CSC 555: Mining Big Data

Project, Phase 2

In this part of the project, you will various queries using Hive, Pig and Hadoop streaming. The schema is available below, but don’t forget to apply the correct delimiter:

http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/SSBM\_schema\_hive.sql

The data is available at:

http://rasinsrv07.cstcis.cti.depaul.edu/CSC553/data/ (this is Scale4)

<http://rasinsrv07.cstcis.cti.depaul.edu/CSC553/data/Scale14/>

In your submission, please note what instance and what cluster you are using (you can reuse your existing cluster for most of the questions). Please be sure to submit all code (pig, python and Hive). You should also submit the command lines you use and a screenshot of a completed run (just the last page, do not worry about capturing the whole output). An answer submission with screenshot/results but without the code will not receive credit.

I highly recommend creating a small sample input (e.g., by running head lineorder.tbl > lineorder.tbl.sample and testing your code with it, you can use head -n 100 to get first 100 lines).

**Part 1: Data Transformation**

Use Scale4 data to perform data processing, unless otherwise specified.

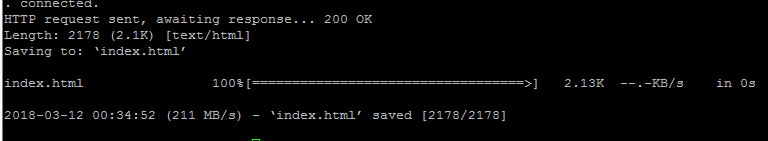
1. **Transform lineorder.tbl table into a comma-separated file: Use Hive, MapReduce with HadoopStreaming and Pig (i.e. 3 different solutions)**

**HIVE:**

cd $HIVE\_HOME

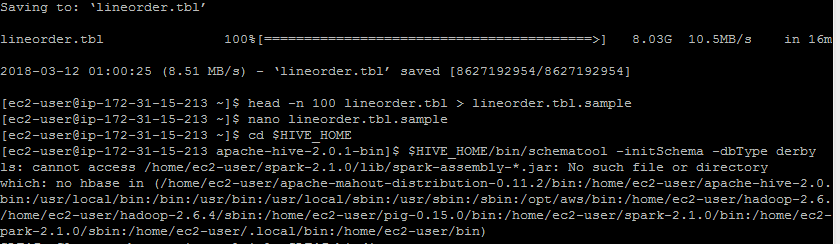
bin/hive

**Downloaded the hive schema**

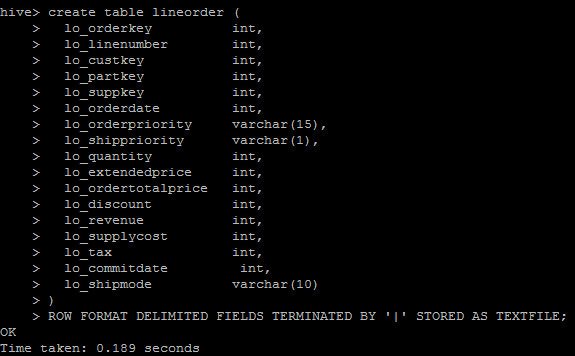


**Downloaded the file lineorder.tbl**

**Converted the large linorder.tbl file to smaller 100 row file called “lineorder.tbl.sample”**



**Created table lineorder expecting input of ‘|’ delimited file**



create table lineorder (

lo\_orderkey int,

lo\_linenumber int,

lo\_custkey int,

lo\_partkey int,

lo\_suppkey int,

lo\_orderdate int,

lo\_orderpriority varchar(15),

lo\_shippriority varchar(1),

lo\_quantity int,

lo\_extendedprice int,

lo\_ordertotalprice int,

lo\_discount int,

lo\_revenue int,

lo\_supplycost int,

lo\_tax int,

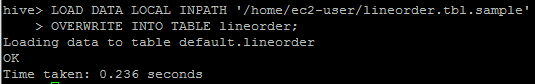
lo\_commitdate int,

lo\_shipmode varchar(10)

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE;

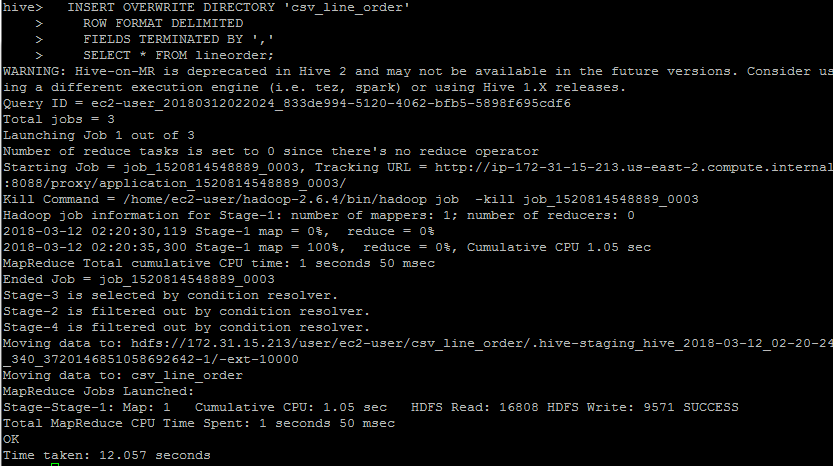
**Loaded smaller file lineorder.tbl.sample file of 100 records into the table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/lineorder.tbl.sample'**

**OVERWRITE INTO TABLE lineorder;**

**Using the smaller file lineorder.tbl.sample within the table linorder; exporting all contents of the table into a .csv file in HDFS user home directory labeled csv\_line\_order (/user/ec2-user/csv\_line\_order)**



INSERT OVERWRITE DIRECTORY 'csv\_line\_order'

ROW FORMAT DELIMITED

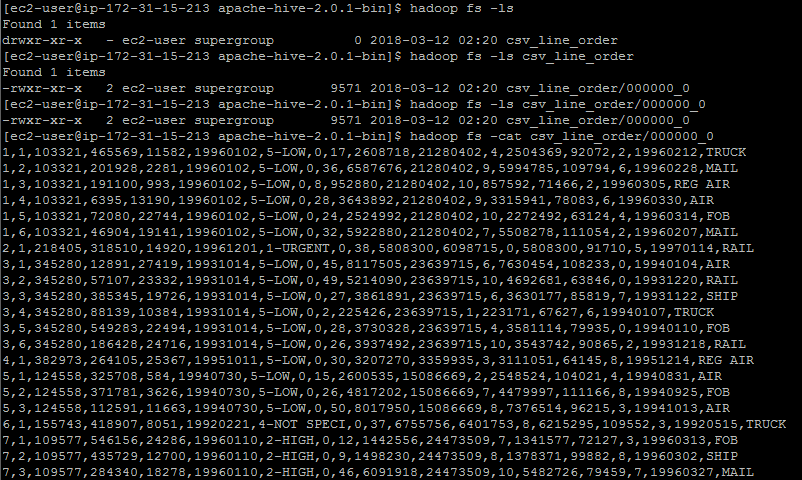
FIELDS TERMINATED BY ','

SELECT \* FROM lineorder;

hadoop fs –ls Command to see what files are located in HDFS

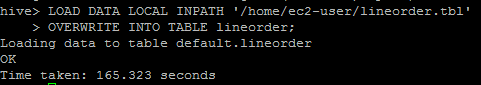
hadoop fs -ls csv\_line\_order Command to see what files are located in csv\_line\_order

hadoop fs -ls csv\_line\_order/000000\_0 Command to see what the file looks like



**ON A (4) NODE CLUSTER – LOAD THE LARGE FILE**

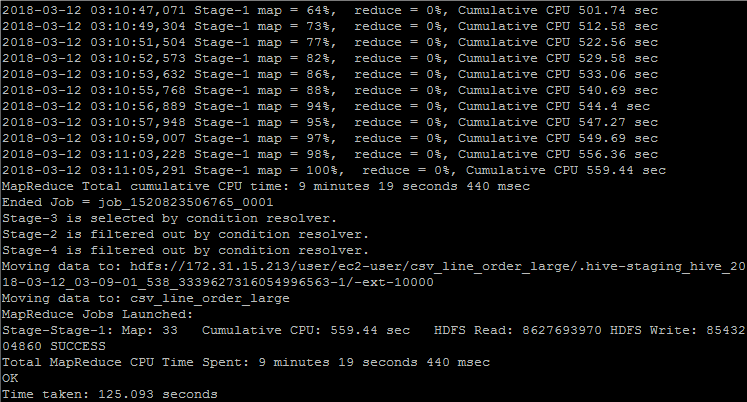
**Loaded entire file lineorder.tbl file of 100 records into the table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/lineorder.tbl'**

**OVERWRITE INTO TABLE lineorder;**

**Using the larger file lineorder.tbl within the table linorder; exporting all contents of the table into a .csv file in HDFS user home directory labeled csv\_line\_order\_large (/user/ec2-user/csv\_line\_order\_large)**



INSERT OVERWRITE DIRECTORY 'csv\_line\_order\_large'

ROW FORMAT DELIMITED

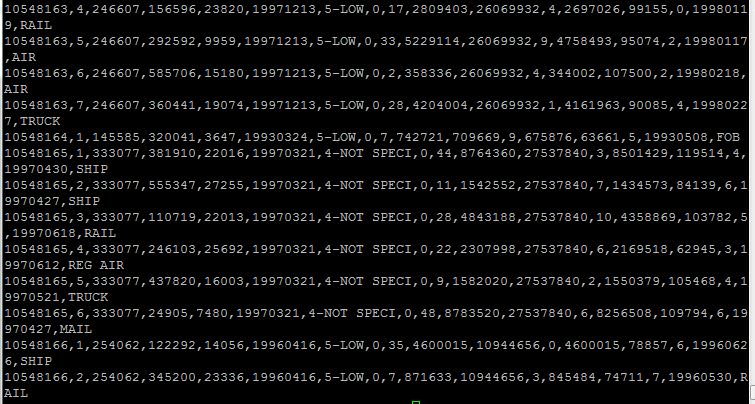
FIELDS TERMINATED BY ','

SELECT \* FROM lineorder;

hadoop fs –ls Command to see what files are located in HDFS

hadoop fs -ls csv\_line\_order\_large Command to see what files are located in csv\_line\_order

hadoop fs -cat csv\_line\_order\_large/000000\_0 Command to see what the file looks like



INSERT OVERWRITE DIRECTORY 'csv\_line\_order\_large\_singlenode'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT \* FROM lineorder;

**B. Extract three of the numeric columns (lo\_quantity, lo\_linenumber, lo\_revenue) for rows where lo\_discount is between 6 and 8 into a space-separated text file (for K-Means clustering later). Use Hive, MapReduce with Hadoop Streaming, and Pig (3 different solutions)**

**Part 2: Querying**

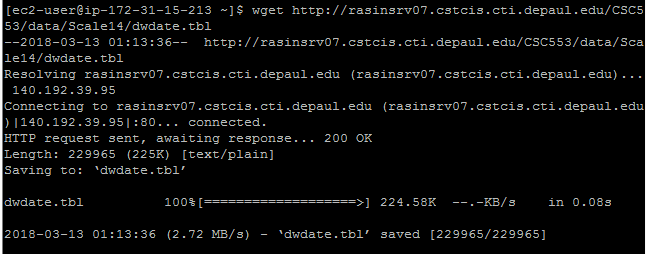
All queries from SSBM benchmark are available here:

http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/SSBM\_queries\_all.sql

Using Scale4 data perform the following data processing and don’t forget to time your results.

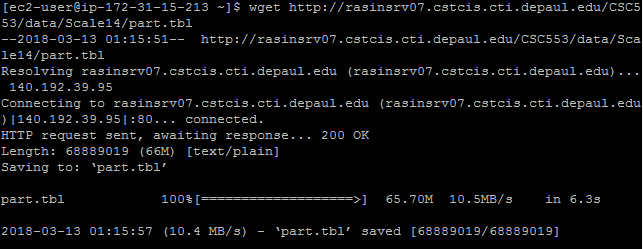
**HIVE**

**Download dwdate.tbl data**



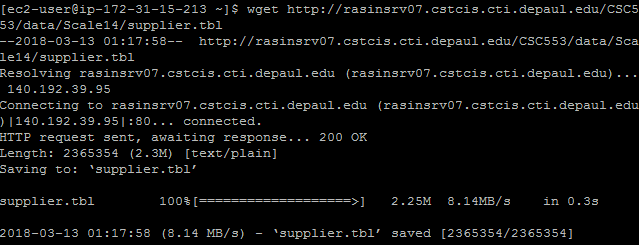
wget <http://rasinsrv07.cstcis.cti.depaul.edu/CSC553/data/Scale14/dwdate.tbl>

**Download part.tbl data**



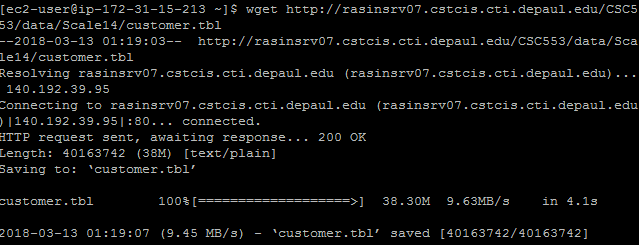
wget <http://rasinsrv07.cstcis.cti.depaul.edu/CSC553/data/Scale14/part.tbl>

**Download supplier.tbl data**



wget <http://rasinsrv07.cstcis.cti.depaul.edu/CSC553/data/Scale14/supplier.tbl>

**Download customer.tbl data**



wget http://rasinsrv07.cstcis.cti.depaul.edu/CSC553/data/Scale14/customer.tbl

**Create Tables part, supplier, customer, dwdate**

create table part (

p\_partkey int,

p\_name varchar(22),

p\_mfgr varchar(6),

p\_category varchar(7),

p\_brand1 varchar(9),

p\_color varchar(11),

p\_type varchar(25),

p\_size int,

p\_container varchar(10)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE;

create table supplier (

s\_suppkey int,

s\_name varchar(25),

s\_address varchar(25),

s\_city varchar(10),

s\_nation varchar(15),

s\_region varchar(12),

s\_phone varchar(15)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE;

create table customer (

c\_custkey int,

c\_name varchar(25),

c\_address varchar(25),

c\_city varchar(10),

c\_nation varchar(15),

c\_region varchar(12),

c\_phone varchar(15),

c\_mktsegment varchar(10)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE;

create table dwdate (

d\_datekey int,

d\_date varchar(19),

d\_dayofweek varchar(10),

d\_month varchar(10),

d\_year int,

d\_yearmonthnum int,

d\_yearmonth varchar(8),

d\_daynuminweek int,

d\_daynuminmonth int,

d\_daynuminyear int,

d\_monthnuminyear int,

d\_weeknuminyear int,

d\_sellingseason varchar(13),

d\_lastdayinweekfl varchar(1),

d\_lastdayinmonthfl varchar(1),

d\_holidayfl varchar(1),

d\_weekdayfl varchar(1)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE;

create table lineorder (

lo\_orderkey int,

lo\_linenumber int,

lo\_custkey int,

lo\_partkey int,

lo\_suppkey int,

lo\_orderdate int,

lo\_orderpriority varchar(15),

lo\_shippriority varchar(1),

lo\_quantity int,

lo\_extendedprice int,

lo\_ordertotalprice int,

lo\_discount int,

lo\_revenue int,

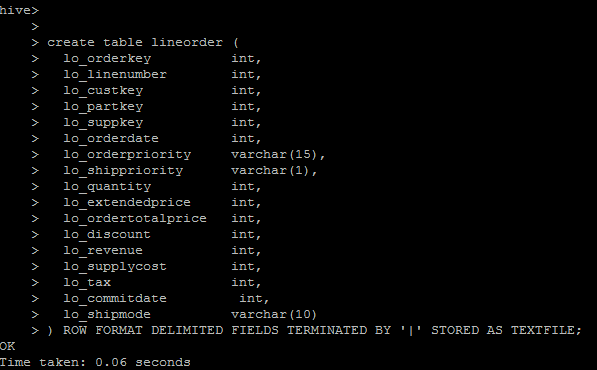
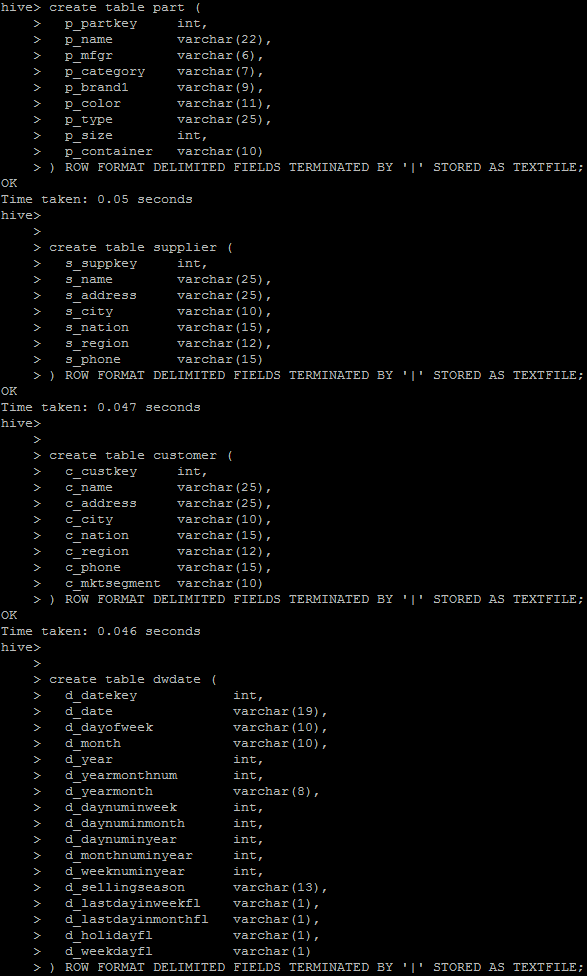
lo\_supplycost int,

lo\_tax int,

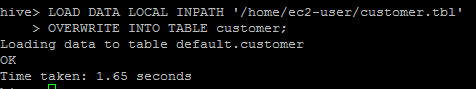
lo\_commitdate int,

lo\_shipmode varchar(10)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE;



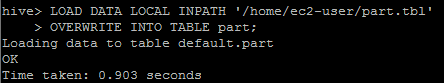
**Load customer.tbl data in customer table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/customer.tbl'**

**OVERWRITE INTO TABLE customer;**

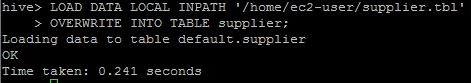
**Load part.tbl data in part table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/part.tbl'**

**OVERWRITE INTO TABLE part;**

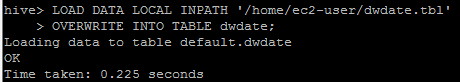
**Load supplier.tbl data in supplier table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/supplier.tbl'**

**OVERWRITE INTO TABLE supplier;**

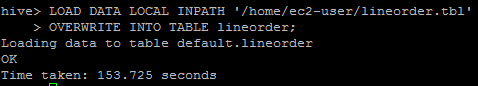
**Load dwdate.tbl data in dwdate table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/dwdate.tbl'**

**OVERWRITE INTO TABLE dwdate;**

**Load lineorder.tbl data in lineorder table**



**LOAD DATA LOCAL INPATH '/home/ec2-user/lineorder.tbl'**

**OVERWRITE INTO TABLE lineorder;**

A. Run SSBM queries 2.1, 3.3 and 4.3 using Hive only (if you have issues running the queries, try placing lineorder table first in the FROM clause of the query)

**Run Query 2.1**

--Q2.1 No simpifications

select sum(lo\_revenue), d\_year, p\_brand1

from lineorder, dwdate, part, supplier

where lo\_orderdate = d\_datekey

and lo\_partkey = p\_partkey

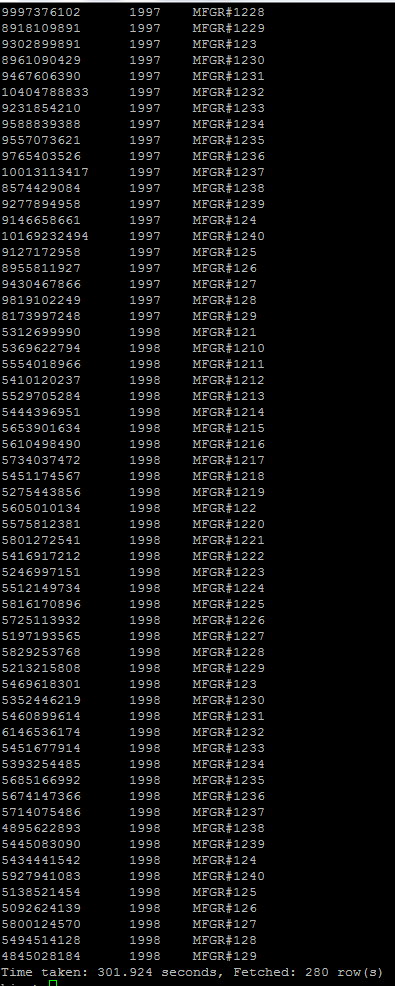
and lo\_suppkey = s\_suppkey

and p\_category = 'MFGR#12'

and s\_region = 'AMERICA'

group by d\_year, p\_brand1

order by d\_year, p\_brand1;



**Run Query 3.3**

select c\_city, s\_city, d\_year, sum(lo\_revenue) as revenue

from customer, lineorder, supplier, dwdate

where lo\_custkey = c\_custkey

and lo\_suppkey = s\_suppkey

and lo\_orderdate = d\_datekey

and c\_nation = 'UNITED KINGDOM'

and c\_city='UNITED KI1'

and s\_city='UNITED KI1'

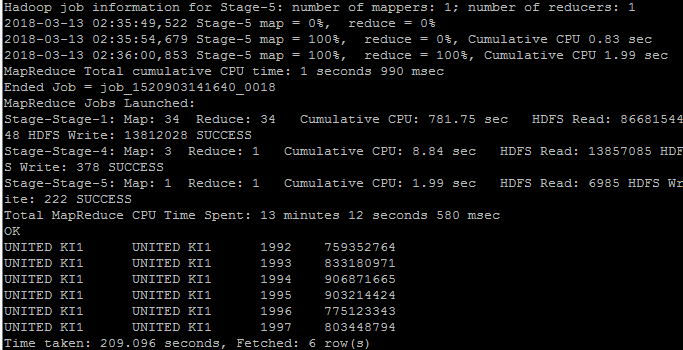
and s\_nation = 'UNITED KINGDOM'

and d\_year between 1992 and 1997

-- and d\_year >= 1992 and d\_year <= 1997

group by c\_city, s\_city, d\_year

order by d\_year asc, revenue asc;



**Run Query 4.3**

select d\_year, s\_city, p\_brand1, sum(lo\_revenue) as profit1

from dwdate, customer, supplier, part, lineorder

where lo\_custkey = c\_custkey

and lo\_suppkey = s\_suppkey

and lo\_partkey = p\_partkey

and lo\_orderdate = d\_datekey

and c\_region = 'AMERICA'

and s\_nation = 'UNITED STATES'

and d\_year = 1997

and p\_category = 'MFGR#14'

group by d\_year, s\_city, p\_brand1;

**B. Create a pre-join (i.e. a new data file) that corresponds to the following query below. You can think of it as a materialized view. What is the size of the new file? Use Hive and Pig (2 different solutions and be sure to report the file size for both).**

INSERT OVERWRITE DIRECTORY 'hive\_pre-join\_file'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT lo\_partkey, lo\_suppkey, s\_suppkey, d\_year, lo\_revenue

FROM lineorder, dwdate, lo\_supplier

WHERE lo\_orderdate = d\_datekey and lo\_suppkey = s\_suppkey;

**Part 3: Clustering**

Using the file you have created in 1-B, run KMeans clustering using 11 clusters.

A. Using Mahout synthetic clustering as you have in a previous assignment on sample data. This entrails running the same clustering command, but substituting your own input data instead of the sample.

**NOTE:** if you get a java.lang.OutOfMemoryError error, you will need to reconfigure Hadoop to supply the java virtual machine with more memory. You can do this by editing the mapred-site.xml (Mapper should not need much RAM):

*<property>*

*<name> mapreduce.reduce.java.opts</name>*

*<value>-Xmx1024m</value>*

*</property>*

The amount of memory can be tweaked (you can go higher, but keep in mind how much physical memory your machine has). If you **still** run out of memory in 3-A submit the screenshot of that change and you will get full credit for the question.

B. Using Hadoop streaming perform three iterations manually (initially with randomly chosen centers). This would require passing a text file with cluster centers using -file option, opening the centers.txt in the mapper with open(‘centers.txt’, ‘r’) and assigning a key to each point based on which center is the closest to each particular point. Your reducer would then compute the new centers, and at that point the iteration is done and the output of the reducer can be given to the next pass.

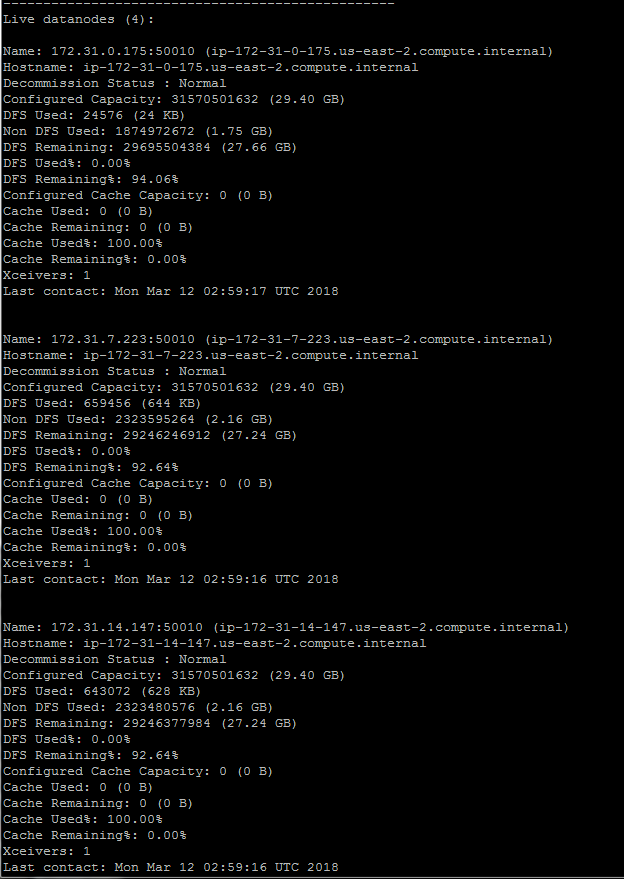
**NOTE**: Not attempting to answer this question will result in an additional grade penalty

**Part 4: Performance**

Compare the performance given following combinations.

**A. All three of your solutions to Part-1A with**

**a. Scale4: a single node cluster and a cluster of at least 4 nodes**



B. Both of your solutions for 2-B.

a. Scale4: a single node and a cluster of at least 4 nodes

C. Summarize the results and cluster performance/scaling in at least a paragraph.

**Extra Credit**

Research and describe the most affordable way to build a 10-Petabyte drive. The drive should be built to own, not to rent (Dropbox or similar services doesn’t count, even if it does say “unlimited” storage).

Submit a single document containing your written answers. Be sure that this document contains your name and “CSC 555 Project Phase 2” at the top.