Ronaldlee Ejalu

CSC 555 Project Phase 1

In this part of the project (which will serve as our take-home midterm), you will 1) Set up a 3-node cluster and 2) perform data warehousing and transformation queries using Hive, Pig and Hadoop streaming on that cluster. The modified Hive-style schema is:
http://cdmgcsarprd01.dpu.depaul.edu/CSC555/SSBM1/SSBM_schema_hive.sql
(**you still have to add the delimiter to table definitions**)
It is based on SSBM benchmark (derived from industry standard TPCH benchmark). The data is at Scale1, or the smallest unit – lineorder is the largest table at about 0.6GB. You can use wget to download the following links. Keep in mind that data is |-separated.
http://cdmgcsarprd01.dpu.depaul.edu/CSC555/SSBM1/dwdate.tbl
http://cdmgcsarprd01.dpu.depaul.edu/CSC555/SSBM1/lineorder.tbl
http://cdmgcsarprd01.dpu.depaul.edu/CSC555/SSBM1/part.tbl
http://cdmgcsarprd01.dpu.depaul.edu/CSC555/SSBM1/supplier.tbl
http://cdmgcsarprd01.dpu.depaul.edu/CSC555/SSBM1/customer.tbl
Please be sure to submit all code (pig, python and HiveQL).

Part 1: Multi-node cluster

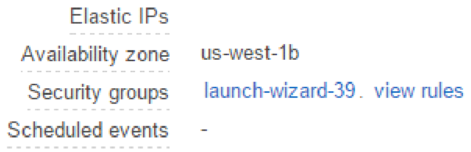
1) Your first step is to setup a multi-node cluster and re-run wordcount. For this part, you will create a 3-node cluster (with a total of 1 master + 2 worker nodes). Include your master node in the workers file, to make sure **all 3** nodes are working.
   You need to perform the following steps:

   1. Create a medium machine on AWS (which will serve as your master). It is possible, but I do not recommend trying to reconfigure your existing Hadoop setup into this new cluster (it is much easier to make 3 new nodes for a total of 4).

      a. When creating a node I recommend changing the default 8G hard drive to 20G.

      b. Change your security group setting to open firewall access. We need to open the ports in two different ways. We will open port 50070 for the web interface in order to be able to see the cluster status in a browser. We will also set 0-64000 range opening up all ports. However, we will ensure that the ports are open only **within** the cluster and not to the world.

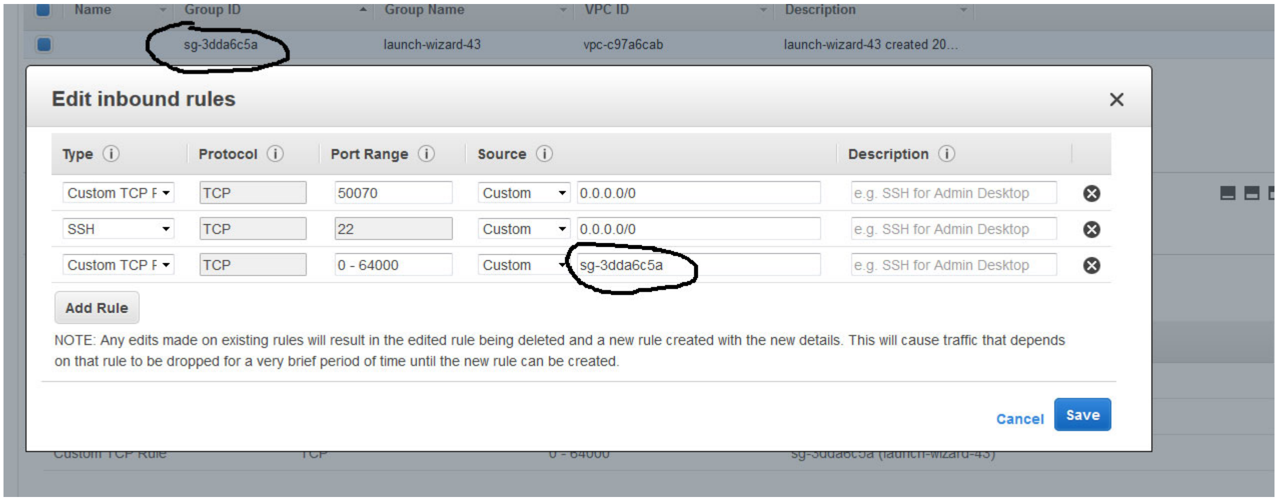
In order to make changes, you need to do the following. Access the cluster security group (launch-wizard-xx).



Elastic IPs	
Availability zone	us-west-1b
Security groups	launch-wizard-39 . view rules
Scheduled events	-

Right click on the security group and choose Edit inbound rules

Note that the first line below is opening port 50070. The second line below is the default (port 22 is required for regular SSH connections). The third line opens all ports but ONLY for the same security group (assuming that all of your nodes in the cluster share the same security group). Please note that we previously had some issues with machines being hacked without that last limitation, so please don't skip this step



Name	Group ID	Group Name	VPC ID	Description
	sg-3dda6c5a	launch-wizard-43	vpc-c97a6cab	launch-wizard-43 created 20...

Edit inbound rules

Type	Protocol	Port Range	Source		Description
Custom TCP F	TCP	50070	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP F	TCP	0 - 64000	Custom	sg-3dda6c5a	e.g. SSH for Admin Desktop

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

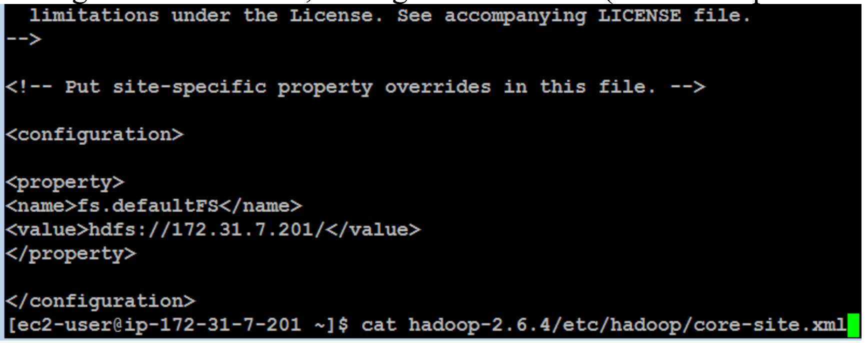
c. Create two new small machines and make sure they are using the same security group that you have configured on the master. You would need to change the security group settings so that both of the workers are sharing the same security group. For that, you can go to "Networking", "Change Security Groups" and check the security group you want.
NOTE: Please make sure to label the machines so that they are easy to find, as it may get a little cluttered.

2. Connect to the master and set up Hadoop similarly to what you did previously. Use the following link:

   http://dbgroup.cdm.depaul.edu/Courses/CSC555/hadoop-2.6.4.tar.gz

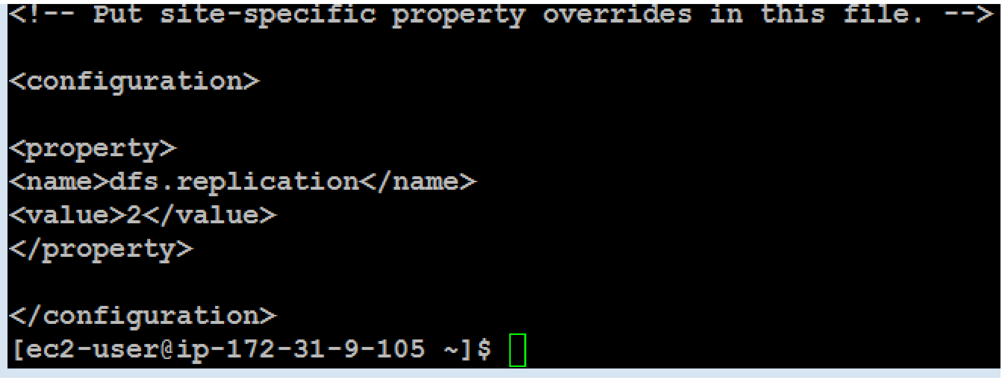
   Do not set up Hadoop on the workers – you will only need to configure up Hadoop once.

   a. Configure core-site.xml, adding the **PrivateIP** (do not use public IP) of the master.



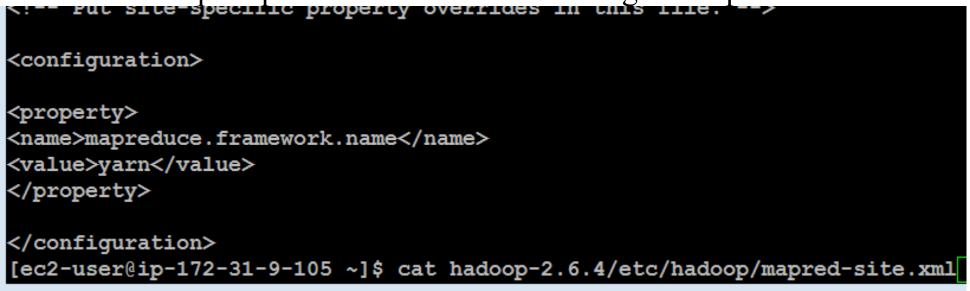
```
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>
<name>fs.defaultFS</name>
<value>hdfs://172.31.7.201/</value>
</property>

</configuration>
[ec2-user@ip-172-31-7-201 ~]$ cat hadoop-2.6.4/etc/hadoop/core-site.xml
```

   b. Configure hdfs-site and set replication factor to 2.



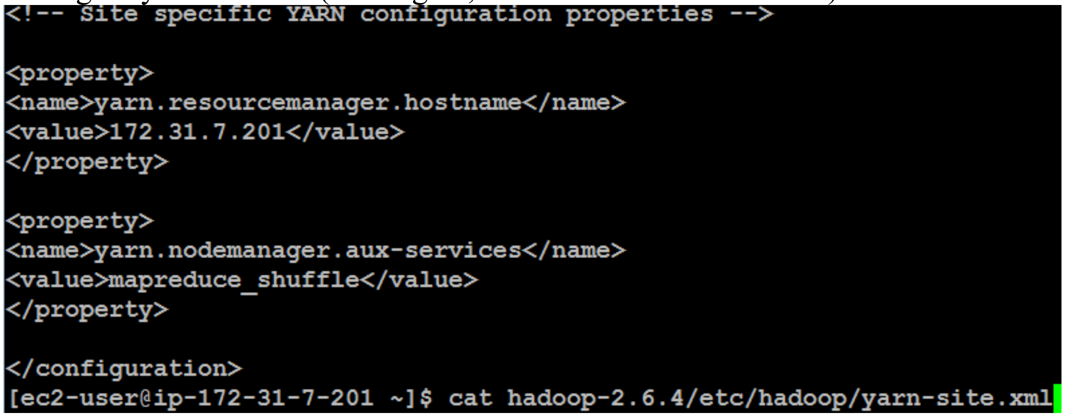
```
<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>
<name>dfs.replication</name>
<value>2</value>
</property>

</configuration>
[ec2-user@ip-172-31-9-105 ~]$ 
```

c. cp hadoop-2.6.4/etc/hadoop/mapred-site.xml.template hadoop-2.6.4/etc/hadoop/mapred-site.xml and then configure mapred-site.xml



```
<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>

</configuration>
[ec2-user@ip-172-31-9-105 ~]$ cat hadoop-2.6.4/etc/hadoop/mapred-site.xml
```

d. Configure yarn-site.xml (once again, use PrivateIP of the master)



```
<!-- Site specific YARN configuration properties -->

<property>
<name>yarn.resourcemanager.hostname</name>
<value>172.31.7.201</value>
</property>

<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>

</configuration>
[ec2-user@ip-172-31-7-201 ~]$ cat hadoop-2.6.4/etc/hadoop/yarn-site.xml
```

Finally, edit the workers file and list your 3 nodes (master and 2 workers) using Private IPs

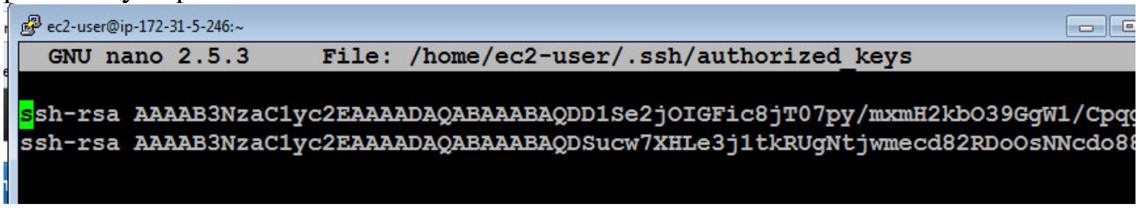
```
[ec2-user@ip-172-31-7-201 ~]$ cat hadoop-2.6.4/etc/hadoop/workers
172.31.7.201
172.31.5.246
…
```

Make sure that you use private IP (private DNS is also ok) for your configuration files (such as conf/masters and conf/workers or the other 3 config files). The advantage of the Private IP is that it does not change after your instance is stopped (if you use the Public IP, the cluster would need to be reconfigured every time it is stopped). The downside of the Private IP is that it is only meaningful within the Amazon EC2 network. So all nodes in EC2 can talk to each other using

Private IP, but you cannot connect to your instance from the outside (e.g., from your laptop) because Private IP has no meaning for your laptop (since your laptop is not part of the Amazon EC2 network).

Now, we will pack up and move Hadoop to the workers. All you need to do is to generate and then copy the public key to the worker nodes to achieve passwordless access across your cluster.

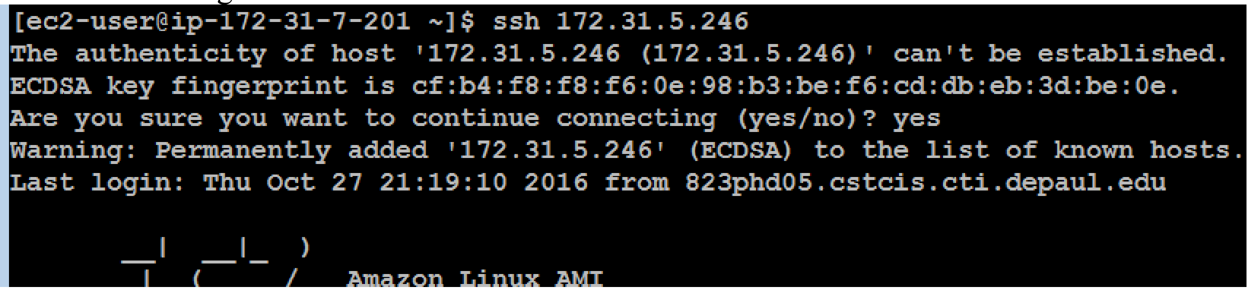
1. Run ssh-keygen -t rsa (and enter empty values for the passphrase) on the master node. That will generate .ssh/id_rsa and .ssh/id_rsa.pub (private and public key). You now need to manually copy the .ssh/id_rsa.pub and append it to ~/.ssh/authorized_keys **on each worker.**
   Keep in mind that this is a single-line public key and accidentally introducing a line break (like discussed in class) would prevent the key from matching it's private key pair.
   Note that the example below is NOT the master, but one of the workers (ip-172-31-5-246). The first public key is the .pem Amazon half and the 2nd public key is the master's public key copied in as one line.



```
  GNU nano 2.5.3      File: /home/ec2-user/.ssh/authorized_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDD1Se2jOIGFic8jT07py/mxmH2kbO39GgW1/Cpq
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDSucw7XHLe3j1tkRUgNtjwmecd82RDoOsNNcdo8
```

   You can add the public key of the master to the master by running this command:
   cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

Make sure that you can ssh to all of the nodes from the master node (by running ssh 54.186.221.92, where the IP address is your worker node) from the master and ensuring that you were able to login. You can exit after successful ssh connection by typing exit (the command prompt will tell you which machine you are connected to, e.g., ec2-user@ip-172-31-37-113). Here's me ssh-ing from master to worker.



```
[ec2-user@ip-172-31-7-201 ~]$ ssh 172.31.5.246
The authenticity of host '172.31.5.246 (172.31.5.246)' can't be established.
ECDSA key fingerprint is cf:b4:f8:f8:f6:0e:98:b3:be:f6:cd:db:eb:3d:be:0e.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.31.5.246' (ECDSA) to the list of known hosts.
Last login: Thu Oct 27 21:19:10 2016 from 823phd05.cstcis.cti.depaul.edu

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
```

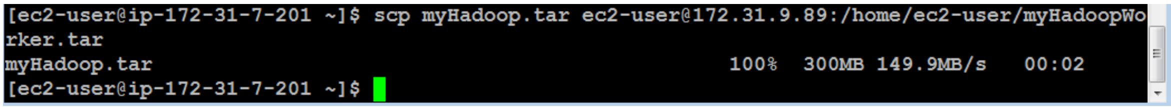
Once you have verified that you can ssh from the master node to every cluster member including the master itself (ssh localhost), you are going to return to the master node (**exit** until your prompt shows the IP address of the master node) and pack the contents of the hadoop directory there. Make sure your Hadoop installation is configured correctly (because from now on, you will have 4 copies of the Hadoop directory and all changes need to be applied in 4 places).

cd (go to root home directory, i.e. /home/ec2-user/)

(pack up the entire Hadoop directory into a single file for transfer. You can optionally compress the file with gzip)
tar cvf myHadoop.tar hadoop-2.6.4
ls -al myHadoop.tar (to verify that the .tar file had been created)

Now, you need to copy the myHadoop.tar file to every non-master node in the cluster. If you had successfully setup public-private key access in the previous step, this command (for each worker node) will do that:

(copies the myHadoop.tar file from the current node to a remote node into a file called myHadoopWorker.tar. Don't forget to replace the IP address with that your worker nodes. By the way, since you are on the Amazon EC2 network, either Public or Private IP will work just fine.)
scp myHadoop.tar ec2-user@54.187.63.189:/home/ec2-user/myHadoopWorker.tar



```
[ec2-user@ip-172-31-7-201 ~]$ scp myHadoop.tar ec2-user@172.31.9.89:/home/ec2-user/myHadoopWo
rker.tar
myHadoop.tar                                          100%  300MB 149.9MB/s   00:02
[ec2-user@ip-172-31-7-201 ~]$
```

Once the tar file containing your Hadoop installation from master node has been copied to each worker node, you need to login to each worker node and unpack the .tar file.
You also need to install Java using **sudo yum install ant**. Without Java on the worker nodes, Hadoop will not start.

Run the following command (on each worker node, not on the master) to untar the hadoop file. We are purposely using a different tar archive name (i.e., **myHadoopWorker.tar**), so if you get "file not found" error, that means you are running this command on the master node or have not yet successfully copied myHadoopWorker.tar file to the worker.

tar xvf myHadoopWorker.tar

Once you are done, run this on the master (nothing needs to be done on the workers to format the cluster unless you are re-formatting, in which case you'll need to delete the dfs directory).
hadoop namenode -format

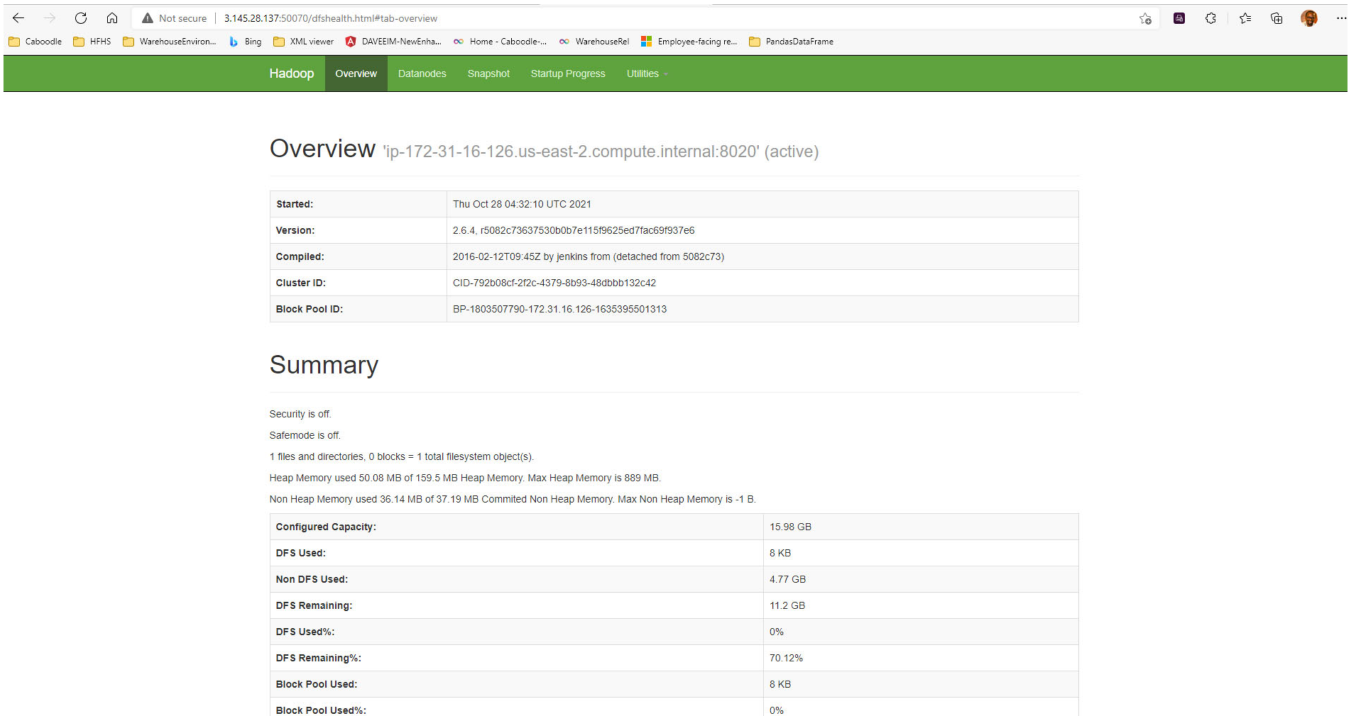
Once you have successfully completed the previous steps, you should can start and use your new cluster by going to the master node and running the start-dfs.sh and start-yarn.sh scripts (you do not need to explicitly start anything on worker nodes – the master will do that for you).

You should verify that the cluster is running by pointing your browser to the link below.

http://[insert-the-public-ip-of-master]:50070/

Make sure that the cluster is operational (you can see the 3 nodes under Datanodes tab).

Submit a screenshot of your cluster status view.



Overview 'ip-172-31-16-126.us-east-2.compute.internal:8020' (active)

Started:	Thu Oct 28 04:32:10 UTC 2021
Version:	2.6.4, r5082c73637530b0b7e115f9625ed7fac69f937e6
Compiled:	2016-02-12T09:45Z by jenkins from (detached from 5082c73)
Cluster ID:	CID-792b08cf-2f2c-4379-8b93-48dbbb132c42
Block Pool ID:	BP-1803507790-172.31.16.126-1635395501313

Summary

Security is off.

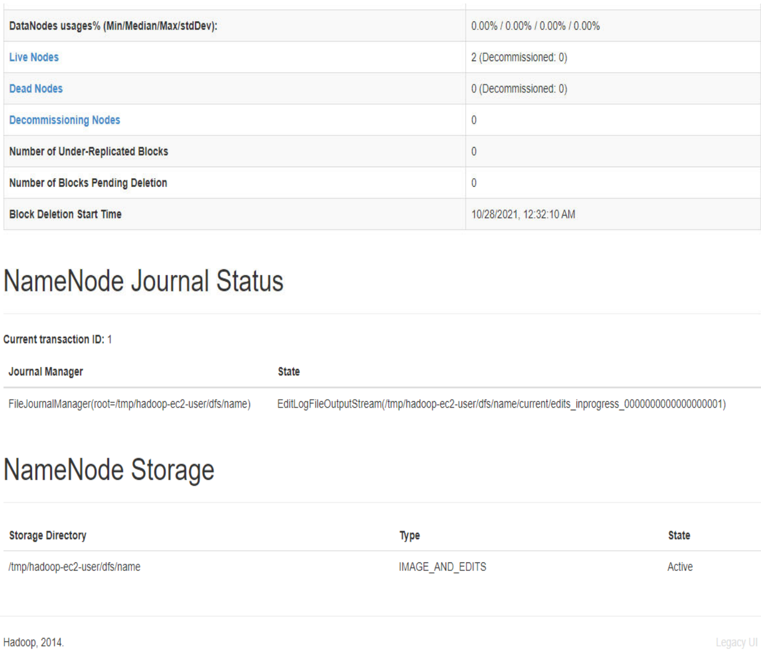
Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

Heap Memory used 50.08 MB of 159.5 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 36.14 MB of 37.19 MB Commited Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	15.98 GB
DFS Used:	8 KB
Non DFS Used:	4.77 GB
DFS Remaining:	11.2 GB
DFS Used%:	0%
DFS Remaining%:	70.12%
Block Pool Used:	8 KB
Block Pool Used%:	0%



DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	2 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0
Block Deletion Start Time	10/28/2021, 12:32:10 AM

NameNode Journal Status

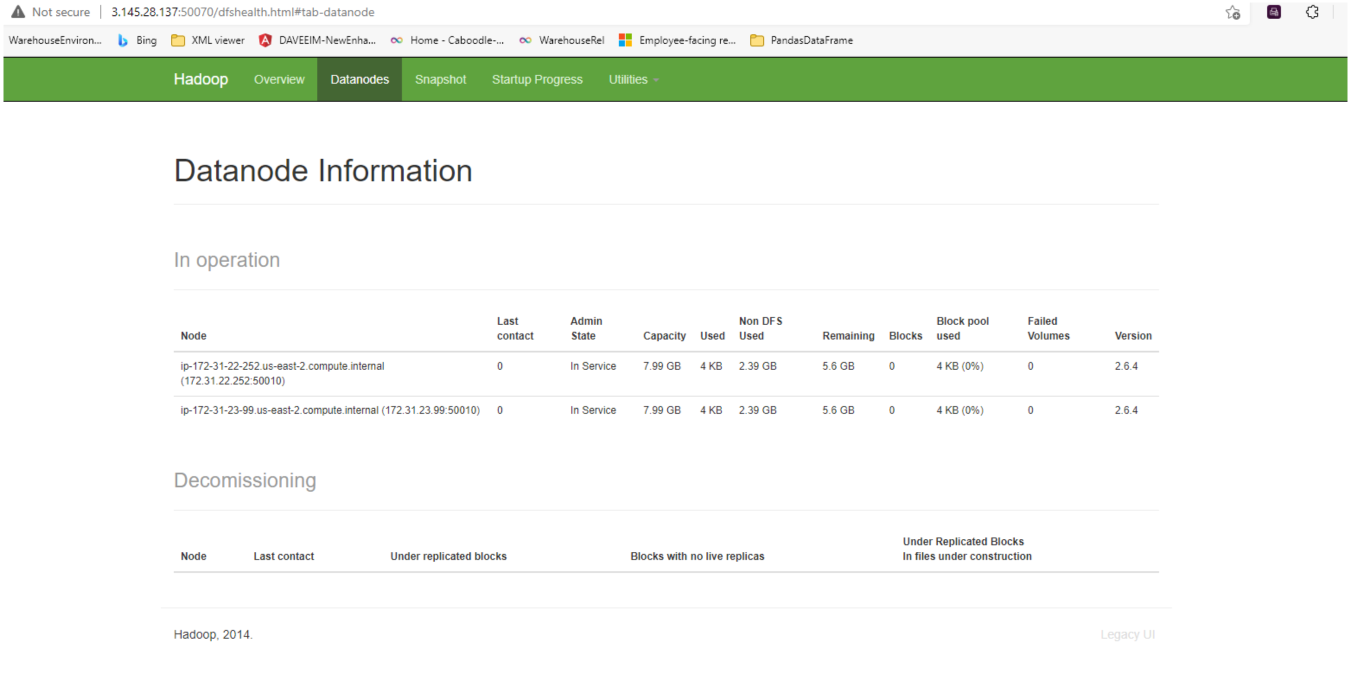
Current transaction ID: 1

Journal Manager	State
FileJournalManager(root=/tmp/hadoop-ec2-user/dfs/name)	EditLogFileOutputStream(/tmp/hadoop-ec2-user/dfs/name/current/edits_inprogress_0000000000000000001)

NameNode Storage

Storage Directory	Type	State
/tmp/hadoop-ec2-user/dfs/name	IMAGE_AND_EDITS	Active

Hadoop, 2014.



Datanode Information

In operation

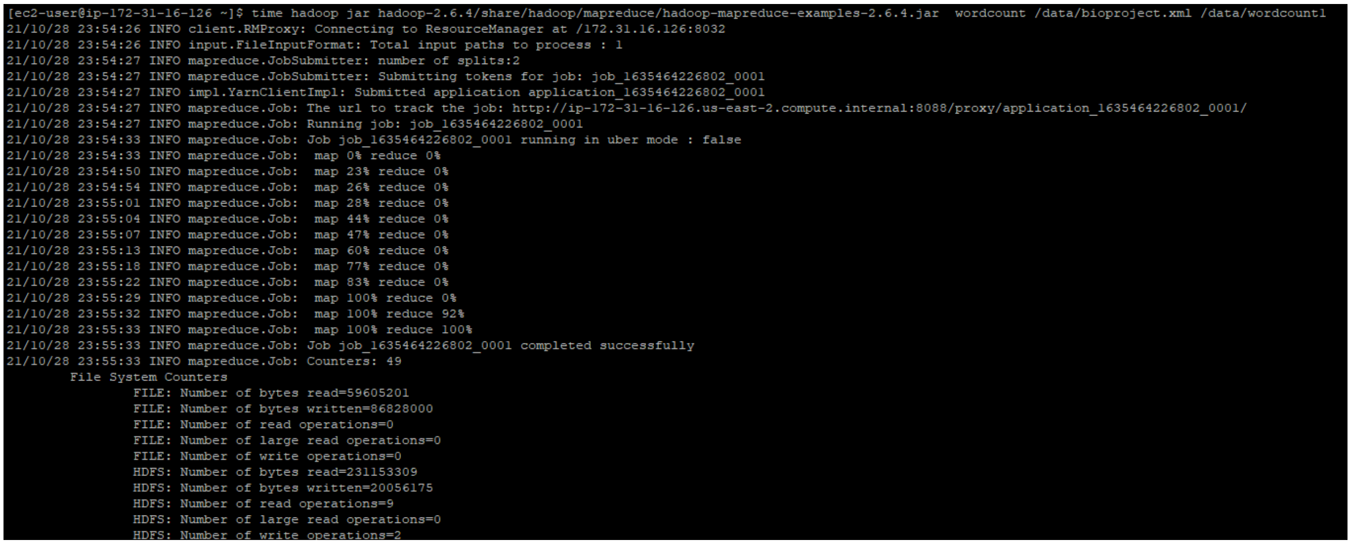
Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
ip-172-31-22-252.us-east-2.compute.internal (172.31.22.252:50010)	0	In Service	7.99 GB	4 KB	2.39 GB	5.6 GB	0	4 KB (0%)	0	2.6.4
ip-172-31-23-99.us-east-2.compute.internal (172.31.23.99:50010)	0	In Service	7.99 GB	4 KB	2.39 GB	5.6 GB	0	4 KB (0%)	0	2.6.4

Decomissioning

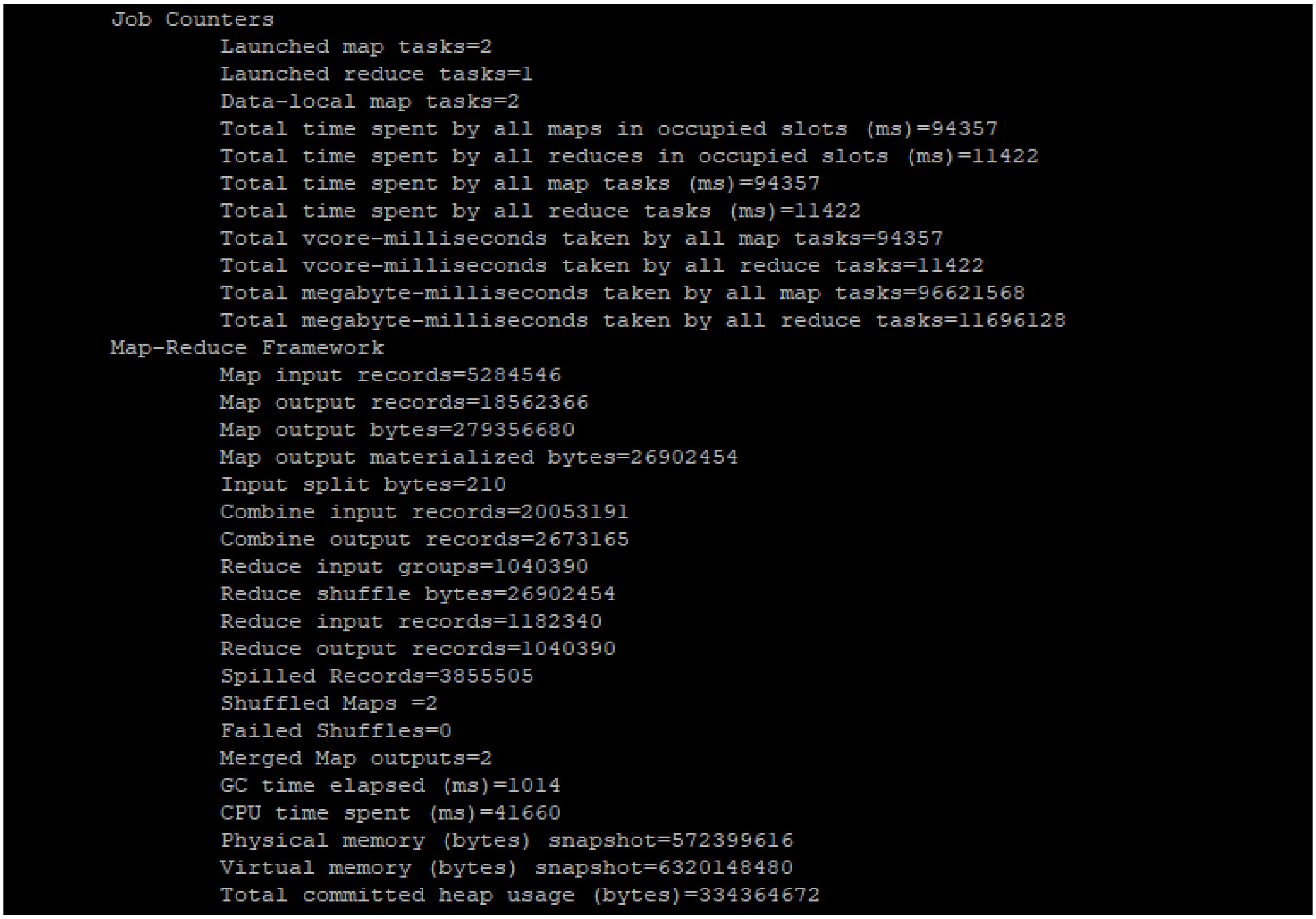
Node	Last contact	Under replicated blocks	Blocks with no live replicas	Under Replicated Blocks In files under construction

Hadoop, 2014.

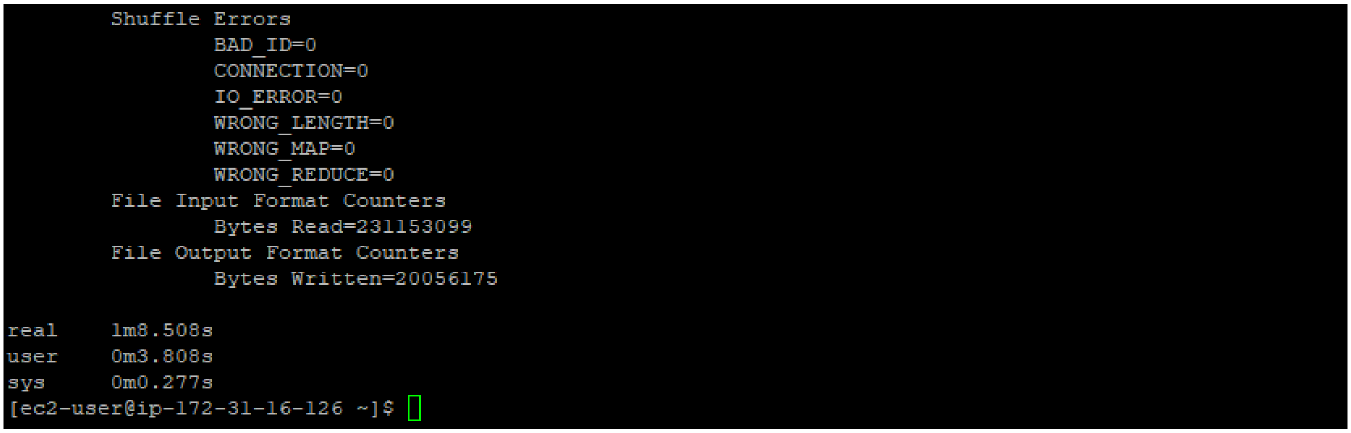
Repeat the steps for wordcount using bioproject.xml from Assignment 2 and submit screenshots of running it.



```
[ec2-user@ip-172-31-16-126 ~]$ time hadoop jar hadoop-2.6.4/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.4.jar  wordcount /data/bioproject.xml /data/wordcount1
21/10/28 23:54:26 INFO client.RMProxy: Connecting to ResourceManager at /172.31.16.126:8032
21/10/28 23:54:26 INFO input.FileInputFormat: Total input paths to process : 1
21/10/28 23:54:27 INFO mapreduce.JobSubmitter: number of splits:2
21/10/28 23:54:27 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1635464226802_0001
21/10/28 23:54:27 INFO impl.YarnClientImpl: Submitted application application_1635464226802_0001
21/10/28 23:54:27 INFO mapreduce.Job: The url to track the job: http://ip-172-31-16-126.us-east-2.compute.internal:8088/proxy/application_1635464226802_0001/
21/10/28 23:54:27 INFO mapreduce.Job: Running job: job_1635464226802_0001
21/10/28 23:54:33 INFO mapreduce.Job: Job job_1635464226802_0001 running in uber mode : false
21/10/28 23:54:33 INFO mapreduce.Job:  map 0% reduce 0%
21/10/28 23:54:50 INFO mapreduce.Job:  map 23% reduce 0%
21/10/28 23:54:54 INFO mapreduce.Job:  map 26% reduce 0%
21/10/28 23:55:01 INFO mapreduce.Job:  map 28% reduce 0%
21/10/28 23:55:04 INFO mapreduce.Job:  map 44% reduce 0%
21/10/28 23:55:07 INFO mapreduce.Job:  map 47% reduce 0%
21/10/28 23:55:13 INFO mapreduce.Job:  map 60% reduce 0%
21/10/28 23:55:18 INFO mapreduce.Job:  map 77% reduce 0%
21/10/28 23:55:22 INFO mapreduce.Job:  map 83% reduce 0%
21/10/28 23:55:29 INFO mapreduce.Job:  map 100% reduce 0%
21/10/28 23:55:32 INFO mapreduce.Job:  map 100% reduce 92%
21/10/28 23:55:33 INFO mapreduce.Job:  map 100% reduce 100%
21/10/28 23:55:33 INFO mapreduce.Job: Job job_1635464226802_0001 completed successfully
21/10/28 23:55:33 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=59605201
                FILE: Number of bytes written=86828000
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=231153309
                HDFS: Number of bytes written=20056175
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
```



```
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=94357
                Total time spent by all reduces in occupied slots (ms)=11422
                Total time spent by all map tasks (ms)=94357
                Total time spent by all reduce tasks (ms)=11422
                Total vcore-milliseconds taken by all map tasks=94357
                Total vcore-milliseconds taken by all reduce tasks=11422
                Total megabyte-milliseconds taken by all map tasks=96621568
                Total megabyte-milliseconds taken by all reduce tasks=11696128
        Map-Reduce Framework
                Map input records=5284546
                Map output records=18562366
                Map output bytes=279356680
                Map output materialized bytes=26902454
                Input split bytes=210
                Combine input records=20053191
                Combine output records=2673165
                Reduce input groups=1040390
                Reduce shuffle bytes=26902454
                Reduce input records=1182340
                Reduce output records=1040390
                Spilled Records=3855505
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=1014
                CPU time spent (ms)=41660
                Physical memory (bytes) snapshot=572399616
                Virtual memory (bytes) snapshot=6320148480
                Total committed heap usage (bytes)=334364672
```



```
        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=231153099
        File Output Format Counters
                Bytes Written=20056175

real    1m8.508s
user    0m3.808s
sys     0m0.277s
[ec2-user@ip-172-31-16-126 ~]$
```

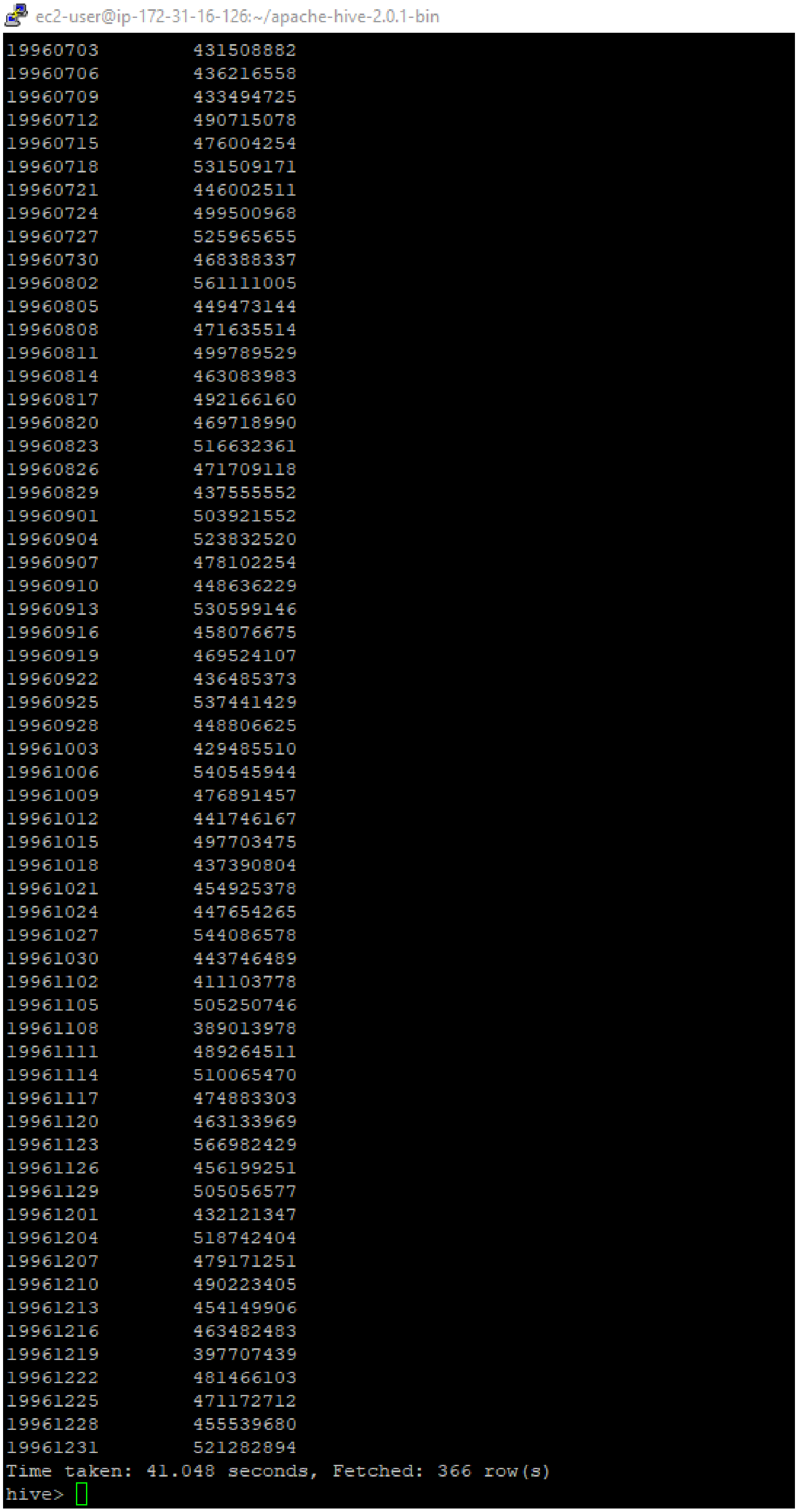
Submit a short paragraph with a discussion about how the results compare (faster? slower? How much faster/slower?)

The word count on the cluster runs faster than the one in assignment 2, which we ran on a single cluster. There is a difference of 4 mins in the run time.

Part 2: Hive

1) Run the following query in Hive and report the time it takes to execute:

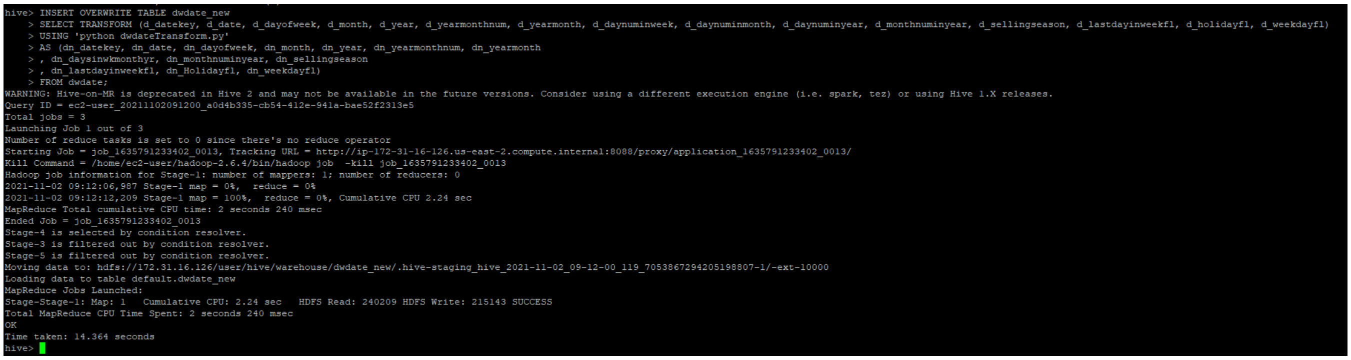
```
select lo_orderdate, sum(lo_extendedprice) as revenue
from lineorder, dwdate
where lo_orderdate = d_datekey
  and d_year = 1996
  and lo_discount between 4 and 6
  and lo_quantity < 22
GROUP BY lo_orderdate;

It takes 41.048 seconds to execute as indicated below:
```



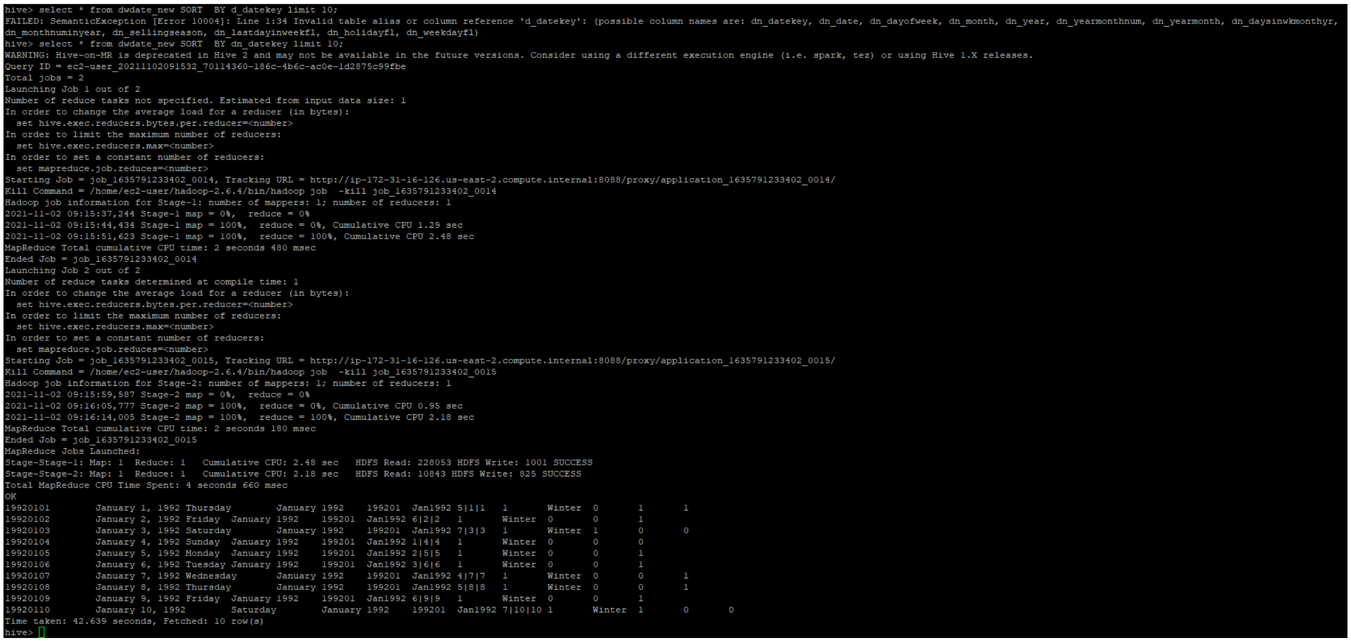
ec2-user@ip-172-31-16-126:~/apache-hive-2.0.1-bin

```
19960703        431508882
19960706        436216558
19960709        433494725
19960712        490715078
19960715        476004254
19960718        531509171
19960721        446002511
19960724        499500968
19960727        525965655
19960730        468388337
19960802        561111005
19960805        449473144
19960808        471635514
19960811        499789529
19960814        463083983
19960817        492166160
19960820        469718990
19960823        516632361
19960826        471709118
19960829        437555552
19960901        503921552
19960904        523832520
19960907        478102254
19960910        448636229
19960913        530599146
19960916        458076675
19960919        469524107
19960922        436485373
19960925        537441429
19960928        448806625
19961003        429485510
19961006        540545944
19961009        476891457
19961012        441746167
19961015        497703475
19961018        437390804
19961021        454925378
19961024        447654265
19961027        544086578
19961030        443746489
19961102        411103778
19961105        505250746
19961108        389013978
19961111        489264511
19961114        510065470
19961117        474883303
19961120        463133969
19961123        566982429
19961126        456199251
19961129        505056577
19961201        432121347
19961204        518742404
19961207        479171251
19961210        490223405
19961213        454149906
19961216        463482483
19961219        397707439
19961222        481466103
19961225        471172712
19961228        455539680
19961231        521282894
Time taken: 41.048 seconds, Fetched: 366 row(s)
hive> 
```

2) Perform the following transform operation using SELECT TRANSFORM on the dwdate table by creating a new table. The new dwdate table will combine d_daynuminweek, d_daynuminmonth, and d_daynuminyear into a single column in the new table using a delimiter of your choice. You should also eliminate the following 2 columns: d_lastdayinmonthfl and d_weeknuminyear. The final table will have fewer columns than the original table because you merge 3 columns into 1 and remove 2 columns.



```
hive> INSERT OVERWRITE TABLE dwdate_new
    > SELECT TRANSFORM (d_datekey, d_date, d_dayofweek, d_month, d_year, d_yearmonthnum, d_yearmonth, d_daynuminweek, d_daynuminmonth, d_daynuminyear, d_monthnuminyear, d_sellingseason, d_lastdayinweekfl, d_holidayfl, d_weekdayfl)
    > USING 'python dwdateTransform.py'
    > AS (dn_datekey, dn_date, dn_dayofweek, dn_month, dn_year, dn_yearmonthnum, dn_yearmonth
    > , dn_daysinwkmonthyr, dn_monthnuminyear, dn_sellingseason
    > , dn_lastdayinweekfl, dn_Holidayfl, dn_weekdayfl)
    > FROM dwdate;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = ec2-user_20211102091200_a0d4b335-cb54-412e-941a-bae52f2313e5
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1635791233402_0013, Tracking URL = http://ip-172-31-16-126.us-east-2.compute.internal:8088/proxy/application_1635791233402_0013/
Kill Command = /home/ec2-user/hadoop-2.6.4/bin/hadoop job  -kill job_1635791233402_0013
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2021-11-02 09:12:06,987 Stage-1 map = 0%,  reduce = 0%
2021-11-02 09:12:12,209 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.24 sec
MapReduce Total cumulative CPU time: 2 seconds 240 msec
Ended Job = job_1635791233402_0013
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://172.31.16.126/user/hive/warehouse/dwdate_new/.hive-staging_hive_2021-11-02_09-12-00_119_7053867294205198807-1/-ext-10000
Loading data to table default.dwdate_new
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 2.24 sec   HDFS Read: 240209 HDFS Write: 215143 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 240 msec
OK
Time taken: 14.364 seconds
hive>
```



```
hive> select * from dwdate_new SORT  BY d_datekey limit 10;
FAILED: SemanticException [Error 10004]: Line 1:34 Invalid table alias or column reference 'd_datekey': (possible column names are: dn_datekey, dn_date, dn_dayofweek, dn_month, dn_year, dn_yearmonthnum, dn_yearmonth, dn_daysinwkmonthyr,
dn_monthnuminyear, dn_sellingseason, dn_lastdayinweekfl, dn_holidayfl, dn_weekdayfl)
hive> select * from dwdate_new SORT  BY dn_datekey limit 10;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = ec2-user_20211102091532_70114360-186c-4b6c-ac0e-1d2875c99fbe
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1635791233402_0014, Tracking URL = http://ip-172-31-16-126.us-east-2.compute.internal:8088/proxy/application_1635791233402_0014/
Kill Command = /home/ec2-user/hadoop-2.6.4/bin/hadoop job  -kill job_1635791233402_0014
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-11-02 09:15:37,244 Stage-1 map = 0%,  reduce = 0%
2021-11-02 09:15:44,434 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.29 sec
2021-11-02 09:15:51,623 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.48 sec
MapReduce Total cumulative CPU time: 2 seconds 480 msec
Ended Job = job_1635791233402_0014
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1635791233402_0015, Tracking URL = http://ip-172-31-16-126.us-east-2.compute.internal:8088/proxy/application_1635791233402_0015/
Kill Command = /home/ec2-user/hadoop-2.6.4/bin/hadoop job  -kill job_1635791233402_0015
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2021-11-02 09:15:59,587 Stage-2 map = 0%,  reduce = 0%
2021-11-02 09:16:05,777 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 0.95 sec
2021-11-02 09:16:14,005 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 2.18 sec
MapReduce Total cumulative CPU time: 2 seconds 180 msec
Ended Job = job_1635791233402_0015
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.48 sec   HDFS Read: 228053 HDFS Write: 1001 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.18 sec   HDFS Read: 10843 HDFS Write: 825 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 660 msec
OK
19920101        January 1, 1992 Thursday        January 1992    199201  Jan1992 5|1|1   1       Winter  0       1       1
19920102        January 2, 1992 Friday  January 1992    199201  Jan1992 6|2|2   1       Winter  0       0       1
19920103        January 3, 1992 Saturday        January 1992    199201  Jan1992 7|3|3   1       Winter  1       0       0
19920104        January 4, 1992 Sunday  January 1992    199201  Jan1992 1|4|4   1       Winter  0       0       0
19920105        January 5, 1992 Monday  January 1992    199201  Jan1992 2|5|5   1       Winter  0       0       1
19920106        January 6, 1992 Tuesday January 1992    199201  Jan1992 3|6|6   1       Winter  0       0       1
19920107        January 7, 1992 Wednesday       January 1992    199201  Jan1992 4|7|7   1       Winter  0       0       1
19920108        January 8, 1992 Thursday        January 1992    199201  Jan1992 5|8|8   1       Winter  0       0       1
19920109        January 9, 1992 Friday  January 1992    199201  Jan1992 6|9|9   1       Winter  0       0       1
19920110        January 10, 1992        Saturday        January 1992    199201  Jan1992 7|10|10 1       Winter  1       0       0
Time taken: 42.639 seconds, Fetched: 10 row(s)
hive>
```

Hive scripts:

```
create table dwdate_new (
  dn_datekey            int,
  dn_date               varchar(19),
  dn_dayofweek          varchar(10),
  dn_month              varchar(10),
  dn_year               int,
  dn_yearmonthnum       int,
  dn_yearmonth           varchar(8),
  dn_daysinwkmonthyr varchar(50),
  dn_monthnuminyear int,
  dn_sellingseason      varchar(13),
  dn_lastdayinweekfl    varchar(1),
  dn_Holidayfl          varchar(1),
  dn_weekdayfl          varchar(1)
)
ROW FORMAT DELIMITED FIELDS
TERMINATED BY '\t' STORED AS TEXTFILE;

add FILE dwdateTransform.py;

INSERT OVERWRITE TABLE dwdate_new
SELECT TRANSFORM (d_datekey, d_date, d_dayofweek, d_month, d_year, d_yearmonthnum
, d_yearmonth, d_daynuminweek, d_daynuminmonth, d_daynuminyear, d_monthnuminyear,
 d_sellingseason, d_lastdayinweekfl, d_holidayfl, d_weekdayfl)
USING 'python dwdateTransform.py'
AS (dn_datekey, dn_date, dn_dayofweek, dn_month, dn_year, dn_yearmonthnum, dn_yea
rmonth
, dn_daysinwkmonthyr, dn_monthnuminyear, dn_sellingseason
, dn_lastdayinweekfl, dn_Holidayfl, dn_weekdayfl)
FROM dwdate;
```

dwdateTransform.py

```
#!/usr/bin/python
# Author: Ronaldlee Ejalu
# CSC 555 Mining Big Data
# MidTerm Exam
# dwdateTransform.py
import sys

for lines in sys.stdin:
        columnList = []
        strippedLines = lines.strip()         # remove any white spaces
        lines = strippedLines.split('\t')  # split the string to create a list of
 words
        d_datekey  = lines[0]
        d_date  = lines[1]
        d_dayofweek = lines[2]
        d_month = lines[3]
        d_year  = lines[4]
        d_yearmonthnum = lines[5]
        d_yearmonth =  lines[6]
        d_daynuminweek  = lines[7]
        d_daynuminmonth = lines[8]
        d_daynuminyear  = lines[9]
        d_daysinwkmonthyr  = d_daynuminweek  + '|' + d_daynuminmonth  + '|' + d_d
aynuminyear
        d_monthnuminyear = lines[10]
        d_sellingseason = lines[11]
        d_lastdayinweekfl = lines[12]
        d_holidayfl = lines[13]
        d_weekdayfl = lines[14]


        print(d_datekey + '\t' + d_date + '\t' + d_dayofweek + '\t' + d_month + '
\t' + d_year + '\t' + d_yearmonthnum + '\t' + d_yearmonth + '\t' + d_daysinwkmont
hyr + '\t' +  d_monthnuminyear + '\t' + d_sellingseason + '\t' + d_lastdayinweekf
l + '\t' + d_holidayfl + '\t' + d_weekdayfl)
```

Part 3: Pig

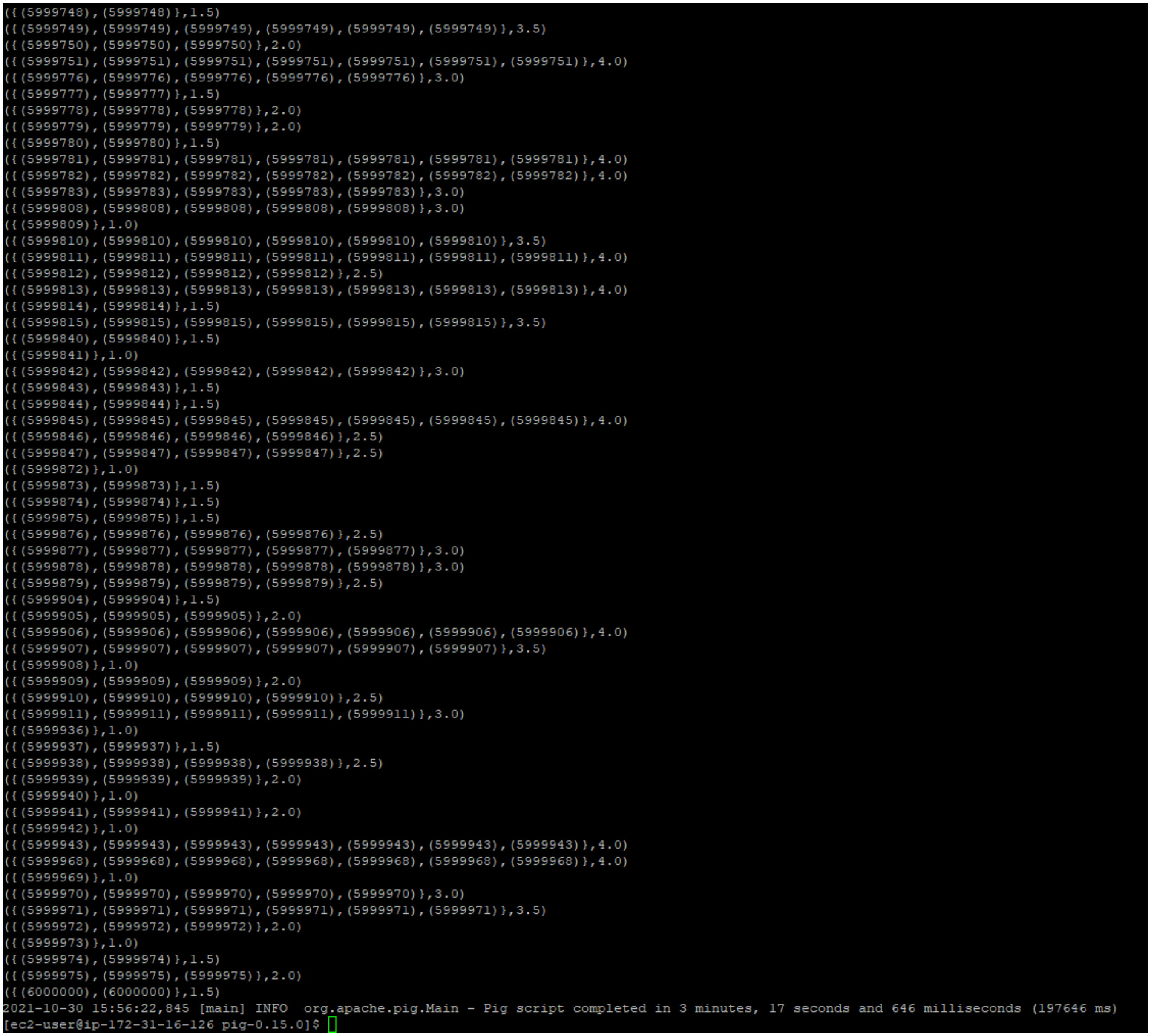
Convert and load the data into Pig, implementing and timing the following queries:

```
SELECT lo_discount, AVG(lo_extendedprice)
FROM lineorder
GROUP BY lo_discount;
```

```
SELECT lo_quantity, SUM(lo_revenue)
FROM lineorder
WHERE lo_discount > 8 AND lo_quantity > 33
GROUP BY lo_quantity;
```

One easy way to time Pig is as follows: put your sequence of pig commands, including LOAD, into a text file and then run, from command line in pig directory (e.g., [ec2-user@ip-172-31-6-39 pig-0.15.0]$), **bin/pig -f pig_script.pig** (which will report how long the pig script took to run).

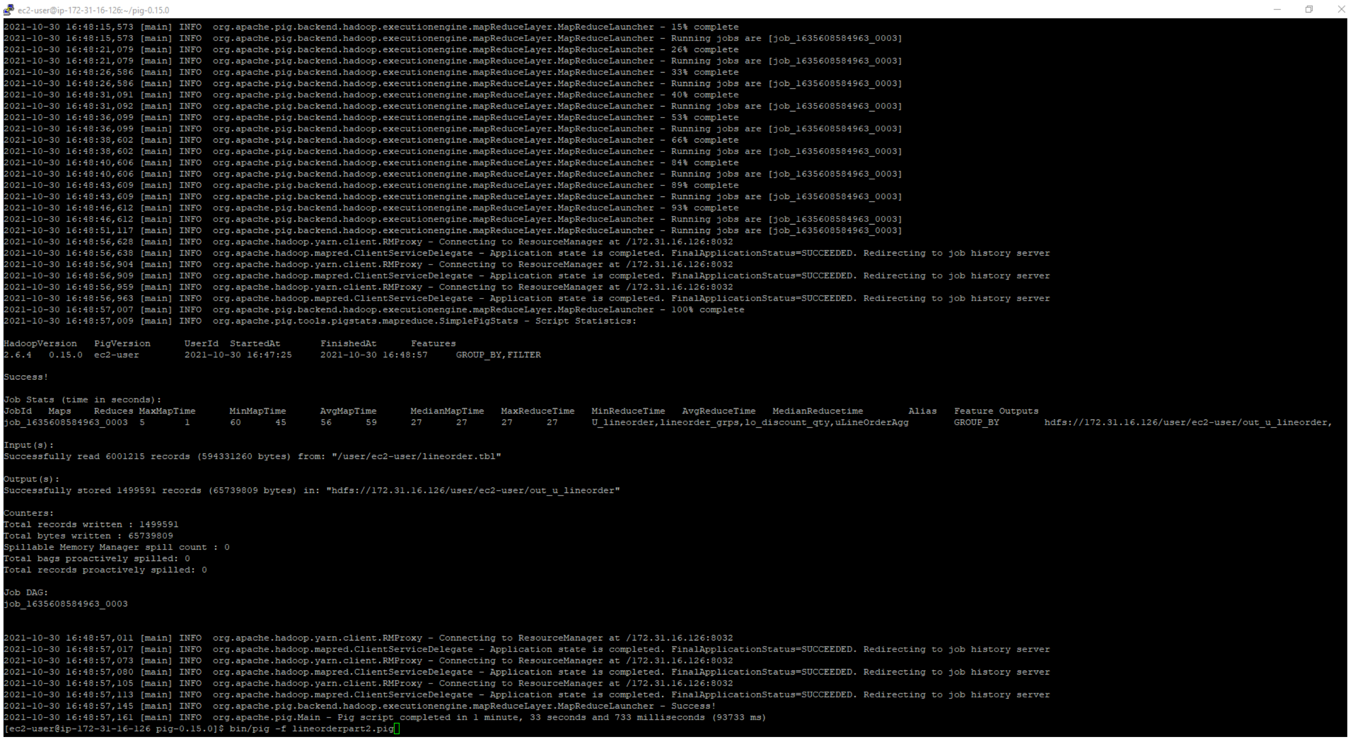
My first script completed in 3 minutes, 17 seconds and 646 milliseconds as indicated below in the screen shot.



```
({(5999748),(5999748)},1.5)
({(5999749),(5999749),(5999749),(5999749),(5999749),(5999749)},3.5)
({(5999750),(5999750),(5999750)},2.0)
({(5999751),(5999751),(5999751),(5999751),(5999751),(5999751),(5999751)},4.0)
({(5999776),(5999776),(5999776),(5999776),(5999776)},3.0)
({(5999777),(5999777)},1.5)
({(5999778),(5999778),(5999778)},2.0)
({(5999779),(5999779),(5999779)},2.0)
({(5999780),(5999780)},1.5)
({(5999781),(5999781),(5999781),(5999781),(5999781),(5999781),(5999781)},4.0)
({(5999782),(5999782),(5999782),(5999782),(5999782),(5999782),(5999782)},4.0)
({(5999783),(5999783),(5999783),(5999783),(5999783)},3.0)
({(5999808),(5999808),(5999808),(5999808),(5999808)},3.0)
({(5999809)},1.0)
({(5999810),(5999810),(5999810),(5999810),(5999810),(5999810)},3.5)
({(5999811),(5999811),(5999811),(5999811),(5999811),(5999811),(5999811)},4.0)
({(5999812),(5999812),(5999812),(5999812)},2.5)
({(5999813),(5999813),(5999813),(5999813),(5999813),(5999813),(5999813)},4.0)
({(5999814),(5999814)},1.5)
({(5999815),(5999815),(5999815),(5999815),(5999815),(5999815)},3.5)
({(5999840),(5999840)},1.5)
({(5999841)},1.0)
({(5999842),(5999842),(5999842),(5999842),(5999842)},3.0)
({(5999843),(5999843)},1.5)
({(5999844),(5999844)},1.5)
({(5999845),(5999845),(5999845),(5999845),(5999845),(5999845),(5999845)},4.0)
({(5999846),(5999846),(5999846),(5999846)},2.5)
({(5999847),(5999847),(5999847),(5999847)},2.5)
({(5999872)},1.0)
({(5999873),(5999873)},1.5)
({(5999874),(5999874)},1.5)
({(5999875),(5999875)},1.5)
({(5999876),(5999876),(5999876),(5999876)},2.5)
({(5999877),(5999877),(5999877),(5999877),(5999877)},3.0)
({(5999878),(5999878),(5999878),(5999878),(5999878)},3.0)
({(5999879),(5999879),(5999879),(5999879)},2.5)
({(5999904),(5999904)},1.5)
({(5999905),(5999905),(5999905)},2.0)
({(5999906),(5999906),(5999906),(5999906),(5999906),(5999906),(5999906)},4.0)
({(5999907),(5999907),(5999907),(5999907),(5999907),(5999907)},3.5)
({(5999908)},1.0)
({(5999909),(5999909),(5999909)},2.0)
({(5999910),(5999910),(5999910),(5999910)},2.5)
({(5999911),(5999911),(5999911),(5999911),(5999911)},3.0)
({(5999936)},1.0)
({(5999937),(5999937)},1.5)
({(5999938),(5999938),(5999938),(5999938)},2.5)
({(5999939),(5999939),(5999939)},2.0)
({(5999940)},1.0)
({(5999941),(5999941),(5999941)},2.0)
({(5999942)},1.0)
({(5999943),(5999943),(5999943),(5999943),(5999943),(5999943),(5999943)},4.0)
({(5999968),(5999968),(5999968),(5999968),(5999968),(5999968),(5999968)},4.0)
({(5999969)},1.0)
({(5999970),(5999970),(5999970),(5999970),(5999970)},3.0)
({(5999971),(5999971),(5999971),(5999971),(5999971),(5999971)},3.5)
({(5999972),(5999972),(5999972)},2.0)
({(5999973)},1.0)
({(5999974),(5999974)},1.5)
({(5999975),(5999975),(5999975)},2.0)
({(6000000),(6000000)},1.5)
2021-10-30 15:56:22,845 [main] INFO  org.apache.pig.Main - Pig script completed in 3 minutes, 17 seconds and 646 milliseconds (197646 ms)
[ec2-user@ip-172-31-16-126 pig-0.15.0]$ 
```

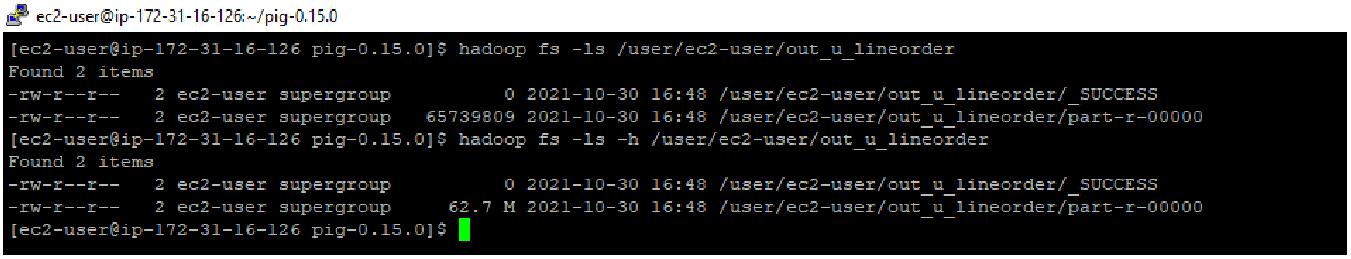
Store the results of the 2nd Pig query into HDFS and report the size of the output.

The second Pig script (lineorderpart2.pig), completed in 1 minute, 33 seconds and 733 milliseconds as indicated below in the screen shot:



```
2021-10-30 16:48:15,573 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 15% complete
2021-10-30 16:48:15,573 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:21,079 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 26% complete
2021-10-30 16:48:21,079 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:26,586 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 33% complete
2021-10-30 16:48:26,586 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:31,091 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 40% complete
2021-10-30 16:48:31,092 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:36,099 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 53% complete
2021-10-30 16:48:36,099 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:38,602 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 66% complete
2021-10-30 16:48:38,602 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:40,606 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 84% complete
2021-10-30 16:48:40,606 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:43,609 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 89% complete
2021-10-30 16:48:43,609 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:46,612 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 93% complete
2021-10-30 16:48:46,612 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:51,117 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Running jobs are [job_1635608584963_0003]
2021-10-30 16:48:56,628 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /172.31.16.126:8032
2021-10-30 16:48:56,638 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2021-10-30 16:48:56,904 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /172.31.16.126:8032
2021-10-30 16:48:56,909 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2021-10-30 16:48:56,959 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /172.31.16.126:8032
2021-10-30 16:48:56,963 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2021-10-30 16:48:57,007 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2021-10-30 16:48:57,009 [main] INFO  org.apache.pig.tools.pigstats.mapreduce.SimplePigStats - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.4   0.15.0  ec2-user        2021-10-30 16:47:25     2021-10-30 16:48:57     GROUP_BY,FILTER

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTime      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime   MedianReducetime        Alias   Feature Outputs
job_1635608584963_0003  5       1       60      45      56      59      27      27      27      27      U_lineorder,lineorder_grps,lo_discount_qty,uLineOrderAgg        GROUP_BY        hdfs://172.31.16.126/user/ec2-user/out_u_lineorder,

Input(s):
Successfully read 6001215 records (594331260 bytes) from: "/user/ec2-user/lineorder.tbl"

Output(s):
Successfully stored 1499591 records (65739809 bytes) in: "hdfs://172.31.16.126/user/ec2-user/out_u_lineorder"

Counters:
Total records written : 1499591
Total bytes written : 65739809
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1635608584963_0003


2021-10-30 16:48:57,011 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /172.31.16.126:8032
2021-10-30 16:48:57,017 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2021-10-30 16:48:57,073 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /172.31.16.126:8032
2021-10-30 16:48:57,080 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2021-10-30 16:48:57,105 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /172.31.16.126:8032
2021-10-30 16:48:57,113 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalApplicationStatus=SUCCEEDED. Redirecting to job history server
2021-10-30 16:48:57,145 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2021-10-30 16:48:57,161 [main] INFO  org.apache.pig.Main - Pig script completed in 1 minute, 33 seconds and 733 milliseconds (93733 ms)
[ec2-user@ip-172-31-16-126 pig-0.15.0]$ bin/pig -f lineorderpart2.pig
```

The size of the output file is: 62.7 M as shown below in the screen shot:



```
[ec2-user@ip-172-31-16-126 pig-0.15.0]$ hadoop fs -ls /user/ec2-user/out_u_lineorder
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2021-10-30 16:48 /user/ec2-user/out_u_lineorder/_SUCCESS
-rw-r--r--   2 ec2-user supergroup   65739809 2021-10-30 16:48 /user/ec2-user/out_u_lineorder/part-r-00000
[ec2-user@ip-172-31-16-126 pig-0.15.0]$ hadoop fs -ls -h /user/ec2-user/out_u_lineorder
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2021-10-30 16:48 /user/ec2-user/out_u_lineorder/_SUCCESS
-rw-r--r--   2 ec2-user supergroup     62.7 M 2021-10-30 16:48 /user/ec2-user/out_u_lineorder/part-r-00000
[ec2-user@ip-172-31-16-126 pig-0.15.0]$
```

PigScripts:

```
-- Author: Ronaldlee Ejalu
-- CSC 555 Mining Big Data
-- MidTerm Exam
-- Script 0ne

/* SELECT lo_discount, AVG(lo_extendedprice)
FROM lineorder
GROUP BY lo_discount;
*/

/*
To run this script at the command line shell:

bin/pig -f lineorderpart1.pig
*/


Ulineorder = LOAD '/user/ec2-user/lineorder.tbl' USING PigStorage('|')
 AS (lo_discount:int, lo_extendedprice:int);
 lineorder_groups = GROUP Ulineorder BY lo_discount;
 lineorder_avgs = FOREACH lineorder_groups GENERATE Ulineorder.lo_discount,
AVG(Ulineorder.lo_extendedprice);
 DUMP lineorder_avgs;
```

```
-- Author: Ronaldlee Ejalu
-- CSC 555 Mining Big Data
-- MidTerm Exam
-- Script two


/*
SELECT lo_quantity, SUM(lo_revenue)
FROM lineorder
WHERE lo_discount > 8 AND lo_quantity > 33
GROUP BY lo_quantity;
*/

/*
To run this script at the command line shell:

bin/pig -f lineorderpart2.pig
*/

U_lineorder = LOAD '/user/ec2-user/lineorder.tbl' USING PigStorage('|')
 AS (lo_quantity:int, lo_revenue:int, lo_discount:int);
 lo_discount_qty = FILTER U_lineorder BY lo_discount > 8 AND lo_quantity > 33;
 lineorder_grps = GROUP lo_discount_qty BY lo_quantity;
 uLineOrderAgg = FOREACH lineorder_grps GENERATE lo_discount_qty.lo_quantity, SUM
(lo_discount_qty.lo_revenue);
 STORE uLineOrderAgg INTO 'out_u_lineorder' USING PigStorage('|');
--DUMP uLineOrderAgg;
```

Part 4: Hadoop Streaming

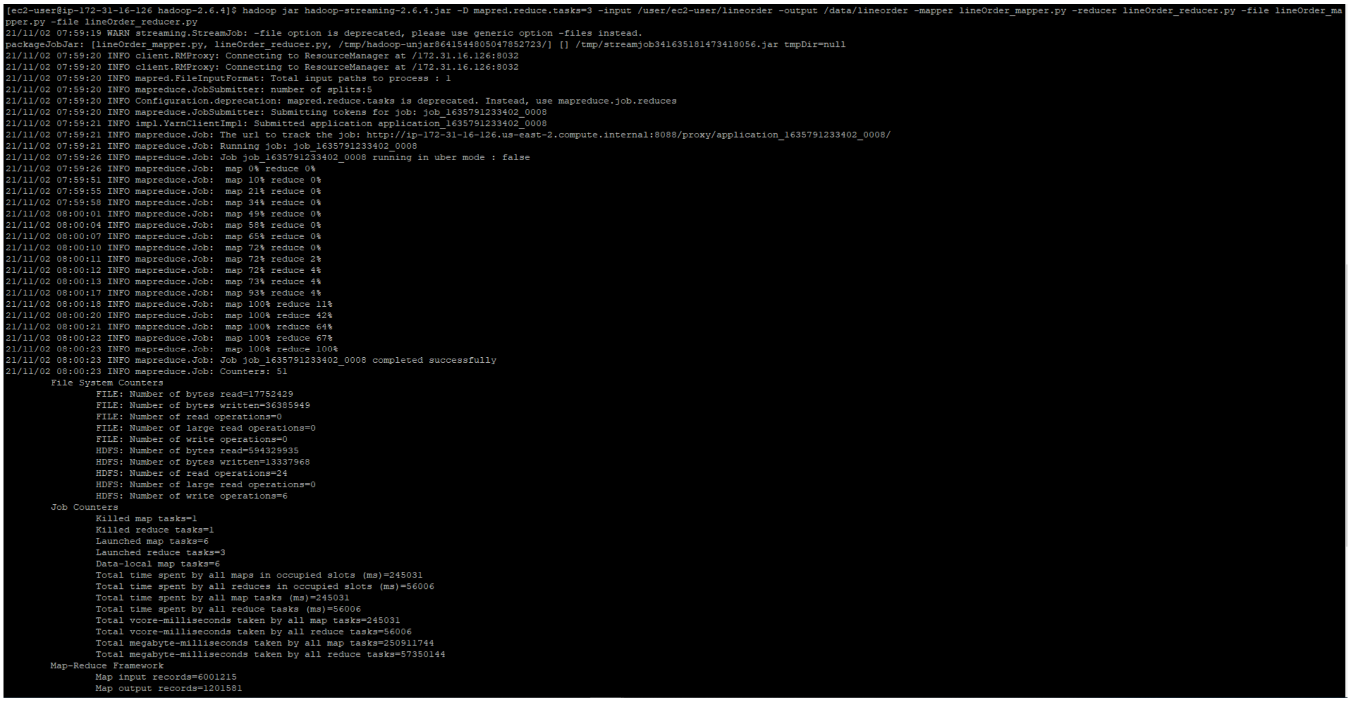
Implement, run and time the following query using Hadoop streaming with python.

```
SELECT lo_quantity, MAX(lo_revenue)
FROM (SELECT lo_revenue, MAX(lo_quantity) as lo_quantity,
                         MAX(lo_discount) as lo_discount
      FROM lineorder
      WHERE lo_orderpriority LIKE '%URGENT'
      GROUP BY lo_revenue)
WHERE lo_discount BETWEEN 4 AND 8
GROUP BY lo_quantity;
```

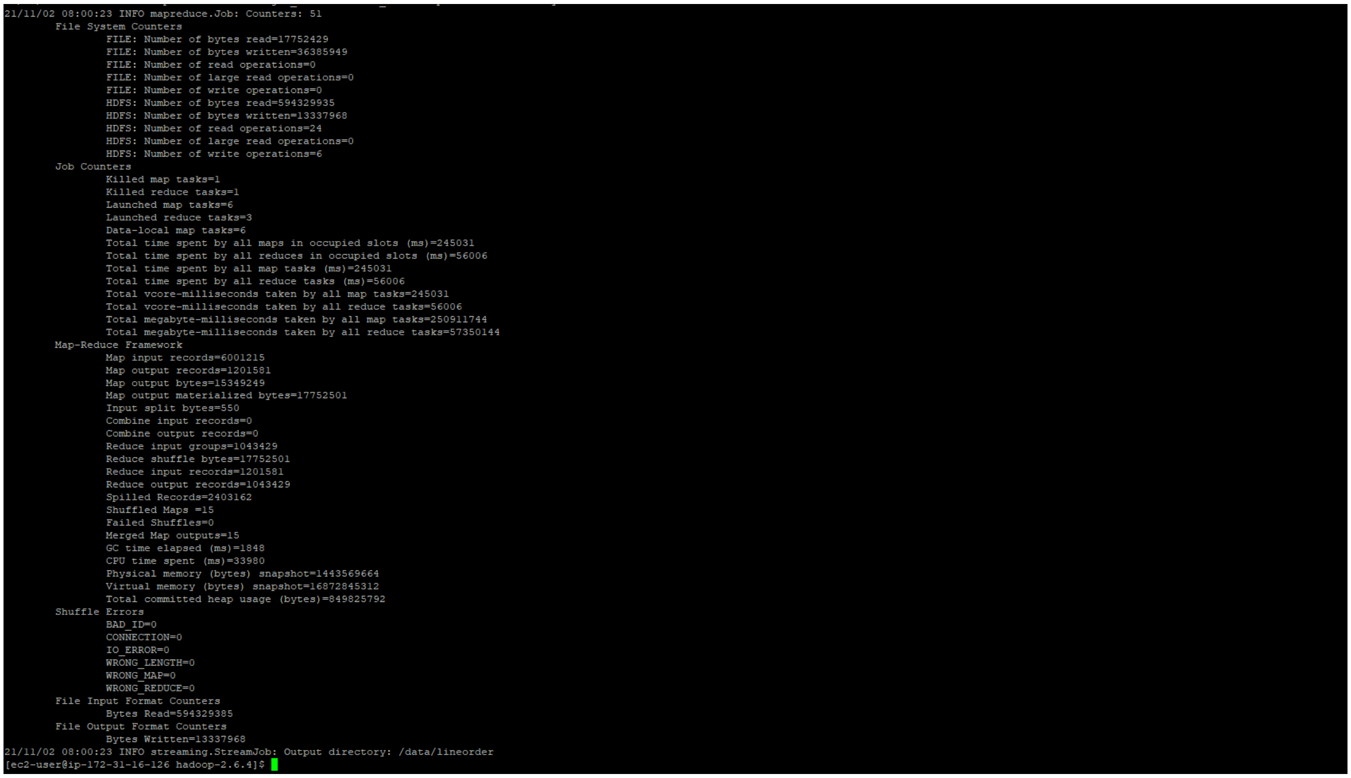
This requires running two different map reduce jobs. First, you would write a job that executes the subquery and produces an output in HDFS. Then you would write a second job that uses output of the first job as the input.

Don’t forget to submit your python code, and the command line you used to run Hadoop streaming jobs.

The first job command lines:



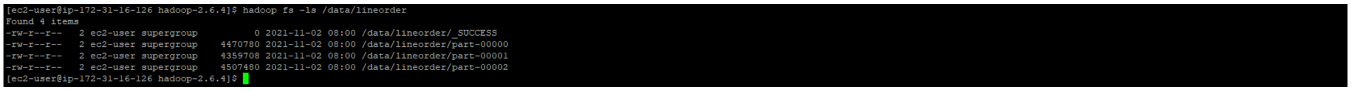
```
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$ hadoop jar hadoop-streaming-2.6.4.jar -D mapred.reduce.tasks=3 -input /user/ec2-user/lineorder -output /data/lineorder -mapper lineOrder_mapper.py -reducer lineOrder_reducer.py -file lineOrder_ma
pper.py -file lineOrder_reducer.py
21/11/02 07:59:19 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [lineOrder_mapper.py, lineOrder_reducer.py, /tmp/hadoop-unjar8641544805047852723/] [] /tmp/streamjob3416351814734l8056.jar tmpDir=null
21/11/02 07:59:20 INFO client.RMProxy: Connecting to ResourceManager at /172.31.16.126:8032
21/11/02 07:59:20 INFO client.RMProxy: Connecting to ResourceManager at /172.31.16.126:8032
21/11/02 07:59:20 INFO mapred.FileInputFormat: Total input paths to process : 1
21/11/02 07:59:20 INFO mapreduce.JobSubmitter: number of splits:5
21/11/02 07:59:20 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
21/11/02 07:59:20 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1635791233402_0008
21/11/02 07:59:21 INFO impl.YarnClientImpl: Submitted application application_1635791233402_0008
21/11/02 07:59:21 INFO mapreduce.Job: The url to track the job: http://ip-172-31-16-126.us-east-2.compute.internal:8088/proxy/application_1635791233402_0008/
21/11/02 07:59:21 INFO mapreduce.Job: Running job: job_1635791233402_0008
21/11/02 07:59:26 INFO mapreduce.Job: Job job_1635791233402_0008 running in uber mode : false
21/11/02 07:59:26 INFO mapreduce.Job:  map 0% reduce 0%
21/11/02 07:59:51 INFO mapreduce.Job:  map 10% reduce 0%
21/11/02 07:59:55 INFO mapreduce.Job:  map 21% reduce 0%
21/11/02 07:59:58 INFO mapreduce.Job:  map 34% reduce 0%
21/11/02 08:00:01 INFO mapreduce.Job:  map 49% reduce 0%
21/11/02 08:00:04 INFO mapreduce.Job:  map 58% reduce 0%
21/11/02 08:00:07 INFO mapreduce.Job:  map 65% reduce 0%
21/11/02 08:00:10 INFO mapreduce.Job:  map 72% reduce 0%
21/11/02 08:00:11 INFO mapreduce.Job:  map 72% reduce 2%
21/11/02 08:00:12 INFO mapreduce.Job:  map 72% reduce 4%
21/11/02 08:00:13 INFO mapreduce.Job:  map 73% reduce 4%
21/11/02 08:00:17 INFO mapreduce.Job:  map 93% reduce 4%
21/11/02 08:00:18 INFO mapreduce.Job:  map 100% reduce 11%
21/11/02 08:00:20 INFO mapreduce.Job:  map 100% reduce 42%
21/11/02 08:00:21 INFO mapreduce.Job:  map 100% reduce 64%
21/11/02 08:00:22 INFO mapreduce.Job:  map 100% reduce 67%
21/11/02 08:00:23 INFO mapreduce.Job:  map 100% reduce 100%
21/11/02 08:00:23 INFO mapreduce.Job: Job job_1635791233402_0008 completed successfully
21/11/02 08:00:23 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=17752429
                FILE: Number of bytes written=36385949
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=594329935
                HDFS: Number of bytes written=13337968
                HDFS: Number of read operations=24
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=6
        Job Counters
                Killed map tasks=1
                Killed reduce tasks=1
                Launched map tasks=6
                Launched reduce tasks=3
                Data-local map tasks=6
                Total time spent by all maps in occupied slots (ms)=245031
                Total time spent by all reduces in occupied slots (ms)=56006
                Total time spent by all map tasks (ms)=245031
                Total time spent by all reduce tasks (ms)=56006
                Total vcore-milliseconds taken by all map tasks=245031
                Total vcore-milliseconds taken by all reduce tasks=56006
                Total megabyte-milliseconds taken by all map tasks=250911744
                Total megabyte-milliseconds taken by all reduce tasks=57350144
        Map-Reduce Framework
                Map input records=6001215
                Map output records=1201581
```



```
21/11/02 08:00:23 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=17752429
                FILE: Number of bytes written=36385949
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=594329935
                HDFS: Number of bytes written=13337968
                HDFS: Number of read operations=24
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=6
        Job Counters
                Killed map tasks=1
                Killed reduce tasks=1
                Launched map tasks=6
                Launched reduce tasks=3
                Data-local map tasks=6
                Total time spent by all maps in occupied slots (ms)=245031
                Total time spent by all reduces in occupied slots (ms)=56006
                Total time spent by all map tasks (ms)=245031
                Total time spent by all reduce tasks (ms)=56006
                Total vcore-milliseconds taken by all map tasks=245031
                Total vcore-milliseconds taken by all reduce tasks=56006
                Total megabyte-milliseconds taken by all map tasks=250911744
                Total megabyte-milliseconds taken by all reduce tasks=57350144
        Map-Reduce Framework
                Map input records=6001215
                Map output records=1201581
```

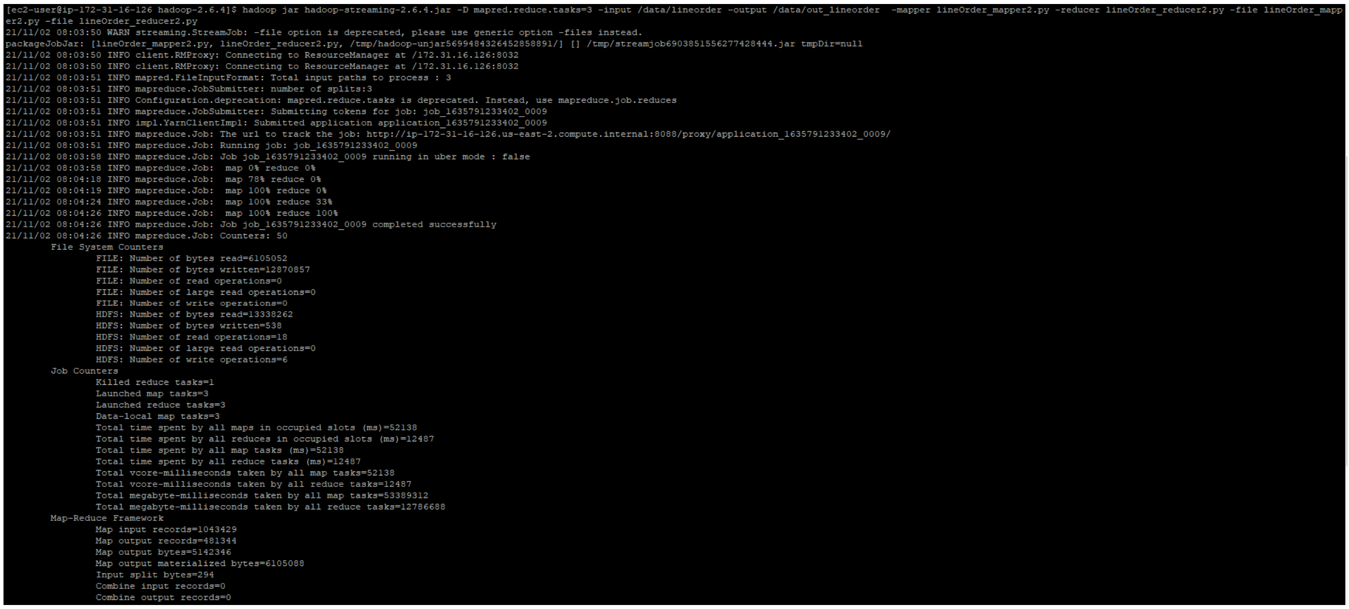
```
                Map output bytes=15349249
                Map output materialized bytes=17752501
                Input split bytes=550
                Combine input records=0
                Combine output records=0
                Reduce input groups=1043429
                Reduce shuffle bytes=17752501
                Reduce input records=1201581
                Reduce output records=1043429
                Spilled Records=2403162
                Shuffled Maps =15
                Failed Shuffles=0
                Merged Map outputs=15
                GC time elapsed (ms)=1848
                CPU time spent (ms)=33980
                Physical memory (bytes) snapshot=1443569664
                Virtual memory (bytes) snapshot=16872845312
                Total committed heap usage (bytes)=849825792
        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=594329385
        File Output Format Counters
                Bytes Written=13337968
21/11/02 08:00:23 INFO streaming.StreamJob: Output directory: /data/lineorder
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$ █
```

The output files from the first job:

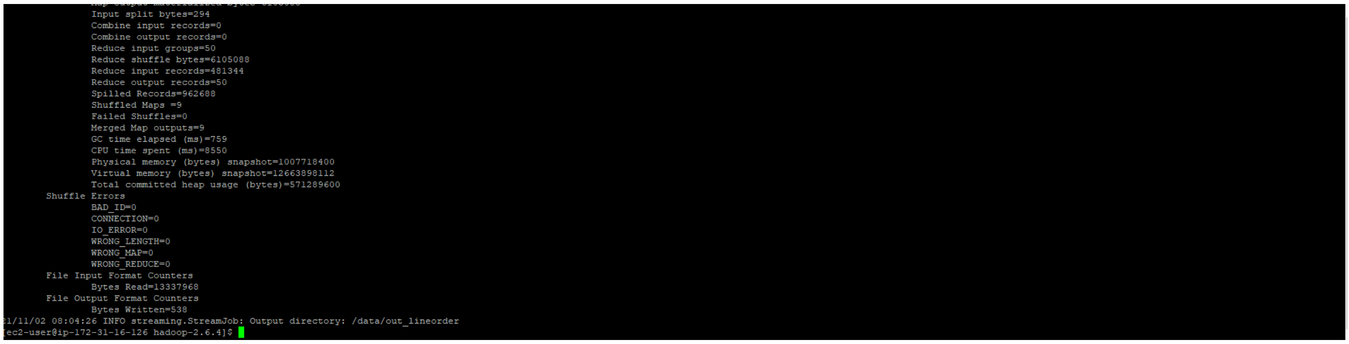


```
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$ hadoop fs -ls /data/lineorder
Found 4 items
-rw-r--r--   2 ec2-user supergroup          0 2021-11-02 08:00 /data/lineorder/_SUCCESS
-rw-r--r--   2 ec2-user supergroup    4470780 2021-11-02 08:00 /data/lineorder/part-00000
-rw-r--r--   2 ec2-user supergroup    4359708 2021-11-02 08:00 /data/lineorder/part-00001
-rw-r--r--   2 ec2-user supergroup    4507480 2021-11-02 08:00 /data/lineorder/part-00002
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$
```

The commands from the second job are as follows:

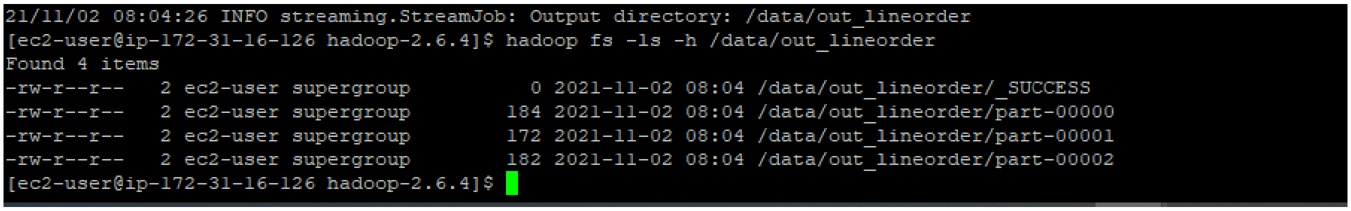


```
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$ hadoop jar hadoop-streaming-2.6.4.jar -D mapred.reduce.tasks=3 -input /data/lineorder -output /data/out_lineorder  -mapper lineOrder_mapper2.py -reducer lineOrder_reducer2.py -file lineOrder_mapp
er2.py -file lineOrder_reducer2.py
21/11/02 08:03:50 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [lineOrder_mapper2.py, lineOrder_reducer2.py, /tmp/hadoop-unjar5699484326452858891/] [] /tmp/streamjob6903851556277428444.jar tmpDir=null
21/11/02 08:03:50 INFO client.RMProxy: Connecting to ResourceManager at /172.31.16.126:8032
21/11/02 08:03:50 INFO client.RMProxy: Connecting to ResourceManager at /172.31.16.126:8032
21/11/02 08:03:51 INFO mapred.FileInputFormat: Total input paths to process : 3
21/11/02 08:03:51 INFO mapreduce.JobSubmitter: number of splits:3
21/11/02 08:03:51 INFO Configuration.deprecation: mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
21/11/02 08:03:51 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1635791233402_0009
21/11/02 08:03:51 INFO impl.YarnClientImpl: Submitted application application_1635791233402_0009
21/11/02 08:03:51 INFO mapreduce.Job: The url to track the job: http://ip-172-31-16-126.us-east-2.compute.internal:8088/proxy/application_1635791233402_0009/
21/11/02 08:03:51 INFO mapreduce.Job: Running job: job_1635791233402_0009
21/11/02 08:03:58 INFO mapreduce.Job: Job job_1635791233402_0009 running in uber mode : false
21/11/02 08:03:58 INFO mapreduce.Job:  map 0% reduce 0%
21/11/02 08:04:18 INFO mapreduce.Job:  map 78% reduce 0%
21/11/02 08:04:19 INFO mapreduce.Job:  map 100% reduce 0%
21/11/02 08:04:24 INFO mapreduce.Job:  map 100% reduce 33%
21/11/02 08:04:26 INFO mapreduce.Job:  map 100% reduce 100%
21/11/02 08:04:26 INFO mapreduce.Job: Job job_1635791233402_0009 completed successfully
21/11/02 08:04:26 INFO mapreduce.Job: Counters: 50
        File System Counters
                FILE: Number of bytes read=6105052
                FILE: Number of bytes written=12870857
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=13338262
                HDFS: Number of bytes written=538
                HDFS: Number of read operations=18
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=6
        Job Counters
                Killed reduce tasks=1
                Launched map tasks=3
                Launched reduce tasks=3
                Data-local map tasks=3
                Total time spent by all maps in occupied slots (ms)=52138
                Total time spent by all reduces in occupied slots (ms)=12487
                Total time spent by all map tasks (ms)=52138
                Total time spent by all reduce tasks (ms)=12487
                Total vcore-milliseconds taken by all map tasks=52138
                Total vcore-milliseconds taken by all reduce tasks=12487
                Total megabyte-milliseconds taken by all map tasks=53389312
                Total megabyte-milliseconds taken by all reduce tasks=12786688
        Map-Reduce Framework
                Map input records=1043429
                Map output records=481344
                Map output bytes=5142346
                Map output materialized bytes=6105088
                Input split bytes=294
                Combine input records=0
                Combine output records=0
```



```
                Input split bytes=294
                Combine input records=0
                Combine output records=0
                Reduce input groups=50
                Reduce shuffle bytes=6105088
                Reduce input records=481344
                Reduce output records=50
                Spilled Records=962688
                Shuffled Maps =9
                Failed Shuffles=0
                Merged Map outputs=9
                GC time elapsed (ms)=759
                CPU time spent (ms)=8550
                Physical memory (bytes) snapshot=1007718400
                Virtual memory (bytes) snapshot=12663898112
                Total committed heap usage (bytes)=571289600
        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=13337968
        File Output Format Counters
                Bytes Written=538
21/11/02 08:04:26 INFO streaming.StreamJob: Output directory: /data/out_lineorder
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$
```

The output files from the second job:



```
21/11/02 08:04:26 INFO streaming.StreamJob: Output directory: /data/out_lineorder
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$ hadoop fs -ls -h /data/out_lineorder
Found 4 items
-rw-r--r--   2 ec2-user supergroup          0 2021-11-02 08:04 /data/out_lineorder/_SUCCESS
-rw-r--r--   2 ec2-user supergroup        184 2021-11-02 08:04 /data/out_lineorder/part-00000
-rw-r--r--   2 ec2-user supergroup        172 2021-11-02 08:04 /data/out_lineorder/part-00001
-rw-r--r--   2 ec2-user supergroup        182 2021-11-02 08:04 /data/out_lineorder/part-00002
[ec2-user@ip-172-31-16-126 hadoop-2.6.4]$
```

Code for the first job:

lineOrder_mapper.py

```
#Ronaldlee Ejalu
#CSC 555 Big Data Mining
# lineOrder_mapper.py
#!/usr/bin/python
import sys
for line in sys.stdin:
        line = line.strip()
        lineorder = line.split('|')
        lo_orderpriority = lineorder[6]
        lo_quantity = lineorder[8]
        lo_discount = lineorder[11]
        lo_revenue = lineorder[12]
        if 'URGENT' in lo_orderpriority:
                print('%s\t%s\t%s' %(lo_revenue, lo_quantity, lo_discount))
```

lineOrder_reducer.py

```
#Ronaldlee Ejalu
#CSC 555 Big Data Mining
# lineOrder_reducer.py
#!/usr/bin/python
import sys
currentKey = None
loQuantityL = []
loDiscountL = []
for line in sys.stdin:
    line = line.strip()
    splittedLinesL = line.split('\t')                               # split the l
ine to create a list of items  e.g lo_revenue \t  lo_quantity \t lo_dis$

    key = splittedLinesL[0]                                         # pick up the
 key   [lo_revenue, lo_quantity, lo_discount]

    lo_quantity = splittedLinesL[1]
    lo_discount = splittedLinesL[2]
    if currentKey == key: #same key
        loQuantityL.append(int(lo_quantity))
        loDiscountL.append(int(lo_discount))
    else:
        if currentKey: # derive the maximum quantity and discount

            lenQuantity = len(loQuantityL)
            lenDiscount = len(loDiscountL)
            if (lenQuantity * lenDiscount > 0):
                # derive the maximum quantity from a list of lo_quantities
                # derive the maximum discount from a list of lo_discount
                print('%s\t%s\t%s' %(currentKey, str(max(loQuantityL)), str(max(l
oDiscountL))))

        loQuantityL = []                                        # re-
initialize the two lists when the keys are not the same (new key) before adding$
        loDiscountL = []

        currentKey = key
        loQuantityL.append(int(lo_quantity))
        loDiscountL.append(int(lo_discount))

# output the last key
# and computer the maximum quantity and discount of all key's values in the diffe
rent list
if currentKey == key:
```

Python Code for the second job:

lineOrder_mapper2.py

```
# Ronaldlee Ejalu
# CSC 555 Big Data Mining
# lineOrder_mapper2.py

#!/usr/bin/python
import sys
for line in sys.stdin:
    line = line.strip()
    lineorder = line.split('\t')
    #print(lineorder)
    lo_revenue = lineorder[0]
    lo_quantity = lineorder[1]
    lo_discount = lineorder[2]
    if 4 <= int(lo_discount) <= 8: # if lo_discount is between 4 and 8
        print('%s\t%s' %(lo_quantity, lo_revenue))
```

lineOrder_reducer2.py

```
# Ronaldlee Ejalu
# CSC 555 Big Data Mining
# lineOrder_reducer2.py

#!/usr/bin/python
import sys

loRevenueL = []
cur_id =  None
key = ''
for line in sys.stdin:

    line = line.strip()
    splittedLines = line.split('\t')
    key = splittedLines[0]
    # lo_revenue = splittedLines[1]
    # print(type(cur_id))
    if cur_id == key:
        loRevenueL.append(int(splittedLines[1]))
    else:
        if cur_id:
            print('%s\t%s'%(cur_id, str(max(loRevenueL))))

        loRevenueL = []
        cur_id = key
        # print(type(cur_id))
        loRevenueL.append(int(splittedLines[1]))
# output the last key
if cur_id == key:
    print('%s\t%s'%(cur_id, str(max(loRevenueL))))
```

NOTE: You may implement this part in Java if you prefer.

Submit a single document containing your written answers. Be sure that this document contains your name and “CSC 555 Project Phase 1” at the top.