**CSC 555: Mining Big Data**

Project, Phase 1

In this part of the project, you will 1) Set up a 3-node cluster and 2) perform data warehousing and transformation queries using Hive, Pig and Hadoop streaming. The modified Hive-style schema is still available at:

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

It is based on SSBM benchmark (derived from industry standard TPCH benchmark). I modified it from SQL to HiveQL. This is 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 (not csv).

<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 and python).

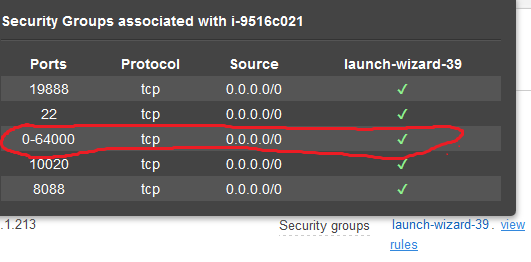
# Part 1: Multi-node cluster

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:

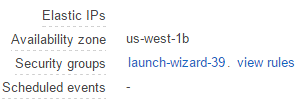
1. Create a new node of a medium size. It is possible, but I do not recommend trying to reconfigure your existing Hadoop into this new cluster (it is much easier to make 3 new nodes for a total of 4 in your AWS account).
   1. When creating a node I recommend changing the default 8G hard drive to 30G so that you do not run out of space easily.
   2. Change your security group setting to open firewall access. Rather than figure out all individual port, you can set 0-64000 range opening up all ports (not the

most secure setting in the long term, but fine for us)

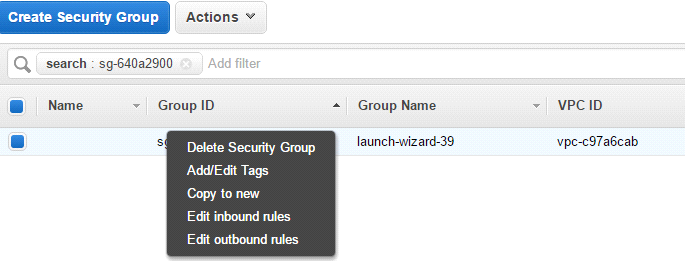


* 1. Step by step instructions on how to make the change to open up the ports:

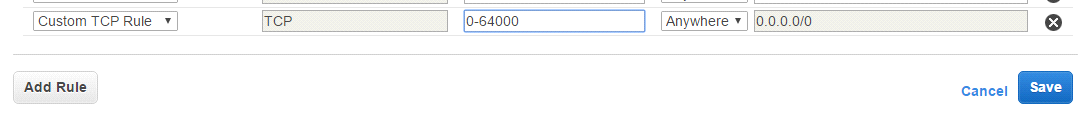
Click on security group (launch-wizard-x)



Right click on the security group and choose Edit inbound rules



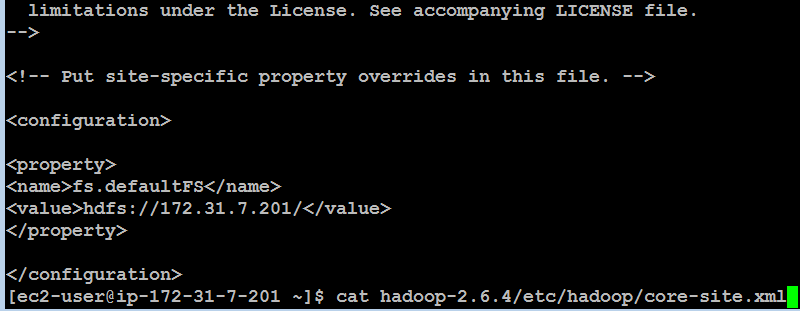
Add a new rule and put in the ports 0-64000 and “Anywhere” and click save.

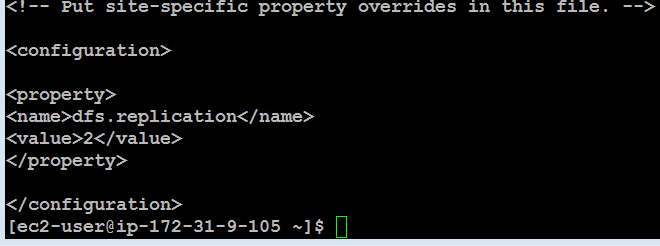


This will open the firewall completely for all ports.

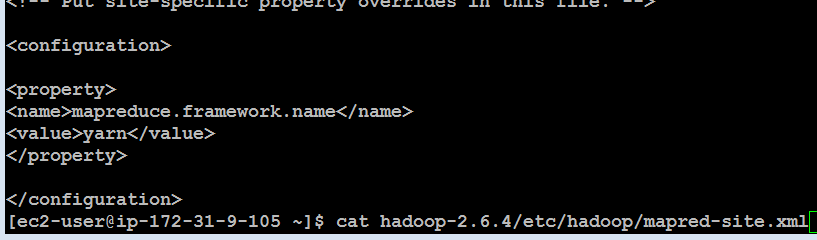
* 1. Finally, right click on the Master node and choose “create more like this” to create 2 more nodes with same settings. If you configure the network settings on master first, security group information will be copied.

NOTE: Hard drive size will not be copied and default to 8G unless you change it.

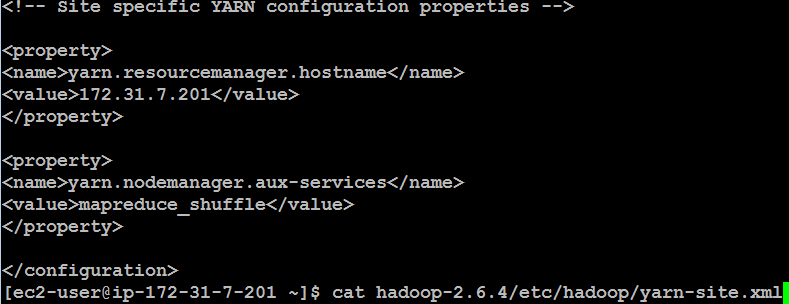
1. Connect to the master and set up Hadoop similarly to what you did previously. Do not attempt to repeat these steps on workers yet – you will only need to set up Hadoop once.
   1. Configure core-site.xml, adding the **PrivateIP** (do not use public IP) of the master. 
   2. Configure hdfs-site and set replication factor to 2.



* 1. 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



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



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

172.31.11.50

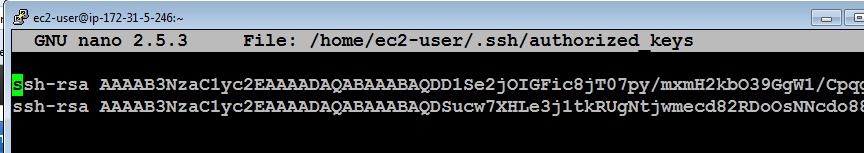
Make sure that you use private IP (private DNS is also ok) for your configuration files (such as conf/masters and conf/slaves or the other 3 config files). The advantage of the Private IP is that it does not change after your instance is stopped (if you use the Public IP, the cluster would need to be 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 break it. For example:

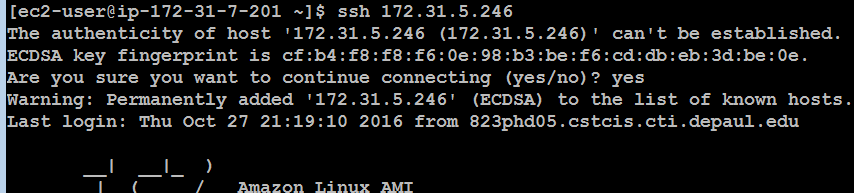
Note that this is NOT the master, but one of the workers (ip-172-31-5-246). The first public key is the .pem Amazon half and the 2nd public key is the master’s public key copied in as one line.



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.



Once you are 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 up the contents of the hadoop directory there. Make sure your Hadoop installation is configured correctly (because from now on, you will have 3 copies of the Hadoop directory and all changes may need to be applied in 3 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)

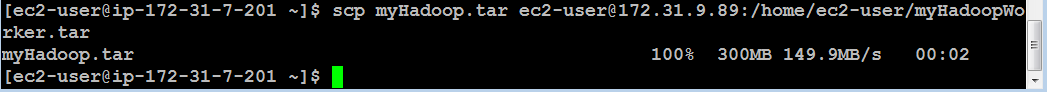
tar cvf myHadoop.tar hadoop-2.6.4

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



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 successfully copied myHadoopWorker.tar file.

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

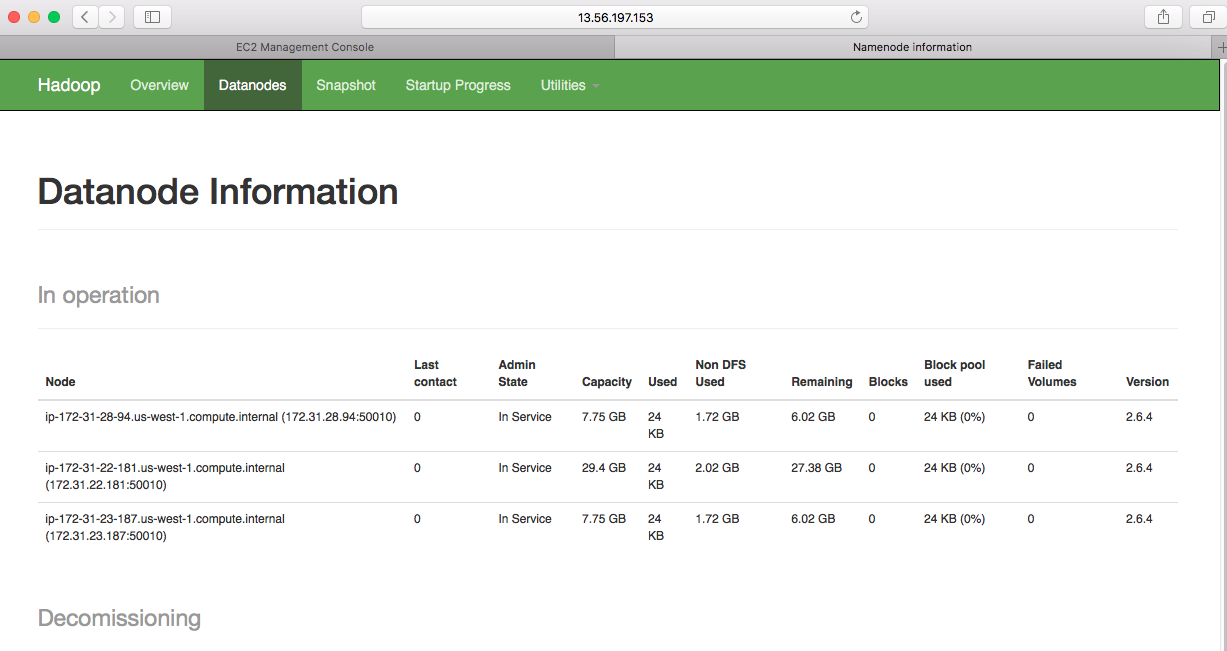
Once you have successfully completed the previous steps, you should be able to start and use your new cluster by going to the master node and running the start-dfs.sh and start-yarn.sh scripts (you do not need to explicitly start anything on worker nodes – the master will do that for you).

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

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

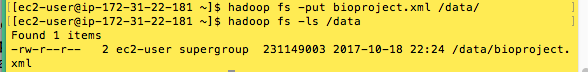
Make sure that the cluster is operational (you can see the 3 nodes under Datanodes tab).

Submit a screenshot of your cluster status view.



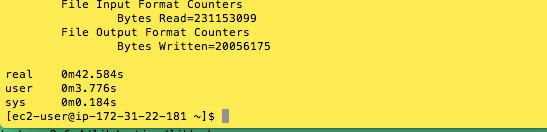
Repeat the steps for wordcount using bioproject.xml from Assignment 1 and submit screenshots of running it.

hadoop fs -put bioproject.xml /data/



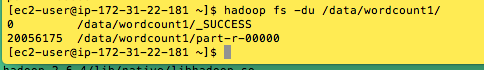
Report the time:

time hadoop jar hadoop-2.6.4/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.4.jar wordcount /data/bioproject.xml /data/wordcount1



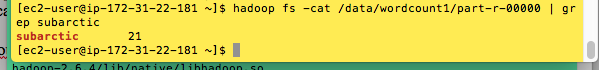
Report the size of the wordcount file

hadoop fs -du /data/wordcount1/



Report the count of grep the word “subarctic”

hadoop fs -cat /data/wordcount1/part-r-00000 | grep subarctic



# Part 2: Hive

Run the following five (1.1, 1.2, 1.3 and 2.1, 2.2) queries in Hive and record the time they take to execute: <http://rasinsrv07.cstcis.cti.depaul.edu/CSC555/SSBM1/SSBM_queries.sql>

<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>

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 '|';

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

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 '|';

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

create table part (

p\_partkey int,

p\_name varchar(22),

p\_mfgr varchar(6),

p\_category varchar(7),

p\_brand1 varchar(9),

p\_color varchar(11),

p\_type varchar(25),

p\_size int,

p\_container varchar(10)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|';

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

create table supplier (

s\_suppkey int,

s\_name varchar(25),

s\_address varchar(25),

s\_city varchar(10),

s\_nation varchar(15),

s\_region varchar(12),

s\_phone varchar(15)

) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|';

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

1.1)

select lo\_orderdate, sum(lo\_extendedprice) as revenue

from lineorder, dwdate

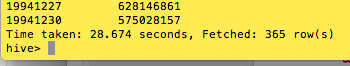
where lo\_orderdate = d\_datekey

and d\_year = 1994

and lo\_discount between 1 and 3

and lo\_quantity < 25

GROUP BY lo\_orderdate;



Ans: 28.674 sec.

1.2)

select sum(lo\_extendedprice) as revenue

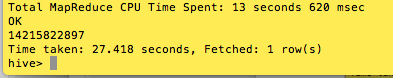
from lineorder, dwdate

where lo\_orderdate = d\_datekey

and d\_yearmonth = 'Jan1993'

and lo\_discount between 5 and 6

and lo\_quantity between 25 and 35;



Ans: 27.418 sec.

1.3)

select sum(lo\_extendedprice) as revenue

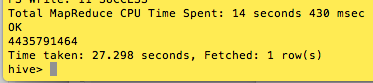
from lineorder, dwdate

where lo\_orderdate = d\_datekey

and d\_weeknuminyear = 6 and d\_year = 1994

and lo\_discount between 5 and 8

and lo\_quantity between 36 and 41;



Ans: 27.298 sec.

2.1)

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

from lineorder, dwdate, part, supplier

where lo\_orderdate = d\_datekey

and lo\_partkey = p\_partkey

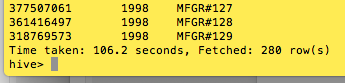
and lo\_suppkey = s\_suppkey

and p\_category = 'MFGR#12'

and s\_region = 'AMERICA'

group by d\_year, p\_brand1

order by d\_year, p\_brand1;



Ans: 106.2 sec.

2.2)

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

from lineorder, dwdate, part, supplier

where lo\_orderdate = d\_datekey

and lo\_partkey = p\_partkey

and lo\_suppkey = s\_suppkey

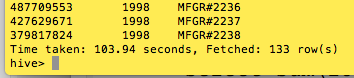
and p\_brand1 between 'MFGR#2221'

and 'MFGR#2238'

and s\_region = 'ASIA'

group by d\_year, p\_brand1

order by d\_year, p\_brand1;



Ans: 103.94 sec.

# Part 3: Pig

Convert and load the data into Pig, implementing queries 0.1, 0.2, 0.3. You only need to do all of the queries for Pig. Check disk storage, if your disk usage is over 90% Pig may hang without an error or a warning.

One easy way to time Pig is as follows: put your sequence of pig commands 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 inform you how long the pig script took to run).

--Q0.1

SELECT AVG(lo\_revenue)

FROM lineorder;

Parse the followings on the script and then execute in the PIG directory

lod = LOAD '/user/ec2-user/lineorder.tbl' USING PigStorage('|')

AS (lo\_orderkey :float,

lo\_linenumber :float,

lo\_custkey :float,

lo\_partkey :float,

lo\_suppkey :float,

lo\_orderdate :float,

lo\_orderpriority :chararray,

lo\_shippriority : chararray,

lo\_quantity :float,

lo\_extendedprice :float,

lo\_ordertotalprice :float,

lo\_discount :float,

lo\_revenue :float,

lo\_supplycost :float,

lo\_tax :float,

lo\_commitdate :float,

lo\_shipmode : chararray);

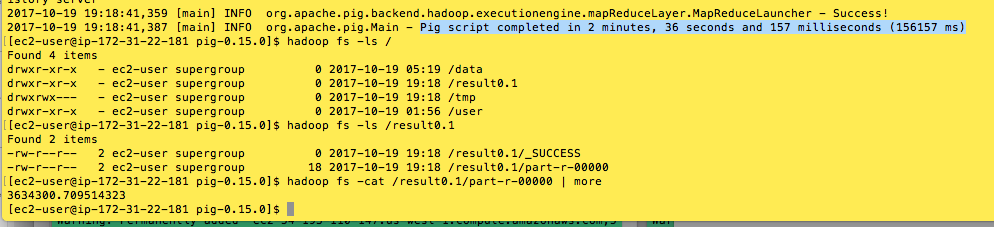
by\_all= group lod ALL;

all\_avg = FOREACH by\_all GENERATE AVG( lod.lo\_revenue);

Dump all\_avg;

STORE all\_avg INTO '/result0.1' USING PigStorage ('\t');

Output:



Ans: 2min 36sec 157 ms was taken

--Q0.2

SELECT lo\_discount, COUNT(lo\_extendedprice)

FROM lineorder

GROUP BY lo\_discount;

Parse the followings on the script and then execute in the PIG directory

lod = LOAD '/user/ec2-user/lineorder.tbl' USING PigStorage('|')

AS (lo\_orderkey :float,

lo\_linenumber :float,

lo\_custkey :float,

lo\_partkey :float,

lo\_suppkey :float,

lo\_orderdate :float,

lo\_orderpriority :chararray,

lo\_shippriority : chararray,

lo\_quantity :float,

lo\_extendedprice :float,

lo\_ordertotalprice :float,

lo\_discount :float,

lo\_revenue :float,

lo\_supplycost :float,

lo\_tax :float,

lo\_commitdate :float,

lo\_shipmode : chararray);

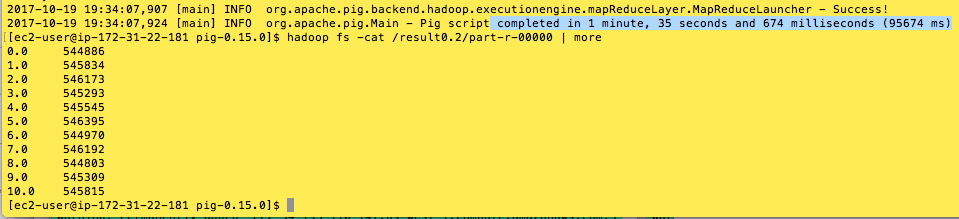
by\_discount = group lod by lo\_discount;

discount\_count = FOREACH by\_discount GENERATE group as lo\_discount, COUNT(lod);

dump discount\_count;

store discount\_count into '/result0.2' using PigStorage('\t');

Output:



Ans: 1min 35sec 674ms

--Q0.3 Added simple test query

SELECT lo\_quantity, SUM(lo\_revenue)

FROM lineorder

WHERE lo\_discount < 3

GROUP BY lo\_quantity;

Parse the followings on the script and then execute in the PIG directory

lod = LOAD '/user/ec2-user/lineorder.tbl' USING PigStorage('|')

AS (lo\_orderkey :float,

lo\_linenumber :float,

lo\_custkey :float,

lo\_partkey :float,

lo\_suppkey :float,

lo\_orderdate :float,

lo\_orderpriority :chararray,

lo\_shippriority : chararray,

lo\_quantity : chararray,

lo\_extendedprice :float,

lo\_ordertotalprice :float,

lo\_discount :float,

lo\_revenue :float,

lo\_supplycost :float,

lo\_tax :float,

lo\_commitdate :float,

lo\_shipmode : chararray);

LowDis = filter lod by lo\_discount < 3;

lod = foreach lod generate lo\_quantity as qua, lo\_revenue as rev;

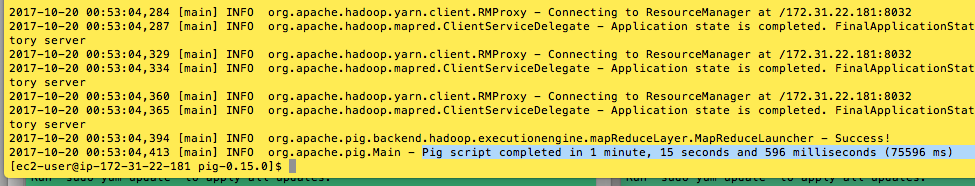
by\_lo\_quantity = group lod by qua;

sumTab = FOREACH by\_lo\_quantity GENERATE group as qua, SUM(lod.rev);

dump sumTab;

store sumTab into '/result0.3' using PigStorage('\t');

Output:



Ans: 1min 15sec 596ms was taken

# Part 4: Hadoop Streaming

Implement query **0.3** using Hadoop streaming with python (you may implement this part in Java if you prefer).

SELECT lo\_quantity, SUM(lo\_revenue)

FROM lineorder

WHERE lo\_discount < 3

GROUP BY lo\_quantity;

Ans:

Python Code (Mapper): col8: lo\_quantity, col12: lo\_revenue, col11: lo\_discount

#!/usr/bin/python

import sys

for line in sys.stdin:

    words=line.strip()

    words=words.split('|')

    print '%s\t%s\t%s' % (words[8], words[12], words[11])

Python Code (Reducer): col0: lo\_quantity, col1: lo\_revenue, col2: lo\_discount

#!/usr/bin/env python

import sys

qua=None #Preset the quantity is None

rev=None #Preset the revenue is None

for line in sys.stdin:

    words=line.strip()

    words=words.split('\t')

    if float(words[2]) < 3 :

        if qua == None:

            qua = words[0]

            rev = float(words[1])

        elif words[0] != qua :

            print '%s\t%s' % (qua, rev)

    qua = words[0]

    rev = float(words[1])

elif words[0] == qua:

    rev += float(words[1])

print '%s\t%s' % (qua, rev)

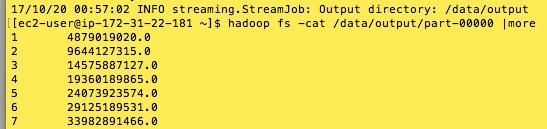
Testing in Terminal

[ec2-user@ip-172-31-22-181 ~]$ cat lineorder.tbl | python project\_mapper1.py | sort -n | python project\_reducer1.py

Hadoop streaming Code

hadoop jar /home/ec2-user/hadoop-2.6.4/share/hadoop/tools/lib/hadoop-streaming-2.6.4.jar -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFieldBasedComparator -D mapred.text.key.comparator.options=-n -input /projectData -output /data/output -mapper project\_mapper1.py -reducer project\_reducer1.py -file project\_reducer1.py -file project\_mapper1.py

Output:



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.