# Hadoop, Hive & Spark Lab

Personalised version for Yousef Abdalla

Before starting the lab, you will need to connect to the server using either the X2Go or Putty approach. Note that you do not need to repeat the security part – we just need to have a terminal window.

In this document commands/text you type is displayed in *courier italics.*

## Start up the server processes

Note: The 1st two commands take a while to run.

You may also see an ‘ECDSA message and be prompted if you wish to continue connecting’ during the start-up process – enter **Yes**

*start-dfs.sh*

*start-yarn.sh*

*jps*

Output will be similar to that below (numbers may be different)

6944 NodeManager

6337 DataNode

6164 NameNode

7270 Jps

6763 ResourceManager

6605 SecondaryNameNode

## HADOOP Exercises

*cd WordCount1*

*wget http://www.gutenberg.org/files/46/46-0.txt*

*mv 46-0.txt file01*

*hdfs dfs -mkdir -p /user/hduser/input*

*hdfs dfs -copyFromLocal file01 input/file01*

*hdfs dfs -ls input*

Output that will be shown (date/time will differ)

Found 1 items

-rw-r--r-- 1 hduser supergroup 1586488 2020-11-19 09:45 /input/file01

*hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.5.jar wordcount input output*

*hdfs dfs -ls*

*hdfs dfs -ls output*

Output that will be shown (date/time will differ)

Found 2 items

-rw-r--r-- 1 hduser supergroup 0 2020-11-19 09:59 output/\_SUCCESS

-rw-r--r-- 1 hduser supergroup 530683 2020-11-19 09:59 output/part-r-00000

*hdfs dfs -cat output/part-r-00000*

*hdfs dfs -cat output/part-r-00000 | grep -i Peter*

*hdfs dfs -copyToLocal output/part-r-00000 part-r-0000*

*cp /usr/local/hadoop/examples/WordCount.java WordCount.java*

*hadoop com.sun.tools.javac.Main WordCount.java*

*jar cf wc.jar WordCount\*.class*

*hadoop jar wc.jar org.apache.hadoop.examples.WordCount input output\_2*

*hdfs dfs -ls output\_2*

*hdfs dfs -cat output\_2/part-r-00000*

*hdfs dfs -cat output\_2/part-r-00000 | grep -i Peter*

*hdfs dfs -copyToLocal output\_2/part-r-00000 part-r-0000\_2*

## HIVE Exercises

*cd*

*cd WordCount2*

*rm -rf metastore\_db*

*java -jar /usr/local/derby/lib/derbyrun.jar server start &*

hduser@ML-RefVm-45584:~$ Security manager installed using the Basic server security policy.

Apache Derby Network Server - 10.4.2.0 - (689064) started and ready to accept connections on port 1527 at 2020-11-19 09:27:18.429 GMT

*schematool -initSchema -dbType derby*

Output that will be shown (date/time will differ)

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: See h[ttp://www.slf4j.org/codes.html#multiple\_bindings](ttp://www.slf4j.org/codes.html#multiple_bindings ) for an explanation.

SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Metastore connection URL: jdbc:derby://localhost:1527/metastore\_db;create=true

Metastore Connection Driver : org.apache.derby.jdbc.EmbeddedDriver

Metastore connection User: APP

Starting metastore schema initialization to 2.3.0

Initialization script hive-schema-2.3.0.derby.sql

Initialization script completed

schemaTool completed

*hive*

Output that will be shown (date/time will differ)

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl- .6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: See http://www.slf4j.org/codes.html#multiple\_bindings for an explanation.

SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-2.3.7.jar!/hive-log4j2.properties Async: true

Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.

*create database words ;*

*use words ;*

*CREATE TABLE docs(words string);*

*load data inpath 'input/file01' into table docs;*

*CREATE TABLE word\_count AS*

*SELECT word, count(\*) AS count FROM*

*(SELECT explode(split(words, '\\W+')) AS word FROM docs) w*

*GROUP BY word;*

*select \* from word\_count order by count desc limit 10;*

*select \* from word\_count where word = 'Peter';*

*drop table word\_count;*

*drop table docs;*

*drop database words;*

*exit;*

## SPARK Exercises

*cd*

*cd Spark*

*hdfs dfs -ls input*

\*\*\* NOTE: If you DO NOT see the file 'input/file01' listed (it is probable you will not), type the three lines below

*wget http://www.gutenberg.org/files/46/46-0.txt*

*mv 46-0.txt file01*

*hdfs dfs -copyFromLocal file01 input/file01*

*pyspark*

*text\_file = sc.textFile("hdfs://localhost:9000/user/hduser/input/file01")*

*counts = text\_file.flatMap(lambda line: line.split(" ")) \*

*.map(lambda word: (word, 1)) \*

*.reduceByKey(lambda a, b: a + b)*

*counts.saveAsTextFile("hdfs://localhost:9000/user/hduser/spark")*

*exit()*

*hdfs dfs -ls spark*

*hdfs dfs -copyToLocal spark/part-00000 p0*

*hdfs dfs -copyToLocal spark/part-00001 p1*

*grep -i Peter p0*

*grep -i Peter p1*

Take a screen shot of the output from this part - you will need to include this in your report

## Shutdown server processes

*java -jar /usr/local/derby/lib/derbyrun.jar server shutdown*

*stop-yarn.sh*

*stop-dfs.sh*

## Shutdown / Power down

Following the instructions appropriate the method you used to connect, shut down and then (from labs.azure.com) power down the server.