



RUNNING A MAPREDUCE PROGRAM ON CDH INSTANCE ON AWS

PREREQUISITES

- Please ensure that you have installed the following on your Windows machine:
 1. [WinSCP](#) tool
 2. [Notepad++](#)

IMPORTANT INSTRUCTIONS

- The following notations have been used while running the Java API code.

[ec2-user@ip-10-0-0-14 ~]\$ **hadoop command**

Output of the command

As shown above, the command to be run is written in **bold**. The output of the command is written in *italics*. The [ec2-user@ip-10-0-0-14 ~] tells us the user through which the command is to be executed.

- Please be careful with the spaces in the commands.
- If a series of commands is given in a particular order, make sure that you run them in the same order.

NOTE: Before starting with the document below, it is necessary to have created the EC2 instance with Cloudera installed on it and to have connected to it as well. If not so, kindly go through [Video 1](#) and [Video 2](#) before getting started with this document.

STEPS TO RUN THE WORDCOUNT PROGRAM ON AMAZON EC2

- To check whether Java is available or not, do the following:

1. Switch to the root user using **sudo-i**
2. Now, run the following command

```
[root@ip-10-0-0-105 ~]# ls /usr/java/jdk1.7.0_67-cloudera/
```

- Set the JAVA_HOME and JRE_HOME in the /etc/profile location.

1. Open the file using the command given below.

vi /etc/profile

2. Add the following at the end of the file as shown below. Please change to the insert mode by pressing **i** before pasting the following lines.

export JAVA_HOME=/usr/java/jdk1.7.0_67-cloudera/

export JRE_HOME=/usr/java/jdk1.7.0_67-cloudera/jre/

export PATH=\$JAVA_HOME/bin:\$PATH

```
for i in /etc/profile.d/*.sh ; do
    if [ -r "$i" ]; then
        if [ "${-#*i}" != "$-" ]; then
            . "$i"
        else
            . "$i" >/dev/null
        fi
    fi
done

unset i
unset -f pathmunge
export JAVA_HOME=/usr/java/jdk1.7.0_67-cloudera/
export JRE_HOME=/usr/java/jdk1.7.0_67-cloudera/jre/
export PATH=$JAVA_HOME/bin:$PATH
-- INSERT --
```

3. Now, save and exit the file. It is important to exit from the insert mode and enter the following in the command mode while using the vi editor:

:wq!

- Now run the following commands as shown below:

```
[root@ip-10-0-0-105 ~]# source /etc/profile
```

```
[root@ip-10-0-0-105 ~]# echo $JAVA_HOME
```

```
/usr/java/jdk1.7.0_67-cloudera/
```

```
[root@ip-10-0-0-105 ~]# java -version
```

```
java version "1.7.0_67"
```

```
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
```

```
[root@ip-10-0-0-105 ~]# javac -version
```

```
javac 1.7.0_67
```

```
[root@ip-10-0-0-105 ~]# source /etc/profile
[root@ip-10-0-0-105 ~]# echo $JAVA_HOME
/usr/java/jdk1.7.0_67-cloudera/
[root@ip-10-0-0-105 ~]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[root@ip-10-0-0-105 ~]# javac
Usage: javac <options> <source files>
where possible options include:
  -g               Generate all debugging info
  -g:none          Generate no debugging info
  -g:{lines,vars,source} Generate only some debugging info
  -nowarn          Generate no warnings
  -verbose         Output messages about what the compiler is doing
  -deprecation     Output source locations where deprecated APIs are used
  -classpath <path> Specify where to find user class files and annotation processors
  -cp <path>       Specify where to find user class files and annotation processors
  -sourcepath <path> Specify where to find input source files
  -bootclasspath <path> Override location of bootstrap class files
  -extdirs <dirs>   Override location of installed extensions
  -endorseddirs <dirs> Override location of endorsed standards path
  -proc:{none,only} Control whether annotation processing and/or compilation is done.
  -processor <class1>[,<class2>,<class3>...] Names of the annotation processors to run; bypasses default discovery process
  -processorpath <path> Specify where to find annotation processors
  -d <directory>    Specify where to place generated class files
  -s <directory>    Specify where to place generated source files
  -implicit:{none,class} Specify whether or not to generate class files for implicitly referenced files
  -encoding <encoding> Specify character encoding used by source files
  -source <release> Provide source compatibility with specified release
  -target <release> Generate class files for specific VM version
  -version          Version information
  -help            Print a synopsis of standard options
  -Akey[=value]    Options to pass to annotation processors
  -X              Print a synopsis of nonstandard options
  -J<flag>         Pass <flag> directly to the runtime system
  -Werror          Terminate compilation if warnings occur
  @<filename>      Read options and filenames from file

[root@ip-10-0-0-105 ~]# javac -version
javac 1.7.0_67
```

Copying the “WordCount.Java” program to the CDH Instance on AWS

Linux/Mac users:

Use the below command:

scp -i <enter the path of the .pem file> WordCount.java ec2-user@<your Public IP>:/home/ec2-user/

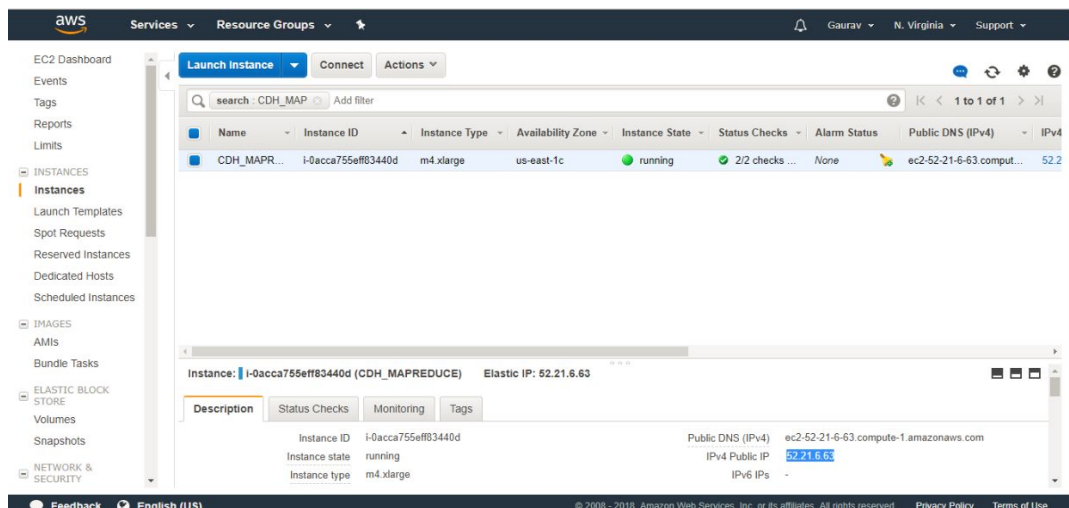
```
Vibhore-MacBook-Pro:Downloads vibhore$ chmod 400 cdh_hdfs.pem
Vibhore-MacBook-Pro:Downloads vibhore$ scp -i cdh_hdfs.pem abt.java ec2-user@52.21.6.63:/home/ec2-user/
/etc/profile.d/lang.sh: line 19: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or directory
abt.java
```

Windows users:

- WinSCP is a tool to transfer a file from a Windows machine to a Linux machine (EC2 instance).

- Open WinSCP.
- Enter the following credentials

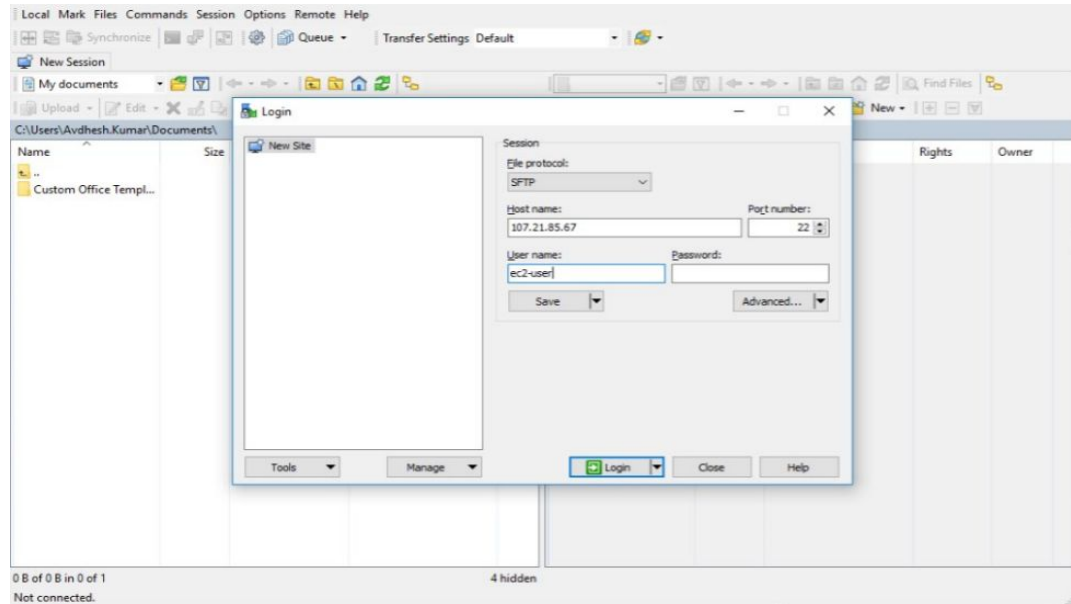
Hostname: Provide the public IP from the EC2 dashboard.



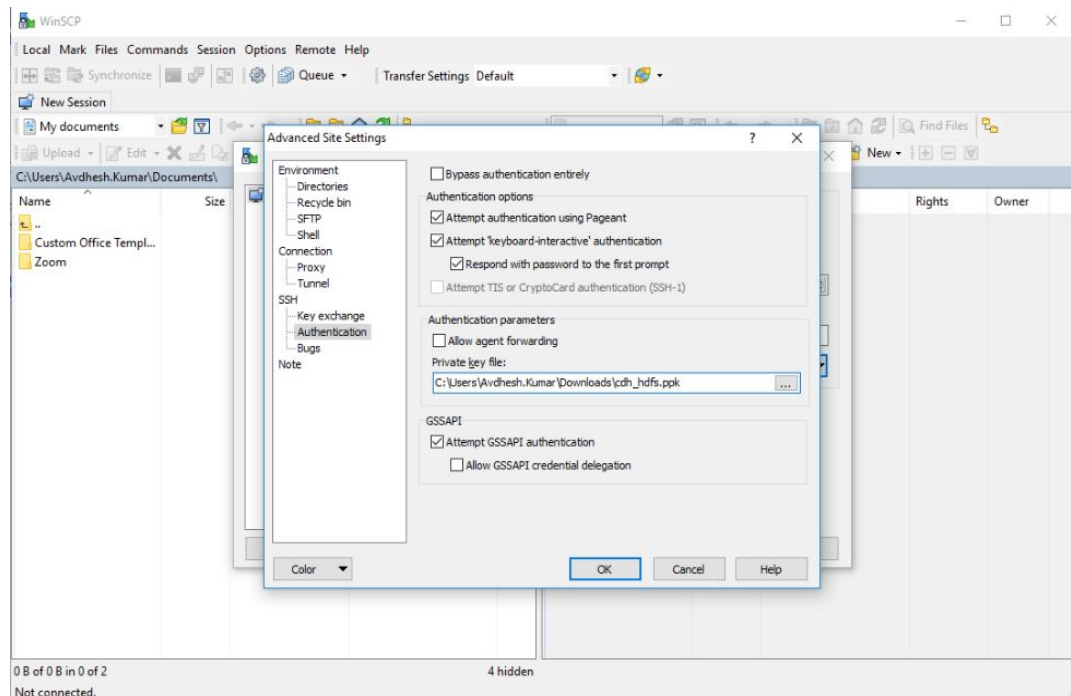
The screenshot shows the AWS Management Console interface. On the left, the navigation menu includes 'EC2 Dashboard', 'Events', 'Tags', 'Reports', 'Limits', 'INSTANCES', 'Launch Templates', 'Spot Requests', 'Reserved Instances', 'Dedicated Hosts', 'Scheduled Instances', 'IMAGES', 'AMIs', 'Bundle Tasks', 'ELASTIC BLOCK STORE', 'Volumes', 'Snapshots', and 'NETWORK & SECURITY'. The main content area displays a table of EC2 instances. The instance 'CDH_MAPREDUCE' is highlighted, showing its ID 'i-0acca755eff83440d', type 'm4.xlarge', availability zone 'us-east-1c', state 'running', and public IP '52.21.6.63'. Below the table, the 'Description' tab is selected, showing details for the instance 'i-0acca755eff83440d (CDH_MAPREDUCE)'. The details include the instance ID, state 'running', type 'm4.xlarge', public DNS (IPv4) 'ec2-52-21-6-63.compute-1.amazonaws.com', IPv4 public IP '52.21.6.63', and IPv6 IPs.

Username: **ec2-user**

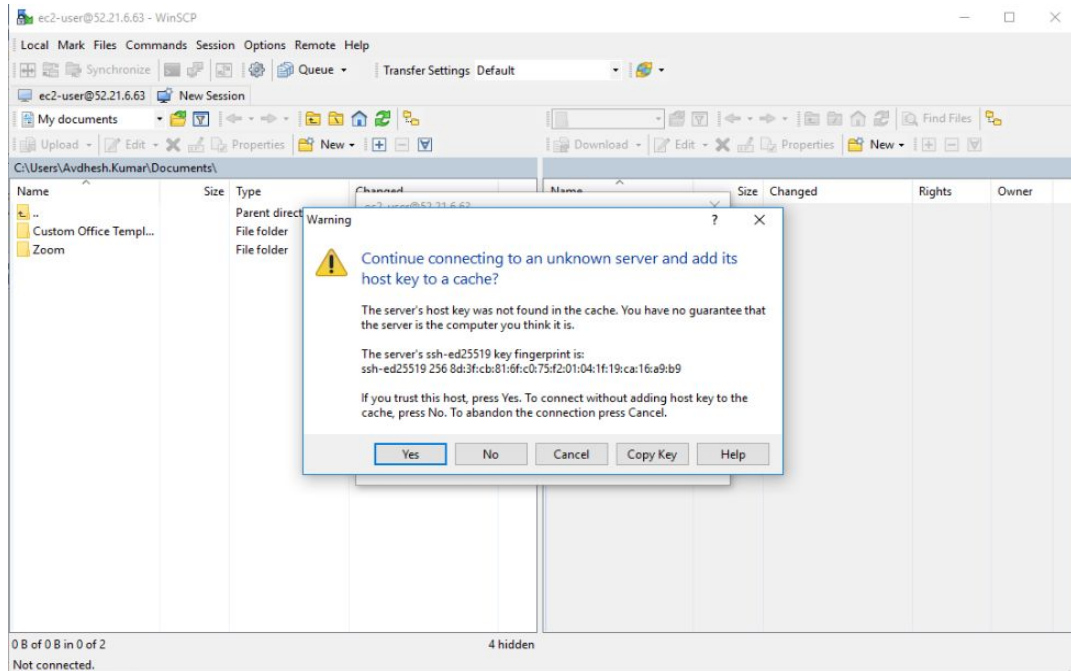
Then, click on '**Advanced**'.



3. After clicking on '**Authentication**', enter the path of your PPK file.



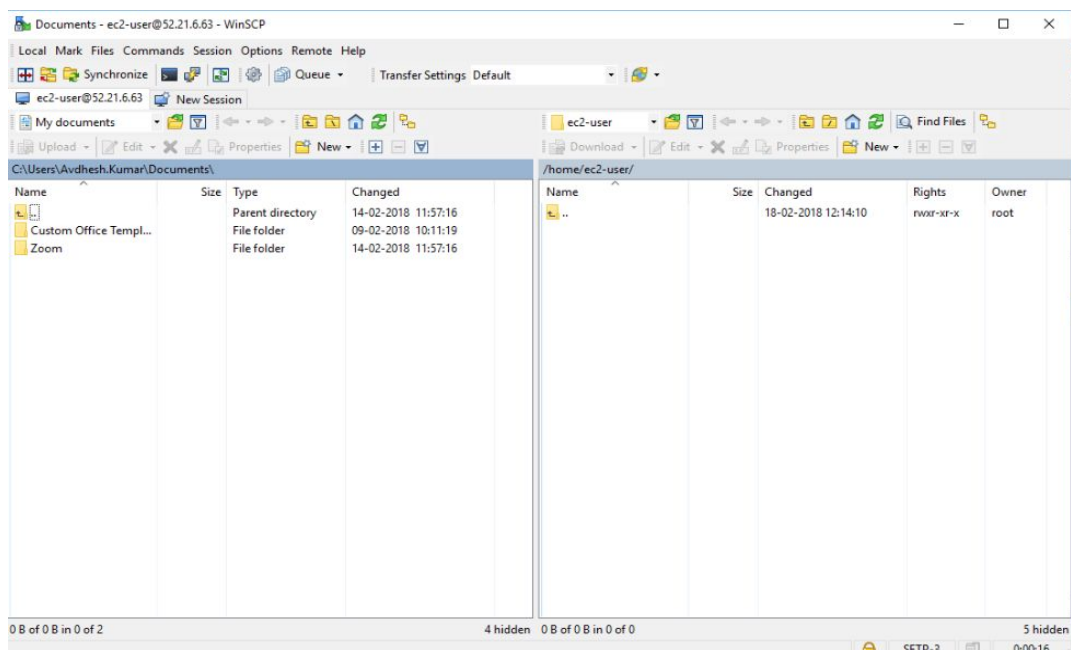
- Click on 'OK' followed by 'Login' after which the a pop-up will appear. Click on 'Yes'.



- The following screen appears:

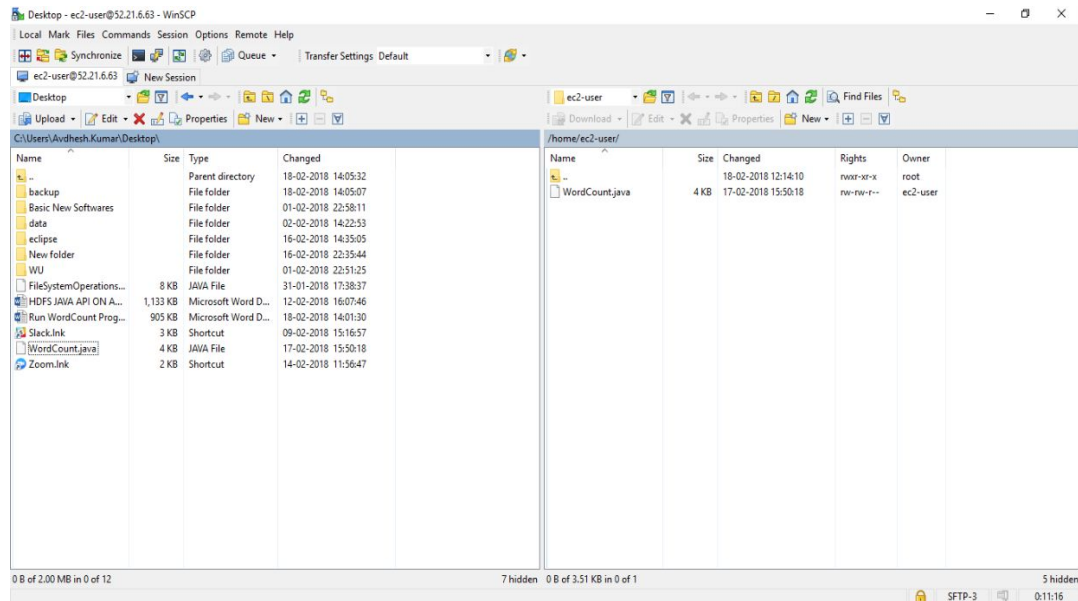
Left side screen: your local machine (Windows, in our case)

Right side screen: your linux machine (AWS EC2 instance)





6. Browse to the 'WordCount.java' file on the left side and drag and drop it to the right. Click on 'OK' on the prompt which appears.



7. We have now successfully copied the 'WordCount.java' from our local machine to our EC2 instance.

- Now, go back to AWS EC2 instance. Copy the 'WordCount.java' from '/home/ec2-user/' to the root home directory i.e '/root' using the command given below and verify it using the 'ls' command.

```
[root@ip-10-0-0-105 ~]# cp /home/ec2-user/Wordcount.java /root
```

```
[root@ip-10-0-0-105 ~]# cp /home/ec2-user/WordCount.java /root
[root@ip-10-0-0-105 ~]# ls
anaconda-ks.cfg  cloudera-manager-installer.bin  mysql-community-release-el7-5.noarch.rpm  original-ks.cfg  WordCount.java
[root@ip-10-0-0-105 ~]#
```

Creating a file using a root user

- Create a file 'input.txt' in your local filesystem using the 'cat' command and enter the desired text in the file as shown below. Use 'Ctrl+d' to save and exit the file.

```
cat > input.txt
```

```
[root@ip-10-0-0-105 ~]# cat > input.txt
The course 2 of the big data program is the most important part of the entire program
[root@ip-10-0-0-105 ~]#
```



- Now verify whether the file has been created or not using the 'ls' command.

```
[root@ip-10-0-0-105 ~]# ls
anaconda-ks.cfg  cloudera-manager-installer.bin  input.txt  mysql-community-release-el7-5.noarch.rpm  original-ks.cfg  WordCount.java
[root@ip-10-0-0-105 ~]#
```

Create a directory inside the HDFS and changing it's owner

- The commands used below demonstrate how to create a directory in the HDFS.

Note: A directory can be created in hadoop only using the hdfs user. So now, switch to the hdfs user. Please note there is a space between - and **hdfs** in the command used below:

```
[root@ip-10-0-0-105 ~]# su - hdfs
```

```
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -mkdir /user/root/
```

- You can verify the directory created by running the command shown below.

```
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -ls /user/
```

Found 6 items

```
drwxrwxrwx - mapred hadoop      0 2018-02-09 09:28 /user/history
```

```
drwxrwxr-t - hive  hive        0 2018-02-09 09:30 /user/hive
```

```
drwxrwxr-x - hue   hue         0 2018-02-09 09:30 /user/hue
```

```
drwxrwxr-x - oozie oozie      0 2018-02-09 09:30 /user/oozie
```

```
drwxr-xr-x - hdfs  supergroup  0 2018-02-12 05:58 /user/root
```

```
drwxr-x--x - spark spark      0 2018-02-09 09:29 /user/spark
```

Now, as seen above, the owner of the directory created is **hdfs** (underlined above). To send a file from any user to hdfs, the owner of the directory inside hdfs should be changed to the user sending the file. For example: If you have to send a file from the root user to a directory inside hdfs, the owner of that particular directory inside hdfs should be changed to root.

- To change the owner of the directory created from hdfs to root, run the following command:

```
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -chown root /user/root
```


- You can verify the same using the command shown below:

```
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -ls /user/
```

Found 6 items

```
drwxrwxrwx - mapred hadoop          0 2018-02-18 07:16 /user/history
```

```
drwxrwxr-t - hive  hive            0 2018-02-18 07:17 /user/hive
```

```
drwxrwxr-x - hue  hue              0 2018-02-18 07:18 /user/hue
```

```
drwxrwxr-x - oozie oozie           0 2018-02-18 07:18 /user/oozie
```

```
drwxr-xr-x - root supergroup       0 2018-02-18 09:49 /user/root
```

```
drwxr-x--x - spark spark           0 2018-02-18 07:17 /user/spark
```

You can see that the owner has changed from **hdfs** to **root**.

```
root@ip-10-0-0-105 ~]# su - hdfs
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -mkdir /user/root/
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -ls /user/
Found 6 items
drwxrwxrwx - mapred hadoop          0 2018-02-18 07:16 /user/history
drwxrwxr-t - hive  hive            0 2018-02-18 07:17 /user/hive
drwxrwxr-x - hue  hue              0 2018-02-18 07:18 /user/hue
drwxrwxr-x - oozie oozie           0 2018-02-18 07:18 /user/oozie
drwxr-xr-x - hdfs supergroup       0 2018-02-18 09:49 /user/root
drwxr-x--x - spark spark           0 2018-02-18 07:17 /user/spark
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -chown root /user/root
[hdfs@ip-10-0-0-105 ~]$ hadoop fs -ls /user/
Found 6 items
drwxrwxrwx - mapred hadoop          0 2018-02-18 07:16 /user/history
drwxrwxr-t - hive  hive            0 2018-02-18 07:17 /user/hive
drwxrwxr-x - hue  hue              0 2018-02-18 07:18 /user/hue
drwxrwxr-x - oozie oozie           0 2018-02-18 07:18 /user/oozie
drwxr-xr-x - root supergroup       0 2018-02-18 09:49 /user/root
drwxr-x--x - spark spark           0 2018-02-18 07:17 /user/spark
[hdfs@ip-10-0-0-105 ~]$
```

- Now, use the 'exit' command to shift from the hdfs user to the root user.

```
[hdfs@ip-10-0-0-105 ~]$ exit;
```

logout

```
[root@ip-10-0-0-105 ~]#
```

```
[hdfs@ip-10-0-0-105 ~]$ exit;
logout
[root@ip-10-0-0-105 ~]#
```



- Now, since the owner has changed from hdfs to root, the root user will be able to create a new directory 'wordcount' inside /user/root/ on the hdfs.

```
[root@ip-10-0-0-105 ~]# hadoop fs -mkdir /user/root/wordcount
```

```
[root@ip-10-0-0-105 ~]# hadoop fs -mkdir /user/root/wordcount  
[root@ip-10-0-0-105 ~]#
```

- Use the 'put' command to put the 'input.txt' file into the /user/root/wordcount directory. Verify the same using the 'ls' command.

```
[root@ip-10-0-0-105 ~]# hadoop fs -put input.txt /user/root/wordcount/
```

```
[root@ip-10-0-0-105 ~]# hadoop fs -ls /user/root/wordcount/
```

```
[root@ip-10-0-0-105 ~]# hadoop fs -put input.txt /user/root/wordcount/  
[root@ip-10-0-0-105 ~]# hadoop fs -ls /user/root/wordcount/  
Found 1 items  
-rw-r--r--  3 root supergroup      86 2018-02-18 10:06 /user/root/wordcount/input.txt  
[root@ip-10-0-0-105 ~]#
```

- Now, create a new directory 'test_classes' using the 'mkdir' command to store the class files after the compilation of the Java code.

```
[root@ip-10-0-0-105 ~]# mkdir test_classes
```

- Set the environment variable for the Hadoop classpath using the below command:

```
[root@ip-10-0-0-105 ~]# export HADOOP_CLASSPATH=$(hadoop classpath)
```

```
[root@ip-10-0-0-105 ~]# mkdir test_classes  
[root@ip-10-0-0-105 ~]# export HADOOP_CLASSPATH=$(hadoop classpath)  
[root@ip-10-0-0-105 ~]#
```

- Run the classpath using the following command:

```
[root@ip-10-0-0-105 ~]# javac -classpath ${HADOOP_CLASSPATH} -d /root/test_classes WordCount.java
```

```
root@ip-10-0-0-105 ~]# javac -classpath ${HADOOP_CLASSPATH} -d /root/test_classes WordCount.java  
root@ip-10-0-0-105 ~]#
```



- Verify whether the classes have been created or not after the compilation of the Java code.

```
[root@ip-10-0-0-105 ~]# cd test_classes/
```

```
[root@ip-10-0-0-105 test_classes]# ls
```

You can see that the classes have been created.

```
[root@ip-10-0-0-105 ~]# cd test_classes/
[root@ip-10-0-0-105 test_classes]# ls
WordCount.class  WordCount$IntSumReducer.class  WordCount$TokenizerMapper.class
[root@ip-10-0-0-105 test_classes]#
```

- Go back from the test_classes directory using 'cd ..' command.

```
[root@ip-10-0-0-105 test_classes]# cd ..
```

```
[root@ip-10-0-0-105 ~]#
```

```
[root@ip-10-0-0-105 test_classes]# ls
WordCount.class  WordCount$IntSumReducer.class  WordCount$TokenizerMapper.class
[root@ip-10-0-0-105 test_classes]# cd ..
[root@ip-10-0-0-105 ~]#
```

Creating a JAR file

- To create a JAR file, run the command shown below:

The syntax for creating a jar file is:

```
jar -cvf jarname.jar -C classfolder_name / .
```

```
[root@ip-10-0-0-105 ~]# jar -cvf WordCount.jar -C test_classes/ .
```

```
[root@ip-10-0-0-105 ~]# jar -cvf WordCount.jar -C test_classes/ .
added manifest
adding: WordCount$TokenizerMapper.class(in = 1736) (out= 755) (deflated 56%)
adding: WordCount$IntSumReducer.class(in = 1739) (out= 741) (deflated 57%)
adding: WordCount.class(in = 1501) (out= 809) (deflated 46%)
[root@ip-10-0-0-105 ~]#
```

Running the JAR file using hadoop jar commands

- Run the JAR file created in the above step using the command shown below:

Syntax: `hadoop jar <JARfile name> <classfile> <input path> <output path>`

```
[root@ip-10-0-0-105 ~]# hadoop jar WordCount.jar WordCount  
/user/root/wordcount/input.txt /user/root/wordcount/output
```

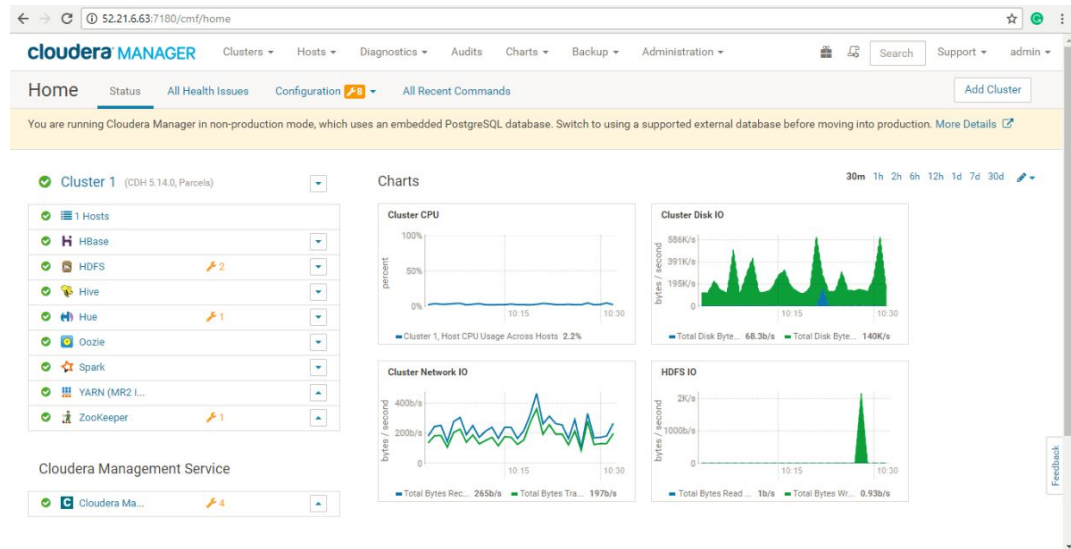
```
[root@ip-10-0-0-105 ~]# hadoop jar WordCount.jar WordCount /user/root/wordcount/input.txt /user/root/wordcount/output  
18/02/18 10:27:10 INFO client.RMProxy: Connecting to ResourceManager at ip-10-0-0-105.ec2.internal/10.0.0.105:8032  
18/02/18 10:27:10 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
18/02/18 10:27:11 INFO input.FileInputFormat: Total input paths to process : 1  
18/02/18 10:27:11 INFO mapreduce.JobSubmitter: number of splits:1  
18/02/18 10:27:11 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1518938193435_0001  
18/02/18 10:27:12 INFO impl.YarnClientImpl: Submitted application application_1518938193435_0001  
18/02/18 10:27:12 INFO mapreduce.Job: The url to track the job: http://ip-10-0-0-105.ec2.internal:8088/proxy/application_1518938193435_0001/  
18/02/18 10:27:12 INFO mapreduce.Job: Running job: job_1518938193435_0001
```

As you can see, the process gets stuck. To kill the process, use 'Ctrl+C'.

```
^C[root@ip-10-0-0-105 ~]#
```

- To solve this problem, we need to visit the YARN Configuration page and edit a few properties.
 - Go to your browser and open cloudera manager. To access the cloudera manager page, enter your public ip address followed by ':7180' as shown below:

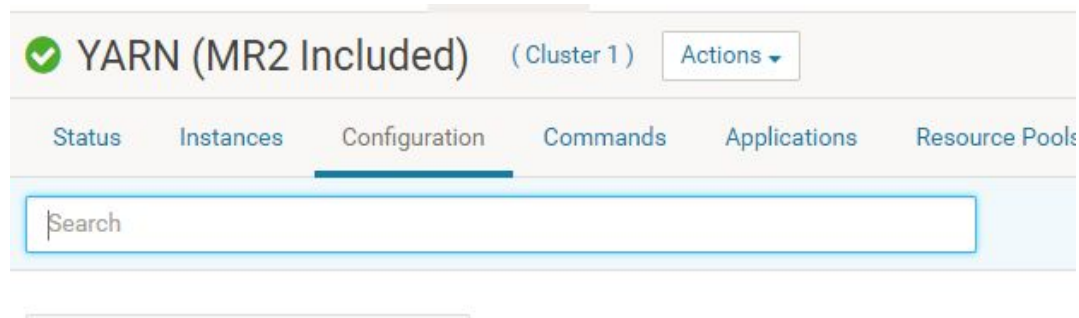
<Your Public IP>:7180



2. Click on '**YARN (MR2 Included)**' after which the following page appears:



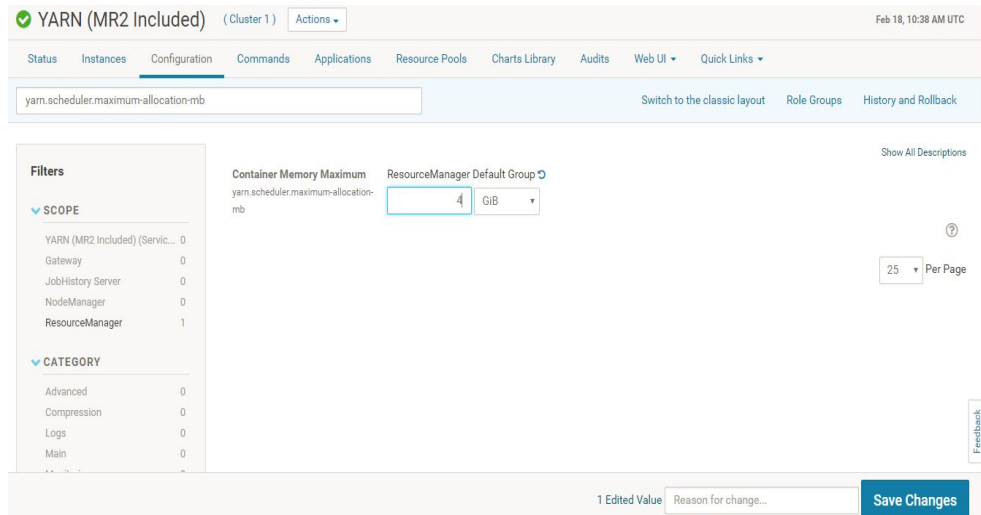
3. Click on '**Configuration**'.



4. Now we have to increase the RAM. To do so, enter the following properties in the search tab. Increase the RAM of each property to the number mentioned below:

- a. **yarn.scheduler.maximum-allocation-mb**

The default value is 1 GB. Change it to 4GB and then click on ‘Save Changes’.



YARN (MR2 Included) (Cluster 1) Actions

Feb 18, 10:38 AM UTC

Status Instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI Quick Links

yarn.scheduler.maximum-allocation-mb Switch to the classic layout Role Groups History and Rollback

Filters

SCOPE

YARN (MR2 Included) (Service...)	0
Gateway	0
JobHistory Server	0
NodeManager	0
ResourceManager	1

CATEGORY

Advanced	0
Compression	0
Logs	0
Main	0

Container Memory Maximum
yarn.scheduler.maximum-allocation-mb

ResourceManager Default Group

4 GiB

Show All Descriptions

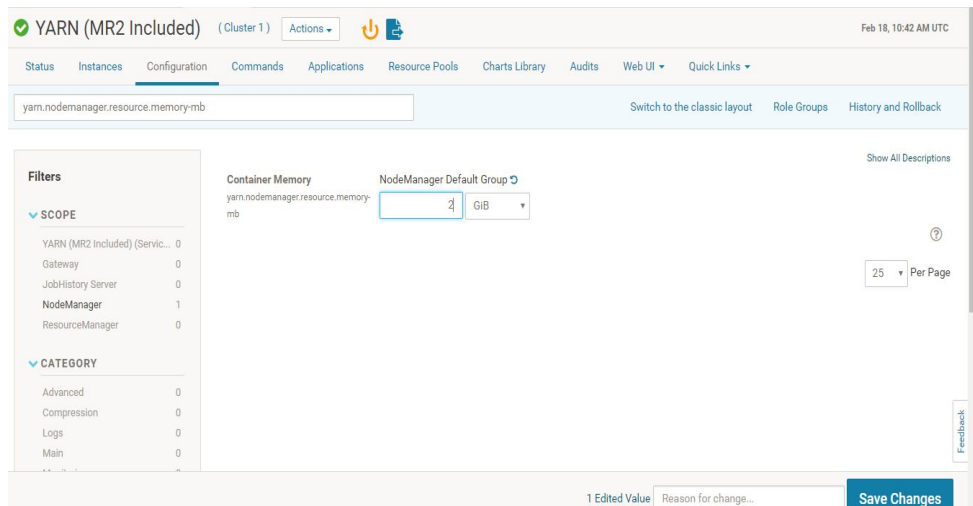
25 Per Page

1 Edited Value Reason for change...

Save Changes

b. yarn.nodemanager.resource.memory-mb

The default value is 1 GB. Change it to 2GB and then click on ‘Save Changes’.



YARN (MR2 Included) (Cluster 1) Actions

Feb 18, 10:42 AM UTC

Status Instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI Quick Links

yarn.nodemanager.resource.memory-mb Switch to the classic layout Role Groups History and Rollback

Filters

SCOPE

YARN (MR2 Included) (Service...)	0
Gateway	0
JobHistory Server	0
NodeManager	1
ResourceManager	0

CATEGORY

Advanced	0
Compression	0
Logs	0
Main	0

Container Memory
yarn.nodemanager.resource.memory-mb

NodeManager Default Group

2 GiB

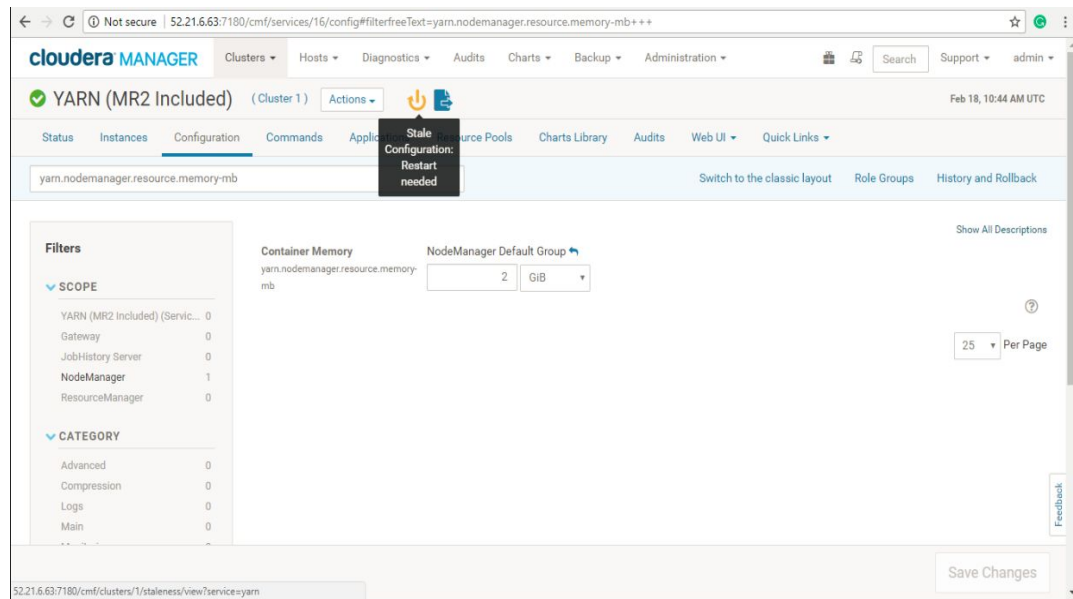
Show All Descriptions

25 Per Page

1 Edited Value Reason for change...

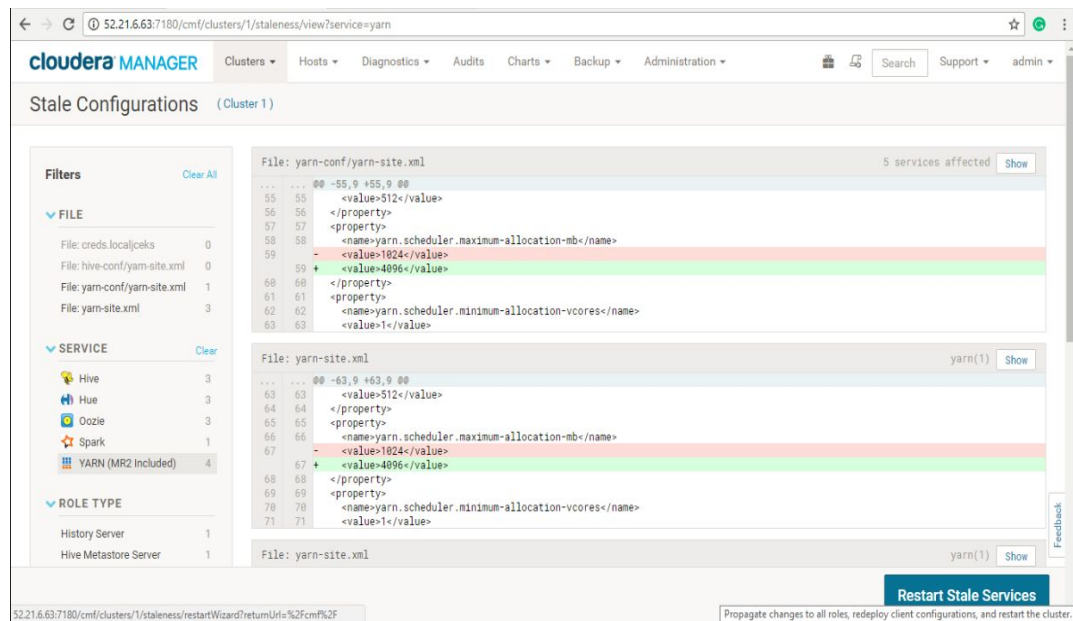
Save Changes

- Restart the YARN service by clicking on the ‘**State Configuration Restart needed**’ icon as shown in the image below:



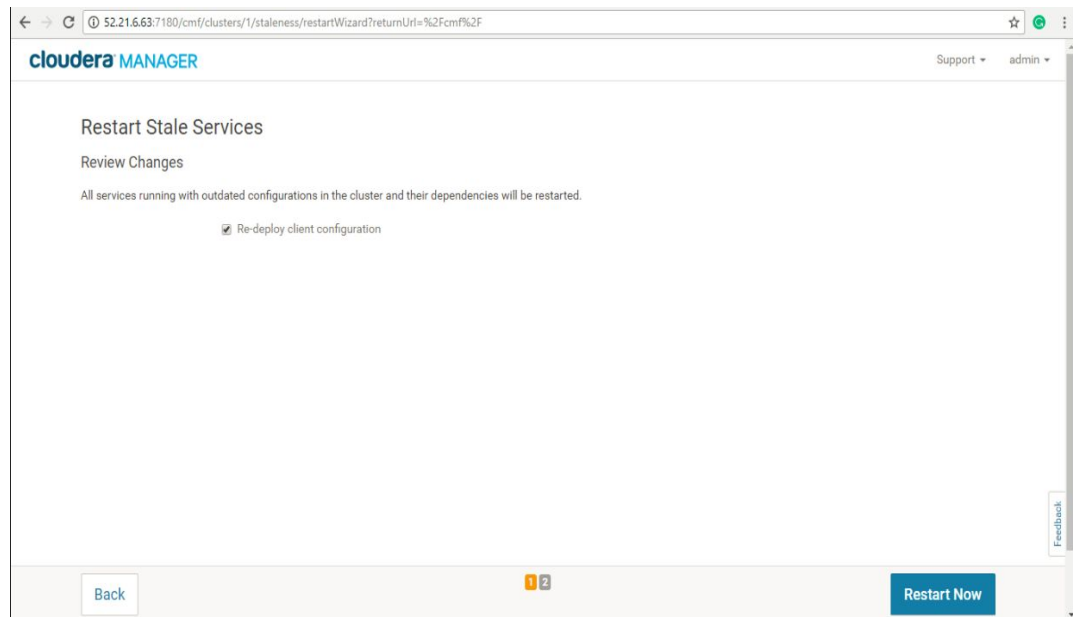
The screenshot shows the Cloudera Manager interface for the YARN (MR2 Included) cluster. The 'Configuration' tab is active, and the search filter is set to 'yarn.nodemanager.resource.memory-mb'. A tooltip indicates that the configuration is 'Stale Configuration: Restart needed'. The configuration value is set to 2 GiB. The left sidebar shows filters for SCOPE (YARN, Gateway, JobHistory Server, NodeManager, ResourceManager) and CATEGORY (Advanced, Compression, Logs, Main). The bottom right has a 'Save Changes' button.

6. Now, click on 'Restart Stale services'.

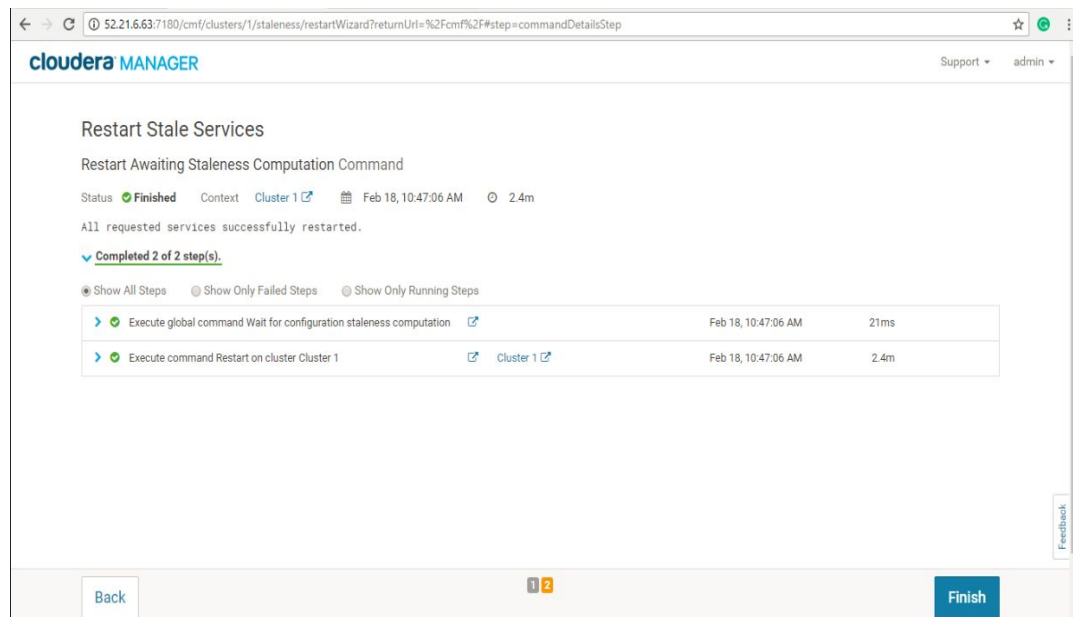


The screenshot shows the Cloudera Manager 'State Configurations' page for the YARN service. The left sidebar shows filters for FILE (yarn-conf/yarn-site.xml, yarn-site.xml) and SERVICE (Hive, Hue, Oozie, Spark, YARN). The YARN service is selected, showing 4 instances. The right pane displays the XML configuration for yarn-site.xml, with a red highlight indicating a stale configuration. A 'Restart Stale Services' button is visible at the bottom right.

7. Click on 'Restart Now'.



8. It generally takes a few minutes to restart the service. Wait till both the green ticks appear. Click on '**Finish**'.



9. Verify that the properties have been changed to the respective values shown below:

- a. yarn.scheduler.maximum-allocation-mb - 4GB
- b. yarn.nodemanager.resource.memory-mb - 2GB

Running the JAR file and checking its output

- Now, again run the JAR file using the command shown below:

```
[root@ip-10-0-0-105 ~]# hadoop jar WordCount.jar WordCount  
/user/root/wordcount/input.txt /user/root/wordcount/output
```

```
root@ip-10-0-0-105 ~]# hadoop jar WordCount.jar WordCount /user/root/wordcount/input.txt /user/root/wordcount/output
02/18 10:51:20 INFO client.RMProxy: Connecting to ResourceManager at ip-10-0-0-105.ec2.internal/10.0.0.105:8032
02/18 10:51:20 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application
Runner to remedy this.
02/18 10:51:20 INFO input.FileInputFormat: Total input paths to process : 1
02/18 10:51:21 INFO mapreduce.JobSubmitter: number of splits:1
02/18 10:51:21 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1518950844391_0001
02/18 10:51:21 INFO impl.YarnClientImpl: Submitted application application_1518950844391_0001
02/18 10:51:21 INFO mapreduce.Job: The url to track the job: http://ip-10-0-0-105.ec2.internal:8088/proxy/application_1518950844391_0001/
02/18 10:51:21 INFO mapreduce.Job: Running job: job_1518950844391_0001
02/18 10:51:30 INFO mapreduce.Job: Job job_1518950844391_0001 running in uber mode : false
02/18 10:51:30 INFO mapreduce.Job: map 0% reduce 0%
02/18 10:51:35 INFO mapreduce.Job: map 100% reduce 0%
02/18 10:51:42 INFO mapreduce.Job: map 100% reduce 50%
02/18 10:51:47 INFO mapreduce.Job: map 100% reduce 100%
02/18 10:51:47 INFO mapreduce.Job: Job job_1518950844391_0001 completed successfully
02/18 10:51:47 INFO mapreduce.Job: Counters: 49
  File System Counters
    FILE: Number of bytes read=173
    FILE: Number of bytes written=446400
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=219
    HDFS: Number of bytes written=93
    HDFS: Number of read operations=5
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=4
  Job Counters
    Launched map tasks=1
    Launched reduce tasks=2
    Data-local map tasks=1
    Total time spent by all maps in occupied slots (ms)=3387
    Total time spent by all reduces in occupied slots (ms)=7043
    Total time spent by all map tasks (ms)=3387
    Total time spent by all reduce tasks (ms)=7043
    Total vcore-milliseconds taken by all map tasks=3387
    Total vcore-milliseconds taken by all reduce tasks=7043
    Total megabyte-milliseconds taken by all map tasks=3468268
    Total megabyte-milliseconds taken by all reduce tasks=7212032
  Map-Reduce Framework
    Map input records=1
    Map output records=17
```



- Verify whether the program ran successfully by accessing the Resource Manager (RM).
 1. Access the Resource Manager by typing your Public IP address followed by '8088' as shown: **<Your Public IP>:8088**

The screenshot shows the Hadoop Resource Manager (RM) web interface. The title is 'All Applications'. The interface includes a sidebar with navigation links: Cluster, About, Nodes, Applications, Scheduler, and Tools. The main content area displays 'Cluster Metrics' and 'User Metrics for dr.who'. A table lists applications, with one application 'application_1518950844391_0001' in a 'SUCCEEDED' state, highlighted in blue.

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB
application_1518950844391_0001	root	word count	MAPREDUCE	root.users.root	Sun Feb 18 16:21:21 +0550 2018	Sun Feb 18 16:21:46 +0550 2018	FINISHED	SUCCEEDED	N/A	N/A	N/A

2. As highlighted in the above image, it should show '**SUCCEEDED**'.

- To check the output of the WordCount program, follow these steps:
 1. We need to access Hue. There are 2 ways to do so:
 - a. Direct Access - using your Public IP followed by '8088' as shown below:

<Public IP>:8088

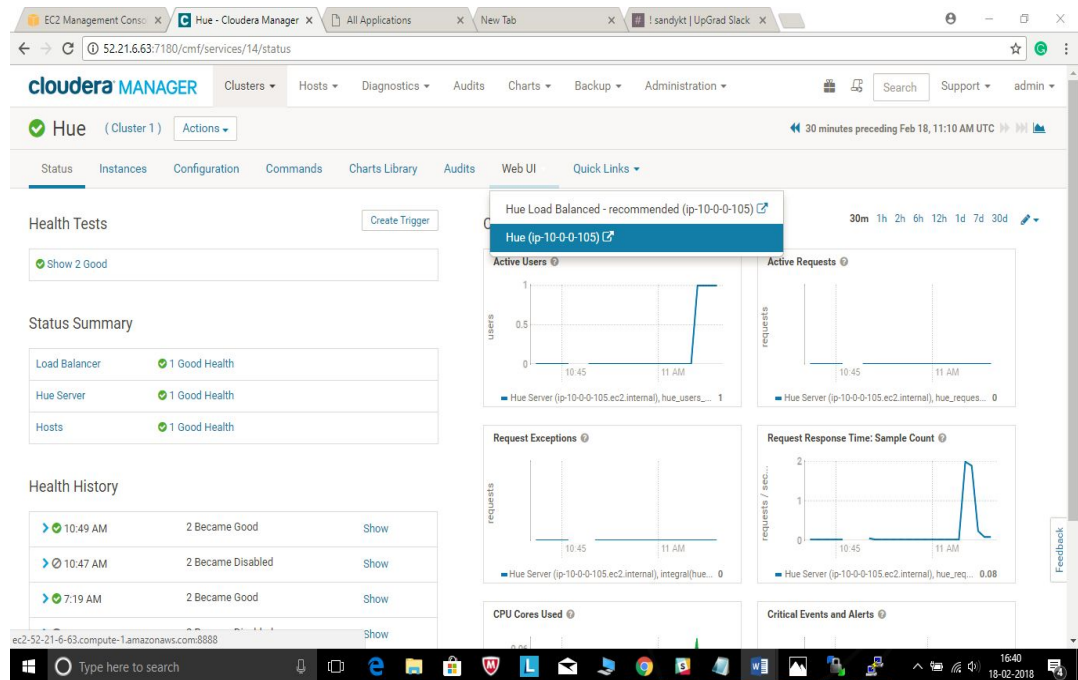
Username: admin

Password: admin
 - b. Cloudera Manager

Access Cloudera Manager by typing your Public IP address followed by '7180' as shown below:

<Public IP>:7180

2. Click on 'Web UI' and select the second option for Hue.

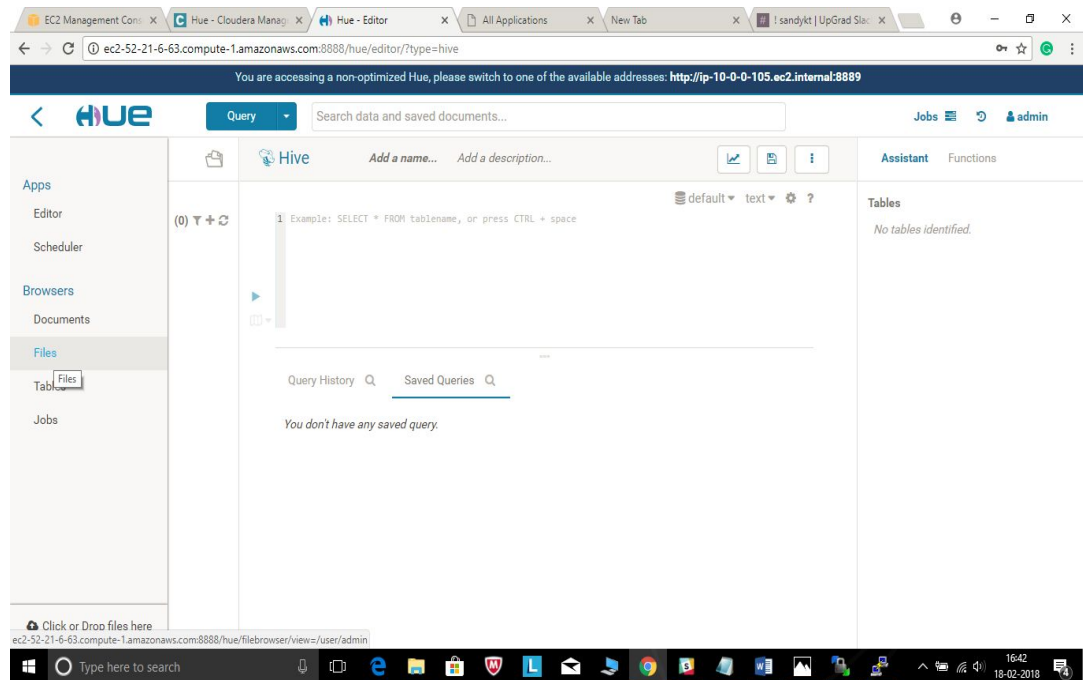


3. After clicking on Hue, login using the following credentials:

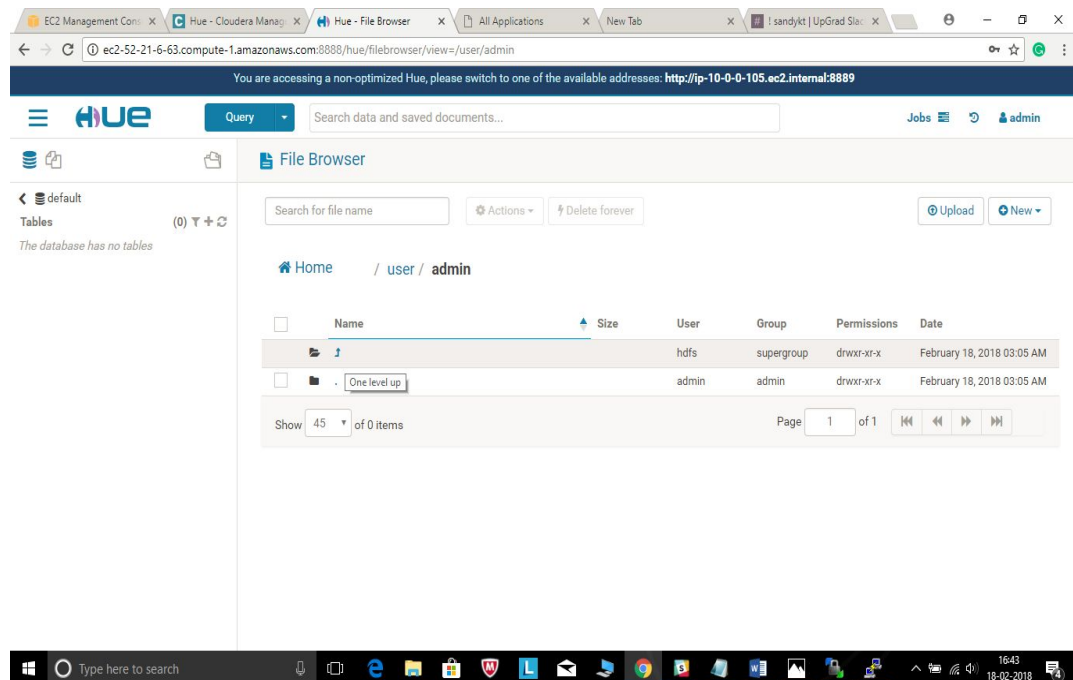
Username: admin

Password: admin

Then, click on 'Files'.



4. Click on ↑ icon as shown in the image.



5. Click on 'root'.



[Home](#) / [user](#)

<input type="checkbox"/>	Name	Size	User	Group	Permissions	Date
<input type="checkbox"/>	↑		hdfs	supergroup	drwxr-xr-x	February 17, 2018 11:16 PM
<input type="checkbox"/>	.		hdfs	supergroup	drwxr-xr-x	February 18, 2018 03:05 AM
<input type="checkbox"/>	admin		admin	admin	drwxr-xr-x	February 18, 2018 03:05 AM
<input type="checkbox"/>	history		mapred	hadoop	drwxrwxrwx	February 17, 2018 11:16 PM
<input type="checkbox"/>	hive		hive	hive	drwxrwxr-x	February 17, 2018 11:17 PM
<input type="checkbox"/>	hue		hue	hue	drwxrwxr-x	February 17, 2018 11:18 PM
<input type="checkbox"/>	oozie		oozie	oozie	drwxrwxr-x	February 17, 2018 11:18 PM
<input type="checkbox"/>	root		root	supergroup	drwxr-xr-x	February 18, 2018 02:27 AM
<input type="checkbox"/>	spark		spark	spark	drwxr-xr-x	February 17, 2018 11:17 PM

6. Click on **'wordcount'**.

[Home](#) / [user](#) / [root](#)

<input type="checkbox"/>	Name	Size	User	Group	Permissions	Date
<input type="checkbox"/>	↑		hdfs	supergroup	drwxr-xr-x	February 18, 2018 03:05 AM
<input type="checkbox"/>	.		root	supergroup	drwxr-xr-x	February 18, 2018 02:27 AM
<input type="checkbox"/>	.staging		root	supergroup	drwx-----	February 18, 2018 02:51 AM
<input type="checkbox"/>	wordcount		root	supergroup	drwxr-xr-x	February 18, 2018 02:51 AM

Show of 2 items

Page of 1

[⏪](#) [⏩](#) [⏴](#) [⏵](#)

7. Then, click on **'output'**.

[Home](#) / [user](#) / [root](#) / [wordcount](#)

<input type="checkbox"/>	Name	Size	User	Group	Permissions	Date
<input type="checkbox"/>	↑		root	supergroup	drwxr-xr-x	February 18, 2018 02:27 AM
<input type="checkbox"/>	.		root	supergroup	drwxr-xr-x	February 18, 2018 02:51 AM
<input type="checkbox"/>	input.txt	86 bytes	root	supergroup	-rw-r--r--	February 18, 2018 02:06 AM
<input type="checkbox"/>	output		root	supergroup	drwxr-xr-x	February 18, 2018 02:51 AM

Show of 2 items

Page of 1

[⏪](#) [⏩](#) [⏴](#) [⏵](#)

8. Finally, click on **'part-r-00000'** and **'part-r-00001'** and verify the output.



Home / user / root / wordcount / output

<input type="checkbox"/>	Name	Size	User	Group	Permissions	Date
<input type="checkbox"/>			root	supergroup	drwxr-xr-x	February 18, 2018 02:51 AM
<input type="checkbox"/>			root	supergroup	drwxr-xr-x	February 18, 2018 02:51 AM
<input type="checkbox"/>	_SUCCESS	0 bytes	root	supergroup	-rw-r--r--	February 18, 2018 02:51 AM
<input type="checkbox"/>	part-r-00000	49 bytes	root	supergroup	-rw-r--r--	February 18, 2018 02:51 AM
<input type="checkbox"/>	part-r-00001	44 bytes	root	supergroup	-rw-r--r--	February 18, 2018 02:51 AM

Show 45 of 3 items Page 1 of 1

EC2 Management Console | Hue - Cloudera Manager | Hue - File Browser | All Applications | New Tab | | sandykt | UpGrad Slides |

ec2-52-21-6-63.compute-1.amazonaws.com:8888/hue/filebrowser/view=/user/root/wordcount/output/part-r-00000

You are accessing a non-optimized Hue, please switch to one of the available addresses: <http://ip-10-0-0-105.ec2.internal:8889>

HUE Query Search data and saved documents... Jobs admin

File Browser

View as binary Home Page 1 to 1 of 1

Edit file / user / root / wordcount / output / part-r-00000

Download

View file location

Refresh

Last modified 02/18/2018 10:51 AM

User root

Group supergroup

Size 49 B

Mode 100644

```
The 1
course 1
entire 1
most 1
of 2
part 1
the 3
```

Type here to search

EC2 Management Console | Hue - Cloudera Manager | Hue - File Browser | All Applications | New Tab | | sandykt | UpGrad Slides |

ec2-52-21-6-63.compute-1.amazonaws.com:8888/hue/filebrowser/view=/user/root/wordcount/output/part-r-00001

You are accessing a non-optimized Hue, please switch to one of the available addresses: <http://ip-10-0-0-105.ec2.internal:8889>

HUE Query Search data and saved documents... Jobs admin

File Browser

View as binary Home Page 1 to 1 of 1

Edit file / user / root / wordcount / output / part-r-00001

Download

View file location

Refresh

Last modified 02/18/2018 10:51 AM

User root

Group supergroup

Size 44 B

Mode 100644

```
2 1
big 1
data 1
important 1
is 1
program 2
```

Type here to search