Running the TPC-H Benchmark on Hive[1]

Yuntao Jia

August 10, 2009

1 Introduction

TPC-H (1) is a decision support benchmark. It consists of a suite of business oriented ad-hoc queries and concurrent data modifications. It is widely used today in evaluating the performance of computer systems. Hive (2) is a data warehouse infrastructure built on top of Hadoop (3) that provides tools to enable easy data summarization, adhoc querying and analysis of large datasets data. In this report, we applied the TPC-H benchmark onto Hive for two particular reasons.

- Find new features to put into Hive so that Hive supports common data warehouse queries.
- Measure the performance of Hive to identify bottlenecks and improve Hive based on that information.

In this report, we will introduce our TPC-H benchmark results on a Hive system. We will first explain the system configuration, including both Hardware and software information. We will then describe the timings results and Price/Performance metrics measured on the system. Because TPC-H benchmark is illustrated in the SQL language which is different from the query language of Hive, we will explain how we rewrote the TPC-H queries to run them on Hive. We also included all the Hive queries in appendix.

2 Benchmark Environment

In this benchmark, we have tested Hive of a latest version which is Hive truck version 799148[2]. To support Hive, we have installed Hadoop and the LZO compression library (4). We mostly used the default configurations of Hive and Hadoop coming with their installation packages except a few changes. In particular, we configured Hadoop to use the LZO for compressing the intermediate map output data. Each hadoop task tracker is configured to run up to four map and four reduce tasks simultaneously. All the configurations can be found in the Hive TPCH package (5).

We set up our Hive system on an eleven node cluster running Linux. In order to make Hive/Hadoop running, we selected one node to be the Hadoop master/name node/job tracker. All ten others are used as Hadoop slave nodes which run the data nodes and task trackers. All the nodes are in the same rack. Their hardware and software information are summarized in Table 1.

Table 1 Cluster hardware and software information

CPU	2 Dual-Core AMD Opteron(tm) 280 Processors, 2.4 GHz, cache 1 MB
Memory	8 GB
Disk	4 hard drives, total 1.6 T
Ethernet	1 Gbps
Linux	2.6.12-1.1398_FC4smp
Lzo lib	2.02

[1] If you have any questions or suggestions about this benchmark, please email to Yuntao Jia (yjia@facebook.com) or comment on the Hive JIRA at https://issues.apache.org/jira/browse/HIVE-600.

[2] Latest as of August 2009.

Hadoop	0.18.3
Hive	Hive truck version 799148

3 Timing Results

We tested the Hive system with a 100GB standard dataset generated with the TPC-H DBGEN program (6). We pre-loaded the data onto the Hadoop Distributed File System (HDFS) before running all the queries. We do not consider loading time as part of the benchmark results. All the query results were saved into Hive tables which are stored in HDFS. We included those storing time in the results.

There are totally twenty two queries and two refresh functions in the TPC-H benchmark. Due to limited time, we only considered the queries in this report.

According to the TPC-H Benchmark requirement (7), we ran those queries for 6 times and collected their timings. All the numbers are shown in Table 4. The average query time is also plotted in Figure 1. In Figure 2, we have plotted the results from an IBM eServer 325 System which were reported in July 2003. The configuration of the IBM eServer 325 system is summarized in Table 2 which is similar to our Hive system. More details of the system and their benchmark results are available in their disclosure report (8).

Table 2 IBM eServer 325 system hardware and software information

CPU	2 Dual-Core AMD Opteron(tm) 246 Processors, 2 GHz
Memory	6 GB
Disk	Total 0.7 T
Ethernet	1 Gbps
Linux	SuSE Linux Enterprise Server 8
Database	IBM DB2 UDB 8.1

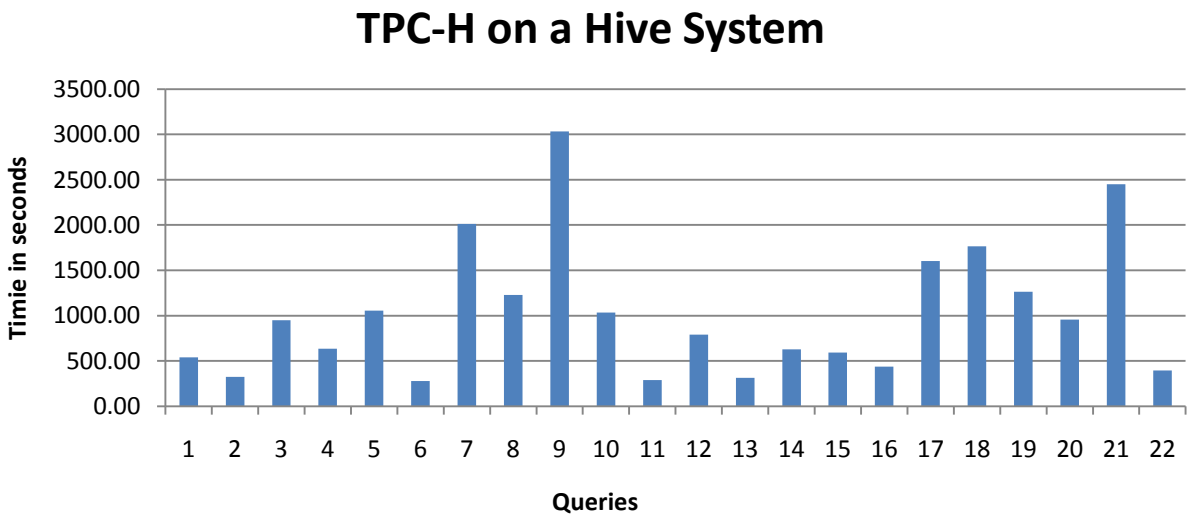

Figure 1 TPC-H benchmark results on a Hive system

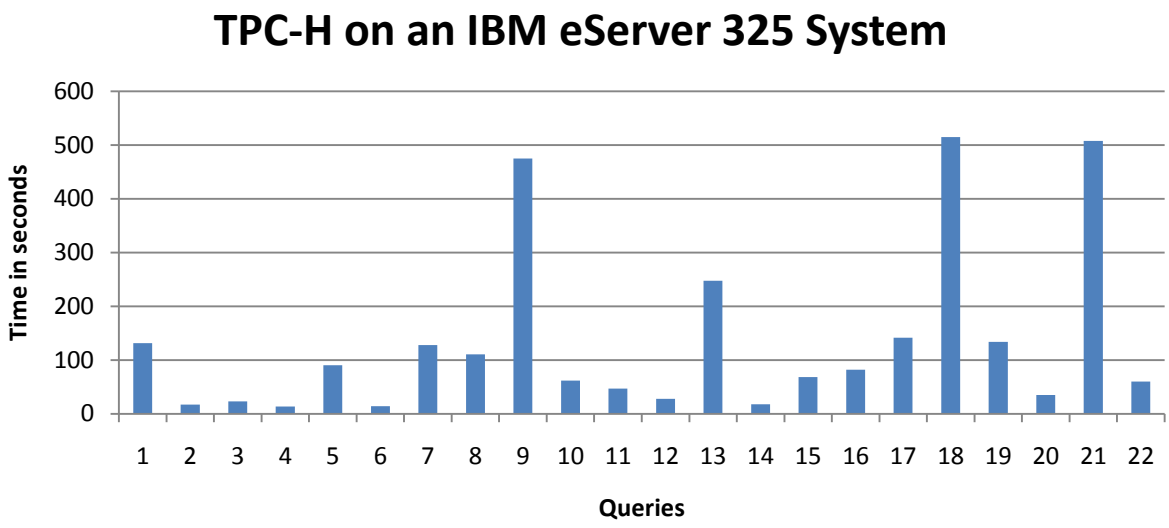

Figure 2 TPC-H Benchmark results on an IBM eServer 325 System

4 Price/Performance Metric

Based the timings, we have computed TPC-H Price/Performance ($/QphH) Metric of the Hive system. Since both Hive and Hadoop are open source software, they are free. All other software come for free too, including Linux and LZO. Each machine costs 3000$ for the hardware. Thus total price of the Hive system is 3000 * 11 = 33000 $.

The TPC-H power is computed using the following equation. Compared to the original equation in the TPC-H benchmark specification (7), we ignored the timings of the refresh functions.

$$\text{TPC-H Power@Size} = \frac{3600 * \text{SF}}{\sqrt[22]{\prod_{i=1}^{i=22} QI(i,0)}}$$

The computed Price/Performance of the Hive system is shown in Table 3.

Table 3 TPC-H Price/Performace Metric of the Hive system

TPC-H Power@Size	TPC-H Throughput@Size	QphH@Size	Price	Price/Performance
436.2926511	350.916137	391.282675	33000$	84.33800449

Table 4 TPC-H Benchmark timings on a Hive system

Timings	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Average
Query 1	524.11	533.09	519.77	582.44	504.11	583.39	541.15
Query 2	340.01	362.01	312.33	313.33	304.06	314.12	324.31
Query 3	979.39	976.28	930.50	920.26	922.41	971.27	950.02
Query 4	628.56	660.64	613.54	628.59	651.7	634.75	636.30
Query 5	1022.49	1082.55	1028.34	1024.36	1073.64	1104.8	1056.03
Query 6	276.87	279.36	276.31	261.78	281.87	286.91	277.18
Query 7	1956.39	1983.31	1976.32	2091.78	2048.39	2013.43	2011.60
Query 8	1195.95	1313.96	1171.67	1199.73	1208.92	1275.9	1227.69
Query 9	2913.2	3141.66	2906.19	3087.54	3072.56	3075.6	3032.79
Query 10	1045.55	1024.37	1011.38	1022.43	1040.41	1065.45	1034.93
Query 11	285.81	285.31	295.92	285.91	280.86	290.89	287.45
Query 12	790.86	782.93	796.05	799.91	784.87	780.99	789.27
Query 13	312.91	309.9	300.92	334.97	299.9	320.93	313.26
Query 14	620.54	617.58	636.54	608.5	630.91	645.91	626.66
Query 15	595.45	597.59	562.42	598.61	591.5	598.97	590.76
Query 16	407.13	469.28	422.12	425.17	472.25	428.19	437.36
Query 17	1573.25	1613.55	1570.50	1620.59	1600.62	1636.62	1602.52
Query 18	1706.73	1817.61	1746.08	1781.09	1766.01	1778.07	1765.93
Query 19	1258.83	1264.41	1253.80	1264.88	1273.91	1274.97	1265.13
Query 20	936.18	969.26	962.28	939.21	929.38	995.37	955.28
Query 21	2419.21	2459.56	2416.42	2450.3	2472.59	2483.36	2450.24
Query 22	332.97	434.15	395.19	408.1	398.29	393.1	393.63

5 Rewriting TPC-H Queries

The TPC-H queries are described with the SQL language. Hive provides a similar query language called Hive QL (9). It does not support all features in SQL yet. However, most TPC-H queries can be rewritten in Hive QL without changing the semantics. In particular, eleven of them require small modifications (e.g. selecting from multiple tables are rewritten to joins); six of them need moderate changes (e.g. sub-queries are rewritten to individual queries) and the remaining five require relative large changes (e.g. UDFs rewritten with individual queries).

In this section, we will explain why and how TPC-H queries are rewritten in Hive QL. We will go through several TPC-H queries as examples and describe the rewritten queries in Hive QL. Since the original queries are usually long and complex, we have neglected some irrelevant details so that we can focus on the changes. The full TPC-H queries can be found in the TPC-H Benchmark specification. The full Hive queries are described in appendix.

5.1 Shipping priority query (Q3)

Hive QL does not support selecting from multiple tables. So in this query, selecting from multiple tables is rewritten to joins and "where" clauses become "on" clauses in the joins. This change is very common across all the queries. The original TPC-H SQL query is:

```
Select
  ...
from
  customer,
  orders,
  lineitem
where
  c_mktsegment = '[SEGMENT]'
  and c_custkey = o_custkey
  and l_orderkey = o_orderkey
  and o_orderdate < date '[DATE]'
  and l_shipdate > date '[DATE]'
...
```

The rewritten Hive query looks like:

```
Select
  ...
from
  customer c join orders o
  on
    c.c_mktsegment = '[SEGMENT]'
    and c.c_custkey = o.o_custkey
    and o.o_orderdate < date '[DATE]'
  join lineitem l
  on
    l.l_orderkey = o.o_orderkey
    and l.l_shipdate > date '[DATE]'
...
```

5.2 Minimum Cost Supplier Query (Q2)

Because sub queries in Hive QL return only tables but not values. So sub query expecting values are rewritten to separate queries. Take this TPC-H query as an example, the original query is

```
Select
  ...
from
  part,
  partsupp,
  ...
where
  ...
  and ps_supplycost = (
    select
      min(ps_supplycost)
    from
      partsupp,
      ...
    where
      p_partkey = ps_partkey
      and ...
  )
...
```

The sub query is rewritten to a separate query and the result is saved to a table named “tmp”. Please note that since “ps_supplycost” is grouped based on “p_partkey”/”ps_partkey” in the sub query, we have to explicitly specify the “group by” operation in the rewritten query.

```
Insert overwrite table tmp
select
  min(ps_supplycost) as min_ps_supplycost,
  ps_partkey
from
  partsupp ps join ...
  ...
group by ps_partkey
```

After rewritten, the main query looks like follows. A “join” with table “tmp” is added to account for the original sub query.

```
Select
  ...
from
  part p join partsupp ps
  on
    ...
  join tmp t
  on
    ps.ps_supplycost = t.min_ps_supplycost
    and ps.ps_partkey = t.ps_partkey
  join ...
...
```

5.3 Order Priority Checking Query (Q4)

Currently, Hive QL does not support "exist" which is a user defined function (UDF). So the “exist” clause in this TPC-H query is rewritten to a separate query. The original query is

```
Select
  ...
from
  orders
where
  ...
  and exists (
    select
      *
    from
      lineitem
    where
      l_orderkey = o_orderkey
      and ...
  )
...
```

The “exist” clause is rewritten to a sub query as follows and the result is saved to a table named “tmp”. For each “l_orderkey” we only need to store one record. So we added a keyword “DISTINCH” before it. This has the same effect as the “group by” operator.

```
insert overwrite table tmp
select
  DISTINCT l_orderkey
from
  lineitem
where
  ...
```

The main query is rewritten as follows. We added a join with table “tmp” to account for the “exist” UDF.

```
Select
  ...
from
  orders o join tmp t
  on
    o.o_orderkey = t.l_orderkey
    and ...
...
```

5.4 Volume Shipping Query (Q7)

This TPC-H query is first rewritten to multiple joins. It turns out that one of the joins includes a “or” clause. However, Hive QL currently does not support "or" clauses in joins. As a result, that join is replaced with a “UNION ALL” of two joins. The original TPC-H SQL query is:

```
Select
  ...
from
  nation n1,
  nation n2,
  supplier,
  customer,
  ...
where
  (
    (n1.n_name = '[NATION1]' and n2.n_name = '[NATION2]')
    or (n1.n_name = '[NATION2]' and n2.n_name = '[NATION1]')
  )
  and s_nationkey = n1.n_nationkey
  and c_nationkey = n2.n_nationkey
  and ...
```

The join between “n1” and “n2” with a “or” clause is rewritten to a separate query and the result is saved to a table named “tmp”.

```
Insert overwrite table tmp
select * from(
  select
    n1.n_name as supp_nation,
    n2.n_name as cust_nation,
    n1.n_nationkey as supp_nationkey,
    n2.n_nationkey as cust_nationkey
  from
    nation n1 join nation n2
    on
      n1.n_name = '[NATION1]'
      and n2.n_name = '[NATION2]'
  UNION ALL
  select
    n1.n_name as supp_nation,
    n2.n_name as cust_nation,
    n1.n_nationkey as supp_nationkey,
    n2.n_nationkey as cust_nationkey
  from
    nation n1 join nation n2
    on
      n1.n_name = '[NATION2]'
      and n2.n_name = '[NATION1]'
) a
```

The main query is rewritten as follows.

```
Select
  ...
from
  supplier s join tmp t
  on
    s.s_nationkey = t.supp_nationkey
  join customer c
  on
    c.c_nationkey = t.cust_nationkey
    and ..
  join ...
```

6 Acknowledgement

I would like to thank the data infrastructure team in facebook for their valuable discussions. Particularly, I thank Zheng Shao for helping me rewrite many of the queries and revising the report. I also thank Rama Ramasamy for helping me set up the cluster.

7 Appendix

We have described all the rewritten Hive queries in this section. There are totally twenty two of them. Please note that "-- comment" stands for comments in the Hive QL.

7.1 Pricing Summary Report Query (Q1)

```
DROP TABLE lineitem;
DROP TABLE q1_pricing_summary_report;

-- create tables and load data
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
-- create the target table
CREATE TABLE q1_pricing_summary_report ( L_RETURNFLAG STRING, L_LINESTATUS STRING, SUM_QTY DOUBLE,
SUM_BASE_PRICE DOUBLE, SUM_DISC_PRICE DOUBLE, SUM_CHARGE DOUBLE, AVE_QTY DOUBLE, AVE_PRICE DOUBLE,
AVE_DISC DOUBLE, COUNT_ORDER INT);
-- the query
INSERT OVERWRITE TABLE q1_pricing_summary_report
SELECT
  L_RETURNFLAG, L_LINESTATUS, SUM(L_QUANTITY), SUM(L_EXTENDEDPRICE), SUM(L_EXTENDEDPRICE*(1-
L_DISCOUNT)), SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT)*(1+L_TAX)), AVG(L_QUANTITY),
AVG(L_EXTENDEDPRICE), AVG(L_DISCOUNT), COUNT(1)
FROM
  lineitem
WHERE
  L_SHIPDATE<='1998-09-02'
GROUP BY L_RETURNFLAG, L_LINESTATUS
ORDER BY L_RETURNFLAG, L_LINESTATUS;
```

7.2 Minimum Cost Supplier Query (Q2)

```
DROP TABLE part;
DROP TABLE supplier;
DROP TABLE partsupp;
DROP TABLE nation;
DROP TABLE region;
DROP TABLE q2_minimum_cost_supplier;
DROP TABLE q2_minimum_cost_supplier_tmp1;
DROP TABLE q2_minimum_cost_supplier_tmp2;
-- create the tables and load the data
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table partsupp (PS_PARTKEY INT, PS_SUPPKEY INT, PS_AVAILQTY INT, PS_SUPPLYCOST
DOUBLE, PS_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
LOCATION'/tpch/partsupp';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
create external table region (R_REGIONKEY INT, R_NAME STRING, R_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/region';
-- create result tables
create table q2_minimum_cost_supplier_tmp1 (s_acctbal double, s_name string, n_name string,
p_partkey int, ps_supplycost double, p_mfgr string, s_address string, s_phone string, s_comment
string);
create table q2_minimum_cost_supplier_tmp2 (p_partkey int, ps_min_supplycost double);
```

```
create table q2_minimum_cost_supplier (s_acctbal double, s_name string, n_name string, p_partkey
int, p_mfgr string, s_address string, s_phone string, s_comment string);
-- the query
insert overwrite table q2_minimum_cost_supplier_tmp1
select
  s.s_acctbal, s.s_name, n.n_name, p.p_partkey, ps.ps_supplycost, p.p_mfgr, s.s_address,
s.s_phone, s.s_comment
from
  nation n join region r
  on
    n.n_regionkey = r.r_regionkey and r.r_name = 'EUROPE'
  join supplier s
  on
    s.s_nationkey = n.n_nationkey
  join partsupp ps
  on
    s.s_suppkey = ps.ps_suppkey
  join part p
  on
    p.p_partkey = ps.ps_partkey and p.p_size = 15 and p.p_type like '%BRASS' ;
insert overwrite table q2_minimum_cost_supplier_tmp2
select
  p_partkey, min(ps_supplycost)
from
  q2_minimum_cost_supplier_tmp1
group by p_partkey;
insert overwrite table q2_minimum_cost_supplier
select
  t1.s_acctbal, t1.s_name, t1.n_name, t1.p_partkey, t1.p_mfgr, t1.s_address, t1.s_phone,
t1.s_comment
from
  q2_minimum_cost_supplier_tmp1 t1 join q2_minimum_cost_supplier_tmp2 t2
on
  t1.p_partkey = t2.p_partkey and t1.ps_supplycost=t2.ps_min_supplycost
order by s_acctbal desc, n_name, s_name, p_partkey
limit 100;
```

7.3 Shipping Priority Query (Q3)

```
DROP TABLE orders;
DROP TABLE lineitem;
DROP TABLE customer;
DROP TABLE q3_shipping_priority;
-- create tables and load data
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
-- create the target table
create table q3_shipping_priority (l_orderkey int, revenue double, o_orderdate string,
o_shippriority int);
-- the query
Insert overwrite table q3_shipping_priority
select
  l_orderkey, sum(l_extendedprice*(1-l_discount)) as revenue, o_orderdate, o_shippriority
from
```

```
  customer c join orders o
    on c.c_mktsegment = 'BUILDING' and c.c_custkey = o.o_custkey
  join lineitem l
    on l.l_orderkey = o.o_orderkey
where
  o_orderdate < '1995-03-15' and l_shipdate > '1995-03-15'
group by l_orderkey, o_orderdate, o_shippriority
order by revenue desc, o_orderdate
limit 10;
```

7.4 Order Priority Checking Query (Q4)

```
DROP TABLE orders;
DROP TABLE lineitem;
DROP TABLE q4_order_priority_tmp;
DROP TABLE q4_order_priority;
-- create tables and load data
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
-- create the target table
CREATE TABLE q4_order_priority_tmp (O_ORDERKEY INT);
CREATE TABLE q4_order_priority (O_ORDERPRIORITY STRING, ORDER_COUNT INT);
-- the query
INSERT OVERWRITE TABLE q4_order_priority_tmp
select
  DISTINCT l_orderkey
from
  lineitem
where
  l_commitdate < l_receiptdate;
INSERT OVERWRITE TABLE q4_order_priority
select o_orderpriority, count(1) as order_count
from
  orders o join q4_order_priority_tmp t
  on
    o.o_orderkey = t.o_orderkey and o.o_orderdate >= '1993-07-01' and o.o_orderdate < '1993-10-
01'
group by o_orderpriority
order by o_orderpriority;
```

7.5 Local Supplier Volume Query (Q5)

```
DROP TABLE customer;
DROP TABLE orders;
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE nation;
DROP TABLE region;
DROP TABLE q5_local_supplier_volume;
-- create tables and load data
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
```

```
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
create external table region (R_REGIONKEY INT, R_NAME STRING, R_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/region';

-- create the target table
create table q5_local_supplier_volume (N_NAME STRING, REVENUE DOUBLE);
-- the query
insert overwrite table q5_local_supplier_volume
select
  n_name, sum(l_extendedprice * (1 - l_discount)) as revenue
from
  customer c join
    ( select n_name, l_extendedprice, l_discount, s_nationkey, o_custkey from orders o join
      ( select n_name, l_extendedprice, l_discount, l_orderkey, s_nationkey from lineitem l join
        ( select n_name, s_suppkey, s_nationkey from supplier s join
          ( select n_name, n_nationkey
            from nation n join region r
            on n.n_regionkey = r.r_regionkey and r.r_name = 'ASIA'
          ) n1 on s.s_nationkey = n1.n_nationkey
        ) s1 on l.l_suppkey = s1.s_suppkey
      ) l1 on l1.l_orderkey = o.o_orderkey and o.o_orderdate >= '1994-01-01'
              and o.o_orderdate < '1995-01-01'
    ) o1
    on c.c_nationkey = o1.s_nationkey and c.c_custkey = o1.o_custkey
group by n_name
order by revenue desc;
```

7.6 Forecasting Revenue Change Query (Q6)

```
DROP TABLE lineitem;
DROP TABLE q6_forecast_revenue_change;
-- create tables and load data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
-- create the target table
create table q6_forecast_revenue_change (revenue double);
-- the query
insert overwrite table q6_forecast_revenue_change
select
  sum(l_extendedprice*l_discount) as revenue
from
  lineitem
where
  l_shipdate >= '1994-01-01'
  and l_shipdate < '1995-01-01'
  and l_discount >= 0.05 and l_discount <= 0.07
  and l_quantity < 24;
```

7.7 Volume Shipping Query (Q7)

```
DROP TABLE customer;
DROP TABLE orders;
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE nation;
```

```
DROP TABLE q7_volume_shipping;
DROP TABLE q7_volume_shipping_tmp;
-- create tables and load data
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
-- create the target table
create table q7_volume_shipping (supp_nation string, cust_nation string, l_year int, revenue
double);
create table q7_volume_shipping_tmp(supp_nation string, cust_nation string, s_nationkey int,
c_nationkey int);
-- the query
insert overwrite table q7_volume_shipping_tmp
select
  *
from
  (
    select
      n1.n_name as supp_nation, n2.n_name as cust_nation, n1.n_nationkey as s_nationkey,
      n2.n_nationkey as c_nationkey
    from
      nation n1 join nation n2
      on
        n1.n_name = 'FRANCE' and n2.n_name = 'GERMANY'
    UNION ALL
    select
      n1.n_name as supp_nation, n2.n_name as cust_nation, n1.n_nationkey as s_nationkey,
      n2.n_nationkey as c_nationkey
    from
      nation n1 join nation n2
      on
        n2.n_name = 'FRANCE' and n1.n_name = 'GERMANY'
) a;
insert overwrite table q7_volume_shipping
select
  supp_nation, cust_nation, l_year, sum(volume) as revenue
from
  (
    select
      supp_nation, cust_nation, year(l_shipdate) as l_year,
      l_extendedprice * (1 - l_discount) as volume
    from
      q7_volume_shipping_tmp t join
        (select l_shipdate, l_extendedprice, l_discount, c_nationkey, s_nationkey
         from supplier s join
           (select l_shipdate, l_extendedprice, l_discount, l_suppkey, c_nationkey
            from customer c join
              (select l_shipdate, l_extendedprice, l_discount, l_suppkey, o_custkey
               from orders o join lineitem l
```

```
              on
                o.o_orderkey = l.l_orderkey and l.l_shipdate >= '1995-01-01'
                and l.l_shipdate <= '1996-12-31'
              ) l1 on c.c_custkey = l1.o_custkey
            ) l2 on s.s_suppkey = l2.l_suppkey
          ) l3 on l3.c_nationkey = t.c_nationkey and l3.s_nationkey = t.s_nationkey
   ) shipping
group by supp_nation, cust_nation, l_year
order by supp_nation, cust_nation, l_year;
```

7.8 National Market Share Query (Q8)

```
DROP TABLE customer;
DROP TABLE orders;
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE nation;
DROP TABLE region;
DROP TABLE part;
DROP TABLE q8_national_market_share;
-- create the tables and load the data
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
create external table region (R_REGIONKEY INT, R_NAME STRING, R_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/region';
-- create the result table
create table q8_national_market_share(o_year string, mkt_share double);
-- the query
insert overwrite table q8_national_market_share
select
  o_year, sum(case when nation = 'BRAZIL' then volume else 0.0 end) / sum(volume) as mkt_share
from
  (
    select
      year(o_orderdate) as o_year, l_extendedprice * (1-l_discount) as volume,
      n2.n_name as nation
    from
      nation n2 join
        (select o_orderdate, l_discount, l_extendedprice, s_nationkey
         from supplier s join
          (select o_orderdate, l_discount, l_extendedprice, l_suppkey
           from part p join
             (select o_orderdate, l_partkey, l_discount, l_extendedprice, l_suppkey
              from lineitem l join
                (select o_orderdate, o_orderkey
                 from orders o join
```

```
                  (select c.c_custkey
                   from customer c join
                     (select n1.n_nationkey
                      from nation n1 join region r
                      on n1.n_regionkey = r.r_regionkey and r.r_name = 'AMERICA'
                      ) n11 on c.c_nationkey = n11.n_nationkey
                   ) c1 on c1.c_custkey = o.o_custkey
                ) o1 on l.l_orderkey = o1.o_orderkey and o1.o_orderdate >= '1995-01-01'
                      and o1.o_orderdate < '1996-12-31'
             ) l1 on p.p_partkey = l1.l_partkey and p.p_type = 'ECONOMY ANODIZED STEEL'
          ) p1 on s.s_suppkey = p1.l_suppkey
       ) s1 on s1.s_nationkey = n2.n_nationkey
  ) all_nation
group by o_year
order by o_year;
```

7.9 Product Type Profit Measure Query (Q9)

```
DROP TABLE part;
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE orders;
DROP TABLE partsupp;
DROP TABLE nation;
DROP TABLE q9_product_type_profit;
-- create the tables and load the data
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table partsupp (PS_PARTKEY INT, PS_SUPPKEY INT, PS_AVAILQTY INT, PS_SUPPLYCOST
DOUBLE, PS_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
LOCATION'/tpch/partsupp';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
-- create the result table
create table q9_product_type_profit (nation string, o_year string, sum_profit double);
-- the query
insert overwrite table q9_product_type_profit
select
  nation, o_year, sum(amount) as sum_profit
from
  (
    select
      n_name as nation, year(o_orderdate) as o_year,
      l_extendedprice * (1 - l_discount) -  ps_supplycost * l_quantity as amount
    from
      orders o join
      (select l_extendedprice, l_discount, l_quantity, l_orderkey, n_name, ps_supplycost
       from part p join
         (select l_extendedprice, l_discount, l_quantity, l_partkey, l_orderkey,
                 n_name, ps_supplycost
          from partsupp ps join
```

```
            (select l_suppkey, l_extendedprice, l_discount, l_quantity, l_partkey,
                    l_orderkey, n_name
             from
               (select s_suppkey, n_name
                from nation n join supplier s on n.n_nationkey = s.s_nationkey
               ) s1 join lineitem l on s1.s_suppkey = l.l_suppkey
            ) l1 on ps.ps_suppkey = l1.l_suppkey and ps.ps_partkey = l1.l_partkey
         ) l2 on p.p_name like '%green%' and p.p_partkey = l2.l_partkey
     ) l3 on o.o_orderkey = l3.l_orderkey
  )profit
group by nation, o_year
order by nation, o_year desc;
```

7.10 Returned Item Reporting Query (Q10)

```
DROP TABLE lineitem;
DROP TABLE orders;
DROP TABLE customer;
DROP TABLE nation;
DROP TABLE q10_returned_item;
-- create the tables and load the data
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
-- create the result table
create table q10_returned_item (c_custkey int, c_name string, revenue double, c_acctbal string,
n_name string, c_address string, c_phone string, c_comment string);
-- the query
insert overwrite table q10_returned_item
select
  c_custkey, c_name, sum(l_extendedprice * (1 - l_discount)) as revenue,
  c_acctbal, n_name, c_address, c_phone, c_comment
from
  customer c join orders o
  on
    c.c_custkey = o.o_custkey and o.o_orderdate >= '1993-10-01' and o.o_orderdate < '1994-01-01'
  join nation n
  on
    c.c_nationkey = n.n_nationkey
  join lineitem l
  on
    l.l_orderkey = o.o_orderkey and l.l_returnflag = 'R'
group by c_custkey, c_name, c_acctbal, c_phone, n_name, c_address, c_comment
order by revenue desc
limit 20;
```

7.11 Important Stock Identification Query (Q11)

```
DROP TABLE partsupp;
DROP TABLE supplier;
DROP TABLE nation;
DROP TABLE q11_important_stock;
DROP TABLE q11_part_tmp;
DROP TABLE q11_sum_tmp;
-- create tables and load data
```

```
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
create external table partsupp (PS_PARTKEY INT, PS_SUPPKEY INT, PS_AVAILQTY INT, PS_SUPPLYCOST
DOUBLE, PS_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
LOCATION'/tpch/partsupp';
-- create the target table
create table q11_important_stock(ps_partkey INT, value DOUBLE);
create table q11_part_tmp(ps_partkey int, part_value double);
create table q11_sum_tmp(total_value double);
-- the query
insert overwrite table q11_part_tmp
select
  ps_partkey, sum(ps_supplycost * ps_availqty) as part_value
from
  nation n join supplier s
  on
    s.s_nationkey = n.n_nationkey and n.n_name = 'GERMANY'
  join partsupp ps
  on
    ps.ps_suppkey = s.s_suppkey
group by ps_partkey;
insert overwrite table q11_sum_tmp
select
  sum(part_value) as total_value
from
  q11_part_tmp;
insert overwrite table q11_important_stock
select
  ps_partkey, part_value as value
from
  (
    select ps_partkey, part_value, total_value
    from q11_part_tmp join q11_sum_tmp
  ) a
where part_value > total_value * 0.0001
order by value desc;
```

7.12 Shipping Modes and Order Priority Query (Q12)

```
DROP TABLE lineitem;
DROP TABLE orders;
DROP TABLE q12_shipping;
-- create the tables and load the data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
-- create the result table
create table q12_shipping(l_shipmode string, high_line_count double, low_line_count double);
-- the query
insert overwrite table q12_shipping
select
  l_shipmode,
  sum(case
    when o_orderpriority ='1-URGENT'
         or o_orderpriority ='2-HIGH'
```

```
    then 1
    else 0
    end
  ) as high_line_count,
  sum(case
    when o_orderpriority <> '1-URGENT'
         and o_orderpriority <> '2-HIGH'
    then 1
    else 0
    end
  ) as low_line_count
from
  orders o join lineitem l
  on
    o.o_orderkey = l.l_orderkey and l.l_commitdate < l.l_receiptdate
    and l.l_shipdate < l.l_commitdate and l.l_receiptdate >= '1994-01-01'
    and l.l_receiptdate < '1995-01-01'
where
  l.l_shipmode = 'MAIL' or l.l_shipmode = 'SHIP'
group by l_shipmode
order by l_shipmode;
```

7.13 Customer Distribution Query (Q13)

```
DROP TABLE customer;
DROP TABLE orders;
DROP TABLE q13_customer_distribution;
-- create the tables and load the data
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
-- create the result table
create table q13_customer_distribution (c_count int, custdist int);
-- the query
insert overwrite table q13_customer_distribution
select
  c_count, count(1) as custdist
from
  (select
     c_custkey, count(o_orderkey) as c_count
   from
     customer c left outer join orders o
     on
       c.c_custkey = o.o_custkey and not o.o_comment like '%special%requests%'
   group by c_custkey
   ) c_orders
group by c_count
order by custdist desc, c_count desc;
```

7.14 Promotion Effect Query (Q14)

```
DROP TABLE lineitem;
DROP TABLE part;
DROP TABLE q14_promotion_effect;
-- create the tables and load the data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
```

```
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
-- create the result table
create table q14_promotion_effect(promo_revenue double);
-- the query
insert overwrite table q14_promotion_effect
select
  100.00 * sum(case
               when p_type like 'PROMO%'
               then l_extendedprice*(1-l_discount)
               else 0.0
               end
  ) / sum(l_extendedprice * (1 - l_discount)) as promo_revenue
from
  lineitem l join part p
  on
    l.l_partkey = p.p_partkey and l.l_shipdate >= '1995-09-01' and l.l_shipdate < '1995-10-01';
```

7.15 Top Supplier Query (Q15)

```
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE revenue;
DROP TABLE max_revenue;
DROP TABLE q15_top_supplier;
-- create the tables and load the data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
-- create result tables
create table revenue(supplier_no int, total_revenue double);
create table max_revenue(max_revenue double);
create table q15_top_supplier(s_suppkey int, s_name string, s_address string, s_phone string,
total_revenue double);
-- the query
insert overwrite table revenue
select
  l_suppkey as supplier_no, sum(l_extendedprice * (1 - l_discount)) as total_revenue
from
  lineitem
where
  l_shipdate >= '1996-01-01' and l_shipdate < '1996-04-01'
group by l_suppkey;
insert overwrite table max_revenue
select
  max(total_revenue)
from
  revenue;
insert overwrite table q15_top_supplier
select
  s_suppkey, s_name, s_address, s_phone, total_revenue
from supplier s join revenue r
  on
    s.s_suppkey = r.supplier_no
  join max_revenue m
  on
    r.total_revenue = m.max_revenue
```

```
order by s_suppkey;
```

7.16 Parts/Supplier Relationship Query (Q16)

```
DROP TABLE partsupp;
DROP TABLE part;
DROP TABLE supplier;
DROP TABLE q16_parts_supplier_relationship;
DROP TABLE q16_tmp;
DROP TABLE supplier_tmp;
-- create the tables and load the data
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
create external table partsupp (PS_PARTKEY INT, PS_SUPPKEY INT, PS_AVAILQTY INT, PS_SUPPLYCOST
DOUBLE, PS_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
LOCATION'/tpch/partsupp';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
-- create the result table
create table q16_parts_supplier_relationship(p_brand string, p_type string, p_size int,
supplier_cnt int);
create table q16_tmp(p_brand string, p_type string, p_size int, ps_suppkey int);
create table supplier_tmp(s_suppkey int);
-- the query
insert overwrite table supplier_tmp
select
  s_suppkey
from
  supplier
where
  not s_comment like '%Customer%Complaints%';
insert overwrite table q16_tmp
select
  p_brand, p_type, p_size, ps_suppkey
from
  partsupp ps join part p
  on
    p.p_partkey = ps.ps_partkey and p.p_brand <> 'Brand#45'
    and not p.p_type like 'MEDIUM POLISHED%'
  join supplier_tmp s
  on
    ps.ps_suppkey = s.s_suppkey;
insert overwrite table q16_parts_supplier_relationship
select
  p_brand, p_type, p_size, count(distinct ps_suppkey) as supplier_cnt
from
  (select
     *
   from
     q16_tmp
   where p_size = 49 or p_size = 14 or p_size = 23 or
         p_size = 45 or p_size = 19 or p_size = 3 or
         p_size = 36 or p_size = 9
) q16_all
group by p_brand, p_type, p_size
order by supplier_cnt desc, p_brand, p_type, p_size;
```

7.17 Small-Quantity-Order Revenue Query (Q17)

```
DROP TABLE lineitem;
DROP TABLE part;
DROP TABLE q17_small_quantity_order_revenue;
DROP TABLE lineitem_tmp;
```

```
-- create the tables and load the data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
-- create the result table
create table q17_small_quantity_order_revenue (avg_yearly double);
create table lineitem_tmp (t_partkey int, t_avg_quantity double);
-- the query
insert overwrite table lineitem_tmp
select
  l_partkey as t_partkey, 0.2 * avg(l_quantity) as t_avg_quantity
from
  lineitem
group by l_partkey;
insert overwrite table q17_small_quantity_order_revenue
select
  sum(l_extendedprice) / 7.0 as avg_yearly
from
  (select l_quantity, l_extendedprice, t_avg_quantity from
   lineitem_tmp t join
     (select
        l_quantity, l_partkey, l_extendedprice
      from
        part p join lineitem l
        on
          p.p_partkey = l.l_partkey
          and p.p_brand = 'Brand#23'
          and p.p_container = 'MED BOX'
     ) l1 on l1.l_partkey = t.t_partkey
   ) a
where l_quantity < t_avg_quantity;
```

7.18 Large Volume Customer Query (Q18)

```
DROP TABLE lineitem;
DROP TABLE orders;
DROP TABLE customer;
DROP TABLE q18_tmp;
DROP TABLE q18_large_volume_customer;
-- create the tables and load the data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
-- create the result tables
create table q18_tmp(l_orderkey int, t_sum_quantity double);
create table q18_large_volume_customer(c_name string, c_custkey int, o_orderkey int, o_orderdate
string, o_totalprice double, sum_quantity double);
-- the query
insert overwrite table q18_tmp
select
```

```
  l_orderkey, sum(l_quantity) as t_sum_quantity
from
  lineitem
group by l_orderkey;
insert overwrite table q18_large_volume_customer
select
  c_name,c_custkey,o_orderkey,o_orderdate,o_totalprice,sum(l_quantity)
from
  customer c join orders o
  on
    c.c_custkey = o.o_custkey
  join q18_tmp t
  on
    o.o_orderkey = t.l_orderkey and t.t_sum_quantity > 300
  join lineitem l
  on
    o.o_orderkey = l.l_orderkey
group by c_name,c_custkey,o_orderkey,o_orderdate,o_totalprice
order by o_totalprice desc,o_orderdate
limit 100;
```

7.19 Discounted Revenue Query (Q19)

```
DROP TABLE lineitem;
DROP TABLE part;
DROP TABLE q19_discounted_revenue;
-- create the tables and load the data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table part (P_PARTKEY INT, P_NAME STRING, P_MFGR STRING, P_BRAND STRING, P_TYPE
STRING, P_SIZE INT, P_CONTAINER STRING, P_RETAILPRICE DOUBLE, P_COMMENT STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/part';
-- create the result table
create table q19_discounted_revenue(revenue double);
-- the query
insert overwrite table q19_discounted_revenue
select
  sum(l_extendedprice * (1 - l_discount) ) as revenue
from
  lineitem l join part p
  on
    p.p_partkey = l.l_partkey
where
  (
    p_brand = 'Brand#12'
    and p_container REGEXP 'SM CASE||SM BOX||SM PACK||SM PKG'
    and l_quantity >= 1 and l_quantity <= 11
    and p_size >= 1 and p_size <= 5
    and l_shipmode REGEXP 'AIR||AIR REG'
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
  or
  (
    p_brand = 'Brand#23'
    and p_container REGEXP 'MED BAG||MED BOX||MED PKG||MED PACK'
    and l_quantity >= 10 and l_quantity <= 20
    and p_size >= 1 and p_size <= 10
    and l_shipmode REGEXP 'AIR||AIR REG'
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
```

```
  or
  (
    p_brand = 'Brand#34'
    and p_container REGEXP 'LG CASE||LG BOX||LG PACK||LG PKG'
    and l_quantity >= 20 and l_quantity <= 30
    and p_size >= 1 and p_size <= 15
    and l_shipmode REGEXP 'AIR||AIR REG'
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
```

7.20 Potential Part Promotion Query (Q20)

```
DROP TABLE partsupp;
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE nation;
DROP TABLE q20_tmp1;
DROP TABLE q20_tmp2;
DROP TABLE q20_tmp3;
DROP TABLE q20_tmp4;
DROP TABLE q20_potential_part_promotion;
-- create tables and load data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
create external table partsupp (PS_PARTKEY INT, PS_SUPPKEY INT, PS_AVAILQTY INT, PS_SUPPLYCOST
DOUBLE, PS_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
LOCATION'/tpch/partsupp';
-- create the target table
create table q20_tmp1(p_partkey int);
create table q20_tmp2(l_partkey int, l_suppkey int, sum_quantity double);
create table q20_tmp3(ps_suppkey int, ps_availqty int, sum_quantity double);
create table q20_tmp4(ps_suppkey int);
create table q20_potential_part_promotion(s_name string, s_address string);

-- the query
insert overwrite table q20_tmp1
select distinct p_partkey
from
  part
where
  p_name like 'forest%';
insert overwrite table q20_tmp2
select
  l_partkey, l_suppkey, 0.5 * sum(l_quantity)
from
  lineitem
where
  l_shipdate >= '1994-01-01'
  and l_shipdate < '1995-01-01'
group by l_partkey, l_suppkey;
insert overwrite table q20_tmp3
select
  ps_suppkey, ps_availqty, sum_quantity
from
  partsupp ps join q20_tmp1 t1
```

```
  on
    ps.ps_partkey = t1.p_partkey
  join q20_tmp2 t2
  on
ps.ps_partkey = t2.l_partkey and ps.ps_suppkey = t2.l_suppkey;
insert overwrite table q20_tmp4
select
  ps_suppkey
from
  q20_tmp3
where
  ps_availqty > sum_quantity
group by ps_suppkey;
insert overwrite table q20_potential_part_promotion
select
  s_name, s_address
from
  supplier s join nation n
  on
    s.s_nationkey = n.n_nationkey
    and n.n_name = 'CANADA'
  join q20_tmp4 t4
  on
    s.s_suppkey = t4.ps_suppkey
order by s_name;
```

7.21 Suppliers Who Kept Orders Waiting Query (Q21)

```
DROP TABLE orders;
DROP TABLE lineitem;
DROP TABLE supplier;
DROP TABLE nation;
DROP TABLE q21_tmp1;
DROP TABLE q21_tmp2;
DROP TABLE q21_suppliers_who_kept_orders_waiting;
-- create tables and load data
create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT,
L_QUANTITY DOUBLE, L_EXTENDEDPRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING,
L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING, L_RECEIPTDATE STRING, L_SHIPINSTRUCT
STRING, L_SHIPMODE STRING, L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION '/tpch/lineitem';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
create external table supplier (S_SUPPKEY INT, S_NAME STRING, S_ADDRESS STRING, S_NATIONKEY INT,
S_PHONE STRING, S_ACCTBAL DOUBLE, S_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION '/tpch/supplier';
create external table nation (N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/nation';
-- create target tables
create table q21_tmp1(l_orderkey int, count_suppkey int, max_suppkey int);
create table q21_tmp2(l_orderkey int, count_suppkey int, max_suppkey int);
create table q21_suppliers_who_kept_orders_waiting(s_name string, numwait int);
-- the query
insert overwrite table q21_tmp1
select
  l_orderkey, count(distinct l_suppkey), max(l_suppkey) as max_suppkey
from
  lineitem
group by l_orderkey;
insert overwrite table q21_tmp2
select
  l_orderkey, count(distinct l_suppkey), max(l_suppkey) as max_suppkey
```

```
from
  lineitem
where
  l_receiptdate > l_commitdate
group by l_orderkey;
insert overwrite table q21_suppliers_who_kept_orders_waiting
select
  s_name, count(1) as numwait
from
  (select s_name from
    (select s_name, t2.l_orderkey, l_suppkey, count_suppkey, max_suppkey
     from q21_tmp2 t2 right outer join
      (select s_name, l_orderkey, l_suppkey from
         (select s_name, t1.l_orderkey, l_suppkey, count_suppkey, max_suppkey
          from
            q21_tmp1 t1 join
            (select s_name, l_orderkey, l_suppkey
             from
               orders o join
               (select s_name, l_orderkey, l_suppkey
                from
                  nation n join supplier s
                  on
                    s.s_nationkey = n.n_nationkey
                    and n.n_name = 'SAUDI ARABIA'
                  join lineitem l
                  on
                    s.s_suppkey = l.l_suppkey
                where
                  l.l_receiptdate > l.l_commitdate
                ) l1 on o.o_orderkey = l1.l_orderkey and o.o_orderstatus = 'F'
            ) l2 on l2.l_orderkey = t1.l_orderkey
          ) a
          where
           (count_suppkey > 1) or ((count_suppkey=1) and (l_suppkey <> max_suppkey))
      ) l3 on l3.l_orderkey = t2.l_orderkey
    ) b
    where
     (count_suppkey is null) or ((count_suppkey=1) and (l_suppkey = max_suppkey))
  )c
group by s_name
order by numwait desc, s_name
limit 100;
```

7.22 Global Sales Opportunity Query (Q22)

```
DROP TABLE customer;
DROP TABLE orders;
DROP TABLE q22_customer_tmp;
DROP TABLE q22_customer_tmp1;
DROP TABLE q22_orders_tmp;
DROP TABLE q22_global_sales_opportunity;
-- create tables and load data
create external table customer (C_CUSTKEY INT, C_NAME STRING, C_ADDRESS STRING, C_NATIONKEY INT,
C_PHONE STRING, C_ACCTBAL DOUBLE, C_MKTSEGMENT STRING, C_COMMENT STRING) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/customer';
create external table orders (O_ORDERKEY INT, O_CUSTKEY INT, O_ORDERSTATUS STRING, O_TOTALPRICE
DOUBLE, O_ORDERDATE STRING, O_ORDERPRIORITY STRING, O_CLERK STRING, O_SHIPPRIORITY INT, O_COMMENT
STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION '/tpch/orders';
-- create target tables
create table q22_customer_tmp(c_acctbal double, c_custkey int, cntrycode string);
create table q22_customer_tmp1(avg_acctbal double);
create table q22_orders_tmp(o_custkey int);
```

```
create table q22_global_sales_opportunity(cntrycode string, numcust int, totacctbal double);
-- the query
insert overwrite table q22_customer_tmp
select
  c_acctbal, c_custkey, substr(c_phone, 1, 2) as cntrycode
from
  customer
where
  substr(c_phone, 1, 2) = '13' or
  substr(c_phone, 1, 2) = '31' or
  substr(c_phone, 1, 2) = '23' or
  substr(c_phone, 1, 2) = '29' or
  substr(c_phone, 1, 2) = '30' or
  substr(c_phone, 1, 2) = '18' or
  substr(c_phone, 1, 2) = '17';
insert overwrite table q22_customer_tmp1
select
  avg(c_acctbal)
from
  q22_customer_tmp
where
  c_acctbal > 0.00;
insert overwrite table q22_orders_tmp
select
  o_custkey
from
  orders
group by
  o_custkey;
insert overwrite table q22_global_sales_opportunity
select
  cntrycode, count(1) as numcust, sum(c_acctbal) as totacctbal
from
  (select cntrycode, c_acctbal, avg_acctbal from
   q22_customer_tmp1 ct1 join
    (select cntrycode, c_acctbal from
       q22_orders_tmp ot
       right outer join q22_customer_tmp ct
       on
         ct.c_custkey = ot.o_custkey
     where
       o_custkey is null
    ) ct2
  ) a
where
  c_acctbal > avg_acctbal
group by cntrycode
order by cntrycode;
```

8 Bibliography

1. TPC-H Benchmark. [Online] http://www.tpc.org/tpch/.

2. Hive. [Online] http://hadoop.apache.org/hive/.

3. Hadoop. [Online] http://hadoop.apache.org/.

4. LZO Compression Library. [Online] http://www.oberhumer.com/opensource/lzo/.

5. TPC-H Benchmark Package on Hive. [Online] https://issues.apache.org/jira/browse/HIVE-600.

6. TPC-H DBGEN. [Online] http://www.tpc.org/tpch/spec/tpch_2_8_0.zip.

7. TPC-H Benchmark Document. [Online] http://www.tpc.org/tpch/spec/tpch2.8.0.pdf.

8. **IBM.** TPC BenchmarkTM H Full Disclosure Report for IBM® ™ 325 using IBM DB2® Universal Database 8.1. [Online] 7 2003. http://www.tpc.org/tpch/results/tpch_result_detail.asp?id=103072902.

9. HiveQL Language. [Online] http://wiki.apache.org/hadoop/Hive/LanguageManual.