

MPA-1 Big data for Machine Learning

Group-15

Tania Khan

V S Shabarish

Vinod A

Submitted on

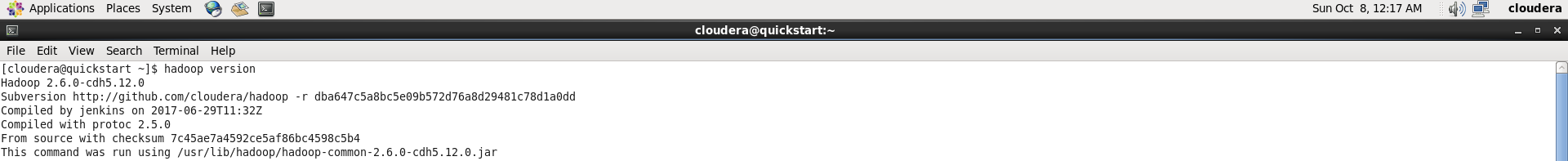
08-10-2023

# Write HDFS shell commands for the following. (18 marks):

## 1.To Print Version of installed Hadoop (1 mark)

To check the version of Hadoop installed in the cloudera machine, we use the command line argument in the cloudera shell or command prompt called as Hadoop version.

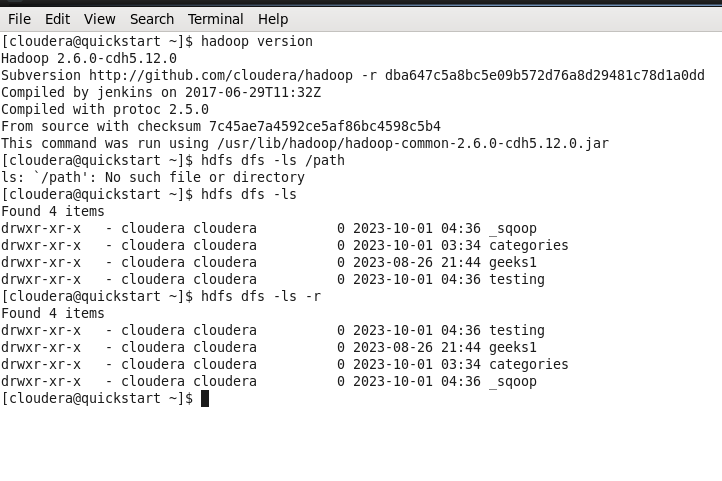
The command will be written in shell as shown in the below image.



***Output:*** The output shows the current version of Hadoop installed in the Virtual machine along with its compilation and packaging methods used for installing and deploying Hadoop in the specific virtual machine.

## 2. For listing the files and directories present in HDFS ‘path’ directory present under root (/path) (1 mark)

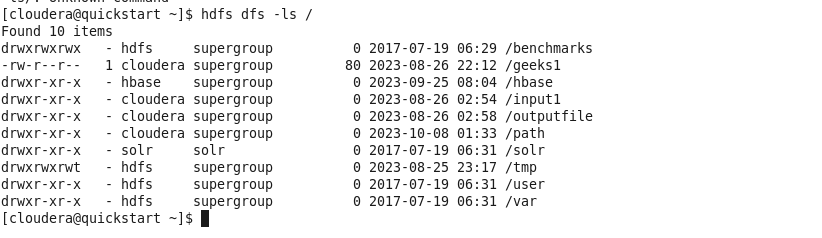
To list the files present under HDFS we command line argument in the cloudera shell or command prompt called as hdfs dfs –ls <path>.

In the below image we can see that there is no folder called path hence now we will make a folder named path and create a file inside the new folder.

Now we are creating a new file called path in HDFS distributed file system.



Now we execute the below command we can see the path folder listed by hdfs.

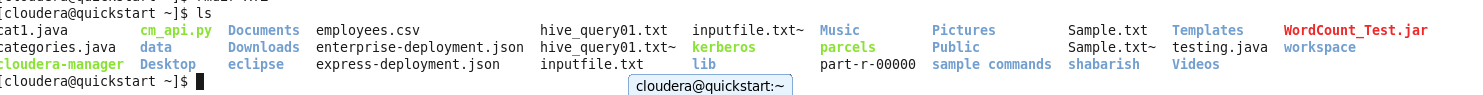


Output: The output shows the list of files present inside the HDFS dfs base directory.

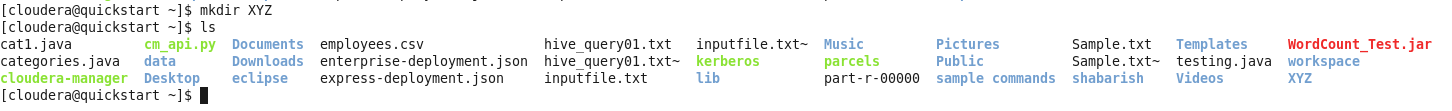
## 3. To Delete an empty directory named as XYZ.

### 3.1 Under normal Directory

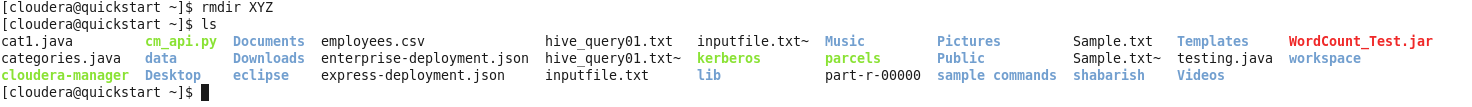
To check the existing files we use the ***ls*** command



We create the new folder named XYZ using mkdir

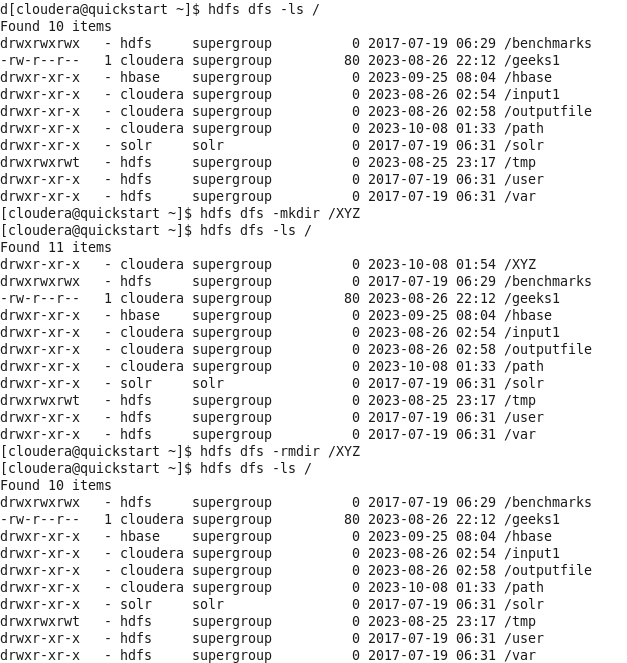


Now to delete the folder we use the command rmdir



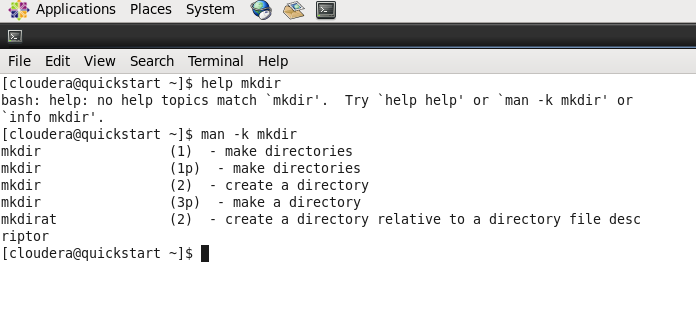
### 3.2 Under HDFS

We follow the same commands as used above with the addition of hdfs dfs – in front of each commands used the results can be seen below.



## 4.To fetch the usage instructions of mkdir command .

To get the usage of mkdir command we use man –k mkdir.

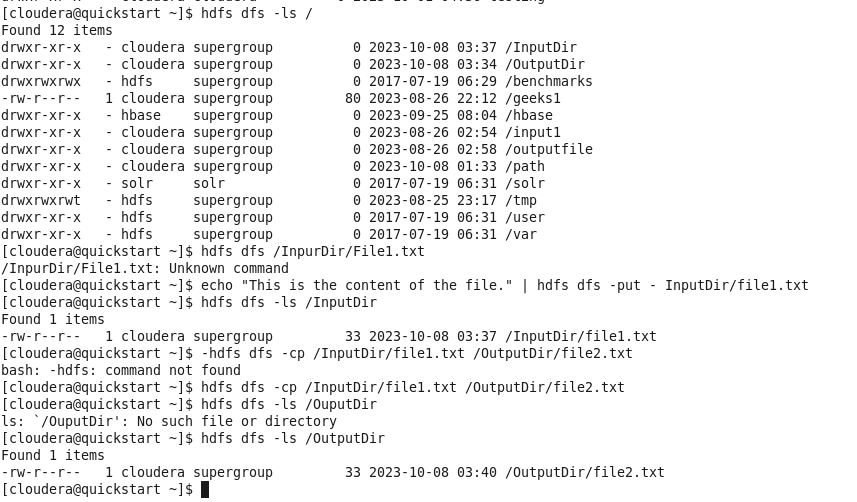


## 5. To Copy ‘file1.txt’ from ‘InputDir’ to ‘OutputDir’ as file2.

To copy and rename an input file to output file path we use below command.

* hdfs dfs -cp /InputDir/file1.txt /OutputDir/file2.txt

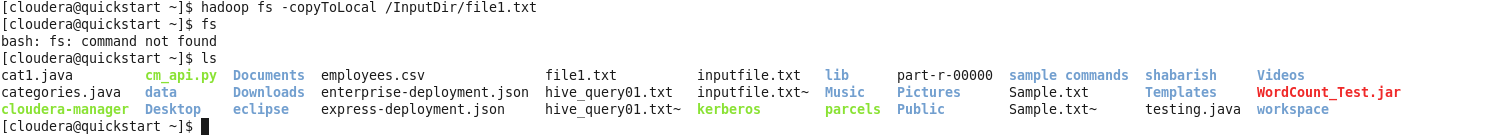
The below image shows the sample file moved from InputDir to OutputDir



## 6.Write command for copy the ‘testfile’ of the hadoop filesystem (present under root) to the local file system (pwd).

To copy a file named 'testfile' from the Hadoop Distributed File System (HDFS) to the local file system in your current working directory, you can use the `hadoop fs -copyToLocal` command. Here's the command:

hadoop fs -copyToLocal /testfile .



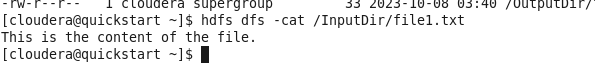
In the above image we are copying a file called file1.txt from the inputDir in HDFS to the local path.

After running this command, 'testfile' from HDFS will be copied to your current working directory in the local file system.

## 7.Write command for display the content of the ‘sample’ file present in newDataFlair directory of HDFS (under root).

* hdfs dfs -cat /newDataFlair/sample

here we are opening a file named file1.text and viewing its contents.



To display the content of the 'sample' file present in the 'newDataFlair' directory of HDFS (under root), you can use the hdfs dfs -cat

## 8.Write command for copy local file named file1 under the present working directory(pwd) of local file system to the Hadoop filesystem under root. (3 marks)

* hdfs dfs -copyFromLocal file1 /
* **file1** is the name of the local file we want to copy.
* **/** represents the root directory in HDFS where we want to copy the file.

After running this command, the 'file1' from the local file system will be copied to the root directory of HDFS.

# Illustrate how MapReduce processes WordCount job ? (7 marks)

The WordCount example is a classic illustration of how MapReduce processes work in Hadoop. It's a simple program used to count the frequency of each word in a large collection of text documents. Here's how a WordCount job is processed in MapReduce:

Consider Sample Input:

"Apple Orange Mango

Orange Grapes Plum”

“Apple Plum Mango

Apple Apple Plum”

Below is the illustration of the above Word count problem processed using Map-Reduce paradigm.

A diagram of a business process

Description automatically generated

***1. Input Data Splitting (Map Input):*** The input data, which is typically a collection of text documents, is split into smaller chunks called Input Splits. Each Input Split represents a portion of the input data and is assigned to a map task. The size of Input Splits is configurable.

***2. Map Phase (Mapper):*** Map tasks are responsible for processing each Input Split. In the WordCount example, the Mapper's main function is to tokenize the text, split it into words, and emit a key-value pair for each word encountered. The key is the word itself, and the value is set to 1 (indicating the presence of one occurrence of the word).

Mapper Output: ("Apple", 1), ("Orange", 1), ("Mango", 1), ("Orange", 1), ("Grapes", 1), ("Plum", 1), ("Apple", 1), ("Plum", 1), ("Mango", 1), ("Apple", 1), ("Apple", 1), ("Plum", 1)

***3. Shuffle and Sort Phase:*** After the Map phase, the framework performs a shuffle and sort operation. It collects all the key-value pairs emitted by the Mappers, groups them by keys, and sorts them. This step ensures that all occurrences of the same word are grouped together.

Example Intermediate Key-Value Pairs:

- ("Apple", [1, 1,1,1])

- ("Grapes", [1])

- ("Mango", [1,1])

- ("Orange", [1,1])

- ("Plum", [1,1,1])

***4. Reduce Phase (Reducer):*** Reduce tasks receive the sorted key-value pairs from the shuffle and sort phase. For each unique key, the Reducer aggregates the values (counts) associated with that key. In the WordCount example, it sums up the counts for each word to calculate the total frequency.

Example Reducer Output:

- ("Apple", 4)

- ("Grapes", 1)

- ("Mango", 2)

- ("Orange", 2)

- ("Plum", 3)

***5. Output:*** The final output, which consists of key-value pairs representing each unique word and its count, is typically written to an output directory in HDFS or another storage system.

Example Output:

```

Apple 4

Grapes 1

Mango 2

Orange 2

Plum 2

```

***6. Job Completion:*** Once all the Map and Reduce tasks have completed, the MapReduce job is considered finished, and the output is available for further processing or analysis.

This is a high-level overview of how a WordCount job is processed in Hadoop MapReduce. MapReduce can be scaled to handle large volumes of data and distributed across a cluster of machines, making it suitable for big data processing tasks.