TPC Transaction Processing Performance Council

TPC Express Big Bench
TPCx-BB

Standard Specification
Version 0.0.17 (Formal Review)

November 13th, 2015

Transaction Processing Performance Council (TPC)

www.tpc.org
info@tpc.org

Legal Notice

No Warranty

Trademarks

Acknowledgments

The TPC acknowledges the work and contributions of the TPC BigBench subcommittee member companies:

Jeffery Buell, Dave Rorke, Meikel Poess, Wayne Smith, John Poleman, Paul Cao, Matt Emmerton, Andy Bond, Da Qi Ren, Seetha Lakshmi, Tilmann Rabl, Nicholas Wakou, Yanpei Chen, Reza Taheri, Tariq Magdon-Ismail, Raghunath Nambiar, Andy Masland, Bhaskar Gowda, Michael Brey, Jamie Reding, Doug Johnson,Dileep Kumar, Francois Raab.

TPC Membership

(as of March 2015)

Full Members

Actian	Cisco	Cloudera	Dell	Fujitsu
Hewlett Packard Enterprise	Hitachi	Huawei	IBM	Inspur
Intel	Lenovo	MapR	Microsoft	NEC
Oracle	Pivotal	Redhat	SAP	Teradata
Unisys	VMware			

Associate Members

IDEAS International	ITOM International Co.	TTA	

Document Revision History

Date	Version	Description
November 13th 2015	0.0.17 Formal Review	11/13/15 Formal Review Spec.

Typographic Conventions

The following typographic conventions are used in this specification:

Convention	Description
Bold	Bold type is used to highlight terms that are defined in this document
Italics	Italics type is used to highlight a variable that indicates some quantity whose value can be assigned in one place and referenced in many other places.
UPPERCASE	Uppercase letters names such as tables and column names. In addition, most acronyms are in uppercase.

Table of Contents

Table of Figures

TABLE OF TABLES

Clause 0 -- PREAMBLE

0.1 Introduction

Big data analytics is an ever growing field of research and business. Due to the drastic decrease of cost of storage and computation, more and more data sources become profitable for data mining. A perfect example is online stores. Earlier online shopping systems used to only record successful transactions, whereas modern systems record every single interaction of a user with the website. The former allowed for simple basket analysis techniques, while current level of detail in monitoring makes detailed user modeling possible. The growing demands on data management systems and the new forms of analysis have led to the development of a new breed of **Big Data Analytics Systems** (**BDAS**).

Similar to the advent of **Database Management Systems**, there is a vastly growing ecosystem of diverse approaches. This leads to a dilemma for customers of **BDAS**, since there are no realistic and proven measures to compare different offerings. To this end, TPCx-BB (BigBench) has been developed, the first proposal for an end to end big data analytics benchmark. TPCx-BB Benchmark was developed to cover essential functional and business aspects of big data use cases. The benchmark allows for an objective measure of hardware and operating system and provides the industry with verifiable performance, price/performance, and availability metrics.

0.1.1 Restrictions and Limitations

Despite the fact that TPC benchmarks offer a rich environment that represents many typical IT applications, these benchmarks do not reflect the entire range of customer IT requirements. In addition, the extent to which a customer can achieve the **Results** reported by a vendor is highly dependent on how closely the TPCx-BB measurements and configuration approximates the customer application. The relative performance of systems derived from these benchmarks does not necessarily hold for other workloads or environments. Extrapolations to any other environments are not recommended.

Benchmark **Results** are highly dependent upon workload, specific application requirements, and systems design, and implementation. Relative system performance and environments will vary because of these and other factors. Therefore, TPCx-BB **Results** should not be used as a substitute for specific customer application benchmarking when critical capacity planning and/or product evaluation decisions are contemplated.

Test Sponsors are permitted various possible implementation designs, insofar as they adhere to the model described and pictorially illustrated in this specification and other TPC specifications. A **Full Disclosure Report (FDR)** of the implementation details, as specified in Clause 8, must be made available along with the reported TPCx-BB metrics.

Comment: While separated from the main text for readability, comments are a part of the standard and must be enforced.

0.2 TPCx-BB Kit and Licensing

The TPCx-BB kit is available from the TPC website (see www.tpc.org/tpcx-bb for more information). Users must sign-up and agree to the TPCx-BB End User Licensing Agreement (EULA) to download the kit. All related work (such as collaterals, papers, derivatives) must acknowledge the TPC and include the TPCx-BB copyright. The TPCx-BB kit includes: TPCx-BB Specification document (this document), TPCx-BB Users Guide documentation, shell scripts to set up the benchmark environment, Java code to execute the benchmark workload, Data Generator, query files, and Benchmark Driver.

0.3 General Implementation Guidelines

The purpose of TPC benchmarks is to provide relevant, objective performance data to industry users. To achieve that purpose, TPC Benchmark Specifications require that benchmark tests be implemented with systems, products, technologies, and pricing that:

- are generally available to users
- are relevant to the market segment that the individual TPC benchmark models or represents (for example, TPCx-BB models and represents a Big Data Analytics System such as Hadoop ecosystem or Hadoop file system API compatible systems

0.3.1 Benchmark Specials

The use of new systems, products, technologies (hardware or software) and pricing is encouraged so long as they meet the requirements above. Specifically prohibited are benchmark systems, products, technologies, pricing (hereafter referred to as "implementations") whose primary purpose is optimization of TPC Benchmark **Results** without any corresponding applicability to real-world applications and environments. The intent is to disallow "**Benchmark Special**" implementations that improve benchmark results but not real-world performance, pricing, or energy consumption.

The following characteristics should be used as a guide to judge whether a particular implementation is a **Benchmark Special**. It is not required that each point below be met, but that the cumulative weight of the evidence be considered to identify an unacceptable implementation. Absolute certainty or certainty beyond a reasonable doubt is not required to make a judgment on this complex issue. The question that must be answered is this: based on the available evidence, does the clear preponderance (the greater share or weight) of evidence indicate that this implementation is a **Benchmark Special**?

0.3.2 Benchmark Special Characteristics

The following characteristics should be used to judge whether a particular implementation is a **Benchmark Special**:

- Is the implementation generally available, documented, and supported?
- Does the implementation have significant restrictions on its use or applicability that limits its use beyond TPC benchmarks?
- Is the implementation or part of the implementation poorly integrated into the larger product?
- Does the implementation take special advantage of the limited nature of TPC benchmarks (e.g., limited duration, use of virtualized capabilities not found in the commercially available product) in a manner that would not be generally applicable to the environment the benchmark represents?
- Is the use of the implementation discouraged by the vendor? (This includes failing to promote the implementation in a manner similar to other products and technologies.)
- Does the implementation require uncommon sophistication on the part of the end-user, datacenter facility manager, programmer, or system administrator?
- Does the implementation use knowledge of the variability of the possible components to enhance the **Result** in such a way as to be significantly different from what a typical customer would experience?
- Is the implementation being used (including beta) or purchased by end-users in the market area the benchmark represents? How many? Multiple sites? If the implementation is not currently being used by end-users, is there any evidence to indicate that it will be used by a significant number of users?

0.4 General Measurement Guidelines

TPCx-BB Results are expected to be accurate representations of system performance. Therefore, there are certain guidelines that are expected to be followed when measuring those results. The approach or methodology to be used in the measurements are either explicitly described in the specification or implemented by the TPCx-BB Kit Clause 2.2 When not described in the specification, the methodologies and approaches used must meet the following requirements:

- The approach is an accepted engineering practice or standard.
- The approach does not enhance the **Results**.
- The equipment used in measuring **Results** must conform to the requirements in Clause 3.
- Fidelity and candor are maintained in reporting any anomalies in the **Results**, even if not specified in the benchmark requirements.

The use of new methodologies and approaches is encouraged so long as they meet the requirements above.

0.5 Definitions

A _____

Attestation Letter

The **TPC-Certified Auditor**'s opinion regarding the compliance of a **Result** must be consigned in an **Attestation Letter** delivered directly to the **Test Sponsor**.

Availability Date

The date when all products necessary to achieve the stated performance and energy characteristics will be available (stated as a single date on the **Executive Summary**).

B _____

Benchmark Special

Any aspect of the benchmark implementation with the primary purpose of the optimization of TPC Benchmark **Results** without any corresponding applicability to real-world applications and environments.

C _____

D _____

BDAS

A **Big Data Analytics System** (BDAS) is the commercially available software that manages the Big Data portion of the TPC Benchmark Standard transactions.

SUT

The **TPC Benchmark SUT** implements the TPC Benchmark Standard transactions. The **SUT** includes

- commercially available server or servers
- commercially available storage
- commercially available Operating System

- commercially available BDAS

E _____

Executive Summary

The term **Executive Summary** refers to the Adobe Acrobat PDF file required by each TPC benchmark. The contents of the **Executive Summary** are defined in each of the TPC Benchmark Standards.

F _____

Full Disclosure Report (FDR)

The **Full Disclosure Report** is a set of files required by the TPC Benchmarks. The purpose of the **Full Disclosure Report** is to document how a benchmark **Result** was implemented and executed in sufficient detail so that the **Result** can be reproduced given the appropriate hardware and software products.

Frame Work

The Framework is collection of Software, API's, Engines etc working together to run the workload.

G _____

H _____

I _____

J _____

K _____

L _____

M _____

N _____

O _____

Operating System/OS

The term **Operating System** refers to a commercially available program that, after being initially loaded into the computer by a boot program, manages all the other programs in a computer, or in a **VM**. The **Operating System** provides a software platform on top of which all other programs run. Without the **Operating System** and the core services that it provides no other programs can run and the computer would be non-functional. Other programs make use of the **Operating System** by making requests for services through a defined application program interface (API). All major computer platforms require an **Operating System**. The functions and services supplied by an **Operating System** include but are not limited to the following:

- manages a dedicated set of processor and memory resources
- maintains and manages a file system
- loads applications into memory
- ensures that the resources allocated to one application are not used by another application in an unauthorized manner
- determines which applications should run in what order, and how much time should be allowed to run the application before giving another application a turn to use the systems resources
- manages the sharing of internal memory among multiple applications
- handles input and output to and from attached hardware devices such as hard disks, network interface cards etc.

Some examples of **Operating Systems** are listed below:

- Windows
- Unix (Solaris, AIX)
- Linux(Red Hat, SUSE)
- Mac OS

P _____

Performance Metric

The reported throughput as expressed in BigBench queries per minute.

Performance Run

Of the two TPCx-BB test runs, the **Performance Run** is defined as the run with the lower TPCx-BB **Performance Metric.**

Priced Configuration

The **Priced Configuration** consists of all components to be priced defined in the TPCx-BB Benchmark Standard including all hardware, software and maintenance.

Price/Performance Metric

The **Price/Performance Metric** is the total price of the **SUT** divided by the TPCx-BB **Performance Metric**.

Q _____

Query/ies

A Query is an implementation of one or more usecase part of the 30.

R _____

Repeatability Run

Of the two TPCx-BB test runs, the **Repeatability Run** is defined as the run with the higher TPCx-BB **Performance Metric.**

Report

The term **Report** refers to the Adobe Acrobat PDF file in the Report folder in the **FDR**. The contents of the **Report** are defined in Clause 8.

Reported

The term **Reported** refers to an item that is part of the **FDR**.

Result

A performance test, documented by an **FDR** and **Executive Summary** submitted to the TPC, claiming to meet the requirements of the TPCx-BB Benchmark Standard.

S _____

Software Version

A **Software Version** uniquely identifies a software product, its release level, update level, and/or patch level. It is typically a string of alphanumeric characters that allows the software manufacturer to uniquely identify the software.

Substitution

Substitution is the use of components in the **Priced Configuration** which are different than those used in the measured configuration. This also requires compliance with the TPC Pricing Specification.

Supporting Files

Supporting Files refers to the contents of the **Supporting Files** folder in the **FDR**. The contents of this folder, consisting of various source files, scripts, and listing files, are defined in Clause 8.

System Under Test (SUT)

System Under Test (SUT) – is defined to be the sum of the components utilized in running a benchmark as specified in Clause 3.

T _____

Test Sponsor

The **Test Sponsor** is the company officially submitting the **Result** with the **FDR** and will be charged the filing fee. Although multiple companies may sponsor a **Result** together, for the purposes of the TPC's processes the **Test Sponsor** must be a single company. A **Test Sponsor** need not be a TPC member. The **Test Sponsor** is responsible for maintaining the **FDR** with any necessary updates or corrections. The **Test Sponsor** is also the name used to identify the **Result**.

TPC-Certified Auditor (Auditor)

The term **TPC-Certified Auditor** is used to indicate that the TPC has reviewed the qualification of the **Auditor** and has certified his/her ability to verify that benchmark **Results** are in compliance with a specification. (Additional details regarding the **Auditor** certification process and the audit process can be found in Section 9 of the TPC Policies document.)

U _____

Usecase

A Usecase defines the problem Big Data Analytics System is solving, a Usecase is Framework and Syntax agnostic and can be implemented in many ways. In this kit we implement all 30 usecases in form of Queries.

V _____

W _____

X _____

Y _____

Z _____

1.1 **Overview of Data Model**

TPCx-BB is an end-to-end big data benchmark. This choice highly sped up the development of TPCx-BB and made it possible to start from a solid and proven foundation. A high-level overview of the data model is presented in Figure 1.

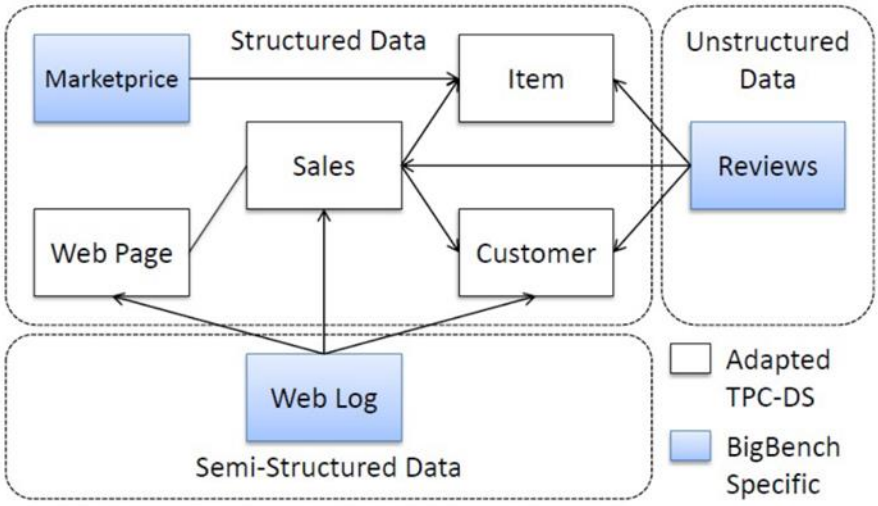


Figure 1 TPCx-BB Schema

TPCx-BB is designed with a multiple-snowflake schema inspired by TPC-DS using a retail model consisting of five fact tables, representing three sales channels, store sales, catalog sales, and online sales, each with a sales and a returns fact table.Since they have decreasing significance in retail business Catalog table is not present in TPCx-BB. As shown in Figure 1, big data specific dimensions were added. The Marketprice is a traditional relational table storing competitors' prices.

1.1.1 **Semi-structured and Unstructured Data**

TPC-DS is populated with structured data allowing the exercise of all aspects of commercial decision support systems, built with a modern **BDAS**. TPCx-BB's schema uses the data of the store and web sales distribution channel of TPC-DS and augments it with semi-structured and unstructured data. The semi-structured part captures registered and guest user clicks on the retailer's website in order to analyze the user behavior. Some of these clicks are for completing a customer order. This part of the data set is semi-structured, since different entries in the weblog represent different user actions and thus have different format.

The clickstream log contains data from URLs which are extracted from an Apache Web server log. Typically, database and Big Data systems convert the Apache log to a table with the following five columns (DateID, TimeID, SalesID, WebPageID, UserID). To ease testing, such a table is generated in advance eliminating the need to extract and convert the Apache log information.

The unstructured part of the schema is generated in form of product reviews, which are, for example, used for sentiment analysis. Figure 1 shows product reviews in the right part and their relationship to Date, Time, Item, Users and Sales tables in the structured part. The implementation of the product reviews is a single table with a structure like (DateID, TimeID, SalesID, ItemID, Re-viewRating, ReviewText).

1.1.2 **Queries**

TPCx-BB features thirty complex queries, ten of which are inspired from TPC-DS, the others were specifically developed for TPCx-BB. The queries are covering major areas of big data analytics. As a result, the queries cannot be expressed by pure SQL queries since they include machine learning techniques, sentiment analysis, and procedural computations. An example solution set is to use built in functions that are internally processed in a MapReduce fashion. The benchmark, however, does not dictate a specific implementation; the benchmark can be implemented in various ways.

Clause 2 -- WORKLOAD AND EXECUTION

2.1 Introduction

This clause defines TPCx-BB Kit, workload and execution.

2.2 Benchmark Kit

2.2.1 Kit Contents

2.2.1.1 The TPCx-BB kit contains the following:

2.2.1.2 TPCx-BB Specification document.

2.2.1.3 TPCx-BB Users Guide documentation.

2.2.1.4 Configuration files to adapt important parameters to the **SUT.**

2.2.1.5 Bash scripts which controls the benchmarking execution.

2.2.1.6 A driver written Java and Bash which implements the higher run logic, time measurement and result computation

2.2.1.7 A set of bash scripts which are called by the driver to perform benchmark and query operations.

2.2.1.8 Reference result set from for the SF 1GB.

2.2.1.9 Set of scripts to automate result verification, checks on result caridnality and report generation.

2.2.2 Kit Usage

To submit a compliant TPCx-BB **Result**, the **Test Sponsor** is required to use the TPCx-BB kit as provided with the following two exceptions:

- The setting of Kit Parameters files specified in Clause 2.2.4.
- Test Sponsor Kit Modifications explicitly allowed by Clause 2.2.5.
- The kit must be used as outlined in the TPCx-BB Users Guide.

2.2.3 Kit Run report

2.2.3.1 The output of the TPCx-BB kit is called the run report which includes the following:

2.2.3.2 Version number of TPCx-BB kit

2.2.3.3 An echo of the kit parameter (Clause 2.2.4) values used for the **Result**.

2.2.3.4 The start, end and elapsed times for the 3 tests (Clause 2.4) of the **Performance Run**.

2.2.3.5 The start, end and elapsed times for the 3 tests (Clause 2.4) of the **Repeatibility Run**.

2.2.3.6 The output from the validation run to ensure the validation run was successful on the SUT (Clause 2.4.2)

2.2.3.7 Output of any errors encountered during the Benchmark Execution.

2.2.3.8 The computed TPCx-BB **Performance Metrics** (Clause 5) for the **Performance Ru**n and **Reliability Run**.

2.2.3.9 Representative system configuration gathered from a node in the cluster.

2.2.3.10 If there is a conflict between the TPCx-BB Specification and the TPCx-BB kit, the TPCx-BB kit implementation prevails.

2.2.4 Kit Parameter settings

2.2.4.1 The following files controls the kit and and parameters may be set by the **Test Sponsor**.

2.2.4.2 Generic Benchmark parmeters defined in Appendix D

2.2.4.3 Query parameters defined in Appendix C, these parameters have been tested to provide results for SF1 and are expected to produce results for larger scale factor test runs. Test sponsor can make syntactic changes are allowed but no values can be changed.

2.2.4.4 Global parameters are engine specific, the **Test Sponsor** can set their own parameters and must disclose as part of FDR. For example please see below.

a) For Hive Global parameter file is located under $Big-Data-Benchmark-for-Big-Bench/engines/hive/conf/engineSettings.%files% E.g. Appendix Eshows and example of Hive engine parameters however the list is not exhaustive.

b) Global Framework parameters for those frameworks which do not use HIVE can place their engine specific Global parameter file under be $Big-Data-Benchmark-for-Big-Bench/engines/%engine%/conf/%enginesettings.%files%.

2.2.5 Test Sponsor Kit Modificaitons

2.2.5.1 Test Sponsor modifications to the provided scripts and configuration files in the TPCx-BB kit to facilitate system, platform and framework diffetrences are allowed without TPC approval. The allowed Test Sponsor modificaitons are as follows:

- Script changes necessary for the kit scripts to execute on a particular Operating System as long as the changes do not alter the execution logic of the script.
- Query specific optimization Framework parameters which can not make use of Global parameters as defined in 2.2.4.4 can place their query parameters under $Big-Data-Benchmark-for-Big-Bench/engines/%Engine%/queries/q%%/enginelocalsettings.%files% file Appendix Fprovides example of how these parameters can be defined and passed with the benchmark.
- Custom metastore population scripts which can be passed using "-v" are placed under Big-Data-Benchmark-for-Big-Bench/engines/%enginename%/population/ and disclosed in the **FDR**..
- For non-hive frameworks custom engine settings can be passed using "-z" or place it under Big-Data-Benchmark-for-Big-Bench/engines/%enginename%/conf/enginesettings.conf and disclosed in the **FDR.**

2.2.5.2 No modifications are allowed to the Java code provided in the TPCx-BB kit.

2.2.5.3 No JAR file optimizers are allowed to be used.

2.2.5.4 Any kit modifications not specified in Clause 2.2.5.1 must be brought forward to the Subcommittee as specified in Clause 2.3.

2.3 **Benchmark Kit Modifications**

For kit changes or modifications other than those allowed by Clause **Error! Reference source not found.**, any TPC Member, company or individual may bring forward proposed kit changes to the TPCx-BB Benchmark Subcommittee.. There are two methods of bringing forward these proposed kit changes.

Direct Method – The TPC Member, company, or individual may propose the kit changes directly to the TPCx-BB benchmark committee.

Indirect Method – If the TPC Member, company, or individual wishes to remain anonymous then a **TPC Certified Auditor** can be used as an intermediary to interact with the TPCx-BB benchmark Subcommittee.

Regardless of which method is used the individual that will be interacting with the TPCx-BB benchmark committee becomes the Change Sponsor.

2.3.1 **Simple Review of Kit Modificaitons**

For Third Tier (Clause 2.3.4.4) or Minor kit (Clause 2.3.4.2) modifications, the Change Sponsor shall present the proposed changes to the Subcommittee. The Subcommittee through its normal course of business will review the proposed changes, make the appropriate kit changes and bring forward the changes to the Council as a new revision of the TPCx-BB Benchmark.

If the proposed changes are significant, the Subcommittee may require that the Change Sponsor follow the Formal Review Process defined in Clause 2.3.2.

2.3.2 **Formal Review of Kit Modifications**

For a Major (Clause 2.3.4.1) kit modificaitons, at the request to the Subcommittee or if the Change Spsonsor so desires, the Change Sponsor shall adhere to the following Formal Review Process.

2.3.2.1 Formal Proposal of Kit Modificaitons

Step 1: The Change Sponsor must submit to the chair of the TPCx-BB benchmark Subcommittee the following information:

- The proposed code changes or new framework code
- The reason for proposing the changes
- Result set from the proposed changes
- Hardware and software list required to validate and review the proposed changes and provides hardware and software access to the reviewers to perform the independent reviews.
- Complete source code access if the proposed change prototype is available

Step 2: The chair of the TPC-BB benchmark committee will add a discussion of the proposed changes to the agenda of the next committee meeting that can be attended by the Change Sponsor.

Step 3: The Change Sponsor will present the proposed changes to the TPCx-BB benchmark Subcommittee.

Step 4: The TPCx-BB benchmark committee will vote on one of three courses of action for the proposed changes.

I. Reject the proposed changes.

II. Review the proposed changes as a Minor Kit Modification.

III. Review the proposed changes as a Major Kit Modification.

If the proposed changes are rejected, no further action is necessary. Otherwise, the proposed changes immediately enter a Proposed Change Review period.

2.3.2.2 Formal Review of Proposed Major Kit Modifications

If the proposed changes were voted to be a Major Kit Modification, then the Subcommittee chair will select at least three members of the Subcommittee to act as primary reviewers of the proposed changes. The primary reviewer's job is to examine and test the proposed changes. The primary reviewers are to give their recommendation to the Subcommittee.

2.3.2.3 Formal Review of Proposed Minor Kit Modification – Six week review period

If the proposed changes were voted to be a Minor Kit Modification, then the Subcommittee chair will select at least two members of the committee to act as primary reviewers of the proposed changes. The primary reviewer's job is to examine and test the proposed changes. The primary reviewers are to give their recommendation to the committee no more than six weeks later.

2.3.2.4 Formal Rewiew by Subcomittee

Once the review period ends and the primary reviewers have given their recommendations, the subcommittee will vote on whether to accept the proposed changes into the TPCx-BB benchmark kit.

If the changes are accepted, then the changes will be added to the kit.

2.3.3 Kit Validation

Before any kit can be submitted for approval as a new revision of the TPCx-BB Benchmark Standard, any all changes must still pass the self-validation tests in the kit.

2.3.4 Classification of Major, Minor and Third Tier Kit Modifications

It is necessary to ensure that the kit remains in sync with fast changing industry and technology land scape. The guidelines below illustrate the current structure of the Kit and helps the Subcommittee to make a decision in a timely manner when evaluating a change proposal. These guidelines will help the Subcommittee do its due diligence and uses its discretion to classify and process the change proposals. Modifications to the kit are divided into three types that follow the Revision clasifications defined in the TPC Policies.

2.3.4.1 Major Kit Modifications:

Major Kit Modifications result in a significant change to the usecases or intent of the TPCx-BB Benchmark as to make results from the new version non-comparable with the results of the current TPCx-BB version.

These are a few examples of Major Kit Modificaitons:

- additions, deletions, and modifications to a Use Case
- changes to Primary Benchmark Metric
- changes which may alter the reference result set
- changes made to run rules and Benchmark execution process

2.3.4.2 Minor Kit Modifications:

Minor Kit Modifications do not significantly alter the reference result set, the primary benchmark metrics, or the use case. Results are still comparable to the prior version. A few examples of Minor Kit changes:

- addition of a new framework support
- bugfixes through the entire kit
- optimizations to the Framework specific code
- feature additions to Benchmark Driver
- modifications to tuning parameter files

- reference result set changes due to bug fixes
- framework feature support
- updates to independent library files
- changes to the Data generator to support features and bugfixes

2.3.4.3 Queries that use machine learning techniques

Queries that use machine learning techniques (clustering or classification) don't have a known correct answer set and so some other criteria must be applied to determine whether modifications are yielding results that should be considered comparable. There are two general categories of changes that could impact the machine learning queries:

1) Changes to the version/implementation of the SUT's machine learning library (for example a new version of the Spark MLLib library) without any changes to kit itself. The concern in this case is that a new version of the machine learning library could make a different tradeoff in accuracy vs performance compared to earlier versions. The following criteria will be applied to evaluate whether results using a new library version should be comparable to previous results:
    - Results using the new library version must be generated without any changes to code or parameters in the kit (in particular there can be no changes to the input data, the parameters to the algorithm (e.g. number of iterations, number of clusters for KMeans, algorithm initialization parameters including seeds for any random initialization, regularization parameters for classification algorithms, etc).
    - Results should only be considered comparable if the accuracy/evaluation metrics reported by the queries are comparable. For example the clustering queries report the sum of squared distances from cluster centers as an accuracy metric, and the classification queries report precision and AUC metrics. These metrics must demonstrate a level of accuracy for the new library implementation that is at least as good (within margin of error) as the accuracy of the earlier library version used in the comparison.

2) Introduction of new machine learning frameworks (not just new versions of the previously supported framework) that may require actual changes in the kit code or parameters. This case is more subjective, but the general guidelines for considering results from a new ML framework to be comparable are:
    - The same input data must be used
    - To the extent that the new framework accepts similar parameters to existing frameworks (number of iterations, number of clusters, regularization parameters), the values for these parameters should be similar to those used for existing frameworks. If there is a need for the parameters to be different there must be sufficient technical justification provided.
    - The new framework should be initialized using techniques that are comparable to the existing framework (e.g. for clustering the new framework should use the same random initialization approach).
    - The new framework should be capable of reporting the same accuracy/evaluation metrics (sum of squared distance, precision, AUC, etc) as existing ML frameworks and these metrics must demonstrate a level of accuracy for the new framework that is at least as good (within margin of error) as the accuracy of the earlier framework used in the comparison.

2.3.4.4 Third Tier Kit Modifications:

Third Tier Kit Modifications are those changes that clarify some confusing or ambiguous area of the kit, but does not alter the kit code or the use case. Results are still comparable to the prior version. These are a few example of Third Tier changes:

- changes in documentation

2.4 **Benchmark Run**

A valid run consists of separate tests run sequentially. These tests may not overlap in their execution times. For example, the start of Test 2 may not begin until Test 1 is complete, the start of Test 3 may not begin until Test 2 is complete, etc. All tests are initiated by the TPCx-BB master scripts which can be executed from any of the nodes in the **SUT**. The tests are listed below:

- Validation Test
- Load Test
- Power Test
- Throughput Test

2.4.1.1 Driver Parmeters used in the sequence of tasks during the test phases.

2.4.1.2 Validation test should be performed prior to the performance test with same configuration used for the performance SF being tested.

- TPCxBB_Validation.sh
- TPCxBB_Benchmarkrun.sh

2.4.1.3 The elapsed time for each test must be reported except Validation phase.

2.4.1.4 Tasks BENCHMARK_START and BENCHMARK_STOP determine the overall elapsed time for the benchmark execution.

2.4.2 **Validation Test**

The Engine validation performs data generation, load, power and validation run with scale factor 1 to perform an exact result validation against the reference result set in the kit . Validation phase ensures that the engine used by the test sponsor to produce the publication can match the reference result set generated. Validation test has three sub-phases.

2.4.2.1 Each Benchmark execution should successfully pass Validation phase as defined in clause 4.1.2

2.4.2.2 The elapsed time for Validation Test is not counted as part of Benchmark Execution.

2.4.2.3 Elapsed time for Validation Test is not included as part of Benchmark Metric calculation.

2.4.2.4 ENGINE_VALIDATION_DATAGENERATION: This phase generates a dataset at a fixed scale factor for 1 (SF1).

2.4.2.5 ENGINE_VALIDATION_LOAD_TEST: During this phase, the data generated at 2.4.2.4 will be loaded into the metastore.

2.4.2.6 ENGINE_VALIDATION_POWER_TEST: During this phase, all 30 queries will be run in sequence and the results are stored in the HDFS storage.

2.4.2.7 ENGINE_VALIDATION_RESULT_VALIDATION: During this automated phase, the benchmark driver compares the results from all 30 queries against a known reference results packaged with the kit.

2.4.3 Load Test

The first test loads the data into the metastore. The Load Test measures the time it takes to load the generated data into one of the engines' optimized storage formats (e.g., ORC or PARQUET).

2.4.4 Power Test

The second test determines the maximum speed an engine can process all 30 queries. The queries must run sequentially in ascending order, one after another.

2.4.5 Throughput Test

The throughput test gives insight into how the **SUT** behaves with an increased number of concurrent query processes. Throughput Test runs 30 queries using concurrent streams, each stream runs all 30 queries in a random order. The number of concurrent streams is configurable.

2.4.6 Query placement in throughput test

The automatic random shuffling always takes place when there is no entry in the BigBench properties for a certain throughput test stream. For shuffling, a Java RNG is used which is always seeded with the same seed at the beginning of the driver execution:

long seed = 1234567890L;
Random rnd = new Random(seed);
Whenever there is a missing entry for a throughput test stream, the list holding the query order is shuffled with this Java call:
List<Integer> queryList = ...
Collections.shuffle(queryList, rnd);
Before shuffling for the first time, the list is initialized as a copy of the power_test_0 list. This way, the query ordering is deterministic. Every shuffle generates the same query order.

2.5 **Benchmark Execution**

2.5.1 The TPCx-BB benchmark test consists of two runs, Run 1 and Run 2, which must follow the Run tests as described in Clause 2.4. No activities except file system cleanup are allowed between Run 1 and Run 2. The **Performance Run** is defined as the run with the lower TPCx-BB Performance Metric. The **Repeatability Run** is defined as the run with the higher TPCx-BB **Performance Metric**. The **Reported Performance Metric** is the TPCx-BB Performance Metric for the **Performance Run**. There should not be any interruption between the four phases, and all four phases should be run without intervention.

No part of the **SUT** may be restarted during the runs. If there is a non-recoverable error reported by any of the applications, operating system, or hardware in any of the four tests (Clause 2.4) or between Run 1 and Run 2, the run is considered invalid. If a recoverable error is detected in any of the tests, and is automatically dealt with or corrected by the applications, operating system, or hardware, then the run is considered valid provided the run meets all other requirements. However, manual intervention by the **Test Sponsor** is not allowed. If a recoverable error requires manual intervention to deal with or correct, then the run is considered invalid.

2.6 **Configuration and Tuning**

The **SUT** cannot be reconfigured, changed, or re-tuned by the **Test Sponsor** during or between any of the 4 tests described in Clause 2.4. Any manual tunings to the **SUT** must be performed before the beginning of the benchmark execution described in clause 2.4, and must be fully disclosed. Automated changes and tuning performed between any of the tests are allowed. Any changes to default tunings or parameters of the applications, **Operating Systems**, or hardware of the **SUT** must be disclosed.

Clause 3 – SYSTEM UNDER TEST

3.1 Logical Breakdown of System Under Test

The tested and **reported** configuration is composed of the hardware and software components that are employed in the TPCx-BB benchmark test and whose cost and performance are described by the benchmark metrics.

3.1.1 System Under Test

- big Data Benchmark and Driver Software: TPCx-BB kit provides fully integrated benchmark and driver software to run on SUT.
- compute Software: Distributed compute software runs on Compute Hardware providing required software capabilities to successfully execute the benchmark.
- data Storage Software: Data Storage software runs on Data Storage hardware providing required software to create, store, and access input, output, intermediate, and temp data during the benchmark execution.
- compute Hardware: compute hardware provides multi-device compute capable hardware to execute the benchmark.
- data Storage Hardware: Data Storage hardware provides data storage capability using various kinds of persistient storage mediums.
- network Hardware and Software: Network Hardware and software provides capability to connect hardware and software in the SUT to communicate and perform data transfer over the network.
- devices, for example compute devices and/or data storage devices, including hardware and software components
- any hardware and software devices of all networks required to connect and support the **SUT** systems
- Each deviceruns compute and storage software commercially available and supported.
- Device running driver hardware and software resides on a separate system facilitating the execution of the benchmark without interfering and influencing the SUT, this device , this device is part of the **SUT** and contains necessary SW and configuration to interact with the SUT and can be in form of Desktop, Notebook, or a Server.

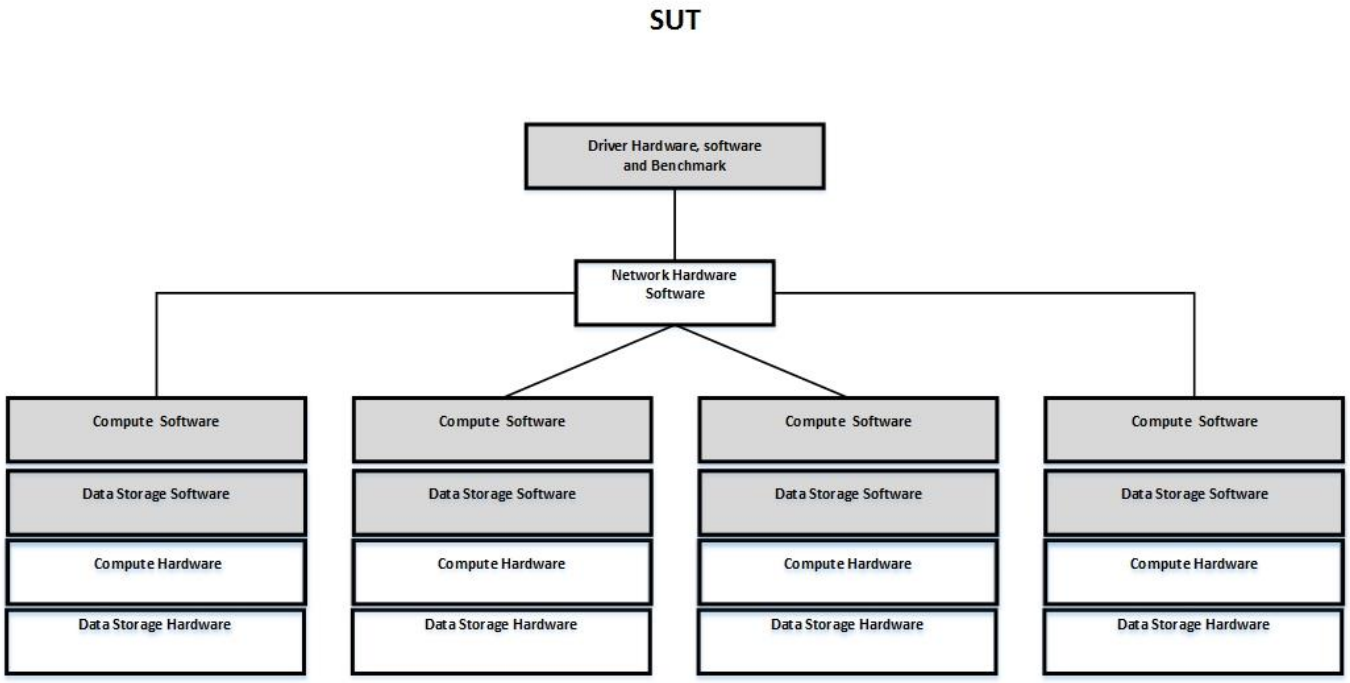


Figure 2 System under Test

3.1.2 Commercially Available Products

Except for the benchmark driver, all **SUT** components must be commercially available products. The source code of any non-commercially available products used to implement the SUT (including but not limited to scripts used to install, configure and tune the SUT) must be disclosed.

3.1.3 Data Durability

The following clauses describe required characterstics of the SUT. The failures described are not induced during the benchmark run.

3.1.3.1 Data Durability.

The System Under Test must be configured to satisfy the requirements for Data Durability described in this clause. Data Durability, is demonstrated by the SUT being able to maintain operations with full data access during and after the permanent irrecoverable failure of any single Durable Medium containing tables, input, output, or metadata.

Definition of Terms

- Data Durability: The ability to have no permanent data loss after the permanent irrecoverable failure of any single Durable Medium containing tables, input data, output data, or metadata.

  Durable Medium: A data storage medium that is either:

  a) An inherently non-volatile medium (e.g., magnetic disk, magnetic tape, optical disk, solid state disk, Phase Change Memory, or techonolgy similar to Phase Change Memory. etc.) or;
  b) A volatile medium with its own self-contained power supply that will retain and permit the transfer of data, before any data is lost, to an inherently non-volative medium after the failure of external power.

Comment: A configured and priced Uninterruptible Power Supply (UPS) is not considered external power.

Comment: DRAM can be considered a durable medium if it can preserve data long enough to satisfy the requirement (b) above. For example, if memory is accompanied by an Uninterruptible Power Supply, and the contents of memory can be transferred to an inherently non-volatile medium during the failure, then the memory is considered durable. Note that no distinction is made between main memory and memory performing similar permanent or temporary data storage in other parts of the system (e.g., disk controller caches).

- Metadata: Descriptive information about the database including names and definitions of tables, indexes, and other schema objects. Various terms commonly used to refer collectively to the metadata include metastore, information schema, data dictionary, or system catalog.

3.1.3.2 Data Durability Requirements

- The test sponsor must guarantee that the test system will not loose data due to a permanent irrecoverable failure of any single durable medium containing TPCx-BB data, e.g. tables, results, or metadata. If required by the Pre-publishing Sub-Committee or an Auditor, Test Sponsors SUT must demonstrate that the Benchmark input data, Metastore, and output data does not change even after SUT reboots, or after failure of a single Durable Medium.
  a) For HDFS this can be done by running $hadoop dfsadmin –report or $hadoop fs –du –h %Benchmark Table Data path on HDFS% and %metastore% - Appendix H
  b) For other storage systems not using HDFS durabilityhas to be proved by providing a report similar to above HDFS report.

Comment: Queries may fail due to a permanent irrecoverable failure of any single durable medium containing TPCx-BB data, e.g. tables, results, or metadata.

Comment: Input Data, Metastore Data and Output Data must be set to replication factor minimum of three for HDFS on JBOD and other systems must demonstrate data durability equivalent to using replication factor three in HDFS.

Clause 4 -- SCALE FACTORS AND RESULT VALIDATION

4.1 Scale Factor

4.1.1.1 The TPCx-BB benchmark defines a set of discrete scaling points ("scale factors") based on the approximate size of the raw data produced by the datagenerator in Gigabytes.

4.1.1.2 Each defined scale factor has an associated value for SF, a unit-less quantity, roughly equivalent to the number of gigabytes of data present on the storage. The relationship between scale factors and SF is summarized in the below table 4-1 Scale Factor and SF.

4.1.1.3 Test sponsors may choose any scale factor from the defined series. No other scale factors may be used for a TPCx-BB result. Except SF1 which is used for result validation only.

Scale Factor	Approximate Size in GB
1	0.9
1000	923
3000	2794
10000	9450
30000	28740
100000	96923
300000	292792
1000000	989482

Table 4-1

4.1.1.4 Below Table 4-2 provides an example of table sizes from SF 1-1000000.

4.1.1.5 Test Dataset is the dataset used to execute the Perforamnce Test and the Repeatiblilty Test.

4.1.1.6 The required dataset size for each permissible scale factor and each table in the test database is detailed in Table below.

4.1.1.7 The SF 1 is used solely for the Result validation for a test and is included here as reference only.

4.1.1.8 The dataset size information provided is an estimate, and may vary from one benchmark submission to another depending on the data storage compression methods used. The estimate is provided solely to assist benchmark sponsors in the sizing of benchmark configurations. The datagenerator uses linear, log 1.5, log5, and sqrt scaling, depending on individual tables. The ratio of scaling between nominal scaling and generated data for a given SF is approximately 1.0.

	SF							
Table	1	1000	3000	10000	30000	100000	300000	1000000
customer	13278712	430117660	747445365	1367378006	2377941518	4363809796	7582697935	13871582504
customer_address	5387229	172752087	300060160	548756695	951300132	1743384578	3027719105	5536928616
customer_demographic	77753083	77753083	77753083	77753083	77753083	77753083	77753083	77753083
date_dim	15395800	15395800	15395800	15395800	15395800	15395800	15395800	15395800
household_demograph	155100	155100	155100	155100	155100	155100	155100	155100
income_band	327	327	327	327	327	327	327	327
inventory	424943678	77664727242	156238037400	331739218142	633776875697	1293521273503	2415494531507	4925822989989
item	6790524	215463472	373234327	682287215	1182602573	2159997620	3742044474	6840618357
item_marketprices	3711674	126239429	219873857	406723339	799689718	1473792473	2564881978	4735682637
product_reviews	56355389	3021887410	6384172302	16859932281	44026790100	133125073965	377223465009	1218098621548
promotion	46244	850113	978213	1121277	1252057	1394319	1523477	1670104
reason	3086	56556	65001	74215	82954	92271	101043	110695
ship_mode	1220	1220	1220	1220	1220	1220	1220	1220
store	3380	105396	184834	338015	586004	1070158	1854140	3385951
store_returns	4337400	8685981116	27127231309	93595021305	288053401659	981301766589	2984443417935	10106945551151
store_sales	92764700	183354740939	571831621868	1969036493318	6047041807477	20553753727503	62391854918853	210851767590672
time_dim	5107780	5107780	5107780	5107780	5107780	5107780	5107780	5107780
warehouse	580	3092	3565	4051	4397	4846	5196	5696
web_clickstreams	178722140	376294553420	1178035219723	4106502744827	12740495809700	43441045579753	132005626695048	450199467042795
web_page	8040	148030	169870	193999	216895	241948	264069	288927
web_returns	5383513	10651735211	33312613665	114760697723	352909055052	1203357418071	3662345334934	12380413321989
web_sales	133791863	262743522378	819457678582	2815131490325	8628922472597	29308035558237	88938499519180	299769274971262
web_site	8807	8806	8807	8807	8807	8807	8807	8807
Total Generated	1023950270	923461306667	2794127012158	9450730896850	28740638310647	96923982607747	292792507396000	989482878785010
Total in GB	1.02395027	923.4613067	2794.127012	9451	28740.63831	96923.98261	292792.5074	989482.8788
Nominal	1000000000	1000000000000	3000000000000	10000000000000	30000000000000	100000000000000	300000000000000	1000000000000000
Ratio Nominal to Gener	0.976609929	1.082882404	1.073680612	1.058119219	1.043818153	1.031736391	1.024616383	1.010628907

4-2 Dataset table sizes.

4.1.2 **Result Validation**

4.1.2.1 The validation result set for SF1 is the reference result used to validate the test SUT for result correctness during the test.

4.1.2.2 The intent is that the functionality is exercised by running the validation queries against SF1 and compare it against the reference result set packaged with the benchmark kit. This is the same set the one that is exercised against the SUT before the the performance test as part of Engine Validation phase. Validation phases are described in 2.4.2.

4.1.2.3 Populate the SUT with SF1 dataset and schema information.

4.1.2.4 Execute the queries using standard query parameters as defined in the Queryparameters.sql see Appendix C Verify the report generated by the driver matching the output to the reference result set.

4.1.2.5 The generated report shall state that the output matches the reference result set for all 30 queries.

4.1.3 Output data for Validation phase.

4.1.3.1 After the execution of validation phase, a query returns one or more rows. The rows are called the output data.

4.1.3.2 Output data shall adhere to the following guidelines.

4.1.3.3 Output appears in the form specified by the query description for queries outputting data from SQL and procedural jobs.

4.1.3.4 Data will be formatted using the TPCx-BB result validation tool.

4.1.3.5 Non-integer numbers including prices are expressed in decimal notation with two digits behind the decimal point.

4.1.3.6 Software library versions for Machine Learning and Natural Language Processing are defined below and must be used to in the result validation.

Library	Distro	Version
OpenNLP Tools	Apache	1.6.0
OpenNLP-Maxent	Sourceforge	3.0.3

4.1.3.7 Strings are case-sensitive and must be displayed as such. Leading or trailing blanks are acceptable.

4.1.3.8 The amount of white space between columns is not specified.

4.1.3.9 The order of a query output data must match the order of the validation output data, except for queries that do not specify an order for their output data.

4.1.3.10 TPCx-BB automates result validation by strictly matching the outputed results for SF1 with reference resultset provided with the kit, the driver looks for exact result match. However, to accomodate situations where the validation results fails to match the reference result set, output data from a SF1 validation test shall adhere to the following rules to still qualify as valid test run.

- All tuples are part of the result and have the same values as reference result.
- An external post processing sorting with a bash script to bring the tuples into total global order for validation against the reference result set is acceptable.
- Use of the orderby feature, if supported to post process the data, is acceptable.
- For singleton column values and results from COUNT aggregates, the values must exactly match the corresponding values in the expected answer sets.
- For other numeric expressions including aggregates, the resulting values must be within 1% of the corresponding values in the expected answer sets.

Comment: Hadoop SQL frameworks do not yet fully support the complete SQL standard. So does Machine learning libraries which are still evolving4.1.3.10 provide frameworks to be run a successful test where results are present but can not match strict reference result set due to missing feature and ordering.(E.g. Kit uses hive.optimize.sampling.orderby which is available in HIVE 0.14 but not in other versions).

4.1.3.11 For all other scale factors, the benchmark driver at the end of the benchmark execution run checks for the presence of output data from the power phase and throughput phase in order to qualify successful benchmark execution.

Clause 5 **METRICS**

5.1 **TPCx-BB Primary Metrics**

TPCx-BB defines the following primary metrics:

- **BBQpm@SF** , the **Performance Metric**, reflecting the TPCx-BB queries per minute throughput; where SF is the Scale Factor (see Clause 4.1)
- $/BBQpm@SF , the **Price/Performance Metric**
- System **Availability Date** as defined by the TPC Pricing Specification
- When TPC-Energy option is chosen for reporting, the TPCx-BB energy metric reports the power per performance and is expressed as Watts/BBpm@SF. TPC energy specification located at www.tpc.org. TPC-Energy metric is not mandatory for this TPCx-BB.

5.2 **Performance Metric (BBQpm@SF)**

The **Performance Metric** of the TPCx-BB benchmark, BBQpm@SF, is computed by combining metric components representing the load, power, and throughput tests.

- SF is the scale factor (4.1).
- T_{LD} is the load factor computed as:

$$T_{LD} = 0.1 * T_{Load}$$

where T_{Load} is the elapsed time of the Load Test (2.4.3) in seconds and 0.1 is a multiplication factor to adjust the contribution of Load test in the performance metric.

T_{PT} is the geometric mean of the elapsed time Q in seconds of each of the 30 queries as measured during the Power Test (2.4.4), multiplied by the number of queries in the benchmark :

$$T_{PT} = M * \sqrt[M]{\prod_{i=1}^{i=30} Q(i)}$$

where Q(i) is the elapsed time in seconds of query *i* during the Power Test and *M* is the number of queries in the Benchmark.

T_{TT} is the throughput test metric computed as the total elapsed time of the throughput test divided by the number of streams as measured during the Throughput Test (2.4.5).

- T_{Tput} is the elapsed time of all streams from the Throughput Test.

- n is the number of streams in the Throughput Test (2.4.5).

$$T_{TT} = \frac{1}{n} T_{Tput}$$

Given the above definitions the overall **Performance Metric** BBQpm@SF is defined as:

$$BBQpm@SF = \frac{SF * 60 * M}{T_{LD} + \sqrt[2]{T_{PT} * T_{TT}}}$$

where M is the number of queries in the Benchmark, SF is the Scale Factor and the factor of 60 in minutes in an hour.

5.3 Price Performance Metric ($/BBQpm@SF)

The **Price/Performance Metric** for the benchmark is defined as:

$$\$ / BBQpm@SF = \frac{C}{BBQpm@SF}$$

Where:

- C is the total cost of ownership of the **SUT**.

If a benchmark configuration is priced in a currency other than US dollars, the units of the **Price/Performance Metrics** must be adjusted to employ the appropriate currency.

5.4 System Availability Date

The System **Availability Date** is defined in the TPC Pricing Specification.

5.5 Fair Metric Comparison

A TPCx-BB **Result** is only comparable with other TPCx-BB **Result**s of the same **Scale Factor** (see Clause 4.1).

- **Results** at the different Scale Factors are not comparable, due to the substantially different computational challenges found at different data volumes. Similarly, the system **Price/Performance Metric** may not scale down linearly with a decrease in dataset size due to configuration changes required by changes in dataset size.

- If **Results** measured against different Scale Factors appear in a printed or electronic communication, then each reference to a **Result** or metric must clearly indicate the Scale Factors against which it was obtained. In particular, all textual references to TPCx-BB metrics (**Performance Metric** or **Price/Performance Metric**) appearing must be expressed in the form that includes the Scale Factor as an integral part of the metric's name; i.e. including the "@SF" suffix. This applies to metrics quoted in text or tables as well as those used to annotate charts or graphs. If metrics are presented in graphical form, then the Scale Factor on which metric is based must be immediately discernible either by appropriate axis labeling or data point labeling. In addition, the Results must be accompanied by a disclaimer stating: "The TPC believes that comparisons of TPCx-BB Results measured against different Scale Factors are misleading and discourages such comparisons".

5.6 **Secondary Metrics**

- Load time
- Power Test total elapsed time
- Throughput Test Elapsed time
- Elapsed time for each query in Power test and Throughput test

Clause 6 **PRICING**

6.1 **Introduction**

This section defines the components, functional requirements of what is priced, and what **Substitutions** are allowed. Rules for pricing the **Priced System** and associated software and maintenance are included in the TPC Pricing Specification located at www.tpc.org.

6.2 **Priced Configuration**

The system to be priced shall include the hardware and software components present in the **System Under Test** (**SUT**), a communication interface that can support user interface devices, additional operational components configured on the test system, and maintenance on all of the above

Calculation of the priced system consists of:

- price of the **SUT** as tested and defined in Clause 3.1
- price of all software licenses for software used in the **SUT**
- price of a communication interface capable of supporting the required number of user interface devices defined in Clause 6.3
- price of additional products (software or hardware) required for customary operation, administration and maintenance of the **SUT** for a period of 3 years
- price of all products required to create, execute, administer, and maintain the executables or necessary to create and populate the test environment

Specifically excluded from the priced system calculation are:

- end-user communication devices and related cables, connectors, and switches
- equipment and tools used exclusively in the production of the **Full Disclosure Report**

6.3 **Additional Operational Components**

Additional products included on a customer installed configuration are also to be included in the priced system if explicitly required for the operation, administration, or maintenance of the priced system. Examples of such products are:

- operator console
- user interface terminal
- CD drive
- software, if required for initial load or maintenance updates
- all cables used to connect components of the **SUT** (except as noted in Clause 6.2)

6.4 **Allowable Substitutions**

6.4.1 **Substitution** is defined as a deliberate act to replace components of the **Priced Configuration** by the **Test Sponsor** as a result of failing the availability requirements of the TPC Pricing Specification or when the part number for a component changes.

Comment: Corrections or "fixes" to components of the **Priced Configuration** are often required during the life of products. These changes are not considered **Substitutions** so long as the part number of the priced component does not change. Suppliers of hardware and software may update the components of the **Priced Configuration**, but these updates must not negatively impact the reported **Performance Metric** or numerical quantities more than two percent.

The following are not considered **Substitutions**:

- Software patches to resolve a security vulnerability
- Silicon revision to correct errors
- New supplier of functionally equivalent components (for example memory chips, disk drives, etc).

6.4.2 Some hardware components of the **Priced Configuration** may be substituted after the **Test Sponsor** has demonstrated to the **Auditor's** satisfaction that the substituting components do not negatively impact the reported **Performance Metric** or numerical quantities. All **Substitutions** must be **Reported** in the **FDR** and noted by the **Auditor** . The following hardware components may be substituted:

- durable medium (for example disk drives) and cables
- durable medium enclosure
- network interface card
- router
- bridge
- repeater

Comment: If any hardware component is substituted, then the result must be audited by an Auditor (see Clause 9.3.1).

Clause 7 – ENERGY – (OPTIONAL)

This section contains the rules and methodologies for measuring and reporting energy metric in TPCx-BB benchmarks. Please refer to Energy Specification on

http://www.tpc.org/TPC_Documents_Current_Versions/pdf/TPC%20Energy%20Specification%201.5.0.pdf

Clause 8 -- FULL DISCLOSURE REPORT

8.1 Full Disclosure Report Requirements

8.1.1 A **Full Disclosure Report (FDR)** is required. This section specifies the requirements of the **FDR**.

The **FDR** is a zip file of a directory structure containing the following:

- a **Report** in Adobe Acrobat PDF format
- an **Executive Summary Statement** in Adobe Acrobat PDF format
- The **Supporting Files** consisting of any source files or scripts modified by the **Test Sponsor** and the output files generated by the TPCx-BB kit. Requirements for the **FDR** file directory structure are described below.

8.1.2 The **FDR** should be sufficient to allow an interested reader to evaluate and, if necessary, recreate an implementation of TPCx-BB **Result** given the appropriate hardware and software products. If any sections in the **FDR** refer to another section of the **FDR**, the names of the referenced scripts/programs must be clearly labeled in each section. Unless explicitly stated otherwise, "disclosed" or "reported" refers to disclosing or reporting in the **FDR**.

Comment: Since the building test environment may consist of a set of scripts and corresponding input files, it is important to disclose and clearly identify, by name, scripts and input files in the **FDR**.

8.2 Format Guidelines

8.2.1 While established practice or practical limitations may cause a particular benchmark disclosure to differ from the examples provided in various small ways, every effort should be made to conform to the format guidelines. The intent is to make it as easy as possible for a reviewer to read, compare, and evaluate material in different benchmark disclosures.

8.2.2 All sections of the report, including appendices, must be printed using font sizes of a minimum of 8 points.

8.2.3 The **Executive Summary** must be included near the beginning of the **Report**.

8.2.4 The order and titles of sections in the **Report** and **Supporting Files** must correspond with the order and titles of sections from the TPCx-BB Specification (i.e., this document). The intent is to make it as easy as possible for readers to compare and contrast material in different Reports.

8.2.5 The directory structure of the **FDR** has three folders:

- *ExecutiveSummaryStatement* - contains the **Executive Summary** statement
- *Report* - contains the **Report**
- *SupportingFiles* - contains the **Supporting Files**

8.3 General Items

8.3.1 The **FDR** must follow all reporting rules of the TPC Pricing Specification, located at www.tpc.org. For clarity and readability, the TPC Pricing Specification requirements may be repeated in the TPCx-BB Specification.

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

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

- Configuration parameters and options for server, storage, network and other hardware components used by the **SUT**.
- Configuration parameters and options for the **Operating System** and file system components used by the **SUT**.
- Configuration parameters and options for any other software components (e.g. compiler optimization options) used by the **SUT**.

Comment 1: In the event that some parameters and options are set multiple times, it must be easily discernible by an interested reader when the parameter or option was modified and what new value it received each time.

Comment 2: This requirement can be satisfied by providing a full list of all parameters and options, as long as all those that have been modified from their default values have been clearly identified and these parameters and options are only set once.

8.3.4 Explicit response to individual disclosure requirements specified in the body of earlier sections of this document must be provided.

8.3.5 Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:

- total number of nodes used
- total number and type of processors used/total number of cores used/total number of threads used (including sizes of L2 and L3 caches)
- size of allocated memory, and any specific mapping/partitioning of memory unique to the test;
- number and type of disk units (and controllers, if applicable)
- number of channels or bus connections to disk units, including their protocol type
- number of LAN (for example, Ethernet) connections and speed for switches and other hardware components physically used in the test or are incorporated into the pricing structure
- type and the run-time execution location of software components

The following Figure 3 Sample Configuration Diagram illustrates a measured benchmark configuration where each server using Ethernet, an external driver, and two processors each with sixteen cores and two threads per core in the **SUT**. Note that this diagram does not depict or imply any optimal configuration for the TPCx-BB benchmark measurement.

Depending on the implementation of the **SUT**, the Name Node, Secondary Name Node, Data Node, Job/Task Tracker, Resource Manager/Node Manager, etc. or the functional equivalents must be specified in the diagram.

Figure 3 Sample Configuration Diagram

- n x Server Rack in scale out configuration.
- n x My Server Model B, 4/32/64 My CPU Model Z (2.7 GHz, 20MB cache, 130W), 128GB, My RAID Controller with 1GB BBWC
- n x My Storage Array Model A with 8 X 1TB 10K SAS HDD
- nx My Switch Model X 10GbE
- Nx Top of the Rack switch.

Comment: Detailed diagrams for system configurations and architectures can vary widely, and it is impossible to provide exact guidelines suitable for all implementations. The intent here is to describe the system components and connections in sufficient detail to allow independent reconstruction of the measurement environment. This example diagram shows homogeneous nodes. This does not preclude tests sponsors from using heterogeneous nodes as long as the system diagram reflects the correct system configuration.

8.3.6 The distribution of dataset across all media must be explicitly described using a format similar to that shown in the following example for the tested system.

8.4

Table 1 Example Layout Description

Server	Role(s)	Count	Virtual	Host Name(s)	HW/SWConfiguration	Storage Setup
Worker	Yarn NM/Hive Server/Spark Worker	50	N	TPCx-BB[100] - [BB150]	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version, Framework SW Model and version. Details of Additional HW/SW if any.	**OS: Model x GB SSD, Intermediate/Shuffle/Temp Data: x Model x GB SSD,Distributed FS: x Model 12x SAS/SATA Harddrive/**
Distro Manger	Hadoop Manager	1	N	TPCx-BB-CDH	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version, Framework SW Model and version. Details of Additional HW/SW if any.	**OS: Model x GB SSD.**
Benchmark SUT Driver	YARN/SPARK,HIVE Gateway	1	N	TPCxBB-Driver1	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version, Framework SW Model and version. Details of Additional HW/SW if any.	
Name Node/Resource Manager	YARN/NN/ZooKeeper	1	N	TPCx-BB_PNN1	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version, Framework SW Model and version. Details of Additional HW/SW if any.	

8.4.1 The distribution of various software components across the system must be explicitly described using a format similar to that shown in the following example for both the tested and priced systems.

Comment: The software components might vary from implementation to implementation.

8.4.2 The distributed file system implementation (for example Apache HDFS, Red Hat Storage, IBM GPFS, EMC Isilon OneFS) and corresponding Hadoop File System API version must be disclosed.

8.4.3 The Engine implementation (for example, Apache Map/Reduce, Spark, Flink, IBM Platform Symphony) and corresponding version must be disclosed.

8.4.4 Frameworks and Engine used in the benchmark should be disclosed in the report.

8.4.5 If there were any additional vendor supported patches applied to the SUT, details of such patches should be disclosed.

8.5 **Workload Related Items**

8.5.1 Script or text used to set for all hardware and software tunable parameters must be **Included** in the **Report**.

8.5.2 Version number of TPCx-BB kit must be **Included** in the **Report**.

8.5.3 The run report generated by TPCx-BB benchmark kit must be **reported** in the **Report**.

8.5.4 Elapsed times of all power and throughput queries needs to be reported from the Performance run, grouped respectively as Structured, semi-structured and unstructured buckets. For groupings please see clause B.5 in Appendix B.

8.5.5 Query completion times for Individual queries run as part of Performance Run should be included in the report.

8.5.6 Output report from successful Engine Validation Run (2.4.2.1)

8.5.7 Global framework parameter settings files per (2.2.4.4)

8.5.8 Kit modification files (2.2.5)

8.6 **SUT Related Items**

8.6.1 Specialized Hardware/Software used in the SUT must be included.

8.6.2 All framework configuration files from SUT, for the performance run E.g Yarn-Site.xml, Hdfs-site.xml etc.

8.6.3 SUT Environment info in form of envinfo.log from a representative node from every role in the server. See Appendix G

8.6.4 The data storage ratio must be disclosed. It is computed by dividing the total physical data storage present in the **Priced Configuration** (expressed in TB) by the chosen Scale Factor as defined in Clause 4.1. Let r be the ratio. The **Reported** value for r must be rounded to the nearest 0.01. That is, reported value=round(r, 2). For example, a system configured with 96 disks of 1TB capacity for a 1TB Scale Factor has a data storage ratio of 96.

Comment: For the reporting of configured disk capacity, terabyte (TB) is defined to be 10^12 bytes.

8.6.5 The Scale Factor to memory ratio must be disclosed. It is computed by dividing the Scale Factor by the total physical memory present in the **Priced Configuration**. Let r be this ratio. The **Reported** ratio must be rounded to the nearest 0.01. That is, reported value=round(r,2). For example, a system configured with 1TB of physical memory for a 10TB Scale Factor has a memory ratio of 10.00.

8.7 Metrics and Scale Factors

8.7.1 The **Reported Performance Metric** (BBQpm@SF for the **Performance Run**) must be disclosed in the **Report**.

8.7.2 The **Performance Metric** (BBQpm@SF) for the **Repeatability Run** must be disclosed in the **Report**.

8.7.3 The **Price/Performance Metric** ($/BBQpm@SF) for the **Performance Run** must be disclosed in the **Report**.

8.7.4 The Scale Factor used for the **Result** must be disclosed in the **Report.**

8.7.5 The number of streams in the throughput run used for the **Result** must be disclosed in the **Report.**

8.7.6 The total elapsed time for the execution of the **Performance Run** and **Repeatability Run** must be disclosed in the **Report.**

8.7.7 The time for each of the three phases (excluding validation) (Clause 2.4) must be disclosed for the **Performance Run** and **Repeatability Run.**

8.8 Audit Related Items

If the benchmark is audited by an independent **Auditor**, the **Auditor's** agency name, address, phone number, and **Attestation Letter** with a brief audit summary report indicating compliance must be included in the **Report**. A statement should be included specifying whom to contact in order to obtain further information regarding the audit process.

8.9 Executive Summary Statement

The **Executive Summary** is meant to be a high level overview of a TPCx-BB implementation. It should provide the salient characteristics of a benchmark execution (metrics, configuration, pricing, etc.) without the exhaustive detail found in the **FDR**. When the TPC-Energy optional reporting is selected by the **Test Sponsor**, the additional requirements and format of TPC-Energy related items in the **Executive Summary** are included in the TPC Energy Specification, located at www.tpc.org.

8.9.1 The **Executive Summary** has three components:

- Implementation Overview
- Pricing Spreadsheet
- Numerical Quantities

8.9.2 Page Layout

Each component of the **Executive Summary** should appear on a page by itself. Each page should use a standard header and format, including:

- 1/2 inch margins, top and bottom
- 3/4 inch left margin, 1/2 inch right margin
- 2 pt. frame around the body of the page. All interior lines should be 1 pt.

8.9.3 **Implementation Overview**

The implementation overview page contains five sets of data, each laid out across the page as a sequence of boxes using 1 pt. rule, with a title above the required quantity. Both titles and quantities should use a 9-12 pt. Times font unless otherwise noted.

8.9.3.1 The first section contains information about the sponsor and system identification.

Table 2 Sponsor and System Identification

Title	Font
Sponsor Name or Logo	16-20 pt. Bold (for Name)
System Identification	16-20 pt. Bold
Version Numbers for TPCx-BB, TPC-Pricing and TPC-Energy (if reported)	16-20 pt. Bold
Report Date	16-20 pt. Bold

Comment 1: It is permissible to use or include company logos when identifying the sponsor.

Comment 2: The report date must be disclosed with a precision of 1 day. The precise format is left to the **Test Sponsor**.

8.9.3.2 The second section contains the Total System Cost, the TPCx-BB **Reported Performance Metric** and the **Price/Performance Metric**.

Table 3 Benchmark Results

Title	Quantity	Precision	Font
Total System Cost	3 yr. Cost of ownership (see Clause 6.2)	1	16-20 pt. Bold
Reported Performance Metric	BBQpm (5.2)	0.01	16-20 pt. Bold
Price/Performance	$/ BBQpm (5.3)	0.01	16-20 pt. Bold
SF (Scale Factor)	SF in as defined in 4.1	1	16-20pt. Bold
Streams	Concurrent Streams used in 2.4.5	1	16-20pt. Bold

Depending on the currency used for publication this sign has to be exchanged with the ISO currency symbol.

8.9.3.3 The third section contains detailed the system configuration.

Table 4 System Configuration Information

Title	Quantity	Font

Framework /Engine Software	Product name andProduct Version	9-12 pt. Times
Operating System	Product name, **Software Version** of OS, File System Type and Version	9-12 pt. Times
Other Software	Product name and **Software Version** of other software components (example Java)	9-12 pt. Times
System **Availability Date**	The **Availability Date** of the system, defined in the TPC Pricing Specification Clause 6	9-12 pt. Times

Comment: The **Software Version** must uniquely identify the orderable software product referenced in the **Priced Configuration** (for example, RALF/2000 4.2.1).

8.9.3.4 The fourth section contains the storage and memory ratios, see Clause 8.6.

Table 5 Storage and Memory Ratios

Title	Precision	Font
Physical Storage/Scale Factor	0.01	9-12 pt. Times
Scale Factor/Physical Memory	0.01	9-12 pt. Times

8.9.3.5 The fifth section contains the components, including:

- total number of nodes used/total number of processors used with their types and speeds in GHz/ total number of cores used/total number of threads used, see Clause 8.3.5
- main and cache memory sizes
- network speed and topology (e.g Top of the Rack switch, in-rack switch)
- storage type, quantity and configuraok tion

8.9.4 **Pricing Spreadsheet**

8.9.4.1 The major categories in the Price Spreadsheet, as appropriate, are as follows:

- network(s)
- server(s) /node(s)
- storage
- software

8.9.4.2 Discounts (may optionally be included with above major category subtotal calculations).

8.9.5 **Numerical Quantities Summary**

8.9.5.1 The Numerical Quantities Summary page contains three sets of data, presented in tabular form, detailing the Load Phase, Power Phase, and throughput phase execution timings for the **Performance Run** and **Repeatability Run**. Each set of data should be headed by its given title and clearly separated from the other tables.

8.9.5.2 The first section contains the Scale Factor, Number of streams, and Engine Validation Phase status for the reported benchmark execution **Result**.

8.9.5.3 The second section contains measurement results and metric from the **Performance Run**.

Table 6 Measurement Results for Performance Run

Performance Run	
Item Title	**Precision**
Benchmark Overall Run Start Time	yyyy-mm-dd hh:mm:ss
Benchmark Overall Run End Time	yyyy-mm-dd hh:mm:ss
Benchmark Overall Run Total Elapsed Time	ss.sss
Start of Load Test	yyyy-mm-dd hh:mm:ss
End of Load Test	yyyy-mm-dd hh:mm:ss
Load Test Elapsed Time	ss.sss
Start of Power Test	yyyy-mm-dd hh:mm:ss
End of Power Test	yyyy-mm-dd hh:mm:ss
Power Test Elapsed Time	ss.sss
Throughput Test	yyyy-mm-dd hh:mm:ss
Throughput Test	yyyy-mm-dd hh:mm:ss
Throughput Test Elapsed Time	ss.sss
Performance Metric (BBQpm@SF)	x,xxx.xx

8.9.5.4 The third section contains the measurement results for the **Repeatability Run**. See Table 6 for contents and precision.

8.9.6 **TPCx-BB Run Report – 2.2.3**

The run report from TPCx-BB must be included in the **Executive Summary.**

8.10 Availability of the Full Disclosure Report

8.10.1 The **Full Disclosure Report** must be readily available to the public. The **Report** and must be made available when the **Result** is made public. In order to use the phrase "TPCx-BB Benchmark", the **Full Disclosure Report** must have been previously submitted electronically to the TPC using the procedure described in the TPC Policies and Guidelines document.

8.10.2 The official **Full Disclosure Report** must be available in English but may be translated to additional languages.

8.11 Revisions to the Full Disclosure Report

The requirements for a revision to an **FDR** are specified in the TPC Pricing Specification.

Clause 9 – AUDITING

9.1 TPC Pricing

All audited requirements specified in the version of TPC pricing Specification located at www.tpc.org must be followed. The TPCx-BB pricing information included in the Report must be audited by a TPC certified Auditor or by the sub-committee part of pre-publication board. Test Sponsor should submit the Pricing data specified in the version of TPC Pricing Specification located at www.tpc.org.

9.2 Optional TPC-Energy Results

When the TPC-Energy optional reporting is selected by the **Test Sponsor**, the rules for auditing of TPC-Energy related items are included in the TPC Energy Specification located at www.tpc.org. If TPC-Energy metrics are **Reported**, the TPCx-BB Energy results must be audited by a TPC-Energy certified **Auditor**.

9.3 General Rules

Before publication, a TPCx-BB **Result** must be certified to be compliant with the spirt and letter of the TPCx-BB Benchmark Standard by an Independent Certified TPC Auditor or a TPCx-BB Pre-Publication Board. The **Test Sponsor** may chose the certification be performed by either by a Certified TPC Auditor or a Pre-Publication Board.

9.3.1 Independent Audit

9.3.1.1 The term independent is defined as "the outcome of the benchmark carries no financial benefit to the auditing agency other than fees earned directly related to the audit." The independent audit of the benchmark must meet all of the following criteria:

- The **Auditor** holds an active TPC certification for the TPCx-BB benchmark.
- The **Auditor** must be independent from the **Test Sponsor** in that the outcome of the benchmark carries no financial benefit to the **Auditor**, other than fees earned as a compensation for performing the audit.
- The **Auditor** is not allowed to be financially related to the **Test Sponsor** or to any one of the suppliers of a measured/priced component (e.g., the **Auditor** cannot be an employee of an entity owned wholly or in part by the **Test Sponsor** or by the supplier of a benchmarked component, and the **Auditor** cannot own a significant share of stocks from the **Test Sponsor** or from the supplier of any benchmarked component, etc.)
- The **Auditor** is not allowed to have supplied any performance consulting for the TPCx-BB **Result** under audit.

9.3.1.2 The **Auditor**'s opinion regarding the compliance of a **Result** must be consigned in an **Attestation Letter** delivered directly to the **Test Sponsor**. To document that a **Result** has been audited, the **Attestation Letter** must be included in the **Report** and made readily available to the public. Upon request, and after approval from the **Test Sponsor**, a detailed audit report may be produced by the **Auditor**.

9.3.2 Pre-Publication Board

The term Pre-Publication Board as defined by the TPC Policies consists of one or more individuals that have been chosen by the TPCx-BB Benchmark Subcommittee to certify **Results** for publication. For TPCx-BB **Results** the Pre-Publication Board consists of 3 members from the TPCx-BB Benchmark Subcommittee. Each member serves a period of three months. The membership will be rotated through the TPCx-BB Benchmark Subcommittee membership. The submission is confidential to the Pre-Publication Board until the **Result** is published. The Pre-Publication Board must complete the review within 10 business days. If no issues are raised within the 10 business day period, the **Result** is considered certified for publication.

9.3.3 Results Based on Existing TPCx-BB Results

A **Test Sponsor** can demonstrate compliance of a new **Result** produced without running any performance test by referring to the certification of a **Result**, if the following conditions are all met:

- The referenced **Result** has already been published by the same or by another **Test Sponsor**.
- The new **Result** must have the same hardware and software architecture and configuration as the referenced **Result**. The only exceptions allowed are for elements not involved in the processing logic of the **SUT** (e.g., number of peripheral slots, power supply, cabinetry, fans, etc.)
- The **Test Sponsor** of the already published **Result** gives written approval for its use as referenced by the **Test Sponsor** of the new **Result**.
- The **Auditor** verifies that there are no significant functional differences between the priced components used for both **Results** (i.e., differences are limited to labeling, packaging and pricing.)
- The **Auditor** or Pre-Publication Board reviews the **FDR** of the new **Result** for compliance.
- If certification is performed by an independent **Auditor**, a new **Attestation Letter** must be included in the **Report** of the new **Result**.

Comment 1: The intent of this clause is to allow publication of benchmarks for systems with different packaging and model numbers that are considered to be identical using the same benchmark run. For example, a rack mountable system and a freestanding system with identical electronics can use the same benchmark run for publication, with, appropriate changes in pricing.

Comment 2: Although it should be apparent to a careful reader that the **FDR** for the two **Results** are based on the same set of performance tests, the **FDR** for the new **Result** is not required to explicitly state that it is based on the performance tests of another published **Result**.

Comment 3: When more than one **Result** is published based on the same set of performance tests, only one of the **Results** from this group can occupy a numbered slot in each of the benchmark **Result** "Top Ten" lists published by the TPC. The **Test Sponsors** of this group of **Results** must all agree on which **Result** from the group will occupy the single slot. In case of disagreement among the **Test Sponsors**, the decision will default to the **Test Sponsor** of the earliest publication from the group.

9.4 Audit Checklist

A generic audit checklist is provided as part of this specification. The generic audit checklist specifies the requirements that must be checked to ensure a **Result** is compliant with the TPCx-BB Specification. Not only should the TPCx-BB requirements be checked for accuracy but the **Auditor** or Pre-Publication Board must ensure that the **FDR** accurately reflects the **Result**.

Comment: An independent **Auditor** must be used for those audit checklist items that refer to pricing or energy.

9.4.1.1 Verify that the TPCx-BB provided kit is used and its version.

9.4.1.2 Verify that all 3 tests (Load, Power, Throughput) (Clause 2.4) of the **Performance Run** and **Repeatability Run** completed with no error reported.

9.4.1.3 Verify Validation tests (2.4) of Performance Run and Repatability Run completed with no error reported.

9.4.1.4 Verify Benchamark has been executed according Clause 2.4 and 2.5.

9.4.1.5 Verify the Validation Phase results reported for SF1 matches with reference result set provided with the TPCx-BB kit (Clause 4.1.2.5) If the Validation Phase run results do not match with the reference result set use manually verify validation Phase results as defined in (Clause 4.1.3)

9.4.1.6 Verify that all scripts and source code to implement the benchmark has been included in the **Report**.

9.4.1.7 Verify Kit report contains all information mentioned through clause 2.2.3

9.4.1.8 Verify Clause 2.2.4 has been followed to ensure the parameter settings was performed as defined in the specification and required reports, files are provided as part of the FDR.

9.4.1.9 Verify Clause 2.2.5 is followed and according the defined Test Sponsor Kit modification.

9.4.1.10 Verify Clause 2.2.5.2 is followed and no JAR files get modified and no JAR file optimizers were used.

9.4.1.11 Verify the test execution has produced the required output by checking the logfiles to see if the query created an output.

9.4.1.12 Verify that all components of the **SUT** are commercially available as per the TPC Pricing Specification.

9.4.1.13 Verify that all components of the **SUT** are included in the pricing.

9.4.1.14 Verify no aspect of SUT, including the dataset size, tuning parameters were changed between performance and repeatability runs.

9.4.1.15 Verify that the SF used for publication is valid according to Clause 4.1.

9.4.1.16 Verify that the metrics are reported as per the requirements in Clause 5

9.4.1.17 Verify that the SUT Pricing Report is in compliance with the TPC Pricing specification.

9.4.2 Verify that the Energy report is in compliance with the TPC Energy specification (if reported).

9.4.3 Verify that Full Disclosure Report and Executive Summary report are accurate and comply with the reporting requirements. This includes:

- metric calculation
- system availability
- the diagrams of both measured and priced configuration
- system pricing
- the numerical quantity summary
- parameter files required as part of **FDR** are provided.

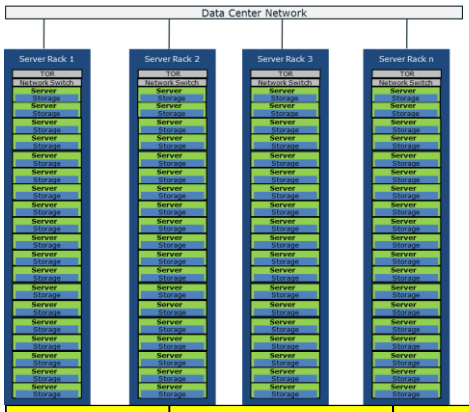
Appendix A. SAMPLE EXECUTIVE SUMMARY

The following page provides a template of the TPCx-BB **Executive Summary.**

My Company Logo	My Server Model B	TPCx-BB Rev. 1.1.0 TPC-Pricing Rev. 2.0.1 Report Date: December 15, 2014
Total System Cost	Performance Metric	Price / Performance
$99,996.13 USD	390.99 BBQpm@3000	$255.76 USD $ / BBQpm @3000

Scale Factor	Streams	Apache Hadoop Compatible software	Operating System	Other Software	Availability Date
3000	4	My HDFS Software 1.0	My OS V2.0	None	December 15, 2014

System Configuration



Server	Role(s)	Count	Host	HW/SWConfiguration	Storage Setup
Worker	Yarn NM/Hive Server/Spark Worker	50	TPCx-BB[100] - [BB150]	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version,	**OS: Model x GB SSD, Intermediate/Shuffle/ Temp Data: x Model x GB**
Distro Manger	Hadoop Manager	1	TPCx-BB-CDH	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version,	**OS: Model x GB SSD.**
Benchmark SUT Driver	YARN/SPARK,HIVE Gateway	1	TPCxBB-Driver1	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version,	
Name Node/Resource Manager	YARN/NN/ZooKeeper	1	TPCx-BB_PNN1	Vendor Server Model Name HW/SW Config (Processor Model, socket count, Frequency, Core count, DRAM capacity, Storage x HDD Model, Network and BW link speed) OS Model version,	

Physical Storage /Scale Factor	Scale Factor/Physical Memory
Servers	4 x My Server Model B
Processors/Cores/Threads/Model	4/32/64 My CPU Model Z (2.7 GHz, 20MB

	cache, 130W)
Memory	128GB
Storage	2 x 600GB 10K SFF SAS (internal) 1 x My Storage Array Model A with 8 X 1TB 7.2K SAS LFF HDD
Network:	2x My Switch Model X 10GbE

My Company Logo	My Server Model B			TPCx-BB Rev. 1.1.0 TPC-Pricing Rev. 2.0.1		
				Report Date:		5-Dec-2014
Description	Part Number	Source	Unit Price	Qty	Extended Price	3 Year Maint. Price
My Server Model B, 4 My CPU Model Z, 128GB, 2 x 600GB 10K SFF SAS	MY-S-001	1	12,100.77	4	$48,403	$100
My Storage Array Model A	MY-SE-002	1	1,988.00	4	$7,952	$200
My HDD Model xyz 1TB SATA 7.2K LFF	MY-HDD-011	1	800.47	40	$32,019	
My OS	MY-OS	1	485.24	4	$1,941	
My HDFS SoftwareSoftware	MY-Hadoop	1	2,700.00	4	$10,800	
My Switch Model X	My-Switch	1	1,922.12	2	$3,844	
				Subtotal	$104,959	$300
Large Purchase Discount	5.0%	1			-$5,248	-$15

Pricing: 1=My Company	Three-Year Cost of Ownership:	$99,996.1
Audited by My Auditor		
All discounts are based on US list prices and for similar quantities and configurations. The discounts are based on the overall specific components pricing from respective vendors in this single quotation. Discounts for similarly sized configurations will be similar to those quoted here, but may vary based on the components in the configuration.	3BQpm:	1,100.1
	$/BBQpm:	$90.9

Prices used in TPC benchmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specifications. If you find that the stated prices are not available according to these terms, please inform at pricing@tpc.org. Thank you.

Measurement Results

Scale Factor	3000
Number of Streams	4

Performance Run

Start of Run	10/02/2014 02:01:09
End of Run	10/02/2014 08:11:31
Run Elapsed Time	6:10:22
Start of Load Test	10/02/2014 02.01:09
End of Load Test	10/02/2014 02:01:16
Load Test Elapsed Time	3:10:22
Start of Power Test	10/02/2014 03:08:26
End of Power Test	10/02/2014 03.08:27
Power Test Elapsed Time	3:10:22
Start of Throughput Test	10/02/2014 03:08:26
End of Throughput Test	10/02/2014 03.08:27
Throughput Test Elapsed Time	3:10:22
Performance Metric (BBQpm@SF)	390.99 @ BBQpm SF

Repeatability Run

Start of Run	10/02/2014 02:01:09
End of Run	10/02/2014 08:11:31
Run Elapsed Time	6:10:22
Start of Load Test	10/02/2014 02.01:09
End of Load Test	10/02/2014 02:01:16
Load Test Elapsed Time	3:10:22
Start of Power Test	10/02/2014 03:08:26
End of Power Test	10/02/2014 03.08:27
Power Test Elapsed Time	3:10:22
Start of Throughput Test	10/02/2014 03:08:26
End of Throughput Test	10/02/2014 03.08:27
Throughput Test Elapsed Time	3:10:22
Performance Metric (BBQpm@SF)	393.99 @ BBQpm SF

Appendix B. LOGICAL DATABASE DESIGN

The following clause provides an overview of the data model and all table columns implemented by the TPCx-BB kit. If there is a conflict between the descriptions provided in the TPCx-BB Specification and the TPCx-BB kit implementation, the TPCx-BB kit prevails.

B.1 Table Columns Used by Queries

Minimal data description (contains only columns used by queries) (~122 columns).

B.1.1 date_dim

date_dim	Type	NULL?	Table is used by queries:
d_date_sk	BIGINT	NOT NULL	Q4 Q6 Q7 Q9 Q13 Q16 Q17 Q19 Q21 Q22 Q23
d_date	DATE		Q4 Q16 Q19 Q22
d_month_seq	INTEGER		Q7
d_week_seq	INTEGER		Q19
d_year	INTEGER		Q6 Q7 Q9 Q13 Q17 Q21 Q23
d_moy	INTEGER		Q7 Q17 Q21 Q23

B.1.2 time_dim

time_dim	Type	NULL?	Table is used by queries:
t_time_sk	BIGINT	NOT NULL	Q4 Q14
t_time	INTEGER		Q4
t_hour	INTEGER		Q14

B.1.3 customer

Customer	Type	NULL?	Table is used by queries:
c_customer_sk	BIGINT	NOT NULL	Q5 Q6 Q7 Q13 Q17
c_customer_id	CHAR (16)	NOT NULL	Q6 Q13
c_current_cdemo_sk	BIGINT		Q5
c_current_addr_sk	BIGINT		Q7 Q17
c_first_name	CHAR (20)		Q6 Q13
c_last_name	CHAR (30)		Q6 Q13
c_preferred_cust_flag	CHAR (1)		Q6
c_birth_country	VARCHAR (20)		Q6
c_login	CHAR (13)		Q6
c_email_address	CHAR (50)		Q6

B.1.4 customer_address

customer_address	Type	NULL?	Table is used by queries:
ca_address_sk	BIGINT	NOT NULL	Q7 Q9 Q17
ca_state	CHAR (2)		Q7 Q9

ca_country	VARCHAR (20)		Q9
ca_gmt_offset	DECIMAL (5 ,2)		Q17

B.1.5 customer_demographics

customer_ demographics	Type	NULL?	Table is used by queries:
cd_demo_sk	BIGINT	NOT NULL	Q5 Q8 Q9
cd_gender	CHAR (1)		Q5
cd_marital_status	CHAR (1)		Q9
cd_education_status	CHAR (20)		Q5 Q9

B.1.6 household_demographics

household_de mographics	Type	NULL?	Table is used by queries:
hd_demo_sk	BIGINT	NOT NULL	Q14
hd_dep_count	INTEGER		Q14

B.1.7 item

item	Type	NULL?	Table is used by queries:
i_item_sk	BIGINT	NOT NULL	Q5 Q7 Q12 Q15 Q16 Q17 Q19 Q21 Q22 Q23 Q24 Q26 Q29 Q30
i_item_id	CHAR (16)	NOT NULL	Q16 Q21 Q22
i_item_desc	VARCHAR (200)		Q21
i_current_price	DECIMAL (7 ,2)		Q7 Q22 Q24
i_class_id	INTEGER		Q26
i_category_id	INTEGER		Q1 Q15 Q29 Q30
i_category	CHAR (50)		Q5 Q7 Q12 Q17 Q26

B.1.8 item_marketprices

item_marketprices	Type	NULL?	Table is used by queries:
imp_item_sk	BIGINT	NOT NULL	Q24
imp_competitor_ price	DECIMAL (7 ,2)		Q24
imp_start_date	BIGINT		Q24
imp_end_date	BIGINT		Q24

B.1.9 inventory

inventory	Type	NULL?	Table is used by queries:
inv_date_sk	BIGINT	NOT NULL	Q22 Q23
inv_item_sk	BIGINT	NOT NULL	Q22 Q23
inv_warehouse_sk	BIGINT	NOT NULL	Q22 Q23
inv_quantity_on_hand	INTEGER		Q22 Q23

B.1.10 promotion

promotion	Type	NULL?	Table is used by queries:
p_channel_dmail	CHAR (1)		Q17
p_channel_email	CHAR (1)		Q17
p_channel_tv,	CHAR (1)		Q17

B.1.11 product_reviews

product_reviews	Type	NULL?	Table is used by queries:
pr_review_sk	BIGINT	NOT NULL	Q27 Q28
pr_review_date	DATE		Q18
pr_review_rating	INT	NOT NULL	Q11 Q28
pr_item_sk	BIGINT	NOT NULL	Q10 Q11 Q19 Q27 Q28
pr_review_content	TEXT	NOT NULL	Q10 Q18 Q19 Q27 Q28

B.1.12 store

store	Type	NULL?	Table is used by queries:
s_store_sk	BIGINT	NOT NULL	Q9 Q17 Q18 Q21
s_store_id	CHAR (16)	NOT NULL	Q21
s_store_name	VARCHAR (50)		Q18 Q21
s_gmt_offset	DECIMAL (5 ,2)		Q17

B.1.13 store_sales

store_sales	Type	NULL?	Table is used by queries:
ss_sold_date_sk	BIGINT default 9999999 ,		Q6 Q7 Q9 Q12 Q13 Q15 Q17 Q18 Q20 Q21 Q24 Q25
ss_sold_time_sk	BIGINT		Q12
ss_item_sk,	BIGINT	NOT NULL	Q1 Q7 Q12 Q15 Q17 Q20 Q21 Q24 Q26
ss_customer_sk	BIGINT		Q6 Q7 Q12 Q13 Q17 Q20 Q21 Q25 Q26
ss_cdemo_sk	BIGINT		Q9
ss_addr_sk	BIGINT		Q9
ss_store_sk	BIGINT		Q1 Q9 Q15 Q17 Q18 Q21
ss_promo_sk	BIGINT		Q17
ss_ticket_number	BIGINT	NOT NULL	Q1 Q20 Q21 Q25
ss_quantity	INTEGER		Q21 Q24
ss_sales_price	DECIMAL (7 ,2)		Q9
ss_ext_discount_amt	DECIMAL (7 ,2)		Q6
ss_ext_sales_price	DECIMAL (7 ,2)		Q6 Q17
ss_ext_wholesale_cost	DECIMAL (7 ,2)		Q6
ss_ext_list_price	DECIMAL (7 ,2)		Q6
ss_net_paid	DECIMAL (7 ,2)		Q13 Q15 Q17 Q20 Q25
ss_net_profit	DECIMAL (7 ,2)		Q9

B.1.14 store_returns

store_returns	Type	NULL?	Table is used by queries:
sr_returned_date_sk	BIGINT default 9999999		Q19 Q20 Q21
sr_item_sk,	BIGINT	NOT NULL	Q19 Q20 Q21
sr_customer_sk,	BIGINT		Q20 Q21
sr_ticket_number	BIGINT	NOT NULL	Q20 Q21
sr_return_quantity,	INTEGER		Q19 Q21
sr_return_amt	DECIMAL (7 ,2)		Q20

B.1.15 web_sales

web_sales	Type	NULL?	Table is used by queries:
ws_sk	BIGINT	NOT NULL	Q8
ws_sold_date_sk	BIGINT default 9999999 ,		Q6 Q8 Q11 Q13 Q16 Q21 Q24 Q25
ws_sold_time_sk	BIGINT		Q14
ws_item_sk	BIGINT	NOT NULL	Q11 Q16 Q21 Q24 Q29
ws_bill_customer_sk	BIGINT		Q6 Q13 Q21 Q25 Q29
ws_ship_hdemo_sk	BIGINT		Q14
ws_web_page_sk	BIGINT		Q14
ws_warehouse_sk	BIGINT		Q16 Q22
ws_order_number	BIGINT	NOT NULL	Q16 Q25
ws_quantit	INTEGER		Q21 Q24
ws_sales_price	DECIMAL (7 ,2)		Q16
ws_ext_discount_amt	DECIMAL (7 ,2)		Q6
ws_ext_sales_price	DECIMAL (7 ,2)		Q6
ws_ext_wholesale_cost	DECIMAL (7 ,2)		Q6
ws_ext_list_price	DECIMAL (7 ,2)		Q6
ws_net_paid	DECIMAL (7 ,2)		Q8 Q11 Q13 Q25

B.1.16 web_returns

web_returns	Type	NULL?	Table is used by queries:
wr_returned_date_sk	BIGINT default 9999999		Q19
wr_item_sk	BIGINT	NOT NULL	Q16 Q19
wr_order_number	BIGINT		Q16
wr_return_quantity	INTEGER		Q19
wr_refunded_cash	DECIMAL (7 ,2)		Q16

B.1.17 web_clickstreams

web_clickstreams	Type	NULL?	Table is used by queries:
wcs_click_sk	BIGINT	NOT NULL	
wcs_click_date_sk	BIGINT		Q3 Q4 Q8 Q12
wcs_click_time_sk	BIGINT		Q3 Q4 Q8 Q12

Refresh system			
${REFRESH_PHASE}		0	0==rows[0,REFRESH_PERCENTAGE*Table.Size]; 1==rows[REFRESH_PERCENTAGE*Table.Size+1, Table.Size] .Has effect only if

wcs_sales_sk	BIGINT		Q3 Q8
wcs_item_sk	BIGINT	can be null	Q2 Q3 Q4 Q5 Q8 Q12 Q30
wcs_web_page_sk	BIGINT		Q8
wcs_user_sk	BIGINT	can be null	Q2 Q3 Q4 Q5 Q8 Q12 Q30

B.1.18 warehouse

warehouse	Type	NULL?	Table is used by queries:
w_warehouse_sk	BIGINT	NOT NULL	Q16 Q23
w_warehouse_name	VARCHAR (20)		Q22 Q23
w_state	CHAR (2)		Q16

B.1.19 web_page

web_page	Type	NULL?	Table is used by queries:
wp_web_page_sk	BIGINT	NOT NULL	Q4 Q8 Q14
wp_type	CHAR (50)		Q4 Q8
wp_char_count	INTEGER		Q14

B.1.20 web_site

(UNUSED/UNREFERENCED) only ref: web_sales

web_site	Type	NULL?	Table is used by queries:

B.1.21 reason

only referenced by sr_reason_sk and wr_reason_sk (both not used in queries)

reason	Type	NULL?	Table is used by queries:

B.1.22 ship_mode

(UNUSED/UNREFERENCED)

ship_mode	Type	NULL?	Table is used by queries:

B.1.23 income_band

(NOT USED)

income_band	Type	NULL?	Table is used by queries:

B.2 Variables

		${REFRESH_SYSTEM_ENABLED}==1.
${REFRESH_SYSTEM_ENABLED}	1	
${REFRESH_PERCENTAGE}	1.0d - 0.01d	default 1%==0.01

Global parameters (affect multiple tables)

NULL_CHANCE	0.00025	If a column is not "NOT NULL" some of the values may be null with the specified percentage
serial keys like customer_sk date_sk etc.. start at 0 or at 1 or...		
${SK_ID_OFFSET}	0	Serial key id offset. Determines where serial keys of tables start
datetime format: yyyy-MM-dd HH:mm:ss		
${date_begin_date}	01.01.1900 00:00	
${date_end_date}	01.01.2200 00:00	
${CURRENT_DAY}	03.08.2005 00:00	TODAY
${one_day_in_milliseconds}	24.0 * 60.0 * 60.0 * 1000.0	one day
${avg_competitors_per_item}	5	
${anonymous_reviews_per_item}	5	
${reviews_per_user}	0.2	
${reviews_per_sale}	0.01	
${pages_per_item}	4	
${pages_to_buy}	4	
${items_per_cart}	12	
${buy_ratio}	4	

address		Used in many tables like store_sales, web_sales, warehouse, etc..
${address_street_number_min}	1.0	
${address_street_number_max}	1000.0	
${address_suite_number_min}	1.0	
${address_suite_number_max}	10.0	
${address_zip_min}	10000.0	
${address_zip_max}	99999.0	

Table scaling types

${SF}	1	root" scalefactor. SF 1 ~1GB SF 10 ~10GB
${SF_log_1.5}	Math.log(${SF}) / Math.log(1.5d) + 1.0d	
${SF_log_5}	(Math.log(${SF}) / Math.log(5.0d) + 1.0d)	
${SF_sqrt}	Math.sqrt(${SF})	
${SF_linear}	${SF} * (2.0d - (${SF_log_5} * ${SF_sqrt} / ${SF}))	

Table specific properties

customer

${preferred_cust_likelihood}	0.5

customer_demographics

${gender_likelihood}	0.5
${married_likelihood}	0.3
${divorced_likelihood}	0.2
${single_likelihood}	0.2
${widowed_likelihood}	0.2

income_band

${income_band_stepsize}	10000

inventory

${inventory_begin_date}	01.01.2001 00:00
${inventory_end_date}	02.01.2006 00:00
${inventory_weeks}	(${inventory_end_date} - ${inventory_begin_date}) / ${one_day_in_milliseconds} / 7
${inventory_days_since_date_begin_date}	(${inventory_begin_date} - ${date_begin_date}) / ${one_day_in_milliseconds}

item

${item_begin_date}	03.01.2000 00:00
${item_end_date}	05.01.2004 00:00

promotion

${dmail_likelihood}	0.5
${email_likelihood}	0.1
${catalog_likelihood}	0.1
${tv_likelihood}	0.1
${radio_likelihood}	0.1
${press_likelihood}	0.1
${event_likelihood}	0.1
${demo_likelihood}	0.1
${discount_active_likelihood}	0.1

store

${store_begin_date}	03.01.2000 00:00
${store_end_date}	05.01.2004 00:00
${STORE_MIN_TAX_PERCENTAGE}	0.00
${STORE_MAX_TAX_PERCENTAGE}	0.11

store_returns

${return_store_sale_likelihood}	0.1
${SR_SAME_CUSTOMER}	0.8

store_sales

${SS_QUANTITY_MAX}	100
${SS_WHOLESALE_MAX}	100.00
${SS_MARKUP_MAX}	1.00
${SS_DISCOUNT_MAX}	1.00
${store_sales_begin_date}	01.01.2001 00:00
${store_sales_end_date}	02.01.2006 00:00
${store_sales_days_since_date_begin_date}	(${store_sales_begin_date} - ${date_begin_date}) / ${one_day_in_milliseconds}
${store_sales_days_within}	(${store_sales_end_date} - ${store_sales_begin_date}) / ${one_day_in_milliseconds}
${SS_ITEMS_PER_ORDER_MIN}	1
${SS_ITEMS_PER_ORDER_MAX}	14

web_clickstreams

${visitor_likelihood}	0.8
${visitor_known_likelihood}	0.5
${mean_clicks_per_visitor}	16
${mean_clicks_per_buyer}	4
${clickstreams_chunksize}	5

web_page

${web_page_begin_date}	03.01.2000 00:00
${web_page_end_date}	05.01.2004 00:00
${WP_AUTOGEN_PCT}	0.30
${WP_LINK_MIN}	2
${WP_LINK_MAX}	25
${WP_IMAGE_MIN}	1
${WP_IMAGE_MAX}	7
${WP_AD_MIN}	0
${WP_AD_MAX}	4

warehouse

${W_SQFT_MIN}	50000
${W_SQFT_MAX}	1000000

web_returns

${return_web_sale_likelihood}	0.1

web_sales

${WS_QUANTITY_MAX}	100
${WS_WHOLESALE_MAX}	100.00
${WS_MARKUP_MAX}	2.00
${WS_DISCOUNT_MAX}	1.00
${WS_MIN_SHIP_DELAY}	1

${WS_MAX_SHIP_DELAY}	120
${WS_ITEMS_PER_ORDER_MIN}	1
${WS_ITEMS_PER_ORDER_MAX}	14
${WS_GIFT_PCT}	0.07
${web_sales_begin_date}	01.01.2001 00:00
${web_sales_end_date}	02.01.2006 00:00
${web_sales_days_since_date_begin_date}	(${web_sales_begin_date} - ${date_begin_date}) / ${one_day_in_milliseconds}
${web_sales_days_within}	(${web_sales_end_date} - ${web_sales_begin_date}) / ${one_day_in_milliseconds}

table sizes formulas (Scaling of tables with increasing scale factor SF)
Not used tables:

${income_band_size}	20
${reason_size}	35 * ${SF_log_1.5}
${ship_mode_size}	20
${web_site_size}	30

static (fixed size) tables

${date_dim_size}	(${date_end_date} - ${date_begin_date}) / ${one_day_in_milliseconds}
${time_dim_size}	${one_day_in_milliseconds} / 1000
${customer_demographics_size}	1920800
${household_demographics_size}	7200

normal not refreshed tables

${store_size}	12 * ${SF_sqrt}
${promotion_size}	300 * ${SF_log_1.5}
${warehouse_size}	5.0d * ${SF_log_5}
${web_page_size}	60 * ${SF_log_1.5}

refreshed tables

Refresh tables generate extra data during refresh (e.g. add new data to the existing tables) In normal-Phase generate table rows: [0 In refresh-Phase generate table rows: [REFRESH_PERCENTAGE*Table.Size+1	REFRESH_PERCENTAGE*Table.Size]; Table.Size]
${customer_size}	100000.0d * ${SF_sqrt}
${customer_address_size}	${customer_size} / 2
${item_size}	18000.0d * ${SF_sqrt}
${item_marketprices_size}	${item_size} * ${avg_competitors_per_item}
${store_sales_size}	90000.0d * ${SF_linear}
${store_returns_size}	${return_store_sale_likelihood} * ${store_sales_size}
${inventory_size}	${inventory_weeks} * ${item_size} * ${warehouse_size}
${web_sales_size}	90000.0d * ${SF_linear}
${web_returns_size}	${return_web_sale_likelihood} * ${web_sales_size}

${product_reviews_size}	(${item_size} * ${anonymous_reviews_per_item}) + (${customer_size} * ${reviews_per_user}) + (${web_sales_size} * ${reviews_per_sale}) * ${SF}	${product_reviews_size}"
${web_clickstreams_size}	(${web_sales_size} * (${pages_per_item} + (${pages_to_buy} / ${items_per_cart}))) + (${web_sales_size} * ${buy_ratio} * ${pages_per_item}) * ${SF}	${web_clickstreams_size}"

B.3 Table Data Generation Rules

B.3.1 date_dim

${date_dim_size} = 73049 (fixed, does not scale) one row per day in range:

(${date_end_date} - ${date_begin_date}) / ${one_day_in_milliseconds})

date_dim	Type	NULL?	Table is used by queries:	Description From: ${date_begin_date to: ${date_end_date}
d_date_sk	BIGINT	NOT NULL	Q4 Q6 Q7 Q9 Q13 Q16 Q17 Q19 Q21 Q22 Q23	Key starting at 1
d_date_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
d_date	DATE		Q4 Q16 Q19 Q22	From: ${date_begin_date to: ${date_end_date} Format: yyyy-MM-dd
d_month_seq	INTEGER		Q7	Starts at 0 Counts every month from start- to end-date
d_week_seq	INTEGER		Q19	Dense unique sequence. Starts at ${SK_ID_OFFSET} Counts every week from start- to end-date
d_quarter_seq	INTEGER			Starts at ${SK_ID_OFFSET} Counts every quater from start- to end-date
d_year	INTEGER		Q6 Q7 Q9 Q13 Q17 Q21 Q23	Year Part of d_date: yyyy
d_dow	INTEGER			Day of week: 1-7, 1==Monday
d_moy	INTEGER		Q7 Q17 Q21 Q23	Month of year: 1-12, 1==Januar
d_dom	INTEGER			Day of Month 1-31
d_qoy	INTEGER			Quarter of Year 1-4
d_fy_year	INTEGER			Financial: d_year + ½ year
d_fy_quarter_seq	INTEGER			Financial: d_quarter + ½ year
d_fy_week_seq	INTEGER			Financial: d_week + ½ year
d_day_name	CHAR (9)			Day of week d_dow as string {Monday,…,Sunday}
d_quarter_name	CHAR (6)			Quarter of year d_qoy as string yyyyQ{1..4}: example: 1990Q2
d_holiday	CHAR (1)			N/Y (true/false)
d_weekend	CHAR (1)			N/Y (true/false)
d_following_holiday	CHAR (1)			N/Y (true/false)
d_first_dom	INTEGER			First day of month in Julian calendar (Julian day number e.g: 2415021)
d_last_dom	INTEGER			Last day of month in Julian calendar (Julian day number e.g: 2415021)

d_same_day_ly	INTEGER	Same day in Julian calendar (Julian day number e.g: 2415021)
d_same_day_lq	INTEGER	Same day in Julian calendar (Julian day number e.g: 2415021)
d_current_day	CHAR (1)	N/Y (true/false) d_date_sk==CURRENT_DAY ? Y:N;
d_current_week	CHAR (1)	N/Y (true/false) d_week_seq==CURRENT_WEEK ? Y:N;
d_current_month	CHAR (1)	N/Y (true/false) d_moy==CURRENT_MONTH ? Y:N;
d_current_quarter	CHAR (1)	N/Y (true/false) d_qoy==CURRENT_QUATER ? Y:N;
d_current_year	CHAR (1)	N/Y (true/false) d_year==CURRENT_YEAR ? Y:N;

Notes:

DISTRIBUTE BY REPLICATION ;

B.3.2 time_dim

${time_dim_size} = 86400 (fixed, does not scale) one row for every second in one day

time_dim	Type	NULL?	Table is used by queries:	Description Example: 0|AAAAAAAABAAAAAAA|0|0|0|0|AM|third|night||
t_time_sk	BIGINT	NOT NULL	Q4 Q14	Dense unique sequence. Starts at ${SK_ID_OFFSET}
t_time_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
t_time	INTEGER		Q4	Starts at 0 time_id == t_time_sk
t_hour	INTEGER		Q14	time_id/60/60 modulo 24
t_minute	INTEGER			time_id/60 modulo 60
t_second	INTEGER			time_id modulo 60
t_am_pm	CHAR (2)			See Weighted list "purchase_band" value col:1 weightColumn: 0
t_shift	CHAR (20)			See Weighted list "purchase_band" value col:2 weightColumn: 0
t_sub_shift	CHAR (20)			See Weighted list "purchase_band" value col:3 weightColumn: 0
t_meal_time	CHAR (20)			See Weighted list "purchase_band" value col:4 weightColumn: 0

Notes:

DISTRIBUTE BY REPLICATION ;

B.3.3 customer

${customer_size} = 100000 * ${SF_sqrt

customer	Type	NULL?	Table is used by queries:	Description
c_customer_sk	BIGINT	NOT NULL	Q5 Q6 Q7 Q13 Q17	Dense unique sequence. Starts at ${SK_ID_OFFSET}
c_customer_id	CHAR (16)	NOT NULL	Q6 Q13	Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
c_current_cdemo_sk	BIGINT		Q5	Random reference to table: customer_demographics cd_demo_sk
c_current_hdemo_sk	BIGINT			Random reference to table: household_demographics hd_demo_sk
c_current_addr_sk	BIGINT		Q7 Q17	Random reference to table: customer_address ca_address_sk
c_first_shipto_date_sk	BIGINT			Random reference to table: date_dim d_date_sk

c_first_sales_date_sk	BIGINT		Random reference to table: date_dim d_date_sk
c_salutation	CHAR (10)		See Weighted list "salutations" value col:0 weightColumn: 0 or 1 Salution must match gender as implicitly choosen by: c_first_name
c_first_name	CHAR (20)	Q6 Q13	See Weighted list "first_names" value col:0 weightColumn: 0
c_last_name	CHAR (30)	Q6 Q13	See Weighted list "last_names" value col:0 weightColumn: 0
c_preferred_cust_flag	CHAR (1)	Q6	Propability : value ${preferred_cust_likelihood}: Y 1-${preferred_cust_likelihood}: N
c_birth_day	INTEGER		Random number [1, 31]
c_birth_month	INTEGER		Random number [1, 12]
c_birth_year	INTEGER		Random number: [1924, 1992]
c_birth_country	VARCHAR (20)	Q6	See Weighted list "countries" value col:0 weightColumn: 0
c_login	CHAR (13)	Q6	Random string len: [1-13]
c_email_address	CHAR (50)	Q6	Pattern: C_first_name.c_last_name@randomProvider.tld
c_last_review_date	CHAR (10)		Min: ${CURRENT_DAY} – 1 Year Max: ${CURRENT_DAY}

Note:

DISTRIBUTE BY HASH (c_customer_sk);

B.3.4 customer_address

${customer_address_size} = ${customer_size} / 2

customer_address	Type	NULL?	Table is used by queries:	Description
ca_address_sk	BIGINT	NOT NULL	Q7 Q9 Q17	Dense unique sequence. Starts at ${SK_ID_OFFSET}
ca_address_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: “ABCDEFGHIJKLMNOPQRSTUVWXYZ” Example: AAAAAAAAOKJNECAA
ca_street_number	CHAR (10)			Random number: [1, 1000]
ca_street_name	VARCHAR (60)			Probability: 50% 1 word “%s” 50% 2 Words “%s %s” From Weighted list "street_names", valueCol:0 weightCoL 0
ca_street_type	CHAR (15)			See Weighted list "street_type" value col:0 weightColumn: 0
ca_suite_number	CHAR (10)			Random String len: [1, 10]
ca_city	VARCHAR (60)			See Weighted list "cities" value col:0 weightColumn: 0
ca_county	VARCHAR (30)			See Weighted list "fips_county" value col:county: weightColumn: uniform
ca_state	CHAR (2)		Q7 Q9	See Weighted list "fips_county" value col: st weightColumn: uniform Same entry as ca_county (state must match county)
ca_zip	CHAR (10)			Random number [10000, 99999]
ca_country	VARCHAR (20)		Q9	'United States'
ca_gmt_offset	DECIMAL (5 ,2)		Q17	See Weighted list "fips_county" value col:gmt weightColumn: uniform Same entry as ca_county (state must match county)
ca_location_type	CHAR (20)			See Weighted list "location_type" value col:0 weightColumn: 0

Note:

DISTRIBUTE BY HASH (ca_address_sk);

B.3.5 customer_demographics

${customer_demographics_size} = 1920800 (fixed, does not scale)

customer_demographics	Type	NULL?	Table is used by queries:	Description
cd_demo_sk	BIGINT	NOT NULL	Q5 Q8 Q9	Dense unique sequence. Starts at ${SK_ID_OFFSET}
cd_gender	CHAR (1)		Q5	See Weighted list "gender" value col:0 weightColumn: 0
cd_marital_status	CHAR (1)		Q9	See Weighted list "marital_status" value col:0 weightColumn: 0
cd_education_status	CHAR (20)		Q5 Q9	See Weighted list " education" value col:0 weightColumn: 0
cd_purchase_estimate	INTEGER			See Weighted list "purchase_band" value col:0 weightColumn: 0
cd_credit_rating	CHAR (10)			See Weighted list "credit_rating" value col:0 weightColumn: 0
cd_dep_count	INTEGER			See Weighted list "dependent_count" value col:0 weightColumn: 0
cd_dep_employed_count	INTEGER			See Weighted list "dependent_count" value col:0 weightColumn: 0
cd_dep_college_count	INTEGER			See Weighted list "dependent_count" value col:0 weightColumn: 0

Note:

DISTRIBUTE BY REPLICATION ;

B.3.6 household_demographics

${household_demographics_size} = 7200 (fixed, does not scale)

household_demographics	Type	NULL?	Table is used by queries:	Description
hd_demo_sk	BIGINT	NOT NULL	Q14	Dense unique sequence. Starts at ${SK_ID_OFFSET} referenced by: • ws_ship_hdemo_sk • c_current_hdemo_sk • ss_hdemo_sk • sr_hdemo_sk • ws_bill_hdemo_sk • ws_ship_hdemo_sk • wr_refunded_hdemo_sk • wr_returning_hdemo_sk
hd_income_band_sk	BIGINT			Random reference to table: "income_band" ib_income_band_sk
hd_buy_potential	CHAR (15)			See Weighted list "buy_potentialt" value col:0 weightColumn: 0
hd_dep_count	INTEGER		Q14	See Weighted list "dependent_count" value col:0 weightColumn: 0
hd_vehicle_count	INTEGER			See Weighted list "vehicle_count" value col:0 weightColumn: 0

Notes:

DISTRIBUTE BY REPLICATION ;

B.3.7 item

${item_size} = 18000.0 * ${SF_sqrt}

item	Type	NULL?	Table is used by queries:	Description
i_item_sk	BIGINT	NOT NULL	Q5 Q7 Q12 Q15 Q16 Q17 Q19 Q21 Q22 Q23 Q24 Q26 Q29 Q30	Dense unique sequence. Starts at ${SK_ID_OFFSET}
i_item_id	CHAR (16)	NOT NULL	Q16 Q21 Q22	Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
i_rec_start_date	DATE			Date from: ${item_begin_date}

			to: ${inventory_begin_date} Format: yyyy-MM-dd With: i_rec_start_date{n} < i_rec_start_date{n+1} where n= **i_item_sk**
i_rec_end_date,	DATE		No end date for the moment. Value: “” Elese: 50% Empty 50%: i_rec_start_date + rand[2years, 4years]
i_item_desc	VARCHAR (200)	Q21	Sentences following pseudo englisch gramatic Example: Clear circumstances know then further white companies. Typical budgets take both required children. Appeals must not make civil, financial representatives. Emotional areas shall wear only.
i_current_price	DECIMAL (7 ,2)	Q7 Q22 Q24	Random decimal [0.09, 99.99]
i_wholesale_cost	DECIMAL (7 ,2)		Random decimal [0.02, 87,36]
i_brand_id	INTEGER		Radnom integer [1001001, 10016017]
i_brand	CHAR (50)		Random string len [1, 50]
i_class_id	INTEGER	Q26	Unique ID identifying i_class. starts at a
i_class	CHAR (50)		Class must depend on selected i_category ! See the following WeightedLists mathing the selected i_category: Women -> women_class Men -> men_class Children -> children_class Shoes -> shoe_class Music -> music_class Jewelry -> jewelry_class Home -> home_class Sports -> sport_class Books -> book_class Electronics -> electronic_class
i_category_id	INTEGER	Q1 Q15 Q29 Q30	Unique id identifying i_category. Starts at 1
i_category	CHAR (50)	Q5 Q7 Q12 Q17 Q26	See Weighted list "categories" value col:0 weightColumn: 0
i_manufact_id	CHAR (50)		Random integer [1, 1000]
i_manufact	CHAR (50)		Random String len [1, 50]
i_size	CHAR (20)		See Weighted list "sizes" value col:0 weightColumn: 0
i_formulation,	CHAR (20)		Random String len [1, 20]
i_color	CHAR (20)		See Weighted list "color" value col:0 weightColumn: 0
i_units	CHAR (10)		See Weighted list "units" value col:0 weightColumn: 0
i_container	CHAR (10)		See Weighted list "container" value col:0 weightColumn: 0
i_manager_id	INTEGER		Random integer [1, 1000] distibuted like: Weighted list "i_manager_id"
i_product_name	CHAR (50)		Random String len [1, 50]

Notes:

DISTRIBUTE BY HASH (i_item_sk);

B.3.8 item_marketprices

${item_marketprices_size} = ${item_size} * ${avg_competitors_per_item}

item_marketprices	Type	NULL?	Table is used by queries:	Description
imp_sk	BIGINT	NOT NULL		Dense unique sequence. Starts at ${SK_ID_OFFSET}
imp_item_sk	BIGINT	NOT NULL	Q24	Random reference to table:: item i_item_sk
imp_competitor	VARCHAR (20)			Random String len [1, 20]
imp_competitor_ price	DECIMAL (7 ,2)		Q24	Random decimal [0.09, 99.99]
imp_start_date	BIGINT		Q24	Random reference to table: date d_date_sk

imp_end_date	BIGINT	Q24	Random reference to table: date d_date_sk > imp_start_date

Notes:

DISTRIBUTE BY HASH (imp_sk);

B.3.9 inventory

${inventory_size}= ${inventory_weeks} * ${item_size} * ${warehouse_size}

inventory	Type	NULL?	Table is used by queries:	Description
inv_date_sk	BIGINT	NOT NULL	Q22 Q23	(id or row) / ${item_size} / ${warehouse_size}) * 7 + ${inventory_days_since_date_begin_date}
inv_item_sk	BIGINT	NOT NULL	Q22 Q23	(id or row) modulo ${item_size}
inv_warehouse_sk	BIGINT	NOT NULL	Q22 Q23	(id or row) / ${item_size} modlulo ${warehouse_size}
inv_quantity_on_hand	INTEGER		Q22 Q23	Random integer between [0, 1000]

Notes:

DISTRIBUTE BY HASH (inv_item_sk);

B.3.10 promotion

${promotion_size} = 300 * ${SF_log_1.5}

promotion	Type	NULL?	Table is used by queries:	Description
p_promo_sk	BIGINT	NOT NULL	Q17	Dense unique sequence. Starts at ${SK_ID_OFFSET}
p_promo_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
p_start_date_sk	BIGINT			Random reference to table: date d_date_sk
p_end_date_sk	BIGINT			Random reference to table: date d_date_sk > p_start_date_sk
p_item_sk	BIGINT			Random reference to table: item i_item_sk
p_cost	DECIMAL (15 ,2)			Random decimal [10.00, 1000.00]
p_response_target	INTEGER			1
p_promo_name	CHAR (50)			See Weighted list "syllables" value col:0 weightColumn: 0
p_channel_dmail	CHAR (1)		Q17	Propability : value ${dmail_likelihood}: Y 1-${dmail_likelihood}: N
p_channel_email	CHAR (1)		Q17	Propability : value ${email_likelihood}: Y 1-${email_likelihood}: N
p_channel_catalog	CHAR (1)			Propability : value ${catalog_likelihood}: Y 1-${catalog_likelihood}: N
p_channel_tv,	CHAR (1)		Q17	Propability : value ${tv_likelihood}: Y 1-${tv_likelihood}: N
p_channel_radio	CHAR (1)			Propability : value ${radio_likelihood}: Y 1-${radio_likelihood}: N
p_channel_press	CHAR (1)			Propability : value ${press_likelihood}: Y 1-${press_likelihood}: N
p_channel_event	CHAR (1)			Propability : value ${event_likelihood}: Y 1-${event_likelihood}: N
p_channel_demo	CHAR (1)			Propability : value ${channel_likelihood}: Y 1-${channel_likelihood}: N
p_channel_details	VARCHAR (100)			Sentences following pseudo englisch gramatic Example: Clear circumstances know then further white companies. Typical budgets take both required children. Appeals must not make civil, financial representatives. Emotional areas shall wear only.
p_purpose,	CHAR (15)			create promo_purpose; set types = (varchar);

				set weights = 1; add ("Unknown": 4);
p_discount_active	CHAR (1)			Propability : value ${discount_active_likelihood}: Y 1-${discount_active_likelihood}: N

Note:

) DISTRIBUTE BY REPLICATION ;

B.3.11 product_reviews

${product_reviews_size} = (${item_size} * ${anonymous_reviews_per_item}) + (${web_sales_size} * ${reviews_per_sale})

pr_review_content must contain sentences which match the refrenced item type and rating.

The benchmark contains many semantic analysis queries, trying to classify the reviews based on the user written text. Therfore, pr_review_content must resemble a human written review text as close as possible!

If the referenced item is a DVD-Player with rating 5, a human reader should be able to recognize that the computer generated review is indeed talking about such a DVD-Player product and that the writer was satisfied. A rating of 1 should reflect a negative review.

product_reviews	Type	NULL?	Table is used by queries:	Description
pr_review_sk	BIGINT	NOT NULL	Q27 Q28	Dense unique sequence. Starts at ${SK_ID_OFFSET}
pr_review_date	DATE		Q18	Date from: ${date_begin_date} to: ${date_begin_date} Format: yyyy-MM-dd With: i_rec_start_date{n} < i_rec_start_date{n+1} where n= **i_item_sk**
pr_review_time	CHAR(6)			Random reference to table: time_dim t_time_sk
pr_review_rating	INT	NOT NULL	Q11 Q28	1-5, See Weighted list "ratingWeights" value col:0 weightColumn: 0
pr_item_sk	BIGINT	NOT NULL	Q10 Q11 Q19 Q27 Q28	Random reference to a ws_item_sk of referenced order_sk in pr_order_sk
pr_user_sk	BIGINT			Random reference to ws_user_sk of referenced order_sk in pr_order_sk
pr_order_sk	BIGINT			Random reference to web_sales order_id
pr_review_content	TEXT	NOT NULL	Q10 Q18 Q19 Q27 Q28	pr_review_content must contain sentences which match the refrenced item's type (i_category) and pr_review_rating.

Notes:

DISTRIBUTE BY HASH (pr_review_sk);

B.3.12 store

${store_size} =12 * ${SF_sqrt}

store	Type	NULL?	Table is used by queries:	Description Example: 1|AAAAAAAAAAAAAAAA|1997-03-13||2451189|ought|245|5250760|8AM-4PM|William Ward|2|Unknown|Enough high areas stop expectations. Elaborate, local is|Charles Bartley|1|Unknown|1|Unknown|767|Spring |Wy|Suite 250|Midway|Williamson County|TN|31904|United States|-5|0.03|
s_store_sk	BIGINT	NOT NULL	Q9 Q17 Q18 Q21	Dense unique sequence. Starts at ${SK_ID_OFFSET}
s_store_id	CHAR (16)	NOT NULL	Q21	Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
s_rec_start_date	DATE			Date from: ${store_begin_date} to: ${store_begin_date}

			Format: yyyy-MM-dd With: s_rec_start_date{n} < s_rec_start_date{n+1} where n= **s_store_sk**
s_rec_end_date	DATE		No end date for the moment. Value: “” Elese: 50% Empty 50%: wp_rec_start_date + rand[2years, 4years]
s_closed_date_sk	BIGINT		With STORE_CLOSED_PCT probabiltity a store is closed. If closed: ref to table date d_date_sk
s_store_name	VARCHAR (50)	Q18 Q21	One random word from Weighted List ‘syllables’
s_number_employees	INTEGER		Random integer between: [200, 300]
s_floor_space	INTEGER		Random integer between: [5000000, 10000000]
s_hours	CHAR (20)		Weighted List ‘call_center_hours’, value_col= 0; weight_col: 0
s_manager	VARCHAR (40)		Pattern: "%s %s" Weighted List ‘first_names’, value_col= 0; weight_col: 0 Weighted List ‘last_names’; value_col= 0; weight_col: 0
s_market_id	INTEGER		Random integer between: [2, 10]
s_geography_class	VARCHAR (100)		Value: "Unknown"
s_market_desc	VARCHAR (100)		Sentences following pseudo englisch gramatic Example: Clear circumstances know then further white companies. Typical budgets take both required children. Appeals must not make civil, financial representatives. Emotional areas shall wear only.
s_market_manager	VARCHAR (40)		Pattern: "%s %s" Weighted List ‘first_names’, value_col= 0; weight_col: 0 Weighted List ‘last_names’; , value_col= 0; weight_col: 0
s_division_id	INTEGER		Value: 1
s_division_name	VARCHAR (50)		Value: "Unknown"
s_company_id	INTEGER		Value: 1
s_company_name	VARCHAR (50)		Value: "Unknown"
s_street_number	VARCHAR (10)		Address like in warehouse
s_street_name	VARCHAR (60)		Address like in warehouse
s_street_type	CHAR (15)		Address like in warehouse
s_suite_number	CHAR (10)		Address like in warehouse
s_city	VARCHAR (60)		Address like in warehouse
s_county	VARCHAR (30)		Address like in warehouse
s_state	CHAR (2)		Address like in warehouse
s_zip	CHAR (10)		Address like in warehouse
s_country	VARCHAR (20)		Address like in warehouse
s_gmt_offset	DECIMAL (5 ,2)	Q17	Address like in warehouse
s_tax_precentage	DECIMAL (5 ,2)		UNIFORM RAND DECIMAL between [${STORE_MIN_TAX_PERCENTAGE}, ${STORE_MAX_TAX_PERCENTAGE}]

Note:

DISTRIBUTE BY REPLICATION ;

B.3.13 store_sales

${store_sales_size} = 90000.0d * ${SF_linear}

One logical sale consists of random[${SS_ITEMS_PER_ORDER_MIN}, ${SS_ITEMS_PER_ORDER_MAX] itmes.

Logical sale	N = random[${SS_ITEMS_PER_ORDER_MIN}, ${SS_ITEMS_PER_ORDER_MAX] 1=same for every N Write N lines for a logical Sale into store_sales table
ss_sold_date_sk	1
ss_sold_time_sk	1
ss_item_sk	N
ss_customer_sk	1
ss_cdemo_sk	1
ss_hdemo_sk	1
ss_addr_sk	1
ss_store_sk	1
ss_promo_sk	1
ss_ticket_number	1
ss_quantity	N
ss_wholesale_cost	N
ss_list_price	N
ss_sales_price	N
ss_ext_discount_amt	N
ss_ext_sales_price	N
ss_ext_wholesale_cost	N
ss_ext_list_price	N
ss_ext_tax	N
ss_coupon_amt	N
ss_net_paid	N
ss_net_paid_inc_tax	N
ss_net_profit	N

store_sales	Type	NULL?	Table is used by queries:	Description
ss_sold_date_sk	BIGINT default 9999999 ,		Q6 Q7 Q9 Q12 Q13 Q15 Q17 Q18 Q20 Q21 Q24 Q25	${store_sales_days_since_date_begin_date} + Math.floor((current row or id) * (${store_sales_days_within} - 1) / ${store_sales_size})) Implicit references to date d_date_sk
ss_sold_time_sk	BIGINT		Q12	Random reference to table: time_dim t_time_sk
ss_item_sk,	BIGINT	NOT NULL	Q1 Q7 Q12 Q15 Q17 Q20 Q21 Q24 Q26	PrimaryKey; Random reference to item i_item_sk A logical sale (same ss_ticket_number) consist of random[${SS_ITEMS_PER_ORDER_MIN}, ${SS_ITEMS_PER_ORDER_MAX] itmes.

ss_sold_date_sk	ss_item_sk,	ss_customer_sk	ss_ticket_number	ss_quantity	ss_wholesale_cost	ss_list_price	ss_sales_price	ss_ext_list_price

ss_customer_sk	BIGINT		Q6 Q7 Q12 Q13 Q17 Q20 Q21 Q25 Q26	Random reference to table: customer c_customer_sk
ss_cdemo_sk	BIGINT		Q9	same cdemo_sk as referenced customer selected in ss_customer_sk
ss_hdemo_sk	BIGINT			same hdemo_sk as referenced customer selected in ss_customer_sk
ss_addr_sk	BIGINT		Q9	same addr_sk as referenced customer selected in ss_customer_sk
ss_store_sk	BIGINT		Q1 Q9 Q15 Q17 Q18 Q21	Random reference to s_store_sk
ss_promo_sk	BIGINT		Q17	Random reference to p_promo_sk
ss_ticket_number	BIGINT	NOT NULL	Q1 Q20 Q21 Q25	Dense unique sequence. Starts at ${SK_ID_OFFSET}
ss_quantity	INTEGER		Q21 Q24	Purchased quantity of item Random integer from [1, ${SS_QUANTITY_MAX}]
ss_wholesale_cost	DECIMAL (7 ,2)			Random decimal from [1, ${SS_WHOLESALE_MAX}]
ss_list_price	DECIMAL (7 ,2)			List price of single item: ss_wholesale_cost * (1 + random[0.00, ${SS_MARKUP_MAX}])
ss_sales_price	DECIMAL (7 ,2)		Q9	Sales price of single item: ss_listPrice * (1 - random[0.00, ${SS_DISCONUT_MAX}])
ss_ext_discount_amt	DECIMAL (7 ,2)		Q6	Discount of item times quantity: ss_ext_list_price - ss_ext_sales_price
ss_ext_sales_price	DECIMAL (7 ,2)		Q6 Q17	Sales price of item times quantity: ss_sales_price * ss_quantity
ss_ext_wholesale_cost	DECIMAL (7 ,2)		Q6	Wholesale cost of item times quantity ss_wholesale_cost * ss_quantity
ss_ext_list_price	DECIMAL (7 ,2)		Q6	List price of item times quantity ss_list_price * ss_quantity
ss_ext_tax	DECIMAL (7 ,2)			Random[0.00, 0.09] * ss_net_paid
ss_coupon_amt	DECIMAL (7 ,2)			Coupon discount Probability: 0.8: value: 0.00 0.2: Value: ss_ext_sales_price * random[0.00, 1.00]
ss_net_paid	DECIMAL (7 ,2)		Q13 Q15 Q17 Q20 Q25	Net paid of item times quantity ss_ext_sales_price * ss_coupon_amt
ss_net_paid_inc_tax	DECIMAL (7 ,2)			Net paid including tax of item times quantity ss_net_paid +ss_ext_tax
ss_net_profit	DECIMAL (7 ,2)		Q9	Profit on that item purchase ss_net_paid - ss_ext_wholesale_cost

Notes:

DISTRIBUTE BY HASH (ss_item_sk);

B.3.14 store_returns

${store_returns_size} = ${return_store_sale_likelihood} * ${store_sales_size}

Store_returns contains returned items for store_sales. A logical store_sale is identified by ss_ticket_id. This table must not contain more the one logical return entry for the same ss_ticket_id.

If a store sale is returned, a customer may not return the complete order, but only some items from it. Additionally he may have purchased 10 units of a certain item, but only returns, e.g., 5 of them. Return not all but random 1-N items from a selected store_sale. Like in store_sales, one logical "store_return" contains multiple items and produces a store_sales table row per returned item.

04.09.2004	23	2345	1	5	45,40	54,35	26,44	271,73
04.09.2004	76	2345	1	1	23,23	29,62	8,67	29,62
04.09.2004	365	2345	1	7	25,52	37,92	33,65	265,41
05.09.2004	637	734	2	1	65,52	92,03	26,69	276,08
05.09.2004	345	734	2	3	24,48	48,45	1,80	48,45

sr_return_date_sk	sr_item_sk,	sr_customer_sk	sr_ticket_number	sr_quantity	sr_wholesale_cost	sr_list_price	sr_sales_price	sr_ext_list_price
19.09.2004	23	2345	1	5 (of 5)	45,40	54,35	26,44	271,73
19.09.2004	365	2345	1	1 (of 7)	25,52	37,92	33,65	37,92
08.12.2008	637	734	2	2 (of 3)	65,52	92,03	26,69	184,06
05.09.2004	345	734	2	3	24,48	48,45	1,80	48,45

Logical sale

Pick a random unique store_sale ticket_number. The selected store_sale consists of N items. From these N items return random M items.

M=random[1, N]

1=same for every M

Write M lines for a logical return into store_returns table

sr_returned_date_sk	**1**
sr_return_time_sk,	1
sr_item_sk,	M
sr_customer_sk,	1
sr_cdemo_sk,	1
sr_hdemo_sk,	1
sr_addr_sk,	1
sr_store_sk,	1
sr_reason_sk,	N
sr_ticket_number	1
sr_return_quantity,	N
sr_return_amt	M
sr_return_tax	M
sr_return_amt_inc_tax	M
sr_fee	M
sr_return_ship_cost	M
sr_refunded_cash	M
sr_reversed_charge	M
sr_store_credit	M
sr_net_loss,	1

store_returns	Type	NULL?	Table is used by queries:	Description
sr_returned_date_sk	BIGINT default 9999999		Q19 Q20 Q21	Random reference to date after! referenced store_sales ss_soled_date_sk with same ticket number

sr_return_time_sk,	BIGINT			Random reference to time_dim t_time_sk
sr_item_sk,	BIGINT	NOT NULL	Q19 Q20 Q21	Random [1-N] item_sk's from ss_item_sk's in referenced store_sales ss_ticket_number (not necessary only one or all items from a store_sales ticket are returned)
sr_customer_sk,	BIGINT		Q20 Q21	Reference to customer_sk, same as in store_sales with same ticket number
sr_cdemo_sk,	BIGINT			Reference to cdemo_sk, same as in store_sales with same ticket number
sr_hdemo_sk,	BIGINT			Reference to hdemo_sk, same as in store_sales with same ticket number
sr_addr_sk,	BIGINT			Reference to addr_sk, same as in store_sales with same ticket number
sr_store_sk,	BIGINT			Reference to store_sk, same as in store_sales with same ticket number
sr_reason_sk,	BIGINT			random Reference to reason r_reason_sk for every returned item
sr_ticket_number	BIGINT	NOT NULL	Q20 Q21	Reference a uniqe existing ticket from store_sales ss_ticket_number
sr_return_quantity,	INTEGER		Q19 Q21	M, Number of returned items in this logical return.
sr_return_amt	DECIMAL (7 ,2)		Q20	ss_sales_price * sr_return_quantity
sr_return_tax	DECIMAL (7 ,2)			sr_return_amt * tax_pct with tax_pct = random decimal between [0.00, 0.09]
sr_return_amt_inc_tax	DECIMAL (7 ,2)			sr_return_amt + sr_return_tax
sr_fee	DECIMAL (7 ,2)			Random decimal between [0.50, 100.00]
sr_return_ship_cost	DECIMAL (7 ,2)			ss_list_price * shipping(=randDecimal[0.00, 1.00] * sr_return_quantity
sr_refunded_cash	DECIMAL (7 ,2)			rand[0.0,1.0] * sr_return_amt
sr_reversed_charge	DECIMAL (7 ,2)			rand[0.01, 1.00] * (sr_return_amt - sr_refunded_cash)
sr_store_credit	DECIMAL (7 ,2)			sr_return_amt - sr_reversed_charge - sr_refunded_cash
sr_net_loss,	DECIMAL (7 ,2)			sr_net_loss = sr_return_amt + sr_return_ship_cost + sr_return_tax - sr_store_credit - sr_refunded_cash - sr_reversed_charge + sr_fee

Notes:

DISTRIBUTE BY HASH (sr_item_sk);

B.3.15 web_sales

${web_sales_size} = 90000.0d * ${SF_linear}

One logical sale consists of random[${WS_ITEMS_PER_ORDER_MIN}, ${WS_ITEMS_PER_ORDER_MAX] itmes.

Logical sale	N = random[${WS_ITEMS_PER_ORDER_MIN}, ${WS_ITEMS_PER_ORDER_MAX] 1=same for every N Write N lines for a logical Sale into web_sales table
ws_sk	**1**
ws_sold_date_sk	1
ws_sold_time_sk,	1
ws_ship_date_sk,	1
ws_item_sk,	N
ws_bill_customer_sk,	1
ws_bill_cdemo_sk,	1
ws_bill_hdemo_sk,	1
ws_bill_addr_sk,	1
ws_ship_customer_sk,	1

ws_ship_cdemo_sk,	1
ws_ship_hdemo_sk,	1
ws_ship_addr_sk,	1
ws_web_page_sk,	1
ws_web_site_sk,	1
ws_ship_mode_sk,	1
ws_warehouse_sk,	1
ws_promo_sk,	1
ws_order_number,	1
ws_quantity,	N
ws_wholesale_cost	N
ws_list_price	N
ws_sales_price	N
ws_ext_discount_amt	N
ws_ext_sales_price	N
ws_ext_wholesale_cost	N
ws_ext_list_price	N
ws_ext_tax	N
ws_coupon_amt	N
ws_ext_ship_cost	N
ws_net_paid	N
ws_net_paid_inc_tax	N
ws_net_paid_inc_ship	N
ws_net_paid_inc_ship_tax	N
ws_net_profit	N

B.3.16 web_returns

	Type		NULL?	Table is used by queries:	Description
wr_returned_date_sk	BIGINT	default 9999999		Q19	Random reference to date after! referenced web_sales ws_soled_date_sk with same order number
wr_returned_time_sk	BIGINT				Random reference to time_dim t_time_sk
wr_item_sk	BIGINT		NOT NULL	Q16 Q19	Random [1-N] item_sk's from ws_item_sk's in referenced web_sales ws_order_number (not necessary only one or all items from a store_sales ticket are returned)
wr_refunded_customer_sk	BIGINT				Probability choice ${WS_GIFT_PCT} : Random reference to table: customer c_customer_sk 1 - ${WS_GIFT_PCT} : same as ws_ship_customer_sk
wr_refunded_cdemo_sk	BIGINT				same cdemo_sk as referenced customer selected in wr_refundedl_customer_sk
wr_refunded_hdemo_sk	BIGINT				same hdemo_sk as referenced customer selected in wr_refundedl_customer_sk
wr_refunded_addr_sk	BIGINT				same addr_sk as referenced customer selected in wr_refundedl_customer_sk
wr_returning_customer_sk	BIGINT				Same as wr_refundedl_customer_sk
wr_returning_cdemo_sk	BIGINT				Same as wr_refunded_cdemo_sk
wr_returning_hdemo_sk	BIGINT				Same as wr_refunded_hdemo_sk
wr_returning_addr_sk	BIGINT				Same as wr_refunded_addr_sk
wr_web_page_sk	BIGINT				Reference to ws_web_page_sk, same as in web_sales with same order_number
wr_reason_sk	BIGINT				random Reference to reason r_reason_sk for every returned item

wr_order_number	BIGINT	Q16	Reference a uniqe existing order_number from web_sales ws_order_number
wr_return_quantity	INTEGER	Q19	Random number of returned pieces for every returned sr_item_sk: Random[1, ss_quantity]
wr_return_amt	DECIMAL (7 ,2)		ws_sales_price * wr_return_quantity
wr_return_tax	DECIMAL (7 ,2)		wr_return_amt * tax_pct with tax_pct = random decimal between [0.00, 0.09]
wr_return_amt_inc_tax	DECIMAL (7 ,2)		wr_return_amt + wr_return_tax
wr_fee	DECIMAL (7 ,2)		Random decimal between [0.50, 100.00]
wr_return_ship_cost	DECIMAL (7 ,2)		ws_list_price * random[0.00, 1.00] * wr_return_quantity
wr_refunded_cash	DECIMAL (7 ,2)	Q16	rand[0.0,1.0] * wr_return_amt
wr_reversed_charge	DECIMAL (7 ,2)		rand[0.01, 1.00] * (wr_return_amt -wr_refunded_cash)
wr_account_credit	DECIMAL (7 ,2)		wr_return_amt -wr_reversed_charge - wr_refunded_cash
wr_net_loss	DECIMAL (7 ,2)		wr_return_amt + wr_return_ship_cost + wr_return_tax - wr_store_credit - wr_refunded_cash - wr_reversed_charge + wr_fee
DISTRIBUTE BY HASH (wr_item_sk);			

B.3.17 web_returns

${web_returns_size} = ${return_web_sale_likelihood} * ${web_sales_size}

web_returns contains returned items for web_sales. A logical web_sale is identified by ws_ticket_id. This table must not contain more the one logical return entry for the same ws_order_number.

Return not all but random 1-N items from a selected web_sale. Like in web_sales, one logical "web_return" contains multiple items and prodcues a web_sales table row per returned item.

Logical sale	Pick a random unique web_sale ws_order_number. The selected web_sale consists of N items. From these N items return random M items. M=random[1, N] 1=same for every M Write M lines for a logical return into web_returns table
wr_returned_date_sk,	1
wr_returned_time_sk,	1
wr_item_sk,	M
wr_refunded_customer_sk,	1
wr_refunded_cdemo_sk	1
wr_refunded_hdemo_sk	1
wr_refunded_addr_sk	1
wr_returning_customer_sk	1
wr_returning_cdemo_sk	1
wr_returning_hdemo_sk	1
wr_returning_addr_sk	1
wr_web_page_sk	M
wr_reason_sk	M
wr_order_number	1
wr_return_quantity	M
wr_return_amt,	M
wr_return_tax	M

wr_return_amt_inc_tax	M
wr_fee	M
wr_return_ship_cost	M
wr_refunded_cash	M
wr_reversed_charge	M
wr_account_credit	M
wr_net_loss	M

B.3.18 web_clickstreams

size: ${clickstreams_chunksize} * ${web_sales_size}

Web-clickstream contains information about each click (e.g. clicking on a link on a webpage) during a visitor's session.

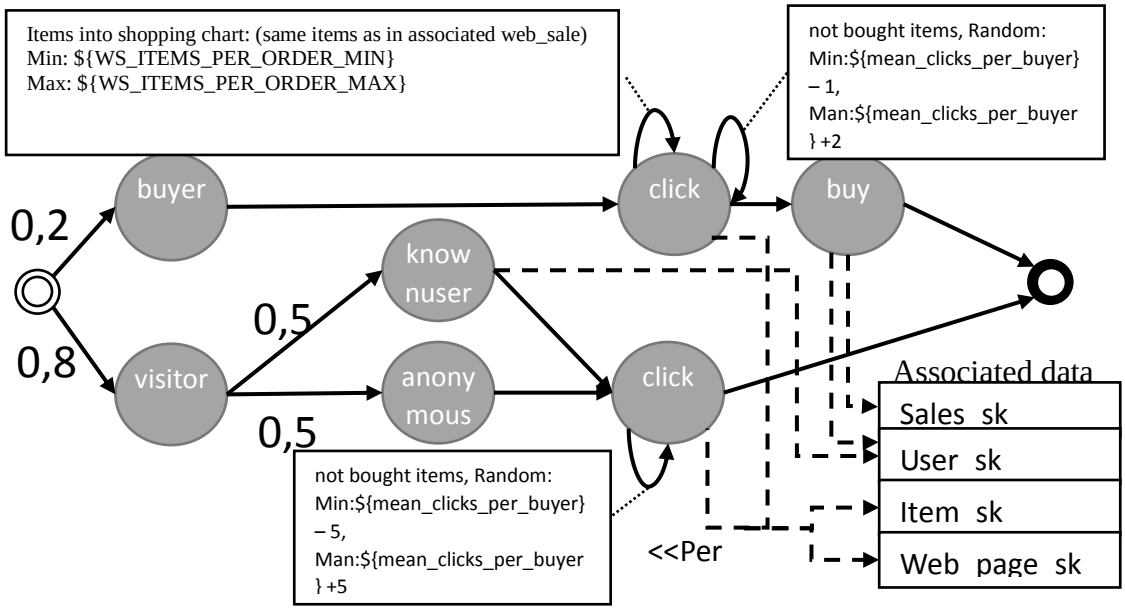
Every visitor generates a "chunk" of n-lines with the same wcs_click_sk in the web_clickstreams table. The table lines of each "chunk" are not continuous but interleaved with lines from other "chunks" (as they would be in a real "clickstream" log file as seen by the webserver).

Every clickstream "chunk" consists of multiple clicks with a total between: random [mean_clicks_per_visitor-1, mean_clicks_per_visitor+5].

Depending on the user type (buyer/visitor), there are different associated paths and data fields.

20% of all clicks are "buyers". Buyers are registered users with a user_sk and a buy has an associated sales_sk. User_sk and sales_sk are linked to corresponding entries from the web_sales table. Obviously, every item bought in web_sales was "clicked" by a user. In additon to the items bought, a user may have clicked on addional rand[${mean_clicks_per_buyer}-1, ${mean_clicks_per_buyer}+2] items he or she only viewed. It is important that the implicit referential integrity between web_sales and web_clickstreams is consistent.

80% of all clicks are "visitors". A visitor clickstream does not end in a purchase. Nevertheless, a "visitor" can still be a logged in user with an associated user_sk. 50% of the visitors are logged in users and 50% are anonymous. Both, known and anonymous users, share the same behavior of doing rand[${mean_clicks_per_buyer}-5, ${mean_clicks_per_buyer}+5] clicks during their session.



web_clickstreams	Type	NULL?	Table is used by queries:	Description
wcs_click_sk	BIGINT	NOT NULL		
wcs_click_date_sk	BIGINT		Q3 Q4 Q8 Q12	Probability choice: ${visitor_likelihood} : Visitor 1- ${visitor_likelihood} : Buyer Case Visitor: ${web_sales_days_since_date_begin_date} + Math.floor((current ID or row) * (($web_sales_days_within} - 1) / ${web_clickstreams_size})) Case Buyer: The clickstream must have the same reference to web_sales_ws_sold_date_sk as the associate web_sale (choosen by: wcs_user_sk)
wcs_click_time_sk	BIGINT		Q3 Q4 Q8 Q12	Probability choice: same choice as wcs_click_date_sk, ${visitor_likelihood} : Visitor 1- ${visitor_likelihood} : Buyer Case Visitor: Random referece to time_dim t_time_sk Case Buyer: Random referece to web_sales_ws_sold_time_sk
wcs_sales_sk	BIGINT		Q3 Q8	Probability choice: same choice as wcs_click_date_sk, ${visitor_likelihood} : Visitor 1- ${visitor_likelihood} : Buyer Case Visitor: Value: “”
wcs_item_sk	BIGINT	can be null	Q2 Q3 Q4 Q5 Q8 Q12 Q30	Case Buyer: (Current row or id) * 1 / ${clickstreams_chunksize} Probability choice: same choice as wcs_click_date_sk, ${visitor_likelihood} : Visitor 1- ${visitor_likelihood} : Buyer Case Visitor: Random referece to item i_item_sk Case Buyer: The clickstream must contain all ws_item_sk from the associated web_sale (choosen by wcs_user_sk) plus additional random[${mean_clicks_per_buyer} – 1, ${mean_clicks_per_buyer} +2] clicked items (random references to item i_item_sk) which where not purchased.
wcs_web_page_sk	BIGINT		Q8	Probability choice: same choice as wcs_click_date_sk, ${visitor_likelihood} : Visitor 1- ${visitor_likelihood} : Buyer Case Visitor: Random referece to web_page wp_web_page_sk Case Buyer: The clickstream must contain all ws_web_page_sk’s from the associated web_sale plus additional random[${mean_clicks_per_buyer} – 1, ${mean_clicks_per_buyer} +2] clicked ws_web_page_sk’s (random references to web_page wp_web_page_sk) .
wcs_user_sk	BIGINT	can be null	Q2 Q3 Q4 Q5 Q8 Q12 Q30	One random web_page_sk for every random wcs_item sk (same random choice as wcs_item_sk) Probability choice: same choice as wcs_click_date_sk, ${visitor_likelihood} : Visitor 1- ${visitor_likelihood} : Buyer Case Visitor: Probability choice: ${visitor_known_likelihood}: known visitor

1- ${visitor_known_likelihood} :unknown visitor

Case Buyer:
Choose a buying user from ws_user_sk
Note: wcs_click_date_sk, wcs_item_sk and wcs_web_page_sk must reflect the values from the associated web_sale (purchasing multiple items in one “clickstream-session”): ws_user_sk, ws_click_date_sk, ws_item_sk and ws_web_page_sk

Notes

DISTRIBUTE BY HASH (wcs_click_sk);

B.3.19 warehouse

${warehouse_size} =5.0d * ${SF_log_5}

warehouse	Type	NULL?	Table is used by queries:	Description
w_warehouse_sk	BIGINT	NOT NULL	Q16 Q23	Dense unique sequence. Starts at ${SK_ID_OFFSET}
w_warehouse_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: “ABCDEFGHIJKLMNOPQRSTUVWXYZ” Example: AAAAAAAAOKJNECAA
w_warehouse_name	VARCHAR (20)		Q22 Q23	Text (multiple words) len(min/max): 5
w_warehouse_sq_ft	INTEGER			Unifrom between ${W_SQFT_MIN}, ${W_SQFT_MAX}
w_street_number	CHAR (10)			DIST_UNIFORM, 1, 1000,
w_street_name	VARCHAR (60)			Probability: 50% 1 word “%s” 50% 2 Words “%s %s” From Weighted list "street_names", valueCol:0 weightCoL 0
w_street_type	CHAR (15)			Weighted list "street_type", valueCol:0 weightCoL 0,
w_suite_number	CHAR (10)			Fromat: “Suite %d" DIST_UNIFORM, 1, 100 suite number is alphabetic 50% of the time
w_city	VARCHAR (60)			ity is picked from a distribution which maps to large/medium/small Weighted list "cities". Value:0 weight:5
w_county	VARCHAR (30)			Weighted list "fips_county" value column “county”, choose a “region” use same region for all cols: county, state, zip, country, gmt_offset
w_state	CHAR (2)		Q16	Weighted list "fips_county” value column “st” match region and country
w_zip	CHAR (10)			Random number [10000, 99999]
w_country	VARCHAR (20)			Allways "United States"
w_gmt_offset	DECIMAL (5 ,2)			Weighted list "fips_county” value column “gmt” match state and county

Notes:

DISTRIBUTE BY REPLICATION ;

B.3.20 web_page

${web_page_size} =60 * ${SF_log_1.5}

web_page	Type	NULL?	Table is used by queries:	Description
wp_web_page_sk	BIGINT	NOT NULL	Q4 Q8 Q14	Dense unique sequence. Starts at ${SK_ID_OFFSET}
wp_web_page_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: “ABCDEFGHIJKLMNOPQRSTUVWXYZ”

				Example: AAAAAAAAOKJNECAA
wp_rec_start_date	DATE			Date from: ${web_page_begin_date} to: ${web_sales_begin_date} Format: yyyy-MM-dd With: wp_rec_start_date{n} < wp_rec_start_date{n+1} where n= **wp_web_page_sk**
wp_rec_end_date	DATE			No end date for the moment. Value: “” Elese: 50% Empty 50%: wp_rec_start_date + rand[2years, 4years]
wp_creation_date_sk	BIGINT			Random reference to table: date d_date_sk
wp_access_date_sk	BIGINT			Random reference to table: date d_date_sk Else: wp_rec_access_date >= wp_rec_creation_date
wp_autogen_flag	CHAR (1)			Propability : value ${WP_AUTOGEN_PCT}: 1 1-${WP_AUTOGEN_PCT}: 0
wp_customer_sk	BIGINT			Random reference to table: customer c_customer_sk
wp_url	VARCHAR (100)			“http://www.” + RANDOMSTRING_[4, 85] +“.com”
wp_type	CHAR (50)	Q4 Q8		Weighted list " web_page_use” value column “0”
wp_char_count	INTEGER	Q14		Radom integer between: min =wp_link_count * 125 + wp_image_count * 50 max =wp_link_count * 300 + wp_image_count * 150
wp_link_count	INTEGER			Random integer between: [${WP_LINK_MIN}, ${WP_LINK_MIN}]
wp_image_count	INTEGER			Random integer between: [${WP_IMAGE_MIN}, ${WP_IMAGE_MIN}]
wp_max_ad_count	INTEGER			Random integer between: [${WP_AD_MIN}, ${WP_AD_MIN}]
Note:				
DISTRIBUTE BY REPLICATION ;				

B.3.21 web_site

(UNUSED/UNREFERENCED) only ref: web_sales

web_site	Type	NULL?	Table is used by querys:	Description
web_site_sk	BIGINT	NOT NULL		Dense unique sequence. Starts at ${SK_ID_OFFSET} referenced by web_sales
web_site_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: “ABCDEFGHIJKLMNOPQRSTUVWXYZ” Example: AAAAAAAAOKJNECAA
web_rec_start_date	DATE			Date from: 1997-08-16 to: 2001-08-16 Format: yyyy-MM-dd With: iweb_rec_start_date {n} < web_rec_start_date {n+1} where n= **web_site_sk**
web_rec_end_date	DATE			No end date for the moment. Value: “” Elese: 50% Empty 50%: wp_rec_start_date + rand[2years, 4years]
web_name	VARCHAR (50)			Template: „site_%d“ with %d = current_row
web_open_date_sk	BIGINT			Random reference to date table d_date_sk
web_close_date_sk	BIGINT			Radom d_date_sk > web_open_date_sk
web_class	VARCHAR (50)			Value: "Unknown"
web_manager	VARCHAR (40)			Pattern: "%s %s" Weighted List ‘first_names’, value_col= 0; weight_col: 0 Weighted List ‘last_names’; , value_col= 0; weight_col: 0

web_mkt_id	INTEGER	Random integer between: [1, 6]
web_mkt_class	VARCHAR (50)	Sentences following pseudo englisch gramatic Example: Clear circumstances know then further white companies. Typical budgets take both required children. Appeals must not make civil, financial representatives. Emotional areas shall wear only.
web_mkt_desc	VARCHAR (100)	Sentences following pseudo englisch gramatic Example: Clear circumstances know then further white companies. Typical budgets take both required children. Appeals must not make civil, financial representatives. Emotional areas shall wear only.
web_market_manager	VARCHAR (40)	Pattern: "%s %s" Weighted List 'first_names', value_col= 0; weight_col: 0 Weighted List 'last_names'; , value_col= 0; weight_col: 0
web_company_id	INTEGER	Random integer between: [1, 6]
web_company_name	CHAR (50)	One random word from Weighted List 'syllables'
web_street_number	CHAR (10)	Address like in warehouse
web_street_name	VARCHAR (60)	Address like in warehouse
web_street_type	CHAR (15)	Address like in warehouse
web_suite_number	CHAR (10)	Address like in warehouse
web_city	VARCHAR (60)	Address like in warehouse
web_county	VARCHAR (30)	Address like in warehouse
web_state	CHAR (2)	Address like in warehouse
web_zip	CHAR (10)	Address like in warehouse
web_country	VARCHAR (20)	Address like in warehouse
web_gmt_offset	DECIMAL (5 ,2)	Address like in warehouse
web_tax_percentage	DECIMAL (5 ,2)	Random decimal betweem [0.00, 0.12]

Notes:

DISTRIBUTE BY REPLICATION ;

B.3.22 reason

only referenced by sr_reason_sk and wr_reason_sk (both not used in queries)

size: 35 * ${SF_log_1.5}

reason	Type	NULL?	Table is used by queries:	Description Example: 1|AAAAAAAABAAAAAAA|Package was damaged|
r_reason_sk	BIGINT	NOT NULL		Dense unique sequence. Starts at ${SK_ID_OFFSET}
r_reason_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
r_reason_desc	CHAR (100)			Weighted List 'return_reasons', row = r_reason_sk; value_col= 0; weight column: 0

Notes.

DISTRIBUTE BY REPLICATION ;

B.3.23 ship_mode

(UNUSED/UNREFERENCED)

size: fixed size of 20

ship_mode	Type	NULL?	Table is used by querys:	Description Example: 1|AAAAAAAABAAAAAAA|EXPRESS|AIR|UPS|YvxVaJI10|
sm_ship_mode_sk	BIGINT	NOT NULL		Dense unique sequence. Starts at ${SK_ID_OFFSET}
sm_ship_mode_id	CHAR (16)	NOT NULL		Unique String, len: 16 charset: "ABCDEFGHIJKLMNOPQRSTUVWXYZ" Example: AAAAAAAAOKJNECAA
sm_type	CHAR (30)			Weighted list "ship_mode". Value:0 weight:0
sm_code	CHAR (10)			Weighted list "ship_mode_code". Value:0 weight:0
sm_carrier	CHAR (20)			Weighted list "ship_mode_carrier ". Value:0 weight:0
sm_contract	CHAR (20)			RandString ALPHANUM, min/max: RS_SM_CONTRACT, SM_CONTRACT

Notes:

DISTRIBUTE BY REPLICATION ;

B.3.24 income_band

(NOT USED!)

size: fixed 20

income_band	Type	NULL?	Table is used by querys:	Description Example: 1|0|10000|
ib_income_band_sk,	BIGINT	NOT NULL		Dense unique sequence. Starts at ${SK_ID_OFFSET}
ib_lower_bound	INTEGER			Weighted List 'income_band' ; row= ib_income_band_sk, valueCol=0
ib_upper_bound	INTEGER			Weighted List 'income_band' ; row= ib_income_band_sk, valueCol=1

Note:

DISTRIBUTE BY REPLICATION ;

B.4 Data Generation

The data generator used is based on an extension of the Parallel Data Generation Framework (PDGF). PDGF is a parallel data generator that is capable of producing large amounts of data for an arbitrary schema. The existing PDGF can be used to generate the structured part of the BigBench model. However, it is not capable of generating the unstructured product reviews text. First, PDGF is enhanced to produce a key-value data set for a fixed set of required and optional keys. This is sufficient to generate the weblogs part of BigBench.

The main challenge in product reviews is producing the unstructured text. This is achieved by an algorithm that produces synthetic text based on sample input text. The algorithm uses a Markov Chain technique that extracts key words and builds a dictionary based on these key words. The new algorithm is applied for BigBench by using some real product reviews from an online retailer for the initial sample data. PDGF interacts with the review generator through an API sending a product category as input and receiving a product review text for that category.

The volume dimension model is far simpler than the variety discussion and previous data generators had a good handle on that. PDGF handles the volume well since it can scale the size of the data based on a scale factor. It also runs efficiently for large scale factors since it runs in parallel and can leverage large systems dedicated for the benchmark.

B.5 Query Overview

This section illustrates a high level overview of the 30 queries of BigBench. It is structured into a general overview of the different query types, a textual description of the 30 queries as well as specific characteristics of implementation.

B.5.1 Query types

The queries used in BigBench can be grouped into three categories: Structured, semi-structured and unstructured. The following table illustrates the data types that the queries access as specified in Clause B.1.

1	Structured	16	Structured
2	Semi-Structured	17	Structured
3	Semi-Structured	18	Un-Structured
4	Semi-Structured	19	Un-Structured
5	Semi-Structured	20	Structured
6	Structured	21	Structured
7	Structured	22	Structured
8	Semi-Structured	23	Structured
9	Structured	24	Structured
10	Un-Structured	25	Structured
11	Structured	26	Structured
12	Semi-Structured	27	Un-Structured
13	Structured	28	Un-Structured
14	Structured	29	Structured
15	Structured	30	Semi-Structured

B.5.2 Query Grouping

The overall number of the thirty queries has been grouped into four categories: Pure Hive queries, Hive queries with MapReduce programs, Hive queries using natural language processing, and queries using Apache MLLIB. In the following, an example for each of the different flavors of queries will be given. The distribution of the different query types is shown in the following table.

Use case	Method	Use case	Method
1	UDF/UDTF	16	Pure QL
2	Map Reduce	17	Pure QL
3	Map Reduce	18	UDF/UDTF/NLP
4	Map Reduce	19	UDF/UDTF/NLP
5	ML	20	ML
6	Pure QL	21	Pure QL
7	Pure QL	22	Pure QL
8	Map Reduce	23	Pure QL
9	Pure QL	24	Pure QL
10	UDF/UDTF/NLP	25	ML
11	Pure QL	26	ML
12	Pure QL	27	UDF/UDTF/NLP
13	Pure QL	28	ML
14	Pure QL	29	UDF/UDTF
15	Pure QL	30	UDF/UDTF/Map Reduce

It should be noted that queries that use NLTK and Mahout also require preprocessing by Hive. Therefore, Apache Hive is critical to all data processing activities in this implementation of BigBench.

B.6 Query Descriptions

This section gives a textual description of each query.

B.6.1 Query 01

Find top 100 products that are sold together frequently in given stores. Only products in certain categories sold in specific stores are considered, and "sold together frequently" means at least 50 customers bought these productstogether in a transaction.

B.6.2 Query 02

Find the top 30 products that are mostly viewed together with a given product in online store. Note that the order of products viewed does not matter, and "viewed together" relates to a web_clickstreams click_session of a known user with a session timeout of 60min.If the duration between two click of a user is greater then the session timeout, a new session begins. With a session timeout of 60min.

B.6.3 Query 03

For a given product get a top 30 list sorted by number of views in descending order of the last 5 products that are mostly viewed before the product was purchased online. For the viewed products, consider only products in certain item categories and viewed within 10days before the purchase date.

B.6.4 Query 04

Web_clickstream shopping cart abandonment analysis: For users who added products in their shopping carts but did not check out in the online store during their session, find the average number of pages they visited during their sessions. A "session" relates to a click_session of a known user with a session time-out of 60min.If the duration between two clicks of a user is greater then the session time-out, a new session begins.

B.6.5 Query 05

Build a model using logistic regression for a visitor to an online store: based on existing users online activities (interest in items of different categories) and demographics. This model will be used to predict if the visitor is interested in a given item category. Output the precision, accuracy and confusion matrix of model.

Note: no need to actually classify existing users, as it will be later used to predict interests of unknown visitors.

B.6.6 Query 06

Identifies customers shifting their purchase habit from store to web sales. Find customers who spend in relation more money in the second year following a given year in the web_sales channel then in the store sales channel. Report customers details: first name, last name, their country of origin, login name and email address) and identify if they are preferred customer, for the top 100 customers with the highest increase in their second year web purchase ratio.

B.6.7 Query 07

List top 10 states in descending order with at least 10 customers who during a given month bought products with the price tag at least 20% higher than the average price of products in the same category.

B.6.8 Query 08

For online sales, compare the total sales monetary amount in which customers checked online reviews before making the purchase and that of sales in which customers did not read reviews. Consider only online sales for a specific category in a given year.

B.6.9 Query 09

Aggregate total amount of sold items over different given types of combinations of customers based on selected groups of marital status, education status, sales price and different combinations of state and sales profit.

B.6.10 Query 10

For all products, extract sentences from its product reviews that contain positive or negative sentiment and display for each item the sentiment polarity of the extracted sentences (POS OR NEG) and the sentence and word in sentence leading to this classification.

B.6.11 Query 11

For a given product, measure the correlation of sentiments, including the number of reviews and average review ratings, on product monthly revenues within a given time frame.

B.6.12 Query 12

Find all customers who viewed items of a given category on the web in a given month and year that was followed by an instore purchase of an item from the same category in the three consecutive months.

B.6.13 Query 13

Display customers with both store and web sales in consecutive years for whom the increase in web sales exceeds the increase in store sales for a specified year.

B.6.14 Query 14

What is the ratio between the number of items sold over the internet in the morning (7 to 8am) to the number of items sold in the evening (7 to 8pm) of customers with a specified number of dependents. Consider onlywebsites with a high amount of content.

B.6.15 Query 15

Find the categories with flat or declining sales for in store purchases during a given year for a given store.

B.6.16 Query 16

Compute the impact of an item price change on the store sales by computing the total sales for items in a 30 day period before and after the price change. Group the items by location of warehouse where they were delivered from.

B.6.17 Query 17

Find the ratio of items sold with and without promotions in a given month and year. Only items in certain categories sold to customersliving in a specific time zone are considered.

B.6.18 Query 18

Identify the stores with flat or declining sales in 3 consecutive months, check if there are any negative reviews regarding these stores available online.

B.6.19 Query 19

Retrieve the items with the highest number of returns where the number of returns was approximately equivalent across all store and web channels (within a tolerance of +/ 10%), within the week ending given dates. Analyse the online reviews for these items to see if there are any major negative reviews.

B.6.20 Query 20

Customer segmentation for return analysis: Customers are separated along the following dimensions: return frequency, return order ratio (total number of orders partially or fully returned versus the total number of orders), return item ratio (total number of items returned versus the number of items purchased), return amount ration (total monetary amount of items returned versus the amount purchased), return order ratio. Consider the store returns during a given year for the computation.

B.6.21 Query 21

Get all items that were sold in stores in a given month and year and which were returned in the next 6 months and repurchased by the returning customer afterwards through the web sales channel in the following three years. For those items, compute the total quantity sold through the store, the quantity returned and the quantity purchased through the web. Group this information by item and store.

B.6.22 Query 22

For all items whose price was changed on a given date, compute the percentage change in inventory between the 30day period BEFORE the price change and the 30day period AFTER the change. Group this information by warehouse.

B.6.23 Query 23

This query contains multiple, related iterations: Iteration 1: Calculate the coefficient of variation and mean of every item and warehouse of the given and the consecutive month. Iteration 2: Find items that had a coefficient of variation of 1.3 or larger in the given and the consecutive month

B.6.24 Query 24

For a given product, measure the effect of competitor's prices on products' instore and online sales. Compute the crossprice elasticity of demand for a given product.

B.6.25 Query 25

Customer segmentation analysis: Customers are separated along the following key shopping dimensions: recency of last visit, frequency of visits and monetary amount. Use the store and online purchase data during a given year to compute. After model of separation is build, report for the analysed customers to which "group" they where assigned.

B.6.26 Query 26

Cluster customers into book buddies/club groups based on their in store book purchasing histories. After model of separation is build, report for the analysed customers to which "group" they where assigned.

B.6.27 Query 27

Extract competitor product names and model names (if any) from online product reviews for a given product.

B.6.28 Query 28

Build text classifier for online review sentiment classification (Positive, Negative, Neutral), using 90% of available reviews for training and the remaining 10% for testing. Display classifier accuracy on testing data and classification result for the 10% testing data: <reviewSK>,<originalRating>,<classificationResult>.

B.6.29 Query 29

Perform category affinity analysis for products purchased together online. Note that the order of products viewed does not matter,

B.6.30 Query 30

Perform category affinity analysis for products viewed together online. Note that the order of products viewed does not matter,and "viewed together" relates to a click_session of a user with a session timeout of 60min. If the duration between two clicks of a user is greater then the session timeout, a new session begins.

B.7 Schema

In the following, the complete schema definition for TPCx-BB Hive is listed in Appendix I

B.8 Weighted lists

See files: weightedList_probabilities.txt and productReviews_weighted_list_probabilities.txt.

Appendix C.

Query Parameters

```
-- !echo ============================;
-- !echo <settings from queryParameters.sql>;
-- !echo ============================;
--new (dates all Mondays, dateranges complete weeks):
--store: 2000-01-03, 2004-01-05 (1463 days, 209 weeks)
--item: 2000-01-03, 2004-01-05 (1463 days, 209 weeks)
--web_page: 2000-01-03, 2004-01-05 (1463 days, 209 weeks)
--store_sales: 2001-01-01, 2006-01-02 (1827 days, 261 weeks)
--web_sales: 2001-01-01, 2006-01-02 (1827 days, 261 weeks)
--inventory: 2001-01-01, 2006-01-02 (1820 days, 261 weeks)


-- READ ME
-- ITEM_SK
-- Datagenerator ensures that item_sk's 10000-10002 are very frequent accross
all scalefactors

-------- Q01 -----------
--category_ids:
--1 Home & Kitchen
--2 Music
--3 Books
--4 Clothing & Accessories
--5 Electronics
--6 Tools & Home Improvement
--7 Toys & Games
--8 Movies & TV
--9 Sports & Outdoors
set q01_i_category_id_IN=1, 2 ,3;
-- sf1 -> 11 stores, 90k sales in 820k lines
set q01_ss_store_sk_IN=10, 20, 33, 40, 50;
set q01_viewed_together_count=50;
set q01_limit=100;

-------- Q02 -----------
-- q02_pid1_IN=<pid>, <pid>, ..
--pid == item_sk
--sf 1 item count: 17999c
set q02_item_sk=10001;
set q02_MAX_ITEMS_PER_BASKET=5000000;
set q02_limit=30;
set q02_session_timeout_inSec=3600;


-------- Q03 -----------
set q03_days_in_sec_before_purchase=864000;
set q03_views_before_purchase=5;
set q03_purchased_item_IN=10001;
--see q1 for categories
set q03_purchased_item_category_IN=2,3;
set q03_limit=30;

-------- Q04 -----------
set q04_session_timeout_inSec=3600;

-------- Q05 -----------
```

```
set q05_i_category='Books';
set q05_cd_education_status_IN='Advanced Degree', 'College', '4 yr Degree', '2
yr Degree';
set q05_cd_gender='M';


-------- Q06 -----------
SET q06_LIMIT=100;
--web_sales and store_sales date
SET q06_YEAR=2001;


-------- Q07 -----------
SET q07_HIGHER_PRICE_RATIO=1.2;
--store_sales date
SET q07_YEAR=2004;
SET q07_MONTH=7;
SET q07_HAVING_COUNT_GE=10;
SET q07_LIMIT=10;

-------- Q08 -----------
set q08_category=review;
-- web_clickstreams date range
set q08_startDate=2001-09-02;
-- + 1year
set q08_endDate=2002-09-02;
set q08_days_before_purchase=1;


-------- Q09 -----------
--store_sales date
set q09_year=2001;

set q09_part1_ca_country=United States;
set q09_part1_ca_state_IN='KY', 'GA', 'NM';
set q09_part1_net_profit_min=0;
set q09_part1_net_profit_max=2000;
set q09_part1_education_status=4 yr Degree;
set q09_part1_marital_status=M;
set q09_part1_sales_price_min=100;
set q09_part1_sales_price_max=150;

set q09_part2_ca_country=United States;
set q09_part2_ca_state_IN='MT', 'OR', 'IN';
set q09_part2_net_profit_min=150;
set q09_part2_net_profit_max=3000;
set q09_part2_education_status=4 yr Degree;
set q09_part2_marital_status=M;
set q09_part2_sales_price_min=50;
set q09_part2_sales_price_max=200;

set q09_part3_ca_country=United States;
set q09_part3_ca_state_IN='WI', 'MO', 'WV';
set q09_part3_net_profit_min=50;
set q09_part3_net_profit_max=25000;
set q09_part3_education_status=4 yr Degree;
set q09_part3_marital_status=M;
set q09_part3_sales_price_min=150;
set q09_part3_sales_price_max=200;

-------- Q10 -----------
```

```
--no params

-------- Q11 -----------
--web_sales date range
set q11_startDate=2003-01-02;
-- +30days
set q11_endDate=2003-02-02;


-------- Q12 -----------
--web_clickstreams start_date - endDate1
--store_sales   start_date - endDate2
set q12_startDate=2001-09-02;
set q12_endDate1=2001-10-02;
set q12_endDate2=2001-12-02;
set q12_i_category_IN='Books', 'Electronics';

-------- Q13 -----------
--store_sales date
set q13_Year=2001;

set q13_limit=100;

-------- Q14 -----------
set q14_dependents=5;
set q14_morning_startHour=7;
set q14_morning_endHour=8;
set q14_evening_startHour=19;
set q14_evening_endHour=20;
set q14_content_len_min=5000;
set q14_content_len_max=6000;

-------- Q15 -----------
--store_sales date range
set q15_startDate=2001-09-02;
--+1year
set q15_endDate=2002-09-02;
set q15_store_sk=10;


-------- Q16 -----------
-- web_sales/returns date
set q16_date=2001-03-16;

-------- Q17 -----------
set q17_gmt_offset=-5;
--store_sales date
set q17_year=2001;
set q17_month=12;
set q17_i_category_IN='Books', 'Music';

-------- Q18 -----------
-- store_sales date range
set q18_startDate=2001-05-02;
--+90days
set q18_endDate=2001-09-02;

-------- Q19 -----------
set q19_storeReturns_date_IN='2004-03-8' ,'2004-08-02' ,'2004-11-15', '2004-12-
20';
```

```
set  q19_webReturns_date_IN='2004-03-8' ,'2004-08-02' ,'2004-11-15', '2004-12-
20';
set q19_store_return_limit=100;

-------- Q20 -----------
--no params

-------- Q21 -----------
--store_sales/returns web_sales/returns date
-- ss_date_sk range at SF 1
--36890  2001-01-01
--38697  2005-12-13
set q21_year=2003;
set q21_month=1;
set q21_limit=100;

-------- Q22 -----------
--inventory date
set q22_date=2001-05-08;
set q22_i_current_price_min=0.98;
set q22_i_current_price_max=1.5;

-------- Q23 -----------
--inventory date
set q23_year=2001;
set q23_month=1;
set q23_coefficient=1.5;

-------- Q24 -----------
set q24_i_item_sk_IN=10000, 10001;

-------- Q25 -----------
-- store_sales and web_sales date
set q25_date=2002-01-02;

-------- Q26 -----------
set q26_i_category_IN='Books';
set q26_count_ss_item_sk=5;

-------- Q27 -----------
set q27_pr_item_sk=10002;

-------- Q28 -----------
--no params

-------- Q29 -----------
set q29_limit=100;
set q29_session_timeout_inSec=3600;


-------- Q30 -----------
set q30_limit=100;
set q30_session_timeout_inSec=3600;


-- !echo ============================;
-- !echo </settings from queryParameters.sql>;
-- !echo ============================;
```

Appendix D.

– Benchmark generic Parameters.

```
## ==========================
## JAVA environment
## ==========================
export BIG_BENCH_JAVA="java"

## ==========================
## default settings for benchmark
## ==========================
export BIG_BENCH_DEFAULT_DATABASE="bigbenchORC"
export BIG_BENCH_DEFAULT_ENGINE="hive"
export BIG_BENCH_DEFAULT_MAP_TASKS="80"
export BIG_BENCH_DEFAULT_SCALE_FACTOR="10"
export BIG_BENCH_DEFAULT_NUMBER_OF_PARALLEL_STREAMS="2"
export BIG_BENCH_DEFAULT_BENCHMARK_PHASE="run_query"

## ==========================
## HADOOP environment
## ==========================

##folder containing the cluster setup *-site.xml files like core-site.xml
export BIG_BENCH_HADOOP_CONF="/etc/hadoop/conf.cloudera.hdfs"
export BIG_BENCH_HADOOP_LIBS_NATIVE="/opt/cloudera/parcels/CDH/lib/hadoop/lib/native"

## memory used by sub-processes spawned by Hive queries (like streaming M/R jobs etc.)
## Suggestion for value: (YarnConatiner_MB - hive_MB)*0.7 e.g. (2000Mb-500Mb)*0.7=1050
export BIG_BENCH_java_child_process_xmx=" -Xmx1024m "

## ==========================
## HDFS config and paths
## ==========================
export BIG_BENCH_USER="$USER"
export BIG_BENCH_HDFS_ABSOLUTE_PATH="/user/$BIG_BENCH_USER" ##working dir of benchmark.
export BIG_BENCH_HDFS_RELATIVE_HOME="benchmarks/bigbench"
export
BIG_BENCH_HDFS_RELATIVE_INIT_DATA_DIR="$BIG_BENCH_HDFS_RELATIVE_HOME/data"
export
BIG_BENCH_HDFS_RELATIVE_REFRESH_DATA_DIR="$BIG_BENCH_HDFS_RELATIVE_HOME/data_
refresh"
export
BIG_BENCH_HDFS_RELATIVE_QUERY_RESULT_DIR="$BIG_BENCH_HDFS_RELATIVE_HOME/query
Results"
export BIG_BENCH_HDFS_RELATIVE_TEMP_DIR="$BIG_BENCH_HDFS_RELATIVE_HOME/temp"
export
BIG_BENCH_HDFS_ABSOLUTE_HOME="$BIG_BENCH_HDFS_ABSOLUTE_PATH/$BIG_BENCH_HDF
S_RELATIVE_HOME"
```

```
export
BIG_BENCH_HDFS_ABSOLUTE_INIT_DATA_DIR="$BIG_BENCH_HDFS_ABSOLUTE_PATH/$BIG_BENCH_HDFS_RELATIVE_INIT_DATA_DIR"
export
BIG_BENCH_HDFS_ABSOLUTE_REFRESH_DATA_DIR="$BIG_BENCH_HDFS_ABSOLUTE_PATH/$BIG_BENCH_HDFS_RELATIVE_REFRESH_DATA_DIR"
export
BIG_BENCH_HDFS_ABSOLUTE_QUERY_RESULT_DIR="$BIG_BENCH_HDFS_ABSOLUTE_PATH/$BIG_BENCH_HDFS_RELATIVE_QUERY_RESULT_DIR"
export
BIG_BENCH_HDFS_ABSOLUTE_TEMP_DIR="$BIG_BENCH_HDFS_ABSOLUTE_PATH/$BIG_BENCH_HDFS_RELATIVE_TEMP_DIR"

# ------------------------------------------
# Hadoop data generation options
# ------------------------------------------
# specify JVM arguments like: -Xmx2000m;
# default of: 800m is sufficient if the datagen only uses 1 worker thread per map task
# Add +100MB per addition worker if you modified: BIG_BENCH_DATAGEN_HADOOP_OPTIONS
export BIG_BENCH_DATAGEN_HADOOP_JVM_ENV="$BIG_BENCH_JAVA -Xmx800m"

# if you increase -workers, you must also increase the -Xmx setting in BIG_BENCH_DATAGEN_HADOOP_JVM_ENV;
#-ap:=automatic progress ,3000ms intervall; prevents hadoop from killing long running jobs
#-workers:=limit hadoop based data generator to use 1 CPU core per map task.
export BIG_BENCH_DATAGEN_HADOOP_OPTIONS=" -workers 1 -ap 3000 "

#(recommended:1 to save space, 3 default) replication count for files written by the datagenerator to HDFS into dir: BIG_BENCH_HDFS_ABSOLUTE_INIT_DATA_DIR and BIG_BENCH_HDFS_ABSOLUTE_REFRESH_DATA_DIR
export BIG_BENCH_DATAGEN_DFS_REPLICATION="1"

# if empty, generate all tables.
# Else: specify tables to generate e.g.: BIG_BENCH_DATAGEN_TABLES="item customer store"
# Tables to choose from: customer customer_address customer_demographics date_dim household_demographics income_band inventory item item_marketprices product_reviews promotion reason ship_mode store store_returns store_sales time_dim warehouse web_clickstreams web_page web_returns web_sales web_site
export BIG_BENCH_DATAGEN_TABLES=""

# if distributed data generation fails, re run with BIG_BENCH_DATAGEN_HADOOP_EXEC_DEBUG="-testDebugMessages" to retrieve more information.
export BIG_BENCH_DATAGEN_HADOOP_EXEC_DEBUG=""

# the default behaviour is to stop when a query error occurs
# set this to 0 to keep on running when an error occurs
export BIG_BENCH_STOP_AFTER_FAILURE="1"

## requires "snakebite" to be installed https://github.com/spotify/snakebite
## yum install epel-release
## yum install -y python-pip
```

```
## pip install snakebite
#0==off 1==on
export BIG_BENCH_USE_SNAKEBITE_HDFSCLIENT="0"
```

Appendix E.

```
--########### README ################
-- The default way to set hive options is doing it globally for your whole cluster (e.g. cloudera manager, ambari, hive-site.xml, ...)
-- However, if for some reasons you cant or wont change your cluster global config, you can enable hive specific tuning options in this file.
-- Below are listed some commonly used settings. The values you see in this file may not apply to your own cluster! we used some of them on our 3 node (16cores 60gb ram) test instances
--#####################################


--##########################
-- EXECUTION ENGINE
--##########################
-- values: mr, tez, spark
-- set hive.execution.engine=mr;

-- ##########################
-- parallel order by. required by queries:
-- ##########################
set bigbench.hive.optimize.sampling.orderby=true;
set bigbench.hive.optimize.sampling.orderby.number=20000;
set bigbench.hive.optimize.sampling.orderby.percent=0.1;

-- ##########################
-- output and itermediate table settings
-- ##########################
-- if you cluster has good cpu's but limited network bandwith, this could speed up the exchange of intermediate results (this option should be turund on if you cluster has high 'net wait i/o%'
-- set hive.exec.compress.intermediate=true;
-- set mapred.map.output.compression.codec=org.apache.hadoop.io.compress.SnappyCodec;

-- default is to keep the created result tables human readable.
-- set hive.exec.compress.output=false;
-- set mapred.output.compression.codec=org.apache.hadoop.io.compress.DefaultCodec;

-- set hive.default.fileformat=ORC;

-- ##########################
-- mappers settings
-- ##########################
-- Number of mappers used by HIVE, based on table sizes. If you experience underutilization or to much mappers/reducers, you can play with these settings
-- The number of physical files a table consists of is irrelevant for hives metric for estimating number of mappers. (Hive uses HiveCombineInputFormat, joining the files)
-- the following two parameters are most effective in influencing hives estimation of mappers. To low settings may result in to many map tasks, while to high size settings result in to few map tasks and underutilization of the cluster.
```

```
-- both extremes are harmful to the performance. For small data set sizes of 1-100GB a good value for max.split.size may be 134217728 (128MB). As an estimation, take a medium sized table and divide its size by the number of map tasks you need to utilize your cluster.

-- set mapreduce.input.fileinputformat.split.minsize=1048576;
-- set mapreduce.input.fileinputformat.split.maxsize=67108864;

-- ############################
-- reducer settings
-- ############################
-- Number of reducers used by HIVE
-- hives metric for estimating reducers is mostly controlled by the following settings. Node: Some Query functions like count(*) or Distinct will lead to hive always using only 1 reducer
-- 1GB default
-- set hive.exec.reducers.bytes.per.reducer=33554432;

-- ############################
-- optimizations for joins.
-- ############################
-- things like mapjoins are done in memory and require a lot of it
-- README!
-- Hive 0.12 bug, hive ignores 'hive.mapred.local.mem' resulting in out of memory errors in map joins!
-- (more exactly: bug in Hadoop 2.2 where hadoop-env.cmd sets the -xmx parameter multiple times, effectively overriding the user set hive.mapred.local.mem setting. see: https://issues.apache.org/jira/browse/HADOOP-10245
-- There are 3 workarounds:
-- 1) assign more memory to the local!! Hadoop JVM client (not! mapred.map.memory)-> map-join child vm will inherit the parents jvm settings
-- 2) reduce "hive.smalltable.filesize" to ~1MB (depends on your cluster settings for the local JVM)
-- 3) turn off "hive.auto.convert.join" to prevent hive from converting the join to a mapjoin.

-- MAP join settings:
-- set hive.auto.convert.join.noconditionaltask.size=100000;

-- set hive.auto.convert.join=true;
-- set hive.optimize.mapjoin.mapreduce=true;
-- set hive.mapred.local.mem=1024;
-- default:25MB, max size of tables considered for local in memory map join. Beware! ORC files have only little file size but huge in memory data size! a 25MB ORC easily consumes 512MB.. related: https://issues.apache.org/jira/browse/HIVE-2601
-- set hive.mapjoin.smalltable.filesize=10000;
-- set hive.mapjoin.localtask.max.memory.usage=0.90;
-- set hive.auto.convert.sortmerge.join=true;
-- set hive.auto.convert.sortmerge.join.noconditionaltask=true;
-- set hive.auto.convert.join.noconditionaltask.size=100000;
-- set hive.optimize.bucketmapjoin=true;
-- set hive.optimize.bucketmapjoin.sortedmerge=false;
-- set hive.optimize.skewjoin=true; --READ FIRST: https://issues.apache.org/jira/browse/HIVE-5888
-- set hive.optimize.skewjoin.compiletime=true;
-- set hive.groupby.skewindata=true;
```

```
-- ##########################
-- Other tuning options
-- ##########################
-- exec.parallel is still considered unstable, but has the potential to increase you utilization by running multiple independent stages of a query in parallel
-- set hive.exec.parallel=true;
-- set hive.exec.parallel.thread.number=8;

-- you should really turn these options on for your whole cluster, not just for bigbench
-- predicate pushdown for ORC-files (eager filtering of columns)
-- set hive.optimize.ppd=true;
-- set hive.optimize.ppd.storage=true;
-- set hive.ppd.recognizetransivity=false;
-- set hive.optimize.index.filter=true;
-- set hive.stats.autogather=true;
-- set hive.auto.convert.sortmerge.join=true;
-- set hive.vectorized.execution.enabled=true;
-- set hive.vectorized.execution.reduce.enabled=true;
-- set hive.cbo.enable=true;
-- set hive.compute.query.using.stats=true;
-- set hive.stats.fetch.column.stats=true;
-- set hive.stats.fetch.partition.stats=true;
-- set hive.script.operator.truncate.env=true;


-- ============================;
-- Print most important properties;
-- ============================;
--exec engine and optimizer
set hive.execution.engine;
set hive.cbo.enable;
set hive.stats.fetch.partition.stats;
set hive.script.operator.truncate.env;
set hive.compute.query.using.stats;
set hive.vectorized.execution.enabled;
set hive.vectorized.execution.reduce.enabled;
set hive.stats.autogather;
--input output
set mapreduce.input.fileinputformat.split.minsize;
set mapreduce.input.fileinputformat.split.maxsize;
set hive.exec.reducers.bytes.per.reducer;
set hive.exec.reducers.max;
set hive.exec.parallel;
set hive.exec.parallel.thread.number;
set hive.exec.compress.intermediate;
set hive.exec.compress.output;
set mapred.map.output.compression.codec;
set mapred.output.compression.codec;
set hive.default.fileformat;
--join optimizations
set hive.auto.convert.sortmerge.join;
```

```
set hive.auto.convert.sortmerge.join.noconditionaltask;
set hive.optimize.bucketmapjoin;
set hive.optimize.bucketmapjoin.sortedmerge;
set hive.auto.convert.join.noconditionaltask.size;
set hive.auto.convert.join;
set hive.optimize.mapjoin.mapreduce;
set hive.mapred.local.mem;
set hive.mapjoin.smalltable.filesize;
set hive.mapjoin.localtask.max.memory.usage;
set hive.optimize.skewjoin;
set hive.optimize.skewjoin.compiletime;
-- filter optimizations (predicate pushdown to storage level)
set hive.optimize.ppd;
set hive.optimize.ppd.storage;
set hive.ppd.recognizetransivity;
set hive.optimize.index.filter;
--other
set hive.optimize.sampling.orderby=true;
set hive.optimize.sampling.orderby.number;
set hive.optimize.sampling.orderby.percent;
set bigbench.hive.optimize.sampling.orderby;
set bigbench.hive.optimize.sampling.orderby.number;
set bigbench.hive.optimize.sampling.orderby.percent;
set hive.groupby.skewindata;
set hive.exec.submit.local.task.via.child;

-- Database - DO NOT DELETE OR CHANGE
CREATE DATABASE IF NOT EXISTS ${env:BIG_BENCH_DATABASE};
use ${env:BIG_BENCH_DATABASE};
```

Appendix F.

```
-- !echo ============================;
-- !echo <settings from engineLocalSettings.sql/conf>;
-- !echo ============================;

-------- Q01 ----------- Example only.
set hive.mapjoin.localtask.max.memory.usage = 3556
set hive.auto.convert.join = false
```

Appendix G.

```
############################
#    Hardware    #
############################

##### /proc/cpuinfo #####

processor       : 0-31
vendor_id       : GenuineIntel
cpu family      : 6
model           : 63
model name      : Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
stepping        : 2
microcode       : 37
cpu MHz                 : 2394.725
cache size      : 30720 KB
physical id     : 0
siblings : 20
core id         : 0
cpu cores       : 10
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good xtopology nonstop_tsc aperfmperf unfair_spinlock pni pclmulqdq monitor est ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm ida xsaveopt fsgsbase bmi1 avx2 smep bmi2 erms invpcid
bogomips        : 4789.45
clflush size    : 64
cache_alignment         : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:



##### /proc/meminfo #####

MemTotal:    165237704 kB
MemFree:    19945712 kB
Buffers:     1179232 kB
Cached:     129025956 kB
SwapCached:      0 kB
Active:     20598904 kB
Inactive:    119510804 kB
Active(anon):  9977840 kB
```

Inactive(anon): 176888 kB
Active(file): 10621064 kB
Inactive(file): 119333916 kB
Unevictable: 0 kB
Mlocked: 0 kB
SwapTotal: 0 kB
SwapFree: 0 kB
Dirty: 289028 kB
Writeback: 0 kB
AnonPages: 9885636 kB
Mapped: 456168 kB
Shmem: 269516 kB
Slab: 3992408 kB
SReclaimable: 3920760 kB
SUnreclaim: 71648 kB
KernelStack: 16640 kB
PageTables: 48880 kB
NFS_Unstable: 0 kB
Bounce: 0 kB
WritebackTmp: 0 kB
CommitLimit: 82618852 kB
Committed_AS: 27066612 kB
VmallocTotal: 34359738367 kB
VmallocUsed: 427252 kB
VmallocChunk: 34275747196 kB
HardwareCorrupted: 0 kB
AnonHugePages: 489472 kB
HugePages_Total: 0
HugePages_Free: 0
HugePages_Rsvd: 0
HugePages_Surp: 0
Hugepagesize: 2048 kB
DirectMap4k: 8188 kB
DirectMap2M: 167763968 kB

lscpu

Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
CPU(s): 40
On-line CPU(s) list: 0-31
Off-line CPU(s) list: 32-39
Thread(s) per core: 1
Core(s) per socket: 10
Socket(s): 2
NUMA node(s): 2
Vendor ID: GenuineIntel
CPU family: 6
Model: 63
Stepping: 2

```
CPU MHz:       2394.725
BogoMIPS:      4788.60
Hypervisor vendor:   Xen
Virtualization type:  full
L1d cache:      32K
L1i cache:      32K
L2 cache:       256K
L3 cache:       30720K
NUMA node0 CPU(s):   0-9,20-29
NUMA node1 CPU(s):   10-19,30,31

##### lspci #####

00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 01)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:03.0 Unassigned class [ff80]: XenSource, Inc. Xen Platform Device (rev 01)

##### lsblk #####

NAME  MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda  202:0  0 250G 0 disk
└─xvda1 202:1  0 250G 0 part /
xvdb  202:16  0 500G 0 disk /hdfs

##### ifconfig #####

eth0   Link encap:Ethernet HWaddr 02:EC:2E:D9:52:AF
    inet addr:172.31.40.36 Bcast:172.31.47.255 Mask:255.255.240.0
    inet6 addr: fe80::ec:2eff:fed9:52af/64 Scope:Link
    UP BROADCAST RUNNING MULTICAST MTU:9001 Metric:1
    RX packets:1551206455 errors:0 dropped:0 overruns:0 frame:0
    TX packets:913371611 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:7619845808430 (6.9 TiB) TX bytes:8075427723927 (7.3 TiB)
    Interrupt:172

lo    Link encap:Local Loopback
    inet addr:127.0.0.1 Mask:255.0.0.0
    inet6 addr: ::1/128 Scope:Host
    UP LOOPBACK RUNNING MTU:65536 Metric:1
    RX packets:474992213 errors:0 dropped:0 overruns:0 frame:0
    TX packets:474992213 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:0
    RX bytes:6400983832792 (5.8 TiB) TX bytes:6400983832792 (5.8 TiB)

##############################
#    Software    #
##############################
```

```
##### linux release #####

LSB_VERSION=base-4.0-amd64:base-4.0-noarch:core-4.0-amd64:core-4.0-noarch:graphics-4.0-
amd64:graphics-4.0-noarch:printing-4.0-amd64:printing-4.0-noarch
Red Hat Enterprise Linux Server release 6.6 (Santiago)
Red Hat Enterprise Linux Server release 6.6 (Santiago)

##### kernel release #####

Linux bmarktest01.local.com 2.6.32-504.23.4.el6.x86_64 #1 SMP Fri May 29 10:16:43 EDT 2015 x86_64 x86_64
x86_64 GNU/Linux

##### date #####

Sat Aug 15 00:14:54 EDT 2015

##### hadoop version #####

Hadoop 2.6.0-cdh5.4.4
Subversion http://github.com/cloudera/hadoop -r b739cd891f6269da5dd22766d7e75bd2c9db73b6
Compiled by jenkins on 2015-07-07T00:02Z
Compiled with protoc 2.5.0
From source with checksum 4acea6ac185376e0b48b33695e88e7a7
This command was run using /opt/cloudera/parcels/CDH-5.4.4-1.cdh5.4.4.p0.4/jars/hadoop-common-2.6.0-
cdh5.4.4.jar

##### hadoop classpath #####

/etc/hadoop/conf:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop/lib/*:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop/.//*:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop-hdfs/./:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop-hdfs/lib/*:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop-hdfs/.//*:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop-yarn/lib/*:/opt/cloudera/parcels/CDH-5.4.4-
1.cdh5.4.4.p0.4/lib/hadoop/libexec/../../hadoop-yarn/.//*:/opt/cloudera/parcels/CDH/lib/hadoop-
mapreduce/lib/*:/opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/.//*

##### java version #####

java version "1.7.0_79"
OpenJDK Runtime Environment (rhel-2.5.5.3.el6_6-x86_64 u79-b14)
OpenJDK 64-Bit Server VM (build 24.79-b02, mixed mode)

##### environment #####

BASH=/bin/bash
BASHOPTS=cmdhist:extquote:force_fignore:hostcomplete:interactive_comments:progcomp:promptvars:so
urcepath
BASH_ALIASES=()
```

```
BASH_ARGC=([0]="11")
BASH_ARGV=([0]="-U" [1]="300" [2]="-m" [3]="1000" [4]="-f" [5]="LOAD_TEST" [6]="-i" [7]="1000" [8]="-f"
[9]="-b" [10]="zipQueryLogs")
BASH_CMDS=()
BASH_LINENO=([0]="465" [1]="0")
BASH_SOURCE=([0]="/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/bin/bigBench"        [1]="/home/ec2-user/Big-Data-Benchmark-for-Big-
Bench-MasterVersion_14_Aug_incl_kmeans/bin/bigBench")
BASH_VERSINFO=([0]="4" [1]="1" [2]="2" [3]="1" [4]="release" [5]="x86_64-redhat-linux-gnu")
BASH_VERSION='4.1.2(1)-release'
BIG_BENCH_BENCHMARK_PHASE=run_query
BIG_BENCH_BIN_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/bin
BIG_BENCH_CLEAN_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/engines/hive/clean
BIG_BENCH_CLEAN_METASTORE_FILE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/engines/hive/clean/dropTables.sql
BIG_BENCH_CONF_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/conf
BIG_BENCH_DATABASE=bigbenchORC
BIG_BENCH_DATAGEN_CORE_SITE=/etc/hadoop/conf.cloudera.hdfs/core-site.xml
BIG_BENCH_DATAGEN_DFS_REPLICATION=3
BIG_BENCH_DATAGEN_HADOOP_EXEC_DEBUG=
BIG_BENCH_DATAGEN_HADOOP_JVM_ENV='java -Xmx800m'
BIG_BENCH_DATAGEN_HADOOP_OPTIONS=' -workers 1 -ap 3000 '
BIG_BENCH_DATAGEN_HDFS_SITE=/etc/hadoop/conf.cloudera.hdfs/hdfs-site.xml
BIG_BENCH_DATAGEN_STAGE_LOG=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/logs/dataGeneration-run_query.log
BIG_BENCH_DATAGEN_TABLES=
BIG_BENCH_DATA_GENERATOR_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/data-generator
BIG_BENCH_DEFAULT_BENCHMARK_PHASE=run_query
BIG_BENCH_DEFAULT_DATABASE=bigbenchORC
BIG_BENCH_DEFAULT_ENGINE=hive
BIG_BENCH_DEFAULT_MAP_TASKS=80
BIG_BENCH_DEFAULT_NUMBER_OF_PARALLEL_STREAMS=2
BIG_BENCH_DEFAULT_SCALE_FACTOR=10
BIG_BENCH_ENGINE=hive
BIG_BENCH_ENGINE_BIN_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/engines/hive/bin
BIG_BENCH_ENGINE_CONF_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/engines/hive/conf
BIG_BENCH_ENGINE_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/engines/hive
BIG_BENCH_ENGINE_HIVE_MAHOUT_EXECUTION=sequential
BIG_BENCH_ENGINE_SETTINGS_FILE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-
MasterVersion_14_Aug_incl_kmeans/engines/hive/conf/hiveSettings.sql
BIG_BENCH_EXPERT_MODE=1
BIG_BENCH_HADOOP_CONF=/etc/hadoop/conf.cloudera.hdfs
BIG_BENCH_HADOOP_LIBS_NATIVE=/opt/cloudera/parcels/CDH/lib/hadoop/lib/native
BIG_BENCH_HDFS_ABSOLUTE_HOME=/user/ec2-user/benchmarks/bigbench
```

BIG_BENCH_HDFS_ABSOLUTE_INIT_DATA_DIR=/user/ec2-user/benchmarks/bigbench/data
BIG_BENCH_HDFS_ABSOLUTE_PATH=/user/ec2-user
BIG_BENCH_HDFS_ABSOLUTE_QUERY_RESULT_DIR=/user/ec2-user/benchmarks/bigbench/queryResults
BIG_BENCH_HDFS_ABSOLUTE_REFRESH_DATA_DIR=/user/ec2-user/benchmarks/bigbench/data_refresh
BIG_BENCH_HDFS_ABSOLUTE_TEMP_DIR=/user/ec2-user/benchmarks/bigbench/temp
BIG_BENCH_HDFS_RELATIVE_HOME=benchmarks/bigbench
BIG_BENCH_HDFS_RELATIVE_INIT_DATA_DIR=benchmarks/bigbench/data
BIG_BENCH_HDFS_RELATIVE_QUERY_RESULT_DIR=benchmarks/bigbench/queryResults
BIG_BENCH_HDFS_RELATIVE_REFRESH_DATA_DIR=benchmarks/bigbench/data_refresh
BIG_BENCH_HDFS_RELATIVE_TEMP_DIR=benchmarks/bigbench/temp
BIG_BENCH_HOME=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans
BIG_BENCH_JAVA=java
BIG_BENCH_LOADING_STAGE_LOG=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/logs/populateMetastore-run_query.log
BIG_BENCH_LOGS_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/logs
BIG_BENCH_MAP_TASKS=300
BIG_BENCH_NUMBER_OF_PARALLEL_STREAMS=2
BIG_BENCH_POPULATE_METASTORE_FILE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/population/hiveCreateLoad_decimal.sql
BIG_BENCH_POPULATION_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/population
BIG_BENCH_QUERIES_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/queries
BIG_BENCH_QUERY_PARAMS_FILE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/conf/queryParameters.sql
BIG_BENCH_REFRESH_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/refresh
BIG_BENCH_REFRESH_METASTORE_FILE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/refresh/hiveRefreshCreateLoad_decimal.sql
BIG_BENCH_SCALE_FACTOR=1000
BIG_BENCH_STOP_AFTER_FAILURE=0
BIG_BENCH_STREAM_NUMBER=0
BIG_BENCH_TOOLS_DIR=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/tools
BIG_BENCH_USER=ec2-user
BIG_BENCH_USE_SNAKEBITE_HDFSCLIENT=0
BIG_BENCH_hive_default_fileformat_result_table=TEXTFILE
BIG_BENCH_hive_default_fileformat_source_table=ORC
BIG_BENCH_java_child_process_xmx=' -Xmx1024m '
BINARY=/usr/bin/hive
BINARY_PARAMS=()
CVS_RSH=ssh
DIRSTACK=()
ENGINE_RUN_METHOD=runEngineCmd
ENGINE_SETTINGS=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/engines/hive/conf/engineSettings.conf

ENV_INFO_FILE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/logs/envInfo.log
EUID=500
FUNCNAME=([0]="logEnvInformation" [1]="main")
GROUPS=()
G_BROKEN_FILENAMES=1
HISTCONTROL=ignoredups
HISTSIZE=1000
HOME=/home/ec2-user
HOSTNAME=bmarktest01.local.com
HOSTTYPE=x86_64
IFS=$' \t\n'
JAVA_HOME=/usr/lib/jvm/jre
LANG=en_US.UTF-8
LESSOPEN='||/usr/bin/lesspipe.sh %s'
LIST_OF_USER_OPTIONS='-b -f 1000 -i LOAD_TEST -f 1000 -m 300 -U'
LOGNAME=ec2-user
LS_COLORS='rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;05;37;41:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz=01;31:*.xz=01;31:*.bz2=01;31:*.tbz=01;31:*.tbz2=01;31:*.bz=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.axv=01;35:*.anx=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=01;36:*.au=01;36:*.flac=01;36:*.mid=01;36:*.midi=01;36:*.mka=01;36:*.mp3=01;36:*.mpc=01;36:*.ogg=01;36:*.ra=01;36:*.wav=01;36:*.axa=01;36:*.oga=01;36:*.spx=01;36:*.xspf=01;36:'
MACHTYPE=x86_64-redhat-linux-gnu
MAIL=/var/spool/mail/ec2-user
MODULE=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/bin/zipQueryLogs
MODULE_HELP_METHOD=helpModule
MODULE_NAME=zipQueryLogs
MODULE_RUN_METHOD=runModule
NLSPATH=/usr/dt/lib/nls/msg/%L/%N.cat
OLDPWD=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans
OPT='?'
OPTERR=1
OPTIND=11
OSTYPE=linux-gnu
PATH=/usr/lib64/qt-3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/ec2-user/bin
PIPESTATUS=([0]="0")
PPID=63520
PS4='+ '
PWD=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans
QTDIR=/usr/lib64/qt-3.3
QTINC=/usr/lib64/qt-3.3/include
QTLIB=/usr/lib64/qt-3.3/lib
SHELL=/bin/bash

SHELLOPTS=braceexpand:hashall:interactive-comments
SHLVL=4
SHOW_HELP=0
SSH_CLIENT='172.31.22.134 58617 22'
SSH_CONNECTION='172.31.22.134 58617 172.31.40.36 22'
SSH_TTY=/dev/pts/1
TERM=xterm
UID=500
USER=ec2-user
USER_DRIVER_WORKLOAD=LOAD_TEST
USER_EXPERT_MODE=1
USER_MAP_TASKS=300
USER_PRINT_STD_OUT=1
USER_SCALE_FACTOR=1000
USER_SETTINGS=/home/ec2-user/Big-Data-Benchmark-for-Big-Bench-MasterVersion_14_Aug_incl_kmeans/conf/userSettings.conf
XFILESEARCHPATH=/usr/dt/app-defaults/%L/Dt

Appendix H.

```
====================================
[root@server-01 ~]# sudo -u hdfs hdfs fsck -blocks
Connecting to namenode via http://server-01.local.com:50070
FSCK started by hdfs (auth:SIMPLE) from /132.233.52.19 for path / at Tue Sep 01 16:31:39 PDT 2015
.....
.....
.....
.....Status: HEALTHY
 Total size:  223603888 B
 Total dirs:  3523
 Total files:  351
 Total symlinks:        0
 Total blocks (validated):   350 (avg. block size 638868 B)
 Minimally replicated blocks:  350 (100.0 %)
 Over-replicated blocks:    0 (0.0 %)
 Under-replicated blocks:    0 (0.0 %)
 Mis-replicated blocks:     0 (0.0 %)
 Default replication factor:  3
 Average block replication:   3.0
 Corrupt blocks:        0
 Missing replicas:      0 (0.0 %)
 Number of data-nodes:     4
 Number of racks:        1
FSCK ended at Tue Sep 01 16:31:39 PDT 2015 in 358 milliseconds


The filesystem under path '/' is HEALTHY

====================================

[root@Server-01]$ hadoop fs -du -h /user/hadoop/benchmarks/bigbench/data
T 78.4 T /user/hadoop/benchmarks/bigbench/data
```

Appendix I.

```
set hdfsDataPath=${env:BIG_BENCH_HDFS_ABSOLUTE_INIT_DATA_DIR};
set fieldDelimiter=|;
set tableFormat=${env:BIG_BENCH_hive_default_fileformat_source_table};
set temporaryTableSuffix=_temporary;

set customerTableName=customer;
set customerAddressTableName=customer_address;
set customerDemographicsTableName=customer_demographics;
set dateTableName=date_dim;
set householdDemographicsTableName=household_demographics;
set incomeTableName=income_band;
set itemTableName=item;
set promotionTableName=promotion;
set reasonTableName=reason;
set shipModeTableName=ship_mode;
set storeTableName=store;
set timeTableName=time_dim;
set warehouseTableName=warehouse;
set webSiteTableName=web_site;
set webPageTableName=web_page;
set inventoryTableName=inventory;
set storeSalesTableName=store_sales;
set storeReturnsTableName=store_returns;
set webSalesTableName=web_sales;
set webReturnsTableName=web_returns;

set marketPricesTableName=item_marketprices;
set clickstreamsTableName=web_clickstreams;
set reviewsTableName=product_reviews;



-- !echo Create temporary table: ${hiveconf:customerTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:customerTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:customerTableName}${hiveconf:temporaryTableSuffix}
 ( c_customer_sk            bigint          --not null
 , c_customer_id            string          --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             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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:customerTableName}'
;


-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:customerTableName};
DROP TABLE IF EXISTS ${hiveconf:customerTableName};
CREATE TABLE ${hiveconf:customerTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:customerTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:customerTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:customerTableName}${hiveconf:temporaryTableSuffix};



-- !echo Create temporary table:
${hiveconf:customerAddressTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:customerAddressTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE
${hiveconf:customerAddressTableName}${hiveconf:temporaryTableSuffix}
  ( ca_address_sk            bigint            --not null
  , ca_address_id            string            --not null
  , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION
'${hiveconf:hdfsDataPath}/${hiveconf:customerAddressTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:customerAddressTableName};
DROP TABLE IF EXISTS ${hiveconf:customerAddressTableName};
CREATE TABLE ${hiveconf:customerAddressTableName}
STORED AS ${hiveconf:tableFormat}
```

```
AS
SELECT * FROM ${hiveconf:customerAddressTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table:
${hiveconf:customerAddressTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:customerAddressTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table:
${hiveconf:customerDemographicsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS
${hiveconf:customerDemographicsTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE
${hiveconf:customerDemographicsTableName}${hiveconf:temporaryTableSuffix}
 ( cd_demo_sk              bigint              ----not null
 , 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 '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION
'${hiveconf:hdfsDataPath}/${hiveconf:customerDemographicsTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table:
${hiveconf:customerDemographicsTableName};
DROP TABLE IF EXISTS ${hiveconf:customerDemographicsTableName};
CREATE TABLE ${hiveconf:customerDemographicsTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:customerDemographicsTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table:
${hiveconf:customerDemographicsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:customerDemographicsTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:dateTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:dateTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:dateTableName}${hiveconf:temporaryTableSuffix}
 ( d_date_sk               bigint            --not null
 , d_date_id               string            --not null
 , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:dateTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:dateTableName};
DROP TABLE IF EXISTS ${hiveconf:dateTableName};
CREATE TABLE ${hiveconf:dateTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:dateTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:dateTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:dateTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table:
${hiveconf:householdDemographicsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS
${hiveconf:householdDemographicsTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE
${hiveconf:householdDemographicsTableName}${hiveconf:temporaryTableSuffix}
  ( hd_demo_sk              bigint              --not null
  , hd_income_band_sk       bigint
  , hd_buy_potential        string
```

```
 , hd_dep_count            int
 , hd_vehicle_count        int
 )
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION
'${hiveconf:hdfsDataPath}/${hiveconf:householdDemographicsTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table:
${hiveconf:householdDemographicsTableName};
DROP TABLE IF EXISTS ${hiveconf:householdDemographicsTableName};
CREATE TABLE ${hiveconf:householdDemographicsTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:householdDemographicsTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table:
${hiveconf:householdDemographicsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:householdDemographicsTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:incomeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:incomeTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:incomeTableName}${hiveconf:temporaryTableSuffix}
 ( ib_income_band_sk       bigint          --not null
 , ib_lower_bound          int
 , ib_upper_bound          int
 )
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:incomeTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:incomeTableName};
DROP TABLE IF EXISTS ${hiveconf:incomeTableName};
CREATE TABLE ${hiveconf:incomeTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:incomeTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:incomeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:incomeTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:itemTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:itemTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:itemTableName}${hiveconf:temporaryTableSuffix}
 ( i_item_sk               bigint          --not null
 , i_item_id               string          --not null
 , i_rec_start_date        string
```

```
  , i_rec_end_date          string
  , i_item_desc             string
  , i_current_price         decimal(7,2)
  , i_wholesale_cost        decimal(7,2)
  , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:itemTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:itemTableName};
DROP TABLE IF EXISTS ${hiveconf:itemTableName};
CREATE TABLE ${hiveconf:itemTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:itemTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:itemTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:itemTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:promotionTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:promotionTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:promotionTableName}${hiveconf:temporaryTableSuffix}
  ( p_promo_sk              bigint            --not null
  , p_promo_id              string            --not null
  , p_start_date_sk         bigint
  , p_end_date_sk           bigint
  , p_item_sk               bigint
  , p_cost                  decimal(15,2)
  , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:promotionTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:promotionTableName};
DROP TABLE IF EXISTS ${hiveconf:promotionTableName};
CREATE TABLE ${hiveconf:promotionTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:promotionTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:promotionTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:promotionTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:reasonTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:reasonTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:reasonTableName}${hiveconf:temporaryTableSuffix}
  ( r_reason_sk             bigint            --not null
  , r_reason_id             string            --not null
  , r_reason_desc           string
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:reasonTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:reasonTableName};
DROP TABLE IF EXISTS ${hiveconf:reasonTableName};
CREATE TABLE ${hiveconf:reasonTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:reasonTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:reasonTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:reasonTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:shipModeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:shipModeTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:shipModeTableName}${hiveconf:temporaryTableSuffix}
  ( sm_ship_mode_sk         bigint            --not null
```

```
  , sm_ship_mode_id          string            --not null
  , sm_type                 string
  , sm_code                 string
  , sm_carrier              string
  , sm_contract             string
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:shipModeTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:shipModeTableName};
DROP TABLE IF EXISTS ${hiveconf:shipModeTableName};
CREATE TABLE ${hiveconf:shipModeTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:shipModeTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:shipModeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:shipModeTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:storeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:storeTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:storeTableName}${hiveconf:temporaryTableSuffix}
  ( s_store_sk              bigint            --not null
  , s_store_id              string            --not null
  , s_rec_start_date        string
  , s_rec_end_date          string
  , s_closed_date_sk        bigint
  , 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            decimal(5,2)
  , s_tax_precentage        decimal(5,2)
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:storeTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:storeTableName};
DROP TABLE IF EXISTS ${hiveconf:storeTableName};
CREATE TABLE ${hiveconf:storeTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:storeTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:storeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:storeTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:timeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:timeTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:timeTableName}${hiveconf:temporaryTableSuffix}
  ( t_time_sk               bigint            --not null
  , t_time_id               string            --not null
  , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:timeTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:timeTableName};
DROP TABLE IF EXISTS ${hiveconf:timeTableName};
CREATE TABLE ${hiveconf:timeTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:timeTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:timeTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:timeTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:warehouseTableName}${hiveconf:temporaryTableSuffix};
```

```
DROP TABLE IF EXISTS ${hiveconf:warehouseTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:warehouseTableName}${hiveconf:temporaryTableSuffix}
 ( w_warehouse_sk          bigint            --not null
 , w_warehouse_id          string            --not null
 , 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            decimal(5,2)
 )
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:warehouseTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:warehouseTableName};
DROP TABLE IF EXISTS ${hiveconf:warehouseTableName};
CREATE TABLE ${hiveconf:warehouseTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:warehouseTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:warehouseTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:warehouseTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:webSiteTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:webSiteTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:webSiteTableName}${hiveconf:temporaryTableSuffix}
 ( web_site_sk             bigint            --not null
 , web_site_id             string            --not null
 , web_rec_start_date      string
 , web_rec_end_date        string
 , web_name                string
 , web_open_date_sk        bigint
 , web_close_date_sk       bigint
 , 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          decimal(5,2)
  , web_tax_percentage      decimal(5,2)
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:webSiteTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:webSiteTableName};
DROP TABLE IF EXISTS ${hiveconf:webSiteTableName};
CREATE TABLE ${hiveconf:webSiteTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:webSiteTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:webSiteTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:webSiteTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:webPageTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:webPageTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:webPageTableName}${hiveconf:temporaryTableSuffix}
  ( wp_web_page_sk          bigint          --not null
  , wp_web_page_id          string          --not null
  , wp_rec_start_date       string
  , wp_rec_end_date         string
  , wp_creation_date_sk     bigint
  , wp_access_date_sk       bigint
  , wp_autogen_flag         string
  , wp_customer_sk          bigint
  , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:webPageTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:webPageTableName};
```

```
DROP TABLE IF EXISTS ${hiveconf:webPageTableName};
CREATE TABLE ${hiveconf:webPageTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:webPageTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:webPageTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:webPageTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:inventoryTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:inventoryTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:inventoryTableName}${hiveconf:temporaryTableSuffix}
 ( inv_date_sk             bigint             --not null
 , inv_item_sk             bigint             --not null
 , inv_warehouse_sk        bigint             --not null
 , inv_quantity_on_hand    int
 )
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:inventoryTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:inventoryTableName};
DROP TABLE IF EXISTS ${hiveconf:inventoryTableName};
CREATE TABLE ${hiveconf:inventoryTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:inventoryTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:inventoryTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:inventoryTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:storeSalesTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:storeSalesTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:storeSalesTableName}${hiveconf:temporaryTableSuffix}
 ( 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             int
 , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:storeSalesTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:storeSalesTableName};
DROP TABLE IF EXISTS ${hiveconf:storeSalesTableName};
CREATE TABLE ${hiveconf:storeSalesTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:storeSalesTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:storeSalesTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:storeSalesTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:storeReturnsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:storeReturnsTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:storeReturnsTableName}${hiveconf:temporaryTableSuffix}
  ( 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       int
  , sr_return_amt            decimal(7,2)
  , sr_return_tax            decimal(7,2)
  , sr_return_amt_inc_tax    decimal(7,2)
  , sr_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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:storeReturnsTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:storeReturnsTableName};
DROP TABLE IF EXISTS ${hiveconf:storeReturnsTableName};
CREATE TABLE ${hiveconf:storeReturnsTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:storeReturnsTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:storeReturnsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:storeReturnsTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:webSalesTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:webSalesTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:webSalesTableName}${hiveconf:temporaryTableSuffix}
  ( 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              int
  , 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)
  , ws_net_profit             decimal(7,2)
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:webSalesTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:webSalesTableName};
DROP TABLE IF EXISTS ${hiveconf:webSalesTableName};
CREATE TABLE ${hiveconf:webSalesTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:webSalesTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:webSalesTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:webSalesTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:webReturnsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:webReturnsTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:webReturnsTableName}${hiveconf:temporaryTableSuffix}
  ( 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        int
  , 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 '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:webReturnsTableName}'
;
```

```
-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:webReturnsTableName};
DROP TABLE IF EXISTS ${hiveconf:webReturnsTableName};
CREATE TABLE ${hiveconf:webReturnsTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:webReturnsTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:webReturnsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:webReturnsTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:marketPricesTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:marketPricesTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:marketPricesTableName}${hiveconf:temporaryTableSuffix}
 ( imp_sk               bigint             --not null
 , imp_item_sk          bigint             --not null
 , imp_competitor       string
 , imp_competitor_price  decimal(7,2)
 , imp_start_date       bigint
 , imp_end_date         bigint

 )
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:marketPricesTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:marketPricesTableName};
DROP TABLE IF EXISTS ${hiveconf:marketPricesTableName};
CREATE TABLE ${hiveconf:marketPricesTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:marketPricesTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:marketPricesTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:marketPricesTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:clickstreamsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:clickstreamsTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:clickstreamsTableName}${hiveconf:temporaryTableSuffix}
(   wcs_click_date_sk      bigint
 , wcs_click_time_sk      bigint
 , wcs_sales_sk           bigint
 , wcs_item_sk            bigint
 , wcs_web_page_sk        bigint
 , wcs_user_sk            bigint
 )
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
 STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:clickstreamsTableName}'
```

```
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:clickstreamsTableName};
DROP TABLE IF EXISTS ${hiveconf:clickstreamsTableName};
CREATE TABLE ${hiveconf:clickstreamsTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:clickstreamsTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:clickstreamsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:clickstreamsTableName}${hiveconf:temporaryTableSuffix};


-- !echo Create temporary table: ${hiveconf:reviewsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE IF EXISTS ${hiveconf:reviewsTableName}${hiveconf:temporaryTableSuffix};
CREATE EXTERNAL TABLE ${hiveconf:reviewsTableName}${hiveconf:temporaryTableSuffix}
(   pr_review_sk           bigint             --not null
  , pr_review_date         string
  , pr_review_time         string
  , pr_review_rating       int                --not null
  , pr_item_sk             bigint             --not null
  , pr_user_sk             bigint
  , pr_order_sk            bigint
  , pr_review_content      string --not null
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '${hiveconf:fieldDelimiter}'
  STORED AS TEXTFILE LOCATION '${hiveconf:hdfsDataPath}/${hiveconf:reviewsTableName}'
;

-- !echo Load text data into ${hiveconf:tableFormat} table: ${hiveconf:reviewsTableName};
DROP TABLE IF EXISTS ${hiveconf:reviewsTableName};
CREATE TABLE ${hiveconf:reviewsTableName}
STORED AS ${hiveconf:tableFormat}
AS
SELECT * FROM ${hiveconf:reviewsTableName}${hiveconf:temporaryTableSuffix}
;

-- !echo Drop temporary table: ${hiveconf:reviewsTableName}${hiveconf:temporaryTableSuffix};
DROP TABLE ${hiveconf:reviewsTableName}${hiveconf:temporaryTableSuffix};
```