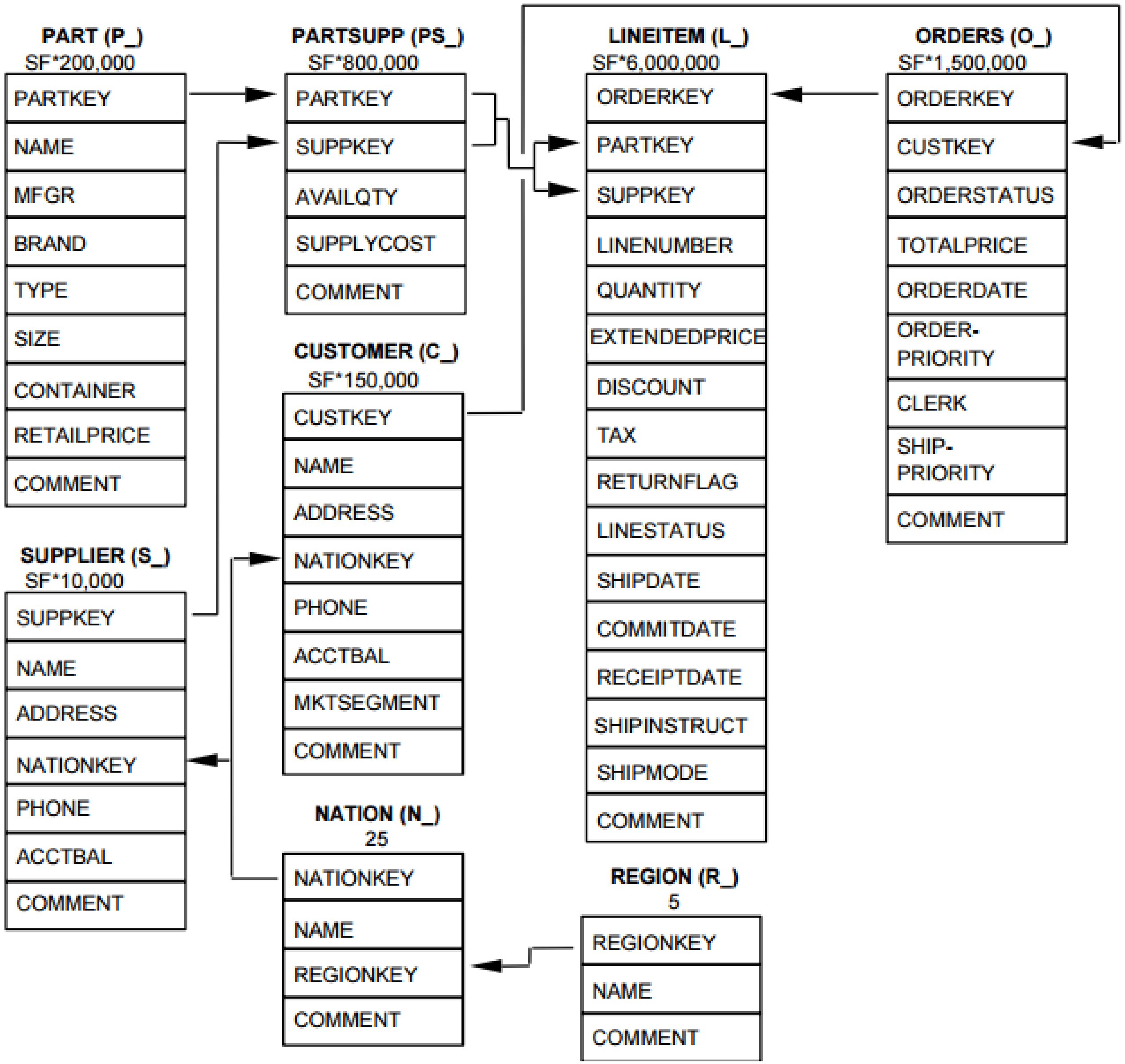
# CS425 – Project Assignment

- Tune the TPC-H queries in a database of your choice…
- What is TPC-H?
  - Industry standard benchmark for testing database performance for complex SQL
    - Decision support
    - Analytics
    - Ad-hoc
- TPC-H specification is here:
  - http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.17.1.pdf
    - We're not actually going to run the full benchmark according to specs
    - Just focus on making the queries run faster

# TPC-H Data Model



**PART (P_)**
SF*200,000
PARTKEY
NAME
MFGR
BRAND
TYPE
SIZE
CONTAINER
RETAILPRICE
COMMENT

**PARTSUPP (PS_)**
SF*800,000
PARTKEY
SUPPKEY
AVAILQTY
SUPPLYCOST
COMMENT

**LINEITEM (L_)**
SF*6,000,000
ORDERKEY
PARTKEY
SUPPKEY
LINENUMBER
QUANTITY
EXTENDEDPRICE
DISCOUNT
TAX
RETURNFLAG
LINESTATUS
SHIPDATE
COMMITDATE
RECEIPTDATE
SHIPINSTRUCT
SHIPMODE
COMMENT

**ORDERS (O_)**
SF*1,500,000
ORDERKEY
CUSTKEY
ORDERSTATUS
TOTALPRICE
ORDERDATE
ORDER-PRIORITY
CLERK
SHIP-PRIORITY
COMMENT

**CUSTOMER (C_)**
SF*150,000
CUSTKEY
NAME
ADDRESS
NATIONKEY
PHONE
ACCTBAL
MKTSEGMENT
COMMENT

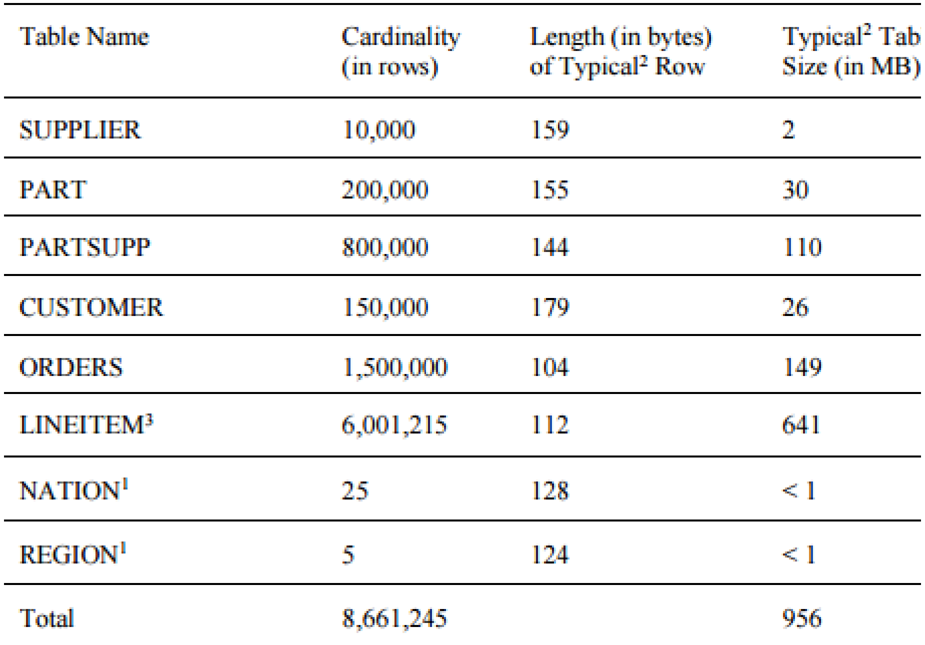
**SUPPLIER (S_)**
SF*10,000
SUPPKEY
NAME
ADDRESS
NATIONKEY
PHONE
ACCTBAL
COMMENT

**NATION (N_)**
25
NATIONKEY
NAME
REGIONKEY
COMMENT

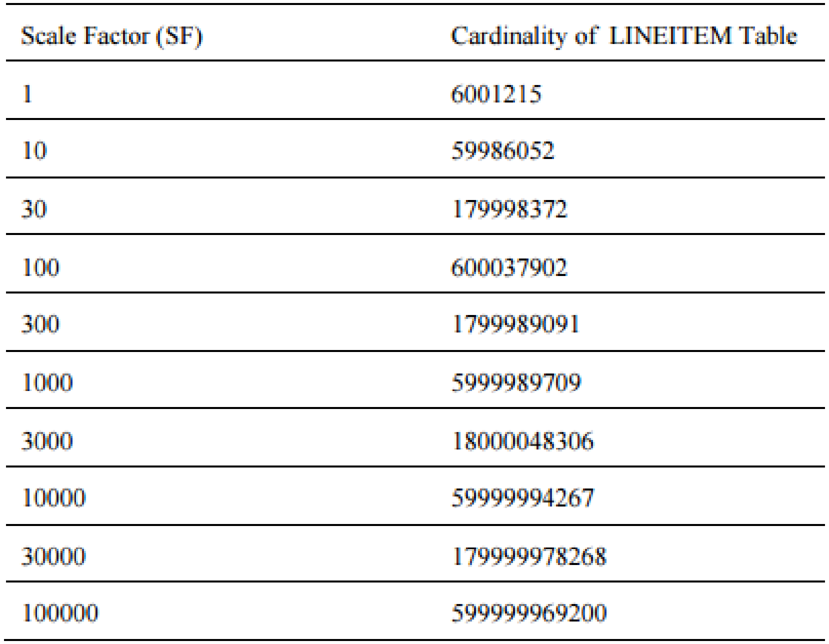
**REGION (R_)**
5
REGIONKEY
NAME
COMMENT

# TPC-H Scale Factor

- Scale Factor (SF) 1 is approximately 1GB of data
- Defined scale factors are:
  - 1, 10, 30, 100, 300, 1000, 3000, 10000, 30000, 100000



| Table Name | Cardinality (in rows) | Length (in bytes) of Typical[2] Row | Typical[2] Tab Size (in MB) |
|---|---|---|---|
| SUPPLIER | 10,000 | 159 | 2 |
| PART | 200,000 | 155 | 30 |
| PARTSUPP | 800,000 | 144 | 110 |
| CUSTOMER | 150,000 | 179 | 26 |
| ORDERS | 1,500,000 | 104 | 149 |
| LINEITEM[3] | 6,001,215 | 112 | 641 |
| NATION[1] | 25 | 128 | < 1 |
| REGION[1] | 5 | 124 | < 1 |
| Total | 8,661,245 | | 956 |



| Scale Factor (SF) | Cardinality of LINEITEM Table |
|---|---|
| 1 | 6001215 |
| 10 | 59986052 |
| 30 | 179998372 |
| 100 | 600037902 |
| 300 | 1799989091 |
| 1000 | 5999989709 |
| 3000 | 18000048306 |
| 10000 | 59999994267 |
| 30000 | 179999978268 |
| 100000 | 599999969200 |

# Defining the TPC-H Tables

- Section 1.4 in the specification gives all the info you need for figuring out what your CREATE TABLE statements should look like.
- Whether or not to implement primary keys and foreign keys is your choice
- ORDERS Table Layout
- Column Name Datatype Requirements Comment
- O_ORDERKEY Identifier SF*1,500,000 are sparsely populated
- O_CUSTKEY Identifier Foreign Key to C_CUSTKEY
- O_ORDERSTATUS fixed text, size 1
- O_TOTALPRICE Decimal
- O_ORDERDATE Date
- O_ORDERPRIORITY fixed text, size 15
- O_CLERK fixed text, size 15
- O_SHIPPRIORITY Integer
- O_COMMENT variable text, size 79
- Primary Key: O_ORDERKEY

# Loading Your Tables

- First you have to generate the data to be loaded
- This is done with the dbgen tool
  - http://www.tpc.org/tpc_documents_current_versions/download_programs/tools-download-request.asp?BM=TPC-H&mode=CURRENT-ONLY
- DBGEN is delivered as source code and has to be compiled
- On linux copy makefile.suite to Makefile, and edit
  - DATABASE=DB2
  - MACHINE =LINUX
- The run, make.
- This assumes you have gcc installed (yum install gcc.x86_64)
- ./dbgen –s 1 (would generate data for SF1)
- Output files end in .tbl
- They are pipe delimited text files

# Loading Your Tables 2

- Now that you have the data generated, you need to load the data into your tables
- This process is different for each database
- For Postgres, see the COPY command
- For MySQL, see the LOAD DATA command
- For DB2, see the LOAD command
- Etc…

# TPC-H Queries

- There are 22 TPC-H queries
- The SQL is given in section 2.4 of the specification
- The queries have substitution parameters
- Use the parameters specified in the query validation info for each query
- Change the SQL as little as possible
- The biggest challenge is that the syntax for the date stuff is different from database to database
- Some other calls like SUBSTRING (SUBSTR) might need to be modified slightly
- You also may want to qualify all the table names with a schema
- Before you do any changes more severe than this type of thing, check with me

# How to Benchmark

- Save your SQL in a text file
- All database have a way that you can ask them to execute whatever SQL is in a text file
  - Postgres – psql –f
  - The mysql command plus input redirection
  - Etc…
- But, we also need to know how long each query takes
  - Postgres - put “\timing on” in your script
    - Should also add “\pset pager off”
  - MySQL – gives you elapsed time by default
- Probably want to redirect your output as well, so that it gets saved
- One run is usually not good enough for us to have much confidence in the numbers
- A better approach is to do 3 runs and take the mode

# How to Tune

- You have many options available to make the SQL statements faster
  - Database specific extensions to the CREATE TABLE statement
  - Look at options on CREATE DATABASE / CREATE TABLESPACE statements
  - Buffer manager buffer sizes
  - Indexes
  - System parameters
    - postgresql.conf
    - my.cnf
    - Etc…

# How to Tune 2

- Some system parameters can usually be changed online via some command
- Others cannot…
- Regardless, to make the change permanent, you normally need to edit the configuration file
- Good benchmarking practice says to make as few changes as possible and then retest
  - Sometimes, changing 1 parameter requires changes to something else for it all to work effectively and that's OK
  - But, make your change set as small as possible
- If re-testing shows improvement, keep the change
- If not, revert it
- Keep track of all the changes you keep
  - Also track the performance #s before and after

# What to Turn In

- Start with SF1 and tune it the best you can
- Then go to SF10 and see if there's any additional tuning that can be done (there may or may not be)

- Here's what you will need to submit on Blackboard
  - HW specs – physical and virtual (cores, mem)
  - Database type and version
  - The exact SQL you ended up using for the queries
  - For the best results you could get at SF1…
    - All your DDL
    - Any system parms that were changed from defaults
    - Elapsed time for each query + the total for all queries
    - A log of how you got there (changed x to y and #s went from a to b)

# What to Turn In 2

- Also include results for SF10
  - If any DDL was different, state this
  - If any system parms were different, state this
  - Elapsed time for each query + the total for all queries
  - If there were changes from SF1, a log of how you got there starting from SF1 settings
- Lessons learned
  - Successes and failures ☺
- For classroom section students, please prepare a short slide deck of no more than 10 minutes to share your results and what you learned with the class
  - This part will not be graded, it’s really so that you can learn from each other