

***U18ISI5202 – Big Data Technologies
Lab Record***

Name : Monika M

Roll no : 20BIS025

Date : 13.12.2022

Course : U18ISI5202 – Big Data technologies

INDEX

S. NO	NAME OF THE EXPERIMENTS
1	Perform setting up and Installing Hadoop in its three operatingmodes: Standalone, Pseudo distributed, fully distributed
2	Implement the following file management tasks in Hadoop: <ul style="list-style-type: none">• Adding files and directories• Retrieving files• Deleting files
3	Run a basic Word Count Map Reduce program to understand MapReduce Paradigm
4	Implement a Map Reduce Program to analyse time-temperaturestatistics and generate report with max/min temperature
5	Implement a Map Reduce Program Using multiplication
6	Perform Joining data with streaming in Hadoop using Map Reduce
7	Collect, aggregate and transport large amount of streaming data fromTwitter data using Apache Flume
8	Perform simple join using Mapper in Spark
9	Install and Run Hive then use Hive to create, alter, and dropdatabases, tables, views,functions, and indexes
10	Verify, Sparse and perform advance join of data using spark

LAB EXERCISE-1

Title of the exercise/experiment: Perform setting up and Installing Hadoop in its three operating modes: Standalone, Pseudo distributed, fully distributed

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

- Installation of Hadoop Framework, its components and study the Hadoop Ecosystem. Perform setting up and installing Hadoop in its three operating modes: Standalone, Pseudo distributed, fully distributed.

THEORY:

Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models.

Hadoop software can be installed in three modes of operation:

- Stand Alone Mode: Hadoop is a distributed software and is designed to run on a commodity of machines. However, we can install it on a single node in stand-alone mode. In this mode, Hadoop software runs as a single monolithic java process. This mode is extremely useful for debugging purpose. You can first test run your Map-Reduce application in this mode on small data, before actually executing it on cluster with big data.
- Pseudo Distributed Mode: In this mode also, Hadoop software is installed on a Single Node. Various daemons of Hadoop will run on the same machine as separate java processes. Hence all the daemons namely Name Node, DataNode, SecondaryNameNode, Job Tracker, Task Tracker run on single machine.
- Fully Distributed Mode: In Fully Distributed Mode, the daemons Name Node, Job Tracker, SecondaryNameNode (Optional and can be run on a separate node) run on the Master Node. The daemons DataNode and Task Tracker run on the Slave Node.

1. Installation done :

```
hadoop@ip-172-31-23-74:~$ sudo mv hadoop-3.3.1 /usr/local/hadoop
hadoop@ip-172-31-23-74:~$ sudo mkdir /usr/local/hadoop/logs
hadoop@ip-172-31-23-74:~$ sudo chown -R hadoop:hadoop /usr/local/hadoop
hadoop@ip-172-31-23-74:~$ sudo nano ~/.bashrc
hadoop@ip-172-31-23-74:~$ source ~/.bashrc
hadoop@ip-172-31-23-74:~$ which javac
/usr/bin/javac
hadoop@ip-172-31-23-74:~$ readlink -f /usr/bin/javac
/usr/lib/jvm/java-11-openjdk-amd64/bin/javac
hadoop@ip-172-31-23-74:~$ sudo nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh
hadoop@ip-172-31-23-74:~$ cd /usr/local/hadoop/lib
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo wget https://jcenter.bintray.com/javax/activation/javax.activation-api/1.2.0/jar
--2022-12-13 05:53:20 - https://jcenter.bintray.com/javax/activation/javax.activation-api/1.2.0/javax.activation-api-1.2.0.jar
Resolving jcenter.bintray.com (jcenter.bintray.com)... 34.95.74.180
Connecting to jcenter.bintray.com (jcenter.bintray.com)|34.95.74.180|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 56674 (55K) [application/java-archive]
Saving to: 'javax.activation-api-1.2.0.jar'

javax.activation-ap 100%[=====] 55.35K --.KB/s in 0.002s
2022-12-13 05:53:20 (21.7 MB/s) - 'javax.activation-api-1.2.0.jar' saved [56674/56674]

hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ hadoop version
Hadoop 3.3.1
Source code repository https://github.com/apache/hadoop.git -r a3b9c37a397ad4188041dd80621bdeefc46885f2
Compiled by ubuntu on 2021-06-15T05:13Z
Compiled with protoc 3.7.1
From source with checksum 88a4ddb2299aca054416d6b7f81ca55
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-3.3.1.jar
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo nano $HADOOP_HOME/etc/hadoop/core-site.xml
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo mkdir -p /home/hadoop/hdfs/{namenode,datanode}
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo chown -R hadoop:hadoop /home/hadoop/hdfs
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo nano $HADOOP_HOME/etc/hadoop/mapred-site.xml
hadoop@ip-172-31-23-74:/usr/local/hadoop/lib$ sudo nano $HADOOP_HOME/etc/hadoop/yarn-site.xml
```

2. start all resources :

```
hadoop@ip-172-31-23-74:~$ start-dfs.sh
Starting namenodes on [0.0.0.0]
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts.
Starting datanodes
Starting secondary namenodes [ip-172-31-23-74]
ip-172-31-23-74: Warning: Permanently added 'ip-172-31-23-74,172.31.23.74' (ECDSA) to the list of known hosts.
hadoop@ip-172-31-23-74:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@ip-172-31-23-74:~$ jps
4323 NameNode
5155 NodeManager
4486 DataNode
5000 ResourceManager
4763 SecondaryNameNode
5437 Jps
hadoop@ip-172-31-23-74:~$ █
```

3. Web Interface :

The screenshot shows the HDFS Health Overview page at localhost:9870/dfshealth.html#tab-overview. The top navigation bar includes tabs for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The Utilities tab is currently selected, displaying a dropdown menu with options: Browse the file system, Logs, Log Level, Metrics, Configuration, Process Thread Dump, and Network Topology.

Overview '0.0.0.0:9000' (active)

Started:	Tue Dec 13 05:58:36 +0000 2022
Version:	3.3.1, ra3b9c37a397ad4188041dd80621bdeefc46885f2
Compiled:	Tue Jun 15 05:13:00 +0000 2021 by ubuntu from (HEAD detached at release-3.3.1-RC3)
Cluster ID:	CID-8e378a60-55a9-436a-b62a-9e6a3ca2b7be
Block Pool ID:	BP-738289719-172.31.23.74-1670911094300

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 77.51 MB of 231 MB Heap Memory. Max Heap Memory is 1.92 GB.
Non Heap Memory used 50.85 MB of 54.56 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	19.31 GB
Configured Remote Capacity:	0 B
DFS Used:	24 KB (0%)

All Applications

Cluster Metrics		Used Resources	Total Reso			
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>
0	0	0	0	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>

Cluster Nodes Metrics		Decommissioning Nodes		Decommissioned Nodes		Lost Nodes	
Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Decommissioned Nodes	Decommissioned Nodes	Decommissioned Nodes	Decommissioned Nodes
1	0	0	0	0	0	0	0

Scheduler Metrics		Scheduling Resource Type		Minimum Allocation		Maximum Allocation									
Scheduler Type	Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	<memory:8192, vCores:4>	<memory:8192, vCores:4>	<memory:8192, vCores:4>								
Show 20 entries	ID	User	Name	Application Type	Application Tags	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	All Metrics

No data available in table

Showing 0 to 0 of 0 entries

DataNode on ip-172-31-23-74.ap-south-1.compute.internal:9866

Cluster ID:	CID-8e378a60-55a9-436a-b62a-9e6a3ca2b7be
Version:	3.3.1, ra3b9c37a397ad4188041dd80621bdeefc46885f2

Block Pools

Namenode Address	Block Pool ID	Actor State	Last Heartbeat	Last Block Report	Last Block Report Size (Max Size)
0.0.0.0:9000	BP-738289719-172.31.23.74-1670911094300	RUNNING	2s	11 minutes	0 B (128 MB)

Volume Information

Directory	StorageType	Capacity Used	Capacity Left	Capacity Reserved	Reserved Space for Replicas	Blocks
/home/hadoop/hdfs/datanode	DISK	28 KB	10.21 GB	0 B	0 B	0

Hadoop 2.0.2

Exercise 2

Implement the following file management tasks in Hadoop:

- Adding files and directories

- Retrieving files

- Deleting files

1.Creating directory :

```
hdfs dfs -mkdir/moni/
```

The screenshot shows the Hadoop Web UI interface. At the top, there's a navigation bar with links for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. Below the navigation bar, the main area is titled "Browse Directory". A search bar at the top of the list table contains the path "/". The table lists one entry: a directory named "moni" with owner "hadoop", group "supergroup", size "0 B", last modified "Dec 13 06:10", replication "0", and block size "0 B". The table has columns for Name, Block Size, Replication, Last Modified, Size, Group, Owner, and Permission. At the bottom of the table, it says "Showing 1 to 1 of 1 entries".

2.Listing the contents :

```
hadoop@ip-172-31-23-74:~$ hdfs dfs -ls /
Found 1 items
drwxr-xr-x - hadoop supergroup          0 2022-12-13 06:10 /moni
hadoop@ip-172-31-23-74:~$
```

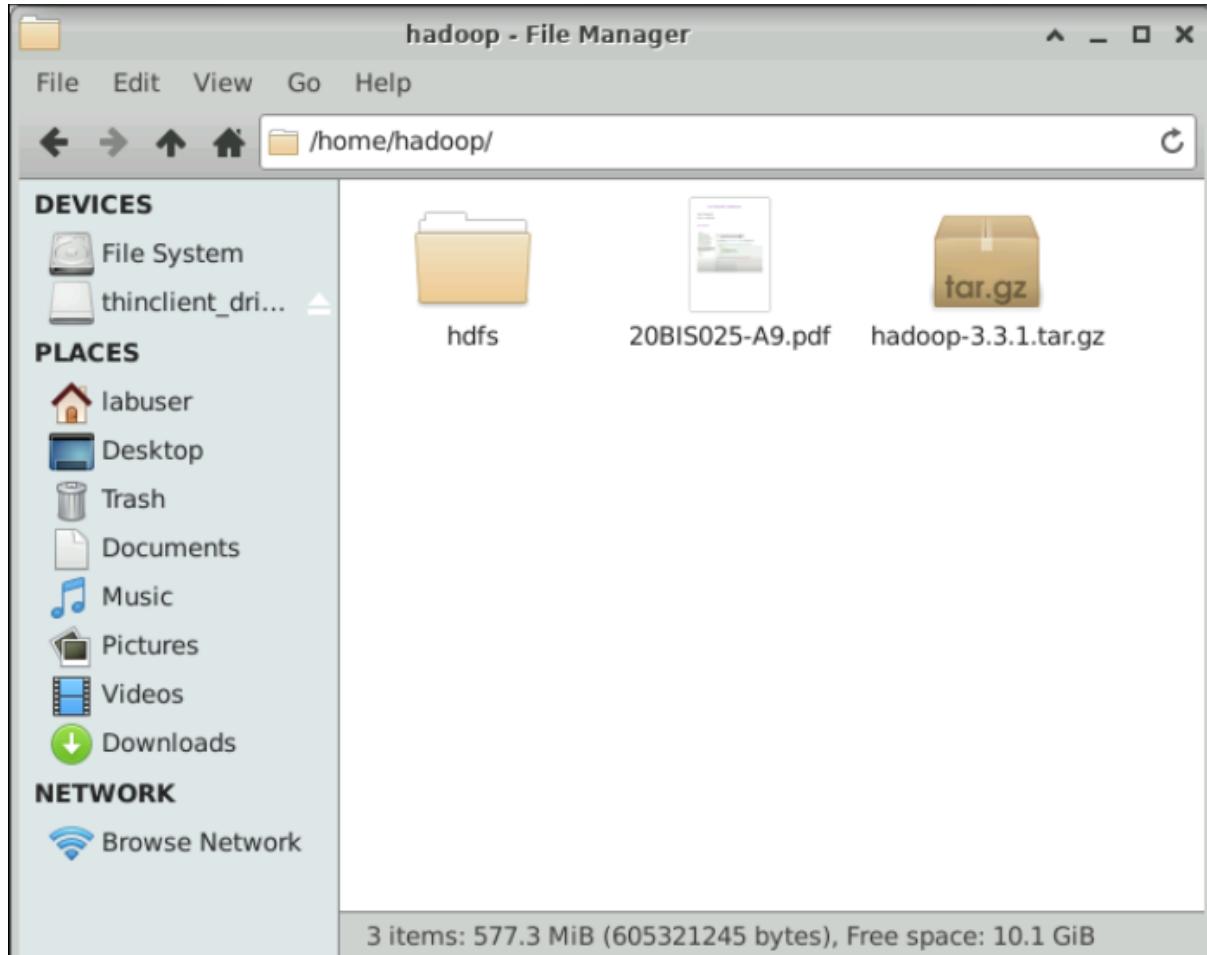
3.Uploading file on created directory :

```
hadoop@ip-172-31-23-74:~$ hdfs dfs -put '/home/labuser/Downloads/20BIS025-A9.pdf' /moni/
hadoop@ip-172-31-23-74:~$
```

The screenshot shows the Hadoop Web UI interface, similar to the previous one but with a different path in the search bar: "/moni". The main area is titled "Browse Directory". The table lists one file: "20BIS025-A9.pdf" with owner "hadoop", group "supergroup", size "130.83 KB", last modified "Dec 13 06:24", replication "1", and block size "128 MB". The table has columns for Name, Block Size, Replication, Last Modified, Size, Group, Owner, and Permission. At the bottom of the table, it says "Showing 1 to 1 of 1 entries".

4. Retrieving from Hadoop to local :

```
hadoop@ip-172-31-23-74:~$ hdfs dfs -get '/moni/20BIS025-A9.pdf' /home/hadoop/
hadoop@ip-172-31-23-74:~$
```



5. Cat command :

```
hadoop@ip-172-31-23-74:~$ hadoop fs -cat /moni/sample.txt
Hello , I am Monika . So i here with the presentation titled " infrastructure based and infrastructure less network .
before getting into the topic , lets discuss sbout wireless network .
Wireless network allows device to be connected or to be communicated without any physical cables .
Example when the devices are connected to WIFI.
Based on range and coverage , wireless network is classified into WAN,LAN and PAN
And based on topology , it classified as infrastructure and adhoc
```

6.Copying content of one directory to another :

```
hadoop@ip-172-31-23-74:~$ hadoop fs -cp /moni/sample.txt /moni2/
cp: `/moni2/': No such file or directory: `hdfs://0.0.0.0:9000/moni2'
hadoop@ip-172-31-23-74:~$ hdfs dfs -mkdir /moni2/
hadoop@ip-172-31-23-74:~$ hdfs dfs -cp /moni/sample.txt /moni2/
hadoop@ip-172-31-23-74:~$
```

The screenshot shows the HDFS browser interface. The URL is `localhost:9870/explorer.html#/`. The top navigation bar includes links for Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The main title is "Browse Directory" with a path of "/". A search bar and a file operations toolbar are present. A table lists two entries: "moni" and "moni2". Both entries have permissions `drwxr-xr-x`, owner "hadoop", group "supergroup", size 0 B, last modified Dec 13 07:05, replication 0, and block size 0 B. The "moni2" entry was added on Dec 13 07:08. The footer indicates "Hadoop, 2021."

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	<code>drwxr-xr-x</code>	hadoop	supergroup	0 B	Dec 13 07:05	0	0 B	moni
□	<code>drwxr-xr-x</code>	hadoop	supergroup	0 B	Dec 13 07:08	0	0 B	moni2

The screenshot shows the HDFS browser interface. The URL is `localhost:9870/explorer.html#/moni2`. The top navigation bar is identical to the first screenshot. The main title is "Browse Directory" with a path of "/moni2". A search bar and a file operations toolbar are present. A table lists one entry: "sample.txt". It has permissions `-rw-r--r--`, owner "hadoop", group "supergroup", size 2.39 KB, last modified Dec 13 07:08, replication 1, and block size 128 MB. The footer indicates "Hadoop, 2021."

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	<code>-rw-r--r--</code>	hadoop	supergroup	2.39 KB	Dec 13 07:08	1	128 MB	sample.txt

7.Copying from local to Hadoop :

```
hadoop@ip-172-31-23-74:~$ hadoop fs -copyFromLocal '/home/labuser/Downloads/bigdata.txt' /moni2
hadoop@ip-172-31-23-74:~$
```

The screenshot shows the HDFS browser interface. The URL is `localhost:9870/explorer.html#/moni2`. The top navigation bar is identical to the previous screenshots. The main title is "Browse Directory" with a path of "/moni2". A search bar and a file operations toolbar are present. A table lists two entries: "bigdata.txt" and "sample.txt". Both entries have permissions `-rw-r--r--`, owner "hadoop", group "supergroup", size 77 B and 2.39 KB respectively, last modified Dec 13 07:12 and 07:08, replication 1, and block size 128 MB. The footer indicates "Hadoop, 2021."

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	<code>-rw-r--r--</code>	hadoop	supergroup	77 B	Dec 13 07:12	1	128 MB	bigdata.txt
□	<code>-rw-r--r--</code>	hadoop	supergroup	2.39 KB	Dec 13 07:08	1	128 MB	sample.txt

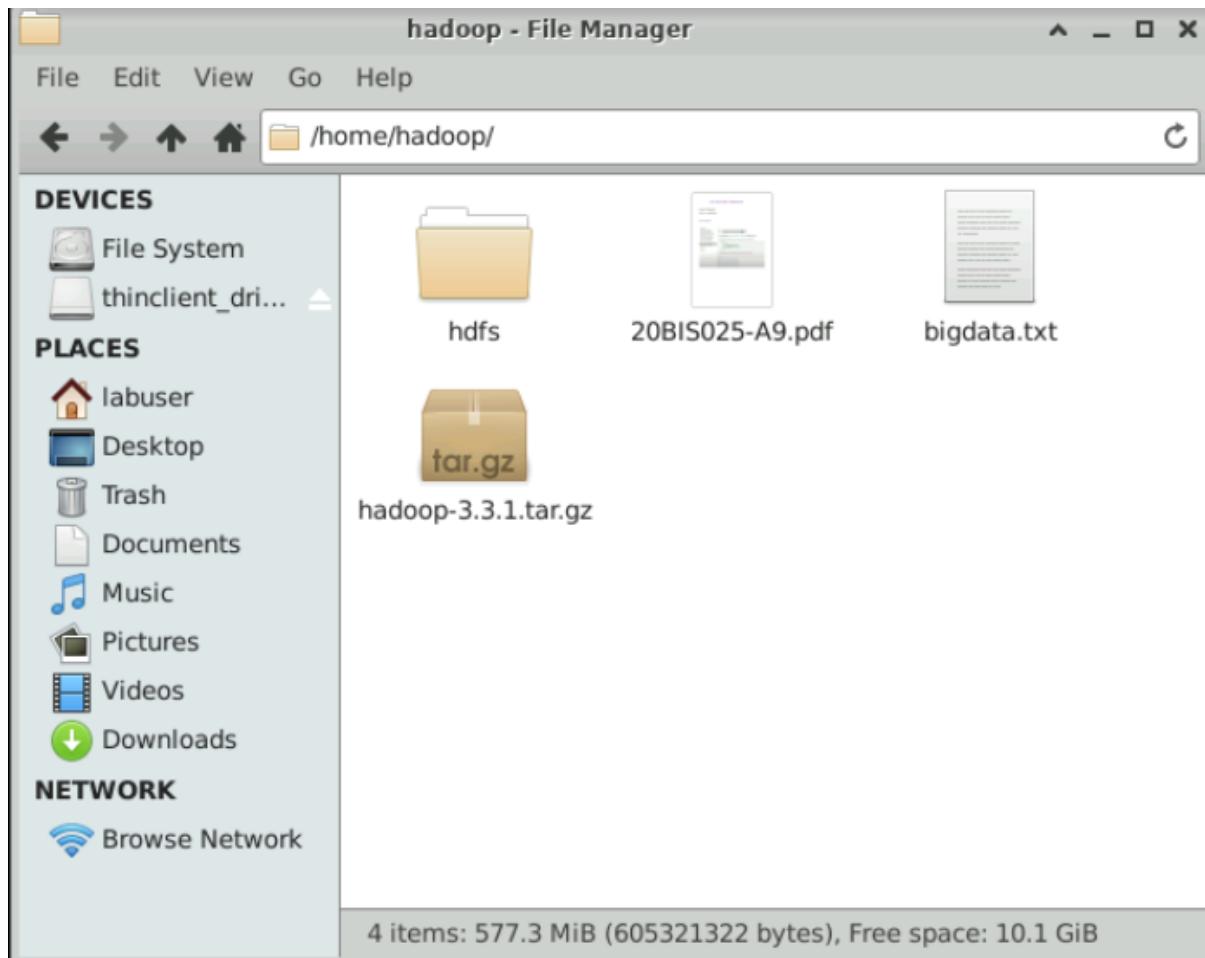
Browse Directory

The screenshot shows the HDFS browser interface. The URL is `localhost:9870/explorer.html#/moni2`. The top navigation bar is identical to the previous screenshots. The main title is "Browse Directory" with a path of "/moni2". A search bar and a file operations toolbar are present. A table lists two entries: "bigdata.txt" and "sample.txt". Both entries have permissions `-rw-r--r--`, owner "hadoop", group "supergroup", size 77 B and 2.39 KB respectively, last modified Dec 13 07:12 and 07:08, replication 1, and block size 128 MB. The footer indicates "Hadoop, 2021."

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	<code>-rw-r--r--</code>	hadoop	supergroup	77 B	Dec 13 07:12	1	128 MB	bigdata.txt
□	<code>-rw-r--r--</code>	hadoop	supergroup	2.39 KB	Dec 13 07:08	1	128 MB	sample.txt

8. Copying from Hadoop to local :

```
hadoop@ip-172-31-23-74:~$ hadoop fs -copyToLocal /moni2/bigdata.txt '/home/hadoop/'  
hadoop@ip-172-31-23-74:~$
```



9. Moving file from one directory(source) to another(destination):

The file is moved to destination and wont be available on source

```
hadoop@ip-172-31-23-74:~$ hdfs dfs -mv /moni2/bigdata.txt /moni/  
hadoop@ip-172-31-23-74:~$
```

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hadoop	supergroup	130.83 KB	Dec 13 06:24	1	128 MB	20BIS025-A9.pdf
-rw-r--r--	hadoop	supergroup	77 B	Dec 13 07:12	1	128 MB	bigdata.txt
-rw-r--r--	hadoop	supergroup	2.39 KB	Dec 13 07:05	1	128 MB	sample.txt

10.Remove a file or directory :

File remove : hdfs dfs -rm /moni2/sample.txt

Directory remove : hdfs -dfs -rmdir /moni2/

```
hadoop@ip-172-31-23-74:~$ hdfs dfs -rm /moni2/sample.txt
Deleted /moni2/sample.txt
hadoop@ip-172-31-23-74:~$ hdfs dfs -rmdir /moni2/
hadoop@ip-172-31-23-74:~$
```

Browse Directory

/									Go!	File	Folder	Image	Text
Show 25 entries		Search:											
	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name					
<input type="checkbox"/>	drwxr-xr-x	hadoop	supergroup	0 B	Dec 13 07:16	0	0 B	moni					
Showing 1 to 1 of 1 entries													
Hadoop, 2021.													

11.Display last few lines of a file :

```
hadoop@ip-172-31-23-74:~$ hadoop fs -tail /moni/sample.txt/
for better understanding

First , infrastructure based network
The demo consists of a switch,router, an access point ,2pcs(one is connected physically , another is wireless) , a laptop , a printer , tablet and a mobile phone.
here i establishes both the wired as well as wireless network to demonstrate how the communication occur but i am running out of time so i am doing it . focus on The wireless
side that is infrastructure based. I need to transferring a package from PC2 to tablet . So i made them as source and destination and the transmission happening , so in progr
ess
The packet received in tablet and in response to that the tablet is sending acknowledgement .
So pc2 received acknowledgement so the communication via access point is successfully

now adhoc
already said adhoc is for temporary network . I used a speaker and a music player and connected them via bluetooth . Lets check connectivity , if they connected , speaker make
s sound.

comes the end of presentation , thank you !
hadoop@ip-172-31-23-74:~$
```

12.Display the aggregate length of a file:

```
hadoop@ip-172-31-23-74:~$ hadoop fs -du /moni/sample.txt
2452 2452 /moni/sample.txt
hadoop@ip-172-31-23-74:~$
```

Exercise-3

Title of the exercise/experiment: Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

➤ Implementing distinct word count problem using Map-Reduce
Theory:

To run a MapReduce job, users should furnish a map function, a reducefunction, input data, and an output data location.

When executed, Hadoop carries out the following steps:

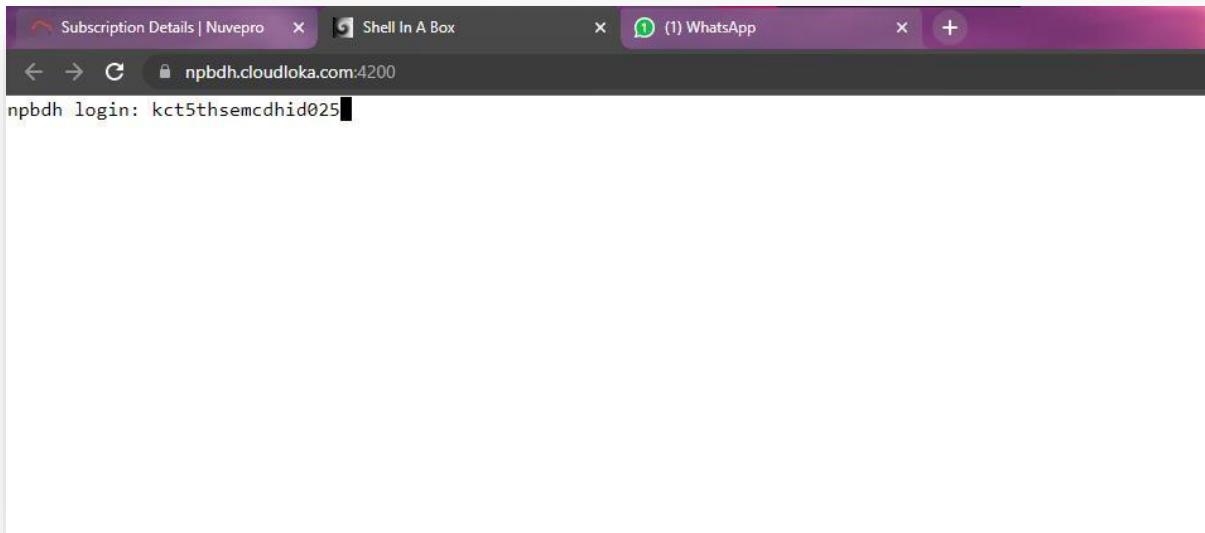
1. Hadoop breaks the input data into multiple data items by new linesand runs the map function once for each data item, giving the item as the input for the function. When executed, the map function outputs one or more key- value pairs.
2. Hadoop collects all the key-value pairs generated from the mapfunction, sorts them by the key, and groups together the valueswith the same key.
3. For each distinct key, Hadoop runs the reduce function once whilepassing the key and list of values for that key as input.
4. The reduce function may output one or more key-value pairs, andHadoop writes them to a file as the final result.

WORDCOUNT EXERCISE:

- 1) LOGIN TO THE WEB SHELL USING THE USERNAME AND PAAWORD

Access Details

Login username	kct5thsemcdhid025
Login password	<input type="password" value="KCT5thSem"/>  
SSH login server	npbdh.cloudloka.com



```
npbdh login: kct5thsemcdhid025
kct5thsemcdhid025@npbdh.cloudloka.com's password:
Last login: Sat Dec 10 03:13:13 2022 from ec2-65-1-45-35.ap-south-1.compute.amazonaws.com
[kct5thsemcdhid025@ip-10-1-1-204 ~]$
```

2) CREATE A FOLDER/ DIRECTORY FOR WORDCOUNT IN HOME

COMMAND: mkdir wordcount

```
npbdh login: kct5thsemcdhid025
kct5thsemcdhid025@npbdh.cloudloka.com's password:
Last login: Sat Dec 10 05:13:30 2022 from ec2-65-1-45-35.ap-south-1.compute.amazonaws.com
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ mkdir Wordcount
```

3) MOVE TO THAT FOLDER/DIRECTORY:

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ cd Wordcount
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$
```

4) CREATE A INPUT FILE INSIDE THE WORDCOUNT FOLDER:

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ touch input.txt
```

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ vi input.txt
```

PASTE THE INPUT IN THE INPUT FILE:

```
Hlo
Ise
Gokila
KCT
Hi
SMALL
LARGE
MEDIUM
Hello
KCT
BIG
DATA
BIG DATA
DATA
SMALL
LARGE
MEDIUM
-- INSERT --
```

35,7

Bot

5) CREATE A WORDCOUNT JAVA FILE:

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ touch WordCount.java
```

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ vi WordCount.java
```

PASTE THE SOURCE CODE IN THE FILE:

```
public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
int sum=0;
for(IntWritable x: values) {
sum+=x.get();
}
context.write(key, new IntWritable(sum));
}
}

public static void main(String[] args) throws Exception {

Configuration conf= new Configuration();
Job job = new Job(conf,"My Word Count Program");
job.setJarByClass(WordCount.class);
-- INSERT --
58,1 64%
```

SOURCE CODE:

```
import
java.io.IOException;
import
java.util.StringTokenize
r;
import
org.apache.hadoop.io.IntWrita
ble; import
org.apache.hadoop.io.LongWrit
able; import
org.apache.hadoop.io.Text;
import
org.apache.hadoop.mapreduce.Ma
pper; import
org.apache.hadoop.mapreduce.Re
ducer; import
```

```
org.apache.hadoop.conf.Configuration; import  
org.apache.hadoop.mapreduce.Job;  
;  
import  
org.apache.hadoop.mapreduce.lib.input.TextInputFormat; import  
org.apache.hadoop.mapreduce.lib.output.TextOutputFormat; import  
org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import  
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; import org.apache.hadoop.fs.Path;  
  
public class WordCount {  
    public static class Map extends  
Mapper<LongWritable,Text,Text,IntWritable> {public  
void map(LongWritable key, Text value,Context context)  
throws
```

```
IOException,InterruptedException  
Exception{String line =  
value.toString();  
StringTokenizer tokenizer = new  
StringTokenizer(line);while  
(tokenizer.hasMoreTokens()) {  
value.set(tokenizer.nextToken());  
context.write(value, new IntWritable(1));  
}  
}  
}  
  
public static class Reduce extends  
Reducer<Text,IntWritable,Text,IntWritable> {public  
void reduce(Text key, Iterable<IntWritable>  
values,Context context) throws  
IOException,InterruptedException {  
int sum=0;  
for(IntWritable x:  
values)  
{  
sum+=x.get();  
}  
context.write(key, new IntWritable(sum));  
}  
}  
  
public static void main(String[] args) throws Exception {  
Configuration conf= new Configuration();  
Job job = new Job(conf,"My Word  
Count Program");
```

```
job.setJarByClass(WordCount.class);
job.setMapperClass(Map.class);
job.setReducerClass(Reduce.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.c
lass);
job.setInputFormatClass(TextInputFor
mat.class);
job.setOutputFormatClass(TextOutput
Format.class); Path outputPath = new
Path(args[1]);
//Configuring the input/output path from the
filesystem into the job
FileInputFormat.addInputPath(job, new
Path(args[o]));
```

```
FileOutputFormat.setOutputPath(job, new Path(args[1]));
//deleting the output path automatically from hdfs so that we don't
have to delete it explicitly
outputPath.getFileSystem(conf).delete(outputPath);
//exiting the job only if the flag value
becomes false
System.exit(job.waitForCompletion(t
rue) ? 0 : 1);
}
}
```

6) CREATE A DIRECTORY/ FOLDER CLASSES INSIDE THE WORDCOUNT FOLDER:

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ mkdir classes
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ ll
total 12
drwxrwxr-x 2 kct5thsemcdhid025 kct5thsemcdhid025 4096 Dec 10 05:20 classes
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 113 Dec 10 05:16 input.txt
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 2327 Dec 10 05:19 WordCount.java
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ █
```

6) SET THE PATH FOR JAVA FILE

COMMAND: `export HADOOP_CLASSPATH=$(hadoop classpath)`

```
echo $HADOOP_CLASSPATH
```

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ export HADOOP_CLASSPATH=$(hadoop classpath)
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ echo $HADOOP_CLASSPATH
/etc/hadoop/conf:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/./:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/../../opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/../../opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-yarn/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-yarn/../*
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$
```

7) COMPILE THE JAVAFILE:

COMMAND: javac -classpath \${HADOOP_CLASSPATH} -d
'/home/<NAME>/wordcount/classes'
'/home/<NAME>/wordcount/WordCount.java'

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ javac -classpath ${HADOOP_CLASSPATH} -d '/home/kct5thsemcdhid025/Wordcount/classes' '/home/kct5thsemcdhid025/WordCount.java'
Note: /home/kct5thsemcdhid025/WordCount.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$
```

8) CREATE A JAR FILE:

COMMAND: jar -cvf WordCount.jar -C '/home/<NAME>/wordcount/classes' .

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ jar -cvf WordCount.jar -C '/home/kct5thsemcdhid025/Wordcount/classes' .
added manifest
adding: WordCount.class(in = 1814) (out= 914)(deflated 49%)
adding: WordCount$Map.class(in = 1657) (out= 691)(deflated 58%)
adding: WordCount$Reduce.class(in = 1627) (out= 686)(deflated 57%)
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$
```

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ ll
total 16
drwxrwxr-x 2 kct5thsemcdhid025 kct5thsemcdhid025 4096 Dec 10 05:32 classes
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 113 Dec 10 05:16 input.txt
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 3021 Dec 10 05:34 WordCount.jar
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 2327 Dec 10 05:19 WordCount.java
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$
```

9) CREATE A DIRECTORY IN HADOOP

```
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir -p /user/kct5thsemcidhid025/wordcount  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$
```

10) PUT THE INPUT FILE IN LOCAL SYSTEM TO HADOOP DIRECTORY:

```
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -put /home/kct5thsemcidhid025/Wordcount/input.txt /user/kct5thsemcidhid025/wordcount/input/  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$
```

11) CREATE A OUTPUT DIRECTORY IN HDFS INSIDE THE WORDCOUNT

```
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcidhid025/wordcount/output  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$
```

12) RUN THE MAP REDUCE PROGRAM:

COMMAND: `hadoop jar /home/<NAME>/wordcount/WordCount.jar' WordCount /user/<NAME>/wordcount/input /user/<NAME>/wordcount/output`

```
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ hadoop jar /home/kct5thsemcdhid025/WordCount/WordCount.jar WordCount /user/kct5thsemcdhid025/wordcount/input /user/kct5thsemcdhid025/wordcount/output
WARNING: Use "yarn jar" to launch YARN applications.
22/12/10 05:39:41 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/10 05:39:42 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
22/12/10 05:39:42 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/kct5thsemcdhid025/.staging/job_1663041244711_19338
22/12/10 05:39:42 INFO input.FileInputFormat: Total input files to process : 1
22/12/10 05:39:42 INFO mapreduce.JobSubmitter: number of splits:1
22/12/10 05:39:42 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
22/12/10 05:39:42 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1663041244711_19338
22/12/10 05:39:42 INFO mapreduce.JobSubmitter: Executing with tokens: []
22/12/10 05:39:42 INFO conf.Configuration: resource-types.xml not found
22/12/10 05:39:42 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
22/12/10 05:39:42 INFO impl.YarnClientImpl: Submitted application application_1663041244711_19338
22/12/10 05:39:42 INFO mapreduce.Job: The url to track the job: http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_19338/
22/12/10 05:39:42 INFO mapreduce.Job: Running job: job_1663041244711_19338
22/12/10 05:39:51 INFO mapreduce.Job: Job job_1663041244711_19338 running in uber mode : false
22/12/10 05:39:51 INFO mapreduce.Job: map 0% reduce 0%
22/12/10 05:39:57 INFO mapreduce.Job: map 100% reduce 0%
22/12/10 05:40:11 INFO mapreduce.Job: map 100% reduce 40%
22/12/10 05:40:12 INFO mapreduce.Job: map 100% reduce 60%
22/12/10 05:40:13 INFO mapreduce.Job: map 100% reduce 80%
22/12/10 05:40:14 INFO mapreduce.Job: map 100% reduce 100%
22/12/10 05:40:14 INFO mapreduce.Job: Job job_1663041244711_19338 completed successfully
22/12/10 05:40:14 INFO mapreduce.Job: Counters: 54
File System Counters
FILE: Number of bytes read=262
FILE: Number of bytes written=1338661
FILE: Number of read operations=0
```

```
hdfs: number of bytes read erasure-coded=0
Job Counters
Launched map tasks=1
Launched reduce tasks=5
Data-local map tasks=1
Total time spent by all maps in occupied slots (ms)=3503
Total time spent by all reduces in occupied slots (ms)=66189
Total time spent by all map tasks (ms)=3503
Total time spent by all reduce tasks (ms)=66189
Total vcore-milliseconds taken by all map tasks=3503
Total vcore-milliseconds taken by all reduce tasks=66189
Total megabyte-milliseconds taken by all map tasks=3587072
Total megabyte-milliseconds taken by all reduce tasks=67777536
Map-Reduce Framework
Map input records=35
Map output records=19
Map output bytes=172
Map output materialized bytes=242
Input split bytes=133
Combine input records=0
Combine output records=0
Reduce input groups=11
Reduce shuffle bytes=242
Reduce input records=19
Reduce output records=11
Spilled Records=38
Shuffled Maps =5
Failed Shuffles=0
Merged Map outputs=5
GC time elapsed (ms)=1742
CPU time spent (ms)=8020
Physical memory (bytes) snapshot=1794891776
Virtual memory (bytes) snapshot=15591702528
Total committed heap usage (bytes)=2802843648
Peak Mem Physical memory (bytes)=519364608
```

13) VERIFY THE OUTPUT:

COMMAND: hdfs dfs -cat /user/<Name>/wordcount/output/*

```
bytes written=0
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$ hdfs dfs -cat /user/kct5thsemcdhid025/wordcount/output/*
Ise      1
BIG      2
Hello    1
Gokila   1
Hlo      1
KCT      2
MEDIUM   2
DATA     3
Hi       2
SMALL    2
LARGE    2
[kct5thsemcdhid025@ip-10-1-1-204 Wordcount]$
```

LAB EXERCISE-4

Title of the exercise/experiment: Implementation of Matrix Multiplication using MapReduce

Theory:

In mathematics, matrix multiplication or the matrix product is a binary operation that produces a matrix from two matrices. In more detail, if A is an $n \times m$ matrix and B is an $m \times p$ matrix, their matrix product AB is an $n \times p$ matrix, in which the m entries across a row of A are multiplied with the m entries down a column of B and summed to produce an entry of AB. When two linear transformations are represented by matrices, then the matrix product represents the composition of the two transformations.

Source Code:

```
import  
java.io.IOException;  
import java.util.*;  
import  
java.util.AbstractMap.SimpleEntry;  
import java.util.Map.Entry;  
import  
org.apache.hadoop.fs.Path;  
import  
org.apache.hadoop.conf.*;  
import org.apache.hadoop.io.*;  
import org.apache.hadoop.mapreduce.*;  
import  
org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import  
org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
```

```
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.Text
OutputFormat; public class
TwoStepMatrixMultiplication {
    public static class Map extends Mapper<LongWritable,
    Text, Text, Text> {public void map(LongWritable key,
    Text value, Context context)
    throws IOException, InterruptedException
    { String line = value.toString();
    String[] indicesAndValue =
    line.split(",");
    Text outputKey =
    new Text();
    Text outputValue = new Text();
    if (indicesAndValue[0].equals("A")) {
    outputKey.set(indicesAndValue[2]);
    outputValue.set("A," + indicesAndValue[1] +
    "," + indicesAndValue[3]);
    context.write(outputKey, outputValue);
    } else {
    outputKey.set(indicesAndValue[1]);
    outputValue.set("B," +
    indicesAndValue[2] + "," +
    indicesAndValue[3]);
    context.write(outputKey, outputValue);
    }
    }
    }
    public static class Reduce extends Reducer<Text,
    Text, Text, Text> {public void reduce(Text key,
    Iterable<Text> values,
```

```
Context context) throws IOException,  
InterruptedException { String[] value;  
ArrayList<Entry<Integer, Float>>  
listA = new  
ArrayList<Entry<Integer,  
Float>>();  
ArrayList<Entry<Integer, Float>>  
listB = new  
ArrayList<Entry<Integer,  
Float>>();  
for (Text val : values) {  
value = val.toString().split(",");  
if (value[0].equals("A")) {  
listA.add(new  
SimpleEntry<Integer,  
Float>(Integer.parseInt(value[1]), Float.parseFloat(value[2])));  
} else {  
listB.add(new SimpleEntry<Integer,  
Float>(Integer.parseInt(value[1]),  
Float.parseFloat(value[2])));  
}  
}  
String i;  
float  
a_ij;  
String k;  
floatb_jk  
;
```

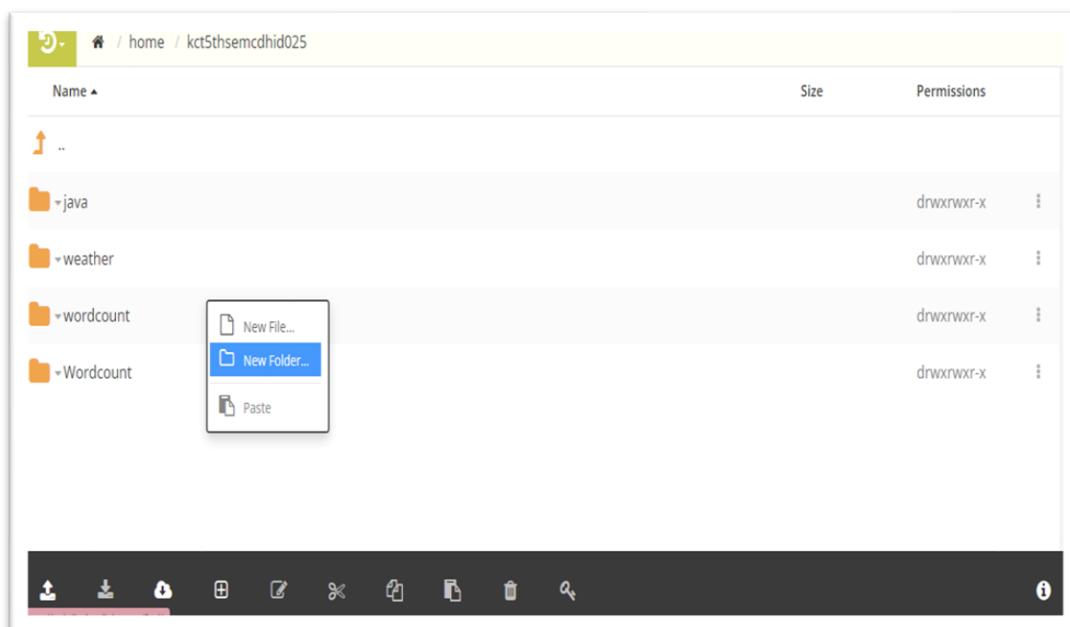
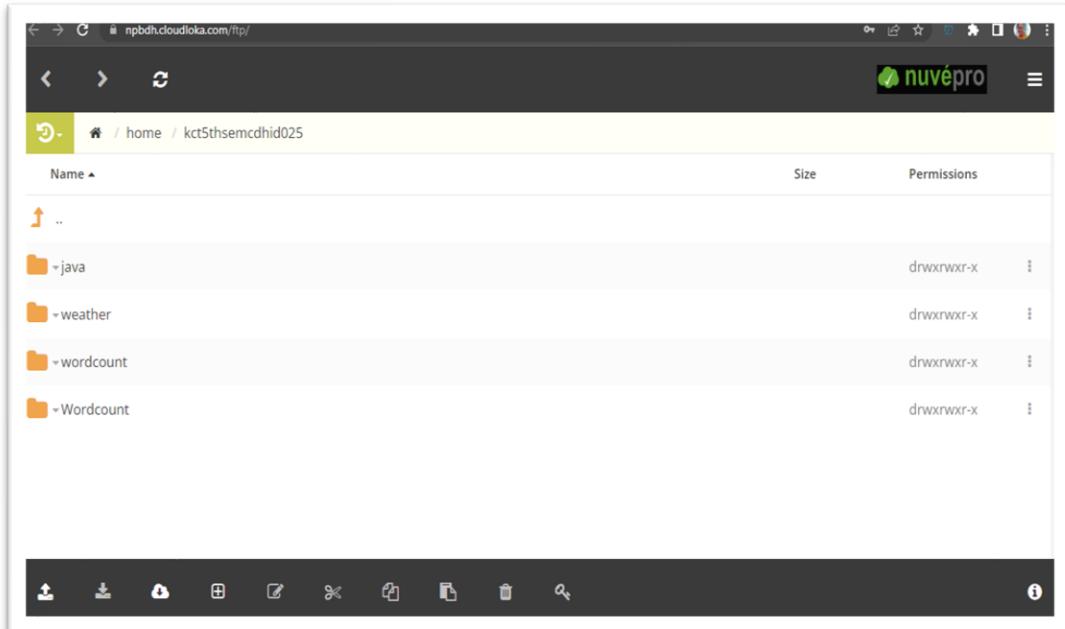
```
Text outputValue = new Text();
for (Entry<Integer, Float> a :
listA) { i =
Integer.toString(a.getKey());
a_ij = a.getValue();
for (Entry<Integer, Float> b :
listB) { k =
Integer.toString(b.getKey());
b_jk = b.getValue();
```

```
        outputValue.set(i + "," + k +
        "," +
        Float.toString(a_ij*b_jk));
    context.write(null,
        outputValue);
    }
}
}
}

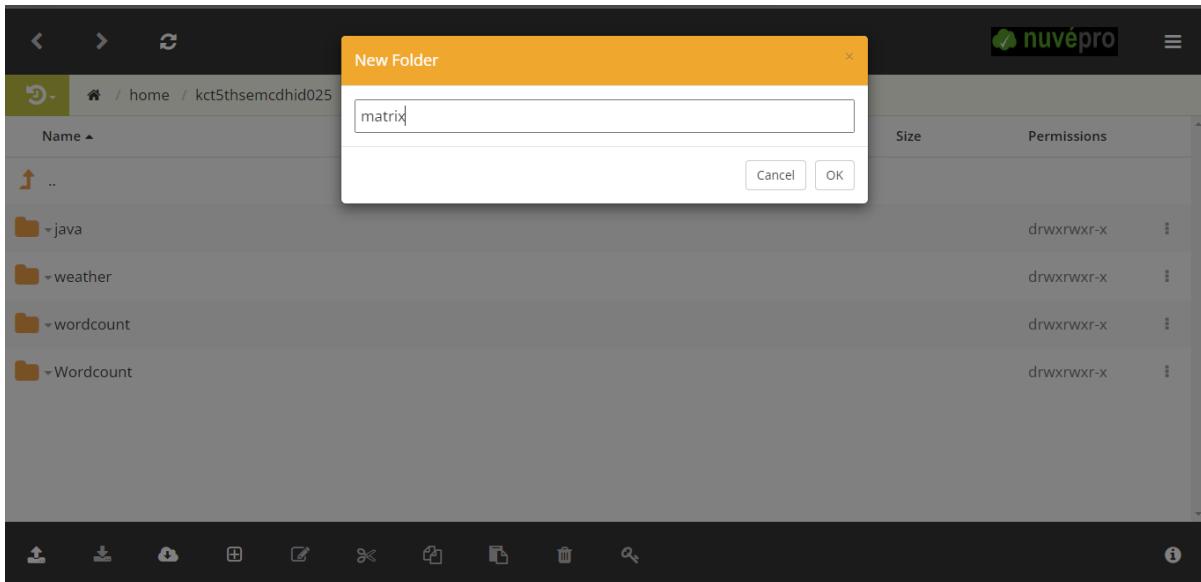
public static void main(String[] args)
throws Exception {
    Configuration conf =
    new Configuration();
    Job job = new Job(conf,
        "MatrixMatrixMultiplicationTwoSteps");
    job.setJarByClass(TwoStepMatrixMultiplication.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
    job.setMapperClass(Map.class);
    job.setReducerClass(Reduce.class);
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);
    FileInputFormat.addInputPath(job, new Path("hdfs://
127.0.0.1:9000/matrixin"));
    FileOutputFormat.setOutputPat
    h(job, new
    Path("hdfs://127.0.0.1:9000/ma
trixout"));
    job.waitForCompletion(true);
}
}
```

MATRIX MULTIPLICATION:

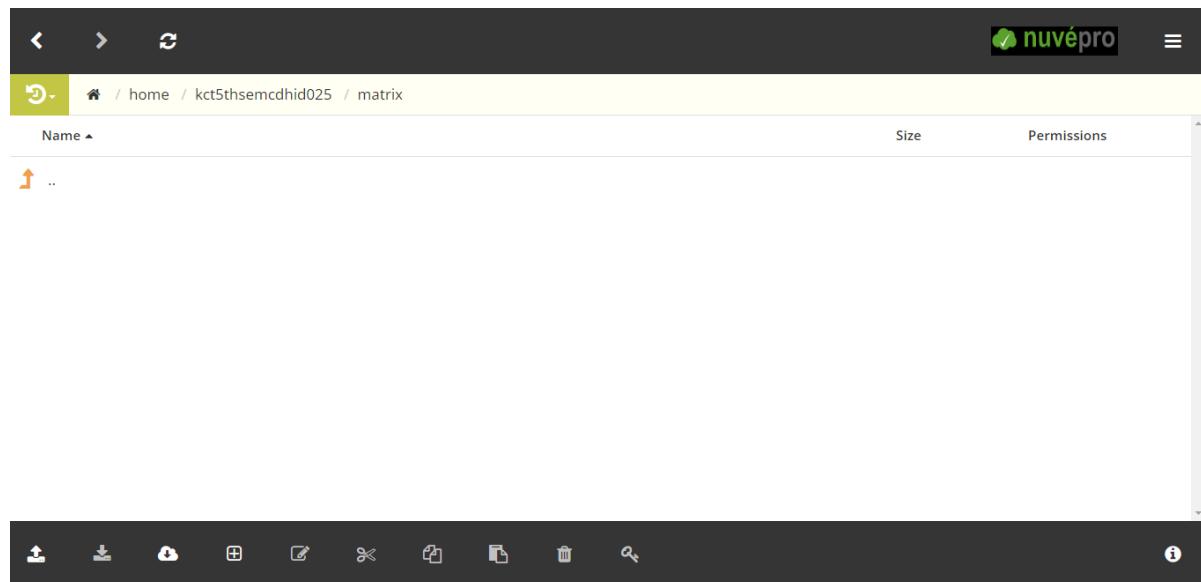
1) LOGIN WITH THE USERNAME AND PASSWORD



CREATE A FOLDER FOR MATRIX

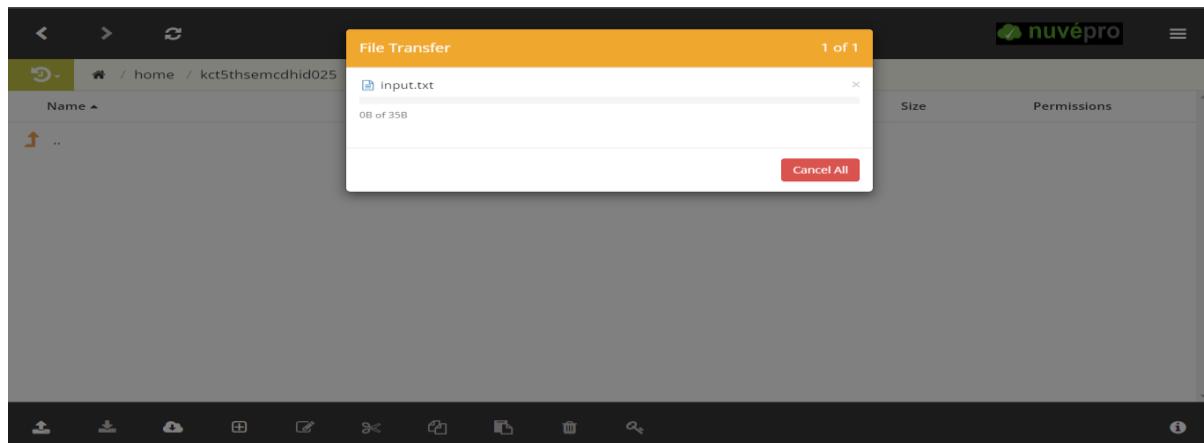


INSIDE THE MATRIX FOLDER UPLOAD THE INPUT FILE AND JAVA FILE

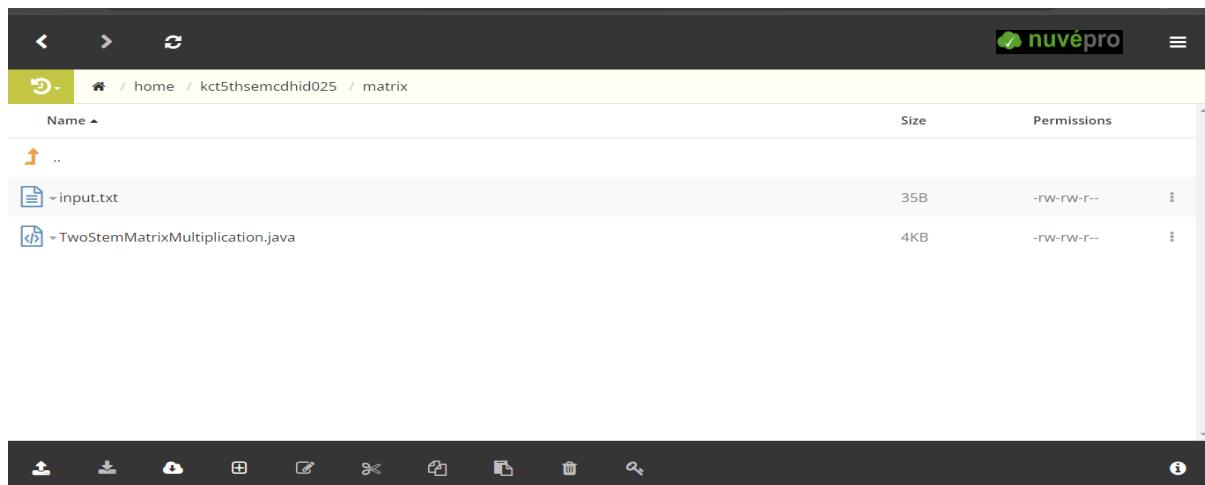


CAN UPLOAD/EDIT/DELETE THE FILES BY USING THESE OPTIONS:

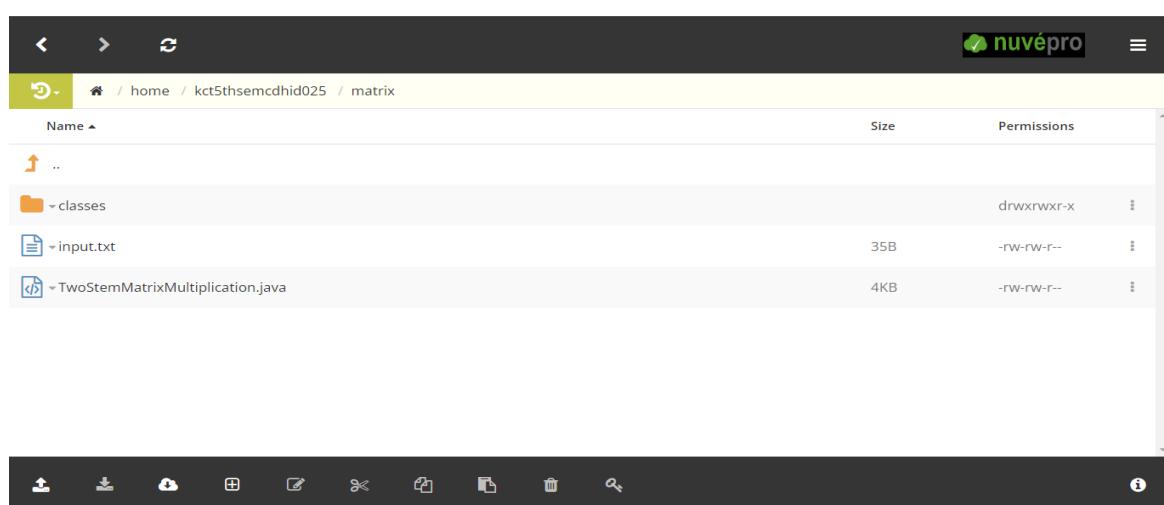
UPLOAD THE INPUT FILE:



UPLOAD THE JAVA FILE:



CREATE A CLASS FOLDER:



IN WEB SHELL GO TO THE MATRIX FOLDER:

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ cd matrix
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ ll
total 12
drwxrwxr-x 2 kct5thsemcdhid025 kct5thsemcdhid025 4096 Dec 12 05:50 classes
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 35 Dec 12 05:47 input.txt
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 3666 Dec 12 05:50 TwoStepMatrixMultiplication.java
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ █
```

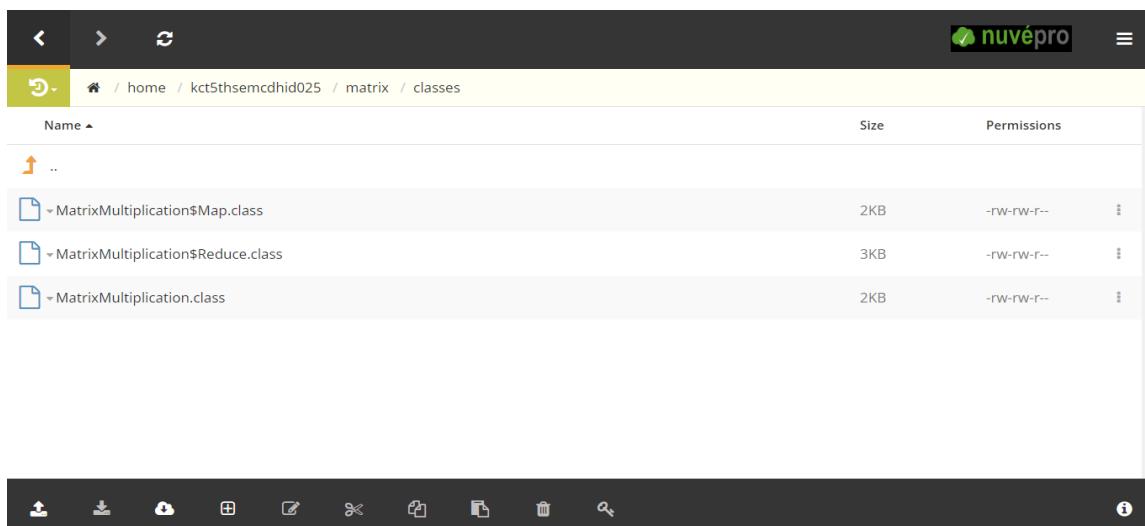
SET THE PATH FOR JAVA FILE:

```
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ export HADOOP_CLASSPATH=$(hadoop classpath)
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ echo $HADOOP_CLASSPATH
/etc/hadoop/conf:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop-hdfs/./:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop-hdfs/.*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop-hdfs/..:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop-mapreduce/.*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop-yarn/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/.../hadoop-yarn/...
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ █
```

COMPILE THE JAVA FILE

```
javac -classpath ${HADOOP_CLASSPATH} -d '/home/<NAME> / matrix /classes'
'/home/<NAME>/ matrix /
TwoStepMatrixMultiplication.java'
```

```
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ javac -classpath ${HADOOP_CLASSPATH} -d '/home/kct5thsemcdhid025/matrix/classes' '/home/kct5thsemcdhid025/matrix/TwoStepMatrixMultiplication.java'
Note: /home/kct5thsemcdhid025/matrix/TwoStepMatrixMultiplication.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ █
```

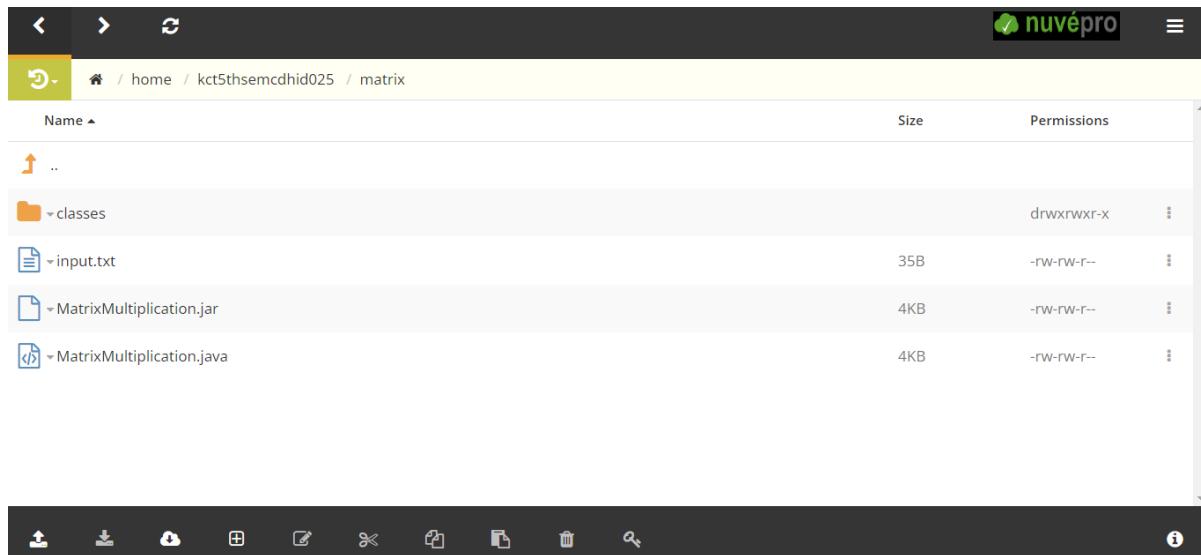


CHECK THE CLASSES FOLDER:

CREATE A JAR FILE

```
jar -cvf TwoStepMatrixMultiplication.jar -C '/home/<NAME>/matrix /classes' .
```

```
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ jar -cvf MatrixMultiplication.jar -C '/home/kct5thsemcdhid025/matrix/classes' .  
added manifest  
adding: MatrixMultiplication$Reduce.class(in = 2985) (out= 1292)(deflated 56%)  
adding: MatrixMultiplication.class(in = 1845) (out= 954)(deflated 48%)  
adding: MatrixMultiplication$Map.class(in = 2411) (out= 1012)(deflated 58%)  
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$
```



```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcdhid025/matrix  
[kct5thsemcdhid025@ip-10-1-1-204 ~]$
```

CREATE A DIRECTORY IN HDFS

CREATE An INPUT AND OUTPUT FOLDER INSIDE THE DIRECTORY:

```
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcidhid025/matrix/input  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcidhid025/matrix/output  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$
```

PUT THE INPUT TEXT FILE INSIDE THE INPUT FOLDER:

```
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -put '/home/kct5thsemcidhid025/matrix/input.txt' '/user/kct5thsemcidhid025/matrix/input/'  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$
```

CHECK THE INPUT FILE

```
[kct5thsemcidhid025@ip-10-1-1-204 ~]$ hdfs dfs -cat /user/kct5thsemcidhid025/matrix/input/*  
M,0,0,1  
M,0,1,2  
M,1,0,3  
M,1,1,4  
[kct5thsemcidhid025@ip-10-1-1-204 ~]$
```

```
[kct5thsemcidhid025@ip-10-1-1-204 matrix]$ hadoop jar /home/kct5thsemcidhid025/matrix/MatMul.jar MatMul /user/kct5thsemcidhid025/Matrix/input /user/kct5thsemcidhid025/Matrix/output  
WARNING: Use "yarn jar" to launch YARN applications.  
22/12/12 15:42:14 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032  
22/12/12 15:42:14 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
22/12/12 15:42:14 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/kct5thsemcidhid025/.staging/job_1663041244711_21353  
22/12/12 15:42:15 INFO input.FileInputFormat: Total input files to process : 1  
22/12/12 15:42:15 INFO mapreduce.JobSubmitter: number of splits:1  
22/12/12 15:42:15 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled  
22/12/12 15:42:15 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1663041244711_21353  
22/12/12 15:42:15 INFO mapreduce.JobSubmitter: Executing with tokens: []  
22/12/12 15:42:15 INFO conf.Configuration: resource-types.xml not found  
22/12/12 15:42:15 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
22/12/12 15:42:15 INFO impl.YarnClientImpl: Submitted application application_1663041244711_21353  
22/12/12 15:42:15 INFO mapreduce.Job: The url to track the job: http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21353/  
22/12/12 15:42:15 INFO mapreduce.Job: Running job: job_1663041244711_21353
```

```
[kct5thsemcidhid025@ip-10-1-1-204 matrix]$ hadoop jar /home/kct5thsemcidhid025/matrix/MatMul.jar MatMul /user/kct5thsemcidhid025/MATRIX/input /user/kct5thsemcidhid025/MATRIX/output  
WARNING: Use "yarn jar" to launch YARN applications.  
22/12/13 10:46:55 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032  
22/12/13 10:46:56 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.  
22/12/13 10:46:56 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/kct5thsemcidhid025/.staging/job_1663041244711_21821  
22/12/13 10:46:56 INFO input.FileInputFormat: Total input files to process : 1  
22/12/13 10:46:56 INFO mapreduce.JobSubmitter: number of splits:1  
22/12/13 10:46:56 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled  
22/12/13 10:46:56 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1663041244711_21821  
22/12/13 10:46:56 INFO mapreduce.JobSubmitter: Executing with tokens: []  
22/12/13 10:46:56 INFO conf.Configuration: resource-types.xml not found  
22/12/13 10:46:56 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.  
22/12/13 10:46:56 INFO impl.YarnClientImpl: Submitted application application_1663041244711_21821  
22/12/13 10:46:56 INFO mapreduce.Job: The url to track the job: http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21821/  
22/12/13 10:46:56 INFO mapreduce.Job: Running job: job_1663041244711_21821 running in uber mode : false  
22/12/13 10:47:05 INFO mapreduce.Job: map 0% reduce 0%  
22/12/13 10:47:11 INFO mapreduce.Job: map 100% reduce 0%
```

```
npbdh.cloudoka.com:4200
Total megabyte-milliseconds taken by all map tasks=4665280
Total megabyte-milliseconds taken by all reduce tasks=111104000
Map-Reduce Framework
Map input records=8
Map output records=8
Map output bytes=64
Map output materialized bytes=148
Input split bytes=130
Combine input records=0
Combine output records=0
Reduce input groups=2
Reduce shuffle bytes=148
Reduce input records=8
Reduce output records=8
Spilled Records=16
Shuffled Maps =5
Failed Shuffles=0
Merged Map outputs=5
GC time elapsed (ms)=2166
CPU time spent (ms)=6510
Physical memory (bytes) snapshot=1527721984
Virtual memory (bytes) snapshot=15533948928
Total committed heap usage (bytes)=1595932672
Peak Map Physical memory (bytes)=493154304
Peak Map Virtual memory (bytes)=2577018880
Peak Reduce Physical memory (bytes)=234246144
Peak Reduce Virtual memory (bytes)=2592227328
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=72
File Output Format Counters
Bytes Written=70
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$
```

OUTPUT:

```
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$ hdfs dfs -cat /user/kct5thsemcdhid025/MATRIX/output/*
1,1,3.0
1,0,4.0
0,1,21.0
0,0,28.0
1,1,18.0
1,0,45.0
0,1,12.0
0,0,30.0
[kct5thsemcdhid025@ip-10-1-1-204 matrix]$
```

LAB EXERCISE-5

Title of the exercise/experiment: Implement a Map Reduce Program to analyse time- temperature statistics and generate report with max/min temperature

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

- To implement a Map, Reduce Program to analyse time- temperature statistics and generate report with max/min temperature

PROBLEM STATEMENT:

1. The system receives temperatures of various cities(Austin, Boston,etc) of USA captured at regular intervals of time on each day in an input file.
2. System will process the input data file and generates a report with Maximum and Minimum temperatures of each day along with time.
3. Generates a separate output report for each city. Ex:

Austin-r-

ooooo

Boston-r-

ooooo

Newjersy-r-

ooooo

Baltimore-r-

ooooo

California-r-

ooooo

Newyork-r-

ooooo

THEORY:

The very first thing which is required for any map reduce problem is to

understand what will be the type of keyIn, ValueIn, KeyOut, ValueOut for the given Mapper class and followed by type of map method parameters.

- public class WhetherForcastMapper extends Mapper <Object, Text, Text, Text>
- Object (keyIn) - Offset for each line, line number 1, 2...
- Text (ValueIn) - Whole string for each line (CA_25-Jan-2014 00:12:345)
- Text (KeyOut) - City information with date information as string
- Text (ValueOut) - Temperature and time information which need to be passed to reducer as string.
- public void map(Object keyOffset, Text dayReport, Context con) {}
- KeyOffset is like line number for each line in input file.

- dayreport is input to map method - whole string present in one line of input file
- con is context where we write mapper output and it is used by reducer. Reducer class and reducer method:-

Similarly, we have to decide what will be the type of keyIn, ValueIn, KeyOut, ValueOut for the given Reducer class and followed by type of reducer method parameters.

- public class WhetherForcastReducer extends Reducer<Text, Text, Text, Text>
- Text(keyIn) - it is same as keyOut of Mapper.
- Text(ValueIn)- it is same as valueOut of Mapper.
- Text(KeyOut)- date as string
- text(ValueOut) - reducer writes max and min temperature with time as string
- public void reduce(Text key, Iterable<Text> values, Context context)

- Text key is value of mapper output. i.e:- City & date information
- Iterable<Text> values - values stores multiple temperature values for a given city and date. context object is where reducer write its processed outcome and finally written in file.

MultipleOutputs:-

In general, reducer generates output file(i.e: part_r_oooo), however in this use case we want to generate multiple output files. In order to deal with such scenario we need to use MultipleOutputs of "org.apache.hadoop.mapreduce.lib.output.MultipleOutputs" which provides a way to write multiple file depending on reducer outcome. For each reducer task multipleoutput object is created and key/result is written to appropriate file.

SOURCE CODE:

```

import
java.io.IOException;
import
java.util.StringTokenizer;
import
org.apache.hadoop.io.Tex
t;
import

```

```
org.apache.hadoop.mapreduce.Mapper;
import
org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.output.MultipleOutputs; import
org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; import
org.apache.hadoop.mapreduce.lib.output.TextOutputFormat; public class
CalculateMaxAndMinTemeratureWithTime {
public static String calOutputName
="California"; publicstatic String
nyOutputName = "Newyork"; public static
```

```
String njOutputName = "Newjersy";
public static String ausOutputName =
"Austin"; public static String
bosOutputName = "Boston";

public static String balOutputName
="Baltimore"; public static class
WhetherForcastMapper extends
Mapper<Object, Text, Text, Text> {
public void map(Object keyOffset, Text dayReport, Context
con) throws IOException, InterruptedException {
 StringTokenizer strTokens =
new StringTokenizer(
dayReport.toString(), "\t");
 int counter = 0;
 Float currnetTemp = null;
 Float minTemp = Float.MAX_VALUE;
 Float maxTemp = Float.MIN_VALUE;
 String date = null;
 String currentTime = null;
 String minTempANDTime = null; String
maxTempANDTime = null; while
(strTokens.hasMoreElements()) { if
(counter
== 0) {
date = strTokens.nextToken();
} else {
if (counter % 2 == 1) {
currentTime = strTokens.nextToken();
} else {
```

```
currnetTemp  
=Float.parseFloat(strTokens.nextToken());  
if(minTemp > currnetTemp) {  
    minTemp = currnetTemp;  
    minTempANDTime = minTemp +  
        "AND" + currentTime;
```

```
}

if (maxTemp <
currnetTemp) {

maxTemp =
currnetTemp;

maxTempANDTime =
maxTemp + "AND" +
currentTime;

}

}

}

counter++;

}

// Write to context - MinTemp,
MaxTemp and corresponding time

Text temp = new Text();
temp.set(maxTempANDTi
me); TextdateText = new
Text(); dateText.set(date);
try {
con.write(dateText, temp);
} catch
(Exception e) {
e.printStackTrace();
}
temp.set(minTempAN
DTime);
dateText.set(date);
con.write(dateText,
```

```
temp);  
}  
}  
  
public static class WhetherForcastReducer extends Reducer<Text, Text, Text,  
Text> {  
  
    MultipleOutputs<Text, Text>  
    mos; public void setup(Context  
    context) { mos = new  
    MultipleOutputs<Text,  
    Text>(context);  
}  
  
    public void reduce(Text key,  
    Iterable<Text> values, Context  
    context)  
    throws IOException, InterruptedException { int
```

```
counter = 0;
String
reducerInputStr[] =
null; String f1Time = "";
String f2Time = "";
String f1 = "", f2 = "";
Text result = new
Text(); for (Text value :
values)
{ if (counter == 0) {
reducerInputStr =
value.toString().split("AN
D"); f1 =
reducerInputStr[0];
f1Time =
reducerInputStr[1];
}
else {
reducerInput
Str =
value.toString().split("AN
D"); f2 =
reducerInputStr[0];
f2Time =
reducerInputStr[1];
}
counter = counter + 1;
}
if
(Float.parseFloat(f
1) >
Float.parseFloat(f2
```

```
)} {  
    result = new Text("Time: " + f2Time + " MinTemp: " + f2 + "\t"  
+ "Time: " + f1Time + "MaxTemp: " + f1);  
}  
else {  
  
    result = new Text("Time: " + f1Time + " MinTemp: " + f1 + "\t"+  
"Time: " + f2Time + "MaxTemp: " + f2);  
}  
String fileName = "";  
if (key.toString().substring(0, 2).equals("CA"))  
{  
    fileName =CalculateMaxAndMinTemeratureTime.calOutputName;  
} else if  
(key.toString().substring(0  
,2).equals("NY")) {  
    fileName =CalculateMaxAndMinTemeratureTime.nyOutputName;  
} else if (key.toString().substring(0,  
,
```

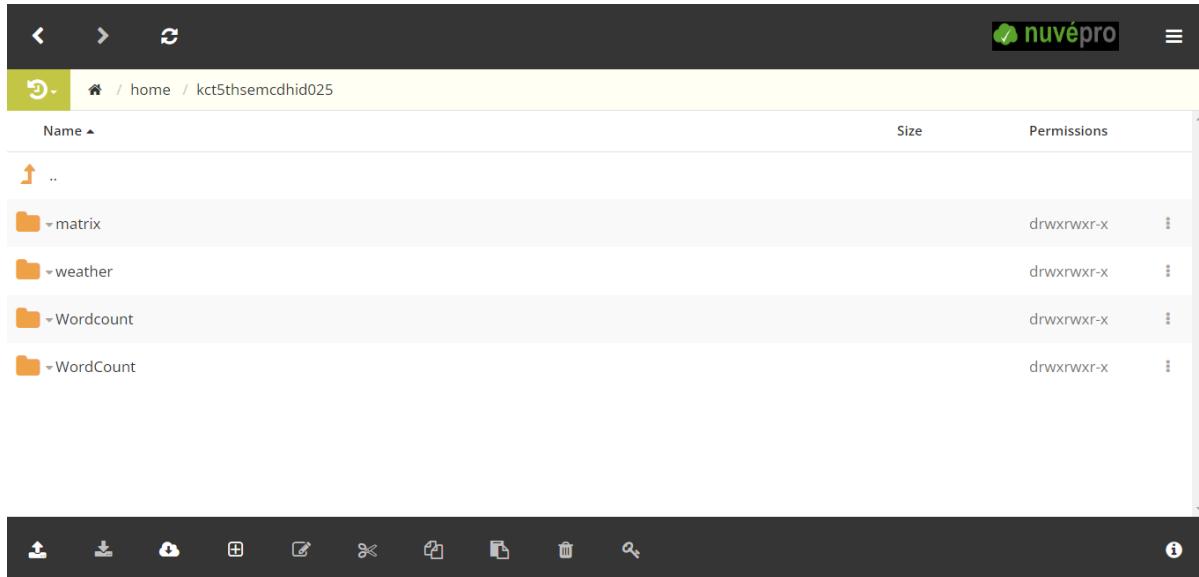
```
2).equals("NJ")) {  
    fileName =CalculateMaxAndMinTemeratureTime.njOutputName;  
}  
} else if  
(key.toString().substring(0  
,3).equals("AUS")) {  
    fileName =CalculateMaxAndMinTemeratureTime.ausOutputName;  
}  
} else if  
(key.toString().substring(0  
,3).equals("BOS")) {  
    fileName =CalculateMaxAndMinTemeratureTime.bosOutputName;  
  
}  
} else if  
(key.toString().substring(0  
,3).equals("BAL")) {  
    fileName =CalculateMaxAndMinTemeratureTime.balOutputName;  
}  
String strArr[] =  
key.toString().split("_");  
key.set(strArr[1]); //Key is  
date value  
mos.write(fileName, key,  
result);  
}  
@Override  
public void cleanup(Context  
context) throws IOException,  
InterruptedException  
{  
    mos.close();  
}
```

```
public static void main(String[]  
args) throws IOException,  
ClassNotFoundException,  
InterruptedException  
{  
    Configuration conf = new  
    Configuration(); Job job =  
    Job.getInstance(conf, "Wheather  
    Statistics of USA");  
    job.setJarByClass(CalculateMaxAndMin  
    TemeratureWithTime.class);  
    job.setMapperClass(WhetherForcastMa  
    pper.class);  
    job.setReducerClass(WhetherForcastRe  
    ducer.class);  
    job.setMapOutputKeyClass(Text.class);
```

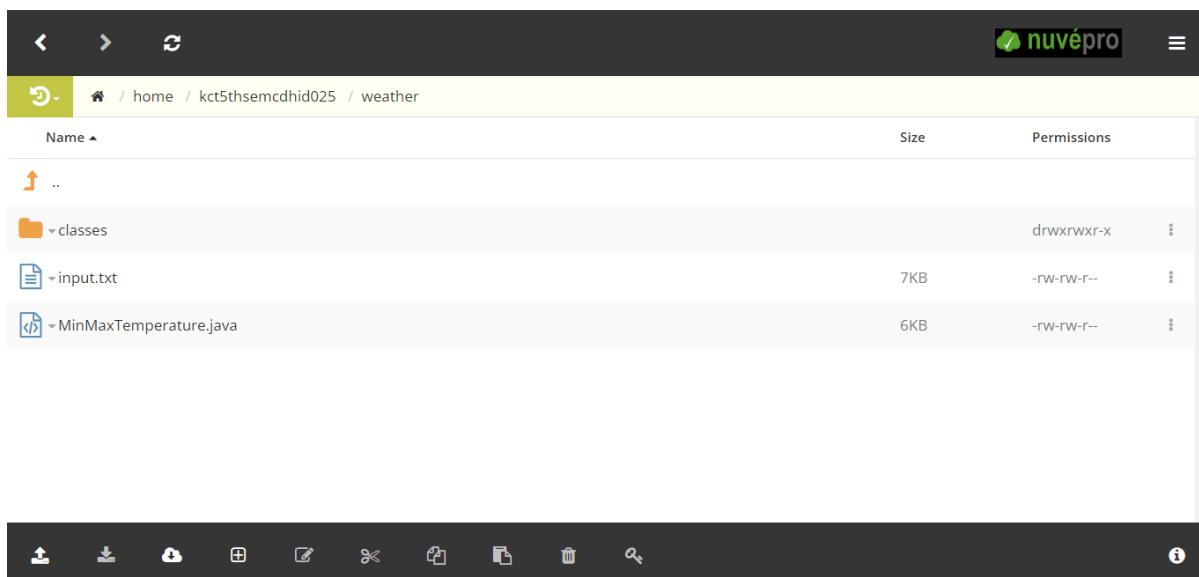
```
job.setMapOutputValueClass(  
    Text.class);  
job.setOutputKeyClass(Text.cl  
ass);  
job.setOutputValueClass(Text.class);  
MultipleOutputs.addNamedOutput(job,calOutputName,TextOutputFormat.class  
, Text.class,Text.class);  
MultipleOutputs.addNamedOutput(job,nyOutputName,TextOutputFormat.cl  
ass, Text.class,Text.class);  
MultipleOutputs.addNamedOutput(job,njOutputName,TextOutputFormat.cl  
ass, Text.class,Text.class);  
  
MultipleOutputs.addNamedOutput(job,bosOutputName,TextOutput  
Format.clas s,Text.class,Text.class);  
MultipleOutputs.addNamedOutput(job,ausOutputName,TextOutput  
Format.clas s,Text.class,Text.class);  
MultipleOutputs.addNamedOutput(job,balOutputName,TextOutputFormat.class  
, Text.class,Text.class);  
// FileInputFormat.addInputPath(job, new Path(args[0]));  
// FileOutputFormat.setOutputPath(job, new Path(args[1]));  
Path pathInput = new  
Path("hdfs://192.168.213.133:54310/weatherInputData/  
input_temp.txt");  
Path pathOutputDir = new Path(  
"hdfs://192.168.213.133:54310/user/hduser1/  
testfs/output_mapred3");  
FileInputFormat.addInputPath(job,  
pathInput);  
FileOutputFormat.setOutputPath(job,pathO  
utputDir); try {  
System.exit(job.waitForCompletion(true) ?
```

```
o :1);  
} catch (Exception e) {  
    // TODO Auto-generated  
    catch block  
    e.printStackTrace();  
  
}  
}  
}
```

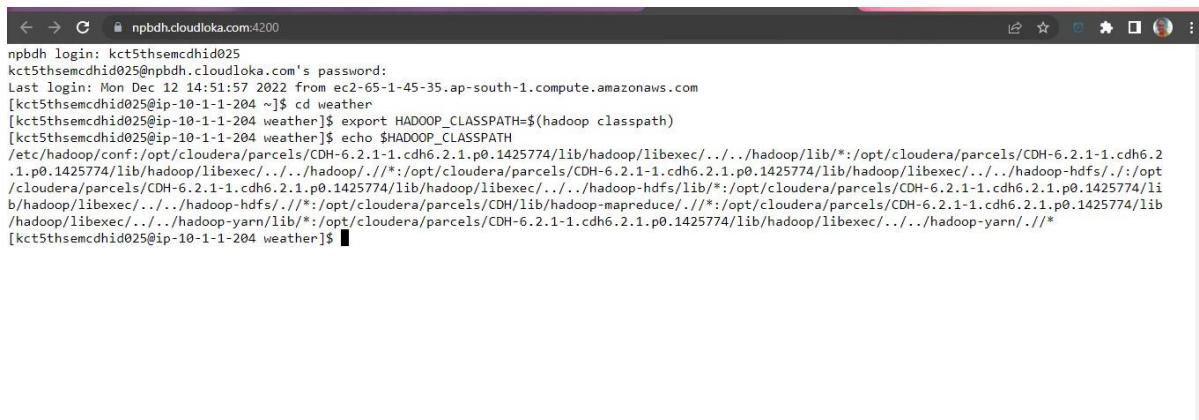
Create a folder weather



Name	Size	Permissions
..		drwxrwxr-x
matrix		drwxrwxr-x
weather		drwxrwxr-x
Wordcount		drwxrwxr-x
WordCount		drwxrwxr-x

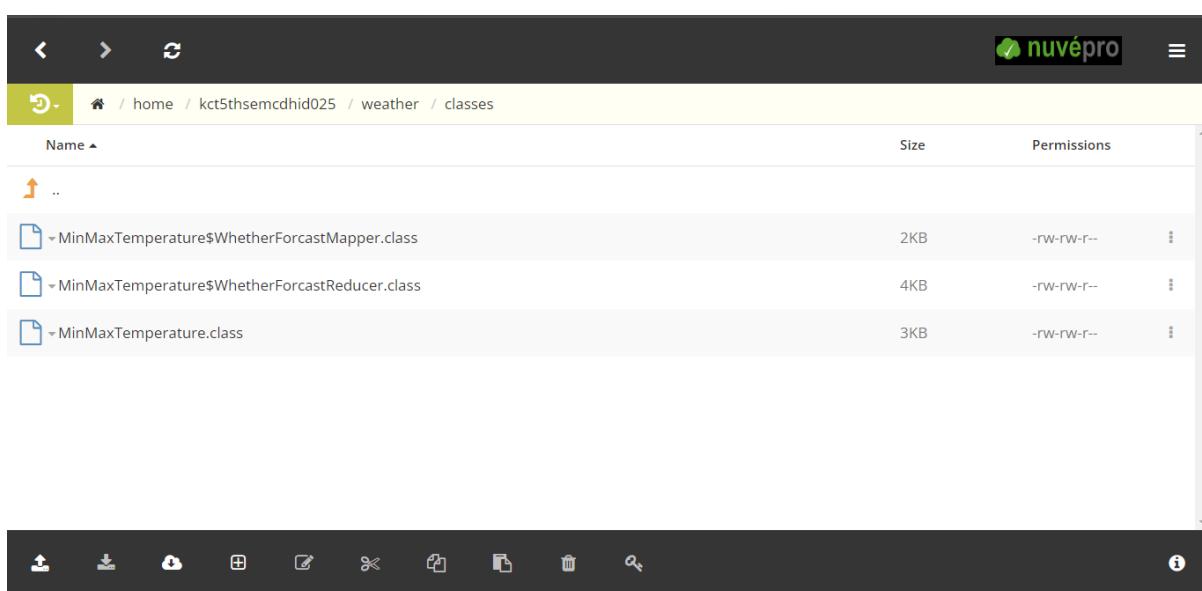


Name	Size	Permissions
..		drwxrwxr-x
classes		drwxrwxr-x
input.txt	7KB	-rw-rw-r--
MinMaxTemperature.java	6KB	-rw-rw-r--

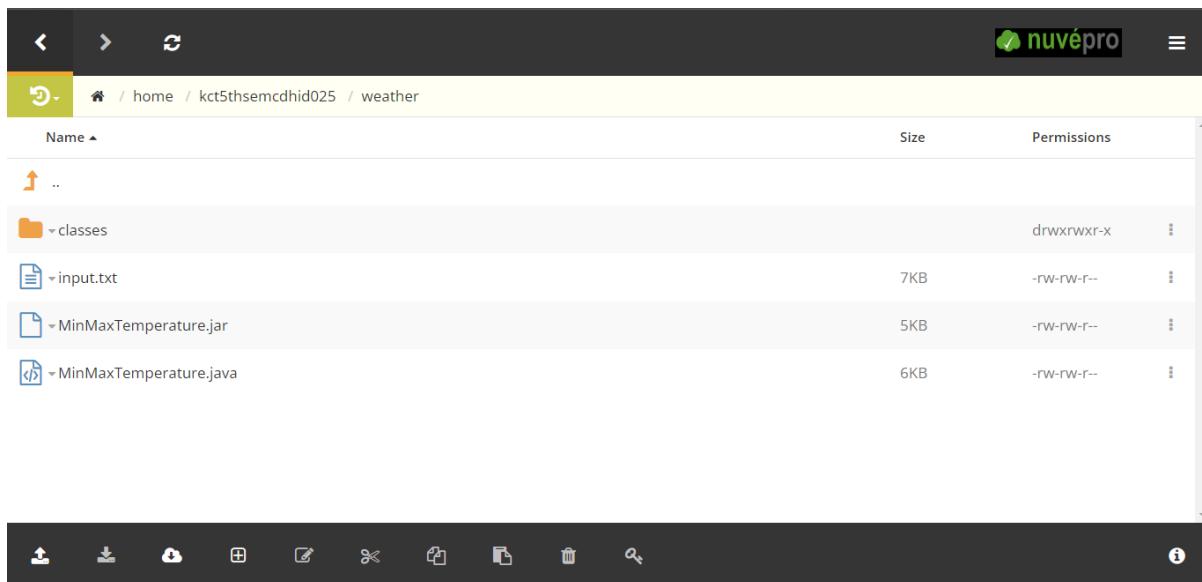


```
npbdh login: kct5thsemcdhid025
kct5thsemcdhid025@npbdh.cloudloka.com's password:
Last login: Mon Dec 12 14:51:57 2022 from ec2-65-1-45-35.ap-south-1.compute.amazonaws.com
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ cd weather
[kct5thsemcdhid025@ip-10-1-1-204 weather]$ export HADOOP_CLASSPATH=$(hadoop classpath)
[kct5thsemcdhid025@ip-10-1-1-204 weather]$ echo $HADOOP_CLASSPATH
/etc/hadoop/conf:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../hadoop/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/./:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-yarn/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-yarn/../*
[kct5thsemcdhid025@ip-10-1-1-204 weather]$
```

```
[kct5thsemcdhid025@ip-10-1-1-204 weather]$ Note: /home/kct5thsemcdhid025/weather/MinMaxTemperature.java uses or overrides a deprecated API.hsemcdhid0  
Note: Recompile with -Xlint:deprecation for details.  
[kct5thsemcdhid025@ip-10-1-1-204 weather]$  
[kct5thsemcdhid025@ip-10-1-1-204 weather]$  
[kct5thsemcdhid025@ip-10-1-1-204 weather]$
```



```
[kct5thsemcdhid025@ip-10-1-1-204 weather]$ jar -cvf MinMaxTemperature.jar -C '/home/kct5thsemcdhid025/weather/classes' .
added manifest
adding: MinMaxTemperature$WhetherForcastReducer.class(in = 3651) (out= 1546)(deflated 57%)
adding: MinMaxTemperature$WhetherForcastMapper.class(in = 2437) (out= 1143)(deflated 53%)
adding: MinMaxTemperature.class(in = 2734) (out= 1349)(deflated 50%)
[kct5thsemcdhid025@ip-10-1-1-204 weather]$
```



```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -ls /user/kct5thsemcdhid025/Weather
Found 2 items
drwxr-xr-x  - kct5thsemcdhid025 kct5thsemcdhid025          0 2022-12-12 06:10 /user/kct5thsemcdhid025/Weather/input
drwxr-xr-x  - kct5thsemcdhid025 kct5thsemcdhid025          0 2022-12-12 06:10 /user/kct5thsemcdhid025/Weather/output
```

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -put '/home/kct5thsemcdhid025/weather/input.txt' '/user/kct5thsemcdhid025/Weather/input'
[kct5thsemcdhid025@ip-10-1-1-204 ~]$
```

```
[kct5thsemdhid025@ip-10-1-1-204 weather]$ hadoop jar /home/kct5thsemdhid025/weather/MinMaxTemperature.jar MinMaxTemperature /user/kct5thsemdhid025/Weather/input /user/kct5thsemdhid025/Weather/output
WARNING: Use "yarn jar" to launch YARN applications.
22/12/12 15:36:48 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/12 15:36:49 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
22/12/12 15:36:49 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/kct5thsemdhid025/.staging/job_1663041244711_21350
22/12/12 15:36:49 INFO input.FileInputFormat: Total input files to process : 1
22/12/12 15:36:49 INFO mapreduce.JobSubmitter: number of splits:1
22/12/12 15:36:49 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
22/12/12 15:36:49 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1663041244711_21350
22/12/12 15:36:49 INFO mapreduce.JobSubmitter: Executing with tokens: []
22/12/12 15:36:49 INFO conf.Configuration: resource-types.xml not found
22/12/12 15:36:49 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
22/12/12 15:36:49 INFO impl.YarnClientImpl: Submitted application application_1663041244711_21350
22/12/12 15:36:49 INFO mapreduce.Job: The url to track the job: http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21350/
22/12/12 15:36:49 INFO mapreduce.Job: Running job: job_1663041244711_21350
```

```
21350/
22/12/12 15:36:49 INFO mapreduce.Job: Running job: job_1663041244711_21350
22/12/12 15:36:58 INFO mapreduce.Job: Job job_1663041244711_21350 running in uber mode : false
22/12/12 15:36:58 INFO mapreduce.Job: map 0% reduce 0%
22/12/12 15:37:05 INFO mapreduce.Job: map 100% reduce 0%
22/12/12 15:37:28 INFO mapreduce.Job: map 100% reduce 40%
22/12/12 15:37:29 INFO mapreduce.Job: map 100% reduce 80%
22/12/12 15:37:30 INFO mapreduce.Job: map 100% reduce 100%
22/12/12 15:37:30 INFO mapreduce.Job: Job job_1663041244711_21350 completed successfully
22/12/12 15:37:30 INFO mapreduce.Job: Counters: 54
```

File System Counters

```
FILE: Number of bytes read=906
FILE: Number of bytes written=1364303
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=7393
HDFS: Number of bytes written=1752
HDFS: Number of read operations=40
HDFS: Number of large read operations=0
HDFS: Number of write operations=34
HDFS: Number of bytes read erasure-coded=0
```

Job Counters

```
Launched map tasks=1
Launched reduce tasks=5
Data-local map tasks=1
Total time spent by all maps in occupied slots (ms)=4156
Total time spent by all reduces in occupied slots (ms)=108370
Total time spent by all map tasks (ms)=4156
Total time spent by all reduce tasks (ms)=108370
Total vcore-milliseconds taken by all map tasks=4156
Total vcore-milliseconds taken by all reduce tasks=108370
Total megabyte-milliseconds taken by all map tasks=4255744
Total megabyte-milliseconds taken by all reduce tasks=110970880
```

Map-Reduce Framework

```
Map output bytes=1584
Map output materialized bytes=886
Input split bytes=131
Combine input records=0
Combine output records=0
Reduce input groups=24
Reduce shuffle bytes=886
Reduce input records=48
Reduce output records=0
Spilled Records=96
Shuffled Maps =5
Failed Shuffles=0
Merged Map outputs=5
GC time elapsed (ms)=2036
CPU time spent (ms)=7400
Physical memory (bytes) snapshot=1538908160
Virtual memory (bytes) snapshot=15550607360
Total committed heap usage (bytes)=1847590912
Peak Map Physical memory (bytes)=527560704
Peak Map Virtual memory (bytes)=2589716480
Peak Reduce Physical memory (bytes)=244539392
Peak Reduce Virtual memory (bytes)=2592980992
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=7262
File Output Format Counters
  Bytes Written=0
[kct5thsemdhid025@ip-10-1-1-204 weather]$
```

OUTPUT:

```
[kct5thsemdhid025@ip-10-1-1-204 weather]$ hdfs dfs -cat /user/kct5thsemdhid025/Weather/output/*
25-Jan-2014 Time: 12:34:542 MinTemp: -22.3 Time: 05:12:345 MaxTemp: 35.7
26-Jan-2014 Time: 22:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 55.7
27-Jan-2014 Time: 02:34:542 MinTemp: -22.3 Time: 05:12:345 MaxTemp: 55.7
29-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 62.9
30-Jan-2014 Time: 22:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 49.2
31-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 03:12:187 MaxTemp: 56.0
29-Jan-2014 Time: 14:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 49.2
30-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 52.9
31-Jan-2014 Time: 15:12:345 MinTemp: -15.7 Time: 03:12:187 MaxTemp: 56.0
28-Jan-2014 Time: 11:19:345 MinTemp: -23.3 Time: 05:12:345 MaxTemp: 35.7
29-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 52.9
30-Jan-2014 Time: 15:12:345 MinTemp: -15.7 Time: 03:12:187 MaxTemp: 56.0
25-Jan-2014 Time: 12:34:542 MinTemp: -22.3 Time: 05:12:345 MaxTemp: 35.7
26-Jan-2014 Time: 04:00:093 MinTemp: -14.0 Time: 05:12:345 MaxTemp: 55.7
27-Jan-2014 Time: 02:34:542 MinTemp: -22.3 Time: 00:14:045 MaxTemp: 35.7
28-Jan-2014 Time: 11:19:345 MinTemp: -23.3 Time: 05:12:345 MaxTemp: 35.7
29-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 52.9
30-Jan-2014 Time: 15:12:345 MinTemp: -15.7 Time: 03:12:187 MaxTemp: 56.0
31-Jan-2014 Time: 22:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 49.2
29-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 52.9
30-Jan-2014 Time: 15:12:345 MinTemp: -15.7 Time: 03:12:187 MaxTemp: 56.0
29-Jan-2014 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 52.9
30-Jan-2014 Time: 15:12:345 MinTemp: -15.7 Time: 03:12:187 MaxTemp: 56.0
31-Jan-2014 Time: 22:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 49.2
[kct5thsemdhid025@ip-10-1-1-204 weather]$ █
```

RESULT:

Thus the minimum and maximum temperature of city is found successfully by the use of Mapand Reduce tasks.

LAB EXERCISE-6

Title of the exercise/experiment: Perform Joining data with streaming in Hadoop using MapReduce

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

- ② To Perform a Joining data with streaming in Hadoop using Map Reduce

THEORY:

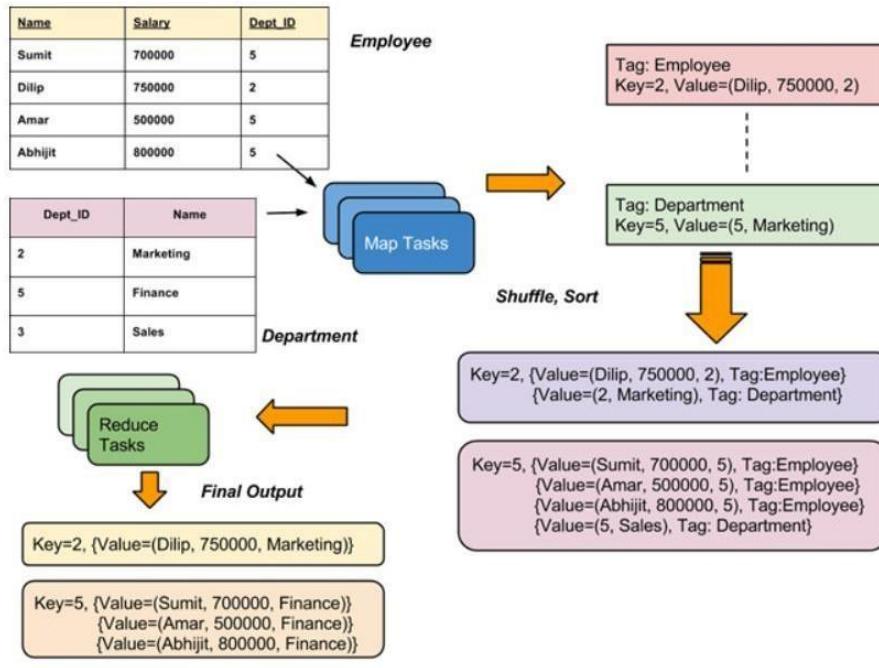
- ② Mapreduce Join operation is used to combine two large datasets. However, this process involves writing lots of code to perform the actual join operation. Joining two datasets begins by comparing the size of each dataset. If one dataset is smaller as compared to the other dataset then smaller dataset is distributed to every data node in the cluster.
- ② Once a join in MapReduce is distributed, either Mapper or Reducer uses the smaller dataset to perform a lookup for matching records from the large dataset and then combine those records to form output records

Types of Join

Depending upon the place where the actual join is performed, joins in Hadoop are classified into-

1. Map-side join – When the join is performed by the mapper, it is called as map-side join. In this type, the join is performed before data is actually consumed by the map function. It is mandatory that the input to each map is in the form of a partition and is in sorted order. Also, there must be an equal number of partitions and it must be sorted by the join key.
2. Reduce-side join – When the join is performed by the reducer, it is called as reduce-side join. There is no necessity in this join to have a dataset in a structured form (or partitioned).

Here, map side processing emits join key and corresponding tuples of both the tables. As an effect of this processing, all the tuples with same join key fall into the same reducer which then joins the records with same join key.



Types of Joins in Hadoop MapReduce

SOURCE CODE:

TextPair.java

```

package
MapReduceJoin;
import java.io.*;
import org.apache.hadoop.io.*;
public class TextPair implements
    WritableComparable<TextPair> { private Textfirst;
private Text
second;public
TextPair() {
    set(new Text(), new Text());
}

}

public TextPair(String first, String
second) { set(newText(first), new
Text(second));
}

```

```
public TextPair(Text first, Text  
second) { set(first,second);  
}  
  
public void set(Text first, Text second) { this.first  
= first;  
this.second = second;
```

```
}

public Text
    getFirst() {
    return first;
}

public Text
    getSecond() {
    return second;
}

@Override
public void write(DataOutput out) throws
    IOException {first.write(out);
second.write(out);}

@Override
public void readFields(DataInput in) throws
    IOException {first.readFields(in);
second.readFields(in);}

@Override
public int hashCode() {
    return first.hashCode() * 163 + second.hashCode();
}

@Override
public boolean
equals(Object o) { if (o
 instanceof TextPair) {
TextPair tp = (TextPair) o;

} }
```

```
        return first.equals(tp.first) && second.equals(tp.second);  
    }  
    return false;  
}
```

```
@Override  
public String toString()  
{ return first + "\t" +  
second;
```

```
}
```

```
@Override  
public int compareTo(TextPair tp) { int cmp  
    = first.compareTo(tp.first); if  
    (cmp != 0) {return cmp;  
}  
return second.compareTo(tp.second);  
}  
// ^^ TextPair  
  
// vv TextPairComparator  
public static class Comparator extends WritableComparator {  
  
    private static final Text.Comparator  
TEXT_COMPARATOR = newText.Comparator();  
  
    public  
        Comparator() {  
        super(TextPair.c  
lass);  
    }  
  
    @Override  
    public int compare(byte[] b1, int s1,  
                      int l1, byte[] b2, int s2, int  
l2) {  
  
        try {  
            int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) +  
readVInt(b1, s1); int firstL2 =  
WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);  
        }  
    }
```

```
int cmp = TEXT_COMPARATOR.compare(b1, s1,
firstL1, b2, s2, firstL2);if (cmp != 0) {
    return cmp;
}
return TEXT_COMPARATOR.compare(b1, s1 + firstL1, l1 - firstL1, b2, s2 +
firstL2, l2
    - firstL2);
} catch (IOException e) {
    throw new IllegalArgumentException(e);
}
}
```

```
}

static {
    WritableComparator.define(TextPair.class, new Comparator());
}
// ^^ TextPairComparator

// vv TextPairFirstComparator
public static class FirstComparator extends WritableComparator {

    private static final Text.Comparator
TEXT_COMPARATOR = newText.Comparator();

    public
        FirstComparator()
    {
        super(TextPair.cla
ss);
    }

    @Override
    public int compare(byte[] b1, int s1,
                      int l1, byte[] b2, int s2, int
l2) {
        try {
            int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) +
readVInt(b1, s1); int firstL2 =
WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);
            return TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2,
firstL2);
        } catch (IOException e) {
```

```
        throw new IllegalArgumentException(e);
    }
}

@Override
public int compare(WritableComparable a,
    WritableComparable b) { if (a instanceof TextPair &&
    b instanceof TextPair) {
    return ((TextPair) a).first.compareTo(((TextPair) b).first);
}
return super.compare(a, b);
```

```
    }
}

// ^^ TextPairFirstComparator
// vv TextPair
}
```

JOIN DRIVER.java:

```
package MapReduceJoin;
import
org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import
org.apache.hadoop.io.Text;
import
org.apache.hadoop.mapred.*;
import
org.apache.hadoop.mapred.lib.MultipleInputs
; import org.apache.hadoop.util.*;
public class JoinDriver extends Configured implements Tool {
    public static class KeyPartitioner implements Partitioner<TextPair, Text>
{
    @Override
    public void
    configure(JobConf job) {}
    @Override
    public int getPartition(TextPair key, Text value, int numPartitions)
{
    return (key.getFirst().hashCode() & Integer.MAX_VALUE)
% numPartitions;
}
```

```
}
```

```
@Override
```

```
public int run(String[] args) throws
```

```
Exception { if(args.length != 3)
```

```
{
```

```
    System.out.println("Usage: <Department Emp Strength input>
```

```
<Department Name input> <output>");
```

```
    return -1;
```

```
}
```

```
JobConf conf = new JobConf(getConf(), getClass());
```

```
conf.setJobName("Join 'Department Emp Strength
```

```
input' with
```

```
'Department Name input");

    Path    AInputPath    =    new
        Path(arg[0]); Path BInputPath =
    new      Path(arg[1]);      Path
    outputPath = new Path(args[2]);
    MultipleInputs.addInputPath(conf,
                                AInputP
ath,                      TextInputFormat.class,
DeptEmpStrengthMapper.class);

    MultipleInputs.addInputPath(conf,
                                BInputP
ath,TextInputFormat.class, DeptNameMapper.class);

    FileOutputFormat.setOutputPath(conf,      outputPath);
    conf.setPartitionerClass(KeyPartitioner.class);
    conf.setOutputValueGroupingComparator(TextPair.FirstComparator.c
las
s);

    conf.setMapOutputKeyClass(TextPair.class);
    conf.setReducerClass(JoinReducer.class);
    conf.setOutputKeyClass(Text.class);
```

```
        JobClient.runJob
        (conf);return o;
    }

    public static void main(String[] args) throws Exception {
        int exitCode = ToolRunner.run(new
        JoinDriver(), args);
        System.exit(exitCode);
    }

}
```

JoinReducer.java:

```
package MapReduceJoin;
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.Text;
import
org.apache.hadoop.mapred.*;
public class JoinReducer extends MapReduceBase implements
Reducer<TextPair,Text, Text, Text> {
    @Override
    public void reduce (TextPair key,
Iterator<Text> values,OutputCollector<Text, Text>
output, Reporter reporter)
        throws IOException
    {
        Text nodeId = new
Text(values.next()); while
(values.hasNext()) {
        Text node = values.next();
```

```

    Text outValue = new Text(nodeId.toString() + "\t\t"
+
node.toString()      output.collect(key.getFirst(), outValue);
);

}

}
}

```

INPUT FILES

Name	Size	Permissions
DeptName.txt	59B	-rw-rw-r--
DeptStrength.txt	50B	-rW-rW-r--
JoinDriver.java	2KB	-rW-rW-r--
JoinReducer.java	660B	-rw-rw-r--
TextPair.java	4KB	-rw-rw-r--

Steps to execute MapReduce word count example

1. Set up the path and configure the java environment:

COMMANDS
1. export HADOOP_CLASSPATH=\$(hadoop classpath) 2.echo \$HADOOP_CLASSPATH

```
[kct5thsencdhid025@ip-10-1-1-204 joindata]$ export HADOOP_CLASSPATH=$(hadoop classpath)
[kct5thsencdhid025@ip-10-1-1-204 joindata]$ echo $HADOOP_CLASSPATH
/etc/hadoop/conf:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/./:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/lib/*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-hdfs/.//*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-mapreduce/.//*:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-yarn/://:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-accessors-smart-1.2.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop-yarn/://:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/lib/activation-1.1.1.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/lib/audience-annotations-0.5.0.jar:/opt/cloud era/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/lib/avro.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/libexec/../../hadoop/awasaki-jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/azure-data-lake-store-sdk-2.2.9.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/libexec/../../hadoop/lib/commons-beanutils-1
```

1. Create an input directory in HDFS.

COMMAND:

hdfs dfs -mkdir / MapReduceJoin/

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcdhid025/MapReduceJoin
[kct5thsemcdhid025@ip-10-1-1-204 ~]$
```

Create a input folder inside of MapReduceJoin directory.

COMMANDS
hdfs dfs -mkdir / MapReduceJoin /input

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcdhid025/MapReduceJoin/input
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ █
```

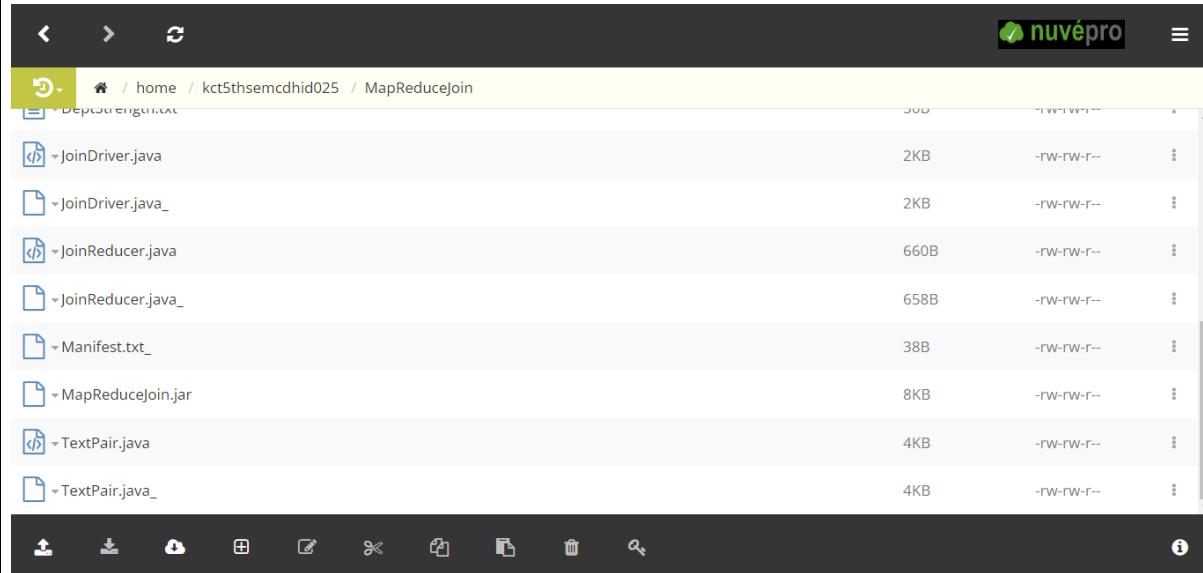
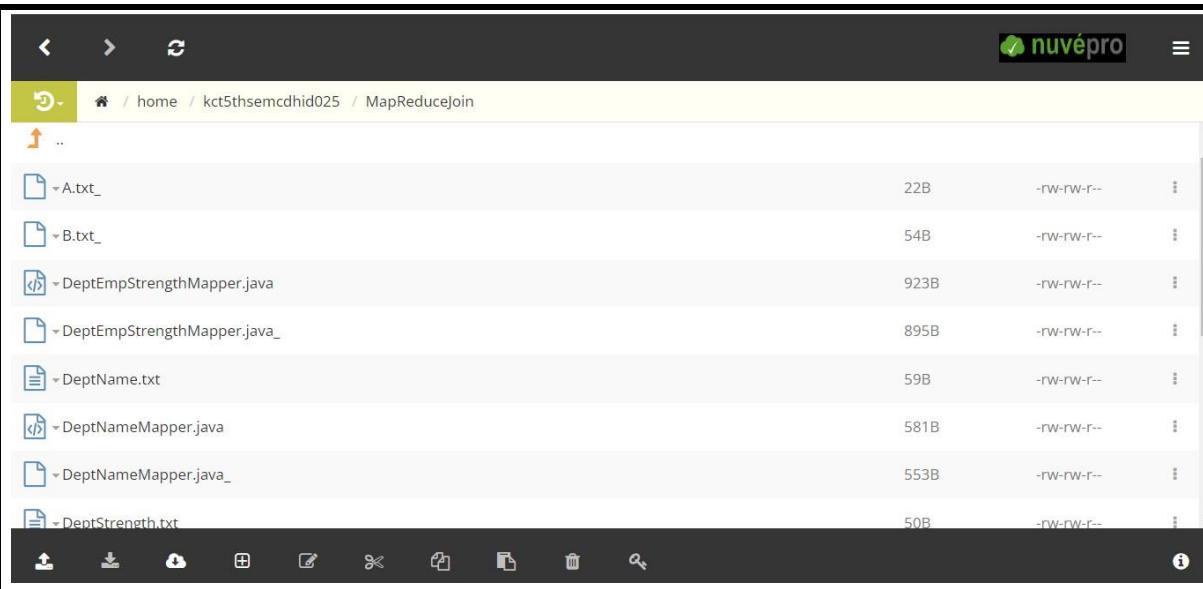
2. Copy the 2 input text file in the input directory (matrixmultiplication) of HDFS.

Copy some text file to hadoop filesystem inside input directory. Here I am copying a input.txt to it. You can copy more than one files.

COMMANDS
1. hdfs dfs -put '/home/gokilanm/MapReduceJoin/DeptName.txt' / MapReduceJoin /input/
2. hdfs dfs -put '/home/gokilanm/MapReduceJoin/DeptStrength.txt' / MapReduceJoin /input/

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -put '/home/kct5thsemcdhid025/joindata/DeptName.txt' '/user/kct5thsemcdhid025/MapReduceJoin/input'
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -put '/home/kct5thsemcdhid025/joindata/DeptStrength.txt' '/user/kct5thsemcdhid025/MapReduceJoin/input'
```

```
[kct5thsemdhid025@ip-10-1-1-204 ~]$ hdfs dfs -cat '/user/kct5thsemdhid025/MapReduceJoin/input/*'  
Dept_ID Dept_Name  
A11 Finance  
B12 HR  
C13 Manufacturing  
Dept_ID Total_Employee  
A11 50  
B12 100  
C13 250  
[kct5thsemdhid025@ip-10-1-1-204 ~]$
```



```
[kct5thsemcdhid025@ip-10-1-1-204 MapReduceJoin]$ ll
total 80
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 22 Dec 12 16:06 A.txt_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 54 Dec 12 16:05 B.txt_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 923 Dec 12 16:05 DeptEmpStrengthMapper.java
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 895 Dec 12 16:04 DeptEmpStrengthMapper.java_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 50 Dec 12 16:11 DeptEmpStrength.txt
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 581 Dec 12 16:04 DeptNameMapper.java
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 553 Dec 12 16:03 DeptNameMapper.java_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 59 Dec 12 15:49 DeptName.txt
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 50 Dec 12 15:49 DeptStrength.txt
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 1905 Dec 12 15:51 JoinDriver.java
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 1903 Dec 12 16:03 JoinDriver.java_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 660 Dec 12 15:52 JoinReducer.java
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 658 Dec 12 16:06 JoinReducer.java_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 38 Dec 12 16:13 Manifest (1).txt_
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 38 Dec 12 16:06 Manifest.txt_
drwxrwxr-x 2 kct5thsemcdhid025 kct5thsemcdhid025 4096 Dec 12 16:15 MapReduceJoin
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 7831 Dec 12 16:09 MapReduceJoin.jar
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 3678 Dec 12 15:53 TextPair.java
-rw-rw-r-- 1 kct5thsemcdhid025 kct5thsemcdhid025 3650 Dec 12 16:06 TextPair.java_
[kct5thsemcdhid025@ip-10-1-1-204 MapReduceJoin]$
```

STEP-9:

Run the program using below command-

COMMANDS

\$HADOOP_HOME/bin/hadoop jar

MapReduceJoin.jar

MapReduceJoin/JoinDriver/DeptStrength.txt

/DeptName.txt /output_mapreducejoin

at org.apache.hadoop.util.NativeCodeLoader.loadNative(NativeCodeLoader.java:24)

```
[kct5thsemcdhid025@ip-10-1-1-204 MapReduceJoin]$ hadoop jar MapReduceJoin.jar /user/kct5thsemcdhid025/MapReduceJoin/input/DeptStrength.txt /user/kct5thsemcdhid025/MapReduceJoin/input/DeptName.txt /user/kct5thsemcdhid025/MapReduceJoin/output_mapreducejoin
```

```
[kct5thsemcdhid025@ip-10-1-1-204 MapReduceJoin]$ hadoop jar MapReduceJoin.jar /user/kct5thsemcdhid025/MapReduceJoin/input/DeptStrength.txt /user/kct5thsemcdhid025/MapReduceJoin/input/DeptName.txt /user/kct5thsemcdhid025/MapReduceJoin/output_mapreducejoin
```

WARNING: Use "yarn jar" to launch YARN applications.

```
22/12/13 05:38:48 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 05:38:48 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 05:38:49 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/kct5thsemcdhid025/.staging/job_1663041244711_21565
22/12/13 05:38:49 INFO mapred.FileInputFormat: Total input files to process : 1
22/12/13 05:38:49 INFO mapred.FileInputFormat: Total input files to process : 1
22/12/13 05:38:49 INFO mapreduce.JobSubmitter: number of splits:4
22/12/13 05:38:49 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
22/12/13 05:38:49 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1663041244711_21565
22/12/13 05:38:49 INFO mapreduce.JobSubmitter: Executing with tokens: []
22/12/13 05:38:49 INFO conf.Configuration: resource-types.xml not found
22/12/13 05:38:49 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
22/12/13 05:38:50 INFO impl.YarnClientImpl: Submitted application application_1663041244711_21565
22/12/13 05:38:50 INFO mapreduce.Job: The url to track the job: http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21565/
22/12/13 05:38:50 INFO mapreduce.Job: Running job: job_1663041244711_21565
22/12/13 05:38:58 INFO mapreduce.Job: Job job_1663041244711_21565 running in uber mode : false
22/12/13 05:38:58 INFO mapreduce.Job: map 0% reduce 0%
22/12/13 05:39:13 INFO mapreduce.Job: map 25% reduce 0%
22/12/13 05:39:14 INFO mapreduce.Job: map 50% reduce 0%
22/12/13 05:39:15 INFO mapreduce.Job: map 100% reduce 0%
```

npbdh.cloudloka.com:4200

System Counters

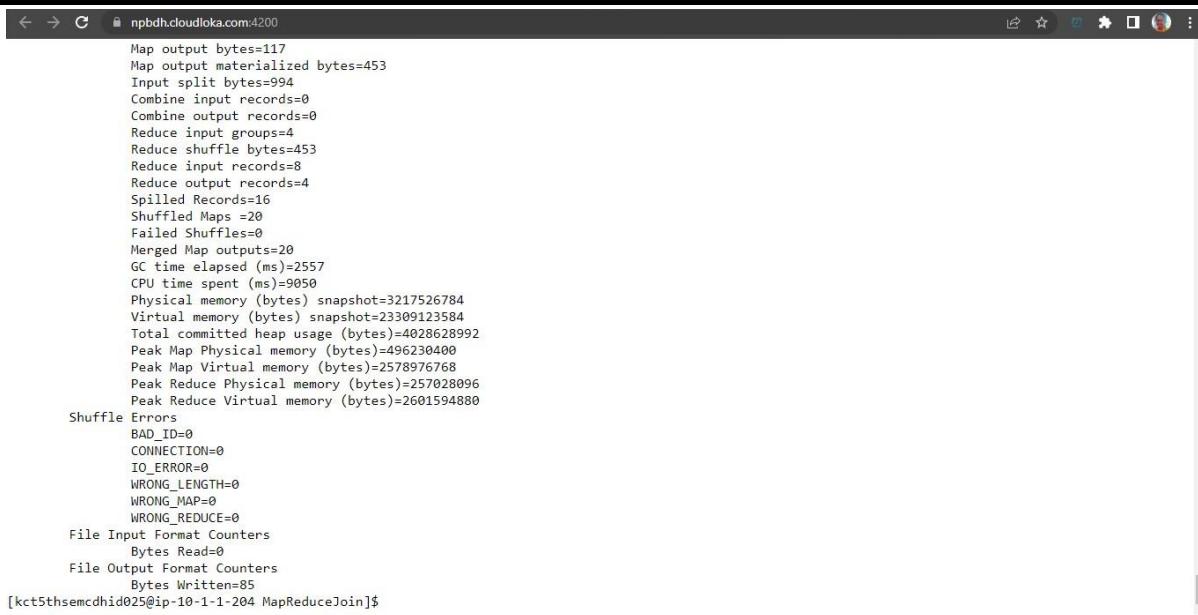
- FILE: Number of bytes read=224
- FILE: Number of bytes written=2015094
- FILE: Number of read operations=0
- FILE: Number of large read operations=0
- FILE: Number of write operations=0
- HDFS: Number of bytes read=1158
- HDFS: Number of bytes written=85
- HDFS: Number of read operations=37
- HDFS: Number of large read operations=0
- HDFS: Number of write operations=10
- HDFS: Number of bytes read erasure-coded=0

Job Counters

- Launched map tasks=4
- Launched reduce tasks=5
- Data-local map tasks=4
- Total time spent by all maps in occupied slots (ms)=56440
- Total time spent by all reduces in occupied slots (ms)=66136
- Total time spent by all map tasks (ms)=56440
- Total time spent by all reduce tasks (ms)=66136
- Total vcore-milliseconds taken by all map tasks=56440
- Total vcore-milliseconds taken by all reduce tasks=66136
- Total megabyte-milliseconds taken by all map tasks=57794560
- Total megabyte-milliseconds taken by all reduce tasks=67723264

Map-Reduce Framework

- Map input records=8
- Map output records=8
- Map output bytes=117
- Map output materialized bytes=453
- Input split bytes=994
- Combine input records=0
- Combine output records=0
- Reduce input groups=4
- Reduce shuffle bytes=453
- Reduce input records=8



The screenshot shows a web browser window with the URL `npbdh.cloudloka.com:4200`. The page displays detailed Hadoop MapReduce job statistics. Key metrics include:

- Map output bytes=117
- Map output materialized bytes=453
- Input split bytes=994
- Combine input records=0
- Combine output records=0
- Reduce input groups=4
- Reduce shuffle bytes=453
- Reduce input records=8
- Reduce output records=4
- Spilled Records=16
- Shuffled Maps =20
- Failed Shuffles=0
- Merged Map outputs=20
- GC time elapsed (ms)=9050
- CPU time spent (ms)=9050
- Physical memory (bytes) snapshot=3217526784
- Virtual memory (bytes) snapshot=23309123584
- Total committed heap usage (bytes)=4028628992
- Peak Map Physical memory (bytes)=496230400
- Peak Map Virtual memory (bytes)=2578976768
- Peak Reduce Physical memory (bytes)=257028096
- Peak Reduce Virtual memory (bytes)=2601594880

Shuffle Errors

- BAD_ID=0
- CONNECTION=0
- IO_ERROR=0
- WRONG_LENGTH=0
- WRONG_MAP=0
- WRONG_REDUCE=0

File Input Format Counters

- Bytes Read=0

File Output Format Counters

- Bytes Written=85

[kct5thsemcdhid025@ip-10-1-1-204 MapReduceJoin]\$

OUTPUT:

```
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -cat /user/kct5thsemcdhid025/MapReduceJoin/output_mapreducejoin/*
B12    100          HR
C13    250          Manufacturing
Dept_ID Total_Employee      Dept_Name
A11     50          Finance
[kct5thsemcdhid025@ip-10-1-1-204 ~]$
```

RESULT:

The join operation in the streaming data can be implemented successfully.

LAB EXERCISE-7

Title of the exercise/experiment: Collect, aggregate and transport large amount of streaming data from Twitter data using Apache Flume

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

To understand how Flume helps in streaming data from various sources.

THEORY:

Apache Flume is a tool for data ingestion in HDFS. It collects, aggregates and transports large amount of streaming data such as log files, events from various sources like network traffic, social media, email messages etc. to HDFS. Flume is a highly reliable & distributed.

The main

idea behind the Flume's design is to capture streaming data from various web servers to HDFS. It has simple and flexible architecture based on streaming data flows. It is fault-tolerant and provides reliability mechanism for Fault tolerance & failure recovery.

Apache Flume

Apache Flume is an open-source tool for collecting, aggregating, and moving huge amounts of streaming data from the external web servers to the central store, say HDFS, HBase, etc. It is a highly available and reliable service which has tunable recovery mechanisms. The main purpose of designing Apache Flume is to move streaming data generated by various applications to Hadoop Distributed FileSystem

Features of Apache Flume

- Apache Flume is a robust, fault-tolerant, and highly available service.
- It is a distributed system with tunable reliability mechanisms for fail-over and recovery.
- Apache Flume is horizontally scalable.
- Apache Flume supports complex data flows such as multi-hop flows, fan-in flows, fan-out flows. Contextual routing etc.
- Apache Flume provides support for large sets of sources, channels, and sinks.
- Apache Flume can efficiently ingest log data from various servers into a centralized repository.
- With Flume, we can collect data from different web servers in

real-time as well as in batch mode.

- We can import large volumes of data generated by social networking sites and e-commerce sites into Hadoop DFS using Apache Flume.

Flume Event

A Flume event is a basic unit of data that needs to be transferred from source to destination.

Flume Agent

Flume agent is an independent JVM process (JVM) in Apache Fl

Source

A source receives data from the data generators. It transfers the received data to one or more channels in the form of events. Flume provides support for several types of sources.

Example – Exec source, Thrift source, Avro source, twitter 1% source, etc. Channel

A channel receives the data or events from the flume source and buffers them till the sinks consume them. It is a transient store. Flume supports different types of channels.

Example – Memory channel, File system channel, JDBC channel, etc. Sink

A sink consumes data from the channel and stores them into the destination. The destination can be a centralized store or other flume agents.

Example – HDFS sink.

Additional Components of Flume Agent

There are few more components other than described above that play a significant role in transferring the events.

Interceptors

They alter or inspect flume events transferred between the flume source and channel.

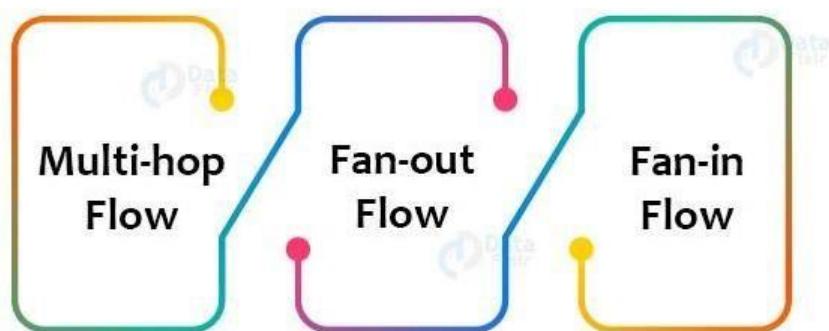
Channel Selectors

They determine which channel is to be chosen for transferring the data when multiple channels exist. Channel selectors are of two types- Default and multiplexing.

Sink Processors

Sink Processors invoke a particular sink from the

Flume Data Flow

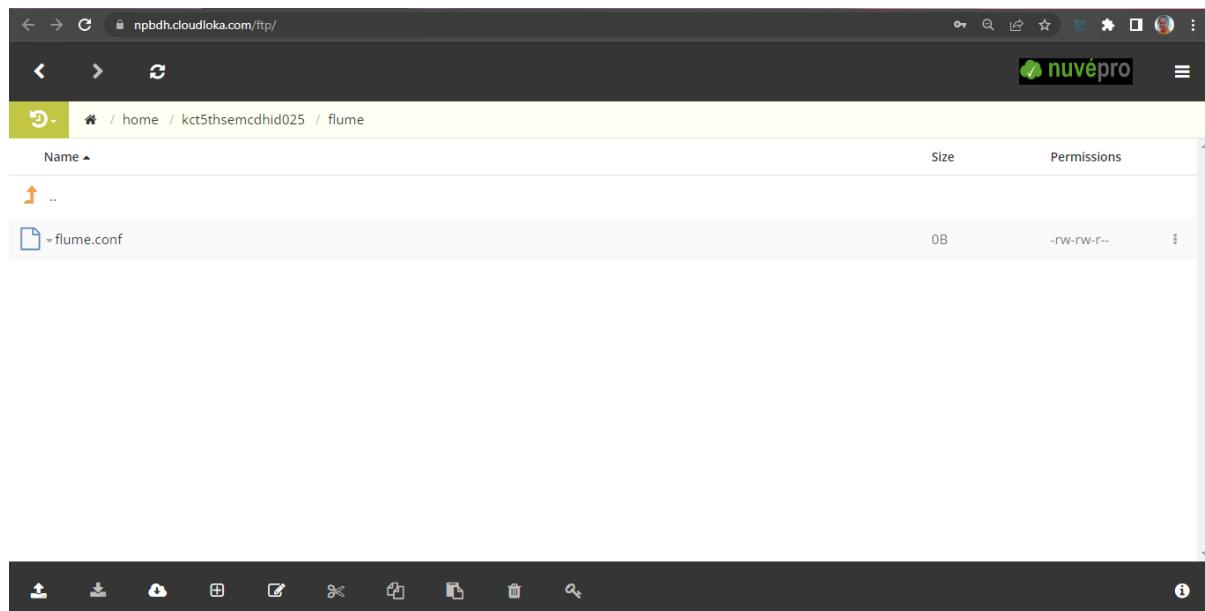


Data ingestion is the initial & important step in order to process & analyze data and then derive business values out of it. There are multiple sources from which data is gathered in an organization.

Collect, aggregate and transport large amount of streaming data from Twitter data using Apache Flume

Apache Flume: Streaming Twitter Data

The screenshot shows the Twitter Developer Apps interface. At the top, there's a purple header with navigation links like 'Developer', 'Use cases', 'Solutions', 'Products', 'Docs', 'Community', and 'Updates'. A message box says '#welcome We have sunset apps.twitter.com. You can manage any of your existing Apps in all of the same ways through this site.' Below the header, a button for 'Create an App' is visible. The main content area has a heading 'No Apps here.' followed by a sub-instruction: 'You'll need an App and API key in order to authenticate and integrate with most Twitter developer products. Create an App to get your API key.' A large central box is titled 'New Key & Secret'. Inside, a question 'Did you save your API Key and Key Secret?' is followed by a note: 'For security, this will be the last time we'll display these. If something happens, you can always regenerate them.' Below this, two sections show generated keys: 'API Key' with value 'nYn4obHurSfhma7EvvQzxd1Mf' and 'Copy' and 'Edit' buttons; and 'API Key Secret' with value 'VLCh23KLmQaywbxS7p4Xkf2y9y2tJ5uhTzQOutlq1IpHQLB Qc5' and similar 'Copy' and 'Edit' buttons. At the bottom, there are 'Cancel' and 'Yes, I saved them' buttons.



SOURCE CODE:

```
# Naming the components on the current
agent. TwitterAgent.sources = Twitter
TwitterAgent.channels = MemChannel
TwitterAgent.sinks = HDFS
TwitterAgent.sources.Twitter.channels =
MemChannel
TwitterAgent.sinks.HDFS.channel =
MemChannel
# Describing/Configuring the source
TwitterAgent.sources.Twitter.type=
org.apache.flume.source.twitter.TwitterSource
```

```
TwitterAgent.sources.Twitter.consumerKey= SN5f21STlXEV8ZC1wuL78Aftc
TwitterAgent.sources.Twitter.consumerSecret=
bounWv4jpXqjshIdUcKH1BL08lU4GvAV3ax2R
oRXRDNaf2uYNb
TwitterAgent.sources.Twitter.accessToken= 1352556616415989761-
aDJel5M1kFdxKMjyuJL2grEFONlkJt
TwitterAgent.sources.Twitter.accessToken
Secret=
Bx9J53CqeJbyKdWvnUh1t9OHov313Ia4X
gofTHoLf3WLP
TwitterAgent.sources.Twitter.keywords= tutorials
point,java, nosql # Describing/Configuring the sink
TwitterAgent.sinks.HDFS.type = hdfs
TwitterAgent.sinks.HDFS.hdfs.path =
hdfs://localhost:9000/twitter
TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream
TwitterAgent.sinks.HDFS.hdfs.writeFormat = Text
TwitterAgent.sinks.HDFS.hdfs.batchSize = 1000
TwitterAgent.sinks.HDFS.hdfs.rollSize = 0
```

TwitterAgent.sinks.HDES.hdfs.rollCount

= 10000

TwitterAgent.sinks.HDFS.hdfs.rollCount

= 600 # Describing/Configuring the
channel

TwitterAgent.channels.MemChannel.type

= memory

TwitterAgent.channels.MemChannel.capacity = 10000

TwitterAgent.channels.MemChannel.byteCapacity = 6912212

TwitterAgent.channels.MemChannel.transactionCapacity = 1000

```

1 TwitterAgent.sources = Twitter
2 TwitterAgent.channels = MemChannel
3 TwitterAgent.sinks = HDFS
4
5 TwitterAgent.sources.Twitter.type = com.cloudera.flume.source.TwitterSource
6 TwitterAgent.sources.Twitter.channels = MemChannel
7 TwitterAgent.sources.Twitter.consumerKey = 6YJdPBEpjmiyQnVRGob8s1F67
8 TwitterAgent.sources.Twitter.consumerSecret = j0c1PGzrulyfkjDnh9nh003VShgoNplwgeiW79tkuvj5GjXYTvG
9 TwitterAgent.sources.Twitter.accessToken = 1596724410051723265-1nEaS1ctFPk0j1qUsrg9o7eie2b
10 TwitterAgent.sources.Twitter.accessTokenSecret = Lg20w4C07UD05Udr16ujjhJKQdqmhnjE0BuItyICtks1
11 TwitterAgent.sources.Twitter.keywords = theinterview, 17YearsOfNash, Warnock, RioCompetition, cpfc, Palace, London, Christmas, New Years
12
13 ##### SINK #####
14 TwitterAgent.sinks.HDFS.channel = MemChannel
15 TwitterAgent.sinks.HDFS.type = hdfs
16 TwitterAgent.sinks.HDFS.hdfs.path = hdfs://user/abhinav9884/Tweets
17 TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream
18 TwitterAgent.sinks.HDFS.hdfs.writeFormat = Text
19
20 TwitterAgent.sinks.HDFS.hdfs.batchSize = 10
21 TwitterAgent.sinks.HDFS.hdfs.rollSize = 0
22 TwitterAgent.sinks.HDFS.hdfs.rollInterval = 600
23 TwitterAgent.sinks.HDFS.hdfs.rollCount = 10000
24
25 ##### CHANNEL #####
26 TwitterAgent.channels.MemChannel.type = memory
27 TwitterAgent.channels.MemChannel.capacity = 100
28 #default - TwitterAgent.channels.MemChannel.capacity = 100
29 TwitterAgent.channels.MemChannel.transactionCapacity = 100

```

▲ /home/kct5thsemcdhid025/flume/flume.conf

Auto-save Save Close

```

[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcdhid025/flume
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hdfs dfs -mkdir /user/kct5thsemcdhid025/flume/tweets
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ █

```

Browse through the Flume home directory and execute the application as shown below.

COMMANDS

```

flume-ng agent -n TwitterAgent ./conf/ -f
$FLUME_HOME/conf/twitter.conf

```

```
← → C npbdb.cloudoka.com:4200
due@parcels:~/cdh6.2.1-p0.1425774-110/flume-ng/..../search/lib/woodstox-core-asl-4.4.1.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/flume-ng/..../search/lib/woodstox-core-5.0.3.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/flume-ng/..../search/lib/zookeeper.jar:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/flume-ng/..../kudu/kudu-flume-sink.jar'-Djava.library.path=/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/bin/..../lib/hadoop/lib/native:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/flume-ng/..../lib/hadoop/lib/native:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/flume/conf
22/12/13 13:58:03 INFO node.PollingPropertiesFileConfigurationProvider: Configuration provider starting
22/12/13 13:58:03 INFO node.PollingPropertiesFileConfigurationProvider: Reloading configuration file:/home/kct5thsemcdhid025/flume/flume.conf
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:MemChannel
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:MemChannel
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:MemChannel
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Added sinks: HDFS Agent: TwitterAgent
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:Twitter
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Processing:HDFS
22/12/13 13:58:03 WARN conf.FlumeConfiguration: Agent configuration for 'TwitterAgent' has no configfilters.
22/12/13 13:58:03 INFO conf.FlumeConfiguration: Post-validation flume configuration contains configuration for agents: [TwitterAgent]
22/12/13 13:58:03 INFO node.AbstractConfigurationProvider: Creating channels
22/12/13 13:58:03 INFO channel.DefaultChannelFactory: Creating instance of channel MemChannel type memory
22/12/13 13:58:03 INFO node.AbstractConfigurationProvider: Created channel MemChannel
22/12/13 13:58:03 INFO source.DefaultSourceFactory: Creating instance of source Twitter, type org.apache.flume.source.twitter.TwitterSource
22/12/13 13:58:03 ERROR node.AbstractConfigurationProvider: Source Twitter has been removed due to an error during configuration
java.lang.InstantiationException: Incompatible source and channel settings defined. source's batch size is greater than the channels transaction capacity. Source: Twitter, batch size = 1000, channel MemChannel, transaction capacity = 100
at org.apache.flume.node.AbstractConfigurationProvider.checkSourceChannelCompatibility(AbstractConfigurationProvider.java:386)
at org.apache.flume.node.AbstractConfigurationProvider.getSourcesChannels(AbstractConfigurationProvider.java:367)
```

OUTPUT:

Show 25 entries

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	814.34 KB	Oct 18 22:07	1	128 MB	FlumeData.1634574999285	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	640.03 KB	Oct 18 22:07	1	128 MB	FlumeData.1634575037854	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	618.03 KB	Oct 18 22:08	1	128 MB	FlumeData.1634575068996	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	679.27 KB	Oct 18 22:08	1	128 MB	FlumeData.1634575099883	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	684.98 KB	Oct 18 22:09	1	128 MB	FlumeData.1634575130889	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	627.18 KB	Oct 18 22:09	1	128 MB	FlumeData.1634575161735	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	619.51 KB	Oct 18 22:10	1	128 MB	FlumeData.1634575191955	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	664.31 KB	Oct 18 22:10	1	128 MB	FlumeData.1634575222988	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	416.28 KB	Oct 18 22:11	1	128 MB	FlumeData.1634575254025	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	0 B	Oct 18 22:11	1	128 MB	FlumeData.1634575310070	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	0 B	Oct 18 22:12	1	128 MB	FlumeData.1634575335146	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	746.78 KB	Oct 18 22:13	1	128 MB	FlumeData.1634575366627	
<input type="checkbox"/>	-rw-r--r--	gokilanm	supergroup	0 B	Oct 18 22:13	1	128 MB	FlumeData.1634575401149.tmp	

Showing 1 to 13 of 13 entries

RESULT:

In short, Apache Flume is an open-source tool for collecting, aggregating, and moving huge amounts of data from the external web servers to the central store. Apache Flume is a highly available and reliable service. Apache Flume can be used for ingesting data from various applications to HDFS. It is useful for various e-commerce sites for understanding customer behavior. The Apache Flume Tutorial had explained the Flume architecture, data flow. It had also enlisted flume features, advantages, and disadvantages.

LAB EXERCISE-8

Title of the exercise/experiment: Perform simple join using Mapper in Spark

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

To Perform simple join using Mapper in Spark.

THEORY:

Spark DataFrame supports all basic SQL Join Types like INNER, LEFT OUTER, RIGHTOUTER, LEFT ANTI, LEFT SEMI, CROSS, SELF JOIN. Spark SQL Joins are wider transformations that result in data shuffling overthe network hence they have huge performance issues when not designed with care.

JOIN QUERY:

Queries can access multiple tables at once, or access the same table in such a way that multiple rows of the table are being processed at the same time. A query that accesses multiple rows of the same or different tables at one time is called a join query.

Parameters

Relation:Specifies the relation to
be joined.join_type:Specifies the
join type.

Syntax:

```
[ INNER ] | CROSS | LEFT [ OUTER ] | [ LEFT ] SEMI | RIGHT [ OUTER ] |
FULL [ OUTER ]
| [ LEFT ] ANTI
```

join_criteria :Specifies how the rows from one relation will be combined with the rows of another relation.

Syntax:

ON boolean_expression | USING (column_name [, ...])

boolean_expression: Specifies an expression with a return type of boolean.

join Operators

join(right: Dataset[_]): DataFrame (1)

join(right: Dataset[_], usingColumn: String):

DataFrame (2) join(right: Dataset[_],

usingColumns: Seq[String]): DataFrame (3)

join(right: Dataset[_], usingColumns: Seq[String],
joinType: String): DataFrame (4)

join(right: Dataset[_], joinExprs: Column): DataFrame (5)

join(right: Dataset[_], joinExprs: Column, joinType: String): DataFrame

(6) Condition-less inner join

Inner join with a single column that exists on both sides Inner join with columns that exist on both sides

Equi-join with explicit join type

Inner join

Join with explicit join type. Self-joins are acceptable.

Perform simple join using Mapper in Spark

1) MOVE INTO THE SPARK SCALA

2) EXECUTE THE COMMANDS IN SCALA:

1) INNER JOIN:

Syntax:

relation [INNER] JOIN relation [join_criteria]

COMMANDS

```
1. val left = Seq((0, "zero"), (1, "one")).toDF("id", "left")
2. val right = Seq((0, "zero"), (2, "two"), (3, "three")).toDF("id", "right")
3. left.join(right, "id").show
```

```
← → C npbdh.cloudloka.com:4200
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ spark shell
-bash: spark: command not found
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/12/13 05:47:25 WARN cluster.YarnSchedulerBackend$YarnSchedulerEndpoint: Attempted to request executors before the AM has registered!
Spark context available as 'sc' (master = yarn, app id = application_1663041244711_21570).
Spark session available as 'spark'.
Welcome to

    __/\__   \_\_\_/\_/\_/\_
   / \ \ \ - \ \ - / \ / \ / \
  / \ \ \ . \ / \ \ \ / \ / \ \
 / \ \ \_/\_ \ \ \_/\_ \ \ \_/\_ \
version 2.4.0-cdh6.2.1

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_181)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val left = Seq((0, "zero"), (1, "one")).toDF("id", "left")
left: org.apache.spark.sql.DataFrame = [id: int, left: string]

scala> val right = Seq((0, "zero"), (2, "two"), (3, "three")).toDF("id", "right")
right: org.apache.spark.sql.DataFrame = [id: int, right: string]

scala> left.join(right, "id").show
+---+---+---+
| id|left|right|
+---+---+---+
|  0|zero| zero|
+---+---+---+

scala>
```

```
< → C npbdh.cloudloka.com:4200
Welcome to
  / \ / \ - \ \ / \ / \ / \
 / \ / . \ / , \ / / \ \ \
 / \ / \ / \ / \ / \ / \
version 2.4.0-cdh6.2.1

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_181)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val left = Seq((0, "zero"), (1, "one")).toDF("id", "left")
left: org.apache.spark.sql.DataFrame = [id: int, left: string]

scala> val right = Seq((0, "zero"), (2, "two"), (3, "three")).toDF("id", "right")
right: org.apache.spark.sql.DataFrame = [id: int, right: string]

scala> left.join(right, "id").show
+---+---+---+
| id|left|right|
+---+---+---+
| 0 |zero| zero|
+---+---+---+

scala> left.join(right, "id").explain
== Physical Plan ==
*(1) Project [id#5, left#6, right#15]
+- *(1) BroadcastHashJoin [id#5], [id#14], Inner, BuildLeft
   :- BroadcastExchange HashedPartitionBroadcastMode(List(cast(input[0, int, false] as bigint)))
   :  +- LocalTableScan [id#5, left#6]
   +- LocalTableScan [id#14, right#15]

scala>
```

2) FULL OUTER JOIN:

In SQL the FULL OUTER JOIN combines the results of both left and right outerjoins and returns all (matched or unmatched) rows from the tables on both sides of the join clause.

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

COMMANDS

```
left.join(right, Seq("id"), "fullouter").show
```

COMMANDS

```
left.join(right, Seq("id"), "fullouter").explain
```

```

scala> left.join(right, "id").explain
== Physical Plan ==
*(1) Project [id#5, left#6, right#15]
+- *(1) BroadcastHashJoin [id#5], [id#14], Inner, BuildLeft
  :- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))
  :  +- LocalTableScan [id#5, left#6]
  :  +- LocalTableScan [id#14, right#15]

scala> left.join(right, Seq("id"), "fullouter").show
+---+---+
| id|left|right|
+---+---+
| 1| one| null|
| 3|null|three|
| 2|null| two|
| 0|zero| zero|
+---+---+

scala> left.join(right, Seq("id"), "fullouter").explain
== Physical Plan ==
*(3) Project [coalesce(id#5, id#14) AS id#54, left#6, right#15]
+- SortMergeJoin [id#5, [id#14], FullOuter
  :- *(1) Sort [id#5 ASC NULLS FIRST], false, 0
    :  +- Exchange hashpartitioning(id#5, 200)
    :  +- LocalTableScan [id#5, left#6]
  +- *(2) Sort [id#14 ASC NULLS FIRST], false, 0
    :  +- Exchange hashpartitioning(id#14, 200)
    :  +- LocalTableScan [id#14, right#15]

scala>

```

3) LEFT ANTI-JOIN

A left anti join returns that all rows from the first table which do not have a match in the second table.

COMM
AN
DS

```
left.join(right, Seq("id"), "leftanti").show
```

COMM
AN
DS

```
left.join(right, Seq("id"), "leftanti").explain
```

```
scala> left.join(right, Seq("id"), "leftanti").show
+---+-----+
| id|left|
+---+-----+
|  1| one|
+---+-----+
```

scala>

```

< > C npbdh.cloudloka.com:4200
at io.netty.buffer.PooledUnsafeDirectByteBuf.setBytes(PooledUnsafeDirectByteBuf.java:288)
at io.netty.buffer.AbstractByteBuf.writeBytes(AbstractByteBuf.java:1106)
at io.netty.channel.socket.nio.NioSocketChannel.doReadBytes(NioSocketChannel.java:343)
at io.netty.channel.nio.AbstractNioByteChannel$NioByteUnsafe.read(AbstractNioByteChannel.java:123)
at io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:645)
at io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.java:580)
at io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:497)
at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:459)
at io.netty.util.concurrent.SingleThreadEventExecutor$5.run(SingleThreadEventExecutor.java:858)
at io.netty.util.concurrent.DefaultThreadFactory$DefaultRunnableDecorator.run(DefaultThreadFactory.java:138)
at java.lang.Thread.run(Thread.java:748)

+---+---+
| id|left|
+---+---+
| 1| one|
+---+---+

scala> left.join(right, Seq("id"), "leftanti").show
+---+---+
| id|left|
+---+---+
| 1| one|
+---+---+

scala> left.join(right, Seq("id"), "leftanti").explain
== Physical Plan ==
*(1) BroadcastHashJoin [id#5], [id#14], LeftAnti, BuildRight
:- LocalTableScan [id#5, left#6]
+- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))
+- LocalTableScan [id#14]

scala>

```

4) BROADCAST JOIN

Spark SQL uses broadcast join (aka broadcast hash join) instead of hash join to optimize join queries when the size of one side data is below spark.sql.autoBroadcastJoinThreshold.

Broadcast join can be very efficient for joins between a large table (fact) with relatively small tables (dimensions) that could then be used to perform a star-schema join. It can avoid sending all data of the large table over the network.

You can use broadcast function or SQL's broadcast hints to mark a dataset to be broadcast when used in a join query.

broadcast join is also called a replicated join (in the distributedsystem community) or a map-side join (in the Hadoop community).

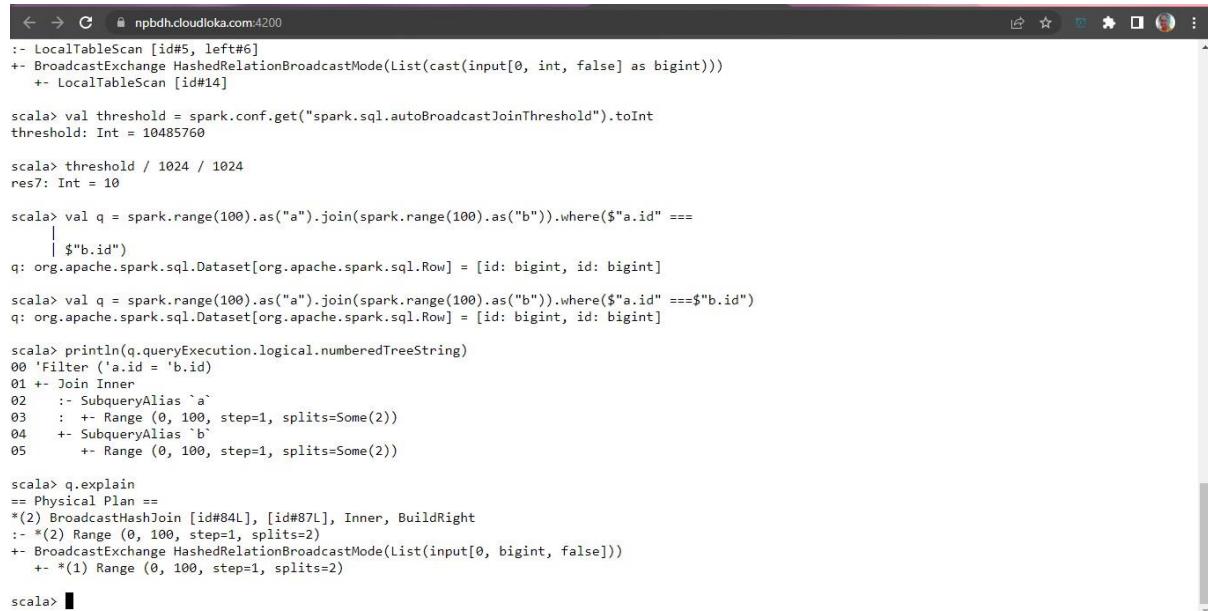
JoinSelection execution planning strategy uses spark.sql.autoBroadcastJoinThreshold property (default: 10M) to control the size of a dataset before broadcasting it to all worker nodes when performing a join.

COMMANDS

```

1. val threshold =
spark.conf.get("spark.sql.autoBroadcastJoinThreshold").toInt 2.threshold /
1024 / 1024
3.val q =
spark.range(100).as("a").join(spark.range(100).as("b")).where($"a.id" ===
$"b.id") 4.println(q.queryExecution.logical.numberedTreeString)
5. q.explain
6. spark.conf.set("spark.sql.autoBroadcastJoin
Threshold", -1)
7.spark.conf.get("spark.sql.autoBroadcastJoin
Threshold")

```



```

← → C npbdh.cloudloka.com:4200
:- LocalTableScan [id#5, left#6]
+- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))
  +- LocalTableScan [id#14]

scala> val threshold = spark.conf.get("spark.sql.autoBroadcastJoinThreshold").toInt
threshold: Int = 10485760

scala> threshold / 1024 / 1024
res7: Int = 10

scala> val q = spark.range(100).as("a").join(spark.range(100).as("b")).where($"a.id" ===
| $"b.id")
q: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> val q = spark.range(100).as("a").join(spark.range(100).as("b")).where($"a.id" ===$"b.id")
q: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> println(q.queryExecution.logical.numberedTreeString)
00 'Filter ('.id = 'b.id)
01 +- Join Inner
02   :- SubqueryAlias `a`
03   :  +- Range (0, 100, step=1, splits=Some(2))
04   +- SubqueryAlias `b`
05     +- Range (0, 100, step=1, splits=Some(2))

scala> q.explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#84L], [id#87L], Inner, BuildRight
:- *(2) Range (0, 100, step=1, splits=2)
+- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
  +- *(1) Range (0, 100, step=1, splits=2)

scala> ■

```

```
← → C npbdh.cloudloka.com:4200
threshold: Int = 10485760

scala> threshold / 1024 / 1024
res7: Int = 10

scala> val q = spark.range(100).as("a").join(spark.range(100).as("b")).where($"a.id" ===
| $"b.id")
q: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> val q = spark.range(100).as("a").join(spark.range(100).as("b")).where($"a.id" === $"b.id")
q: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> println(q.queryExecution.logical.numberedTreeString)
00 'Filter ('a.id = 'b.id)
01 +- Join Inner
02   :- SubqueryAlias `a`
03   :  +- Range (0, 100, step=1, splits=Some(2))
04   +- SubqueryAlias `b`
05     +- Range (0, 100, step=1, splits=Some(2))

scala> q.explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#84L], [id#87L], Inner, BuildRight
:- *(2) Range (0, 100, step=1, splits=2)
+- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
  +- *(1) Range (0, 100, step=1, splits=2)

scala> ark.conf.set("spark.sql.autoBroadcastJoinThreshold", -1)

scala> spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
res11: String = -1

scala> █
```

```

scala> q.explain
== Physical Plan ==
*(5) SortMergeJoin [id#84L], [id#87L], Inner
:- *(2) Sort [id#84L ASC NULLS FIRST], false, 0
:  +- Exchange hashpartitioning(id#84L, 200)
:     +- *(1) Range (0, 100, step=1, splits=2)
+- *(4) Sort [id#87L ASC NULLS FIRST], false, 0
    +- ReusedExchange [id#87L], Exchange hashpartitioning(id#84L, 200)

scala>
<console>:1: error: ';' expected but 'val' found.
val qBroadcast = spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where($"a.id"=="$b.id")val qBroadcast =
^

scala> val qBroadcast =spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where($"a.id"=="$b.id")
qBroadcast: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> qBroadcast.explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#94L], [id#97L], Inner, BuildRight
:- *(2) Range (0, 100, step=1, splits=2)
+- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
    +- *(1) Range (0, 100, step=1, splits=2)

scala>

```

COMMANDS

1. q.explain
2. val qBroadcast =
spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where(\$"a.id" == "\$b.id")
3. qBroadcast.ex
plain
4. val qBroadcastLeft =
"""" SELECT /*+
BROADCAST (lf) */ *
FROM range(100) lf, range(1000) rt
WHERE lf.id
= rt.id""""
5. scala> sql(qBroadcastLeft).explain

```

<- → C npbdh.cloudloka.com:4200
== Physical Plan ==
*(5) SortMergeJoin [id#84L], [id#87L], Inner
:- *(2) Sort [id#84L ASC NULLS FIRST], false, 0
:  +- Exchange hashpartitioning(id#84L, 200)
:    +- *(1) Range (0, 100, step=1, splits=2)
+- *(4) Sort [id#87L ASC NULLS FIRST], false, 0
  +- ReusedExchange [id#87L], Exchange hashpartitioning(id#84L, 200)

scala>
<console>:1: error: ';' expected but 'val' found.
val qBroadcast = spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where($"a.id"=="$b.id")val qBroadcast =
^

scala> val qBroadcast =spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where($"a.id"=="$b.id")
qBroadcast: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> qBroadcast.explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#94L], [id#97L], Inner, BuildRight
:- *(2) Range (0, 100, step=1, splits=2)
+- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
  +- *(1) Range (0, 100, step=1, splits=2)

scala> val qBroadcastLeft = """SELECT /*+ BROADCAST (lf) */ * FROM range(100) lf, range(1000) rt WHERE lf.id = rt.id"""
qBroadcastLeft: String = SELECT /*+ BROADCAST (lf) */ * FROM range(100) lf, range(1000) rt WHERE lf.id = rt.id

scala> sql(qBroadcastLeft).explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#109L], [id#110L], Inner, BuildLeft
:- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
:  +- *(1) Range (0, 100, step=1, splits=2)
+- *(2) Range (0, 1000, step=1, splits=2)

scala> ■

```

COMMANDS

1. val qBroadcastRight = """"
SELECT /*+ MAPJOIN (rt) */
*** FROM range(100) lf,**
range(1000) rt WHERE lf.id
= rt.id
""""

scala> sql(qBroadcastRight).explain

```

<- → C npbdh.cloudloka.com:4200
val qBroadcast = spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where($"a.id"=="$b.id")val qBroadcast =
^

scala> val qBroadcast =spark.range(100).as("a").join(broadcast(spark.range(100)).as("b")).where($"a.id"=="$b.id")
qBroadcast: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: bigint, id: bigint]

scala> qBroadcast.explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#94L], [id#97L], Inner, BuildRight
:- *(2) Range (0, 100, step=1, splits=2)
+- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
  +- *(1) Range (0, 100, step=1, splits=2)

scala> val qBroadcastLeft = """SELECT /*+ BROADCAST (lf) */ * FROM range(100) lf, range(1000) rt WHERE lf.id = rt.id"""
qBroadcastLeft: String = SELECT /*+ BROADCAST (lf) */ * FROM range(100) lf, range(1000) rt WHERE lf.id = rt.id

scala> sql(qBroadcastLeft).explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#109L], [id#110L], Inner, BuildLeft
:- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
:  +- *(1) Range (0, 100, step=1, splits=2)
+- *(2) Range (0, 1000, step=1, splits=2)

scala> val qBroadcastRight = """ SELECT /*+ MAPJOIN (rt) */ * FROM range(100) lf, range(1000) rt WHERE lf.id = rt.id """
qBroadcastRight: String = " SELECT /*+ MAPJOIN (rt) */ * FROM range(100) lf, range(1000) rt WHERE lf.id = rt.id "

scala> sql(qBroadcastRight).explain
== Physical Plan ==
*(2) BroadcastHashJoin [id#116L], [id#117L], Inner, BuildRight
:- *(2) Range (0, 100, step=1, splits=2)
+- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]))
  +- *(1) Range (0, 1000, step=1, splits=2)

scala> ■

```

RESU LT:

simple join using Mapper in Spark can be implemented.

LAB EXERCISE-9

Title of the exercise/experiment: Install and Run Hive then use Hive to create, alter, and drop databases, tables, views,functions, and indexes

a) OBJECTIVE OF THE EXERCISE/EXPERIMENT

Install and Run Hive then use Hive to create, alter, and drop databases, tables, views,functions, and indexes

HIVE-OPERATIONS

COMMANDS
cd \$HIVE_HOME/bin hive



```

< → C npbdh.cloudloka.com:4200
npbdh login: kct5thsemcdhid025
kct5thsemcdhid025@npbdh.cloudloka.com's password:
Last login: Tue Dec 13 05:46:50 2022 from ec2-65-1-45-35.ap-south-1.compute.amazonaws.com
[kct5thsemcdhid025@ip-10-1-1-204 ~]$ hive
WARNING: Use "yarn jar" to launch YARN applications.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/jars/log4j-slf4j-impl-2.8.2.jar!/org/slf4j/impl/StaticLoggerB
inder.class]
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/jars/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBin
der.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2022-12-13 06:11:28,766 main WARN JNDI lookup class is not available because this JRE does not support JNDI. JNDI string lookups will not be availabl
e, continuing configuration. Ignoring java.lang.ClassNotFoundException: org.apache.logging.log4j.core.lookup.JndiLookup
Logging initialized using configuration in jar:file:/opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/jars/hive-common-2.1.1-cdh6.2.1.jar!/hive-l
og4j2.properties Async: false
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive>

```

SHOW DATABASES:

1. SHOW A DATABASE:

At any time, you can see the databases that already exist as follows:

COMMANDS
Show databases;

```
← → C npbdh.cloudloka.com:4200
hotel
hr
hr_data
hr_db
hr_db1
hr_dbrajat
hr_hw_amit
hrdata
hsdb
hsproject
hulk_db
ian
ig
icg1
icg2
icmr
ig
ikhlaque
ikhu_demo1234
imapala_project
impala
impala12
impala123
impala_pok
impala_proj
impala_project
impalaproject
impalaprojectpokemon
imhealth
incedo
indium
indium1
ineuron_db
ineuron_db
i...e...o...7
```

```
← → C npbdh.cloudloka.com:4200
zlatan
zlatan1
zlikhith
zman
zpavan
zsachin
zsambit
zsambit_demo
zsarvesh
zsarvesh_demo
zsekhlar
zsonu
zsuhail
zsuresh
zsureshassignment
zswar
zswaroop
zuned_05oct
zuned_test
zvikram
zzabhinaw
zzassignment
zzassignmentt
zzkhush
zznaresh
zzsonu
zzsunairlines
zzsunretail
zzsunretail_
zzsunyoutube
zzzassignment
zzzzproject
Time taken: 1.653 seconds, Fetched: 3227 row(s)
hive>
```

- 1) Create database:
A database in Hive is a namespace or a collection of tables.

COMMANDS

```
Create database  
bigdata;
```

```
hive> create database samplebigdata;
OK
Time taken: 0.082 seconds
hive> █
```

2) LIST A DATABASE STARTS WITH A SPECIFIC LETTER

```
hive> show databases like 's.*';
OK
s18
s19
s1_demo_retail
sabdar_db
sachin
sachin1
sachin_db
sachinlabs
safdar
safdar_db
sahadev
sahil
sahildemo
sahildemo1
sahil_practice
sai
sai1
saikat10dt87edu
saikatt
saikrishna
saimanvitha
sairohith
sairohithm
sak_123
sakshidani
sakshidani1
sakshidaniimp
sal
salarydb_ext

hive> show databases like 'sampleb.*';
OK
samplebigdata
Time taken: 0.045 seconds, Fetched: 1 row(s)
hive> █
```

3. IF DATABASE ALREADY EXISTS:

```
hive> CREATE DATABASE IF NOT EXISTS lab;
OK
Time taken: 0.056 seconds
hive>
```

4. Add a creator name with date of database

```
hive> CREATE DATABASE mysamplebigdata WITH DBPROPERTIES('creator' = 'sankamethra', 'date'= '2022-12-13');
OK
Time taken: 0.061 seconds
hive>
```

5) DESCRIBE A DATABASE:

```
hive> DESCRIBE DATABASE mysamplebigdata;
OK
mysamplebigdata          hdfs://nameservice1/user/hive/warehouse/mysamplebigdata.db      kct5thsemcdhid025      USER
Time taken: 0.072 seconds, Fetched: 1 row(s)
hive>
```

6) DESCRIBE DATABASE EXTENDED

```
hive> DESCRIBE DATABASE EXTENDED mysamplebigdata;
OK
mysamplebigdata          hdfs://nameservice1/user/hive/warehouse/mysamplebigdata.db      kct5thsemcdhid025      USER {date=2022-12-13, creator=san
kamethra}
Time taken: 0.045 seconds, Fetched: 1 row(s)
hive>
```

7) USE COMMAND

```
hive> use mysamplebigdata;
OK
Time taken: 0.032 seconds
hive>
```

8) SHOW TABLES:

```
hive> show tables;  
OK  
Time taken: 0.09 seconds  
hive>
```

```

hive> create table stu_details(subject varchar(20), mark int, rollno int);
OK
Time taken: 0.255 seconds
hive> show tables;
OK
stu_details
Time taken: 0.063 seconds, Fetched: 1 row(s)
hive> █

```

9) DEFAULT DATABASE:

```

-----, -----
hive> set hive.cli.print.current.db=true;
hive (mysamplebigdata)> USE default;
OK
Time taken: 0.033 seconds
hive (default)> set hive.cli.print.current.db=false;
hive> █

```

10) DROP A DATABASE:

```

hive> drop table if exists stu_details;
OK
Time taken: 0.243 seconds
-----, -----

```

11)ALTER DATABASE:

```

hive> ALTER DATABASE mysamplebigdata SET DBPROPERTIES ('edited-by'= 'sankamethra');
OK
Time taken: 0.282 seconds
hive> █

```

12) CREATING TABLE:

```

hive> CREATE TABLE IF NOT EXISTS mysamplebigdata.employees (name STRING COMMENT 'Employee name', salary FLOAT COMMENT 'Employee salary', subordinates ARRAY<STRING> COMMENT 'Names of subordinates', deductions MAP<STRING, FLOAT>COMMENT 'Keys are deduction names, values are percentages', address STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>COMMENT 'Home address')COMMENT 'Description of the table'LOCATION '/user/hive/warehouse/bigdataise.db/employees' TBLPROPERTIES ('creator'='me', 'created_at'='2012-01-02 10:00:00');
OK
Time taken: 0.471 seconds
hive> █

```

13) Copy the schema of an existing tables:

```
hive> CREATE TABLE IF NOT EXISTS mysamplebigdata.employees2 LIKE mysamplebigdata.employees;
OK
Time taken: 0.038 seconds
hive>
```

14) SHOW TABLES:

← → C npbdh.cloudloka.com:4200

bureau_tm3
bureaut
business_bucket
customers_internal
bx_book_rating
bx_temp
bx_users
c
cab
cab4
cab_data
cab_data1
cab_data2
cab_ola
cabs
cabs_ola
calc
calc1
calc11
calc112
calc5
call
call8
calls
calls1
capstone1
capstone11
capstone12
capstone_hbase
capstone_project
capstoneproject
capstoneproject1
card1
cards
cards1
cards20
cards

```
-----  
hive> use default;  
OK  
Time taken: 0.032 seconds  
hive> show tables in mysamplebigdata;  
OK  
employees  
employees2  
stu_details  
Time taken: 0.042 seconds, Fetched: 3 row(s)  
hive>
```

15) DESCRIBE:

```
hive> describe extended mysamplebigdata.employees;
OK
name          string           Employee name
salary        float            Employee salary
subordinates  array<string>    Names of subordinates
deductions    map<string,float> Keys are deductions names, values are percentages
address       struct<street:string,city:string,state:string,zip:int> Home address

Detailed Table Information  Table(tableName:employees, dbName:mysamplebigdata, owner:kct5thsemcdhid025, createTime:1670913367, lastAccessTime:0, retention:0, sd:StorageDescriptor(cols:[FieldSchema(name:name, type:string, comment:Employee name), FieldSchema(name:salary, type:float, comment:Employee salary), FieldSchema(name:subordinates, type:array<string>, comment:Names of subordinates), FieldSchema(name:deductions, type:map<string,float>, comment:Keys are deductions names, values are percentages), FieldSchema(name:address, type:struct<street:string,city:string,state:string,zip:int>, comment:Home address)], location:hdfs://nameservice1/user/hive/warehouse/bigdataise.db/employees, inputFormat:org.apache.hadoop.mapred.TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.format=1}), bucketCols:[], sortCols:[], parameters:{}), skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[], skewedColValueLocationMaps:{}), storedAsSubDirectories:false), partitionKeys:[], parameters:{transient_lastDdlTime=1670913367, created_at=2012-01-02 10:00:00, creator=me, comment=Description of the table}, viewOriginalText:null, viewExpandedText:null, tableType:MANAGED_TABLE, ownerType:USER)
Time taken: 0.097 seconds, Fetched: 7 row(s)
hive>
```

```

hive> describe mysamplebigdata.employees salary;
OK
salary          float          from deserializer
Time taken: 0.078 seconds, Fetched: 1 row(s)
hive>

```

16) EXTERNAL TABLES:

```

hive> CREATE EXTERNAL TABLE IF NOT EXISTS mysamplebigdata.employees3
>   LIKE mysamplebigdata.employees
>   LOCATION '/data/';

```

17) Partitioned, Managed Tables

```

hive> CREATE TABLE employeedetails ( name STRING, salary FLOAT, subordinates ARRAY<STRING>, deductions MAP<STRING, FLOAT>, address STRUCT<street:STRING, city:STRING,
state:STRING, zip:INT>) PARTITIONED BY (country STRING, state STRING);
OK
Time taken: 0.1 seconds
hive>

```

18) External Partitioned Tables

```

hive> CREATE EXTERNAL TABLE IF NOT EXISTS log_messages ( hms INT,severity STRING, server STRING, process_id INT, message STRING) PARTITIONED BY (year INT, month INT,
day INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
OK
Time taken: 0.083 seconds
hive>

```

19) DESCRIBE PARTITIONS:

```

hive> describe extended log_messages;
OK
hms          int
severity      string
server        string
process_id    int
message       string
year          int
month         int
day           int

# Partition Information
# col_name      data_type      comment
year          int
month         int
day           int

Detailed Table Information  Table(tableName:log_messages, dbName:default, owner:kct5thsemcdhid025, createTime:1670915193, lastAccessTime:0, retention:0, sd:StorageDescriptor(cols:[FieldSchema(name:hms, type:int, comment:null), FieldSchema(name:severity, type:string, comment:null), FieldSchema(name:server, type:string, comment:null), FieldSchema(name:process_id, type:int, comment:null), FieldSchema(name:message, type:string, comment:null), FieldSchema(name:year, type:int, comment:null), FieldSchema(name:month, type:int, comment:null), FieldSchema(name:day, type:int, comment:null)], location:hdfs://nameservice1/user/hive/warehouse/log_messages, inputFormat:org.apache.hadoop.mapred.TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1, serdeInfo:SerdeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.format=t, field.delim=t}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValueLocationMaps:{}), storedAsSubDirectories:false), partitionKeys:[FieldSchema(name:year, type:int, comment:null), FieldSchema(name:month, type:int, comment:null), FieldSchema(name:day, type:int, comment:null)], parameters:{totalSize=0, EXTERNAL=TRUE, numRows=0, rawDataSize=0, COLUMN_STATS_ACCURATE={"BASIC_STATS":"true"}, numFiles=0, numPartitions=0, transient_lastDdlTime=1670915193, numFileErasureCodes=0}, viewOriginalText=null, viewExpandedText=null, tableType:EXTERNAL_TABLE, ownerType:USER)
Time taken: 0.085 seconds, Fetched: 17 row(s)
hive>

```

20) IEW

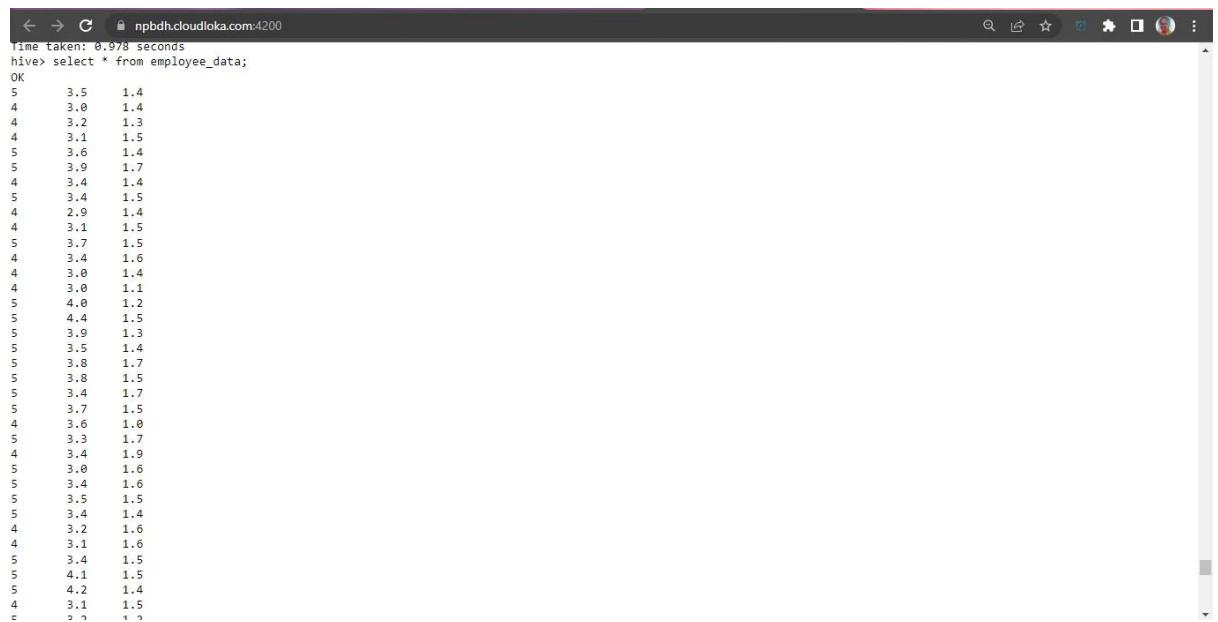
```
hive> Create VIEW Sample_View AS SELECT * FROM mysamplebigdata.employees2 WHERE salary>25000;
OK
Time taken: 0.158 seconds
hive> █
```

FUNCTIONS:**USER DEFINED FUNCTIONS:****Create:**

```
hive> create table employee_data (Id int, Name string , Salary float) row format delimited fields terminated by ',' ;
OK
Time taken: 0.083 seconds
hive> █
```

Load data:

```
hive> load data local inpath '/home/kct5thsemcdhid025/iris.csv' into table employee_data;
Loading data to table default.employee_data
OK
Time taken: 0.978 seconds
hive>
```

Select the data:


5	3.5	1.4
4	3.0	1.4
4	3.2	1.3
4	3.1	1.5
5	3.6	1.4
5	3.9	1.7
4	3.4	1.4
5	3.4	1.5
4	2.9	1.4
4	3.1	1.5
5	3.7	1.5
4	3.4	1.6
4	3.0	1.4
4	3.0	1.1
5	4.0	1.2
5	4.4	1.5
5	3.9	1.3
5	3.5	1.4
5	3.8	1.7
5	3.8	1.5
5	3.4	1.7
5	3.7	1.5
4	3.6	1.0
5	3.3	1.7
4	3.4	1.9
5	3.0	1.6
5	3.4	1.6
5	3.5	1.5
5	3.4	1.4
4	3.2	1.6
4	3.1	1.6
5	3.4	1.5
5	4.1	1.5
5	4.2	1.4
4	3.1	1.5
c	2.9	1.3

Mathematical Functions:

```
hive> select Id, Name, sqrt(Salary) from employee_data ;
Query ID = kct5thsemcdhid025_20221213072422_6a60cc1c-95ea-4a40-b5b7-0f43ee1a1c2f
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
22/12/13 07:24:23 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:24:23 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21632, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21632/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21632
```

```

← → C 🔒 npbdh.cloudloka.com:4200
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
22/12/13 07:24:23 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:24:23 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21632, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21632/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21632
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2022-12-13 07:24:34,371 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:24:45,803 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 4.23 sec
MapReduce Total cumulative CPU time: 4 seconds 230 msec
Ended Job = job_1663041244711_21632
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Cumulative CPU: 4.23 sec HDFS Read: 12917 HDFS Write: 6812 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 230 msec
OK
NULL FIRST_NAME NULL
198 Donald NULL
199 Douglas NULL
200 Jennifer NULL
201 Michael NULL
202 Pat NULL
203 Susan NULL
204 Hermann NULL
205 Shelley NULL
206 William NULL
100 Steven NULL
101 Neena NULL
102 Lex NULL
103 Alexander NULL
104 Bruce NULL
105 David NULL
106 Valli NULL
107 Diana NULL
108 Nancy NULL
109 Daniel NULL
110 John NULL
111 Ismael NULL
112 Jose Manuel NULL
113 Lint NULL
114 null NULL
115 null NULL
116 null NULL
117 null NULL
118 null NULL
119 null NULL
120 null NULL
121 null NULL
122 null NULL
123 null NULL
124 null NULL
125 null NULL
126 null NULL
127 null NULL
128 null NULL
129 null NULL
130 null NULL
131 null NULL
132 null NULL
133 null NULL
134 null NULL
135 null NULL
136 null NULL
137 null NULL
138 null NULL
139 null NULL
140 null NULL
141 null NULL
142 null NULL
143 null NULL
144 null NULL
145 null NULL
146 null NULL
147 null NULL
148 null NULL
149 null NULL
150 null NULL
151 null NULL
152 null NULL
153 null NULL
154 null NULL
155 null NULL
156 null NULL
157 null NULL
158 null NULL
159 null NULL
160 null NULL
161 null NULL
162 null NULL
163 null NULL
164 null NULL
165 null NULL
166 null NULL
167 null NULL
168 null NULL
169 null NULL
170 null NULL
171 null NULL
172 null NULL
173 null NULL
174 null NULL
175 null NULL
176 null NULL
177 null NULL
178 null NULL
179 null NULL
180 null NULL
181 null NULL
182 null NULL
183 null NULL
184 null NULL
185 null NULL
186 null NULL
187 null NULL
188 null NULL
189 null NULL
190 null NULL
191 null NULL
192 null NULL
193 null NULL
194 null NULL
195 null NULL
196 null NULL
197 null NULL
198 null NULL
199 null NULL
200 null NULL
201 null NULL
202 null NULL
203 null NULL
204 null NULL
205 null NULL
206 null NULL
207 null NULL
208 null NULL
209 null NULL
210 null NULL
211 null NULL
212 null NULL
213 null NULL
214 null NULL
215 null NULL
216 null NULL
217 null NULL
218 null NULL
219 null NULL
220 null NULL
221 null NULL
222 null NULL
223 null NULL
224 null NULL
225 null NULL
226 null NULL
227 null NULL
228 null NULL
229 null NULL
230 null NULL
231 null NULL
232 null NULL
233 null NULL
234 null NULL
235 null NULL
236 null NULL
237 null NULL
238 null NULL
239 null NULL
240 null NULL
241 null NULL
242 null NULL
243 null NULL
244 null NULL
245 null NULL
246 null NULL
247 null NULL
248 null NULL
249 null NULL
250 null NULL
251 null NULL
252 null NULL
253 null NULL
254 null NULL
255 null NULL
256 null NULL
257 null NULL
258 null NULL
259 null NULL
260 null NULL
261 null NULL
262 null NULL
263 null NULL
264 null NULL
265 null NULL
266 null NULL
267 null NULL
268 null NULL
269 null NULL
270 null NULL
271 null NULL
272 null NULL
273 null NULL
274 null NULL
275 null NULL
276 null NULL
277 null NULL
278 null NULL
279 null NULL
280 null NULL
281 null NULL
282 null NULL
283 null NULL
284 null NULL
285 null NULL
286 null NULL
287 null NULL
288 null NULL
289 null NULL
290 null NULL
291 null NULL
292 null NULL
293 null NULL
294 null NULL
295 null NULL
296 null NULL
297 null NULL
298 null NULL
299 null NULL
299 null NULL

```

Select max salary from employee table:

```

hive> select max(Salary) from employee_data;
Query ID = kct5thsemcdhid025_20221213072532_1b329cf-43cc-4b88-b19e-0a29f3457eeb
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:25:32 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:25:32 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21635, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21635/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21635

```

```

Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:25:42,747 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:25:59,269 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.21 sec
2022-12-13 07:26:08,662 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 6.13 sec
MapReduce Total cumulative CPU time: 6 seconds 130 msec
Ended Job = job_1663041244711_21635
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 6.13 sec HDFS Read: 16673 HDFS Write: 103 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 130 msec
OK
6.9
Time taken: 38.516 seconds, Fetched: 1 row(s)
hive>
```

Select min salary from employee table:

```

hive> select min(Salary) from employee_data;
Query ID = kct5thsemcdhid025_20221213072659_cd934593-d735-4c54-aa1e-d8df6dc14d7c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:27:00  INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:27:00  INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21640, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21640/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21640
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:27:22,317 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:27:35,762 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.91 sec
2022-12-13 07:27:45,085 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.69 sec
MapReduce Total cumulative CPU time: 5 seconds 690 msec
Ended Job = job_1663041244711_21640
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.69 sec HDFS Read: 16673 HDFS Write: 103 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 690 msec
OK
1.0
Time taken: 47.26 seconds, Fetched: 1 row(s)
hive>
```

Select average salary from employee table:

```

hive> select avg(Salary) from employee_data;
Query ID = kct5thsemcdhid025_20221213072827_ef1406f3-e9c8-4d1d-b40a-159d9e5c860d
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:28:28  INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:28:28  INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21643, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21643/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21643
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:28:41,604 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:28:49,914 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.32 sec
2022-12-13 07:28:57,294 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.17 sec
MapReduce Total cumulative CPU time: 5 seconds 170 msec
Ended Job = job_1663041244711_21643
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.17 sec HDFS Read: 17131 HDFS Write: 118 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 170 msec
OK
3.758666552225747
Time taken: 31.574 seconds, Fetched: 1 row(s)
Query ID = kct5thsemcdhid025_20221213072859_06698cb5-646a-4144-900d-849f21d69df8
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
```

```

← → C 🔒 npbdh.cloudloka.com:4200
2022-12-13 07:28:49,914 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.32 sec
2022-12-13 07:28:57,294 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.17 sec
MapReduce Total cumulative CPU time: 5 seconds 170 msec
Ended Job = job_1663041244711_21643
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.17 sec HDFS Read: 17131 HDFS Write: 118 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 170 msec
OK
3.7586666552225747
Time taken: 31.574 seconds, Fetched: 1 row(s)
Query ID = kct5thsemcdhid025_20221213072859_06698cb5-646a-4144-900d-849f21d69df8
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:28:59 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:28:59 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21644, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:8086/proxy/application_1663041244711_21644/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21644
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:29:19,391 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:29:27,597 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.44 sec
2022-12-13 07:29:41,079 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 6.15 sec
MapReduce Total cumulative CPU time: 6 seconds 150 msec
Ended Job = job_1663041244711_21644
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 6.15 sec HDFS Read: 17131 HDFS Write: 118 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 150 msec
OK
3.7586666552225747
Time taken: 43.73 seconds, Fetched: 1 row(s)
hive> 
```

SUM OF SALARY:

```

hive> select sum(Salary) from employee_data;select sum(Salary) from employee_data;
Query ID = kct5thsemcdhid025_20221213073016_bac54ff9-d579-4c15-8160-77cd0254536
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:30:16 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:30:16 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21647, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:8086/proxy/application_1663041244711_21647/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21647
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:30:28,524 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:30:36,766 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.44 sec
2022-12-13 07:30:44,001 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.42 sec
MapReduce Total cumulative CPU time: 5 seconds 420 msec
Ended Job = job_1663041244711_21647
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.42 sec HDFS Read: 16860 HDFS Write: 117 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 420 msec
OK
563.7999982833862
Time taken: 28.754 seconds, Fetched: 1 row(s)
Query ID = kct5thsemcdhid025_20221213073045_e17acb8c-d2a1-4424-9817-0321cc12c450
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers: 
```

```
Total MapReduce CPU Time Spent: 5 seconds 420 msec
OK
563.7999982833862
Time taken: 28.754 seconds, Fetched: 1 row(s)
Query ID = kct5thsemcdhid025_20221213073045_e17acb8c-d2a1-4424-9817-0321cc12c450
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:30:45 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:30:45 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job job_1663041244711_21649, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21649
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21649
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:30:55,408 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:31:07,685 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.81 sec
2022-12-13 07:31:28,121 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 4.67 sec
MapReduce Total cumulative CPU time: 4 seconds 670 msec
Ended Job = job_1663041244711_21649
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 4.67 sec HDFS Read: 16867 HDFS Write: 117 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 670 msec
OK
563.7999982833862
Time taken: 45.12 seconds, Fetched: 1 row(s)
hive>
```

COUNT:

```
hive> select Count(*) from employee_data;
Query ID = kct5thsemcdhid025_20221213073208_38b199a4-2923-46d3-8338-845ac2fbe574
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:32:08 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:32:08 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job job_1663041244711_21653, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21653
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21653
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:32:22,202 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:32:32,564 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.75 sec
2022-12-13 07:32:41,765 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.43 sec
MapReduce Total cumulative CPU time: 5 seconds 430 msec
Ended Job = job_1663041244711_21653
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.43 sec HDFS Read: 16596 HDFS Write: 103 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 430 msec
OK
202
Time taken: 35.698 seconds, Fetched: 1 row(s)
Query ID = kct5thsemcdhid025_20221213073243_a681cc71-e64a-4b8c-9c3e-71aee1ac2b01
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:

In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:32:44 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:32:44 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job job_1663041244711_21654, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21654
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21654
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:32:54,003 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:33:05,253 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.47 sec
2022-12-13 07:33:16,597 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.26 sec
MapReduce Total cumulative CPU time: 5 seconds 260 msec
Ended Job = job_1663041244711_21654
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.26 sec HDFS Read: 16596 HDFS Write: 103 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 260 msec
OK
202
Time taken: 33.775 seconds, Fetched: 1 row(s)
hive>
```

To fetch the employees whose salary is greater than 25000

```

hive> select sum(Salary) from employee_data;
Query ID = kct5thsemcdhid025_20221213073410_f62e4d2f-395f-471b-a352-cb3302d4e2d9
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/12/13 07:34:10 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
22/12/13 07:34:10 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10.1.1.204:8032
Starting Job = job_1663041244711_21657, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/proxy/application_1663041244711_21657/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.1.p0.1425774/lib/hadoop/bin/hadoop job -kill job_1663041244711_21657
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-12-13 07:34:28,686 Stage-1 map = 0%, reduce = 0%
2022-12-13 07:34:37,894 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 2.79 sec
2022-12-13 07:34:48,260 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.66 sec
MapReduce Total cumulative CPU time: 5 seconds 660 msec
Ended Job = job_1663041244711_21657
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.66 sec HDFS Read: 16867 HDFS Write: 117 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 660 msec
OK
563.7999982833862
Time taken: 40.295 seconds, Fetched: 1 row(s)
hive> ■

```

RESULT:

The Hive operations like create, alter, and drop databases, tables, views, functions, and indexes are implemented successfully.

LAB EXERCISE-10

Title of the exercise/experiment : Verify, Sparse and perform advance join of data using spark

Verify, Sparse and perform advance join of data using spark

1) Broadcast Hash Join

```
scala> spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
res0: String = 10485760
```

scala>

```
scala> val data1 = Seq(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 20, 20, 50)
data1: Seq[Int] = List(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 20, 20, 50)
```

scala>

```
scala> val df1 = data1.toDF("id1")
df1: org.apache.spark.sql.DataFrame = [id1: int]
```

```
scala>
```

```
scala> val data2 = Seq(30, 20, 40, 50)
data2: Seq[Int] = List(30, 20, 40, 50)

scala> ■

scala> val df2 = data2.toDF("id2")
df2: org.apache.spark.sql.DataFrame = [id2: int]

scala>

scala> val dfJoined = df1.join(df2, $"id1" === $"id2")
dfJoined: org.apache.spark.sql.DataFrame = [id1: int, id2: int]

scala>

scala> dfJoined.queryExecution.executedPlan
res1: org.apache.spark.sql.execution.SparkPlan =
*(1) BroadcastHashJoin [id1#3], [id2#8], Inner, BuildRight
:- LocalTableScan [id1#3]
+- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))
   +- LocalTableScan [id2#8]

scala> ■

scala> dfJoined.show
+---+---+
|id1|id2|
+---+---+
| 20| 20|
| 20| 20|
| 30| 30|
| 40| 40|
| 40| 40|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 50| 50|
+---+---+  
scala>
```

2) Shuffle Hash Join

```
scala> spark.conf.set("spark.sql.autoBroadcastJoinThreshold", 2)
scala> spark.conf.set("spark.sql.join.preferSortMergeJoin", "false")
scala> spark.conf.get("spark.sql.join.preferSortMergeJoin")
res5: String = false
scala>


---


scala> spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
res6: String = 2
scala> █


---


scala> val data1 = Seq(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 20, 50)
data1: Seq[Int] = List(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 20, 50)
scala>


---


scala> val df1 = data1.toDF("id1")
df1: org.apache.spark.sql.DataFrame = [id1: int]
scala>


---


scala> val df1 = data1.toDF("id1")
df1: org.apache.spark.sql.DataFrame = [id1: int]
scala> val data2 = Seq(30, 20, 40, 50)
data2: Seq[Int] = List(30, 20, 40, 50)
scala>
```

```

scala> val df2 = data2.toDF("id2")
df2: org.apache.spark.sql.DataFrame = [id2: int]

scala> val dfJoined = df1.join(df2, $"id1" === $"id2")
dfJoined: org.apache.spark.sql.DataFrame = [id1: int, id2: int]

scala>

```

We are setting `spark.sql.autoBroadcastJoinThreshold` to -1
to disable broadcast.

COMMANDS

```

1.scala> spark.conf.get("spark.sql.join.preferSortMergeJoin") res1:
  String
= true
2.scala> spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
  res2: String
= -1
3.scala> val data1 = Seq(10, 20, 20, 30, 40, 10, 40, 20, 20, 20,
  50)
data1: Seq[Int] = List(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 50)
4.scala> val df1 = data1.toDF("id1")
df1: org.apache.spark.sql.DataFrame =
[id1: int]
5.scala> val data2 = Seq(30, 20,
  40, 50)
data2: Seq[Int] = List(30, 20, 40, 50)
6.scala> val df2
= data2.toDF("id2")
df2: org.apache.spark.sql.DataFrame = [id2: int]

```

```

scala> dfJoined.queryExecution.executedPlan
res7: org.apache.spark.sql.execution.SparkPlan =
ShuffledHashJoin [id1#26], [id2#31], Inner, BuildRight
:- Exchange hashpartitioning(id1#26, 200)
: +- LocalTableScan [id1#26]
+- Exchange hashpartitioning(id2#31, 200)
  +- LocalTableScan [id2#31]

scala> dfJoined.show
+---+---+
|id1|id2|
+---+---+
| 40| 40|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 40| 40|
| 50| 50|
| 30| 30|
+---+---+

```

```
scala> █
```

COMMANDS

```

scala> val dfJoined = df1.join(df2, $"id1" >= $"id2")
dfJoined: org.apache.spark.sql.DataFrame = [id1: int, id2: int]

```

COMMANDS

```

scala>
dfJoined.queryExecution.execute
dPlans
scala> dfJoined.show

```

```

← → C npbdh.cloudloka.com:4200
scala> val df1 = data1.toDF("id1")
df1: org.apache.spark.sql.DataFrame = [id1: int]

scala> val data2 = Seq(30, 20, 40, 50)
data2: Seq[Int] = List(30, 20, 40, 50)

scala> val df2 = data2.toDF("id2")
df2: org.apache.spark.sql.DataFrame = [id2: int]

scala> val dfJoined = df1.join(df2, $"id1" === $"id2")
dfJoined: org.apache.spark.sql.DataFrame = [id1: int, id2: int]

scala> dfJoined.queryExecution.executedPlan
res7: org.apache.spark.sql.execution.SparkPlan =
ShuffledHashJoin [id1#26], [id2#31], Inner, BuildRight
:- Exchange hashpartitioning(id1#26, 200)
:+ LocalTableScan [id1#26]
++ Exchange hashpartitioning(id2#31, 200)
++ LocalTableScan [id2#31]

scala> dfJoined.show
+---+---+
|id1|id2|
+---+---+
| 40| 40|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 40| 40|
| 50| 50|
| 30| 30|
+---+---+

```

3) CARTESIAN PRODUCT JOIN

```

scala> spark.conf.get("spark.sql.join.preferSortMergeJoin")
res9: String = false

scala> spark.conf.get("spark.sql.autoBroadcastJoinThreshold")
res10: String = 2

scala> val data1 = Seq(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 20, 50)
data1: Seq[Int] = List(10, 20, 20, 30, 40, 10, 40, 20, 20, 20, 20, 50)

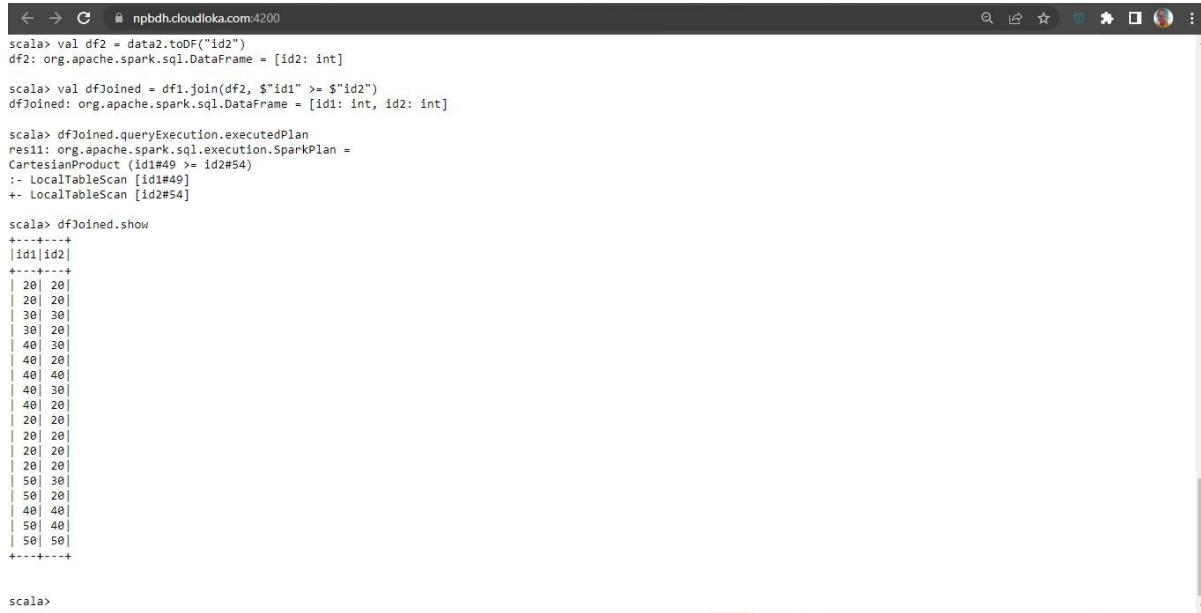
scala> val df1 = data1.toDF("id1")
df1: org.apache.spark.sql.DataFrame = [id1: int]

scala> val data2 = Seq(30, 20, 40, 50)
data2: Seq[Int] = List(30, 20, 40, 50)

scala> val df2 = data2.toDF("id2")
df2: org.apache.spark.sql.DataFrame = [id2: int]

scala>

```



The screenshot shows a Scala REPL session running on a web-based interface at npbdh.cloudloka.com:4200. The session demonstrates performing a join operation between two DataFrames and then displaying the resulting DataFrame's execution plan.

```
scala> val df2 = data2.toDF("id2")
df2: org.apache.spark.sql.DataFrame = [id2: int]

scala> val dfJoined = df1.join(df2, $"id1" >= $"id2")
dfJoined: org.apache.spark.sql.DataFrame = [id1: int, id2: int]

scala> dfJoined.queryExecution.executedPlan
res1: org.apache.spark.sql.execution.SparkPlan =
CartesianProduct (id1#49 >= id2#54)
:- LocalTableScan [id1#49]
++ LocalTableScan [id2#54]

scala> dfJoined.show
+---+---+
|id1|id2|
+---+---+
| 20| 20|
| 20| 20|
| 30| 30|
| 30| 20|
| 40| 30|
| 40| 20|
| 40| 40|
| 40| 30|
| 40| 20|
| 20| 20|
| 20| 20|
| 20| 20|
| 50| 30|
| 50| 20|
| 40| 40|
| 50| 40|
| 50| 50|
+---+---+

```

Conclusion:

To Verify, Sparse and perform advance join of data using spark can be implemented.