## Student: Huynh Vo

#I would love more time

## HU Extension Assignment 06 E63 Big Data Analytics

### Handed out: 10/06/2017 Due by 4:00 PM EST, 10/14/2017

In your solution, please leave the text of every problem as presented here. Add your solution below the problem statement. It is important for us and we will take points if you ignore this request. Please make sure that you provide numeric or textual results of your calculations. If there are no results, we will treat the problem as not addressed. Just providing some code without results will give you 0 point.

**Problem 1)** Lecture notes contain script network-count.py in both Spark Streaming API and Spark Structured Streaming API. Use Linux nc (NetCat) utility to demonstrate that scripts work. Run both scripts on your own VM with Spark 2.2 installation. Cloudera VM with Spark 1.6 does not have Spark Structured Streaming API.

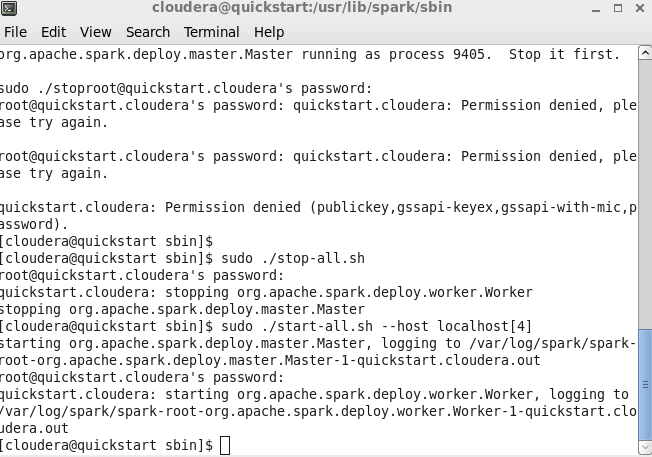
(20%)

#Start master and slave:

[cloudera@quickstart opt]$ cd /usr/lib/spark/sbin

[cloudera@quickstart sbin]$ sudo ./stop-all.sh

[cloudera@quickstart sbin]$ sudo ./start-all.sh --host localhost[4]



On another Linux window, type:

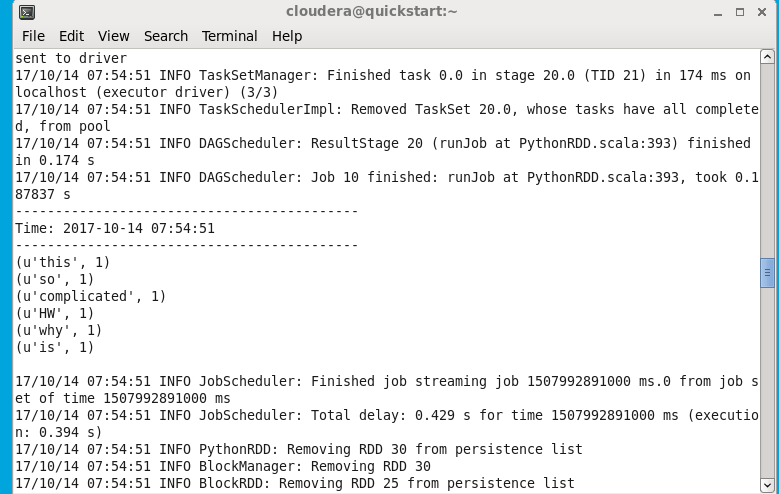
[cloudera@quickstart ~]$ nc -lk 9999

why is this HW so complicated

Screen%20Shot%202017-10-14%20at%2010.51.44%20AM.png

On another Linux window, type:

$ spark-submit --master local[4] network-count.py localhost 9999



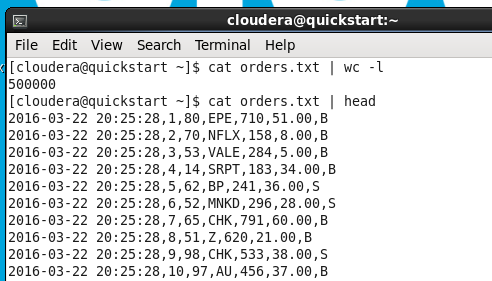
#I’m not sure whats going on with my computer, it keeps running, is it because its streaming? I followed lecture 6 and lab 6.

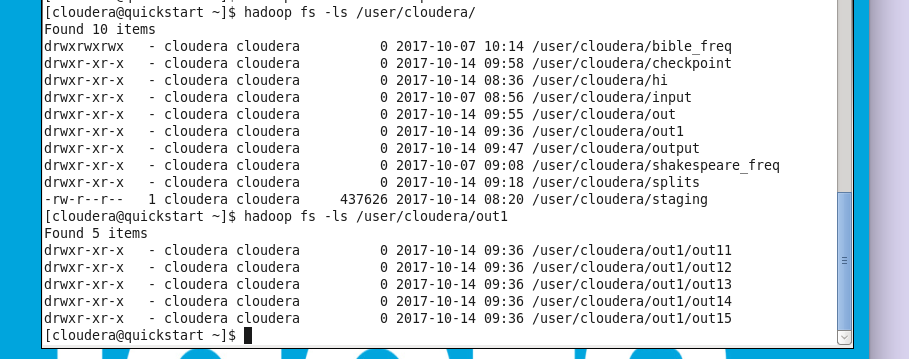
**Problem 2)** Expand provide orders.tar.gz file. Also, download shell scrips splitAndSend.original.sh and splitAndSend.sh and python script count-buys.py. First run splitAndSend.original.sh and count-buys.py. Record the failure mode of count-buys.py. Simply read the error message produced and tell us what is happening. Then run script splitAndSend.sh and Python program count-buys.py and tell us what the results are. In both cases show use contents of your HDFS directories input, output and staging. Subsequently run script splitAndSend.sh which is supposed to reduce or eliminate the race condition and Python program count-buys.py again and tell us what new results are. You might want to rename HDFS directory output from the first run in order to preserve it’s content. In both cases show use contents of your HDFS directories input, output and staging. Locate an output file named part-00000 that is not empty and show its content to us. Run these experiments on Cloudera VM. You need HDFS for these programs to run.

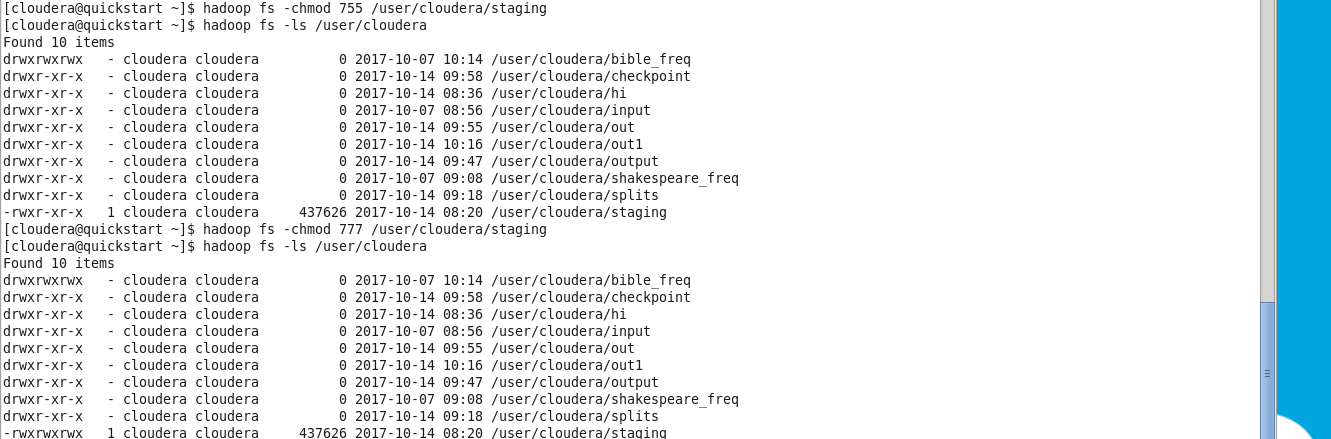
(30%)

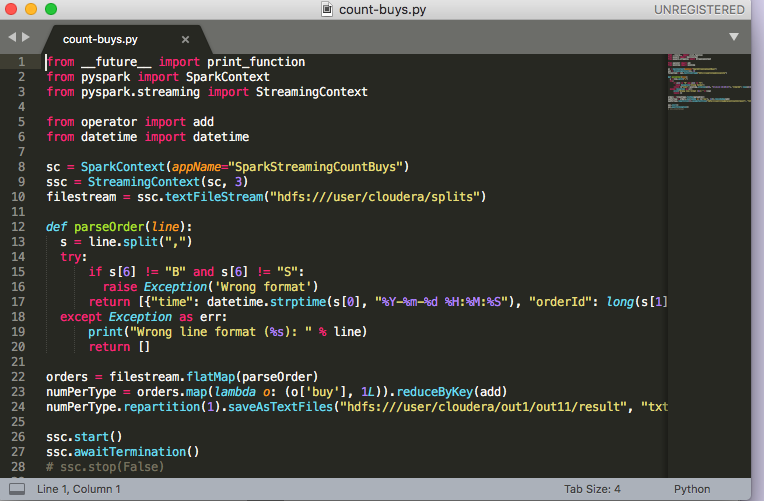
[cloudera@quickstart ~]$ cat orders.txt | wc –l

[cloudera@quickstart ~]$ cat orders.txt | head



#In user/cloudera, I have below directories





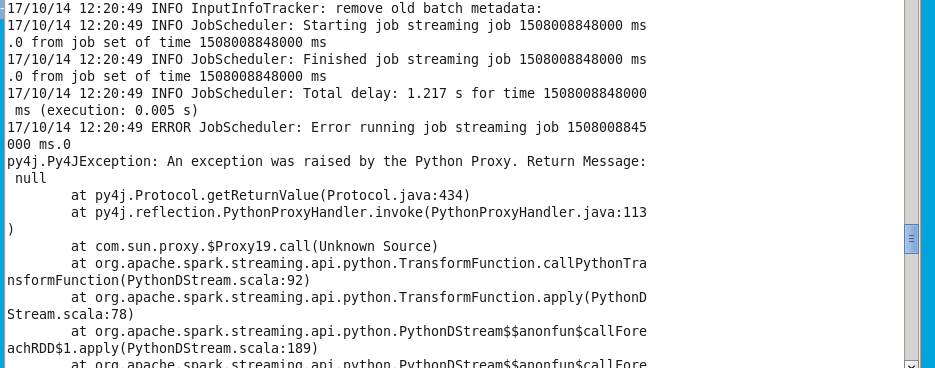
# In a separate linux terminal:

[cloudera@quickstart ~]$ spark-submit --master local[4] count-buys.py

# In a different linux terminal:

[cloudera@quickstart ~]$ chmod +x splitAndSend.original.sh

[cloudera@quickstart ~]$ ./splitAndSend.original.sh splits



#Error message: 17/10/14 12:20:49 INFO JobScheduler: Total delay: 1.217 s for time 1508008848000 ms (execution: 0.005 s)

17/10/14 12:20:49 ERROR JobScheduler: Error running job streaming job 1508008845000 ms.0

py4j.Py4JException: An exception was raised by the Python Proxy. Return Message: null

at py4j.Protocol.getReturnValue(Protocol.java:434)

at py4j.reflection.PythonProxyHandler.invoke(PythonProxyHandler.java:113)

at com.sun.proxy.$Proxy19.call(Unknown Source)

Caused by: java.io.FileNotFoundException: File does not exist: /user/cloudera/splits/chunkai.\_COPYING\_

The error results no file is found in /user/cloudera/out1/out11.

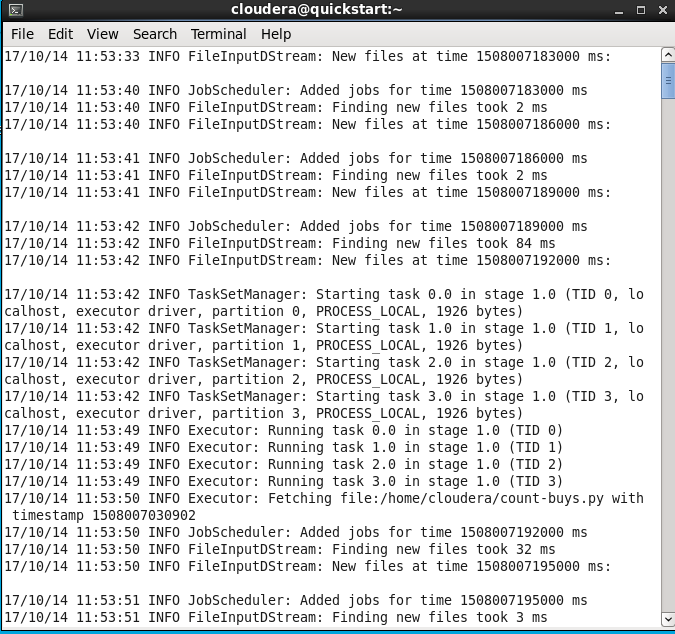
# In a separate linux terminal:

[cloudera@quickstart ~]$ spark-submit --master local[4] count-buys.py

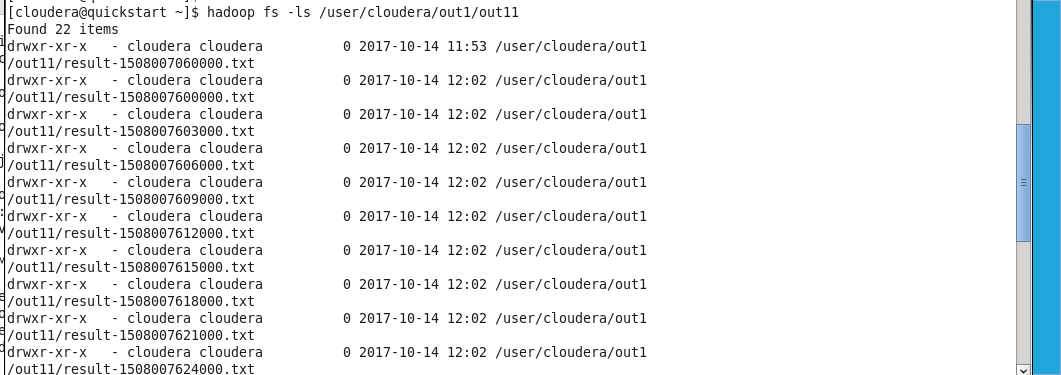
# In a different linux terminal:

[cloudera@quickstart ~]$ chmod +x splitAndSend.sh

[cloudera@quickstart ~]$ ./splitAndSend.sh splits



#Check to see the files



With proper .sh, we found files in /user/cloudera/out1/out11

**Problem 3).** In the second run of the previous problem will notice that many of part-00000 files in your output directory are empty. Could you explain why.

(10%)

[cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/out1/out11/result-1508007621000.txt

{[False, 4970L}

{True, 5030L}

Explanation:

part-00000 is one of the chunks after file orders.txt is splitted into multiple chunks

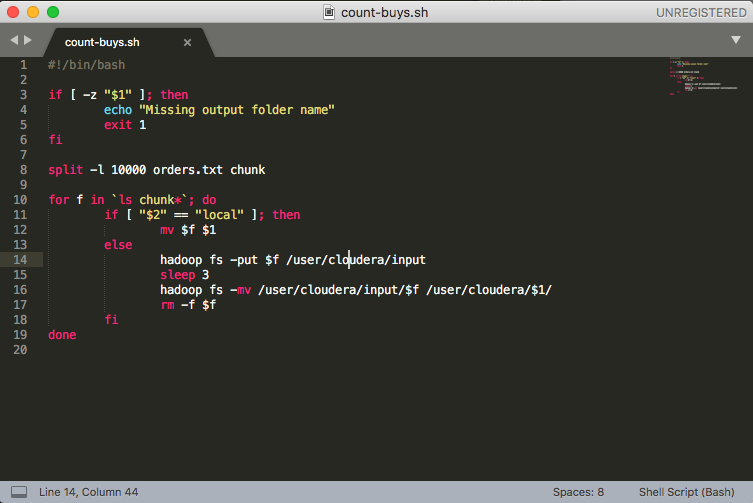
Some chunks are empty because we want the file to split every 3 seconds, and sometimes during that 3 seconds, we don’t have 10000 yet.

It could also because there was no one bought.

It is empty because nothing is making out of the map task.

**Problem 4)** Could you rewrite count-buys.sh in Spark Structured Streaming API. If you do that change script splitAndSend.sh to move generated chunks from the local files system directory staging to local file system directory input. Run this experiment on your VM with Spark 2.2.

(20%)



# In a separate linux terminal:

$spark-submit --master local[4] count-buys.py

# In a different linux terminal:

$chmod +x count-buy.sh

$ ./count-buy.sh splits

**Problem 5)** Examine provided Python program stateful\_wordcount.py. Make it work as is. If there are errors on the code, fix them. Modify the code so that it outputs the number of words starting with letters a and b. Demonstrate that modified program work. You should provide several both positive and negative examples. (20%)

# In a separate linux terminal:

[cloudera@quickstart ~]$ spark-submit --master local[4] stateful\_wordcount.py

# In a different linux terminal:

[cloudera@quickstart ~]$ chmod +x splitAndSend.sh

[cloudera@quickstart ~]$ ./splitAndSend.sh splits