Assignment 5
CSCI E 63 - Big Data Analytics

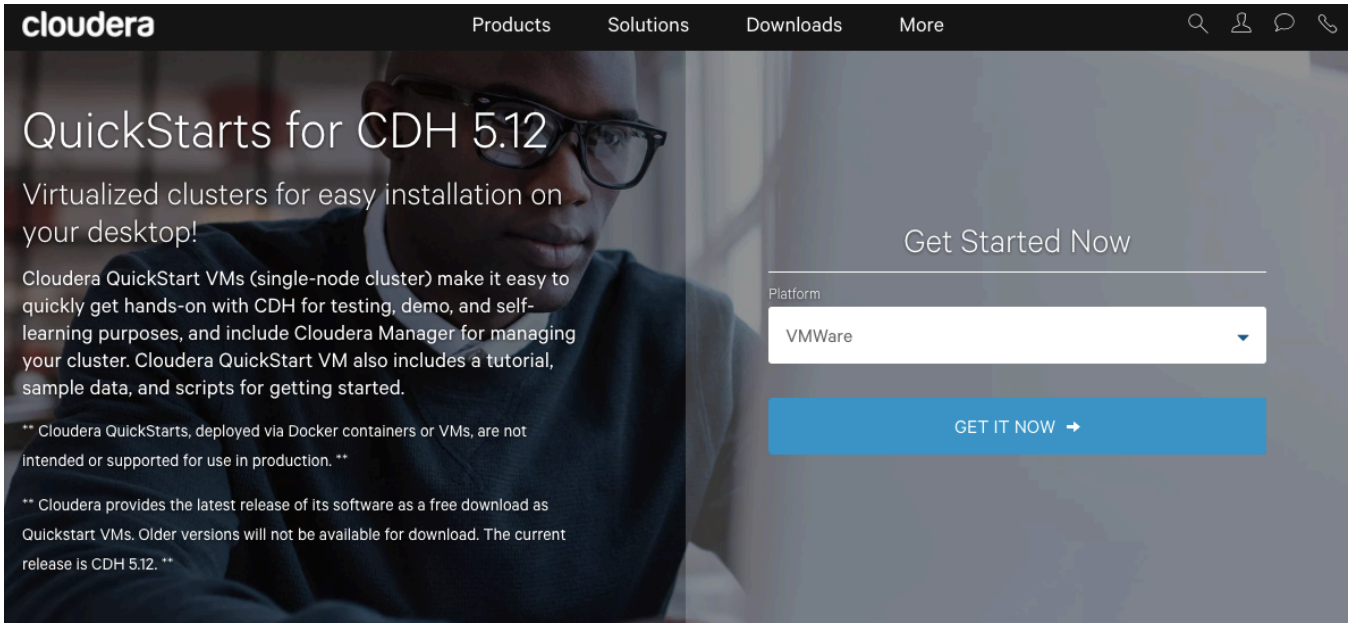
Problem 1.
Download Quick Start VM for CDH 5.12 from https://www.cloudera.com/downloads/quickstart_vms/5-8.html. Start the VM. Please assign to the VM as much memory as you can. Examine whether hadoop-hdfs-* , hadoop-mapreduce-* and hadoop-yarn-* daemons are running. If those daemons are not running start all of them. If any of daemons fails to run, try to fix it.

Answer:

→ Download the VM

Download the Quick Start VM for CDH from:
https://www.cloudera.com/downloads/quickstart_vms/5-12.html

Select Platform: VMWare

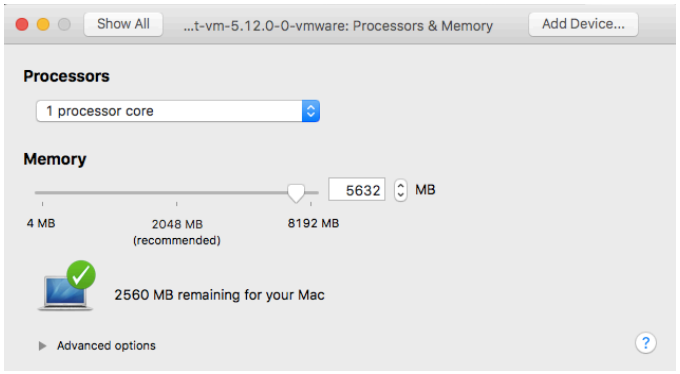


Extract the zip file contents.

→ Start the VM

Using VMWare Fusion, go to File -> Open... and select the .vmx file you just extracted. It will import the VM.

Assign as much memory as you can (Minimum 4GB).



Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

You should be able to start it now. It takes a few minutes so be patient.

→ Examine whether hadoop-hdfs-* , hadoop-mapreduce-* and hadoop-yarn-* daemons are running.

From the terminal, run the commands:

for x in `cd /etc/init.d ; ls hadoop-hdfs-*`; do sudo service $x status ; done
for x in `cd /etc/init.d ; ls hadoop-mapreduce-*` ; do sudo service $x status ; done
for x in `cd /etc/init.d ; ls hadoop-yarn-*` ; do sudo service $x status ; done

```
[cloudera@quickstart init.d]$ for x in `cd /etc/init.d ; ls hadoop-hdfs-*`; do sudo service $x status ; done
Hadoop datanode is running                                 [  OK  ]
Hadoop journalnode is running                              [  OK  ]
Hadoop namenode is running                                 [  OK  ]
Hadoop secondarynamenode is running                        [  OK  ]
```

```
[cloudera@quickstart init.d]$ for x in `cd /etc/init.d ; ls hadoop-mapreduce-*` ; do sudo service $x status ; done
Hadoop historyserver is running                            [  OK  ]
```

```
[cloudera@quickstart init.d]$ for x in `cd /etc/init.d ; ls hadoop-yarn-*` ; do sudo service $x status ; done
Hadoop nodemanager is running                              [  OK  ]
Hadoop proxyserver is dead and pid file exists             [FAILED]
Hadoop resourcemanager is running                          [  OK  ]
```

Understanding the commands:

Run everything in between ‘’ and pass the output(ls response) to the calling environment.
The list of names will be names of services.

```
[cloudera@quickstart ~]$ cd /etc/init.d
[cloudera@quickstart init.d]$ ls hadoop-hdfs-*
hadoop-hdfs-datanode  hadoop-hdfs-journalnode  hadoop-hdfs-namenode  hadoop-hdfs-secondarynamenode
```

Then run each service one by one.

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

Fixing the failure:
Sometimes stopping and starting the service is enough. That can be done by replacing 'status' in the above commands with stop/start.
In this case, that will not work.

Do the following:

cd /etc/hadoop/conf.pseudo
ls
core-site.xml hadoop-metrics.properties log4j.properties README hadoop-env.sh hdfs-site.xml
mapred-site.xml yarn-site.xml
sudo vi yarn-site.xml

To yarn-site.xml file, add the following property:

```
<property>
        <description>web proxy </description> <name>yarn.web-proxy.address</name>
        <value>localhost:3122</value>
</property>
```

for x in `cd /etc/init.d ; ls hadoop-yarn-proxy*` ; do sudo service $x start ; done

```
starting proxyserver, logging to /var/log/hadoop-yarn/yarn-yarn-proxyserver-quickstart.cloudera.out
Started Hadoop proxyserver:                                [  OK  ]
```

```
[cloudera@quickstart ~]$ for x in `cd /etc/init.d ; ls hadoop-yarn-*` ; do sudo service $x status ; done
Hadoop nodemanager is running                              [  OK  ]
Hadoop proxyserver is running                              [  OK  ]
Hadoop resourcemanager is running                          [  OK  ]
```

The Hadoop environment is now running fine.

Assignment 5
CSCI E 63 - Big Data Analytics

Problem 2.
Examine whether there are HDFS home directories for users: spark, hive, oozie, and cloudera. If the directories are present, find the content of those directories. If the directories are not present, create them. Please do not format the namenode.

Answer:

→ To see all directories in HDFS, run command:
sudo –u hdfs hdfs dfs –ls /user/

```
[cloudera@quickstart conf.pseudo]$ sudo -u hdfs hdfs dfs -ls /user/
Found 8 items
drwxr-xr-x   - cloudera cloudera            0 2017-07-19 06:28 /user/cloudera
drwxr-xr-x   - mapred   hadoop              0 2017-07-19 06:29 /user/history
drwxrwxrwx   - hive     supergroup          0 2017-07-19 06:31 /user/hive
drwxrwxrwx   - hue      supergroup          0 2017-07-19 06:30 /user/hue
drwxrwxrwx   - jenkins  supergroup          0 2017-07-19 06:29 /user/jenkins
drwxrwxrwx   - oozie    supergroup          0 2017-07-19 06:30 /user/oozie
drwxrwxrwx   - root     supergroup          0 2017-07-19 06:29 /user/root
drwxr-xr-x   - hdfs     supergroup          0 2017-07-19 06:31 /user/spark
```

→ HDFS home directories are present for users:
- spark
- hive
- oozie
- cloudera

→ To check the contents of these directories, run commands:
sudo -u hdfs hdfs dfs -ls -R /user/spark

```
[root@quickstart ~]# sudo -u hdfs hdfs dfs -ls -R /user/spark
drwxrwxrwx   - spark supergroup          0 2017-10-05 16:41 /user/spark/applicationHistory
```

sudo -u hdfs hdfs dfs -ls -R /user/hive

```
[root@quickstart ~]# sudo -u hdfs hdfs dfs -ls -R /user/hive
drwxrwxrwx   - hive supergroup          0 2017-07-19 06:31 /user/hive/warehouse
```

sudo -u hdfs hdfs dfs -ls -R /user/oozie
(Output: Too many contents to attach)

sudo -u hdfs hdfs dfs -ls -R /user/cloudera
(Output: No content)

Assignment 5
CSCI E 63 - Big Data Analytics

Problem 3. Create new Linux user smith. Make that user a member of the mapred Linux group. Make that user a sudo user. Create the home directory of user smith in HDFS. Download provided files bible.tar and shakespeare.tar. Unzip both tar files and copy the resulting files into HDFS directory input of user smith. As user smith run Hadoop grep on both bible and shakespeare texts. Every Hadoop run requires separate output directory. Examine content of first 20 lines of files generated by Hadoop grep.

Answer:

→ To create a new Linux user 'smith' of the mapred linux group:

1. Log in as *root*, or type command:
$ su -
And enter root's password (default is cloudera)

2. Now, as user *root*, type:
$ useradd -g mapred smith

Note: User running Hadoop MapReduce programs must be a member of mapred group.

3. To create password for new user, type :
$ passwd smith

```
[root@quickstart ~]# useradd -g mapred smith
[root@quickstart ~]# passwd smith
```

→ To create the HDFS home directories for user smith,

1. To create directory, type:
$ sudo -u hdfs hadoop fs -mkdir /user/smith

2. To grant the ownership to smith, type:
$ sudo -u hdfs hadoop fs -chown smith:mapred /user/smith

3. To give full read-write-execute right on the directory, type:
$ sudo -u hdfs hadoop fs -chmod 1777 /user/smith

```
[root@quickstart ~]# sudo -u hdfs hadoop fs -mkdir /user/smith
[root@quickstart ~]# sudo -u hdfs hadoop fs -chown smith:mapred /user/smith
[root@quickstart ~]# sudo -u hdfs hadoop fs -chmod 1777 /user/smith
```

Assignment 5
CSCI E 63 - Big Data Analytics

4. To give sudo privileges:
    a. Login as root or su to get root prompt
    b. Type visudo
    c. And editor will open. Find a line that says:
        root ALL=(ALL) ALL
    d. Add one with username smith below that
        smith ALL=(ALL) ALL

→ Copy the unzipped Bible and Shakespeare files into HDFS directory input of user smith

To create directory input for user smith, run command:
sudo -u hdfs hadoop fs -mkdir /user/smith/input

```
[smith@quickstart root]$ hadoop fs -ls
Found 1 items
drwxr-xr-x   - hdfs mapred          0 2017-10-05 17:56 input
```

To copy the Bible and Shakespeare files into the input directory from local, run commands:
hadoop fs -copyFromLocal /home/cloudera/Documents/all-bible /user/smith/input/bible
hadoop fs -copyFromLocal /home/cloudera/Documents/all-shakespeare /user/smith/input/shakespeare

```
[smith@quickstart root]$ hadoop fs -copyFromLocal /home/cloudera/Documents/all-bible /user/smith/input/bible
[smith@quickstart root]$ hadoop fs -copyFromLocal /home/cloudera/Documents/all-shakespeare /user/smith/input/shakespeare
[smith@quickstart root]$ sudo -u hdfs hdfs dfs -ls -R /user/smith/input
-rw-r--r--   1 smith mapred    5258688 2017-10-05 18:15 /user/smith/input/bible
-rw-r--r--   1 smith mapred    5284231 2017-10-05 18:15 /user/smith/input/shakespeare
```

→ As user smith run Hadoop grep on both bible and shakespeare texts.

Run commands:
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar grep input/bible bible_freq '\w+'

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar grep input/shakespeare shakespeare_freq '\w+'

Check files created:

```
[smith@quickstart ~]$ hadoop fs -ls -R
drwxr-xr-x   - smith mapred          0 2017-10-05 18:57 bible_freq
-rw-r--r--   1 smith mapred          0 2017-10-05 18:57 bible_freq/_SUCCESS
-rw-r--r--   1 smith mapred     147360 2017-10-05 18:57 bible_freq/part-r-00000
drwxr-xr-x   - hdfs  mapred          0 2017-10-05 18:38 input
-rw-r--r--   1 smith mapred    5235872 2017-10-05 18:38 input/bible
-rw-r--r--   1 smith mapred    5284231 2017-10-05 18:15 input/shakespeare
drwxr-xr-x   - smith mapred          0 2017-10-05 19:01 shakespeare_freq
-rw-r--r--   1 smith mapred          0 2017-10-05 19:01 shakespeare_freq/_SUCCESS
-rw-r--r--   1 smith mapred     299379 2017-10-05 19:01 shakespeare_freq/part-r-00000
```

Assignment 5
CSCI E 63 - Big Data Analytics

→ Examine content of first 20 lines of files generated by Hadoop grep.

hadoop fs -cat bible_freq/part-r-00000 | head -20

```
[smith@quickstart ~]$ hadoop fs -cat bible_freq/part-r-00000 | head -20
62229   the
38916   and
34541   of
13452   to
12846   And
12590   that
12388   in
9762    shall
9669    he
8940    unto
8854    I
8385    his
8000    a
7249    for
6972    they
6895    be
6858    is
6649    him
6647    LORD
6572    not
```

hadoop fs -cat shakespeare_freq/part-r-00000 | head -20

```
[smith@quickstart ~]$ hadoop fs -cat shakespeare_freq/part-r-00000 | head -20
25578   the
23027   I
19654   and
17462   to
16444   of
13524   a
12697   you
11296   my
10699   in
8857    is
8851    that
8402    not
8033    me
8020    s
7800    And
7231    with
7165    it
6812    his
6753    be
6246    your
```

Assignment 5
CSCI E 63 - Big Data Analytics

Problem 4. Create your own version of "Hadoop grep" program using Spark. Compare your results with the results of Hadoop grep when applied to the texts of King James Bible, and all of Shakespeare's works, contained in files bible.tar and shakespear.tar respectively. Notice small differences between results obtained by your Spark program and Hadoop grep. Try to explain what causes those differences. Save results of your Spark grep operations both in HDFS and on your local file system. You can implement your solution using one of interactive shells or a standalone program.

Answer:

→ Setup the environment

As root user, change .bash_profile file and add:

```
export JAVA_HOME=/usr/java/jdk1.7.0_67-cloudera
export SPARK_HOME=/usr/lib/spark
export HADOOP_MAPRED_HOME=/usr/lib/hadoop-mapreduce
export PATH=$PATH:$JAVA_HOME/bin:$SPARK_HOME/bin
```

Build using command: source .bash_profile

```
[smith@quickstart root]$ echo $JAVA_HOME
/usr/java/jdk1.7.0_67-cloudera
[smith@quickstart root]$ echo $SPARK_HOME
/usr/lib/spark
```

To find and create log4j.properties, run commands:

find . -name log4j.properties.template -print

```
[root@quickstart /]# find . -name log4j.properties.template -print
./etc/spark/conf.dist/log4j.properties.template
```

sudo cp log4j.properties.template log4j.properties

Replace INFO, WARN with ERROR

To start Spark-Master, run commands:

cd /usr/lib/spark/sbin

./start-master.sh

```
[root@quickstart conf.dist]# cd /usr/lib/spark/sbin
[root@quickstart sbin]# sudo ./start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /var/log/spark/spark-root-org.apache.spark
1-quickstart.cloudera.out
[root@quickstart sbin]# su smith
[smith@quickstart sbin]$ cd ~
[smith@quickstart ~]$ pyspark
```

Assignment 5
CSCI E 63 - Big Data Analytics

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Python version 2.6.6 (r266:84292, Jul 23 2015 15:22:56)
SparkContext available as sc, HiveContext available as sqlContext.
```

→ Create your own version of "Hadoop grep" program using Spark.

Read the file using on of the two options:

Read from HDFS
Note: The default file location is in the HDFS home directory of the current user i.e. cloudera

```
>>> bible = sc.textFile("/user/smith/input/bible")
>>> bible.count()
116634
>>> shakespeare = sc.textFile("/user/smith/input/shakespeare")
>>> shakespeare.count()
173126
```

Read from local file

```
>>> bible = sc.textFile("file:///home/cloudera/Documents/all-bible")
>>> bible.count()
116634
>>> shakespeare = sc.textFile("file:///home/cloudera/Documents/all-shakespeare")
>>> shakespeare.count()
173126
```

Get word counts for the two files:
Run commands:

```
>>> from operator import add
>>> bibleWordCounts = bible.flatMap(lambda x:x.split(" ")).map(lambda x: (x,1)).reduceByKey(add)
>>> bibleExchanged = bibleWordCounts.map(lambda x: (x[1],x[0]))
>>> bibleSorted = bibleExchanged.sortByKey(False)
```

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

```
>>> shakespeareWordCounts = shakespeare.flatMap(lambda x:x.split(" ")).map(lambda x: (x, 1)).reduceByKey(add)
>>> shakespeareExchanged = shakespeareWordCounts.map(lambda x: (x[1],x[0]))
>>> shakespeareSorted = shakespeareExchanged.sortByKey(False)
```

Verify the RDDs:

```
>>> bibleSorted.take(20)
```

```
>>> bibleSorted.take(20)
[(604754, u''), (62221, u'the'), (38643, u'and'), (34505, u'of'), (13435, u'to'), (12735, u'And'), (12465, u'that'), (12223, u'in
'), (9762, u'shall'), (9511, u'he'), (8930, u'unto'), (8708, u'I'), (8362, u'his'), (7998, u'a'), (7162, u'for'), (6895, u'they')
, (6736, u'be'), (6721, u'is'), (5999, u'with'), (5859, u'not')]
```

```
>>> shakespeareSorted.take(20)
```

```
>>> shakespeareSorted.take(20)
[(64531, u''), (25069, u'the'), (18793, u'and'), (16436, u'to'), (16069, u'of'), (15223, u'I'), (12982, u'a'), (11180, u'my'), (10
134, u'in'), (9109, u'you'), (8109, u'is'), (7773, u'that'), (7123, u'not'), (7001, u'with'), (6594, u'his'), (6202, u'be'), (6119
, u'your'), (5955, u'\tAnd'), (5781, u'for'), (5311, u'have')]
```

→ Notice small differences between results obtained by your Spark program and Hadoop grep. Try to explain what causes those differences.

The differences are caused by case differences, trailing spaces and special characters. Hadoop grep uses regex to count just the words. If we make all lower case and use regex to keep just the words, the count will match exactly.

→ Save results of your Spark grep operations both in HDFS and on your local file system.

To save spark objects to HDFS, run commands:

```
>>> bibleSorted.saveAsTextFile("hdfs:///user/smith/output/biblesorted")
>>> shakespeareSorted.saveAsTextFile("hdfs:///user/smith/output/shakespearesorted")
```

To save spark objects to local file system, run commands:

```
>>> bibleSorted.saveAsTextFile("file:///home/smith/output/biblesorted")
>>> shakespeareSorted.saveAsTextFile("file:///home/smith/output/shakespearesorted")
```

→ Verify the outputs:

To verify the outputs on HDFS, run commands:

```
hadoop fs -cat output/biblesorted/part-00000 | head -20
hadoop fs -cat output/shakespearesorted/part-00000 | head -20
```

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

```
[smith@quickstart ~]$ hadoop fs -cat output/biblesorted/part-00000 | head -20
(604754, u'')
(62221, u'the')
(38643, u'and')
(34505, u'of')
(13435, u'to')
(12735, u'And')
(12465, u'that')
(12223, u'in')
(9762, u'shall')
(9511, u'he')
(8930, u'unto')
(8708, u'I')
(8362, u'his')
(7998, u'a')
(7162, u'for')
(6895, u'they')
(6736, u'be')
(6721, u'is')
(5999, u'with')
(5859, u'not')
cat: Unable to write to output stream.
[smith@quickstart ~]$ hadoop fs -cat output/shakespearesorted/part-00000 | head -20
(64531, u'')
(25069, u'the')
(18793, u'and')
(16436, u'to')
(16069, u'of')
(15223, u'I')
(12982, u'a')
(11180, u'my')
(10134, u'in')
(9109, u'you')
(8109, u'is')
(7773, u'that')
(7123, u'not')
(7001, u'with')
(6594, u'his')
(6202, u'be')
(6119, u'your')
(5955, u'\tAnd')
(5781, u'for')
(5311, u'have')
```

To verify the outputs on the local file system, simply navigate to the folder

```
[smith@quickstart output]$ cd /home/smith/output
[smith@quickstart output]$ ls -ltr
total 8
drwxr-xr-x 2 smith mapred 4096 Oct  5 20:54 biblesorted
drwxr-xr-x 2 smith mapred 4096 Oct  5 20:54 shakespearesorted
```

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

You will see that the two generated outputs are folders and not files

cd biblesorted
ls -ltr
cd ..
cd shakespearesorted
ls -ltr

```
total 1060
-rw-r--r-- 1 smith mapred 1081644 Oct  5 20:54 part-00000
-rw-r--r-- 1 smith mapred       0 Oct  5 20:54 _SUCCESS
```

```
total 1700
-rw-r--r-- 1 smith mapred 1738803 Oct  5 20:54 part-00000
-rw-r--r-- 1 smith mapred       0 Oct  5 20:54 _SUCCESS
```

Examine the contents of the file using vi

Assignment 5
CSCI E 63 - Big Data Analytics

Problem 5.
Create your own tables KINGJAMES with columns for words and frequencies and insert into the table the result of your Spark grep program which produces word counts in file bible. Find all words in table KINGJAMES which start with letter "w" and are 4 or more characters long and appear more than 250 times. Write a query that will tell us the number of such words. Before counting turn all words in lower case.
When comparing a word with a string your use LIKE operator, like

word like 'a%' or word like '%th%'

Symbol '%' means any number of characters. You measure the length of a string using function length() and you change the case of a word to all lower characters using function lower().

Answer:

→ Prepare the data
The bibleSorted RDD prepared and saved to file above, contains words in the form u'word'. Before adding it to the table, we need to clean this up.

Run pyspark command:
bibleTablePrep = bibleSorted.map(lambda rec: (rec[0], str(rec[1])))

```
>>> bibleTablePrep = bibleSorted.map(lambda x: " ".join(str(d) for d in x))
>>> bibleTablePrep.take(10)
['604754 ', '62221 the', '38643 and', '34505 of', '13435 to', '12735 And', '12465 that', '12223 in', '9762 shall', '9511 he']
```

Now save this to file using command:
bibleTablePrep.saveAsTextFile("hdfs:///user/cloudera/output/biblesorted_clean")

```
[cloudera@quickstart ~]$ hadoop fs -ls -R /user/cloudera/output/biblesorted_clean
-rw-r--r--   1 cloudera cloudera          0 2017-10-06 18:52 /user/cloudera/output/biblesorted_clean/_SUCCESS
-rw-r--r--   1 cloudera cloudera    1020885 2017-10-06 18:52 /user/cloudera/output/biblesorted_clean/part-00000
```

→ Use the beeline command to open the JDBC client

```
[smith@quickstart ~]$ beeline
Beeline version 1.1.0-cdh5.12.0 by Apache Hive
beeline>
```

→ Connect Beeline to Hive Server using command:
!connect jdbc:hive2://127.0.0.1:10000/default hive cloudera org.apache.hive.jdbc.HiveDriver

```
beeline> !connect jdbc:hive2://127.0.0.1:10000/default hive cloudera org.apache.hive.jdbc.HiveDriver
Connecting to jdbc:hive2://127.0.0.1:10000/default
Connected to: Apache Hive (version 1.1.0-cdh5.12.0)
Driver: Hive JDBC (version 1.1.0-cdh5.12.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://127.0.0.1:10000/default>
```

Assignment 5
CSCI E 63 - Big Data Analytics

This connects using the JDBC driver and provides username/password as hive/cloudera

→ To create the table to accept the data, run command:
create table KINGJAMES (freq INT, word STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ' ' stored as textfile;
describe kingjames

```
+------------+--+
|  tab_name  |
+------------+--+
| kingjames  |
+------------+--+
1 row selected (0.436 seconds)
```

```
+-----------+------------+----------+--+
| col_name  | data_type  | comment  |
+-----------+------------+----------+--+
| freq      | int        |          |
| word      | string     |          |
+-----------+------------+----------+--+
2 rows selected (0.374 seconds)
```

→ To load the from HDFS file system, run command:
LOAD DATA INPATH "/user/cloudera/output/biblesorted_clean/part-00000" INTO TABLE kingjames;

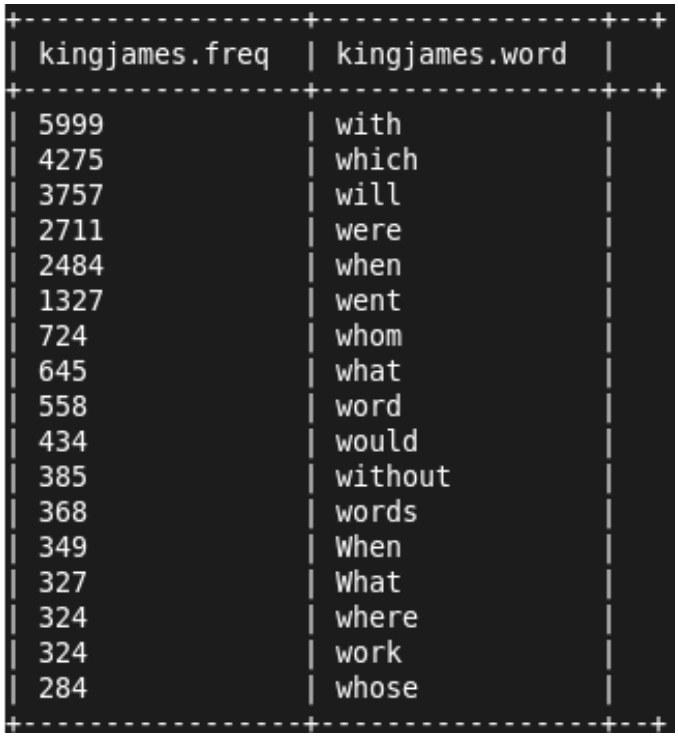
Verify:
select * from kingjames limit 10;

```
+-----------------+-----------------+--+
| kingjames.freq  | kingjames.word  |
+-----------------+-----------------+--+
| 604754          |                 |
| 62221           | the             |
| 38643           | and             |
| 34505           | of              |
| 13435           | to              |
| 12735           | And             |
| 12465           | that            |
| 12223           | in              |
| 9762            | shall           |
| 9511            | he              |
+-----------------+-----------------+--+
```

→ Find all words in table KINGJAMES which start with letter "w" and are 4 or more characters long and appear more than 250 times.
select * from kingjames where lower(word) like 'w%' and length(word) > 4 and freq > 250;

Karan A. Bhandarkar

Assignment 5
CSCI E 63 – Big Data Analytics



kingjames.freq	kingjames.word	
5999	with	
4275	which	
3757	will	
2711	were	
2484	when	
1327	went	
724	whom	
645	what	
558	word	
434	would	
385	without	
368	words	
349	When	
327	What	
324	where	
324	work	
284	whose	

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

Problem 6.
Transfer content of your Hive KINGJAMES table to a Spark DataFrame. Perform the analysis from problem 5 using any available API in Spark. Please note that you are working with Spark 1.6.

Answer:

→ For Spark to locate your hive, do the following:

1. If Hive services are not running, start Hive metadata server
$ hiveserver2 & #
& at the end means "run in the background"

2. Copy hive-site.xml from $HIVE_HOME/conf, which is /etc/hive/conf to $SPARK_HOME/conf, which happens to be /etc/spark/conf
$ sudo cp /etc/hive/conf/hive-site.xml $SPARK_HOME/conf

3. Create HiveContext (in Spark 2, you do not do this, just use spark (SparkSession)
>>> hc = HiveContext(sc)

→ Transfer content of your Hive KINGJAMES table to a Spark DataFrame.
>>> dfs = hc.sql("select * from kingjames")
>>> dfs.count()

```
>>> dfs.count()
50759
```

→ Perform the analysis
>>> from pyspark.sql.functions import length
>>> from pypark.sql.functions import col
>>> dfs.select('word','freq').where(col('word').like("w%")).filter(length(col('word'))>=4).filter(dfs['freq'] > 250).show()

Karan A. Bhandarkar

Assignment 5
CSCI E 63 – Big Data Analytics

word	freq
with	5999
which	4275
will	3757
were	2711
when	2484
went	1327
whom	724
what	645
word	558
would	434
without	385
words	368
where	324
work	324
whose	284

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

Problem 7.
Use Sqoop to transfer the content of MySQL database retail_db which is present on the Cloudera VM into Hive. Demonstrate that new Hive tables are created and correspond to the original MySQL tables. Find the number of rows in each table. Compare those row counts with row counts in MySQL database.

Answer:

→ To use sqoop to import all tables of an existing database schema, at the command prompt, run command:
sqoop import-all-tables -m 1 --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba --password=cloudera --compression-codec=snappy --as-parquetfile --warehouse-dir=/user/hive/warehouse --hive-import

→ Demonstrate that new Hive tables are created and correspond to the original MySQL tables.

To check from sqoop, run command:
sqoop list-tables --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba --password=cloudera

```
categories
customers
departments
order_items
orders
products
```

To check from beeline, run command:
show tables;

```
+--------------+--+
|   tab_name   |
+--------------+--+
| categories   |
| customers    |
| departments  |
| kingjames    |
| order_items  |
| orders       |
| products     |
+--------------+--+
```

To get the list of tables in retail_db in MySQL, run command:
$ mysql --user=retail_dba –p

Assignment 5
CSCI E 63 - Big Data Analytics

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| retail_db          |
+--------------------+
2 rows in set (0.01 sec)

mysql> use retail_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+---------------------+
| Tables_in_retail_db |
+---------------------+
| categories          |
| customers           |
| departments         |
| order_items         |
| orders              |
| products            |
+---------------------+
6 rows in set (0.00 sec)
```

→ Find the number of rows in each table.

Use sqoop eval command to run a SQL query for each table as below:

sqoop eval --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba -password=cloudera --query "SELECT count(*) FROM categories"

categories

```
----------------------
| count(*)           |
----------------------
| 58                 |
----------------------
```

customers

```
----------------------
| count(*)           |
----------------------
| 12435              |
----------------------
```

departments

```
----------------------
| count(*)           |
----------------------
| 6                  |
----------------------
```

order_items

```
----------------------
| count(*)           |
----------------------
| 172198             |
----------------------
```

orders

```
----------------------
| count(*)           |
----------------------
| 68883              |
----------------------
```

products

```
-------------------------
| count(*)              |
-------------------------
| 1345                  |
-------------------------
```

Karan A. Bhandarkar

Assignment 5
CSCI E 63 - Big Data Analytics

→ Compare those row counts with row counts in MySQL database.

```
mysql> select count(*) from categories;
+----------+
| count(*) |
+----------+
|       58 |
+----------+
1 row in set (0.00 sec)

mysql> select count(*) from customers;
+----------+
| count(*) |
+----------+
|    12435 |
+----------+
1 row in set (0.01 sec)

mysql> select count(*) from departments
+----------+
| count(*) |
+----------+
|        6 |
+----------+
1 row in set (0.00 sec)

mysql> select count(*) from order_items
+----------+
| count(*) |
+----------+
|   172198 |
+----------+
1 row in set (0.07 sec)

mysql> select count(*) from orders;
+----------+
| count(*) |
+----------+
|    68883 |
+----------+
1 row in set (0.03 sec)

mysql> select count(*) from products;
+----------+
| count(*) |
+----------+
|     1345 |
+----------+
1 row in set (0.00 sec)
```

Row counts match.

Karan A. Bhandarkar