CSC 555: Mining Big Data

Project, Phase 2 (due Sunday March 24th)

In this part of the project, you will execute queries using Hive, Pig and Hadoop streaming and develop a custom version of KMeans clustering. 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 (this is Scale1, the smallest denomination of this benchmark)
http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/

In your submission, please note what cluster you are using. 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 without 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 run **head -n 500 lineorder.tbl** to get a specific number of lines.

NOTE: the total number of points adds up to 70 because Phase I is worth 30 of the project.

Part 1: Data Transformation (15 pts)

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

In all solutions you must switch odd and even columns (i.e., switch the positions of columns 1 and 2, columns 3 and 4, etc.). You do not need to transform the columns in any way, just a new data file.

Using my multi-node cluster

Hive

```
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;

LOAD DATA LOCAL INPATH '/home/ec2-user/part.tbl' OVERWRITE INTO TABLE part;
```

Python code (colSwitcher.py):

```
#!/usr/bin/python
import sys

for line in sys.stdin:
   line = line.strip().split('\t')
   print '*'.join([line[1], line[0], line[3], line[2], line[5], line[4], line[7], line[6], line[8]])
```

Commands:

ADD FILE /home/ec2-user/colSwitcher.py;

INSERT OVERWRITE DIRECTORY 'partSwitched.tbl' SELECT TRANSFORM (p_partkey, p_name, p_mfgr, p_category, p_brand1, p_color, p_type, p_size, p_container) USING 'colSwitcher.py' AS (p_name, p_partkey, p_category, p_mfgr, p_color, p_brand1, p_size, p_type, p_container) FROM part;

Completed Run:

```
ec2-user@ip-172-31-14-37:~/apache-hive-2.0.1-bin
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1552319673784_0007, Tracking URL = http://ip-172-31-14-37.us-
east-2.compute.internal:8088/proxy/application_1552319673784_0007/
Kill Command = /home/ec2-user/hadoop-2.6.4/bin/hadoop job  -kill job_15523196737
84_0007
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2019-03-20 20:26:28,724 Stage-1 map = 0%,  reduce = 0%
2019-03-20 20:26:34,963 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.43 se
c
MapReduce Total cumulative CPU time: 4 seconds 430 msec
Ended Job = job_1552319673784_0007
Stage-3 is selected by condition resolver.
Stage-2 is filtered out by condition resolver.
Stage-4 is filtered out by condition resolver.
Moving data to: hdfs://172.31.14.37/user/ec2-user/partSwitched.tbl/.hive-staging
_hive_2019-03-20_20-26-22_174_6983635567456908387-1/-ext-10000
Moving data to: partSwitched.tbl
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 4.43 sec   HDFS Read: 17145329 HDFS Writ
e: 21739259 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 430 msec
OK
Time taken: 14.95 seconds
hive>
```

Output, first ten rows:

```
ec2-user@ip-172-31-14-37:~/apache-hive-2.0.1-bin
bm
[ec2-user@ip-172-31-14-37 apache-hive-2.0.1-bin]$ hadoop fs -ls /user/ec2-user
Found 6 items
drwxr-xr-x   - ec2-user supergroup          0 2019-03-12 03:02 /user/ec2-user/als
drwxr-xr-x   - ec2-user supergroup          0 2019-03-12 02:54 /user/ec2-user/dataset
drwxr-xr-x   - ec2-user supergroup          0 2019-03-12 02:54 /user/ec2-user/ml_dataset
drwxr-xr-x   - ec2-user supergroup          0 2019-03-12 02:18 /user/ec2-user/movielens
drwxr-xr-x   - ec2-user supergroup          0 2019-03-20 20:26 /user/ec2-user/partSwitched.tbl
drwxr-xr-x   - ec2-user supergroup          0 2019-03-11 14:28 /user/ec2-user/ssbm
[ec2-user@ip-172-31-14-37 apache-hive-2.0.1-bin]$ hadoop fs -ls /user/ec2-user/partSwitched.tbl
Found 1 items
-rwxr-xr-x   2 ec2-user supergroup   21739259 2019-03-20 20:26 /user/ec2-user/partSwitched.tbl/000000_0
[ec2-user@ip-172-31-14-37 apache-hive-2.0.1-bin]$ hadoop fs -head /user/ec2-user/partSwitched.tbl/000000_0
-head: Unknown command
[ec2-user@ip-172-31-14-37 apache-hive-2.0.1-bin]$ hadoop fs -cat /user/ec2-user/partSwitched.tbl/000000_0 | head
lace spring*1*MFGR#11*MFGR#1*goldenrod*MFGR#1121*7*PROMO BURNISHED COPPER*JUMBO PKG\N\N\N\N\N\N\N\N
rosy metallic*2*MFGR#43*MFGR#4*blush*MFGR#4318*1*LARGE BRUSHED BRASS*LG CASE\N\N\N\N\N\N\N\N
green antique*3*MFGR#32*MFGR#3*dark*MFGR#3210*21*STANDARD POLISHED BRASS*WRAP CASE\N\N\N\N\N\N\N\N
metallic smoke*4*MFGR#14*MFGR#1*chocolate*MFGR#1426*14*SMALL PLATED BRASS*MED DRUM\N\N\N\N\N\N\N\N
blush chiffon*5*MFGR#45*MFGR#4*forest*MFGR#4510*15*STANDARD POLISHED TIN*SM PKG\N\N\N\N\N\N\N\N
ivory azure*6*MFGR#23*MFGR#2*white*MFGR#2325*4*PROMO PLATED STEEL*MED BAG\N\N\N\N\N\N\N\N
blanched tan*7*MFGR#51*MFGR#5*blue*MFGR#513*45*SMALL PLATED COPPER*SM BAG\N\N\N\N\N\N\N\N
khaki cream*8*MFGR#13*MFGR#1*ivory*MFGR#1328*41*PROMO BURNISHED TIN*LG DRUM\N\N\N\N\N\N\N\N
rose moccasin*9*MFGR#41*MFGR#4*thistle*MFGR#4117*12*SMALL BURNISHED STEEL*WRAP CASE\N\N\N\N\N\N\N\N
moccasin royal*10*MFGR#21*MFGR#2*floral*MFGR#2128*44*LARGE BURNISHED STEEL*LG CAN\N\N\N\N\N\N\N\N
cat: Unable to write to output stream.
[ec2-user@ip-172-31-14-37 apache-hive-2.0.1-bin]$
```

Hadoop Streaming

There is no need for a custom mapper for this exercise, so I used the linux cat function as the mapper. The reducer code is:

colSwitcherReducer.py

```
#!/usr/bin/python
import sys

for line in sys.stdin:
   line = line.strip().split('|')
   print "%s*%s*%s*%s*%s*%s*%s*%s*%s" %
(line[1],line[0],line[3],line[2],line[5],line[4],line[7],line[6],line[8])
```

Command:

```
hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/ssbm/part.tbl -output
/data/output110 -mapper /bin/cat -reducer colSwitcherReducer.py -file colSwitcherReducer.py
```

```
ec2-user@ip-172-31-14-37:~
                Combine output records=0
                Reduce input groups=200000
                Reduce shuffle bytes=17739271
                Reduce input records=200000
                Reduce output records=200000
                Spilled Records=400000
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=278
                CPU time spent (ms)=4740
                Physical memory (bytes) snapshot=696291328
                Virtual memory (bytes) snapshot=6440493056
                Total committed heap usage (bytes)=503316480
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=17143355
        File Output Format Counters
                Bytes Written=17139259
19/03/20 21:20:09 INFO streaming.StreamJob: Output directory: /data/output110
[ec2-user@ip-172-31-14-37 ~]$
```

Output, first ten rows:

```
ec2-user@ip-172-31-14-37:~
                Virtual memory (bytes) snapshot=6440493056
                Total committed heap usage (bytes)=503316480
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=17143355
        File Output Format Counters
                Bytes Written=17139259
19/03/20 21:20:09 INFO streaming.StreamJob: Output directory: /data/output110
[ec2-user@ip-172-31-14-37 ~]$ hadoop fs -cat /data/output110/part-00000 | head
cyan floral*100000*MFGR#55*MFGR#5*maroon*MFGR#5535*17*LARGE BURNISHED STEEL*MED BOX
floral pink*100001*MFGR#54*MFGR#5*black*MFGR#5436*37*STANDARD BRUSHED TIN*JUMBO CASE
olive rose*100002*MFGR#12*MFGR#1*peach*MFGR#1226*11*STANDARD ANODIZED NICKEL*WRAP CAN
light violet*100003*MFGR#15*MFGR#1*puff*MFGR#155*41*MEDIUM PLATED BRASS*SM BOX
gainsboro slate*100004*MFGR#25*MFGR#2*hot*MFGR#2511*29*SMALL POLISHED TIN*SM CASE
drab misty*100005*MFGR#23*MFGR#2*grey*MFGR#235*7*SMALL POLISHED STEEL*MED BAG
honeydew navy*100006*MFGR#12*MFGR#1*royal*MFGR#1237*23*STANDARD BURNISHED COPPER*WRAP CASE
moccasin wheat*100007*MFGR#22*MFGR#2*firebrick*MFGR#2211*4*PROMO BURNISHED COPPER*MED PKG
powder burlywood*100008*MFGR#35*MFGR#3*spring*MFGR#3535*19*ECONOMY BRUSHED BRASS*SM PKG
antique aquamarine*100009*MFGR#52*MFGR#5*indian*MFGR#529*41*SMALL BURNISHED STEEL*WRAP BOX
cat: Unable to write to output stream.
[ec2-user@ip-172-31-14-37 ~]$
```

Pig

Load the Data:

PartData = LOAD '/user/ec2-user/ssbm/part.tbl' USING PigStorage('|') AS (p_partkey:int, p_name:chararray, p_mfgr:chararray, p_category:chararray, p_brand1:chararray, p_color:chararray, p_type:chararray, p_size:int, p_container:chararray);

Verify Data Loaded:

PartG = GROUP PartData ALL;
Count = FOREACH PartG GENERATE COUNT(PartData);
DUMP Count;

```
ec2-user@ip-172-31-14-37:~/pig-0.15.0
2019-03-21 00:58:14,122 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10
020. Already tried 4 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MI
LLISECONDS)
2019-03-21 00:58:15,124 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10
020. Already tried 5 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MI
LLISECONDS)
2019-03-21 00:58:16,125 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10
020. Already tried 6 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MI
LLISECONDS)
2019-03-21 00:58:17,127 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10
020. Already tried 7 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MI
LLISECONDS)
2019-03-21 00:58:18,128 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10
020. Already tried 8 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MI
LLISECONDS)
2019-03-21 00:58:19,129 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:10
020. Already tried 9 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MI
LLISECONDS)
2019-03-21 00:58:19,232 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLaunche
r - Unable to retrieve job to compute warning aggregation.
2019-03-21 00:58:19,232 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLaunche
r - Success!
2019-03-21 00:58:19,235 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set...
 will not generate code.
2019-03-21 00:58:19,249 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to
process : 1
2019-03-21 00:58:19,249 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input p
aths to process : 1
(200000)
grunt>
```

Switch Columns:

PartSwitchedPig = FOREACH PartData GENERATE p_name, p_partkey, p_category, p_mfgr, p_color, p_brand1, p_size, p_type, p_container;

Write to file:

STORE PartSwitchedPig INTO 'partOutPig' USING PigStorage('*');

```
ec2-user@ip-172-31-14-37:~/pig-0.15.0
ISECONDS)
2019-03-21 01:16:17,261 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 2 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:18,262 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 3 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:19,263 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 4 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:20,265 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 5 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:21,266 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 6 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:22,268 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 7 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:23,269 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 8 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:24,270 [main] INFO  org.apache.hadoop.ipc.Client - Retrying connect to server: 0.0.0.0/0.0.0.0:100
20. Already tried 9 time(s); retry policy is RetryUpToMaximumCountWithFixedSleep(maxRetries=10, sleepTime=1000 MILL
ISECONDS)
2019-03-21 01:16:24,375 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher
 - Unable to retrieve job to compute warning aggregation.
2019-03-21 01:16:24,375 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher
 - Success!
grunt>
```

Output, first ten rows:

```
ec2-user@ip-172-31-14-37:~
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$
[ec2-user@ip-172-31-14-37 ~]$ hadoop fs -ls partOutPig/part-m-00000
-rw-r--r--   2 ec2-user supergroup   16939259 2019-03-21 01:13 partOutPig/part-m-00000
[ec2-user@ip-172-31-14-37 ~]$ hadoop fs -cat partOutPig/part-m-00000 | head
lace spring*1*MFGR#11*MFGR#1*goldenrod*MFGR#1121*7*PROMO BURNISHED COPPER*JUMBO PKG
rosy metallic*2*MFGR#43*MFGR#4*blush*MFGR#4318*1*LARGE BRUSHED BRASS*LG CASE
green antique*3*MFGR#32*MFGR#3*dark*MFGR#3210*21*STANDARD POLISHED BRASS*WRAP CASE
metallic smoke*4*MFGR#14*MFGR#1*chocolate*MFGR#1426*14*SMALL PLATED BRASS*MED DRUM
blush chiffon*5*MFGR#45*MFGR#4*forest*MFGR#4510*15*STANDARD POLISHED TIN*SM PKG
ivory azure*6*MFGR#23*MFGR#2*white*MFGR#2325*4*PROMO PLATED STEEL*MED BAG
blanched tan*7*MFGR#51*MFGR#5*blue*MFGR#513*45*SMALL PLATED COPPER*SM BAG
khaki cream*8*MFGR#13*MFGR#1*ivory*MFGR#1328*41*PROMO BURNISHED TIN*LG DRUM
rose moccasin*9*MFGR#41*MFGR#4*thistle*MFGR#4117*12*SMALL BURNISHED STEEL*WRAP CASE
moccasin royal*10*MFGR#21*MFGR#2*floral*MFGR#2128*44*LARGE BURNISHED STEEL*LG CAN
cat: Unable to write to output stream.
[ec2-user@ip-172-31-14-37 ~]$
```

Part 2: Querying (25 pts)

Implement the following query:

```
select lo_quantity, c_nation, sum(lo_revenue)
from customer, lineorder
where lo_custkey = c_custkey
  and c_region = 'AMERICA'
  and lo_discount BETWEEN 3 and 5
group by lo_quantity, c_nation;
```

using Hive, MapReduce with HadoopStreaming and Pig (i.e. 3 different solutions). I Hive, this merely requires pasting the query into the Hive prompt and timing it. In Hadoop streaming, this will require a total of 2 passes (one for join and another one for GROUP BY).

Using my multi-node cluster

Hive:

Create and load tables:

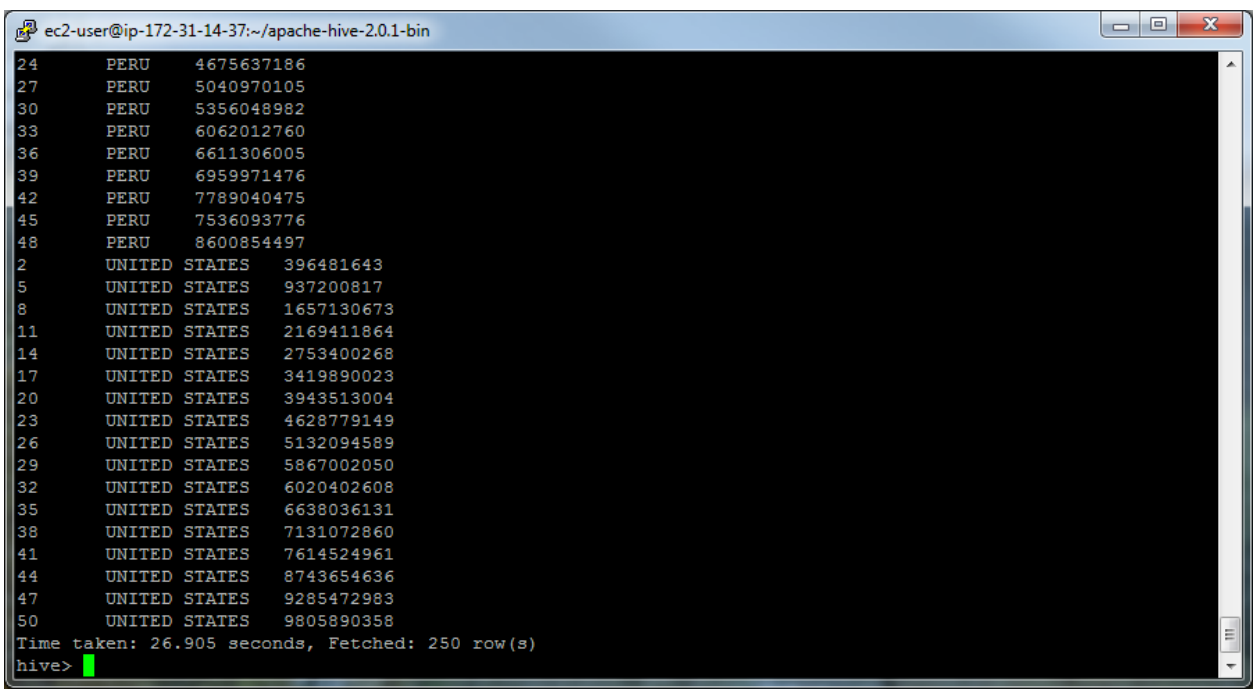
```
CREATE TABLE lineorder (
  lo_orderkey         INT,
  lo_linenumber       INT,
  lo_custkey          INT,
  lo_partkey          INT,
  lo_suppkey          INT,
  lo_orderdate        INT,
  lo_orderpriority    VARCHAR(15),
  lo_shippriority     VARCHAR(1),
  lo_quantity         INT,
  lo_extendedprice    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 DATA LOCAL INPATH '/home/ec2-user/lineorder.tbl' OVERWRITE INTO TABLE lineorder;
```

```
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;

LOAD DATA LOCAL INPATH '/home/ec2-user/customer.tbl' OVERWRITE INTO TABLE customer;
```

Execute sql statement:

```
select lo_quantity, c_nation, sum(lo_revenue)
from customer, lineorder
where lo_custkey = c_custkey
  and c_region = 'AMERICA'
  and lo_discount BETWEEN 3 and 5
group by lo_quantity, c_nation;
```

End of output with time taken:



```
ec2-user@ip-172-31-14-37:~/apache-hive-2.0.1-bin
24      PERU    4675637186
27      PERU    5040970105
30      PERU    5356048982
33      PERU    6062012760
36      PERU    6611306005
39      PERU    6959971476
42      PERU    7789040475
45      PERU    7536093776
48      PERU    8600854497
2       UNITED STATES   396481643
5       UNITED STATES   937200817
8       UNITED STATES   1657130673
11      UNITED STATES   2169411864
14      UNITED STATES   2753400268
17      UNITED STATES   3419890023
20      UNITED STATES   3943513004
23      UNITED STATES   4628779149
26      UNITED STATES   5132094589
29      UNITED STATES   5867002050
32      UNITED STATES   6020402608
35      UNITED STATES   6638036131
38      UNITED STATES   7131072860
41      UNITED STATES   7614524961
44      UNITED STATES   8743654636
47      UNITED STATES   9285472983
50      UNITED STATES   9805890358
Time taken: 26.905 seconds, Fetched: 250 row(s)
hive>
```

Michael Janke CSC 555 Project Phase 2

Hadoop Streaming:

Join

lineCustMapJoin.py

```
#!/usr/bin/python
import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
   line = line.strip().split('|')
   if line[1].startswith('Customer#'):
      if line[5] == 'AMERICA': # Return on matching records
         print line[0], '\t', line[4], '\t', 'customer'
   # lineorder
   else:
      if 3 <= int(line[11]) <= 5: # Return on matching records
         print line[2], '\t', line[8], '\t', line[12], '\t', 'lineorder'
```

lineCustReduceJoin.py

```
#!/usr/bin/python

import sys

currentKey = None
quantity = []
revenue = []
nation = ''

# input comes from STDIN
for line in sys.stdin:

    split = line.strip().split('\t')
    key = split[0] # key is customer id
    value = '\t'.join(split[1:])

    if currentKey == key: # Same key
        if value.endswith('lineorder'):
            quantity.extend([split[1]])
            revenue.extend([split[2]])
        if value.endswith('customer'):
            nation = split[1]
    else:
        # Do not print anything until all records
        # for a key have been seen, this is signaled
```

```
            # by currentKey != key
            # Check for values and then iterate results
            lenQuantity = len(quantity)
            lenNation = len(nation)
            if (lenQuantity*lenNation > 0):
                i = 0
                while i < lenQuantity:
                    print quantity[i], '\t', nation, '\t', revenue[i]
                    i += 1

            # reset values
            quantity = []
            revenue = []
            nation = ''

            if value.endswith('lineorder'):
                quantity.extend([split[1]])
                revenue.extend([split[2]])
            if value.endswith('customer'):
                nation = split[1]

        # set the current key at the end of each iteration
        currentKey = key
```

Commands:

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/phase2 -output /data/phase2_1 -mapper lineCustMapJoin.py -reducer lineCustReduceJoin.py -file lineCustMapJoin.py -file lineCustReduceJoin.py

```
ec2-user@ip-172-31-14-37:~
                Reduce input records=1643225
                Reduce output records=325799
                Spilled Records=3286450
                Shuffled Maps =6
                Failed Shuffles=0
                Merged Map outputs=6
                GC time elapsed (ms)=983
                CPU time spent (ms)=27170
                Physical memory (bytes) snapshot=1781522432
                Virtual memory (bytes) snapshot=15023194112
                Total committed heap usage (bytes)=1328545792
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=597166431
        File Output Format Counters
                Bytes Written=7938763
19/03/22 15:07:38 INFO streaming.StreamJob: Output directory: /data/phase2_1
[ec2-user@ip-172-31-14-37 ~]$
```

hadoop fs -ls /data/phase2_1

```
ec2-user@ip-172-31-14-37:~
                GC time elapsed (ms)=983
                CPU time spent (ms)=27170
                Physical memory (bytes) snapshot=1781522432
                Virtual memory (bytes) snapshot=15023194112
                Total committed heap usage (bytes)=1328545792
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=597166431
        File Output Format Counters
                Bytes Written=7938763
19/03/22 15:07:38 INFO streaming.StreamJob: Output directory: /data/phase2_1
[ec2-user@ip-172-31-14-37 ~]$ hadoop fs -ls /data/phase2_1
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2019-03-22 15:07 /data/phase2_1/_S
UCCESS
-rw-r--r--   2 ec2-user supergroup    7938763 2019-03-22 15:07 /data/phase2_1/pa
rt-00000
[ec2-user@ip-172-31-14-37 ~]$
```

hadoop fs -cat /data/phase2_1/part-00000 | head

```
ec2-user@ip-172-31-14-37:~
        File Input Format Counters
                Bytes Read=597166431
        File Output Format Counters
                Bytes Written=7938763
19/03/22 15:07:38 INFO streaming.StreamJob: Output directory: /data/phase2_1
[ec2-user@ip-172-31-14-37 ~]$ hadoop fs -ls /data/phase2_1
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2019-03-22 15:07 /data/phase2_1/_S
UCCESS
-rw-r--r--   2 ec2-user supergroup    7938763 2019-03-22 15:07 /data/phase2_1/pa
rt-00000
[ec2-user@ip-172-31-14-37 ~]$ hadoop fs -cat /data/phase2_1/part-00000 | head
6       PERU    1031010
8       PERU    1185433
20      PERU    2491299
1       PERU    171431
43      PERU    5189886
30      PERU    3783756
20      PERU    1899318
15      PERU    2307048
7       PERU    778794
23      PERU    3816837
cat: Unable to write to output stream.
[ec2-user@ip-172-31-14-37 ~]$
```

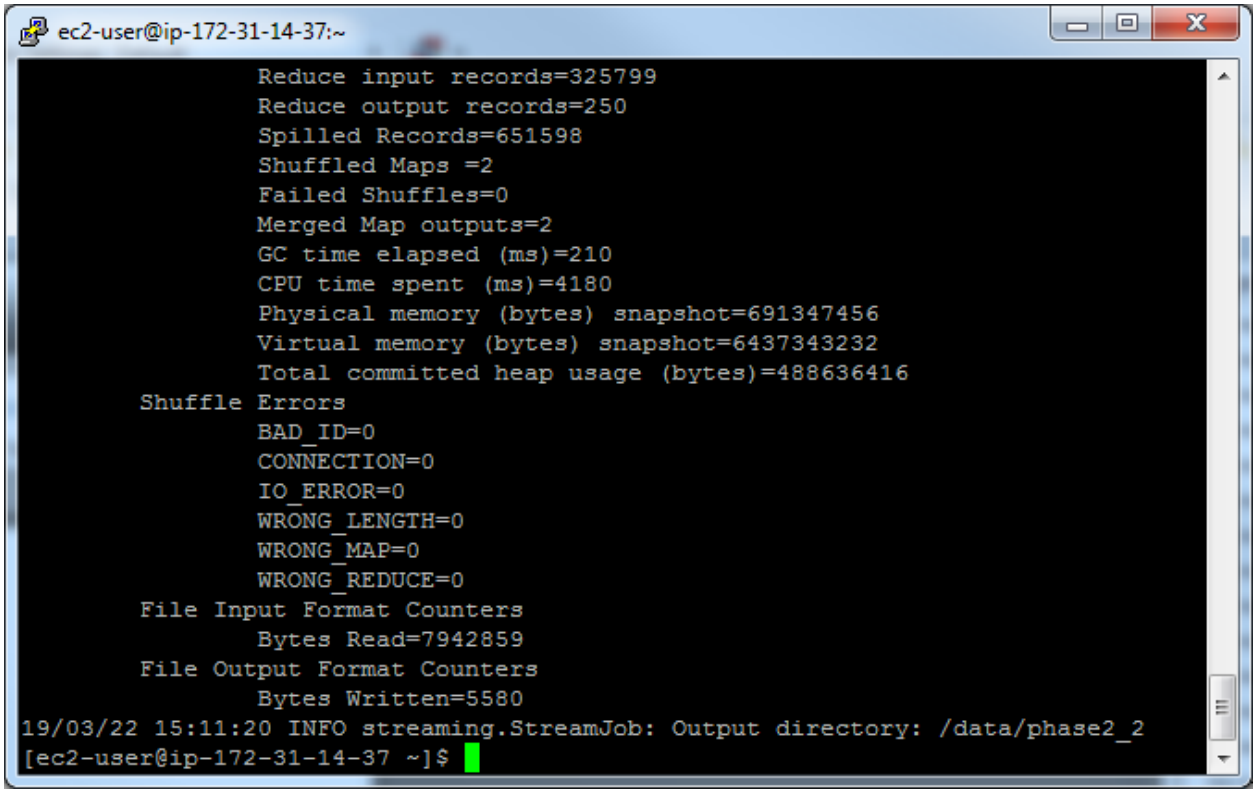
Group:

lineCustReduceGroup.py

```
#!/usr/bin/python
import sys

curr_id = None
curr_tot = 0
id = None

# The input comes from standard input (line by line)
for line in sys.stdin:
    # parse the line and split it by '\t'
    line = line.strip().split('\t')

    # grab the key
    # values include some whitespace, removing here
    id = line[0].strip() + '\t' + line[1].strip()

    # grab the value (int)
    val = int(line[2])
```

```
        if curr_id == id:
            curr_tot += val

        else:
            if curr_id: # output the sum, single key completed
                print '%s\t%d' % (curr_id, curr_tot)
            curr_tot = val

        # set curr_id to id at end of each iteration
        curr_id = id

    # output the last key
    if curr_id == id:
        print '%s\t%d' % (curr_id, curr_tot)
```

hadoop jar hadoop-streaming-2.6.4.jar -D stream.num.map.output.key.fields=2 -input /data/phase2_01/part-00000 -output /data/phase2_06 -mapper /bin/cat -reducer lineCustReduceGroup.py -file lineCustReduceGroup.py



hadoop fs -cat /data/phase2_2/part-00000

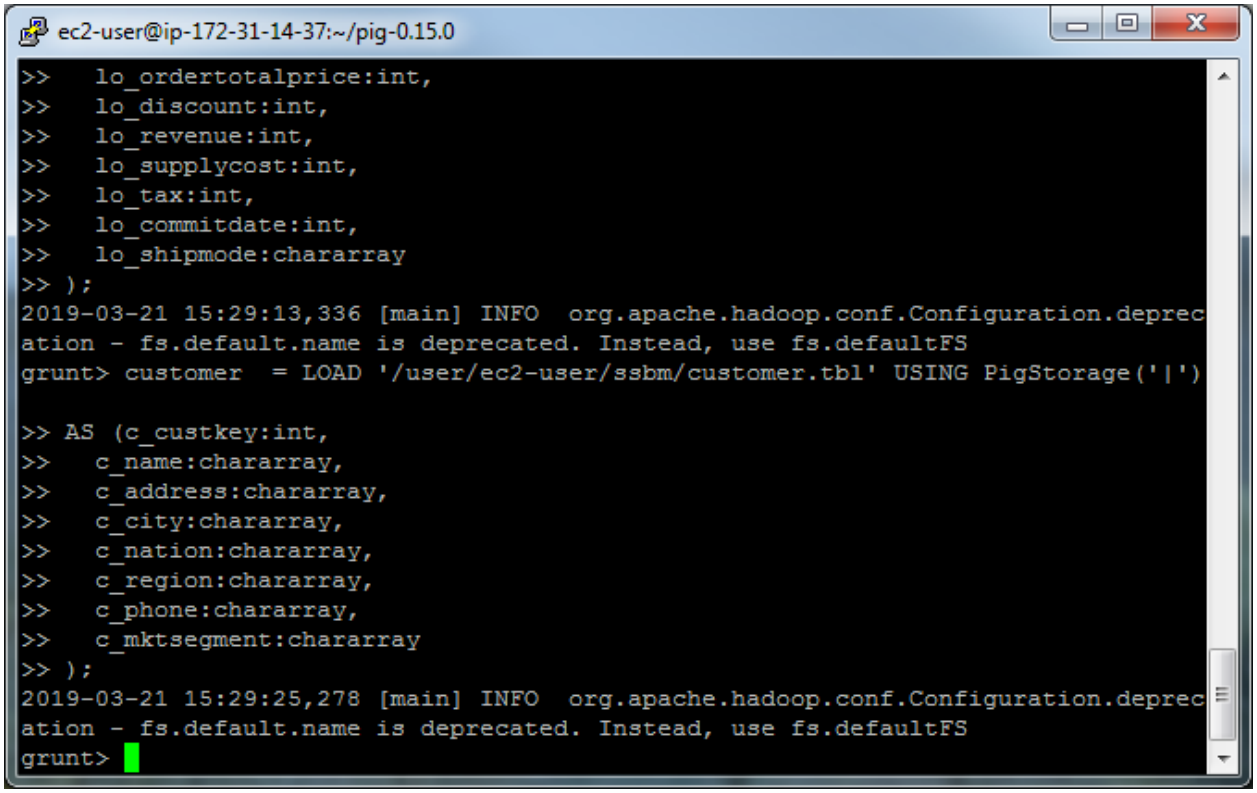
```
ec2-user@ip-172-31-14-37:~
50      CANADA  9267364820
50      PERU    8635363700
50      UNITED STATES   9805890358
6       ARGENTINA       1145246666
6       BRAZIL  1051924508
6       CANADA  1172533337
6       PERU    1114116730
6       UNITED STATES   1142463797
7       ARGENTINA       1294557868
7       BRAZIL  1220257059
7       CANADA  1351039262
7       PERU    1245835983
7       UNITED STATES   1388084746
8       ARGENTINA       1550523311
8       BRAZIL  1420473852
8       CANADA  1543712634
8       PERU    1357428094
8       UNITED STATES   1657130673
9       ARGENTINA       1798422567
9       BRAZIL  1615636240
9       CANADA  1674498414
9       PERU    1596785927
9       UNITED STATES   1748935712
[ec2-user@ip-172-31-14-37 ~]$
```

Pig:

Load Tables:

```
lineorder = LOAD '/user/ec2-user/ssbm/lineorder.tbl' USING PigStorage('|')
AS (lo_orderkey:int,
  lo_linenumber:int,
  lo_custkey:int,
  lo_partkey:int,
  lo_suppkey:int,
  lo_orderdate:int,
  lo_orderpriority:chararray,
  lo_shippriority:chararray,
  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:chararray
);
```

```
customer  = LOAD '/user/ec2-user/ssbm/customer.tbl' USING PigStorage('|')
AS (c_custkey:int,
  c_name:chararray,
  c_address:chararray,
  c_city:chararray,
  c_nation:chararray,
  c_region:chararray,
  c_phone:chararray,
  c_mktsegment:chararray
);
```



Execution steps:

```
FilteredLineorder = FILTER lineorder BY lo_discount >= 3 AND lo_discount <= 5;
FilteredCustomer = FILTER customer BY c_region == 'AMERICA';
JoinedData = JOIN FilteredLineorder BY (lo_custkey), FilteredCustomer BY (c_custkey);
GroupedData = GROUP JoinedData BY (lo_quantity, c_nation);
Result = FOREACH GroupedData GENERATE group, SUM(JoinedData.lo_revenue) as rev;
DUMP Result;
```

I had difficulty with displaying the non-summed columns so I left them out of the command. What's displayed still includes the grouped columns.

```
ec2-user@ip-172-31-14-37:~/pig-0.15.0
((45,CANADA),8587910391)
((45,ARGENTINA),8975171879)
((45,UNITED STATES),8599367420)
((46,PERU),8616030892)
((46,BRAZIL),8537179198)
((46,CANADA),8714853605)
((46,ARGENTINA),8888131785)
((46,UNITED STATES),8583003587)
((47,PERU),8477408521)
((47,BRAZIL),8740804315)
((47,CANADA),8825586261)
((47,ARGENTINA),9172719968)
((47,UNITED STATES),9285472983)
((48,PERU),8600854497)
((48,BRAZIL),9041047056)
((48,CANADA),9404797264)
((48,ARGENTINA),8862047735)
((48,UNITED STATES),9312021356)
((49,PERU),8336553718)
((49,BRAZIL),8869239908)
((49,CANADA),9717656478)
((49,ARGENTINA),9698422577)
((49,UNITED STATES),9257609485)
((50,PERU),8635363700)
((50,BRAZIL),8988626423)
((50,CANADA),9267364820)
((50,ARGENTINA),9370116794)
((50,UNITED STATES),9805890358)
grunt>
```

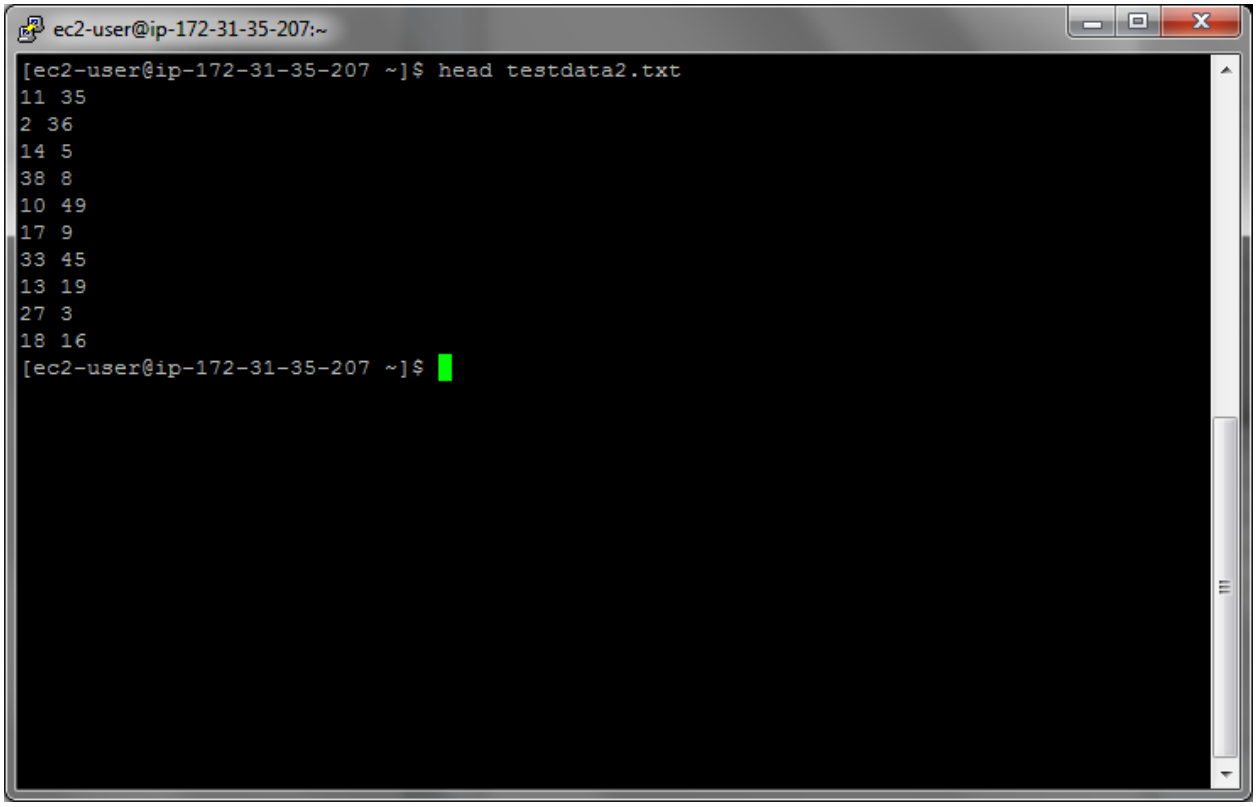
Part 3: Clustering (30 pts)

Create a new numeric file with 25,000 rows and 3 columns, separated by space – you can generate numeric data as you prefer, but submit whatever code that you have used.

A. (5 pts) Using Mahout synthetic clustering as you have in a previous assignment on sample data. This entails running the **same** clustering command, but substituting your own input data instead of the sample.

   Note, I used the single-node Hadoop instance for this exercise.

   First, I used an online random sequence generator to generate 100 x and y variables. Here's a screen shot of the first 10 records. The full list is at the end of this document.



Commands:

hadoop fs –put testdata2.txt testdata/

time mahout org.apache.mahout.clustering.syntheticcontrol.kmeans.Job

```
ec2-user@ip-172-31-35-207:~
        1.0 : [distance=6.1627584600351595]: [1.0,37.0]
{"identifier":"VL-10","r":[5.062,8.326],"c":[28.294,38.824],"n":17}
        Weight : [props - optional]:  Point:
        1.0 : [distance=7.764928695555327]: [33.0,45.0]
        1.0 : [distance=1.2126781251816674]: [28.0,40.0]
        1.0 : [distance=12.649110640673518]: [37.0,48.0]
        1.0 : [distance=9.812956621072928]: [24.0,30.0]
        1.0 : [distance=5.335783750799285]: [27.0,44.0]
        1.0 : [distance=6.8599434057002835]: [29.0,32.0]
        1.0 : [distance=5.319221526413292]: [25.0,43.0]
        1.0 : [distance=14.475130803025783]: [24.0,25.0]
        1.0 : [distance=12.366938848016812]: [23.0,50.0]
        1.0 : [distance=9.54555640383647]: [22.0,46.0]
        1.0 : [distance=9.970544855015797]: [34.0,47.0]
        1.0 : [distance=6.743188284134864]: [31.0,45.0]
        1.0 : [distance=12.888663509411028]: [27.0,26.0]
        1.0 : [distance=8.8284300116492]: [28.0,30.0]
        1.0 : [distance=9.908404038402733]: [27.0,29.0]
        1.0 : [distance=7.92983940197884]: [22.0,34.0]
        1.0 : [distance=13.730601289262806]: [40.0,46.0]
19/03/24 16:30:09 INFO ClusterDumper: Wrote 6 clusters
19/03/24 16:30:09 INFO MahoutDriver: Program took 217980 ms (Minutes: 3.633)

real    3m45.822s
user    0m13.230s
sys     0m3.392s
[ec2-user@ip-172-31-35-207 ~]$
```

mahout clusterdump --input output/clusters-7-final --pointsDir output/clusteredPoints --output clusteranalyze.txt

```
ec2-user@ip-172-31-35-207:~
Found 12 items
-rw-r--r--   1 ec2-user supergroup        194 2019-03-24 16:29 output/_policy
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:30 output/clusteredPoints
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:26 output/clusters-0
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:27 output/clusters-1
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:27 output/clusters-2
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:28 output/clusters-3
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:28 output/clusters-4
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:28 output/clusters-5
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:29 output/clusters-6
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:29 output/clusters-7-final
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:26 output/data
drwxr-xr-x   - ec2-user supergroup          0 2019-03-24 16:26 output/random-seeds
[ec2-user@ip-172-31-35-207 ~]$ mahout clusterdump --input output/clusters-7-final --pointsDir
output/clusteredPoints --output clusteranalyze.txt
Running on hadoop, using /home/ec2-user/hadoop-2.6.4/bin/hadoop and HADOOP_CONF_DIR=
MAHOUT-JOB: /home/ec2-user/apache-mahout-distribution-0.11.2/mahout-examples-0.11.2-job.jar
19/03/24 16:35:37 INFO AbstractJob: Command line arguments: {--dictionaryType=[text], --distan
ceMeasure=[org.apache.mahout.common.distance.SquaredEuclideanDistanceMeasure], --endPhase=[214
7483647], --input=[output/clusters-7-final], --output=[clusteranalyze.txt], --outputFormat=[TE
XT], --pointsDir=[output/clusteredPoints], --startPhase=[0], --tempDir=[temp]}
19/03/24 16:35:38 INFO ClusterDumper: Wrote 6 clusters
19/03/24 16:35:38 INFO MahoutDriver: Program took 1848 ms (Minutes: 0.0308)
```

more clusteranalyze.txt

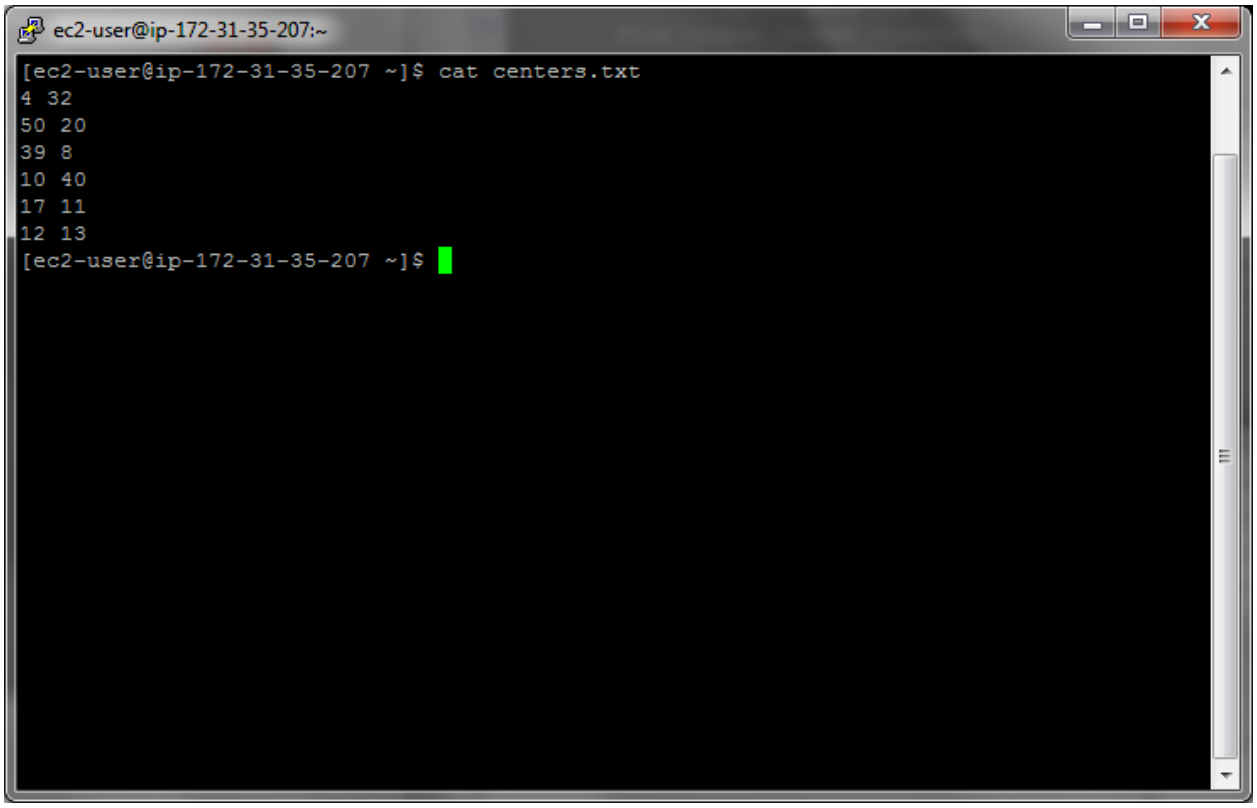
```
ec2-user@ip-172-31-35-207:~
{"identifier":"VL-22","r":[4.815,8.134],"c":[43.75,30.458],"n":24}
        Weight : [props - optional]:  Point:
        1.0 : [distance=5.903888784333448]: [46.0,25.0]
        1.0 : [distance=8.491519462250457]: [43.0,22.0]
        1.0 : [distance=3.550479232128852]: [44.0,34.0]
        1.0 : [distance=6.514540360694015]: [39.0,26.0]
        1.0 : [distance=7.949165749379699]: [41.0,23.0]
        1.0 : [distance=17.683021125110702]: [50.0,47.0]
        1.0 : [distance=4.630252272944889]: [45.0,26.0]
        1.0 : [distance=3.332551991759211]: [46.0,28.0]
        1.0 : [distance=10.875319280115152]: [35.0,24.0]
        1.0 : [distance=12.183564179299559]: [50.0,20.0]
        1.0 : [distance=7.298806028690196]: [39.0,36.0]
        1.0 : [distance=10.174604141903721]: [40.0,21.0]
        1.0 : [distance=11.77663942349902]: [49.0,41.0]
        1.0 : [distance=8.027405316234587]: [41.0,38.0]
        1.0 : [distance=7.212667290569026]: [37.0,33.0]
        1.0 : [distance=13.654275378470706]: [42.0,44.0]
        1.0 : [distance=5.269968637899503]: [49.0,30.0]
        1.0 : [distance=11.6378364016303]: [33.0,26.0]
        1.0 : [distance=11.566009227233236]: [43.0,42.0]
        1.0 : [distance=11.336779500565596]: [50.0,21.0]
        1.0 : [distance=7.229977139966927]: [47.0,24.0]
        1.0 : [distance=9.120813347016334]: [49.0,23.0]
        1.0 : [distance=7.545809175370877]: [44.0,38.0]
        1.0 : [distance=9.54057490114953]: [48.0,39.0]
--More--(23%)
```

B. (25 pts) Using Hadoop streaming perform four iterations manually **using 6 centers** (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 with new centers can be given to the next pass of the same code.

The only difference between first and subsequent iteration is that in first iteration you have to pick the initial centers. Starting from 2nd iteration, the centers will be given to you by a previous pass of KMeans.

Note: I used the single-node Hadoop instance for this exercise.

Using the same 100 points from exercise 3.A, I randomly picked six starting centers from the list:



I used the same testdate2.txt file for this exercise as I did for part 3.A.

Code:

kmeansMapper.py

```
#!/usr/bin/python

import sys
import math

fd = open('centers.txt', 'r')
centers = []

for line in fd:
    line = line.strip()
    vals = line.split(' ')
    centers.extend([vals])
fd.close()

for line in sys.stdin:
```

```
        line = line.strip()
        vals = line.split(' ')

        clusterNum = None
        distance = None
        i = 0

        #compare to each center and store the smallest distance
        for center in centers:

            euclidDist = math.sqrt( (float(vals[0])-float(center[0]))**2 + (float(v$

            if clusterNum:
                if euclidDist < distance:
                    clusterNum = i+1
                    distance = euclidDist
            else: #always record the first cluster
                clusterNum = i+1
                distance = euclidDist

            i += 1

        print  clusterNum, '\t', vals[0], '\t', vals[1]
```

kmeansReducer.py

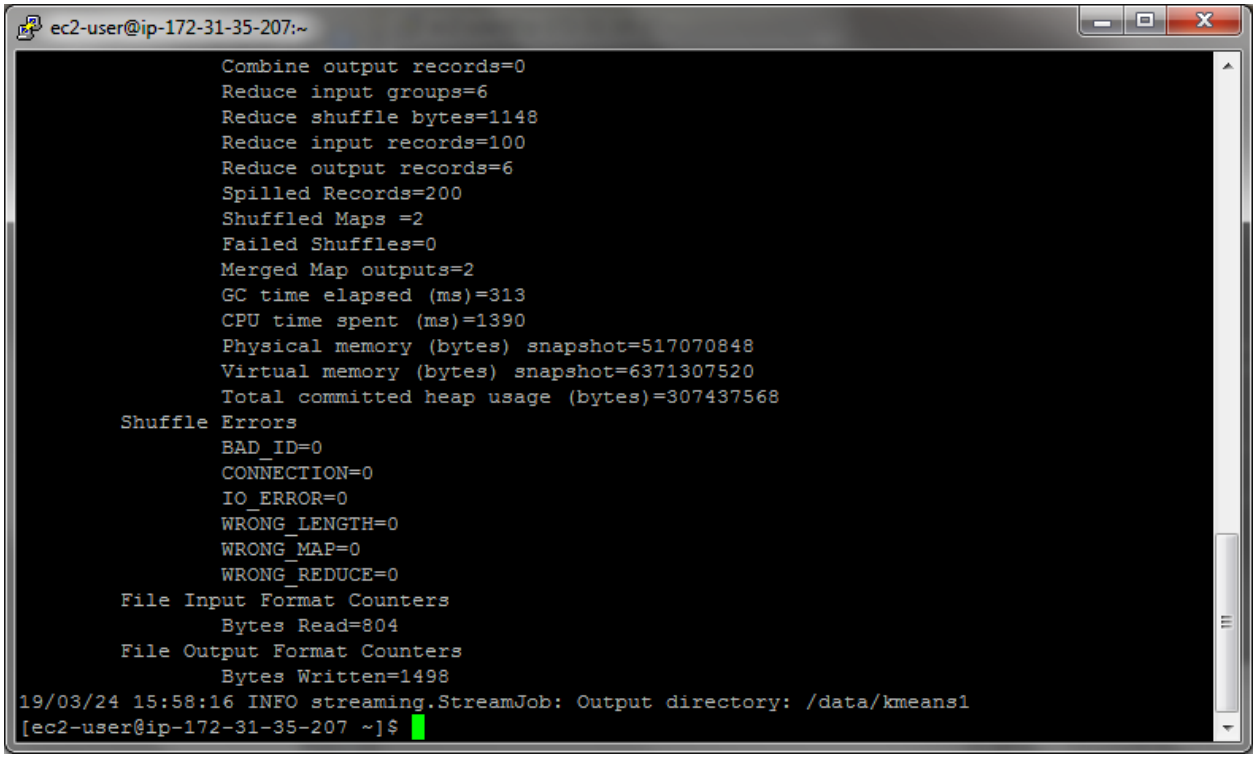
```
#!/usr/bin/python

import sys

currId = None  # this is the "current" key
currXs = []
currYs = []
id = None

# The input comes from standard input (line by line)
for line in sys.stdin:

    line = line.strip()
    ln = line.split('\t')
    id = ln[0]

    if currId == id:
        currXs.append(float(ln[1]))
        currYs.append(float(ln[2]))
    else:
        if currId:
```

```
                #calculate center
                centerX = sum(currXs)/len(currXs)
                centerY = sum(currYs)/len(currYs)
                print '%s %s %s %s' % (centerX, centerY, currId, zip(currXs, cu$

                currXs = []
                currYs = []

            currId = id
            currXs.append(float(ln[1]))
            currYs.append(float(ln[2]))

    # output the last key
    if currId == id:
        #calculate center
        centerX = sum(currXs)/len(currXs)
        centerY = sum(currYs)/len(currYs)
        print '%s %s %s %s' % (centerX, centerY, currId, zip(currXs, currYs))
```

Executions (note cluster output text is also at the end of this file)**:**

Execution 1:

hadoop jar hadoop-streaming-2.6.4.jar -input /data/testdata2.txt -file centers.txt -mapper kmeansMapper.py -file kmeansMapper.py -reducer kmeansReducer.py -file kmeansReducer.py -output /data/kmeans1



```
ec2-user@ip-172-31-35-207:~
                Combine output records=0
                Reduce input groups=6
                Reduce shuffle bytes=1148
                Reduce input records=100
                Reduce output records=6
                Spilled Records=200
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=313
                CPU time spent (ms)=1390
                Physical memory (bytes) snapshot=517070848
                Virtual memory (bytes) snapshot=6371307520
                Total committed heap usage (bytes)=307437568
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1498
19/03/24 15:58:16 INFO streaming.StreamJob: Output directory: /data/kmeans1
[ec2-user@ip-172-31-35-207 ~]$
```

hadoop fs -cat /data/kmeans1/part-00000

```
ec2-user@ip-172-31-35-207:~
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1498
19/03/24 15:58:16 INFO streaming.StreamJob: Output directory: /data/kmeans1
[ec2-user@ip-172-31-35-207 ~]$ hadoop fs -cat /data/kmeans1/part-00000
6.22222222222 30.0 1  [(5.0, 29.0), (7.0, 35.0), (6.0, 32.0), (1.0, 37.0), (14.0, 28.0), (7.0,
 23.0), (2.0, 33.0), (10.0, 31.0), (4.0, 22.0)]
43.619047619 30.1428571429 2  [(39.0, 36.0), (35.0, 24.0), (45.0, 26.0), (49.0, 41.0), (46.0,
28.0), (50.0, 20.0), (40.0, 21.0), (48.0, 15.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (41
.0, 18.0), (49.0, 23.0), (47.0, 24.0), (50.0, 21.0), (43.0, 42.0), (33.0, 26.0), (49.0, 30.0),
 (42.0, 44.0), (37.0, 33.0), (41.0, 38.0)]
38.0 11.0833333333 3  [(36.0, 19.0), (32.0, 9.0), (36.0, 19.0), (45.0, 5.0), (47.0, 9.0), (43.
0, 2.0), (42.0, 11.0), (35.0, 11.0), (34.0, 15.0), (39.0, 8.0), (37.0, 17.0), (30.0, 8.0)]
20.9333333333 40.0666666667 4  [(1.0, 48.0), (25.0, 43.0), (12.0, 38.0), (29.0, 32.0), (27.0,
44.0), (27.0, 29.0), (22.0, 46.0), (28.0, 30.0), (31.0, 45.0), (10.0, 40.0), (8.0, 44.0), (23.
0, 50.0), (34.0, 47.0), (22.0, 34.0), (15.0, 31.0)]
18.8181818182 8.81818181818 5  [(27.0, 26.0), (16.0, 4.0), (13.0, 1.0), (20.0, 8.0), (17.0, 11
.0), (20.0, 4.0), (25.0, 3.0), (24.0, 25.0), (16.0, 5.0), (12.0, 3.0), (17.0, 7.0)]
6.4375 9.0 6  [(14.0, 18.0), (4.0, 10.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0), (
2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (6.0, 12.0), (8.0, 6.0), (8.0, 4.
0), (10.0, 7.0), (10.0, 4.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (11.0, 4.0), (3.0, 4.0), (10
.0, 5.0), (12.0, 6.0), (6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (12.0, 21.0), (12.0, 13.0), (10.0,
 9.0), (2.0, 14.0), (6.0, 18.0), (3.0, 16.0)]
[ec2-user@ip-172-31-35-207 ~]$
```

Note: the first two values are the new center points, the third value is the cluster number and the sets of pairs are the points belonging to those clusters.

Execution 2:

Replace the centers file:

rm centers.txt
hadoop fs -get /data/kmeans1/part-00000 centers.txt

Run with new centers:

hadoop jar hadoop-streaming-2.6.4.jar -input /data/testdata2.txt -file centers.txt -mapper kmeansMapper.py -file kmeansMapper.py -reducer kmeansReducer.py -file kmeansReducer.py -output /data/kmeans2

```
ec2-user@ip-172-31-35-207:~
                Combine output records=0
                Reduce input groups=6
                Reduce shuffle bytes=1148
                Reduce input records=100
                Reduce output records=6
                Spilled Records=200
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=244
                CPU time spent (ms)=1330
                Physical memory (bytes) snapshot=516943872
                Virtual memory (bytes) snapshot=6371332096
                Total committed heap usage (bytes)=307437568
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1533
19/03/24 16:04:33 INFO streaming.StreamJob: Output directory: /data/kmeans2
[ec2-user@ip-172-31-35-207 ~]$
```

hadoop fs -cat /data/kmeans2/part-00000

```
ec2-user@ip-172-31-35-207:~
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1533
19/03/24 16:04:33 INFO streaming.StreamJob: Output directory: /data/kmeans2
[ec2-user@ip-172-31-35-207 ~]$ hadoop fs -cat /data/kmeans2/part-00000
7.23076923077 31.5384615385 1  [(10.0, 40.0), (15.0, 31.0), (7.0, 35.0), (5.0, 29.0), (14.0, 2
8.0), (6.0, 32.0), (12.0, 21.0), (1.0, 37.0), (7.0, 23.0), (1.0, 48.0), (4.0, 22.0), (10.0, 31
.0), (2.0, 33.0)]
43.5263157895 31.5789473684 2  [(40.0, 21.0), (39.0, 36.0), (50.0, 20.0), (35.0, 24.0), (49.0,
 41.0), (46.0, 28.0), (45.0, 26.0), (47.0, 24.0), (50.0, 21.0), (33.0, 26.0), (43.0, 42.0), (4
1.0, 38.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (42.0, 44.0), (49.0, 30.0), (49.0, 23.0)
, (37.0, 33.0)]
38.9285714286 11.8571428571 3  [(48.0, 15.0), (45.0, 5.0), (36.0, 19.0), (41.0, 18.0), (32.0,
9.0), (36.0, 19.0), (43.0, 2.0), (39.0, 8.0), (35.0, 11.0), (34.0, 15.0), (37.0, 17.0), (30.0,
 8.0), (47.0, 9.0), (42.0, 11.0)]
24.2142857143 38.0714285714 4  [(29.0, 32.0), (27.0, 44.0), (12.0, 38.0), (25.0, 43.0), (24.0,
 25.0), (27.0, 26.0), (31.0, 45.0), (27.0, 29.0), (34.0, 47.0), (8.0, 44.0), (22.0, 34.0), (22
.0, 46.0), (23.0, 50.0), (28.0, 30.0)]
17.5555555556 6.77777777778 5  [(20.0, 4.0), (13.0, 1.0), (16.0, 4.0), (17.0, 11.0), (20.0, 8.
0), (25.0, 3.0), (17.0, 7.0), (14.0, 18.0), (16.0, 5.0)]
6.1935483871 8.12903225806 6  [(6.0, 12.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0),
 (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (8.0, 6.0), (8.0, 4.0), (10.0,
7.0), (10.0, 4.0), (12.0, 3.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (11.0, 4.0), (3.0, 4.0), (
10.0, 5.0), (12.0, 6.0), (6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (4.0, 10.0), (3.0, 16.0), (12.0,
 13.0), (6.0, 18.0), (2.0, 14.0), (10.0, 9.0)]
[ec2-user@ip-172-31-35-207 ~]$
```

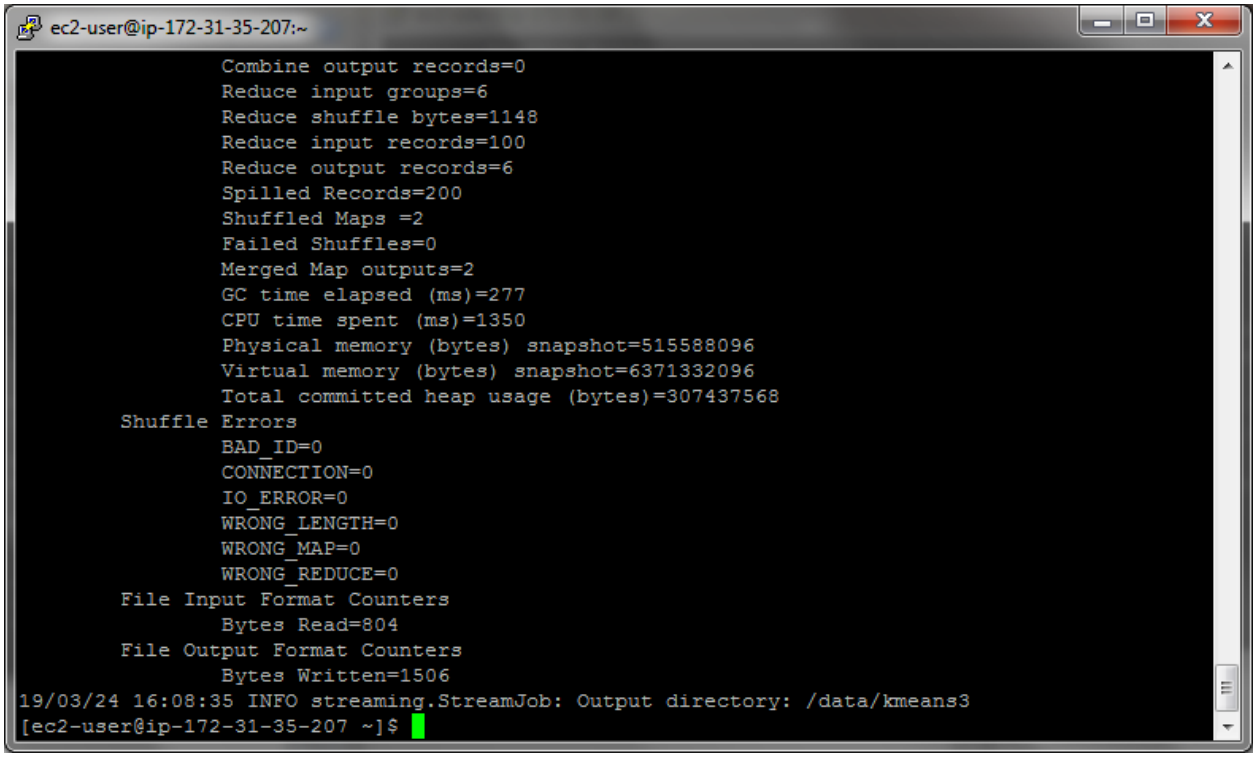
Execution 3:

Replace the centers file:

```
rm centers.txt
hadoop fs -get /data/kmeans2/part-00000 centers.txt
```

Run with new centers:

```
hadoop jar hadoop-streaming-2.6.4.jar -input /data/testdata2.txt -file centers.txt -mapper
kmeansMapper.py -file kmeansMapper.py -reducer kmeansReducer.py -file
kmeansReducer.py -output /data/kmeans3
```



```
ec2-user@ip-172-31-35-207:~
                Combine output records=0
                Reduce input groups=6
                Reduce shuffle bytes=1148
                Reduce input records=100
                Reduce output records=6
                Spilled Records=200
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=277
                CPU time spent (ms)=1350
                Physical memory (bytes) snapshot=515588096
                Virtual memory (bytes) snapshot=6371332096
                Total committed heap usage (bytes)=307437568
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1506
19/03/24 16:08:35 INFO streaming.StreamJob: Output directory: /data/kmeans3
[ec2-user@ip-172-31-35-207 ~]$
```

hadoop fs -cat /data/kmeans3/part-00000

```
ec2-user@ip-172-31-35-207:~
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1506
19/03/24 16:08:35 INFO streaming.StreamJob: Output directory: /data/kmeans3
[ec2-user@ip-172-31-35-207 ~]$ hadoop fs -cat /data/kmeans3/part-00000
7.6 32.8 1  [(10.0, 40.0), (8.0, 44.0), (7.0, 35.0), (15.0, 31.0), (5.0, 29.0), (12.0, 21.0),
(14.0, 28.0), (6.0, 32.0), (1.0, 37.0), (12.0, 38.0), (4.0, 22.0), (10.0, 31.0), (2.0, 33.0),
(1.0, 48.0), (7.0, 23.0)]
43.7222222222 32.1666666667 2  [(35.0, 24.0), (39.0, 36.0), (50.0, 20.0), (49.0, 41.0), (46.0,
 28.0), (45.0, 26.0), (43.0, 42.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (49.0, 23.0), (4
7.0, 24.0), (50.0, 21.0), (33.0, 26.0), (49.0, 30.0), (42.0, 44.0), (37.0, 33.0), (41.0, 38.0)
]
39.0 12.4666666667 3  [(36.0, 19.0), (41.0, 18.0), (45.0, 5.0), (36.0, 19.0), (48.0, 15.0), (3
2.0, 9.0), (43.0, 2.0), (39.0, 8.0), (35.0, 11.0), (34.0, 15.0), (37.0, 17.0), (30.0, 8.0), (4
7.0, 9.0), (42.0, 11.0), (40.0, 21.0)]
26.5833333333 37.5833333333 4  [(25.0, 43.0), (27.0, 44.0), (29.0, 32.0), (34.0, 47.0), (22.0,
 34.0), (28.0, 30.0), (27.0, 26.0), (31.0, 45.0), (23.0, 50.0), (24.0, 25.0), (27.0, 29.0), (2
2.0, 46.0)]
16.5454545455 6.36363636364 5  [(25.0, 3.0), (20.0, 4.0), (17.0, 11.0), (16.0, 4.0), (13.0, 1.
0), (20.0, 8.0), (17.0, 7.0), (14.0, 18.0), (12.0, 6.0), (12.0, 3.0), (16.0, 5.0)]
5.79310344828 8.37931034483 6  [(8.0, 6.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0),
 (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (6.0, 12.0), (8.0, 4.0), (10.0,
 7.0), (10.0, 4.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (11.0, 4.0), (3.0, 4.0), (10.0, 5.0),
(6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (4.0, 10.0), (3.0, 16.0), (2.0, 14.0), (10.0, 9.0), (6.0,
 18.0), (12.0, 13.0)]
[ec2-user@ip-172-31-35-207 ~]$
```

Execution 4:

Replace the centers file:

rm centers.txt
hadoop fs -get /data/kmeans3/part-00000 centers.txt

Run with new centers:

hadoop jar hadoop-streaming-2.6.4.jar -input /data/testdata2.txt -file centers.txt -mapper kmeansMapper.py -file kmeansMapper.py -reducer kmeansReducer.py -file kmeansReducer.py -output /data/kmeans4

```
ec2-user@ip-172-31-35-207:~
                Combine output records=0
                Reduce input groups=6
                Reduce shuffle bytes=1148
                Reduce input records=100
                Reduce output records=6
                Spilled Records=200
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=317
                CPU time spent (ms)=1350
                Physical memory (bytes) snapshot=514560000
                Virtual memory (bytes) snapshot=6371307520
                Total committed heap usage (bytes)=307437568
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1503
19/03/24 16:11:32 INFO streaming.StreamJob: Output directory: /data/kmeans4
[ec2-user@ip-172-31-35-207 ~]$
```

hadoop fs -cat /data/kmeans4/part-00000

```
ec2-user@ip-172-31-35-207:~
        File Input Format Counters
                Bytes Read=804
        File Output Format Counters
                Bytes Written=1503
19/03/24 16:11:32 INFO streaming.StreamJob: Output directory: /data/kmeans4
[ec2-user@ip-172-31-35-207 ~]$ hadoop fs -cat /data/kmeans4/part-00000
7.6 32.8 1  [(10.0, 40.0), (8.0, 44.0), (7.0, 35.0), (15.0, 31.0), (5.0, 29.0), (12.0, 21.0),
(14.0, 28.0), (6.0, 32.0), (1.0, 37.0), (12.0, 38.0), (4.0, 22.0), (10.0, 31.0), (2.0, 33.0),
(1.0, 48.0), (7.0, 23.0)]
43.3529411765 32.8823529412 2  [(45.0, 26.0), (35.0, 24.0), (39.0, 36.0), (49.0, 41.0), (46.0,
 28.0), (43.0, 42.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (49.0, 23.0), (47.0, 24.0), (5
0.0, 21.0), (33.0, 26.0), (49.0, 30.0), (42.0, 44.0), (37.0, 33.0), (41.0, 38.0)]
39.6875 12.9375 3  [(36.0, 19.0), (41.0, 18.0), (45.0, 5.0), (36.0, 19.0), (48.0, 15.0), (32.0
, 9.0), (43.0, 2.0), (39.0, 8.0), (35.0, 11.0), (34.0, 15.0), (37.0, 17.0), (30.0, 8.0), (47.0
, 9.0), (42.0, 11.0), (40.0, 21.0), (50.0, 20.0)]
26.5833333333 37.5833333333 4  [(27.0, 44.0), (29.0, 32.0), (25.0, 43.0), (34.0, 47.0), (22.0,
 34.0), (28.0, 30.0), (27.0, 26.0), (31.0, 45.0), (23.0, 50.0), (24.0, 25.0), (27.0, 29.0), (2
2.0, 46.0)]
16.0833333333 6.16666666667 5  [(25.0, 3.0), (20.0, 4.0), (17.0, 11.0), (16.0, 4.0), (13.0, 1.
0), (20.0, 8.0), (12.0, 6.0), (14.0, 18.0), (12.0, 3.0), (16.0, 5.0), (11.0, 4.0), (17.0, 7.0)
]
5.60714285714 8.53571428571 6  [(8.0, 6.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0),
 (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (6.0, 12.0), (8.0, 4.0), (10.0,
 7.0), (10.0, 4.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (3.0, 4.0), (10.0, 5.0), (6.0, 8.0), (
4.0, 8.0), (7.0, 10.0), (4.0, 10.0), (3.0, 16.0), (2.0, 14.0), (10.0, 9.0), (6.0, 18.0), (12.0
, 13.0)]
[ec2-user@ip-172-31-35-207 ~]$
```

That is the final output.

Extra credit (7 pts): Create the equivalent of KMeans driver from Mahout. That is, write a python script that will automatically execute the hadoop streaming command, then get the new centers from HDFS and repeat the command. This will be easiest to do if you write your reducer to output just the centers (without the key) to HDFS. This way, all you have to do is to execute the get command to get the new centers (you can hard-code the locations of output in HDFS into your script).

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.

testdata2.txt

```
11 35
2 36
14 5
38 8
10 49
17 9
33 45
13 19
27 3
18 16
28 40
4 32
46 25
21 15
29 7
43 22
37 48
44 34
20 12
31 6
39 26
41 23
50 47
24 30
1 42
45 26
46 28
1 48
27 44
35 24
29 32
7 23
16 5
14 18
```

50 20
39 36
40 21
2 33
10 31
4 22
42 11
6 13
47 9
30 8
37 17
3 19
12 38
34 15
25 43
49 41
17 7
35 11
39 8
43 2
48 15
24 25
23 50
41 38
3 16
12 21
22 46
13 1
20 4
37 33
36 19
5 29
34 47
14 28
42 44
49 30
6 18
10 40
31 45
27 26
32 9
28 30
33 26
43 42
16 4
50 21
47 24
27 29

7 35
15 31
49 23
41 18
17 11
2 14
10 9
25 3
22 34
20 8
44 38
40 46
36 19
48 39
12 13
1 37
6 32
45 5

Generated at https://www.random.org/sequences/?mode=advanced

Clustering Output:

Execution 1:

6.22222222222 30.0 1 [(5.0, 29.0), (7.0, 35.0), (6.0, 32.0), (1.0, 37.0), (14.0, 28.0), (7.0, 23.0), (2.0, 33.0), (10.0, 31.0), (4.0, 22.0)]

43.619047619 30.1428571429 2 [(39.0, 36.0), (35.0, 24.0), (45.0, 26.0), (49.0, 41.0), (46.0, 28.0), (50.0, 20.0), (40.0, 21.0), (48.0, 15.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (41.0, 18.0), (49.0, 23.0), (47.0, 24.0), (50.0, 21.0), (43.0, 42.0), (33.0, 26.0), (49.0, 30.0), (42.0, 44.0), (37.0, 33.0), (41.0, 38.0)]

38.0 11.0833333333 3 [(36.0, 19.0), (32.0, 9.0), (36.0, 19.0), (45.0, 5.0), (47.0, 9.0), (43.0, 2.0), (42.0, 11.0), (35.0, 11.0), (34.0, 15.0), (39.0, 8.0), (37.0, 17.0), (30.0, 8.0)]

20.9333333333 40.0666666667 4 [(1.0, 48.0), (25.0, 43.0), (12.0, 38.0), (29.0, 32.0), (27.0, 44.0), (27.0, 29.0), (22.0, 46.0), (28.0, 30.0), (31.0, 45.0), (10.0, 40.0), (8.0, 44.0), (23.0, 50.0), (34.0, 47.0), (22.0, 34.0), (15.0, 31.0)]

18.8181818182 8.81818181818 5 [(27.0, 26.0), (16.0, 4.0), (13.0, 1.0), (20.0, 8.0), (17.0, 11.0), (20.0, 4.0), (25.0, 3.0), (24.0, 25.0), (16.0, 5.0), (12.0, 3.0), (17.0, 7.0)]

6.4375 9.0 6 [(14.0, 18.0), (4.0, 10.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0), (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (6.0, 12.0), (8.0, 6.0), (8.0, 4.0), (10.0, 7.0), (10.0, 4.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (11.0, 4.0), (3.0, 4.0), (10.0, 5.0), (12.0, 6.0), (6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (12.0, 21.0), (12.0, 13.0), (10.0, 9.0), (2.0, 14.0), (6.0, 18.0), (3.0, 16.0)]

Execution 2:

7.23076923077 31.5384615385 1 [(10.0, 40.0), (15.0, 31.0), (7.0, 35.0), (5.0, 29.0), (14.0, 28.0), (6.0, 32.0), (12.0, 21.0), (1.0, 37.0), (7.0, 23.0), (1.0, 48.0), (4.0, 22.0), (10.0, 31.0), (2.0, 33.0)]

43.5263157895 31.5789473684 2 [(40.0, 21.0), (39.0, 36.0), (50.0, 20.0), (35.0, 24.0), (49.0, 41.0), (46.0, 28.0), (45.0, 26.0), (47.0, 24.0), (50.0, 21.0), (33.0, 26.0), (43.0, 42.0), (41.0, 38.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (42.0, 44.0), (49.0, 30.0), (49.0, 23.0), (37.0, 33.0)]

38.9285714286 11.8571428571 3 [(48.0, 15.0), (45.0, 5.0), (36.0, 19.0), (41.0, 18.0), (32.0, 9.0), (36.0, 19.0), (43.0, 2.0), (39.0, 8.0), (35.0, 11.0), (34.0, 15.0), (37.0, 17.0), (30.0, 8.0), (47.0, 9.0), (42.0, 11.0)]

24.2142857143 38.0714285714 4 [(29.0, 32.0), (27.0, 44.0), (12.0, 38.0), (25.0, 43.0), (24.0, 25.0), (27.0, 26.0), (31.0, 45.0), (27.0, 29.0), (34.0, 47.0), (8.0, 44.0), (22.0, 34.0), (22.0, 46.0), (23.0, 50.0), (28.0, 30.0)]

17.5555555556 6.77777777778 5 [(20.0, 4.0), (13.0, 1.0), (16.0, 4.0), (17.0, 11.0), (20.0, 8.0), (25.0, 3.0), (17.0, 7.0), (14.0, 18.0), (16.0, 5.0)]

6.1935483871 8.12903225806 6 [(6.0, 12.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0), (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (8.0, 6.0), (8.0, 4.0), (10.0, 7.0), (10.0, 4.0), (12.0, 3.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (11.0, 4.0), (3.0, 4.0), (10.0, 5.0), (12.0, 6.0), (6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (4.0, 10.0), (3.0, 16.0), (12.0, 13.0), (6.0, 18.0), (2.0, 14.0), (10.0, 9.0)]

Execution 3:

7.6 32.8 1 [(10.0, 40.0), (8.0, 44.0), (7.0, 35.0), (15.0, 31.0), (5.0, 29.0), (12.0, 21.0), (14.0, 28.0), (6.0, 32.0), (1.0, 37.0), (12.0, 38.0), (4.0, 22.0), (10.0, 31.0), (2.0, 33.0), (1.0, 48.0), (7.0, 23.0)]

43.7222222222 32.1666666667 2 [(35.0, 24.0), (39.0, 36.0), (50.0, 20.0), (49.0, 41.0), (46.0, 28.0), (45.0, 26.0), (43.0, 42.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (49.0, 23.0), (47.0, 24.0), (50.0, 21.0), (33.0, 26.0), (49.0, 30.0), (42.0, 44.0), (37.0, 33.0), (41.0, 38.0)]

39.0 12.4666666667 3 [(36.0, 19.0), (41.0, 18.0), (45.0, 5.0), (36.0, 19.0), (48.0, 15.0), (32.0, 9.0), (43.0, 2.0), (39.0, 8.0), (35.0, 11.0), (34.0, 15.0), (37.0, 17.0), (30.0, 8.0), (47.0, 9.0), (42.0, 11.0), (40.0, 21.0)]

26.5833333333 37.5833333333 4 [(25.0, 43.0), (27.0, 44.0), (29.0, 32.0), (34.0, 47.0), (22.0, 34.0), (28.0, 30.0), (27.0, 26.0), (31.0, 45.0), (23.0, 50.0), (24.0, 25.0), (27.0, 29.0), (22.0, 46.0)]

16.5454545455 6.36363636364 5 [(25.0, 3.0), (20.0, 4.0), (17.0, 11.0), (16.0, 4.0), (13.0, 1.0), (20.0, 8.0), (17.0, 7.0), (14.0, 18.0), (12.0, 6.0), (12.0, 3.0), (16.0, 5.0)]

5.79310344828 8.37931034483 6 [(8.0, 6.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0), (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (6.0, 12.0), (8.0, 4.0), (10.0, 7.0), (10.0, 4.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (11.0, 4.0), (3.0, 4.0), (10.0, 5.0), (6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (4.0, 10.0), (3.0, 16.0), (2.0, 14.0), (10.0, 9.0), (6.0, 18.0), (12.0, 13.0)]

Execution 4:

7.6 32.8 1 [(10.0, 40.0), (8.0, 44.0), (7.0, 35.0), (15.0, 31.0), (5.0, 29.0), (12.0, 21.0), (14.0, 28.0), (6.0, 32.0), (1.0, 37.0), (12.0, 38.0), (4.0, 22.0), (10.0, 31.0), (2.0, 33.0), (1.0, 48.0), (7.0, 23.0)]

43.3529411765 32.8823529412 2 [(45.0, 26.0), (35.0, 24.0), (39.0, 36.0), (49.0, 41.0), (46.0, 28.0), (43.0, 42.0), (48.0, 39.0), (40.0, 46.0), (44.0, 38.0), (49.0, 23.0), (47.0, 24.0), (50.0, 21.0), (33.0, 26.0), (49.0, 30.0), (42.0, 44.0), (37.0, 33.0), (41.0, 38.0)]

39.6875 12.9375 3 [(36.0, 19.0), (41.0, 18.0), (45.0, 5.0), (36.0, 19.0), (48.0, 15.0), (32.0, 9.0), (43.0, 2.0), (39.0, 8.0), (35.0, 11.0), (34.0, 15.0), (37.0, 17.0), (30.0, 8.0), (47.0, 9.0), (42.0, 11.0), (40.0, 21.0), (50.0, 20.0)]

26.5833333333 37.5833333333 4 [(27.0, 44.0), (29.0, 32.0), (25.0, 43.0), (34.0, 47.0), (22.0, 34.0), (28.0, 30.0), (27.0, 26.0), (31.0, 45.0), (23.0, 50.0), (24.0, 25.0), (27.0, 29.0), (22.0, 46.0)]

16.0833333333 6.16666666667 5 [(25.0, 3.0), (20.0, 4.0), (17.0, 11.0), (16.0, 4.0), (13.0, 1.0), (20.0, 8.0), (12.0, 6.0), (14.0, 18.0), (12.0, 3.0), (16.0, 5.0), (11.0, 4.0), (17.0, 7.0)]

5.60714285714 8.53571428571 6 [(8.0, 6.0), (6.0, 13.0), (3.0, 19.0), (3.0, 12.0), (3.0, 9.0), (2.0, 5.0), (2.0, 2.0), (4.0, 11.0), (4.0, 3.0), (5.0, 10.0), (6.0, 12.0), (8.0, 4.0), (10.0, 7.0), (10.0, 4.0), (9.0, 3.0), (5.0, 2.0), (2.0, 2.0), (3.0, 4.0), (10.0, 5.0), (6.0, 8.0), (4.0, 8.0), (7.0, 10.0), (4.0, 10.0), (3.0, 16.0), (2.0, 14.0), (10.0, 9.0), (6.0, 18.0), (12.0, 13.0)]