KIT-KALAIGNARKARUNANIDHI INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)
(Accredited by NAAC&NBA with 'A' Grade)
Approved by AICTE & Affiliated to Anna University, Chennai)
Kannampalayam Post, Coimbatore-641402

Department of Artificial Intelligence and Data Science

B19ADP601 – BIG DATA ANALYTICS LABORATORY

Name : ……

Batch : …… Reg.No : ……..

Branch : ……. Year : ……

KIT-KALAIGNARKARUNANIDHI INSTITUTE OF TECHNOLOGY
(ANAUTONOMOUS INSTITUTION)
(Accredited by NAAC & NBA with 'A' Grade)
Approved by AICTE & Affiliated to Anna University, Chennai)
Kannampalayam Post, Coimbatore-641402

Department of Artificial Intelligence and Data Science

BONAFIDE CERTIFICATE

Name : ….……

Roll No. : …… Reg. No.: ……

Branch : B.Tech – Artificial Intelligence and Data Science

Certified that this is Bonafide record work done by Mr./Ms. ……

of III – Year Artificial Intelligence and Data Science during the academic year 2023-2024.

FACULTY IN-CHARGE **HOD**

Submitted for the University Practical Examination held on …….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Instructions for Laboratory Classes

1. Enter the lab with record work book & necessary things.
2. Enter the lab without bags and footwear.
3. Footwear should be kept in the outside shoe rack neatly.
4. Maintain silence during the Lab hours.
5. Read and follow the work instructions inside the laboratory.
6. Handle the computer systems with care.
7. Shutdown the Computer properly and arrange chairs in order before leaving the lab.
8. The program should be written on the left side pages of the record workbook.
9. The record work book should be completed in all aspects and submitted in the next class itself.
10. Experiment number with date should be written at the top left-hand corner of the record work book page.
11. Strictly follow the uniform dress code for Laboratory classes.
12. Maintain punctuality for lab classes.
13. Avoid eatables inside and maintain the cleanliness of the lab.

VISION

To produce competent professionals to the dynamic needs of the emerging field of Artificial Intelligence and Data Science

MISSION

- To empower students with the knowledge and skills necessary to create intelligent systems and innovative solutions that address societal issues.
- Providing technical knowledge on par with Industry to the students through qualified faculty members having knowledge in recent trends and technologies.
- To produce competent engineers who are both professional and life-skills oriented.
- Providing opportunities for students to improve their research skills in order to address a variety of societal concerns through innovative projects.

PROGRAMMEOUTCOMES (POs)

Students graduating from Artificial Intelligence and Data Science should be able to:

PO1 Engineering knowledge : Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex Artificial Intelligence and Data Science problems.

PO2 Problem analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and Artificial Intelligence and Data Sciences.

PO3 Design/development to solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations in the field of Artificial Intelligence and Data Science.

PO4 Conduct investigations of complex problems: Using research-based knowledge and Artificial Intelligence & Data Science oriented research methodologies including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5 Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex Artificial Intelligence and Data Science Engineering activities with an understanding of the limitations.

PO6 The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7 Environment and sustainability: Understand the impact of the professional Artificial Intelligence and Data Science Engineering solutions in societal and environmental contexts, and demonstrate the knowledge, and need for the sustainable development.

PO8 Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9 Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

PO10 Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 Projectmanagementandfinance:Demonstrate knowledge and understanding of the Artificial Intelligence and Data Science engineering and management principles and apply these to one's own work, as a member and leader in a team and, to manage projects in multidisciplinary environments.

PO12 Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will have a strong foundation in mathematics, programming, machine learning, artificial intelligence, and data science, as well as advanced skills in these areas to solve technical problems.

PEO2: Graduates will have the capability to apply their knowledge and skills to identify and solve the issues in real world Artificial Intelligence and Data Science related applications.

PEO3: Graduates will be able to engage in life-long learning by completing advanced software Technologies, certificates, and/or other professional development.

PROGRAM SPECIFIC OUTCOME(PSOs)

Graduates of Artificial Intelligence and Data Science Programmed should be able to:

PSO1: Apply fundamental concepts of Artificial Intelligence and Data Science according to the environmental needs.

PSO2: Ability to develop skills to address and solve Artificial Intelligence based social and environmental problem using Data Science to deal multidisciplinary projects using modern tools.

COURSE OUTCOMES:

Students will be able to:

Course Outcome	Knowledge Level
CO1: Apply the Perform setting up and installing Hadoop in its three operating modes.	K3
CO2: Implement the file management tasks in hadoop	K3
CO3: Build the Map Reduce Paradigm.	K3
CO4: Make use of pig latin scripts sort, group, join, project, and filter your data.	K3
CO5: Experiment with the installation of HIVE.	K3

CO/PO &PSO		PO1 (K3)	PO2 (K4)	PO3 (K5)	PO4 (K5)	PO5 (K6)	PO6 (K3) (A3)	PO7 (K2) (A3)	PO8 (K3) (A3)	PO9 (A3)	PO10 (A3)	PO11 (K3) (A3)	PO12 (A3)	PSO1 (K3,A3)	PSO2 (K3,A3)
CO1	K3	3	3	2	1	3	-	-	-	-	-	-	2	3	3
CO2	K3	3	3	2	1	3	-	-	-	-	-	-	2	3	3
CO3	K3	3	3	2	1	3	-	-	-	-	-	-	2	3	3
CO4	K3	3	3	2	1	3	-	-	-	-	-	-	2	3	3
CO5	K3	3	3	2	1	3	-	-	-	-	-	-	2	3	3
Weighted Average		3	3	2	1	3	-	-	-	-	-	-	2	3	3

SYLLABUS

LIST OF EXPERIMENTS

1. **HADOOP INSTALLATION.**
2. **FILE MANAGEMENT IN HADOOP.**
3. **MAP REDUCE PROGRAM – WORD COUNT.**
4. **MAP REDUCE PROGRAM – WEATHER DATA.**
5. **MAP REDUCE PROGRAM – MATRIX MULTIPLICATION.**
6. **INSTALLATION OF PIG AND HIVE.**
7. **PIG LATIN SCRIPTS – WORD COUNT & TO FIND MAXIMUM TEMPERATURE FOR EACH AND EVERY YEAR.**
8. **HIVE FUNCTIONS.**

Total hours: 45

Practical Record Book Index Page

Sl. No.	Date	Name of the Experiment	Page Number	Aim & Algorithm (20 Marks)	Program (30 Marks)	Output & Inference (15 Marks)	Viva-Voce (10Marks)	Total (75Marks)	Signature of the Faculty Member

Model Exam Marks (25):_____Total (100):_____

Signature of the Faculty Member

Practical Record Book Index Page

Sl. No.	Date	Name of the Experiment	Page Number	Aim & Algorithm (20 Marks)	Program (30 Marks)	Output & Inference (15Marks)	Viva-Voce (10Marks)	Total (75Marks)	Signature of the Faculty Member

Model Exam Marks (25):_____Total (100):____

Signature of the Faculty Member

S.NO	EXPERIMENT	PREREQUISITES	LEARNING OBJECTIVES
1	FOR ALL EXPERIMENTS	PROGRAMMING IN PYTHON	1. To optimize business decisions and create competitive advantage with Big data analytics 2. To practice concepts required for developing map reduce programs. 3. To impart the architectural concepts of Hadoop and introducing map reduce paradigm. 4. To practice programming tools PIG and HIVE in Hadoop eco system. 5. To implement best practices for Hadoop development.

Ex. No:	1	HADOOP INSTALLATION
Date:		

AIM

Setting up and Installing Hadoop Framework using Windows.

DESCRIPTION

Installing Hadoop on Windows can be a bit challenging, as Hadoop is primarily designed to run on Unix-based systems. However, we can use the Hadoop distribution for Windows provided by the Apache Hadoop project or use tools like the Hadoop on Windows distribution by Microsoft.

PREREQUISTIES

1. **Java Installation:**
   https://www.java.com/en/download/windows_offline.jsp
   https://www.oracle.com/java/technologies/downloads/#java8-windows
2. **Download Hadoop**
   https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-3.2.4/hadoop-3.2.4.tar.gz
3. **Download 7-Zip**

   It is a "tar.gz" file which cannot be unzipped by WinRAR we need to install 7-Zip latest version

   https://www.7-zip.org/download.html
4. **Download Hadoop binaries**

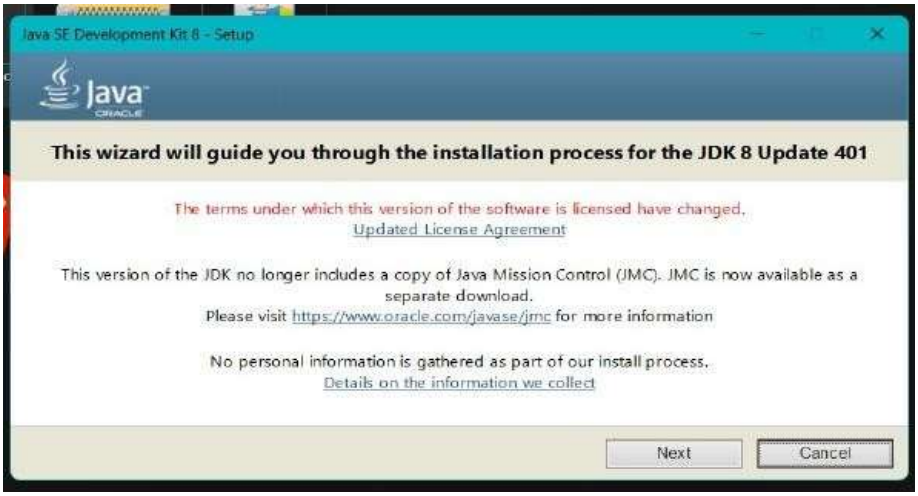
   https://drive.google.com/drive/folders/1TKKhHNClDYPNmn-kCSw-wiZ_sjN32P-L?usp=sharing
5. **Notepad++**

   https://notepad-plus-plus.org/downloads/v8.6.2/

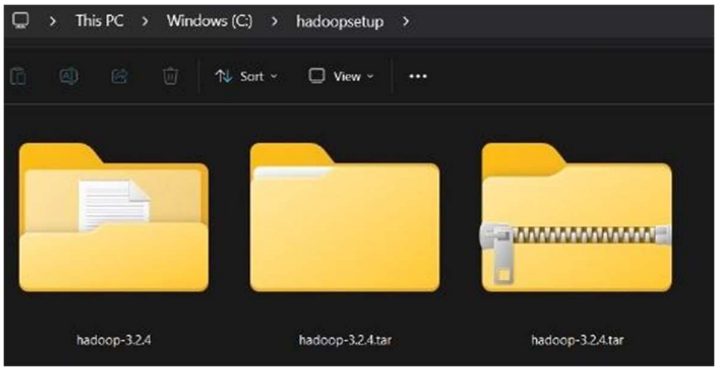
Steps for the Installation:

1. **Install JRE**

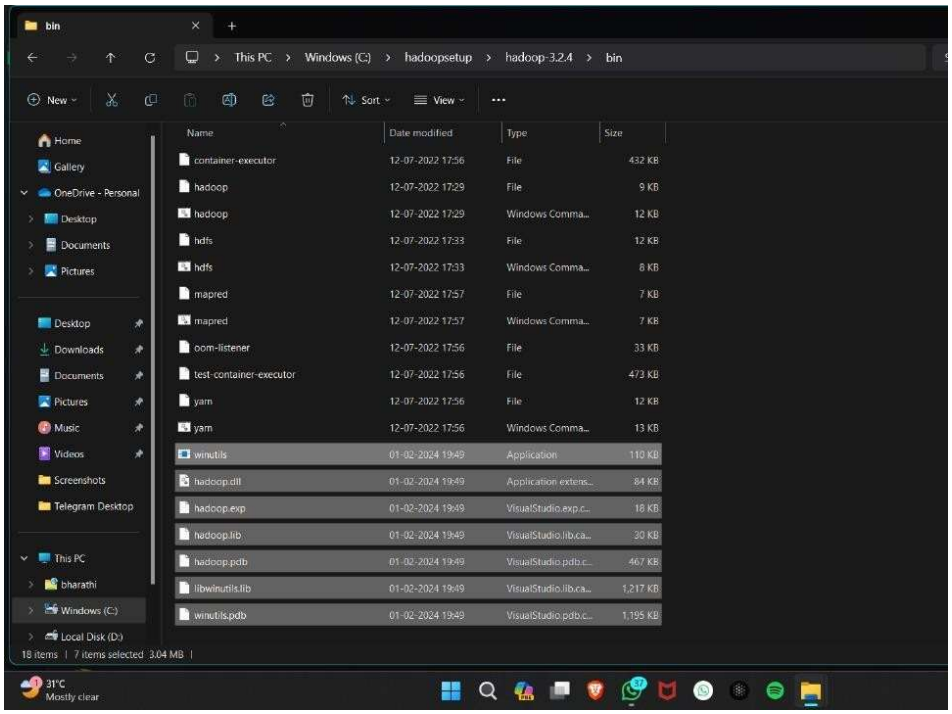
2. **Install JDK**



3. **Create a folder with name "hadoopsetup"** in **C** Disk and place the Hadoop downloaded file in it then with the help of 7-Zip Unzip it Twice.



4. **Download Hadoop libraries** from drive link and place it in bin of Hadoop File

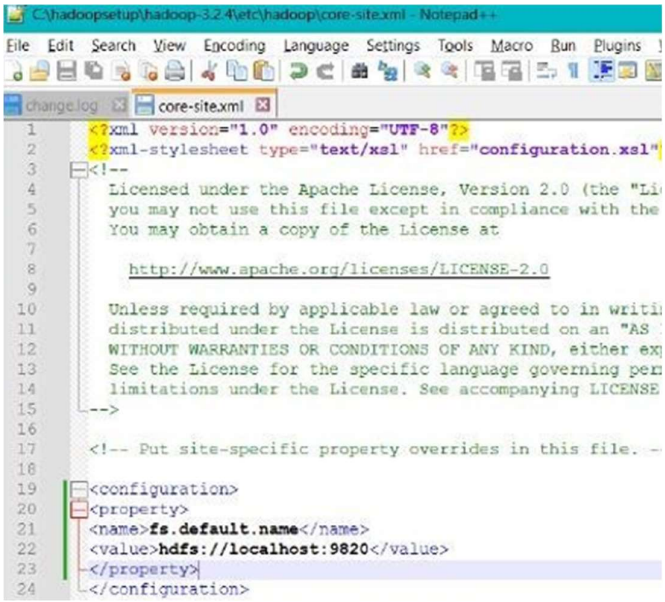


5. **Configuration:**

Navigate to the etc\hadoop directory within the Hadoop installation directory. Edit the **required sub folders** in "etc" folder of hadoop as follows

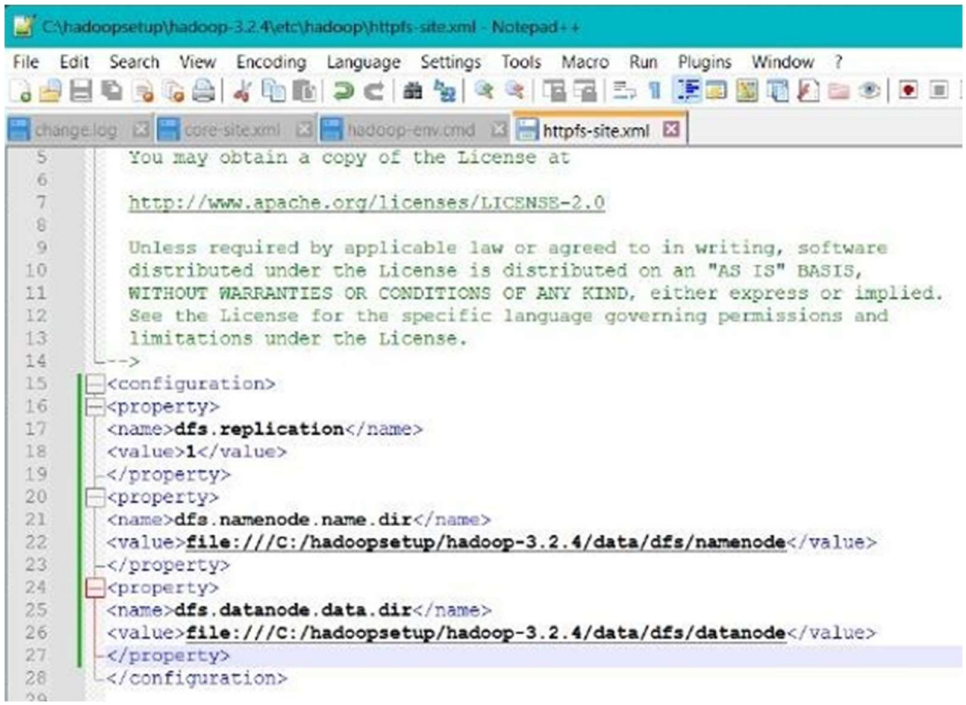
core-site.xml:

```
<configuration>
<property>
<name>fs.defaultFS</name>
        <value>hdfs://localhost:9000</value>
<property>
</configuration>
```

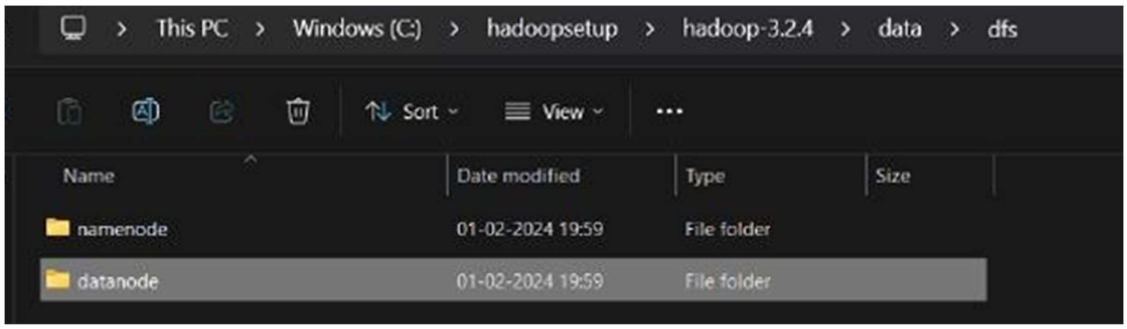


hdfs-site.xml

```
<property>
<name>dfs.replication</name>
<value>1</value>
 </property>
 <property>
<name>dfs.namenode.name.dir</name>
  <value>file:///C:/hadoopsetup/hadoop-3.2.4/data/dfs/namenode</value>
</property>
 <property>
<name>dfs.datanode.data.dir</name>
  <value>file:///C:/hadoopsetup/hadoop-3.2.4/data/dfs/datanode</value>
 </property>
```

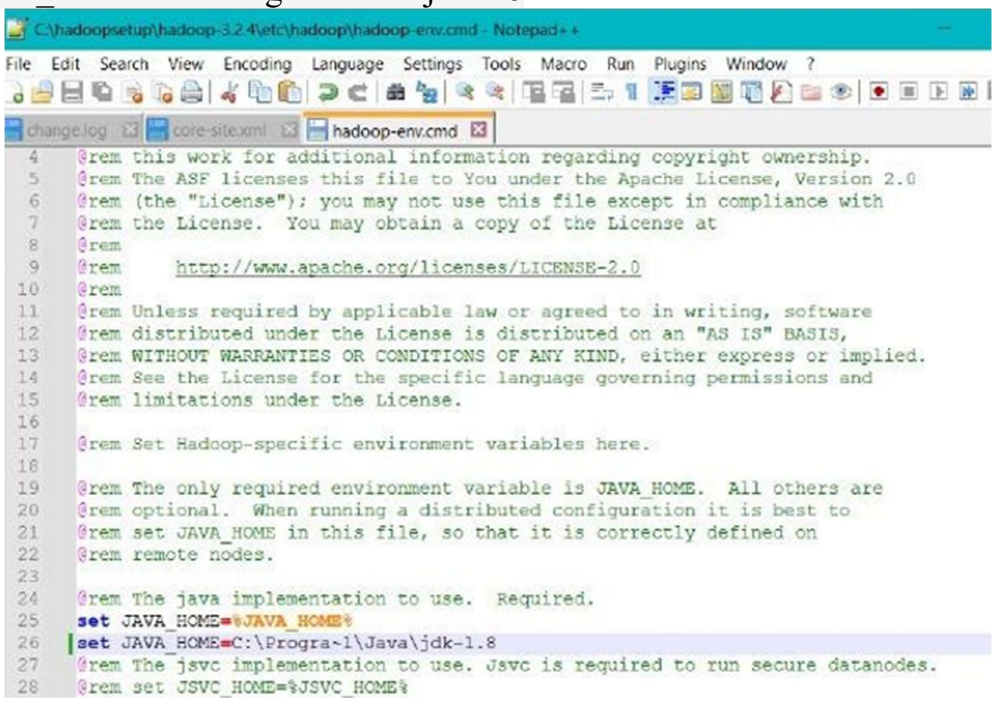


Create a folder "**data**" in Hadoop with a subfolder "**dfs**". In the dfs folder create two other folders with names "**datanode**" & "**namenode**".



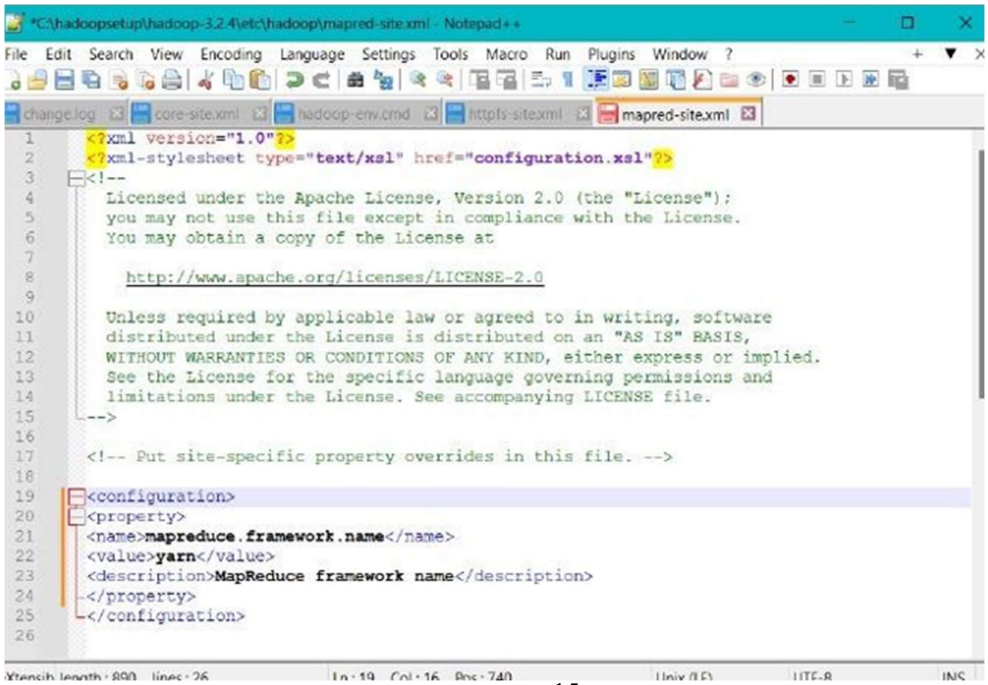
Hadoop-env.xml

set JAVA_HOME=C:\Progra~1\Java\jdk-1.8



Mapred-site.xml

<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
<description>MapReduce framework name</description>
</property>



Yarn-site.xml

```
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
<description>Yarn Node Manager Aux Service</description>
</property>
```

6. Environment Variables:

Add new entries in your system variable:

1. **Variable name: HADOOP_HOME**
   variable value: C:\hadoopsetup\hadoop-3.2.4
2. **Variable name: JAVA_HOME**
   variable value: C:\Program Files\Java\jdk-1.8

Add the following entries to your system's PATH variable:

%HADOOP_HOME%\bin
%HADOOP_HOME%\sbin
%JAVA_HOME%\bin

7. Start Hadoop Services:

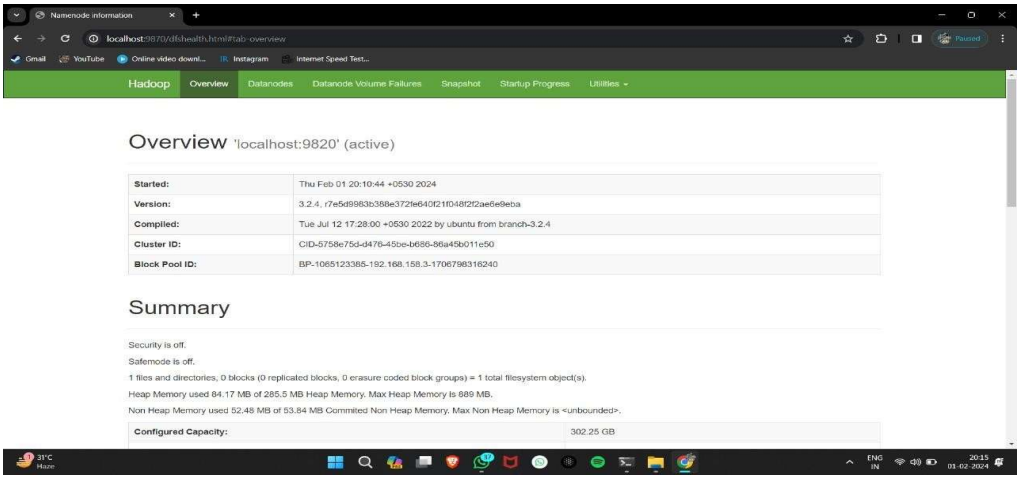
Open a command prompt as an administrator.
Navigate to the Hadoop bin directory and run the following commands:

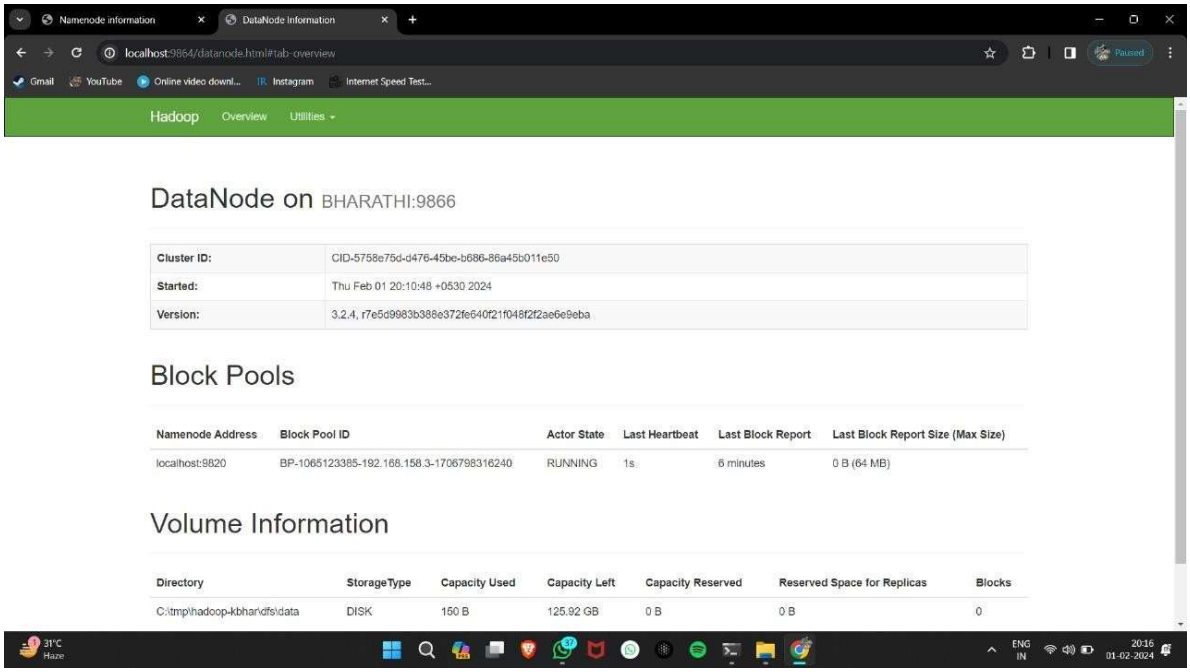
- Check the Versions of Java and Hadoop.
- Open the Powershell Prompt
  "hdfs namenode -format" execute the following command.

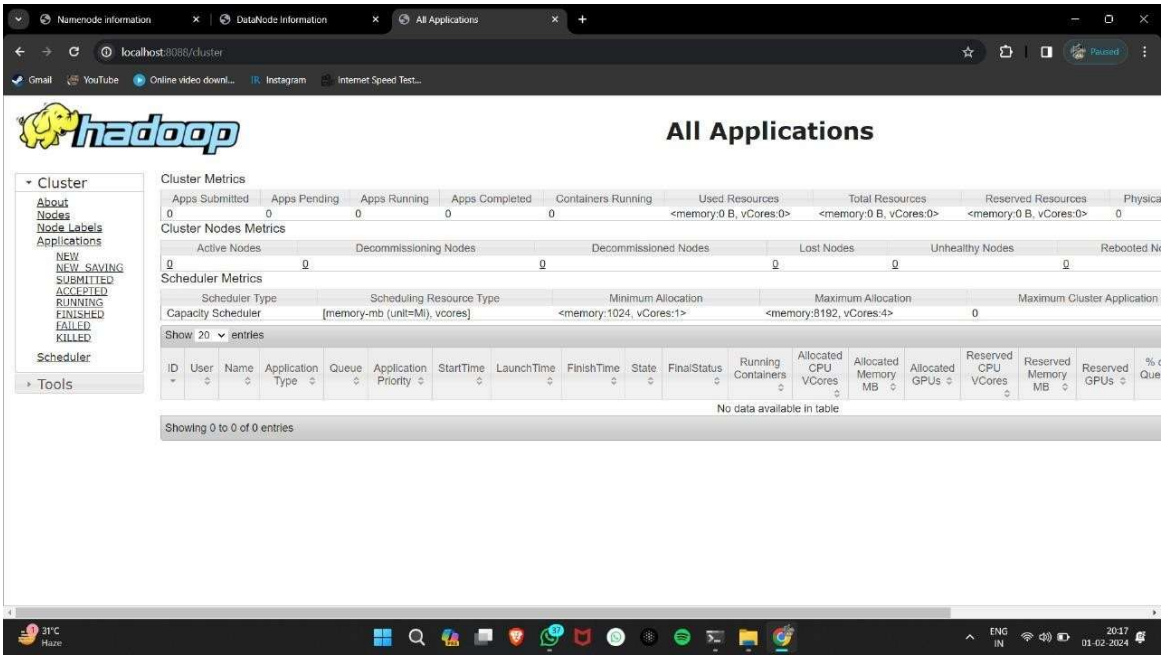
- Start the Hadoop in the Powershell prompt with following Commands
  ".\start-dfs.cmd"
  "./start-yarn.cmd"

- Open a browser and type the following host links to see the status of Hadoop Framework

http://localhost:9870/dfshealth.html
http://localhost:9864/datanode.html
http://localhost:8088/cluster


Namenode information
localhost:9870/dfshealth.html#tab-overview
Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Overview 'localhost:9820' (active)
Started: Thu Feb 01 20:10:44 +0530 2024
Version: 3.2.4, r7e5d9983b388e372fe640f21f048f2f2ae6e9eba
Compiled: Tue Jul 12 17:28:00 +0530 2022 by ubuntu from branch-3.2.4
Cluster ID: CID-5758e75d-d476-45be-b686-86a45b011e50
Block Pool ID: BP-1065123385-192.168.158.3-1706798316240
Summary
Security is off.
Safemode is off.
1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).
Heap Memory used 84.17 MB of 285.5 MB Heap Memory. Max Heap Memory is 889 MB.
Non Heap Memory used 52.48 MB of 53.84 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.
Configured Capacity: 302.25 GB


Namenode information
DataNode Information
localhost:9864/datanode.html#tab-overview
Hadoop
Overview
Utilities
DataNode on BHARATHI:9866
Cluster ID: CID-5758e75d-d476-45be-b686-86a45b011e50
Started: Thu Feb 01 20:10:48 +0530 2024
Version: 3.2.4, r7e5d9983b388e372fe640f21f048f2f2ae6e9eba
Block Pools
Namenode Address | Block Pool ID | Actor State | Last Heartbeat | Last Block Report | Last Block Report Size (Max Size)
localhost:9820 | BP-1065123385-192.168.158.3-1706798316240 | RUNNING | 1s | 6 minutes | 0 B (64 MB)
Volume Information
Directory | StorageType | Capacity Used | Capacity Left | Capacity Reserved | Reserved Space for Replicas | Blocks
C:\tmp\hadoop-kbhar\dfs\data | DISK | 150 B | 125.92 GB | 0 B | 0 B | 0


Namenode information
DataNode Information
All Applications
localhost:8088/cluster
All Applications
Cluster
About
Nodes
Node Labels
Applications
NEW
NEW_SAVING
SUBMITTED
ACCEPTED
RUNNING
FINISHED
FAILED
KILLED
Scheduler
Tools
Cluster Metrics
Apps Submitted | Apps Pending | Apps Running | Apps Completed | Containers Running | Used Resources | Total Resources | Reserved Resources
0 | 0 | 0 | 0 | 0 | <memory:0 B, vCores:0> | <memory:0 B, vCores:0> | <memory:0 B, vCores:0>
Cluster Nodes Metrics
Active Nodes | Decommissioning Nodes | Decommissioned Nodes | Lost Nodes | Unhealthy Nodes
0 | 0 | 0 | 0 | 0
Scheduler Metrics
Scheduler Type | Scheduling Resource Type | Minimum Allocation | Maximum Allocation | Maximum Cluster Application
Capacity Scheduler | [memory-mb (unit=Mi), vcores] | <memory:1024, vCores:1> | <memory:8192, vCores:4> | 0
Show 20 entries
ID | User | Name | Application Type | Queue | Application Priority | StartTime | LaunchTime | FinishTime | State | FinalStatus | Running Containers | Allocated CPU VCores | Allocated Memory MB | Allocated GPUs | Reserved CPU VCores | Reserved Memory MB | Reserved GPUs
No data available in table
Showing 0 to 0 of 0 entries

VIVA VOICE

1) What is the current version of Hadoop?

2) Is Hadoop an open-source framework?

3) What is namenode and datanode?

4) Name the three modes in which Hadoop can run?

5) Give the different Hadoop configuration file.

RESULT

Setting Up the Hadoop Framework is Successfully Installed

Ex. No:	2	FILE MANAGEMENT TASK
Date:		

AIM

To implement simple commands and file commands in Hadoop File System (HDFS).

ALGORITHM:

Step 1: Start the program
Step 2: Create a directory and sub directory
Step 3: Create and read file using put and get command
Step 4: Delete the file and directory
Step 5: Stop

PROGRAM:

Hadoop Version:

hadoop -version

SAMPLE OUTPUT

Hadoop 0.20.2-cdh3u2
Subversionsfile:///tmp/topdir/BUILD/hadoop-0.20.2- cduh3u2 –r 95a824e4oo5b2a94fe1c11f1ef9db4c672ba43cb

Creating directory:

Hadoop fs -mkdir /dir_name

Creating sub - directory:

Hadoop fs -mkdir /dir_name/sub_dir_name

Creating a file in hadoop:

Create a text file with some content and place in C drive

Hadoop fs -put C:\file_name / Hadoop fs -get /file_name

Reading the content in the text file:

Hadoop fs -head /file_name (or)

Hadoop fs -cat /file_name

Listing root directory

hadoop fs –ls /dir_name

File System Utilization:

hadoop fs –df /dir_name

SAMPLE OUTPUT

Filesystem Size Used AvailUse%
/dir_name 18611908608 3482800128 7934210048 18%

Count of files and directories:

hadoop fs –count /dir_name

SAMPLE OUTPUT

152 306 3086411421hdfs://dir_name

Zero byte file:

Hadoop fs -touchz /dir_name/file_name

Deleting file:

Hadoop fs -rm /file_name

Same output:

/file_name deleted.

Deleting directory:

Hadoop fs -rm -r /dir_name

Same output:

/dir_name deleted.

Viva Questions:

1. **Can files in HDFS be modified?**

2. **What is the block size of HDFS file system?**

3. **Where is the Hadoop file located?**

4. **What is the default size in HDFS?**

5. **How the Hadoop files are stored?**

Result:

Thus, the implementation of simple commands and file commands in Hadoop File System (HDFS) is executed successfully.

Ex. No:	3	WORD COUNT USING MAP REDUCE
Date:		

AIM

To implement the Word Count program using Map Reduce with Hadoop streaming utility.

DESCRIPTION:

MapReduce is the heart of Hadoop. It is this programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster. The MapReduce concept is fairly simple to understand for those who are familiar with clustered scale- out data processing solutions. The term MapReduce actually refers to two separate and distinct tasks that Hadoop programs perform. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.

ALGORITHM MAPREDUCE:

Word Count is a simple program which counts the number of occurrences of each word in given text input data set.

- **Mapper**- The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.

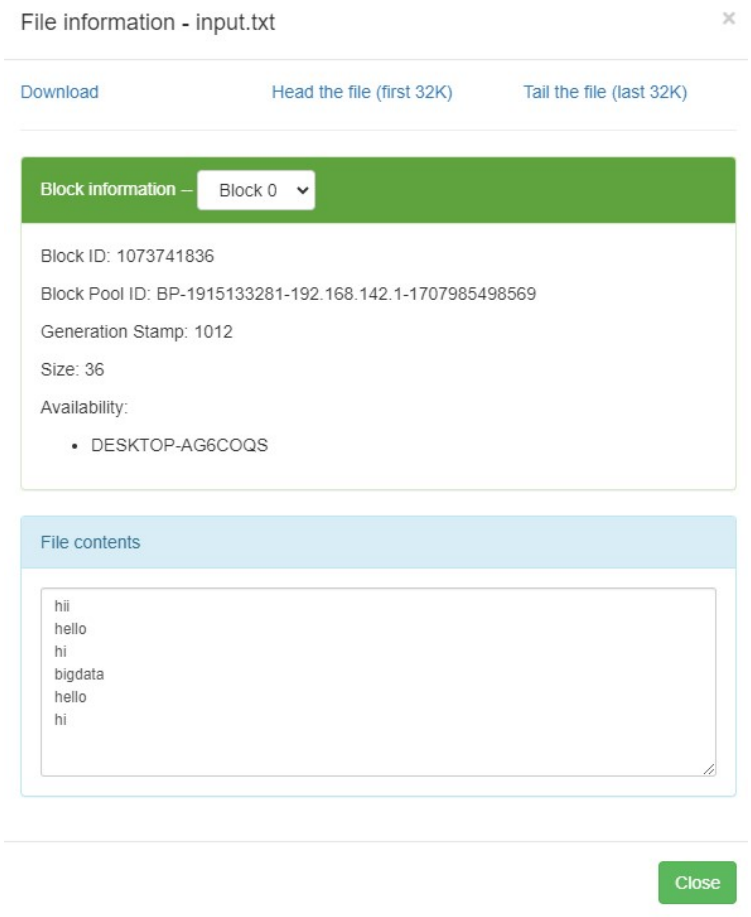
- **Reduce** − This stage is the combination of the **Shuffle** stage and the **Reduce** stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.

Program:

Output:

```
C:\Users\Admin\Documents>icacls mapper.py /grant Everyone:F
processed file: mapper.py
Successfully processed 1 files; Failed processing 0 files

C:\Users\Admin\Documents>icacls reducer.py /grant Everyone:F
processed file: reducer.py
Successfully processed 1 files; Failed processing 0 files
```

```
C:\Users\Admin\Documents>type input.txt | python mapper.py | sort | python reducer.py
bigdata 1
hello   2
hi      2
hii     1
```



Viva Questions:

1. **What is mapper?**

2. **What is the reducer?**

3. **What is the default size of the mapper?**

4. **In windows environment what is the use of icacls command?**

5. **What is the use of "type input.txt | python mapper.py | sort | python reducer.py" command?**

Result:

Thus, the program for Word Count using Map Reduce was implemented Successfully.

Ex. No:	4	WEATHER DATA USING MAP REDUCE
Date:		

AIM

To implement the Weather data program using Map Reduce with Hadoop streaming utility.

DESCRIPTION:

MapReduce is the heart of Hadoop. It is this programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster. The MapReduce concept is fairly simple to understand for those who are familiar with clustered scale- out data processing solutions. The term MapReduce actually refers to two separate and distinct tasks that Hadoop programs perform. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.

ALOGRITHM:

Step 1: Import the datset from ftp://ftp.ncdc.noaa.gov/pub/data/uscrn/products/daily01/.

Step 2: Write mapper and reducer functions for finding the average of the weather.

Step 3: Now with Hadoop streaming jar file load the input, mapper.py, reducer.py and execute the code.

Program:

MAPPER.PY: -

```
import sys
# input comes from STDIN (standard input)
#The mapper will get daily max temperature and group it by month. so output will be(month, max)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split()
    #See the README hosted on the weather website  which help us understand how each position represents a column
    month = line[10:12]
    daily_max = line[38:45]
    daily_max = daily_max.strip()
    # increase counters
    for word in words:
        # write the results to STDOUT (standard output);
        # what we output here will be go through the shuffle proess and then
        # be the input for the Reduce step, i.e. the input for reducer.PY
        # tab-delimited; month and daily max temperature as output
        print '%s\t%s' % (month ,daily_max)
```

REDUCER.PY:-

```
from operator import itemgetter
import sys
#reducer will get the input from stdid which will be a collection of key, value(Key=month , value= daily max temperature)
#reducer logic: will get all the daily max temperature for a month and find max temperature for the month
#shuffle will ensure that key are sorted(month)
current_month = None
current_max = 0
month = None

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # parse the input we got from mapper.py
```

```
    month, daily_max = line.split('\t', 1)

    # convert daily_max (currently a string) to float
    try:
        daily_max = float(daily_max)
    except ValueError:
        # daily_max was not a number, so silently
        # ignore/discard this line
        continue

    # this IF-switch only works because Hadoop shuffle process sorts map output
    # by key (here: month) before it is passed to the reducer
    if current_month == month:
        if daily_max > current_max:
            current_max = daily_max
    else:
        if current_month:
            # write result to STDOUT
            print '%s\t%s' % (current_month, current_max)
        current_max = daily_max
        current_month = month

# output of the last month
if current_month == month:
    print '%s\t%s' % (current_month, current_max)
```

Output:

01 -14.1
02 -11.8
03 -16.5
04 -7.5
05 1.3
06 17.7
07 19.3
08 18.1
09 6.9
10 4.6
11 -2.0
12 -11.8

VIVA VOICE

1. What is Hadoop streaming?

2. What are the different weather parameters?

3. Why weather data analysis is done using big data?

4. Advantages of MapReduce in weather analysis.

5. Current industries using MapReduce concepts in weather analysis.

Result:

Thus, using Map reduce the weather data is successfully observed &executed.

Ex. No:	5	MATRIX MULTIPLICATION USING MAP REDUCE
Date:		

AIM

To implement the Matrix multiplication using Map Reduce with Hadoop streaming utility.

DESCRIPTION:

MapReduce is the heart of Hadoop. It is this programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster. The MapReduce concept is fairly simple to understand for those who are familiar with clustered scale- out data processing solutions. The term MapReduce actually refers to two separate and distinct tasks that Hadoop programs perform. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.

ALGORITHM MAPREDUCE:

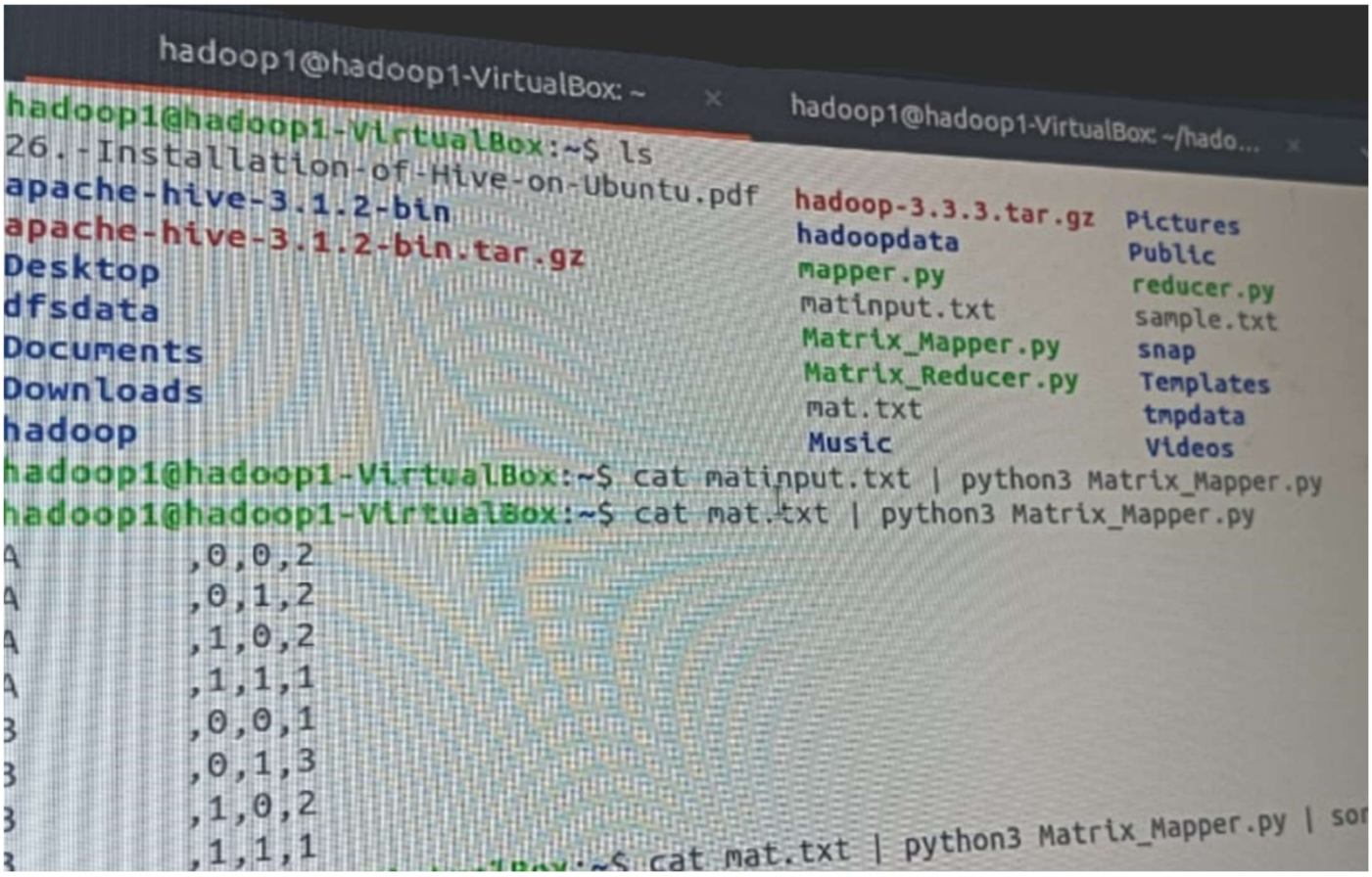
Word Count is a simple program which counts the number of occurrences of each word in given text input data set.

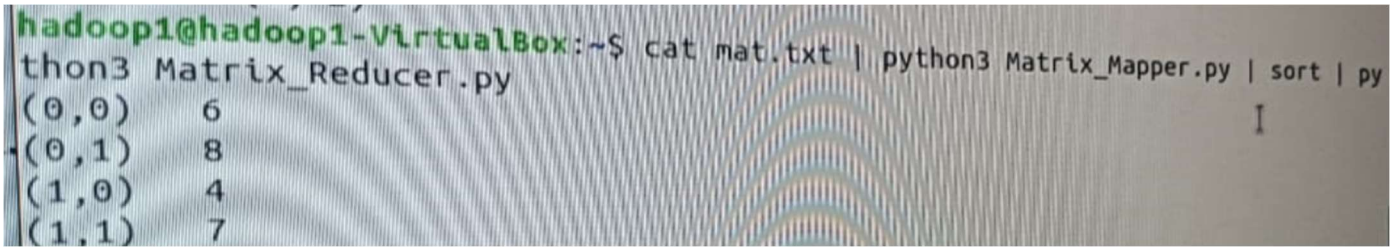
- **Mapper**- The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.

- **Reduce** − This stage is the combination of the **Shuffle** stage and the **Reduce** stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.

Program:

Output:


hadoop1@hadoop1-VirtualBox: ~
hadoop1@hadoop1-VirtualBox: ~/hado...
hadoop1@hadoop1-VirtualBox:~$ ls
26.-Installation-of-Hive-on-Ubuntu.pdf
apache-hive-3.1.2-bin
apache-hive-3.1.2-bin.tar.gz
Desktop
dfsdata
Documents
Downloads
hadoop
hadoop-3.3.3.tar.gz
hadoopdata
mapper.py
matinput.txt
Matrix_Mapper.py
Matrix_Reducer.py
mat.txt
Music
Pictures
Public
reducer.py
sample.txt
snap
Templates
tmpdata
Videos
hadoop1@hadoop1-VirtualBox:~$ cat matinput.txt | python3 Matrix_Mapper.py
hadoop1@hadoop1-VirtualBox:~$ cat mat.txt | python3 Matrix_Mapper.py
A ,0,0,2
A ,0,1,2
A ,1,0,2
A ,1,1,1
B ,0,0,1
B ,0,1,3
B ,1,0,2
B ,1,1,1
cat mat.txt | python3 Matrix_Mapper.py | sor


hadoop1@hadoop1-VirtualBox:~$ cat mat.txt | python3 Matrix_Mapper.py | sort | py
thon3 Matrix_Reducer.py
(0,0) 6
(0,1) 8
(1,0) 4
(1,1) 7

VIVA VOICE

1. Why matrix multiplication is considered in MapReduce?

2. Advantages using MapReduce in matrix multiplication.

3. In how many dimensions of matrix multiplication can be done in MapReduce?

4. What is the role of the Map function in matrix multiplication

5. What are the key steps involved in implementing matrix multiplication using MapReduce?

Result:

Thus, using Map reduce the matrix multiplication is successfully observed &executed.

Ex. No:	6	INSTALLATION OF PIG
Date:		

AIM

To install the PIG using Windows.

Prerequisites:

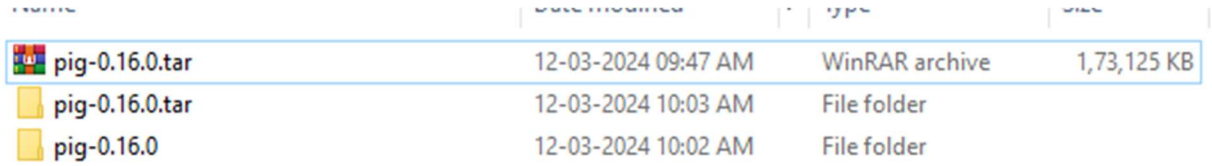
Pig Installation:

https://downloads.apache.org/pig/pig-0.16.0/

Steps for the installation:

Step 1:

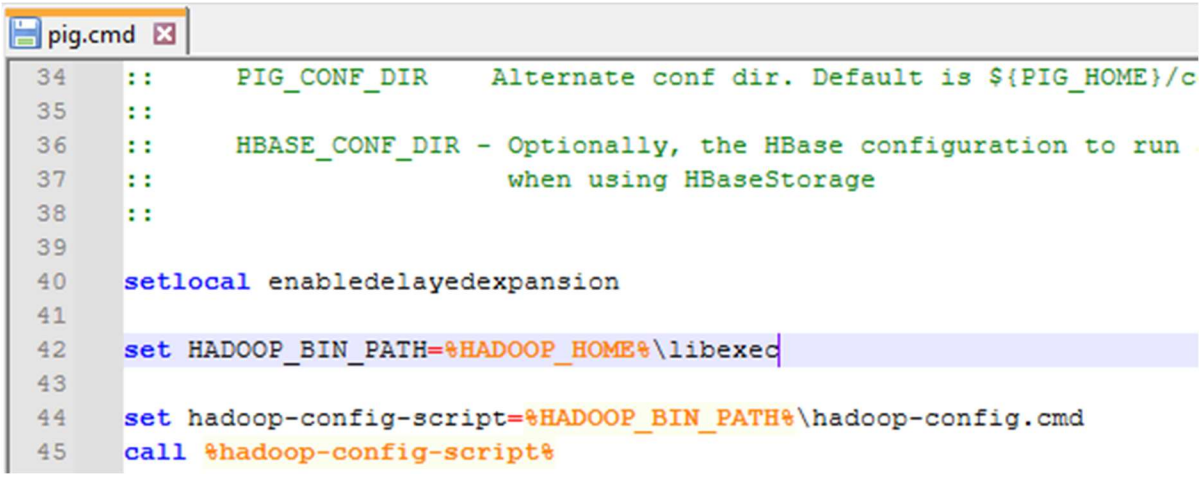
Place the downloaded pig tar file in **C** Disk and with the help of 7-Zip Unzip it Twice.



Step 2:

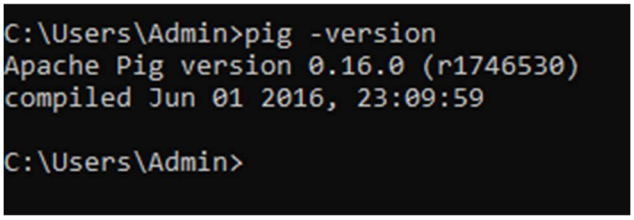
Configure the pig file

"set HADOOP_BIN_PATH=%HADOOP_HOME%\libexec"



Output:

Pig -version



VIVA VOICE

1. What is Apache Pig?

2. What are the steps involved in setting up environment variables for Apache Pig?

3. How do you start and stop Apache Pig services after installation?

4. How do you verify the installation of Apache Pig?

5. Outline the configuration changes required in pig.properties file?

Result:

Thus, using installation of PIG is successfully observed &executed.

Ex. No:	7	INSTALLATION OF HIVE
Date:		

AIM

To install the HIVE using Ubuntu.

Description:

Installing Apache Hive involves several steps, and it's important to follow the recommended procedures for your specific environment. Below are general steps for installing Hive:

Prerequisites:

Make sure you have Java installed on your system. Hive is a Java-based tool, and it requires Java to be set up. Install Hadoop: Hive typically runs on top of Hadoop, so you need to have Hadoop installed and configured before installing Hive.

Installation:

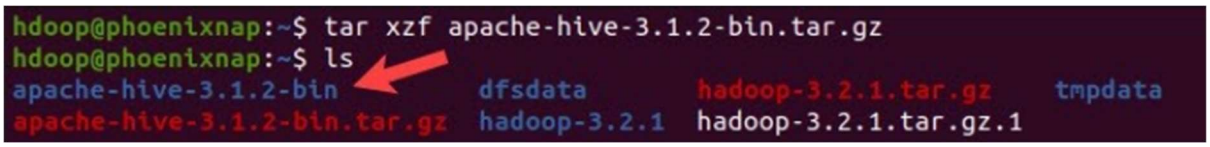
Step 1: Download and Untar Hive

```
hdoop@phoenixnap:~$ wget https://downloads.apache.org/hive/hive-3.1.2/apache-hi
ve-3.1.2-bin.tar.gz
--2020-06-01 08:11:30--  https://downloads.apache.org/hive/hive-3.1.2/apache-hi
ve-3.1.2-bin.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 88.99.95.219, 2a01:4f8
:10a:201a::2
Connecting to downloads.apache.org (downloads.apache.org)|88.99.95.219|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 278813748 (266M) [application/x-gzip]
Saving to: 'apache-hive-3.1.2-bin.tar.gz'

apache-hive-3.1.2-b 100%[==================>] 265.90M  10.9MB/s    in 25s

2020-06-01 08:11:55 (10.7 MB/s) - 'apache-hive-3.1.2-bin.tar.gz' saved [2788137
48/278813748]
```

Access your Ubuntu command line and download the compressed Hive files using and the wget command followed by the download path:

wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz

Once the download process is complete, untar the compressed Hive package: tar xzf apache-hive-3.1.2-bin.tar.gz The Hive binary files are now located in the apache-hive-3.1.2-bin directory

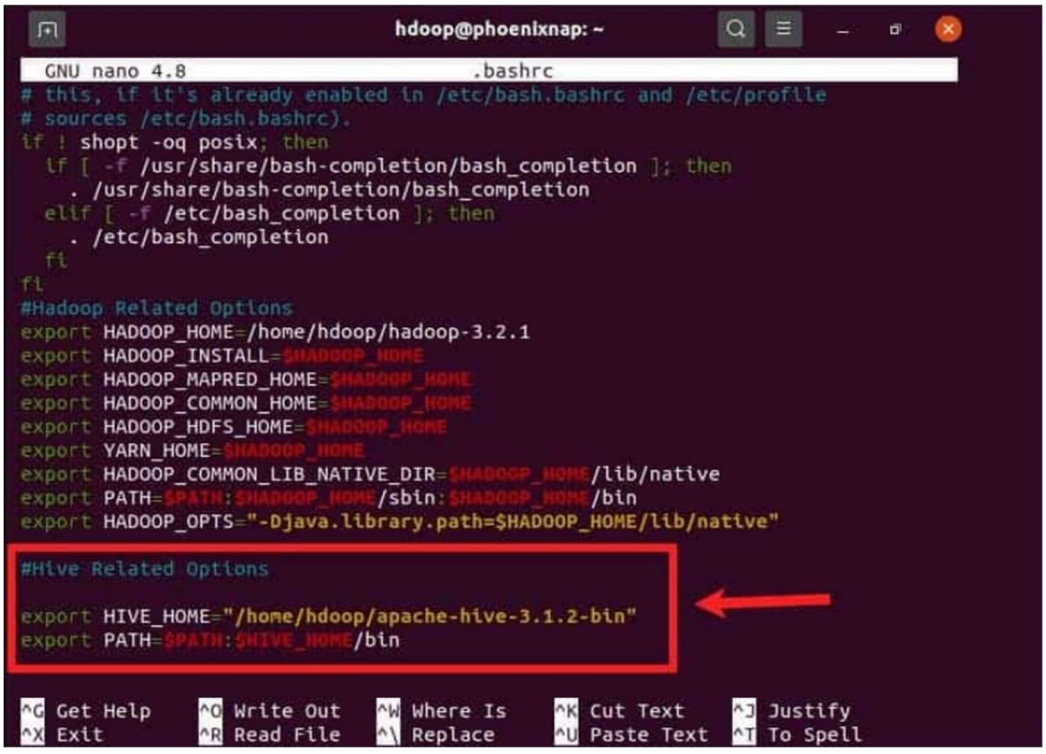


Step 2: Configure Hive Environment Variables (bashrc)

sudo nano .bashrc

Append the following Hive environment variables to the .bashrc file: export

HIVE_HOME= "home/hdoop/apache-hive-3.1.2-bin" export PATH=$PATH:$HIVE_HOME/bin



Step 3: Edit hive-config.sh file

Add the HADOOP_HOME variable and the full path to your Hadoop directory: export **HADOOP_HOME=/home/hdoop/hadoop-3.2.1**

```
# Allow alternate conf dir location.
HIVE_CONF_DIR="${HIVE_CONF_DIR:-$HIVE_HOME/conf}"

export HIVE_CONF_DIR="/home/hdoop/apache-hive-3.1.2-bin/conf"
export HADOOP_HOME=/home/hdoop/hadoop-3.2.1
```

Step 4: Create Hive Directories in HDFS

Create tmp Directory:

hdfs dfs -mkdir /tmp

hdfs dfs -chmod g+w /tmp

hdfs dfs -ls /

Create warehouse Directory:

hdfs dfs -mkdir -p /user/hive/warehouse

hdfs dfs -chmod g+w /user/hive/warehouse

hdfs dfs -ls /user/hive

Step 5: Initiate Derby Database

$HIVE_HOME/bin/schematool -dbType derby -initSchema

```
Initialization script completed
schemaTool completed
hdoop@phoenixnap:~/apache-hive-3.1.2-bin/bin$
```

VIVA VOICE:

1. What is Apache Hive?

2. What are the steps involved in setting up environment variables for Apache Hive?

3. How do you start and stop Apache Hive services after installation?

4. How do you initialize the Hive metastore database schema?

5. Outline the configuration changes required in hive-site.xml file?

RESULT

Thus, Installation of HIVE is successful.

Ex. No:	8a	PIG LATIN SCRIPTS USING WORD COUNT
Date:		

AIM

To implement the WORD COUNT using PIG Latin Script.

DESCRIPTION:

` Pig Latin is a scripting language used for processing and analyzing large datasets in Apache Hadoop. It has a simple syntax and is often used for writing MapReduce programs. If you want to write a Pig Latin script to find the word count in a dataset, you can follow these steps:

ALGORITHM:

Step 1: LOAD the input text
Step 2: Generate the token for each word
Step 3: Group the data
Step 4: Count the grouped data
Step 5: Print the output

Program:

```
hadoop1@hadoop1-VirtualBox:~$pig grunt>
input = LOAD '/word_count/input.txt' AS(line:Chararray);
grunt > words = FOREACH input GENERATE FLATTEN(TOKENIZE(line,' ')) AS word;
grunt > grouped = GROUP words BY word;
grunt > wordcount = FOREACH Grouped GENERATE group, COUNT(words);;
grunt > DUMP wordcount;
```

OUTPUT

VIVA VOCE:

1. What is the purpose of finding word count using Pig Latin Scripts?

2. Explain the structure of a pig latin script?

3. What are the key Pig Latin commands used for processing data?

4. How do you group data in Pig Latin and why is it important for word count?

5. What is Pig Latin?

RESULT

Thus the word Count was executed using the Pig Latin Script.

Ex. No:	8b	MAXIMUM TEMPERATURE USING PIG LATIN
Date:		

AIM

To Implement the Pig Latin Scripts for finding the maximum temperature each and every year.

DESCRIPTION:

Pig Latin is a scripting language used in Apache Hadoop for processing and Analyzing large datasets. If you want to find the maximum temperature for each year using Pig Latin, you'll typically have a dataset with temperature records.

ALGORITHM:

Step 1: LOAD the input text
Step 2: Generate the token for each word
Step 3: Group the data
Step 4: Count the grouped data
Step 5: Print the output

Program:

```
hadoop1@hadoop1-VirtualBox:~$pig
grunt> A = LOAD “input.txt” USING PigStorage() AS (Year:int,Temp:int);
 grunt >B = GROUP A ALL;
grunt >C = FOREACH B GENERATE MAX(A.Temp);
grunt >DUMP C;
```

OUTPUT

VIVA VOCE:

1. What is the purpose of finding the maximum temperature each year using Pig Latin scripts?

2. Can you outline the steps involved in writing a Pig Latin script for this task?

3. How do you load the data containing temperature records into Pig?

4. What Pig Latin function or operator would you use to extract the year from each temperature record?

5. How do you group the temperature data by year in Pig Latin?

Result:

Thus, the installation of PIG Latin Scripts for finding the maximum temperature each and every year is executed successfully

CONTENT BEYOND SYLLABUS

Ex. No:	9	HADOOP – MRJOB PYTHON LIBRARY
Date:		

AIM

Count the number of occurrence of words from a text file using python mrjob.

DESCRIPTION:

mrjob is the famous python library for MapReduce developed by YELP. The library helps developers to write MapReduce code using a Python Programming language. Developers cantest the MapReduce Python code written with mrjob locally on their system or on the cloud using Amazon EMR(Elastic Map Reduce).

Amazon EMR is a cloud-based web service provided by Amazon Web Services for Big Data purposes. mrjob is currently an active Framework for Map Reduce programming or Hadoop Streaming jobs and has good document support for Hadoop with python than any Other library or framework currently available. With mrjob, we can write code for Mapper and Reducer in a single class.

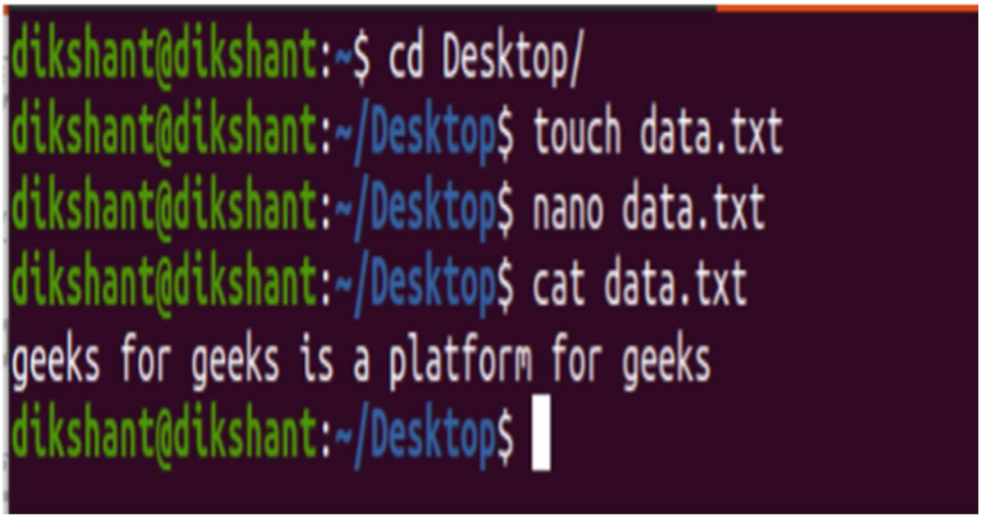
Install mrjob in your system:

pip install mrjob # for python3 use pip3

```
dikshant@dikshant:~$ pip3 install mrjob
Collecting mrjob
  Downloading mrjob-0.7.4-py2.py3-none-any.whl (439 kB)
     |                                | 439 kB 154 kB/s
Requirement already satisfied: PyYAML>=3.10 in /usr/lib/python3/dist-packages (f
rom mrjob) (5.3.1)
Installing collected packages: mrjob
```

PROGRAM:

Step 1: Create a text file with the name data.txt and add some content to it.
touch data.txt //used to create file in linux
nano data.txt // nano is a command line editor in linux
cat data.txt // used to see the inner content of file



Step 2: Create a file with the name CountWord.py at the location where your data.txt file is available.

touch CountWord.py // create the python file with name CountWord

```
from mrjob.job import MRJob
class Count(MRJob):

    def mapper(self, _, line):
        for word in line.split():
            yield(word, 1)

    def reducer(self, word, counts):
        yield(word, sum(counts))

if __name__ == &#39;__main__&#39;:

    Count.run()
```

Below is the image Of My CountWord.py file.

```
dikshant@dikshant:~/Desktop$ cat CountWord.py
from mrjob.job import MRJob
class Count(MRJob):
        def mapper(self, _, line):
                for word in line.split():
                        yield(word, 1)
        def reducer(self, word, counts):
                yield(word, sum(counts))
if __name__ == '__main__':
        Count.run()
dikshant@dikshant:~/Desktop$
```

Step 3: Run the python File in your local machine as shown below to test it is working fine or not(Note: I am using python3).

python CountWord.py data.txt

```
dikshant@dikshant:~/Desktop$ python3 CountWord.py data.txt
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/CountWord.dikshant.20201112.104140.528112
Running step 1 of 1...
job output is in /tmp/CountWord.dikshant.20201112.104140.528112/output
Streaming final output from /tmp/CountWord.dikshant.20201112.104140.528112/outpu
t...
"geeks" 3
"a"     1
"for"   2
"is"    1
"platform"      1
Removing temp directory /tmp/CountWord.dikshant.20201112.104140.528112...
dikshant@dikshant:~/Desktop$
```

Choice Description

-r inline mrjob runs in a single python program(Default Option)
-r local mrjob runs locally in some subprocess along with some Hadoop features
-r hadoop mrjob runs on Hadoop
-r emr mrjob runs on Amazon Elastic MapReduce

OUTPUT:

Syntax:
python <mrjob-pythonfile> -r hadoop <hdfs-path>

Command:
Send your data.txt to HDFS with the help of the below command (NOTE: I have already sent data.txt to the Count content folder on HDFS).

hdfs dfs -put /home/dikshant/Desktop/data.txt /

```
dikshant@dikshant:~$ hdfs dfs -put /home/dikshant/Desktop/data.txt /
dikshant@dikshant:~$ hdfs dfs -ls /
Found 3 items
-rw-r--r--   1 dikshant supergroup         40 2020-11-12 16:51 /data.txt
drwxrwxr-x   - dikshant supergroup          0 2020-11-06 12:17 /tmp
drwxr-xr-x   - dikshant supergroup          0 2020-10-28 18:58 /user
dikshant@dikshant:~$
```

Run the below command to run mrjob on Hadoop.

python CountWord.py -r hadoop hdfs:///content/data.txt

```
job output is in hdfs:///user/dikshant/tmp/mrjob/CountWord.dikshant.20201112.113
907.267357/output
Streaming final output from hdfs:///user/dikshant/tmp/mrjob/CountWord.dikshant.2
0201112.113907.267357/output...
"a"     1
"for"   2
"geeks" 3
"is"    1
"platform"      1
Removing HDFS temp directory hdfs:///user/dikshant/tmp/mrjob/CountWord.dikshant.
20201112.113907.267357...
Removing temp directory /tmp/CountWord.dikshant.20201112.113907.267357...
dikshant@dikshant:~/Desktop$
```

Result:
Hadoop – mrjob Python Library For Map Reduce Program have successfully executed mrjob on the text file available on our HDFS.