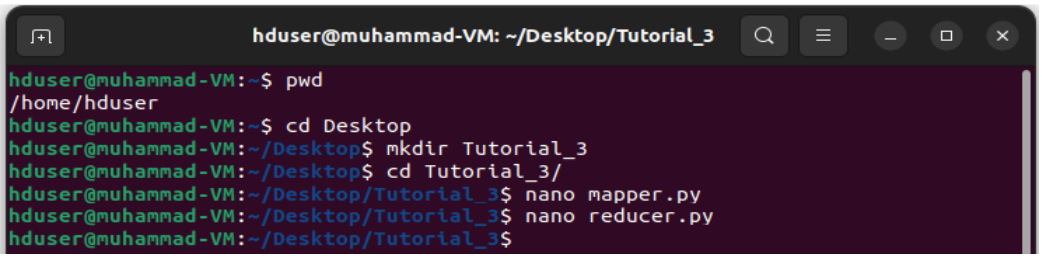
Tutorial 3

Hadoop Streaming using MapReduce Model

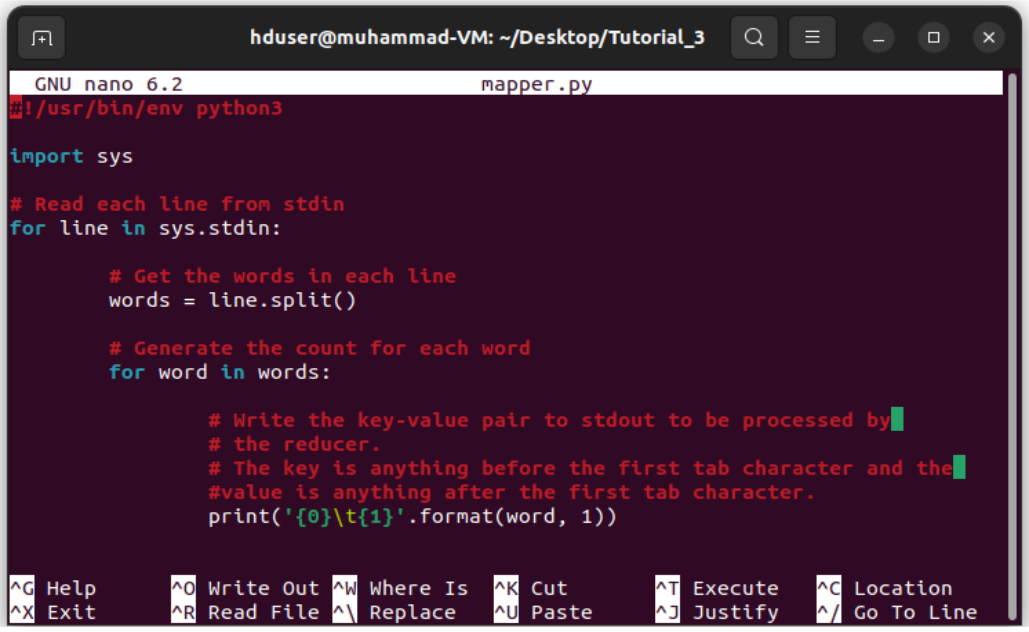
To demonstrate how the Hadoop streaming utility can run Python code as a MapReduce application on the Hadoop cluster, the **WordCount** application can be implemented as two Python programs: (**`mapper.py`** and **`reducer.py`**). **`mapper.py`** is the Python code that implements the logic in the map phase of **WordCount**. It reads data from **stdin**, splits the lines into words, and outputs each word with its intermediate count to **stdout**.

Create a folder Tutorial_3 on Ubuntu Desktop or any other folder name of your choice and write the Python code in the following files named as **`mapper.py`** and **`reducer.py`**. The below screenshot showed the commands to create the folder (Tutorial_3) on Ubuntu Desktop. Then create the file mapper.py and type the code and similarly for reducer.py.



<u>Part I:</u> WordCount Frequency

1) Type the following Python code in the files in the Ubuntu VM using **nano/ gedit** editor as mentioned in the screenshot.



symbol represents the comments in the Python.

reducer.py

```
hduser@muhammad-VM: ~/Desktop/Tutorial_3
  GNU nano 6.2                    reducer.py *
#!/usr/bin/env python3

import sys

curr_word = None
curr_count = 0

# Process each key-value pair from the mapper
for line in sys.stdin:

        # Get the key and value from the current line
        word, count = line.split('\t')

        # Convert the count to an int
        count = int(count)

        # If the current word is the same as the previous word,
        # increment its count, otherwise print the words count to stdout
        if word == curr_word:
                curr_count += count
        else:
                # Write word and its number of occurrences as a key-value
                # pair to stdout
                if curr_word:
                        print('{0}\t{1}'.format(curr_word, curr_count))

                curr_word = word
                curr_count = count

# Output the count for the last word
if curr_word == word:
        print('{0}\t{1}'.format(curr_word, curr_count))

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

2) Before hadoop streaming, we provide the executable privileges to **`mapper.py`** and **`reducer.py`** files by using the command. (Check Tutorial 1 for further details).

`$chmod 700 mapper.py`

`$chmod 700 reducer.py`

```
hduser@muhammad-VM: ~/Downloads/Tutorial_3
hduser@muhammad-VM:~$ cd Downloads
hduser@muhammad-VM:~/Downloads$ cd Tutorial_3/
hduser@muhammad-VM:~/Downloads/Tutorial_3$ ls -l
total 144
-rw------- 1 hduser hadoopgroup 135287 Feb 25  2022 britney-spears.txt
-rw------- 1 hduser hadoopgroup    426 Feb 25  2022 mapper.py
-rw------- 1 hduser hadoopgroup    744 Feb 25  2022 reducer.py
hduser@muhammad-VM:~/Downloads/Tutorial_3$ chmod 700 mapper.py
hduser@muhammad-VM:~/Downloads/Tutorial_3$ chmod 700 reducer.py
hduser@muhammad-VM:~/Downloads/Tutorial_3$ ls -l
total 144
-rw------- 1 hduser hadoopgroup 135287 Feb 25  2022 britney-spears.txt
-rwx------ 1 hduser hadoopgroup    426 Feb 25  2022 mapper.py
-rwx------ 1 hduser hadoopgroup    744 Feb 25  2022 reducer.py
hduser@muhammad-VM:~/Downloads/Tutorial_3$
```

Check that the python code is working correctly or not. Use the following Linux command on Ubuntu terminal/ shell as mentioned below

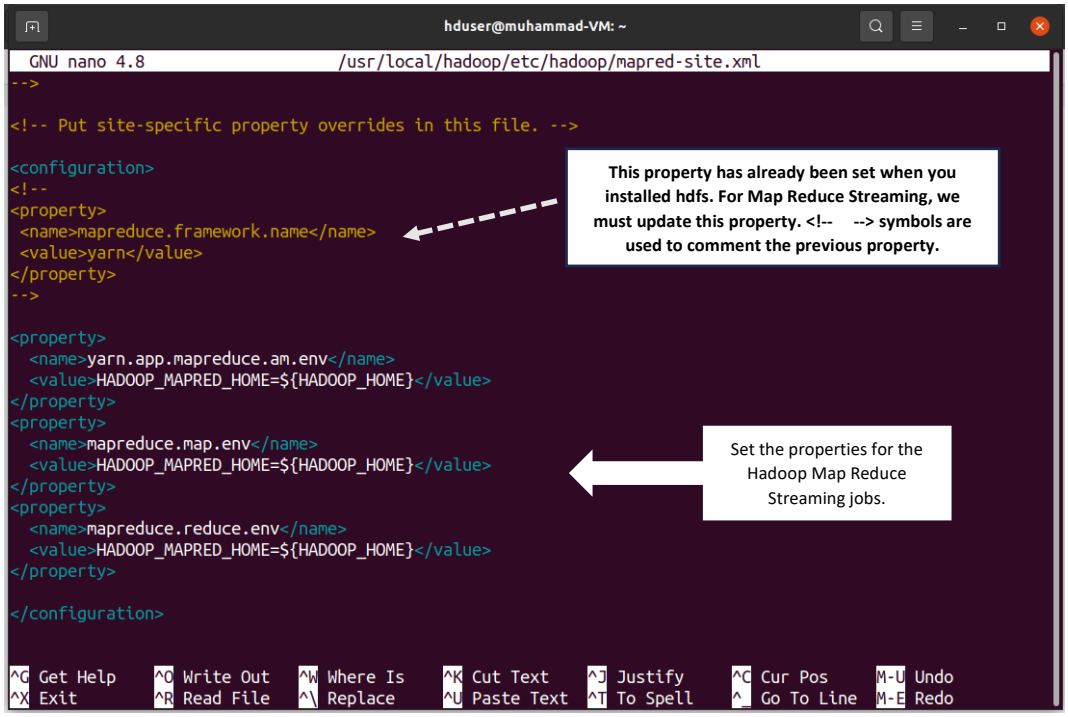
```
hduser@muhammad-VM: ~/Downloads/Tutorial_3
hduser@muhammad-VM:~/Downloads/Tutorial_3$ echo 'jack be nimble jack be quick' | ./mapper.py | sort -t 1 | ./reducer.py
be      2
jack    2
nimble  1
quick   1
hduser@muhammad-VM:~/Downloads/Tutorial_3$
```

This shows that the syntax of the python code is correct. If you are facing difficulty in writing the Python code, you can download the files from Moodle and copy them into Tutorial_3 folder.

3) The MapReduce framework's streaming functionality is one of the properties that can be configured in Hadoop using the **`mapred-site.xml`** file. Without writing a Java code, Hadoop Streaming is a tool that lets you design and execute MapReduce tasks using arbitrary executables as the mapper and/or reducer. Before the execution of Hadoop streaming, you must update **mapred-site.xml** file in the folder at the location **/usr/local/hadoop/etc/hadoop/**

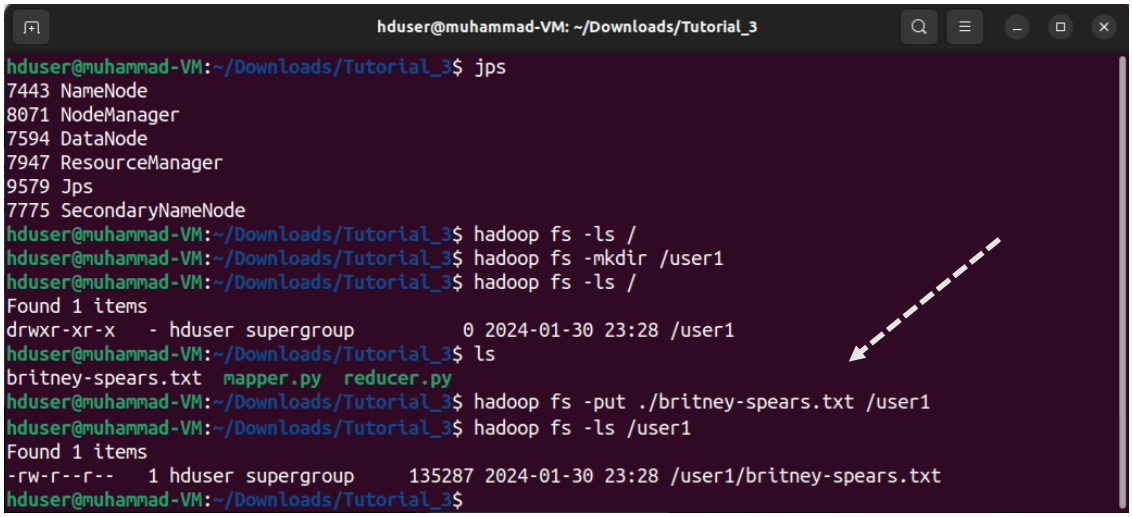
Comment the already written property in the **`mapred.xml`** file and write the below mentioned code. The file after update looks like as mentioned below

nano /usr/local/hadoop/etc/hadoop/mapred-site.xml



Comment the previous property in the **mapred-site.xml** file and append as the three new properties for Hadoop environment as shown in the above screenshot. These are essential for the execution of Hadoop streaming jobs.

4) Download a text input file from the Moodle named as **britney-spears.txt** (Song lyrics). This file will store in the Downloads folder in Ubuntu VM automatically. Move this text file from your Downloads folder into **hdfs** for Map Reduce streaming jobs. The steps to move the file from Downloads folder to the "user1" folder on **hdfs** is shown below

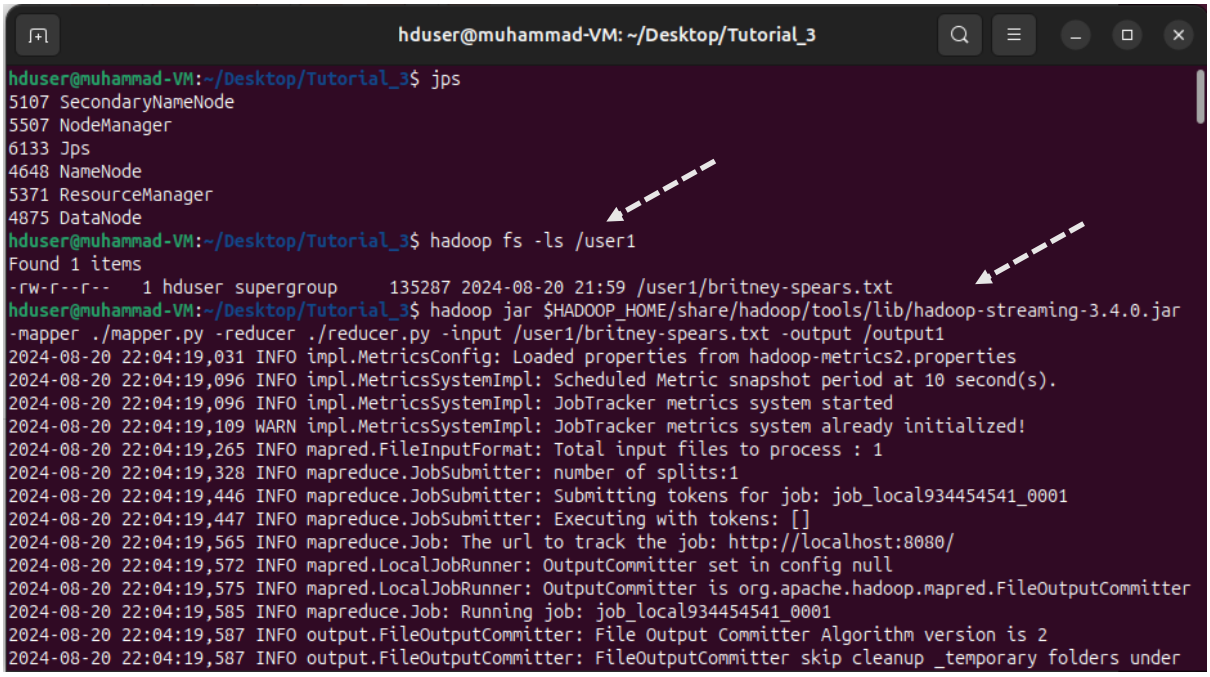


```
$cd Downloads
$hadoop fs -put ./britney-spears.txt /user1
$hadoop fs -ls /user1
```

If the **user1** is not present on hadoop distributed file system (hdfs), create a folder **user1** on the Hadoop distributed file system using the command.

```
$hadoop fs -mkdir /user1
```

5) Now the input file (**britney-spears.txt**) is ready for the Hadoop streaming, type the following command as mentioned below highlighted with an arrow

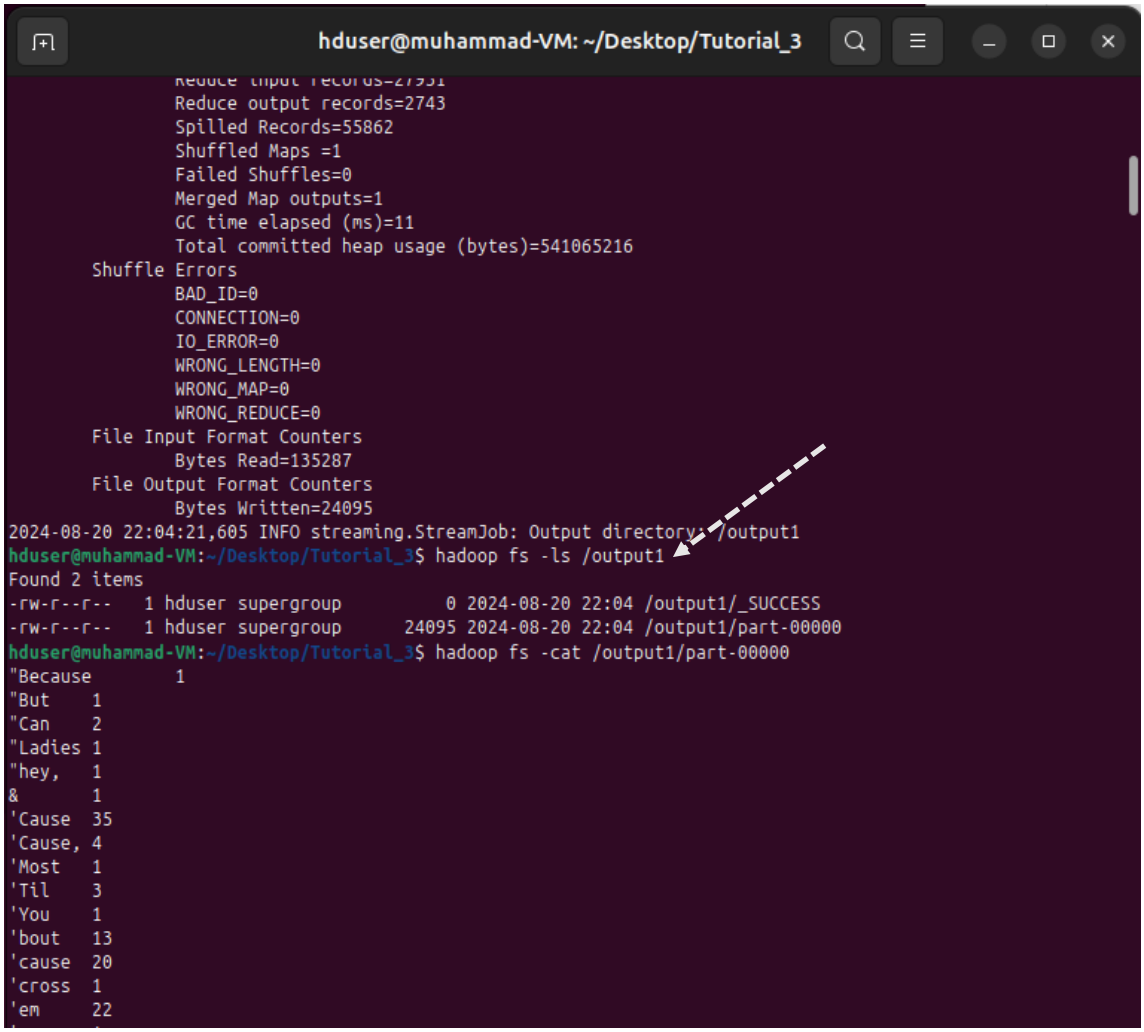


6) The separate command for Hadoop streaming from the above screenshot is mentioned below

Note: The last part of the message will be displayed after the successful execution of Hadoop streaming job as mentioned below

7) The output of the Hadoop streaming job will be stored in the output folder (named as "**output1**" in the above command) on the Hadoop cluster. It is not possible to reuse or rewrite into the output1 folder name on hdfs. It is necessary for either creating a new folder name or delete the existing one and start over.

You can display the contents of the files by using the following commands as mentioned below using a dotted-dashed arrow as shown in the screenshot.



The wordcount example using Hadoop streaming is successfully completed as clearly illustrated in the output as clearly shown in the screenshot.

You have finished a successful execution of Hadoop streaming job. Now you can consider a large data input file and count the frequency of various words.

- If you would like to see the details of Hadoop cluster, you can explore further details of hadoop cluster by using **localhost:9870** on the web browser (mozilla firefox or google chrome on Ubuntu VM). Explore different options of hadoop in the browser.

8) If you would like download the processed data from the hdfs drive to local drive for your continuous assessment. You can use the following commands as mentioned below

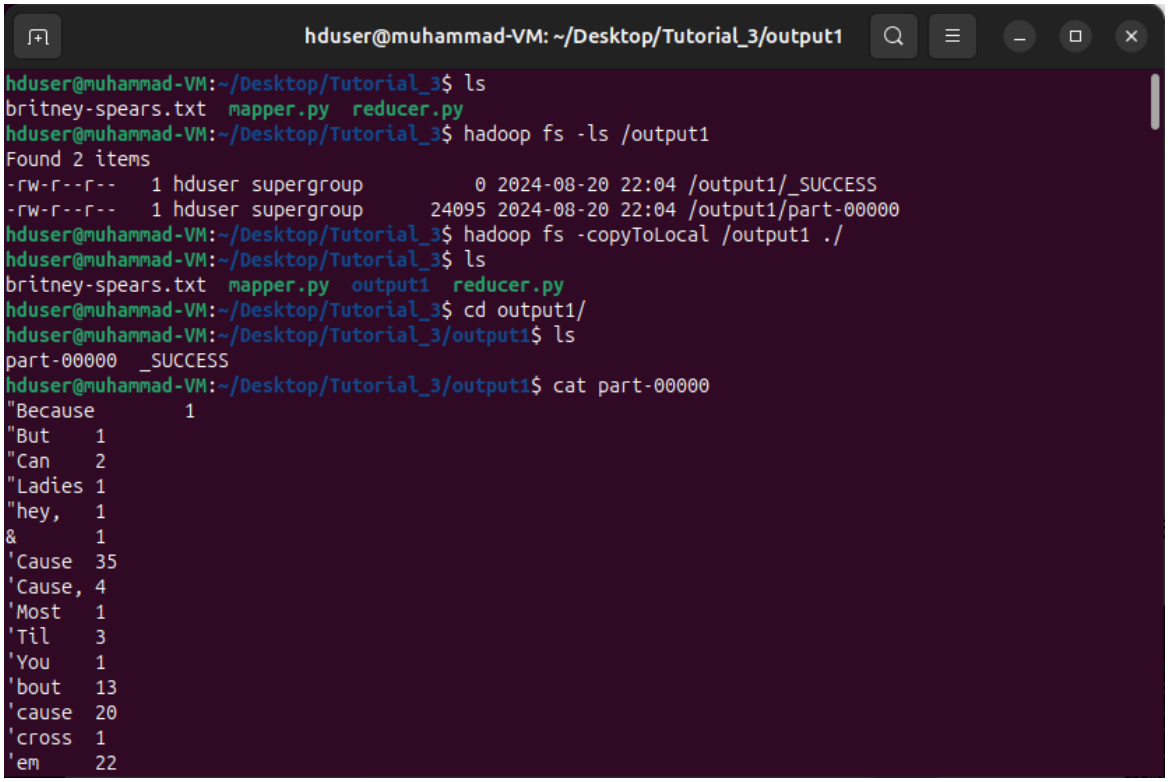
```
$cd /home/hduser/Desktop/Tutorial_3/
$hadoop fs -ls /output1
```

This command will copy the folder from hdfs to your local Tutorial_3 folder on Ubuntu VM.

```
$hadoop fs -copyToLocal /output1 ./
$ls
$cd output1
```

```
$cat part-00000
```

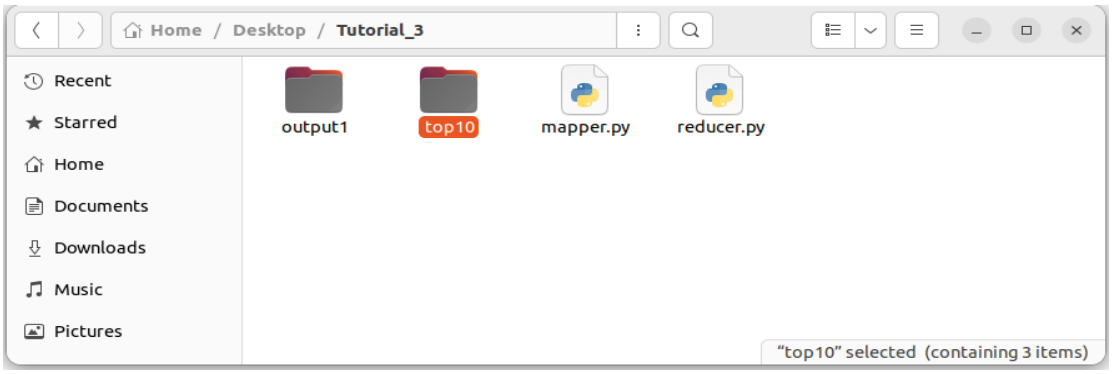
The screenshot for the above-mentioned commands is shown below.



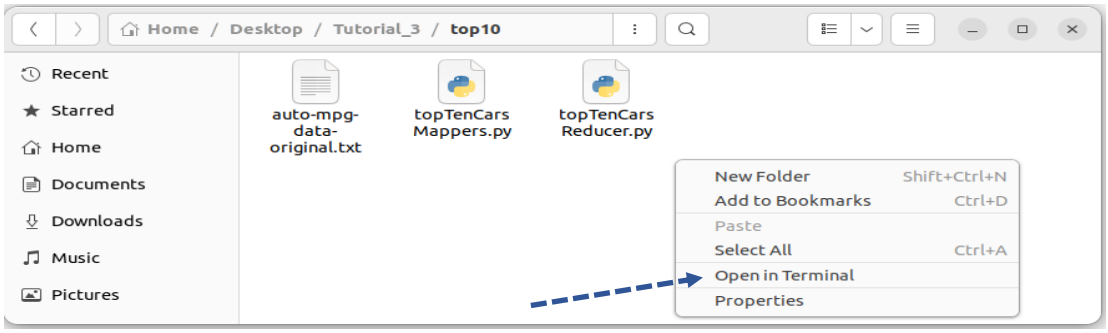
Now create another folder in Tutorial_3 and named as "**top10**".

Part II: Top10 Design Pattern

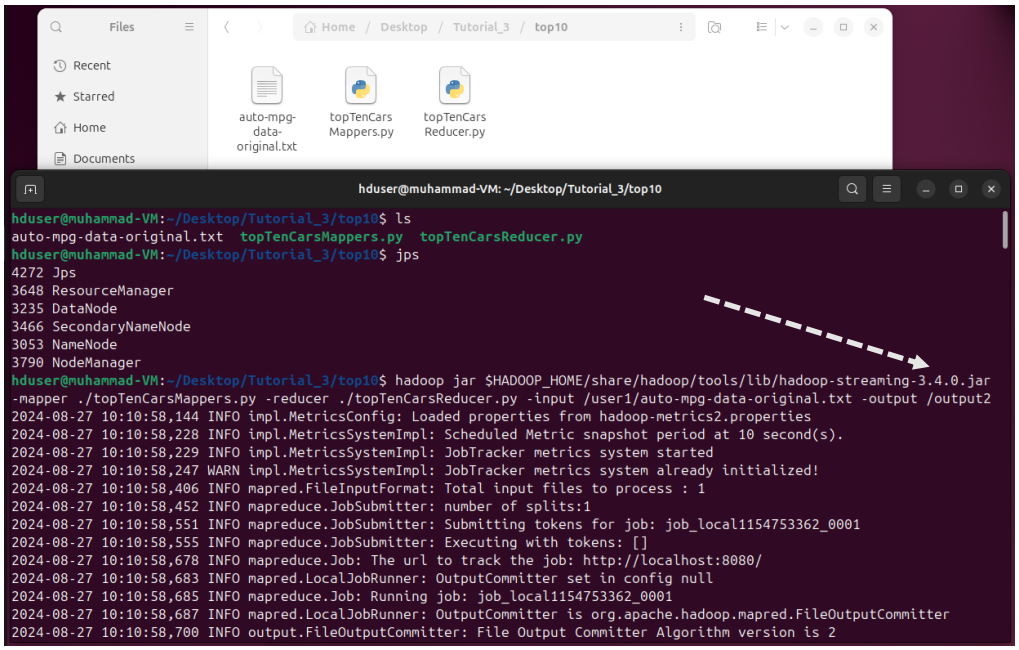
1) Download the folder **top10** MapReduce Design pattern from the Moodle. Unzip the folder and move the folder to **Tutorial_3** on Ubuntu Desktop as shown below.



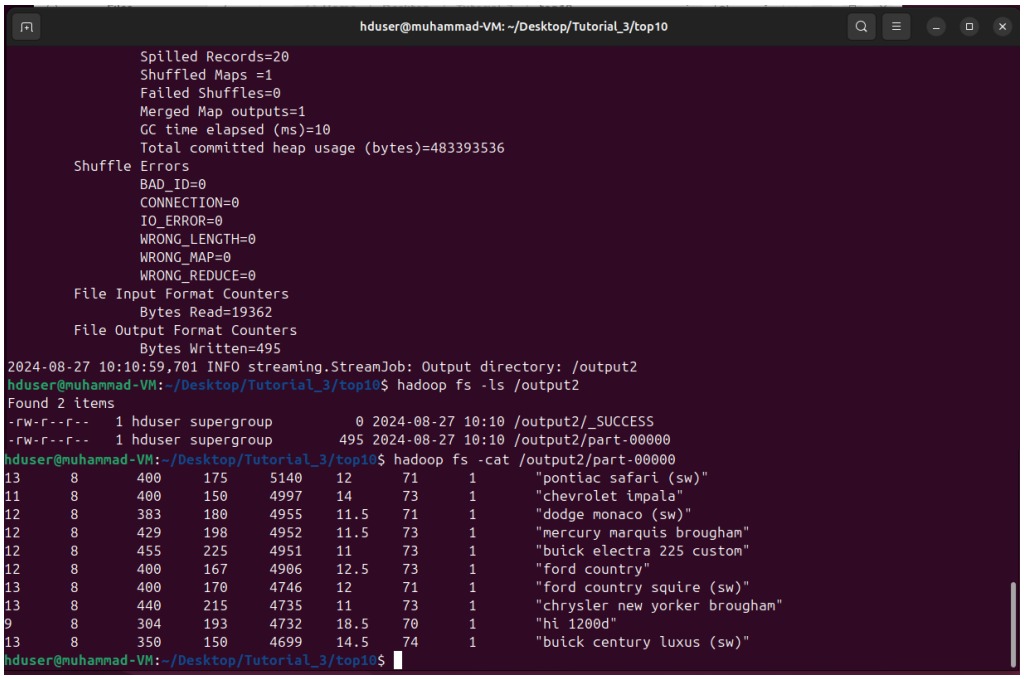
When you open **top10** folder inside **Tutorial_3**, three files (Mapper, Reducer Python files and text file) are present as shown below



Right click inside the folder and open the terminal as shown below



2) Ater successful completion of MapReduce job, the following screenshot shows that **output2** folder is created on **hdfs**. We can display the output by using the hadoop **`$ls`** command.



3) The output is based on the top 10 records based on the dataset as shown in the above screenshot.

References:

- https://hadoop.apache.org/docs/current/hadoop-streaming/HadoopStreaming.html
- https://github.com/Virksaabnavjot/Mapreduce-Python/tree/master