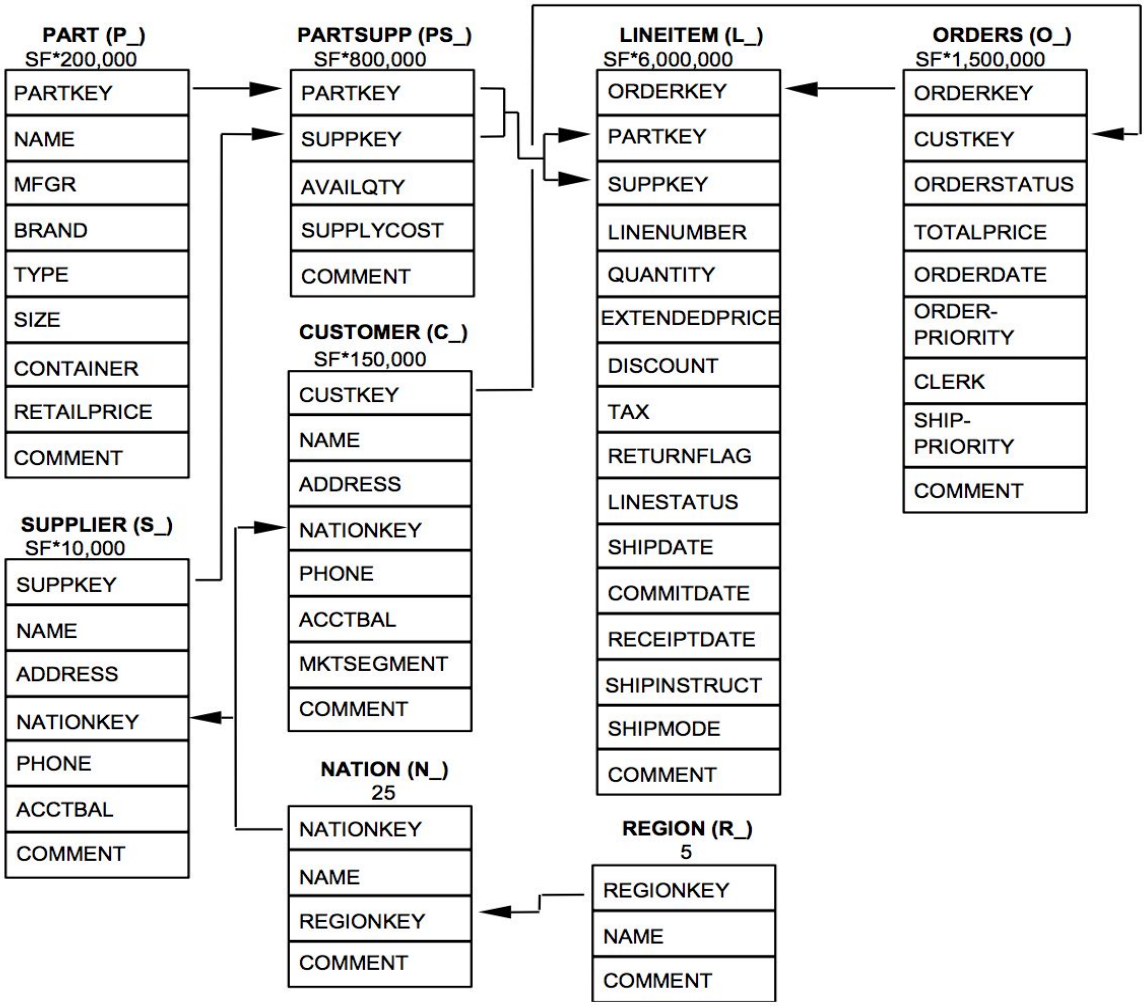
Big Data 2018 - Homework 2

Please download and run the Greenplum Virtual Machine before starting this assignment (see instructions in the class site). The machine comes with a database instance of TPC-H. We're going to introduce below its schema and the tools you'll need for this assignment.

1. The Schema -- TPC-H is a very common benchmark that depicts a realistic sales database using in decision support. More details can be found at http://www.tpc.org/tpch/. The layout is given below

Figure 2: The TPC-H Schema



Legend:

- The parentheses following each table name contain the prefix of the column names for that table;
- The arrows point in the direction of the one-to-many relationships between tables;
- The number/formula below each table name represents the cardinality (number of rows) of the table. Some are factored by SF, the Scale Factor, to obtain the chosen database size. The cardinality for the LINEITEM table is approximate (see Clause 4.2.5).

2. The EXPLAIN tool gives you a glimpse on how the database is solving a query. Here is an example

```
tpch=# select count(*) from supplier, nation
tpch-# where s_nationkey = n_nationkey and n_name = 'BRAZIL';
 count
-------
  1980
(1 row)

tpch=# explain select count(*) from supplier, nation
tpch-#where s_nationkey = n_nationkey and n_name = 'BRAZIL';
                                                QUERY PLAN

-------------------------------------------------------------------------------------
----------------------------
 Aggregate  (cost=0.00..868.80 rows=1 width=8)
   ->  Gather Motion 2:1  (slice2; segments: 2)
         ->  Aggregate  (cost=0.00..868.80 rows=1 width=8)
               ->  Hash Join  (cost=0.00..868.80 rows=1000 width=1)
                     Hash Cond: supplier.s_nationkey = nation.n_nationkey
                     ->  Table Scan on supplier
                     ->  Hash  (cost=431.00..431.00 rows=1 width=4)
                     ->  Broadcast Motion 2:2  (slice1; segments: 2)
                           ->  Table Scan on nation
                                 Filter: n_name = 'BRAZIL'::bpchar
 Optimizer status: PQO version 2.56.3
(11 rows)
```

What the plan is telling us is that Greenplum is replicating the result of scanning table NATION and then doing a local HASH JOIN with table SUPPLIER. Each local count is then sent to a chose node and the final count(*) is calculated.

3. The \timing feature from PSQL. You can roughly approximate the time to run a query and to transport all its results in the following way. (Don't forget the last line of the script. It restores the ability to see the results of a query interactively.)

```
tpch=# \o /dev/null
tpch=# \timing
Timing is on.
tpch=#select count(*) from supplier, nation where s_nationkey = n_nationkey and
n_name = 'BRAZIL';
Time: 84.463 ms;
tpch=#\o
```

4. All the tables in the schema were create with column oriented storage. To create an exact same table with row oriented storage you can do the following. Please don't forget to run the ANALYZE at the end to update the tables statistics

```
tpch=# CREATE TABLE customer2 (LIKE customer)
tpch-# WITH (appendonly=true, orientation=row);
NOTICE:  Table doesn't have 'DISTRIBUTED BY' clause, defaulting to distribution
columns from LIKE table
CREATE TABLE
tpch=# INSERT INTO customer2 SELECT * FROM customer;
tpch=# ANALYZE;
```

   Note that now you have two tables, CUSTOMER and CUSTOMER2, with the exact same content differing only by their underlying storage layout.

Your assignment is then

- To come up with two queries, one that works best when an underlying table is column oriented and one the other way around. (So you'll really two pairs of queries, the variation over the row-oriented table and the one over the column-oriented one). Your query must use at least two tables.
- To check if the plan of the queries variants differ and to comment why you think the faster plan is better.