TPC BENCHMARK ™ DS

Standard Specification

Version 2.6.0

September, 2017

Transaction Processing Performance Council (TPC)

www.tpc.org

info@tpc.org

Legal Notice

The TPC reserves all right, title, and interest to this document and associated source code as provided under U.S. and international laws, including without limitation all patent and trademark rights therein.

Permission to copy without fee all or part of this document is granted provided that the TPC copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the Transaction Processing Performance Council. To copy otherwise requires specific permission.

No Warranty

Trademarks

TPC Benchmark, TPC-DS and QphDS are trademarks of the Transaction Processing Performance Council.

Acknowledgments

Document Revision History

Date	Version	Description
08-28-2015	2.0.0	Mail ballot version
11-12-2015	2.1.0	Includes FogBugz entries 937, 991, 1002, 1033 1053, 1060, 1121, 1128, 1135, 1136
06-09-2016	2.2.0	Includes FogBugz entries 1571 1559 1539 1538 1537 1531 1502 1501 1480 1479 1474 1473 1472 1470 1393 1322 1263
08-05-2016	2.3.0	Includes FogBugz entries 1676, 1627, 1531, 1501 and 616
02-24-2017	2.4.0	Includes FogBugz entries 1728, 1697, 1696 and 1654
06-08-2017	2.5.0	Includes FogBugz entries 1756, 1894, 1909, 1912, 1980 and 1981
0926-2017	2.6.0	Includes FogBugz entries 1556, 2031, 2043, 1984, 2030, 2036, 2143, 2035, 20412037, 2039, 2040, 2045, 2046, 2047

■

TPC Membership
(as of June 2017)

Full Members

actian.	CISCO	DELL	DataCore SOFTWARE	FUJITSU
Hewlett Packard Enterprise	HITACHI	HUAWEI	IBM	inspur
(intel)	Lenovo	Microsoft	NUTANIX	ORACLE
Pivotal.	redhat	SAP	TERADATA Raising Intelligence	TTA Telecommunications Technology Association
vmware				

Associate Members

IDEAS INTERNATIONAL	CAICT 中国信息通信研究院	· U · C ·

Table of Contents

0   PREAMBLE

0.1   Introduction

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

This benchmark illustrates decision support systems that:

- Examine large volumes of data;
- Give answers to real-world business questions;
- Execute queries of various operational requirements and complexities (e.g., ad-hoc, reporting, iterative OLAP, data mining);
- Are characterized by high CPU and IO load;
- Are periodically synchronized with source OLTP databases through database maintenance functions.
- Run on "Big Data" solutions, such as RDBMS as well as Hadoop/Spark based systems.

A benchmark result measures query response time in single user mode, query throughput in multi user mode and data maintenance performance for a given hardware, operating system, and data processing system configuration under a controlled, complex, multi-user decision support workload.

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

0.2   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 benchmark tests be implemented with systems, products, technologies and pricing that:

a)   Are generally available to users;

b)   Are relevant to the market segment that the individual TPC benchmark models or represents (e.g., TPC-DS models and represents complex, high data volume, decision support environments);

c)   Would plausibly be implemented by a significant number of users in the market segment modeled or represented by the benchmark.

In keeping with these requirements, the TPC-DS database must be implemented using commercially available data processing software, and its queries must be executed via SQL interface.

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 or pricing (hereafter referred to as "implementations") whose primary purpose is performance optimization of TPC benchmark results without any corresponding applicability to real-world applications and environments. In other words, all "benchmark special" implementations, which improve benchmark results but not real-world performance or pricing, are prohibited.

A number of characteristics shall be evaluated in order 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: "Based on the available evidence, does the clear preponderance (the greater share or weight) of evidence indicate this implementation is a benchmark special?"

The following characteristics shall be used to judge whether a particular implementation is a benchmark special:

a) Is the implementation generally available, documented, and supported?
b) Does the implementation have significant restrictions on its use or applicability that limits its use beyond TPC benchmarks?
c) Is the implementation or part of the implementation poorly integrated into the larger product?
d) Does the implementation take special advantage of the limited nature of TPC benchmarks (e.g., query templates, query mix, concurrency and/or contention, etc.) in a manner that would not be generally applicable to the environment the benchmark represents?
e) 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.)
f) Does the implementation require uncommon sophistication on the part of the end-user, programmer, or system administrator?
g) Is the pricing unusual or non-customary for the vendor or unusual or non-customary compared to normal business practices? The following pricing practices are suspect:
   • Availability of a discount to a small subset of possible customers;
   • Discounts documented in an unusual or non-customary manner;
   • Discounts that exceed 25% on small quantities and 50% on large quantities;
   • Pricing featured as a close-out or one-time special;
   • Unusual or non-customary restrictions on transferability of product, warranty or maintenance on discounted items.
h) Is the implementation (including beta-release components) being purchased or used for applications in the market segment the benchmark represents? How many sites implemented it? How many end-users benefit from it? If the implementation is not currently being purchased or used, is there any evidence to indicate that it will be purchased or used by a significant number of end-user sites?

0.3 General Measurement Guidelines

TPC benchmark results are expected to be accurate representations of system performance. Therefore, there are specific 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 left to the discretion of the test sponsor.

When not described in the specification, the methodologies and approaches used must meet the following requirements:

a) The approach is an accepted engineering practice or standard;
b) The approach does not enhance the result;
c) Equipment used in measuring the results is calibrated according to established quality standards;
d) Fidelity and candor is maintained in reporting any anomalies in the results, even if not specified in the benchmark requirements.

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

0.4   Workload Independence

TPC-DS uses terminology and metrics which are similar to other benchmarks originated by the TPC and others. Such similarity in terminology does not in any way imply that TPC-DS results are comparable to other benchmarks. The only benchmark results comparable to TPC-DS are other TPC-DS results compliant with the same major revision of the benchmark specification and with the same scale factor.

While this benchmark offers a rich environment representative of many decision support systems, it does not reflect the entire range of decision support requirements. In addition, the extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-DS approximates the customer's application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon workload, specific application requirements, and systems design and implementation. As a result of these and other factors, relative system performance will vary. Therefore, TPC-DS should not be used as a substitute for a specific customer application benchmarking when critical capacity planning and/or product evaluation decisions are contemplated.

Benchmark sponsors are permitted to employ several possible system designs and a broad degree of implementation freedom within the constraints detailed in this specification. A full disclosure report (FDR) of the implementation details must be made available along with the reported results.

0.5   Associated Materials

In addition to this document, TPC-DS relies on material which is only available electronically. While not included in the printed version of the specification, this material is integral to the submission of a compliant TPC-DS benchmark result. Table 0-1 summarizes the electronic material related to the TPC-DS specification that is available for download from the TPC web site.

This material is maintained, versioned and revised independently of the specification itself. Refer to **Appendix F** to determine which version(s) of the electronic content are compliant with this revision of the specification.

Table 0-1 Electronically Available Specification Material

Content	File Name/Location	Usage	Additional Information
Data generator	dsdgen	Used to generate the data sets for the benchmark	Clause 3.4
Query generator	dsqgen	Used to generate the query sets for the benchmark	Clause 4.1.2
Query Templates	query_templates/	Used by **dsqgen** to generate executable query text	Clause 4.1.3
Query Template Variants	query_variants/	Used by dsqgen to generate alternative executable query text	Appendix C
Table definitions in ANSI SQL	tpcds.sql tpcds_source.sql	Sample implementation of the logical schema for the data warehouse.	Appendix A
Data	data_maintenance/	Sample implementation of the SQL	Clause 5.3

Content	File Name/Location	Usage	Additional Information
Maintenance Functions in ANSI SQL		needed for the Data Maintenance phase of the benchmark	
Answer Sets	answer_sets/	Used to verify the initial population of the data warehouse.	Clause 7.3
Reference Data Set	run dsdgen with –validate flag	Set of files for each scale factor to compare the correct data generation of base data, refresh data and **dsqgen** data	

0.5.1   The rules for pricing are included in the current revision of the TPC Pricing Specification located on the TPC website (http://www.tpc.org).

Comment: There is a non-binding How_To_Guide.doc guide electronically available. The purpose of this guide is to describe the most common tasks necessary to implement a TPC-DS benchmark. The target audience is individuals who want to install, populate, run and analyze the database, queries and data maintenance workloads for TPC-DS.

1 Business and Benchmark Model

1.1 Overview

TPC Benchmark™ DS (TPC-DS) contains benchmark components that can be used to asses a broad range of system topologies and implementation methodologies in a technically rigorous and directly comparable, vendor-neutral manner. In order to ease the learning curve for users and benchmark sponsors who are new to TPC-DS, the benchmark has been mapped to a typical business environment. This clause outlines the business modeling assumptions that were adopted during the development of the benchmark, and their impact on the benchmark environment.

TPC-DS models the decision support functions of a retail product supplier. The supporting schema contains vital business information, such as customer, order, and product data. The benchmark models the two most important components of any mature decision support system:

- User queries, which convert operational facts into business intelligence.
- Data maintenance, which synchronizes the process of management analysis with the operational external data source on which it relies.

The benchmark abstracts the diversity of operations found in an information analysis application, while retaining essential performance characteristics. As it is necessary to execute a great number of queries and data transformations to completely manage any business analysis environment, no benchmark can succeed in exactly mimicking a particular environment and remain broadly applicable.

While TPC-DS does not aspire to be a model of how to build an actual information analysis application, the workload has been granted a realistic context. It imitates the activity of a multi-channel retailer; thus tracking store, web and catalog sales channels.

The goal of selecting a retail business model is to assist the reader in relating intuitively to the components of the benchmark, without tracking that industry segment so tightly as to minimize the relevance of the benchmark. The TPC-DS workload may be used to characterize any industry that must transform operational and external data into business intelligence.

Although the emphasis is on information analysis, the benchmark recognizes the need to periodically refresh its data. The data represents a reasonable image of a business operation as they progress over time.

Some TPC benchmarks model the operational aspect of the business environment where transactions are executed on a real time basis. Other benchmarks address the simpler, more static model of decision support. The TPC-DS benchmark, models the challenges of business intelligence systems where operational data is used both to support the making of sound business decisions in near real time and to direct long-range planning and exploration.

Figure 1-1 illustrates TPC-DS benchmark components.
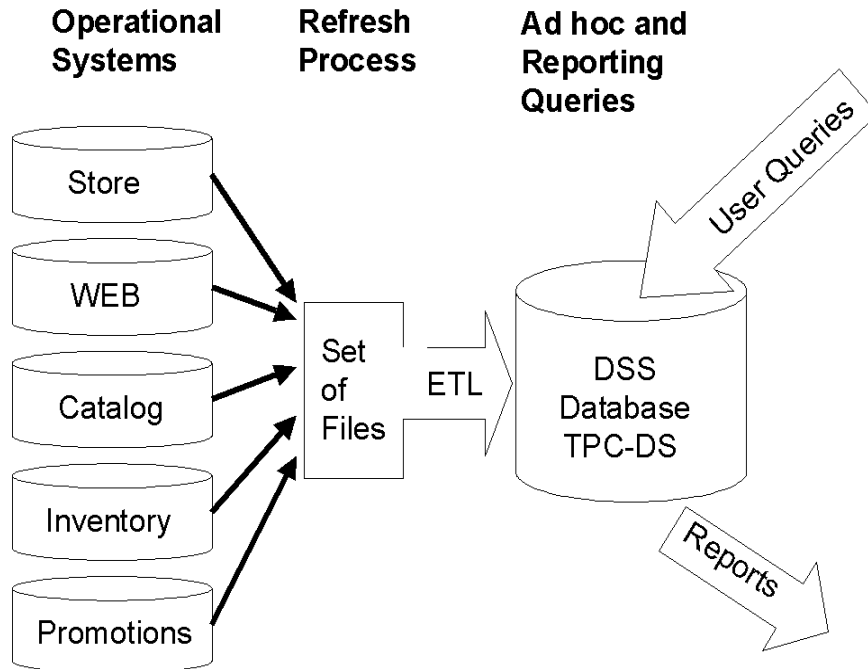
Figure 1-1: TPC-DS benchmark components

1.2   Business Model

TPC-DS models any industry that must manage, sell and distribute products (e.g., food, electronics, furniture, music and toys etc.). It utilizes the business model of a large retail company having multiple stores located nation-wide. Beyond its brick and mortar stores, the company also sells goods through catalogs and the Internet. Along with tables to model the associated sales and returns, it includes a simple inventory system and a promotion system.

The following are examples of business processes of this retail company:

- Record customer purchases (and track customer returns) from any sales channel
- Modify prices according to promotions
- Maintain warehouse inventory
- Create dynamic web pages
- Maintain customer profiles (Customer Relationship Management)

TPC-DS does not benchmark the operational systems. It is assumed that the channel sub-systems were designed at different times by diverse groups having dissimilar functional requirements. It is also recognized that they may be operating on significantly different hardware configurations, software configurations and data model semantics. All three channel sub-systems are autonomous and retain possibly redundant information regarding customers, addresses, etc. For more information in the benchmarking of operational system, please see the TPC website (http://www.tpc.org).

TPC-DS' modeling of the business environment falls into three broad categories:

- Data Model and Data Access Assumptions (see Clause 1.3)
- Query and User Model Assumptions (see Clause 1.4)

- Data Maintenance Assumptions(see Clause 1.5)

1.3 Data Model and Data Access Assumptions

1.3.1   TPC-DS models a system that allows potentially long running and multi-part queries where the DBA can assume that the data processing system is quiescent for queries during any particular period.

1.3.2   The TPC-DS data tracks, possibly with some delay, the state of an operational database through data maintenance functions, which include a number of modifications impacting some part of the decision support system.

1.3.3   The TPC-DS schema is a snowflake schema. It consists of multiple dimension and fact tables. Each dimension has a single column surrogate key. The fact tables join with dimensions using each dimension table's surrogate key. The dimension tables can be classified into one of the following types:

- **Static**:   The contents of the dimension are loaded once during database load and do not change over time. The date dimension is an example of a static dimension.
- **Historical**: The history of the changes made to the dimension data is maintained by creating multiple rows for a single business key value. Each row includes columns indicating the time period for which the row is valid. The fact tables are linked to the dimension values that were active at the time the fact was recorded, thus maintaining "historical truth". Item is an example of a historical dimension.
- **Non-Historical**: The history of the changes made to the dimension data is not maintained. As dimension rows are updated, the previous values are overwritten and this information is lost. All fact data is associated with the most current value of the dimension. Customer is an example of a Non-Historical dimension.

1.3.4   To achieve the optimal compromise between performance and operational consistency, the system administrator can set, once and for all, the locking levels and the concurrent scheduling rules for queries and data maintenance functions.

1.3.5   The size of a DSS system – more precisely the size of the data captured in a DSS system – may vary from company to company and within the same company based on different time frames. Therefore, the TPC-DS benchmark will model several different sizes of the DSS (a.k.a. benchmark scaling or scale factor).

1.4 Query and User Model Assumptions

The users and queries modeled by the benchmark exhibit the following characteristics:

a) They address complex business problems
b) They use a variety of access patterns, query phrasings, operators, and answer set constraints
c) They employ query parameters that change across query executions

In order to address the enormous range of query types and user behaviors encountered by a decision support system, TPC-DS utilizes a generalized query model. This model allows the benchmark to capture important aspects of the interactive, iterative nature of on-line analytical processing (OLAP) queries, the longer-running complex queries of data mining and knowledge discovery, and the more planned behavior of well known report queries.

1.4.1   Query Classes

The size of the schema and its three sales channels allow for amalgamating the above query classes, especially ad-hoc and reporting, into the same benchmark. An ad-hoc querying workload simulates an environment in which users connected to the system send individual queries that are not known in advance. The system's administrator (DBA) cannot optimize the system specifically for this set of queries. Consequently, execution time for those queries can be very long. In contrast, queries in a reporting workload are very well known in advance. As a result, the DBA can optimize the system specifically for these queries to execute them very rapidly by using clever data placement methods (e.g. partitioning and clustering) and auxiliary data structures (e.g. materialized views and indexes). Amalgamating both types of queries has been traditionally difficult in benchmark environments since per the definition of a benchmark all queries, apart from bind variables, are known in advance. TPC-DS accomplishes this fusion by dividing the schema into reporting and ad-hoc parts. The catalog sales channel is dedicated for the reporting part, while the store and web channels are dedicated for the ad-hoc part. The catalog sales channel was chosen as the reporting part because its data accounts for 40% of the entire data set. The reporting and ad-hoc parts of the schema differ in what kind of auxiliary data structures can be created.. The idea behind this approach is that the queries accessing the ad-hoc part constitute the ad-hoc query set while the queries accessing the reporting part are considered the reporting queries.

A sophisticated decision support system must support a diverse user population. While there are many ways to categorize those diverse users and the queries that they generate, TPC-DS has defined four broad classes of queries that characterize most decision support queries:

- Reporting queries
- Ad hoc queries
- Iterative OLAP queries
- Data mining queries

TPC-DS provides a wide variety of queries in the benchmark to emulate these diverse query classes.

1.4.1.1 Reporting Queries

These queries capture the "reporting" nature of a DSS system. They include queries that are executed periodically to answer well-known, pre-defined questions about the financial and operational health of a business. Although reporting queries tend to be static, minor changes are common. From one use of a given reporting query to the next, a user might choose to shift focus by varying a date range, geographic location or a brand name.

1.4.1.2 Ad hoc Queries

These queries capture the dynamic nature of a DSS system in which impromptu queries are constructed to answer immediate and specific business questions. The central difference between *ad hoc* queries and reporting queries is the limited degree of foreknowledge that is available to the System Administrator (SysAdmin) when planning for an *ad hoc* query.

1.4.1.3 Iterative OLAP Queries

OLAP queries allow for the exploration and analysis of business data to discover new and meaningful relationships and trends. While this class of queries is similar to the "Ad hoc Queries" class, it is distinguished by a scenario-based user session in which a sequence of queries is submitted. Such a sequence may include both complex and simple queries.

1.4.1.4 Data Mining Queries

Data mining is the process of sifting through large amounts of data to produce data content relationships. It can predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. This class of queries typically consists of joins and large aggregations that return large data result sets for possible extraction.

1.5   Data Maintenance Assumptions

A data warehouse is only as accurate and current as the operational data on which it is based. Accordingly, the migration of data from operational OLTP systems to analytical DSS systems is crucial. The migration tends to vary widely from business to business and application to application. Previous benchmarks evaluated the data analysis component of decision support systems while excluding a realistic data refresh process. TPC-DS offers a more balanced view.

Decision support database refresh processes usually involve three distinct and important steps:

- **Data Extraction:** This phase consists of the accurate extraction of pertinent data from production OLTP databases and other relevant data sources. In a production environment, the extraction step may include numerous separate extract operations executed against multiple OLTP databases and auxiliary data sources. While selection and tuning of the associated systems and procedures is important to the success of the production system, it is separate from the purchase and configuration of the decision support servers. Accordingly, the data extract step of the ETL process (E) is not modeled in the benchmark. The TPC-DS data maintenance process starts from generated flat files that are assumed to be the output of this external Extraction process.
- **Data Transformation:** This is when the extracted data is cleansed and massaged into a common format suitable for assimilation by the decision support database.
- **Data Load:** This is the actual insertion, modification and deletion of data within the decision support database.

Taken together, the progression of Extraction, Transformation and Load is more commonly known by its acronym, ETL. In TPC-DS, the modeling of Transformation and Load is known as Data Maintenance (DM) or Data Refresh. In this specification the two terms are used interchangeably.

The DM process of TPC-DS includes the following tasks that result from such a complex business environment as shown in Figure 1-2:

i) Load the refresh data set, which consists of new, deleted and changed data destined for the data warehouse in its operational format.
ii) Load refresh data set into the data warehouse applying data transformations, e.g.:
- Data denormalization (3rd Normal form to snowflake). During this step the source tables are mapped into the data warehouse by:
    - Direct source to target mapping. This type of mapping is the most common. It applies to tables in the data warehouse that have an equivalent table in the operational schema.
    - Multiple data warehouse source tables are joined and the result is mapped to one target table. This mapping translates the third normal form of the operational schema into the de-normalized form of the data warehouse.
    - One source table is mapped to multiple target tables. This mapping is the least common. It occurs if, for efficiency reason, the schema of the operational system is less normalized than the data warehouse schema.
- Syntactically cleanse data
- De-normalize
iii) Insert new fact records and delete fact records by date.

The structure and relationships between the flat files is provided in form of a table description and the ddl of the tables that represent the hypothetical operational database in Appendix A.
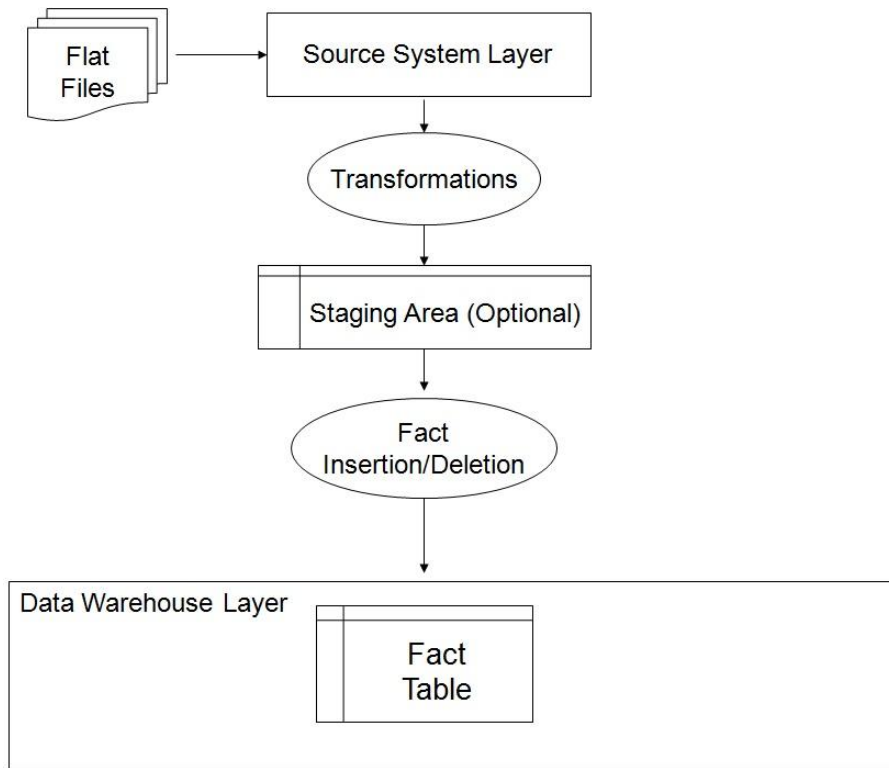
Figure 1-2: Execution Overview of the Data Maintenance Process

2   Logical Database Design

2.1   Schema Overview

The TPC-DS schema models the sales and sales returns process for an organization that employs three primary sales channels: stores, catalogs, and the Internet. The schema includes seven fact tables:

- A pair of fact tables focused on the product sales and returns for each of the three channels
- A single fact table that models inventory for the catalog and internet sales channels.

In addition, the schema includes 17 dimension tables that are associated with all sales channels. The following clauses specify the logical design of each table:

- The name of the table, along with its abbreviation (listed parenthetically)
- A logical diagram of each fact table and its related dimension tables
- The high-level definitions for each table and its relationship to other tables, using the format defined in Clause 2.2
- The scaling and cardinality information for each column

2.2   Column Definitions

2.2.1   Column Name

2.2.1.1   Each column is uniquely named, and each column name begins with the abbreviation of the table in which it appears.

2.2.1.2   Columns that are part of the table's primary key are indicated in the column called Primary Key (Sections 2.3 and 2.4). If a table uses a composite primary key, then for convenience of reading the order of a given column in a table's primary key is listed in parentheses following the column name.

2.2.1.3   Columns that are part of a business key are indicated with (B) appearing after the column name (Sections 2.3 and 2.4). A business key is neither a primary key nor a foreign key in the context of the data warehouse schema. It is only used to differentiate new data from update data of the source tables during the data maintenance operations.

2.2.2   Datatype

2.2.2.1   Each column employs one of the following datatypes:
   a) Identifier means that the column shall be able to hold any key value generated for that column.
   b) Integer means that the column shall be able to exactly represent integer values (i.e., values in increments of 1) in the range of at least (-2^{n-1}) to $(2^{n-1}-1)$, where n is 64.
   c) Decimal(d, f) means that the column shall be able to represent decimal values up to and including d digits, of which f shall occur to the right of the decimal place; the values can be either represented exactly or interpreted to be in this range.
   d) Char(N) means that the column shall be able to hold any string of characters of a fixed length of N.

   Comment: If the string that a column of datatype char(N) holds is shorter than N characters, then trailing spaces shall be stored in the database or the database shall automatically pad with spaces upon retrieval such that a CHAR_LENGTH() function will return N.

   e) Varchar(N) means that the column shall be able to hold any string of characters of a variable length with a maximum length of N. Columns defined as "varchar(N)" may optionally be implemented as "char(N)".
   f) Date means that the column shall be able to express any calendar day between January 1, 1900 and December 31, 2199.

2.2.2.2   The datatypes do not correspond to any specific SQL-standard datatype. The definitions are provided to highlight the properties that are required for a particular column. The benchmark implementer may employ any internal representation or SQL datatype that meets those requirements.

2.2.2.3   The implementation chosen by the test sponsor for a particular datatype definition shall be applied consistently to all the instances of that datatype definition in the schema, except for identifier columns, whose datatype may be selected to satisfy database scaling requirements.

2.2.3   NULLs

If a column definition includes an 'N' in the **NULLs** column this column is populated in every row of the table for all scale factors. If the field is blank this column may contain NULLs.

2.2.4   Foreign Key

If the values in this column join with another column, the foreign columns name is listed in the **Foreign Key** field of the column definition.

2.3   Fact Table Definitions

2.3.1   Store Sales (SS)

2.3.1.1   Store Sales ER-Diagram
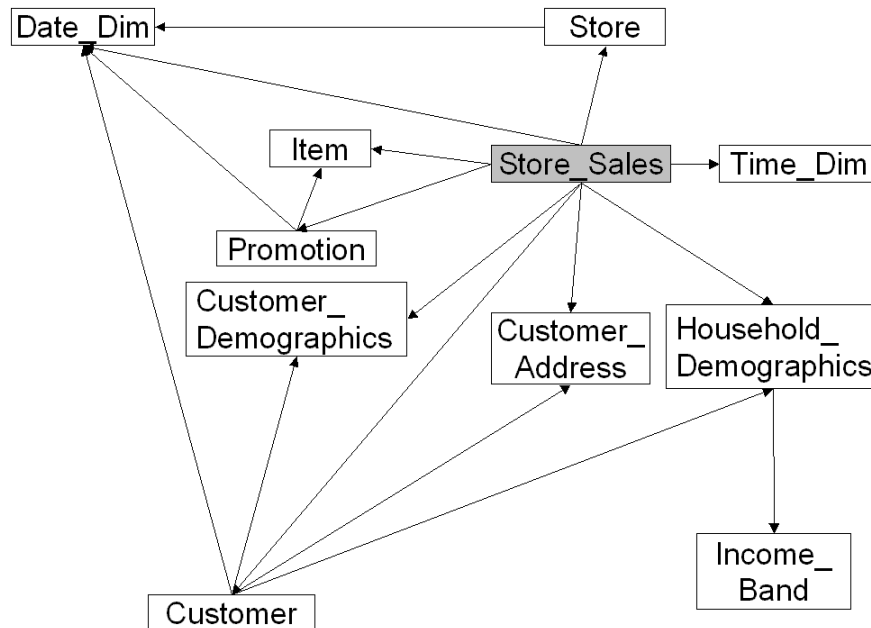
2.3.1.2   Store Sales Column Definitions

Each row in this table represents a single lineitem for a sale made through the store channel and recorded in the store_sales fact table.

Table 2-1 Store_sales Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
ss_sold_date_sk	identifier			d_date_sk
ss_sold_time_sk	identifier			t_time_sk
ss_item_sk (1)	identifier	N	Y	i_item_sk
ss_customer_sk	identifier			c_customer_sk
ss_cdemo_sk	identifier			cd_demo_sk
ss_hdemo_sk	identifier			hd_demo_sk
ss_addr_sk	identifier			ca_address_sk
ss_store_sk	identifier			s_store_sk
ss_promo_sk	identifier			p_promo_sk

Column	Datatype	NULLs	Primary Key	Foreign Key
ss_ticket_number (2)	identifier	N	Y	
ss_quantity	integer			
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)			

2.3.2 Store Returns (SR)

2.3.2.1 Store Returns ER-Diagram

2.3.2.2 Store Returns Column Definition

Each row in this table represents a single lineitem for the return of an item sold through the store channel and recorded in the store_returns fact table.

Table 2-2 Store_returns Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
sr_returned_date_sk	identifier			d_date_sk
sr_return_time_sk	identifier			t_time_sk
sr_item_sk (1)	identifier	N	Y	i_item_sk,ss_item_sk
sr_customer_sk	identifier			c_customer_sk
sr_cdemo_sk	identifier			cd_demo_sk
sr_hdemo_sk	identifier			hd_demo_sk
sr_addr_sk	identifier			ca_address_sk
sr_store_sk	identifier			s_store_sk
sr_reason_sk	identifier			r_reason_sk
sr_ticket_number (2)	identifier	N	Y	ss_ticket_number
sr_return_quantity	integer			
sr_return_amt	decimal(7,2)			
sr_return_tax	decimal(7,2)			

Column	Datatype	NULLs	Primary Key	Foreign Key
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)			

2.3.3 Catalog Sales (CS)

2.3.3.1 Catalog Sales ER-Diagram

2.3.3.2 Catalog Sales Column Definition

Each row in this table represents a single lineitem for a sale made through the catalog channel and recorded in the catalog_sales fact table.

Table 2-3 Catalog Sales Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
cs_sold_date_sk	identifier			d_date_sk
cs_sold_time_sk	identifier			t_time_sk
cs_ship_date_sk	identifier			d_date_sk
cs_bill_customer_sk	identifier			c_customer_sk
cs_bill_cdemo_sk	identifier			cd_demo_sk
cs_bill_hdemo_sk	identifier			hd_demo_sk
cs_bill_addr_sk	identifier			ca_address_sk
cs_ship_customer_sk	identifier			c_customer_sk
cs_ship_cdemo_sk	identifier			cd_demo_sk
cs_ship_hdemo_sk	identifier			hd_demo_sk
cs_ship_addr_sk	identifier			ca_address_sk
cs_call_center_sk	identifier			cc_call_center_sk
cs_catalog_page_sk	identifier			cp_catalog_page_sk
cs_ship_mode_sk	identifier			sm_ship_mode_sk
cs_warehouse_sk	identifier			w_warehouse_sk
cs_item_sk (1)	identifier	N	Y	i_item_sk
cs_promo_sk	identifier			p_promo_sk
cs_order_number (2)	identifier	N	Y	
cs_quantity	integer			
cs_wholesale_cost	decimal(7,2)			
cs_list_price	decimal(7,2)			
cs_sales_price	decimal(7,2)			

Column	Datatype	NULLs	Primary Key	Foreign Key
cs_ext_discount_amt	decimal(7,2)			
cs_ext_sales_price	decimal(7,2)			
cs_ext_wholesale_cost	decimal(7,2)			
cs_ext_list_price	decimal(7,2)			
cs_ext_tax	decimal(7,2)			
cs_coupon_amt	decimal(7,2)			
cs_ext_ship_cost	decimal(7,2)			
cs_net_paid	decimal(7,2)			
cs_net_paid_inc_tax	decimal(7,2)			
cs_net_paid_inc_ship	decimal(7,2)			
cs_net_paid_inc_ship_tax	decimal(7,2)			
cs_net_profit	decimal(7,2)			

2.3.4 Catalog Returns (CR)

2.3.4.1 Catalog Returns ER-Diagram
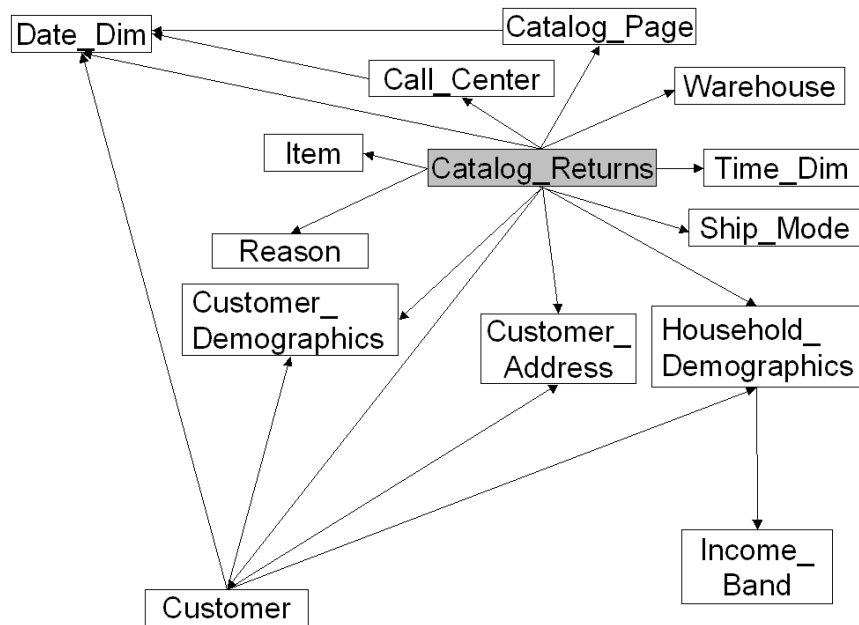
2.3.4.2 Catalog Returns Column Definition

Each row in this table represents a single lineitem for the return of an item sold through the catalog channel and recorded in the catalog_returns table.

Table 2-4 Catalog_returns Column Definition

Colum	Datatype	NULLs	Primary Key	Foreign Key
cr_returned_date_sk	identifier			d_date_sk
cr_returned_time_sk	identifier			t_time_sk
cr_item_sk (1)	identifier	N	Y	i_item_sk,cs_item_sk
cr_refunded_customer_sk	identifier			c_customer_sk
cr_refunded_cdemo_sk	identifier			cd_demo_sk
cr_refunded_hdemo_sk	identifier			hd_demo_sk
cr_refunded_addr_sk	identifier			ca_address_sk
cr_returning_customer_sk	identifier			c_customer_sk
cr_returning_cdemo_sk	identifier			cd_demo_sk
cr_returning_hdemo_sk	identifier			hd_demo_sk
cr_returning_addr_sk	identifier			ca_address_sk
cr_call_center_sk	identifier			cc_call_center_sk
cr_catalog_page_sk	identifier			cp_catalog_page_sk

Colum	Datatype	NULLs	Primary Key	Foreign Key
cr_ship_mode_sk	identifier			sm_ship_mode_sk
cr_warehouse_sk	identifier			w_warehouse_sk
cr_reason_sk	identifier			r_reason_sk
cr_order_number (2)	identifier	N	Y	cs_order_number
cr_return_quantity	integer			
cr_return_amount	decimal(7,2)			
cr_return_tax	decimal(7,2)			
cr_return_amt_inc_tax	decimal(7,2)			
cr_fee	decimal(7,2)			
cr_return_ship_cost	decimal(7,2)			
cr_refunded_cash	decimal(7,2)			
cr_reversed_charge	decimal(7,2)			
cr_store_credit	decimal(7,2)			
cr_net_loss	decimal(7,2)			

2.3.5 Web Sales (WS)

2.3.5.1 Web Sales ER-Diagram
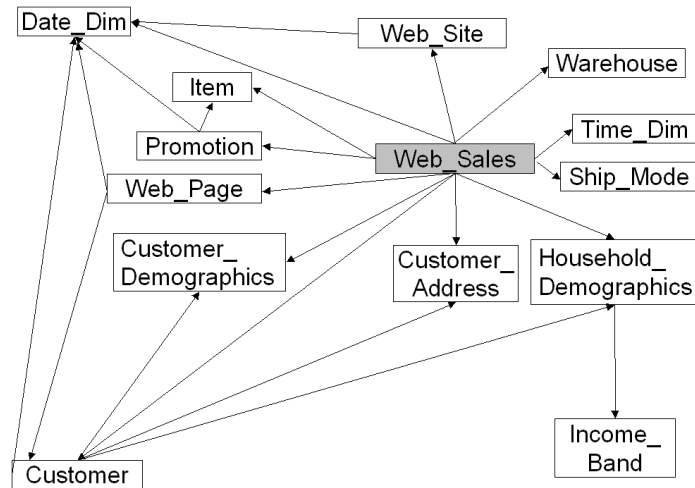
2.3.5.2 Web Sales Column Definition

Each row in this table represents a single lineitem for a sale made through the web channel and recorded in the web_sales fact table.

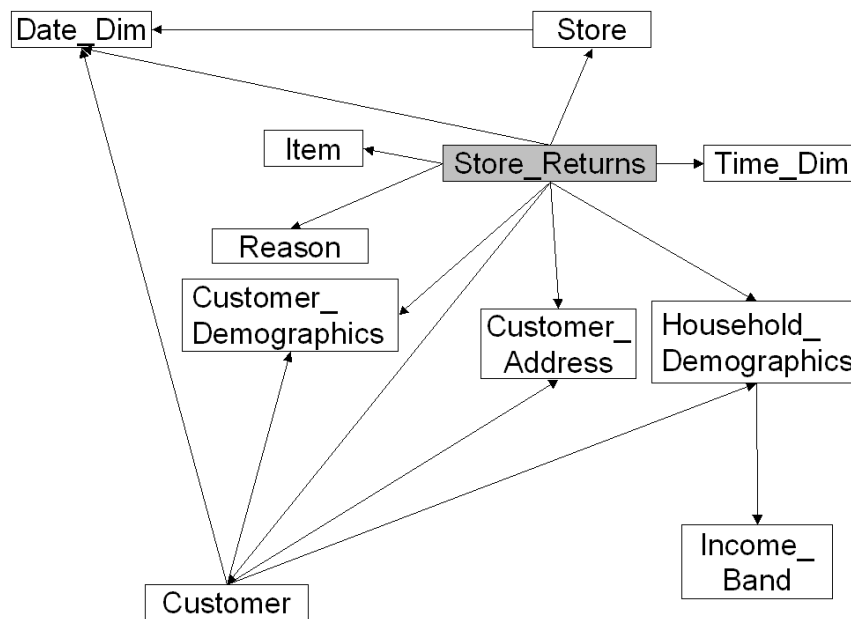**Table 2-5 Web_sales Column Definitions**

Column	Datatype	NULLs	Primary Key	Foreign Key
ws_sold_date_sk	identifier			d_date_sk
ws_sold_time_sk	identifier			t_time_sk
ws_ship_date_sk	identifier			d_date_sk
ws_item_sk (1)	identifier	N	Y	i_item_sk
ws_bill_customer_sk	identifier			c_customer_sk
ws_bill_cdemo_sk	identifier			cd_demo_sk
ws_bill_hdemo_sk	identifier			hd_demo_sk
ws_bill_addr_sk	identifier			ca_address_sk
ws_ship_customer_sk	identifier			c_customer_sk
ws_ship_cdemo_sk	identifier			cd_demo_sk
ws_ship_hdemo_sk	identifier			hd_demo_sk
ws_ship_addr_sk	identifier			ca_address_sk
ws_web_page_sk	identifier			wp_web_page_sk
ws_web_site_sk	identifier			web_site_sk
ws_ship_mode_sk	identifier			sm_ship_mode_sk
ws_warehouse_sk	identifier			w_warehouse_sk
ws_promo_sk	identifier			p_promo_sk

Column	Datatype	NULLs	Primary Key	Foreign Key
ws_order_number (2)	identifier	N	Y	
ws_quantity	integer			
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)			

2.3.6 Web Returns (WR)

2.3.6.1 Web Returns ER-Diagram
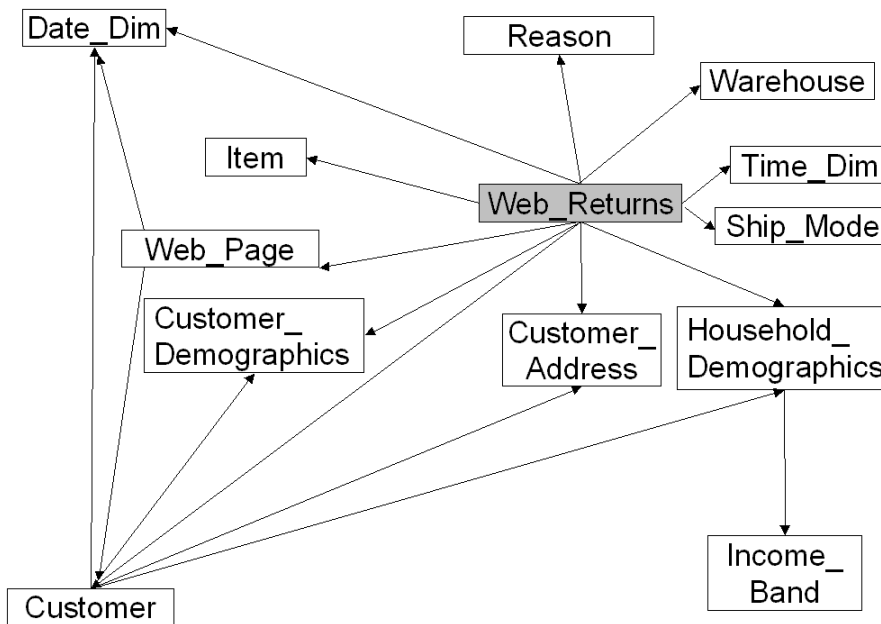
2.3.6.2 Web Returns Column Definition

Each row in this table represents a single lineitem for the return of an item sold through the web sales channel and recorded in the web_returns table.
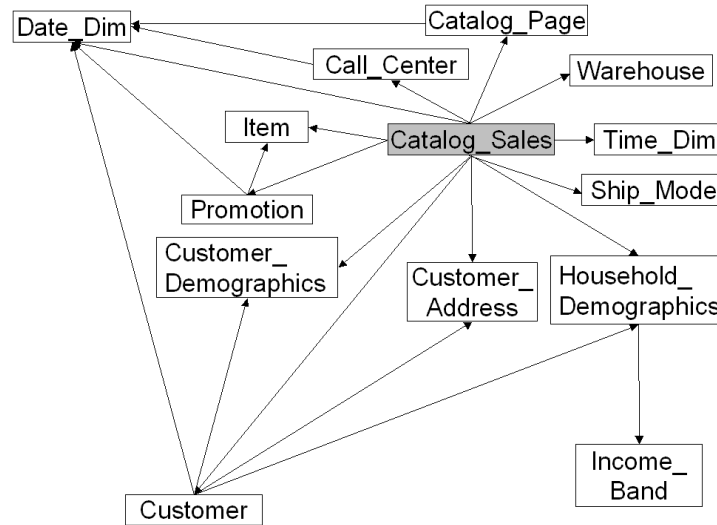
Table 2-6 Web_returns Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
wr_returned_date_sk	identifier			d_date_sk
wr_returned_time_sk	identifier			t_time_sk
wr_item_sk (2)	identifier	N	Y	i_item_sk,ws_item_sk
wr_refunded_customer_sk	identifier			c_customer_sk
wr_refunded_cdemo_sk	identifier			cd_demo_sk
wr_refunded_hdemo_sk	identifier			hd_demo_sk
wr_refunded_addr_sk	identifier			ca_address_sk
wr_returning_customer_sk	identifier			c_customer_sk
wr_returning_cdemo_sk	identifier			cd_demo_sk

Column	Datatype	NULLs	Primary Key	Foreign Key
wr_returning_hdemo_sk	identifier			hd_demo_sk
wr_returning_addr_sk	identifier			ca_address_sk
wr_web_page_sk	identifier			wp_web_page_sk
wr_reason_sk	identifier			r_reason_sk
wr_order_number (1)	identifier	N	Y	ws_order_number
wr_return_quantity	integer			
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)			

2.3.7 Inventory (INV)

2.3.7.1 Inventory ER-Diagram
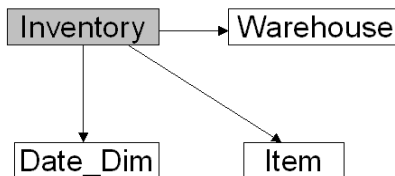
2.3.7.2 Inventory Column Definition

Each row in this table represents the quantity of a particular item on-hand at a given warehouse during a specific week.

Table 2-7 Inventory Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
inv_date_sk (1)	identifier	N	Y	d_date_sk
inv_item_sk (2)	identifier	N	Y	i_item_sk
inv_warehouse_sk (3)	identifier	N	Y	w_warehouse_sk
inv_quantity_on_hand	integer			

2.4 Dimension Table Definitions

2.4.1 Store (S)

Each row in this dimension table represents details of a store.

Table 2-8: Store Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
s_store_sk	identifier	N	Y	
s_store_id (B)	char(16)	N		
s_rec_start_date	date			
s_rec_end_date	date			
s_closed_date_sk	identifier			d_date_sk
s_store_name	varchar(50)			

Column	Datatype	NULLs	Primary Key	Foreign Key
s_number_employees	integer			
s_floor_space	integer			
s_hours	char(20)			
S_manager	varchar(40)			
S_market_id	integer			
S_geography_class	varchar(100)			
S_market_desc	varchar(100)			
s_market_manager	varchar(40)			
s_division_id	integer			
s_division_name	varchar(50)			
s_company_id	integer			
s_company_name	varchar(50)			
s_street_number	varchar(10)			
s_street_name	varchar(60)			
s_street_type	char(15)			
s_suite_number	char(10)			
s_city	varchar(60)			
s_county	varchar(30)			
s_state	char(2)			
s_zip	char(10)			
s_country	varchar(20)			
s_gmt_offset	decimal(5,2)			
s_tax_percentage	decimal(5,2)			

2.4.2 Call Center (CC)

Each row in this table represents details of a call center.

Table 2-9 Call_center Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
cc_call_center_sk	identifier	N	Y	
cc_call_center_id (B)	char(16)	N		
cc_rec_start_date	date			
cc_rec_end_date	date			
cc_closed_date_sk	identifier			d_date_sk
cc_open_date_sk	identifier			d_date_sk
cc_name	varchar(50)			
cc_class	varchar(50)			
cc_employees	integer			
cc_sq_ft	integer			
cc_hours	char(20)			
cc_manager	varchar(40)			
cc_mkt_id	integer			
cc_mkt_class	char(50)			
cc_mkt_desc	varchar(100)			
cc_market_manager	varchar(40)			
cc_division	integer			
cc_division_name	varchar(50)			
cc_company	integer			
cc_company_name	char(50)			
cc_street_number	char(10)			
cc_street_name	varchar(60)			
cc_street_type	char(15)			
cc_suite_number	char(10)			
cc_city	varchar(60)			
cc_county	varchar(30)			
cc_state	char(2)			
cc_zip	char(10)			
cc_country	varchar(20)			
cc_gmt_offset	decimal(5,2)			
cc_tax_percentage	decimal(5,2)			

2.4.3 Catalog_page (CP)

Each row in this table represents details of a catalog page.

Table 2-10 Catalog Page Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
cp_catalog_page_sk	identifier	N	Y	
cp_catalog_page_id (B)	char(16)	N		
cp_start_date_sk	identifier			d_date_sk
cp_end_date_sk	identifier			d_date_sk
cp_department	varchar(50)			
cp_catalog_number	integer,			
cp_catalog_page_number	integer,			
cp_description	varchar(100)			
cp_type	varchar(100)			

2.4.4　　　　Web_site (WEB)

Each row in this table represents details of a web site.

Table 2-11 Web_site Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
web_site_sk	identifier	N	Y	
web_site_id (B)	char(16)	N		
web_rec_start_date	date			
web_rec_end_date	date			
web_name	varchar(50)			
web_open_date_sk	identifier			d_date_sk
web_close_date_sk	identifier			d_date_sk
web_class	varchar(50)			
web_manager	varchar(40)			
web_mkt_id	integer			
web_mkt_class	varchar(50)			
web_mkt_desc	varchar(100)			
web_market_manager	varchar(40)			
web_company_id	integer			
web_company_name	char(50)			
web_street_number	char(10)			
web_street_name	varchar(60)			
web_street_type	char(15)			
web_suite_number	char(10)			
web_city	varchar(60)			
web_county	varchar(30)			
web_state	char(2)			
web_zip	char(10)			
web_country	varchar(20)			
web_gmt_offset	decimal(5,2)			
web_tax_percentage	decimal(5,2)			

2.4.5　　　　Web_page (WP)

Each row in this table represents details of a web page within a web site.

Table 2-12 Web_page Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
wp_web_page_sk	identifier	N	Y	
wp_web_page_id (B)	char(16)	N		
wp_rec_start_date	date			
wp_rec_end_date	date			
wp_creation_date_sk	identifier			d_date_sk
wp_access_date_sk	identifier			d_date_sk
wp_autogen_flag	char(1)			
wp_customer_sk	identifier			c_customer_sk
wp_url	varchar(100)			
wp_type	char(50)			

Column	Datatype	NULLs	Primary Key	Foreign Key
wp_char_count	integer			
wp_link_count	integer			
wp_image_count	integer			
wp_max_ad_count	integer			

2.4.6 Warehouse (W)

Each row in this dimension table represents a warehouse where items are stocked.

Table 2-13 Warehouse Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
w_warehouse_sk	identifier	N	Y	
w_warehouse_id (B)	char(16)	N		
w_warehouse_name	varchar(20)			
w_warehouse_sq_ft	integer			
w_street_number	char(10)			
w_street_name	varchar(60)			
w_street_type	char(15)			
w_suite_number	char(10)			
w_city	varchar(60)			
w_county	varchar(30)			
w_state	char(2)			
w_zip	char(10)			
w_country	varchar(20)			
w_gmt_offset	decimal(5,2)			

2.4.7 Customer (C)

Each row in this dimension table represents a customer.

Table 2-14: Customer Table Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
c_customer_sk	identifier	N	Y	
c_customer_id (B)	char(16)	N		
c_current_cdemo_sk	identifier			cd_demo_sk
c_current_hdemo_sk	identifier			hd_demo_sk
c_current_addr_sk	identifier			ca_addres_sk
c_first_shipto_date_sk	identifier			d_date_sk
c_first_sales_date_sk	identifier			d_date_sk
c_salutation	char(10)			
c_first_name	char(20)			
c_last_name	char(30)			
c_preferred_cust_flag	char(1)			
c_birth_day	integer			
c_birth_month	integer			
c_birth_year	integer			
c_birth_country	varchar(20)			
c_login	char(13)			
c_email_address	char(50)			
c_last_review_date_sk	identifier			d_date_sk

2.4.8 Customer_address (CA)

Each row in this table represents a unique customer address (each customer can have more than one address)

Table 2-15 Customer_address Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
ca_address_sk	identifier	N	Y	
ca_address_id (B)	char(16)	N		
ca_street_number	char(10)			

Column	Datatype			
ca_street_name	varchar(60)			
ca_street_type	char(15)			
ca_suite_number	char(10)			
ca_city	varchar(60)			
ca_county	varchar(30)			
ca_state	char(2)			
ca_zip	char(10)			
ca_country	varchar(20)			
ca_gmt_offset	decimal(5,2)			
ca_location_type	char(20)			

2.4.9 Customer_demographics (CD)

The customer demographics table contains one row for each unique combination of customer demographic information.

Table 2-16 Customer_demographics Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
cd_demo_sk	identifier	N	Y	
cd_gender	char(1)			
cd_marital_status	char(1)			
cd_education_status	char(20)			
cd_purchase_estimate	integer			
cd_credit_rating	char(10)			
cd_dep_count	integer			
cd_dep_employed_count	integer			
cd_dep_college_count	integer			

2.4.10 Date_dim (D)

Each row in this table represents one calendar day. The surrogate key (d_date_sk) for a given row is derived from the julian date being described by the row.

Table 2-17 Date_dim Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
d_date_sk	identifier	N	Y	
d_date_id (B)	char(16)	N		
d_date	date			
d_month_seq	integer			
d_week_seq	integer			
d_quarter_seq	integer			
d_year	integer			
d_dow	integer			
d_moy	integer			
d_dom	integer			
d_qoy	integer			
d_fy_year	integer			
d_fy_quarter_seq	integer			
d_fy_week_seq	integer			
d_day_name	char(9)			
d_quarter_name	char(6)			
d_holiday	char(1)			
d_weekend	char(1)			
d_following_holiday	char(1)			
d_first_dom	integer			
d_last_dom	integer			
d_same_day_ly	integer			
d_same_day_lq	integer			
d_current_day	char(1)			
d_current_week	char(1)			
d_current_month	char(1)			
d_current_quarter	char(1)			
d_current_year	char(1)			

2.4.11 Household_demographics (HD)

Each row of this table defines a household demographic profile.

Table 2-18 Household_demographics Column Definition

Column	Datatype	NULLs	Primary Key	Foreign Key
hd_demo_sk	identifier	N	Y	
hd_income_band_sk	identifier			ib_income_band_sk
hd_buy_potential	char(15)			
hd_dep_count	integer			
hd_vehicle_count	integer			

2.4.12 Item (I)

Each row in this table represents a unique product formulation (e.g., size, color, manufactuer, etc.).

Table 2-19 Item Column Definition

Column	Datatype	NULLs	Primary Key	Foreign Key
i_item_sk	identifier	N	Y	
i_item_id (B)	char(16)	N		
i_rec_start_date	date			
i_rec_end_date	date			
i_item_desc	varchar(200)			
i_current_price	decimal(7,2)			
i_wholesale_cost	decimal(7,2)			
i_brand_id	integer			
i_brand	char(50)			
i_class_id	integer			
i_class	char(50)			
i_category_id	integer			
i_category	char(50)			
i_manufact_id	integer			
i_manufact	char(50)			
i_size	char(20)			
i_formulation	char(20)			
i_color	char(20)			
i_units	char(10)			
i_container	char(10)			
i_manager_id	integer			
i_product_name	char(50)			

2.4.13 Income_band (IB)

Each row in this table represents details of an income range.

Table 2-20: Income_band Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
ib_income_band_sk	identifier	N	Y	
ib_lower_bound	integer			
ib_upper_bound	integer			

2.4.14 Promotion (P)

Each row in this table represents details of a specific product promotion (e.g., advertising, sales, PR).

Table 2-21: Promotion Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
p_promo_sk	identifier	N	Y	
p_promo_id (B)	char(16)	N		
p_start_date_sk	identifier			d_date_sk
p_end_date_sk	identifier			d_date_sk
p_item_sk	identifier			i_item_sk

Column	Datatype	NULLs	Primary Key	Foreign Key
p_cost	decimal(15,2)			
p_response_target	integer			
p_promo_name	char(50)			
p_channel_dmail	char(1)			
p_channel_email	char(1)			
p_channel_catalog	char(1)			
p_channel_tv	char(1)			
p_channel_radio	char(1)			
p_channel_press	char(1)			
p_channel_event	char(1)			
p_channel_demo	char(1)			
p_channel_details	varchar(100)			
p_purpose	char(15)			
p_discount_active	char(1)			

2.4.15 Reason (R)

Each row in this table represents a reason why an item was returned.

Table 2-22: Reason Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
r_reason_sk	identifier	N	Y	
r_reason_id (B)	char(16)	N		
r_reason_desc	char(100)			

2.4.16 Ship_mode (SM)

Each row in this table represents a shipping mode.

Table 2-23: Ship_mode Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
sm_ship_mode_sk	identifier	N	Y	
sm_ship_mode_id (B)	char(16)	N		
sm_type	char(30)			
sm_code	char(10)			
sm_carrier	char(20)			
sm_contract	char(20)			

2.4.17 Time_dim (T)

Each row in this table represents one second.

Table 2-24: Time_dim Column Definitions

Column	Datatype	NULLs	Primary Key	Foreign Key
t_time_sk	Identifier	N	Y	
t_time_id (B)	char(16)	N		
t_time	Integer			
t_hour	Integer			
t_minute	Integer			
t_second	Integer			
t_am_pm	char(2)			
t_shift	char(20)			
t_sub_shift	char(20)			
t_meal_time	char(20)			

2.4.18   dsdgen_version

This table is not employed during the benchmark. A flat file is generated by dsdgen (see Appendix F), and it can be helpful in assuring that the current data set was built with the correct version of the TPC-DS toolkit. It is included here for completeness.

Table 2-25: dsdgen_version Column Definitions

Column	Datatype	NULLs	Foreign Key
dv_version	Varchar(16)	N	
dv_create_date	date	N	
dv_create_time	time	N	
dv_cmdline_args	Varchar(200)	N	

2.5   Implementation Requirements

2.5.1   Definition of Terms

2.5.1.1   The tables defined in Clause 2.3 and Clause 2.4 are referred to as base tables. The flat file data generated by **dsdgen** corresponding to each base table and loaded into each base table is referred to as base table data. A structure containing base table data is referred to as a base table data structure.

2.5.1.2   Other than the base table data structures, any database structure that contains a copy of, reference to, or data computed from base table data is defined as an auxiliary data structures (**ADS**). The data in the ADS is materialized from the base table data; references are a form of materialization. There is an essential distinction between base table data contained in a base table data structure and data contained in auxiliary data structures. Because auxiliary data structures contain *copies of*, *references to, or data computed from* base table data, deleting data from an auxiliary data structure does not result in the loss of base table data in that it is still contained in the base table data structure. In contrast, deleting data from a base table data structure (in the absence of copies in any auxiliary data structures) does result in the loss of base table data.

2.5.1.3   There are two types of auxiliary data structures: Implicit and explicit. An explicit auxiliary data structure (EADS) is created as a consequence of a directive (e.g. DDL, session options, global configuration parameters). These directives are called EADS Directives. Any ADS which is not an EADS is by definition an Implict ADS (IADS).

           Comment: In contrast to an implicit **ADS**, an **EADS** would not have been created without the directive.

2.5.1.4   The assignment of groups of rows from a table or EADS to different files, disks, or areas is defined as horizontal partitioning.

2.5.1.5   The assignment of groups of columns of one or more rows to files, disks, or areas different from those storing the other columns of these rows is defined as vertical partitioning.

2.5.1.6   A Primary Key is one or more columns that uniquely identifies a row. None of the columns that are part of the Primary Key may be nullable. A table must have no more than one Primary Key. A primary key may be enforced, e.g. by a primary key constraint.

2.5.1.7   A Foreign Key is a column or combination of columns used to establish a link between the data in two tables. A link is created between two tables by adding the column or columns that hold one table's Primary Key values to the other table. This column becomes a Foreign Key in the second table. A foreign key may be enforced, e.g. by a foreign key constraint.Referential Integrity is a data property whereby a Foreign Key in one table has a corresponding Primary key in a different table.

2.5.1.8   The definition of primary and foreign keys is optional.

2.5.1.9   Whenever this specification refers to a set of primary and foreign keys it refers to the set of primary and foreign keys defined in clauses 2.3 and 2.4.

2.5.2   Data Processing System & Tables

2.5.2.1   The data processing system shall be implemented using a generally available and supported system (DBMS).

2.5.2.2   The SQL data definition statements and associated scripts used to implement the logical schema definition are defined as the **DDL**.

2.5.2.3   The database which is built and utilized to run the **Query Validation** test is defined as the qualification database.

2.5.2.4   The database which is built and utilized for performance reporting is defined as the test database.

2.5.2.5   The physical clustering of records of different tables within the database is allowed as long as this clustering does not alter the logical relationships of each table.

   Comment: The intent of this clause is to permit flexibility in the physical layout of a database and based upon the defined TPC-DS schema.

2.5.2.6   Table names should match those provided in Clause 2.3 and Clause 2.4. If the data processing system prevents the use of the table names specified in Clause 2.3 and Clause 2.4, they may be altered provided that:
- The name changes are minimal (e.g., short prefix or suffix.)
- The name changes have no performance impact
- The name changes are also made to the query set, in compliance with Clause 4.2.3

2.5.2.7   Each table listed in Clause 2.3 and Clause 2.4, shall be implemented according to the column definitions provided above.

2.5.2.8   The column names used in the benchmark implementation shall match those defined for each column specified in Clause 2.3 and Clause 2.4. If the data processing system prevents the use of the column names specified in Clause 2.3 and Clause 2.4, they may be altered provided:
- The name changes are the minimal changes required (e.g., short prefix or suffix or character substitution.)
- The changed names are required to follow the documented naming convention employed in the system used for the benchmark implementation
- The names used must provide no performance benefit compared to any other names that might be chosen.
- The identical name changes must also be made to the query set, in compliance with Clause 4.2.3

2.5.2.9   The columns within a given table may be implemented in any order, but all columns listed in the table definition shall be implemented and there shall be no columns added to the tables.

2.5.2.10   Each table column defined in Clause 2.3 and Clause 2.4 shall be implemented as discrete columns and shall be independently accessible by the data processing system as a single column. Specifically:
- Columns shall not be merged. For example, C_LOGIN and C_EMAIL_ADDRESS cannot be implemented as two sub-parts of a single discrete column C_DATA.
- Columns shall not be split. For example, P_TYPE cannot be implemented as two discrete columns P_TYPE_SUBSTR1 and P_TYPE_SUBSTR2.

2.5.2.11   The database shall allow for insertion of arbitrary data values that conform to the column's datatype and optional constraints defined in accordance with Clause 2.5.4.

2.5.3   **Explicit Auxiliary Data Structures (EADS)**

2.5.3.1   Except as provided in this section, replication of database objects (i.e., tables, rows or columns) is prohibited.

2.5.3.2   An EADS which does not include data materialized from Catalog_Sales or Catalog_Returns is subject to the following limitations:
- It may materialize data from no more than one **base table**.
- It may materialize all or some of the following three items:
  1. The primary key or any subset of PK columns if the PK is a compound key
  2. Pointers or references to corresponding base table rows (e.g., "row IDs").
  3. At most one of the following:
     a) A foreign key or any subset of the FK columns if the FK is a compound key
     b) A column having a date data type
     c) A column that is a business key

2.5.3.3   An EADS which includes data materialized from Catalog_Sales or Catalog_Returns may not include any data materialized from Store_Sales, Store_Returns, Web_Sales, Web_Returns or Inventory.

2.5.3.4   An EADS which materializes data from both fact and dimension tables must be the result of joining on FK – PK related columns.

2.5.3.5   An EADS which materializes data from one or more dimension tables without simultaneously materializing data from Catalog_Sales and/or Catalog_Returns is disallowed, unless otherwise permitted by Clause 2.5.3.2. An EADS which materializes data from one or more dimension tables must materialize at least one dimension row for every fact table row, unless the foreign key value for a dimension row is null.

Comment: The intent is to prohibit the creation of EADS on only dimension tables, except as allowed by clause 2.5.3.3.

2.5.3.6   The benchmark implementation of EADS may involve replication of selected data from the base tables provided that:
- All replicated data are managed by the system used for the benchmark implementation
- All replications are transparent to all data manipulation operations

2.5.3.7   The creation of all EADS must be included in the database load test (see Clause 7.4.3). EADS may not be created or deleted during the performance test.

2.5.3.8   Partitioning

2.5.3.8.1   A logical table space is a named collection of physical storage devices referenced as a single, logically contiguous, non-divisible entity.

2.5.3.8.2   The **DDL** may **i**nclude syntax that directs a table in its entirety to be stored in a particular logical table space.

2.5.3.8.3   Horizontal partitioning of **base tables** or **EADS** is allowed. If the partitioning is a function of data in the table or auxiliary data structure, the assignment shall be based on the values in the partitioning column(s). Only primary keys, foreign keys, date columns and date surrogate keys may be used as partitioning columns. If partitioning DDL uses directives that specify explicit partition values for the partitioning columns, they shall satisfy the following conditions:
- They may not rely on any knowledge of the data stored in the partitioning column(s) except the minimum and maximum values for those columns, and the definition of data types for those columns provided in Clause 2.
- Within the limitations of integer division, they shall define each partition to accept an equal portion of the range between the minimum and maximum values of the partitioning column(s).
- For date-based partitions, it is permissible to partition into equally sized domains based upon an integer granularity of days, weeks, months, or years; all using the Gregorian calendar (e.g., 30 days, 4 weeks, 1 month, 1 year, etc.). For date-based partition granularities other than days, a partition boundary may extend beyond the minimum or maximum boundaries as established in that table's data characteristics as defined in Clause 3.4
- The directives shall allow the insertion of values of the partitioning column(s) outside the range covered by the minimum and maximum values, as required by Clause 1.5.

If any directives or DDL are used to horizontally partition data, the directives, DDL, and other details necessary to replicate the partitioning behavior shall be disclosed.

Multi-level partitioning of base tables or auxiliary data structures is allowed only if each level of partitioning satisfies the conditions stated above.

2.5.3.8.4   Vertical partitioning of **base tables** or **EADS** is allowed when meeting all of the following requirements:
- SQL DDL that explicitly partitions data vertically is prohibited.
- SQL DDL must not contain partitioning directives which influence the physical placement of data on durable media.

- The row must be logically presented as an atomic set of columns.

Comment: This implies that vertical partitioning which does not rely upon explicit partitioning directives is allowed. Explicit partitioning directives are those that assign groups of columns of one row to files, disks or areas different from those storing the other columns in that row.

2.5.4 Constraints

2.5.4.1 The use of both enforced and unenforced constraints is permitted. If constraints are used, they shall satisfy the following requirements:
- Enforced constraints shall be enforced either at the statement level or at the transaction level
- Unenforced constraints must be validated after all data is loaded during the Load Test and before the start of the Performance Test
- They are limited to primary key, foreign key, and NOT NULL constraints
- NOT NULL constraints are allowed on EADSs and tables. Only columns that are marked 'N' in their logical table definition (or columns in EADSs derived from such columns) can be constrained with NOT NULL.

2.5.4.2 If foreign key constraints are defined and enforced, there is no specific requirement for a particular delete/update action when enforcing a constraint (e.g., ANSI SQL RESTRICT, CASCADE, NO ACTION, are all acceptable).

2.6 Data Access Transparency Requirements

2.6.1 Data Access Transparency is the property of the system that removes from the query text any knowledge of the physical location and access mechanisms of partitioned data. No finite series of tests can prove that the system supports complete data access transparency. The requirements below describe the minimum capabilities needed to establish that the system provides transparent data access. A benchmark implementation that uses horizontal partitioning shall meet the requirements for transparent data access described in Clauses 2.6.2 and 2.6.3.

Comment: The intent of this clause is to require that access to physically and/or logically partitioned data be provided directly and transparently by services implemented by generally available layers such as the interactive SQL interface, the data processing system, the operating system (OS), the hardware, or any combination of these.

2.6.2 Each of the tables described in Clause 2.3 and Clause 2.4 shall be identifiable by names that have no relationship to the partitioning of tables. All data manipulation operations in the executable query text (see Clause 3) shall use only these names.

2.6.3 Using the names which satisfy Clause 2.6.2, any arbitrary non-TPC-DS query shall be able to reference any set of rows or columns that is:

- Identifiable by any arbitrary condition supported by the underlying system
- Using the names described in Clause 2.6.2 and using the same data manipulation semantics and syntax for all tables

For example, the semantics and syntax used to query an arbitrary set of rows in any one table shall also be usable when querying another arbitrary set of rows in any other table.

Comment: The intent of this clause is that each TPC-DS query uses general purpose mechanisms to access data in the database.

3   Scaling and Database Population

This clause defines the database population and how it scales.

3.1   Scaling Model

3.1.1   The TPC-DS benchmark defines a set of discrete scaling points ("**scale factors**") based on the approximate size of the raw data produced by **dsdgen**. The actual byte count may vary depending on individual hardware and software platforms.

3.1.2   The set of scale factors defined for TPC-DS is:

- 1TB, 3TB, 10TB, 30TB, 100TB

where terabyte (TB) is defined to be 2^{40} bytes.

Comment: The maximum size of the test database for a valid performance test is currently set at 100TB. The TPC recognizes that additional benchmark development work is necessary to allow TPC-DS to scale beyond that limit.

3.1.3   Each defined scale factor has an associated value for **SF**, a unit-less quantity, roughly equivalent to the number of gigabytes of data present in the data warehouse. The relationship between scale factors and SF is summarized in Table 3-1 Scale Factor and SF.

Table 3-1 Scale Factor and SF

Scale Factor	SF
1TB	1000
3TB	3000
10TB	10000
30TB	30000
100TB	100000

3.1.4   Test sponsors may choose any scale factor from the defined series. No other scale factors may be used for a TPC-DS result.

3.1.5   Results at the different scale factors are not comparable, due to the substantially different computational challenges found at different data volumes.

3.2   Test Database Scaling

3.2.1   **Test database** is the database used to execute the database load test and the performance test (see Clause 7.4)

3.2.2   The required row count for each permissible scale factor and each table in the test database is detailed in Table 3-2 Database Row Counts.

Comment: The 1GB entries are used solely for the qualification database (see Clause 3.3.1) and are included here for ease of reference.

3.2.3   The row size information provided is an estimate, and may vary from one benchmark submission to another depending on the precise data base implementation that is selected. It is provided solely to assist benchmark sponsors in the sizing of benchmark configurations.

Table 3-2 Database Row Counts

Table	Avr Row Size in bytes	Sample Row Counts. Number of rows are within 1/100th Percent of these numbers					
		1GB	1TB	3TB	10TB	30TB	100TB
call_center	305	6	42	48	54	60	60
catalog_page	139	11,718	30,000	36,000	40,000	46,000	50,000
catalog_returns	166	144,067	143,996,756	432,018,033	1,440,033,112	4,319,925,093	14,400,175,879
catalog_sales	226	1,441,548	1,439,980,416	4,320,078,880	14,399,964,710	43,200,404,822	143,999,334,399
customer	132	100,000	12,000,000	30,000,000	65,000,000	80,000,000	100,000,000
customer_address	110	50,000	6,000,000	15,000,000	32,500,000	40,000,000	50,000,000
customer_demographics	42	1,920,800	1,920,800	1,920,800	1,920,800	1,920,800	1,920,800
date_dim	141	73,049	73,049	73,049	73,049	73,049	73,049
household_demographics	21	7,200	7,200	7,200	7,200	7,200	7,200
income_band	16	20	20	20	20	20	20
inventory	16	11,745,000	783,000,000	1,033,560,000	1,311,525,000	1,627,857,000	1,965,337,830
item	281	18,000	300,000	360,000	402,000	462,000	502,000
promotions	124	300	1,500	1,800	2,000	2,300	2,500
reason	38	35	65	67	70	72	75
ship_mode	56	20	20	20	20	20	20
store	263	12	1,002	1,350	1,500	1,704	1,902
store_returns	134	287,514	287,999,764	863,989,652	2,879,970,104	8,639,952,111	28,800,018,820
store_sales	164	2,880,404	2,879,987,999	8,639,936,081	28,799,983,563	86,399,341,874	287,997,818,084
time_dim	59	86,400	86,400	86,400	86,400	86,400	86,400
warehouse	117	5	20	22	25	27	30
web_page	96	60	3,000	3,600	4,002	4,602	5,004
web_returns	162	71,763	71,997,522	216,003,761	720,020,485	2,160,007,345	7,199,904,459
web_sales	226	719,384	720,000,376	2,159,968,881	7,199,963,324	21,600,036,511	71,999,670,164
web_site	292	30	54	66	78	84	96

3.3 Qualification Database Scaling

3.3.1 The **Qualification database** is the database used to execute the query validation test (see Clause 7.3)

3.3.2 The intent is that the functionality exercised by running the validation queries against the qualification database be the same as that exercised against the test database during the performance test. To this end, the qualification database must be identical to the test database in virtually every regard (except size), including but not limited to:

a) Column definitions
b) Method of data generation and loading (but not degree of parallelism)
c) Statistics gathering method
d) Data accessibility implementation
e) Type of partitioning (but not degree of partitioning)

f) Replication
g) Table type (if there is a choice)
h) EADS (e.g., indices)

3.3.3   The qualification database may differ from the test database only if the difference is directly related to the difference in sizes. For example, if the test database employs horizontal partitioning (see Clause 2.5.3.7), then the qualification database must also employ horizontal partitioning, though the number of partitions may differ in each case. As another example, the qualification database could be configured such that it uses a representative sub-set of the CPUs, memory and disks used by the test database configuration. If the qualification database configuration differs from the test database configuration in any way, the differences must be disclosed

3.3.4   The qualification database must be populated using **dsdgen**, and use a scale factor of 1GB.

3.3.5   The row counts of the qualification database are defined in Clause 3.2.

3.4   dsdgen and Database Population

3.4.1   The test database and the qualification database must be populated with data produced by **dsdgen**, the TPC-supplied data generator for TPC-DS. The major and minor version number of **dsdgen** must match that of the TPC-DS specification. The source code for **dsdgen** is provided as part of the electronically downloadable portion of this specification (see Appendix F).

3.4.2   The data generated by **dsdgen** are meant to be compliant with Table 3-2 and Table 5-2. In case of differences between the table and the data generated by **dsdgen**, Table 3-2 and Table 5-2 prevail.

3.4.3   Vendors are allowed to modify the **dsdgen** code for both the initial database population and the data maintenance. However, the resultant data must meet the following requirements in order to be considered correct:

a) The content of individual columns must be identical to that produced by **dsdgen**.
b) The data format of individual columns must be identical to that produced by **dsdgen**.
c) The number of rows generated for a given scale factor must be identical to that specified in Table 3-2 and Table 5-2.

If a modified version of dsdgen is used, the modified source code must be disclosed in full. In addition, the auditor must verify that the modified source code which is disclosed matches the data generation program used in the benchmark execution.

Comment: The intent of this clause is to allow for modification of the dsdgen code required for portability or speed, while precluding any change that affects the resulting data. Minor changes for portability or bugs are permitted in dsdgen for both initial database population and data maintenance.

3.4.4   If modifications are restricted to a subset of the source code, the vendor may publish only the individual dsdgen source code files which have been modified.

3.4.5   The output of dsdgen is text. The content of each field is terminated by '|'. A '|' in the first position of a row indicates that the first column of the row is empty. Two consecutive '|' indicate that the given column value is empty. Empty column values are only generated for columns that are NULL-able as specified in the logical database design. Empty column values, as generated by dsdgen, must be treated as NULL values in the data processing system, i.e. the data processing system must be able to retrieve NULL-able columns using 'is null' predicates.

Comment: The data generated by dsdgen includes some international characters. Examples of international characters are Ô and É. The database must preserve these characters during loading and processing by using a character encoding such as ISO/IEC 8859-1 that includes these characters.

3.5 Data Validation

The test database must be verified for correct data content. This must be done after the initial database load and prior to any performance tests. A validation data set is produced using **dsdgen** with the "-validate" and "-vcount" options. The minimum value for "-vcount" is 50, which produces 50 rows of validation data for most tables. The exceptions being the "returns" fact tables which will only have 5 rows each on average and the dimension tables with fewer than 50 total rows.

All rows produced in the validation data set must exist in the test database.

4   Query Overview

4.1   General Requirements and Definitions for Queries

4.1.1   Query Definition and Availability

4.1.1.1   Each query is described by the following components:
   a) A business question, which illustrates the business context in which the query could be used. The business questions are listed in Appendix B.
   b) The functional query definition, as specified in the TPC-supplied query template (see Clause 4.1.2 for a discussion of Functional Query Definitions)
   c) The substitution parameters, which describe the substitution values needed to generate the executable query text
   d) The answer set, which is used in query validation (see Clause 7.3)

Comment: Some functional query definitions include a limit on the number of rows to be returned by the query. These limits are omitted from the business question.

Comment: In cases where the business question does not accurately describe the functional query definition, the latter will prevail.

4.1.1.2   Due to the large size of the TPC-DS query set, this document does not contain all of the query components. Refer to Table 0-1 Electronically Available Specification Material for information on obtaining the query set.

4.1.2   Functional Query Definitions

4.1.2.1   The functionality of each query is defined by its query template and dsqgen.

4.1.3   **dsqgen** translates the query templates into fully functional SQL, which is known as executable query text (EQT). The major and minor version number of **dsqgen** must match that of the TPC-DS specification. The source code for **dsqgen** is provided as part of the electronically downloadable portion of this specification (see Table 0-1 Electronically Available Specification Material).

4.1.3.1   The query templates are primarily phrased in compliance with SQL1999 core (with OLAP amendments). A template includes the following, non-standard additions:
   • They are annotated, where necessary, to specify the number of rows to be returned
   • They include substitution tags that, in conjunction with **dsqgen**, allow a single template to generate a large number of syntactically distinct queries, which are functionally equivalent

4.1.3.2   The executable query text for each query in a compliant implementation must be taken from either the functional query definition or an approved query variant (see Clause Appendix C). Except as specifically allowed in Clauses 4.2.3, **Error! Reference source not found.** and 4.2.5, executable query text must be used in ull, exactly as provided by the TPC.

4.1.3.3   Any query template whose EQT does not match the functionality of the corresponding EQT produced by the TPC-supplied template is invalid.

4.1.3.4   All query templates and their substitution parameters shall be disclosed.

4.1.3.5   Benchmark sponsors are allowed to alter the precise phrasing of a query template to allow for minor differences in product functionality or query dialect as defined in Clause 4.2.3.

4.1.3.6   If the alterations allowed by Clause 4.2.3 are not sufficient to permit a benchmark sponsor to produce EQT that can be executed by the DBMS selected for their benchmark submission, they may submit an alternate query template for approval by the TPC (see Clause 4.2.3.4).

4.1.3.7   If the query template used in a benchmark submission is not identical to a template supplied by the TPC, it must satisfy the compliance requirements of Clauses 4.2.3, **Error! Reference source not found.** and 4.2.5.

4.2   Query Modification Methods

4.2.1   The queries must be expressed in a commercially available implementation of the SQL language. Since the ISO SQL language is continually evolving, the TPC-DS benchmark specification permits certain deviations from the SQL phrasing used in the TPC-supplied query templates.

4.2.2   There are four types of permissible deviations:

   a) Minor query modifications, defined in Clause 4.2.3
   b) Modifications to limit row counts, defined in clause 4.2.4
   c) Modifications for extraction queries, defined in clause 4.2.5
   d) Approved query variants, defined in Appendix C

4.2.3   Minor Query Modifications

4.2.3.1   It is recognized that implementations require specific adjustments for their operating environment and the syntactic variations of its dialect of the SQL language. The query modifications described in Clause 4.2.3.4:
   • Are defined to be minor
   • Do not require approval
   • May be used in conjunction with any other minor query modifications
   • May be used to modify either a functional query definition or an approved variant of that definition

Modifications that do not fall within the bounds described in Clause 4.2.3.4 are not minor and are not compliant unless they are an integral part of an approved query variant (see Appendix C).

Comment: The only exception is for the queries that require a given number of rows to be returned. The requirements governing this exception are given in Clause 4.2.4.1

4.2.3.2   The application of minor query modifications to functional query definitions or approved variants must be consistent over the query set. For example, if a particular vendor-specific date expression or table name syntax is used in one query, it must be used in all other queries involving date expressions or table names.

4.2.3.3   The use of minor modifications shall be disclosed and justified (see Clause 10.3.4.4).

4.2.3.4   The following query modifications are minor:

a) Tables:

   1. Table names - The table and view names found in the CREATE TABLE, CREATE VIEW, DROP VIEW and FROM clause of each query may be modified to reflect the customary naming conventions of the system under test.
   2. Tablespace references - CREATE TABLE statements may be augmented with a tablespace reference conforming to the requirements of Clause 3.
   3. WITH() clause - Queries using the "with()" syntax, also known as common table sub-expressions, can be replaced with semantically equivalent derived tables or views.

b) Joins:

   1. Outer Join - For outer join queries, vendor specific syntax may be used instead of the specified syntax. For example, the join expression "CUSTOMER LEFT OUTER JOIN ORDERS ON C_CUSTKEY = O_CUSTKEY" • may be replaced by adding CUSTOMER and ORDERS to the from clause and adding a specially-marked join predicate (e.g., C_CUSTKEY *= O_CUSTKEY).
   2. Inner Join - For inner join queries, vendor specific syntax may be used instead of the specified syntax. For example, the join expression "FROM CUSTOMER, ORDERS WHERE C_CUSTKEY = O_CUSTKEY" may be modified to use a JOIN clause such as "FROM CUSTOMER JOIN ORDERS ON C_CUSTKEY = O_CUSTKEY".

c) Operators:

1. Explicit ASC - ASC may be explicitly appended to columns in an ORDER BY clause.
2. Relational operators - Relational operators used in queries such as "<", ">", "<>", "<=", and "=", may be replaced by equivalent vendor-specific operators, for example ".LT.", ".GT.", "!=" or "^=", ".LE.", and "==", respectively.
3. String concatenation operator - For queries which use string concatenation operators, vendor specific syntax can be used (e.g. || can be substituted with +).
4. Rollup operator - an operator of the form "rollup (x,y)" may be substituted with the following operator: "x,y with rollup". x,y are expressions.

d) Control statements:

1. Command delimiters - Additional syntax may be inserted at the end of the executable query text for the purpose of signaling the end of the query and requesting its execution. Examples of such command delimiters are a semicolon or the word "GO".
2. Transaction control statements - A CREATE/DROP TABLE or CREATE/DROP VIEW statement may be followed by a COMMIT WORK statement or an equivalent vendor-specific transaction control statement.
3. Dependent views - If an implementation is using variants involving views and the implementation only supports "DROP RESTRICT" semantics (i.e., all dependent objects must be dropped first), then additional DROP statements for the dependent views may be added.

e) Alias:

1. Select-list expression aliases - For queries that include the definition of an alias for a SELECT-list item (e.g., "AS" clause), vendor-specific syntax may be used instead of the specified syntax. Examples of acceptable implementations include "TITLE <string>", or "WITH HEADING <string>". Use of a select-list expression alias is optional.
2. GROUP BY and ORDER BY - For queries that utilize a view, nested table-expression, or select-list alias solely for the purposes of grouping or ordering on an expression, vendors may replace the view, nested table-expression or select-list alias with a vendor-specific SQL extension to the GROUP BY or ORDER BY clause. Examples of acceptable implementations include "GROUP BY <ordinal>", "GROUP BY <expression>", "ORDER BY <ordinal>", and "ORDER BY <expression>".
3. Correlation names - Table-name aliases may be added to the executable query text. The keyword "AS" before the table-name alias may be omitted.
4. Nested table-expression aliasing - For queries involving nested table-expressions, the nested keyword "AS" before the table alias may be omitted.
5. Column alias - column name alias may be added for columns in any SELECT list of an executable query text. These column aliases may be used to refer to the column in later portions of the query, such as GROUP BY or ORDER BY clauses.

f) Expressions and functions:

1. Date expressions - For queries that include an expression involving manipulation of dates (e.g., adding/subtracting days/months/years, or extracting years from dates), vendor-specific syntax may be used instead of the specified syntax. Examples of acceptable implementations include "YEAR(<column>)" to extract the year from a date column or "DATE(<date>) + 3 MONTHS" to add 3 months to a date.
2. Output formatting functions - Scalar functions whose sole purpose is to affect output formatting (such as treatment of null strings) or intermediate arithmetic result precision (such as COALESCE or CAST) may be applied to items in the outermost SELECT list of the query.
3. Aggregate functions - At large scale factors, the aggregates may exceed the range of the values supported by an integer. The aggregate functions AVG and COUNT may be replaced with equivalent vendor-specific functions to handle the expanded range of values (e.g., AVG_BIG and COUNT_BIG).
4. Substring Scalar Functions - For queries which use the SUBSTRING() scalar function, vendor-specific syntax may be used instead of the specified syntax. For example, "SUBSTRING(S_ZIP, 1, 5)".

5. Standard Deviation Function - For queries which use the standard deviation function (stddev_samp), vendor specific syntax may be used (e.g. stdev, stddev).
6. Explicit Casting - Scalar functions (such as CAST) whose sole purpose is to affect result precision for operations involving integer columns or values may be applied. The resulting syntax must have equivalent semantic behavior.
7. Mathematical functions - Vendors specific mathematical expressions may be used to implement mathematical functions in the executable query text. The replacement syntax must implement the full semantic behavior (e.g. handling for NULLs) of the mathematical functions as defined in the ISO SQL standard. For example, avg() may be replaced by average() or by a mathematical expressions such as sum()/count().
8. Date casting - Explicit casting of columns that are of the date datatype, as defined in Clause 2.2.2, and date constant strings, expressed in month, day and year, into a datatype that allows for date arithmetic in expressions is permissible. Replacement syntax must have equivalent semantic behavior.
9. Casting syntax: - Vendor specific casting syntax may be used to implement casting functions present in the executable query text provided that the vendor specific casting syntax is semantically equivalent to the syntax provided in the executable query text.
10. Existing scalar functions - Existing scalar functions (such as CAST) in the query templates whose sole purpose is to affect output formatting or result precision may be modified. The resulting syntax must be consistent with the query template's original intended semantic behavior.

Comment: At higher scale factors some of the existing scalar functions might need adjustments to enable the benchmark to be run successfully at the intended scale factor. For example, to avoid numeric overflow at the intended scale factor, changing the CAST of a column from decimal(15, 4) to wider decimal(31, 4) is allowed."

g) General

1. Delimited identifiers - In cases where identifier names conflict with reserved words in a given implementation, delimited identifiers may be used.
2. Parentheses - Adding or removing parentheses around expressions and sub-queries is allowed. Both an opening parenthesis '(' and its corresponding closing parenthesis ')' must be added or removed together.
3. Ordinals - Ordinals can be exchanged with the referenced column name, or vice versa. E.g. "select a,b from T order by 2;" can be rewritten to "select a,b from T order by b;".

Comment: The application of all minor query modifications must result in queries that have equivalent ISO SQL semantic behavior as the queries generated from the TPC-supplied query templates.

Comment: All query modifications are labeled minor based on the assumption that they do not significantly impact the performance of the queries

4.2.4 Row Limit Modifications

4.2.4.1 Some queries require that a given number of rows be returned (e.g., "Return the first 10 selected rows"). If N is the number of rows to be returned, the query must return exactly the first N rows unless fewer than N rows qualify, in which case all rows must be returned. There are four permissible ways of satisfying this requirement:
- Vendor-specific control statements supported by a test sponsor's interactive SQL interface may be used (e.g., SET ROWCOUNT n) to limit the number of rows returned.
- Control statements recognized by the implementation specific layer (see Clause 8.2.4) and used to control a loop which fetches the rows may be used to limit the number of rows returned (e.g., while rowcount <= n).
- Vendor-specific SQL syntax may be added to the SELECT statement of a query template to limit the number of rows returned (e.g., SELECT FIRST n). This syntax is not classified as a minor query modification since it completes the functional requirements of the functional query definition and there is no standardized syntax defined. In all other respects, the query must satisfy the requirements of Clause 4.1.2. The syntax added must deal solely with the size of the answer set, and must not make any additional explicit reference, for example, to tables, indices, or access paths.

- Enclosing the outer most SQL statement (or statements in case of iterative OLAP queries) with a select clause and a row limiting predicate. For example, if Q is the original query text. Then the modification would be: SELECT * FROM (Q) where rownum<=n. This syntax is not classified as a minor query modification since it completes the functional requirements of the functional query definition and there is no standardized syntax defined. In all other respects, the query must satisfy the requirements of Clause 4.1.2. The syntax added must deal solely with the size of the answer set, and must not make any additional explicit reference, for example, to tables, indices, or access paths.

A test sponsor must select one of these methods and use it consistently for all the queries that require that a specified number of rows be returned.

4.2.5   Extract Query Modifications

4.2.5.1   Some queries return large result sets. These queries correspond to the queries described in Clause 1.4 as those that produce large result sets for extraction; the results are to be saved for later analysis. The benchmark allows for alternative methods for a DBMS to extract these result rows to files in addition to the normal method of processing them through a SQL front-end tool and using the front-end tool to output the rows to a file. If a query for any stream returns 10,000 or more result rows, the vendor may extract the rows for that query in all streams to files using one of the following permitted vendor-specific extraction tools or methods:

- Vendor-specific SQL syntax may be added to the SELECT statement of a query template to redirect the rows returned to a file. For example, "Unload to file 'outputfile' Select c1, c2 …"

- Vendor-specific control statements supported by a test sponsor's interactive SQL interface may be used. For example,

```
set output_file = 'outputfile'

select c1, c2…;

unset output_file;
```

- Control statements recognized by the implementation specific layer (see Clause 8.2.4) and used to invoke an extraction tool or method.

4.2.5.2   If one of these alternative extract options is used, the output shall be formatted as delimited or fixed-width ASCII text.

4.2.5.3   If one of these alternative extract options is used, they must meet the following conditions:

A test sponsor may select only one of the options in 4.2.5.1. That method must be used consistently for all the queries that are eligible as extract queries.

- If the extraction syntax modifies the query SQL, in all other respects the query must satisfy the requirements of Clause 4.1.2. The syntax added must deal solely with the extraction tool or method, and must not make any additional explicit reference, for example, to tables, indices, or access paths.
- The test sponsor must demonstrate that the file names used, and the extract facility itself, does not provide hints or optimizations in the DBMS such that the query has additional performance gains beyond any benefits from accelerating the extraction of rows.

The tool or method used must meet all ACID requirements for the queries used in combination with the tool or method.

4.2.6   Query Variants

4.2.6.1   A **Query Variant** is an alternate query template, which has been created to allow a vendor to overcome specific functional barriers or product deficiencies that could not be address by minor query modifications.

4.2.6.2   Approval of any new query variant is required prior to using such variant to produce compliant TPC-DS results. The approval process is defined Clause 4.2.7.

4.2.6.3　Query variants that have already been approved are summarized in Appendix C.

Comment: Since the soft appendix is updated each time a new variant is approved, test sponsors should obtain the latest version of this appendix prior to implementing the benchmark. See Appendix F **Tool Set Requirements** for more information)

4.2.7　Query Variant Approval

4.2.7.1　New query variants will be considered for approval if they meet one of the following criteria:
   a) The vendor requesting the variant cannot successfully run the executable query text against the qualification database using the functional query definition or an approved variant even after applying appropriate minor query modifications as per Clause 4.2.3.
   b) The proposed variant contains new or enhanced SQL syntax, relevant to the benchmark, which is defined in an Approved Committee Draft of a new ISO SQL standard.
   c) The variant contains syntax that brings the proposed variant closer to adherence to an ISO SQL standard.
   d) The proposed variant contains minor syntax differences that have a straightforward mapping to ISO SQL syntax used in the functional query definition and offers functionality substantially similar to the ISO SQL standard.

4.2.7.2　To be approved, a proposed variant should have the following properties. Not all of the properties are specifically required. Rather, the cumulative weight of each property satisfied by the proposed variant will be the determining factor in approving the variant.
   a) Variant is syntactic only, seeking functional compatibility and not performance gain.
   b) Variant is minimal and restricted to correcting a missing functionality.
   c) Variant is based on knowledge of the business question rather than on knowledge of the system under test (SUT) or knowledge of specific data values in the test database.
   d) Variant has broad applicability among different vendors.
   e) Variant is non procedural.
   f) Variant is an approved ISO SQL syntax to implement the functional query definition.
   g) Variant is sponsored by a vendor who can implement it and who intends on using it in an upcoming implementation of the benchmark.

4.2.7.3　To be approved, the proposed variant shall conform to the implementation guidelines defined in Clause 4.2.8 and the coding standards defined in Clause 4.2.9.

4.2.7.4　Approval of proposed query variants will be at the sole discretion of the TPC-DS subcommittee, subject to TPC policy.

4.2.7.5　All proposed query variants that are submitted for approval will be recorded, along with a rationale describing why they were or were not approved.

4.2.8　Variant Implementation Guidelines

4.2.8.1　When a proposed query variant includes the creation of a table, the datatypes shall conform to Clause 2.2.2.

4.2.8.2　When a proposed query variant includes the creation of a new entity (e.g., cursor, view, or table) the entity name shall ensure that newly created entities do not interfere with other query sessions and are not shared between multiple query sessions.

4.2.8.3　Any entity created within a proposed query variant must also be deleted within that variant.

4.2.8.4　If CREATE TABLE statements are used within a proposed query variant, they may include a tablespace reference (e.g., IN <tablespacename>). A single tablespace must be used for all tables created within a proposed query variant.

4.2.9　Coding Style

4.2.9.1　Implementers may code the executable query text in any desired coding style, including
   a) use of line breaks, tabs or white space
   b) choice of upper or lower case text

4.2.9.2   The coding style used shall have no impact on the performance of the system under test, and must be consistently applied throughout the entire query set.

Comment: The auditor may require proof that the coding style does not affect performance.

4.3 Substitution Parameter Generation

4.3.1 Each query has one or more substitution parameters. Dsqgen must be used to generate executable query texts for the query streams. In order to generate the required number of query streams, dsqgen must be used with the RNGSEED, INPUT and STREAMS options. The value for the RNGSEED option, <SEED>, is selected as the timestamp of the end of the database load time (Load End Time) expressed in the format mmddhhmmsss as defined in Clause 7.4.3.8. The value for the STREAMS option, <S>, is two times the number of streams, S_q, to be executed during each Throughput Test (S=2* S_q). The value of the INPUT option, <input.txt>, is a file containing the location of all 99 query templates in numerical order.

Comment: RNGSEED guarantees that the query substitution parameter values are not known prior to running the power and throughput tests. Called with a value of <S> for the STREAMS parameter, dsqgen generates S+1 files, named query_0.sql through query_[S].sql. Each file contains a different permutation of the 99 queries.

4.3.2 Query_0.sql is the sequence of queries to be executed during the Power Test; files query_1.sql through query_[S_q].sql are the sequences of queries to be executed during the first Throughput Test; and files query_[S_q+1].sql through query_[2*S_q].sql are the sequences of queries to be executed during the second Throughput Test.

Comment: The substitution parameter values for the qualification queries are provided in 11.9.1.1Appendix B:. They must be manually inserted into the query templates.

5   Data Maintenance

5.1   Implementation Requirements and Definitions

5.1.1   Data maintenance operations are performed as part of the benchmark execution. These operations consist of processing refresh runs.The total number of refresh runs in the benchmark equals the number of query streams in one Throughput Test. All data maintenance functions defined in Clause 5.3 are executed in each refresh run. Each refresh run has its own data set as generated by dsdgen and must be used in the order generated by dsdgen. Data maintenance operations execute separately from queries. Refresh runs do not overlap; at most one refresh run is running at any time.

5.1.2   Each refresh run includes all data maintenance functions defined in Clause 5.3 on the refresh data defined in Clause 5.2. All data maintenance functions need to have finished in refresh run n before any data maintenance function can commence in refresh run n+1 (see Clause 7.4.8.5).

5.1.3   Data maintenance functions can be decomposed or combined into any number of database operations and the execution order of the data maintenance functions can be freely chosen as long as the following conditions are met. Particularly, the functions in each refresh run may be run sequentially or in parallel.

   a) Data Accessibility properties (See Clause 6.1);
   b) All primary/foreign key relationships must be preserved regardless of whether they have been enforced by constraint (see Clause 2.5.4). This does not imply that referential integrity constraints must be defined explicitly.
   c) A time-stamped output message is sent when the data maintenance process is finished.

   Comment: The intent of this clause is to maintain primary and foreign key referential integrity.

   Comment: Implementers can assume that if all DM operations complete successfully that the PK/FK relationship is preserved. Any exceptions are bugs that need to be fixed in the spec.

5.1.4   All existing and enabled EADS affected by any data maintenance operation must be updated within those data maintenance operations. All updates performed by the refresh process must be visible to queries that start after the updates are completed.

5.1.5   The data maintenance functions must be implemented in SQL or procedural SQL. The proper implementation of the data maintenance function must be validated by the auditor who may request additional tests to ascertain that the data maintenance functions were implemented and executed in accordance with the benchmark requirements.

   Comment: Procedural SQL can be SQL embedded in other programs, interpreted or compiled.

5.1.6   The **staging area** is an optional collection of database objects (e.g. tables, indexes, views, etc.) used to implement the data maintenance functions. Database objects created in the staging area can only be used during execution of the data maintenance phase and cannot be used during any other phase of the benchmark. Any object created in the staging area needs to be disclosed in the FDR.

5.1.7   Any disk storage used for the staging area must be priced. Any mapping or virtualization of disk storage must be disclosed.

5.2   Refresh Data

5.2.1   The refresh data consists of a series of refresh data sets, numbered 1, 2, 3…n. <n> is identical to the number of streams used in the Throughput Tests of the benchmark. Each refresh data set consists of <N> flat files. The content of the flat files can be used to populate the source schema, defined in Appendix A. However, populating the source schema is not mandated. The flat files generated for each refresh data set and their corresponding source schema tables are denoted in the following table.

Table 5-1 Flat File to Source Schema Table Mapping and Flat File Size at Scale Factor 1

Flat File Name	Approximate Size at SF=1[1]		Source Schema Table Name
	Bytes	Number of rows	
s_catalog_order.dat	116505	682	s_catalog_order
s_catalog_order_lineitem.dat	592735	6138	s_catalog_order_lineitem
s_catalog_returns.dat	112182	578	s_catalog_returns
s_inventory.dat	26764259	540000	s_inventory
s_purchase.dat	142552	1022	s_purchase
s_purchase_lineitem.dat	1312480	12264	s_purchase_lineitem
s_store_returns.dat	159306	1235	s_store_returns
s_web_order.dat	43458	256	s_web_order
s_web_order_lineitem.dat	324160	3072	s_web_order_lineitem
s_web_returns.dat	42165	295	s_web_returns
inventory_delete	66	3	inventory_delete
delete	66	3	delete

---

[1] The number of rows are correct to within 0.001%. However, the number of bytes can vary from refresh set to refresh set due to NULL values.

Table 5-2 Approximate Number of rows in the update sets

Source Schema Table Name Flat File Name (with .dat extension)	Approximate Number of Rows[2] at Scale Factors:					
	1	1000	3000	10000	30000	100000
delete_1.dat	3	3	3	3	3	3
inventory_delete_1.dat	3	3	3	3	3	3
s_catalog_order_1.dat	682	681062	2043188	6810626	20431878	68106258
s_catalog_order_lineitem_1.dat	6138	6129558	18388692	61295634	183886902	612956322
s_catalog_returns_1.dat	595	612485	1838772	6128994	18382810	61291609
s_inventory_1.dat	270000	18000000	23760000	30150000	37422000	90360000
s_purchase_1.dat	1022	1021594	3064780	10215938	30647816	102159386
s_purchase_lineitem_1.dat	12264	12259128	36777360	122591256	367773792	1225912632
s_store_returns_1.dat	1200	1226054	3676450	12259852	36777217	122600683
s_web_order_1.dat	256	255398	766196	2553984	7661954	25539846
s_web_order_lineitem_1.dat	3072	3064776	9194352	30647808	91943448	306478152
s_web_returns_1.dat	320	306222	918594	3061569	9190618	30642220

Table 5-3 Approximate size of update data sets in bytes

Source Schema Table Name Flat File Name (with .dat extension)	Approximate Number of Bytes[3] at Scale Factors:					
	1	1000	3000	10000	30000	100000
s_catalog_order	116505	118319211	356209093	1189890226	3582266543	11966927381
s_catalog_order_lineitem	592735	613833353	1853028767	6200096417	18729687588	62665689954
s_catalog_returns	112182	120659364	363309171	1212531153	3648947224	12186641092
s_inventory	26764259	1784226065	2355173608	2988571049	3709394541	4478391128
s_purchase	142552	145457806	438594877	1464772384	41069025492	14749907338
s_purchase_lineitem	1312480	1347883261	4070341609	13601008735	41069025492	137232918564
s_store_returns	159306	165441528	501145568	1677710639	5088325652	17029150799
s_web_order	43458	44295571	133116152	445523894	1338776975	4480621920
s_web_order_lineitem	324160	332423806	999959825	3354924415	10091449245	33855757519
s_web_page	482	24016	28815	31982	36801	40013
s_web_returns	42165	44803099	134594520	450275312	1353093091	4533145920
inventory_delete	66	66	66	66	66	66
delete	66	66	66	66	66	66

5.2.2   The number of rows present in each refresh set at scale factor 1 for each of the flat files is summarized in Table 5-1.

5.2.3   The refresh data set of each data maintenance function must be generated using **dsdgen**. The execution of **dsdgen** is not timed. The output of **dsdgen** is a text file. The storage to hold the refresh data sets must be part of the priced configuration.

5.2.4   The refresh data set produced by dsdgen can be modified in the following way: The output file for each table of the refresh data set can be split into n files where each file contains approximately 1/n of the total number of rows of the original output file. The order of the rows in the original output file must be preserved, such that the concatenation of all n files is identical to the original file.

5.2.5   Reading the refresh data is a timed part of the data maintenance process. The data set for a specific refresh run must be loaded and timed as part of the execution of the refresh run. The loading of data must be performed via generic processes inherent to the data processing system or by the loader utility the database software provides and supports for general data loading. It is explicitly prohibited to use a loader tool that has been specifically developed for TPC-DS.

---

[2] The number of rows are correct to within 0.001%.

[3] The number of bytes can vary from refresh set to refresh set due to NULL values.

5.3   Data Maintenance Functions

5.3.1   Data maintenance functions perform insertand delete operations that are defined in pseudo code. Depending on which operation they perform and on which type of table, they are categorized as Method1 through Method3. They are:

Method 1: fact insert data maintenance

Method 2: fact delete data maintenance

Method 3: inventory delete data maintenance

5.3.2   The following table lists all data maintenance functions, their type of operation and target table. The number of rows in the views must be equal to the rowcounts in the source schema tables listed in column 6 of Table 5-4. The rowcounts of the source schema tables are listed in Table 5-2.

Table 5-4 Data Maintenance Function Summary

Data Maintenance Function ID	Data Maintenance Function	Type of Operation	View Name	Target Table	Source Schema Table
1	LF_CR(Clause 5.3.11.6)	Method 1	crv	catalog_returns	s_catalog_returns
2	LF_CS(Clause 5.3.11.5)	Method 1	csv	catalog_sales	s_catalog_sales
3	LF_I(Clause 5.3.11.7)	Method 1	iv	inventory	s_inventory
4	LF_SR(Clause 5.3.11.2)	Method 1	srv	store_returns	s_store_returns
5	LF_SS(Clause 5.3.11.1)	Method 1	ssv	store_sales	s_purchase_lineitem
6	LF_WR(Clause 5.3.11.4)	Method 1	wrv	web_returns	s_web_returns
7	LF_WS(Clause 5.3.11.3)	Method 1	wsv	web_sales	s_web_order_lineitem
8	DF_CS(Clause 5.3.11.10)	Method 2	-	catalog_sales [S], catalog_returns [R]	-
9	DF_SS(Clause 5.3.11.9)	Method 2	-	store_sales [S], store_returns [R]	-
10	DF_WS(Clause 5.3.11.11)	Method 2	-	web_sales [S], web_returns [R]	-
11	DF_I(Clause 5.3.11.12)	Method 3	-	Inventory [I]	-

5.3.3   Data maintenance function method 1 reads rows from a view V (see column View Name of table in Clause 5.3.2) and insert rows into a data warehouse table T. Both V and T are defined as part of the data maintenance function. T is created as part of the initial load of the data warehouse. V is a logical table that does not need to be instantiated.

5.3.4   The primary key of V is defined in the data maintenance function. Each data maintenance function contains a table with column mapping between its view V and its data warehouse table T. The primary key of V is denoted in bold letters on the left side of this mapping table (e.g. Table 5-5).

5.3.5   Business keys are the primary keys from the source schema. Business keys are denoted in bold letters on the right side of the mapping table for the data maintenance function (e.g. Table 5-5).

5.3.6   Generating a new primary key value for a dimension table is defined as generating the next largest value in the dense sequence of the table's primary key values. That is, assuming the largest current primary key value is x then the next value is x+1.

5.3.7   Method 1: Fact Table Load

```
for every row v in view V corresponding to fact table F
        get row v into local variable lv
        for every type 1 business key column bkc in v
            get row d from dimension table D corresponding to bkc
                where the business keys of v and d are equal
            update bkc of lv with surrogate key of d
        end for
        for every type 2 business key column bkc in v
            get row d from dimension table D corresponding to bkc
                where the business keys of v and d are equal and
```

```
                                                    rec_end_date is NULL
                    update bkc of lv with surrogate key of d
            end for
            insert lv into F

        end for
```

5.3.8 Method 2: Sales and Returns Fact Table Delete

```
Delete rows from R with corresponding rows in S
        where d_date between Date1 and Date2
Delete rows from S
        where d_date between Date1 and Date2
```

Comment: D_date is a column of the date_dim dimension. D_date has to be obtained by joining to the date_dim dimension on sales date surrogate key. The sales date surrogate key for the store sales is ss_sold_date_sk, for catalog it is cs_sold_date_sk and for web sales it is ws_sold_date_sk.

5.3.9 Method 3: Inventory Fact Table Delete

```
Delete rows from I where d_date between Date1 and Date2
```

Comment: D_date is a column of the date_dim dimension. D_date has to be obtained by joining to the date_dim dimension on inv_date_sk.

5.3.10 Each data maintenance function inserting or updating rows in dimension and fact tables is defined by the following components:

a) **Descriptor**, indicating the name of the data maintenance function in the form of DM_<abbreviation of data warehouse table> for dimensions and LF_<abbreviation of the data warehouse fact table> for fact tables. The extension indicates the data warehouse table that is populated with this data maintenance function.
b) The **data maintenance method** describes the pseudo code of the data maintenance function.
c) A **SQL view V** describing which tables of the source schema need to be joined to obtain the correct rows to be loaded.
d) The **column mapping** defining which source schema columns map to which data warehouse columns;

5.3.11 Each data maintenance function deleting rows from fact tables is defined by the following components:

a) **Descriptor**, indicating the name of the data maintenance function in the form of DF_<abbreviation of data warehouse fact table>. The extension indicates the data warehouse fact table from which rows are deleted.
b) Tables: **S** and **R,** or **I** in case of inventory
c) Two dates: **Date1** and **Date2**
d) The **data maintenance method** indicates how data is deleted

Comment: In the flat files generated by dsdgen for data maintenance there are 2 files which relate to deletes. One flat file (delete_<n>.dat) associated with deletes applies to sales and returns for store, web and catalog where <n> denotes the set number, defined in Clause 5.1.2). The second flat file (inventory_delete_<n>.dat) applies to inventory only where <n> denotes the set number,d efined in Clause 5.1.2). In each delete flat file there are 3 sets of start and end dates for the delete function.   Each of the 3 sets of dates must be applied.

5.3.11.1   LF_SS

```
CREATE view ssv as
SELECT  d_date_sk ss_sold_date_sk,
        t_time_sk ss_sold_time_sk,
        i_item_sk ss_item_sk,
        c_customer_sk ss_customer_sk,
        c_current_cdemo_sk ss_cdemo_sk,
        c_current_hdemo_sk ss_hdemo_sk,
        c_current_addr_sk ss_addr_sk,
        s_store_sk ss_store_sk,
        p_promo_sk ss_promo_sk,
        purc_purchase_id ss_ticket_number,
        plin_quantity ss_quantity,
        i_wholesale_cost ss_wholesale_cost,
        i_current_price ss_list_price,
        plin_sale_price ss_sales_price,
        (i_current_price-plin_sale_price)*plin_quantity ss_ext_discount_amt,
        plin_sale_price * plin_quantity ss_ext_sales_price,
        i_wholesale_cost * plin_quantity ss_ext_wholesale_cost,
        i_current_price * plin_quantity ss_ext_list_price,
        i_current_price * s_tax_precentage ss_ext_tax,
        plin_coupon_amt ss_coupon_amt,
        (plin_sale_price * plin_quantity)-plin_coupon_amt ss_net_paid,
        ((plin_sale_price * plin_quantity)-plin_coupon_amt)*(1+s_tax_precentage) ss_net_paid_inc_tax,
        ((plin_sale_price * plin_quantity)-plin_coupon_amt)-(plin_quantity*i_wholesale_cost)
ss_net_profit
FROM    s_purchase
LEFT OUTER JOIN customer ON (purc_customer_id = c_customer_id)
LEFT OUTER JOIN store ON (purc_store_id = s_store_id)
LEFT OUTER JOIN date_dim ON (cast(purc_purchase_date as date) = d_date)
LEFT OUTER JOIN time_dim ON (PURC_PURCHASE_TIME = t_time)
JOIN s_purchase_lineitem ON (purc_purchase_id = plin_purchase_id)
LEFT OUTER JOIN promotion ON plin_promotion_id = p_promo_id
LEFT OUTER JOIN item ON plin_item_id = i_item_id
WHERE   purc_purchase_id = plin_purchase_id
    AND i_rec_end_date is NULL
    AND s_rec_end_date is NULL;
```

Table 5-5: Column mapping for the store_sales fact table

Source Schema Column	Target Column
SS_SOLD_DATE_SK	SS_SOLD_DATE_SK
SS_SOLD_TIME_SK	SS_SOLD_TIME_SK
SS_ITEM_SK	**SS_ITEM_SK**
SS_CUSTOMER_SK	SS_CUSTOMER_SK
SS_CDEMO_SK	SS_CDEMO_SK
SS_HDEMO_SK	SS_HDEMO_SK
SS_ADDR_SK	SS_ADDR_SK
SS_STORE_SK	SS_STORE_SK
SS_PROMO_SK	SS_PROMO_SK
SS_TICKET_NUMBER	**SS_TICKET_NUMBER**
SS_QUANTITY	SS_QUANTITY
SS_WHOLESALE_COST	SS_WHOLESALE_COST
SS_LIST_PRICE	SS_LIST_PRICE
SS_SALES_PRICE	SS_SALES_PRICE
SS_EXT_DISCOUNT_AMT	SS_EXT_DISCOUNT_AMT
SS_EXT_SALES_PRICE	SS_EXT_SALES_PRICE
SS_EXT_WHOLESALE_COST	SS_EXT_WHOLESALE_COST
SS_EXT_LIST_PRICE	SS_EXT_LIST_PRICE
SS_EXT_TAX	SS_EXT_TAX
SS_COUPON_AMT	SS_COUPON_AMT
SS_NET_PAID	SS_NET_PAID
SS_NET_PAID_INC_TAX	SS_NET_PAID_INC_TAX
SS_NET_PROFIT	SS_NET_PROFIT

5.3.11.2   LF_SR

```
CREATE view srv as
SELECT d_date_sk sr_returned_date_sk
       ,t_time_sk sr_return_time_sk
       ,i_item_sk sr_item_sk
       ,c_customer_sk sr_customer_sk
       ,c_current_cdemo_sk sr_cdemo_sk
       ,c_current_hdemo_sk sr_hdemo_sk
       ,c_current_addr_sk sr_addr_sk
       ,s_store_sk sr_store_sk
       ,r_reason_sk sr_reason_sk
       ,sret_ticket_number sr_ticket_number
       ,sret_return_qty sr_return_quantity
       ,sret_return_amt sr_return_amt
       ,sret_return_tax sr_return_tax
       ,sret_return_amt + sret_return_tax sr_return_amt_inc_tax
       ,sret_return_fee sr_fee
       ,sret_return_ship_cost sr_return_ship_cost
       ,sret_refunded_cash sr_refunded_cash
       ,sret_reversed_charge sr_reversed_charge
       ,sret_store_credit sr_store_credit
       ,sret_return_amt+sret_return_tax+sret_return_fee
        -sret_refunded_cash-sret_reversed_charge-sret_store_credit sr_net_loss
FROM s_store_returns
LEFT OUTER JOIN date_dim
  ON (cast(sret_return_date as date) = d_date)
LEFT OUTER JOIN time_dim
  ON (( cast(substr(sret_return_time,1,2) AS integer)*3600
       +cast(substr(sret_return_time,4,2) AS integer)*60
       +cast(substr(sret_return_time,7,2) AS integer)) = t_time)
LEFT OUTER JOIN item ON (sret_item_id = i_item_id)
LEFT OUTER JOIN customer ON (sret_customer_id = c_customer_id)
LEFT OUTER JOIN store ON (sret_store_id = s_store_id)
LEFT OUTER JOIN reason ON (sret_reason_id = r_reason_id)
WHERE i_rec_end_date IS NULL
  AND s_rec_end_date IS NULL;
```

Table 5-6: Column mapping for the store_returns fact table

Source Schema Column	Target Column
SR_RETURNED_DATE_SK	SR_RETURNED_DATE_SK
SR_RETURN_TIME_SK	SR_RETURN_TIME_SK
SR_ITEM_SK	**SR_ITEM_SK**
SR_CUSTOMER_SK	SR_CUSTOMER_SK
SR_CDEMO_SK	SR_CDEMO_SK
SR_HDEMO_SK	SR_HDEMO_SK
SR_ADDR_SK	SR_ADDR_SK
SR_STORE_SK	SR_STORE_SK
SR_REASON_SK	SR_REASON_SK
SR_TICKET_NUMBER	**SR_TICKET_NUMBER**
SR_RETURN_QUANTITY	SR_RETURN_QUANTITY
SR_RETURN_AMT	SR_RETURN_AMT
SR_RETURN_TAX	SR_RETURN_TAX
SR_RETURN_AMT_INC_TAX	SR_RETURN_AMT_INC_TAX
SR_FEE	SR_FEE
SR_RETURN_SHIP_COST	SR_RETURN_SHIP_COST
SR_REFUNDED_CASH	SR_REFUNDED_CASH
SR_REVERSED_CHARGE	SR_REVERSED_CHARGE
SR_STORE_CREDIT	SR_STORE_CREDIT
SR_NET_LOSS	SR_NET_LOSS

5.3.11.3   LF_WS

```
CREATE VIEW wsv AS
SELECT  d1.d_date_sk ws_sold_date_sk,
        t_time_sk ws_sold_time_sk,
        d2.d_date_sk ws_ship_date_sk,
        i_item_sk ws_item_sk,
        c1.c_customer_sk ws_bill_customer_sk,
        c1.c_current_cdemo_sk ws_bill_cdemo_sk,
        c1.c_current_hdemo_sk ws_bill_hdemo_sk,
        c1.c_current_addr_sk ws_bill_addr_sk,
        c2.c_customer_sk ws_ship_customer_sk,
        c2.c_current_cdemo_sk ws_ship_cdemo_sk,
        c2.c_current_hdemo_sk ws_ship_hdemo_sk,
        c2.c_current_addr_sk ws_ship_addr_sk,
        wp_web_page_sk ws_web_page_sk,
        web_site_sk ws_web_site_sk,
        sm_ship_mode_sk ws_ship_mode_sk,
        w_warehouse_sk ws_warehouse_sk,
        p_promo_sk ws_promo_sk,
        word_order_id ws_order_number,
        wlin_quantity ws_quantity,
        i_wholesale_cost ws_wholesale_cost,
        i_current_price ws_list_price,
        wlin_sales_price ws_sales_price,
        (i_current_price-wlin_sales_price)*wlin_quantity ws_ext_discount_amt,
        wlin_sales_price * wlin_quantity ws_ext_sales_price,
        i_wholesale_cost * wlin_quantity ws_ext_wholesale_cost,
        i_current_price * wlin_quantity ws_ext_list_price,
        i_current_price * web_tax_percentage ws_ext_tax,
        wlin_coupon_amt ws_coupon_amt,
        wlin_ship_cost * wlin_quantity WS_EXT_SHIP_COST,
        (wlin_sales_price * wlin_quantity)-wlin_coupon_amt ws_net_paid,
        ((wlin_sales_price * wlin_quantity)-wlin_coupon_amt)*(1+web_tax_percentage) ws_net_paid_inc_tax,
        ((wlin_sales_price * wlin_quantity)-wlin_coupon_amt)-(wlin_quantity*i_wholesale_cost)
WS_NET_PAID_INC_SHIP,
        (wlin_sales_price * wlin_quantity)-wlin_coupon_amt + (wlin_ship_cost * wlin_quantity)
        + i_current_price * web_tax_percentage WS_NET_PAID_INC_SHIP_TAX,
        ((wlin_sales_price * wlin_quantity)-wlin_coupon_amt)-(i_wholesale_cost * wlin_quantity)
WS_NET_PROFIT
FROM    s_web_order
LEFT OUTER JOIN date_dim d1 ON (cast(word_order_date as date) = d1.d_date)
LEFT OUTER JOIN time_dim ON (word_order_time = t_time)
LEFT OUTER JOIN customer c1 ON (word_bill_customer_id = c1.c_customer_id)
LEFT OUTER JOIN customer c2 ON (word_ship_customer_id = c2.c_customer_id)
LEFT OUTER JOIN web_site ON (word_web_site_id = web_site_id AND web_rec_end_date IS NULL)
LEFT OUTER JOIN ship_mode ON (word_ship_mode_id = sm_ship_mode_id)
JOIN s_web_order_lineitem ON (word_order_id = wlin_order_id)
LEFT OUTER JOIN date_dim d2 ON (cast(wlin_ship_date as date) = d2.d_date)
LEFT OUTER JOIN item ON (wlin_item_id = i_item_id AND i_rec_end_date IS NULL)
LEFT OUTER JOIN web_page ON (wlin_web_page_id = wp_web_page_id AND wp_rec_end_date IS NULL)
LEFT OUTER JOIN warehouse ON (wlin_warehouse_id = w_warehouse_id)
LEFT OUTER JOIN promotion ON (wlin_promotion_id = p_promo_id);
```

Table 5-7: Column mapping for the web_sales fact table

Source Schema Column	Target Column
WS_SOLD_DATE_SK	WS_SOLD_DATE_SK
WS_SOLD_TIME_SK	WS_SOLD_TIME_SK
WS_SHIP_DATE_SK	WS_SHIP_DATE_SK
WS_ITEM_SK	**WS_ITEM_SK**
WS_BILL_CUSTOMER_SK	WS_BILL_CUSTOMER_SK
WS_BILL_CDEMO_SK	WS_BILL_CDEMO_SK
WS_BILL_HDEMO_SK	WS_BILL_HDEMO_SK
WS_BILL_ADDR_SK	WS_BILL_ADDR_SK
WS_SHIP_CUSTOMER_SK	WS_SHIP_CUSTOMER_SK
WS_SHIP_CDEMO_SK	WS_SHIP_CDEMO_SK
WS_SHIP_HDEMO_SK	WS_SHIP_HDEMO_SK
WS_SHIP_ADDR_SK	WS_SHIP_ADDR_SK
WS_WEB_PAGE_SK	WS_WEB_PAGE_SK
WS_WEB_SITE_SK	WS_WEB_SITE_SK

Source Schema Column	Target Column
WS_SHIP_MODE_SK	WS_SHIP_MODE_SK
WS_WAREHOUSE_SK	WS_WAREHOUSE_SK
WS_PROMO_SK	WS_PROMO_SK
WS_ORDER_NUMBER	**WS_ORDER_NUMBER**
WS_QUANTITY	WS_QUANTITY
WS_WHOLESALE_COST	WS_WHOLESALE_COST
WS_LIST_PRICE	WS_LIST_PRICE
WS_SALES_PRICE	WS_SALES_PRICE
WS_EXT_DISCOUNT_AMT	WS_EXT_DISCOUNT_AMT
WS_EXT_SALES_PRICE	WS_EXT_SALES_PRICE
WS_EXT_WHOLESALE_COST	WS_EXT_WHOLESALE_COST
WS_EXT_LIST_PRICE	WS_EXT_LIST_PRICE
WS_EXT_TAX	WS_EXT_TAX
WS_COUPON_AMT	WS_COUPON_AMT
WS_EXT_SHIP_COST	WS_EXT_SHIP_COST
WS_NET_PAID	WS_NET_PAID
WS_NET_PAID_INC_TAX	WS_NET_PAID_INC_TAX
WS_NET_PAID_INC_SHIP	WS_NET_PAID_INC_SHIP
WS_NET_PAID_INC_SHIP_TAX	WS_NET_PAID_INC_SHIP_TAX
WS_NET_PROFIT	WS_NET_PROFIT

5.3.11.4   LF_WR

```
CREATE VIEW wrv AS
SELECT d_date_sk wr_return_date_sk
      ,t_time_sk wr_return_time_sk
      ,i_item_sk wr_item_sk
      ,c1.c_customer_sk wr_refunded_customer_sk
      ,c1.c_current_cdemo_sk wr_refunded_cdemo_sk
      ,c1.c_current_hdemo_sk wr_refunded_hdemo_sk
      ,c1.c_current_addr_sk wr_refunded_addr_sk
      ,c2.c_customer_sk wr_returning_customer_sk
      ,c2.c_current_cdemo_sk wr_returning_cdemo_sk
      ,c2.c_current_hdemo_sk wr_returning_hdemo_sk
      ,c2.c_current_addr_sk wr_returing_addr_sk
      ,wp_web_page_sk wr_web_page_sk
      ,r_reason_sk wr_reason_sk
      ,wret_order_id wr_order_number
      ,wret_return_qty wr_return_quantity
      ,wret_return_amt wr_return_amt
      ,wret_return_tax wr_return_tax
      ,wret_return_amt + wret_return_tax AS wr_return_amt_inc_tax
      ,wret_return_fee wr_fee
      ,wret_return_ship_cost wr_return_ship_cost
      ,wret_refunded_cash wr_refunded_cash
      ,wret_reversed_charge wr_reversed_charge
      ,wret_account_credit wr_account_credit
      ,wret_return_amt+wret_return_tax+wret_return_fee
       -wret_refunded_cash-wret_reversed_charge-wret_account_credit wr_net_loss
FROM s_web_returns LEFT OUTER JOIN date_dim ON (cast(wret_return_date as date) = d_date)
LEFT OUTER JOIN time_dim ON ((CAST(SUBSTR(wret_return_time,1,2) AS integer)*3600
+CAST(SUBSTR(wret_return_time,4,2) AS integer)*60+CAST(SUBSTR(wret_return_time,7,2) AS integer))=t_time)
LEFT OUTER JOIN item ON (wret_item_id = i_item_id)
LEFT OUTER JOIN customer c1 ON (wret_return_customer_id = c1.c_customer_id)
LEFT OUTER JOIN customer c2 ON (wret_refund_customer_id = c2.c_customer_id)
LEFT OUTER JOIN reason ON (wret_reason_id = r_reason_id)
LEFT OUTER JOIN web_page ON (wret_web_page_id = WP_WEB_PAGE_id)
WHERE i_rec_end_date IS NULL AND wp_rec_end_date IS NULL;
```

Table: 5-8: Column mapping for the web_returns fact table

Source Schema Column	Target Column
WR_RETURNED_DATE_SK	WR_RETURNED_DATE_SK
WR_RETURNED_TIME_SK	WR_RETURNED_TIME_SK
WR_SHIP_DATE_SK	WR_SHIP_DATE_SK
WR_ITEM_SK	**WR_ITEM_SK**
WR_REFUNDED_CUSTOMER_SK	WR_REFUNDED_CUSTOMER_SK
WR_REFUNDED_CDEMO_SK	WR_REFUNDED_CDEMO_SK
WR_REFUNDED_HDEMO_SK	WR_REFUNDED_HDEMO_SK
WR_REFUNDED_ADDR_SK	WR_REFUNDED_ADDR_SK
WR_RETURNING_CUSTOMER_SK	WR_RETURNING_CUSTOMER_SK
WR_RETURNING_CDEMO_SK	WR_RETURNING_CDEMO_SK
WR_RETURNING_HDEMO_SK	WR_RETURNING_HDEMO_SK
WR_RETURNING_ADDR_SK	WR_RETURNING_ADDR_SK
WR_WEB_PAGE_SK	WR_WEB_PAGE_SK
WR_SHIP_MODE_SK	WR_SHIP_MODE_SK
WR_REASON_SK	WR_REASON_SK
WR_WAREHOUSE_SK	WR_WAREHOUSE_SK
WR_ORDER_NUMBER	**WR_ORDER_NUMBER**
WR_RETURN_QUANTITY	WR_RETURN_QUANTITY
WR_RETURN_AMT	WR_RETURN_AMT
WR_RETURN_TAX	WR_RETURN_TAX
WR_RETURN_AMT_INC_TAX	WR_RETURN_AMT_INC_TAX
WR_FEE	WR_FEE
WR_RETURN_SHIP_COST	WR_RETURN_SHIP_COST
WR_REFUNDED_CASH	WR_REFUNDED_CASH
WR_REVERSED_CHARGE	WR_REVERSED_CHARGE
WR_ACCOUNT_CREDIT	WR_ACCOUNT_CREDIT
WR_NET_LOSS	WR_NET_LOSS

5.3.11.5   LF_CS

```
CREATE view csv as
SELECT d1.d_date_sk cs_sold_date_sk
      ,t_time_sk cs_sold_time_sk
      ,d2.d_date_sk cs_ship_date_sk
      ,c1.c_customer_sk cs_bill_customer_sk
      ,c1.c_current_cdemo_sk cs_bill_cdemo_sk
      ,c1.c_current_hdemo_sk cs_bill_hdemo_sk
      ,c1.c_current_addr_sk cs_bill_addr_sk
      ,c2.c_customer_sk cs_ship_customer_sk
      ,c2.c_current_cdemo_sk cs_ship_cdemo_sk
      ,c2.c_current_hdemo_sk cs_ship_hdemo_sk
      ,c2.c_current_addr_sk cs_ship_addr_sk
      ,cc_call_center_sk cs_call_center_sk
      ,cp_catalog_page_sk cs_catalog_page_sk
      ,sm_ship_mode_sk cs_ship_mode_sk
      ,w_warehouse_sk cs_warehouse_sk
      ,i_item_sk cs_item_sk
      ,p_promo_sk cs_promo_sk
      ,cord_order_id cs_order_number
      ,clin_quantity cs_quantity
      ,i_wholesale_cost cs_wholesale_cost
      ,i_current_price cs_list_price
      ,clin_sales_price cs_sales_price
      ,(i_current_price-clin_sales_price)*clin_quantity cs_ext_discount_amt
      ,clin_sales_price * clin_quantity cs_ext_sales_price
      ,i_wholesale_cost * clin_quantity cs_ext_wholesale_cost
      ,i_current_price * clin_quantity CS_EXT_LIST_PRICE
      ,i_current_price * cc_tax_percentage CS_EXT_TAX
      ,clin_coupon_amt cs_coupon_amt
      ,clin_ship_cost * clin_quantity CS_EXT_SHIP_COST
      ,(clin_sales_price * clin_quantity)-clin_coupon_amt cs_net_paid
      ,((clin_sales_price * clin_quantity)-clin_coupon_amt)*(1+cc_tax_percentage) cs_net_paid_inc_tax
      ,(clin_sales_price * clin_quantity)-clin_coupon_amt + (clin_ship_cost * clin_quantity) CS_NET_PAID_INC_SHIP
      ,(clin_sales_price * clin_quantity)-clin_coupon_amt + (clin_ship_cost * clin_quantity)
       + i_current_price * cc_tax_percentage CS_NET_PAID_INC_SHIP_TAX
      ,((clin_sales_price * clin_quantity)-clin_coupon_amt)-(clin_quantity*i_wholesale_cost) cs_net_profit
FROM    s_catalog_order
LEFT OUTER JOIN date_dim d1 ON
  (cast(cord_order_date as date) = d1.d_date)
LEFT OUTER JOIN time_dim ON (cord_order_time = t_time)
LEFT OUTER JOIN customer c1 ON (cord_bill_customer_id = c1.c_customer_id)
LEFT OUTER JOIN customer c2 ON (cord_ship_customer_id = c2.c_customer_id)
LEFT OUTER JOIN call_center ON (cord_call_center_id = cc_call_center_id AND cc_rec_end_date IS NULL)
LEFT OUTER JOIN ship_mode ON (cord_ship_mode_id = sm_ship_mode_id)
JOIN s_catalog_order_lineitem ON (cord_order_id = clin_order_id)
LEFT OUTER JOIN date_dim d2 ON
  (cast(clin_ship_date as date) = d2.d_date)
LEFT OUTER JOIN catalog_page ON
  (clin_catalog_page_number = cp_catalog_page_number and clin_catalog_number = cp_catalog_number)
LEFT OUTER JOIN warehouse ON (clin_warehouse_id = w_warehouse_id)
LEFT OUTER JOIN item ON (clin_item_id = i_item_id AND i_rec_end_date IS NULL)
LEFT OUTER JOIN promotion ON (clin_promotion_id = p_promo_id);
```

Table 5-9: Column mapping for the catalog_sales fact table

Source Schema Column	Target Column
CS_SOLD_DATE_SK	CS_SOLD_DATE_SK
CS_SOLD_TIME_SK	CS_SOLD_TIME_SK
CS_SHIP_DATE_SK	CS_SHIP_DATE_SK
CS_BILL_CUSTOMER_SK	CS_BILL_CUSTOMER_SK
CS_BILL_CDEMO_SK	CS_BILL_CDEMO_SK
CS_BILL_HDEMO_SK	CS_BILL_HDEMO_SK
CS_BILL_ADDR_SK	CS_BILL_ADDR_SK
CS_SHIP_CUSTOMER_SK	CS_SHIP_CUSTOMER_SK
CS_SHIP_CDEMO_SK	CS_SHIP_CDEMO_SK
CS_SHIP_HDEMO_SK	CS_SHIP_HDEMO_SK
CS_SHIP_ADDR_SK	CS_SHIP_ADDR_SK
CS_CALL_CENTER_SK	CS_CALL_CENTER_SK
CS_CATALOG_PAGE_SK	CS_CATALOG_PAGE_SK
CS_SHIP_MODE_SK	CS_SHIP_MODE_SK
CS_WAREHOUSE_SK	CS_WAREHOUSE_SK
CS_ITEM_SK	**CS_ITEM_SK**
CS_PROMO_SK	CS_PROMO_SK
CS_ORDER_NUMBER	**CS_ORDER_NUMBER**
CS_QUANTITY	CS_QUANTITY

Source Schema Column	Target Column
CS_WHOLESALE_COST	CS_WHOLESALE_COST
CS_LIST_PRICE	CS_LIST_PRICE
CS_SALES_PRICE	CS_SALES_PRICE
CS_EXT_DISCOUNT_AMT	CS_EXT_DISCOUNT_AMT
CS_EXT_SALES_PRICE	CS_EXT_SALES_PRICE
CS_EXT_WHOLESALE_COST	CS_EXT_WHOLESALE_COST
CS_EXT_LIST_PRICE	CS_EXT_LIST_PRICE
CS_EXT_TAX	CS_EXT_TAX
CS_COUPON_AMT	CS_COUPON_AMT
CS_EXT_SHIP_COST	CS_EXT_SHIP_COST
CS_NET_PAID	CS_NET_PAID
CS_NET_PAID_INC_TAX	CS_NET_PAID_INC_TAX
CS_NET_PAID_INC_SHIP	CS_NET_PAID_INC_SHIP
CS_NET_PAID_INC_SHIP_TAX	CS_NET_PAID_INC_SHIP_TAX
CS_NET_PROFIT	CS_NET_PROFIT

5.3.11.6   LF_CR

```
CREATE VIEW crv as
SELECT d_date_sk cr_return_date_sk
      ,t_time_sk cr_return_time_sk
      ,i_item_sk cr_item_sk
      ,c1.c_customer_sk cr_refunded_customer_sk
      ,c1.c_current_cdemo_sk cr_refunded_cdemo_sk
      ,c1.c_current_hdemo_sk cr_refunded_hdemo_sk
      ,c1.c_current_addr_sk cr_refunded_addr_sk
      ,c2.c_customer_sk cr_returning_customer_sk
      ,c2.c_current_cdemo_sk cr_returning_cdemo_sk
      ,c2.c_current_hdemo_sk cr_returning_hdemo_sk
      ,c2.c_current_addr_sk cr_returing_addr_sk
      ,cc_call_center_sk cr_call_center_sk
      ,cp_catalog_page_sk CR_CATALOG_PAGE_SK
      ,sm_ship_mode_sk CR_SHIP_MODE_SK
      ,w_warehouse_sk CR_WAREHOUSE_SK
      ,r_reason_sk cr_reason_sk
      ,cret_order_id cr_order_number
      ,cret_return_qty cr_return_quantity
      ,cret_return_amt cr_return_amt
      ,cret_return_tax cr_return_tax
      ,cret_return_amt + cret_return_tax AS cr_return_amt_inc_tax
      ,cret_return_fee cr_fee
      ,cret_return_ship_cost cr_return_ship_cost
      ,cret_refunded_cash cr_refunded_cash
      ,cret_reversed_charge cr_reversed_charge
      ,cret_merchant_credit cr_merchant_credit
      ,cret_return_amt+cret_return_tax+cret_return_fee
        -cret_refunded_cash-cret_reversed_charge-cret_merchant_credit cr_net_loss
FROM s_catalog_returns
LEFT OUTER JOIN date_dim
  ON (cast(cret_return_date as date) = d_date)
LEFT OUTER JOIN time_dim ON
  ((CAST(substr(cret_return_time,1,2) AS integer)*3600
   +CAST(substr(cret_return_time,4,2) AS integer)*60
   +CAST(substr(cret_return_time,7,2) AS integer)) = t_time)
LEFT OUTER JOIN item ON (cret_item_id = i_item_id)
LEFT OUTER JOIN customer c1 ON (cret_return_customer_id = c1.c_customer_id)
LEFT OUTER JOIN customer c2 ON (cret_refund_customer_id = c2.c_customer_id)
LEFT OUTER JOIN reason ON (cret_reason_id = r_reason_id)
LEFT OUTER JOIN call_center ON (cret_call_center_id = cc_call_center_id)
LEFT OUTER JOIN catalog_page ON (cret_catalog_page_id = cp_catalog_page_id)
LEFT OUTER JOIN ship_mode ON (cret_shipmode_id = sm_ship_mode_id)
LEFT OUTER JOIN warehouse ON (cret_warehouse_id = w_warehouse_id)
WHERE i_rec_end_date IS NULL AND cc_rec_end_date IS NULL;
```

Table 5-10: Column mapping for the catalog_returns fact table

Source Schema Column	Target Column
CR_RETURNED_DATE_SK	CR_RETURNED_DATE_SK
CR_RETURNED_TIME_SK	CR_RETURNED_TIME_SK
CR_SHIP_DATE_SK	CR_SHIP_DATE_SK
CR_ITEM_SK	**CR_ITEM_SK**
CR_REFUNDED_CUSTOMER_SK	CR_REFUNDED_CUSTOMER_SK
CR_REFUNDED_CDEMO_SK	CR_REFUNDED_CDEMO_SK
CR_REFUNDED_HDEMO_SK	CR_REFUNDED_HDEMO_SK
CR_REFUNDED_ADDR_SK	CR_REFUNDED_ADDR_SK
CR_RETURNING_CUSTOMER_SK	CR_RETURNING_CUSTOMER_SK
CR_RETURNING_CDEMO_SK	CR_RETURNING_CDEMO_SK
CR_RETURNING_HDEMO_SK	CR_RETURNING_HDEMO_SK
CR_RETURNING_ADDR_SK	CR_RETURNING_ADDR_SK
CR_CALL_CENTER_SK	CR_CALL_CENTER_SK
CR_CATALOG_PAGE_SK	CR_CATALOG_PAGE_SK
CR_SHIP_MODE_SK	CR_SHIP_MODE_SK
CR_WAREHOUSE_SK	CR_WAREHOUSE_SK
CR_REASON_SK	CR_REASON_SK
CR_ORDER_NUMBER	**CR_ORDER_NUMBER**
CR_RETURN_QUANTITY	CR_RETURN_QUANTITY
CR_RETURN_AMOUNT	CR_RETURN_AMOUNT

Source Schema Column	Target Column
CR_RETURN_TAX	CR_RETURN_TAX
CR_RETURN_AMT_INC_TAX	CR_RETURN_AMT_INC_TAX
CR_FEE	CR_FEE
CR_RETURN_SHIP_COST	CR_RETURN_SHIP_COST
CR_REFUNDED_CASH	CR_REFUNDED_CASH
CR_REVERSED_CHARGE	CR_REVERSED_CHARGE
CR_ACCOUNT_CREDIT	CR_ACCOUNT_CREDIT
CR_NET_LOSS	CR_NET_LOSS

5.3.11.7   LF_I:

5.3.11.8

```
CREATE view iv AS
SELECT d_date_sk inv_date_sk,
       i_item_sk inv_item_sk,
       w_warehouse_sk inv_warehouse_sk,
       invn_qty_on_hand inv_quantity_on_hand
FROM s_inventory
LEFT OUTER JOIN warehouse ON (invn_warehouse_id=w_warehouse_id)
LEFT OUTER JOIN item ON (invn_item_id=i_item_id AND i_rec_end_date IS NULL)
LEFT OUTER JOIN date_dim ON (d_date=invn_date);
```

Table 5-11: Column mapping for the inventory fact table

Source Schema Column	Target Column
inv_date_sk	**inv_date_sk**
inv_item_sk	**inv_item_sk**
inv_warehouse_sk	**inv_warehouse_sk**
inv_quantity_on_hand	**inv_quantity_on_hand**

5.3.11.9   DF_SS:
S=store_sales
R=store_returns
Date1 as generated by dsdgen
Date2 as generated by dsdgen

5.3.11.10   DF_CS:
S=catalog_sales
R=catalog_returns
Date1 as generated by dsdgen
Date2 as generated by dsdgen

5.3.11.11   DF_WS:
S=web_sales
R=web_returns
Date1 as generated by dsdgen
Date2 as generated by dsdgen

5.3.11.12   DF_I:
I=Inventory
Date1 as generated by dsdgen
Date2 as generated by dsdgen

6   Data Accessibility Properties

6.1   The Data Accessibility Properties

The System Under Test must be configured to satisfy the requirements for Data Accessibility described in this clause. Data Accessibility 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, EADS, or metadata. Data Accessibility tests are conducted by inducing failure of a Durable Medium within the SUT.

6.1.1   Definition of Terms

6.1.1.1   Data Accessibility: The ability to maintain operations with full data access after the permanent irrecoverable failure of any single Durable Medium containing tables, EADS, or metadata.

6.1.1.2   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, persistent 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-volatile medium after the failure of external power.

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

Comment: Memory 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).

6.1.1.3   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.

6.1.2   Data Accessibility Requirements

6.1.2.1   The test sponsor shall guarantee the test system will continue executing queries and data maintenance functions with full data access during and after the permanent irrecoverable failure of any single durable medium containing TPC-DS database objects, e.g. tables, EADS, or metadata. The medium to be failed is to be chosen at random by the auditor.

6.1.2.2   The Data Accessibility Test is performed by causing the failure of a single Durable Medium during the execution of the first Data Maintenance Test as described in Clause 7.4.   The Data Accessibility Test is successful if all in-flight and subsequent queries and data maintenance functions complete successfully.

6.1.2.3   The Data Accessibility Test must be performed as part of the Performance Test that is used as the basis for reporting the performance metric and results, while running against the test database at the full reported scale factor.

7   Performance Metrics and Execution Rules

7.1   Definition of Terms

7.1.1   The **Benchmark** is defined as the execution of the **Load Test** followed by the **Performance Test**.

7.1.2   The **Load Test** is defined as all activity required to bring the **System Under Test** to the configuration that immediately precedes the beginning of the **Performance Test**. The **Load Test** must not include the execution of any of the queries in the **Power Test** or **Throughput Test** or any similar query.

7.1.3   The **Performance Test** is defined as the **Power Test,** both **Throughput Tests** and both **Data Maintenance Tests**.

7.1.4   A **query stream** is defined as the sequential execution of a permutation of queries submitted by a single emulated user. A query stream consists of the 99 queries defined in Clause 4.

7.1.5   A **session** is defined as a uniquely identified process context capable of supporting the execution of user-initiated database activity.

7.1.6   A **query session** is a session executing activity on behalf of a Power Test or a Throughput Test.

7.1.7   A **refresh run** is defined as the execution of one set of data maintenance functions.

7.1.8   A **refresh session** is a session executing activity on behalf of a refresh run.

7.1.9   A **Throughput Test** consists of S_q query sessions each running a single query stream.

7.1.10   A **Power Test** consists of exactly one query session running a single query stream.

7.1.11   A **Data Maintenance Test** consists of the execution of a series of refresh sstreamss .

7.1.12   A **query** is an ordered set of one or more valid SQL statements resulting from applying the required parameter substitutions to a given query template. The order of the SQL statements is defined in the query template.

7.1.13   The **SUT** consists of a collection of configured components used to complete the benchmark.

7.1.14   The mechanism used to submit queries to the SUT and to measure their execution time is called a **driver**.

7.1.15   A **timestamp** must be taken in the time zone the SUT is located in. It is defined as any representation equivalent to yyyy-mm-dd hh:mm:ss.s, where:

- yyyy is the 4 digit representation of year
- mm is the 2 digit representation of month
- dd is the 2 digit representation of day
- hh is the 2 digit representation of hour in 24-hour clock notation
- mm is the 2 digit representation of minute
- ss.s is the 3 digit representation of second with a precision of at least 1/10 of a second

7.1.16   **Elapsed time** is measured in seconds rounded up to the nearest 0.1 second.

7.1.17   **Test Database** is the loaded data and created meta data required to execute the TPC-DS benchmark, i.e. Load test, Power test, Throughput test, Data maintenance test and all tests required by the auditor.

7.1.18   **Database Location** is the location of loaded data that is directly accessible (read/write) by the test database to query or apply dml operations on the TPC-DS tables defined in Clause 2 as required by Load test, Power test, Throughput test, Data maintenance test and all tests required by the auditor.

7.2   Configuration Rules

7.2.1   The driver is a logical entity that can be implemented using one or more physical programs, processes, or systems (see Clause 8.3).

7.2.2   The communication between the driver and the SUT must be limited to one session per query. These sessions are prohibited from communicating with one another except for the purpose of scheduling Data Maintenance functions (see Clause 5.3).

7.2.3   All query sessions must be initialized in exactly the same way. All refresh sessions must be initialized in exactly the same way. The initialization of a refresh session may be different than that of the query session.

7.2.4   All session initialization parameters, settings and commands must be disclosed.

Comment: The intent of this clause is to provide the information needed to precisely recreate the execution environment of any given stream as it exists prior to the submission of the first query or data maintenance function.

7.2.5   The driver shall submit each TPC-DS query for execution by the SUT via the session associated with the corresponding query stream.

7.2.6   In the case of the data maintenance functions, the driver is only required to submit the commands necessary to cause the execution of each data maintenance function.

7.2.7   The driver's submittal of the queries to the SUT during the performance test shall be limited to the transmission of the query text to the data processing system and whatever additional information is required to conform to the measurement and data gathering requirements defined in this document. In addition:

- The interaction between the driver and the SUT shall not have the purpose of indicating to the SUT or any of its components an execution strategy or priority that is time-dependent or query-specific;
- The interaction between the driver and the SUT shall not have the purpose of indicating to the SUT, or to any of its components, the insertion of time delays;
- The driver shall not insert time delays before, after, or between the submission of queries to the SUT;
- The interaction between the driver and the SUT shall not have the purpose of modifying the behavior or configuration of the SUT (i.e., data processing system or operating system settings) on a query-by-query basis. These parameters shall not be altered during the execution of the performance test.

Comment: One intent of this clause is to prohibit the pacing of query submission by the driver.

7.2.8   Environmental Assumptions

7.2.8.1   The configuration and initialization of the SUT, the database, or the session, including any relevant parameter, switch or option settings, shall be based only on externally documented capabilities of the system that can be reasonably interpreted as useful for a decision support workload. This workload is characterized by:
- Sequential scans of large amounts of data;
- Aggregation of large amounts of data;
- Multi-table joins;
- Possibly extensive sorting.

7.2.8.2   While the configuration and initialization can reflect the general nature of this expected workload, it shall not take special advantage of the limited functions actually exercised by the benchmark. The queries actually chosen in the benchmark are merely examples of the types of queries that might be used in such an environment, not necessarily actual user queries. Due to this limit in the scope of the queries and test environment, TPC-DS has chosen to restrict the use of some database technologies (see Clause 2.5). In general, the effect of the configuration on benchmark performance should be representative of its expected effect on the performance of the class of applications modeled by the benchmark.

7.2.8.3   The features, switches or parameter settings that comprise the configuration of the operating system, the data processing system or the session must be such that it would be reasonable to expect a database administrator with the following characteristics be able to decide to use them:

- Knowledge of the general characteristics of the workload as defined above;
- Knowledge of the logical and physical database layout;
- Access to operating system and database documentation;
- No knowledge of product internals beyond what is documented externally.

Each feature, switch or parameter setting used in the configuration and initialization of the operating system, the data processing system or the session must meet the following criteria:

- It shall remain in effect without change throughout the performance test;
- It shall not make reference to specific tables, indices or queries for the purpose of providing hints to the query optimizer.

7.2.9   The collection of statistics requested through the use of directives must be part of the database load. If these directives request the collection of different levels of statistics for different columns, they must adhere to the following rules.:

1) The level of statistics collected for a given column must be based on the column's membership in a class.
2) Class definitions must rely solely on the following column attributes from the logical database design (as defined in Clause 2):
   - Datatype;
   - Nullable;
   - Foreign Key;
   - Primary Key.
3) Class definitions may combine column attributes using AND, OR and NOT operators. (for example, one class could contain all columns satisfying the following combination of attributes: [Identifier Datatype] AND [NOT nullable OR Foreign Key]);
4) Class membership must be applied consistently on all columns across all tables;
5) Statistics that operate in sets, such as distribution statistics, should employ a fixed set appropriate to the scale factor used. Knowledge of the cardinality, values or distribution of a non-key column (as specified in Clause 3) must not be used to tailor statistics gathering.

7.2.10   Profile-Directed Optimization

7.2.10.1   Special rules apply to the use of so-called profile-directed optimization (PDO), in which binary executables are reordered or otherwise optimized to best suit the needs of a particular workload. These rules do not apply to the routine use of PDO by a database vendor in the course of building commercially available and supported database products; such use is not restricted. Rather, the rules apply to the use of PDO by a test sponsor to optimize executables of a database product for a particular workload. Such optimization is permissible if all of the following conditions are satisfied:

- The use of PDO or similar procedures by the test sponsor must be disclosed.
- The procedure and any scripts used to perform the optimization must be disclosed.
- The procedure used by the test sponsor could reasonably be used by a customer on a shipped database executable.
- The optimized database executables resulting from the application of the procedure must be supported by the database software vendor.
- The workload used to drive the optimization is described in Clause 7.2.10.2.
- The same set of executables must be used for all phases of the benchmark.

7.2.10.2   If profile-directed optimization is used, the workload used to drive it can be the execution of any subset of the TPC-DS queries or any data maintenance functions, in any order, against a TPC-DS database of any desired scale factor, with default substitution parameters applied. The query/data maintenance function set, used in PDO, must be reported.

7.3 Query Validation

7.3.1 All query templates used in a benchmark submission shall be validated.

7.3.2 The validation process is defined as follows:

1. Populate the qualification database (see Clause 3.3) ;
2. Execute the query template using qualification substitution parameters as defined in 11.9.1.1Appendix B:;
3. Compare the output to the answer set defined for the query.

7.3.3 The output data shall match the answer set defined for the query, subject to the constraints defined in Clause 7.5

7.4 Execution Rules

7.4.1 General Requirements

7.4.1.1 If the load test, power test, either throughput test, or either data maintenance test fail, the benchmark run is invalid.

7.4.1.2 All tables created with explicit directives during the execution of the benchmark tests must meet the data accessibility requirements defined in Clause 6.

7.4.1.3 The SUT, including any database server(s), shall not be restarted at any time after the power test begins until after all tests have completed.

7.4.1.4 The driver shall submit queries through one or more sessions on the SUT. Each session corresponds to one, and only one, query stream on the SUT.

7.4.1.5 Parallel activity within the SUT directed toward the execution of a single query or data maintenance function (e.g. intra-query parallelism) is not restricted.

7.4.1.6 The real-time clock used by the driver to compute the timing intervals must measure time with a resolution of at least 0.01 second.

7.4.2 The benchmark must use the following sequence of tests:

a) Database Load Test
b) Power Test
c) Throughput Test 1
d) Data Maintenance Test 1
e) Throughput Test 2
f) Data Maintenance Test 2

7.4.3 Database Load Test

7.4.3.1 The process of building the test database is known as database load. Database load consists of timed and un-timed components.

7.4.3.2 The population of the test database, as defined in Clause 2.1, consists of two logical phases:
a) Generation: the process of using dsdgen to create data in a format suitable for presentation to the load facility. The generated data may be stored in memory, or in flat files on tape or disk.
b) Loading: the process of storing the generated data to the Database Location.

Generation and loading of the data can be accomplished in two ways:

a) **Load from flat files: dsdgen** is used to generate flat files that are stored in or copied to a location on the SUT or on external storage, which is different from the Database Location, i.e. this data is a copy of the TPC-DS data. The records in these files may optionally be permuted and relocated to the SUT or external storage. Before benchmark execution data must be loaded from these flat files into the Database Location. In this case, only the loading into the Database Location contributes to the database load time.

b) **In-line load**: dsdgen is used to generate data that is directly loaded into the Database Location using an "in-line" load facility. In this case, generation and loading occur concurrently and both contribute to the database load time.

Comment: For option a) The TPC-DS data stored in the Database Location must be a full copy of the flat files. I.e. if the flat files were deleted the benchmark could be executed. The reason for this is that the storing of dsdgen data into the Database Location must result in a new copy of the data, i.e. logical copying is not allowed.

7.4.3.3   The resources used to generate, permute, relocate to the SUT or hold dsdgen data may optionally be distinct from those used to run the actual benchmark. For example:
a) For load from flat files, a separate system or a distinct storage subsystem may be used to generate, store and permute dsdgen data into the flat files used for the database load.
b) For in-line load, separate and distinct processing elements may be used to generate and permute data and to deliver it to the Database Location.

7.4.3.4   Resources used only in the generation phase of the population of the test database must be treated as follows:

For load from flat files,

a) Any processing element (e.g., CPU or memory) used exclusively to generate and hold dsdgen data or relocate it to the SUT prior to the load phase shall not be included in the total priced system and shall be physically removed from or made inaccessible to the SUT prior to the start of the Load Testusing vendor supported methods;
b) Any storage facility (e.g., disk drive, tape drive or peripheral controller) used exclusively to generate and deliver data to the SUT during the load phase shall not be included in the total priced system. The test sponsor must demonstrate to the satisfaction of the auditor that this facility is not being used in the Performance Tests.

For in-line load, any processing element (e.g., CPU or memory) or storage facility (e.g., disk drive, tape drive or peripheral controller) used exclusively to generate and deliver dsdgen data to the SUT during the load phase shall not be included in the total priced system and shall be physically removed from or made inaccessible to the SUT prior to the start of the Performance Tests.

Comment: The intent is to isolate the cost of resources required to generate data from those required to load data into the Database Location.

7.4.3.5   An implementation may require additional programs to transfer dsdgen data into the database tables (from either flat file or in-line load). If non-commercial programs are used for this purpose, their source code must be disclosed. If commercially available programs are used for this purpose, their vendors and configurations shall be disclosed. Whether or not the software is commercially available, use of the software's functionality's shall be limited to:
1. Permutation of the data generated by dsdgen ;
2. Delivery of the data generated by dsdgen to the data processing system.

7.4.3.6   The database load must be implemented using commercially available utilities (invoked at the command level or through an API) or an SQL programming interface (such as embedded SQL or ODBC).

7.4.3.7   Database Load Time

7.4.3.7.1   The elapsed time to prepare the Test Database for the execution of the performance test is called the Database Load Time (T_{LOAD}), and must be disclosed. It includes all of the elapsed time to create the tables defined in Clause 2.1, load data, create and populate EADS, define and validate constraints, gather statistics for the test database, configure the system under test to execute the performance test, and to ensure that the test database meets the data accessibility requirements including syncing loaded data on RAID devices and the taking of a backup of the data processing system, when necessary.

7.4.3.8   The Database Load Time, known as T_{LOAD} is the difference between Load Start Time and Load End Time.

- Load Start Time is defined as the timestamp taken at the start of the creation of the tables defined in Clause 2.1 or when the first character is read from any of the flat files or, in case of in-line load, when the first character is generated by dsdgen, whichever happens first
- Load End Time is defined as the timestamp taken when the Test Database is fully populated, all EADS are created, a database backup has completed (if applicable) and the SUT is configured, as it will be during the performance test

Comment: Since the time of the end of the database load is used to seed the random number generator for the substitution parameters, that time cannot be delayed in any way that would make it predictable to the test sponsor.

7.4.3.8.1 There are five classes of operations which may be excluded from database load time:

a) Any operation that does not affect the state of the data processing system (e.g., data generation into flat files, relocation of flat files to the SUT, permutation of data in flat files, operating-system-level disk partitioning or configuration);

b) Any modification to the state of the data processing system that is not specific to the TPC-DS workload (e.g. logical tablespace creation or database block formatting);

c) The time required to install or remove physical resources (e.g. CPU, memory or disk) on the SUT that are not priced;

d) An optional backup of the test database performed at the test sponsor's discretion. However, if a backup is required to ensure that the data accessibility properties can be met, it must be included in the load time;

e) Operations that create RAID devices.

f) Tests required to fulfill data validation test (see Clause 3.5)

7.4.3.8.2 There cannot be any manual intervention during the Database Load.

7.4.3.8.3 The SUT or any component of it must not be restarted after the start of the Load Test and before the start of the Performance Test.

Comment: The intent of this Clause is that when the timing ends the system under test be capable of executing the Performance Test without any further change. The database load may be decomposed into several phases. Database load time is the sum of the elapsed times of all phases during which activity other than that detailed in Clause 7.4.3.8.1 occurred on the SUT.

7.4.4 Power Test

7.4.4.1 The Power Test is executed immediately following the load test.

7.4.4.2 The Power Test measures the ability of the system to process a sequence of queries in the least amount of time in a single stream fashion.

7.4.4.3 The Power Test shall execute queries submitted by the driver through a single query stream with stream identification number 0 and using a single session on the SUT.

7.4.4.4 The queries in the Power Test shall be executed in the order assigned to its stream identification number and defined in 11.9.1.1Appendix D:.

7.4.4.5 Only one query shall be active at any point of time during the Power Test.

7.4.4.6

7.4.5 Power Test Timing

7.4.5.1 The elapsed time of the Power Test, known as T_{Power} is the difference between
- Power Test Start Time, which is the timestamp that must be taken before the first character of the executable query text of the first query of Stream 0 is submitted to the SUT by the driver; and
- Power Test End Time, which is the timestamp that must be taken after the last character of output data from the last query of Stream 0 is received by the driver from the SUT.

- The elapsed time of the Power Test shall be disclosed.

7.4.6   Throughput Tests

7.4.6.1   The Throughput Tests measure the ability of the system to process the most queries in the least amount of time with multiple users.

7.4.6.2   Throughput Test 1 immediately follows the Power Test. The sequencing of Throughput Tests and Data Maintenance Tests is as follows:

- Throughput Test 1 followed by Data Maintenance Test1 followed by Throughput Test 2 followed by Data Maintenance Test 2.

7.4.6.3   Any explicitly created aggregates, as defined in Clause 5.1.4, present and enabled during any portion of Throughput Test 1or 2 must be present and enabled at all times that queries are being processed.

7.4.6.4   Each query stream contains a distinct permutation of the query templates defined for TPC-DS. The permutation of queries for the first 20 query streams is shown in 11.9.1.1Appendix D:.

7.4.6.5   Only one query shall be active on any of the sessions at any point of time during a Throughput Test.

7.4.6.6   The Throughput Test shall execute queries submitted by the driver through a sponsor-selected number of query streams (S_q). There must be one session per query stream on the SUT and each stream must execute queries serially (i.e. one after another).

7.4.6.7   Each query stream is uniquely identified by a **stream identification number** s ranging from 1 to S, where S is the number of query streams in the Throughput Tests (Throughput Test 1 plus Throughput Test 2).

7.4.6.8   Once a stream identification number has been generated and assigned to a given query stream, the same number must be used for that query stream for the duration of the test.

7.4.6.9   The value of S_q is any even number larger than or equal to 4.

7.4.6.10   The same value of S_q shall be used for bothThroughput Tests, and shall remain constant throughout each Throughput Test.

7.4.6.11   The queries in each query stream shall be executed in the order assigned to the stream identification number and defined in 11.9.1.1Appendix D:.

7.4.7   Throughput Test Timing

7.4.7.1   For a given query template **t**, used to produce the **i**[th] query within query stream **s**, the **query elapsed time**, QD(s, i, t), is the difference between:
- The timestamp when the first character of the executable query text is submitted to the SUT by the driver;
- The timestamp when the last character of the output is returned from the SUT to the driver and a success message is sent to the driver.

Comment: All the operations that are part of the execution of a query (e.g., creation and deletion of a temporary table or a view) must be included in the elapsed time of that query.

7.4.7.2   The elapsed time of each query in each stream shall be disclosed for each Throughput Test and Power Test.

7.4.7.3   The elapsed time of Throughput Test 1, known as T_{TT1} is the difference between Throughput Test 1 Start Time and Throughput Test 1 End Time.

7.4.7.4   Throughput Test 1 Start Time, which is the timestamp that must be taken before the first character of the executable query text of the first query stream of Throughput Test 1 is submitted to the SUT by the driver.

7.4.7.5   Throughput Test 1 End Time, which is the timestamp that must be taken after the last character of output data from the last query of the last query stream of Throughput Test 1 is received by the driver from the SUT.

Comment: In this clause a query stream is said to be first if it starts submitting queries before any other query streams. The last query stream is defined to be that query stream whose output data are received last by the driver.

7.4.7.6 The elapsed time of Throughput Test 2, known as T_{TT2} is the difference between Throughput Test 2 Start Time and Throughput Test 2 End Time,

7.4.7.7 Throughput Test 2 Start Time is defined as a timestamp identical to Data Maintenance Test 1 End Time.

7.4.7.8 Throughput Test 2 End Time, which is the timestamp that must be taken after the last character of output data from the last query of the last query stream of Throughput Test 2 is received by the driver from the SUT.

7.4.7.9 The elapsed time of each Throughput Test shall be disclosed.

7.4.8 Data Maintenance Tests

7.4.8.1 The Data Maintenance Tests 1 and 2 measure the ability to perform desired data changes to the TPC-DS data set.

7.4.8.2 Each Data Maintenance Test shall execute $S_q/2$ refresh runs.

7.4.8.3 Each refresh run uses its own data set as generated by dsdgen. Refresh runs must be executed in the order generated by dsdgen.

7.4.8.4 Any explicitly created aggregates, as defined in clause 5.1.4, present and enabled during any portion of Throughput Test 1 must conform to clause 7.4.6.3.

7.4.8.5 Refresh runs do not overlap; at most one refresh run is running at any time. All data maintenance functions need to have finished in refresh run n before any data maintenance function can commence on refresh run n+1.

Comment: Each set of data maintenance functions runs with its own refresh data set. The order of refresh runs is determined by dsdgen.

7.4.8.6 The scheduling of each data maintenance function within refresh runs is left to the test sponsor.

7.4.8.7 The Durable Medium failure required as part of the Data Accessibility Test (Clause 6.1.2) must be triggered during Data Maintenance Test 1 (at some time after the starting timestamp of the first refresh run in Data Maintenance Test 1, and before the ending timestamp of the last refresh run in Data Maintenance Test 2).

7.4.9 Data Maintenance Timing

7.4.9.1 The elapsed time, DI(i,s), for the execution of the data maintenance function ,i, of the sth refresh run (e.g. applying the sth refresh data set on the data maintenance function i), is the difference between:
- The timestamp, DS(i,s), when the first character of the data maintenance function i executing in refresh run s is submitted to the SUT by the driver, or when the first character requesting the execution of Data Maintenance function i is submitted to the SUT by the driver, whichever happens first;
- The timestamp, DE(i,s) end time, when the last character of output data from the last data maintenance function of the last refresh run is received by the driver from the SUT and a success message has been received by the driver from the SUT.

7.4.9.2 The elapsed time, DI(s), for the execution of all data maintenance functions of refresh run s is the difference between the start timestamp of refresh run s, DS(s) and the end timestamp of refresh run s, DE(s). DS(s) is defined as DS(i,s), where i denotes the first data maintenance function executed in refresh run s. DE(s) is defined as DS(j,s), where j is the last data maintenance function executed in refresh run s.

7.4.9.3 The elapsed time of Data Maintenance Test 1, known as T_{DM1} is the difference between Data Maintenance Test 1 Start Time and Data Maintenance Test 1 End Time,

7.4.9.4 Data Maintenance Test 1 Start Time is defined as the starting timestamp DS of the first refresh run in Data Maintenance Test 1.

7.4.9.5 Data Maintenance Test 1 End Time is defined as the ending timestamp DE of the last refresh run in Data Maintenance Test 1, including all EADS updates.

7.4.9.6   The elapsed time of Data Maintenance Test 2, known as T_{DM2} is the difference between Data Maintenance Test 2 Start Time and Data Maintenance Test 2 End Time,

7.4.9.7   Data Maintenance Test 2 Start Time is defined as the starting timestamp DS of the first refresh run in Data Maintenance Test 2.

7.4.9.8   Data Maintenance Test 2 End Time is defined as the ending timestamp DE of the last refresh run in Data Maintenance Test 2, including all EADS updates.

7.4.9.9   The elapsed time of each data maintenance function within each refresh run must be disclosed, i.e. all DI(i,s) must be disclosed.

7.4.9.10   The timestamp of the start and end times and the elapsed time of each refresh run must be disclosed, i.e. for all refresh run s DS(s), DE(s) and DI(s) must be disclosed.

7.5   Output Data

7.5.1   After execution, a query returns one or more rows. The rows are called the output data.

7.5.2   Output data shall adhere to the following guidelines:

a) Columns appear in the order specified by the SELECT list of the query.
b) Column headings are optional.
c) Non-integer expressions including prices are expressed in decimal notation with at least two digits behind the decimal point.
d) Integer quantities contain no leading zeros.
e) Dates are expressed in a format that includes the year, month and day in integer form, in that order (e.g., YYYY-MM-DD). The delimiter between the year, month and day is not specified. Other date representations, for example the number of days since 1970-01-01, are specifically not allowed.
f) Strings are case-sensitive and must be displayed as such. Leading or trailing blanks are acceptable.
g) The amount of white space between columns is not specified.
h) 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.
i) NULLs must always be printed by the same string pattern of zero or more characters.

Comment: The intent of this clause is to assure that output data is expressed in a format easily readable by a non-sophisticated computer user, and can be compared with known output data for query validation.

Comment: Since the reference answer set provided in the specification originated from different data processing systems, the reference answer set does not consistently express NULL values with the same string pattern.

7.5.3   The precision of all values contained in the output data shall adhere to the following rules:

a) For singleton column values and results from COUNT aggregates, the values must exactly match the query validation output data.
b) For ratios, results must be within 1% of the query validation output data when reported to the nearest 1/100th, rounded up.
c) For results from SUM money aggregates, the resulting values must be within $100 of the query validation output data.
d) For results from AVG aggregates, the resulting values must be within 1% of the query validation output data when reported to the nearest 1/100th, rounded up.

7.6   Metrics

7.6.1   TPC-DS defines three primary metrics:

a) A Performance Metric, QphDS@SF, reflecting the TPC-DS query throughput (see Clause 7.6.3);
b) A Price-Performance metric, $/QphDS@SF (see Clause 7.6.4);
c) System availability date (see Clause 7.6.5).

7.6.2 TPC-DS also defines several secondary metrics. The secondary metrics are:

a) Load time, as defined in Clause 7.4.3.7;
b) Power Test Elapsed time as defined in Clause 7.4.4 and the elapsed time of each query in the Power Test;
c) Throughput Test 1 and Throughput Test 2 elapsed times, as defined in clauses 7.4.7.3 and 7.4.7.6.
d) When TPC_Energy option is chosen for reporting, the TPC-DS energy metric reports the power per performance and is expressed as Watts/QphDS@SF. (see TPC-Energy specification for additional requirements).

Each secondary metric shall be referenced in conjunction with the scale factor at which it was achieved. For example, Load Time references shall take the form of Load Time @ SF, or "Load Time = 10 hours @ 1000".

7.6.3 The Performance Metric (QphDS@SF)

7.6.3.1 The primary performance metric of the benchmark is QphDS@SF, defined as:

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

Where:

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

7.6.3.2

Comment: The floor symbol ($\lfloor \quad \rfloor$) in the above equation truncates any fractional part.

7.6.4 The Price Performance Metric ($/QphDS@SF)

7.6.4.1 The price-performance metric for the benchmark is defined as:

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

Where:

P is the price of the Priced System as defined in Clause 9.1.1.

QphDS@SF is the reported performance metric as defined in Clause 7.6.3

7.6.4.2   If a benchmark configuration is priced in a currency other than US dollars, the units of the price-performance metrics may be adjusted to employ the appropriate currency.

7.6.5   The System Availability Date, as defined in the TPC Pricing Specification Version 1 must be disclosed in any references to either the performance or price-performance metric of the benchmark.

7.6.6   Fair Metric Comparison

7.6.6.1   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 may not scale down linearly with a decrease in database size due to configuration changes required by changes in database size.

If results measured against different database sizes (i.e., with different scale factors) appear in a printed or electronic communication, then each reference to a result or metric must clearly indicate the database size against which it was obtained. In particular, all textual references to TPC-DS metrics (performance or price/performance) appearing must be expressed in the form that includes the size of the test database as an integral part of the metric's name; i.e. including the "@size" 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 test database size 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 TPC-DS results measured against different database sizes are misleading and discourages such comparisons".

7.6.6.2   Any TPC-DS result is comparable to other TPC-DS results regardless of the number of query streams used during the test (as long as the scale factors chosen for their respective test databases were the same).

7.6.7   Required Reporting Components

To be compliant with the TPC-DS standard and the TPC's fair use policies, all public references to TPC-DS results for a given configuration must include the following components:

- The size of the test database, expressed separately or as part of the metric's names (e.g., QphDS@10GB);
- The TPC-DS Performance Metric, QphDS@Size;
- The TPC-DS Price/Performance metric, $/QphDS@Size;
- The Availability Date of the complete configuration (see TPC Pricing Specification located on the TPC website (http://www.tpc.org).

Following are two examples of compliant reporting of TPC-DS results:

Example 1: At 10GB the RALF/3000 Server has a TPC-DS Query-per-Hour metric of 3010 when run against a 10GB database yielding a TPC-DS Price/Performance of $1,202 per query-per-hour and will be available 1-Apr-06.

Example 2: The RALF/3000 Server, which will start shipping on 1-Apr-06, is rated 3,010 QphDS@10GB and 1202 $/QphDS@10GB.

8   SUT AND DRIVER IMPLEMENTATION

This clause defines the System Under Test (SUT) and the benchmark driver.

8.1   Models of Tested Configurations

8.1.1   The tested and reported configuration(s) is composed of a driver that submits queries to a system under test (SUT). The SUT executes these queries and replies to the driver. The driver resides on the SUT hardware and software.

8.1.2   Figure 8-1 illustrates examples of driver/SUT configurations. The driver is the shaded area. The diagram also depicts the driver/SUT boundary (see Clause 7.1.16 and Clause 7.4) where timing intervals are measured.
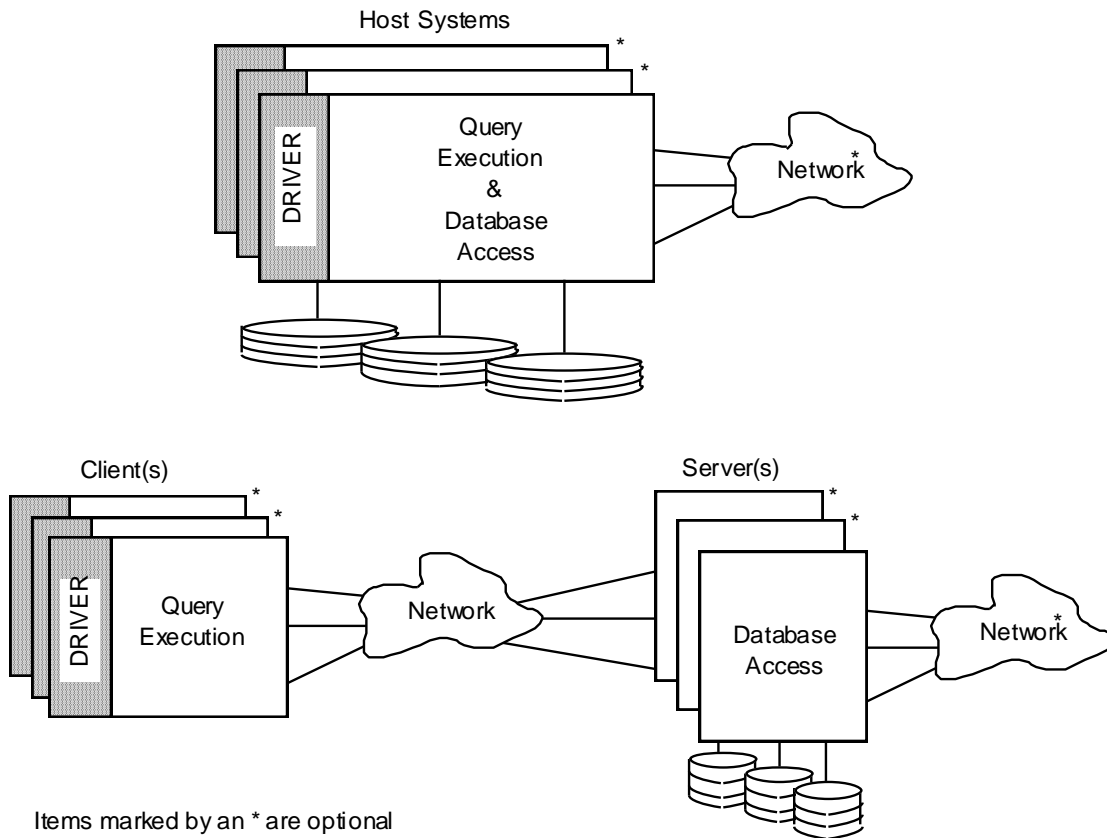
Figure 8-1: Two driver/SUT configurations, a "host-based" and a "client/server" configuration

8.2   System Under Test (SUT) Definition

8.2.1   The SUT consists of:

   a) The host system(s) or server(s), including hardware and software supporting access to the database employed in the performance test and whose cost and performance are described by the benchmark metrics
   b) Any client processing units (e.g., front-end processors, workstations, etc.) used to execute the queries
   c) The hardware and software components needed to communicate with user interface devices
   d) The hardware and software components of all networks required to connect and support the SUT components

e) Data storage media sufficient to satisfy scaling rules in Clause 3, data accessibility properties in Clause 6.1 and data described in Clause 7.5.

8.2.2   All SUT components, as described in Clause 8.2.1, shall be commercially available software or hardware products.

8.2.3   An implementation-specific layer can be implemented on the SUT. This layer shall be logically located between the driver and the SUT, as depicted by Figure 8-2.

Figure 8-2: Implementation Specific Layer
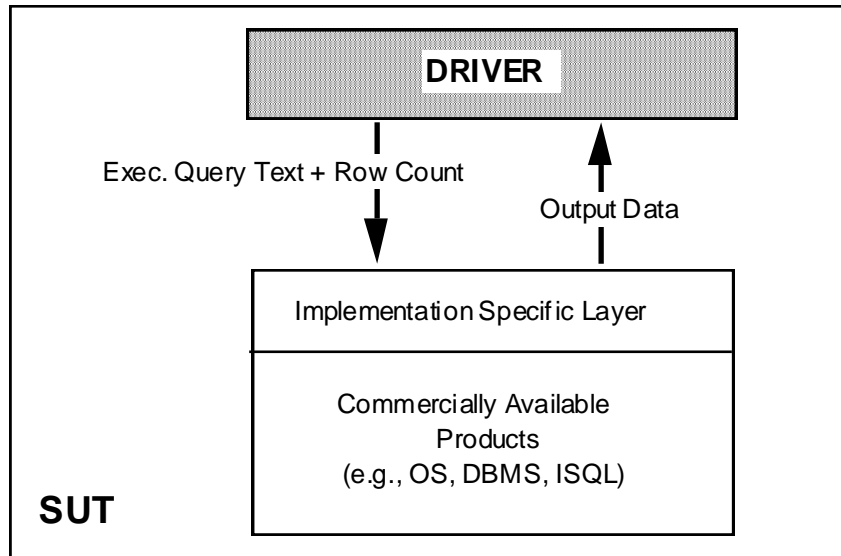
8.2.4   If present on the SUT, an implementation-specific layer, shall be minimal and general purpose (i.e., not limited to the TPC-DS queries). The source code shall be disclosed. The functions performed by an implementation specific layer shall be strictly limited to the following:

a) Database transaction control operations before and after each query execution
b) Cursor control and manipulation operations around the executable query text
c) Definition of procedures and data structures required to process dynamic SQL, including the communication of the executable query text to the commercially available layers of the SUT and the reception of the query output data
d) Communication with the commercially available layers of the SUT
e) Buffering of the query output data
f) Communication with the drivere it

The following are examples of functions that the implementation-specific layer shall not perform:
a) Any modification of the executable query text;
b) Any use of stored procedures to execute the queries;
c) Any sorting or translation of the query output data;
d) Any function prohibited by the requirements of Clause 7.2.8.1.

8.3   Driver Definition

8.3.1   The driver presents the workload to the SUT. The driver is a logical entity that can be implemented using one or more programs, processes, or systems. The driver shall perform only the following functions:

a) Generate a unique stream ID, starting with 1 for each query stream
b) Sequence queries for execution by the query
c) Activate, schedule, and/or synchronize the execution of data maintenance functions

d) Generate the executable query text for each query

e) Generate values for the substitution parameters of each query

f) Complete the executable query text by replacing the substitution parameters by the values generated for them and, if needed, replacing the text-tokens by the query stream ID

g) Submit each complete executable query text to the SUT for execution, including the number of rows to be returned when specified by the functional query definition

h) Submit each data maintenance function to the SUT for execution

i) Receive the output data resulting from each query execution from the SUT

j) Measure the execution times of the queries and the data maintenance functions and compute measurement statistics

k) Maintain an audit log of query text and query execution output

8.3.2   The generation of executable query text used by the driver to submit queries to the SUT does not need to occur on the SUT and does not have to be included in any timing interval.

8.3.3   The driver shall not perform any function other than those described in Clause 8.3.1. Specifically, the driver shall not perform any of the following functions:

a) Performing, activating, or synchronizing any operation other than those mentioned in Clause 8.3.1

b) Delaying the execution of any query after the execution of the previous query other than for delays necessary to process the functions described in Clause 8.3.1. This delay must be reported and can not exceed half a second between any two consecutive queries of the same query stream

c) Modifying the compliant executable query text prior to its submission to the SUT

d) Embedding the executable query text within a stored procedure definition or an application program

e) Submitting to the SUT the values generated for the substitution parameters of a query other than as part of the executable query text submitted

f) Submitting to the SUT any data other than the instructions to execute the data maintenance functions, the compliant executable query text and, when specified by the functional query definition, the number of rows to be returned

g) Artificially extending the execution time of any query.

8.3.4   The driver is not required to be priced.

9   PRICING

This section defines the components, functional requirements of what is priced, and what substitutions are allowed. Rules for pricing the Priced Configuration and associated software and maintenance are included in the current revision of the TPC Pricing Specification Version 1 located on the TPC website (http://www.tpc.org).

9.1   Priced System

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

9.1.1   System Under Test

Calculation of the priced system consists of:

- Price of the SUT as tested and defined in Clause 8;
- Price of a communication interface capable of supporting the required number of user interface devices defined in Clause 8;
- Price of on-line storage for the database as described in Clause 9.1.3 and storage for all software included in the priced configuration;
- 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 executable query texts or necessary to create and populate the test database.

Specifically excluded from the priced system calculation are:

- End-user communication devices and related cables, connectors, and concentrators;
- Equipment and tools used exclusively in the production of the full disclosure report;
- Equipment and tools used exclusively for the execution of the **dsdgen** or **dsqgen** (see Appendix F) programs.0

9.1.2   User Interface Devices and Communications

9.1.2.1   The priced system must include the hardware and software components of a communication interface capable of supporting a number of user interface devices (e.g., terminals, workstations, PCs, etc.) at least equal to 10 times the minimum number of query streams or the actual number of query streams, whichever is greater.

Comment: Test sponsors are encouraged to configure the SUT with a general-purpose communication interface capable of supporting a large number of user interface devices.

9.1.2.2   Only the interface is to be priced. Not to be included in the priced system are the user interface devices themselves and the cables, connectors and concentrators used to connect the user interface devices to the SUT. For example, in a configuration that includes an Ethernet interface to communicate with PCs, the Ethernet card and supporting software must be priced, but not the Ethernet cables and the PCs.

Comment: Active components (e.g., workstations, PCs, concentrators, etc.) can only be excluded from the priced system under the assumption that their role is strictly limited to submitting executable query text and receiving output data and that they do not participate in the query execution. All query processing performed by the tested configuration is considered part of the performance test and can only be done by components that are included in the priced system.

9.1.2.3   The communication interface used must be an industry standard interface, such as Ethernet, Token Ring, or RS232.

9.1.2.4   The following diagram illustrates the boundary between what is priced (on the right) and what is not (on the left). In the event the driver is a commercial product its price should not be included in the price of the SUT:
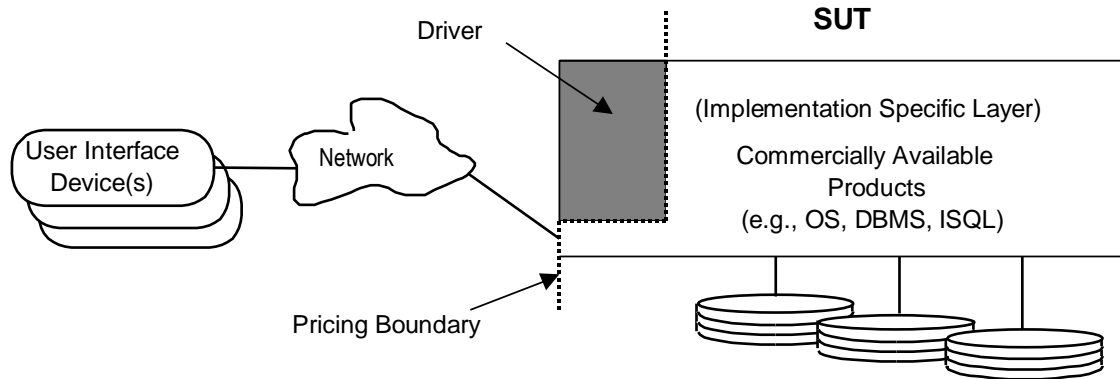
Figure 9-1: Pricing Boundary

9.1.3   Database Storage

9.1.3.1   The storage that is required to be priced includes:
- storage required to execute the benchmark;
- storage and media needed to assure that the test database meets the data accessibility requirements;
- storage used to hold optional staging area data.

9.1.3.2   All storage required for the priced system must be present on the tested system.

9.1.4   Additional Operational Components

9.1.4.1   Additional products that might be included on a customer installed configuration, such as operator consoles and magnetic tape drives, are also to be included in the priced system if explicitly required for the operation, administration, or maintenance, of the priced system.

9.1.4.2   Copies of the software, on appropriate media, and a software load device, if required for initial load or maintenance updates, must be included.

9.1.4.3   The price of an Uninterruptible Power Supply, if specifically contributing to a durability solution, must be included.

9.1.4.4   The price of all cables used to connect components of the system (except as noted in Clause9.1.2.2) must be included.

9.2   Allowable Substitution

9.2.1   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 version 1 or when the Part Number for a component changes.

9.2.2   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 TPC-DS Performance Metric. All Substitutions must be reported in the Report and noted in the Auditor's Attestation Letter. The following hardware component may be substituted:

- Durable Medium

10   FULL DISCLOSURE

10.1   Reporting Requirements

10.1.1   A Full Disclosure Report (FDR) is required for a benchmark publication. The FDR is a zip file of a directory structure containing the following:

10.1.2   A Report in Adobe Acrobat PDF format,

10.1.3   An Executive Summary Statement (ES) in Adobe Acrobat PDF format,

10.1.4   An XML document ("ES.xml") with approximately the same information as in the Executive Summary Statement,

10.1.5   The Supporting Files consisting of various source files, scripts, and listing files.

10.1.6   Requirements for the FDR file directory structure are described below.

10.1.7   The intent of this disclosure is to simplify comparison between results and for a customer to be able to replicate the results of this benchmark given appropriate documentation and products.

10.2   Format Guidelines

10.2.1   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.

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

10.2.3   The Executive Summary must be included near the beginning of the full disclosure report.

10.2.4   The directory structure of the FDR has three folders:

- *ExecutiveSummaryStatement* - contains the Executive Summary Statement and ES.xml
- *Report* - contains the Report,
- *SupportingFiles* - contains the Supporting Files.

10.3   Full Disclosure Report Contents

The FDR should be sufficient to allow an interested reader to evaluate and, if necessary, recreate an implementation of TPC-DS. If any sections in the FDR refer to another section of the report (e.g., an appendix), the names of the referenced scripts/programs must be clearly labeled in each section.

Comment: Since the building of a database 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.

The order and titles of sections in the test sponsor's full disclosure report must correspond with the order and titles of sections from the TPC-DS standard specification (i.e., this document).

10.3.1   General Items

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

10.3.1.2 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:
a) Database tuning options;
b) Optimizer/Query execution options;
c) Query processing tool/language configuration parameters;
d) Recovery/commit options;

e) Consistency/locking options;

f) Operating system and configuration parameters;

g) Configuration parameters and options for any other software component incorporated into the pricing structure;

h) Compiler optimization options.

Comment: 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: 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.

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

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

a) Number and type of processors (including size of L2 cache);

b) Size of allocated memory, and any specific mapping/partitioning of memory unique to the test;

c) Number and type of disk units (and controllers, if applicable);

d) Number of channels or bus connections to disk units, including their protocol type;

e) Number of LAN (e.g., Ethernet) connections, including routers, workstations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure;

f) Type and the run-time execution location of software components (e.g., data processing system, query processing tools/languages, middleware components, software drivers, etc.).

The following sample diagram illustrates a measured benchmark configuration using Ethernet, an external driver, and four processors in the SUT. Note that this diagram does not depict or imply any optimal configuration for the TPC-DS benchmark measurement.

Cluster of 4 Systems
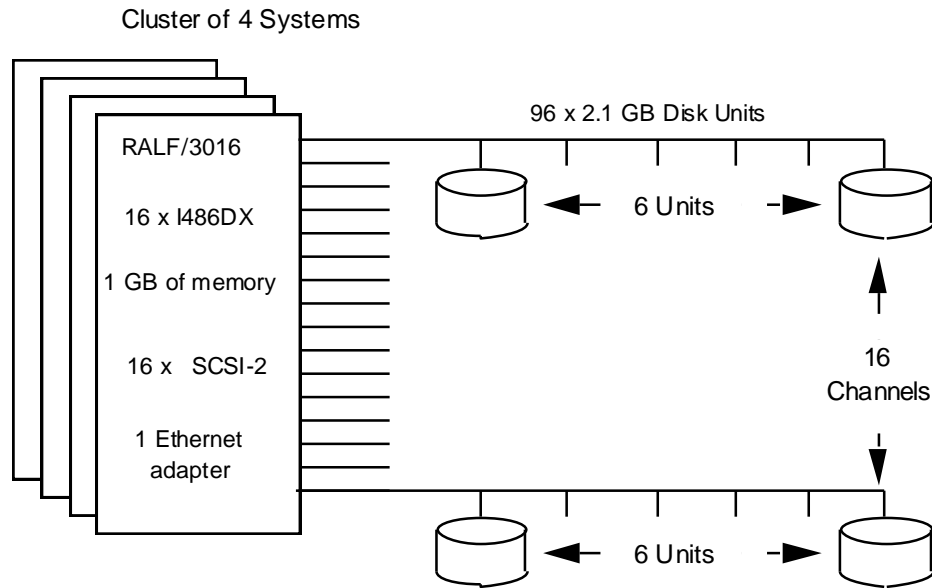
LAN: Ethernet using NETplus routers
CPU: 16 x a243DX 50MHz with 256 KByte Second Level Cache
        1 gigabyte of main memory
        4 x SCSI-2 Fast Controllers
Disk: 96 x 2.1 gigabyte SCSI-2 drives

Figure 10-1: Sample Configuration Diagram

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.

10.3.2   Clause 2- Logical Database Design Related Items

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

10.3.2.2   The physical organization of tables and indices within the test and qualification databases must be disclosed. If the column ordering of any table is different from that specified in Clause2.3 or 2.4,, it must be noted.

Comment: The concept of physical organization includes, but is not limited to: record clustering (i.e., rows from different logical tables are co-located on the same physical data page), index clustering (i.e., rows and leaf nodes of an index to these rows are co-located on the same physical data page), and partial fill-factors (i.e., physical data pages are left partially empty even though additional rows are available to fill them).

10.3.2.3   If any directives to DDLs are used to horizontally partition tables and rows in the test and qualification databases, these directives, DDLs, and other details necessary to replicate the partitioning behavior must be disclosed.

10.3.2.4   Any replication of physical objects must be disclosed and must conform to the requirements of Clause 2.5.3.

10.3.3   Clause 3 - Scaling and Database Population Related Items

10.3.3.1   The cardinality (e.g., the number of rows) of each table of the test database, as it existed at the completion of the database load (see Clause 7.1.2) must be disclosed.

10.3.3.2   The distribution of tables and logs across all media must be explicitly described using a format similar to that shown in the following example for both the tested and priced systems.

Comment: Detailed diagrams for layout of database tables on disks can widely vary, and it is difficult to provide exact guidelines suitable for all implementations. The intent is to provide sufficient detail to allow independent reconstruction of the test database. The figure that follows is an example of database layout descriptions and is not intended to describe any optimal layout for the TPC-DS database.

Controller	Disk Drive	Description of Content
40A	0	Operating system, root
	1	System page and swap
	2	Physical log
	3	100% of store_sales and store tables
40B	0	33% of store_sales, catalog_sales and catalog_returns tables
	1	33% of store_sales, catalog_sales and catalog_returns tables
	2	34% of store_sales, catalog_sales and catalog_returns tables
	3	100% of date_dim, time_dim and reason tables

10.3.3.3   Figure 10-2: Sample Database Layout Description

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

Comment: The intent is to provide sufficient detail about partitioning and replication to allow independent reconstruction of the test database.

10.3.3.5   Implementations may use some form of RAID. The RAID level used must be disclosed for each device. If RAID is used in an implementation, the logical intent of its use must be disclosed. Three levels of usage are defined:
    a) Base tables only: In this case only the Base Tables (see Clause 2) are protected by any form of RAID;
    b) Base tables and EADS: in addition to the protection of the base tables, implementations in this class must also employ RAID to protect all EADS;

c) Everything: implementations in this usage category must employ RAID to protect all database storage, including temporary or scratch space in addition to the base tables and EADS.

10.3.3.6   The version number (i.e., the major revision number, the minor revision number, and third tier number) of dsdgen must be disclosed. Any modifications to the dsdgen source code (see Appendix B:) must be disclosed. In the event that a program other than dsdgen was used to populate the database, it must be disclosed in its entirety.

10.3.3.7   The database load time for the test database (see Clause 7.4.3.7) must be disclosed.

10.3.3.8   The data storage ratio must be disclosed. It is computed by dividing the total data storage of the priced configuration (expressed in GB) by SF corresponding to the scale factor chosen for the test database as defined in Clause 3.1. The ratio must be reported to the nearest 1/100th, rounded up. For example, a system configured with 96 disks of 2.1 GB capacity for a 100GB test database has a data storage ratio of 2.02.

Comment: For the reporting of configured disk capacity, gigabyte (GB) is defined to be 2^30 bytes. Since disk manufacturers typically report disk size using base ten (i.e., GB = 10^9), it may be necessary to convert the advertised size from base ten to base two.

10.3.3.9   The details of the database load must be disclosed, including a block diagram illustrating the overall process. Disclosure of the load procedure includes all steps, scripts, input and configuration files required to completely reproduce the test and qualification databases.

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

10.3.4   Clause 4 and 5 - Query and Data Maintenance -Related Items

10.3.4.1   The query language used to implement the queries must be identified (e.g., "RALF/SQL-Plus").

10.3.4.2   The method of verification for the random number generation must be described unless the supplied dsdgen and dsqgen were used.

10.3.4.3   The method used to generate values for substitution parameters must be disclosed. The version number (i.e., the major revision number, the minor revision number, and third tier number) of dsqgen must be disclosed..

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

Comment: For query output of more than 10 rows, only the first 10 need to be disclosed in the FDR. The remaining rows must be made available upon request.

10.3.4.5   All the query substitution parameters used during the performance test must be disclosed in tabular format, along with the seeds used to generate these parameters.

10.3.4.6   All query and refresh session initialization parameters, settings and commands must be disclosed (see Clauses 7.2.2 through 7.2.7).

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

10.3.4.8   Any object created in the staging area (see Clause 5.1.8 for definition and usage restrictions) used to implement the data maintenance functions must be disclosed. Also, any disk storage used for the staging area must be priced, and any mapping or virtualization of disk storage must be disclosed.

10.3.5   Clause 6– Data Persistence Properties Related Items

10.3.5.1   The results of the data accessibility tests must be disclosed along with a description of how the data accessibility requirements were met. This includes disclosure of the code written to implement the data accessibility Query.

10.3.6   Clause 7- Performance Metrics and Execution Rules Related Items

10.3.6.1   Any system activity on the SUT that takes place between the conclusion of the load test and the beginning of the performance test must be fully disclosed including listings of scripts or command logs.

10.3.6.2   The details of the steps followed to implement the performance test must be disclosed.

10.3.6.3   The timing intervals defined in Clause 7 must be disclosed.

10.3.6.4   For each Throughput Test, the minimum, the 25th percentile, the median, the 75th percentile, and the maximum times for each query shall be reported.

10.3.6.5   The start time and finish time for each query stream must be reported.

10.3.6.6   The start time and finish time for each data maintenance function in the refresh run must be reported for the Throughput Tests.

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

10.3.7   Clause 8 - SUT and Driver Implementation Related Items

10.3.7.1   A detailed textual description of how the driver performs its functions, how its various components interact and any product functionalities or environmental settings on which it relies must be provided. All related source code, scripts and configuration files must be disclosed. The information provided should be sufficient for an independent reconstruction of the driver.

10.3.7.2   If an implementation specific layer is used, then a detailed description of how it performs its functions, how its various components interact and any product functionalities or environmental setting on which it relies must be provided. All related source code, scripts and configuration files must be disclosed. The information provided should be sufficient for an independent reconstruction of the implementation specific layer.

10.3.7.3   If profile-directed optimization as described in Clause 7.2.10 is used, such use must be disclosed. In particular, the procedure and any scripts used to perform the optimization must be disclosed.

10.3.8   Clause 9 - Pricing Related Items

10.3.8.1   A detailed list of hardware and software used in the priced system must be reported. The rules for pricing are included in the current revision of the TPC Pricing Specification located on the TPC website (http://www.tpc.org).

10.3.8.2   The System Availability Date (see Clause 7.6.5) must be the single availability date reported on the first page of the executive summary. The full disclosure report must report Availability Dates individually for at least each of the categories for which a pricing subtotal must be. All Availability Dates required to be reported must be disclosed to a precision of 1 day, but the precise format is left to the test sponsor.

              Comment: A test sponsor may disclose additional detail on the availability of the system's components in the Notes section of the Executive Summary and may add a footnote reference to the System Availability Date.

10.3.8.3   Additional Clause 7 related items may be included in the full disclosure report for each country specific priced configuration.

10.3.9   Clause 11 - Audit Related Items

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

10.4   Executive Summary

The executive summary is meant to be a high level overview of a TPC-DS 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.

The executive summary has three components:

Implementation and Cost of Ownership Overview

Pricing Spreadsheet

Numerical Quantities

10.4.1   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

a) 1/2 inch margins, top and bottom;
b) 3/4 inch left margin, 1/2 inch right margin;
c) 2 pt. frame around the body of the page. All interior lines should be 1 pt;
d) Sponsor identification and System identification, each set apart by a 1 pt. rule, in 16-20 pt. Times Bold font.
e) TPC-DS, TPC-Pricing, TPC-Energy (if reported), with three tier versioning (e.g., 1.2.3), and report date, separated from other header items and each other by a 1 pt. Rule, in 9-12 pt. Times font.

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

Comment: The report date must be disclosed with a precision of 1 day. The precise format is left to the test sponsor.

Comment: Appendix E contains a sample executive summary. It is meant to help clarify the requirements in Clause 10.4 and is provided solely as an example.

10.4.2   Implementation Overview

Implementation and Cost of Ownership Overview

The implementation overview page contains six 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.

The middle portion of the page must contain two diagrams, which must be of equal size and fill out the entire space. The left diagram shows the benchmarked configuration and the right diagram shows a pie chart with the percentages of the total time and the total times for the Load Test, Throughput Test 1, and Throughput Test 2.

The next section must contain a synopsis of the SUT's major system 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;
• Main and cache memory sizes;
• Network and I/O connectivity;
• Disk quantity and geometry.

If the implementation used a two-tier architecture, front-end and back-end systems must be detailed separately.

10.4.2.1   The first section contains the results that were obtained from the reported runs of the Performance test.

Title	Quantity	Precisio	Units	Font

		n		
Total System Cost	3 yr. Cost of ownership (See Clause 7)	1	$1	16-20 pt. Bold
TPC-DS Composite Query per Hour Metric	QphDS (see Clause 7.6	0.1	QphDS@nnGB	16-20 pt. Bold
Price/Performance	$/QphDS (see Clause 7.6.4)	1	$/QphDS@nnGB	16-20 pt. Bold

The next section details the system configuration

Title	Quantity	Precision	Units	Font
Data Set Size	Raw data size of test database	1	GB	9-12 pt. Times
Data Processing System	Brand, Software Version of Data Processing System used			9-12 pt. Times
Operating System	Brand, Software Version of OS used			9-12 pt. Times
Other Software	Brand, Software Version of other software components			9-12 pt. Times
System Availability Date	System Availability Date	1 day		9-12 pt. Times
Clustered Or Not	Yes/No			9-12 pt. Times

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

10.4.2.2 The middle portion of the page must contain two diagrams, which must be of equal size and fill out the width of the entire space. The left diagram shows the benchmarked configuration and the right diagram shows a pie chart with the percentages of the total time and the total times for the Load Test, Throughput Test 1 and Throughput Test 2.

10.4.2.3 This section contains the database load and RAID information

Title	Quantity	Precision	Units	Font
Load includes backup	Yes/No	N/A	N/A	9-12 pt. Times
RAID	None / Base tables only / Explicit Auxiliary Data Structures / Everything	N/A	N/A	9-12 pt. Times

10.4.2.4 The next section of the Implementation Overview shall contain a synopsis of the SUT's major components, including:

Node and/or processor count and speed in GHz;

Main and cache memory sizes;

Network and IO connectivity;

Disk quantity and geometry

Total mass storage in the priced system.

If the implementation used a two-tier architecture, front-end and back-end systems should be defined separately.

10.4.2.5 The final section of the Implementation Overview shall contain a note stating:
"Database Size includes only raw data (i.e., no temp, index, redundant storage space, etc.)."

10.4.3 Pricing Spreadsheet

The pricing spreadsheet, required by Clause 10.4, must be reproduced in its entirety. Refer to Appendix E for a sample pricing spreadsheet.

10.4.4 Numerical Quantities Summary

The Numerical Quantities Summary page contains three sets of data.

1. The first set is the number of query streams.
2. The second set contains the Start Date, Start Time, End Date, End Time, and Elapsed Time for:
   - Database Load
   - Power Test
   - Throughput Test 1
   - Data Maintenance Test 1
   - Throughput Test 2
   - Data Maintenance Test 2.
3. The third set is a table which contains the information required by Clause 10.3.6.4 .

10.4.5 ES.xml Requirements

The schema of the ES.xml document is defined by the XML schema document tpcds-es.xsd located on the TPC website (http://www.tpc.org). The ES.xml file must conform to the tpcds-es.xsd (established by XML schema validation).

Comment: The **Sponsor** is responsible for verifying that the ES.xml file they provide in the **Full Disclosure Report** conforms to the TPC-DS XML schema. A validation tool will be provided on the TPC web site to facilitate this verification.

Appendix G describes the structure of the XML schema, defines the individual fields, and explains how to use the schema.

10.5 Availability of the Full Disclosure Report

10.5.1 The full disclosure report must be readily available to the public at a reasonable charge, similar to charges for comparable documents by that test sponsor. The report must be made available when results are made public. In order to use the phrase "TPC Benchmark DS", the full disclosure report must have been submitted electronically to the TPC using the procedure described in the TPC Policies document.

10.5.2 The official full disclosure report must be available in English but may be translated to additional languages.

10.6 Revisions to the Full Disclosure Report

Revisions to the full disclosure documentation shall be handled as follows:

a) Fully documented price changes can be reflected in a new published price/performance. The benchmark need not be rerun to remain compliant.
b) Hardware or software product substitutions within the SUT, with the exception of equipment emulated as allowed under Clause 8, require the benchmark to be re-run with the new components in order to re-establish compliance. For any substitution of equipment emulated during the benchmark, a new demonstration must be provided.
c) The revised report shall be submitted as defined in Clause10.2.1.

Comment: During the normal product life cycle, problems will be uncovered that require changes, sometimes referred to as patches or updates. When the cumulative result of applied changes causes the performance metric (see Clause 7.6.3) to decrease by more than 2% from the reported value, then the test sponsor is required to re-validate the benchmark results.

a) Fully documented price changes can be reflected in a new published price/performance.
b) When cumulative price changes have resulted in a worsening of the reported price/performance by 2% or more the test sponsor must submit revised price/performance results to the TPC within 30 days of the effective date of the price change(s) to remain in compliance. The benchmark need not be re-run to remain in compliance.

Comment: The intent of this Clause is that published price/performance reflect actual current price/performance.

a) A change in the committed availability date for the priced system can be reflected in a new published availability date.
b) A report may be revised to add or delete Clause 9 related items for country-specific priced configurations.
c) Full disclosure report revisions may be required for other reasons as specified in the TPC Policies and Guidelines document, and must be submitted using the mechanisms described therein.

10.7 Derived Results

10.7.1   TPC-DS results can be used as the basis for new TPC-DS results if and only if:

a) The auditor ensures that the hardware and software products are the same as those used in the prior result;
b) The auditor reviews the FDR of the new results and ensures that they match what is contained in the original sponsor's FDR;
c) The auditor can attest to the validity of the pricing used in the new FDR.
d) The intent of this clause is to allow a reseller of equipment from a given supplier to publish under the re-seller's name a TPC-DS result already published by the supplier.

10.8   Supporting Files Index Table

An index for all files required by Clause 10.2.4 **Supporting Files** must be provided in the **Report**. The **Supporting Files** index is presented in a tabular format where the columns specify the following:

- The first column denotes the clause in the TPC Specification
- The second column provides a short description of the file contents
- The third column contains the path name for the file starting at the SupportingFiles directory.

If there are no **Supporting Files** provided then the description column must indicate that there is no supporting file and the path name column must be left blank.

The following table is an example of the **Supporting Files** Index Table that must be **reported** in the **Report**.

Clause	Description	Pathname
Introduction	Database Tunable Parameters	SupportingFiles/Introduction/DBtune.txt
	OS Tunable Parameters	SupportingFiles/Introduction/OStune.txt
Clause 2	Table creation scripts	SupportingFiles/Clause2/createTables.sh
	Index creation scripts	SupportingFiles/Clause2/createIndex.sh
Clause 3	Load transaction scripts	SupportingFiles/Clause3/doLoad.sh
Clause 4		
Clause 5	Data maintenance scripts	SupportingFiles/Clause5/doRefresh.sh
Clause 6	Data Accessibility Scripts	SupportingFiles/Clause6/runACID.sh
	Output of data accessibility tests	SupportingFiles/Clause6/ACID.out
Clause 7		
Clause 8		
Clause 9		

10.9 Supporting Files

The Supporting Files contain human readable and machine executable (i.e., able to be performed by the appropriate program without modification) scripts, executables and source code that are required to recreate the benchmark Result. If there is a choice of using a GUI or a script, then the machine executable script must be provided in the Supporting Files. If no corresponding script is available for a GUI, then the Supporting Files must contain a detailed step by step description of how to manipulate the GUI.

The combination of the following rules shall allow anybody to reproduce the benchmark result.

- All software developed specifically for the benchmark must be included in the supporting files if the software was used to cover the requirements of a clause of the benchmark specification or to conduct a benchmark run with the SUT. This includes machine executable code in the form of scripts (e.g., .sql, .sh, .tcsh, .cmd, or .bat files) or source code (e.g., .cpp, .cc, .cxx, .c files). Specifically developed executables (e.g., .exe files) need to be included unless their source code has been provided in the supporting files with detailed instructions (e.g., make files) how to re-generate the executables from the source for the hardware and operating system used for the benchmark.
- References (e.g., URLs) need to be provided for all software available for general purchase or download which has NOT been developed specifically for the benchmark. The software must be available under the location provided by the references for the time the benchmark is published on the TPC website.
- All command line options used to invoke any of the above programs need to be disclosed. If a GUI is used, detailed instructions on how to navigate the GUI as used to reproduce the benchmark result need to be disclosed.

The directory structure under SupportingFiles must follow the clause numbering from the TPC-DS Standard Specification (i.e., this document). The directory name is specified by the 10.9 third level Clauses immediately preceding the fourth level **Supporting Files** reporting requirements. If there is more than one instance of one type of file, subfolders may be used for each instance. For example if multiple **Tier A** machines were used in the benchmark, there may be a folder for each **Tier A** machine.

File names should be chosen to indicate to the casual reader what is contained within the file. For example, if the requirement is to provide the scripts for all table definition statements and all other statements used to set-up the database, file names of 1, 2, 3, 4 or 5 are unacceptable. File names that include the text "tables", "index" or "frames" should be used to convey to the reader what is being created by the script.

10.9.1 SupportingFiles/Introduction Directory

All scripts required to configure the hardware must be **reported** in the **Supporting Files**.

All scripts required to configure the software must be **reported** in the **Supporting Files**. This includes any **Tunable Parameters** and options which have been changed from the defaults in commercially available products, including but not limited to:

- Database tuning options.
- Recovery/commit options.
- Consistency/locking options.
- **Operating System** and application configuration parameters.
- Compilation and linkage options and run-time optimizations used to create/install applications, OS, and/or databases.
- Parameters, switches or flags that can be changed to modify the behavior of the product.

Comment: This requirement can be satisfied by providing a full list of all parameters and options.

10.9.2 SupportingFiles/Clause2 Directory

Scripts must be provided for all table definition statements and all other statements used to set-up the database. All scripts must be human readable and machine executable (i.e., able to be performed by the appropriate program without modification). All scripts are to be **reported** in the **Supporting Files**.

10.9.3   SupportingFiles/Clause3 Directory

Scripts must be provided for all **dsdgen** invocations used to populate the database with content. All scripts must be human readable and machine executable (i.e., able to be performed by the appropriate program without modification). All scripts are to be **reported** in the **Supporting Files**.

10.9.4   SupportingFiles/Clause4 Directory

The implementation of each query of the benchmark as defined per Clause 4 must be **reported** in the **Supporting Files**. This includes, but is not limited to, the code implementing the queries of this benchmark.

10.9.5   SupportingFiles/Clause5 Directory

Scripts must be provided for all steps used to maintain the database content in order to implement Clause 5. All scripts must be human readable and machine executable (i.e., able to be performed by the appropriate program without modification). All scripts are to be **reported** in the **Supporting Files**.

10.9.6   SupportingFiles/Clause6 Directory

Scripts must be provided for all steps used to validate Clause 6. All scripts must be human readable and machine executable (i.e., able to be performed by the appropriate program without modification). All scripts and the output of the scripts are to be **reported** in the **Supporting Files**.

10.9.7   SupportingFiles/Clause7 Directory

No requirements

10.9.8   SupportingFiles/Clause8 Directory

No requirements

10.9.9   SupportingFiles/Clause9 Directory

No requirements

11   AUDIT

This clause defines the audit requirements for TPC-DS. The auditor needs to ensure that the benchmark under audit complies with the TPC-DS specification. Rules for auditing Pricing information are included in the TPC Pricing Specification located at www.tpc.org. 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.

11.1   General Rules

11.1.1   An independent audit of the benchmark results by a TPC certified auditor is required. 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." In addition, the auditing agency cannot have supplied any performance consulting under contract for the benchmark.

In addition, the following conditions must be met:

a)   The auditing agency cannot be financially related to the sponsor. For example, the auditing agency is financially related if it is a dependent division of the sponsor, the majority of its stock is owned by the sponsor, etc.

b)   The auditing agency cannot be financially related to any one of the suppliers of the measured/priced configuration, e.g., the data processing system supplier, the disk supplier, etc.

11.1.2   The auditor's attestation letter is to be made readily available to the public as part of the full disclosure report. A detailed report from the auditor is not required.

11.1.3   TPC-DS results can be used as the basis for new TPC-DS results if and only if:

The auditor ensures that the hardware and software products are the same as those used in the prior result;

The auditor reviews the FDR of the new results and ensures that they match what is contained in the original sponsor's FDR;

The auditor can attest to the validity of the pricing used in the new FDR.

Comment: The intent of this clause is to allow a reseller of equipment from a given supplier to publish under the re-seller's name a TPC-DS result already published by the supplier.

Comment: In the event that all conditions listed in Clause 11.1.2 are met, the auditor is not required to follow the remaining auditor's check list items from Clause 11.2.

11.1.4   In the event that a remote audit procedure is used in the context of a change-based audit, a remote connection to the SUT must be available for the auditor to verify selected audit items from Clause 9.2

11.2   Auditor's Check List

11.2.1   This clause defines the minimal audit checks that the auditor is required to conduct for TPC-DS. In order for the auditor to ensure that the benchmark under audit complies with the TPC-DS specification the auditor is allowed to ask for additional checks.

11.2.2   Clause 2 Related Items

11.2.2.1   Verify that the data types used for each column are conformant. For example, verify that decimal columns can be incremented by 0.01 from -9,999,999,999.99.

11.2.2.2   Verify that the tables have the required list of columns.

11.2.2.3   Verify that the implementation rules are met by the test database.

11.2.2.4   Verify that the test database meets the data access transparency requirements.

11.2.2.5 Verify that conforming arbitrary data values can be inserted into any of the tables. Examples of verification tests include:

11.2.3 Inserting a row that is a complete duplicate of an existing row except for a distinct primary key;

11.2.4 Inserting a row with column values within the domain of the data type and check constraints but beyond the range of existing values.

11.2.4.1 Ensure that all EADS satisfy the requirements of Clause 2.5.3

11.2.4.2 Verify that the set of EADS that are present and enabled at the end of the load test are the same set that are present and enabled at the end of the performance test as required by Clause 2.5.3.6. A similar check may be performed at any point during the performance test at the discretion of the auditor. Note the method used to verify that this requirement has been met.

[This auditor clause states a requirement that does not appear to be stated before (that no ADS can be created during the test). If such a requirement exists it should be stated in clause 2.]

11.2.4.3 Clause 3 Related Items

11.2.4.4 Verify that the qualification database is properly scaled and populated.

11.2.4.5 Verify that the qualification and test databases were constructed in the same manner so that correct behavior on the qualification database is indicative of correct behavior on the test database.

11.2.4.6 *Note the method u*sed to populate the database (i.e., dsdgen or modified version of dsdgen). Note the version number (i.e., the major revision number, the minor revision number, and third tier number) of dsdgen, and the names of the dsdgen files which have been modified. Verify that the version matches the benchmark specification.

11.2.4.7 Verify that storage and processing elements that are not included in the priced configuration are physically removed or made inaccessible during the performance test using a vendor supported method.

11.2.4.8 Verify that the validation data sets are proven consistent with the data loaded into the database according to clause 3.5.

11.2.4.9 Verify referential integrity in the database after the initial load. Referential Integrity is a data property that can be VERIFIED BY CHECKING THAT EVERY FOREIGN KEY HAS A CORRESPONDING PRIMARY KEY.

11.3 Clause 4 Related Items

11.3.1.1 Verify that the basis for the SQL used for each query is the functional query definition or an approved variant or meets Clause 9.2.3.2.

11.3.1.2 Verify that any deviation in the SQL from either the functional query definition or an approved variant is compliant with the specified minor query modifications. Verify that minor query modifications have been applied consistently to the set of functional query definitions or approved variants used.

11.3.1.3 Verify that the executable query text produces the required output when executed against the qualification database using the validation values for substitution parameters.

11.3.1.4 Note the method used to generate the values for substitution parameters (i.e., dsqgen, modified version of dsqgen, other method). If dsqgen was used, note the version number (i.e., the major revision number, the minor revision number, and third tier number) of dsqgen. Verify that the version matches the benchmark specification.

11.3.1.5 Verify that the generated substitution parameters are correctly generated. For each stream take 10 random queries and verify their substitution values.

11.3.1.6 Verify that no aspect of the system under test, except for the database size, has changed between the demonstration of compliance against the qualification database and the execution of the reported measurements.

11.4   Clause 5 Related Items

11.4.1.1   Verify immediately after the performance test that all EADS that were created as part of the database load are correctly maintained. This test is to be conducted with a script that performs the following two types of subtests:

1. For any index measure the response time for index lookups of two keys, one that was loaded during the database load test and one that was loaded during the data maintenance test. For test Type 1 verify that both keys are in the index and verify that the query response times are not substantially differrent from each other, as would be the case if the index was not maintained. (e.g. having a difference of more than 50%)

2. Create another instance for all non–index EADS using the same directives as used for the original EADS. Verify that the creation of the second instance does not query the original EADS. Verify that their content is logically identical.

11.4.1.2   Verify that the data maintenance functions are implemented according to their definition.

11.4.1.3   Verify that the data maintenance functions update, insert and delete the correct update data. For each data maintenance function in a random stream verify that 2 random rows have been correctly updated, inserted and deleted.

11.4.1.4   Verify that the order of the data maintenance functions is in accordance with Clause 5.

11.4.1.5   Note the method used to execute database maintenance operations.

11.4.2   Verify that the refresh data loaded as part of each data maintenance function is in accordance with Clause 5.2.4

11.5   Clause 6 Related Items

11.5.1.1   Verify that the required data accessibility properties are supported by the system under test as configured for the execution of the reported measurements.

11.6   Clause 7 Related Items

11.6.1.1   Verify that the execution rules are followed for the Load Test, Power Test, Throughput Tests 1 and 2, and Data Maintenance Tests 1 and 2.

11.6.1.2   Verify that the database load time is measured according to the requirements.

11.6.1.3   Verify that the queries are executed against the test database.

11.6.1.4   Verify that the query sequencing rules are followed.

11.6.1.5   Verify that the measurement interval for the Throughput Tests is measured as required.

11.6.1.6   Verify that the method used to measure the timing intervals is compliant.

11.6.1.7   Verify that the metrics are computed as required.

11.6.1.8   Verify that any profile-directed optimization performed by the test sponsor conforms to the requirements of Clause 7.2.

11.6.1.9   Verify the set of EADS that exist at the end of the load test exist and are valid and up to date at the end of the performance test by querying the meta data of the test database before the Power Test and after Throughput Test 2. If there is any doubt that the EADS are not maintained the auditor shall run additional tests.

11.7   Clause 8 Related Items

11.7.1.1   Verify that the driver meets the requirements of Clauses 8.3.

11.8   Clause 9 Related Items

11.8.1.1   Verify that the composition of the SUT is in compliance with the Clause 9 and that its components will be commercially available products according to the current version of TPC pricing specification.

11.8.1.2   Note whether an implementation specific layer is used and verify its compliance with Clause 9.1.

11.8.1.3   Verify that all required components of the SUT are priced according to the current version of TPC pricing specification.

11.8.1.4   Verify that a user communication interface is included in the SUT.

11.8.1.5   Verify that all required maintenance is priced according to the current version of TPC pricing specification.

11.8.1.6   Verify that any discount used is generally available and complies according to the current version of TPC pricing specification.

11.8.1.7   Verify that any third-party pricing complies with the requirements of TPC pricing specification.

11.8.2   Verify that the pricing spreadsheet includes all hardware and software licenses, warranty coverage, and additional maintenance costs as required according to the current version of TPC pricing specification.

Comment: Since final pricing for new products is typically set very close to the product announcement date, the auditor is not required to verify the final pricing of the tested system.

11.9   Clause 10 Related Items

11.9.1.1   Verify that major portions of the full disclosure report are accurate and comply with the reporting requirements. This includes:
- The executive summary;
- The numerical quantity summary;
- The diagrams of both measured and priced configurations;
- The block diagram illustrating the database load process.

Appendix A: Logical Representation of the Refresh Data Set

A.1   Refresh Data Set DDL

The following DDL statements define a detailed structure of the flat files, generated by dsdgen, that constitute the refresh data set. The datatypes correspond to those in Clause 2.2.

Table A-1: Column definition s_zip_to_gmt

Column	Datatype	NULLs	Foreign Key
zipg_zip	char(5)	N	
zipg_gmt_offset	integer	N	

Table A-2: Column definition s_purchase_lineitem

Column	Datatype	NULLs	Foreign Key
plin_purchase_id	identifier	N	
plin_line_number	integer	N	
plin_item_id	char(16)		i_item_id
plin_promotion_id	char(16)		p_promo_id
plin_quantity	integer		
plin_sale_price	numeric(7,2)		
plin_coupon_amt	numeric(7,2)		
plin_comment	char(100		

Table A-3: Column definition s_customer

Column	Datatype	NULLs	Foreign Key
cust_customer_id	identifier	N	c_customer_id
cust_salutation	char(10)		
cust_last_name	char(20)		
cust_first_name	char(20)		
cust_preffered_flag	char(1)		
cust_birth_date	char(10)		
cust_birth_country	char(20)		
cust_login_id	char(13)		
cust_email_address	char(50)		
cust_last_login_chg_date	char(10)		
cust_first_shipto_date	char(10)		
cust_first_purchase_date	char(10)		
cust_last_review_date	char(10)		
cust_primary_machine_id	char(15)		
cust_secondary_machine_id	char(15)		
cust_street_number	char(10),		
cust_suite_number	char(10)		
cust_street_name1	char(30)		
cust_street_name2	char(30)		
cust_street_type	char(15)		
cust_city	char(60)		
cust_zip	char(10)		
cust_county	char(30)		
cust_state	char(2)		
cust_country	char(20)		
cust_loc_type	char(20)		
cust_gender	char(1)		cd_gender
cust_marital_status	char(1)		cd_marital_status
cust_educ_status	char(20)		cd_education_status
cust_credit_rating	char(10)		cd_credit_rating
cust_purch_est	numeric(7,2)		cd_purchase_estimate
cust_buy_potential	char(15)		hd_buy_potential
cust_depend_cnt	integer		cd_dep_count
cust_depend_emp_cnt	integer		cd_dep_employed_count
cust_depend_college_cnt	integer		cd_dep_college_count
cust_vehicle_cnt	integer		hd_vehicle_count
cust_annual_income	numeric(9,2)		ib_lower_bound, ib_upper_bound

Table A-4: Column definition s_purchase

Column	Datatype	NULLs	Foreign Key
purc_purchase_id	identifier	N	plin_purchase_id
purc_store_id	char(16)		s_store_id
purc_customer_id	char(16)		s_customer_id
purc_purchase_date	char(10)		d_date
purc_purchase_time	integer		t_time
purc_register_id	integer		
purc_clerk_id	integer		
purc_comment	char(100)		

Table A-5: Column definition s_catalog_order

Column	Datatype	NULLs	Foreign Key
cord_order_id	identifier	N	clin_order_id
cord_bill_customer_id	char(16)		c_customer_id
cord_ship_customer_id	char(16)		c_customer_id
cord_order_date	char(10)		d_date
cord_order_time	integer		t_time
cord_ship_mode_id	char(16)		sm_ship_mode_id
cord_call_center_id	char(16)		cc_call_center_id
cord_order_comments	varchar(100)		

Table A-6: Column definition s_web_order

Column	Datatype	NULLs	Foreign Key
word_order_id	identifier	N	wlin_order_id
word_bill_customer_id	char(16)		c_customer_id
word_ship_customer_id	char(16)		c_customer_id
word_order_date	char(10)		d_date
word_order_time	integer		t_time
word_ship_mode_id	char(16)		sm_ship_mode_id
word_web_site_id	char(16)		web_site_id
word_order_comments	char(100)		

Table A-7: Column definition s_item

Column	Datatype	NULLs	Foreign Key
item_item_id	char(16)	N	i_item_id
item_item_description	char(200)		
item_list_price	numeric(7,2)		
item_wholesale_cost	numeric(7,2)		
item_size	char(20)		
item_formulation	char(20)		
item_color	char(20)		
item_units	char(10)		
item_container	char(10)		
item_manager_id	integer		

Table A-8: Column definition s_catalog_order_lineitem

Column	Datatype	NULLs	Foreign Key
clin_order_id	identifier	N	cord_order_id
clin_line_number	integer		
clin_item_id	char(16)		i_item_id
clin_promotion_id	char(16)		p_promo_id
clin_quantity	integer		
clin_sales_price	numeric(7,2)		
clin_coupon_amt	numeric(7,2)		
clin_warehouse_id	char(16)		w_warehouse_id
clin_ship_date	char(10)		
clin_catalog_number	integer		
clin_catalog_page_number	integer		
clin_ship_cost	numeric(7,2)		

Figure A-9: Column definition s_web_order_lineitem

Column	Datatype	NULLs	Foreign Key
wlin_order_id	identifier	N	word_order_id
wlin_line_number	integer	N	
wlin_item_id	char(16)		i_item_id
wlin_promotion_id	char(16)		p_promo_id
wlin_quantity	integer		
wlin_sales_price	numeric(7,2)		
wlin_coupon_amt	numeric(7,2)		
wlin_warehouse_id	char(16)		w_warehouse_id
wlin_ship_date	char(10)		d_date
wlin_ship_cost	numeric(7,2)		
wlin_web_page_id	char(16)		wp_web_page

Figure A-10: Column definition s_store

Column	Datatype	NULLs	Foreign Key
stor_store_id	char(16)	N	s_store_id
stor_closed_date	char(10)		d_date
stor_name	char(50)		
stor_employees	integer		
stor_floor_space	integer		
stor_hours	char(20)		
stor_store_manager	char(40)		
stor_market_id	integer		
stor_geography_class	char(100)		
stor_market_manager	char(40)		
stor_tax_percentage	numeric(5,2)		

Table A-11: Column definition s_call_center

Column	Datatype	NULLs	Foreign Key
call_center_id	char(16)	N	cc_center_id
call_open_date	char(10)		d_date
call_closed_date	char(10)		d_date
call_center_name	char(50)		
call_center_class	char(50)		
call_center_employees	integer		
call_center_sq_ft	integer		
call_center_hours	char(20)		
call_center_manager	char(40)		
call_center_tax_percentage	numeric(7,2)		

Table A-12: Column definition s_web_site

Column	Datatype	NULLs	Foreign Key
wsit_web_site_id	char(16)	N	web_site_id
wsit_open_date	char(10)		d_date
wsit_closed_date	char(10)		d_date
wsit_site_name	char(50)		
wsit_site_class	char(50)		
wsit_site_manager	char(40),		
wsit_tax_percentage	decimal(5,2)		

Table A-13: Column definition s_warehouse

Column	Datatype	NULLs	Foreign Key
wrhs_warehouse_id	char(16)	N	w_warehouse_id
wrhs_warehouse_desc	char(200)		
wrhs_warehouse_sq_ft	integer		

Table A-14: Column definition s_web_page

Column	Datatype	NULLs	Foreign Key
wpag_web_page_id	char(16)	N	web_page_id
wpag_create_date	char(10)		d_date

Column	Datatype	NULLs	Foreign Key
wpag_access_date	char(10)		d_date
wpag_autogen_flag	char(1)		
wpag_url	char(100)		
wpag_type	char(50)		
wpag_char_cnt	integer		
wpag_link_cnt	integer		
wpag_image_cnt	integer		
wpag_max_ad_cnt	integer		

Table A-15: Column definition s_promotion

Column	Datatype	NULLs	Foreign Key
prom_promotion_id	char(16)	N	
prom_promotion_name	char(30)		
prom_start_date	char(10)		d_date
prom_end_date	char(10)		d_date
prom_cost	numeric(7,2)		
prom_response_target	char(1)		
prom_channel_dmail	char(1)		
prom_channel_email	char(1)		
prom_channel_catalog	char(1)		
prom_channel_tv	char(1)		
prom_channel_radio	char(1)		
prom_channel_press	char(1)		
prom_channel_event	char(1)		
prom_channel_demo	char(1)		
prom_channel_details	char(100)		
prom_purpose	char(15)		
prom_discount_active	char(1)		

Table A-16: Column definition s_store_returns

Column	Datatype	NULLs	Foreign Key
sret_store_id	char(16)		s_store_id
sret_purchase_id	char(16)	N	
sret_line_number	integer	N	
sret_item_id	char(16)	N	
sret_customer_id	char(16)		s_customer_id
sret_return_date	char(10)		d_date
sret_return_time	char(10)		t_time
sret_ticket_number	char(20)		
sret_return_qty	integer		
sret_return_amount	numeric(7,2)		
sret_return_tax	numeric(7,2)		
sret_return_fee	numeric(7,2)		
sret_return_ship_cost	numeric(7,2)		
sret_refunded_cash	numeric(7,2)		
sret_reversed_charge	numeric(7,2)		
sret_store_credit	numeric(7,2)		
sret_reason_id	char(16)		r_reason_id

Table A-17: Column definition s_catalog_returns

Column	Datatype	NULLs	Foreign Key
cret_call_center_id	char(16)		cc_call_center_id
cret_order_id	integer	N	
cret_line_number	integer	N	
cret_item_id	char(16)	N	i_item_id
cret_return_customer_id	char(16)		c_customer_id
cret_refund_customer_id	char(16)		c_customer_id
cret_return_date	char(10)		d_date
cret_return_time	char(10)		t_time
cret_return_qty	integer		
cret_return_amt	numeric(7,2)		
cret_return_tax	numeric(7,2)		

Column	Datatype	NULLs	Foreign Key
cret_return_fee	numeric(7,2)		
cret_return_ship_cost	numeric(7,2)		
cret_refunded_cash	numeric(7,2)		
cret_reversed_charge	numeric(7,2)		
cret_merchant_credit	numeric(7,2)		
cret_reason_id	char(16)		r_reason_id
cret_shipmode_id	char(16)		
cret_catalog_page_id	char(16)		
cret_warehouse_id	char(16)		

Table A-18: Column definition s_web_returns

Column	Datatype	NULLs	Foreign Key
wret_web_page_id	char(16)		wp_web_page_id
wret_order_id	integer	N	
wret_line_number	integer	N	
wret_item_id	char(16)	N	i_item_id
wret_return_customer_id	char(16)		c_customer_id
wret_refund_customer_id	char(16)		c_customer_id
wret_return_date	char(10)		d_date
wret_return_time	char(10)		t_time
wret_return_qty	integer		
wret_return_amt	numeric(7,2)		
wret_return_tax	numeric(7,2)		
wret_return_fee	numeric(7,2)		
wret_return_ship_cost	numeric(7,2)		
wret_refunded_cash	numeric(7,2)		
wret_reversed_charge	numeric(7,2)		
wret_account_credit	numeric(7,2)		
wret_reason_id	char(16)		r_reason_id

Table A-19: Column definition s_inventory

Column	Datatype	NULLs	Foreign Key
invn_warehouse_id	char(16),	N	w_warehouse_id
invn_item_id	char(16),	N	i_item_id
invn_date	char(10)	N	d_date
invn_qty_on_hand	integer		

Table A-20: Column definition s_catalog_page

Column	Datatype	NULLs	Foreign Key
cpag_catalog_number	integer	N	
cpag_catalog_page_number	integer	N	
cpag_department	char(20)		
cpag_id	char(16)		
cpag_start_date	char(10)		d_date
cpag_end_date	char(10)		d_date
cpag_description	varchar(100)		
cpag_type	varchar(100)		

A.2   Relationships between source schema tables

The following relationships are defined between source schema tables:

Table A-21: Column definition

Source Schema Table 1	Source Schema Table 2	Join Condition
s_purchase	s_purchse_lineitem	purc_purchase_id = plin_purchase_id
s_web_order	s_web_order_lineitem	word_order_id = wlin_order_id
s_catalog_order	s_catalog_order_lineitem	cord_order_id = clin_order_id

Appendix B: Business Questions

Comment: The leading zeros in the numerical suffix used when parameters hold multiple values match the output of dsqgen. The leading zeros do not appear in the query templates.

B.1   query1.tpl

Find customers who have returned items more than 20% more often than the average customer returns for a store in a given state for a given year.

Qualification Substitution Parameters:

- YEAR.01=2000
- STATE.01=TN
- AGG_FIELD.01 = SR_RETURN_AMT

B.2   query2.tpl

Report the increase of weekly web and catalog sales from one year to the next year for each week. That is, compute the increase of Monday, Tuesday, ... Sunday sales from one year to the following.

Qualification Substitution Parameters:

- YEAR.01=2001

B.3   query3.tpl

Report the total extended sales price per item brand of a specific manufacturer for all sales in a specific month of the year.

Qualification Substitution Parameters:

- MONTH.01=11
- *MANUFACT* =128
- AGGC = ss_ext_sales_price

B.4   query4.tpl

Find customers who spend more money via catalog than in stores. Identify preferred customers and their country of origin.

Qualification Substitution Parameters:

- YEAR.01=2001
- SELECTONE.01= t_s_secyear.customer_preferred_cust_flag

B.5   query5.tpl

Report sales, profit, return amount, and net loss in the store, catalog, and web channels for a 14-day window. Rollup results by sales channel and channel specific sales method (store for store sales, catalog page for catalog sales and web site for web sales)

Qualification Substitution Parameters:

- SALES_DATE.01=2000-08-23
- YEAR.01=2000

B.6 query6.tpl

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

Qualification Substitution Parameters:

- MONTH.01=1
- YEAR.01=2001

B.7 query7.tpl

Compute the average quantity, list price, discount, and sales price for promotional items sold in stores where the promotion is not offered by mail or a special event. Restrict the results to a specific gender, marital and educational status.

Qualification Substitution Parameters:

- YEAR.01=2000
- ES.01=College
- MS.01=S
- GEN.01=M

B.8 query8.tpl

Compute the net profit of stores located in 400 Metropolitan areas with more than 10 preferred customers.

Qualification Substitution Parameters:

ZIP.01=24128	ZIP.81=57834	ZIP.161=13354	ZIP.241=15734	ZIP.321=78668
ZIP.02=76232	ZIP.82=62878	ZIP.162=45375	ZIP.242=63435	ZIP.322=22245
ZIP.03=65084	ZIP.83=49130	ZIP.163=40558	ZIP.243=25733	ZIP.323=15798
ZIP.04=87816	ZIP.84=81096	ZIP.164=56458	ZIP.244=35474	ZIP.324=27156
ZIP.05=83926	ZIP.85=18840	ZIP.165=28286	ZIP.245=24676	ZIP.325=37930
ZIP.06=77556	ZIP.86=27700	ZIP.166=45266	ZIP.246=94627	ZIP.326=62971
ZIP.07=20548	ZIP.87=23470	ZIP.167=47305	ZIP.247=53535	ZIP.327=21337
ZIP.08=26231	ZIP.88=50412	ZIP.168=69399	ZIP.248=17879	ZIP.328=51622
ZIP.09=43848	ZIP.89=21195	ZIP.169=83921	ZIP.249=15559	ZIP.329=67853
ZIP.10=15126	ZIP.90=16021	ZIP.170=26233	ZIP.250=53268	ZIP.330=10567
ZIP.11=91137	ZIP.91=76107	ZIP.171=11101	ZIP.251=59166	ZIP.331=38415
ZIP.12=61265	ZIP.92=71954	ZIP.172=15371	ZIP.252=11928	ZIP.332=15455
ZIP.13=98294	ZIP.93=68309	ZIP.173=69913	ZIP.253=59402	ZIP.333=58263
ZIP.14=25782	ZIP.94=18119	ZIP.174=35942	ZIP.254=33282	ZIP.334=42029
ZIP.15=17920	ZIP.95=98359	ZIP.175=15882	ZIP.255=45721	ZIP.335=60279
ZIP.16=18426	ZIP.96=64544	ZIP.176=25631	ZIP.256=43933	ZIP.336=37125
ZIP.17=98235	ZIP.97=10336	ZIP.177=24610	ZIP.257=68101	ZIP.337=56240
ZIP.18=40081	ZIP.98=86379	ZIP.178=44165	ZIP.258=33515	ZIP.338=88190
ZIP.19=84093	ZIP.99=27068	ZIP.179=99076	ZIP.259=36634	ZIP.339=50308
ZIP.20=28577	ZIP.100=39736	ZIP.180=33786	ZIP.260=71286	ZIP.340=26859
ZIP.21=55565	ZIP.101=98569	ZIP.181=70738	ZIP.261=19736	ZIP.341=64457
ZIP.22=17183	ZIP.102=28915	ZIP.182=26653	ZIP.262=58058	ZIP.342=89091
ZIP.23=54601	ZIP.103=24206	ZIP.183=14328	ZIP.263=55253	ZIP.343=82136
ZIP.24=67897	ZIP.104=56529	ZIP.184=72305	ZIP.264=67473	ZIP.344=62377
ZIP.25=22752	ZIP.105=57647	ZIP.185=62496	ZIP.265=41918	ZIP.345=36233
ZIP.26=86284	ZIP.106=54917	ZIP.186=22152	ZIP.266=19515	ZIP.346=63837
ZIP.27=18376	ZIP.107=42961	ZIP.187=10144	ZIP.267=36495	ZIP.347=58078

- ZIP.28=38607   ZIP.108=91110   ZIP.188=64147   ZIP.268=19430   ZIP.348=17043
- ZIP.29=45200   ZIP.109=63981   ZIP.189=48425   ZIP.269=22351   ZIP.349=30010
- ZIP.30=21756   ZIP.110=14922   ZIP.190=14663   ZIP.270=77191   ZIP.350=60099
- ZIP.31=29741   ZIP.111=36420   ZIP.191=21076   ZIP.271=91393   ZIP.351=28810
- ZIP.32=96765   ZIP.112=23006   ZIP.192=18799   ZIP.272=49156   ZIP.352=98025
- ZIP.33=23932   ZIP.113=67467   ZIP.193=30450   ZIP.273=50298   ZIP.353=29178
- ZIP.34=89360   ZIP.114=32754   ZIP.194=63089   ZIP.274=87501   ZIP.354=87343
- ZIP.35=29839   ZIP.115=30903   ZIP.195=81019   ZIP.275=18652   ZIP.355=73273
- ZIP.36=25989   ZIP.116=20260   ZIP.196=68893   ZIP.276=53179   ZIP.356=30469
- ZIP.37=28898   ZIP.117=31671   ZIP.197=24996   ZIP.277=18767   ZIP.357=64034
- ZIP.38=91068   ZIP.118=51798   ZIP.198=51200   ZIP.278=63193   ZIP.358=39516
- ZIP.39=72550   ZIP.119=72325   ZIP.199=51211   ZIP.279=23968   ZIP.359=86057
- ZIP.40=10390   ZIP.120=85816   ZIP.200=45692   ZIP.280=65164   ZIP.360=21309
- ZIP.41=18845   ZIP.121=68621   ZIP.201=92712   ZIP.281=68880   ZIP.361=90257
- ZIP.42=47770   ZIP.122=13955   ZIP.202=70466   ZIP.282=21286   ZIP.362=67875
- ZIP.43=82636   ZIP.123=36446   ZIP.203=79994   ZIP.283=72823   ZIP.363=40162
- ZIP.44=41367   ZIP.124=41766   ZIP.204=22437   ZIP.284=58470   ZIP.364=11356
- ZIP.45=76638   ZIP.125=68806   ZIP.205=25280   ZIP.285=67301   ZIP.365=73650
- ZIP.46=86198   ZIP.126=16725   ZIP.206=38935   ZIP.286=13394   ZIP.366=61810
- ZIP.47=81312   ZIP.127=15146   ZIP.207=71791   ZIP.287=31016   ZIP.367=72013
- ZIP.48=37126   ZIP.128=22744   ZIP.208=73134   ZIP.288=70372   ZIP.368=30431
- ZIP.49=39192   ZIP.129=35850   ZIP.209=56571   ZIP.289=67030   ZIP.369=22461
- ZIP.50=88424   ZIP.130=88086   ZIP.210=14060   ZIP.290=40604   ZIP.370=19512
- ZIP.51=72175   ZIP.131=51649   ZIP.211=19505   ZIP.291=24317   ZIP.371=13375
- ZIP.52=81426   ZIP.132=18270   ZIP.212=72425   ZIP.292=45748   ZIP.372=55307
- ZIP.53=53672   ZIP.133=52867   ZIP.213=56575   ZIP.293=39127   ZIP.373=30625
- ZIP.54=10445   ZIP.134=39972   ZIP.214=74351   ZIP.294=26065   ZIP.374=83849
- ZIP.55=42666   ZIP.135=96976   ZIP.215=68786   ZIP.295=77721   ZIP.375=68908
- ZIP.56=66864   ZIP.136=63792   ZIP.216=51650   ZIP.296=31029   ZIP.376=26689
- ZIP.57=66708   ZIP.137=11376   ZIP.217=20004   ZIP.297=31880   ZIP.377=96451
- ZIP.58=41248   ZIP.138=94898   ZIP.218=18383   ZIP.298=60576   ZIP.378=38193
- ZIP.59=48583   ZIP.139=13595   ZIP.219=76614   ZIP.299=24671   ZIP.379=46820
- ZIP.60=82276   ZIP.140=10516   ZIP.220=11634   ZIP.300=45549   ZIP.380=88885
- ZIP.61=18842   ZIP.141=90225   ZIP.221=18906   ZIP.301=13376   ZIP.381=84935
- ZIP.62=78890   ZIP.142=58943   ZIP.222=15765   ZIP.302=50016   ZIP.382=69035
- ZIP.63=49448   ZIP.143=39371   ZIP.223=41368   ZIP.303=33123   ZIP.383=83144
- ZIP.64=14089   ZIP.144=94945   ZIP.224=73241   ZIP.304=19769   ZIP.384=47537
- ZIP.65=38122   ZIP.145=28587   ZIP.225=76698   ZIP.305=22927   ZIP.385=56616
- ZIP.66=34425   ZIP.146=96576   ZIP.226=78567   ZIP.306=97789   ZIP.386=94983
- ZIP.67=79077   ZIP.147=57855   ZIP.227=97189   ZIP.307=46081   ZIP.387=48033
- ZIP.68=19849   ZIP.148=28488   ZIP.228=28545   ZIP.308=72151   ZIP.388=69952
- ZIP.69=43285   ZIP.149=26105   ZIP.229=76231   ZIP.309=15723   ZIP.389=25486
- ZIP.70=39861   ZIP.150=83933   ZIP.230=75691   ZIP.310=46136   ZIP.390=61547
- ZIP.71=66162   ZIP.151=25858   ZIP.231=22246   ZIP.311=51949   ZIP.391=27385
- ZIP.72=77610   ZIP.152=34322   ZIP.232=51061   ZIP.312=68100   ZIP.392=61860
- ZIP.73=13695   ZIP.153=44438   ZIP.233=90578   ZIP.313=96888   ZIP.393=58048
- ZIP.74=99543   ZIP.154=73171   ZIP.234=56691   ZIP.314=64528   ZIP.394=56910
- ZIP.75=83444   ZIP.155=30122   ZIP.235=68014   ZIP.315=14171   ZIP.395=16807
- ZIP.76=83041   ZIP.156=34102   ZIP.236=51103   ZIP.316=79777   ZIP.396=17871
- ZIP.77=12305   ZIP.157=22685   ZIP.237=94167   ZIP.317=28709   ZIP.397=35258
- ZIP.78=57665   ZIP.158=71256   ZIP.238=57047   ZIP.318=11489   ZIP.398=31387
- ZIP.79=68341   ZIP.159=78451   ZIP.239=14867   ZIP.319=25103   ZIP.399=35458

- ZIP.80=25003   ZIP.160=54364 ZIP.240=73520 ZIP.320=32213 ZIP.400=35576
- QOY.01=2
- YEAR.01=1998

B.9 query9.tpl

Categorize store sales transactions into 5 buckets according to the number of items sold. Each bucket contains the average discount amount, sales price, list price, tax, net paid, paid price including tax, or net profit..

Qualification Substitution Parameters:

- AGGCTHEN.01= ss_ext_discount_amt
- AGGCELSE.01= ss_net_paid
- RC.01=74129
- RC.02=122840
- RC.03=56580
- RC.04=10097
- RC.05=165306

B.10 query10.tpl

Count the customers with the same gender, marital status, education status, purchase estimate, credit rating, dependent count, employed dependent count and college dependent count who live in certain counties and who have purchased from both stores and another sales channel during a three month time period of a given year.

Qualification Substitution Parameters:

- YEAR.01 = 2002
- MONTH.01 = 1
- COUNTY.01 = Rush County
- COUNTY.02 = Toole County
- COUNTY.03 = Jefferson County
- COUNTY.04 = Dona Ana County
- COUNTY.05 = La Porte County

B.11 query11.tpl

Find customers whose increase in spending was large over the web than in stores this year compared to last year.

Qualification Substitution Parameters:

- YEAR.01 = 2001
- SELECTONE = t_s_secyear.customer_preferred_cust_flag

B.12 query12.tpl

Compute the revenue ratios across item classes: For each item in a list of given categories, during a 30 day time period, sold through the web channel compute the ratio of sales of that item to the sum of all of the sales in that item's class.

Qualification Substitution Parameters

- CATEGORY.01 = Sports
- CATEGORY.02 = Books
- CATEGORY.03 = Home
- SDATE.01 = 1999-02-22

- YEAR.01 = 1999

B.13 query13.tpl

Calculate the average sales quantity, average sales price, average wholesale cost, total wholesale cost for store sales of different customer types (e.g., based on marital status, education status) including their household demographics, sales price and different combinations of state and sales profit for a given year.

Qualification Substitution Parameters:

- STATE.01 = TX
- STATE.02 = OH
- STATE.03 = TX
- STATE.04 = OR
- STATE.05 = NM
- STATE.06 = KY
- STATE.07 = VA
- STATE.08 = TX
- STATE.09 = MS
- ES.01 = Advanced Degree
- ES.02 = College
- ES.03 = 2 yr Degree
- MS.01 = M
- MS.02 = S
- MS.03 = W

B.14 query14.tpl)

This query contains multiple iterations:

Iteration 1: First identify items in the same brand, class and category that are sold in all three sales channels in two consecutive years. Then compute the average sales (quantity*list price) across all sales of all three sales channels in the same three years (average sales). Finally, compute the total sales and the total number of sales rolled up for each channel, brand, class and category. Only consider sales of cross channel sales that had sales larger than the average sale.

Iteration 2: Based on the previous query compare December store sales.

Qualification Substitution Parameters:

- DAY.01 = 11
- YEAR.01 = 1999

B.15 query15.tpl

Report the total catalog sales for customers in selected geographical regions or who made large purchases for a given year and quarter.

Qualification Substitution Parameters:

- QOY.01 = 2
- YEAR.01 = 2001

B.16 query16.tpl

Report number of orders, total shipping costs and profits from catalog sales of particular counties and states for a given 60 day period for non-returned sales filled from an alternate warehouse.

Qualification Substitution Parameters:

- COUNTY_E.01 = Williamson County
- COUNTY_D.01 = Williamson County
- COUNTY_C.01 = Williamson County
- COUNTY_B.01 = Williamson County
- COUNTY_A.01 = Williamson County
- STATE.01 = GA
- MONTH.01 = 2
- YEAR.01 = 2002

B.17 query17.tpl

Analyze, for each state, all items that were sold in stores in a particular quarter and returned in the next three quarters and then re-purchased by the customer through the catalog channel in the three following quarters.

Qualification Substitution Parameters:

- YEAR.01 = 2001

B.18 query18.tpl

Compute, for each county, the average quantity, list price, coupon amount, sales price, net profit, age, and number of dependents for all items purchased through catalog sales in a given year by customers who were born in a given list of six months and living in a given list of seven states and who also belong to a given gender and education demographic.

Qualification Substitution Parameters:

- MONTH.01 = 1
- MONTH.02 = 6
- MONTH.03 = 8
- MONTH.04 = 9
- MONTH.05 = 12
- MONTH.06 = 2
- STATE.01 = MS
- STATE.02 = IN
- STATE.03 = ND
- STATE.04 = OK
- STATE.05 = NM
- STATE.06 = VA
- STATE.07 = MS
- ES.01 = Unknown
- GEN.01 = F
- YEAR.01 = 1998

B.19 query19.tpl

Select the top revenue generating products bought by out of zip code customers for a given year, month and manager. Qualification Substitution Parameters

- MANAGER.01 = 8

- MONTH.01 = 11
- YEAR.01 = 1998

B.20 query20.tpl

Compute the total revenue and the ratio of total revenue to revenue by item class for specified item categories and time periods.

Qualification Substitution Parameters:

- CATEGORY.01 = Sports
- CATEGORY.02 = Books
- CATEGORY.03 = Home
- SDATE.01 = 1999-02-22
- YEAR.01 = 1999

B.21 query21.tpl

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

Qualification Substitution Parameters:

- SALES_DATE.01 = 2000-03-11
- YEAR.01 = 2000

B.22 query22.tpl

For each product name, brand, class, category, calculate the average quantity on hand. Rollup data by product name, brand, class and category.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.23 query23.tpl

This query contains multiple, related iterations:

Find frequently sold items that were sold more than 4 times on any day during four consecutive years through the store sales channel. Compute the maximum store sales made by any given customer in a period of four consecutive years (same as above). Compute the best store customers that are in the 5th percentile of sales.

Finally, compute the total web and catalog sales in a particular month made by our best store customers buying our most frequent store items.

Qualification Substitution Parameters:

- MONTH.01 = 2
- YEAR.01 = 2000
- TOPPERCENT=50

B.24 query24.tpl

This query contains multiple, related iterations:

Iteration 1: Calculate the total specified monetary value of items in a specific color for store sales transactions by customer name and store, in a specific market, from customers who currently don't live in their birth countries and in the neighborhood of the store, and list only those customers for whom the total specified monetary value is greater than 5% of the average value

 Iteration 2: Calculate the total specified monetary value of items in a specific color for store sales transactions by customer name and store, in a specific market, from customers who currently don't live in their birth countries and in the neighborhood of the store, and list only those customers for whom the total specified monetary value is greater than 5% of the average value

Qualification Substitution Parameters:

- MARKET = 8
- COLOR.1 = peach
- COLOR.2 = saddle
- AMOUNTONE = ss_net_paid

B.25 query25.tpl

Get all items that were

- sold in stores in a particular month and year and
- returned and re-purchased by the customer through the catalog channel in the same month and in the six following months.

For these items, compute the sum of net profit of store sales, net loss of store loss and net profit of catalog . Group this information by item and store.

Qualification Substitution Parameters:

- YEAR.01 = 2001
- AGG.01 = sum

B.26 query26.tpl

Computes the average quantity, list price, discount, sales price for promotional items sold through the catalog channel where the promotion was not offered by mail or in an event for given gender, marital status and educational status.

Qualification Substitution Parameters:

- YEAR.01 = 2000
- ES.01 = College
- MS.01 = S
- GEN.01 = M

B.27 query27.tpl

For all items sold in stores located in six states during a given year, find the average quantity, average list price, average list sales price, average coupon amount for a given gender, marital status, education and customer demographic.

Qualification Substitution Parameters:

- STATE_F.01 = TN

- STATE_E.01 = TN
- STATE_D.01 = TN
- STATE_C.01 = TN
- STATE_B.01 = TN
- STATE_A.01 = TN
- ES.01 = College
- MS.01 = S
- GEN.01 = M
- YEAR.01 = 2002

B.28   query28.tpl

Calculate the average list price, number of non empty (null) list prices and number of distinct list prices of six different sales buckets of the store sales channel. Each bucket is defined by a range of distinct items and information about list price, coupon amount and wholesale cost.

Qualification Substitution Parameters:

- WHOLESALECOST.01=57
- WHOLESALECOST.02=31
- WHOLESALECOST.03=79
- WHOLESALECOST.04=38
- WHOLESALECOST.05=17
- WHOLESALECOST.06=7
- COUPONAMT.01=459
- COUPONAMT.02=2323
- COUPONAMT.03=12214
- COUPONAMT.04=6071
- COUPONAMT.05=836
- COUPONAMT.06=7326
- LISTPRICE.01=8
- LISTPRICE.02=90
- LISTPRICE.03=142
- LISTPRICE.04=135
- LISTPRICE.05=122
- LISTPRICE.06=154

B.29   query29.tpl

Get all items that were sold in stores in a specific month and year and which were returned in the next six months of the same year and re-purchased by the returning customer afterwards through the catalog sales channel in the following three years.

For those these items, compute the total quantity sold through the store, the quantity returned and the quantity purchased through the catalog. Group this information by item and store.

Qualification Substitution Parameters:

- MONTH.01 = 9
- YEAR.01 = 1999
- AGG.01 = sum

B.30 query30.tpl

Find customers and their detailed customer data who have returned items, which they bought on the web, for an amount that is 20% higher than the average amount a customer returns in a given state in a given time period across all items. Order the output by customer data.

Qualification Substitution Parameters:

- YEAR.01 = 2002
- STATE.01 = GA

B.31 query31.tpl

List counties where the percentage growth in web sales is consistently higher compared to the percentage growth in store sales in the first three consecutive quarters for a given year.

Qualification Substitution Parameters:

- YEAR.01 = 2000
- AGG.01 = ss1.ca_county

B.32 query32.tpl

Compute the total discounted amount for a particular manufacturer in a particular 90 day period for catalog sales whose discounts exceeded the average discount by at least 30%.

Qualification Substitution Parameters:

- CSDATE.01 = 2000-01-27
- YEAR.01 = 2000
- IMID.01 = 977

B.33 query33.tpl

What is the monthly sales figure based on extended price for a specific month in a specific year, for manufacturers in a specific category in a given time zone. Group sales by manufacturer identifier and sort output by sales amount, by channel, and give Total sales.

Qualification Substitution Parameters:

- CATEGORY.01 = Electronics
- GMT.01 = -5
- MONTH.01 = 5
- YEAR.01 = 1998

B.34 query34.tpl

Display all customers with specific buy potentials and whose dependent count to vehicle count ratio is larger than 1.2, who in three consecutive years made purchases with between 15 and 20 items in the beginning or the end of each month in stores located in 8 counties.

Qualification Substitution Parameters:

- COUNTY_H.01 = Williamson County
- COUNTY_G.01 = Williamson County
- COUNTY_F.01 = Williamson County
- COUNTY_E.01 = Williamson County
- COUNTY_D.01 = Williamson County
- COUNTY_C.01 = Williamson County

- COUNTY_B.01 = Williamson County
- COUNTY_A.01 = Williamson County
- YEAR.01 = 1999
- BPTWO.01 = Unknown
- BPONE.01 = >10000

B.35   query35.tpl

For the groups of customers living in the same state, having the same gender and marital status who have purchased from stores and from either the catalog or the web during a given year, display the following:

- state, gender, marital status, count of customers
- min, max, avg, count distinct of the customer's dependent count
- min, max, avg, count distinct of the customer's employed dependent count
- min, max, avg, count distinct of the customer's dependents in college count

Display / calculate the "count of customers" multiple times to emulate a potential reporting tool scenario.

Qualification Substitution Parameters:

- YEAR.01 = 2002
- AGGONE = min
- AGGTWO = max
- AGGTHREE = avg

B.36   query36.tpl

Compute store sales gross profit margin ranking for items in a given year for a given list of states.\

Qualification Substitution Parameters:

- STATE_H.01 = TN
- STATE_G.01 = TN
- STATE_F.01 = TN
- STATE_E.01 = TN
- STATE_D.01 = TN
- STATE_C.01 = TN
- STATE_B.01 = TN
- STATE_A.01 = TN
- YEAR.01 = 2001

B.37   query37.tpl

List all items and current prices sold through the catalog channel from certain manufacturers in a given $30 price range and consistently had a quantity between 100 and 500 on hand in a 60-day period.

Qualification Substitution Parameters:

- PRICE.01 = 68
- MANUFACT_ID.01 = 677
- MANUFACT_ID.02 = 940
- MANUFACT_ID.03 = 694
- MANUFACT_ID.04 = 808
- INVDATE.01 = 2000-02-01

B.38 query38.tpl

Display count of customers with purchases from all 3 channels in a given year.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.39 query39.tpl

This query contains multiple, related iterations:

Iteration 1: Calculate the coefficient of variation and mean of every item and warehouse of two consecutive months

Iteration 2: Find items that had a coefficient of variation in the first months of 1.5 or large

Qualification Substitution Parameters:

- YEAR.01 = 2001
- MONTH.01 = 1

B.40 query40.tpl

Compute the impact of an item price change on the 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.

Qualification Substitution Parameters

- SALES_DATE.01 = 2000-03-11
- YEAR.01 = 2000

B.41 query41.tpl

How many items do we carry with specific combinations of color, units, size and category.

Qualification Substitution Parameters

- MANUFACT.01 = 738
- SIZE.01 = medium
- SIZE.02 = extra large
- SIZE.03 = N/A
- SIZE.04 = small
- SIZE.05 = petite
- SIZE.06 = large
- UNIT.01 = Ounce
- UNIT.02 = Oz
- UNIT.03 = Bunch
- UNIT.04 = Ton
- UNIT.05 = N/A
- UNIT.06 = Dozen
- UNIT.07 = Box
- UNIT.08 = Pound
- UNIT.09 = Pallet
- UNIT.10 = Gross
- UNIT.11 = Cup
- UNIT.12 = Dram

- UNIT.13 = Each
- UNIT.14 = Tbl
- UNIT.15 = Lb
- UNIT.16 = Bundle
- COLOR.01 = powder
- COLOR.02 = khaki
- COLOR.03 = brown
- COLOR.04 = honeydew
- COLOR.05 = floral
- COLOR.06 = deep
- COLOR.07 = light
- COLOR.08 = cornflower
- COLOR.09 = midnight
- COLOR.10 = snow
- COLOR.11 = cyan
- COLOR.12 = papaya
- COLOR.13 = orange
- COLOR.14 = frosted
- COLOR.15 = forest
- COLOR.16 = ghost

B.42   query42.tpl

For each item and a specific year and month calculate the sum of the extended sales price of store transactions.

Qualification Substitution Parameters:

- MONTH.01 = 11
- YEAR.01 = 2000

B.43   query43.tpl

Report the sum of all sales from Sunday to Saturday for stores in a given data range by stores.

Qualification Substitution Parameters:

- YEAR.01 = 2000
- GMT.01 = -5

B.44   query44.tpl

List the best and worst performing products measured by net profit.

Qualification Substitution Parameters:

- NULLCOLSS.01 = ss_addr_sk
- STORE.01 = 4

B.45   query45.tpl

Report the total web sales for customers in specific zip codes, cities, counties or states, or specific items for a given year and quarter. .

Qualification Substitution Parameters:

- QOY.01 = 2
- YEAR.01 = 2001

- GBOBC = ca_city

B.46 query46.tpl

Compute the per-customer coupon amount and net profit of all "out of town" customers buying from stores located in 5 cities on weekends in three consecutive years. The customers need to fit the profile of having a specific dependent count and vehicle count. For all these customers print the city they lived in at the time of purchase, the city in which the store is located, the coupon amount and net profit

Qualification Substitution Parameters:

- CITY_E.01 = Fairview
- CITY_D.01 = Fairview
- CITY_C.01 = Fairview
- CITY_B.01 = Midway
- CITY_A.01 = Fairview
- VEHCNT.01 = 3
- YEAR.01 = 1999
- DEPCNT.01 = 4

B.47 query47.tpl

Find the item brands and categories for each store and company, the monthly sales figures for a specified year, where the monthly sales figure deviated more than 10% of the average monthly sales for the year, sorted by deviation and store. Report deviation of sales from the previous and the following monthly sales.

Qualification Substitution Parameters

- YEAR.01 = 1999
- SELECTONE = v1.i_category, v1.i_brand, v1.s_store_name, v1.s_company_name
- SELECTTWO = ,v1.d_year, v1.d_moy

B.48 query48.tpl

Calculate the total sales by different types of customers (e.g., based on marital status, education status), sales price and different combinations of state and sales profit.

Qualification Substitution Parameters:

- MS.01=M
- MS.02=D
- MS.03=S
- ES.01=4 yr Degree
- ES.02=2 yr Degree
- ES.03=College
- STATE.01=CO
- STATE.02=OH
- STATE.03=TX
- STATE.04=OR
- STATE.05=MN
- STATE.06=KY
- STATE.07=VA
- STATE.08=CA
- STATE.09=MS
- YEAR.01=2000

B.49 query49.tpl

Report the worst return ratios (sales to returns) of all items for each channel by quantity and currency sorted by ratio. Quantity ratio is defined as total number of sales to total number of returns. Currency ratio is defined as sum of return amount to sum of net paid.

Qualification Substitution Parameters:

- MONTH.01 = 12
- YEAR.01 = 2001

B.50 query50.tpl

For each store count the number of items in a specified month that were returned after 30, 60, 90, 120 and more than 120 days from the day of purchase.

Qualification Substitution Parameters:

- MONTH.01 = 8
- YEAR.01 = 2001

B.51 query51.tpl

Compute the count of store sales resulting from promotions, the count of all store sales and their ratio for specific categories in a particular time zone and for a given year and month.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.52 query52.tpl

Report the total of extended sales price for all items of a specific brand in a specific year and month.

Qualification Substitution Parameters

- MONTH.01=11
- YEAR.01=2000

B.53 query53.tpl

Find the ID, quarterly sales and yearly sales of those manufacturers who produce items with specific characteristics and whose average monthly sales are larger than 10% of their monthly sales.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.54 query54.tpl

Find all customers who purchased items of a given category and class on the web or through catalog in a given month and year that was followed by an in-store purchase at a store near their residence in the three consecutive months. Calculate a histogram of the revenue by these customers in $50 segments showing the number of customers in each of these revenue generated segments.

Qualification Substitution Parameters:

- CLASS.01 = maternity
- CATEGORY.01 = Women
- MONTH.01 = 12

- YEAR.01 = 1998

B.55 query55.tpl

For a given year, month and store manager calculate the total store sales of any combination all brands.

Qualification Substitution Parameters:

- MANAGER.01 = 28
- MONTH.01 = 11
- YEAR.01 = 1999

B.56 query56.tpl

Compute the monthly sales amount for a specific month in a specific year, for items with three specific colors across all sales channels. Only consider sales of customers residing in a specific time zone. Group sales by item and sort output by sales amount.

Qualification Substitution Parameters:

- COLOR.01 = slate
- COLOR.02 = blanched
- COLOR.03 = burnished
- GMT.01 = -5
- MONTH.01 = 2
- YEAR.01 = 2001

B.57 query57.tpl

Find the item brands and categories for each call center and their monthly sales figures for a specified year, where the monthly sales figure deviated more than 10% of the average monthly sales for the year, sorted by deviation and call center. Report the sales deviation from the previous and following month.

Qualification Substitution Parameters:

- YEAR.01 = 1999
- SELECTONE = v1.i_category, v1.i_brand, v1.cc_name
- SELECTTWO = ,v1.d_year, v1.d_moy

B.58 query58.tpl

Retrieve the items generating the highest revenue and which had a revenue that was approximately equivalent across all of store, catalog and web within the week ending a given date.

Qualification Substitution Parameters:

- SALES_DATE.01 = 2000-01-03

B.59 query59.tpl

Report the increase of weekly store sales from one year to the next year for each store and day of the week.

Qualification Substitution Parameters:

- DMS.01 = 1212

B.60 query60.tpl

What is the monthly sales amount for a specific month in a specific year, for items in a specific category, purchased by customers residing in a specific time zone. Group sales by item and sort output by sales amount.

Qualification Substitution Parameters:

- CATEGORY.01 = Music
- GMT.01 = -5
- MONTH.01 = 9
- YEAR=1998

B.61 query61.tpl

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

Qualification Substitution Parameters:

- GMT.01 = -5
- CATEGORY.01 = Jewelry
- MONTH.01 = 11
- YEAR.01 = 1998

B.62 query62.tpl

For web sales, create a report showing the counts of orders shipped within 30 days, from 31 to 60 days, from 61 to 90 days, from 91 to 120 days and over 120 days within a given year, grouped by warehouse, shipping mode and web site.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.63 query63.tpl

For a given year calculate the monthly sales of items of specific categories, classes and brands that were sold in stores and group the results by store manager. Additionally, for every month and manager print the yearly average sales of those items.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.64 query64.tpl

Find those stores that sold more cross-sales items from one year to another. Cross-sale items are items that are sold over the Internet, by catalog and in store.

Qualification Substitution Parameters:

- YEAR.01 = 1999
- PRICE.01 = 64
- COLOR.01 = purple
- COLOR.02 = burlywood
- COLOR.03 = indian
- COLOR.04 = spring
- COLOR.05 = floral
- COLOR.06 = medium

B.65   query65.tpl

In a given period, for each store, report the list of items with revenue less than 10% the average revenue for all the items in that store.

Qualification Substitution Parameters:

- DMS.01 = 1176

B.66   query66.tpl

Compute web and catalog sales and profits by warehouse. Report results by month for a given year during a given 8-hour period.

Qualification Substitution Parameters

- SALESTWO.01 = cs_sales_price
- SALESONE.01 = ws_ext_sales_price
- NETTWO.01 = cs_net_paid_inc_tax
- NETONE.01 = ws_net_paid
- SMC.01 = DHL
- SMC.02 = BARIAN
- TIMEONE.01 = 30838
- YEAR.01 = 2001

B.67   query67.tpl

Find top stores for each category based on store sales in a specific year.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.68   query68.tpl

Compute the per customer extended sales price, extended list price and extended tax for "out of town" shoppers buying from stores located in two cities in the first two days of each month of three consecutive years. Only consider customers with specific dependent and vehicle counts.

Qualification Substitution Parameters:

- CITY_B.01 = Midway
- CITY_A.01 = Fairview
- VEHCNT.01 = 3
- YEAR.01 = 1999
- DEPCNT.01 = 4

B.69   query69.tpl

Count the customers with the same gender, marital status, education status, education status, purchase estimate and credit rating who live in certain states and who have purchased from stores but neither form the catalog nor from the web during a two month time period of a given year.

Qualification Substitution Parameters:

- STATE.01 = KY
- STATE.02 = GA
- STATE.03 = NM

- YEAR.01 = 2001
- MONTH.01 = 4

B.70 query70.tpl

Compute store sales net profit ranking by state and county for a given year and determine the five most profitable states.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.71 query71.tpl

Select the top revenue generating products, sold during breakfast or dinner time for one month managed by a given manager across all three sales channels.

Qualification Substitution Parameters:

- MANAGER.01 = 1
- MONTH.01 = 11
- YEAR.01 = 1999

B.72 query72.tpl

For each item, warehouse and week combination count the number of sales with and without promotion.

Qualification Substitution Parameters:

- BP.01 = >10000
- MS.01 = D
- YEAR.01 = 1999

Comment: The adding of the scalar number 5 to d1.d_date in the predicate "d3.d_date > d1.d_date + 5" means that 5 days are added to d1.d_date.

B.73 query73.tpl

Count the number of customers with specific buy potentials and whose dependent count to vehicle count ratio is larger than 1 and who in three consecutive years bought in stores located in 4 counties between 1 and 5 items in one purchase. Only purchases in the first 2 days of the months are considered.

Qualification Substitution Parameters:

- COUNTY_D.01 = Orange County
- COUNTY_C.01 = Bronx County
- COUNTY_B.01 = Franklin Parish
- COUNTY_A.01 = Williamson County
- YEAR.01 = 1999
- BPTWO.01 = Unknown
- BPONE.01 = >10000

B.74 query74.tpl

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.

Qualification Substitution Parameters:

- YEAR.01 = 2001
- AGGONE.01 = sum
- ORDERC.01 = 1
- ORDERC.02 = 1
- ORDERC.03 = 1

B.75 query75.tpl

For two consecutive years track the sales of items by brand, class and category.

Qualification Substitution Parameters:

- CATEGORY.01 = Books
- YEAR.01 = 2002

B.76 query76.tpl

Computes the average quantity, list price, discount, sales price for promotional items sold through the web channel where the promotion is not offered by mail or in an event for given gender, marital status and educational status.

Qualification Substitution Parameters:

- NULLCOLCS.01 = cs_ship_addr_sk
- NULLCOLWS.01 = ws_ship_customer_sk
- NULLCOLSS.01 = ss_store_sk

B.77 query77.tpl

Report the total sales, returns and profit for all three sales channels for a given 30 day period. Roll up the results by channel and a unique channel location identifier.

Qualification Substitution Parameters:

- SALES_DATE.01 = 2000-08-23

B.78 query78.tpl

Report the top customer / item combinations having the highest ratio of store channel sales to all other channel sales (minimum 2 to 1 ratio), for combinations with at least one store sale and one other channel sale. Order the output by highest ratio.

Qualification Substitution Parameters:

- YEAR.01 = 2000
- SELECTONE.01 = ss_sold_year, ss_item_sk, ss_customer_sk

B.79 query79.tpl

Compute the per customer coupon amount and net profit of Monday shoppers. Only purchases of three consecutive years made on Mondays in large stores by customers with a certain dependent count and with a large vehicle count are considered.

Qualification Substitution Parameters:

- VEHCNT.01 = 2
- YEAR.01 = 1999
- DEPCNT.01 = 6

B.80 query80.tpl

Report extended sales, extended net profit and returns in the store, catalog, and web channels for a 30 day window for items with prices larger than $50 not promoted on television, rollup results by sales channel and channel specific sales means (store for store sales, catalog page for catalog sales and web site for web sales)

Qualification Substitution Parameters:

- SALES_DATE.01 = 2000-08-23

B.81 query81.tpl

Find customers and their detailed customer data who have returned items bought from the catalog more than 20 percent the average customer returns for customers in a given state in a given time period. Order output by customer data.

Qualification Substitution Parameters:

- YEAR.01 = 2000
- STATE.01 = GA

B.82 query82.tpl

Find customers who tend to spend more money (net-paid) on-line than in stores.

Qualification Substitution Parameters

- MANUFACT_ID.01 = 129
- MANUFACT_ID.02 = 270
- MANUFACT_ID.03 = 821
- MANUFACT_ID.04 = 423
- INVDATE.01 = 2000-05-25
- PRICE.01 = 62

B.83 query83.tpl

Retrieve the items with the highest number of returns where the number of returns was approximately equivalent across all store, catalog and web channels (within a tolerance of +/- 10%), within the week ending a given date.

Qualification Substitution Parameters

- RETURNED_DATE_THREE.01 = 2000-11-17
- RETURNED_DATE_TWO.01 = 2000-09-27
- RETURNED_DATE_ONE.01 = 2000-06-30

B.84 query84.tpl

List all customers living in a specified city, with an income between 2 values.

Qualification Substitution Parameters

- INCOME.01 = 38128
- CITY.01 = Edgewood

B.85 query85.tpl

For all web return reason calculate the average sales, average refunded cash and average return fee by different combinations of customer and sales types (e.g., based on marital status, education status, state and sales profit).

Qualification Substitution Parameters:

- YEAR.01 = 2000
- STATE.01 = IN
- STATE.02 = OH
- STATE.03 = NJ
- STATE.04 = WI
- STATE.05 = CT
- STATE.06 = KY
- STATE.07 = LA
- STATE.08 = IA
- STATE.09 = AR
- ES.01 = Advanced Degree
- ES.02 = College
- ES.03 = 2 yr Degree
- MS.01 = M
- MS.02 = S
- MS.03 = W

B.86 query86.tpl

Rollup the web sales for a given year by category and class, and rank the sales among peers within the parent, for each group compute sum of sales, location with the hierarchy and rank within the group.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.87 query87.tpl

Count how many customers have ordered on the same day items on the web and the catalog and on the same day have bought items in a store.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.88   query88.tpl

How many items do we sell between pacific times of a day in certain stores to customers with one dependent count and 2 or less vehicles registered or 2 dependents with 4 or fewer vehicles registered or 3 dependents and five or less vehicles registered. In one row break the counts into sells from 8:30 to 9, 9 to 9:30, 9:30 to 10 ... 12 to 12:30

Qualification Substitution Parameters:

- STORE.01=Unknown
- HOUR.01=4
- HOUR.02=2
- HOUR.03=0

B.89   query89.tpl

Within a year list all month and combination of item categories, classes and brands that have had monthly sales larger than 0.1 percent of the total yearly sales.

Qualification Substitution Parameters:

- CLASS_F.01 = dresses
- CAT_F.01 = Women
- CLASS_E.01 = birdal
- CAT_E.01 = Jewelry
- CLASS_D.01 = shirts
- CAT_D.01 = Men
- CLASS_C.01 = football
- CAT_C.01 = Sports
- CLASS_B.01 = stereo
- CAT_B.01 = Electronics
- CLASS_A.01 = computers
- CAT_A.01 = Books
- YEAR.01 = 1999

B.90   query90.tpl

What is the ratio between the number of items sold over the internet in the morning (8 to 9am) to the number of items sold in the evening (7 to 8pm) of customers with a specified number of dependents. Consider only websites with a high amount of content.

Qualification Substitution Parameters:

- HOUR_PM.01 = 19
- HOUR_AM.01 = 8
- DEPCNT.01 = 6

B.91   query91.tpl

Display total returns of catalog sales by call center and manager in a particular month for male customers of unknown education or female customers with advanced degrees with a specified buy potential and from a particular time zone.

Qualification Substitution Parameters:

- YEAR.01 = 1998
- MONTH.01 = 11

- BUY_POTENTIAL.01 = Unknown
- GMT.01 = -7

B.92   query92.tpl

Compute the total discount on web sales of items from a given manufacturer over a particular 90 day period for sales whose discount exceeded 30% over the average discount of items from that manufacturer in that period of time.

Qualification Substitution Parameters:

- IMID.01 = 350
- WSDATE.01 = 2000-01-27

B.93   query93.tpl

For a given merchandise return reason, report on customers' total cost of purchases minus the cost of returned items.

Qualification Substitution Parameters:

- REASON.01 = reason 28

B.94   query94.tpl

Produce a count of web sales and total shipping cost and net profit in a given 60 day period to customers in a given state from a named web site for non returned orders shipped from more than one warehouse.

Qualification Substitution Parameters:

- YEAR.01 = 1999
- MONTH.01 = 2
- STATE.01 = IL

B.95   query95.tpl

Produce a count of web sales and total shipping cost and net profit in a given 60 day period to customers in a given state from a named web site for returned orders shipped from more than one warehouse.

Qualification Substitution Parameters:

- STATE.01=IL
- MONTH.01=2
- YEAR.01=1999

B.96   query96.tpl

Compute a count of sales from a named store to customers with a given number of dependents made in a specified half hour period of the day.

Qualification Substitution Parameters:

- HOUR.01 = 20
- DEPCNT.01 = 7

B.97   query97.tpl

Generate counts of promotional sales and total sales, and their ratio from the web channel for a particular item category and month to customers in a given time zone.

Qualification Substitution Parameters:

- DMS.01 = 1200

B.98   query98.tpl

Report on items sold in a given 30 day period, belonging to the specified category.

Qualification Substitution Parameters

- YEAR.01 = 1999
- SDATE.01 = 1999-02-22
- CATEGORY.01 = Sports
- CATEGORY.02 = Books
- CATEGORY.03 = Home

B.99   query99.tpl

For catalog sales, create a report showing the counts of orders shipped within 30 days, from 31 to 60 days, from 61 to 90 days, from 91 to 120 days and over 120 days within a given year, grouped by warehouse, call center and shipping mode.

Qualification Substitution Parameters

- DMS.01 = 1200

Appendix C: <u>Approved Query Variants</u>

The following Query variants are approved. See Table 0-1 for location of Original Query Template and Approved Query Variant Templates.

Original Query Template	Approved Query Variant Template
Query10.tpl	Query10a.tpl
Query18.tpl	Query18a.tpl
Query27.tpl	Query27a.tpl
Query35.tpl	Query35a.tpl
Query36.tpl	Query36a.tpl
Query51.tpl	Query51a.tpl
Query70.tpl	Query70a.tpl
Query77.tpl	Query77a.tpl
Query80.tpl	Query80a.tpl
Query86.tpl	Query86a.tpl

Appendix D: Query Ordering

The order of query templates in each query stream is determined by dsqgen. For convenience the following table displays the order of query templates for the first 21 streams. The order is the same for all scale factors.

Table 11-1 Required Query Sequences

SEQ Num	Stream Number																				
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18
2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35
3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16
4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6
5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22
6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61
7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67
8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54
9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60
10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43
11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76
12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90
13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91
14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21
15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92
16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31
17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98
18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9
19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38
20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86
21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36
22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34
23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17
24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2
25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50
26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15
27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37
28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69
29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78
30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63
31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72
32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82
33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70
34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93
35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56
36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28
37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71
38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64
39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33
40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48
41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32
42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96
43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24
44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80
45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25
46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94
47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83
48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53
49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29
50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87
51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99
52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19
53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10
54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20

SEQ Num	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73
56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75
57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44
58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97
59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8
60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77
61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1
62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7
63	8	19	58	6	80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81
64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40
65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11
66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85
67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89
68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41
69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14	21	36	28	69	14
70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4
71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51
72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23	46
73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52
74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66	88
75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27
76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39	66
77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68
78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95
79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30	59
80	84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42
81	54	38	97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65
82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79
83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3
84	2	7	32	98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47
85	26	10	78	77	87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57
86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74
87	72	71	65	20	64	12	1	96	83	56	89	79	73	34	70	57	15	43	95	68	23
88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55	22	33	85	26
89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45	3	75	30
90	18	45	3	75	30	59	52	93	4	44	92	24	62	41	17	94	99	76	74	48	49
91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13	91	13
92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84
93	4	44	92	24	62	41	17	94	99	76	74	48	49	9	25	16	27	63	8	19	58
94	99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5
95	68	23	46	51	11	86	40	90	18	45	3	75	30	59	52	93	4	44	92	24	62
96	83	56	89	79	73	34	70	57	15	43	95	68	23	46	51	11	86	40	90	18	45
97	61	42	47	35	67	82	55	22	33	85	26	10	78	77	87	72	71	65	20	64	12
98	5	39	66	88	53	29	60	50	31	37	81	54	38	97	61	42	47	35	67	82	55
99	76	74	48	49	9	25	16	27	63	8	19	58	6	80	84	2	7	32	98	5	39

Appendix E: Sample Executive Summary

Sponsor (Optional Logo)	System Identification	TPC-DS 1.0.0 TPC Pricing 1.1.0
		Report Date November 22, 2006
Total System Cost	TPC-DS Throughput	Price / Performance	Availability Date
$1,197,918 (USD)	30013.4 QphDS@1000GB	4.252 $ / QphDS@1000GB	November 22, 2006
Database Size	Database Manager	Operating System	Other Software
1000GB	Database System	OS	

Benchmarked configuration

Elapsed Time

Load Includes Backup = No		RAID = No	
System Component		System Total	Per Node
Nodes		8	na
Processors/Cores/Threads: XXXX DualCore/gg Processor Y 3.4GHz, 1x5MB L3 Cache		8/16/16	1/2/2
Memory		64GB	8GB
OS Disk Drives: 36GB 15Krpm HDD Ultra320		24	3
Network Interfaces: on board Gigk		16	2
Host Bus Adapters		16	2
Storage Switch		2	na
Storage Subsystem XXXX Storage Array		5	na
Storage Subsystem Disk Drives: 36GB 15Krpm HDD Ultra320		208	na
Total Mass Storage		7062.1 GdB	na
* Database Size includes only new data (i.e., no temp, index, redundant storage space, etc.)			

Sponsor	System Identification	

Description	Part Number	Price Source	Unit Price	Qty.	Extended Price	3-yr Maint. Price
Server Hardware:						
SrvS: XXX, 16x 9040 1.6GHz, 32GB Mem, Service Processor & Server Mngmt S/W	XXX-1121165-100 (not incl.)	1	$70,500	1	$70,500	$14,400
MEM: 4GB XXX Memory, 2GB DIMMs	XXX116201-ICA	1	$1,280	5	$16,240	
MEM: 32GB XXX Memory, 2GB DIMMs	XXX116201-32G	1	$11,552	6	$69,312	
CTRL: Fibre Channel HBA, 2Gb 2-Port PCI	XXX5203-PCX	1	$2,118	37	$78,106	
CTRL: Fibre Channel HBA, 4Gb 2-Port PCI	XXX6M021-PCX	1	$1,899	3	$5,691	
KIT: 1x RAID Card, 1x 36GB Disk (boot media)	XXX112061-IDK	1	$719	1	$719	
DISK: 1x 36GB 10Krpm SAS (boot media)	XXX403601-X2S	1	$425	1	$425	
IF: Monitor, 17-inch LCD, Kybrd, Mse & Cable	XXX-U1F	1	$665	1	$665	
CBL: Ethernet, Cat5e, RM5 Conn's, 4m	XXX600004-T0T	1	$19	1	$19	
CAB: 40U x 19" x 42" Open Front Cabinet	XXX401M21-FSS	1	$2,410	1	$2,410	
CAB: 40U Open Front Trim Kit	XXX401M21-OFK	1	$235	1	$235	
ACC: Installation Maintenance Materials	XXX000-INS	1	$166	1	$166	
ACC: XXX Call Home Service	XXX11x00-CH	1	$0	1	$0	
			Server Subtotal		$244,620	$14,400
Storage Hardware:						
DISK: 13GB, 15Krpm 2Gb FC *	XXX315-UGF	1	$1,201	33	$39,631	Spares1
CTRL: DPE, 2 RAID Card, 1GB Cache ea., 0 Disk	XXX800-DPE	1	$13,300	1	$13,300	$1,005
RCK: DAE Storage Enclosure, FC 2Gb Optical I/F	XXX2-UFD	1	$5,605	1	$5,605	$632
SrvS MGT: Storage, Navisphere Manager	XXX300-WRK	1	$2,138	1	$2,138	$1,005
DISK: 36GB, 15Krpm 4Gb FC *	XXX403615-4F	1	$426	601	$256,026	Spares1
DISK: 13GB, 15Krpm 4Gb FC *	XXX40T315-4F	1	$996	125	$124,500	Spares1
RCK: Storage Enclosure, FC 4Gb Optical Optical I/F	XXX4000-DAS	1	$5,441	47	$256,009	$40,605
RCK: Rack Mount Kit, Disk Enclosure	XXX4000-RMK	1	$256	47	$12,032	
RCK: SFP Transceiver, 4Gb LC Optical	XXX40101F	1	$204	79	$16,116	$2,330
CBL: FC, Optical, LC->LC Conn's, 10m *	XXXA2LLM10-M	3	$48	83	$4,116	Spares1
PAR: Distribution Strip, 8-Socket, 220V	XXX524-STv	1	$404	13	$5,252	
CAB: 40U x 19" x 42" Open Front Cabinet	XXX401M21-AUX	1	$1,805	4	$7,220	
CAB: 40U Open Front Trim Kit	XXX401M21-OFK	1	$235	4	$952	
			Storage Subtotal		$699,197	$40,426
Server Software:						
OS: XXX, 16P, 1yr Lic/h.	XXX123-M	1	$35,113	1	$35,113	$17,434
OS: XXX Lims1 Subscrptn, 1yr.	DUS20031x-TSP	1	$3,173	2	$6,346	
APP: XXX Db Instance Lic's.	510-04170	2	$5,315	1	$5,315	** Incl. below
APP: XXX Db Client License	359-0W11	2	$156	55	$8,580	** Incl. below
ACC: XXX C+	254-00170	2	$109	1	$109	** Incl. below
ACC: XXX Basic	04x-0036	2	$109	1	$109	** Incl. below
SRvC: XXX Problem Resolution Services **		2	$243	1		$243
			Software Subtotal		$55,511	$17,669
VAR: 10% Volume Cash Discount (Ext Price where Price Source=1)					($118,515)	
			Total USD		$1,120,423	$77,495

Notes:

1. * 10% or minimum 2 spares are added in place of onsite service.
2. H/W & S/W maintenance at 24 x 7 w/4 hr. max. response time for spares.
3. Price Source: 1 = VAR Product Price, XXX Maintenance Price
   2 = YYY   3 = WebSpecials

3-Year Cost of Ownership:	$1,197,918	USD
QphDS @ 1000GB:	30,013.4	
$ / QphDS @ 1000GB:	$39.92	USD

Benchmark result and test methodology audited by the Auditor, Audit R Us, Inc.

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

Number of Query Streams: 11

Start and End Times

	Start Date	Start Time	End Date	End Time	Elapsed Time
Database Load	11/7/2008	0:00:10	11/7/2008	1:01:10	1:00:00
Power Test	11/7/2008	1:01:10	11/7/2008	1:01:10	1:00:00
Throughput Run 1	11/7/2008	2:01:10	11/7/2008	1:01:10	1:00:00
Query Run 1	11/7/2008	3:01:10	11/7/2008	1:01:10	1:00:00
Refresh Run 1	11/7/2008	4:01:10	11/7/2008	1:01:10	1:00:00
Throughput Run 2	11/7/2008	5:01:10	11/7/2008	1:01:10	1:00:00
Query Run 2	11/7/2008	6:01:10	11/7/2008	1:01:10	1:00:00
Refresh Run 2	11/7/2008	7:01:10	11/7/2008	1:01:10	1:00:00

TPC-DS Timing Intervals (in seconds)

Query	Minimum		25th Percentile		Median		75th percentile		Maximum	
	Run1	Run2	Run1	Run2	Run1	Run2	Run1	Run2	Run1	Run2
1	7.8	8.8	10.2	13.4	9.1	11.3	10.1	10.8	30.4	32.2
2	1.4	1.6	1.8	2.4	1.6	2.1	1.8	2.0	5.5	5.9
3	2.7	3.1	3.5	4.7	3.2	3.9	3.5	3.8	10.6	11.2
4	38.3	43.0	49.6	65.6	44.3	55.1	49.1	52.7	148.3	157.2
5	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
6	0.7	0.8	0.9	1.2	0.8	1.0	0.9	1.0	2.8	2.9
7	6.1	6.8	7.9	10.4	7.0	8.7	7.8	8.4	23.5	24.9
8	51.3	57.7	66.5	88.0	59.4	73.9	65.9	70.7	199.0	210.9
9	38.3	43.0	49.6	65.6	44.3	55.1	49.1	52.7	148.3	157.2
10	63.6	71.4	82.4	109.0	73.5	91.5	81.6	87.6	246.4	261.2
11	6.2	6.9	8.0	10.6	7.1	8.9	7.9	8.5	23.9	25.4
12	2.6	2.9	3.4	4.5	3.0	3.8	3.4	3.6	10.1	10.7
13	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
14	38.3	43.0	49.6	65.6	44.3	55.1	49.1	52.7	148.3	157.2
15	76.6	86.1	99.3	131.4	88.7	110.3	98.4	105.6	297.1	314.9
16	4.0	4.5	5.2	6.9	4.7	5.8	5.2	5.6	15.7	16.6
17	0.6	0.7	0.8	1.0	0.7	0.9	0.8	0.8	2.3	2.4
18	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
19	4.2	4.7	5.4	7.1	4.8	6.0	5.3	5.7	16.1	17.1
20	41.0	46.0	53.1	70.3	47.4	59.0	52.6	56.5	158.9	168.4
21	41.9	47.1	54.4	71.9	48.5	60.4	53.8	57.8	162.6	172.3
22	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
23	6.4	7.2	8.3	11.0	7.4	9.2	8.2	8.8	24.9	26.4
24	64.9	72.9	84.1	111.3	75.1	93.3	83.3	89.4	251.5	266.5
25	93.7	105.3	121.5	160.8	108.5	134.9	120.3	129.2	363.4	385.2
26	41.9	47.1	54.4	71.9	48.5	60.4	53.8	57.8	162.6	172.3
27	10.6	11.9	13.7	18.1	12.2	15.2	13.6	14.6	41.0	43.4
28	41.0	46.0	53.1	70.3	47.4	59.0	52.6	56.5	158.9	168.4
29	1.1	1.2	1.4	1.8	1.2	1.5	1.4	1.5	4.1	4.4
30	525.6	590.5	681.4	901.5	608.2	756.4	674.7	724.4	2037.4	2159.7
31	104.4	117.3	135.4	179.1	120.8	150.3	134.1	143.9	404.8	429.1
32	77.9	87.6	101.0	133.7	90.2	112.2	100.1	107.4	302.1	320.2
33	76.4	85.8	99.0	131.0	88.4	109.9	98.1	105.3	296.1	313.9
34	14.6	16.4	18.9	25.1	16.9	21.0	18.8	20.1	56.6	60.0
35	52.7	59.3	68.4	90.5	61.0	75.9	67.7	72.7	204.5	216.8

40	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
41	52.7	59.3	68.4	90.5	61.0	75.9	67.7	72.7	204.5	216.8
42	9.3	10.4	12.0	15.9	10.7	13.3	11.9	12.8	35.9	38.1
43	6.7	7.5	8.6	11.4	7.7	9.6	8.5	9.2	25.8	27.3
44	9.1	10.3	11.9	15.7	10.6	13.2	11.7	12.6	35.5	37.6
45	1.1	1.2	1.4	1.8	1.2	1.5	1.4	1.5	4.1	4.4
46	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
47	29.0	32.6	37.6	49.7	33.5	41.7	37.2	40.0	112.4	119.1
48	77.7	87.3	100.7	133.3	89.9	111.8	99.7	107.1	301.2	319.3
49	92.2	103.6	119.5	158.1	106.7	132.7	118.4	127.1	357.4	378.8
50	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
51	65.9	74.1	85.5	113.1	76.3	94.9	84.6	90.9	255.6	270.9
52	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
53	41.9	47.1	54.4	71.9	48.5	60.4	53.8	57.8	162.6	172.3
54	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
55	911.8	1024.4	1182.2	1564.0	1055.1	1312.2	1170.6	1256.8	3534.7	3746.8
56	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
57	51.7	58.1	67.0	88.6	59.8	74.4	66.3	71.2	200.3	212.4
58	51.6	57.9	66.8	88.4	59.7	74.2	66.2	71.1	199.9	211.9
59	6.4	7.2	8.3	11.0	7.4	9.2	8.2	8.8	24.9	26.4
60	5.0	5.6	6.5	8.6	5.8	7.2	6.4	6.9	19.3	20.5
61	3.8	4.3	4.9	6.5	4.4	5.5	4.9	5.2	14.7	15.6
62	52.7	59.3	68.4	90.5	61.0	75.9	67.7	72.7	204.5	216.8
63	6.4	7.2	8.3	11.0	7.4	9.2	8.2	8.8	24.9	26.4
64	53.8	60.5	69.8	92.3	62.3	77.4	69.1	74.2	208.6	221.1
65	385.4	433.0	499.7	661.1	446.0	554.6	494.8	531.2	1494.0	1583.6
66	9.1	10.3	11.9	15.7	10.6	13.2	11.7	12.6	35.5	37.6
67	105.6	118.7	136.9	181.2	122.2	152.0	135.6	145.6	409.4	434.0
68	92.4	103.8	119.8	158.5	107.0	133.0	118.7	127.4	358.3	379.8
69	91.0	102.2	118.0	156.1	105.3	131.0	116.8	125.4	352.8	373.9
70	91.1	102.4	118.1	156.3	105.4	131.1	117.0	125.6	353.2	374.4
71	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
72	41.9	47.1	54.4	71.9	48.5	60.4	53.8	57.8	162.6	172.3
73	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
74	9.1	10.3	11.9	15.7	10.6	13.2	11.7	12.6	35.5	37.6
75	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
76	7.8	8.8	10.2	13.4	9.1	11.3	10.1	10.8	30.4	32.2
77	51.6	57.9	66.8	88.4	59.7	74.2	66.2	71.1	199.9	211.9
78	7.8	8.8	10.2	13.4	9.1	11.3	10.1	10.8	30.4	32.2
79	5.0	5.6	6.5	8.6	5.8	7.2	6.4	6.9	19.3	20.5
80	3.8	4.3	4.9	6.5	4.4	5.5	4.9	5.2	14.7	15.6
81	5.2	5.9	6.8	9.0	6.0	7.5	6.7	7.2	20.3	21.5
82	6.4	7.2	8.3	11.0	7.4	9.2	8.2	8.8	24.9	26.4
83	53.8	60.5	69.8	92.3	62.3	77.4	69.1	74.2	208.6	221.1
84	385.4	433.0	499.7	661.1	446.0	554.6	494.8	531.2	1494.0	1583.6
85	9.1	10.3	11.9	15.7	10.6	13.2	11.7	12.6	35.5	37.6
86	105.6	118.7	136.9	181.2	122.2	152.0	135.6	145.6	409.4	434.0
87	92.4	103.8	119.8	158.5	107.0	133.0	118.7	127.4	358.3	379.8
88	42.1	47.2	54.5	72.1	48.7	60.5	54.0	58.0	163.0	172.8
89	91.1	102.4	118.1	156.3	105.4	131.1	117.0	125.6	353.2	374.4
90	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
91	51.7	58.1	67.0	88.6	59.8	74.4	66.3	71.2	200.3	212.4
92	5.3	6.0	6.9	9.2	6.2	7.7	6.9	7.4	20.7	22.0
93	41.0	46.0	53.1	70.3	47.4	59.0	52.6	56.5	158.9	168.4
94	4.0	4.5	5.2	6.9	4.7	5.8	5.2	5.6	15.7	16.6
95	42.2	47.4	54.7	72.3	48.8	60.7	54.1	58.1	163.5	173.3
96	41.0	46.0	53.1	70.3	47.4	59.0	52.6	56.5	158.9	168.4
97	4.2	4.7	5.4	7.1	4.8	6.0	5.3	5.7	16.1	17.1
98	552.9	621.2	716.8	948.4	639.8	795.7	709.8	762.1	2143.4	2272.0
99	76.6	86.1	99.3	131.4	88.7	110.3	98.4	105.6	297.1	314.9

Appendix F: Tool Set Requirements

F.1   Introduction

In addition to this document, TPC-DS relies on material that is only available electronically. While not included in the printed version of the specification, this "soft appendix" is integral to the submission of a compliant TPC-DS benchmark result.

F.2   Availability

Need to confirm and any other legalese with TPC

The electronically available specification content may be downloaded from the TPC-DS section of the TPC web site located on the TPC website (http://www.tpc.org) free of charge. It is solely intended for use in conjunction with the execution of a TPC-DS benchmark. Any other use is prohibited without the express, prior written consent of the TPC.

F.3   Compatibility

This material is maintained, versioned and revised independently of the specification itself. It is the benchmark sponsor's responsibility to assure that any benchmark submission relies on a revision of the soft appendix that is compliant with the revision of the TPC-DS specification against which the result is being submitted.

The soft appendix includes a version number similar to that used in the specification, with a major version number, a minor version number and a third tier level, each separated by a decimal point. The major and minor revision numbers are tied to those of the TPC-DS specification with which the soft appendix is compliant. The third tier level of the soft appendix is incremented whenever the appendix itself is updated, and is independent of revision changes or updates to the specification.

A revision of the soft appendix may be used to submit a TPC-DS benchmark result provided that the major revision number of the soft appendix matches that of a specification revision that is eligible for benchmark submission;

Comment: The intent of this clause is to allow for the possibly lengthy tuning and preparation cycle that precedes a benchmark submission, during which a third tier revision could be released.

Benchmark sponsors are encouraged to use the most recent patch level of a given soft appendix version, as it will contain the latest clarifications and bug fixes, but any third tier level may be used to produce a compliant benchmark submission as long as the prior conditions are met.

Appendix G: XML Schema Guide

G.1   Overview

The schema of the ES.xml document is defined by the XML schema document tpcds-es.xsd available at located on the TPC website (http://www.tpc.org). The ES.xml file must conform to the tpcds-es.xsd (established by XML schema validation).

G.2   Schema Structure

An XML document conforming to the tpcds-es.xsd schema contains a single element named tpcdsResult of type RootType. The main complex types are explained in the sections below. The other types not included here can be found in tpcds-es.xsd.

G.3   The RootType contains the following attributes:

Attribute	Type	Description
SponsorName	string	The sponsor's name.
SystemIdentification	string	The name of measured server.
SpecVersion	SpecVersionType	*TPC-DS SPECIFICATION VERSION NUMBER UNDER WHICH THE BENCHMARK IS PUBLISHED*
PricingSpecVersion	SpecVersionType	*TPC-PRICING SPECIFICATION VERSION NUMBER UNDER WHICH THE BENCHMARK IS PUBLISHED*
ReportDate	date	*THE DATE THAT THE RESULT IS SUBMITTED TO THE TPC.*
RevisionDate	date	*THE DATE THAT A REVISION IS SUBMITTED TO THE TPC, IF APPLICABLE; OTHERWISE OMITTED.*
AvailabilityDate	date	Availability Date (see TPC Pricing Specification)
TpcdsThroughput	tpcdsType	Reported Throughput in QphDS@SF (see Clause 7.6)
PricePerf	PriceType	Price/Performance Metric ($/QphDS@SF)
Currency	CurrencyType	The currency in which the result is priced.
TotalSystemCost	PriceType	Total System Cost (see TPC Pricing Specification)
AuditorName	AuditorType	The name of the Auditor who certified the result.
Cluster	YesNoType	"Y" or "N" indicating if the result was implemented on a clustered server configuration.
AllRaidProtected	YesNoType	"Y" or "N" indicating if the result was implemented on a server with raid protection at all levels.
SchemaVersion	SchemaVersionType	The schema version, initially "1.0".

G.4   The RootType contains the following elements:

Element	Type	Description
DBServer	DBServerType	The DBServer element contains information about the database server.
StorageSubsystem	StorageSubsystemType	The StorageSubsystem element contains information about the storage subsystem used for the benchmark run.

DatabaseLoad	DatabaseLoadRunDetailType	The DatabaseLoad element contains information about database load run performance.
PowerRun	PowerRunDetailType	The PowerRun element contains information about the Power Run performance.
QueryRun1	QueryRunDetailType	The QueryRun1 element contains information about the Throughput Test 1 performance.
RefreshGroup1	RefreshGroupDetailType	The RefreshRun1 element contains information about the Refresh Run 1 performance.
QueryRun2	QueryRunDetailType	The QueryRun2 element contains information about the Throughput Test 2 performance.
RefreshGroup2	RefreshGroupDetailType	The RefreshRun2 element contains information about the Refresh Run 2 performance.

G.5   The DBServerType contains the following attributes:

Attribute	Type	Description
DBName	string	THE NAME OF THE DATABASE MANAGEMENT SYSTEM (DBMS).
DBVersion	string	The version of the DBMS.
DBMiscInfo	string	ANY MISCELLANEOUS INFORMATION NEEDED TO INDICATE PRECISELY WHICH VERSION OF THE DBMS WAS TESTED (E.G., "SERVICE PACK 1" OR "BUILD 1298").
OSName	string	THE NAME OF THE OPERATING SYSTEM (OS) ON WHICH THE DBMS WAS RUNNING.
OSVersion	string	THE OS VERSION.
OSMiscInfo	string	ANY MISCELLANEOUS INFORMATION NEEDED TO INDICATE PRECISELY WHICH VERSION OF THE OS WAS TESTED (E.G., "SERVICE PACK 1" OR "BUILD 1298").
ProcessorName	string	THE PROCESSOR NAME AND TYPE (E.G., "INTEL XEON 2.8 GHZ")
ProcessorCount	positiveInteger	THE TOTAL NUMBER OF PROCESSORS IN THE Database Server.
CoreCount	positiveInteger	THE TOTAL NUMBER OF CORES IN THE Database Server.
ThreadCount	positiveInteger	THE TOTAL NUMBER OF THREADS IN THE Database Server.
Memory	decimal	THE AMOUNT OF MEMORY (IN GB) CONFIGURED ON THE Database Server.
InitialDBSize	positiveInteger	Initial Database Size in GB
RedundancyLevel	string	The Redundancy Level
SpindleCount	positiveInteger	Priced number of Durable Media (disks) on the Database Server.
SpindleSize	positiveInteger	Size of the priced Durable Media (disks) on the Database Server.
HostBusAdapterCount	positiveInteger	Number of host adapters in the priced system configuration

-

G.6   The DBServerType has the following elements:

Element	Type	Description
PerNodeHardware	PerNodeHardwareType	*THE HARDWARE CONFIGURATION USED FOR THE NODES.* *NOTE: MULTIPLE OCCURRENCES POSSIBLE.*

G.7   The StorageSubsystemType contains the following attributes:

Attribute	Type	Description
StorageSubsystem	string	Description or name of the storage subsystem
RaidLevel	string	RAID level used for this storage subsystem
ArrayCount	positiveInteger	Number of storage arrays used in the priced configuration
SpindleTechnology	string	Description of the durable media technology used in the priced configuration
SpindleSize	double	Size of the disk
SpindleRPM	positiveInteger	Rotations per minute of the spindles
TotalMassStorage	double	Total amount of storage provided by the storage subsystem

G.8   The StorageSubsystemType contains the following elements:

Element	Type	Description
StorageArray	StorageArrayType	Description of the storage array
StorageSwitch	StorageSwitchType	Description of the storage switch

G.9   The StorageArrayType consists of the following attributes:

Attribute	Type	Description
RaidLevel	string	*RAID LEVEL USED SPECIFICALLY FOR THIS STORAGE ARRAY*
SpindleTechnology	string	*SPINDLE TECHNOLOGY USED SPECIFICALLY FOR THIS ARRAY*
SpindleCount	positiveInteger	*NUMBER OF SPINDLES IN THIS STORAGE ARRAY*
SpindleRPM	positiveInteger	*ROTATIONS PER MINUTE OF THE SPINDLES USED SPECIFICALLY FOR THIS STORAGE ARRAY*

G.10   The StorageSwitchType consists of the following attributes:

Attribute	Type	Description
StorageSwitchDescription	string	*DESCRIPTION OF THE STORAGE SWITCH.*
StorageSwitchCount	positiveInteger	*NUMBER OF THE STORAGE SWITCH IN THE CONFIGURATION*
StorageSwitchTechnology	string	*DESCRIPTION OF THE STORAGE SWITCH TECHNOLOGY*

G.11   The DatabaseLoadRunDetailType contains the following attributes:

Attribute	Type	Description
LoadTimeIncludesBackup	YesNoType	*'Y' OR 'N' DEPENDING ON IF THE DATABASE LOAD TIMING INCLUDES THE TIME FOR PERFORMING A BACKUP.*

G.12   The DatabaseLoadRunDetailType contains the following elements:

Element	Type	Description
RunTiming	RunTimingDataType	*THE RUN TIMING ELEMENT CONTAINS TIMING INFORMATION SUCH AS START AND END TIMES OF THE DATABASE LOAD RUN.*

G.13   The PowerRunDetailType contains the following elements:

Element	Type	Description
RunTiming	RunTimingDataType	*THE RUN TIMING ELEMENT CONTAINS TIMING INFORMATION SUCH AS START AND END TIMES OF THE POWER RUN.*
PowerQuery	PowerQueryDataType	*A QUERY ELEMENT FOR EACH QUERY IN THE BENCHMARK REPORTS ON PERFORMANCE OF THIS QUERY IN THE POWER RUN.*

G.14   The RefreshGroupDetailType contains the following elements:

Element	Type	Description
RunTiming	RunTimingDataType	*THE RUN TIMING ELEMENT CONTAINS TIMING INFORMATION SUCH AS START AND END TIMES OF A REFRESH RUN.*
RefreshFunction	RefreshDataType	*A DATA MAINTENANCE FUNCTION ELEMENT FOR EACH DATA MAINTENANCE FUNCTION PERFORMED BY EACH REFRESH RUN*

G.15   The QueryRunDetailType contains the following elements:

Element	Type	Description
RunTiming	RunTimingDataType	*THE RUN TIMING ELEMENT CONTAINS TIMING INFORMATION SUCH AS START AND END TIMES OF A THROUGHPUT TEST.*
Query	QueryDataType	*A QUERY ELEMENT FOR EACH QUERY IN THE RUN.*

G.16   PowerQueryDataType contains the following attributes:

Attribute	Type	Description
QueryNumber	positiveInteger	*QUERY NUMBER*
RT	RTType	*RUN TIME OF THE QUERY AS DEFINED BY CLAUSE 7.4.9*

G.17   QueryDataType contains the following attributes:

Attribute	Type	Description
QueryNumber	positiveInteger	*NUMBER OF THE QUERY*
RTMin	RTType	*MINIMUM RUN TIME OF THIS QUERY ACROSS ALL STREAMS IN THIS RUN*
RTMax	RTTYpe	*MAXIMUM RUN TIME OF THIS QUERY ACROSS ALL STREAMS IN THIS RUN*
RTMedian	RTType	*MEDIAN RUN TIME OF THIS QUERY ACROSS ALL STREAMS IN THIS RUN*
RT25th	RTType	*25-PERCENTILE RUN TIME OF THIS QUERY ACROSS ALL STREAMS IN THIS RUN*
RT75th	RTType	*75-PERCENTILE RUN TIME OF THIS QUERY ACROSS ALL STREAMS IN THIS RUN*

G.18   RefreshDataType contains the following attributes:

Attribute	Type	Description
RefreshFunctionName	RefreshFunctionNameDataType	*NAME OF THE DATA MAINTENANCE FUNCTION. THE NAMES MUST MATCH ONE OF THE VALUES DEFINED IN* Table 5-4.
RT	RTType	*RUN TIME OF THIS DATA MAINTENANCE FUNCTION IN THIS RUN.*