street_name           string,
  web_street_type           string,
  web_suite_number          string,
  web_city                  string,
  web_county                string,
  web_state                 string,
  web_zip                   string,
  web_country               string,
  web_gmt_offset
decimal(5,2),
  web_tax_precentage
decimal(5,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/web_site'
tblproperties
('serialization.null.format'='')
;


create external table et_reason
(
  r_reason_sk               tinyint,
  r_reason_id               string,
  r_reason_desc             string
```

```
  ws_net_profit
decimal(7,2)
)
row format delimited fields terminated
by '|'
location '/tpcds10000g/web_sales'
tblproperties
('serialization.null.format'='')
;


create external table et_web_site
(
  web_site_sk               tinyint,
  web_site_id               string,
  web_rec_start_date        timestamp,
  web_rec_end_date          timestamp,
  web_name                  string,
  web_open_date_sk          int,
  web_close_date_sk         int,
  web_class                 string,
  web_manager               string,
  web_mkt_id                bigint,
  web_mkt_class             string,
  web_mkt_desc              string,
  web_market_manager        string,
  web_company_id            bigint,
  web_company_name          st



)
row format delimited fields terminated
by '|'
location '/tpcds10000g/reason'
tblproperties
('serialization.null.format'='')
;


create external table et_warehouse
(
  w_warehouse_sk            tinyint,
  w_warehouse_id            string,
  w_warehouse_name          string,
  w_warehouse_sq_ft         bigint,
  w_street_number           string,
  w_street_name             string,
  w_street_type             string,
  w_suite_number            string,
  w_city                    string,
  w_county                  string,
  w_state                   string,
  w_zip                     string,
  w_country                 string,
  w_gmt_offset
decimal(5,2)
)
```

```
row format delimited fields terminated
by '|'
location '/tpcds10000g/warehouse'
tblproperties
('serialization.null.format'='')
;
```

```
show tables;
EOF
```

D.3 Hive 0.13

```
-- Use the following to execute this
script and create the tables in Hive:
--   $HIVE_HOME/bin/hive -hiveconf
DB_NAME=300 -f
$testhome/ddl/065.hive.create-
tables.ddl

CREATE DATABASE IF NOT EXISTS
TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B_N
EW
  COMMENT 'For TPCDS tables at
${hiveconf:DB_NAME} scale factor' ;

USE
TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B_N
EW;

create external table customer_address
(
    ca_address_sk             int,
    ca_address_id             string,
    ca_street_number          string,
    ca_street_name            string,
    ca_street_type            string,
    ca_suite_number           string,
    ca_city                   string,
    ca_county                 string,
    ca_state                  string,
    ca_zip                    string,
    ca_country                string,
    ca_gmt_offset             double,
    ca_location_type          string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table
customer_demographics
(
    cd_demo_sk                int,
    cd_gender                 string,
    cd_marital_status         string,
    cd_education_status       string,
    cd_purchase_estimate      int,
    cd_credit_rating          string,
    cd_dep_count              int,
    cd_dep_employed_count     int,
    cd_dep_college_count      int
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table date_dim
(
    d_date_sk                 int,
    d_date_id                 string,
    d_date                    string,
    d_month_seq               int,
    d_week_seq                int,
    d_quarter_seq             int,
    d_year                    int,
    d_dow                     int,
    d_moy                     int,
    d_dom                     int,
    d_qoy                     int,
    d_fy_year                 int,
    d_fy_quarter_seq          int,
    d_fy_week_seq             int,
    d_day_name                string,
    d_quarter_name            string,
    d_holiday                 string,
    d_weekend                 string,
    d_following_holiday       string,
    d_first_dom               int,
    d_last_dom                int,
    d_same_day_ly             int,
    d_same_day_lq             int,
    d_current_day             string,
    d_current_week            string,
    d_current_month           string,
    d_current_quarter         string,
    d_current_year            string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;
```

```
create external table warehouse
(
    w_warehouse_sk            int,
    w_warehouse_id            string,
    w_warehouse_name          string,
    w_warehouse_sq_ft         int,
    w_street_number           string,
    w_street_name             string,
    w_street_type             string,
    w_suite_number            string,
    w_city                    string,
    w_county                  string,
    w_state                   string,
    w_zip                     string,
    w_country                 string,
    w_gmt_offset              double
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table ship_mode
(
    sm_ship_mode_sk           int,
    sm_ship_mode_id           string,
    sm_type                   string,
    sm_code                   string,
    sm_carrier                string,
    sm_contract               string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table time_dim
(
    t_time_sk                 int,
    t_time_id                 string,
    t_time                    int,
    t_hour                    int,
    t_minute                  int,
    t_second                  int,
    t_am_pm                   string,
    t_shift                   string,
    t_sub_shift               string,
    t_meal_time               string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;
```

```
create external table reason
(
    r_reason_sk               int,
    r_reason_id               string,
    r_reason_desc             string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table income_band
(
    ib_income_band_sk         int,
    ib_lower_bound            int,
    ib_upper_bound            int
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table item
(
    i_item_sk                 int,
    i_item_id                 string,
    i_rec_start_date          string,
    i_rec_end_date            string,
    i_item_desc               string,
    i_current_price           double,
    i_wholesale_cost          double,
    i_brand_id                int,
    i_brand                   string,
    i_class_id                int,
    i_class                   string,
    i_category_id             int,
    i_category                string,
    i_manufact_id             int,
    i_manufact                string,
    i_size                    string,
    i_formulation             string,
    i_color                   string,
    i_units                   string,
    i_container               string,
    i_manager_id              int,
    i_product_name            string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;
```

```
create external table store
(
    s_store_sk                int,
    s_store_id                string,
    s_rec_start_date          string,
    s_rec_end_date            string,
    s_closed_date_sk          int,
    s_store_name              string,
    s_number_employees        int,
    s_floor_space             int,
    s_hours                   string,
    s_manager                 string,
    s_market_id               int,
    s_geography_class         string,
    s_market_desc             string,
    s_market_manager          string,
    s_division_id             int,
    s_division_name           string,
    s_company_id              int,
    s_company_name            string,
    s_street_number           string,
    s_street_name             string,
    s_street_type             string,
    s_suite_number            string,
    s_city                    string,
    s_county                  string,
    s_state                   string,
    s_zip                     string,
    s_country                 string,
    s_gmt_offset              double,
    s_tax_precentage          double
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table call_center
(
    cc_call_center_sk         int,
    cc_call_center_id         string,
    cc_rec_start_date         string,
    cc_rec_end_date           string,
    cc_closed_date_sk         int,
    cc_open_date_sk           int,
    cc_name                   string,
    cc_class                  string,
    cc_employees              int,
    cc_sq_ft                  int,
    cc_hours                  string,
    cc_manager                string,
    cc_mkt_id                 int,
    cc_mkt_class              string,
    cc_mkt_desc               string,
    cc_market_manager         string,
    cc_division               int,
    cc_division_name          string,
    cc_company                int,
    cc_company_name           string,
    cc_street_number          string,
    cc_street_name            string,
    cc_street_type            string,
    cc_suite_number           string,
    cc_city                   string,
    cc_county                 string,
    cc_state                  string,
    cc_zip                    string,
    cc_country                string,
    cc_gmt_offset             double,
    cc_tax_percentage         double
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table customer
(
    c_customer_sk             int,
    c_customer_id             string,
    c_current_cdemo_sk        int,
    c_current_hdemo_sk        int,
    c_current_addr_sk         int,
    c_first_shipto_date_sk    int,
    c_first_sales_date_sk     int,
    c_salutation              string,
    c_first_name              string,
    c_last_name               string,
    c_preferred_cust_flag     string,
    c_birth_day               int,
    c_birth_month             int,
    c_birth_year              int,
    c_birth_country           string,
    c_login                   string,
    c_email_address           string,
    c_last_review_date        string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table web_site
(
    web_site_sk               int,
    web_site_id               string,
    web_rec_start_date        string,
    web_rec_end_date          string,
    web_name                  string,
```

```
    web_open_date_sk          int,
    web_close_date_sk         int,
    web_class                 string,
    web_manager               string,
    web_mkt_id                int,
    web_mkt_class             string,
    web_mkt_desc              string,
    web_market_manager        string,
    web_company_id            int,
    web_company_name          string,
    web_street_number         string,
    web_street_name           string,
    web_street_type           string,
    web_suite_number          string,
    web_city                  string,
    web_county                string,
    web_state                 string,
    web_zip                   string,
    web_country               string,
    web_gmt_offset            double,
    web_tax_percentage        double
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table store_returns
(
    sr_returned_date_sk       int,
    sr_return_time_sk         int,
    sr_item_sk                int,
    sr_customer_sk            int,
    sr_cdemo_sk               int,
    sr_hdemo_sk               int,
    sr_addr_sk                int,
    sr_store_sk               int,
    sr_reason_sk              int,
    sr_ticket_number          int,
    sr_return_quantity        int,
    sr_return_amt             double,
    sr_return_tax             double,
    sr_return_amt_inc_tax     double,
    sr_fee                    double,
    sr_return_ship_cost       double,
    sr_refunded_cash          double,
    sr_reversed_charge        double,
    sr_store_credit           double,
    sr_net_loss               double
)
--row format delimited
--fields terminated by '|'
clustered by (sr_item_sk)
sorted by (sr_ticket_number,
sr_item_sk) into 271 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table
household_demographics
(
    hd_demo_sk                int,
    hd_income_band_sk         int,
    hd_buy_potential          string,
    hd_dep_count              int,
    hd_vehicle_count          int
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table web_page
(
    wp_web_page_sk            int,
    wp_web_page_id            string,
    wp_rec_start_date         string,
    wp_rec_end_date           string,
    wp_creation_date_sk       int,
    wp_access_date_sk         int,
    wp_autogen_flag           string,
    wp_customer_sk            int,
    wp_url                    string,
    wp_type                   string,
    wp_char_count             int,
    wp_link_count             int,
    wp_image_count            int,
    wp_max_ad_count           int
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table promotion
(
    p_promo_sk                int,
    p_promo_id                string,
    p_start_date_sk           int,
    p_end_date_sk             int,
    p_item_sk                 int,
    p_cost                    double,
    p_response_target         int,
    p_promo_name              string,
    p_channel_dmail           string,
    p_channel_email           string,
    p_channel_catalog         string,
    p_channel_tv              string,
    p_channel_radio           string,
```

```
    p_channel_press           string,
    p_channel_event           string,
    p_channel_demo            string,
    p_channel_details         string,
    p_purpose                 string,
    p_discount_active         string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table catalog_page
(
    cp_catalog_page_sk        int,
    cp_catalog_page_id        string,
    cp_start_date_sk          int,
    cp_end_date_sk            int,
    cp_department             string,
    cp_catalog_number         int,
    cp_catalog_page_number    int,
    cp_description            string,
    cp_type                   string
)
row format delimited
fields terminated by '|'
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table inventory
(
    inv_date_sk               int,
    inv_item_sk               int,
    inv_warehouse_sk          int,
    inv_quantity_on_hand      bigint
)
--row format delimited
--fields terminated by '|'
clustered by (inv_item_sk)
sorted by (inv_date_sk, inv_item_sk,
inv_warehouse_sk) into 89 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table catalog_returns
(
    cr_returned_date_sk       int,
    cr_returned_time_sk       int,
    cr_item_sk                int,
    cr_refunded_customer_sk   int,
    cr_refunded_cdemo_sk      int,
    cr_refunded_hdemo_sk      int,
    cr_refunded_addr_sk       int,
    cr_returning_customer_sk  int,
    cr_returning_cdemo_sk     int,
    cr_returning_hdemo_sk     int,
    cr_returning_addr_sk      int,
    cr_call_center_sk         int,
    cr_catalog_page_sk        int,
    cr_ship_mode_sk           int,
    cr_warehouse_sk           int,
    cr_reason_sk              int,
    cr_order_number           int,
    cr_return_quantity        int,
    cr_return_amount          double,
    cr_return_tax             double,
    cr_return_amt_inc_tax     double,
    cr_fee                    double,
    cr_return_ship_cost       double,
    cr_refunded_cash          double,
    cr_reversed_charge        double,
    cr_store_credit           double,
    cr_net_loss               double
)
--row format delimited
--fields terminated by '|'
clustered by (cr_item_sk)
sorted by (cr_order_number,
cr_item_sk) into 271 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table web_returns
(
    wr_returned_date_sk       int,
    wr_returned_time_sk       int,
    wr_item_sk                int,
    wr_refunded_customer_sk   int,
    wr_refunded_cdemo_sk      int,
    wr_refunded_hdemo_sk      int,
    wr_refunded_addr_sk       int,
    wr_returning_customer_sk  int,
    wr_returning_cdemo_sk     int,
    wr_returning_hdemo_sk     int,
    wr_returning_addr_sk      int,
    wr_web_page_sk            int,
    wr_reason_sk              int,
    wr_order_number           int,
    wr_return_quantity        int,
    wr_return_amt             double,
    wr_return_tax             double,
    wr_return_amt_inc_tax     double,
    wr_fee                    double,
    wr_return_ship_cost       double,
    wr_refunded_cash          double,
    wr_reversed_charge        double,
    wr_account_credit         double,
    wr_net_loss               double
)
--row format delimited
```

```
--fields terminated by '|'
clustered by (wr_item_sk)
sorted by (wr_order_number,
wr_item_sk) into 271 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table web_sales
(
    ws_sold_date_sk           int,
    ws_sold_time_sk           int,
    ws_ship_date_sk           int,
    ws_item_sk                int,
    ws_bill_customer_sk       int,
    ws_bill_cdemo_sk          int,
    ws_bill_hdemo_sk          int,
    ws_bill_addr_sk           int,
    ws_ship_customer_sk       int,
    ws_ship_cdemo_sk          int,
    ws_ship_hdemo_sk          int,
    ws_ship_addr_sk           int,
    ws_web_page_sk            int,
    ws_web_site_sk            int,
    ws_ship_mode_sk           int,
    ws_warehouse_sk           int,
    ws_promo_sk               int,
    ws_order_number           int,
    ws_quantity               int,
    ws_wholesale_cost         double,
    ws_list_price             double,
    ws_sales_price            double,
    ws_ext_discount_amt       double,
    ws_ext_sales_price        double,
    ws_ext_wholesale_cost     double,
    ws_ext_list_price         double,
    ws_ext_tax                double,
    ws_coupon_amt             double,
    ws_ext_ship_cost          double,
    ws_net_paid               double,
    ws_net_paid_inc_tax       double,
    ws_net_paid_inc_ship      double,
    ws_net_paid_inc_ship_tax  double,
    ws_net_profit             double
)
--row format delimited
--fields terminated by '|'
clustered by (ws_item_sk)
sorted by (ws_order_number,
ws_item_sk) into 271 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table catalog_sales
(
    cs_sold_date_sk           int,
    cs_sold_time_sk           int,
    cs_ship_date_sk           int,
    cs_bill_customer_sk       int,
    cs_bill_cdemo_sk          int,
    cs_bill_hdemo_sk          int,
    cs_bill_addr_sk           int,
    cs_ship_customer_sk       int,
    cs_ship_cdemo_sk          int,
    cs_ship_hdemo_sk          int,
    cs_ship_addr_sk           int,
    cs_call_center_sk         int,
    cs_catalog_page_sk        int,
    cs_ship_mode_sk           int,
    cs_warehouse_sk           int,
    cs_item_sk                int,
    cs_promo_sk               int,
    cs_order_number           int,
    cs_quantity               int,
    cs_wholesale_cost         double,
    cs_list_price             double,
    cs_sales_price            double,
    cs_ext_discount_amt       double,
    cs_ext_sales_price        double,
    cs_ext_wholesale_cost     double,
    cs_ext_list_price         double,
    cs_ext_tax                double,
    cs_coupon_amt             double,
    cs_ext_ship_cost          double,
    cs_net_paid               double,
    cs_net_paid_inc_tax       double,
    cs_net_paid_inc_ship      double,
    cs_net_paid_inc_ship_tax  double,
    cs_net_profit             double
)
--row format delimited
--fields terminated by '|'
clustered by (cs_item_sk)
sorted by (cs_order_number,
cs_item_sk) into 271 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;

create external table store_sales
(
    ss_sold_date_sk           int,
    ss_sold_time_sk           int,
    ss_item_sk                int,
    ss_customer_sk            int,
    ss_cdemo_sk               int,
    ss_hdemo_sk               int,
    ss_addr_sk                int,
    ss_store_sk               int,
    ss_promo_sk               int,
    ss_ticket_number          int,
    ss_quantity               int,
    ss_wholesale_cost         double,
    ss_list_price             double,
```

```
    ss_sales_price            double,
    ss_ext_discount_amt       double,
    ss_ext_sales_price        double,
    ss_ext_wholesale_cost     double,
    ss_ext_list_price         double,
    ss_ext_tax                double,
    ss_coupon_amt             double,
    ss_net_paid               double,
    ss_net_paid_inc_tax       double,
    ss_net_profit             double
```

```
)
--row format delimited
--fields terminated by '|'
clustered by (ss_item_sk)
sorted by (ss_ticket_number,
ss_item_sk) into 271 buckets
stored as ORC
tblproperties
('serialization.null.format'='')
;
```

Appendix E: Query Text

Queries for all vendors are generated from query templates. Specific parameter values depend on both the context the query is run (scale factor, single or multi-stream), and the seed for the random number generator. A common seed (20140815) for the random number generator was used across the 3 distributions, thus making all queries across all distributions the same.

The queries were executed in a different order for each stream based on the standard TPC-DS specification. However, the query order for each stream was identical across all vendors.

Following are the query text for the 46 common queries, as used during the single-stream test:

E.1 Big SQL Queries:

```
-- start query 1 in stream 0 using template
query96.tpl and seed 550831069
select  count(*)
from store_sales
    ,household_demographics
    ,time_dim, store
where ss_sold_time_sk = time_dim.t_time_sk
    and ss_hdemo_sk =
household_demographics.hd_demo_sk
    and ss_store_sk = s_store_sk
    and time_dim.t_hour = 15
    and time_dim.t_minute >= 30
    and household_demographics.hd_dep_count
= 6
    and store.s_store_name = 'ese'
order by count(*)
 fetch first 100 rows only;

-- end query 1 in stream 0 using template
query96.tpl
-- start query 2 in stream 0 using template
query7.tpl and seed 997258328
select  i_item_id,
        avg(cast(ss_quantity as double))
agg1,
        avg(ss_list_price) agg2,
        avg(ss_coupon_amt) agg3,
        avg(ss_sales_price) agg4
 from store_sales, customer_demographics,
date_dim, item, promotion
 where ss_sold_date_sk = d_date_sk and
       ss_item_sk = i_item_sk and
       ss_cdemo_sk = cd_demo_sk and
       ss_promo_sk = p_promo_sk and
       cd_gender = 'M' and
       cd_marital_status = 'W' and
       cd_education_status = '2 yr Degree'
and
       (p_channel_email = 'N' or
p_channel_event = 'N') and
       d_year = 1999
 group by i_item_id
 order by i_item_id
  fetch first 100 rows only;

-- end query 2 in stream 0 using template
query7.tpl

-- start query 5 in stream 0 using template
query39.tpl and seed 1420791654
with inv as
(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy
       ,stdev,mean, case mean when 0 then
null else stdev/mean end cov
 from(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy

,stddev_samp(inv_quantity_on_hand)
stdev,avg(cast(inv_quantity_on_hand as
double)) mean
      from inventory
          ,item
          ,warehouse
          ,date_dim
      where inv_item_sk = i_item_sk
        and inv_warehouse_sk =
w_warehouse_sk
```

```
        and inv_date_sk = d_date_sk
        and d_year =2000
      group by
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy) foo
 where case mean when 0 then 0 else
stdev/mean end > 1)
select
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean, inv1.cov

,inv2.w_warehouse_sk,inv2.i_item_sk,inv2.d_
moy,inv2.mean, inv2.cov
from inv inv1,inv inv2
where inv1.i_item_sk = inv2.i_item_sk
  and inv1.w_warehouse_sk =
inv2.w_warehouse_sk
  and inv1.d_moy=2
  and inv2.d_moy=2+1
order by
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean,inv1.cov
        ,inv2.d_moy,inv2.mean, inv2.cov
;
with inv as
(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy
       ,stdev,mean, case mean when 0 then
null else stdev/mean end cov
 from(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy

,stddev_samp(inv_quantity_on_hand)
stdev,avg(cast(inv_quantity_on_hand as
double)) mean
      from inventory
          ,item
          ,warehouse
          ,date_dim
      where inv_item_sk = i_item_sk
        and inv_warehouse_sk =
w_warehouse_sk
        and inv_date_sk = d_date_sk
        and d_year =2000
      group by
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy) foo
 where case mean when 0 then 0 else
stdev/mean end > 1)
select
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean, inv1.cov

,inv2.w_warehouse_sk,inv2.i_item_sk,inv2.d_
moy,inv2.mean, inv2.cov
from inv inv1,inv inv2
where inv1.i_item_sk = inv2.i_item_sk
  and inv1.w_warehouse_sk =
inv2.w_warehouse_sk
  and inv1.d_moy=2
  and inv2.d_moy=2+1
  and inv1.cov > 1.5
order by
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean,inv1.cov
        ,inv2.d_moy,inv2.mean, inv2.cov
;

-- end query 5 in stream 0 using template
query39.tpl

-- start query 7 in stream 0 using template
query32.tpl and seed 944563352
select  sum(cs_ext_discount_amt)  as
"excess discount amount"
from
   catalog_sales
   ,item
   ,date_dim
where
i_manufact_id = 353
and i_item_sk = cs_item_sk
and d_date between '2000-01-16' and
        (cast('2000-01-16' as date) + 90
days)
and d_date_sk = cs_sold_date_sk
and cs_ext_discount_amt
     > (
         select
            1.3 * avg(cs_ext_discount_amt)
         from
            catalog_sales
           ,date_dim
         where
              cs_item_sk = i_item_sk
          and d_date between '2000-01-16'
and
                                (cast('2000-
01-16' as date) + 90 days)
          and d_date_sk = cs_sold_date_sk
      )
 fetch first 100 rows only;

-- end query 7 in stream 0 using template
query32.tpl

-- start query 14 in stream 0 using
template query21.tpl and seed 614834996
select  *
 from(select w_warehouse_name
           ,i_item_id
           ,sum(case when (cast(d_date as
date) < cast ('1998-06-27' as date))
                     then
inv_quantity_on_hand
                     else 0 end) as
inv_before
           ,sum(case when (cast(d_date as
date) >= cast ('1998-06-27' as date))
                     then
inv_quantity_on_hand
                     else 0 end) as
inv_after
   from inventory
       ,warehouse
       ,item
       ,date_dim
   where i_current_price between 0.99 and
1.49
     and i_item_sk          = inv_item_sk
     and inv_warehouse_sk   =
w_warehouse_sk
```

```
     and inv_date_sk    = d_date_sk
     and d_date between (cast ('1998-06-27'
as date) - 30 days)
                    and (cast ('1998-06-27'
as date) + 30 days)
   group by w_warehouse_name, i_item_id) x
 where (case when inv_before > 0
             then cast(inv_after as double)
/ cast(inv_before as double)
             else null
             end) between 2.0/3.0 and
3.0/2.0
 order by w_warehouse_name
         ,i_item_id
  fetch first 100 rows only;

-- end query 14 in stream 0 using template
query21.tpl
-- start query 15 in stream 0 using
template query43.tpl and seed 959608359
select  s_store_name, s_store_id,
        sum(case when (d_day_name='Sunday')
then ss_sales_price else null end)
sun_sales,
        sum(case when (d_day_name='Monday')
then ss_sales_price else null end)
mon_sales,
        sum(case when
(d_day_name='Tuesday') then ss_sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then
ss_sales_price else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then ss_sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then ss_sales_price else null end)
fri_sales,
        sum(case when
(d_day_name='Saturday') then ss_sales_price
else null end) sat_sales
 from date_dim, store_sales, store
 where d_date_sk = ss_sold_date_sk and
       s_store_sk = ss_store_sk and
       s_gmt_offset = -8 and
       d_year = 1998
 group by s_store_name, s_store_id
 order by s_store_name,
s_store_id,sun_sales,mon_sales,tue_sales,we
d_sales,thu_sales,fri_sales,sat_sales
  fetch first 100 rows only;

-- end query 15 in stream 0 using template
query43.tpl
-- start query 16 in stream 0 using
template query27.tpl and seed 331218716
select  i_item_id,
        s_state, grouping(s_state) g_state,
        avg(cast(ss_quantity as double))
agg1,
        avg(ss_list_price) agg2,
        avg(ss_coupon_amt) agg3,
        avg(ss_sales_price) agg4
 from store_sales, customer_demographics,
date_dim, store, item
 where ss_sold_date_sk = d_date_sk and
       ss_item_sk = i_item_sk and
       ss_store_sk = s_store_sk and
       ss_cdemo_sk = cd_demo_sk and
       cd_gender = 'F' and
       cd_marital_status = 'W' and
       cd_education_status = '4 yr Degree'
and
       d_year = 1999 and
       s_state in ('OH','IL', 'LA', 'GA',
'CO', 'AL')
 group by rollup (i_item_id, s_state)
 order by i_item_id
         ,s_state
  fetch first 100 rows only;

-- end query 16 in stream 0 using template
query27.tpl

-- start query 19 in stream 0 using
template query58.tpl and seed 1844319395
with ss_items as
 (select i_item_id item_id
        ,sum(ss_ext_sales_price)
ss_item_rev
 from store_sales
     ,item
     ,date_dim
 where ss_item_sk = i_item_sk
   and d_date in (select d_date
                  from date_dim
                  where d_week_seq =
(select d_week_seq
                                        from
date_dim
                                        where
d_date = '1998-05-29'))
   and ss_sold_date_sk   = d_date_sk
 group by i_item_id),
 cs_items as
 (select i_item_id item_id
        ,sum(cs_ext_sales_price)
cs_item_rev
  from catalog_sales
      ,item
      ,date_dim
 where cs_item_sk = i_item_sk
  and  d_date in (select d_date
                  from date_dim
                  where d_week_seq =
(select d_week_seq
                                        from
date_dim
                                        where
d_date = '1998-05-29'))
  and  cs_sold_date_sk = d_date_sk
 group by i_item_id),
 ws_items as
 (select i_item_id item_id
        ,sum(ws_ext_sales_price)
ws_item_rev
  from web_sales
      ,item
      ,date_dim
 where ws_item_sk = i_item_sk
  and  d_date in (select d_date
                  from date_dim
```

```
                  where d_week_seq =(select
d_week_seq
                                     from
date_dim
                                     where
d_date = '1998-05-29'))
  and ws_sold_date_sk   = d_date_sk
 group by i_item_id)
  select  ss_items.item_id
       ,ss_item_rev

,ss_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 ss_dev
       ,cs_item_rev

,cs_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 cs_dev
       ,ws_item_rev

,ws_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 ws_dev

,(ss_item_rev+cs_item_rev+ws_item_rev)/3
average
 from ss_items,cs_items,ws_items
 where ss_items.item_id=cs_items.item_id
   and ss_items.item_id=ws_items.item_id
   and ss_item_rev between 0.9 *
cs_item_rev and 1.1 * cs_item_rev
   and ss_item_rev between 0.9 *
ws_item_rev and 1.1 * ws_item_rev
   and cs_item_rev between 0.9 *
ss_item_rev and 1.1 * ss_item_rev
   and cs_item_rev between 0.9 *
ws_item_rev and 1.1 * ws_item_rev
   and ws_item_rev between 0.9 *
ss_item_rev and 1.1 * ss_item_rev
   and ws_item_rev between 0.9 *
cs_item_rev and 1.1 * cs_item_rev
 order by item_id
         ,ss_item_rev
  fetch first 100 rows only;

-- end query 19 in stream 0 using template
query58.tpl

-- start query 22 in stream 0 using
template query33.tpl and seed 248487088
with ss as (
 select

i_manufact_id,sum(ss_ext_sales_price)
total_sales
 from
       store_sales,
       date_dim,
         customer_address,
         item
 where
         i_manufact_id in (select
  i_manufact_id
from
 item
where i_category in ('Books'))
 and     ss_item_sk              =
i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -7
 group by i_manufact_id),
 cs as (
 select

i_manufact_id,sum(cs_ext_sales_price)
total_sales
 from
       catalog_sales,
       date_dim,
         customer_address,
         item
 where
         i_manufact_id               in
(select
  i_manufact_id
from
 item
where i_category in ('Books'))
 and     cs_item_sk              =
i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -7
 group by i_manufact_id),
 ws as (
 select

i_manufact_id,sum(ws_ext_sales_price)
total_sales
 from
       web_sales,
       date_dim,
         customer_address,
         item
 where
         i_manufact_id               in
(select
  i_manufact_id
from
 item
where i_category in ('Books'))
 and     ws_item_sk              =
i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -7
 group by i_manufact_id)
  select  i_manufact_id ,sum(total_sales)
total_sales
 from  (select * from ss
        union all
        select * from cs
```

```
        union all
        select * from ws) tmp1
 group by i_manufact_id
 order by total_sales
 fetch first 100 rows only;

-- end query 22 in stream 0 using template
query33.tpl

-- start query 24 in stream 0 using
template query62.tpl and seed 800775315
select
   substr(w_warehouse_name,1,20)
  ,sm_type
  ,web_name
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk <= 30 ) then 1 else 0 end)
as "30 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 30) and
                 (ws_ship_date_sk -
ws_sold_date_sk <= 60) then 1 else 0 end )
as "31-60 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 60) and
                 (ws_ship_date_sk -
ws_sold_date_sk <= 90) then 1 else 0 end)
as "61-90 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 90) and
                 (ws_ship_date_sk -
ws_sold_date_sk <= 120) then 1 else 0 end)
as "91-120 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk  > 120) then 1 else 0 end)
as ">120 days"
from
   web_sales
  ,warehouse
  ,ship_mode
  ,web_site
  ,date_dim
where
    d_month_seq between 1201 and 1201 + 11
and ws_ship_date_sk   = d_date_sk
and ws_warehouse_sk   = w_warehouse_sk
and ws_ship_mode_sk   = sm_ship_mode_sk
and ws_web_site_sk    = web_site_sk
group by
   substr(w_warehouse_name,1,20)
  ,sm_type
  ,web_name
order by substr(w_warehouse_name,1,20)
        ,sm_type
       ,web_name
 fetch first 100 rows only;

-- end query 24 in stream 0 using template
query62.tpl

-- start query 27 in stream 0 using
template query63.tpl and seed 812633773
select  *
from (select i_manager_id
             ,sum(ss_sales_price) sum_sales
             ,avg(sum(ss_sales_price)) over
(partition by i_manager_id)
avg_monthly_sales
      from item
          ,store_sales
          ,date_dim
          ,store
      where ss_item_sk = i_item_sk
        and ss_sold_date_sk = d_date_sk
        and ss_store_sk = s_store_sk
        and d_month_seq in
(1178,1178+1,1178+2,1178+3,1178+4,1178+5,11
78+6,1178+7,1178+8,1178+9,1178+10,1178+11)
        and ((    i_category in
('Books','Children','Electronics')
              and i_class in
('personal','portable','refernece','self-
help')
              and i_brand in
('scholaramalgamalg #14','scholaramalgamalg
#7',

'exportiunivamalg #9','scholaramalgamalg
#9'))
           or(    i_category in
('Women','Music','Men')
              and i_class in
('accessories','classical','fragrances','pa
nts')
              and i_brand in ('amalgimporto
#1','edu packscholar #1','exportiimporto
#1',

'importoamalg #1')))
group by i_manager_id, d_moy) tmp1
where case when avg_monthly_sales > 0 then
abs (sum_sales - avg_monthly_sales) /
avg_monthly_sales else null end > 0.1
order by i_manager_id
        ,avg_monthly_sales
        ,sum_sales
 fetch first 100 rows only;

-- end query 27 in stream 0 using template
query63.tpl
-- start query 28 in stream 0 using
template query69.tpl and seed 1390437346
select
  cd_gender,
  cd_marital_status,
  cd_education_status,
  count(*) cnt1,
  cd_purchase_estimate,
  count(*) cnt2,
  cd_credit_rating,
  count(*) cnt3
 from
  customer c,customer_address
ca,customer_demographics
 where
  c.c_current_addr_sk = ca.ca_address_sk
and
  ca_state in ('AL','VA','GA') and
  cd_demo_sk = c.c_current_cdemo_sk and
  exists (select *
          from store_sales,date_dim
```

```
          where c.c_customer_sk =
ss_customer_sk and
                ss_sold_date_sk = d_date_sk
and
                d_year = 2004 and
                d_moy between 3 and 3+2)
and
   (not exists (select *
            from web_sales,date_dim
            where c.c_customer_sk =
ws_bill_customer_sk and
                  ws_sold_date_sk =
d_date_sk and
                  d_year = 2004 and
                  d_moy between 3 and 3+2)
and
    not exists (select *
            from catalog_sales,date_dim
            where c.c_customer_sk =
cs_ship_customer_sk and
                  cs_sold_date_sk =
d_date_sk and
                  d_year = 2004 and
                  d_moy between 3 and 3+2))
 group by cd_gender,
          cd_marital_status,
          cd_education_status,
          cd_purchase_estimate,
          cd_credit_rating
 order by cd_gender,
          cd_marital_status,
          cd_education_status,
          cd_purchase_estimate,
          cd_credit_rating
  fetch first 100 rows only;

-- end query 28 in stream 0 using template
query69.tpl
-- start query 29 in stream 0 using
template query60.tpl and seed 374071684
with ss as (
 select
          i_item_id,sum(ss_ext_sales_price)
total_sales
 from
       store_sales,
       date_dim,
         customer_address,
         item
 where
        i_item_id in (select
  i_item_id
from
 item
where i_category in ('Jewelry'))
 and     ss_item_sk              =
i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 cs as (
 select
          i_item_id,sum(cs_ext_sales_price)
total_sales
 from
       catalog_sales,
       date_dim,
         customer_address,
         item
 where
        i_item_id               in (select
  i_item_id
from
 item
where i_category in ('Jewelry'))
 and     cs_item_sk              =
i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 ws as (
 select
          i_item_id,sum(ws_ext_sales_price)
total_sales
 from
       web_sales,
       date_dim,
         customer_address,
         item
 where
        i_item_id               in (select
  i_item_id
from
 item
where i_category in ('Jewelry'))
 and     ws_item_sk              =
i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id)
  select
  i_item_id
,sum(total_sales) total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_item_id
 order by i_item_id
      ,total_sales
  fetch first 100 rows only;

-- end query 29 in stream 0 using template
query60.tpl
-- start query 30 in stream 0 using
template query59.tpl and seed 1976435349
with wss as
 (select d_week_seq,
```

```
        ss_store_sk,
        sum(case when (d_day_name='Sunday')
then ss_sales_price else null end)
sun_sales,
        sum(case when (d_day_name='Monday')
then ss_sales_price else null end)
mon_sales,
        sum(case when
(d_day_name='Tuesday') then ss_sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then
ss_sales_price else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then ss_sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then ss_sales_price else null end)
fri_sales,
        sum(case when
(d_day_name='Saturday') then ss_sales_price
else null end) sat_sales
 from store_sales,date_dim
 where d_date_sk = ss_sold_date_sk
 group by d_week_seq,ss_store_sk
 )
  select
s_store_name1,s_store_id1,d_week_seq1

,sun_sales1/sun_sales2,mon_sales1/mon_sales
2

,tue_sales1/tue_sales1,wed_sales1/wed_sales
2,thu_sales1/thu_sales2

,fri_sales1/fri_sales2,sat_sales1/sat_sales
2
 from
 (select s_store_name
s_store_name1,wss.d_week_seq d_week_seq1
        ,s_store_id s_store_id1,sun_sales
sun_sales1
        ,mon_sales mon_sales1,tue_sales
tue_sales1
        ,wed_sales wed_sales1,thu_sales
thu_sales1
        ,fri_sales fri_sales1,sat_sales
sat_sales1
  from wss,store,date_dim d
  where d.d_week_seq = wss.d_week_seq and
        ss_store_sk = s_store_sk and
        d_month_seq between 1189 and 1189 +
11) y,
 (select s_store_name
s_store_name2,wss.d_week_seq d_week_seq2
        ,s_store_id s_store_id2,sun_sales
sun_sales2
        ,mon_sales mon_sales2,tue_sales
tue_sales2
        ,wed_sales wed_sales2,thu_sales
thu_sales2
        ,fri_sales fri_sales2,sat_sales
sat_sales2
  from wss,store,date_dim d
  where d.d_week_seq = wss.d_week_seq and
        ss_store_sk = s_store_sk and
        d_month_seq between 1189+ 12 and
1189 + 23) x
 where s_store_id1=s_store_id2
   and d_week_seq1=d_week_seq2-52
 order by
s_store_name1,s_store_id1,d_week_seq1
 fetch first 100 rows only;

-- end query 30 in stream 0 using template
query59.tpl

-- start query 32 in stream 0 using
template query98.tpl and seed 1900673199
select i_item_desc
      ,i_category
      ,i_class
      ,i_current_price
      ,sum(ss_ext_sales_price) as
itemrevenue

,sum(ss_ext_sales_price)*100/sum(sum(ss_ext
_sales_price)) over
          (partition by i_class) as
revenueratio
from
        store_sales
        ,item
        ,date_dim
where
        ss_item_sk = i_item_sk
        and i_category in ('Shoes',
'Music', 'Home')
        and ss_sold_date_sk = d_date_sk
        and d_date between cast('1999-05-
25' as date)
                                  and
(cast('1999-05-25' as date) + 30 days)
group by
        i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price
order by
        i_category
        ,i_class
        ,i_item_id
        ,i_item_desc
        ,revenueratio;

-- end query 32 in stream 0 using template
query98.tpl

-- start query 36 in stream 0 using
template query28.tpl and seed 24799953
select  *
from (select avg(ss_list_price) B1_LP
            ,count(ss_list_price) B1_CNT
            ,count(distinct ss_list_price)
B1_CNTD
      from store_sales
      where ss_quantity between 0 and 5
        and (ss_list_price between 47 and
47+10
             or ss_coupon_amt between 11713
and 11713+1000
```

```
             or ss_wholesale_cost between
55 and 55+20)) B1,
     (select avg(ss_list_price) B2_LP
            ,count(ss_list_price) B2_CNT
            ,count(distinct ss_list_price)
B2_CNTD
      from store_sales
      where ss_quantity between 6 and 10
        and (ss_list_price between 93 and
93+10
          or ss_coupon_amt between 7733 and
7733+1000
          or ss_wholesale_cost between 43
and 43+20)) B2,
     (select avg(ss_list_price) B3_LP
            ,count(ss_list_price) B3_CNT
            ,count(distinct ss_list_price)
B3_CNTD
      from store_sales
      where ss_quantity between 11 and 15
        and (ss_list_price between 32 and
32+10
          or ss_coupon_amt between 11517
and 11517+1000
          or ss_wholesale_cost between 26
and 26+20)) B3,
     (select avg(ss_list_price) B4_LP
            ,count(ss_list_price) B4_CNT
            ,count(distinct ss_list_price)
B4_CNTD
      from store_sales
      where ss_quantity between 16 and 20
        and (ss_list_price between 147 and
147+10
          or ss_coupon_amt between 509 and
509+1000
          or ss_wholesale_cost between 78
and 78+20)) B4,
     (select avg(ss_list_price) B5_LP
            ,count(ss_list_price) B5_CNT
            ,count(distinct ss_list_price)
B5_CNTD
      from store_sales
      where ss_quantity between 21 and 25
        and (ss_list_price between 16 and
16+10
          or ss_coupon_amt between 2401 and
2401+1000
          or ss_wholesale_cost between 32
and 32+20)) B5,
     (select avg(ss_list_price) B6_LP
            ,count(ss_list_price) B6_CNT
            ,count(distinct ss_list_price)
B6_CNTD
      from store_sales
      where ss_quantity between 26 and 30
        and (ss_list_price between 11 and
11+10
          or ss_coupon_amt between 916 and
916+1000
          or ss_wholesale_cost between 6
and 6+20)) B6
 fetch first 100 rows only;

-- end query 36 in stream 0 using template
query28.tpl
```

```
-- start query 39 in stream 0 using
template query66.tpl and seed 1688498284
select
         w_warehouse_name
        ,w_warehouse_sq_ft
        ,w_city
        ,w_county
        ,w_state
        ,w_country
        ,ship_carriers
        ,year
        ,sum(jan_sales) as jan_sales
        ,sum(feb_sales) as feb_sales
        ,sum(mar_sales) as mar_sales
        ,sum(apr_sales) as apr_sales
        ,sum(may_sales) as may_sales
        ,sum(jun_sales) as jun_sales
        ,sum(jul_sales) as jul_sales
        ,sum(aug_sales) as aug_sales
        ,sum(sep_sales) as sep_sales
        ,sum(oct_sales) as oct_sales
        ,sum(nov_sales) as nov_sales
        ,sum(dec_sales) as dec_sales
        ,sum(jan_sales/w_warehouse_sq_ft)
as jan_sales_per_sq_foot
        ,sum(feb_sales/w_warehouse_sq_ft)
as feb_sales_per_sq_foot
        ,sum(mar_sales/w_warehouse_sq_ft)
as mar_sales_per_sq_foot
        ,sum(apr_sales/w_warehouse_sq_ft)
as apr_sales_per_sq_foot
        ,sum(may_sales/w_warehouse_sq_ft)
as may_sales_per_sq_foot
        ,sum(jun_sales/w_warehouse_sq_ft)
as jun_sales_per_sq_foot
        ,sum(jul_sales/w_warehouse_sq_ft)
as jul_sales_per_sq_foot
        ,sum(aug_sales/w_warehouse_sq_ft)
as aug_sales_per_sq_foot
        ,sum(sep_sales/w_warehouse_sq_ft)
as sep_sales_per_sq_foot
        ,sum(oct_sales/w_warehouse_sq_ft)
as oct_sales_per_sq_foot
        ,sum(nov_sales/w_warehouse_sq_ft)
as nov_sales_per_sq_foot
        ,sum(dec_sales/w_warehouse_sq_ft)
as dec_sales_per_sq_foot
        ,sum(jan_net) as jan_net
        ,sum(feb_net) as feb_net
        ,sum(mar_net) as mar_net
        ,sum(apr_net) as apr_net
        ,sum(may_net) as may_net
        ,sum(jun_net) as jun_net
        ,sum(jul_net) as jul_net
        ,sum(aug_net) as aug_net
        ,sum(sep_net) as sep_net
        ,sum(oct_net) as oct_net
        ,sum(nov_net) as nov_net
        ,sum(dec_net) as dec_net
 from (
    (select
        w_warehouse_name
        ,w_warehouse_sq_ft
        ,w_city
        ,w_county
        ,w_state
        ,w_country
```

```
       ,'MSC' || ',' || 'USPS' as
ship_carriers
      ,d_year as year
       ,sum(case when d_moy = 1
                then ws_sales_price*
ws_quantity else 0 end) as jan_sales
       ,sum(case when d_moy = 2
                then ws_sales_price*
ws_quantity else 0 end) as feb_sales
       ,sum(case when d_moy = 3
                then ws_sales_price*
ws_quantity else 0 end) as mar_sales
       ,sum(case when d_moy = 4
                then ws_sales_price*
ws_quantity else 0 end) as apr_sales
       ,sum(case when d_moy = 5
                then ws_sales_price*
ws_quantity else 0 end) as may_sales
       ,sum(case when d_moy = 6
                then ws_sales_price*
ws_quantity else 0 end) as jun_sales
       ,sum(case when d_moy = 7
                then ws_sales_price*
ws_quantity else 0 end) as jul_sales
       ,sum(case when d_moy = 8
                then ws_sales_price*
ws_quantity else 0 end) as aug_sales
       ,sum(case when d_moy = 9
                then ws_sales_price*
ws_quantity else 0 end) as sep_sales
       ,sum(case when d_moy = 10
                then ws_sales_price*
ws_quantity else 0 end) as oct_sales
       ,sum(case when d_moy = 11
                then ws_sales_price*
ws_quantity else 0 end) as nov_sales
       ,sum(case when d_moy = 12
                then ws_sales_price*
ws_quantity else 0 end) as dec_sales
       ,sum(case when d_moy = 1
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jan_net
       ,sum(case when d_moy = 2
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as feb_net
       ,sum(case when d_moy = 3
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as mar_net
       ,sum(case when d_moy = 4
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as apr_net
       ,sum(case when d_moy = 5
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as may_net
       ,sum(case when d_moy = 6
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jun_net
       ,sum(case when d_moy = 7
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jul_net
       ,sum(case when d_moy = 8
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as aug_net
       ,sum(case when d_moy = 9
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as sep_net
       ,sum(case when d_moy = 10
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as oct_net
       ,sum(case when d_moy = 11
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as nov_net
       ,sum(case when d_moy = 12
                then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as dec_net
     from
          web_sales
         ,warehouse
         ,date_dim
         ,time_dim
          ,ship_mode
     where
            ws_warehouse_sk =
w_warehouse_sk
        and ws_sold_date_sk = d_date_sk
        and ws_sold_time_sk = t_time_sk
       and ws_ship_mode_sk =
sm_ship_mode_sk
        and d_year = 2002
       and t_time between 18036 and
18036+28800
       and sm_carrier in ('MSC','USPS')
     group by
        w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
      ,d_year
       )
 union all
    (select
       w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,'MSC' || ',' || 'USPS' as
ship_carriers
      ,d_year as year
       ,sum(case when d_moy = 1
                then cs_ext_sales_price*
cs_quantity else 0 end) as jan_sales
       ,sum(case when d_moy = 2
                then cs_ext_sales_price*
cs_quantity else 0 end) as feb_sales
       ,sum(case when d_moy = 3
                then cs_ext_sales_price*
cs_quantity else 0 end) as mar_sales
       ,sum(case when d_moy = 4
```

```
               then cs_ext_sales_price*
cs_quantity else 0 end) as apr_sales
       ,sum(case when d_moy = 5
               then cs_ext_sales_price*
cs_quantity else 0 end) as may_sales
       ,sum(case when d_moy = 6
               then cs_ext_sales_price*
cs_quantity else 0 end) as jun_sales
       ,sum(case when d_moy = 7
               then cs_ext_sales_price*
cs_quantity else 0 end) as jul_sales
       ,sum(case when d_moy = 8
               then cs_ext_sales_price*
cs_quantity else 0 end) as aug_sales
       ,sum(case when d_moy = 9
               then cs_ext_sales_price*
cs_quantity else 0 end) as sep_sales
       ,sum(case when d_moy = 10
               then cs_ext_sales_price*
cs_quantity else 0 end) as oct_sales
       ,sum(case when d_moy = 11
               then cs_ext_sales_price*
cs_quantity else 0 end) as nov_sales
       ,sum(case when d_moy = 12
               then cs_ext_sales_price*
cs_quantity else 0 end) as dec_sales
       ,sum(case when d_moy = 1
               then cs_net_profit *
cs_quantity else 0 end) as jan_net
       ,sum(case when d_moy = 2
               then cs_net_profit *
cs_quantity else 0 end) as feb_net
       ,sum(case when d_moy = 3
               then cs_net_profit *
cs_quantity else 0 end) as mar_net
       ,sum(case when d_moy = 4
               then cs_net_profit *
cs_quantity else 0 end) as apr_net
       ,sum(case when d_moy = 5
               then cs_net_profit *
cs_quantity else 0 end) as may_net
       ,sum(case when d_moy = 6
               then cs_net_profit *
cs_quantity else 0 end) as jun_net
       ,sum(case when d_moy = 7
               then cs_net_profit *
cs_quantity else 0 end) as jul_net
       ,sum(case when d_moy = 8
               then cs_net_profit *
cs_quantity else 0 end) as aug_net
       ,sum(case when d_moy = 9
               then cs_net_profit *
cs_quantity else 0 end) as sep_net
       ,sum(case when d_moy = 10
               then cs_net_profit *
cs_quantity else 0 end) as oct_net
       ,sum(case when d_moy = 11
               then cs_net_profit *
cs_quantity else 0 end) as nov_net
       ,sum(case when d_moy = 12
               then cs_net_profit *
cs_quantity else 0 end) as dec_net
     from
          catalog_sales
         ,warehouse
         ,date_dim
         ,time_dim
         ,ship_mode
     where
            cs_warehouse_sk =
w_warehouse_sk
        and cs_sold_date_sk = d_date_sk
        and cs_sold_time_sk = t_time_sk
        and cs_ship_mode_sk =
sm_ship_mode_sk
        and d_year = 2002
        and t_time between 18036 AND
18036+28800
        and sm_carrier in ('MSC','USPS')
     group by
        w_warehouse_name
        ,w_warehouse_sq_ft
        ,w_city
        ,w_county
        ,w_state
        ,w_country
        ,d_year
     )
 ) x
 group by
        w_warehouse_name
        ,w_warehouse_sq_ft
        ,w_city
        ,w_county
        ,w_state
        ,w_country
        ,ship_carriers
       ,year
 order by w_warehouse_name
  fetch first 100 rows only;

-- end query 39 in stream 0 using template
query66.tpl
-- start query 40 in stream 0 using
template query90.tpl and seed 1949014749
select  cast(amc as decimal(15,4))/cast(pmc
as decimal(15,4)) am_pm_ratio
 from ( select count(*) amc
       from web_sales,
household_demographics , time_dim, web_page
       where ws_sold_time_sk =
time_dim.t_time_sk
         and ws_ship_hdemo_sk =
household_demographics.hd_demo_sk
         and ws_web_page_sk =
web_page.wp_web_page_sk
         and time_dim.t_hour between 11 and
11+1
         and
household_demographics.hd_dep_count = 9
         and web_page.wp_char_count between
5000 and 5200) at,
      ( select count(*) pmc
       from web_sales,
household_demographics , time_dim, web_page
       where ws_sold_time_sk =
time_dim.t_time_sk
         and ws_ship_hdemo_sk =
household_demographics.hd_demo_sk
         and ws_web_page_sk =
web_page.wp_web_page_sk
         and time_dim.t_hour between 18 and
18+1
```

```
         and
household_demographics.hd_dep_count = 9
         and web_page.wp_char_count between
5000 and 5200) pt
 order by am_pm_ratio
  fetch first 100 rows only;

-- end query 40 in stream 0 using template
query90.tpl

-- start query 44 in stream 0 using
template query92.tpl and seed 643980925
select
   sum(ws_ext_discount_amt)  as "Excess
Discount Amount"
from
    web_sales
   ,item
   ,date_dim
where
i_manufact_id = 926
and i_item_sk = ws_item_sk
and d_date between '1999-01-03' and
        (cast('1999-01-03' as date) + 90
days)
and d_date_sk = ws_sold_date_sk
and ws_ext_discount_amt
     > (
         SELECT
            1.3 * avg(ws_ext_discount_amt)
         FROM
            web_sales
           ,date_dim
         WHERE
              ws_item_sk = i_item_sk
          and d_date between '1999-01-03'
and
                              (cast('1999-
01-03' as date) + 90 days)
          and d_date_sk = ws_sold_date_sk
      )
order by sum(ws_ext_discount_amt)
 fetch first 100 rows only;

-- end query 44 in stream 0 using template
query92.tpl
-- start query 45 in stream 0 using
template query3.tpl and seed 691662667
select  dt.d_year
       ,item.i_brand_id brand_id
       ,item.i_brand brand
       ,sum(ss_net_profit) sum_agg
 from  date_dim dt
      ,store_sales
      ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
   and store_sales.ss_item_sk =
item.i_item_sk
   and item.i_manufact_id = 596
   and dt.d_moy=12
 group by dt.d_year
      ,item.i_brand
      ,item.i_brand_id
 order by dt.d_year
         ,sum_agg desc
         ,brand_id
  fetch first 100 rows only;

-- end query 45 in stream 0 using template
query3.tpl

-- start query 49 in stream 0 using
template query9.tpl and seed 937436805
select case when (select count(*)
                  from store_sales
                  where ss_quantity between
1 and 20) > 62316685
            then (select
avg(ss_ext_sales_price)
                  from store_sales
                  where ss_quantity between
1 and 20)
            else (select
avg(ss_net_paid_inc_tax)
                  from store_sales
                  where ss_quantity between
1 and 20) end bucket1 ,
       case when (select count(*)
                  from store_sales
                  where ss_quantity between
21 and 40) > 19045798
            then (select
avg(ss_ext_sales_price)
                  from store_sales
                  where ss_quantity between
21 and 40)
            else (select
avg(ss_net_paid_inc_tax)
                  from store_sales
                  where ss_quantity between
21 and 40) end bucket2,
       case when (select count(*)
                  from store_sales
                  where ss_quantity between
41 and 60) > 365541424
            then (select
avg(ss_ext_sales_price)
                  from store_sales
                  where ss_quantity between
41 and 60)
            else (select
avg(ss_net_paid_inc_tax)
                  from store_sales
                  where ss_quantity between
41 and 60) end bucket3,
       case when (select count(*)
                  from store_sales
                  where ss_quantity between
61 and 80) > 216357808
            then (select
avg(ss_ext_sales_price)
                  from store_sales
                  where ss_quantity between
61 and 80)
            else (select
avg(ss_net_paid_inc_tax)
                  from store_sales
                  where ss_quantity between
61 and 80) end bucket4,
       case when (select count(*)
                  from store_sales
                  where ss_quantity between
81 and 100) > 184483884
```

```
            then (select
avg(ss_ext_sales_price)
                  from store_sales
                  where ss_quantity between
81 and 100)
            else (select
avg(ss_net_paid_inc_tax)
                  from store_sales
                  where ss_quantity between
81 and 100) end bucket5
from reason
where r_reason_sk = 1
;

-- end query 49 in stream 0 using template
query9.tpl

-- start query 52 in stream 0 using
template query93.tpl and seed 1821797098
select  ss_customer_sk
            ,sum(act_sales) sumsales
      from (select ss_item_sk
                  ,ss_ticket_number
                  ,ss_customer_sk
                  ,case when
sr_return_quantity is not null then
(ss_quantity-
sr_return_quantity)*ss_sales_price

else (ss_quantity*ss_sales_price) end
act_sales
            from store_sales left outer
join store_returns on (sr_item_sk =
ss_item_sk

and sr_ticket_number = ss_ticket_number)
               ,reason
            where sr_reason_sk =
r_reason_sk
              and r_reason_desc = 'reason
74') t
      group by ss_customer_sk
      order by sumsales, ss_customer_sk
 fetch first 100 rows only;

-- end query 52 in stream 0 using template
query93.tpl

-- start query 55 in stream 0 using
template query22.tpl and seed 635815297
select  i_product_name
             ,i_brand
             ,i_class
             ,i_category
             ,avg(cast(inv_quantity_on_hand
as double)) qoh
       from inventory
           ,date_dim
           ,item
           ,warehouse
       where inv_date_sk=d_date_sk
              and inv_item_sk=i_item_sk
              and inv_warehouse_sk =
w_warehouse_sk
              and d_month_seq between 1199
and 1199 + 11
       group by rollup(i_product_name
                      ,i_brand
                      ,i_class
                      ,i_category)
order by qoh, i_product_name, i_brand,
i_class, i_category
 fetch first 100 rows only;

-- end query 55 in stream 0 using template
query22.tpl
-- start query 56 in stream 0 using
template query89.tpl and seed 2079706651
select  *
from(
select i_category, i_class, i_brand,
       s_store_name, s_company_name,
       d_moy,
       sum(ss_sales_price) sum_sales,
       avg(sum(ss_sales_price)) over
         (partition by i_category, i_brand,
s_store_name, s_company_name)
        avg_monthly_sales
from item, store_sales, date_dim, store
where ss_item_sk = i_item_sk and
      ss_sold_date_sk = d_date_sk and
      ss_store_sk = s_store_sk and
      d_year in (1999) and
        ((i_category in
('Books','Jewelry','Men') and
          i_class in
('history','birdal','pants')
         )
      or (i_category in
('Music','Home','Shoes') and
          i_class in
('pop','furniture','athletic')
        ))
group by i_category, i_class, i_brand,
         s_store_name, s_company_name,
d_moy) tmp1
where case when (avg_monthly_sales <> 0)
then (abs(sum_sales - avg_monthly_sales) /
avg_monthly_sales) else null end > 0.1
order by sum_sales - avg_monthly_sales,
s_store_name
 fetch first 100 rows only;

-- end query 56 in stream 0 using template
query89.tpl
-- start query 59 in stream 0 using
template query52.tpl and seed 223505300
select  dt.d_year
       ,item.i_brand_id brand_id
       ,item.i_brand brand
       ,sum(ss_ext_sales_price) ext_price
 from date_dim dt
     ,store_sales
     ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
    and store_sales.ss_item_sk =
item.i_item_sk
    and item.i_manager_id = 1
    and dt.d_moy=11
    and dt.d_year=1999
 group by dt.d_year
       ,item.i_brand
       ,item.i_brand_id
```

```
 order by dt.d_year
       ,ext_price desc
       ,brand_id
 fetch first 100 rows only ;

-- end query 59 in stream 0 using template
query52.tpl
-- start query 60 in stream 0 using
template query50.tpl and seed 1718577076
select
   s_store_name
  ,s_company_id
  ,s_street_number
  ,s_street_name
  ,s_street_type
  ,s_suite_number
  ,s_city
  ,s_county
  ,s_state
  ,s_zip
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk <= 30 ) then 1 else 0 end)
as "30 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 30) and
                 (sr_returned_date_sk -
ss_sold_date_sk <= 60) then 1 else 0 end )
as "31-60 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 60) and
                 (sr_returned_date_sk -
ss_sold_date_sk <= 90) then 1 else 0 end)
as "61-90 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 90) and
                 (sr_returned_date_sk -
ss_sold_date_sk <= 120) then 1 else 0 end)
as "91-120 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk  > 120) then 1 else 0 end)
as ">120 days"
from
   store_sales
  ,store_returns
  ,store
  ,date_dim d1
  ,date_dim d2
where
    d2.d_year = 1999
and d2.d_moy  = 10
and ss_ticket_number = sr_ticket_number
and ss_item_sk = sr_item_sk
and ss_sold_date_sk   = d1.d_date_sk
and sr_returned_date_sk   = d2.d_date_sk
and ss_customer_sk = sr_customer_sk
and ss_store_sk = s_store_sk
group by
   s_store_name
  ,s_company_id
  ,s_street_number
  ,s_street_name
  ,s_street_type
  ,s_suite_number
  ,s_city
  ,s_county
  ,s_state
  ,s_zip
order by s_store_name
        ,s_company_id
        ,s_street_number
        ,s_street_name
        ,s_street_type
        ,s_suite_number
        ,s_city
        ,s_county
        ,s_state
        ,s_zip
 fetch first 100 rows only;

-- end query 60 in stream 0 using template
query50.tpl
-- start query 61 in stream 0 using
template query42.tpl and seed 709936855
select  dt.d_year
       ,item.i_category_id
       ,item.i_category
       ,sum(ss_ext_sales_price)
 from   date_dim dt
       ,store_sales
       ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
       and store_sales.ss_item_sk =
item.i_item_sk
       and item.i_manager_id = 1
       and dt.d_moy=12
       and dt.d_year=2000
 group by      dt.d_year
               ,item.i_category_id
               ,item.i_category
 order by       sum(ss_ext_sales_price)
desc,dt.d_year
               ,item.i_category_id
               ,item.i_category
 fetch first 100 rows only ;

-- end query 61 in stream 0 using template
query42.tpl
-- start query 62 in stream 0 using
template query41.tpl and seed 944250029
select  distinct(i_product_name)
 from item i1
 where i_manufact_id between 716 and 716+40
   and (select count(*) as item_cnt
        from item
        where (i_manufact = i1.i_manufact
and
        ((i_category = 'Women' and
        (i_color = 'spring' or i_color =
'hot') and
        (i_units = 'Carton' or i_units =
'Tbl') and
        (i_size = 'large' or i_size =
'N/A')
        ) or
        (i_category = 'Women' and
        (i_color = 'magenta' or i_color =
'goldenrod') and
        (i_units = 'Cup' or i_units = 'Oz')
and
        (i_size = 'economy' or i_size =
'extra large')
        ) or
        (i_category = 'Men' and
```

```
        (i_color = 'cyan' or i_color =
'antique') and
        (i_units = 'Dozen' or i_units =
'Case') and
        (i_size = 'medium' or i_size =
'petite')
        ) or
        (i_category = 'Men' and
        (i_color = 'moccasin' or i_color =
'black') and
        (i_units = 'Box' or i_units =
'Pallet') and
        (i_size = 'large' or i_size =
'N/A')
        ))) or
       (i_manufact = i1.i_manufact and
        ((i_category = 'Women' and
        (i_color = 'azure' or i_color =
'light') and
        (i_units = 'Gross' or i_units =
'Each') and
        (i_size = 'large' or i_size =
'N/A')
        ) or
        (i_category = 'Women' and
        (i_color = 'mint' or i_color =
'burnished') and
        (i_units = 'N/A' or i_units =
'Unknown') and
        (i_size = 'economy' or i_size =
'extra large')
        ) or
        (i_category = 'Men' and
        (i_color = 'floral' or i_color =
'midnight') and
        (i_units = 'Pound' or i_units =
'Ton') and
        (i_size = 'medium' or i_size =
'petite')
        ) or
        (i_category = 'Men' and
        (i_color = 'navy' or i_color =
'blue') and
        (i_units = 'Bundle' or i_units =
'Ounce') and
        (i_size = 'large' or i_size =
'N/A')
        )))) > 0
 order by i_product_name
  fetch first 100 rows only;

-- end query 62 in stream 0 using template
query41.tpl

-- start query 64 in stream 0 using
template query12.tpl and seed 918962166
select  i_item_desc
      ,i_category
      ,i_class
      ,i_current_price
      ,sum(ws_ext_sales_price) as
itemrevenue

,sum(ws_ext_sales_price)*100/sum(sum(ws_ext
_sales_price)) over
          (partition by i_class) as
revenueratio
from
        web_sales
        ,item
        ,date_dim
where
        ws_item_sk = i_item_sk
        and i_category in ('Jewelry',
'Men', 'Books')
        and ws_sold_date_sk = d_date_sk
        and d_date between cast('2002-06-
11' as date)
                                   and
(cast('2002-06-11' as date) + 30 days)
group by
        i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price
order by
        i_category
        ,i_class
        ,i_item_id
        ,i_item_desc
        ,revenueratio
 fetch first 100 rows only;

-- end query 64 in stream 0 using template
query12.tpl
-- start query 65 in stream 0 using
template query20.tpl and seed 711739272
select  i_item_desc
       ,i_category
       ,i_class
       ,i_current_price
       ,sum(cs_ext_sales_price) as
itemrevenue

,sum(cs_ext_sales_price)*100/sum(sum(cs_ext
_sales_price)) over
           (partition by i_class) as
revenueratio
 from   catalog_sales
     ,item
     ,date_dim
 where cs_item_sk = i_item_sk
   and i_category in ('Jewelry', 'Music',
'Men')
   and cs_sold_date_sk = d_date_sk
 and d_date between cast('2000-02-09' as
date)
                                  and
(cast('2000-02-09' as date) + 30 days)
 group by i_item_id
         ,i_item_desc
         ,i_category
         ,i_class
         ,i_current_price
 order by i_category
         ,i_class
         ,i_item_id
         ,i_item_desc
         ,revenueratio
 fetch first 100 rows only;

-- end query 65 in stream 0 using template
query20.tpl
```

```
-- start query 66 in stream 0 using
template query88.tpl and seed 1924183468
select  *
from
 (select count(*) h8_30_to_9
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 8
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s1,
 (select count(*) h9_to_9_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 9
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s2,
 (select count(*) h9_30_to_10
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 9
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s3,
 (select count(*) h10_to_10_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 10
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s4,
 (select count(*) h10_30_to_11
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 10
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s5,
 (select count(*) h11_to_11_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
```

```
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 11
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s6,
 (select count(*) h11_30_to_12
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 11
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s7,
 (select count(*) h12_to_12_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 12
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s8
;

-- end query 66 in stream 0 using template
query88.tpl

-- start query 72 in stream 0 using
template query71.tpl and seed 1436004490
select i_brand_id brand_id, i_brand
brand,t_hour,t_minute,
       sum(ext_price) ext_price
 from item, (select ws_ext_sales_price as
ext_price,
                        ws_sold_date_sk as
sold_date_sk,
                        ws_item_sk as
sold_item_sk,
                        ws_sold_time_sk as
time_sk
                from web_sales,date_dim
                where d_date_sk =
ws_sold_date_sk
                  and d_moy=12
                  and d_year=1998
                union all
                select cs_ext_sales_price
as ext_price,
                        cs_sold_date_sk as
sold_date_sk,
                        cs_item_sk as
sold_item_sk,
                        cs_sold_time_sk as
time_sk
                from
catalog_sales,date_dim
                where d_date_sk =
cs_sold_date_sk
                  and d_moy=12
                  and d_year=1998
                union all
                select ss_ext_sales_price
as ext_price,
                        ss_sold_date_sk as
sold_date_sk,
                        ss_item_sk as
sold_item_sk,
                        ss_sold_time_sk as
time_sk
                from store_sales,date_dim
                where d_date_sk =
ss_sold_date_sk
                  and d_moy=12
                  and d_year=1998
                ) as tmp,time_dim
 where
   sold_item_sk = i_item_sk
   and i_manager_id=1
   and time_sk = t_time_sk
   and (t_meal_time = 'breakfast' or
t_meal_time = 'dinner')
 group by i_brand,
i_brand_id,t_hour,t_minute
 order by ext_price desc, i_brand_id
 ;
```

```
-- end query 72 in stream 0 using template
query71.tpl
-- start query 73 in stream 0 using
template query34.tpl and seed 1451328249
select c_last_name
       ,c_first_name
       ,c_salutation
       ,c_preferred_cust_flag
       ,ss_ticket_number
       ,cnt from
   (select ss_ticket_number
          ,ss_customer_sk
          ,count(*) cnt
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and (date_dim.d_dom between 1 and 3 or
date_dim.d_dom between 25 and 28)
    and
(household_demographics.hd_buy_potential =
'1001-5000' or

household_demographics.hd_buy_potential =
'5001-10000')
    and
household_demographics.hd_vehicle_count > 0
    and (case when
household_demographics.hd_vehicle_count > 0
       then
cast(household_demographics.hd_dep_count as
double) /
cast(household_demographics.hd_vehicle_coun
t as double)
       else null
       end)  > 1.2
    and date_dim.d_year in
(1999,1999+1,1999+2)
    and store.s_county in ('Sierra
County','Lunenburg County','Jackson
County','Harmon County',
                           'Mesa
County','Pipestone County','Pennington
County','Perry County')
    group by
ss_ticket_number,ss_customer_sk)
dn,customer
    where ss_customer_sk = c_customer_sk
      and cnt between 15 and 20
    order by
c_last_name,c_first_name,c_salutation,c_pre
ferred_cust_flag desc;

-- end query 73 in stream 0 using template
query34.tpl

-- start query 78 in stream 0 using
template query77.tpl and seed 1879081522
with ss as
 (select s_store_sk,
        sum(ss_ext_sales_price) as sales,
        sum(ss_net_profit) as profit
 from store_sales,
      date_dim,
      store
 where ss_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and (cast('2002-08-24' as
date) +  30 days)
       and ss_store_sk = s_store_sk
 group by s_store_sk)
 ,
 sr as
 (select s_store_sk,
        sum(sr_return_amt) as returns,
        sum(sr_net_loss) as profit_loss
 from store_returns,
      date_dim,
      store
 where sr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and (cast('2002-08-24' as
date) +  30 days)
       and sr_store_sk = s_store_sk
 group by s_store_sk),
 cs as
 (select cs_call_center_sk,
        sum(cs_ext_sales_price) as sales,
        sum(cs_net_profit) as profit
 from catalog_sales,
      date_dim
 where cs_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and (cast('2002-08-24' as
date) +  30 days)
 group by cs_call_center_sk
 ),
 cr as
 (select
        sum(cr_return_amount) as returns,
        sum(cr_net_loss) as profit_loss
 from catalog_returns,
      date_dim
 where cr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and (cast('2002-08-24' as
date) +  30 days)
 ),
 ws as
 ( select wp_web_page_sk,
        sum(ws_ext_sales_price) as sales,
        sum(ws_net_profit) as profit
 from web_sales,
      date_dim,
      web_page
 where ws_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and (cast('2002-08-24' as
date) +  30 days)
       and ws_web_page_sk = wp_web_page_sk
 group by wp_web_page_sk),
 wr as
 (select wp_web_page_sk,
```

```
        sum(wr_return_amt) as returns,
        sum(wr_net_loss) as profit_loss
 from web_returns,
      date_dim,
      web_page
 where wr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and (cast('2002-08-24' as
date) +  30 days)
       and wr_web_page_sk = wp_web_page_sk
 group by wp_web_page_sk)
  select  channel
        , id
        , sum(sales) as sales
        , sum(returns) as returns
        , sum(profit) as profit
 from
 (select 'store channel' as channel
        , ss.s_store_sk as id
        , sales
        , coalesce(returns, 0) as returns
        , (profit -
coalesce(profit_loss,0)) as profit
 from   ss left join sr
        on  ss.s_store_sk = sr.s_store_sk
 union all
 select 'catalog channel' as channel
        , cs_call_center_sk as id
        , sales
        , returns
        , (profit - profit_loss) as profit
 from  cs
       , cr
 union all
 select 'web channel' as channel
        , ws.wp_web_page_sk as id
        , sales
        , coalesce(returns, 0) returns
        , (profit -
coalesce(profit_loss,0)) as profit
 from   ws left join wr
        on  ws.wp_web_page_sk =
wr.wp_web_page_sk
 ) x
 group by rollup (channel, id)
 order by channel
         ,id
  fetch first 100 rows only;

-- end query 78 in stream 0 using template
query77.tpl
-- start query 79 in stream 0 using
template query73.tpl and seed 413577677
select c_last_name
       ,c_first_name
       ,c_salutation
       ,c_preferred_cust_flag
       ,ss_ticket_number
       ,cnt from
   (select ss_ticket_number
          ,ss_customer_sk
          ,count(*) cnt
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and date_dim.d_dom between 1 and 2
    and
(household_demographics.hd_buy_potential =
'501-1000' or

household_demographics.hd_buy_potential =
'5001-10000')
    and
household_demographics.hd_vehicle_count > 0
    and case when
household_demographics.hd_vehicle_count > 0
then

household_demographics.hd_dep_count/
household_demographics.hd_vehicle_count
else null end > 1
    and date_dim.d_year in
(1999,1999+1,1999+2)
    and store.s_county in ('Lea
County','West Feliciana Parish','Nowata
County','Jackson County')
    group by
ss_ticket_number,ss_customer_sk)
dj,customer
    where ss_customer_sk = c_customer_sk
      and cnt between 1 and 5
    order by cnt desc;

-- end query 79 in stream 0 using template
query73.tpl
-- start query 80 in stream 0 using
template query84.tpl and seed 1842474049
select  c_customer_id as customer_id
       ,c_last_name || ', ' ||
coalesce(c_first_name,'') as customername
 from customer
     ,customer_address
     ,customer_demographics
     ,household_demographics
     ,income_band
     ,store_returns
 where ca_city          =  'Mount Zion'
   and c_current_addr_sk = ca_address_sk
   and ib_lower_bound   >=  50749
   and ib_upper_bound   <=  50749 + 50000
   and ib_income_band_sk =
hd_income_band_sk
   and cd_demo_sk = c_current_cdemo_sk
   and hd_demo_sk = c_current_hdemo_sk
   and sr_cdemo_sk = cd_demo_sk
 order by c_customer_id
  fetch first 100 rows only;

-- end query 80 in stream 0 using template
query84.tpl

-- start query 82 in stream 0 using
template query55.tpl and seed 1117454508
select  i_brand_id brand_id, i_brand brand,
        sum(ss_ext_sales_price) ext_price
 from date_dim, store_sales, item
```

```
 where d_date_sk = ss_sold_date_sk
       and ss_item_sk = i_item_sk
       and i_manager_id=48
       and d_moy=11
       and d_year=2001
 group by i_brand, i_brand_id
 order by ext_price desc, i_brand_id
 fetch first 100 rows only ;

-- end query 82 in stream 0 using template
query55.tpl
-- start query 83 in stream 0 using
template query56.tpl and seed 1152645577
with ss as (
 select i_item_id,sum(ss_ext_sales_price)
total_sales
 from
       store_sales,
       date_dim,
         customer_address,
         item
 where i_item_id in (select
     i_item_id
from item
where i_color in
('maroon','powder','lawn'))
 and     ss_item_sk              =
i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 cs as (
 select i_item_id,sum(cs_ext_sales_price)
total_sales
 from
       catalog_sales,
       date_dim,
         customer_address,
         item
 where
         i_item_id               in (select
  i_item_id
from item
where i_color in
('maroon','powder','lawn'))
 and     cs_item_sk              =
i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 ws as (
 select i_item_id,sum(ws_ext_sales_price)
total_sales
 from
       web_sales,
       date_dim,
         customer_address,
         item
 where
         i_item_id               in (select
  i_item_id
from item
where i_color in
('maroon','powder','lawn'))
 and     ws_item_sk              =
i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id)
  select  i_item_id ,sum(total_sales)
total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_item_id
 order by total_sales
  fetch first 100 rows only;

-- end query 83 in stream 0 using template
query56.tpl
-- start query 84 in stream 0 using
template query2.tpl and seed 1528114170
with wscs as
 (select sold_date_sk
        ,sales_price
  from (select ws_sold_date_sk sold_date_sk
              ,ws_ext_sales_price
sales_price
        from web_sales) x
        union all
       (select cs_sold_date_sk sold_date_sk
              ,cs_ext_sales_price
sales_price
        from catalog_sales)),
 wswscs as
 (select d_week_seq,
        sum(case when (d_day_name='Sunday')
then sales_price else null end) sun_sales,
        sum(case when (d_day_name='Monday')
then sales_price else null end) mon_sales,
        sum(case when
(d_day_name='Tuesday') then sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then sales_price
else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then sales_price else null end) fri_sales,
        sum(case when
(d_day_name='Saturday') then sales_price
else null end) sat_sales
 from wscs
     ,date_dim
 where d_date_sk = sold_date_sk
```

```
 group by d_week_seq)
 select d_week_seq1
       ,round(sun_sales1/sun_sales2,2)
       ,round(mon_sales1/mon_sales2,2)
       ,round(tue_sales1/tue_sales2,2)
       ,round(wed_sales1/wed_sales2,2)
       ,round(thu_sales1/thu_sales2,2)
       ,round(fri_sales1/fri_sales2,2)
       ,round(sat_sales1/sat_sales2,2)
 from
 (select wswscs.d_week_seq d_week_seq1
        ,sun_sales sun_sales1
        ,mon_sales mon_sales1
        ,tue_sales tue_sales1
        ,wed_sales wed_sales1
        ,thu_sales thu_sales1
        ,fri_sales fri_sales1
        ,sat_sales sat_sales1
  from wswscs,date_dim
  where date_dim.d_week_seq =
wswscs.d_week_seq and
        d_year = 1998) y,
 (select wswscs.d_week_seq d_week_seq2
        ,sun_sales sun_sales2
        ,mon_sales mon_sales2
        ,tue_sales tue_sales2
        ,wed_sales wed_sales2
        ,thu_sales thu_sales2
        ,fri_sales fri_sales2
        ,sat_sales sat_sales2
  from wswscs
      ,date_dim
  where date_dim.d_week_seq =
wswscs.d_week_seq and
        d_year = 1998+1) z
 where d_week_seq1=d_week_seq2-53
 order by d_week_seq1;

-- end query 84 in stream 0 using template
query2.tpl
-- start query 85 in stream 0 using
template query26.tpl and seed 1427200905
select  i_item_id,
        avg(cast(cs_quantity as double))
agg1,
        avg(cs_list_price) agg2,
        avg(cs_coupon_amt) agg3,
        avg(cs_sales_price) agg4
 from catalog_sales, customer_demographics,
date_dim, item, promotion
 where cs_sold_date_sk = d_date_sk and
       cs_item_sk = i_item_sk and
       cs_bill_cdemo_sk = cd_demo_sk and
       cs_promo_sk = p_promo_sk and
       cd_gender = 'M' and
       cd_marital_status = 'D' and
       cd_education_status = 'Advanced
Degree' and
       (p_channel_email = 'N' or
p_channel_event = 'N') and
       d_year = 2000
 group by i_item_id
 order by i_item_id
  fetch first 100 rows only;

-- end query 85 in stream 0 using template
query26.tpl
```

```
-- start query 86 in stream 0 using
template query40.tpl and seed 600490395
select
   w_state
  ,i_item_id
  ,sum(case when (cast(d_date as date) <
cast ('2000-04-27' as date))
              then cs_sales_price -
coalesce(cr_refunded_cash,0) else 0 end) as
sales_before
  ,sum(case when (cast(d_date as date) >=
cast ('2000-04-27' as date))
              then cs_sales_price -
coalesce(cr_refunded_cash,0) else 0 end) as
sales_after
 from
   catalog_sales left outer join
catalog_returns on
       (cs_order_number = cr_order_number
        and cs_item_sk = cr_item_sk)
  ,warehouse
  ,item
  ,date_dim
 where
     i_current_price between 0.99 and 1.49
 and i_item_sk          = cs_item_sk
 and cs_warehouse_sk    = w_warehouse_sk
 and cs_sold_date_sk    = d_date_sk
 and d_date between (cast ('2000-04-27' as
date) - 30 days)
                and (cast ('2000-04-27' as
date) + 30 days)
 group by
    w_state,i_item_id
 order by w_state,i_item_id
 fetch first 100 rows only;

-- end query 86 in stream 0 using template
query40.tpl

-- start query 88 in stream 0 using
template query53.tpl and seed 1796782974
select  * from
(select i_manufact_id,
sum(ss_sales_price) sum_sales,
avg(sum(ss_sales_price)) over (partition by
i_manufact_id) avg_quarterly_sales
from item, store_sales, date_dim, store
where ss_item_sk = i_item_sk and
ss_sold_date_sk = d_date_sk and
ss_store_sk = s_store_sk and
d_month_seq in
(1198,1198+1,1198+2,1198+3,1198+4,1198+5,11
98+6,1198+7,1198+8,1198+9,1198+10,1198+11)
and
((i_category in
('Books','Children','Electronics') and
i_class in
('personal','portable','reference','self-
help') and
i_brand in ('scholaramalgamalg
#14','scholaramalgamalg #7',
               'exportiunivamalg
#9','scholaramalgamalg #9'))
or(i_category in ('Women','Music','Men')
and
```

```
i_class in
('accessories','classical','fragrances','pa
nts') and
i_brand in ('amalgimporto #1','edu
packscholar #1','exportiimporto #1',
               'importoamalg #1')))
group by i_manufact_id, d_qoy ) tmp1
where case when avg_quarterly_sales > 0
       then abs (sum_sales -
avg_quarterly_sales)/ avg_quarterly_sales
       else null end > 0.1
order by avg_quarterly_sales,
         sum_sales,
         i_manufact_id
 fetch first 100 rows only;

-- end query 88 in stream 0 using template
query53.tpl
-- start query 89 in stream 0 using
template query79.tpl and seed 2112737383
select

c_last_name,c_first_name,substr(s_city,1,30
),ss_ticket_number,amt,profit
  from
   (select ss_ticket_number
          ,ss_customer_sk
          ,store.s_city
          ,sum(ss_coupon_amt) amt
          ,sum(ss_net_profit) profit
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and
(household_demographics.hd_dep_count = 3 or
household_demographics.hd_vehicle_count >
1)
    and date_dim.d_dow = 1
    and date_dim.d_year in
(2000,2000+1,2000+2)
    and store.s_number_employees between
200 and 295
    group by
ss_ticket_number,ss_customer_sk,ss_addr_sk,
store.s_city) ms,customer
    where ss_customer_sk = c_customer_sk
 order by
c_last_name,c_first_name,substr(s_city,1,30
), profit
 fetch first 100 rows only;

-- end query 89 in stream 0 using template
query79.tpl

-- start query 96 in stream 0 using
template query83.tpl and seed 593789178
with sr_items as
 (select i_item_id item_id,
        sum(sr_return_quantity) sr_item_qty
 from store_returns,
      item,
      date_dim
 where sr_item_sk = i_item_sk
 and   d_date    in
       (select d_date
       from date_dim
       where d_week_seq in
              (select d_week_seq
              from date_dim
         where d_date in ('1999-06-
14','1999-08-26','1999-11-06')))
 and   sr_returned_date_sk   = d_date_sk
 group by i_item_id),
 cr_items as
 (select i_item_id item_id,
        sum(cr_return_quantity) cr_item_qty
 from catalog_returns,
      item,
      date_dim
 where cr_item_sk = i_item_sk
 and   d_date    in
       (select d_date
       from date_dim
       where d_week_seq in
              (select d_week_seq
              from date_dim
         where d_date in ('1999-06-
14','1999-08-26','1999-11-06')))
 and   cr_returned_date_sk   = d_date_sk
 group by i_item_id),
 wr_items as
 (select i_item_id item_id,
        sum(wr_return_quantity) wr_item_qty
 from web_returns,
      item,
      date_dim
 where wr_item_sk = i_item_sk
 and   d_date    in
       (select d_date
       from date_dim
       where d_week_seq in
              (select d_week_seq
              from date_dim
              where d_date in ('1999-06-
14','1999-08-26','1999-11-06')))
 and   wr_returned_date_sk   = d_date_sk
 group by i_item_id)
  select  sr_items.item_id
       ,sr_item_qty
       ,cast(sr_item_qty as
double)/(cast(sr_item_qty+cr_item_qty+wr_it
em_qty as double))/3.0 * 100 sr_dev
       ,cr_item_qty
       ,cast(cr_item_qty as
double)/(cast(sr_item_qty+cr_item_qty+wr_it
em_qty as double))/3.0 * 100 cr_dev
       ,wr_item_qty
       ,cast(wr_item_qty as
double)/(cast(sr_item_qty+cr_item_qty+wr_it
em_qty as double))/3.0 * 100 wr_dev

,(sr_item_qty+cr_item_qty+wr_item_qty)/3.0
average
 from sr_items
     ,cr_items
     ,wr_items
 where sr_items.item_id=cr_items.item_id
   and sr_items.item_id=wr_items.item_id
```

```
 order by sr_items.item_id
         ,sr_item_qty
  fetch first 100 rows only;

-- end query 96 in stream 0 using template
query83.tpl
-- start query 97 in stream 0 using
template query61.tpl and seed 1770420976
select  promotions,total,cast(promotions as
decimal(15,4))/cast(total as
decimal(15,4))*100
from
  (select sum(ss_ext_sales_price)
promotions
   from  store_sales
        ,store
        ,promotion
        ,date_dim
        ,customer
        ,customer_address
        ,item
   where ss_sold_date_sk = d_date_sk
   and   ss_store_sk = s_store_sk
   and   ss_promo_sk = p_promo_sk
   and   ss_customer_sk= c_customer_sk
   and   ca_address_sk = c_current_addr_sk
   and   ss_item_sk = i_item_sk
   and   ca_gmt_offset = -7
   and   i_category = 'Electronics'
   and   (p_channel_dmail = 'Y' or
p_channel_email = 'Y' or p_channel_tv =
'Y')
   and   s_gmt_offset = -7
   and   d_year = 1999
   and   d_moy  = 12) promotional_sales,
  (select sum(ss_ext_sales_price) total
   from  store_sales
        ,store
        ,date_dim
        ,customer
        ,customer_address
        ,item
   where ss_sold_date_sk = d_date_sk
   and   ss_store_sk = s_store_sk
   and   ss_customer_sk= c_customer_sk
   and   ca_address_sk = c_current_addr_sk
   and   ss_item_sk = i_item_sk
   and   ca_gmt_offset = -7
   and   i_category = 'Electronics'
   and   s_gmt_offset = -7
   and   d_year = 1999
   and   d_moy  = 12) all_sales
order by promotions, total
 fetch first 100 rows only;

-- end query 97 in stream 0 using template
query61.tpl

-- start query 99 in stream 0 using
template query76.tpl and seed 945056756
select  channel, col_name, d_year, d_qoy,
i_category, COUNT(*) sales_cnt,
SUM(ext_sales_price) sales_amt FROM (
        SELECT 'store' as channel,
'ss_hdemo_sk' col_name, d_year, d_qoy,
i_category, ss_ext_sales_price
ext_sales_price
         FROM store_sales, item, date_dim
         WHERE ss_hdemo_sk IS NULL
           AND ss_sold_date_sk=d_date_sk
           AND ss_item_sk=i_item_sk
        UNION ALL
        SELECT 'web' as channel,
'ws_web_page_sk' col_name, d_year, d_qoy,
i_category, ws_ext_sales_price
ext_sales_price
         FROM web_sales, item, date_dim
         WHERE ws_web_page_sk IS NULL
           AND ws_sold_date_sk=d_date_sk
           AND ws_item_sk=i_item_sk
        UNION ALL
        SELECT 'catalog' as channel,
'cs_ship_addr_sk' col_name, d_year, d_qoy,
i_category, cs_ext_sales_price
ext_sales_price
         FROM catalog_sales, item, date_dim
         WHERE cs_ship_addr_sk IS NULL
           AND cs_sold_date_sk=d_date_sk
           AND cs_item_sk=i_item_sk) foo
GROUP BY channel, col_name, d_year, d_qoy,
i_category
ORDER BY channel, col_name, d_year, d_qoy,
i_category
 fetch first 100 rows only;

-- end query 99 in stream 0 using template
query76.tpl
```

E.2 Impala Queries:

```
-- start query 1 in stream 0 using template
query96.tpl and seed 550831069
select  count(*)
from store_sales
    ,household_demographics
    ,time_dim, store
where ss_sold_time_sk = time_dim.t_time_sk
    and ss_hdemo_sk =
household_demographics.hd_demo_sk
    and ss_store_sk = s_store_sk
    and time_dim.t_hour = 15
    and time_dim.t_minute >= 30
    and household_demographics.hd_dep_count
= 6
    and store.s_store_name = 'ese'
order by count(*)
 limit 100;

-- end query 1 in stream 0 using template
query96.tpl
-- start query 2 in stream 0 using template
query7.tpl and seed 997258328
select  i_item_id,
        avg(ss_quantity) agg1,
        avg(ss_list_price) agg2,
        avg(ss_coupon_amt) agg3,
        avg(ss_sales_price) agg4
 from store_sales, customer_demographics,
date_dim, item, promotion
 where ss_sold_date_sk = d_date_sk and
```

```
       ss_item_sk = i_item_sk and
       ss_cdemo_sk = cd_demo_sk and
       ss_promo_sk = p_promo_sk and
       cd_gender = 'M' and
       cd_marital_status = 'W' and
       cd_education_status = '2 yr Degree'
and
       (p_channel_email = 'N' or
p_channel_event = 'N') and
       d_year = 1999
 group by i_item_id
 order by i_item_id
  limit 100;

-- end query 2 in stream 0 using template
query7.tpl
-- start query 5 in stream 0 using template
query39.tpl and seed 1420791654
with inv as
(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy
       ,stdev,mean, case mean when 0 then
null else stdev/mean end cov
 from(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy

,cast(stddev_samp(inv_quantity_on_hand) as
double)  stdev,avg(inv_quantity_on_hand)
mean
      from inventory
          ,item
          ,warehouse
          ,date_dim
      where inv_item_sk = i_item_sk
        and inv_warehouse_sk =
w_warehouse_sk
        and inv_date_sk = d_date_sk
        and d_year =2000
      group by
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy) foo
 where case mean when 0 then 0 else
stdev/mean end > 1)
select
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean, inv1.cov

,inv2.w_warehouse_sk,inv2.i_item_sk,inv2.d_
moy,inv2.mean, inv2.cov
from inv inv1,inv inv2
where inv1.i_item_sk = inv2.i_item_sk
  and inv1.w_warehouse_sk =
inv2.w_warehouse_sk
  and inv1.d_moy=2
  and inv2.d_moy=2+1
order by
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean,inv1.cov
       ,inv2.d_moy,inv2.mean, inv2.cov
;
with inv as
(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy
       ,stdev,mean, case mean when 0 then
null else stdev/mean end cov
 from(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy

,cast(stddev_samp(inv_quantity_on_hand) as
double) stdev,avg(inv_quantity_on_hand)
mean
      from inventory
          ,item
          ,warehouse
          ,date_dim
      where inv_item_sk = i_item_sk
        and inv_warehouse_sk =
w_warehouse_sk
        and inv_date_sk = d_date_sk
        and d_year =2000
      group by
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy) foo
 where case mean when 0 then 0 else
stdev/mean end > 1)
select
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean, inv1.cov

,inv2.w_warehouse_sk,inv2.i_item_sk,inv2.d_
moy,inv2.mean, inv2.cov
from inv inv1,inv inv2
where inv1.i_item_sk = inv2.i_item_sk
  and inv1.w_warehouse_sk =
inv2.w_warehouse_sk
  and inv1.d_moy=2
  and inv2.d_moy=2+1
  and inv1.cov > 1.5
order by
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean,inv1.cov
       ,inv2.d_moy,inv2.mean, inv2.cov
;

-- end query 5 in stream 0 using template
query39.tpl
-- start query 7 in stream 0 using template
query32.tpl and seed 944563352
select  sum(cs_ext_discount_amt) as "excess
discount amount"
from
   catalog_sales
   ,item
   ,date_dim
      ,(
         select
            cs_item_sk,
            1.3 * avg(cs_ext_discount_amt)
avg_cs_ext_discount_amt130
         from
            catalog_sales
           ,date_dim
         where
          d_date between '2000-01-16' and
                         (cast('2000-01-16'
as timestamp) + interval 90 days)
          and d_date_sk = cs_sold_date_sk
          group by cs_item_sk
      ) tmp1
where
i_manufact_id = 353
and i_item_sk = catalog_sales.cs_item_sk
and d_date between '2000-01-16' and
        (cast('2000-01-16' as timestamp) +
interval 90 days)
and d_date_sk = cs_sold_date_sk
and tmp1.cs_item_sk = i_item_sk
and cs_ext_discount_amt  >
avg_cs_ext_discount_amt130
 limit 100;

-- end query 7 in stream 0 using template
query32.tpl
-- start query 14 in stream 0 using
template query21.tpl and seed 614834996
select  *
 from(select w_warehouse_name
           ,i_item_id
           ,sum(case when (d_date < cast
('1998-06-27' as timestamp))
```

```
                        then
inv_quantity_on_hand
                      else 0 end) as
inv_before
            ,sum(case when (d_date >= cast
('1998-06-27' as timestamp))
                      then
inv_quantity_on_hand
                      else 0 end) as
inv_after
   from inventory
       ,warehouse
       ,item
       ,date_dim
   where i_current_price between 0.99 and
1.49
     and i_item_sk          = inv_item_sk
     and inv_warehouse_sk   =
w_warehouse_sk
     and inv_date_sk    = d_date_sk
     and d_date between (cast ('1998-06-27'
as timestamp) - interval 30 days)
                    and (cast ('1998-06-27'
as timestamp) + interval 30 days)
   group by w_warehouse_name, i_item_id) x
 where (case when inv_before > 0
             then inv_after / inv_before
             else null
             end) between 2.0/3.0 and
3.0/2.0
 order by w_warehouse_name
         ,i_item_id
  limit 100;

-- end query 14 in stream 0 using template
query21.tpl
-- start query 15 in stream 0 using
template query43.tpl and seed 959608359
select  s_store_name, s_store_id,
        sum(case when (d_day_name='Sunday')
then ss_sales_price else null end)
sun_sales,
        sum(case when (d_day_name='Monday')
then ss_sales_price else null end)
mon_sales,
        sum(case when
(d_day_name='Tuesday') then ss_sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then
ss_sales_price else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then ss_sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then ss_sales_price else null end)
fri_sales,
        sum(case when
(d_day_name='Saturday') then ss_sales_price
else null end) sat_sales
 from date_dim, store_sales, store
 where d_date_sk = ss_sold_date_sk and
       s_store_sk = ss_store_sk and
       s_gmt_offset = -8 and
       d_year = 1998
 group by s_store_name, s_store_id
 order by s_store_name,
s_store_id,sun_sales,mon_sales,tue_sales,we
d_sales,thu_sales,fri_sales,sat_sales
  limit 100;

-- end query 15 in stream 0 using template
query43.tpl
-- start query 16 in stream 0 using
template query27.tpl and seed 331218716
with results as
 (select i_item_id,
        s_state, 0 as g_state,
        avg(ss_quantity) agg1,
        avg(ss_list_price) agg2,
        avg(ss_coupon_amt) agg3,
        avg(ss_sales_price) agg4
 from store_sales, customer_demographics,
date_dim, store, item
 where ss_sold_date_sk = d_date_sk and
       ss_item_sk = i_item_sk and
       ss_store_sk = s_store_sk and
       ss_cdemo_sk = cd_demo_sk and
       cd_gender = 'F' and
       cd_marital_status = 'W' and
       cd_education_status = '4 yr Degree'
and
       d_year = 1999 and
       s_state in ('OH','IL', 'LA', 'GA',
'CO', 'AL')
 group by i_item_id, s_state)

  select  i_item_id,
  s_state, g_state, agg1, agg2, agg3, agg4
   from (
        select i_item_id, s_state, g_state,
agg1, agg2, agg3, agg4 from results
         union
        select i_item_id, NULL AS s_state,
1 AS g_state, avg(agg1) agg1, avg(agg2)
agg2, avg(agg3) agg3,
         avg(agg4) agg4 from results
        group by i_item_id
         union
        select NULL AS i_item_id, NULL as
s_state, 1 as g_state, avg(agg1) agg1,
avg(agg2) agg2, avg(agg3) agg3,
         avg(agg4) agg4 from results
        ) foo
  order by i_item_id, s_state
  limit 100;

-- end query 16 in stream 0 using template
query27.tpl
-- start query 19 in stream 0 using
template query58.tpl and seed 1844319395
with ss_items as
 (select i_item_id item_id
        ,sum(ss_ext_sales_price)
ss_item_rev
 from store_sales
     ,item
     ,date_dim
     JOIN (select d1.d_date
           from date_dim d1
               JOIN date_dim d2
                   ON d1.d_week_seq =
d2.d_week_seq
           where d2.d_date = '1998-05-29')
sub
         ON date_dim.d_date = sub.d_date
 where ss_item_sk = i_item_sk
   and ss_sold_date_sk   = d_date_sk
 group by i_item_id),
 cs_items as
 (select i_item_id item_id
        ,sum(cs_ext_sales_price)
cs_item_rev
  from catalog_sales
      ,item
      ,date_dim
     JOIN (select d1.d_date
           from date_dim d1
               JOIN date_dim d2
                   ON d1.d_week_seq =
d2.d_week_seq
           where d2.d_date = '1998-05-29')
sub
         ON date_dim.d_date = sub.d_date
```

```
 where cs_item_sk = i_item_sk
  and  cs_sold_date_sk = d_date_sk
 group by i_item_id),
 ws_items as
 (select i_item_id item_id
        ,sum(ws_ext_sales_price)
ws_item_rev
  from web_sales
      ,item
      ,date_dim
     JOIN (select d1.d_date
           from date_dim d1
               JOIN date_dim d2
                   ON d1.d_week_seq =
d2.d_week_seq
           where d2.d_date = '1998-05-29')
sub
         ON date_dim.d_date = sub.d_date
 where ws_item_sk = i_item_sk
  and ws_sold_date_sk   = d_date_sk
 group by i_item_id)
  select  ss_items.item_id
       ,ss_item_rev

,ss_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 ss_dev
       ,cs_item_rev

,cs_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 cs_dev
       ,ws_item_rev

,ws_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 ws_dev

,(ss_item_rev+cs_item_rev+ws_item_rev)/3
average
 from ss_items,cs_items,ws_items
 where ss_items.item_id=cs_items.item_id
   and ss_items.item_id=ws_items.item_id
   and ss_item_rev between 0.9 *
cs_item_rev and 1.1 * cs_item_rev
   and ss_item_rev between 0.9 *
ws_item_rev and 1.1 * ws_item_rev
   and cs_item_rev between 0.9 *
ss_item_rev and 1.1 * ss_item_rev
   and cs_item_rev between 0.9 *
ws_item_rev and 1.1 * ws_item_rev
   and ws_item_rev between 0.9 *
ss_item_rev and 1.1 * ss_item_rev
   and ws_item_rev between 0.9 *
cs_item_rev and 1.1 * cs_item_rev
 order by item_id
        ,ss_item_rev
  limit 100;

-- end query 19 in stream 0 using template
query58.tpl
-- start query 22 in stream 0 using
template query33.tpl and seed 248487088
with ss as (
 select

item.i_manufact_id,sum(ss_ext_sales_price)
total_sales
 from
       store_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select
i_manufact_id
                        from item
                        where i_category in
('Books')
        ) ssi on item.i_manufact_id =
ssi.i_manufact_id
 where   ss_item_sk              =
i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -7
 group by item.i_manufact_id),
 cs as (
 select

item.i_manufact_id,sum(cs_ext_sales_price)
total_sales
 from
       catalog_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select
i_manufact_id
                        from item
                        where i_category in
('Books')
        ) csi on item.i_manufact_id =
csi.i_manufact_id
 where   cs_item_sk              =
i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -7
 group by item.i_manufact_id),
 ws as (
 select

item.i_manufact_id,sum(ws_ext_sales_price)
total_sales
 from
       web_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select
i_manufact_id
                        from item
                        where i_category in
('Books')
        ) wsi on item.i_manufact_id =
wsi.i_manufact_id
 where   ws_item_sk              =
i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -7
 group by item.i_manufact_id)
  select  i_manufact_id ,sum(total_sales)
total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_manufact_id
 order by total_sales
```

```
 limit 100;

-- end query 22 in stream 0 using template
query33.tpl
-- start query 24 in stream 0 using
template query62.tpl and seed 800775315
select
   substr(w_warehouse_name,1,20)
  ,sm_type
  ,web_name
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk <= 30 ) then 1 else 0 end)
as "30 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 30) and
                (ws_ship_date_sk -
ws_sold_date_sk <= 60) then 1 else 0 end )
as "31-60 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 60) and
                (ws_ship_date_sk -
ws_sold_date_sk <= 90) then 1 else 0 end)
as "61-90 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 90) and
                (ws_ship_date_sk -
ws_sold_date_sk <= 120) then 1 else 0 end)
as "91-120 days"
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk  > 120) then 1 else 0 end)
as ">120 days"
from
   web_sales
  ,warehouse
  ,ship_mode
  ,web_site
  ,date_dim
where
    d_month_seq between 1201 and 1201 + 11
and ws_ship_date_sk   = d_date_sk
and ws_warehouse_sk   = w_warehouse_sk
and ws_ship_mode_sk   = sm_ship_mode_sk
and ws_web_site_sk    = web_site_sk
group by
   substr(w_warehouse_name,1,20)
  ,sm_type
  ,web_name
order by substr(w_warehouse_name,1,20)
        ,sm_type
       ,web_name
 limit 100;

-- end query 24 in stream 0 using template
query62.tpl
-- start query 27 in stream 0 using
template query63.tpl and seed 812633773
with cte1 as
(select i_manager_id
            ,sum(ss_sales_price) sum_sales
      from item
          ,store_sales
          ,date_dim
          ,store
      where ss_item_sk = i_item_sk
        and ss_sold_date_sk = d_date_sk
        and ss_store_sk = s_store_sk
        and d_month_seq in
(1178,1178+1,1178+2,1178+3,1178+4,1178+5,11
78+6,1178+7,1178+8,1178+9,1178+10,1178+11)
        and ((    i_category in
('Books','Children','Electronics')
              and i_class in
('personal','portable','refernece','self-
help')
              and i_brand in
('scholaramalgamalg #14','scholaramalgamalg
#7',
'exportiunivamalg #9','scholaramalgamalg
#9'))
          or(    i_category in
('Women','Music','Men')
              and i_class in
('accessories','classical','fragrances','pa
nts')
              and i_brand in ('amalgimporto
#1','edu packscholar #1','exportiimporto
#1',

'importoamalg #1')))

group by i_manager_id, d_moy),

cte2 as
(
select
  i_manager_id,
  avg(sum_sales) avg_monthly_sales
from cte1
  group by
  i_manager_id)

 select
  cte1.i_manager_id
  ,cte1.sum_sales
  ,cte2.avg_monthly_sales
from cte1
     cross join cte2
where
  (cte1.i_manager_id = cte2.i_manager_id
  or
  (cte1.i_manager_id is NULL and
cte2.i_manager_id is NULL))
  and case when avg_monthly_sales > 0 then
abs (sum_sales - avg_monthly_sales) /
avg_monthly_sales else null end > 0.1
order by cte1.i_manager_id
        ,avg_monthly_sales
        ,sum_sales
 limit 100;

-- end query 27 in stream 0 using template
query63.tpl
-- start query 28 in stream 0 using
template query69.tpl and seed 1390437346
select
  cd_gender,
  cd_marital_status,
  cd_education_status,
  count(*) cnt1,
  cd_purchase_estimate,
  count(*) cnt2,
  cd_credit_rating,
  count(*) cnt3
 from
  customer c,customer_address
ca,customer_demographics
  left semi join
         (select ss_customer_sk as
customer_sk
          from store_sales,date_dim
          where store_sales.ss_sold_date_sk
= date_dim.d_date_sk
          and d_year = 2004 and
                d_moy between 3 and 3+2) ss
      on c.c_customer_sk = ss.customer_sk
  left outer join
           (select ws_bill_customer_sk as
customer_sk
            from web_sales,date_dim
            where web_sales.ws_sold_date_sk
= date_dim.d_date_sk
            and d_year = 2004 and
```

```
                d_moy between 3 and 3+2)
ws
      on c.c_customer_sk = ws.customer_sk
  left outer join
           (select cs_ship_customer_sk as
customer_sk
            from catalog_sales,date_dim
            where
catalog_sales.cs_sold_date_sk =
date_dim.d_date_sk
            and d_year = 2004 and
                  d_moy between 3 and 3+2)
cs
      on c.c_customer_sk = cs.customer_sk
 where
  c.c_current_addr_sk = ca.ca_address_sk
and
  ca_state in ('AL','VA','GA') and
  cd_demo_sk = c.c_current_cdemo_sk and
  ws.customer_sk is NULL and
  cs.customer_sk is NULL
 group by cd_gender,
          cd_marital_status,
          cd_education_status,
          cd_purchase_estimate,
          cd_credit_rating
 order by cd_gender,
          cd_marital_status,
          cd_education_status,
          cd_purchase_estimate,
          cd_credit_rating
  limit 100;

-- end query 28 in stream 0 using template
query69.tpl
-- start query 29 in stream 0 using
template query60.tpl and seed 374071684
with ss as (
 select

item.i_item_id,sum(ss_ext_sales_price)
total_sales
 from
       store_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select i_item_id
                        from item
                         where i_category
in ('Jewelry')
       ) iss on item.i_item_id =
iss.i_item_id
 where   ss_item_sk              =
i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 cs as (
 select

item.i_item_id,sum(cs_ext_sales_price)
total_sales
 from
       catalog_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select i_item_id
                        from item
                           where i_category
in ('Jewelry')
       ) ics on item.i_item_id =
ics.i_item_id
 where   cs_item_sk              =
i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 ws as (
 select

item.i_item_id,sum(ws_ext_sales_price)
total_sales
 from
       web_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select i_item_id
                        from item
                        where i_category in
('Jewelry')
       ) iws on item.i_item_id =
iws.i_item_id
 where   ws_item_sk              =
i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id)
  select
  i_item_id
,sum(total_sales) total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_item_id
 order by i_item_id
      ,total_sales
  limit 100;

-- end query 29 in stream 0 using template
query60.tpl
-- start query 30 in stream 0 using
template query59.tpl and seed 1976435349
with wss as
 (select d_week_seq,
        ss_store_sk,
        sum(case when (d_day_name='Sunday')
then ss_sales_price else null end)
sun_sales,
        sum(case when (d_day_name='Monday')
then ss_sales_price else null end)
mon_sales,
        sum(case when
(d_day_name='Tuesday') then ss_sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then
ss_sales_price else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then ss_sales_price
else null end) thu_sales,
```

```
        sum(case when (d_day_name='Friday')
then ss_sales_price else null end)
fri_sales,
        sum(case when
(d_day_name='Saturday') then ss_sales_price
else null end) sat_sales
 from store_sales,date_dim
 where d_date_sk = ss_sold_date_sk
 group by d_week_seq,ss_store_sk
 )
  select
s_store_name1,s_store_id1,d_week_seq1

,sun_sales1/sun_sales2,mon_sales1/mon_sales
2

,tue_sales1/tue_sales1,wed_sales1/wed_sales
2,thu_sales1/thu_sales2

,fri_sales1/fri_sales2,sat_sales1/sat_sales
2
 from
 (select s_store_name
s_store_name1,wss.d_week_seq d_week_seq1
        ,s_store_id s_store_id1,sun_sales
sun_sales1
        ,mon_sales mon_sales1,tue_sales
tue_sales1
        ,wed_sales wed_sales1,thu_sales
thu_sales1
        ,fri_sales fri_sales1,sat_sales
sat_sales1
  from wss,store,date_dim d
  where d.d_week_seq = wss.d_week_seq and
        ss_store_sk = s_store_sk and
        d_month_seq between 1189 and 1189 +
11) y,
 (select s_store_name
s_store_name2,wss.d_week_seq d_week_seq2
        ,s_store_id s_store_id2,sun_sales
sun_sales2
        ,mon_sales mon_sales2,tue_sales
tue_sales2
        ,wed_sales wed_sales2,thu_sales
thu_sales2
        ,fri_sales fri_sales2,sat_sales
sat_sales2
  from wss,store,date_dim d
  where d.d_week_seq = wss.d_week_seq and
        ss_store_sk = s_store_sk and
        d_month_seq between 1189+ 12 and
1189 + 23) x
 where s_store_id1=s_store_id2
   and d_week_seq1=d_week_seq2-52
 order by
s_store_name1,s_store_id1,d_week_seq1
 limit 100;

-- end query 30 in stream 0 using template
query59.tpl
-- start query 32 in stream 0 using
template query98.tpl and seed 1900673199
with cte1 as
( select
    i_item_id
    ,i_item_desc
    ,i_category
    ,i_class
    ,i_current_price
    ,sum(ss_ext_sales_price) as itemrevenue
    from store_sales
         ,item
         ,date_dim
    where
        ss_item_sk = i_item_sk
       and i_category in ('Shoes',
'Music', 'Home')
        and ss_sold_date_sk = d_date_sk
       and d_date between cast('1999-05-
25' as timestamp)
                and (cast('1999-05-25' as
timestamp) + interval 30 days)
    group by
        i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price ),
cte2 as
( select
    i_class
    ,sum(itemrevenue) as sumitemrevenue
  from cte1
  group by
    i_class)
select
   i_item_desc
   ,i_category
   ,i_class
   ,i_current_price
   ,itemrevenue
   ,revenueratio
from (
select
   cte1.i_item_id
   ,cte1.i_item_desc
   ,cte1.i_category
   ,cte1.i_class
   ,cte1.i_current_price
   ,cte1.itemrevenue

,cte1.itemrevenue*100/cte2.sumitemrevenue
as revenueratio
from cte1
     cross join cte2
where
   cte1.i_class = cte2.i_class or
(cte1.i_class is NULL and cte2.i_class is
NULL)
) v1
order by
   i_category
   ,i_class
   ,i_item_id
   ,i_item_desc
   ,revenueratio
;

-- end query 32 in stream 0 using template
query98.tpl
-- start query 36 in stream 0 using
template query28.tpl and seed 24799953
select  *
from (select avg(ss_list_price) B1_LP
            ,count(ss_list_price) B1_CNT
            ,count(distinct ss_list_price)
B1_CNTD
      from store_sales
      where ss_quantity between 0 and 5
        and (ss_list_price between 47 and
47+10
             or ss_coupon_amt between 11713
and 11713+1000
             or ss_wholesale_cost between
55 and 55+20)) B1
cross join
     (select avg(ss_list_price) B2_LP
            ,count(ss_list_price) B2_CNT
            ,count(distinct ss_list_price)
B2_CNTD
      from store_sales
      where ss_quantity between 6 and 10
        and (ss_list_price between 93 and
93+10
```

```
          or ss_coupon_amt between 7733 and
7733+1000
          or ss_wholesale_cost between 43
and 43+20)) B2
cross join
     (select avg(ss_list_price) B3_LP
            ,count(ss_list_price) B3_CNT
            ,count(distinct ss_list_price)
B3_CNTD
      from store_sales
      where ss_quantity between 11 and 15
        and (ss_list_price between 32 and
32+10
          or ss_coupon_amt between 11517
and 11517+1000
          or ss_wholesale_cost between 26
and 26+20)) B3
cross join
     (select avg(ss_list_price) B4_LP
            ,count(ss_list_price) B4_CNT
            ,count(distinct ss_list_price)
B4_CNTD
      from store_sales
      where ss_quantity between 16 and 20
        and (ss_list_price between 147 and
147+10
          or ss_coupon_amt between 509 and
509+1000
          or ss_wholesale_cost between 78
and 78+20)) B4
cross join
     (select avg(ss_list_price) B5_LP
            ,count(ss_list_price) B5_CNT
            ,count(distinct ss_list_price)
B5_CNTD
      from store_sales
      where ss_quantity between 21 and 25
        and (ss_list_price between 16 and
16+10
          or ss_coupon_amt between 2401 and
2401+1000
          or ss_wholesale_cost between 32
and 32+20)) B5
cross join
     (select avg(ss_list_price) B6_LP
            ,count(ss_list_price) B6_CNT
            ,count(distinct ss_list_price)
B6_CNTD
      from store_sales
      where ss_quantity between 26 and 30
        and (ss_list_price between 11 and
11+10
          or ss_coupon_amt between 916 and
916+1000
          or ss_wholesale_cost between 6
and 6+20)) B6
 limit 100;

-- end query 36 in stream 0 using template
query28.tpl
-- start query 39 in stream 0 using
template query66.tpl and seed 1688498284
select
         w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,ship_carriers
       ,year
       ,sum(jan_sales) as jan_sales
       ,sum(feb_sales) as feb_sales
       ,sum(mar_sales) as mar_sales
       ,sum(apr_sales) as apr_sales
       ,sum(may_sales) as may_sales
       ,sum(jun_sales) as jun_sales
       ,sum(jul_sales) as jul_sales
       ,sum(aug_sales) as aug_sales
       ,sum(sep_sales) as sep_sales
       ,sum(oct_sales) as oct_sales
       ,sum(nov_sales) as nov_sales
       ,sum(dec_sales) as dec_sales
       ,sum(jan_sales/w_warehouse_sq_ft)
as jan_sales_per_sq_foot
       ,sum(feb_sales/w_warehouse_sq_ft)
as feb_sales_per_sq_foot
       ,sum(mar_sales/w_warehouse_sq_ft)
as mar_sales_per_sq_foot
       ,sum(apr_sales/w_warehouse_sq_ft)
as apr_sales_per_sq_foot
       ,sum(may_sales/w_warehouse_sq_ft)
as may_sales_per_sq_foot
       ,sum(jun_sales/w_warehouse_sq_ft)
as jun_sales_per_sq_foot
       ,sum(jul_sales/w_warehouse_sq_ft)
as jul_sales_per_sq_foot
       ,sum(aug_sales/w_warehouse_sq_ft)
as aug_sales_per_sq_foot
       ,sum(sep_sales/w_warehouse_sq_ft)
as sep_sales_per_sq_foot
       ,sum(oct_sales/w_warehouse_sq_ft)
as oct_sales_per_sq_foot
       ,sum(nov_sales/w_warehouse_sq_ft)
as nov_sales_per_sq_foot
       ,sum(dec_sales/w_warehouse_sq_ft)
as dec_sales_per_sq_foot
       ,sum(jan_net) as jan_net
       ,sum(feb_net) as feb_net
       ,sum(mar_net) as mar_net
       ,sum(apr_net) as apr_net
       ,sum(may_net) as may_net
       ,sum(jun_net) as jun_net
       ,sum(jul_net) as jul_net
       ,sum(aug_net) as aug_net
       ,sum(sep_net) as sep_net
       ,sum(oct_net) as oct_net
       ,sum(nov_net) as nov_net
       ,sum(dec_net) as dec_net
 from (
    (select
       w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
        ,concat('MSC', ',', 'USPS') as
ship_carriers
      ,d_year as year
       ,sum(case when d_moy = 1
                 then ws_sales_price*
ws_quantity else 0 end) as jan_sales
       ,sum(case when d_moy = 2
                 then ws_sales_price*
ws_quantity else 0 end) as feb_sales
       ,sum(case when d_moy = 3
                 then ws_sales_price*
ws_quantity else 0 end) as mar_sales
       ,sum(case when d_moy = 4
                 then ws_sales_price*
ws_quantity else 0 end) as apr_sales
       ,sum(case when d_moy = 5
                 then ws_sales_price*
ws_quantity else 0 end) as may_sales
       ,sum(case when d_moy = 6
                 then ws_sales_price*
ws_quantity else 0 end) as jun_sales
       ,sum(case when d_moy = 7
                 then ws_sales_price*
ws_quantity else 0 end) as jul_sales
       ,sum(case when d_moy = 8
                 then ws_sales_price*
ws_quantity else 0 end) as aug_sales
```

```
       ,sum(case when d_moy = 9
                 then ws_sales_price*
ws_quantity else 0 end) as sep_sales
       ,sum(case when d_moy = 10
                 then ws_sales_price*
ws_quantity else 0 end) as oct_sales
       ,sum(case when d_moy = 11
                 then ws_sales_price*
ws_quantity else 0 end) as nov_sales
       ,sum(case when d_moy = 12
                 then ws_sales_price*
ws_quantity else 0 end) as dec_sales
       ,sum(case when d_moy = 1
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jan_net
       ,sum(case when d_moy = 2
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as feb_net
       ,sum(case when d_moy = 3
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as mar_net
       ,sum(case when d_moy = 4
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as apr_net
       ,sum(case when d_moy = 5
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as may_net
       ,sum(case when d_moy = 6
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jun_net
       ,sum(case when d_moy = 7
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jul_net
       ,sum(case when d_moy = 8
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as aug_net
       ,sum(case when d_moy = 9
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as sep_net
       ,sum(case when d_moy = 10
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as oct_net
       ,sum(case when d_moy = 11
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as nov_net
       ,sum(case when d_moy = 12
                 then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as dec_net
     from
          web_sales
         ,warehouse
         ,date_dim
         ,time_dim
          ,ship_mode
     where
            ws_warehouse_sk =
w_warehouse_sk
        and ws_sold_date_sk = d_date_sk
        and ws_sold_time_sk = t_time_sk
       and ws_ship_mode_sk =
sm_ship_mode_sk
        and d_year = 2002
        and t_time between 18036 and
18036+28800
        and sm_carrier in ('MSC','USPS')
      group by
         w_warehouse_name
        ,w_warehouse_sq_ft
        ,w_city
        ,w_county
        ,w_state
        ,w_country
       ,d_year
        )
 union all
    (select
        w_warehouse_name
        ,w_warehouse_sq_ft
        ,w_city
        ,w_county
        ,w_state
        ,w_country
         ,concat('MSC', ',', 'USPS') as
ship_carriers
       ,d_year as year
       ,sum(case when d_moy = 1
                 then cs_ext_sales_price*
cs_quantity else 0 end) as jan_sales
       ,sum(case when d_moy = 2
                 then cs_ext_sales_price*
cs_quantity else 0 end) as feb_sales
       ,sum(case when d_moy = 3
                 then cs_ext_sales_price*
cs_quantity else 0 end) as mar_sales
       ,sum(case when d_moy = 4
                 then cs_ext_sales_price*
cs_quantity else 0 end) as apr_sales
       ,sum(case when d_moy = 5
                 then cs_ext_sales_price*
cs_quantity else 0 end) as may_sales
       ,sum(case when d_moy = 6
                 then cs_ext_sales_price*
cs_quantity else 0 end) as jun_sales
       ,sum(case when d_moy = 7
                 then cs_ext_sales_price*
cs_quantity else 0 end) as jul_sales
       ,sum(case when d_moy = 8
                 then cs_ext_sales_price*
cs_quantity else 0 end) as aug_sales
       ,sum(case when d_moy = 9
                 then cs_ext_sales_price*
cs_quantity else 0 end) as sep_sales
       ,sum(case when d_moy = 10
                 then cs_ext_sales_price*
cs_quantity else 0 end) as oct_sales
       ,sum(case when d_moy = 11
                 then cs_ext_sales_price*
cs_quantity else 0 end) as nov_sales
       ,sum(case when d_moy = 12
                 then cs_ext_sales_price*
cs_quantity else 0 end) as dec_sales
       ,sum(case when d_moy = 1
                 then cs_net_profit *
cs_quantity else 0 end) as jan_net
       ,sum(case when d_moy = 2
                 then cs_net_profit *
cs_quantity else 0 end) as feb_net
       ,sum(case when d_moy = 3
                 then cs_net_profit *
cs_quantity else 0 end) as mar_net
       ,sum(case when d_moy = 4
                 then cs_net_profit *
cs_quantity else 0 end) as apr_net
       ,sum(case when d_moy = 5
                 then cs_net_profit *
cs_quantity else 0 end) as may_net
       ,sum(case when d_moy = 6
                 then cs_net_profit *
cs_quantity else 0 end) as jun_net
       ,sum(case when d_moy = 7
                 then cs_net_profit *
cs_quantity else 0 end) as jul_net
```

```
       ,sum(case when d_moy = 8
                 then cs_net_profit *
cs_quantity else 0 end) as aug_net
       ,sum(case when d_moy = 9
                 then cs_net_profit *
cs_quantity else 0 end) as sep_net
       ,sum(case when d_moy = 10
                 then cs_net_profit *
cs_quantity else 0 end) as oct_net
       ,sum(case when d_moy = 11
                 then cs_net_profit *
cs_quantity else 0 end) as nov_net
       ,sum(case when d_moy = 12
                 then cs_net_profit *
cs_quantity else 0 end) as dec_net
     from
          catalog_sales
         ,warehouse
         ,date_dim
         ,time_dim
         ,ship_mode
     where
            cs_warehouse_sk =
w_warehouse_sk
        and cs_sold_date_sk = d_date_sk
        and cs_sold_time_sk = t_time_sk
        and cs_ship_mode_sk =
sm_ship_mode_sk
        and d_year = 2002
        and t_time between 18036 and
18036+28800
        and sm_carrier in ('MSC','USPS')
     group by
        w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,d_year
     )
 ) x
 group by
        w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,ship_carriers
       ,year
 order by w_warehouse_name
  limit 100;

-- end query 39 in stream 0 using template
query66.tpl
-- start query 40 in stream 0 using
template query90.tpl and seed 1949014749
select  cast(amc as decimal(15,4))/cast(pmc
as decimal(15,4)) am_pm_ratio
 from ( select count(*) amc
       from web_sales,
household_demographics , time_dim, web_page
       where ws_sold_time_sk =
time_dim.t_time_sk
         and ws_ship_hdemo_sk =
household_demographics.hd_demo_sk
         and ws_web_page_sk =
web_page.wp_web_page_sk
         and time_dim.t_hour between 11 and
11+1
         and
household_demographics.hd_dep_count = 9
         and web_page.wp_char_count between
5000 and 5200) at
cross join
      ( select count(*) pmc
       from web_sales,
household_demographics , time_dim, web_page
       where ws_sold_time_sk =
time_dim.t_time_sk
         and ws_ship_hdemo_sk =
household_demographics.hd_demo_sk
         and ws_web_page_sk =
web_page.wp_web_page_sk
         and time_dim.t_hour between 18 and
18+1
         and
household_demographics.hd_dep_count = 9
         and web_page.wp_char_count between
5000 and 5200) pt
 order by am_pm_ratio
  limit 100;

-- end query 40 in stream 0 using template
query90.tpl
-- start query 44 in stream 0 using
template query92.tpl and seed 643980925
select
   sum(ws_ext_discount_amt)  as "Excess
Discount Amount"
from
    web_sales
   ,item
   ,date_dim
   join(
         SELECT
            ws_item_sk, 1.3 *
avg(ws_ext_discount_amt)
avg_ws_ext_discount_amt_130
         FROM
            web_sales
           ,date_dim
         WHERE
              d_date between '1999-01-03'
and
                                (cast('1999-
01-03' as timestamp) + interval 90 days)
          and d_date_sk = ws_sold_date_sk
         GROUP BY ws_item_sk
      ) wsd on item.i_item_sk =
wsd.ws_item_sk
where
i_manufact_id = 926
and i_item_sk = web_sales.ws_item_sk
and d_date between '1999-01-03' and
        (cast('1999-01-03' as timestamp) +
interval 90 days)
and d_date_sk = ws_sold_date_sk
and ws_ext_discount_amt >
avg_ws_ext_discount_amt_130
order by sum(ws_ext_discount_amt)
 limit 100;

-- end query 44 in stream 0 using template
query92.tpl
-- start query 45 in stream 0 using
template query3.tpl and seed 691662667
select  dt.d_year
       ,item.i_brand_id brand_id
       ,item.i_brand brand
       ,sum(ss_net_profit) sum_agg
 from  date_dim dt
      ,store_sales
      ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
   and store_sales.ss_item_sk =
item.i_item_sk
   and item.i_manufact_id = 596
   and dt.d_moy=12
 group by dt.d_year
      ,item.i_brand
      ,item.i_brand_id
```

```
 order by dt.d_year
         ,sum_agg desc
         ,brand_id
  limit 100;

-- end query 45 in stream 0 using template
query3.tpl
-- start query 49 in stream 0 using
template query9.tpl and seed 937436805
select bucket1, bucket2, bucket3, bucket4,
bucket5
 from

(select case when count1 > 62316685 then
then1 else else1 end bucket1
from (
select count(*) count1,
avg(ss_ext_sales_price) then1,
avg(ss_net_paid_inc_tax) else1
from store_sales
where ss_quantity between 1 and 20
) A1) B1

CROSS JOIN

(select case when count2 > 19045798 then
then2 else else2 end bucket2
from (
select count(*) count2,
avg(ss_ext_sales_price) then2,
avg(ss_net_paid_inc_tax) else2
from store_sales
where ss_quantity between 21 and 40
) A2) B2

CROSS JOIN

(select case when count3 > 365541424 then
then3 else else3 end bucket3
from (
select count(*) count3,
avg(ss_ext_sales_price) then3,
avg(ss_net_paid_inc_tax) else3
from store_sales
where ss_quantity between 41 and 60
) A3) B3

CROSS JOIN

(select case when count4 > 216357808 then
then4 else else4 end bucket4
from (
select count(*) count4,
avg(ss_ext_sales_price) then4,
avg(ss_net_paid_inc_tax) else4
from store_sales
where ss_quantity between 61 and 80
) A4) B4

CROSS JOIN
(select case when count5 > 184483884 then
then5 else else5 end bucket5
from (
select count(*) count5,
avg(ss_ext_sales_price) then5,
avg(ss_net_paid_inc_tax) else5
from store_sales
where ss_quantity between 81 and 100
) A5) B5

CROSS JOIN

reason
where r_reason_sk = 1
;

-- end query 49 in stream 0 using template
query9.tpl
-- start query 52 in stream 0 using
template query93.tpl and seed 1821797098
select  ss_customer_sk
            ,sum(act_sales) sumsales
      from (select ss_item_sk
                  ,ss_ticket_number
                  ,ss_customer_sk
                  ,case when
sr_return_quantity is not null then
(ss_quantity-
sr_return_quantity)*ss_sales_price

else (ss_quantity*ss_sales_price) end
act_sales
            from store_sales left outer
join store_returns on (sr_item_sk =
ss_item_sk

and sr_ticket_number = ss_ticket_number)
               ,reason
            where sr_reason_sk =
r_reason_sk
              and r_reason_desc = 'reason
74') t
      group by ss_customer_sk
      order by sumsales, ss_customer_sk
 limit 100;

-- end query 52 in stream 0 using template
query93.tpl
-- start query 55 in stream 0 using
template query22.tpl and seed 635815297
with results as
(select  i_product_name
             ,i_brand
             ,i_class
             ,i_category
             ,avg(inv_quantity_on_hand) qoh
       from inventory
           ,date_dim
           ,item
           ,warehouse
       where  inv_date_sk=d_date_sk
              and inv_item_sk=i_item_sk
              and inv_warehouse_sk =
w_warehouse_sk
              and d_month_seq between 1199
and 1199 + 11
       group by
i_product_name,i_brand,i_class,i_category),
results_rollup as
(
 select i_product_name, i_brand, i_class,
i_category,qoh from results

 union all

 select i_product_name, i_brand, i_class,
null i_category,sum(qoh) from results
 group by i_product_name,i_brand,i_class

 union all

 select i_product_name, i_brand, null
i_class, null i_category,sum(qoh) from
results
 group by i_product_name,i_brand

 union all

 select i_product_name, null i_brand, null
i_class, null i_category,sum(qoh) from
results
 group by i_product_name
```

```
 union all

 select null i_product_name, null i_brand,
null i_class, null i_category,sum(qoh) from
results
)
 select  i_product_name, i_brand, i_class,
i_category,qoh
      from results_rollup
      order by qoh, i_product_name,
i_brand, i_class, i_category
 limit 100;

-- end query 55 in stream 0 using template
query22.tpl
-- start query 56 in stream 0 using
template query89.tpl and seed 2079706651
with cte1 as
(
select i_category, i_class, i_brand,
       s_store_name, s_company_name,
       d_moy,
       sum(ss_sales_price) sum_sales
from item, store_sales, date_dim, store
where ss_item_sk = i_item_sk and
      ss_sold_date_sk = d_date_sk and
      ss_store_sk = s_store_sk and
      d_year in (1999) and
        ((i_category in
('Books','Jewelry','Men') and
          i_class in
('history','birdal','pants')
         )
      or (i_category in
('Music','Home','Shoes') and
          i_class in
('pop','furniture','athletic')
        ))
group by i_category, i_class, i_brand,
         s_store_name, s_company_name,
d_moy),

cte2 as
(
select
i_category, i_brand, s_store_name,
s_company_name,
avg(sum_sales) avg_monthly_sales
from cte1
group by
i_category, i_brand, s_store_name,
s_company_name
)
 select  cte1.i_category, cte1.i_class,
cte1.i_brand,
       cte1.s_store_name,
cte1.s_company_name, cte1.d_moy,
       cte1.sum_sales,
cte2.avg_monthly_sales
from cte1 cross join cte2
where
       (cte1.i_category = cte2.i_category
       or (cte1.i_category is NULL and
cte2.i_category is NULL))

       and (cte1.i_brand = cte2.i_brand
       or (cte1.i_brand is NULL and
cte2.i_brand is NULL))

       and (cte1.s_store_name =
cte2.s_store_name
       or (cte1.s_store_name is NULL and
cte2.s_store_name is NULL))
       and (cte1.s_company_name =
cte2.s_company_name
       or (cte1.s_company_name is NULL and
cte2.s_company_name is NULL))
       --and cte1.d_moy = cte2.d_moy
       and case when (avg_monthly_sales <>
0) then (abs(sum_sales - avg_monthly_sales)
/ avg_monthly_sales) else null end > 0.1
order by sum_sales - avg_monthly_sales,
cte1.s_store_name
 limit 100;

-- end query 56 in stream 0 using template
query89.tpl
-- start query 59 in stream 0 using
template query52.tpl and seed 223505300
select  dt.d_year
       ,item.i_brand_id brand_id
       ,item.i_brand brand
       ,sum(ss_ext_sales_price) ext_price
 from date_dim dt
     ,store_sales
     ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
    and store_sales.ss_item_sk =
item.i_item_sk
    and item.i_manager_id = 1
    and dt.d_moy=11
    and dt.d_year=1999
 group by dt.d_year
       ,item.i_brand
       ,item.i_brand_id
 order by dt.d_year
       ,ext_price desc
       ,brand_id
 limit 100 ;

-- end query 59 in stream 0 using template
query52.tpl
-- start query 60 in stream 0 using
template query50.tpl and seed 1718577076
select
   s_store_name
  ,s_company_id
  ,s_street_number
  ,s_street_name
  ,s_street_type
  ,s_suite_number
  ,s_city
  ,s_county
  ,s_state
  ,s_zip
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk <= 30 ) then 1 else 0 end)
as "30 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 30) and
                (sr_returned_date_sk -
ss_sold_date_sk <= 60) then 1 else 0 end )
as "31-60 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 60) and
                (sr_returned_date_sk -
ss_sold_date_sk <= 90) then 1 else 0 end)
as "61-90 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 90) and
                (sr_returned_date_sk -
ss_sold_date_sk <= 120) then 1 else 0 end)
as "91-120 days"
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk  > 120) then 1 else 0 end)
as ">120 days"
from
   store_sales
  ,store_returns
  ,store
  ,date_dim d1
  ,date_dim d2
where
```

```
    d2.d_year = 1999
and d2.d_moy  = 10
and ss_ticket_number = sr_ticket_number
and ss_item_sk = sr_item_sk
and ss_sold_date_sk   = d1.d_date_sk
and sr_returned_date_sk   = d2.d_date_sk
and ss_customer_sk = sr_customer_sk
and ss_store_sk = s_store_sk
group by
   s_store_name
  ,s_company_id
  ,s_street_number
  ,s_street_name
  ,s_street_type
  ,s_suite_number
  ,s_city
  ,s_county
  ,s_state
  ,s_zip
order by s_store_name
        ,s_company_id
        ,s_street_number
        ,s_street_name
        ,s_street_type
        ,s_suite_number
        ,s_city
        ,s_county
        ,s_state
        ,s_zip
 limit 100;

-- end query 60 in stream 0 using template
query50.tpl
-- start query 61 in stream 0 using
template query42.tpl and seed 709936855
select  dt.d_year
       ,item.i_category_id
       ,item.i_category
       ,sum(ss_ext_sales_price)
 from   date_dim dt
       ,store_sales
       ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
       and store_sales.ss_item_sk =
item.i_item_sk
       and item.i_manager_id = 1
       and dt.d_moy=12
       and dt.d_year=2000
 group by      dt.d_year
               ,item.i_category_id
               ,item.i_category
 order by       sum(ss_ext_sales_price)
desc,dt.d_year
               ,item.i_category_id
               ,item.i_category
 limit 100 ;

-- end query 61 in stream 0 using template
query42.tpl
-- start query 62 in stream 0 using
template query41.tpl and seed 944250029
select  distinct(i_product_name)
 from item i1
 JOIN  (select i_manufact, count(*) as
item_cnt
        from item
        where (
        ((i_category = 'Women' and
        (i_color = 'spring' or i_color =
'hot') and
        (i_units = 'Carton' or i_units =
'Tbl') and
        (i_size = 'large' or i_size =
'N/A')
        ) or
        (i_category = 'Women' and
        (i_color = 'magenta' or i_color =
'goldenrod') and
        (i_units = 'Cup' or i_units = 'Oz')
and
        (i_size = 'economy' or i_size =
'extra large')
        ) or
        (i_category = 'Men' and
        (i_color = 'cyan' or i_color =
'antique') and
        (i_units = 'Dozen' or i_units =
'Case') and
        (i_size = 'medium' or i_size =
'petite')
        ) or
        (i_category = 'Men' and
        (i_color = 'moccasin' or i_color =
'black') and
        (i_units = 'Box' or i_units =
'Pallet') and
        (i_size = 'large' or i_size =
'N/A')
        ))) or
       (
        ((i_category = 'Women' and
        (i_color = 'azure' or i_color =
'light') and
        (i_units = 'Gross' or i_units =
'Each') and
        (i_size = 'large' or i_size =
'N/A')
        ) or
        (i_category = 'Women' and
        (i_color = 'mint' or i_color =
'burnished') and
        (i_units = 'N/A' or i_units =
'Unknown') and
        (i_size = 'economy' or i_size =
'extra large')
        ) or
        (i_category = 'Men' and
        (i_color = 'floral' or i_color =
'midnight') and
        (i_units = 'Pound' or i_units =
'Ton') and
        (i_size = 'medium' or i_size =
'petite')
        ) or
        (i_category = 'Men' and
        (i_color = 'navy' or i_color =
'blue') and
        (i_units = 'Bundle' or i_units =
'Ounce') and
        (i_size = 'large' or i_size =
'N/A')
        )))
     group by i_manufact) i2
ON i1.i_manufact = i2.i_manufact
 where i1.i_manufact_id between 716 and
716+40
 and i2.item_cnt > 0
 order by i_product_name
  limit 100;

-- end query 62 in stream 0 using template
query41.tpl
-- start query 64 in stream 0 using
template query12.tpl and seed 918962166
with cte1 as
( select
    i_item_id
    ,i_item_desc
    ,i_category
    ,i_class
    ,i_current_price
    ,sum(ws_ext_sales_price) as itemrevenue
    from web_sales
```

```
         ,item
         ,date_dim
    where
        ws_item_sk = i_item_sk
        and ws_sold_date_sk = d_date_sk
       and i_category in ('Jewelry',
'Men', 'Books')
       and d_date between cast('2002-06-
11' as timestamp)
                and (cast('2002-06-11' as
timestamp) + interval 30 days)
    group by
        i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price ),
cte2 as
( select
    i_class
    ,sum(itemrevenue) as sumitemrevenue
  from cte1
  group by
    i_class)
 select  i_item_desc
   ,i_category
   ,i_class
   ,i_current_price
   ,itemrevenue
   ,revenueratio
from (
select
   cte1.i_item_id
   ,cte1.i_item_desc
   ,cte1.i_category
   ,cte1.i_class
   ,cte1.i_current_price
   ,cte1.itemrevenue

,cte1.itemrevenue*100/cte2.sumitemrevenue
as revenueratio
from cte1
     cross join cte2
where
   (cte1.i_class = cte2.i_class
   or
   (cte1.i_class is NULL and cte2.i_class
is NULL))
) v1
order by
   i_category
   ,i_class
   ,i_item_id
   ,i_item_desc
   ,revenueratio
 limit 100;

-- end query 64 in stream 0 using template
query12.tpl
-- start query 65 in stream 0 using
template query20.tpl and seed 711739272
with cte1 as
( select
    i_item_id
    ,i_item_desc
    ,i_category
    ,i_class
    ,i_current_price
    ,sum(cs_ext_sales_price) as itemrevenue
    from catalog_sales
         ,item
         ,date_dim
    where
        cs_item_sk = i_item_sk
    and i_category in ('Jewelry', 'Music',
'Men')
    and cs_sold_date_sk = d_date_sk
    and d_date between cast('2000-02-09' as
timestamp)
       and (cast('2000-02-09' as
timestamp) + interval 30 days)
    group by
        i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price ),
cte2 as
( select
    i_class
    ,sum(itemrevenue) as sumitemrevenue
  from cte1
  group by
    i_class)
  select  i_item_desc
   ,i_category
   ,i_class
   ,i_current_price
   ,itemrevenue
   ,revenueratio
from (
select
   cte1.i_item_id
   ,cte1.i_item_desc
   ,cte1.i_category
   ,cte1.i_class
   ,cte1.i_current_price
   ,cte1.itemrevenue

,cte1.itemrevenue*100/cte2.sumitemrevenue
as revenueratio
from cte1
     cross join cte2
where
   (cte1.i_class = cte2.i_class
   or
   (cte1.i_class is NULL and cte2.i_class
is NULL))
) v1
order by
   i_category
   ,i_class
   ,i_item_id
   ,i_item_desc
   ,revenueratio
 limit 100;

-- end query 65 in stream 0 using template
query20.tpl
-- start query 66 in stream 0 using
template query88.tpl and seed 1924183468
select  *
from
 (select count(*) h8_30_to_9
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 8
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
```

```
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s1
cross join
 (select count(*) h9_to_9_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 9
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s2
cross join
 (select count(*) h9_30_to_10
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 9
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s3
cross join
 (select count(*) h10_to_10_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 10
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s4
cross join
 (select count(*) h10_30_to_11
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 10
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s5
cross join
 (select count(*) h11_to_11_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 11
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s6
cross join
 (select count(*) h11_30_to_12
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 11
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
```

```
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s7
cross join
 (select count(*) h12_to_12_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 12
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s8
;

-- end query 66 in stream 0 using template
query88.tpl
-- start query 72 in stream 0 using
template query71.tpl and seed 1436004490
select i_brand_id brand_id, i_brand
brand,t_hour,t_minute,
       sum(ext_price) ext_price
 from item, (select ws_ext_sales_price as
ext_price,
                        ws_sold_date_sk as
sold_date_sk,
                        ws_item_sk as
sold_item_sk,
                        ws_sold_time_sk as
time_sk
                 from web_sales,date_dim
                 where d_date_sk =
ws_sold_date_sk
                   and d_moy=12
                   and d_year=1998
                 union all
                 select cs_ext_sales_price
as ext_price,
                        cs_sold_date_sk as
sold_date_sk,
                        cs_item_sk as
sold_item_sk,
                        cs_sold_time_sk as
time_sk
                 from
catalog_sales,date_dim
                 where d_date_sk =
cs_sold_date_sk
                   and d_moy=12
                   and d_year=1998
                 union all
                 select ss_ext_sales_price
as ext_price,
                        ss_sold_date_sk as
sold_date_sk,
                        ss_item_sk as
sold_item_sk,
                        ss_sold_time_sk as
time_sk
                 from store_sales,date_dim
                 where d_date_sk =
ss_sold_date_sk
                   and d_moy=12
                   and d_year=1998
                 ) as tmp,time_dim
 where
   sold_item_sk = i_item_sk
   and i_manager_id=1
   and time_sk = t_time_sk
   and (t_meal_time = 'breakfast' or
t_meal_time = 'dinner')
 group by i_brand,
i_brand_id,t_hour,t_minute
 order by ext_price desc, i_brand_id
 ;

-- end query 72 in stream 0 using template
query71.tpl
-- start query 73 in stream 0 using
template query34.tpl and seed 1451328249
select c_last_name
       ,c_first_name
       ,c_salutation
       ,c_preferred_cust_flag
       ,ss_ticket_number
       ,cnt from
   (select ss_ticket_number
          ,ss_customer_sk
          ,count(*) cnt
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and (date_dim.d_dom between 1 and 3 or
date_dim.d_dom between 25 and 28)
    and
(household_demographics.hd_buy_potential =
'1001-5000' or

household_demographics.hd_buy_potential =
'5001-10000')
    and
household_demographics.hd_vehicle_count > 0
    and (case when
household_demographics.hd_vehicle_count > 0
       then
household_demographics.hd_dep_count/
household_demographics.hd_vehicle_count
       else null
       end)  > 1.2
    and date_dim.d_year in
(1999,1999+1,1999+2)
    and store.s_county in ('Sierra
County','Lunenburg County','Jackson
County','Harmon County',
                           'Mesa
County','Pipestone County','Pennington
County','Perry County')
    group by
ss_ticket_number,ss_customer_sk)
dn,customer
    where ss_customer_sk = c_customer_sk
      and cnt between 15 and 20
    order by
c_last_name,c_first_name,c_salutation,c_pre
ferred_cust_flag desc;
```

```
-- end query 73 in stream 0 using template
query34.tpl
-- start query 78 in stream 0 using
template query77.tpl and seed 1879081522
with ss as
 (select s_store_sk,
         sum(ss_ext_sales_price) as sales,
         sum(ss_net_profit) as profit
 from store_sales,
      date_dim,
      store
 where ss_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as timestamp)
                  and (cast('2002-08-24' as
timestamp) +  interval 30 days)
       and ss_store_sk = s_store_sk
 group by s_store_sk)
 ,
 sr as
 (select s_store_sk,
         sum(sr_return_amt) as returnz,
         sum(sr_net_loss) as profit_loss
 from store_returns,
      date_dim,
      store
 where sr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as timestamp)
                  and (cast('2002-08-24' as
timestamp) +  interval 30 days)
       and sr_store_sk = s_store_sk
 group by s_store_sk),
 cs as
 (select cs_call_center_sk,
        sum(cs_ext_sales_price) as sales,
        sum(cs_net_profit) as profit
 from catalog_sales,
      date_dim
 where cs_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as timestamp)
                  and (cast('2002-08-24' as
timestamp) +  interval 30 days)
 group by cs_call_center_sk
 ),
 cr as
 (select
        sum(cr_return_amount) as returnz,
        sum(cr_net_loss) as profit_loss
 from catalog_returns,
      date_dim
 where cr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as timestamp)
                  and (cast('2002-08-24' as
timestamp) +  interval 30 days)
 ),
 ws as
 ( select wp_web_page_sk,
        sum(ws_ext_sales_price) as sales,
        sum(ws_net_profit) as profit
 from web_sales,
      date_dim,
      web_page
 where ws_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as timestamp)
                  and (cast('2002-08-24' as
timestamp) +  interval 30 days)
       and ws_web_page_sk = wp_web_page_sk
 group by wp_web_page_sk),
 wr as
 (select wp_web_page_sk,
        sum(wr_return_amt) as returnz,
        sum(wr_net_loss) as profit_loss
 from web_returns,
      date_dim,
      web_page
 where wr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as timestamp)
                  and (cast('2002-08-24' as
timestamp) +  interval 30 days)
       and wr_web_page_sk = wp_web_page_sk
 group by wp_web_page_sk)
 ,
 results as
 (select channel
        , id
        , sum(sales) as sales
        , sum(returnz) as returnz
        , sum(profit) as profit
 from
 (select 'store channel' as channel
        , ss.s_store_sk as id
        , sales
        , coalesce(returnz, 0) as returnz
        , (profit -
coalesce(profit_loss,0)) as profit
 from   ss left join sr
        on  ss.s_store_sk = sr.s_store_sk
 union all
 select 'catalog channel' as channel
        , cs_call_center_sk as id
        , sales
        , returnz
        , (profit - profit_loss) as profit
 from  cs
       cross join cr
 union all
 select 'web channel' as channel
        , ws.wp_web_page_sk as id
        , sales
        , coalesce(returnz, 0) returnz
        , (profit -
coalesce(profit_loss,0)) as profit
 from   ws left join wr
        on  ws.wp_web_page_sk =
wr.wp_web_page_sk
 ) x
 group by channel, id )

  select  channel
       , id
       , sales
       , returnz
       , profit
 from (
 select channel, id, sales, returnz, profit
from  results
 union
 select channel, NULL AS id, sum(sales) as
sales, sum(returnz) as returnz, sum(profit)
as profit from  results group by channel
 union
 select NULL AS channel, NULL AS id,
sum(sales) as sales, sum(returnz) as
returnz, sum(profit) as profit from
results
) foo
order by channel, id
  limit 100;

-- end query 78 in stream 0 using template
query77.tpl
-- start query 79 in stream 0 using
template query73.tpl and seed 413577677
select c_last_name
       ,c_first_name
       ,c_salutation
       ,c_preferred_cust_flag
       ,ss_ticket_number
       ,cnt from
```

```
   (select ss_ticket_number
          ,ss_customer_sk
          ,count(*) cnt
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and date_dim.d_dom between 1 and 2
    and
(household_demographics.hd_buy_potential =
'501-1000' or

household_demographics.hd_buy_potential =
'5001-10000')
    and
household_demographics.hd_vehicle_count > 0
    and case when
household_demographics.hd_vehicle_count > 0
then

household_demographics.hd_dep_count/
household_demographics.hd_vehicle_count
else null end > 1
    and date_dim.d_year in
(1999,1999+1,1999+2)
    and store.s_county in ('Lea
County','West Feliciana Parish','Nowata
County','Jackson County')
    group by
ss_ticket_number,ss_customer_sk)
dj,customer
    where ss_customer_sk = c_customer_sk
      and cnt between 1 and 5
    order by cnt desc;

-- end query 79 in stream 0 using template
query73.tpl
-- start query 80 in stream 0 using
template query84.tpl and seed 1842474049
select  c_customer_id as customer_id
       ,concat(c_last_name, ', ',
coalesce(c_first_name,'')) as customername
 from customer
     ,customer_address
     ,customer_demographics
     ,household_demographics
     ,income_band
     ,store_returns
 where ca_city         =  'Mount Zion'
   and c_current_addr_sk = ca_address_sk
   and ib_lower_bound   >=  50749
   and ib_upper_bound   <=  50749 + 50000
   and ib_income_band_sk =
hd_income_band_sk
   and cd_demo_sk = c_current_cdemo_sk
   and hd_demo_sk = c_current_hdemo_sk
   and sr_cdemo_sk = cd_demo_sk
 order by c_customer_id
  limit 100;

-- end query 80 in stream 0 using template
query84.tpl
-- start query 82 in stream 0 using
template query55.tpl and seed 1117454508
select  i_brand_id brand_id, i_brand brand,
        sum(ss_ext_sales_price) ext_price
 from date_dim, store_sales, item
 where d_date_sk = ss_sold_date_sk
       and ss_item_sk = i_item_sk
       and i_manager_id=48
       and d_moy=11
       and d_year=2001
 group by i_brand, i_brand_id
 order by ext_price desc, i_brand_id
 limit 100 ;

-- end query 82 in stream 0 using template
query55.tpl
-- start query 83 in stream 0 using
template query56.tpl and seed 1152645577
with ss as (
 select
item.i_item_id,sum(ss_ext_sales_price)
total_sales
 from
       store_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select i_item_id
                        from item
                        where i_color in
('maroon','powder','lawn')
       ) ssi on item.i_item_id =
ssi.i_item_id
 where   ss_item_sk              =
i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 cs as (
 select
item.i_item_id,sum(cs_ext_sales_price)
total_sales
 from
       catalog_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select i_item_id
                        from item
                        where i_color in
('maroon','powder','lawn')
       ) csi on item.i_item_id =
csi.i_item_id
 where   cs_item_sk              =
i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id),
 ws as (
 select
item.i_item_id,sum(ws_ext_sales_price)
total_sales
 from
       web_sales,
       date_dim,
         customer_address,
         item
        left semi join (
                        select i_item_id
                        from item
                        where i_color in
('maroon','powder','lawn')
       ) wsi on item.i_item_id =
wsi.i_item_id
```

```
 where   ws_item_sk              =
i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 group by i_item_id)
  select  i_item_id ,sum(total_sales)
total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_item_id
 order by total_sales
  limit 100;

-- end query 83 in stream 0 using template
query56.tpl
-- start query 84 in stream 0 using
template query2.tpl and seed 1528114170
with wscs as
 (select sold_date_sk
        ,sales_price
  from (select ws_sold_date_sk sold_date_sk
              ,ws_ext_sales_price
sales_price
        from web_sales) x
        union all
       (select cs_sold_date_sk sold_date_sk
              ,cs_ext_sales_price
sales_price
        from catalog_sales)),
 wswscs as
 (select d_week_seq,
        sum(case when (d_day_name='Sunday')
then sales_price else null end) sun_sales,
        sum(case when (d_day_name='Monday')
then sales_price else null end) mon_sales,
        sum(case when
(d_day_name='Tuesday') then sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then sales_price
else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then sales_price else null end) fri_sales,
        sum(case when
(d_day_name='Saturday') then sales_price
else null end) sat_sales
 from wscs
     ,date_dim
 where d_date_sk = sold_date_sk
 group by d_week_seq)
 select d_week_seq1
       ,round(sun_sales1/sun_sales2,2)
       ,round(mon_sales1/mon_sales2,2)
       ,round(tue_sales1/tue_sales2,2)
       ,round(wed_sales1/wed_sales2,2)
       ,round(thu_sales1/thu_sales2,2)
       ,round(fri_sales1/fri_sales2,2)
       ,round(sat_sales1/sat_sales2,2)
 from
 (select wswscs.d_week_seq d_week_seq1
        ,sun_sales sun_sales1
        ,mon_sales mon_sales1
        ,tue_sales tue_sales1
        ,wed_sales wed_sales1
        ,thu_sales thu_sales1
        ,fri_sales fri_sales1
        ,sat_sales sat_sales1
  from wswscs,date_dim
  where date_dim.d_week_seq =
wswscs.d_week_seq and
        d_year = 1998) y,
 (select wswscs.d_week_seq d_week_seq2
        ,sun_sales sun_sales2
        ,mon_sales mon_sales2
        ,tue_sales tue_sales2
        ,wed_sales wed_sales2
        ,thu_sales thu_sales2
        ,fri_sales fri_sales2
        ,sat_sales sat_sales2
  from wswscs
      ,date_dim
  where date_dim.d_week_seq =
wswscs.d_week_seq and
        d_year = 1998+1) z
 where d_week_seq1=d_week_seq2-53
 order by d_week_seq1;

-- end query 84 in stream 0 using template
query2.tpl
-- start query 85 in stream 0 using
template query26.tpl and seed 1427200905
select  i_item_id,
        avg(cs_quantity) agg1,
        avg(cs_list_price) agg2,
        avg(cs_coupon_amt) agg3,
        avg(cs_sales_price) agg4
 from catalog_sales, customer_demographics,
date_dim, item, promotion
 where cs_sold_date_sk = d_date_sk and
       cs_item_sk = i_item_sk and
       cs_bill_cdemo_sk = cd_demo_sk and
       cs_promo_sk = p_promo_sk and
       cd_gender = 'M' and
       cd_marital_status = 'D' and
       cd_education_status = 'Advanced
Degree' and
       (p_channel_email = 'N' or
p_channel_event = 'N') and
       d_year = 2000
 group by i_item_id
 order by i_item_id
  limit 100;

-- end query 85 in stream 0 using template
query26.tpl
-- start query 86 in stream 0 using
template query40.tpl and seed 600490395
select
   w_state
  ,i_item_id
  ,sum(case when d_date < cast ('2000-04-
27' as timestamp)
              then cs_sales_price -
coalesce(cr_refunded_cash,0) else 0 end) as
sales_before
  ,sum(case when d_date >= cast ('2000-04-
27' as timestamp)
              then cs_sales_price -
coalesce(cr_refunded_cash,0) else 0 end) as
sales_after
 from
   catalog_sales left outer join
catalog_returns on
       (cs_order_number = cr_order_number
        and cs_item_sk = cr_item_sk)
  ,warehouse
  ,item
  ,date_dim
 where
     i_current_price between 0.99 and 1.49
 and i_item_sk          = cs_item_sk
 and cs_warehouse_sk    = w_warehouse_sk
 and cs_sold_date_sk    = d_date_sk
```

```
 and d_date between (cast ('2000-04-27' as
timestamp) - interval 30 days)
                 and (cast ('2000-04-27' as
timestamp) + interval 30 days)
 group by
    w_state,i_item_id
 order by w_state,i_item_id
  limit 100;

-- end query 86 in stream 0 using template
query40.tpl
-- start query 88 in stream 0 using
template query53.tpl and seed 1796782974
with cte1 as
(
select i_manufact_id,
sum(ss_sales_price) sum_sales

from item, store_sales, date_dim, store
where ss_item_sk = i_item_sk and
ss_sold_date_sk = d_date_sk and
ss_store_sk = s_store_sk and
d_month_seq in
(1198,1198+1,1198+2,1198+3,1198+4,1198+5,11
98+6,1198+7,1198+8,1198+9,1198+10,1198+11)
and
((i_category in
('Books','Children','Electronics') and
i_class in
('personal','portable','reference','self-
help') and
i_brand in ('scholaramalgamalg
#14','scholaramalgamalg #7',
                'exportiunivamalg
#9','scholaramalgamalg #9'))
or(i_category in ('Women','Music','Men')
and
i_class in
('accessories','classical','fragrances','pa
nts') and
i_brand in ('amalgimporto #1','edu
packscholar #1','exportiimporto #1',
                'importoamalg #1')))
group by i_manufact_id, d_qoy
),
cte2 as
(
select
  i_manufact_id,
  avg(sum_sales) avg_quarterly_sales
from cte1
  group by
  i_manufact_id)

 select
  cte1.i_manufact_id
  ,cte1.sum_sales
  ,cte2.avg_quarterly_sales
from cte1
     cross join cte2
where
  (cte1.i_manufact_id = cte2.i_manufact_id
  or
  (cte1.i_manufact_id is NULL and
cte2.i_manufact_id is NULL))
  and case when avg_quarterly_sales > 0
       then abs (sum_sales -
avg_quarterly_sales)/ avg_quarterly_sales
       else null end > 0.1
order by avg_quarterly_sales,
         sum_sales,
         cte1.i_manufact_id
 limit 100;

-- end query 88 in stream 0 using template
query53.tpl
```

```
-- start query 89 in stream 0 using
template query79.tpl and seed 2112737383
select

c_last_name,c_first_name,substr(s_city,1,30
),ss_ticket_number,amt,profit
  from
   (select ss_ticket_number
          ,ss_customer_sk
          ,store.s_city
          ,sum(ss_coupon_amt) amt
          ,sum(ss_net_profit) profit
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and
(household_demographics.hd_dep_count = 3 or
household_demographics.hd_vehicle_count >
1)
    and date_dim.d_dow = 1
    and date_dim.d_year in
(2000,2000+1,2000+2)
    and store.s_number_employees between
200 and 295
    group by
ss_ticket_number,ss_customer_sk,ss_addr_sk,
store.s_city) ms,customer
    where ss_customer_sk = c_customer_sk
 order by
c_last_name,c_first_name,substr(s_city,1,30
), profit
 limit 100;

-- end query 89 in stream 0 using template
query79.tpl
-- start query 96 in stream 0 using
template query83.tpl and seed 593789178
with sr_items as
 (select i_item_id item_id,
        sum(sr_return_quantity) sr_item_qty
 from store_returns,
      item,
      date_dim
      left semi join
                (select d_date
                 from date_dim
      left semi join
                (select d_week_seq
                from date_dim
          where d_date in ('1999-06-
14','1999-08-26','1999-11-06')
      ) d3 on date_dim.d_week_seq =
d3.d_week_seq
      ) d2 on date_dim.d_date = d2.d_date
 where sr_item_sk = i_item_sk
 and   sr_returned_date_sk   = d_date_sk
 group by i_item_id),
 cr_items as
 (select i_item_id item_id,
        sum(cr_return_quantity) cr_item_qty
 from catalog_returns,
      item,
      date_dim
      left semi join
                (select d_date
                 from date_dim
      left semi join
                (select d_week_seq
                from date_dim
          where d_date in ('1999-06-
14','1999-08-26','1999-11-06')
```

```
      ) d3 on date_dim.d_week_seq =
d3.d_week_seq
      ) d2 on date_dim.d_date = d2.d_date
 where cr_item_sk = i_item_sk
 and   cr_returned_date_sk   = d_date_sk
 group by i_item_id),
 wr_items as
 (select i_item_id item_id,
        sum(wr_return_quantity) wr_item_qty
 from web_returns,
      item,
      date_dim
      left semi join
                (select d_date
                 from date_dim
      left semi join
                (select d_week_seq
                from date_dim
           where d_date in ('1999-06-
14','1999-08-26','1999-11-06')
      ) d3 on date_dim.d_week_seq =
d3.d_week_seq
      ) d2 on date_dim.d_date = d2.d_date
 where wr_item_sk = i_item_sk
 and   wr_returned_date_sk   = d_date_sk
 group by i_item_id)
  select  sr_items.item_id
       ,sr_item_qty

,sr_item_qty/(sr_item_qty+cr_item_qty+wr_it
em_qty)/3.0 * 100 sr_dev
       ,cr_item_qty

,cr_item_qty/(sr_item_qty+cr_item_qty+wr_it
em_qty)/3.0 * 100 cr_dev
       ,wr_item_qty

,wr_item_qty/(sr_item_qty+cr_item_qty+wr_it
em_qty)/3.0 * 100 wr_dev

,(sr_item_qty+cr_item_qty+wr_item_qty)/3.0
average
 from sr_items
     ,cr_items
     ,wr_items
 where sr_items.item_id=cr_items.item_id
   and sr_items.item_id=wr_items.item_id
 order by sr_items.item_id
         ,sr_item_qty
  limit 100;

-- end query 96 in stream 0 using template
query83.tpl
-- start query 97 in stream 0 using
template query61.tpl and seed 1770420976
select  promotions,total,cast(promotions as
decimal(15,4))/cast(total as
decimal(15,4))*100
from
  (select sum(ss_ext_sales_price)
promotions
   from  store_sales
        ,store
        ,promotion
        ,date_dim
        ,customer
        ,customer_address
        ,item
   where ss_sold_date_sk = d_date_sk
   and   ss_store_sk = s_store_sk
   and   ss_promo_sk = p_promo_sk
   and   ss_customer_sk= c_customer_sk
   and   ca_address_sk = c_current_addr_sk
   and   ss_item_sk = i_item_sk
   and   ca_gmt_offset = -7
   and   i_category = 'Electronics'
   and   (p_channel_dmail = 'Y' or
p_channel_email = 'Y' or p_channel_tv =
'Y')
   and   s_gmt_offset = -7
   and   d_year = 1999
   and   d_moy  = 12) promotional_sales
cross join
  (select sum(ss_ext_sales_price) total
   from  store_sales
        ,store
        ,date_dim
        ,customer
        ,customer_address
        ,item
   where ss_sold_date_sk = d_date_sk
   and   ss_store_sk = s_store_sk
   and   ss_customer_sk= c_customer_sk
   and   ca_address_sk = c_current_addr_sk
   and   ss_item_sk = i_item_sk
   and   ca_gmt_offset = -7
   and   i_category = 'Electronics'
   and   s_gmt_offset = -7
   and   d_year = 1999
   and   d_moy  = 12) all_sales
order by promotions, total
 limit 100;

-- end query 97 in stream 0 using template
query61.tpl
-- start query 99 in stream 0 using
template query76.tpl and seed 945056756
select  channel, col_name, d_year, d_qoy,
i_category, COUNT(*) sales_cnt,
SUM(ext_sales_price) sales_amt FROM (
        SELECT 'store' as channel,
'ss_hdemo_sk' col_name, d_year, d_qoy,
i_category, ss_ext_sales_price
ext_sales_price
         FROM store_sales, item, date_dim
         WHERE ss_hdemo_sk IS NULL
           AND ss_sold_date_sk=d_date_sk
           AND ss_item_sk=i_item_sk
        UNION ALL
        SELECT 'web' as channel,
'ws_web_page_sk' col_name, d_year, d_qoy,
i_category, ws_ext_sales_price
ext_sales_price
         FROM web_sales, item, date_dim
         WHERE ws_web_page_sk IS NULL
           AND ws_sold_date_sk=d_date_sk
           AND ws_item_sk=i_item_sk
        UNION ALL
        SELECT 'catalog' as channel,
'cs_ship_addr_sk' col_name, d_year, d_qoy,
i_category, cs_ext_sales_price
ext_sales_price
         FROM catalog_sales, item, date_dim
         WHERE cs_ship_addr_sk IS NULL
           AND cs_sold_date_sk=d_date_sk
           AND cs_item_sk=i_item_sk) foo
GROUP BY channel, col_name, d_year, d_qoy,
i_category
ORDER BY channel, col_name, d_year, d_qoy,
i_category
 limit 100;

-- end query 99 in stream 0 using template
query76.tpl
```

E.3 Hive 0.13 Queries:

```
-- start query 1 in stream 0 using template
query96.tpl and seed 550831069
select  count(*) count1
from store_sales
```

```
    ,household_demographics
    ,time_dim, store
where ss_sold_time_sk = time_dim.t_time_sk
    and ss_hdemo_sk =
household_demographics.hd_demo_sk
    and ss_store_sk = s_store_sk
    and time_dim.t_hour = 15
    and time_dim.t_minute >= 30
    and household_demographics.hd_dep_count
= 6
    and store.s_store_name = 'ese'
--order by count(*)
order by count1
  limit 100;
-- end query 1 in stream 0 using template
query96.tpl
-- start query 2 in stream 0 using template
query7.tpl and seed 997258328
select  i_item_id,
        avg(ss_quantity) agg1,
        avg(ss_list_price) agg2,
        avg(ss_coupon_amt) agg3,
        avg(ss_sales_price) agg4
 from store_sales, customer_demographics,
date_dim, item, promotion
 where ss_sold_date_sk = d_date_sk and
       ss_item_sk = i_item_sk and
       ss_cdemo_sk = cd_demo_sk and
       ss_promo_sk = p_promo_sk and
       cd_gender = 'M' and
       cd_marital_status = 'W' and
       cd_education_status = '2 yr Degree'
and
       (p_channel_email = 'N' or
p_channel_event = 'N') and
       d_year = 1999
 group by i_item_id
 order by i_item_id
  limit 100;

-- end query 2 in stream 0 using template
query7.tpl
-- start query 5 in stream 0 using template
query39.tpl and seed 1420791654
with inv as
(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy
       ,stdev,mean, case mean when 0 then
null else stdev/mean end cov
 from(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy

,stddev_samp(inv_quantity_on_hand)
stdev,avg(inv_quantity_on_hand) mean
      from inventory
          ,item
          ,warehouse
          ,date_dim
      where inv_item_sk = i_item_sk
        and inv_warehouse_sk =
w_warehouse_sk
        and inv_date_sk = d_date_sk
        and d_year =2000
      group by
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy) foo
 where case mean when 0 then 0 else
stdev/mean end > 1)
select
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean, inv1.cov

,inv2.w_warehouse_sk,inv2.i_item_sk,inv2.d_
moy,inv2.mean, inv2.cov
from inv inv1,inv inv2
where inv1.i_item_sk = inv2.i_item_sk
  and inv1.w_warehouse_sk =
inv2.w_warehouse_sk
  and inv1.d_moy=2
  and inv2.d_moy=2+1
order by
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean,inv1.cov
       ,inv2.d_moy,inv2.mean, inv2.cov
;
with inv as
(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy
       ,stdev,mean, case mean when 0 then
null else stdev/mean end cov
 from(select
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy

,stddev_samp(inv_quantity_on_hand)
stdev,avg(inv_quantity_on_hand) mean
      from inventory
          ,item
          ,warehouse
          ,date_dim
      where inv_item_sk = i_item_sk
        and inv_warehouse_sk =
w_warehouse_sk
        and inv_date_sk = d_date_sk
        and d_year =2000
      group by
w_warehouse_name,w_warehouse_sk,i_item_sk,d
_moy) foo
 where case mean when 0 then 0 else
stdev/mean end > 1)
select
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean, inv1.cov

,inv2.w_warehouse_sk,inv2.i_item_sk,inv2.d_
moy,inv2.mean, inv2.cov
from inv inv1,inv inv2
where inv1.i_item_sk = inv2.i_item_sk
  and inv1.w_warehouse_sk =
inv2.w_warehouse_sk
  and inv1.d_moy=2
  and inv2.d_moy=2+1
  and inv1.cov > 1.5
order by
inv1.w_warehouse_sk,inv1.i_item_sk,inv1.d_m
oy,inv1.mean,inv1.cov
       ,inv2.d_moy,inv2.mean, inv2.cov
;

-- end query 5 in stream 0 using template
query39.tpl
-- start query 7 in stream 0 using template
query32.tpl and seed 944563352
select  sum(cs.cs_ext_discount_amt) as
excess_discount_amount
from
   catalog_sales cs
   join date_dim dd on dd.d_date_sk =
cs.cs_sold_date_sk
   join item i on i.i_item_sk =
cs.cs_item_sk
   join
      (
         select
            cs2.cs_item_sk as cs_item_sk,
            1.3 *
avg(cs2.cs_ext_discount_amt)  as
tmp_cs_ext_discount_amt
         from
            catalog_sales cs2
```

```
            join date_dim dd2 on
dd2.d_date_sk = cs2.cs_sold_date_sk
         where
         dd2.d_date between '2000-01-16'
         and date_add(cast('2000-01-16' as
date), 90)
         group by cs2.cs_item_sk
      ) tmp on tmp.cs_item_sk = i.i_item_sk

where
i.i_manufact_id = 353
and dd.d_date between '2000-01-16'
and date_add(cast('2000-01-16' as date),
90)
and cs.cs_ext_discount_amt >
tmp.tmp_cs_ext_discount_amt
  limit 100;

-- end query 7 in stream 0 using template
query32.tpl
-- start query 14 in stream 0 using
template query21.tpl and seed 614834996
select  *
 from(select w_warehouse_name
            ,i_item_id
            ,sum(case when (cast(d_date as
date) < cast ('1998-06-27' as date))
                      then
inv_quantity_on_hand
                      else 0 end) as
inv_before
            ,sum(case when (cast(d_date as
date) >= cast ('1998-06-27' as date))
                      then
inv_quantity_on_hand
                      else 0 end) as
inv_after
   from inventory
       ,warehouse
       ,item
       ,date_dim
   where i_current_price between 0.99 and
1.49
     and i_item_sk          = inv_item_sk
     and inv_warehouse_sk   =
w_warehouse_sk
     and inv_date_sk    = d_date_sk
     and d_date between date_sub(cast
('1998-06-27' as date), 30)
                    and date_add(cast
('1998-06-27' as date), 30)
   group by w_warehouse_name, i_item_id) x
 where (case when inv_before > 0
             then inv_after / inv_before
             else null
             end) between 2.0/3.0 and
3.0/2.0
 order by w_warehouse_name
         ,i_item_id
   limit 100;

-- end query 14 in stream 0 using template
query21.tpl
-- start query 15 in stream 0 using
template query43.tpl and seed 959608359
select  s_store_name, s_store_id,
        sum(case when (d_day_name='Sunday')
then ss_sales_price else null end)
sun_sales,
        sum(case when (d_day_name='Monday')
then ss_sales_price else null end)
mon_sales,
        sum(case when
(d_day_name='Tuesday') then ss_sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then
ss_sales_price else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then ss_sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then ss_sales_price else null end)
fri_sales,
        sum(case when
(d_day_name='Saturday') then ss_sales_price
else null end) sat_sales
 from date_dim, store_sales, store
 where d_date_sk = ss_sold_date_sk and
       s_store_sk = ss_store_sk and
       s_gmt_offset = -8 and
       d_year = 1998
 group by s_store_name, s_store_id
 order by s_store_name,
s_store_id,sun_sales,mon_sales,tue_sales,we
d_sales,thu_sales,fri_sales,sat_sales
  limit 100;

-- end query 15 in stream 0 using template
query43.tpl
-- start query 16 in stream 0 using
template query27.tpl and seed 331218716
select  i_item_id,
        s_state,
        --grouping(s_state) g_state,
        GROUPING__ID g_state,
        avg(ss_quantity) agg1,
        avg(ss_list_price) agg2,
        avg(ss_coupon_amt) agg3,
        avg(ss_sales_price) agg4
 from store_sales, customer_demographics,
date_dim, store, item
 where ss_sold_date_sk = d_date_sk and
       ss_item_sk = i_item_sk and
       ss_store_sk = s_store_sk and
       ss_cdemo_sk = cd_demo_sk and
       cd_gender = 'F' and
       cd_marital_status = 'W' and
       cd_education_status = '4 yr Degree'
and
       d_year = 1999 and
       s_state in ('OH','IL', 'LA', 'GA',
'CO', 'AL')
 group by i_item_id, s_state with ROLLUP
 order by i_item_id
         ,s_state
   limit 100;

-- end query 16 in stream 0 using template
query27.tpl
-- start query 19 in stream 0 using
template query58.tpl and seed 1844319395
with ss_items as
 (select i_item_id item_id
        ,sum(ss_ext_sales_price)
ss_item_rev
 from store_sales
     JOIN item ON ss_item_sk = i_item_sk
     JOIN date_dim dd0 ON ss_sold_date_sk =
dd0.d_date_sk
     JOIN
        (select dd1.d_date

        from date_dim dd1
             JOIN  date_dim dd2 ON
dd1.d_week_seq = dd2.d_week_seq
        where dd2.d_date = '1998-05-29') v1
ON dd0.d_date = v1.d_date
 group by i_item_id),
cs_items as
 (select i_item_id item_id
```

```
        ,sum(cs_ext_sales_price)
cs_item_rev
 from catalog_sales
     JOIN item ON cs_item_sk = i_item_sk
     JOIN date_dim dd0 ON cs_sold_date_sk
= dd0.d_date_sk
     JOIN
        (select dd1.d_date

        from date_dim dd1
             JOIN  date_dim dd2 ON
dd1.d_week_seq = dd2.d_week_seq
        where dd2.d_date = '1998-05-29') v1
ON dd0.d_date = v1.d_date
 group by i_item_id),
ws_items as
 (select i_item_id item_id
        ,sum(ws_ext_sales_price)
ws_item_rev
 from web_sales
     JOIN item ON ws_item_sk = i_item_sk
     JOIN date_dim dd0 ON ws_sold_date_sk =
dd0.d_date_sk
     JOIN
        (select dd1.d_date

        from date_dim dd1
             JOIN  date_dim dd2 ON
dd1.d_week_seq = dd2.d_week_seq
        where dd2.d_date = '1998-05-29') v1
ON dd0.d_date = v1.d_date
 group by i_item_id)
  select  ss_items.item_id
       ,ss_item_rev

,ss_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 ss_dev
       ,cs_item_rev

,cs_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 cs_dev
       ,ws_item_rev

,ws_item_rev/(ss_item_rev+cs_item_rev+ws_it
em_rev)/3 * 100 ws_dev

,(ss_item_rev+cs_item_rev+ws_item_rev)/3
average
 from ss_items,cs_items,ws_items
 where ss_items.item_id=cs_items.item_id
   and ss_items.item_id=ws_items.item_id
   and ss_item_rev between 0.9 *
cs_item_rev and 1.1 * cs_item_rev
   and ss_item_rev between 0.9 *
ws_item_rev and 1.1 * ws_item_rev
   and cs_item_rev between 0.9 *
ss_item_rev and 1.1 * ss_item_rev
   and cs_item_rev between 0.9 *
ws_item_rev and 1.1 * ws_item_rev
   and ws_item_rev between 0.9 *
ss_item_rev and 1.1 * ss_item_rev
   and ws_item_rev between 0.9 *
cs_item_rev and 1.1 * cs_item_rev
 order by item_id
         ,ss_item_rev
   limit 100;

-- end query 19 in stream 0 using template
query58.tpl
-- start query 22 in stream 0 using
template query33.tpl and seed 248487088
with ss as (
 select

i_manufact_id,sum(ss_ext_sales_price)
total_sales
 from
 (
 select distinct i1.i_manufact_id,
ss_ext_sales_price
 from
       store_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where
         i1.i_manufact_id =
i2.i_manufact_id
 and     i2.i_category in ('Books')
 and     ss_item_sk              =
i1.i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -7
 ) v1
 group by i_manufact_id),
 cs as (
 select

i_manufact_id,sum(cs_ext_sales_price)
total_sales
 from
 (
 select distinct i1.i_manufact_id,
cs_ext_sales_price
 from
       catalog_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where
         i1.i_manufact_id =
i2.i_manufact_id
 and     i2.i_category in ('Books')
 and     cs_item_sk              =
i1.i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -7
 ) v2
 group by i_manufact_id),
 ws as (
 select

i_manufact_id,sum(ws_ext_sales_price)
total_sales
 from
 (
 select distinct i1.i_manufact_id,
ws_ext_sales_price
 from
       web_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where
         i1.i_manufact_id =
i2.i_manufact_id
 and     i2.i_category in ('Books')
 and     ws_item_sk              =
i1.i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
```

```
 and     d_year                  = 2001
 and     d_moy                   = 6
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -7
 ) v3
 group by i_manufact_id)
  select  i_manufact_id ,sum(total_sales)
total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_manufact_id
 order by total_sales
  limit 100;

-- end query 22 in stream 0 using template
query33.tpl
-- start query 24 in stream 0 using
template query62.tpl and seed 800775315
select
   substr(w_warehouse_name,1,20)
w_warehouse_name
  ,sm_type
  ,web_name
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk <= 30 ) then 1 else 0 end)
as 30_days
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 30) and
                 (ws_ship_date_sk -
ws_sold_date_sk <= 60) then 1 else 0 end )
as 31_60_days
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 60) and
                 (ws_ship_date_sk -
ws_sold_date_sk <= 90) then 1 else 0 end)
as 61_90_days
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk > 90) and
                 (ws_ship_date_sk -
ws_sold_date_sk <= 120) then 1 else 0 end)
as 91_120_days
  ,sum(case when (ws_ship_date_sk -
ws_sold_date_sk  > 120) then 1 else 0 end)
as above120_days
from
   web_sales
  ,warehouse
  ,ship_mode
  ,web_site
  ,date_dim
where
    d_month_seq between 1201 and 1201 + 11
and ws_ship_date_sk   = d_date_sk
and ws_warehouse_sk   = w_warehouse_sk
and ws_ship_mode_sk   = sm_ship_mode_sk
and ws_web_site_sk    = web_site_sk
group by
   substr(w_warehouse_name,1,20)
  ,sm_type
  ,web_name
order by substr(w_warehouse_name,1,20)
        ,sm_type
       ,web_name
  limit 100;

-- end query 24 in stream 0 using template
query62.tpl
-- start query 27 in stream 0 using
template query63.tpl and seed 812633773
select  *
from (select i_manager_id
             ,sum(ss_sales_price) sum_sales
             ,avg(sum(ss_sales_price)) over
(partition by i_manager_id)
avg_monthly_sales
      from item
          ,store_sales
          ,date_dim
          ,store
      where ss_item_sk = i_item_sk
        and ss_sold_date_sk = d_date_sk
        and ss_store_sk = s_store_sk
        and d_month_seq in
(1178,1178+1,1178+2,1178+3,1178+4,1178+5,11
78+6,1178+7,1178+8,1178+9,1178+10,1178+11)
        and ((    i_category in
('Books','Children','Electronics')
              and i_class in
('personal','portable','refernece','self-
help')
              and i_brand in
('scholaramalgamalg #14','scholaramalgamalg
#7',

'exportiunivamalg #9','scholaramalgamalg
#9'))
          or(    i_category in
('Women','Music','Men')
              and i_class in
('accessories','classical','fragrances','pa
nts')
              and i_brand in ('amalgimporto
#1','edu packscholar #1','exportiimporto
#1',

'importoamalg #1')))
group by i_manager_id, d_moy) tmp1
where case when avg_monthly_sales > 0 then
abs (sum_sales - avg_monthly_sales) /
avg_monthly_sales else null end > 0.1
order by i_manager_id
        ,avg_monthly_sales
        ,sum_sales
 limit 100;

-- end query 27 in stream 0 using template
query63.tpl
-- start query 28 in stream 0 using
template query69.tpl and seed 1390437346
select
  cd_gender,
  cd_marital_status,
  cd_education_status,
  count(*) cnt1,
  cd_purchase_estimate,
  count(*) cnt2,
  cd_credit_rating,
  count(*) cnt3
 from
  customer c
  JOIN customer_address ca ON
c.c_current_addr_sk = ca.ca_address_sk
  JOIN customer_demographics ON cd_demo_sk
= c.c_current_cdemo_sk
  LEFT SEMI JOIN
  (select ss_customer_sk
          from store_sales,date_dim
          where --c.c_customer_sk =
ss_customer_sk and
                ss_sold_date_sk = d_date_sk
and
                d_year = 2004 and
                d_moy between 3 and 3+2)
ssdd
  ON c.c_customer_sk = ssdd.ss_customer_sk
  LEFT OUTER JOIN
   (select ws_bill_customer_sk
           from web_sales,date_dim
```

```
            where --c.c_customer_sk =
ws_bill_customer_sk and
                  ws_sold_date_sk =
d_date_sk and
                  d_year = 2004 and
                  d_moy between 3 and 3+2)
wsdd
  ON c.c_customer_sk =
wsdd.ws_bill_customer_sk
  LEFT OUTER JOIN
    (select cs_ship_customer_sk
            from catalog_sales,date_dim
            where --c.c_customer_sk =
cs_ship_customer_sk and
                  cs_sold_date_sk =
d_date_sk and
                  d_year = 2004 and
                  d_moy between 3 and 3+2)
csdd  --)
  ON c.c_customer_sk =
csdd.cs_ship_customer_sk

 where
  ca_state in ('AL','VA','GA') and
  wsdd.ws_bill_customer_sk is null and
  csdd.cs_ship_customer_sk is null
 group by cd_gender,
          cd_marital_status,
          cd_education_status,
          cd_purchase_estimate,
          cd_credit_rating
 order by cd_gender,
          cd_marital_status,
          cd_education_status,
          cd_purchase_estimate,
          cd_credit_rating
   limit 100;

-- end query 28 in stream 0 using template
query69.tpl
-- start query 29 in stream 0 using
template query60.tpl and seed 374071684
with ss as (
 select
         i_item_id,sum(ss_ext_sales_price)
total_sales
 from
 (
 select distinct i1.i_item_id,
ss_ext_sales_price
 from
       store_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where
         i1.i_item_id = i2.i_item_id
 and     i2.i_category in ('Jewelry')
 and     ss_item_sk              =
i1.i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -5
 ) v1
 group by i_item_id),
 cs as (
 select
          i_item_id,sum(cs_ext_sales_price)
total_sales
 from
 (
 select distinct i1.i_item_id,
cs_ext_sales_price
 from
       catalog_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where
         i1.i_item_id = i2.i_item_id
 and     i2.i_category in ('Jewelry')
 and     cs_item_sk              =
i1.i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 ) v2
 group by i_item_id),
 ws as (
 select
          i_item_id,sum(ws_ext_sales_price)
total_sales
 from
 (
 select distinct i1.i_item_id,
ws_ext_sales_price
 from
       web_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where
         i1.i_item_id = i2.i_item_id
 and     i2.i_category in ('Jewelry')
 and     ws_item_sk              =
i1.i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2002
 and     d_moy                   = 10
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 ) v3
 group by i_item_id)
  select
  i_item_id
,sum(total_sales) total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_item_id
 order by i_item_id
      ,total_sales
   limit 100;

-- end query 29 in stream 0 using template
query60.tpl
-- start query 30 in stream 0 using
template query59.tpl and seed 1976435349
with wss as
 (select d_week_seq,
        ss_store_sk,
        sum(case when (d_day_name='Sunday')
then ss_sales_price else null end)
sun_sales,
        sum(case when (d_day_name='Monday')
then ss_sales_price else null end)
mon_sales,
```

```
        sum(case when
(d_day_name='Tuesday') then ss_sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then
ss_sales_price else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then ss_sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then ss_sales_price else null end)
fri_sales,
        sum(case when
(d_day_name='Saturday') then ss_sales_price
else null end) sat_sales
 from store_sales,date_dim
 where d_date_sk = ss_sold_date_sk
 group by d_week_seq,ss_store_sk
 )
  select
s_store_name1,s_store_id1,d_week_seq1

,sun_sales1/sun_sales2,mon_sales1/mon_sales
2

,tue_sales1/tue_sales1,wed_sales1/wed_sales
2,thu_sales1/thu_sales2

,fri_sales1/fri_sales2,sat_sales1/sat_sales
2
 from
 (select s_store_name
s_store_name1,wss.d_week_seq d_week_seq1
        ,s_store_id s_store_id1,sun_sales
sun_sales1
        ,mon_sales mon_sales1,tue_sales
tue_sales1
        ,wed_sales wed_sales1,thu_sales
thu_sales1
        ,fri_sales fri_sales1,sat_sales
sat_sales1
  from wss,store,date_dim d
  where d.d_week_seq = wss.d_week_seq and
        ss_store_sk = s_store_sk and
        d_month_seq between 1189 and 1189 +
11) y,
 (select s_store_name
s_store_name2,wss.d_week_seq d_week_seq2
        ,s_store_id s_store_id2,sun_sales
sun_sales2
        ,mon_sales mon_sales2,tue_sales
tue_sales2
        ,wed_sales wed_sales2,thu_sales
thu_sales2
        ,fri_sales fri_sales2,sat_sales
sat_sales2
  from wss,store,date_dim d
  where d.d_week_seq = wss.d_week_seq and
        ss_store_sk = s_store_sk and
        d_month_seq between 1189+ 12 and
1189 + 23) x
 where s_store_id1=s_store_id2
   and d_week_seq1=d_week_seq2-52
 order by
s_store_name1,s_store_id1,d_week_seq1
 limit 100;

-- end query 30 in stream 0 using template
query59.tpl
-- start query 32 in stream 0 using
template query98.tpl and seed 1900673199
select i_item_desc
      ,i_category
      ,i_class
      ,i_current_price
      ,itemrevenue
      ,revenueratio
from
(
select i_item_id
      ,i_item_desc
      ,i_category
      ,i_class
      ,i_current_price
      ,sum(ss_ext_sales_price) as
itemrevenue

,sum(ss_ext_sales_price)*100/sum(sum(ss_ext
_sales_price)) over
          (partition by i_class) as
revenueratio
from
       store_sales
       ,item
       ,date_dim
where
       ss_item_sk = i_item_sk
       and i_category in ('Shoes',
'Music', 'Home')
       and ss_sold_date_sk = d_date_sk
       and d_date between cast('1999-05-
25' as date)
                                  and
date_add(cast('1999-05-25' as date), 30)
group by
       i_item_id
       ,i_item_desc
       ,i_category
       ,i_class
       ,i_current_price
order by
       i_category
       ,i_class
       ,i_item_id
       ,i_item_desc
       ,revenueratio
) z;

-- end query 32 in stream 0 using template
query98.tpl
-- start query 36 in stream 0 using
template query28.tpl and seed 24799953
select  *
from (select avg(ss_list_price) B1_LP
            ,count(ss_list_price) B1_CNT
            ,count(distinct ss_list_price)
B1_CNTD
      from store_sales
      where ss_quantity between 0 and 5
        and (ss_list_price between 47 and
47+10
             or ss_coupon_amt between 11713
and 11713+1000
             or ss_wholesale_cost between
55 and 55+20)) B1,
     (select avg(ss_list_price) B2_LP
            ,count(ss_list_price) B2_CNT
            ,count(distinct ss_list_price)
B2_CNTD
      from store_sales
      where ss_quantity between 6 and 10
        and (ss_list_price between 93 and
93+10
          or ss_coupon_amt between 7733 and
7733+1000
          or ss_wholesale_cost between 43
and 43+20)) B2,
     (select avg(ss_list_price) B3_LP
            ,count(ss_list_price) B3_CNT
            ,count(distinct ss_list_price)
B3_CNTD
      from store_sales
      where ss_quantity between 11 and 15
```

```
        and (ss_list_price between 32 and
32+10
          or ss_coupon_amt between 11517
and 11517+1000
          or ss_wholesale_cost between 26
and 26+20)) B3,
     (select avg(ss_list_price) B4_LP
            ,count(ss_list_price) B4_CNT
            ,count(distinct ss_list_price)
B4_CNTD
      from store_sales
      where ss_quantity between 16 and 20
        and (ss_list_price between 147 and
147+10
          or ss_coupon_amt between 509 and
509+1000
          or ss_wholesale_cost between 78
and 78+20)) B4,
     (select avg(ss_list_price) B5_LP
            ,count(ss_list_price) B5_CNT
            ,count(distinct ss_list_price)
B5_CNTD
      from store_sales
      where ss_quantity between 21 and 25
        and (ss_list_price between 16 and
16+10
          or ss_coupon_amt between 2401 and
2401+1000
          or ss_wholesale_cost between 32
and 32+20)) B5,
     (select avg(ss_list_price) B6_LP
            ,count(ss_list_price) B6_CNT
            ,count(distinct ss_list_price)
B6_CNTD
      from store_sales
      where ss_quantity between 26 and 30
        and (ss_list_price between 11 and
11+10
          or ss_coupon_amt between 916 and
916+1000
          or ss_wholesale_cost between 6
and 6+20)) B6
 limit 100;

-- end query 36 in stream 0 using template
query28.tpl
-- start query 39 in stream 0 using
template query66.tpl and seed 1688498284
select
         w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
        ,ship_carriers
        ,year
       ,sum(jan_sales) as jan_sales
       ,sum(feb_sales) as feb_sales
       ,sum(mar_sales) as mar_sales
       ,sum(apr_sales) as apr_sales
       ,sum(may_sales) as may_sales
       ,sum(jun_sales) as jun_sales
       ,sum(jul_sales) as jul_sales
       ,sum(aug_sales) as aug_sales
       ,sum(sep_sales) as sep_sales
       ,sum(oct_sales) as oct_sales
       ,sum(nov_sales) as nov_sales
       ,sum(dec_sales) as dec_sales
       ,sum(jan_sales/w_warehouse_sq_ft)
as jan_sales_per_sq_foot
       ,sum(feb_sales/w_warehouse_sq_ft)
as feb_sales_per_sq_foot
       ,sum(mar_sales/w_warehouse_sq_ft)
as mar_sales_per_sq_foot
       ,sum(apr_sales/w_warehouse_sq_ft)
as apr_sales_per_sq_foot
       ,sum(may_sales/w_warehouse_sq_ft)
as may_sales_per_sq_foot
       ,sum(jun_sales/w_warehouse_sq_ft)
as jun_sales_per_sq_foot
       ,sum(jul_sales/w_warehouse_sq_ft)
as jul_sales_per_sq_foot
       ,sum(aug_sales/w_warehouse_sq_ft)
as aug_sales_per_sq_foot
       ,sum(sep_sales/w_warehouse_sq_ft)
as sep_sales_per_sq_foot
       ,sum(oct_sales/w_warehouse_sq_ft)
as oct_sales_per_sq_foot
       ,sum(nov_sales/w_warehouse_sq_ft)
as nov_sales_per_sq_foot
       ,sum(dec_sales/w_warehouse_sq_ft)
as dec_sales_per_sq_foot
       ,sum(jan_net) as jan_net
       ,sum(feb_net) as feb_net
       ,sum(mar_net) as mar_net
       ,sum(apr_net) as apr_net
       ,sum(may_net) as may_net
       ,sum(jun_net) as jun_net
       ,sum(jul_net) as jul_net
       ,sum(aug_net) as aug_net
       ,sum(sep_net) as sep_net
       ,sum(oct_net) as oct_net
       ,sum(nov_net) as nov_net
       ,sum(dec_net) as dec_net
 from (
   select
       w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,concat('MSC', ',', 'USPS') as
ship_carriers
      ,d_year as year
       ,sum(case when d_moy = 1
                then ws_sales_price*
ws_quantity else 0 end) as jan_sales
       ,sum(case when d_moy = 2
                then ws_sales_price*
ws_quantity else 0 end) as feb_sales
       ,sum(case when d_moy = 3
                then ws_sales_price*
ws_quantity else 0 end) as mar_sales
       ,sum(case when d_moy = 4
                then ws_sales_price*
ws_quantity else 0 end) as apr_sales
       ,sum(case when d_moy = 5
                then ws_sales_price*
ws_quantity else 0 end) as may_sales
       ,sum(case when d_moy = 6
                then ws_sales_price*
ws_quantity else 0 end) as jun_sales
       ,sum(case when d_moy = 7
                then ws_sales_price*
ws_quantity else 0 end) as jul_sales
       ,sum(case when d_moy = 8
                then ws_sales_price*
ws_quantity else 0 end) as aug_sales
       ,sum(case when d_moy = 9
                then ws_sales_price*
ws_quantity else 0 end) as sep_sales
       ,sum(case when d_moy = 10
                then ws_sales_price*
ws_quantity else 0 end) as oct_sales
       ,sum(case when d_moy = 11
                then ws_sales_price*
ws_quantity else 0 end) as nov_sales
       ,sum(case when d_moy = 12
                then ws_sales_price*
ws_quantity else 0 end) as dec_sales
       ,sum(case when d_moy = 1
```

```
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jan_net
       ,sum(case when d_moy = 2
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as feb_net
       ,sum(case when d_moy = 3
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as mar_net
       ,sum(case when d_moy = 4
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as apr_net
       ,sum(case when d_moy = 5
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as may_net
       ,sum(case when d_moy = 6
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jun_net
       ,sum(case when d_moy = 7
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as jul_net
       ,sum(case when d_moy = 8
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as aug_net
       ,sum(case when d_moy = 9
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as sep_net
       ,sum(case when d_moy = 10
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as oct_net
       ,sum(case when d_moy = 11
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as nov_net
       ,sum(case when d_moy = 12
              then
ws_net_paid_inc_ship_tax * ws_quantity else
0 end) as dec_net
     from
          web_sales
         ,warehouse
         ,date_dim
         ,time_dim
          ,ship_mode
     where
            ws_warehouse_sk =
w_warehouse_sk
        and ws_sold_date_sk = d_date_sk
        and ws_sold_time_sk = t_time_sk
       and ws_ship_mode_sk =
sm_ship_mode_sk
        and d_year = 2002
        and t_time between 18036 and
18036+28800
       and sm_carrier in ('MSC','USPS')
     group by
        w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
      ,d_year
 union all
    select
       w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,concat('MSC', ',', 'USPS') as
ship_carriers
      ,d_year as year
       ,sum(case when d_moy = 1
              then cs_ext_sales_price*
cs_quantity else 0 end) as jan_sales
       ,sum(case when d_moy = 2
              then cs_ext_sales_price*
cs_quantity else 0 end) as feb_sales
       ,sum(case when d_moy = 3
              then cs_ext_sales_price*
cs_quantity else 0 end) as mar_sales
       ,sum(case when d_moy = 4
              then cs_ext_sales_price*
cs_quantity else 0 end) as apr_sales
       ,sum(case when d_moy = 5
              then cs_ext_sales_price*
cs_quantity else 0 end) as may_sales
       ,sum(case when d_moy = 6
              then cs_ext_sales_price*
cs_quantity else 0 end) as jun_sales
       ,sum(case when d_moy = 7
              then cs_ext_sales_price*
cs_quantity else 0 end) as jul_sales
       ,sum(case when d_moy = 8
              then cs_ext_sales_price*
cs_quantity else 0 end) as aug_sales
       ,sum(case when d_moy = 9
              then cs_ext_sales_price*
cs_quantity else 0 end) as sep_sales
       ,sum(case when d_moy = 10
              then cs_ext_sales_price*
cs_quantity else 0 end) as oct_sales
       ,sum(case when d_moy = 11
              then cs_ext_sales_price*
cs_quantity else 0 end) as nov_sales
       ,sum(case when d_moy = 12
              then cs_ext_sales_price*
cs_quantity else 0 end) as dec_sales
       ,sum(case when d_moy = 1
              then cs_net_profit *
cs_quantity else 0 end) as jan_net
       ,sum(case when d_moy = 2
              then cs_net_profit *
cs_quantity else 0 end) as feb_net
       ,sum(case when d_moy = 3
              then cs_net_profit *
cs_quantity else 0 end) as mar_net
       ,sum(case when d_moy = 4
              then cs_net_profit *
cs_quantity else 0 end) as apr_net
       ,sum(case when d_moy = 5
              then cs_net_profit *
cs_quantity else 0 end) as may_net
       ,sum(case when d_moy = 6
              then cs_net_profit *
cs_quantity else 0 end) as jun_net
       ,sum(case when d_moy = 7
              then cs_net_profit *
cs_quantity else 0 end) as jul_net
       ,sum(case when d_moy = 8
              then cs_net_profit *
cs_quantity else 0 end) as aug_net
       ,sum(case when d_moy = 9
              then cs_net_profit *
cs_quantity else 0 end) as sep_net
       ,sum(case when d_moy = 10
              then cs_net_profit *
cs_quantity else 0 end) as oct_net
       ,sum(case when d_moy = 11
              then cs_net_profit *
cs_quantity else 0 end) as nov_net
       ,sum(case when d_moy = 12
```

```
              then cs_net_profit *
cs_quantity else 0 end) as dec_net
     from
          catalog_sales
         ,warehouse
         ,date_dim
         ,time_dim
        ,ship_mode
     where
           cs_warehouse_sk =
w_warehouse_sk
        and cs_sold_date_sk = d_date_sk
        and cs_sold_time_sk = t_time_sk
       and cs_ship_mode_sk =
sm_ship_mode_sk
        and d_year = 2002
        and t_time between 18036 and
18036+28800
       and sm_carrier in ('MSC','USPS')
     group by
        w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
      ,d_year
 ) x
 group by
        w_warehouse_name
       ,w_warehouse_sq_ft
       ,w_city
       ,w_county
       ,w_state
       ,w_country
       ,ship_carriers
      ,year
 order by w_warehouse_name
   limit 100;

-- end query 39 in stream 0 using template
query66.tpl
-- start query 40 in stream 0 using
template query90.tpl and seed 1949014749
select  cast(amc as decimal(15,4))/cast(pmc
as decimal(15,4)) am_pm_ratio
 from ( select count(*) amc
       from web_sales,
household_demographics , time_dim, web_page
       where ws_sold_time_sk =
time_dim.t_time_sk
         and ws_ship_hdemo_sk =
household_demographics.hd_demo_sk
         and ws_web_page_sk =
web_page.wp_web_page_sk
         and time_dim.t_hour between 11 and
11+1
         and
household_demographics.hd_dep_count = 9
         and web_page.wp_char_count between
5000 and 5200) at,
      ( select count(*) pmc
       from web_sales,
household_demographics , time_dim, web_page
       where ws_sold_time_sk =
time_dim.t_time_sk
         and ws_ship_hdemo_sk =
household_demographics.hd_demo_sk
         and ws_web_page_sk =
web_page.wp_web_page_sk
         and time_dim.t_hour between 18 and
18+1
         and
household_demographics.hd_dep_count = 9
         and web_page.wp_char_count between
5000 and 5200) pt
 order by am_pm_ratio
  limit 100;

-- end query 40 in stream 0 using template
query90.tpl
-- start query 44 in stream 0 using
template query92.tpl and seed 643980925
select
   sum(ws.ws_ext_discount_amt)   as
Excess_Discount_Amount
from
    web_sales ws join item i1 on
i1.i_item_sk = ws.ws_item_sk
    join date_dim dd on dd.d_date_sk =
ws.ws_sold_date_sk
    join
      (
       SELECT
          ws_item_sk, 1.3 *
avg(ws_ext_discount_amt) as
avg_ws_ext_discount_amt
       FROM
          web_sales join date_dim on
d_date_sk = ws_sold_date_sk
       WHERE
          d_date between '1999-01-03'
          and date_add(cast('1999-01-03' as
date), 90)
          group by ws_item_sk
       ) tmp on tmp.ws_item_sk =
i1.i_item_sk
where
i1.i_manufact_id = 926
and dd.d_date between '1999-01-03'
and date_add(cast('1999-01-03' as date),
90)
and ws.ws_ext_discount_amt  >
tmp.avg_ws_ext_discount_amt
order by Excess_Discount_Amount
  limit 100;

-- end query 44 in stream 0 using template
query92.tpl
-- start query 45 in stream 0 using
template query3.tpl and seed 691662667
select  dt.d_year
       ,item.i_brand_id brand_id
       ,item.i_brand brand
       ,sum(ss_net_profit) sum_agg
 from  date_dim dt
      ,store_sales
      ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
   and store_sales.ss_item_sk =
item.i_item_sk
   and item.i_manufact_id = 596
   and dt.d_moy=12
 group by dt.d_year
      ,item.i_brand
      ,item.i_brand_id
 order by dt.d_year
         ,sum_agg desc
         ,brand_id
  limit 100;

-- end query 45 in stream 0 using template
query3.tpl
-- start query 49 in stream 0 using
template query9.tpl and seed 937436805
with temp1 as
(
  select 'bucket1' bucket, count(*) cnt,
avg(ss_ext_sales_price) avg_amt,
avg(ss_net_paid_inc_tax) avg_paid
  from store_sales
  where ss_quantity between 1 and 20
),
```

```
temp2 as
(
  select 'bucket2' bucket, count(*) cnt,
avg(ss_ext_sales_price) avg_amt,
avg(ss_net_paid_inc_tax) avg_paid
  from store_sales
  where ss_quantity between 21 and 40
),
temp3 as
(
  select 'bucket3' bucket, count(*) cnt,
avg(ss_ext_sales_price) avg_amt,
avg(ss_net_paid_inc_tax) avg_paid
  from store_sales
  where ss_quantity between 41 and 60
),
temp4 as
(
  select 'bucket4' bucket, count(*) cnt,
avg(ss_ext_sales_price) avg_amt,
avg(ss_net_paid_inc_tax) avg_paid
  from store_sales
  where ss_quantity between 61 and 80
),
temp5 as
(
  select 'bucket5' bucket, count(*) cnt,
avg(ss_ext_sales_price) avg_amt,
avg(ss_net_paid_inc_tax) avg_paid
  from store_sales
  where ss_quantity between 81 and 100
)

select
       case when (temp1.bucket = 'bucket1'
and temp1.cnt > 62316685)
            then temp1.avg_amt
            else temp1.avg_paid
            end bucket1 ,
       case when (temp2.bucket = 'bucket2'
and temp2.cnt > 19045798)
            then temp2.avg_amt
            else temp2.avg_paid
            end bucket2 ,
       case when (temp3.bucket = 'bucket3'
and temp3.cnt > 365541424)
            then temp3.avg_amt
            else temp3.avg_paid
            end bucket3 ,
       case when (temp4.bucket = 'bucket4'
and temp4.cnt > 216357808)
            then temp4.avg_amt
            else temp4.avg_paid
            end bucket4 ,
       case when (temp5.bucket = 'bucket5'
and temp5.cnt > 184483884)
            then temp5.avg_amt
            else temp5.avg_paid
            end bucket5

from temp1, temp2, temp3, temp4, temp5
;

-- end query 49 in stream 0 using template
query9.tpl
-- start query 52 in stream 0 using
template query93.tpl and seed 1821797098
select  ss_customer_sk
            ,sum(act_sales) sumsales
      from (select ss_item_sk
                  ,ss_ticket_number
                  ,ss_customer_sk
                  ,case when
sr_return_quantity is not null then
(ss_quantity-
sr_return_quantity)*ss_sales_price
else (ss_quantity*ss_sales_price) end
act_sales
            from store_sales left outer
join store_returns on (sr_item_sk =
ss_item_sk

and sr_ticket_number = ss_ticket_number)
                ,reason
            where sr_reason_sk =
r_reason_sk
              and r_reason_desc = 'reason
74') t
      group by ss_customer_sk
      order by sumsales, ss_customer_sk
 limit 100;

-- end query 52 in stream 0 using template
query93.tpl
-- start query 55 in stream 0 using
template query22.tpl and seed 635815297
select  i_product_name
             ,i_brand
             ,i_class
             ,i_category
             ,avg(inv_quantity_on_hand) qoh
       from inventory
           ,date_dim
           ,item
           ,warehouse
       where inv_date_sk=d_date_sk
              and inv_item_sk=i_item_sk
              and inv_warehouse_sk =
w_warehouse_sk
              and d_month_seq between 1199
and 1199 + 11
       group by i_product_name
                       ,i_brand
                       ,i_class
                       ,i_category WITH
ROLLUP
order by qoh, i_product_name, i_brand,
i_class, i_category
  limit 100;

-- end query 55 in stream 0 using template
query22.tpl
-- start query 56 in stream 0 using
template query89.tpl and seed 2079706651
select  *
from(
select i_category, i_class, i_brand,
       s_store_name, s_company_name,
       d_moy,
       sum(ss_sales_price) sum_sales,
       avg(sum(ss_sales_price)) over
         (partition by i_category, i_brand,
s_store_name, s_company_name)
        avg_monthly_sales
from item, store_sales, date_dim, store
where ss_item_sk = i_item_sk and
      ss_sold_date_sk = d_date_sk and
      ss_store_sk = s_store_sk and
      d_year in (1999) and
        ((i_category in
('Books','Jewelry','Men') and
          i_class in
('history','birdal','pants')
        )
      or (i_category in
('Music','Home','Shoes') and
          i_class in
('pop','furniture','athletic')
        ))
group by i_category, i_class, i_brand,
         s_store_name, s_company_name,
d_moy) tmp1
```

```
where case when (avg_monthly_sales <> 0)
then (abs(sum_sales - avg_monthly_sales) /
avg_monthly_sales) else null end > 0.1
order by sum_sales - avg_monthly_sales,
s_store_name
 limit 100;

-- end query 56 in stream 0 using template
query89.tpl
-- start query 59 in stream 0 using
template query52.tpl and seed 223505300
select  dt.d_year
       ,item.i_brand_id brand_id
       ,item.i_brand brand
       ,sum(ss_ext_sales_price) ext_price
 from date_dim dt
     ,store_sales
     ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
    and store_sales.ss_item_sk =
item.i_item_sk
    and item.i_manager_id = 1
    and dt.d_moy=11
    and dt.d_year=1999
 group by dt.d_year
       ,item.i_brand
       ,item.i_brand_id
 order by dt.d_year
       ,ext_price desc
       ,brand_id
 limit 100 ;

-- end query 59 in stream 0 using template
query52.tpl
-- start query 60 in stream 0 using
template query50.tpl and seed 1718577076
select
   s_store_name
  ,s_company_id
  ,s_street_number
  ,s_street_name
  ,s_street_type
  ,s_suite_number
  ,s_city
  ,s_county
  ,s_state
  ,s_zip
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk <= 30 ) then 1 else 0 end)
as 30_days
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 30) and
                (sr_returned_date_sk -
ss_sold_date_sk <= 60) then 1 else 0 end )
as 31_60_days
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 60) and
                (sr_returned_date_sk -
ss_sold_date_sk <= 90) then 1 else 0 end)
as 61_90_days
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk > 90) and
                (sr_returned_date_sk -
ss_sold_date_sk <= 120) then 1 else 0 end)
as 91_120_days
  ,sum(case when (sr_returned_date_sk -
ss_sold_date_sk  > 120) then 1 else 0 end)
as above120_days
from
   store_sales
  ,store_returns
  ,store
  ,date_dim d1
  ,date_dim d2
where
    d2.d_year = 1999
and d2.d_moy  = 10
and ss_ticket_number = sr_ticket_number
and ss_item_sk = sr_item_sk
and ss_sold_date_sk   = d1.d_date_sk
and sr_returned_date_sk   = d2.d_date_sk
and ss_customer_sk = sr_customer_sk
and ss_store_sk = s_store_sk
group by
   s_store_name
  ,s_company_id
  ,s_street_number
  ,s_street_name
  ,s_street_type
  ,s_suite_number
  ,s_city
  ,s_county
  ,s_state
  ,s_zip
order by s_store_name
        ,s_company_id
        ,s_street_number
        ,s_street_name
        ,s_street_type
        ,s_suite_number
        ,s_city
        ,s_county
        ,s_state
        ,s_zip
  limit 100;

-- end query 60 in stream 0 using template
query50.tpl
-- start query 61 in stream 0 using
template query42.tpl and seed 709936855
select  dt.d_year
       ,item.i_category_id
       ,item.i_category
       ,sum(ss_ext_sales_price)
sum_ss_ext_sales_price
 from  date_dim dt
       ,store_sales
       ,item
 where dt.d_date_sk =
store_sales.ss_sold_date_sk
       and store_sales.ss_item_sk =
item.i_item_sk
       and item.i_manager_id = 1
       and dt.d_moy=12
       and dt.d_year=2000
 group by      dt.d_year
               ,item.i_category_id
               ,item.i_category
 order by       --sum(ss_ext_sales_price)
desc,dt.d_year
                --4 desc,dt.d_year
                sum_ss_ext_sales_price
desc,dt.d_year
               ,item.i_category_id
               ,item.i_category
  limit 100;

-- end query 61 in stream 0 using template
query42.tpl
-- start query 62 in stream 0 using
template query41.tpl and seed 944250029
select  distinct(i_product_name)
 from item i1
 where i_manufact_id between 716 and 716+40
   and exists (
        select tmp.i_manufact from
        (
        select i_manufact
        from item
        where
        --(i_manufact = i1.i_manufact and
        ((i_category = 'Women' and
```

```
        (i_color = 'spring' or i_color =
'hot') and
        (i_units = 'Carton' or i_units =
'Tbl') and
        (i_size = 'large' or i_size =
'N/A')
        ) or
        (i_category = 'Women' and
        (i_color = 'magenta' or i_color =
'goldenrod') and
        (i_units = 'Cup' or i_units = 'Oz')
and
        (i_size = 'economy' or i_size =
'extra large')
        ) or
        (i_category = 'Men' and
        (i_color = 'cyan' or i_color =
'antique') and
        (i_units = 'Dozen' or i_units =
'Case') and
        (i_size = 'medium' or i_size =
'petite')
        ) or
        (i_category = 'Men' and
        (i_color = 'moccasin' or i_color =
'black') and
        (i_units = 'Box' or i_units =
'Pallet') and
        (i_size = 'large' or i_size =
'N/A')
        ))
        --)
        or
       --(i_manufact = i1.i_manufact and
        ((i_category = 'Women' and
        (i_color = 'azure' or i_color =
'light') and
        (i_units = 'Gross' or i_units =
'Each') and
        (i_size = 'large' or i_size =
'N/A')
        ) or
        (i_category = 'Women' and
        (i_color = 'mint' or i_color =
'burnished') and
        (i_units = 'N/A' or i_units =
'Unknown') and
        (i_size = 'economy' or i_size =
'extra large')
        ) or
        (i_category = 'Men' and
        (i_color = 'floral' or i_color =
'midnight') and
        (i_units = 'Pound' or i_units =
'Ton') and
        (i_size = 'medium' or i_size =
'petite')
        ) or
        (i_category = 'Men' and
        (i_color = 'navy' or i_color =
'blue') and
        (i_units = 'Bundle' or i_units =
'Ounce') and
        (i_size = 'large' or i_size =
'N/A')
        ))
        --)
        ) tmp where tmp.i_manufact =
i1.i_manufact )
 order by i_product_name
   limit 100;

-- end query 62 in stream 0 using template
query41.tpl
-- start query 64 in stream 0 using
template query12.tpl and seed 918962166
select  i_item_desc
      ,i_category
      ,i_class
      ,i_current_price
      ,itemrevenue
      ,revenueratio
from (
select i_item_id
      ,i_item_desc
      ,i_category
      ,i_class
      ,i_current_price
      ,sum(ws_ext_sales_price) as
itemrevenue

,sum(ws_ext_sales_price)*100/sum(sum(ws_ext
_sales_price)) over
          (partition by i_class) as
revenueratio
from
       web_sales
       ,item
       ,date_dim
where
       ws_item_sk = i_item_sk
       and i_category in ('Jewelry',
'Men', 'Books')
       and ws_sold_date_sk = d_date_sk
       and d_date between cast('2002-06-
11' as date)
                                  and
date_add(cast('2002-06-11' as date), 30)
group by
       i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price
order by
       i_category
        ,i_class
        ,i_item_id
        ,i_item_desc
        ,revenueratio
) v1
  limit 100;

-- end query 64 in stream 0 using template
query12.tpl
-- start query 65 in stream 0 using
template query20.tpl and seed 711739272
select  v1.i_item_desc
       ,v1.i_category
       ,v1.i_class
       ,v1.i_current_price
       ,v1.itemrevenue
       ,v1.revenueratio
from
(
select  i_item_id, i_item_desc
       ,i_category
       ,i_class
       ,i_current_price
       ,sum(cs_ext_sales_price) as
itemrevenue

,sum(cs_ext_sales_price)*100/sum(sum(cs_ext
_sales_price)) over
           (partition by i_class) as
revenueratio
 from   catalog_sales
     ,item
     ,date_dim
 where cs_item_sk = i_item_sk
   and i_category in ('Jewelry', 'Music',
'Men')
   and cs_sold_date_sk = d_date_sk
```

```
 and d_date between cast('2000-02-09' as
date)
                 and date_add(cast('2000-
02-09' as date), 30)
 group by i_item_id
         ,i_item_desc
         ,i_category
         ,i_class
         ,i_current_price
 order by i_category
         ,i_class
         ,i_item_id
         ,i_item_desc
         ,revenueratio
) v1
  limit 100
;

-- end query 65 in stream 0 using template
query20.tpl
-- start query 66 in stream 0 using
template query88.tpl and seed 1924183468
select  *
from
 (select count(*) h8_30_to_9
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 8
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s1,
 (select count(*) h9_to_9_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 9
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s2,
 (select count(*) h9_30_to_10
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 9
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s3,
 (select count(*) h10_to_10_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 10
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s4,
 (select count(*) h10_30_to_11
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 10
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s5,
 (select count(*) h11_to_11_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
```

```
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 11
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s6,
 (select count(*) h11_30_to_12
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 11
     and time_dim.t_minute >= 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s7,
 (select count(*) h12_to_12_30
 from store_sales, household_demographics ,
time_dim, store
 where ss_sold_time_sk = time_dim.t_time_sk
     and ss_hdemo_sk =
household_demographics.hd_demo_sk
     and ss_store_sk = s_store_sk
     and time_dim.t_hour = 12
     and time_dim.t_minute < 30
     and
((household_demographics.hd_dep_count = 1
and
household_demographics.hd_vehicle_count<=1+
2) or

(household_demographics.hd_dep_count = 4
and
household_demographics.hd_vehicle_count<=4+
2) or

(household_demographics.hd_dep_count = 2
and
household_demographics.hd_vehicle_count<=2+
2))
     and store.s_store_name = 'ese') s8
;

-- end query 66 in stream 0 using template
query88.tpl
-- start query 72 in stream 0 using
template query71.tpl and seed 1436004490
select i_brand_id brand_id, i_brand
brand,t_hour,t_minute,
       sum(ext_price) ext_price
 from item, (select ws_ext_sales_price as
ext_price,
                        ws_sold_date_sk as
sold_date_sk,
                        ws_item_sk as
sold_item_sk,
                        ws_sold_time_sk as
time_sk
                from web_sales,date_dim
                where d_date_sk =
ws_sold_date_sk
                  and d_moy=12
                  and d_year=1998
                union all
                select cs_ext_sales_price
as ext_price,
                        cs_sold_date_sk as
sold_date_sk,
                        cs_item_sk as
sold_item_sk,
                        cs_sold_time_sk as
time_sk
                from
catalog_sales,date_dim
                where d_date_sk =
cs_sold_date_sk
                  and d_moy=12
                  and d_year=1998
                union all
                select ss_ext_sales_price
as ext_price,
                        ss_sold_date_sk as
sold_date_sk,
                        ss_item_sk as
sold_item_sk,
                        ss_sold_time_sk as
time_sk
                from store_sales,date_dim
                where d_date_sk =
ss_sold_date_sk
                  and d_moy=12
                  and d_year=1998
                ) as tmp,time_dim
 where
   sold_item_sk = i_item_sk
   and i_manager_id=1
   and time_sk = t_time_sk
   and (t_meal_time = 'breakfast' or
t_meal_time = 'dinner')
 group by i_brand,
i_brand_id,t_hour,t_minute
 order by ext_price desc, i_brand_id
 ;

-- end query 72 in stream 0 using template
query71.tpl
-- start query 73 in stream 0 using
template query34.tpl and seed 1451328249
select c_last_name
       ,c_first_name
       ,c_salutation
       ,c_preferred_cust_flag
       ,ss_ticket_number
       ,cnt from
   (select ss_ticket_number
          ,ss_customer_sk
          ,count(*) cnt
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
```

```
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and (date_dim.d_dom between 1 and 3 or
date_dim.d_dom between 25 and 28)
    and
(household_demographics.hd_buy_potential =
'1001-5000' or

household_demographics.hd_buy_potential =
'5001-10000')
    and
household_demographics.hd_vehicle_count > 0
    and (case when
household_demographics.hd_vehicle_count > 0
       then
household_demographics.hd_dep_count/
household_demographics.hd_vehicle_count
       else null
       end)  > 1.2
    and date_dim.d_year in
(1999,1999+1,1999+2)
    and (
    store.s_county = 'Sierra County'
    or store.s_county = 'Lunenburg County'
    or store.s_county = 'Jackson County'
    or store.s_county = 'Harmon County'
    or store.s_county = 'Mesa County'
    or store.s_county = 'Pipestone County'
    or store.s_county = 'Pennington County'
    or store.s_county = 'Perry County')
    group by
ss_ticket_number,ss_customer_sk)
dn,customer
    where ss_customer_sk = c_customer_sk
      and cnt between 15 and 20
    order by
c_last_name,c_first_name,c_salutation,c_pre
ferred_cust_flag desc;

-- end query 73 in stream 0 using template
query34.tpl
-- start query 78 in stream 0 using
template query77.tpl and seed 1879081522
with ss as
 (select s_store_sk,
         sum(ss_ext_sales_price) as sales,
         sum(ss_net_profit) as profit
 from store_sales,
      date_dim,
      store
 where ss_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and date_add(cast('2002-
08-24' as date), 30)
       and ss_store_sk = s_store_sk
 group by s_store_sk)
 ,
 sr as
 (select s_store_sk,
         sum(sr_return_amt) as returns,
         sum(sr_net_loss) as profit_loss
 from store_returns,
      date_dim,
      store
 where sr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and date_add(cast('2002-
08-24' as date), 30)
       and sr_store_sk = s_store_sk
 group by s_store_sk),
 cs as
 (select cs_call_center_sk,
        sum(cs_ext_sales_price) as sales,
        sum(cs_net_profit) as profit
 from catalog_sales,
      date_dim
 where cs_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and date_add(cast('2002-
08-24' as date), 30)
 group by cs_call_center_sk
 ),
 cr as
 (select
        sum(cr_return_amount) as returns,
        sum(cr_net_loss) as profit_loss
 from catalog_returns,
      date_dim
 where cr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and date_add(cast('2002-
08-24' as date), 30)
 ),
 ws as
 ( select wp_web_page_sk,
        sum(ws_ext_sales_price) as sales,
        sum(ws_net_profit) as profit
 from web_sales,
      date_dim,
      web_page
 where ws_sold_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and date_add(cast('2002-
08-24' as date), 30)
       and ws_web_page_sk = wp_web_page_sk
 group by wp_web_page_sk),
 wr as
 (select wp_web_page_sk,
        sum(wr_return_amt) as returns,
        sum(wr_net_loss) as profit_loss
 from web_returns,
      date_dim,
      web_page
 where wr_returned_date_sk = d_date_sk
       and d_date between cast('2002-08-24'
as date)
                  and date_add(cast('2002-
08-24' as date), 30)
       and wr_web_page_sk = wp_web_page_sk
 group by wp_web_page_sk)
  select  channel
        , id
        , sum(sales) as sales
        , sum(returns) as returns
        , sum(profit) as profit
 from
 (select 'store channel' as channel
        , ss.s_store_sk as id
        , sales
        , coalesce(returns, 0) as returns
        , (profit -
coalesce(profit_loss,0)) as profit
 from   ss left join sr
        on  ss.s_store_sk = sr.s_store_sk
 union all
 select 'catalog channel' as channel
        , cs_call_center_sk as id
        , sales
        , returns
        , (profit - profit_loss) as profit
 from  cs
       , cr
 union all
 select 'web channel' as channel
        , ws.wp_web_page_sk as id
        , sales
        , coalesce(returns, 0) returns
        , (profit -
coalesce(profit_loss,0)) as profit
```

```
 from   ws left join wr
        on  ws.wp_web_page_sk =
wr.wp_web_page_sk
 ) x
 group by channel, id WITH ROLLUP
 order by channel
         ,id
   limit 100;

-- end query 78 in stream 0 using template
query77.tpl
-- start query 79 in stream 0 using
template query73.tpl and seed 413577677
select c_last_name
       ,c_first_name
       ,c_salutation
       ,c_preferred_cust_flag
       ,ss_ticket_number
       ,cnt from
   (select ss_ticket_number
          ,ss_customer_sk
          ,count(*) cnt
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and date_dim.d_dom between 1 and 2
    and
(household_demographics.hd_buy_potential =
'501-1000' or

household_demographics.hd_buy_potential =
'5001-10000')
    and
household_demographics.hd_vehicle_count > 0
    and case when
household_demographics.hd_vehicle_count > 0
then

household_demographics.hd_dep_count/
household_demographics.hd_vehicle_count
else null end > 1
    and date_dim.d_year in
(1999,1999+1,1999+2)
    and store.s_county in ('Lea
County','West Feliciana Parish','Nowata
County','Jackson County')
    group by
ss_ticket_number,ss_customer_sk)
dj,customer
    where ss_customer_sk = c_customer_sk
      and cnt between 1 and 5
    order by cnt desc;

-- end query 79 in stream 0 using template
query73.tpl
-- start query 80 in stream 0 using
template query84.tpl and seed 1842474049
select  c_customer_id as customer_id
       ,concat(c_last_name, ', ' ,
c_first_name) as customername
 from customer
     ,customer_address
     ,customer_demographics
     ,household_demographics
     ,income_band
     ,store_returns
 where ca_city         =  'Mount Zion'
   and c_current_addr_sk = ca_address_sk
   and ib_lower_bound   >=  50749
   and ib_upper_bound   <=  50749 + 50000
   and ib_income_band_sk =
hd_income_band_sk
   and cd_demo_sk = c_current_cdemo_sk
   and hd_demo_sk = c_current_hdemo_sk
   and sr_cdemo_sk = cd_demo_sk
 order by customer_id --
   limit 100;

-- end query 80 in stream 0 using template
query84.tpl
-- start query 82 in stream 0 using
template query55.tpl and seed 1117454508
select  i_brand_id brand_id, i_brand brand,
       sum(ss_ext_sales_price) ext_price
 from date_dim, store_sales, item
 where d_date_sk = ss_sold_date_sk
       and ss_item_sk = i_item_sk
       and i_manager_id=48
       and d_moy=11
       and d_year=2001
 group by i_brand, i_brand_id
 order by ext_price desc, i_brand_id
 limit 100 ;

-- end query 82 in stream 0 using template
query55.tpl
-- start query 83 in stream 0 using
template query56.tpl and seed 1152645577
with ss as (
 select i_item_id,sum(ss_ext_sales_price)
total_sales
 from
 (
 select distinct
i1.i_item_id,ss_ext_sales_price
 from
       store_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where i1.i_item_id = i2.i_item_id
 and i2.i_color in
('maroon','powder','lawn')
 and     ss_item_sk              =
i1.i_item_sk
 and     ss_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     ss_addr_sk              =
ca_address_sk
 and     ca_gmt_offset           = -5
 ) v1
 group by i_item_id),
 cs as (
 select i_item_id,sum(cs_ext_sales_price)
total_sales
 from
 (
 select distinct
i1.i_item_id,cs_ext_sales_price
 from
       catalog_sales,
       date_dim,
         customer_address,
         item i1,
         item i2
 where i1.i_item_id = i2.i_item_id
 and i2.i_color in
('maroon','powder','lawn')
 and     cs_item_sk              =
i1.i_item_sk
 and     cs_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
```

```
 and     cs_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 ) v2
 group by i_item_id),
 ws as (
 select i_item_id,sum(ws_ext_sales_price)
total_sales
 from
 (
 select distinct
i1.i_item_id,ws_ext_sales_price
 from
        web_sales,
        date_dim,
          customer_address,
          item i1,
          item i2
 where i1.i_item_id = i2.i_item_id
 and i2.i_color in
('maroon','powder','lawn')
 and     ws_item_sk              =
i1.i_item_sk
 and     ws_sold_date_sk         =
d_date_sk
 and     d_year                  = 2000
 and     d_moy                   = 1
 and     ws_bill_addr_sk         =
ca_address_sk
 and     ca_gmt_offset           = -5
 ) v3
 group by i_item_id)
  select  i_item_id ,sum(total_sales)
total_sales
 from  (select * from ss
        union all
        select * from cs
        union all
        select * from ws) tmp1
 group by i_item_id
 order by total_sales
   limit 100;

-- end query 83 in stream 0 using template
query56.tpl
-- start query 84 in stream 0 using
template query2.tpl and seed 1528114170
with wscs as
 (select sold_date_sk
        ,sales_price
  from (select ws_sold_date_sk sold_date_sk
              ,ws_ext_sales_price
sales_price
        from web_sales
        union all
        select cs_sold_date_sk sold_date_sk
              ,cs_ext_sales_price
sales_price
        from catalog_sales) x ),
 wswscs as
 (select d_week_seq,
        sum(case when (d_day_name='Sunday')
then sales_price else null end) sun_sales,
        sum(case when (d_day_name='Monday')
then sales_price else null end) mon_sales,
        sum(case when
(d_day_name='Tuesday') then sales_price
else  null end) tue_sales,
        sum(case when
(d_day_name='Wednesday') then sales_price
else null end) wed_sales,
        sum(case when
(d_day_name='Thursday') then sales_price
else null end) thu_sales,
        sum(case when (d_day_name='Friday')
then sales_price else null end) fri_sales,
        sum(case when
(d_day_name='Saturday') then sales_price
else null end) sat_sales
 from wscs
     ,date_dim
 where d_date_sk = sold_date_sk
 group by d_week_seq)
 select d_week_seq1
       ,round(sun_sales1/sun_sales2,2)
       ,round(mon_sales1/mon_sales2,2)
       ,round(tue_sales1/tue_sales2,2)
       ,round(wed_sales1/wed_sales2,2)
       ,round(thu_sales1/thu_sales2,2)
       ,round(fri_sales1/fri_sales2,2)
       ,round(sat_sales1/sat_sales2,2)
 from
 (select wswscs.d_week_seq d_week_seq1
        ,sun_sales sun_sales1
        ,mon_sales mon_sales1
        ,tue_sales tue_sales1
        ,wed_sales wed_sales1
        ,thu_sales thu_sales1
        ,fri_sales fri_sales1
        ,sat_sales sat_sales1
  from wswscs,date_dim
  where date_dim.d_week_seq =
wswscs.d_week_seq and
        d_year = 1998) y,
 (select wswscs.d_week_seq d_week_seq2
        ,sun_sales sun_sales2
        ,mon_sales mon_sales2
        ,tue_sales tue_sales2
        ,wed_sales wed_sales2
        ,thu_sales thu_sales2
        ,fri_sales fri_sales2
        ,sat_sales sat_sales2
  from wswscs
      ,date_dim
  where date_dim.d_week_seq =
wswscs.d_week_seq and
        d_year = 1998+1) z
 where d_week_seq1=d_week_seq2-53
 order by d_week_seq1;

-- end query 84 in stream 0 using template
query2.tpl
-- start query 85 in stream 0 using
template query26.tpl and seed 1427200905
select  i_item_id,
        avg(cs_quantity) agg1,
        avg(cs_list_price) agg2,
        avg(cs_coupon_amt) agg3,
        avg(cs_sales_price) agg4
 from catalog_sales, customer_demographics,
date_dim, item, promotion
 where cs_sold_date_sk = d_date_sk and
       cs_item_sk = i_item_sk and
       cs_bill_cdemo_sk = cd_demo_sk and
       cs_promo_sk = p_promo_sk and
       cd_gender = 'M' and
       cd_marital_status = 'D' and
       cd_education_status = 'Advanced
Degree' and
       (p_channel_email = 'N' or
p_channel_event = 'N') and
       d_year = 2000
 group by i_item_id
 order by i_item_id
  limit 100;

-- end query 85 in stream 0 using template
query26.tpl
-- start query 86 in stream 0 using
template query40.tpl and seed 600490395
select
   w_state
  ,i_item_id
```

```
  ,sum(case when (cast(d_date as date) <
cast ('2000-04-27' as date))
               then cs_sales_price -
coalesce(cr_refunded_cash,0) else 0 end) as
sales_before
  ,sum(case when (cast(d_date as date) >=
cast ('2000-04-27' as date))
               then cs_sales_price -
coalesce(cr_refunded_cash,0) else 0 end) as
sales_after
 from
   catalog_sales left outer join
catalog_returns on
       (cs_order_number = cr_order_number
        and cs_item_sk = cr_item_sk)
  ,warehouse
  ,item
  ,date_dim
 where
     i_current_price between 0.99 and 1.49
 and i_item_sk          = cs_item_sk
 and cs_warehouse_sk    = w_warehouse_sk
 and cs_sold_date_sk    = d_date_sk
 and d_date between date_sub(cast ('2000-
04-27' as date), 30)
                and date_add(cast ('2000-
04-27' as date), 30)
 group by
    w_state,i_item_id
 order by w_state,i_item_id
  limit 100;

-- end query 86 in stream 0 using template
query40.tpl
-- start query 88 in stream 0 using
template query53.tpl and seed 1796782974
select  * from
(select i_manufact_id,
sum(ss_sales_price) sum_sales,
avg(sum(ss_sales_price)) over (partition by
i_manufact_id) avg_quarterly_sales
from item, store_sales, date_dim, store
where ss_item_sk = i_item_sk and
ss_sold_date_sk = d_date_sk and
ss_store_sk = s_store_sk and
d_month_seq in
(1198,1198+1,1198+2,1198+3,1198+4,1198+5,11
98+6,1198+7,1198+8,1198+9,1198+10,1198+11)
and
((i_category in
('Books','Children','Electronics') and
i_class in
('personal','portable','reference','self-
help') and
i_brand in ('scholaramalgamalg
#14','scholaramalgamalg #7',
               'exportiunivamalg
#9','scholaramalgamalg #9'))
or(i_category in ('Women','Music','Men')
and
i_class in
('accessories','classical','fragrances','pa
nts') and
i_brand in ('amalgimporto #1','edu
packscholar #1','exportiimporto #1',
               'importoamalg #1')))
group by i_manufact_id, d_qoy ) tmp1
where case when avg_quarterly_sales > 0
       then abs (sum_sales -
avg_quarterly_sales)/ avg_quarterly_sales
       else null end > 0.1
order by avg_quarterly_sales,
         sum_sales,
         i_manufact_id
 limit 100;

-- end query 88 in stream 0 using template
query53.tpl
-- start query 89 in stream 0 using
template query79.tpl and seed 2112737383
select

c_last_name,c_first_name,substr(s_city,1,30
) s_city_part ,ss_ticket_number,amt,profit
  from
   (select ss_ticket_number
          ,ss_customer_sk
          ,store.s_city
          ,sum(ss_coupon_amt) amt
          ,sum(ss_net_profit) profit
    from
store_sales,date_dim,store,household_demogr
aphics
    where store_sales.ss_sold_date_sk =
date_dim.d_date_sk
    and store_sales.ss_store_sk =
store.s_store_sk
    and store_sales.ss_hdemo_sk =
household_demographics.hd_demo_sk
    and
(household_demographics.hd_dep_count = 3 or
household_demographics.hd_vehicle_count >
1)
    and date_dim.d_dow = 1
    and date_dim.d_year in
(2000,2000+1,2000+2)
    and store.s_number_employees between
200 and 295
    group by
ss_ticket_number,ss_customer_sk,ss_addr_sk,
store.s_city) ms,customer
    where ss_customer_sk = c_customer_sk
 order by
c_last_name,c_first_name,s_city_part,
profit
  limit 100;

-- end query 89 in stream 0 using template
query79.tpl
-- start query 96 in stream 0 using
template query83.tpl and seed 593789178
with sr_items as
 (select i_item_id item_id,
        sum(sr_return_quantity) sr_item_qty
 from store_returns
      JOIN item ON sr_item_sk = i_item_sk
      JOIN date_dim dd0 ON
sr_returned_date_sk   = dd0.d_date_sk
      JOIN
        (select dd1.d_date
        from date_dim dd1
             JOIN  date_dim dd2 ON
dd1.d_week_seq = dd2.d_week_seq
        where dd2.d_date in ('1999-06-
14','1999-08-26','1999-11-06')) v1 ON
dd0.d_date    = v1.d_date
 group by i_item_id),
cr_items as
 (select i_item_id item_id,
        sum(cr_return_quantity) cr_item_qty
 from catalog_returns
      JOIN item ON cr_item_sk = i_item_sk
      JOIN date_dim dd0 ON
cr_returned_date_sk   = dd0.d_date_sk
      JOIN
        (select dd1.d_date
        from date_dim dd1
             JOIN  date_dim dd2 ON
dd1.d_week_seq = dd2.d_week_seq
        where dd2.d_date in ('1999-06-
14','1999-08-26','1999-11-06')) v1 ON
dd0.d_date    = v1.d_date
 group by i_item_id),
```

```
wr_items as
 (select i_item_id item_id,
        sum(wr_return_quantity) wr_item_qty
 from web_returns
      JOIN item ON wr_item_sk = i_item_sk
      JOIN date_dim dd0 ON
wr_returned_date_sk   = dd0.d_date_sk
      JOIN
        (select dd1.d_date
        from date_dim dd1
             JOIN  date_dim dd2 ON
dd1.d_week_seq = dd2.d_week_seq
        where dd2.d_date in ('1999-06-
14','1999-08-26','1999-11-06')) v1 ON
dd0.d_date    = v1.d_date
 group by i_item_id)
  select  sr_items.item_id
       ,sr_item_qty

,sr_item_qty/(sr_item_qty+cr_item_qty+wr_it
em_qty)/3.0 * 100 sr_dev
       ,cr_item_qty

,cr_item_qty/(sr_item_qty+cr_item_qty+wr_it
em_qty)/3.0 * 100 cr_dev
       ,wr_item_qty

,wr_item_qty/(sr_item_qty+cr_item_qty+wr_it
em_qty)/3.0 * 100 wr_dev

,(sr_item_qty+cr_item_qty+wr_item_qty)/3.0
average
 from sr_items
     ,cr_items
     ,wr_items
 where sr_items.item_id=cr_items.item_id
   and sr_items.item_id=wr_items.item_id
 order by sr_items.item_id
         ,sr_item_qty
   limit 100;

-- end query 96 in stream 0 using template
query83.tpl
-- start query 97 in stream 0 using
template query61.tpl and seed 1770420976
select  promotions,total,cast(promotions as
decimal(15,4))/cast(total as
decimal(15,4))*100
from
  (select sum(ss_ext_sales_price)
promotions
   from  store_sales
        ,store
        ,promotion
        ,date_dim
        ,customer
        ,customer_address
        ,item
   where ss_sold_date_sk = d_date_sk
   and   ss_store_sk = s_store_sk
   and   ss_promo_sk = p_promo_sk
   and   ss_customer_sk= c_customer_sk
   and   ca_address_sk = c_current_addr_sk
   and   ss_item_sk = i_item_sk
   and   ca_gmt_offset = -7
   and   i_category = 'Electronics'
   and   (p_channel_dmail = 'Y' or
p_channel_email = 'Y' or p_channel_tv =
'Y')
   and   s_gmt_offset = -7
   and   d_year = 1999
   and   d_moy  = 12) promotional_sales,
  (select sum(ss_ext_sales_price) total
   from  store_sales
        ,store
        ,date_dim
        ,customer
        ,customer_address
        ,item
   where ss_sold_date_sk = d_date_sk
   and   ss_store_sk = s_store_sk
   and   ss_customer_sk= c_customer_sk
   and   ca_address_sk = c_current_addr_sk
   and   ss_item_sk = i_item_sk
   and   ca_gmt_offset = -7
   and   i_category = 'Electronics'
   and   s_gmt_offset = -7
   and   d_year = 1999
   and   d_moy  = 12) all_sales
order by promotions, total
 limit 100;

-- end query 97 in stream 0 using template
query61.tpl
-- start query 99 in stream 0 using
template query76.tpl and seed 945056756
select  channel, col_name, d_year, d_qoy,
i_category, COUNT(*) sales_cnt,
SUM(ext_sales_price) sales_amt FROM (
        SELECT 'store' as channel,
'ss_hdemo_sk' col_name, d_year, d_qoy,
i_category, ss_ext_sales_price
ext_sales_price
         FROM store_sales, item, date_dim
         WHERE ss_hdemo_sk IS NULL
           AND ss_sold_date_sk=d_date_sk
           AND ss_item_sk=i_item_sk
        UNION ALL
        SELECT 'web' as channel,
'ws_web_page_sk' col_name, d_year, d_qoy,
i_category, ws_ext_sales_price
ext_sales_price
         FROM web_sales, item, date_dim
         WHERE ws_web_page_sk IS NULL
           AND ws_sold_date_sk=d_date_sk
           AND ws_item_sk=i_item_sk
        UNION ALL
        SELECT 'catalog' as channel,
'cs_ship_addr_sk' col_name, d_year, d_qoy,
i_category, cs_ext_sales_price
ext_sales_price
         FROM catalog_sales, item, date_dim
         WHERE cs_ship_addr_sk IS NULL
           AND cs_sold_date_sk=d_date_sk
           AND cs_item_sk=i_item_sk) foo
GROUP BY channel, col_name, d_year, d_qoy,
i_category
ORDER BY channel, col_name, d_year, d_qoy,
i_category
 limit 100;

-- end query 99 in stream 0 using template
query76.tpl
```

Appendix F: Load & Analayze Scripts:

This appendix contains all scripts used during the load phase of the benchmark.

F.1 Big SQL Load & Analyze scripts:

Load:

```
set schema $schema;

SET HADOOP PROPERTY 'dfs.blocksize'= 536870912;
SET HADOOP PROPERTY 'parquet.block.size' = 536870912;

load hadoop using file url '/HADOOPDS10000G_PARQ/call_center' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
call_center overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/catalog_page' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
catalog_page overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/catalog_returns' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
catalog_returns overwrite WITH LOAD PROPERTIES ('num.map.tasks'='425');

load hadoop using file url '/HADOOPDS10000G_PARQ/catalog_sales' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
catalog_sales overwrite WITH LOAD PROPERTIES ('num.map.tasks'='4250');

load hadoop using file url '/HADOOPDS10000G_PARQ/customer_demographics' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_demographics overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/date_dim' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table date_dim overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/household_demographics' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
household_demographics overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/income_band' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
income_band overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/item' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table item overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/promotion' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table promotion
overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/reason' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table reason overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');
```

```
load hadoop using file url '/HADOOPDS10000G_PARQ/ship_mode' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table ship_mode
overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/store' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table store overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/store_returns' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
store_returns overwrite WITH LOAD PROPERTIES ('num.map.tasks'='700');

load hadoop using file url '/HADOOPDS10000G_PARQ/store_sales' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
store_sales overwrite WITH LOAD PROPERTIES ('num.map.tasks'='5500');

load hadoop using file url '/HADOOPDS10000G_PARQ/time_dim' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table time_dim overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/warehouse/' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table warehouse
overwrite WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/web_page' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table web_page overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');

load hadoop using file url '/HADOOPDS10000G_PARQ/web_returns' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
web_returns overwrite WITH LOAD PROPERTIES ('num.map.tasks'='200');

load hadoop using file url '/HADOOPDS10000G_PARQ/web_sales' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table web_sales
overwrite WITH LOAD PROPERTIES ('num.map.tasks'='2000');

load hadoop using file url '/HADOOPDS10000G_PARQ/web_site' with source properties
('field.delimiter'='|', 'ignore.extra.fields'='true') into table web_site overwrite
WITH LOAD PROPERTIES ('num.map.tasks'='1');


set schema $schema;

SET HADOOP PROPERTY 'dfs.blocksize'= 536870912;
SET HADOOP PROPERTY 'parquet.block.size' = 536870912;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_10_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address overwrite ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_11_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_12_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_13_16.dat' with source
```

```
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_14_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_15_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_16_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_1_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_2_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_3_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_4_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_5_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_6_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_7_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_8_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;
load hadoop using file url
'/HADOOPDS10000G_PARQ/customer_address/customer_address_9_16.dat' with source
properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer_address append ;

set schema $schema;

SET HADOOP PROPERTY 'dfs.blocksize'= 536870912;
SET HADOOP PROPERTY 'parquet.block.size' = 536870912;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_10_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer overwrite ;
```

```
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_11_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_12_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_13_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_14_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_15_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_16_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_1_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_2_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_3_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_4_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_5_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_6_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_7_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_8_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/customer/customer_9_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
customer append ;


set schema $schema;

SET HADOOP PROPERTY 'dfs.blocksize'= 536870912;
SET HADOOP PROPERTY 'parquet.block.size' = 536870912;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_10_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory overwrite ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_11_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory append ;
```

```
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_12_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_13_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_14_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_15_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_16_16.dat'
with source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into
table inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_1_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_2_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_3_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_4_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_5_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_6_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_7_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_8_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
load hadoop using file url '/HADOOPDS10000G_PARQ/inventory/inventory_9_16.dat' with
source properties ('field.delimiter'='|', 'ignore.extra.fields'='true') into table
inventory append ;
```

Analyze:

```
set schema $schema;

ANALYZE TABLE call_center COMPUTE STATISTICS FOR COLUMNS cc_call_center_sk,
cc_call_center_id, cc_rec_start_date, cc_rec_end_date, cc_closed_date_sk,
cc_open_date_sk, cc_name, cc_class, cc_employees, cc_sq_ft, cc_hours, cc_manager,
cc_mkt_id, cc_mkt_class, cc_mkt_desc, cc_market_manager, cc_division,
cc_division_name, cc_company, cc_company_name, cc_street_number, cc_street_name,
cc_street_type, cc_suite_number, cc_city, cc_county, cc_state, cc_zip, cc_country,
cc_gmt_offset, cc_tax_percentage;

ANALYZE TABLE catalog_page COMPUTE STATISTICS FOR COLUMNS cp_catalog_page_sk,
cp_catalog_page_id, cp_start_date_sk, cp_end_date_sk, cp_department,
cp_catalog_number, cp_catalog_page_number, cp_description, cp_type;

ANALYZE TABLE catalog_returns COMPUTE STATISTICS FOR COLUMNS cr_returned_date_sk,
cr_returned_time_sk, cr_item_sk, cr_refunded_customer_sk, cr_refunded_cdemo_sk,
```

```
cr_refunded_hdemo_sk, cr_refunded_addr_sk, cr_returning_customer_sk,
cr_returning_cdemo_sk, cr_returning_hdemo_sk, cr_returning_addr_sk,
cr_call_center_sk, cr_catalog_page_sk, cr_ship_mode_sk, cr_warehouse_sk,
cr_reason_sk, cr_order_number, cr_return_quantity, cr_return_amount, cr_return_tax,
cr_return_amt_inc_tax, cr_fee, cr_return_ship_cost, cr_refunded_cash,
cr_reversed_charge, cr_store_credit, cr_net_loss;

ANALYZE TABLE catalog_sales COMPUTE STATISTICS FOR COLUMNS cs_sold_date_sk,
cs_sold_time_sk, cs_ship_date_sk, cs_bill_customer_sk, cs_bill_cdemo_sk,
cs_bill_hdemo_sk, cs_bill_addr_sk, cs_ship_customer_sk, cs_ship_cdemo_sk,
cs_ship_hdemo_sk, cs_ship_addr_sk, cs_call_center_sk, cs_catalog_page_sk,
cs_ship_mode_sk, cs_warehouse_sk, cs_item_sk, cs_promo_sk, cs_order_number,
cs_quantity, cs_wholesale_cost, cs_list_price, cs_sales_price, cs_ext_discount_amt,
cs_ext_sales_price, cs_ext_wholesale_cost, cs_ext_list_price, cs_ext_tax,
cs_coupon_amt, cs_ext_ship_cost, cs_net_paid, cs_net_paid_inc_tax,
cs_net_paid_inc_ship, cs_net_paid_inc_ship_tax, cs_net_profit;

ANALYZE TABLE customer COMPUTE STATISTICS FOR COLUMNS c_customer_sk, c_customer_id,
c_current_cdemo_sk, c_current_hdemo_sk, c_current_addr_sk, c_first_shipto_date_sk,
c_first_sales_date_sk, c_salutation, c_first_name, c_last_name,
c_preferred_cust_flag, c_birth_day, c_birth_month, c_birth_year, c_birth_country,
c_login, c_email_address, c_last_review_date;

ANALYZE TABLE customer_address COMPUTE STATISTICS FOR COLUMNS ca_address_sk,
ca_address_id, ca_street_number, ca_street_name, ca_street_type, ca_suite_number,
ca_city, ca_county, ca_state, ca_zip, ca_country, ca_gmt_offset, ca_location_type;

ANALYZE TABLE customer_demographics COMPUTE STATISTICS FOR COLUMNS cd_demo_sk,
cd_gender, cd_marital_status, cd_education_status, cd_purchase_estimate,
cd_credit_rating, cd_dep_count, cd_dep_employed_count, cd_dep_college_count;

ANALYZE TABLE date_dim COMPUTE STATISTICS FOR COLUMNS d_date_sk, d_date_id, d_date,
d_month_seq, d_week_seq, d_quarter_seq, d_year, d_dow, d_moy, d_dom, d_qoy,
d_fy_year, d_fy_quarter_seq, d_fy_week_seq, d_day_name, d_quarter_name, d_holiday,
d_weekend, d_following_holiday, d_first_dom, d_last_dom, d_same_day_ly,
d_same_day_lq, d_current_day, d_current_week, d_current_month, d_current_quarter,
d_current_year;

ANALYZE TABLE household_demographics COMPUTE STATISTICS FOR COLUMNS hd_demo_sk,
hd_income_band_sk, hd_buy_potential, hd_dep_count, hd_vehicle_count;

ANALYZE TABLE income_band COMPUTE STATISTICS FOR COLUMNS ib_income_band_sk,
ib_lower_bound, ib_upper_bound;

ANALYZE TABLE inventory COMPUTE STATISTICS FOR COLUMNS inv_date_sk, inv_item_sk,
inv_warehouse_sk, inv_quantity_on_hand;

ANALYZE TABLE item COMPUTE STATISTICS FOR COLUMNS i_item_sk, i_item_id,
i_rec_start_date, i_rec_end_date, i_item_desc, i_current_price, i_wholesale_cost,
i_brand_id, i_brand, i_class_id, i_class, i_category_id, i_category, i_manufact_id,
i_manufact, i_size, i_formulation, i_color, i_units, i_container, i_manager_id,
i_product_name;

ANALYZE TABLE promotion COMPUTE STATISTICS FOR COLUMNS p_promo_sk, p_promo_id,
p_start_date_sk, p_end_date_sk, p_item_sk, p_cost, p_response_target, p_promo_name,
p_channel_dmail, p_channel_email, p_channel_catalog, p_channel_tv, p_channel_radio,
p_channel_press, p_channel_event, p_channel_demo, p_channel_details, p_purpose,
p_discount_active;

ANALYZE TABLE reason COMPUTE STATISTICS FOR COLUMNS r_reason_sk, r_reason_id,
r_reason_desc;

ANALYZE TABLE ship_mode COMPUTE STATISTICS FOR COLUMNS sm_ship_mode_sk,
sm_ship_mode_id, sm_type, sm_code, sm_carrier, sm_contract;

ANALYZE TABLE store COMPUTE STATISTICS FOR COLUMNS s_store_sk, s_store_id,
s_rec_start_date, s_rec_end_date, s_closed_date_sk, s_store_name,
s_number_employees, s_floor_space, s_hours, s_manager, s_market_id,
```

```
s_geography_class, s_market_desc, s_market_manager, s_division_id, s_division_name,
s_company_id, s_company_name, s_street_number, s_street_name, s_street_type,
s_suite_number, s_city, s_county, s_state, s_zip, s_country, s_gmt_offset,
s_tax_precentage;

ANALYZE TABLE store_returns COMPUTE STATISTICS FOR COLUMNS sr_returned_date_sk,
sr_return_time_sk, sr_item_sk, sr_customer_sk, sr_cdemo_sk, sr_hdemo_sk,
sr_addr_sk, sr_store_sk, sr_reason_sk, sr_ticket_number, sr_return_quantity,
sr_return_amt, sr_return_tax, sr_return_amt_inc_tax, sr_fee, sr_return_ship_cost,
sr_refunded_cash, sr_reversed_charge, sr_store_credit, sr_net_loss;

ANALYZE TABLE store_sales COMPUTE STATISTICS FOR COLUMNS ss_sold_date_sk,
ss_sold_time_sk, ss_item_sk, ss_customer_sk, ss_cdemo_sk, ss_hdemo_sk, ss_addr_sk,
ss_store_sk, ss_promo_sk, ss_ticket_number, ss_quantity, ss_wholesale_cost,
ss_list_price, ss_sales_price, ss_ext_discount_amt, ss_ext_sales_price,
ss_ext_wholesale_cost, ss_ext_list_price, ss_ext_tax, ss_coupon_amt, ss_net_paid,
ss_net_paid_inc_tax, ss_net_profit;

ANALYZE TABLE time_dim COMPUTE STATISTICS FOR COLUMNS t_time_sk, t_time_id, t_time,
t_hour, t_minute, t_second, t_am_pm, t_shift, t_sub_shift, t_meal_time;

ANALYZE TABLE warehouse COMPUTE STATISTICS FOR COLUMNS w_warehouse_sk,
w_warehouse_id, w_warehouse_name, w_warehouse_sq_ft, w_street_number,
w_street_name, w_street_type, w_suite_number, w_city, w_county, w_state, w_zip,
w_country, w_gmt_offset;

ANALYZE TABLE web_page COMPUTE STATISTICS FOR COLUMNS wp_web_page_sk,
wp_web_page_id, wp_rec_start_date, wp_rec_end_date, wp_creation_date_sk,
wp_access_date_sk, wp_autogen_flag, wp_customer_sk, wp_url, wp_type, wp_char_count,
wp_link_count, wp_image_count, wp_max_ad_count;

ANALYZE TABLE web_returns COMPUTE STATISTICS FOR COLUMNS wr_returned_date_sk,
wr_returned_time_sk, wr_item_sk, wr_refunded_customer_sk, wr_refunded_cdemo_sk,
wr_refunded_hdemo_sk, wr_refunded_addr_sk, wr_returning_customer_sk,
wr_returning_cdemo_sk, wr_returning_hdemo_sk, wr_returning_addr_sk, wr_web_page_sk,
wr_reason_sk, wr_order_number, wr_return_quantity, wr_return_amt, wr_return_tax,
wr_return_amt_inc_tax, wr_fee, wr_return_ship_cost, wr_refunded_cash,
wr_reversed_charge, wr_account_credit, wr_net_loss;

ANALYZE TABLE web_sales COMPUTE STATISTICS FOR COLUMNS ws_sold_date_sk,
ws_sold_time_sk, ws_ship_date_sk, ws_item_sk, ws_bill_customer_sk,
ws_bill_cdemo_sk, ws_bill_hdemo_sk, ws_bill_addr_sk, ws_ship_customer_sk,
ws_ship_cdemo_sk, ws_ship_hdemo_sk, ws_ship_addr_sk, ws_web_page_sk,
ws_web_site_sk, ws_ship_mode_sk, ws_warehouse_sk, ws_promo_sk, ws_order_number,
ws_quantity, ws_wholesale_cost, ws_list_price, ws_sales_price, ws_ext_discount_amt,
ws_ext_sales_price, ws_ext_wholesale_cost, ws_ext_list_price, ws_ext_tax,
ws_coupon_amt, ws_ext_ship_cost, ws_net_paid, ws_net_paid_inc_tax,
ws_net_paid_inc_ship, ws_net_paid_inc_ship_tax, ws_net_profit;

ANALYZE TABLE web_site COMPUTE STATISTICS FOR COLUMNS web_site_sk, web_site_id,
web_rec_start_date, web_rec_end_date, web_name, web_open_date_sk,
web_close_date_sk, web_class, web_manager, web_mkt_id, web_mkt_class, web_mkt_desc,
web_market_manager, web_company_id, web_company_name, web_street_number,
web_street_name, web_street_type, web_suite_number, web_city, web_county,
web_state, web_zip, web_country, web_gmt_offset, web_tax_percentage;
```

Stats views:

```
DBNAME=$1
schema=$2

db2 connect to ${DBNAME}
db2 -v set schema ${schema}

db2 -v "drop view cr_gview"
db2 -v "drop view sr_gview"
db2 -v "drop view ss_gview"
```

```
db2 -v "drop view wr_gview"
db2 -v "drop view ws_gview"
db2 -v "drop view c_gview"
db2 -v "drop view inv_gview"
db2 -v "drop view sv_date_dim"

db2 -v "create view CR_GVIEW (c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12,
c13, c14, c15, c16, c17, c18, c19, c20, c21, c22, c23, c24, c25, c26, c27, c28,
c29, c30, c31, c32, c33, c34, c35, c36, c37, c38, c39, c40, c41, c42, c43, c44,
c45, c46, c47, c48, c49, c50, c51, c52, c53, c54, c55, c56, c57, c58, c59, c60,
c61, c62, c63, c64, c65, c66, c67, c68, c69, c70, c71, c72, c73, c74, c75, c76,
c77, c78, c79, c80, c81, c82, c83,c84, c85, c86, c87, c88, c89, c90, c91, c92, c93,
c94, c95, c96, c97, c98, c99, d_d_date) as
(
       select T2.*, T3.*, T4.*, T5.*, T6.*, T7.*, DATE(T5.D_DATE) as D_D_DATE
       from   CATALOG_RETURNS as T1,
              CATALOG_PAGE as T2, CUSTOMER_ADDRESS as T3, CUSTOMER as T4,
              DATE_DIM as T5, CUSTOMER_ADDRESS as T6, CUSTOMER as T7
       where  T1.CR_CATALOG_PAGE_SK       = T2.CP_CATALOG_PAGE_SK  and
              T1.CR_REFUNDED_ADDR_SK      = T3.CA_ADDRESS_SK       and
              T1.CR_REFUNDED_CUSTOMER_SK  = T4.C_CUSTOMER_SK       and
              T1.CR_RETURNED_DATE_SK      = T5.D_DATE_SK           and
              T1.CR_RETURNING_ADDR_SK     = T6.CA_ADDRESS_SK       and
              T1.CR_RETURNING_CUSTOMER_SK = T7.C_CUSTOMER_SK
)"


db2 -v "create view SR_GVIEW as
(
       select T2.*, T3.*, T4.*, T5.*, DATE(T3.D_DATE) as D_D_DATE
       from   STORE_RETURNS as T1,
              CUSTOMER as T2, DATE_DIM as T3, TIME_DIM as T4, STORE as T5
       where  T1.SR_CUSTOMER_SK           = T2.C_CUSTOMER_SK       and
              T1.SR_RETURNED_DATE_SK      = T3.D_DATE_SK           and
              T1.SR_RETURN_TIME_SK        = T4.T_TIME_SK           and
              T1.SR_STORE_SK              = T5.S_STORE_SK
)"


db2 -v "create view SS_GVIEW as
(
       select T2.*, T3.*, T4.*, DATE(T2.D_DATE) as D_D_DATE
       from   STORE_SALES as T1,
              DATE_DIM as T2, TIME_DIM as T3, STORE as T4
       where  T1.SS_SOLD_DATE_SK          = T2.D_DATE_SK           and
              T1.SS_SOLD_TIME_SK          = T3.T_TIME_SK           and
              T1.SS_STORE_SK              = T4.S_STORE_SK
)"



db2 -v "create view WR_GVIEW (c1, c2, c3, c4, c5,c6, c7, c8, c9, c10, c11, c12,
c13, c14, c15, c16, c17, c18, c19, c20, c21, c22, c23, c24, c25, c26, c27, c28,
c29, c30, c31, c32, c33, c34, c35, c36, c37, c38, c39, c40, c41, c42, c43, c44,
c45, c46, c47, c48, c49, c50, c51, c52, c53, c54, c55, c56, c57, c58, c59, c60,
c61, c62, c63, c64, c65, c66, c67, c68, c69, c70, c71, c72, c73, c74, c75, c76,
c77, c78, c79, c80, c81, c82, c83,c84, c85, c86, c87, c88, c89, c90, c91, c92, c93,
c94, c95, c96, c97, c98, c99, c100, c101, c102, c103, c104, c105, c106, c107, c108,
D_D_DATE) as
(
       select T2.*, T3.*, T4.*, T5.*, T6.*, T7.*, T8.*, DATE(T5.D_DATE) as D_D_DATE
       from   WEB_RETURNS as T1,
              CUSTOMER_ADDRESS as T2, CUSTOMER_DEMOGRAPHICS as T3, CUSTOMER as T4,
DATE_DIM as T5,
              CUSTOMER_ADDRESS as T6, CUSTOMER_DEMOGRAPHICS as T7, CUSTOMER as T8
       where  T1.WR_REFUNDED_ADDR_SK      = T2.CA_ADDRESS_SK       and
              T1.WR_REFUNDED_CDEMO_SK     = T3.CD_DEMO_SK          and
              T1.WR_REFUNDED_CUSTOMER_SK  = T4.C_CUSTOMER_SK       and
```

```
              T1.WR_RETURNED_DATE_SK      = T5.D_DATE_SK           and
              T1.WR_RETURNING_ADDR_SK     = T6.CA_ADDRESS_SK       and
              T1.WR_RETURNING_CDEMO_SK    = T7.CD_DEMO_SK          and
              T1.WR_RETURNING_CUSTOMER_SK = T8.C_CUSTOMER_SK
)"


db2 -v "create view WS_GVIEW (c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12,
c13, c14, c15, c16, c17, c18, c19, c20, c21, c22, c23, c24, c25, c26, c27, c28,
c29, c30, c31, c32, c33, c34, c35, c36, c37, c38, c39, c40, c41, c42, c43, c44,
c45, c46, c47, c48, c49, c50, c51, c52, c53, c54, c55, c56, c57, c58, c59, c60,
c61, c62, c63, c64, c65, c66, c67, c68, c69, c70, c71, c72, c73, c74, c75, c76,
c77, c78, c79, c80, c81, c82, c83,c84, c85, c86, c87, c88, c89, c90, c91, c92,
D_D_DATE, E_D_DATE) as
(
      select T2.*, T3.*, T4.*, T5.*, DATE(T3.D_DATE) as D_D_DATE, DATE(T5.D_DATE)
as E_D_DATE
      from   WEB_SALES as T1,
             CUSTOMER as T2, DATE_DIM as T3, CUSTOMER as T4, DATE_DIM as T5
      where  T1.WS_BILL_CUSTOMER_SK       = T2.C_CUSTOMER_SK       and
             T1.WS_SHIP_CUSTOMER_SK       = T4.C_CUSTOMER_SK       and
             T1.WS_SHIP_DATE_SK           = T3.D_DATE_SK           and
             T1.WS_SOLD_DATE_SK           = T5.D_DATE_SK
)"

db2 -v "create view C_GVIEW (c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12,
c13, c14, c15, c16, c17, c18, c19, c20, c21, c22, c23, c24, c25, c26, c27, c28,
c29, c30, c31, c32, c33, c34, c35, c36, c37, c38, c39, c40, c41, c42, c43, c44,
c45, c46, c47, c48, c49, c50, c51, c52, c53, c54, c55, c56, c57, c58, c59, c60,
c61, c62, c63, c64, c65, c66, c67, c68, c69, c70, c71, c72, c73, c74, c75, c76,
c77, c78, D_D_DATE, E_D_DATE) as
(
      select T2.*, T3.*, T4.*, T5.*, DATE(T4.D_DATE) as D_D_DATE, DATE(T5.D_DATE)
as E_D_DATE
      from   CUSTOMER as T1,
             CUSTOMER_ADDRESS as T2, CUSTOMER_DEMOGRAPHICS as T3, DATE_DIM as T4,
DATE_DIM as T5
      where  T1.C_CURRENT_ADDR_SK         = T2.CA_ADDRESS_SK       and
             T1.C_CURRENT_CDEMO_SK        = T3.CD_DEMO_SK          and
             T1.C_FIRST_SALES_DATE_SK     = T4.D_DATE_SK           and
             T1.C_FIRST_SHIPTO_DATE_SK    = T5.D_DATE_SK
)"


db2 -v "create view INV_GVIEW as (select T2.*, DATE(T2.D_DATE) as D_D_DATE from
INVENTORY as T1, DATE_DIM as T2 where T1.INV_DATE_SK=T2.D_DATE_SK)"

db2 -v "create view SV_DATE_DIM as (select date(d_date) as d_d_date from DATE_DIM)"

db2 -v "alter view CR_GVIEW enable query optimization"
db2 -v "alter view SR_GVIEW enable query optimization"
db2 -v "alter view SS_GVIEW enable query optimization"
db2 -v "alter view WR_GVIEW enable query optimization"
db2 -v "alter view WS_GVIEW enable query optimization"
db2 -v "alter view C_GVIEW enable query optimization"
db2 -v "alter view INV_GVIEW enable query optimization"
db2 -v "alter view SV_DATE_DIM enable query optimization"


time db2 -v "runstats on table SV_DATE_DIM with distribution"
time db2 -v "runstats on table CR_GVIEW with distribution tablesample BERNOULLI(1)"
time db2 -v "runstats on table SR_GVIEW with distribution tablesample BERNOULLI(1)"
time db2 -v "runstats on table SS_GVIEW with distribution tablesample BERNOULLI(1)"
time db2 -v "runstats on table WR_GVIEW with distribution tablesample BERNOULLI(1)"
time db2 -v "runstats on table WS_GVIEW with distribution tablesample BERNOULLI(1)"
time db2 -v "runstats on table C_GVIEW with distribution tablesample BERNOULLI(1)"
time db2 -v "runstats on table INV_GVIEW with distribution tablesample
BERNOULLI(1)"
```

```
db2 commit
db2 terminate


DBNAME=$1
schema=$2

db2 connect to ${DBNAME}
db2 -v set schema ${schema}

db2 -v "drop view cs_gview1"
db2 -v "drop view cs_gview2"
db2 -v "drop view cs_gview3"
db2 -v "drop view cs_gview4"
db2 -v "drop view cs_gview5"

db2 -v "create view cs_gview1 as (
select t2.* from CATALOG_SALES as t1, CUSTOMER as t2
where
t1.CS_BILL_CUSTOMER_SK=t2.C_CUSTOMER_SK
)"

db2 -v "create view cs_gview2 as (
select t2.* from CATALOG_SALES as t1, CATALOG_PAGE as t2
where
t1.CS_CATALOG_PAGE_SK=t2.CP_CATALOG_PAGE_SK
)"

db2 -v "create view cs_gview3 as (
select t2.* from CATALOG_SALES as t1, CUSTOMER as t2
where
t1.CS_SHIP_CUSTOMER_SK=t2.C_CUSTOMER_SK
)"

db2 -v "create view cs_gview4 as (
select t2.*, DATE(t2.D_DATE) as D_D_DATE from CATALOG_SALES as t1, DATE_DIM as t2
where
t1.CS_SHIP_DATE_SK=t2.D_DATE_SK
)"

db2 -v "create view cs_gview5 as (
select t2.*, DATE(t2.D_DATE) as D_D_DATE from CATALOG_SALES as t1, DATE_DIM as t2
where
t1.CS_SOLD_DATE_SK=t2.D_DATE_SK
)"

db2 -v "alter view cs_gview1 enable query optimization"
db2 -v "alter view cs_gview2 enable query optimization"
db2 -v "alter view cs_gview3 enable query optimization"
db2 -v "alter view cs_gview4 enable query optimization"
db2 -v "alter view cs_gview5 enable query optimization"

time db2 -v "runstats on table cs_gview1 with distribution tablesample
BERNOULLI(1)"
time db2 -v "runstats on table cs_gview2 with distribution tablesample
BERNOULLI(1)"
time db2 -v "runstats on table cs_gview3 with distribution tablesample
BERNOULLI(1)"
time db2 -v "runstats on table cs_gview4 with distribution tablesample
BERNOULLI(1)"
time db2 -v "runstats on table cs_gview5 with distribution tablesample
BERNOULLI(1)"

db2 commit
db2 terminate
```

Informational Constraints:

```
set schema $schema;

-------------------------------------------------------------
--  primary key definitions
-------------------------------------------------------------

alter table call_center
    add primary key (cc_call_center_sk)
    not enforced enable query optimization;
commit work;

alter table catalog_page
    add primary key (cp_catalog_page_sk)
    not enforced enable query optimization;
commit work;

alter table catalog_returns
    add primary key (cr_item_sk, cr_order_number)
    not enforced enable query optimization;
commit work;

alter table catalog_sales
    add primary key (cs_item_sk, cs_order_number)
    not enforced enable query optimization;
commit work;

alter table customer
    add primary key (c_customer_sk)
    not enforced enable query optimization;
commit work;

alter table customer_address
    add primary key (ca_address_sk)
    not enforced enable query optimization;
commit work;

alter table customer_demographics
    add primary key (cd_demo_sk)
    not enforced enable query optimization;
commit work;

alter table date_dim
    add primary key (d_date_sk)
    not enforced enable query optimization;
commit work;

alter table household_demographics
    add primary key (hd_demo_sk)
    not enforced enable query optimization;
commit work;

alter table income_band
    add primary key (ib_income_band_sk)
    not enforced enable query optimization;
commit work;

alter table inventory
    add primary key (inv_date_sk, inv_item_sk, inv_warehouse_sk)
    not enforced enable query optimization;
commit work;

alter table item
    add primary key (i_item_sk)
    not enforced enable query optimization;
```

```
commit work;

alter table promotion
    add primary key (p_promo_sk)
    not enforced enable query optimization;
commit work;

alter table reason
    add primary key (r_reason_sk)
    not enforced enable query optimization;
commit work;

alter table ship_mode
    add primary key (sm_ship_mode_sk)
    not enforced enable query optimization;
commit work;

alter table store
    add primary key (s_store_sk)
    not enforced enable query optimization;
commit work;

alter table store_returns
    add primary key (sr_item_sk, sr_ticket_number)
    not enforced enable query optimization;
commit work;

alter table store_sales
    add primary key (ss_item_sk, ss_ticket_number)
    not enforced enable query optimization;
commit work;

alter table time_dim
    add primary key (t_time_sk)
    not enforced enable query optimization;
commit work;

alter table warehouse
    add primary key (w_warehouse_sk)
    not enforced enable query optimization;
commit work;

alter table web_page
    add primary key (wp_web_page_sk)
    not enforced enable query optimization;
commit work;

alter table web_returns
    add primary key (wr_item_sk, wr_order_number)
    not enforced enable query optimization;
commit work;

alter table web_sales
    add primary key (ws_item_sk, ws_order_number)
    not enforced enable query optimization;
commit work;

alter table web_site
    add primary key (web_site_sk)
    not enforced enable query optimization;
commit work;


------------------------------------------------------------
--  foreign key definitions
------------------------------------------------------------

-- tables with no FKs
```

```
--   customer_address
--   customer_demographics
--   item
--   date_dim
--   warehouse
--   ship_mode
--   time_dim
--   reason
--   income_band

alter table promotion
    add constraint fk1 foreign key  (p_start_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table promotion
    add constraint fk2 foreign key  (p_end_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table promotion
    add constraint fk3 foreign key  (p_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store
    add constraint fk foreign key  (s_closed_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table call_center
    add constraint fk1 foreign key  (cc_closed_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table call_center
    add constraint fk2 foreign key  (cc_open_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table customer
    add constraint fk1 foreign key  (c_current_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table customer
    add constraint fk2 foreign key  (c_current_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table customer
    add constraint fk3 foreign key  (c_current_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table customer
    add constraint fk4 foreign key  (c_first_shipto_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table customer
    add constraint fk5 foreign key  (c_first_sales_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;
```

```
alter table web_site
    add constraint fk1 foreign key  (web_open_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_site
    add constraint fk2 foreign key  (web_close_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_page
    add constraint fk1 foreign key  (cp_start_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_page
    add constraint fk2 foreign key  (cp_end_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table household_demographics
    add constraint fk foreign key  (hd_income_band_sk)
       references income_band (ib_income_band_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_page
    add constraint fk1 foreign key  (wp_creation_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_page
    add constraint fk2 foreign key  (wp_access_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_page
    add constraint fk3 foreign key  (wp_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk1 foreign key  (ss_sold_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk2 foreign key  (ss_sold_time_sk)
       references time_dim (t_time_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk3a foreign key  (ss_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk4 foreign key  (ss_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk5 foreign key  (ss_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table store_sales
```

```
    add constraint fk6 foreign key  (ss_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk7 foreign key  (ss_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk8 foreign key  (ss_store_sk)
       references store (s_store_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_sales
    add constraint fk9 foreign key  (ss_promo_sk)
       references promotion (p_promo_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk1 foreign key  (sr_returned_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk2 foreign key  (sr_return_time_sk)
       references time_dim (t_time_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk3a foreign key  (sr_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk3b foreign key  (sr_item_sk, sr_ticket_number)
       references store_sales (ss_item_sk, ss_ticket_number) NOT ENFORCED ENABLE
QUERY OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk4 foreign key  (sr_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk5 foreign key  (sr_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk6 foreign key  (sr_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk7 foreign key  (sr_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table store_returns
    add constraint fk8 foreign key  (sr_store_sk)
       references store (s_store_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
```

```
commit work;

alter table store_returns
    add constraint fk9 foreign key  (sr_reason_sk)
       references reason (r_reason_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk1 foreign key  (cs_sold_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk2 foreign key  (cs_sold_time_sk)
       references time_dim (t_time_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk3 foreign key  (cs_ship_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk4 foreign key  (cs_bill_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk5 foreign key  (cs_bill_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk6 foreign key  (cs_bill_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk7 foreign key  (cs_bill_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk8 foreign key  (cs_ship_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk9 foreign key  (cs_ship_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk10 foreign key  (cs_ship_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk11 foreign key  (cs_ship_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;
```

```
alter table catalog_sales
    add constraint fk12 foreign key  (cs_call_center_sk)
       references call_center (cc_call_center_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk13 foreign key  (cs_catalog_page_sk)
       references catalog_page (cp_catalog_page_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk14 foreign key  (cs_ship_mode_sk)
       references ship_mode (sm_ship_mode_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk15 foreign key  (cs_warehouse_sk)
       references warehouse (w_warehouse_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk16a foreign key  (cs_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_sales
    add constraint fk17 foreign key  (cs_promo_sk)
       references promotion (p_promo_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk1 foreign key  (cr_returned_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk2 foreign key  (cr_returned_time_sk)
       references time_dim (t_time_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk3 foreign key  (cr_item_sk, cr_order_number)
       references catalog_sales (cs_item_sk, cs_order_number) NOT ENFORCED ENABLE
QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk4 foreign key  (cr_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk5 foreign key  (cr_refunded_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk6 foreign key  (cr_refunded_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk7 foreign key  (cr_refunded_hdemo_sk)
```

```
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk8 foreign key  (cr_refunded_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk9 foreign key  (cr_returning_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk10 foreign key  (cr_returning_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk11 foreign key  (cr_returning_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk12 foreign key  (cr_returning_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk13 foreign key  (cr_call_center_sk)
       references call_center (cc_call_center_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk14 foreign key  (cr_catalog_page_sk)
       references catalog_page (cp_catalog_page_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk15 foreign key  (cr_ship_mode_sk)
       references ship_mode (sm_ship_mode_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk16 foreign key  (cr_warehouse_sk)
       references warehouse (w_warehouse_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table catalog_returns
    add constraint fk17 foreign key  (cr_reason_sk)
       references reason (r_reason_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk1 foreign key  (ws_sold_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
```

```
    add constraint fk2 foreign key  (ws_sold_time_sk)
       references time_dim (t_time_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk3 foreign key  (ws_ship_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk4a foreign key  (ws_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk5 foreign key  (ws_bill_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk6 foreign key  (ws_bill_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk7 foreign key  (ws_bill_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk8 foreign key  (ws_bill_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk9 foreign key  (ws_ship_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk10 foreign key  (ws_ship_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk11 foreign key  (ws_ship_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk12 foreign key  (ws_ship_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk13 foreign key  (ws_web_page_sk)
       references web_page (wp_web_page_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk14 foreign key  (ws_web_site_sk)
       references web_site (web_site_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
```

```
commit work;

alter table web_sales
    add constraint fk15 foreign key  (ws_ship_mode_sk)
       references ship_mode (sm_ship_mode_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk16 foreign key  (ws_warehouse_sk)
       references warehouse (w_warehouse_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_sales
    add constraint fk17 foreign key  (ws_promo_sk)
       references promotion (p_promo_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk1 foreign key  (wr_returned_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk2 foreign key  (wr_returned_time_sk)
       references time_dim (t_time_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk3a foreign key  (wr_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk3b foreign key  (wr_item_sk, wr_order_number)
       references web_sales (ws_item_sk, ws_order_number) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk4 foreign key  (wr_refunded_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk5 foreign key  (wr_refunded_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk6 foreign key  (wr_refunded_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk7 foreign key  (wr_refunded_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk8 foreign key  (wr_returning_customer_sk)
       references customer (c_customer_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;
```

```
alter table web_returns
    add constraint fk9 foreign key  (wr_returning_cdemo_sk)
       references customer_demographics (cd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk10 foreign key  (wr_returning_hdemo_sk)
       references household_demographics (hd_demo_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk11 foreign key  (wr_returning_addr_sk)
       references customer_address (ca_address_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk12 foreign key  (wr_web_page_sk)
       references web_page (wp_web_page_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table web_returns
    add constraint fk13 foreign key  (wr_reason_sk)
       references reason (r_reason_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table inventory
    add constraint fk1 foreign key  (inv_date_sk)
       references date_dim (d_date_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table inventory
    add constraint fk2 foreign key  (inv_item_sk)
       references item (i_item_sk) NOT ENFORCED ENABLE QUERY OPTIMIZATION;
commit work;

alter table inventory
    add constraint fk3 foreign key  (inv_warehouse_sk)
       references warehouse (w_warehouse_sk) NOT ENFORCED ENABLE QUERY
OPTIMIZATION;
commit work;
```

F.2 Impala Load & Analyze scripts:

Load:

```
#!/bin/bash


impala-shell -d tpcds10000g <<EOF
create table date_dim like et_date_dim stored as parquetfile;
insert overwrite table date_dim select * from et_date_dim;


create table time_dim like et_time_dim stored as parquetfile;
insert overwrite table time_dim select * from et_time_dim;


create table customer like et_customer stored as parquetfile;
```

```
insert overwrite table customer select * from et_customer;


create table customer_address like et_customer_address stored as parquetfile;
insert overwrite table customer_address select * from et_customer_address;


create table customer_demographics like et_customer_demographics stored as
parquetfile;
insert overwrite table customer_demographics select * from
et_customer_demographics;


create table household_demographics like et_household_demographics stored as
parquetfile;
insert overwrite table household_demographics select * from
et_household_demographics;


create table item like et_item stored as parquetfile;
insert overwrite table item select * from et_item;


create table promotion like et_promotion stored as parquetfile;
insert overwrite table promotion select * from et_promotion;


create table store like et_store stored as parquetfile;
insert overwrite table store select * from et_store;


create table store_returns like et_store_returns stored as parquetfile;
insert overwrite table store_returns select * from et_store_returns;


create table web_sales like et_web_sales stored as parquetfile;
insert overwrite table web_sales select * from et_web_sales;


create table web_returns like et_web_returns stored as parquetfile;
insert overwrite table web_returns select * from et_web_returns;


create table catalog_sales like et_catalog_sales stored as parquetfile;
insert overwrite table catalog_sales select * from et_catalog_sales;


create table catalog_returns like et_catalog_returns stored as parquetfile;
insert overwrite table catalog_returns select * from et_catalog_returns;


create table store_sales like et_store_sales stored as parquetfile;
insert overwrite table store_sales select * from et_store_sales;


create table call_center like et_call_center stored as parquetfile;
insert overwrite table call_center select * from et_call_center;
```

```
create table income_band like et_income_band stored as parquetfile;
insert overwrite table income_band select * from et_income_band;


create table ship_mode like et_ship_mode stored as parquetfile;
insert overwrite table ship_mode select * from et_ship_mode;


create table reason like et_reason stored as parquetfile;
insert overwrite table reason select * from et_reason;


create table reason like et_reason stored as parquetfile;
insert overwrite table reason select * from et_reason;


create table inventory like et_inventory stored as parquetfile;
insert overwrite table inventory select * from et_inventory;


create table warehouse like et_warehouse stored as parquetfile;
insert overwrite table warehouse select * from et_warehouse;


create table web_site like et_web_site stored as parquetfile;
insert overwrite table web_site select * from et_web_site;


create table web_page like et_web_page stored as parquetfile;
insert overwrite table web_page select * from et_web_page;


create table catalog_page like et_catalog_page stored as parquetfile;
insert overwrite table catalog_page select * from et_catalog_page;


show tables;
EOF
```

Analyze:

```
#!/bin/bash

impala-shell -d TPCDS10000G <<EOF
compute stats call_center;
compute stats catalog_page;
compute stats catalog_returns;
compute stats catalog_sales;
compute stats customer;
compute stats customer_address;
compute stats customer_demographics;
compute stats date_dim;
compute stats household_demographics;
compute stats income_band;
compute stats inventory;
compute stats item;
compute stats promotion;
compute stats reason;
```

```
compute stats ship_mode;
compute stats store;
compute stats store_returns;
compute stats store_sales;
compute stats time_dim;
compute stats warehouse;
compute stats web_page;
compute stats web_returns;
compute stats web_sales;
compute stats web_site;
EOF
```

F.3 Hive0.13 Load & Analyze scripts:

Load:

```
-- Use the following to execute this script and create the tables in Hive:
--   $HIVE_HOME/bin/hive -hiveconf DB_NAME=300 -f
$testhome/ddl/070.hive.populateTables.ORC.sql


USE TPCDS${hiveconf:DB_NAME}G_HIVE;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.customer_address
   SELECT * FROM customer_address;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.customer_demographics
   SELECT * FROM customer_demographics;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.date_dim
   SELECT * FROM date_dim;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.warehouse
   SELECT * FROM warehouse;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.ship_mode
   SELECT * FROM ship_mode;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.time_dim
   SELECT * FROM time_dim;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.reason
   SELECT * FROM reason;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.income_band
   SELECT * FROM income_band;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.item
   SELECT * FROM item;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.store
   SELECT * FROM store;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.call_center
   SELECT * FROM call_center;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.customer
   SELECT * FROM customer;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.web_site
   SELECT * FROM web_site;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.household_demographics
   SELECT * FROM household_demographics;
```

```
INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.web_page
   SELECT * FROM web_page;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.promotion
   SELECT * FROM promotion;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.catalog_page
   SELECT * FROM catalog_page;

set mapred.min.split.size=128000000;
set mapred.max.split.size=128000000;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.store_returns
   SELECT * FROM store_returns;

set mapred.min.split.size=128000000;
set mapred.max.split.size=128000000;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.inventory
   SELECT * FROM inventory;

set mapred.min.split.size=512000000;
set mapred.max.split.size=512000000;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.catalog_returns
   SELECT * FROM catalog_returns;

set mapred.min.split.size=128000000;
set mapred.max.split.size=128000000;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.web_returns
   SELECT * FROM web_returns;

set mapred.min.split.size=128000000;
set mapred.max.split.size=128000000;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.web_sales
   SELECT * FROM web_sales;

set mapred.min.split.size=1000000000;
set mapred.max.split.size=1000000000;
set hive.enforce.bucketing=true;
```

```
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.catalog_sales
   SELECT * FROM catalog_sales;

set mapred.min.split.size=1500000000;
set mapred.max.split.size=1500000000;
set hive.enforce.bucketing=true;
set hive.enforce.sorting=true;
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;
set hive.input.format=org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;

INSERT OVERWRITE TABLE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B.store_sales
   SELECT * FROM store_sales;
```

Analyze:

```
USE TPCDS${hiveconf:DB_NAME}G_HIVE_ORC_B_NEW;

ANALYZE TABLE call_center COMPUTE STATISTICS;
ANALYZE TABLE call_center COMPUTE STATISTICS FOR COLUMNS
       cc_call_center_sk, cc_call_center_id, cc_rec_start_date,
       cc_rec_end_date, cc_closed_date_sk, cc_open_date_sk, cc_name,
       cc_class, cc_employees, cc_sq_ft, cc_hours, cc_manager,
       cc_mkt_id, cc_mkt_class, cc_mkt_desc, cc_market_manager,
       cc_division, cc_division_name, cc_company, cc_company_name,
       cc_street_number, cc_street_name, cc_street_type,
       cc_suite_number, cc_city, cc_county, cc_state, cc_zip,
       cc_country, cc_gmt_offset, cc_tax_percentage;

ANALYZE TABLE catalog_page COMPUTE STATISTICS;
ANALYZE TABLE catalog_page COMPUTE STATISTICS FOR COLUMNS
       cp_catalog_page_sk, cp_catalog_page_id, cp_start_date_sk,
       cp_end_date_sk, cp_department, cp_catalog_number,
       cp_catalog_page_number, cp_description, cp_type;

ANALYZE TABLE catalog_returns  COMPUTE STATISTICS;
ANALYZE TABLE catalog_returns  COMPUTE STATISTICS FOR COLUMNS
       cr_returned_date_sk, cr_returned_time_sk, cr_item_sk,
       cr_refunded_customer_sk, cr_refunded_cdemo_sk,
       cr_refunded_hdemo_sk, cr_refunded_addr_sk,
       cr_returning_customer_sk, cr_returning_cdemo_sk,
       cr_returning_hdemo_sk, cr_returning_addr_sk, cr_call_center_sk,
       cr_catalog_page_sk, cr_ship_mode_sk, cr_warehouse_sk,
       cr_reason_sk, cr_order_number, cr_return_quantity,
       cr_return_amount, cr_return_tax, cr_return_amt_inc_tax, cr_fee,
       cr_return_ship_cost, cr_refunded_cash, cr_reversed_charge,
       cr_store_credit, cr_net_loss;

ANALYZE TABLE catalog_sales  COMPUTE STATISTICS;
ANALYZE TABLE catalog_sales  COMPUTE STATISTICS FOR COLUMNS
       cs_sold_date_sk, cs_sold_time_sk, cs_ship_date_sk,
       cs_bill_customer_sk, cs_bill_cdemo_sk, cs_bill_hdemo_sk,
       cs_bill_addr_sk, cs_ship_customer_sk, cs_ship_cdemo_sk,
       cs_ship_hdemo_sk, cs_ship_addr_sk, cs_call_center_sk,
       cs_catalog_page_sk, cs_ship_mode_sk, cs_warehouse_sk,
       cs_item_sk, cs_promo_sk, cs_order_number, cs_quantity,
       cs_wholesale_cost, cs_list_price, cs_sales_price,
       cs_ext_discount_amt, cs_ext_sales_price, cs_ext_wholesale_cost,
       cs_ext_list_price, cs_ext_tax, cs_coupon_amt, cs_ext_ship_cost,
       cs_net_paid, cs_net_paid_inc_tax, cs_net_paid_inc_ship,
       cs_net_paid_inc_ship_tax, cs_net_profit;
```

```
ANALYZE TABLE customer COMPUTE STATISTICS;
ANALYZE TABLE customer COMPUTE STATISTICS FOR COLUMNS
       c_customer_sk, c_customer_id, c_current_cdemo_sk,
       c_current_hdemo_sk, c_current_addr_sk, c_first_shipto_date_sk,
       c_first_sales_date_sk, c_salutation, c_first_name, c_last_name,
       c_preferred_cust_flag, c_birth_day, c_birth_month,
       c_birth_year, c_birth_country, c_login, c_email_address,
       c_last_review_date;

ANALYZE TABLE customer_address COMPUTE STATISTICS;
ANALYZE TABLE customer_address COMPUTE STATISTICS FOR COLUMNS
       ca_address_sk, ca_address_id, ca_street_number, ca_street_name,
       ca_street_type, ca_suite_number, ca_city, ca_county, ca_state,
       ca_zip, ca_country, ca_gmt_offset, ca_location_type;

ANALYZE TABLE customer_demographics COMPUTE STATISTICS;
ANALYZE TABLE customer_demographics COMPUTE STATISTICS FOR COLUMNS
       cd_demo_sk, cd_gender, cd_marital_status, cd_education_status,
       cd_purchase_estimate, cd_credit_rating, cd_dep_count,
       cd_dep_employed_count, cd_dep_college_count;

ANALYZE TABLE date_dim COMPUTE STATISTICS;
ANALYZE TABLE date_dim COMPUTE STATISTICS FOR COLUMNS
       d_date_sk, d_date_id, d_date, d_month_seq, d_week_seq,
       d_quarter_seq, d_year, d_dow, d_moy, d_dom, d_qoy, d_fy_year,
       d_fy_quarter_seq, d_fy_week_seq, d_day_name, d_quarter_name,
       d_holiday, d_weekend, d_following_holiday, d_first_dom,
       d_last_dom, d_same_day_ly, d_same_day_lq, d_current_day,
       d_current_week, d_current_month, d_current_quarter,
       d_current_year;

ANALYZE TABLE household_demographics COMPUTE STATISTICS;
ANALYZE TABLE household_demographics COMPUTE STATISTICS FOR COLUMNS
       hd_demo_sk, hd_income_band_sk, hd_buy_potential, hd_dep_count,
       hd_vehicle_count;

ANALYZE TABLE income_band COMPUTE STATISTICS;
ANALYZE TABLE income_band COMPUTE STATISTICS FOR COLUMNS
       ib_income_band_sk, ib_lower_bound, ib_upper_bound;

ANALYZE TABLE inventory  COMPUTE STATISTICS;
ANALYZE TABLE inventory  COMPUTE STATISTICS FOR COLUMNS
       inv_date_sk, inv_item_sk, inv_warehouse_sk,
       inv_quantity_on_hand;

ANALYZE TABLE item COMPUTE STATISTICS;
ANALYZE TABLE item COMPUTE STATISTICS FOR COLUMNS
       i_item_sk, i_item_id, i_rec_start_date, i_rec_end_date,
       i_item_desc, i_current_price, i_wholesale_cost, i_brand_id,
       i_brand, i_class_id, i_class, i_category_id, i_category,
       i_manufact_id, i_manufact, i_size, i_formulation, i_color,
       i_units, i_container, i_manager_id, i_product_name;

ANALYZE TABLE promotion COMPUTE STATISTICS;
ANALYZE TABLE promotion COMPUTE STATISTICS FOR COLUMNS
       p_promo_sk, p_promo_id, p_start_date_sk, p_end_date_sk,
       p_item_sk, p_cost, p_response_target, p_promo_name,
       p_channel_dmail, p_channel_email, p_channel_catalog,
       p_channel_tv, p_channel_radio, p_channel_press,
       p_channel_event, p_channel_demo, p_channel_details, p_purpose,
       p_discount_active;

ANALYZE TABLE reason COMPUTE STATISTICS;
ANALYZE TABLE reason COMPUTE STATISTICS FOR COLUMNS
       r_reason_sk, r_reason_id, r_reason_desc;

ANALYZE TABLE ship_mode COMPUTE STATISTICS;
ANALYZE TABLE ship_mode COMPUTE STATISTICS FOR COLUMNS
```

```
      sm_ship_mode_sk, sm_ship_mode_id, sm_type, sm_code, sm_carrier,
      sm_contract;

ANALYZE TABLE store COMPUTE STATISTICS;
ANALYZE TABLE store COMPUTE STATISTICS FOR COLUMNS
      s_store_sk, s_store_id, s_rec_start_date, s_rec_end_date,
      s_closed_date_sk, s_store_name, s_number_employees,
      s_floor_space, s_hours, s_manager, s_market_id,
      s_geography_class, s_market_desc, s_market_manager,
      s_division_id, s_division_name, s_company_id, s_company_name,
      s_street_number, s_street_name, s_street_type, s_suite_number,
      s_city, s_county, s_state, s_zip, s_country, s_gmt_offset,
      s_tax_precentage;

ANALYZE TABLE store_returns  COMPUTE STATISTICS;
ANALYZE TABLE store_returns  COMPUTE STATISTICS FOR COLUMNS
      sr_returned_date_sk, sr_return_time_sk, sr_item_sk,
      sr_customer_sk, sr_cdemo_sk, sr_hdemo_sk, sr_addr_sk,
      sr_store_sk, sr_reason_sk, sr_ticket_number,
      sr_return_quantity, sr_return_amt, sr_return_tax,
      sr_return_amt_inc_tax, sr_fee, sr_return_ship_cost,
      sr_refunded_cash, sr_reversed_charge, sr_store_credit,
      sr_net_loss;

ANALYZE TABLE store_sales  COMPUTE STATISTICS;
ANALYZE TABLE store_sales  COMPUTE STATISTICS FOR COLUMNS
      ss_sold_date_sk, ss_sold_time_sk, ss_item_sk, ss_customer_sk,
      ss_cdemo_sk, ss_hdemo_sk, ss_addr_sk, ss_store_sk, ss_promo_sk,
      ss_ticket_number, ss_quantity, ss_wholesale_cost,
      ss_list_price, ss_sales_price, ss_ext_discount_amt,
      ss_ext_sales_price, ss_ext_wholesale_cost, ss_ext_list_price,
      ss_ext_tax, ss_coupon_amt, ss_net_paid, ss_net_paid_inc_tax,
      ss_net_profit;

ANALYZE TABLE time_dim COMPUTE STATISTICS;
ANALYZE TABLE time_dim COMPUTE STATISTICS FOR COLUMNS
      t_time_sk, t_time_id, t_time, t_hour, t_minute, t_second,
      t_am_pm, t_shift, t_sub_shift, t_meal_time;

ANALYZE TABLE warehouse COMPUTE STATISTICS;
ANALYZE TABLE warehouse COMPUTE STATISTICS FOR COLUMNS
      w_warehouse_sk, w_warehouse_id, w_warehouse_name,
      w_warehouse_sq_ft, w_street_number, w_street_name,
      w_street_type, w_suite_number, w_city, w_county, w_state,
      w_zip, w_country, w_gmt_offset;

ANALYZE TABLE web_page COMPUTE STATISTICS;
ANALYZE TABLE web_page COMPUTE STATISTICS FOR COLUMNS
      wp_web_page_sk, wp_web_page_id, wp_rec_start_date,
      wp_rec_end_date, wp_creation_date_sk, wp_access_date_sk,
      wp_autogen_flag, wp_customer_sk, wp_url, wp_type,
      wp_char_count, wp_link_count, wp_image_count, wp_max_ad_count;

ANALYZE TABLE web_returns  COMPUTE STATISTICS;
ANALYZE TABLE web_returns  COMPUTE STATISTICS FOR COLUMNS
      wr_returned_date_sk, wr_returned_time_sk, wr_item_sk,
      wr_refunded_customer_sk, wr_refunded_cdemo_sk,
      wr_refunded_hdemo_sk, wr_refunded_addr_sk,
      wr_returning_customer_sk, wr_returning_cdemo_sk,
      wr_returning_hdemo_sk, wr_returning_addr_sk, wr_web_page_sk,
      wr_reason_sk, wr_order_number, wr_return_quantity,
      wr_return_amt, wr_return_tax, wr_return_amt_inc_tax, wr_fee,
      wr_return_ship_cost, wr_refunded_cash, wr_reversed_charge,
      wr_account_credit, wr_net_loss;

ANALYZE TABLE web_sales  COMPUTE STATISTICS;
ANALYZE TABLE web_sales  COMPUTE STATISTICS FOR COLUMNS
      ws_sold_date_sk, ws_sold_time_sk, ws_ship_date_sk, ws_item_sk,
```

```
        ws_bill_customer_sk, ws_bill_cdemo_sk, ws_bill_hdemo_sk,
        ws_bill_addr_sk, ws_ship_customer_sk, ws_ship_cdemo_sk,
        ws_ship_hdemo_sk, ws_ship_addr_sk, ws_web_page_sk,
        ws_web_site_sk, ws_ship_mode_sk, ws_warehouse_sk, ws_promo_sk,
        ws_order_number, ws_quantity, ws_wholesale_cost, ws_list_price,
        ws_sales_price, ws_ext_discount_amt, ws_ext_sales_price,
        ws_ext_wholesale_cost, ws_ext_list_price, ws_ext_tax,
        ws_coupon_amt, ws_ext_ship_cost, ws_net_paid,
        ws_net_paid_inc_tax, ws_net_paid_inc_ship,
        ws_net_paid_inc_ship_tax, ws_net_profit;

ANALYZE TABLE web_site COMPUTE STATISTICS;
ANALYZE TABLE web_site COMPUTE STATISTICS FOR COLUMNS
        web_site_sk, web_site_id, web_rec_start_date, web_rec_end_date,
        web_name, web_open_date_sk, web_close_date_sk, web_class,
        web_manager, web_mkt_id, web_mkt_class, web_mkt_desc,
        web_market_manager, web_company_id, web_company_name,
        web_street_number, web_street_name, web_street_type,
        web_suite_number, web_city, web_county, web_state, web_zip,
        web_country, web_gmt_offset, web_tax_percentage;
```

Appendix G: Attestation Letter:

Benchmark sponsor: Berni Schiefer
IBM
8200 Warden Avenue
Markham, Ontario, L6C 1C7

October 24, 2014

At IBM's request I verified the implementation and results of a **10TB Big Data Decision Support** (Hadoop-DS) benchmark, with most features derived from the TPC-DS Benchmark.

The Hadoop-DS benchmark was executed on three identical clusters, each running a different query engine. The test clusters were configured as follows:

IBM x3650BD Cluster - 17 Nodes (configuration per node)

Operating System:	Red Hat Enterprise Linux 6.4
CPUs	2 x Intel Xeon Processor E5-2680 v2 (2.8 GHz, 25MB L3)
Memory	128GB (1866MHz DDR3)
Storage	10 x 2TB SATA 3.5" HDD

The intent of the benchmark was to measure the performance of the following three Hadoop based SQL query engines, all executing an identical workload:

- **IBM BigInsights Big SQL v3.0**
- **Cloudera CDH 5.1.2 Impala v1.4.1**
- **HortonWorks Hive v0.13**

The results were:

	Big SQL	Impala	Hive
Single-User Run Duration (h:m:s)	**0:48:28**	**2:55:36**	**4:25:49**
Multi-User Run Duration (h:m:s)	**1:55:45**	**4:08:40**	**16:32:30**
Qph Hadoop-DS @10TB - Single-User	**5,694**	**1,571**	**1,038**
Qph Hadoop-DS @10TB - Multi-User (x4)	**9,537**	**4,439**	**1,112**

These results are for a non-TPC benchmark. A subset of the TPC-DS Benchmark standard requirements was implemented.

The Hadoop-DS benchmark implementation complied with the following subset of requirements from the latest version of the TPC-DS Benchmark standard.

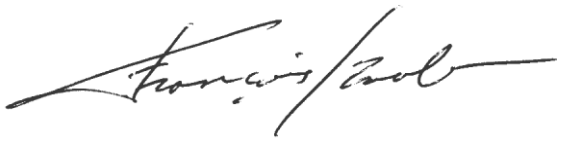
- The database schemas were defined with the proper layout and data types
- The population for the databases was generated using the TPC provided dsdgen
- The three databases were properly scaled to 10TB and populated accordingly
- The auxiliary data structure requirements were met since none were defined
- The query input variables were generated by the TPC provided dsqgen
- The execution times for queries were correctly measured and reported

The following features and requirements from the latest version of the TPC-DS Benchmark standard were not adhered to:

- A subset of 46 queries out of the total set of 99 were executed
- The database load time was neither measured nor reported
- The defined referential integrity constraints were not enforced
- The statistics collection did not meet the required limitations
- The data persistence properties were not demonstrated
- The data maintenance functions were neither implemented nor executed
- A single throughput test was used to measure multi-user performance
- The system pricing was not provided or reviewed
- The report did not meet the defined format and content

The white paper documenting the details of the Hadoop-DS benchmark executed against the three query engines was verified for accuracy.

Respectfully Yours,



François Raab, President