# Hadoop Training Commands

## Module 1: Linux Intro – Very basic Linux commands

### ### list files and directories in the current directory

**ls**

### ### list files with additional attributes like date modified, owner, group size. Everything is case sensitive

**ls -l**

### ### list all files including hidden ones

**ls -al**

### ### Change directory. Case sensitive command

**cd linux\_basics**

### ### The cat command outputs all the text of the file on the screen. This is useful for viewing short files on the command prompt

**cat output.txt**

### ### Clear the current output on the screen

**clear**

### ### Rename the file using the mv command

**mv output.txt input.txt**

### ### Create directory using the mkdir command

**mkdir test**

### ### Show the color of the test directory

**ls -al**

### ### linux is very strict about ownership. Chmod changes permissions of the file to make it writable by everyone (777 permissions)

**chmod 777 -R test/**

### ### Change to root user using sudo su

**sudo su**

**cd /var**

**touch tmp.txt**

### ### Save the file as root and switch to cloudera user and try to access the file. It won’t let you save

**su cloudera**

**cd /var**

**vim tmp.txt**

**:wq! (exit without saving)**

### ### Now try to take the ownership of the tmp.txt file

**chown cloudera:cloudera tmp.txt**

**sudo chown cloudera:cloudera tmp.txt**

**vim tmp.txt**

**:wq**

### ###Chaining Commands together

**cd ~/linux\_basics**

**echo 'hello world' && ls –al**

### ### Pipe Operator for piping output of one command into another program

**ls -al | grep inp**

### ### Redirection Operator for redirecting output into a file

**echo 'hello world' > hello.txt**

**echo 'some more' > hello.txt**

**cat hello.txt**

### ###For Appending Text instead of overriding

**echo 'This line is for appending text' >> hello.txt**

**cat hello.txt**

## Module 2: Interacting with HDFS FileSystem

**hadoop fs -ls /user**

**hadoop fs -copyFromLocal ~/linux\_basics/input.txt /user/cloudera/**

**hadoop fs -cat /user/cloudera/input.txt**

**hadoop fs -rm /user/cloudera/input.txt**

**hadoop fs -copyToLocal /user/cloudera/input.txt ./inputcopy.txt**

## Module 3: Running a MapReduce Program. The File (WordCount.java) is attached with the mail

**hadoop fs -mkdir /user/cloudera/wordcount /user/cloudera/wordcount/input**

**mkdir -p build**

**javac -cp /usr/lib/hadoop/\*:/usr/lib/hadoop-mapreduce/\* WordCount.java -d build –Xlint**

**jar -cvf word\_count.jar -C build/ .**

**echo "Hadoop is an elephant" > file0**

**echo "Hadoop is as yellow as can be" > file1**

**echo "Oh what a yellow fellow is Hadoop" > file2**

**hadoop fs -put file\* /user/cloudera/wordcount/input**

**hadoop jar word\_count.jar org.myorg.WordCount /user/cloudera/wordcount/input /user/cloudera/wordcount/output**

**hadoop fs -cat /user/cloudera/wordcount/output/\***

### Module 4: Interacting with Hive. The Sample students.csv is attached with the mail

**SHOW TABLES;**

**CREATE TABLE students(name string, age int)**

**ROW FORMAT DELIMITED**

**FIELDS TERMINATED BY ','**

**STORED AS TEXTFILE;**

**LOAD DATA LOCAL INPATH '/home/cloudera/hadoop\_basics/students.csv' OVERWRITE INTO TABLE students;**

**SELECT \* from Students;**

**select \* from students where name like '%doe';**

**select \* from students where age > 34;**

**select \* from students where age > 34 ORDER by age;**

**select avg(age) from students;**

**DROP TABLE students;**

## Module 5 : Using Impala instead of Hive to interact with the HDFS data

**INVALIDATE METADATA;**

**SELECT \* FROM STUDENTS;**

**select \* from students where age > 30;**

**select \* from students where age > 35 ORDER BY age;**

## Module 6: Using Pig to read unstructured data (or structured). The Students.csv is attached with the mail

**students = LOAD '/home/cloudera/hadoop\_basics/students.csv' using PigStorage(',')**

**AS (name:chararray, age:int);**

**DESCRIBE students**

**DUMP students;**

**names = FOREACH students GENERATE name;**

**DUMP names;**

**older = FILTER students by age > 35;**

**DUMP older;**

PIG PROGRAM TO FIND WORDCOUNT

lines = LOAD '/user/hadoop/HDFS\_File.txt' AS (line:chararray);

words = FOREACH lines GENERATE FLATTEN(TOKENIZE(line)) as word;

grouped = GROUP words BY word;

wordcount = FOREACH grouped GENERATE group, COUNT(words);

DUMP wordcount;