CSC 555: Mining Big Data

Project, Phase 1 (due Sunday, February 14th)

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://rasinsrv07.cstcis.cti.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://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/dwdate.tbl
http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/lineorder.tbl
http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/part.tbl
http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/supplier.tbl
http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/customer.tbl
Please be sure to submit all code (pig, python and HiveQL).

Part 1: Multi-node cluster

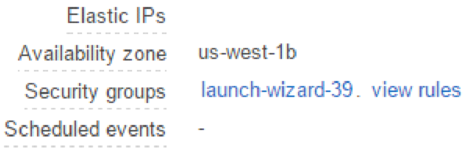
1) Your first step is to setup a multi-node cluster and re-run a simple wordcount. For this part, you will create a 3-node cluster (with a total of 1 master + 2 worker nodes). Include your master node in the “slaves” file, to make sure **all 3** nodes are working.
You need to perform the following steps:

   1. Create a new node of a medium size (you can always switch the size of the node). It is possible, but I do not recommend trying to reconfigure your existing Hadoop into this new cluster (it is much easier to make 3 new nodes for a total of 4 in your AWS account).

      a. **When creating a node I recommend changing the default 8G hard drive to 30G on all nodes.**

      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).

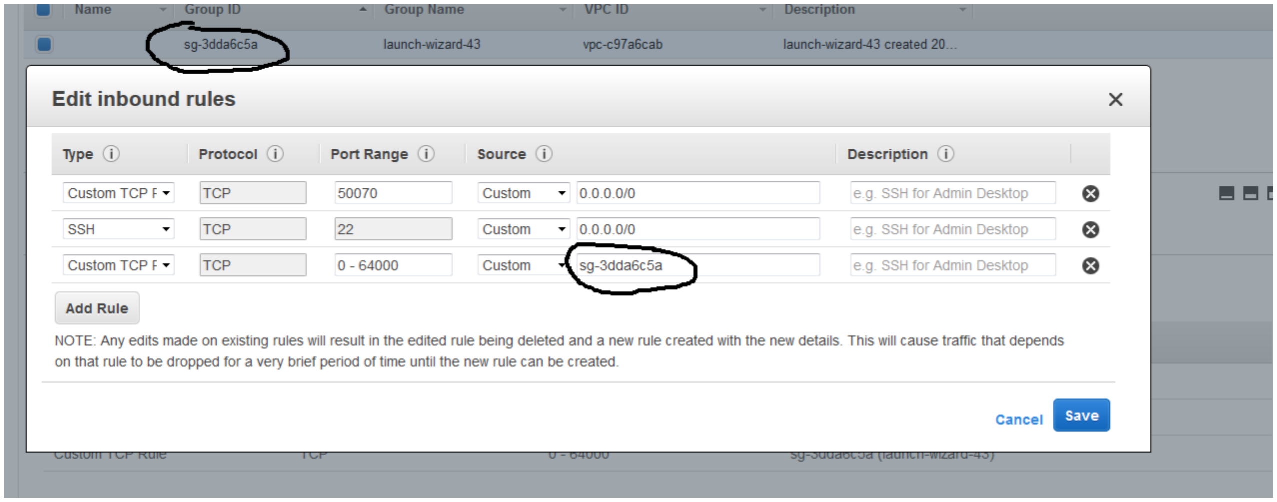


Elastic IPs	
Availability zone	us-west-1b
Security groups	launch-wizard-39 . view rules
Scheduled events	-

Right click on the security group and choose Edit inbound rules

Christian Craig Csc 555 project phase1

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 – that will happen automatically if you use the “create more like this” option when creating instances as specified in part 1-c below). 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

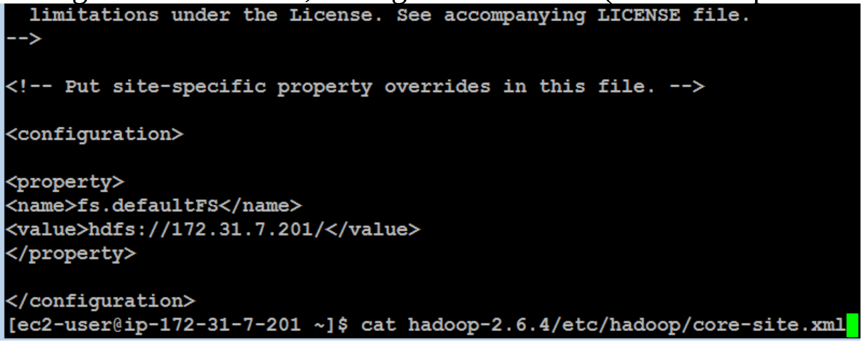
Type	Protocol	Port Range	Source		Description
Custom TCP F	TCP	50070	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP F	TCP	0 - 64000	Custom	sg-3dda6c5a	e.g. SSH for Admin Desktop

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. Right click on the Master node and choose “create more like this” to create 2 more nodes with same settings. If you configure the network settings on master first, security group information will be copied.
NOTE: Hard drive size will not be copied and default to 8G unless you change it.

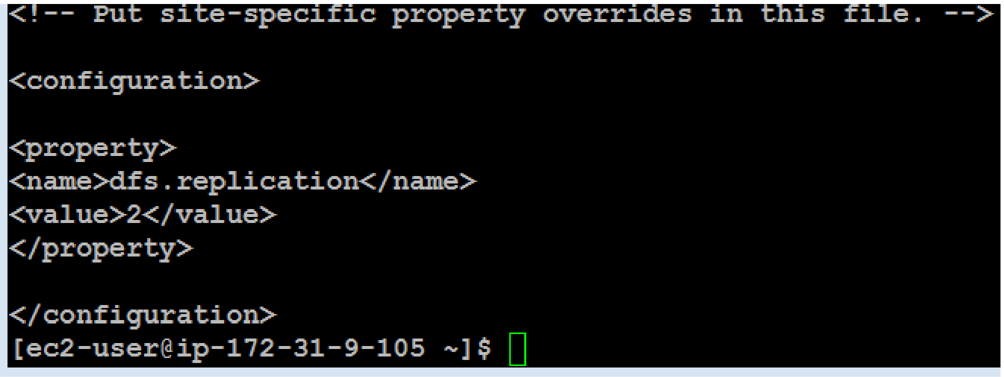
2. Connect to the master and set up Hadoop similarly to what you did previously. Do not attempt to repeat these steps on workers yet – you will only need to set up Hadoop once.

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



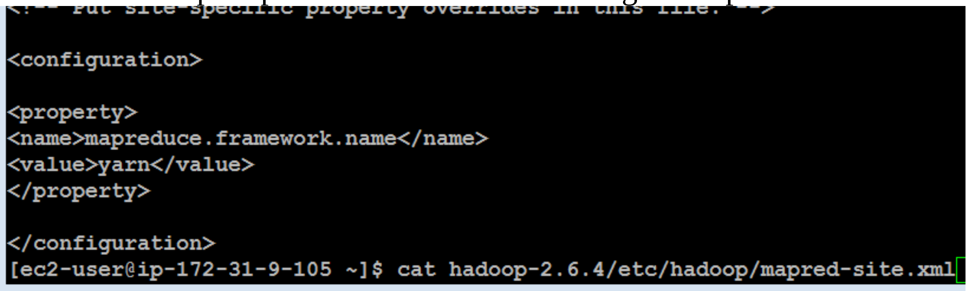
```
  limitations under the License. See accompanying LICENSE file.
-->

<!-- 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.



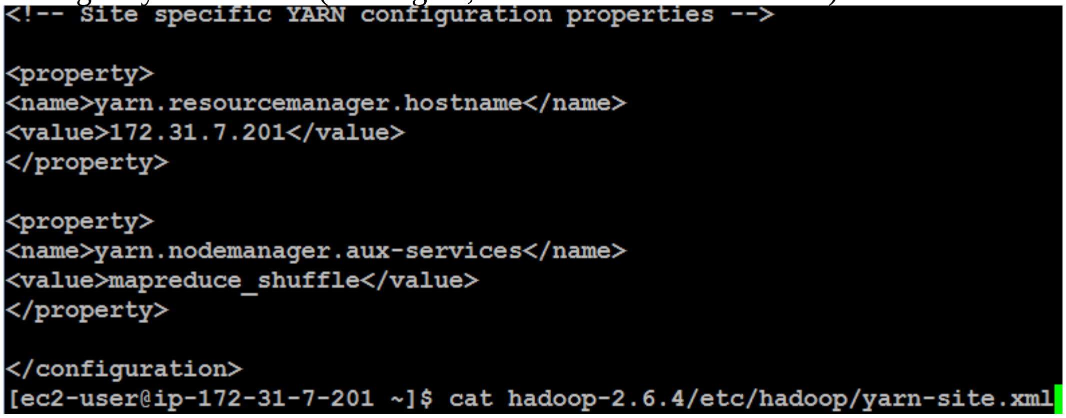
```
<!-- Put site-specific property overrides in this file. -->

<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 slaves file and list your 3 nodes (master and 2 workers) using Private IPs

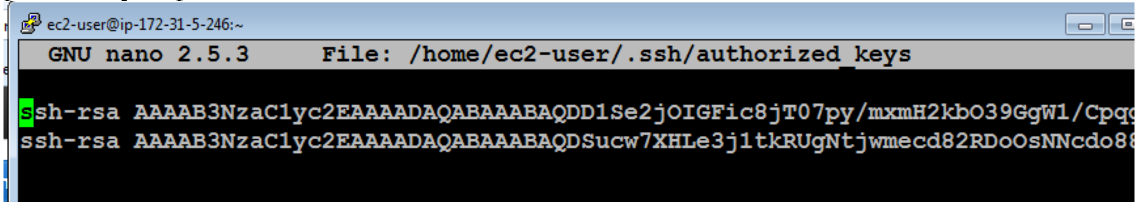
```
[ec2-user@ip-172-31-7-201 ~]$ cat hadoop-2.6.4/etc/hadoop/slaves
172.31.7.201
172.31.5.246
…
```

Make sure that you use private IP (private DNS is also ok) for your configuration files (such as conf/masters and conf/slaves 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.

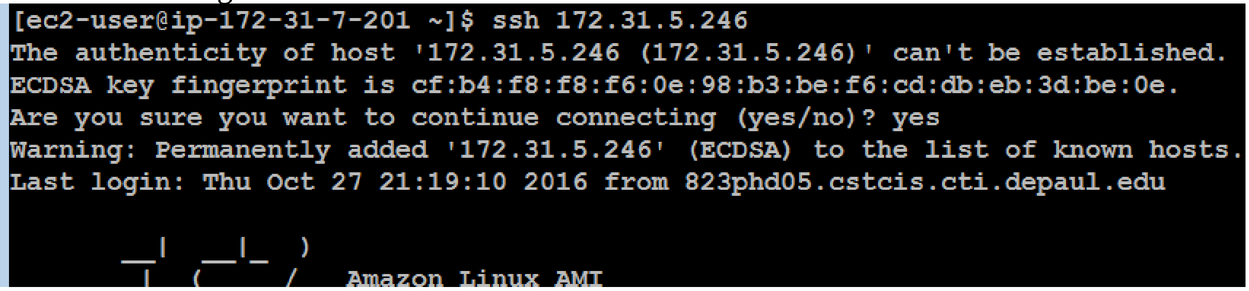
1. Run ssh-keygen -t rsa (and enter empty values for the passphrase) on the master node. That will generate .ssh/id_rsa and .ssh/id_rsa.pub (private and public key). You now need to manually copy the .ssh/id_rsa.pub and append it to ~/.ssh/authorized_keys **on each worker.**
   Keep in mind that this is a single-line public key and accidentally introducing a line break would cause a mismatch.
   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).

/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.272.b10-1.amzn2.0.1.x86_64/jre/bin/java

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)

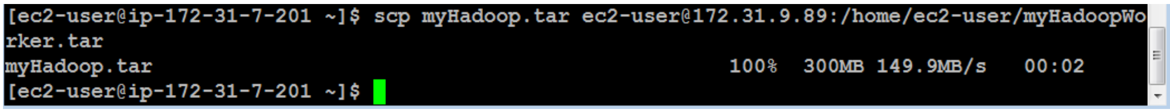
```
tar cvf myHadoop.tar hadoop-2.6.4
```

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 non-master node and unpack the .tar file.

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).

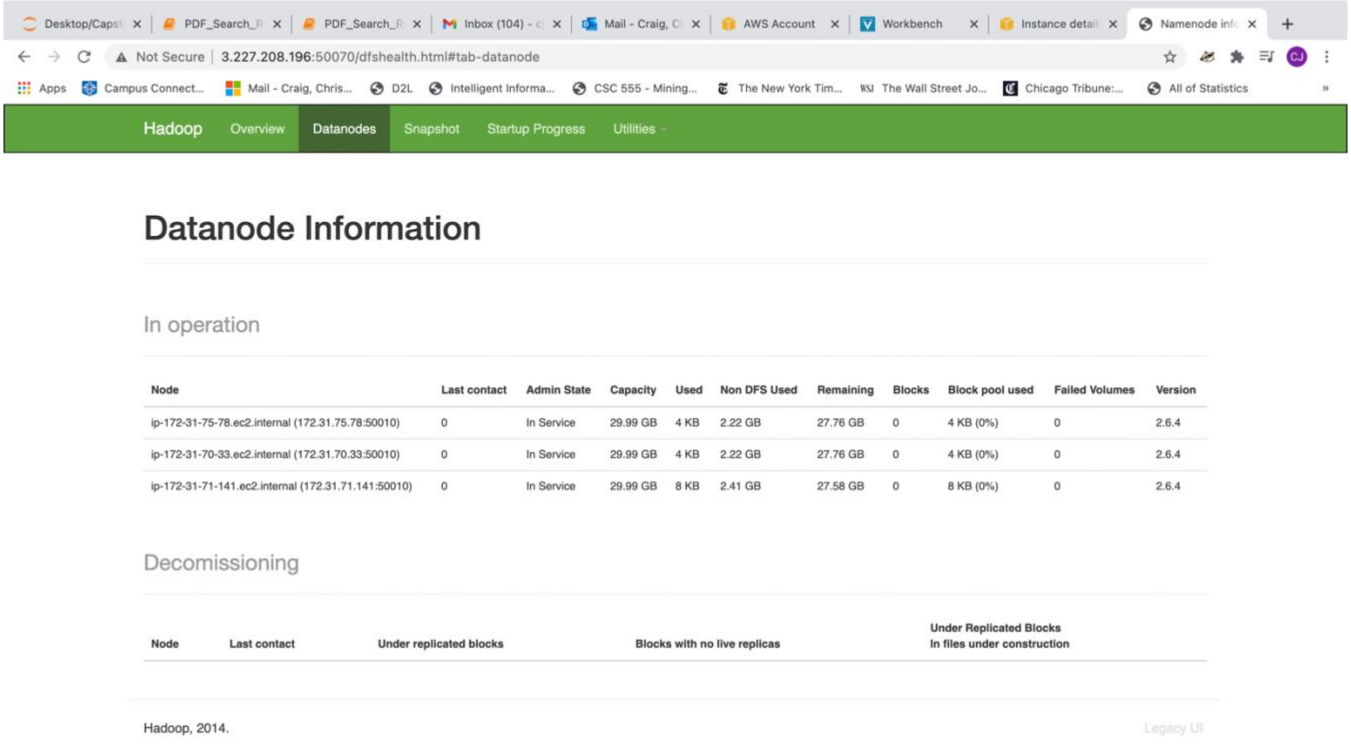
You should verify that the cluster is running by pointing your browser to the link below.

http://[insert-the-public-ip-of-master]:50070/
http://3.227.208.196:50070/

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

Submit a screenshot of your cluster status view.

Christian Craig Csc 555 project phase1



Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
ip-172-31-75-78.ec2.internal (172.31.75.78:50010)	0	In Service	29.99 GB	4 KB	2.22 GB	27.76 GB	0	4 KB (0%)	0	2.6.4
ip-172-31-70-33.ec2.internal (172.31.70.33:50010)	0	In Service	29.99 GB	4 KB	2.22 GB	27.76 GB	0	4 KB (0%)	0	2.6.4
ip-172-31-71-141.ec2.internal (172.31.71.141:50010)	0	In Service	29.99 GB	8 KB	2.41 GB	27.58 GB	0	8 KB (0%)	0	2.6.4

Decomissioning

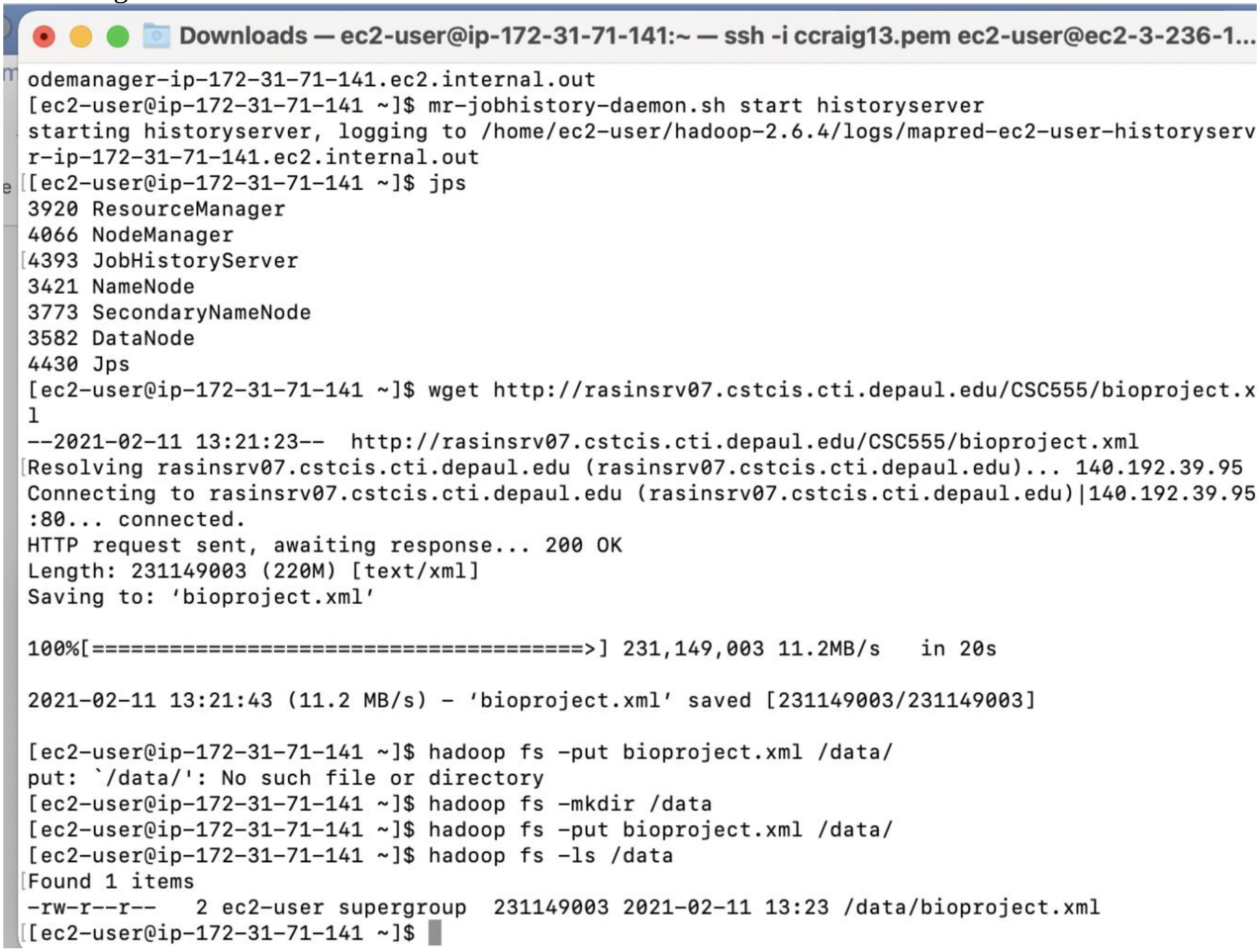
Node	Last contact	Under replicated blocks	Blocks with no live replicas	Under Replicated Blocks In files under construction

Hadoop, 2014.

Legacy UI

Christian Craig Csc 555 project phase1

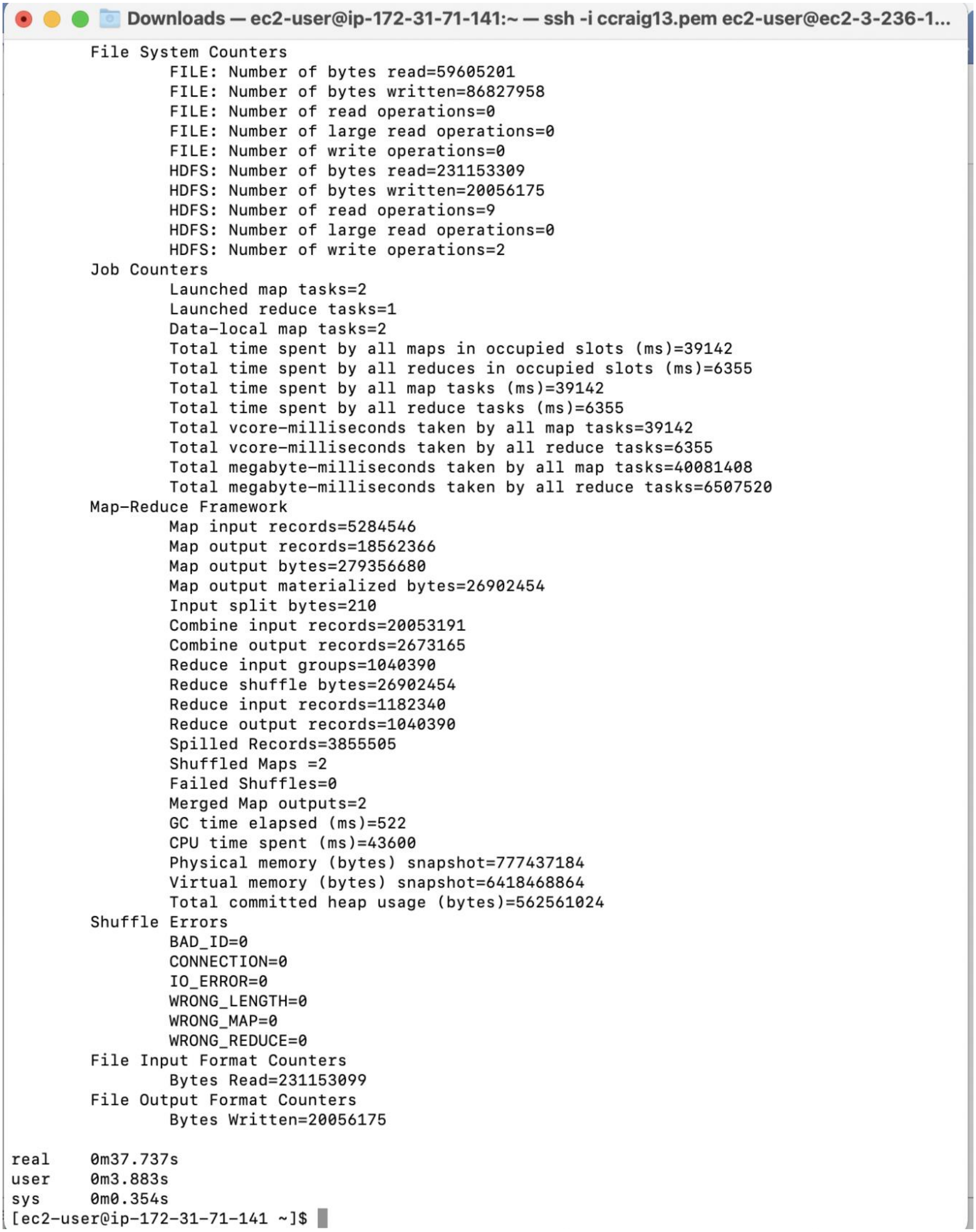
Repeat the steps for wordcount using bioproject.xml from Assignment 1 and submit screenshots of running it.



Downloads — ec2-user@ip-172-31-71-141:~ — ssh -i ccraig13.pem ec2-user@ec2-3-236-1...

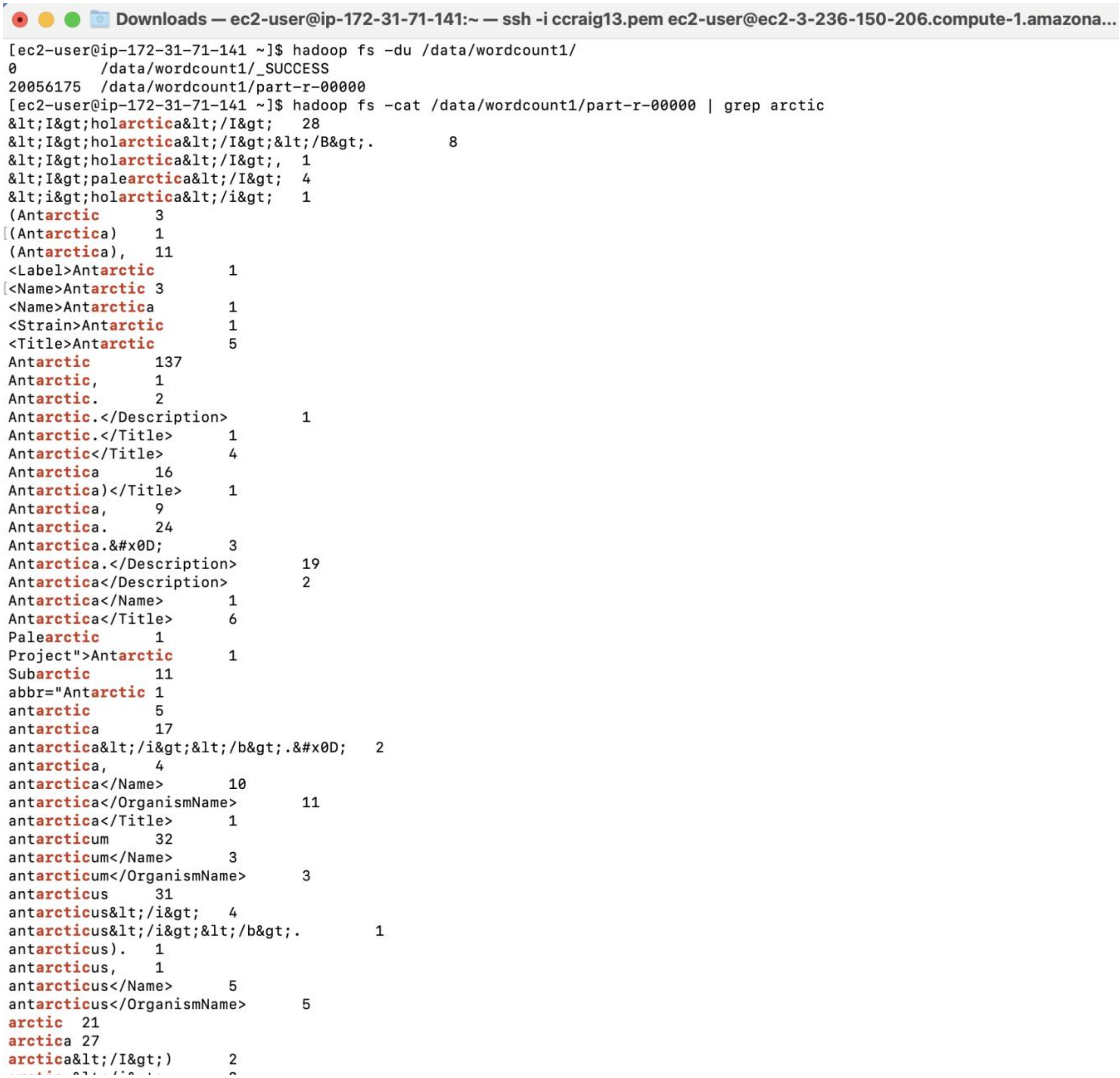
```
odemanager-ip-172-31-71-141.ec2.internal.out
[ec2-user@ip-172-31-71-141 ~]$ mr-jobhistory-daemon.sh start historyserver
starting historyserver, logging to /home/ec2-user/hadoop-2.6.4/logs/mapred-ec2-user-historyserv
r-ip-172-31-71-141.ec2.internal.out
[ec2-user@ip-172-31-71-141 ~]$ jps
3920 ResourceManager
4066 NodeManager
4393 JobHistoryServer
3421 NameNode
3773 SecondaryNameNode
3582 DataNode
4430 Jps
[ec2-user@ip-172-31-71-141 ~]$ wget http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/bioproject.x
l
--2021-02-11 13:21:23--  http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/bioproject.xml
Resolving rasinsrv07.cstcis.cti.depaul.edu (rasinsrv07.cstcis.cti.depaul.edu)... 140.192.39.95
Connecting to rasinsrv07.cstcis.cti.depaul.edu (rasinsrv07.cstcis.cti.depaul.edu)|140.192.39.95
:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 231149003 (220M) [text/xml]
Saving to: ‘bioproject.xml’

100%[======================================>] 231,149,003 11.2MB/s   in 20s

2021-02-11 13:21:43 (11.2 MB/s) - ‘bioproject.xml’ saved [231149003/231149003]

[ec2-user@ip-172-31-71-141 ~]$ hadoop fs -put bioproject.xml /data/
put: `/data/': No such file or directory
[ec2-user@ip-172-31-71-141 ~]$ hadoop fs -mkdir /data
[ec2-user@ip-172-31-71-141 ~]$ hadoop fs -put bioproject.xml /data/
[ec2-user@ip-172-31-71-141 ~]$ hadoop fs -ls /data
Found 1 items
-rw-r--r--   2 ec2-user supergroup  231149003 2021-02-11 13:23 /data/bioproject.xml
[ec2-user@ip-172-31-71-141 ~]$ 
```
]

Christian Craig Csc 555 project phase1


Downloads — ec2-user@ip-172-31-71-141:~ — ssh -i ccraig13.pem ec2-user@ec2-3-236-1...

```
        File System Counters
                FILE: Number of bytes read=59605201
                FILE: Number of bytes written=86827958
                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)=39142
                Total time spent by all reduces in occupied slots (ms)=6355
                Total time spent by all map tasks (ms)=39142
                Total time spent by all reduce tasks (ms)=6355
                Total vcore-milliseconds taken by all map tasks=39142
                Total vcore-milliseconds taken by all reduce tasks=6355
                Total megabyte-milliseconds taken by all map tasks=40081408
                Total megabyte-milliseconds taken by all reduce tasks=6507520
        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)=522
                CPU time spent (ms)=43600
                Physical memory (bytes) snapshot=777437184
                Virtual memory (bytes) snapshot=6418468864
                Total committed heap usage (bytes)=562561024
        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    0m37.737s
user    0m3.883s
sys     0m0.354s
[ec2-user@ip-172-31-71-141 ~]$
```

Christian Craig Csc 555 project phase1


Downloads — ec2-user@ip-172-31-71-141:~ — ssh -i ccraig13.pem ec2-user@ec2-3-236-150-206.compute-1.amazona...

```
[ec2-user@ip-172-31-71-141 ~]$ hadoop fs -du /data/wordcount1/
0         /data/wordcount1/_SUCCESS
20056175  /data/wordcount1/part-r-00000
[ec2-user@ip-172-31-71-141 ~]$ hadoop fs -cat /data/wordcount1/part-r-00000 | grep arctic
&lt;I&gt;holarctica&lt;/I&gt;   28
&lt;I&gt;holarctica&lt;/I&gt;&lt;/B&gt;.        8
&lt;I&gt;holarctica&lt;/I&gt;,  1
&lt;I&gt;palearctica&lt;/I&gt;  4
&lt;i&gt;holarctica&lt;/i&gt;   1
(Antarctic      3
(Antarctica)    1
(Antarctica),   11
<Label>Antarctic        1
<Name>Antarctic 3
<Name>Antarctica        1
<Strain>Antarctic       1
<Title>Antarctic        5
Antarctic       137
Antarctic,      1
Antarctic.      2
Antarctic.</Description>        1
Antarctic.</Title>      1
Antarctic</Title>       4
Antarctica      16
Antarctica)</Title>     1
Antarctica,     9
Antarctica.     24
Antarctica.&#x0D;       3
Antarctica.</Description>       19
Antarctica</Description>        2
Antarctica</Name>       1
Antarctica</Title>      6
Palearctic      1
Project">Antarctic      1
Subarctic       11
abbr="Antarctic 1
antarctic       5
antarctica      17
antarctica&lt;/i&gt;&lt;/b&gt;.&#x0D;   2
antarctica,     4
antarctica</Name>       10
antarctica</OrganismName>       11
antarctica</Title>      1
antarcticum     32
antarcticum</Name>      3
antarcticum</OrganismName>      3
antarcticus     31
antarcticus&lt;/i&gt;   4
antarcticus&lt;/i&gt;&lt;/b&gt;.        1
antarcticus).   1
antarcticus,    1
antarcticus</Name>      5
antarcticus</OrganismName>      5
arctic  21
arctica 27
arctica&lt;/I&gt;)      2
```

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

The results are much faster with the time being 1m13.7s with one node to 37.7s with three nodes. So, the time is cut by 51%. This is due to the fact that the nodes are able to split the work amongst each other. Instead of just one node working the task, there is now 3 that are able to separate the task, and therefore, reduce 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 3 and 4
  and lo_quantity < 26
GROUP BY lo_orderdate;
```

Time taken: 26.847 seconds, Fetched: 366 row(s)

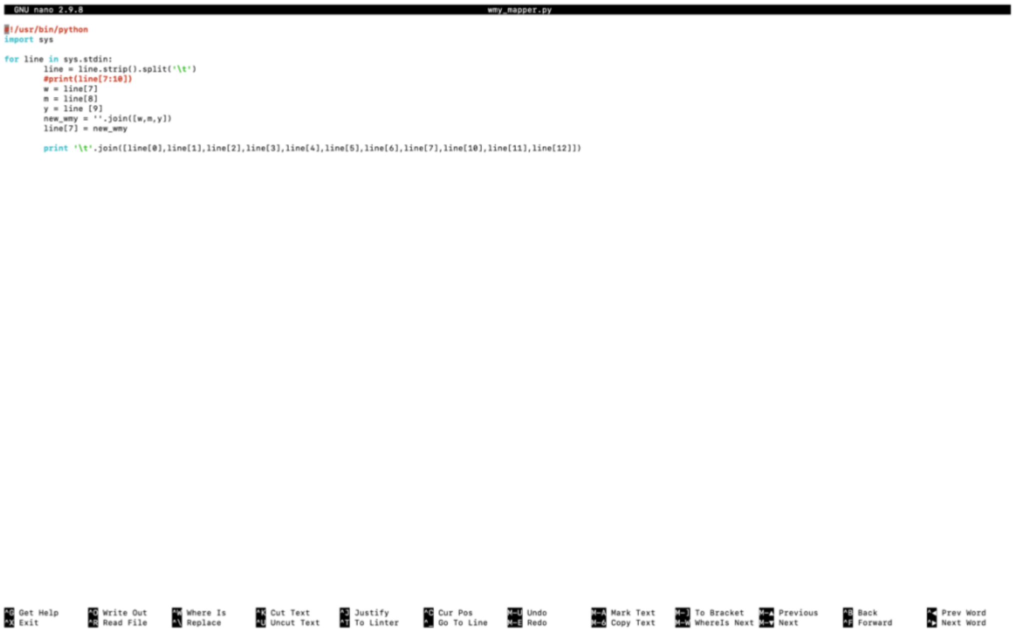
2) Perform the following transform operation using SELECT TRANSFORM on the dwdate table by creating a new table. The new dwdate table will combine d_daynuminweek, d_daynuminmonth, and d_daynuminyear into a single column in the new table. You should also eliminate of the last 3 columns (d_lastdayinmonthfl,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

```
create table dwdate (
  d_datekey           int,
  d_date              varchar(19),
  d_dayofweek         varchar(10),
  d_month             varchar(10),
  d_year              int,
  d_yearmonthnum      int,
  d_yearmonth         varchar(8),
  d_daynuminweek      int,
  d_daynuminmonth     int,
  d_daynuminyear      int,
  d_monthnuminyear    int,
  d_weeknuminyear     int,
  d_sellingseason     varchar(13),
  d_lastdayinweekfl   varchar(1),
  d_lastdayinmonthfl  varchar(1),
  d_holidayfl         varchar(1),
  d_weekdayfl         varchar(1)
)
ROW FORMAT DELIMITED FIELDS
TERMINATED BY '|' STORED AS TEXTFILE;
```

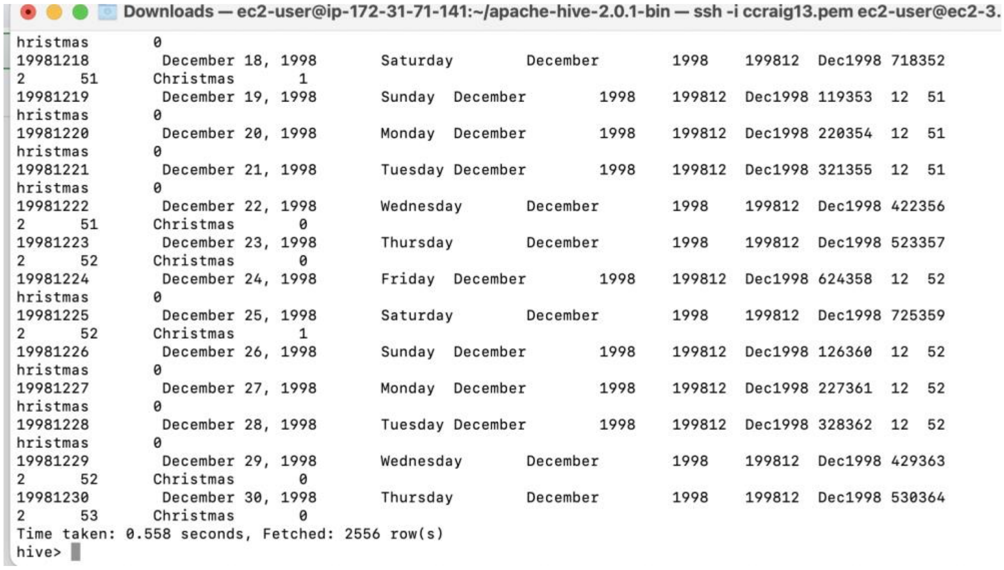
```
LOAD DATA LOCAL INPATH '/home/ec2-user/dwdate.tbl'
OVERWRITE INTO TABLE dwdate;
```

```
create table dwdate_transform (
d_datekey           int,
d_date              varchar(19),
d_dayofweek         varchar(10),
d_month             varchar(10),
d_year              int,
d_yearmonthnum      int,
d_yearmonth         varchar(8),
d_wmy               int,
d_monthnuminyear    int,
d_weeknuminyear     int,
d_sellingseason     varchar(13),
d_lastdayinweekfl   varchar(1)

)

ROW FORMAT DELIMITED FIELDS
TERMINATED BY '\t' STORED AS TEXTFILE;


INSERT OVERWRITE TABLE dwdate_transform

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_weeknuminyear,d_sellingseason,d_lastdayinweekfl,d_lastdayinmonthfl,d_holidayfl,d_weekdayfl) USING 'python wmy_mapper.py'



AS

(d_datekey,d_date,d_dayofweek,d_month,d_year,d_yearmonthnum,d_yearmonth,d_wmy,d_monthnuminyear,d_weeknuminyear,d_sellingseason,d_lastdayinweekfl) FROM dwdate;
```

Christian Craig Csc 555 project phase1



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

for line in sys.stdin:
        line = line.strip().split('\t')
        #print(line[7:10])
        w = line[7]
        m = line[8]
        y = line [9]
        new_wmy = ''.join([w,m,y])
        line[7] = new_wmy

        print '\t'.join([line[0],line[1],line[2],line[3],line[4],line[5],line[6],line[7],line[10],line[11],line[12]])
```


Downloads — ec2-user@ip-172-31-71-141:~/apache-hive-2.0.1-bin — ssh -i ccraig13.pem ec2-user@ec2-3.

```
hristmas       0
19981218        December 18, 1998       Saturday        December        1998    199812  Dec1998 718352
2       51      Christmas       1
19981219        December 19, 1998       Sunday  December        1998    199812  Dec1998 119353  12      51
hristmas       0
19981220        December 20, 1998       Monday  December        1998    199812  Dec1998 220354  12      51
hristmas       0
19981221        December 21, 1998       Tuesday December        1998    199812  Dec1998 321355  12      51
hristmas       0
19981222        December 22, 1998       Wednesday       December        1998    199812  Dec1998 422356
2       51      Christmas       0
19981223        December 23, 1998       Thursday        December        1998    199812  Dec1998 523357
2       52      Christmas       0
19981224        December 24, 1998       Friday  December        1998    199812  Dec1998 624358  12      52
hristmas       0
19981225        December 25, 1998       Saturday        December        1998    199812  Dec1998 725359
2       52      Christmas       1
19981226        December 26, 1998       Sunday  December        1998    199812  Dec1998 126360  12      52
hristmas       0
19981227        December 27, 1998       Monday  December        1998    199812  Dec1998 227361  12      52
hristmas       0
19981228        December 28, 1998       Tuesday December        1998    199812  Dec1998 328362  12      52
hristmas       0
19981229        December 29, 1998       Wednesday       December        1998    199812  Dec1998 429363
2       52      Christmas       0
19981230        December 30, 1998       Thursday        December        1998    199812  Dec1998 530364
2       53      Christmas       0
Time taken: 0.558 seconds, Fetched: 2556 row(s)
hive> 
```

Part 3: Pig

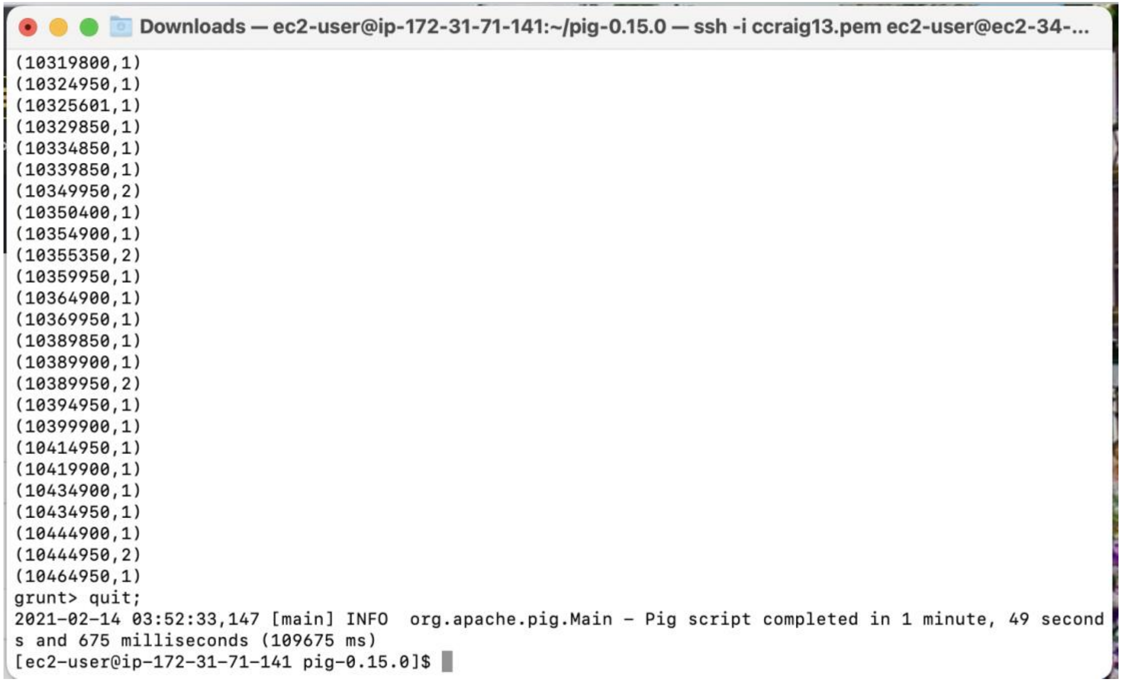
Convert and load the data into Pig, implementing and timing the following queries:One easy way to time Pig is as follows: put your sequence of pig commands, including LOAD, into a text file and then run, from command line in pig directory (e.g., [ec2-user@ip-172-31-6-39 pig-0.15.0]$), **bin/pig -f pig_script.pig**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

SELECT lo_discount, COUNT(lo_extendedprice)

Christian Craig Csc 555 project phase1

```
FROM lineorder
GROUP BY lo_discount;
```

```
bin/pig
lineorder_tbl = LOAD '/user/ec2-user/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_ordertotalpri
ce:int,lo_discount: int,lo_revenue: int,lo_supplycost:int,lo_tax: int,lo_commitdate: int,
lo_shipmode:chararray);

lo_group1 = group lineorder_tbl by $12;


lo_count1 = foreach lo_group1 generate group as disc, COUNT(lineorder_tbl.lo_extendedprice);


dump lo_count1;
quit;
```


Downloads — ec2-user@ip-172-31-71-141:~/pig-0.15.0 — ssh -i ccraig13.pem ec2-user@ec2-34-...

```
(10319800,1)
(10324950,1)
(10325601,1)
(10329850,1)
(10334850,1)
(10339850,1)
(10349950,2)
(10350400,1)
(10354900,1)
(10355350,2)
(10359950,1)
(10364900,1)
(10369950,1)
(10389850,1)
(10389900,1)
(10389950,2)
(10394950,1)
(10399900,1)
(10414950,1)
(10419900,1)
(10434900,1)
(10434950,1)
(10444900,1)
(10444950,2)
(10464950,1)
grunt> quit;
2021-02-14 03:52:33,147 [main] INFO  org.apache.pig.Main - Pig script completed in 1 minute, 49 second
s and 675 milliseconds (109675 ms)
[ec2-user@ip-172-31-71-141 pig-0.15.0]$ 
```

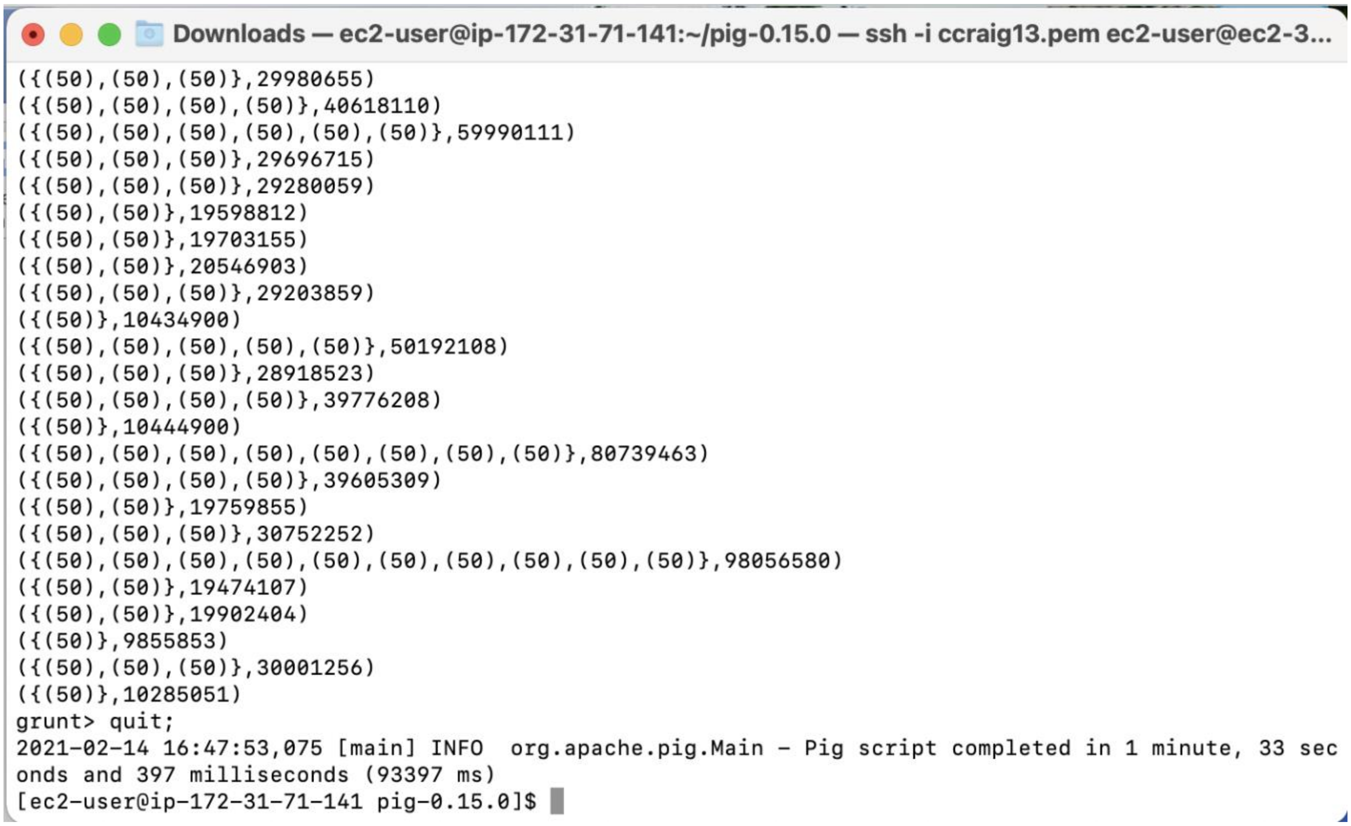
1m49s

```
SELECT lo_quantity, SUM(lo_revenue)
FROM lineorder
WHERE lo_discount > 6 AND lo_quantity > 33
```

Christian Craig Csc 555 project phase1

GROUP BY lo_quantity;

```
lineorder_tbl = LOAD '/user/ec2-user/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);

lo_filter = filter lineorder_tbl by $12 > 6 AND $9 > 33;


lo_group2 = group lo_filter by $9;


lo_sum = foreach lo_group2 generate lo_filter.lo_quantity, SUM(lo_filter.lo_revenue);


dump lo_sum;

quit;
```


Downloads — ec2-user@ip-172-31-71-141:~/pig-0.15.0 — ssh -i ccraig13.pem ec2-user@ec2-3...

```
({(50),(50),(50)},29980655)
({(50),(50),(50),(50)},40618110)
({(50),(50),(50),(50),(50),(50)},59990111)
({(50),(50),(50)},29696715)
({(50),(50),(50)},29280059)
({(50),(50)},19598812)
({(50),(50)},19703155)
({(50),(50)},20546903)
({(50),(50),(50)},29203859)
({(50)},10434900)
({(50),(50),(50),(50),(50)},50192108)
({(50),(50),(50)},28918523)
({(50),(50),(50),(50)},39776208)
({(50)},10444900)
({(50),(50),(50),(50),(50),(50),(50),(50)},80739463)
({(50),(50),(50),(50)},39605309)
({(50),(50)},19759855)
({(50),(50),(50)},30752252)
({(50),(50),(50),(50),(50),(50),(50),(50),(50),(50)},98056580)
({(50),(50)},19474107)
({(50),(50)},19902404)
({(50)},9855853)
({(50),(50),(50)},30001256)
({(50)},10285051)
grunt> quit;
2021-02-14 16:47:53,075 [main] INFO  org.apache.pig.Main - Pig script completed in 1 minute, 33 sec
onds and 397 milliseconds (93397 ms)
[ec2-user@ip-172-31-71-141 pig-0.15.0]$ 
```

Christian Craig Csc 555 project phase1

1m33s

```
Part_tbl = LOAD '/user/ec2-user/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);
```

```
supplier_tbl = LOAD '/user/ec2-user/supplier.tbl' USING PigStorage('|') AS(s_suppkey:int,
s_name:chararray, s_address:chararray, s_city:chararray,s_nation:chararray, s_region:chararray,
s_phone:chararray);
```

```
customer_tbl = LOAD '/user/ec2-user/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);
```

```
dwdate_tbl = LOAD '/user/ec2-user/dwdate.tbl' USING PigStorage('|') AS(d_datekey:int,
d_date:chararray, d_dayofweek:chararray, d_year:int,
d_yearmonthnum:int,d_yearmonth:chararray,d_daynuminweek:int,d_daynuminmonth:int,d_daynuminyear:int,d_monthnuminyear:int,d_weeknuminyear:int,d_sellingseason:chararray,d_lastdayinweekfl:chararray,d_lastdayinmonthfl: chararray,d_holidayfl:chararray,d_weekdayfl:chararray);
```

```
lineorder_tbl = LOAD '/user/ec2-user/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);
```

Part 4: Hadoop Streaming

Implement, run and time the following query using Hadoop streaming with python. STDDEV refers to standard deviation.

```
SELECT lo_shipmode, STDDEV(lo_tax)
FROM lineorder
WHERE lo_quantity BETWEEN 15 AND 18
GROUP BY lo_shipmode;
```

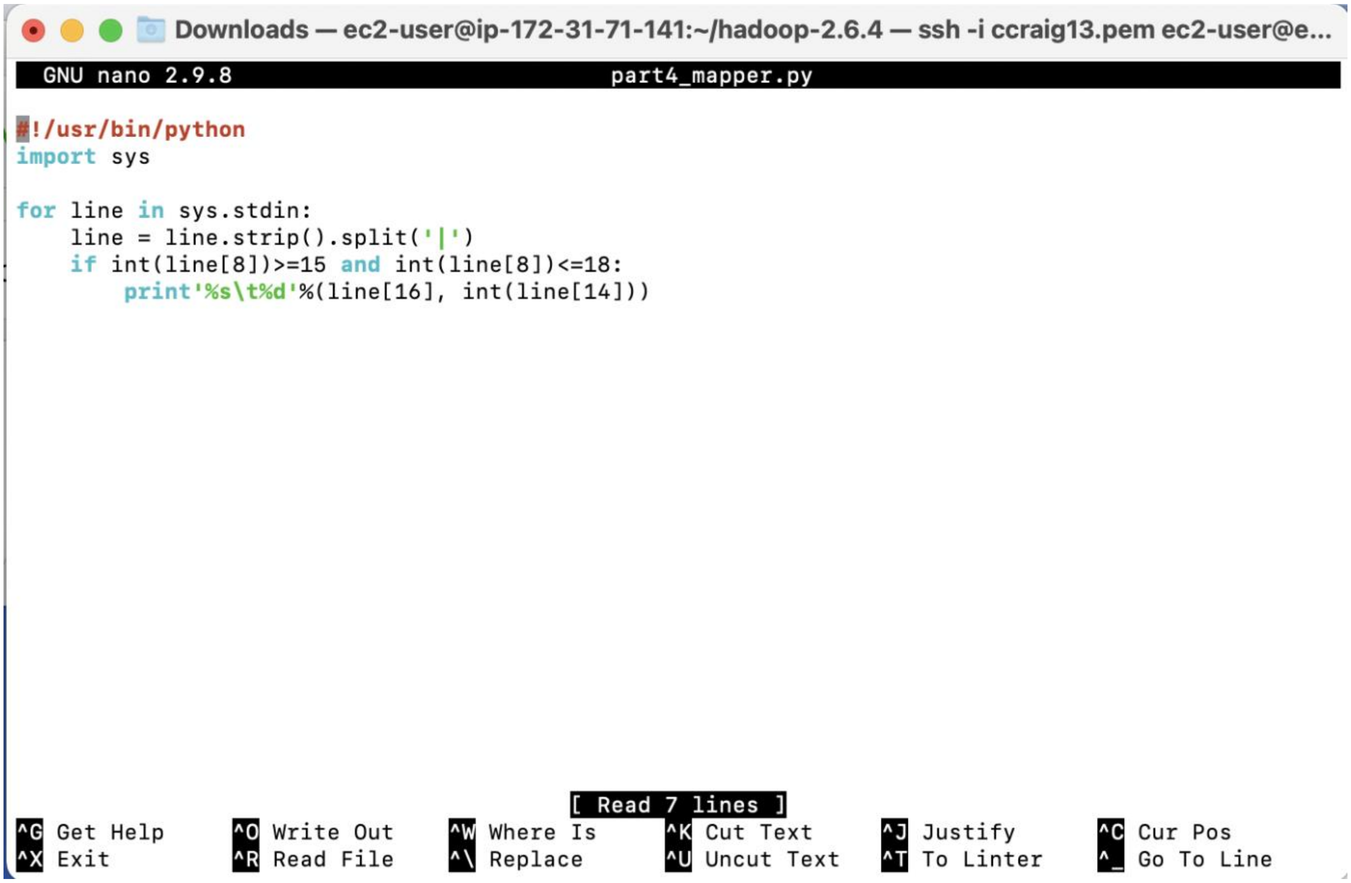
```
create table lineorder (
  lo_orderkey          int,
  lo_linenumber        int,
  lo_custkey           int,
  lo_partkey           int,
  lo_suppkey           int,
  lo_orderdate         int,
  lo_orderpriority     varchar(15),
  lo_shippriority      varchar(1),
  lo_quantity          int,
  lo_extendedprice     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;
```

Christian Craig Csc 555 project phase1

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

OVERWRITE INTO TABLE lineorder;

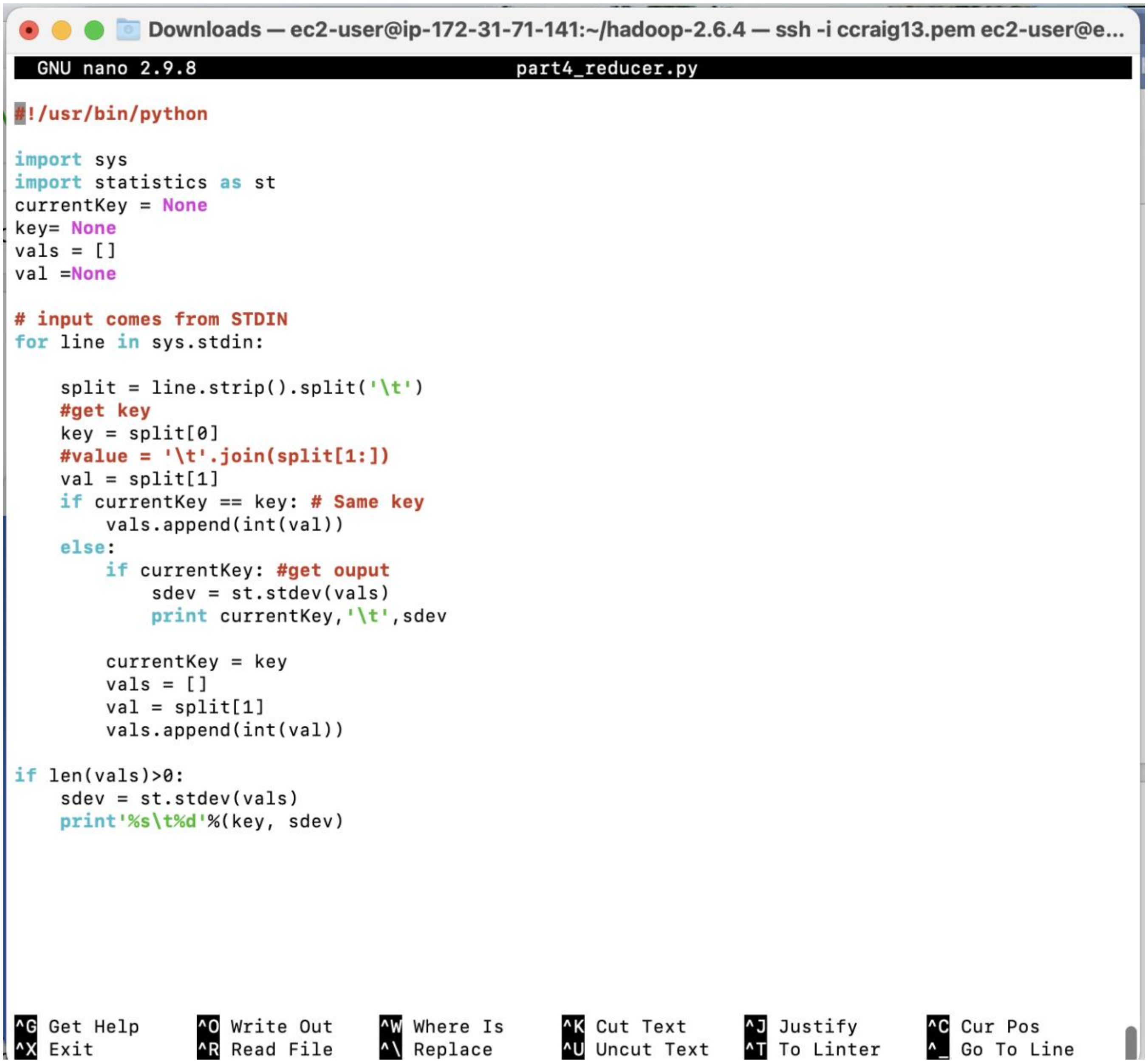
```
-rw-r--r--   2 ec2-user supergroup  594313001 2021-02-13 22:19 lineorder.tbl
```


Downloads — ec2-user@ip-172-31-71-141:~/hadoop-2.6.4 — ssh -i ccraig13.pem ec2-user@e...

```
  GNU nano 2.9.8                     part4_mapper.py

#!/usr/bin/python
import sys

for line in sys.stdin:
    line = line.strip().split('|')
    if int(line[8])>=15 and int(line[8])<=18:
        print'%s\t%d'%(line[16], int(line[14]))






                                  [ Read 7 lines ]
^G Get Help     ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos
^X Exit         ^R Read File    ^\ Replace      ^U Uncut Text   ^T To Linter    ^_ Go To Line
```

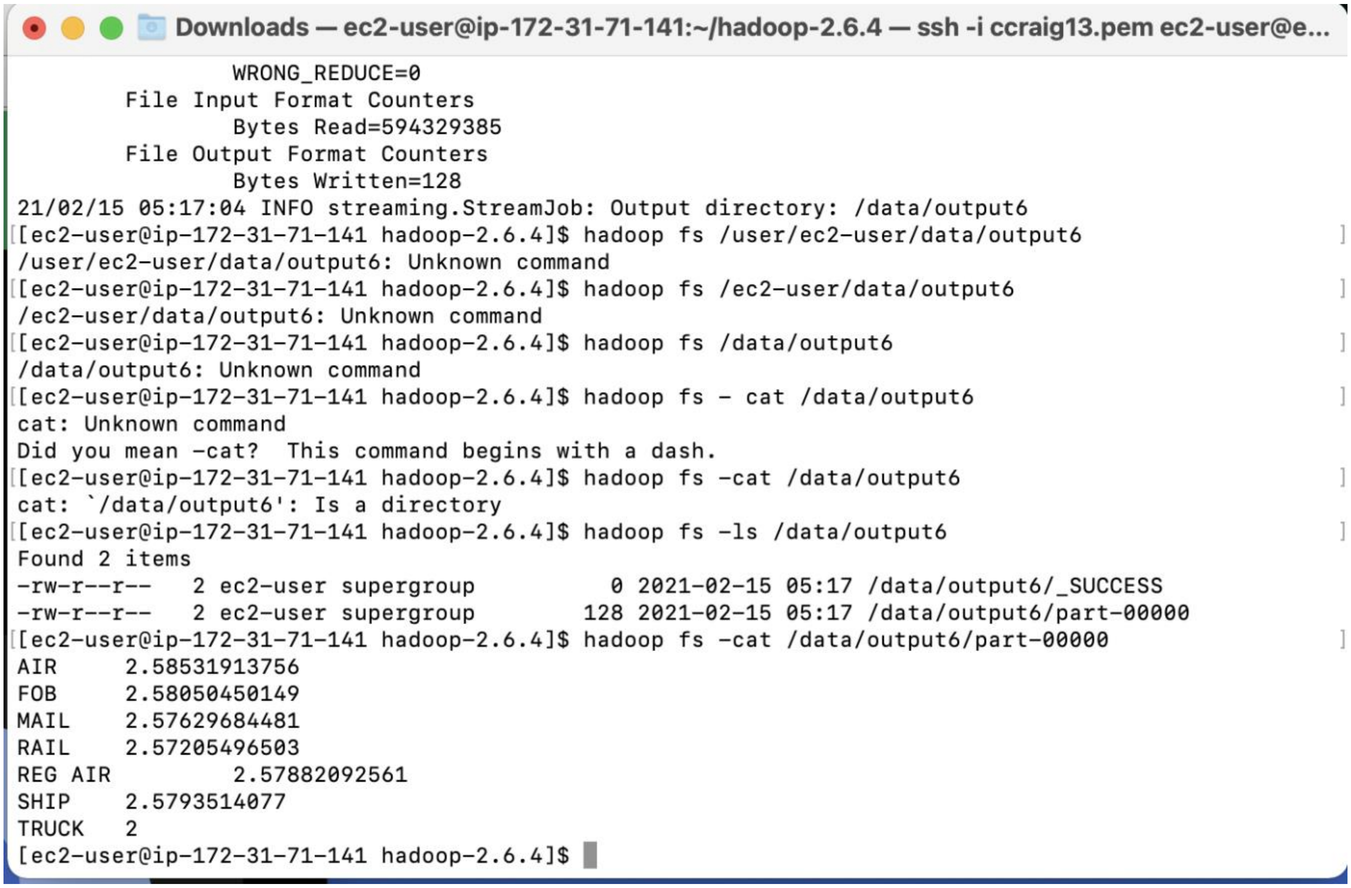
Christian Craig Csc 555 project phase1

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



```python
#!/usr/bin/python

import sys
import statistics as st
currentKey = None
key= None
vals = []
val =None

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

    split = line.strip().split('\t')
    #get key
    key = split[0]
    #value = '\t'.join(split[1:])
    val = split[1]
    if currentKey == key: # Same key
        vals.append(int(val))
    else:
        if currentKey: #get ouput
            sdev = st.stdev(vals)
            print currentKey,'\t',sdev

        currentKey = key
        vals = []
        val = split[1]
        vals.append(int(val))

if len(vals)>0:
    sdev = st.stdev(vals)
    print'%s\t%d'%(key, sdev)
```

Christian Craig Csc 555 project phase1



Downloads — ec2-user@ip-172-31-71-141:~/hadoop-2.6.4 — ssh -i ccraig13.pem ec2-user@e...

```
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=594329385
        File Output Format Counters
                Bytes Written=128
21/02/15 05:17:04 INFO streaming.StreamJob: Output directory: /data/output6
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs /user/ec2-user/data/output6
/user/ec2-user/data/output6: Unknown command
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs /ec2-user/data/output6
/ec2-user/data/output6: Unknown command
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs /data/output6
/data/output6: Unknown command
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs - cat /data/output6
cat: Unknown command
Did you mean -cat?  This command begins with a dash.
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs -cat /data/output6
cat: `/data/output6': Is a directory
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs -ls /data/output6
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2021-02-15 05:17 /data/output6/_SUCCESS
-rw-r--r--   2 ec2-user supergroup        128 2021-02-15 05:17 /data/output6/part-00000
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ hadoop fs -cat /data/output6/part-00000
AIR     2.58531913756
FOB     2.58050450149
MAIL    2.57629684481
RAIL    2.57205496503
REG AIR         2.57882092561
SHIP    2.5793514077
TRUCK   2
[ec2-user@ip-172-31-71-141 hadoop-2.6.4]$ 
```

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.