LAB 3 INSTRUCTIONS

DS8003 – MGT OF BIG DATA AND TOOLS

Ryerson University

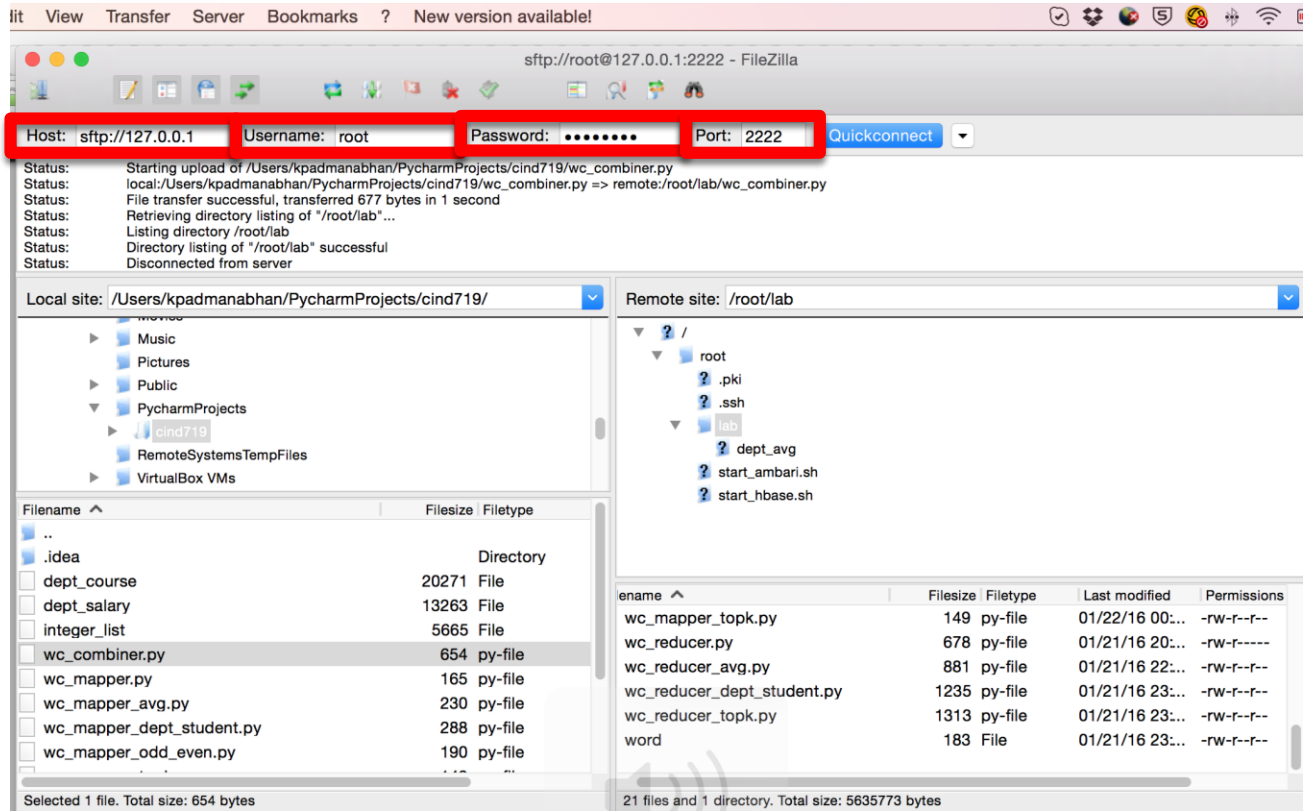
Instructor: Kanchana Padmanabhan

Upload Dataset to the Sandbox

- Download the dataset: shakespeare_100.txt to you local machine (WINDOWS MACHINE If you are in the LAB)–
- The file is available on D2L Brightspace under **Datasets & Scripts -> under Shakespeare**
- Upload the dataset via Filezilla (Lab Computer already has this software installed)
- If you don't have FileZilla, download and install here: https://filezilla-project.org
- Open Filezilla

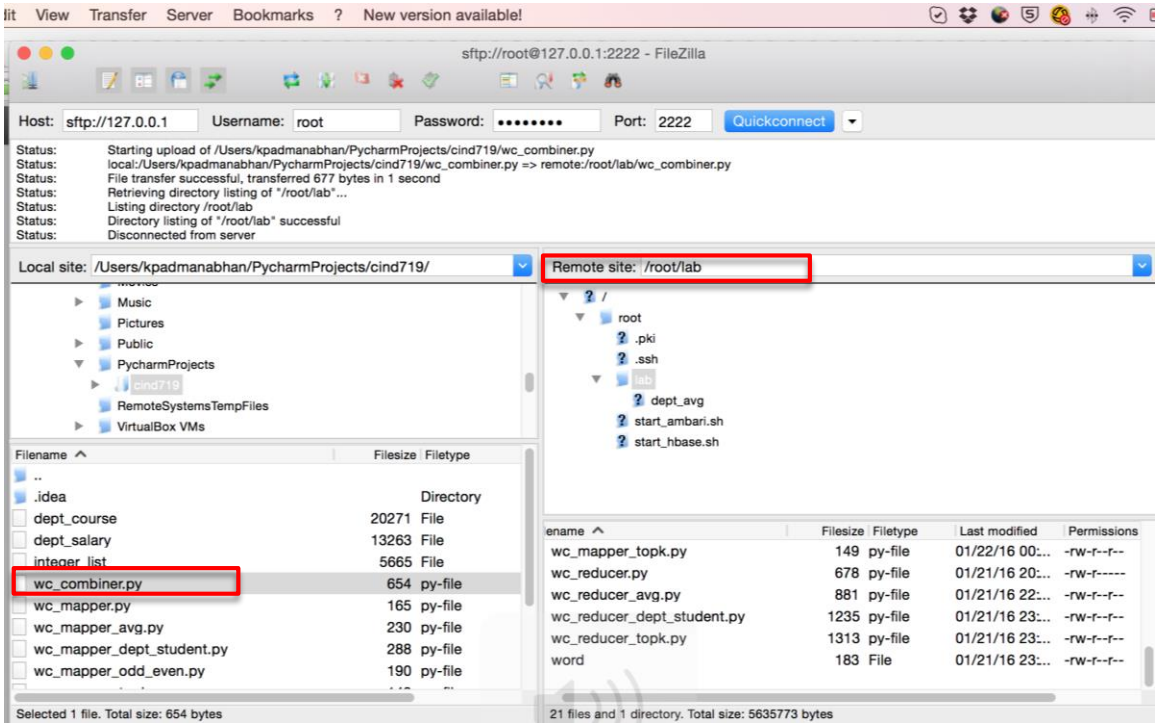
Connect to Virtual box using Filezilla

- Open Filezilla
- Enter Host: sftp://127.0.0.1
- Port: 2222
- Username & Password – same as your virtual box
- Click QuickConnect



Upload Dataset to the Sandbox

- After connecting to the Sandbox access node in Filezilla…
  - The left side shows directories of your local computer (WINDOWS COMPUTER IF USING LAB MACHINE)
  - The right side box shows directories of your remote machine on Linux
    - In this case the HDP Sandbox (Virtual Box)
- Upload shakespeare_100.txt to Sandbox
  - On the right-side box, navigate to /root/lab
  - On the left-side box, navigate and find the shakespeare_100.txt file you downloaded and drag it to /root/lab on the right to the Sandbox



Upload few more files to /root/lab

- Download wc_mapper-2.py and wc-reducer-2.py from D2L undero **Week 3=> MapReduce-Lab => Python MapReduce scripts** and upload to VirtualBox

Copy Shakespere_100.txt into HDFS

- Using the "-put" command we practiced last week

```
[root@sandbox lab]# hdfs dfs -ls /user/root
Found 3 items
drwx------   - root root          0 2016-09-05 17:54 /user/root/.staging
-rw-r--r--   3 root root        318 2016-09-05 17:43 /user/root/test.txt
-rw-r--r--   3 root root        318 2016-09-05 17:54 /user/root/test_copy.txt
[root@sandbox lab]# hdfs dfs -put shakespeare_100.txt /user/root
[root@sandbox lab]# hdfs dfs -ls /user/root
Found 4 items
drwx------   - root root          0 2016-09-05 17:54 /user/root/.staging
-rw-r--r--   3 root root    5589917 2016-09-17 14:33 /user/root/shakespeare_100.
txt
-rw-r--r--   3 root root        318 2016-09-05 17:43 /user/root/test.txt
-rw-r--r--   3 root root        318 2016-09-05 17:54 /user/root/test_copy.txt
[root@sandbox lab]#
```

- The data files have to be in HDFS for MapReduce to start processing

MapReduce

- Today's Lab
  - Java MapReduce WordCount
  - Python Streaming WordCount

WordCount: This is similar to the Apples, Strawberries, and Oranges example we saw in class except that it is counting the occurrences of words in text file

Java MapReduce WordCount

1. Find your mapreduce-example jar file
   - [root@sandbox ~]# find /usr -name *hadoop-mapreduce-example*
2. Find the list of examples that can be run:
   - [root@sandbox lab]# hadoop jar /usr/hdp/2.4.0.0-169/hadoop-mapreduce/hadoop-mapreduce-examples-2.7.1.2.4.0.0-169.jar
3. Run java M/R wordcount example
   - [root@sandbox]# hadoop jar /usr/hdp/2.4.0.0-169/hadoop-mapreduce/hadoop-mapreduce-examples-2.7.1.2.4.0.0-169.jar wordcount /user/root/shakespeare_100.txt /user/root/shakespeare_100_out
   - View results
   - [root@sandbox]# hadoop fs -ls /user/root/shakespeare_100_out
   - [root@sandbox]# hadoop fs -cat /user/root/shakespeare_100_out/part-r-00000 | tail -n 50

Make sure this *.jar file path matches with the result from previous step

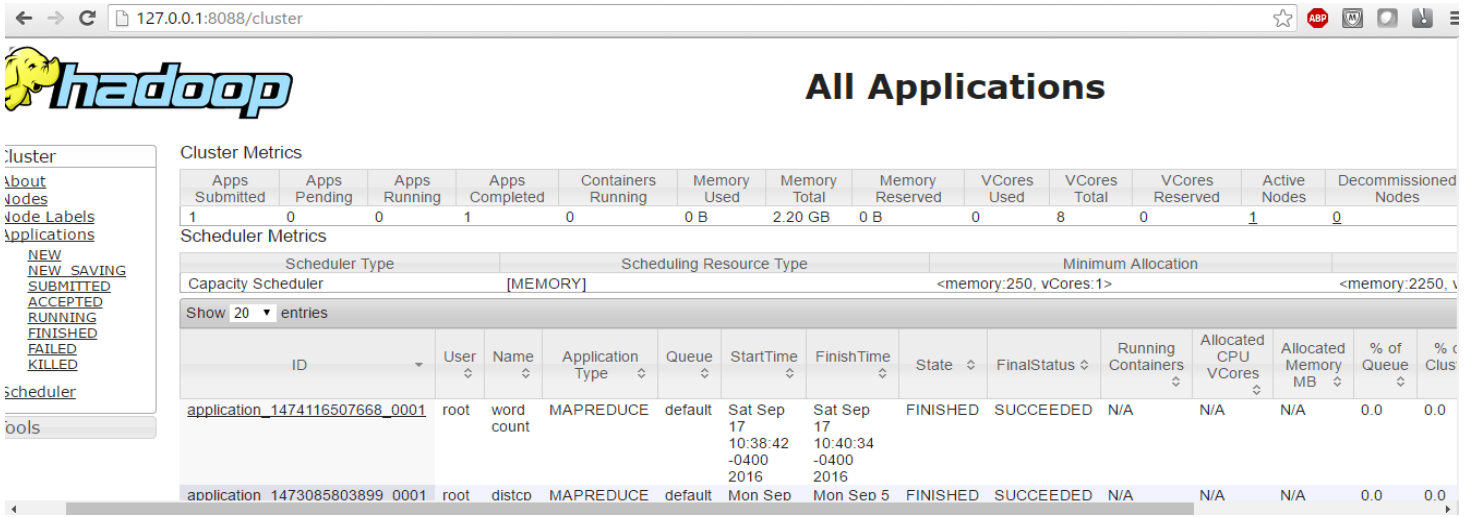
```
}       2
[root@sandbox lab]# hadoop fs -cat /user/root/shakespeare-out-java/part-r-00000 | tail -n 5
zone,   1
zounds! 1
zounds, 1
zwagger'd       1
}       2
[root@sandbox lab]#
```

Java MapReduce WordCount

Launching a job:

```
[root@sandbox lab]# hadoop jar /usr/hdp/2.4.0.0-169/hadoop-mapreduce/hadoop-mapr
educe-examples-2.7.1.2.4.0.0-169.jar wordcount /user/root/shakespeare_100.txt /u
ser/root/shakespeare_100_out
WARNING: Use "yarn jar" to launch YARN applications.
16/09/17 14:38:38 INFO impl.TimelineClientImpl: Timeline service address: http:/
/sandbox.hortonworks.com:8188/ws/v1/timeline/
16/09/17 14:38:39 INFO client.RMProxy: Connecting to ResourceManager at sandbox.
hortonworks.com/10.0.2.15:8050
16/09/17 14:38:41 INFO input.FileInputFormat: Total input paths to process : 1
```

Monitor a job: http://127.0.0.1:8088/cluster



Java MapReduce WordCount

Look at results

```
root@sandbox:~/lab
[root@sandbox lab]# hdfs dfs -ls /user/root/shakespeare_100_out
Found 2 items
-rw-r--r--   3 root root          0 2016-09-17 14:40 /user/root/shakespeare_100_
out/_SUCCESS
-rw-r--r--   3 root root     721004 2016-09-17 14:40 /user/root/shakespeare_100_
out/part-r-00000
[root@sandbox lab]# hdfs dfs -getmerge /user/root/shakespeare_100_out shakespear
e_wordcount
[root@sandbox lab]# ls
dept_course.txt   shakespeare_100.txt    test_two_copies.txt
dept_salary.txt   shakespeare_wordcount  test.txt
integer_list.txt  test_copy.txt
[root@sandbox lab]# head -5 shakespeare_wordcount
"       241
"'Tis   1
"A      4
"AS-IS".        1
"Air,"  1
[root@sandbox lab]#
```

MapReduce

- Today's Lab
  - Java MapReduce WordCount
  - Python Streaming WordCount

Python Streaming WordCount

1. We will use the *wc_mapper-2.py* and *wc_reducer-2.py* *we uploaded previously*
2. Find your mapreduce-example jar file
   - [root@sandbox lab]# find /usr -name *hadoop-streaming*
3. Run Hadoop streaming wordcount
   - [root@sandbox lab]# hadoop jar /usr/hdp/2.4.0.0-169/hadoop-mapreduce/hadoop-streaming-2.7.1.2.4.0.0-169.jar -file /root/lab/wc_mapper-2.py -mapper wc_mapper-2.py -file /root/lab/wc_reducer-2.py -reducer wc_reducer-2.py -input /user/root/shakespeare_100.txt -output /user/root/shakespeare_streaming_out
4. View results
   - [root@sandbox lab]# hadoop fs -ls /user/root/shakespeare_streaming_out
   - [root@sandbox lab]# hadoop fs –cat /user/root/shakespeare_streaming_out/part-00000 | tail -n 15

Python Streaming WordCount

Launching a job

root@sandbox:~/lab

```
[root@sandbox lab]# hadoop jar /usr/hdp/2.4.0.0-169/hadoop-mapreduce/hadoop-stre
aming-2.7.1.2.4.0.0-169.jar -file /root/lab/wc_mapper-2.py -mapper wc_mapper-2.p
y -file /root/lab/wc_reducer-2.py -reducer wc_reducer-2.py -input /user/root/sha
kespeare_100.txt -output /user/root/shakespeare_streaming_out
WARNING: Use "yarn jar" to launch YARN applications.
16/09/17 14:55:29 WARN streaming.StreamJob: -file option is deprecated, please u
se generic option -files instead.
packageJobJar: [/root/lab/wc_mapper-2.py, /root/lab/wc_reducer-2.py] [/usr/hdp/2
.4.0.0-169/hadoop-mapreduce/hadoop-streaming-2.7.1.2.4.0.0-169.jar] /tmp/streamj
ob4173614868856968256.jar tmpDir=null
16/09/17 14:55:32 INFO impl.TimelineClientImpl: Timeline service address: http:/
/sandbox.hortonworks.com:8188/ws/v1/timeline/
16/09/17 14:55:32 INFO client.RMProxy: Connecting to ResourceManager at sandbox.
hortonworks.com/10.0.2.15:8050
16/09/17 14:55:33 INFO impl.TimelineClientImpl: Timeline service address: http:/
/sandbox.hortonworks.com:8188/ws/v1/timeline/
16/09/17 14:55:33 INFO client.RMProxy: Connecting to ResourceManager at sandbox.
hortonworks.com/10.0.2.15:8050
16/09/17 14:55:33 INFO mapred.FileInputFormat: Total input paths to process : 1
16/09/17 14:55:34 INFO mapreduce.JobSubmitter: number of splits:2
16/09/17 14:55:34 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_14
74116507668_0003
16/09/17 14:55:35 INFO impl.YarnClientImpl: Submitted application application_14
74116507668_0003
```

Number of Reducers

- Number of reducers (jobs) can be changed using the option "-D mapred.reduce.tasks=n". where n is the number of reducers
- If n=0 then there will be no reducers. Mappers output will be written out to HDFS
- You may want to use n=0 when you are perform tasks such as filtering
- hadoop jar /usr/hdp/2.4.0.0-169/hadoop-mapreduce/hadoop-streaming-2.7.1.2.4.0.0-169.jar -D mapred.reduce.tasks=4 -file /root/lab/wc_mapper-2.py -mapper wc_mapper-2.py -file /root/lab/wc_reducer-2.py -reducer wc_reducer-2.py -input /user/root/shakespeare_100.txt -output /user/root/shakespeare_streaming_out_2

https://hadoop.apache.org/docs/r1.2.1/streaming.html#Specifying+the+Number+of+Reducers

Hadoop Streaming with Other Languages

- Using Linux Commands:
  - https://www.r-bloggers.com/using-hadoop-streaming-api-to-perform-a-word-count-job-in-r-and-c/
- Using Bash (Check D2L for sample code)
  - https://www.linkedin.com/pulse/20140706111754-176301000-hadoop-streaming-example-job-using-bash
- Using R (Check D2L for sample code)
  - https://coderwall.com/p/imxf6g/running-wordcount-on-hadoop-using-r-script

Word Count in Linux

- [root@sandbox lab]# cat shakespeare_100.txt | head
- [root@sandbox lab]# cat shakespeare_100.txt | head -1
- [root@sandbox lab]# cat shakespeare_100.txt | tr '[:space:]' '\n'| sort | uniq -c | sort -rn | head -15
- [root@sandbox lab]# cat shakespeare_100.txt | tr '[:space:]' '\n'| sort | uniq -c | sort -rn | grep "zealous"

Try yourself

Map-Reduce – ODD/Even Number

- **Input:** A file containing a list of numbers
  - Input file is "integer_list.txt" that can be found downloaded from D2L **Week 3=> MapReduce-Lab => Datasets**
  - Upload file to VirtualBox.
  - Copy the file to "/user/root" in the hdfs
- **Output:** Count the number of odd numbers and even numbers
- **Think about**
  - How do you mathematically check if a number if even or odd?
  - Starting from the WordCount Example- What changes need to be made to the mapper or reducer for this problem?

- Helpful Python code:
  - Casting
    - D ='5'
    - A = int(D)
    - converts string "5" into the number *5*
  - "%" is the modulo parameter;
    - 8%4
    - will be equal to 0 because 8 is completely divisible by 4

Map-Reduce – Average

- **Input:** A file containing Department[SPACE]Salary
  - Input file is "dept_salary.txt" that you downloaded from D2L **Week 3=> MapReduce-Lab => Datasets**
  - Upload to VirtualBox.
  - Copy the file to "/user/root" in the hdfs
- **Output:** Department[SPACE]Average Salary
- **Think about**
  - What required to calculate an average?
  - Starting from the WordCount Example- What changes need to be made to the mapper or reducer for this problem?

Summary

- The two python scripts perform the mapper and reducer tasks. The underlying data shuffling/sorting is taken care of by the mapreduce framework
- Now that you've practiced hadoop filesystem commands and tried to run some mapreduce code, you're ready to move on to the fun part – Pig and Hive!

Recommended Readings

- Hadoop Streaming: http://www.devx.com/opensource/introduction-to-hadoop-streaming.html
- https://hadoop.apache.org/docs/r1.2.1/streaming.html