Practical No. :- 1

Aim:- HDFS: List of Commands (mkdir, touchz, copy from local/put, copy to local/get, move from local, cp, rmr, du,dus, stat)

Theory:-

HDFS Commands and Their Theoretical Explanation

1. **mkdir**
   - Creates a new directory in the HDFS file system.
   - Similar to creating a folder on a local file system but performed on the distributed storage system.
   - Syntax: hadoop fs -mkdir <directory_path>
2. **touchz**
   - Creates an empty file in HDFS if it does not already exist.
   - Useful for creating placeholder files for specific processes.
   - Syntax: hadoop fs -touchz <file_path>
3. **copyFromLocal / put**
   - Copies files from the local file system to the HDFS.
   - copyFromLocal is a more verbose command, while put is shorthand for the same functionality.
   - Syntax: hadoop fs -copyFromLocal <local_path> <hdfs_path>
   - Alternate: hadoop fs -put <local_path> <hdfs_path>
4. **copyToLocal / get**
   - Copies files from HDFS to the local file system.
   - copyToLocal is explicit, while get serves as a shorthand for the same.
   - Syntax: hadoop fs -copyToLocal <hdfs_path> <local_path>
   - Alternate: hadoop fs -get <hdfs_path> <local_path>
5. **moveFromLocal**
   - Moves a file from the local file system to HDFS and deletes the source file after transferring.
   - It ensures the file no longer exists locally.
   - Syntax: hadoop fs -moveFromLocal <local_path> <hdfs_path>
6. **cp**
   - Copies files within the HDFS from one location to another.
   - The source and destination paths must be in HDFS.
   - Syntax: hadoop fs -cp <source_path> <destination_path>
7. **rmr**
   - Removes directories recursively in HDFS.
   - It deletes directories and their contents permanently.
   - Syntax: hadoop fs -rmr <directory_path>
     (Note: This command has been replaced by -rm -r in newer versions.)
8. **du**
   - Displays the disk usage of files and directories in HDFS.
   - Provides the size of each file/directory in bytes.
   - Syntax: hadoop fs -du <directory_path>
9. **dus**
   - Summarizes the disk usage of files and directories in HDFS.
   - Shows the aggregate size of directories in a more user-friendly format.
   - Syntax: hadoop fs -dus <directory_path>

10. stat

- Provides detailed metadata about a file or directory in HDFS.
- Includes information like file size, block size, replication factor, and modification time.
- Syntax: hadoop fs -stat <file_path>

Code:-

1. Create a Directory (mkdir):-

   hadoop fs -mkdir /path/to/directory

2. Create an Empty File (touchz):-

   hadoop fs -touchz /path/to/filename

3. Copy a File from Local to HDFS (copyFromLocal or put):-

   hadoop fs -copyFromLocal /local/path /hdfs/path
   # OR
   hadoop fs -put /local/path /hdfs/path

4. Copy a File from HDFS to Local (copyToLocal or get):-

   hadoop fs -copyToLocal /hdfs/path /local/path

   # OR

   hadoop fs -get /hdfs/path /local/path

5. Move a File from Local to HDFS (moveFromLocal):-

   hadoop fs -moveFromLocal /local/path /hdfs/path

6. **Copy a File within HDFS (cp):-**

hadoop fs -cp /source/path /destination/path

7. Remove a File or Directory (rmr):-

hadoop fs -rmr /path/to/file-or-directory

8. Display File/Directory Size (du):-

hadoop fs -du /path

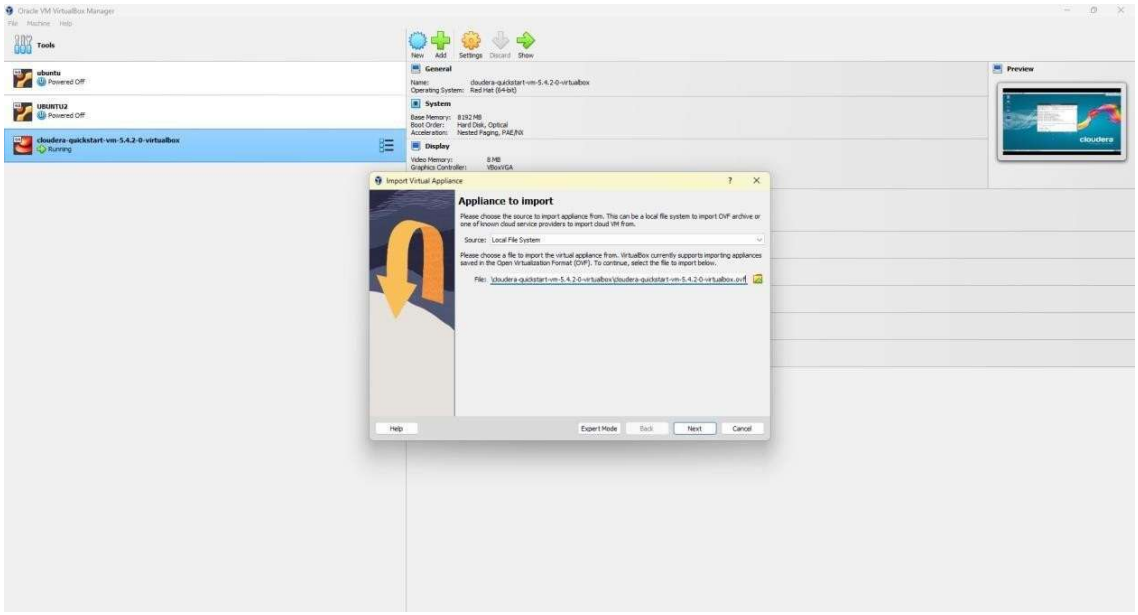
9. Display Summary of Disk Usage (dus):-

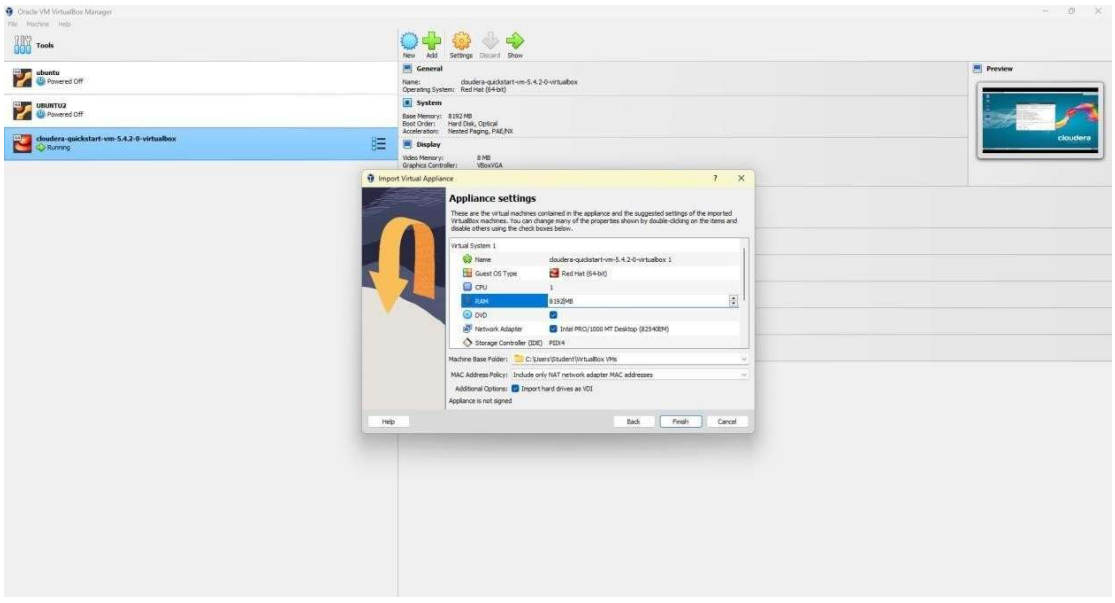
hadoop fs -dus /path

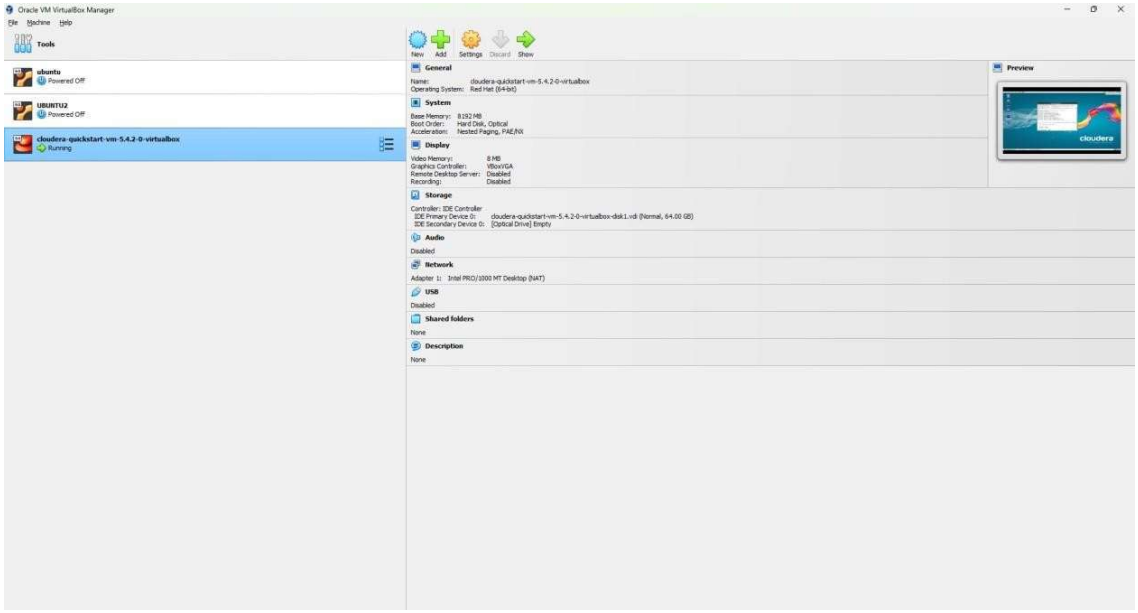
10. Display File Information (stat):-

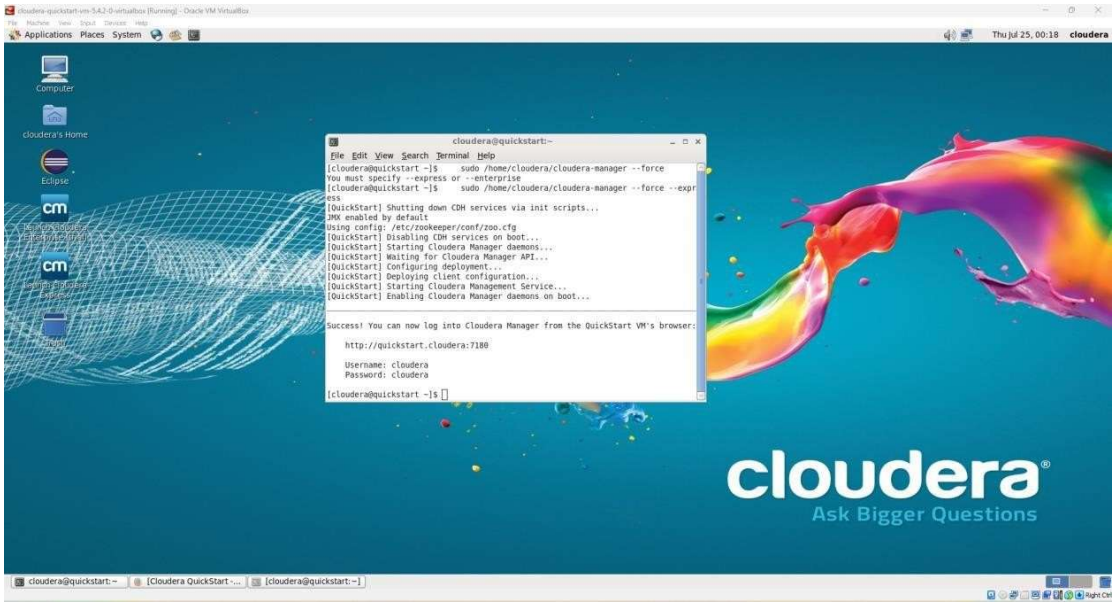
hadoop fs -stat [format] /path

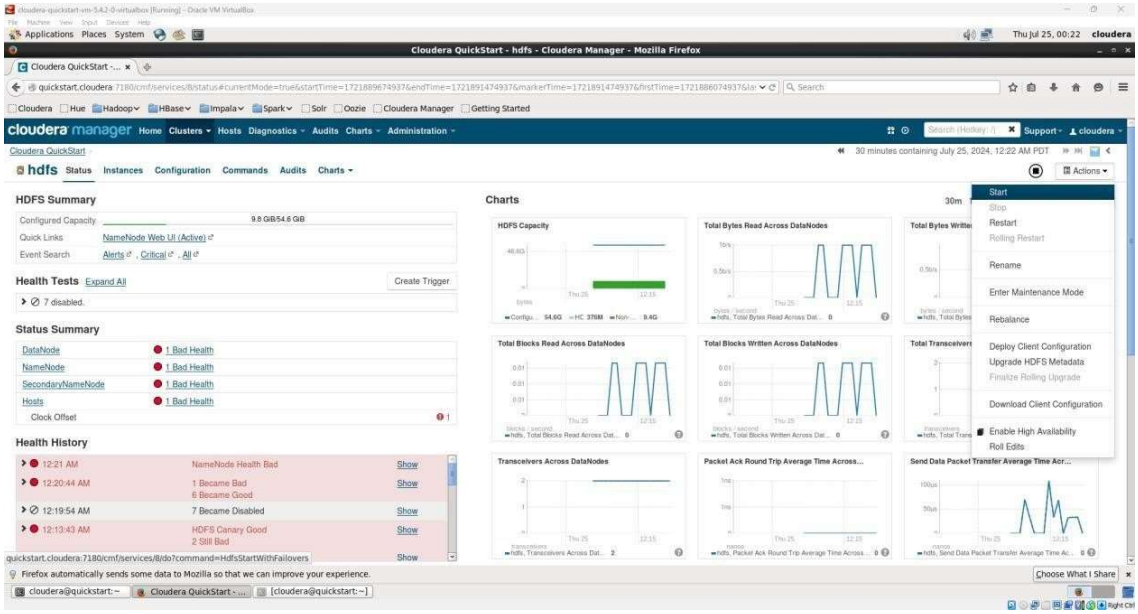
Output:-

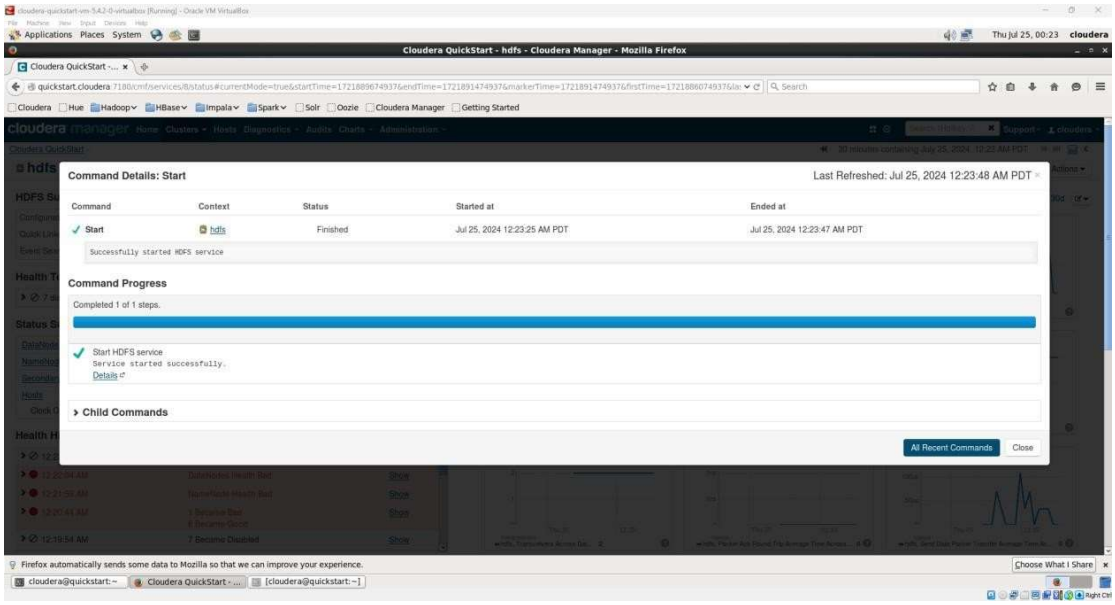


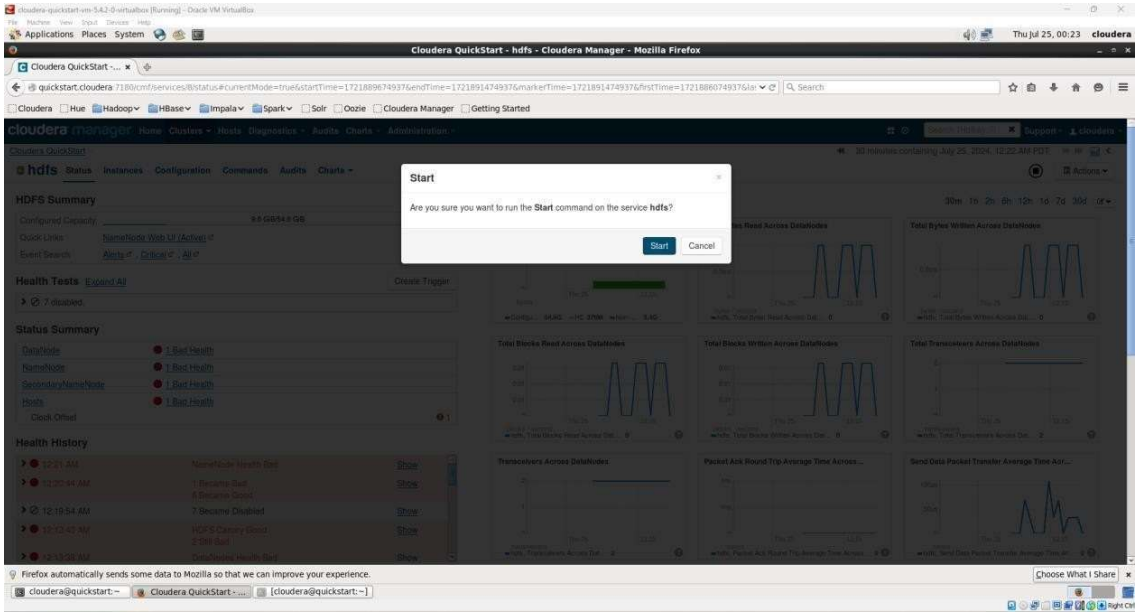


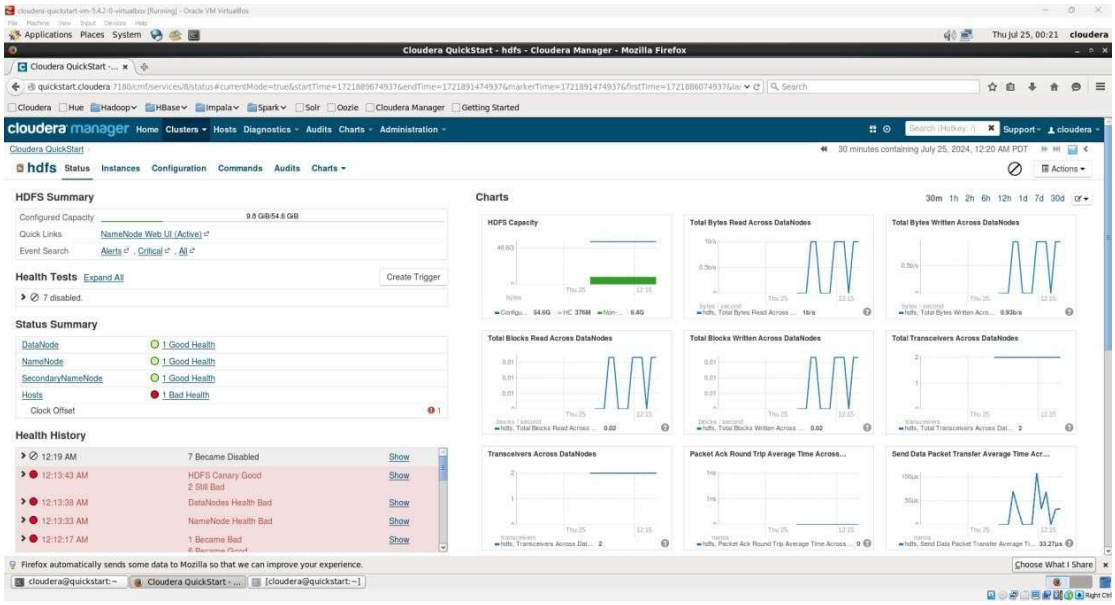

Oracle VM VirtualBox Manager
File Machine Help
Tools
ubuntu
Powered Off
UBUNTU2
Powered Off
cloudera-quickstart-vm-5.4.2-0-virtualbox
Running
New Add Settings Discard Show
General
System
Display
Storage
Audio
Network
USB
Shared folders
Description
None
Preview
cloudera

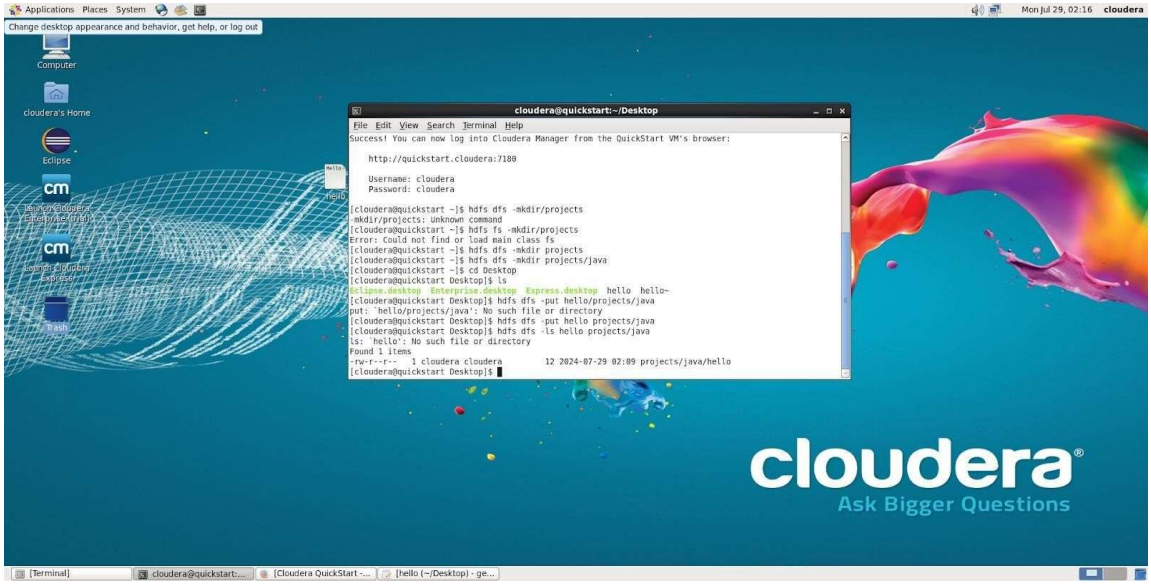

Applications Places System
Thu Jul 25, 00:18 cloudera
Computer
cloudera's Home
Eclipse
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ sudo /home/cloudera/cloudera-manager --force
You must specify --express or --enterprise
[cloudera@quickstart ~]$ sudo /home/cloudera/cloudera-manager --force --express
[QuickStart] Shutting down CDH services via init scripts...
JMX enabled by default
Using config: /etc/zookeeper/conf/zoo.cfg
[QuickStart] Disabling CDH services on boot...
[QuickStart] Starting Cloudera Manager daemons...
[QuickStart] Waiting for Cloudera Manager API...
[QuickStart] Configuring deployment...
[QuickStart] Deploying client configuration...
[QuickStart] Starting Cloudera Management Service...
[QuickStart] Enabling Cloudera Manager daemons on boot...
Success! You can now log into Cloudera Manager from the QuickStart VM's browser:
http://quickstart.cloudera:7180
Username: cloudera
Password: cloudera
[cloudera@quickstart ~]$
cloudera
Ask Bigger Questions
cloudera@quickstart:~
[Cloudera QuickStart -...
[cloudera@quickstart:~]

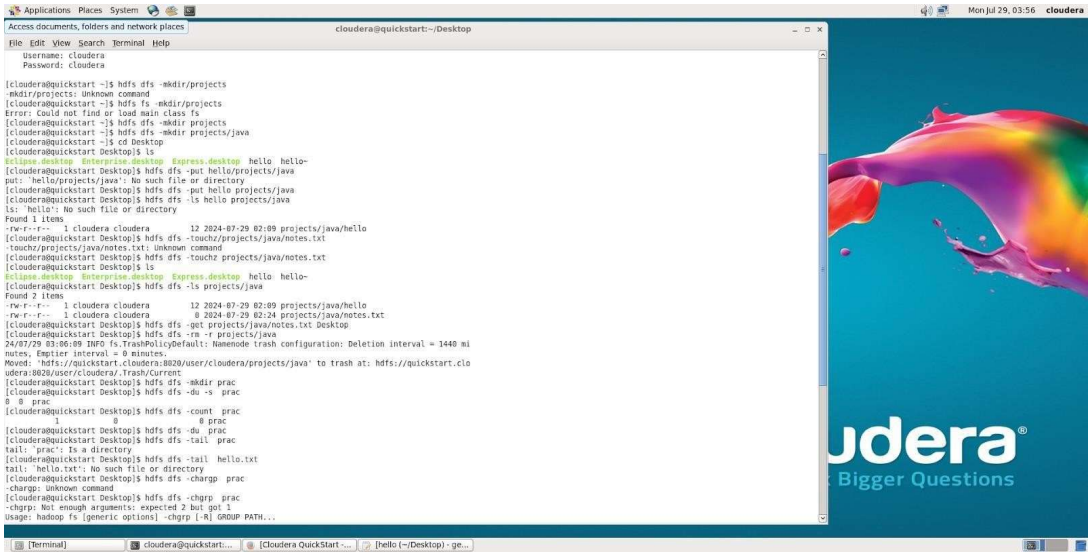

Cloudera QuickStart - hdfs - Cloudera Manager - Mozilla Firefox
cloudera manager
hdfs
HDFS Summary
Health Tests
Status Summary
Health History
Charts
Start
Stop
Restart
Rolling Restart
Rename
Enter Maintenance Mode
Rebalance
Deploy Client Configuration
Upgrade HDFS Metadata
Finalize Rolling Upgrade
Download Client Configuration
Enable High Availability
Roll Edits


Cloudera QuickStart - hdfs - Cloudera Manager - Mozilla Firefox
Command Details: Start
Last Refreshed: Jul 25, 2024 12:23:48 AM PDT
Command
Context
Status
Started at
Ended at
Start
hdfs
Finished
Jul 25, 2024 12:23:25 AM PDT
Jul 25, 2024 12:23:47 AM PDT
Successfully started HDFS service
Command Progress
Completed 1 of 1 steps.
Start HDFS service
Service started successfully.
Details
Child Commands
All Recent Commands
Close


Cloudera QuickStart - hdfs - Cloudera Manager - Mozilla Firefox
Start
Are you sure you want to run the Start command on the service hdfs?
Start
Cancel


Cloudera QuickStart - hdfs - Cloudera Manager - Mozilla Firefox
HDFS Summary
Charts
Health Tests
Status Summary
Health History


Applications Places System
Mon Jul 29, 02:16 cloudera
Change desktop appearance and behavior, get help, or log out
Computer
cloudera's Home
Eclipse
Trash
cloudera@quickstart:~/Desktop
File Edit View Search Terminal Help
Success! You can now log into Cloudera Manager from the QuickStart VM's browser:
http://quickstart.cloudera:7180
Username: cloudera
Password: cloudera
[cloudera@quickstart ~]$ hdfs dfs -mkdir/projects
-mkdir/projects: Unknown command
[cloudera@quickstart ~]$ hdfs fs -mkdir/projects
Error: Could not find or load main class fs
[cloudera@quickstart ~]$ hdfs dfs -mkdir projects
[cloudera@quickstart ~]$ hdfs dfs -mkdir projects/java
[cloudera@quickstart ~]$ cd Desktop
[cloudera@quickstart Desktop]$ ls
Eclipse.desktop Enterprise.desktop Express.desktop hello hello~
[cloudera@quickstart Desktop]$ hdfs dfs -put hello/projects/java
put: `hello/projects/java': No such file or directory
[cloudera@quickstart Desktop]$ hdfs dfs -put hello projects/java
[cloudera@quickstart Desktop]$ hdfs dfs -ls hello projects/java
ls: `hello': No such file or directory
Found 1 items
-rw-r--r-- 1 cloudera cloudera 12 2024-07-29 02:09 projects/java/hello
[cloudera@quickstart Desktop]$
cloudera®
Ask Bigger Questions
[Terminal]
cloudera@quickstart:...
[Cloudera QuickStart -...
[hello (~/Desktop) - ge...


Applications Places System
Mon Jul 29, 03:56 cloudera
Access documents, folders and network places
cloudera@quickstart:~/Desktop
File Edit View Search Terminal Help
Username: cloudera
Password: cloudera
[cloudera@quickstart ~]$ hdfs dfs -mkdir/projects
-mkdir/projects: Unknown command
[cloudera@quickstart ~]$ hdfs fs -mkdir/projects
Error: Could not find or load main class fs
[cloudera@quickstart ~]$ hdfs dfs -mkdir projects
[cloudera@quickstart ~]$ hdfs dfs -mkdir projects/java
[cloudera@quickstart ~]$ cd Desktop
[cloudera@quickstart Desktop]$ ls
Eclipse.desktop Enterprise.desktop Express.desktop hello hello~
[cloudera@quickstart Desktop]$ hdfs dfs -put hello/projects/java
put: `hello/projects/java': No such file or directory
[cloudera@quickstart Desktop]$ hdfs dfs -put hello projects/java
[cloudera@quickstart Desktop]$ hdfs dfs -ls hello projects/java
ls: `hello': No such file or directory
Found 1 items
-rw-r--r-- 1 cloudera cloudera 12 2024-07-29 02:09 projects/java/hello
[cloudera@quickstart Desktop]$ hdfs dfs -touchz/projects/java/notes.txt
-touchz/projects/java/notes.txt: Unknown command
[cloudera@quickstart Desktop]$ hdfs dfs -touchz projects/java/notes.txt
[cloudera@quickstart Desktop]$ ls
Eclipse.desktop Enterprise.desktop Express.desktop hello hello~
[cloudera@quickstart Desktop]$ hdfs dfs -ls projects/java
Found 2 items
-rw-r--r-- 1 cloudera cloudera 12 2024-07-29 02:09 projects/java/hello
-rw-r--r-- 1 cloudera cloudera 0 2024-07-29 02:24 projects/java/notes.txt
[cloudera@quickstart Desktop]$ hdfs dfs -get projects/java/notes.txt Desktop
[cloudera@quickstart Desktop]$ hdfs dfs -rm -r projects/java
24/07/29 03:06:09 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 1440 mi
nutes, Emptier interval = 0 minutes.
Moved: 'hdfs://quickstart.cloudera:8020/user/cloudera/projects/java' to trash at: hdfs://quickstart.clo
udera:8020/user/cloudera/.Trash/Current
[cloudera@quickstart Desktop]$ hdfs dfs -mkdir prac
[cloudera@quickstart Desktop]$ hdfs dfs -du -s prac
0 0 prac
[cloudera@quickstart Desktop]$ hdfs dfs -count prac
1 0 0 prac
[cloudera@quickstart Desktop]$ hdfs dfs -du prac
[cloudera@quickstart Desktop]$ hdfs dfs -tail prac
tail: `prac': Is a directory
[cloudera@quickstart Desktop]$ hdfs dfs -tail hello.txt
tail: `hello.txt': No such file or directory
[cloudera@quickstart Desktop]$ hdfs dfs -chargp prac
-chargp: Unknown command
[cloudera@quickstart Desktop]$ hdfs dfs -chgrp prac
-chgrp: Not enough arguments: expected 2 but got 1
Usage: hadoop fs [generic options] -chgrp [-R] GROUP PATH...
udera®
Bigger Questions
[Terminal]
cloudera@quickstart:...
[Cloudera QuickStart -...
[hello (~/Desktop) - ge...

Practical No. :- 2

Aim:- Map Reduce

1. Write a program in Map Reduce for WordCount operation.

2. Write a program in Map Reduce for Union operation.

Theory:-

1. WordCount Operation

The WordCount program calculates the frequency of each word in a text.

- **Map Phase:** Splits input lines into words, emitting <word, 1> pairs.
- **Shuffle and Sort Phase:** Groups all identical words together.
- **Reduce Phase:** Sums up counts for each word, producing <word, total_count>.
  Output: A list of words with their frequencies.
  Use Case: Text analysis, log analysis, and indexing.

2. Union Operation

The Union program combines records from two datasets into one.

- **Map Phase:** Reads records from both datasets, emitting each record as a key-value pair.
- **Shuffle and Sort Phase:** Prepares grouped records for merging.
- **Reduce Phase:** Outputs all records into a single dataset without removing duplicates.
  Output: A merged dataset.
  Use Case: Data integration and preprocessing.

Code:-

WordMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class WordMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
@Override
    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String line = value.toString();
        for (String word : line.split("\\W+")) {
            if (word.length() > 0) {
                context.write(new Text(word), new IntWritable(1));
            }
        }
    }
}
```

SumReducer.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class SumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
```

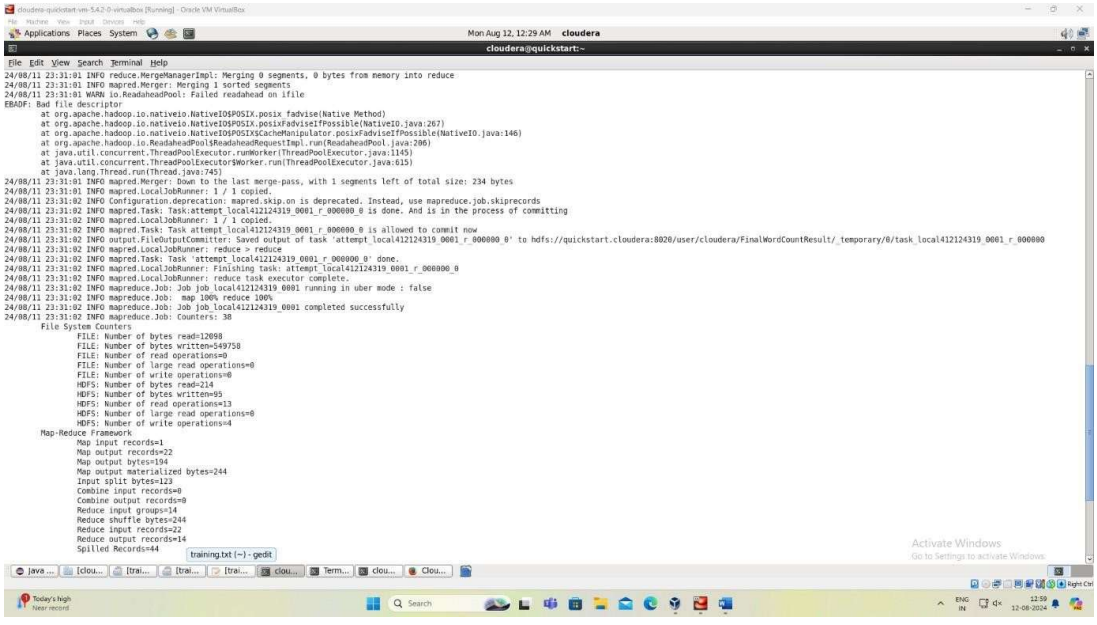
```
@Override
   public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
InterruptedException {
      int wordCount = 0;
      for (IntWritable value : values) {
         wordCount += value.get();
      }
      context.write(key, new IntWritable(wordCount));
 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

WordCount.java

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.Job;

public class WordCount {
   public static void main(String[] args) throws Exception {
      if (args.length != 2) {
        System.out.printf("Usage: WordCount <input dir> <output dir>\n");
        System.exit(-1);
      }
      Job job = new Job();

      job.setJarByClass(WordCount.class);
      job.setJobName("Word Count");
      FileInputFormat.setInputPaths(job, new Path(args[0]));
      FileOutputFormat.setOutputPath(job, new Path(args[1]));
```

```
        job.setMapperClass(WordMapper.class);
        job.setReducerClass(SumReducer.class);

        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(IntWritable.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        boolean success = job.waitForCompletion(true);
        System.exit(success ? 0 : 1);

    }
}
```

Output:-



```
                HDFS: Number of bytes read=314
                HDFS: Number of bytes written=200
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=283776
                Total time spent by all reduces in occupied slots (ms)=322304
                Total time spent by all map tasks (ms)=2217
                Total time spent by all reduce tasks (ms)=2518
                Total vcore-seconds taken by all map tasks=2217
                Total vcore-seconds taken by all reduce tasks=2518
                Total megabyte-seconds taken by all map tasks=283776
                Total megabyte-seconds taken by all reduce tasks=322304
        Map-Reduce Framework
                Map input records=1
                Map output records=31
                Map output bytes=306
                Map output materialized bytes=269
                Input split bytes=129
                Combine input records=0
                Combine output records=0
                Reduce input groups=24
                Reduce shuffle bytes=269
                Reduce input records=31
                Reduce output records=24
                Spilled Records=62
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=41
                CPU time spent (ms)=500
                Physical memory (bytes) snapshot=258859008
                Virtual memory (bytes) snapshot=1401176064
                Total committed heap usage (bytes)=101449728
        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=185
        File Output Format Counters
                Bytes Written=200
[cloudera@quickstart ~]$
```

```
[cloudera@quickstart ~]$ hdfs dfs -cat FinalWordCountResult/part-r-00000
19      4
As      1
COVID   4
The     1
based   1
emails  1
has     1
home    1
in      1
led     1
lured   1
pandemic        1
paradigm        1
people  1
phishing        1
ransomware      1
security        1
shifted 1
the     1
themed  1
through 1
to      2
weaker  1
workplace       1
```

Code:-

AvgMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.FloatWritable;
import  org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class AvgMapper extends Mapper<LongWritable, Text, Text, FloatWritable> {
@Override
    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
       String line = value.toString();

       for (String word : line.split("\\W+")) {
          if (word.length() > 0) {
             context.write(new Text(word), new FloatWritable(1));
          }
       }
   }
}
```

AvgReducer.java

```
import java.io.IOException;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class AvgReducer extends Reducer<Text, FloatWritable, Text, FloatWritable> {
   float wordCount = 0.0f;
   float sum=0.0f;
```

```
@Override

   public void reduce(Text key, Iterable<FloatWritable> values, Context context) throws
IOException, InterruptedException {


       for (FloatWritable value : values) {
          wordCount += value.get();


       }
       sum=sum+1;


       context.write(key, new FloatWritable(wordCount/sum));
   }
}
```

AvgDriver.java

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.Job;

public class AvgDirver {
   public static void main(String[] args) throws Exception {
      if (args.length != 2) {
        System.out.printf("Usage: WordCount <input dir> <output dir>\n");
        System.exit(-1);
       }
       Job job = new Job();
```

```
        job.setJarByClass(AvgDirver.class);
        job.setJobName("Word Count");
        FileInputFormat.setInputPaths(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        job.setMapperClass(AvgMapper.class);
        job.setReducerClass(AvgReducer.class);

        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(FloatWritable.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(FloatWritable.class);

        boolean success = job.waitForCompletion(true);
        System.exit(success ? 0 : 1);

    }
}
```

Output:-

```
[cloudera@quickstart ~]$ hdfs dfs -cat prac2Result/part-r-00000
19      4.0
As      2.5
COVID   3.0
The     2.5
based   2.2
emails  2.0
has     1.8571428
home    1.75
in      1.6666666
led     1.6
lured   1.5454545
pandemic        1.5
paradigm        1.4615384
people  1.4285715
phishing        1.4
ransomware      1.375
security        1.3529412
shifted 1.3333334
the     1.3157895
themed  1.3
through 1.2857143
to      1.3181819
weaker  1.3043479
workplace       1.2916666
```

Practical No. :- 3

Aim:- Map Reduce

1. Write a program in Map Reduce for Intersection operation.

2. Write a program in Map Reduce for Matrix Multiplication.

Theory:-

1. Intersection Operation

The Intersection program identifies common records between two datasets.

- **Map Phase:** Reads records from both datasets, emitting each record as a key with its dataset identifier as the value.
- **Shuffle and Sort Phase:** Groups records with the same key across datasets.
- **Reduce Phase:** Outputs records that appear in both datasets, based on identifiers from both sources.
  Output: Common records between the two datasets.
  Use Case: Useful for data comparison and filtering common entities.

2. Matrix Multiplication

Matrix Multiplication computes the product of two matrices.

- **Map Phase:** Emits key-value pairs where the key is the target cell (i, j), and the value includes the element's row/column data.
- **Shuffle and Sort Phase:** Groups all values for a target cell.
- **Reduce Phase:** Multiplies and sums relevant elements to compute the value for each cell in the result matrix.
  Output: The resultant matrix.
  Use Case: Applied in scientific computing and graph processing.

Code:-

Main Class.java

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.Job;

public class testdriver {
   public static void main(String[] args) throws Exception {
     if (args.length != 2) {
       System.out.printf("Usage: WordCount <input dir> <output dir>\n");
       System.exit(-1);
            }
            Job job = new Job();

            job.setJarByClass(testdriver.class);
            job.setJobName("Word Count");
            FileInputFormat.setInputPaths(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));

            job.setMapperClass(testmap.class);
            job.setReducerClass(testreduce.class);

            job.setMapOutputKeyClass(IntWritable.class);
            job.setMapOutputValueClass(IntWritable.class);

            job.setOutputKeyClass(IntWritable.class);
            job.setOutputValueClass(IntWritable.class);
```

```
        boolean success = job.waitForCompletion(true);
        System.exit(success ? 0 : 1);

    }
}
```

Mapper Class.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class testmap extends Mapper<LongWritable, Text, IntWritable,
IntWritable> {
@Override
    public void map(LongWritable key, Text value, Context context)
throws IOException, InterruptedException {
      String line = value.toString();
        String[] tokens = line.split(","); // This is the delimiter between
        int keypart = Integer.parseInt(tokens[0]);
        int valuePart = Integer.parseInt(tokens[1]);
        context.write(new IntWritable(valuePart), new IntWritable(keypart));
    }
}
```

Reducer Class.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapreduce.Reducer;
public class testreduce extends Reducer<IntWritable, IntWritable,
IntWritable, IntWritable> {
@Override
      public void reduce(IntWritable key, Iterable<IntWritable> values,
Context context)  throws IOException, InterruptedException {

      for (IntWritable value : values) {

         context.write(value,key);

        }
}
}
```

Output:-

```
-rw-r--r--   1 cloudera cloudera       1490 2024-08-19 23:13 /user/cloudera/sortResult1/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat /user/cloudera/sortResult1/part-r-00000
814711606       124
759224212       171
617667090       273
955357205       282
513417565       522
816200339       673
450563752       682
512878119       888
621386563       948
902102267       962
837559306       1266
860673511       1273
728815257       1485
972292029       1673
505716836       1705
341417157       1779
898523128       1815
423331391       2021
994022214       2117
406502997       2125
142278373       2187
795490682       2225
385383069       2269
886494815       2370
963881480       2804
868214595       2847
```

Code:-

Main class.java:-

```
import  org.apache.hadoop.conf.*;
import org.apache.hadoop.fs.Path;
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.TextOutputFormat;

public class MMDriver {

   public static void main(String[] args) throws Exception {
      if (args.length != 2) {
         System.err.println("Usage: MatrixMultiply <in_dir> <out_dir>");
         System.exit(2);
      }
      Configuration conf = new Configuration();
      // M is an m-by-n matrix; N is an n-by-p matrix.
      conf.set("m", "1000");
      conf.set("n", "100");
      conf.set("p", "1000");

      Job job = new Job(conf, "MatrixMultiply");


      job.setJarByClass(MMDriver.class);
      job.setOutputKeyClass(Text.class);
      job.setOutputValueClass(Text.class);

      job.setMapperClass(MMMap.class);
      job.setReducerClass(MMReduce.class);

      job.setInputFormatClass(TextInputFormat.class);
      job.setOutputFormatClass(TextOutputFormat.class);

      FileInputFormat.addInputPath(job, new Path(args[0]));
      FileOutputFormat.setOutputPath(job, new Path(args[1]));

      job.waitForCompletion(true);
   }
}
```

Mapper class.java

```
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

import java.io.IOException;

public class MMMap
  extends Mapper<LongWritable, Text, Text, Text> {
      @Override
      public void map(LongWritable key, Text value, Context context)
                  throws IOException, InterruptedException {
            Configuration conf = context.getConfiguration();
            int m = Integer.parseInt(conf.get("m"));
            int p = Integer.parseInt(conf.get("p"));
            String line = value.toString();
            // (M, i, j, Mij);
            String[] indicesAndValue = line.split(",");
            Text outputKey = new Text();
            Text outputValue = new Text();
            if (indicesAndValue[0].equals("M")) {
                  for (int k = 0; k < p; k++) {
                        outputKey.set(indicesAndValue[1] + "," + k);
                        // outputKey.set(i,k);
                        outputValue.set(indicesAndValue[0] + "," + indicesAndValue[2]
                                    + "," + indicesAndValue[3]);
                        // outputValue.set(M,j,Mij);
                        context.write(outputKey, outputValue);
                  }
            } else {
                  // (N, j, k, Njk);
                  for (int i = 0; i < m; i++) {
                        outputKey.set(i + "," + indicesAndValue[2]);
                        outputValue.set("N," + indicesAndValue[1] + ","
                                    + indicesAndValue[3]);
                        context.write(outputKey, outputValue);
                  }
            }
      }
}
```

Reducer.java

```
import org.apache.hadoop.io.Text;
import java.io.IOException;
```

```
import java.util.HashMap;

public class MMReduce
  extends org.apache.hadoop.mapreduce.Reducer<Text, Text, Text, Text> {
      @Override
      public void reduce(Text key, Iterable<Text> values, Context context)
              throws IOException, InterruptedException {
          String[] value;
          //key=(i,k),
          //Values = [(M/N,j,V/W),..]
          HashMap<Integer, Float> hashA = new HashMap<Integer, Float>();
          HashMap<Integer, Float> hashB = new HashMap<Integer, Float>();
          for (Text val : values) {
              value = val.toString().split(",");
              if (value[0].equals("M")) {
                  hashA.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
              } else {
                  hashB.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
              }
          }
          int n = Integer.parseInt(context.getConfiguration().get("n"));
          float result = 0.0f;
          float m_ij;
          float n_jk;
          for (int j = 0; j < n; j++) {
              m_ij = hashA.containsKey(j) ? hashA.get(j) : 0.0f;
              n_jk = hashB.containsKey(j) ? hashB.get(j) : 0.0f;
              result += m_ij * n_jk;
          }
          if (result != 0.0f) {
              context.write(null,
                      new Text(key.toString() + "," + Float.toString(result)));
          }
      }
}
```

Output:-

```
[cloudera@quickstart ~]$ hdfs dfs -cat prac4Result/part-r-00000
0,0,19.0
0,1,22.0
1,0,43.0
1,1,50.0
```

Practical No. :- 4

Aim:- MongoDB: Installation, Sample Database Creation, Query the Sample Database using MongoDB querying commands, Create Collection, Insert Document, Query Document, Delete Document, Indexing.

Theory:-

1. Installation

- Download MongoDB from the official website and install it for your operating system.
- Start the MongoDB server using mongod, and connect to it using the MongoDB shell (mongo).

2. Sample Database Creation

- Use the command use <database_name> to create or switch to a database. MongoDB automatically creates the database upon the first document insertion.

3. Query the Sample Database

- Use show dbs to list databases and db.collection_name.find() to query data within a collection.

4. Create Collection

- Collections are created automatically upon inserting documents or manually with: db.createCollection('<collection_name>').

5. Insert Document

- Add a document to a collection using: db.collection_name.insertOne({key: value}) or insertMany() for multiple documents.

6. Query Document

- Retrieve documents with: db.collection_name.find({key: value}).

7. Delete Document

- Remove documents using: db.collection_name.deleteOne({key: value}) or deleteMany().

8. Indexing

- Optimize query performance with indexes: db.collection_name.createIndex({key: 1}).

Code:-

1. Start MongoDB Shell:-

   Mongo

2. Create a Sample Database:-

   use SampleDB

3. Create a Collection:-

   db.createCollection("Students").

4. Insert a Document:-

```
db.Students.insertOne({
  student_id: 1,
  name: "John Doe",
  age: 20,
  course: "Computer Science",
  grades: { math: 85, physics: 90 },
  enrolled_on: new Date()
})
```

5. Query the Document:-

```
db.Students.find({ name: "John Doe" })
```

6. Update a Document:-

```
db.Students.updateOne(
  { student_id: 1 },
  { $set: { age: 21, "grades.math": 90 } }
)
```

7. Delete a Document:-

```
db.Students.deleteOne({ student_id: 1 })
```

8. Create an Index:-

```
db.Students.createIndex({ name: 1 })
```

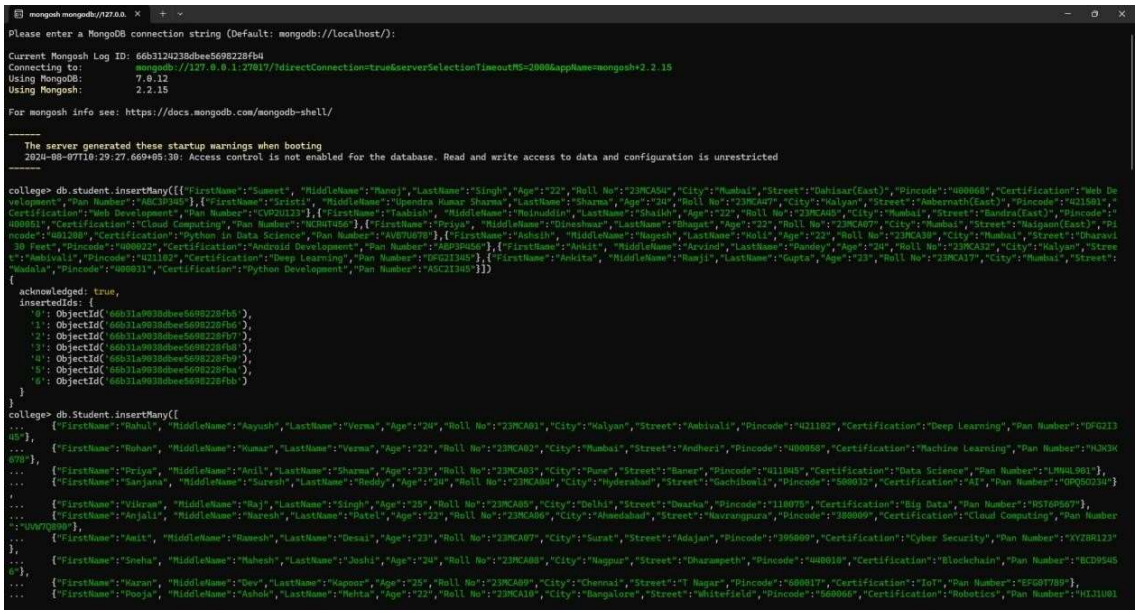
9. Query with the Index:-

```
db.Students.find({ name: "John Doe" }).explain("executionStats")
```

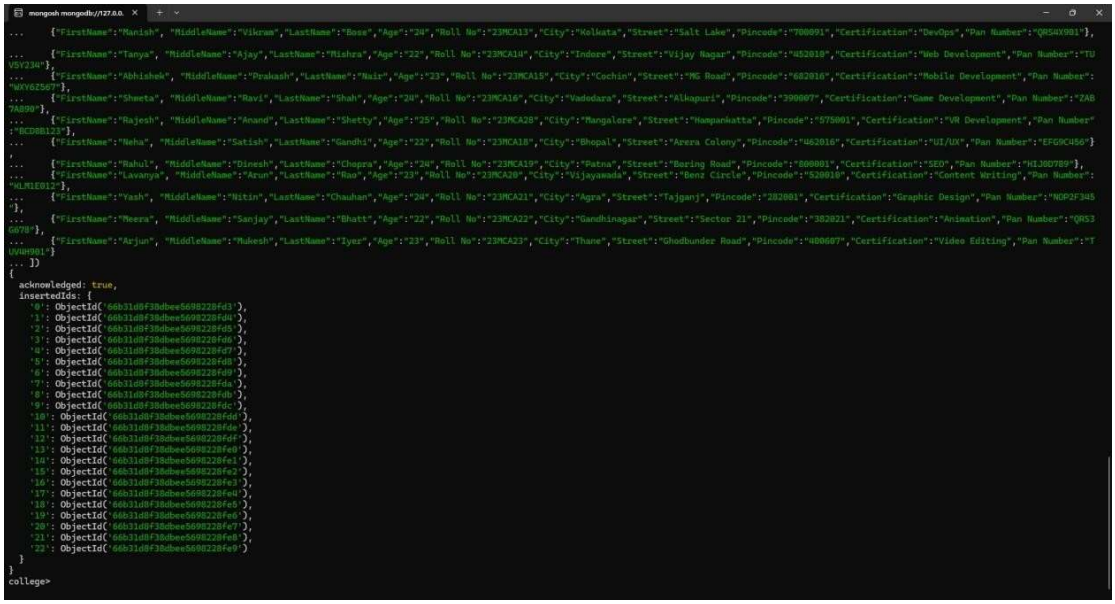
Output:-

1. Create Command:-

- Insert 30 Entries:-

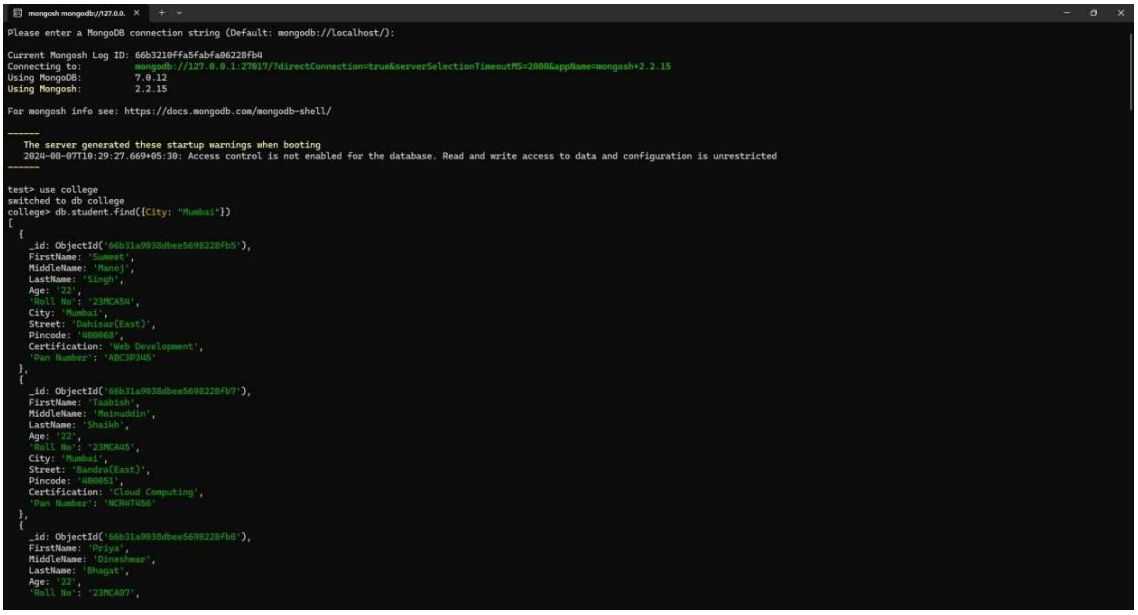


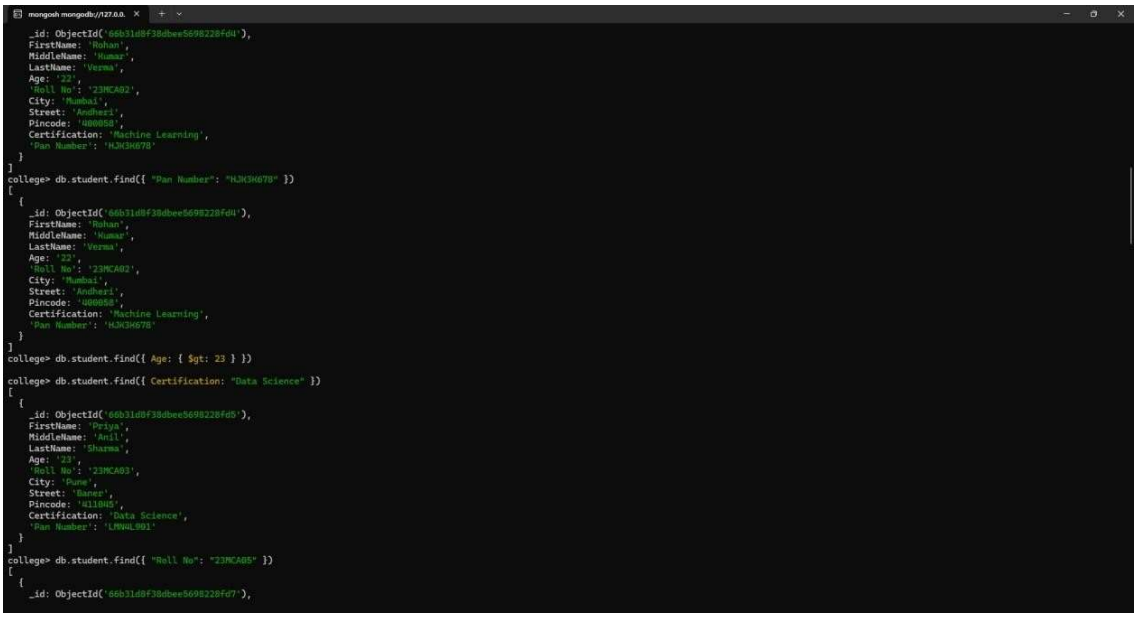

mongosh mongodb://127.0.0.
college> db.student.insertMany([
... {"FirstName":"Rahul", "MiddleName":"Aayush","LastName":"Verma","Age":"24","Roll No":"23MCA01","City":"Kalyan","Street":"Ambivali","Pincode":"421102","Certification":"Deep Learning","Pan Number":"DFG2I345"},
... {"FirstName":"Rohan", "MiddleName":"Kumar","LastName":"Verma","Age":"22","Roll No":"23MCA02","City":"Mumbai","Street":"Andheri","Pincode":"400058","Certification":"Machine Learning","Pan Number":"KJH3K678"},
... {"FirstName":"Priya", "MiddleName":"Anil","LastName":"Sharma","Age":"23","Roll No":"23MCA03","City":"Pune","Street":"Baner","Pincode":"411045","Certification":"Data Science","Pan Number":"LMWL901"},
... {"FirstName":"Sanjana", "MiddleName":"Suresh","LastName":"Reddy","Age":"24","Roll No":"23MCA04","City":"Hyderabad","Street":"Gachibowli","Pincode":"500032","Certification":"AI","Pan Number":"OPQ56234"},
... {"FirstName":"Vikram", "MiddleName":"Raj","LastName":"Singh","Age":"25","Roll No":"23MCA05","City":"Delhi","Street":"Dwarka","Pincode":"110075","Certification":"Big Data","Pan Number":"RST6P567"},
... {"FirstName":"Anjali", "MiddleName":"Naresh","LastName":"Patel","Age":"22","Roll No":"23MCA06","City":"Ahmedabad","Street":"Navrangpura","Pincode":"380009","Certification":"Cloud Computing","Pan Number":"UVW7Q890"},
... {"FirstName":"Amit", "MiddleName":"Ramesh","LastName":"Desai","Age":"23","Roll No":"23MCA07","City":"Surat","Street":"Adajan","Pincode":"395009","Certification":"Cyber Security","Pan Number":"XYZ8R123"},
... {"FirstName":"Sneha", "MiddleName":"Mahesh","LastName":"Joshi","Age":"24","Roll No":"23MCA08","City":"Nagpur","Street":"Dharampeth","Pincode":"440010","Certification":"Blockchain","Pan Number":"BCD95456"},
... {"FirstName":"Karan", "MiddleName":"Dev","LastName":"Kapoor","Age":"25","Roll No":"23MCA09","City":"Chennai","Street":"T Nagar","Pincode":"600017","Certification":"IoT","Pan Number":"EFG0T789"},
... {"FirstName":"Pooja", "MiddleName":"Ashok","LastName":"Mehta","Age":"22","Roll No":"23MCA10","City":"Bangalore","Street":"Whitefield","Pincode":"560066","Certification":"Robotics","Pan Number":"HIJ1U812"},
... {"FirstName":"Nikhil", "MiddleName":"Sunil","LastName":"Jain","Age":"24","Roll No":"23MCA11","City":"Jaipur","Street":"Malviya Nagar","Pincode":"302017","Certification":"AR/VR","Pan Number":"KLM2V345"},
... {"FirstName":"Divya", "MiddleName":"Rajesh","LastName":"Agarwal","Age":"23","Roll No":"23MCA12","City":"Lucknow","Street":"Gomti Nagar","Pincode":"226010","Certification":"Ethical Hacking","Pan Number":"NOP3W678"},
... {"FirstName":"Manish", "MiddleName":"Vikram","LastName":"Bose","Age":"24","Roll No":"23MCA13","City":"Kolkata","Street":"Salt Lake","Pincode":"700091","Certification":"DevOps","Pan Number":"QRS4X901"},
... {"FirstName":"Tanya", "MiddleName":"Ajay","LastName":"Mishra","Age":"22","Roll No":"23MCA14","City":"Indore","Street":"Vijay Nagar","Pincode":"452010","Certification":"Web Development","Pan Number":"TUV5Y234"},
... {"FirstName":"Abhishek", "MiddleName":"Prakash","LastName":"Nair","Age":"23","Roll No":"23MCA15","City":"Cochin","Street":"MG Road","Pincode":"682016","Certification":"Mobile Development","Pan Number":"WXY6Z567"},
... {"FirstName":"Shweta", "MiddleName":"Ravi","LastName":"Shah","Age":"24","Roll No":"23MCA16","City":"Vadodara","Street":"Alkapuri","Pincode":"390007","Certification":"Game Development","Pan Number":"ZAB7A890"},
... {"FirstName":"Rajesh", "MiddleName":"Anand","LastName":"Shetty","Age":"25","Roll No":"23MCA28","City":"Mangalore","Street":"Hampankatta","Pincode":"575001","Certification":"VR Development","Pan Number":"BCD8B123"},
... {"FirstName":"Neha", "MiddleName":"Satish","LastName":"Gandhi","Age":"22","Roll No":"23MCA18","City":"Bhopal","Street":"Arera Colony","Pincode":"462016","Certification":"UI/UX","Pan Number":"EFG9C456"},
... {"FirstName":"Rahul", "MiddleName":"Dinesh","LastName":"Chopra","Age":"20","Roll No":"23MCA19","City":"Patna","Street":"Boring Road","Pincode":"800001","Certification":"SED","Pan Number":"HIJ0D789"},
... {"FirstName":"Lavanya", "MiddleName":"Arun","LastName":"Rao","Age":"23","Roll No":"23MCA20","City":"Vijayawada","Street":"Benz Circle","Pincode":"520010","Certification":"Content Writing","Pan Number":"KLM1E012"},
... {"FirstName":"Yash", "MiddleName":"Nitin","LastName":"Chauhan","Age":"24","Roll No":"23MCA21","City":"Agra","Street":"Tajganj","Pincode":"282001","Certification":"Graphic Design","Pan Number":"NOP2F345"},
... {"FirstName":"Meera", "MiddleName":"Sanjay","LastName":"Bhatt","Age":"22","Roll No":"23MCA22","City":"Gandhinagar","Street":"Sector 21","Pincode":"382021","Certification":"Animation","Pan Number":"QRS3G678"},
... {"FirstName":"Arjun", "MiddleName":"Mukesh","LastName":"Iyer","Age":"23","Roll No":"23MCA23","City":"Thane","Street":"Ghodbunder Road","Pincode":"400607","Certification":"Video Editing","Pan Number":"TUV8H901"}
...])

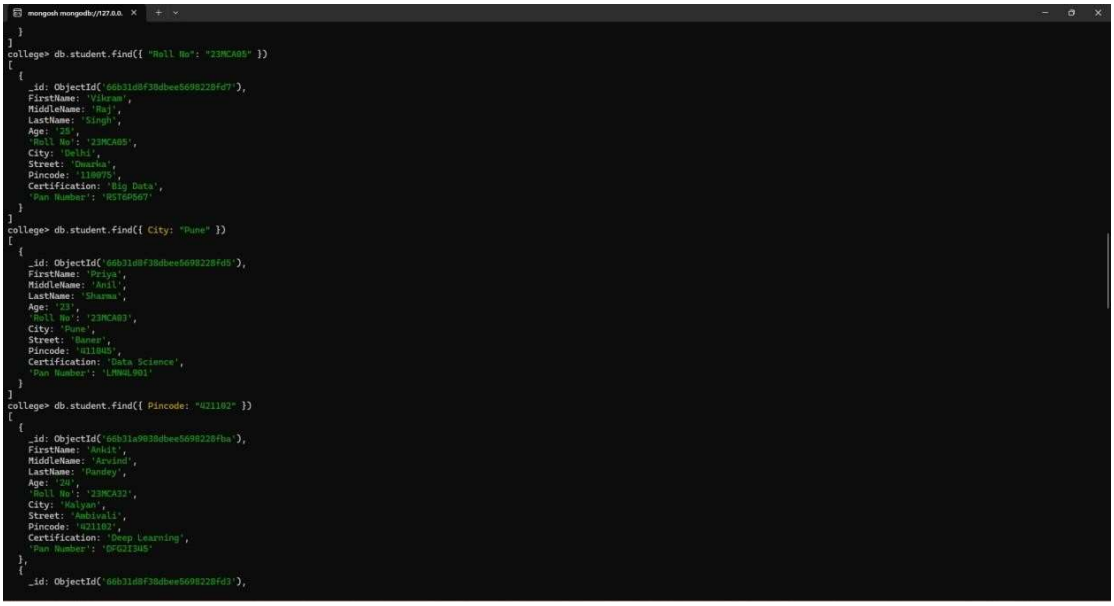

mongosh mongodb://127.0.0.
... {"FirstName":"Manish", "MiddleName":"Vikram","LastName":"Bose","Age":"24","Roll No":"23MCA13","City":"Kolkata","Street":"Salt Lake","Pincode":"700091","Certification":"DevOps","Pan Number":"QRS4X901"},
... {"FirstName":"Tanya", "MiddleName":"Ajay","LastName":"Mishra","Age":"22","Roll No":"23MCA14","City":"Indore","Street":"Vijay Nagar","Pincode":"452010","Certification":"Web Development","Pan Number":"TUV5Y234"},
... {"FirstName":"Abhishek", "MiddleName":"Prakash","LastName":"Nair","Age":"23","Roll No":"23MCA15","City":"Cochin","Street":"MG Road","Pincode":"682016","Certification":"Mobile Development","Pan Number":"WXY6Z567"},
... {"FirstName":"Shweta", "MiddleName":"Ravi","LastName":"Shah","Age":"24","Roll No":"23MCA16","City":"Vadodara","Street":"Alkapuri","Pincode":"390007","Certification":"Game Development","Pan Number":"ZAB7A890"},
... {"FirstName":"Rajesh", "MiddleName":"Anand","LastName":"Shetty","Age":"25","Roll No":"23MCA28","City":"Mangalore","Street":"Hampankatta","Pincode":"575001","Certification":"VR Development","Pan Number":"BCD8B123"},
... {"FirstName":"Neha", "MiddleName":"Satish","LastName":"Gandhi","Age":"22","Roll No":"23MCA18","City":"Bhopal","Street":"Arera Colony","Pincode":"462016","Certification":"UI/UX","Pan Number":"EFG9C456"},
... {"FirstName":"Rahul", "MiddleName":"Dinesh","LastName":"Chopra","Age":"20","Roll No":"23MCA19","City":"Patna","Street":"Boring Road","Pincode":"800001","Certification":"SED","Pan Number":"HIJ0D789"},
... {"FirstName":"Lavanya", "MiddleName":"Arun","LastName":"Rao","Age":"23","Roll No":"23MCA20","City":"Vijayawada","Street":"Benz Circle","Pincode":"520010","Certification":"Content Writing","Pan Number":"KLM1E012"},
... {"FirstName":"Yash", "MiddleName":"Nitin","LastName":"Chauhan","Age":"24","Roll No":"23MCA21","City":"Agra","Street":"Tajganj","Pincode":"282001","Certification":"Graphic Design","Pan Number":"NOP2F345"},
... {"FirstName":"Meera", "MiddleName":"Sanjay","LastName":"Bhatt","Age":"22","Roll No":"23MCA22","City":"Gandhinagar","Street":"Sector 21","Pincode":"382021","Certification":"Animation","Pan Number":"QRS3G678"},
... {"FirstName":"Arjun", "MiddleName":"Mukesh","LastName":"Iyer","Age":"23","Roll No":"23MCA23","City":"Thane","Street":"Ghodbunder Road","Pincode":"400607","Certification":"Video Editing","Pan Number":"TUV8H901"}
...])
{
acknowledged: true,
insertedIds: {
'0': ObjectId('66b31d0f38dbee5698228fd3'),
'1': ObjectId('66b31d0f38dbee5698228fd4'),
'2': ObjectId('66b31d0f38dbee5698228fd5'),
'3': ObjectId('66b31d0f38dbee5698228fd6'),
'4': ObjectId('66b31d0f38dbee5698228fd7'),
'5': ObjectId('66b31d0f38dbee5698228fd8'),
'6': ObjectId('66b31d0f38dbee5698228fd9'),
'7': ObjectId('66b31d0f38dbee5698228fda'),
'8': ObjectId('66b31d0f38dbee5698228fdb'),
'9': ObjectId('66b31d0f38dbee5698228fdc'),
'10': ObjectId('66b31d0f38dbee5698228fdd'),
'11': ObjectId('66b31d0f38dbee5698228fde'),
'12': ObjectId('66b31d0f38dbee5698228fdf'),
'13': ObjectId('66b31d0f38dbee5698228fe0'),
'14': ObjectId('66b31d0f38dbee5698228fe1'),
'15': ObjectId('66b31d0f38dbee5698228fe2'),
'16': ObjectId('66b31d0f38dbee5698228fe3'),
'17': ObjectId('66b31d0f38dbee5698228fe4'),
'18': ObjectId('66b31d0f38dbee5698228fe5'),
'19': ObjectId('66b31d0f38dbee5698228fe6'),
'20': ObjectId('66b31d0f38dbee5698228fe7'),
'21': ObjectId('66b31d0f38dbee5698228fe8'),
'22': ObjectId('66b31d0f38dbee5698228fe9')
}
}
college>

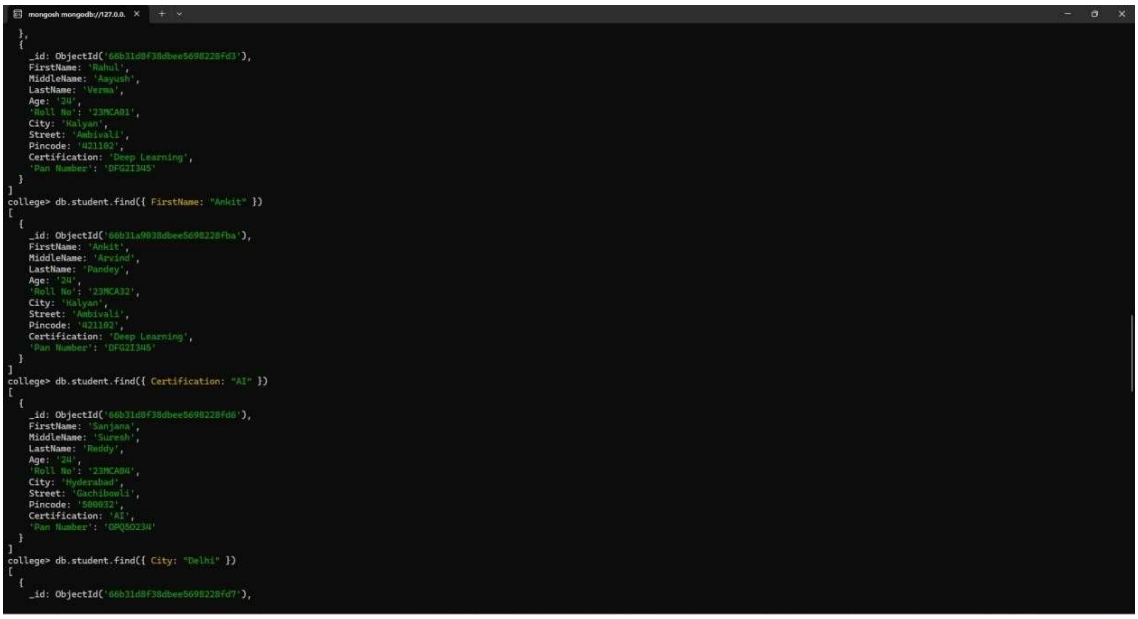
2. Read Command:-

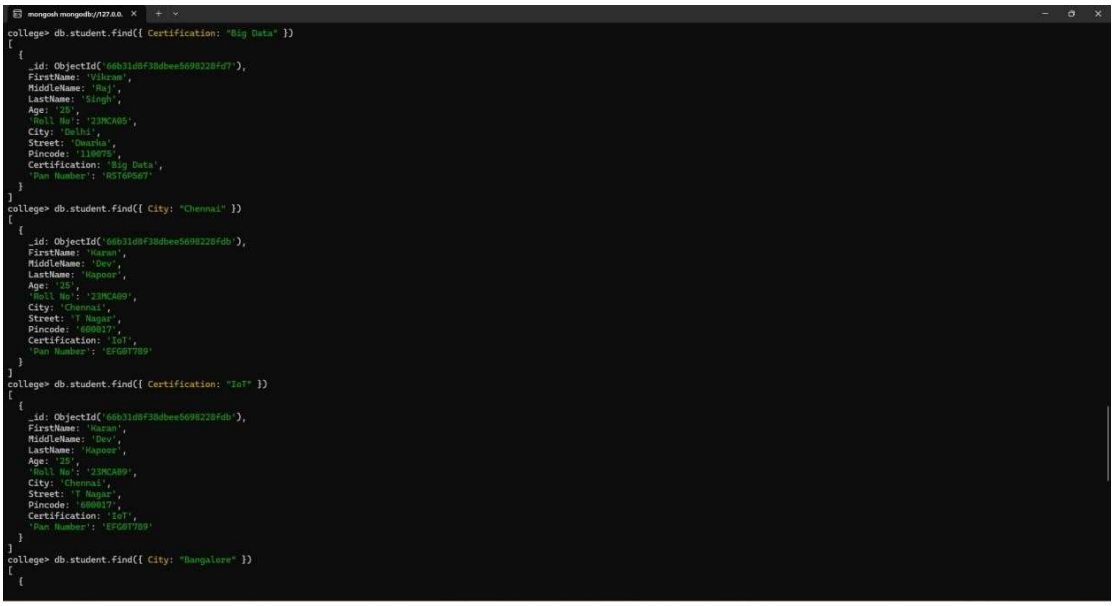
- Read Operations (15 Examples):-

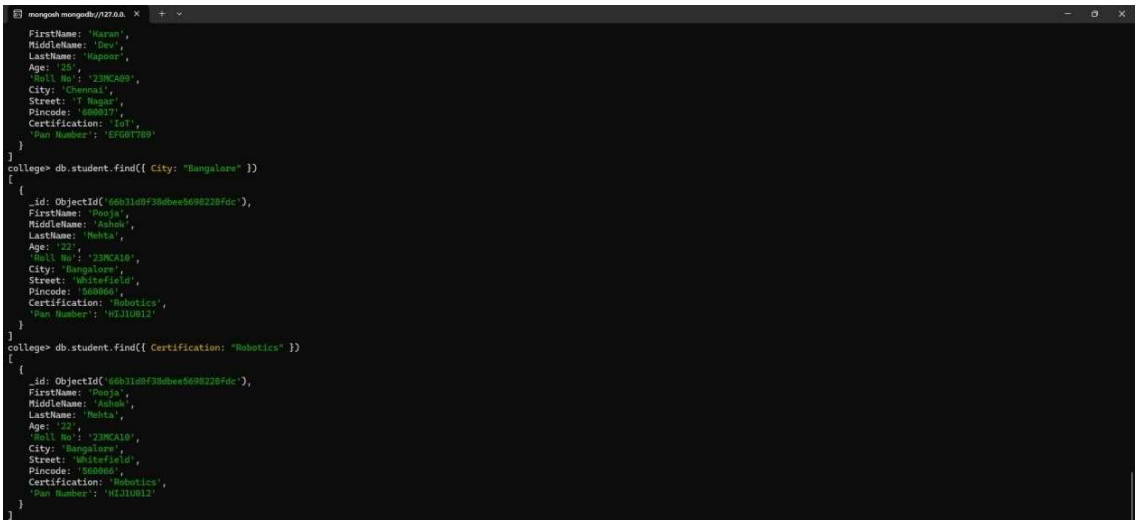



mongosh mongodb://127.0.0.
_id: ObjectId('66b31d8f38dbee5698228fd4'),
FirstName: 'Rohan',
MiddleName: 'Kumar',
LastName: 'Verma',
Age: '22',
'Roll No': '23MCA02',
City: 'Mumbai',
Street: 'Andheri',
Pincode: '400058',
Certification: 'Machine Learning',
'Pan Number': 'HJK3H678'
}
]
college> db.student.find({ "Pan Number": "HJK3H678" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd4'),
FirstName: 'Rohan',
MiddleName: 'Kumar',
LastName: 'Verma',
Age: '22',
'Roll No': '23MCA02',
City: 'Mumbai',
Street: 'Andheri',
Pincode: '400058',
Certification: 'Machine Learning',
'Pan Number': 'HJK3H678'
}
]
college> db.student.find({ Age: { $gt: 23 } })
college> db.student.find({ Certification: "Data Science" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd5'),
FirstName: 'Priya',
MiddleName: 'Anil',
LastName: 'Sharma',
Age: '23',
'Roll No': '23MCA03',
City: 'Pune',
Street: 'Baner',
Pincode: '411045',
Certification: 'Data Science',
'Pan Number': 'LMN4L901'
}
]
college> db.student.find({ "Roll No": "23MCA05" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd7'),


}
]
college> db.student.find({ "Roll No": "23MCA05" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd7'),
FirstName: 'Vikram',
MiddleName: 'Raj',
LastName: 'Singh',
Age: '25',
'Roll No': '23MCA05',
City: 'Delhi',
Street: 'Dwarka',
Pincode: '110075',
Certification: 'Big Data',
'Pan Number': 'RST6P567'
}
]
college> db.student.find({ City: "Pune" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd5'),
FirstName: 'Priya',
MiddleName: 'Anil',
LastName: 'Sharma',
Age: '23',
'Roll No': '23MCA03',
City: 'Pune',
Street: 'Baner',
Pincode: '411045',
Certification: 'Data Science',
'Pan Number': 'LMN4L901'
}
]
college> db.student.find({ Pincode: "421102" })
[
{
_id: ObjectId('66b31a9038dbee5698228fba'),
FirstName: 'Ankit',
MiddleName: 'Arvind',
LastName: 'Pandey',
Age: '24',
'Roll No': '23MCA32',
City: 'Kalyan',
Street: 'Ambivali',
Pincode: '421102',
Certification: 'Deep Learning',
'Pan Number': 'DFG2I3W5'
},
{
_id: ObjectId('66b31d8f38dbee5698228fd3'),

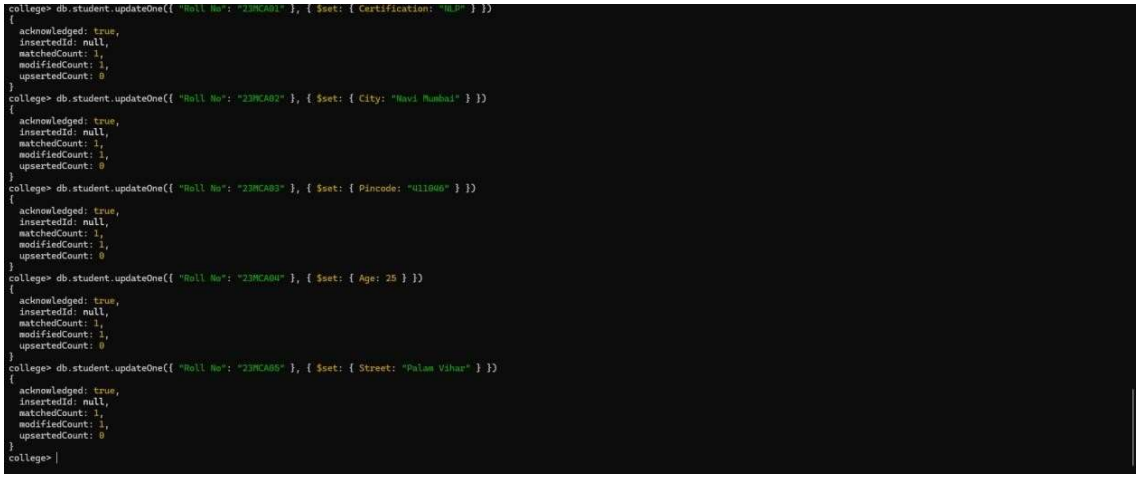

mongosh mongodb://127.0.0.
},
{
_id: ObjectId('66b31d8f38dbee5698228fd3'),
FirstName: 'Rahul',
MiddleName: 'Aayush',
LastName: 'Verma',
Age: '24',
'Roll No': '23MCA01',
City: 'Kalyan',
Street: 'Ambivali',
Pincode: '421102',
Certification: 'Deep Learning',
'Pan Number': 'DFG2I345'
}
]
college> db.student.find({ FirstName: "Ankit" })
[
{
_id: ObjectId('66b31a9038dbee5698228fba'),
FirstName: 'Ankit',
MiddleName: 'Arvind',
LastName: 'Pandey',
Age: '24',
'Roll No': '23MCA32',
City: 'Kalyan',
Street: 'Ambivali',
Pincode: '421102',
Certification: 'Deep Learning',
'Pan Number': 'DFG2I345'
}
]
college> db.student.find({ Certification: "AI" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd6'),
FirstName: 'Sanjana',
MiddleName: 'Suresh',
LastName: 'Reddy',
Age: '24',
'Roll No': '23MCA06',
City: 'Hyderabad',
Street: 'Gachibowli',
Pincode: '500032',
Certification: 'AI',
'Pan Number': 'GPQ5O234'
}
]
college> db.student.find({ City: "Delhi" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd7'),


mongosh mongodb://127.0.0.
college> db.student.find({ Certification: "Big Data" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fd7'),
FirstName: 'Vikram',
MiddleName: 'Raj',
LastName: 'Singh',
Age: '25',
'Roll No': '23MCA05',
City: 'Delhi',
Street: 'Dwarka',
Pincode: '110075',
Certification: 'Big Data',
'Pan Number': 'RST6P567'
}
]
college> db.student.find({ City: "Chennai" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fdb'),
FirstName: 'Karan',
MiddleName: 'Dev',
LastName: 'Kapoor',
Age: '25',
'Roll No': '23MCA09',
City: 'Chennai',
Street: 'T Nagar',
Pincode: '600017',
Certification: 'IoT',
'Pan Number': 'EFGBT789'
}
]
college> db.student.find({ Certification: "IoT" })
[
{
_id: ObjectId('66b31d8f38dbee5698228fdb'),
FirstName: 'Karan',
MiddleName: 'Dev',
LastName: 'Kapoor',
Age: '25',
'Roll No': '23MCA09',
City: 'Chennai',
Street: 'T Nagar',
Pincode: '600017',
Certification: 'IoT',
'Pan Number': 'EFGBT789'
}
]
college> db.student.find({ City: "Bangalore" })
[
{



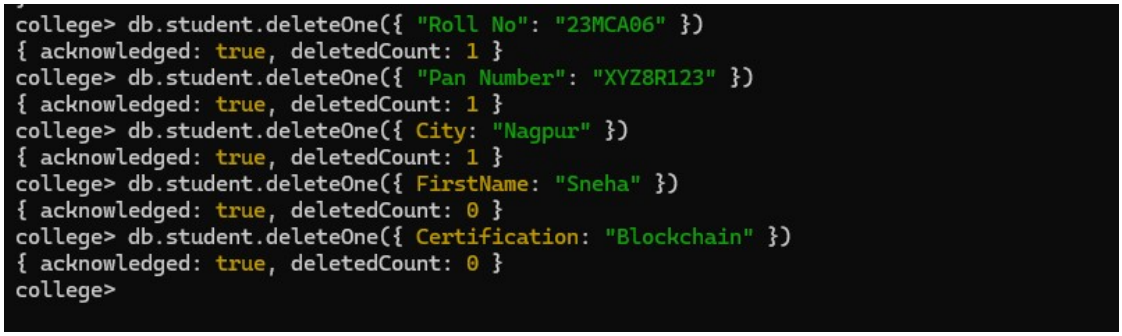
3. Update Command:-

- Update Operations (5 Examples):-



4. Delete Command:-

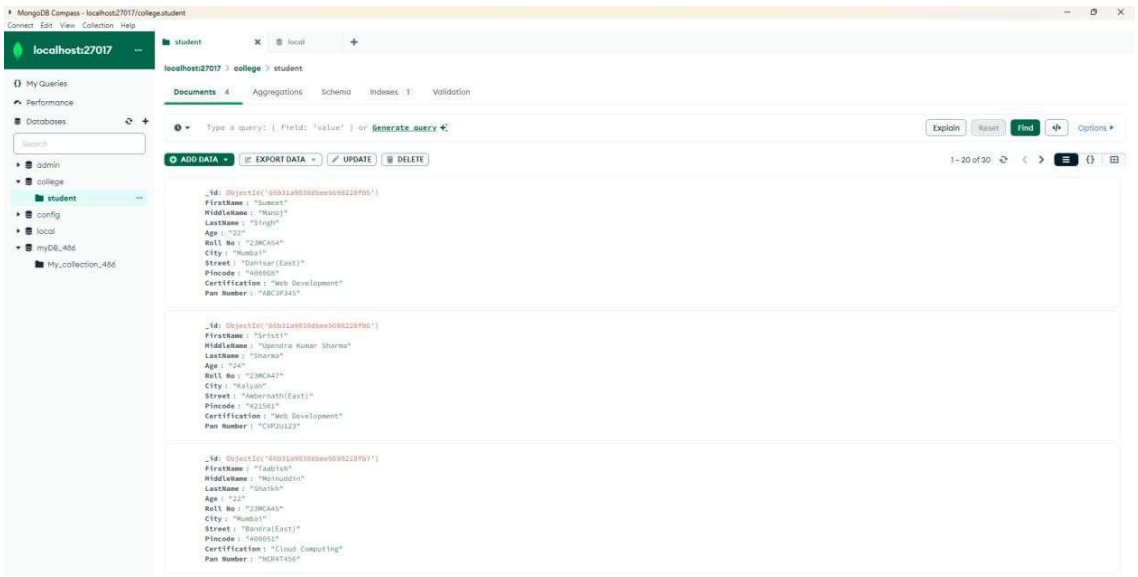
- Delete Operations (5 Examples):-

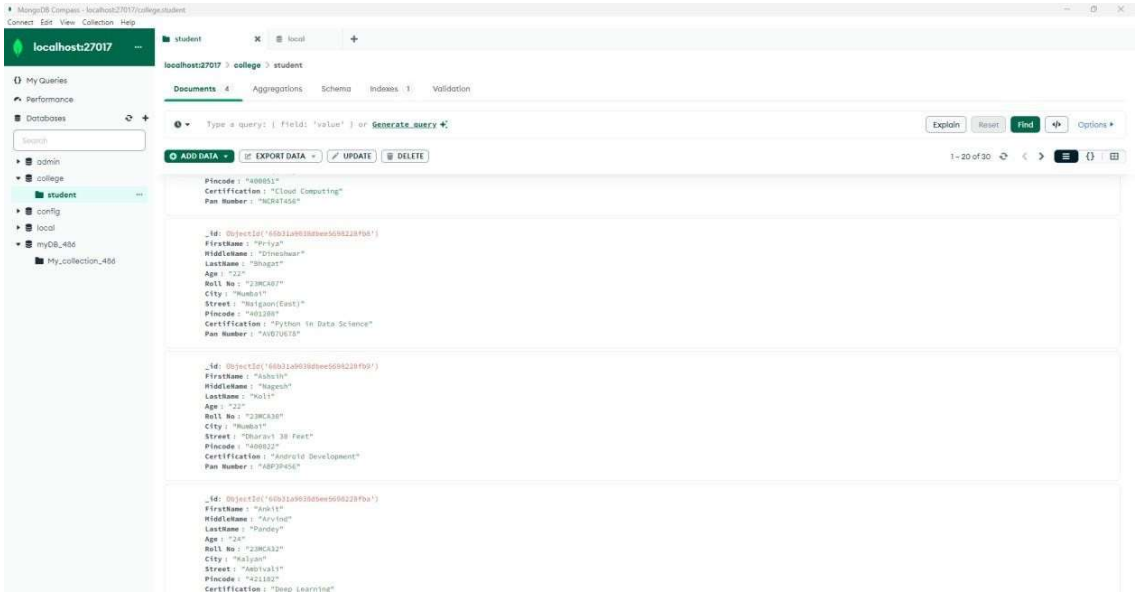


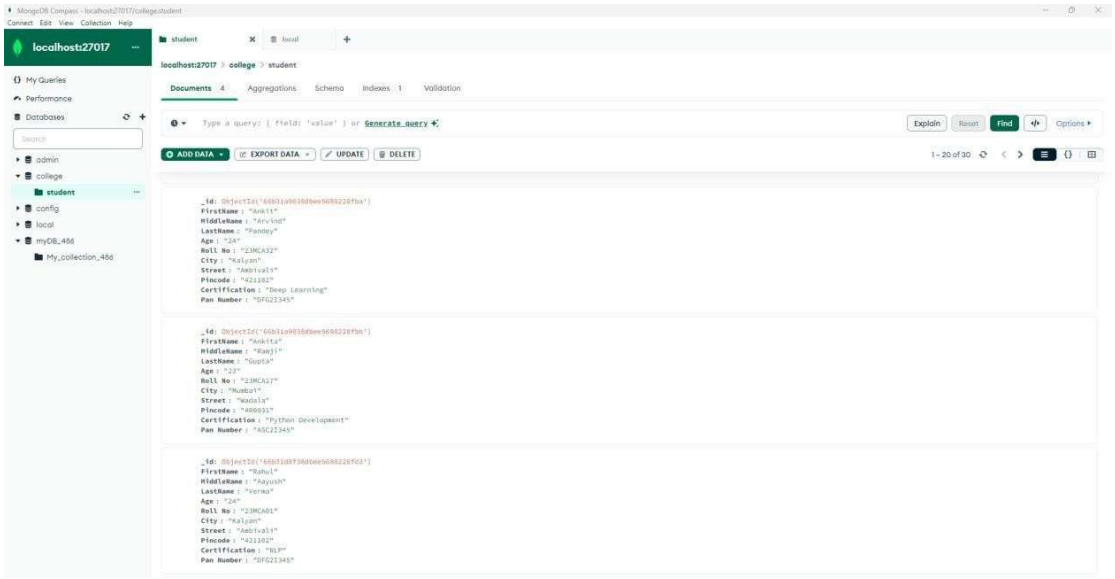
5. Indexing:-

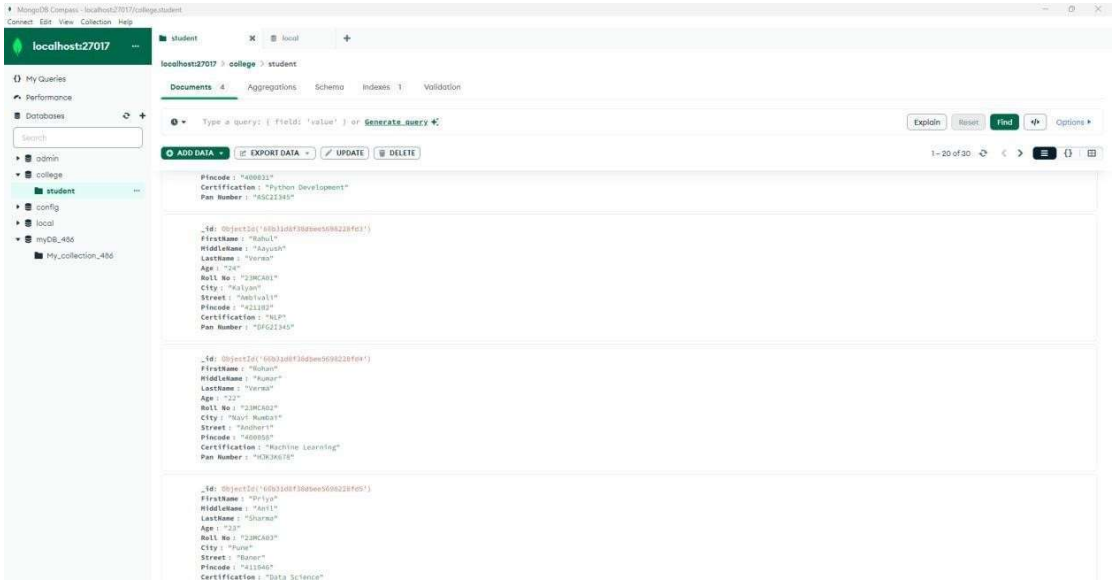
- Create Indexes:-

```
college> db.student.createIndex({ City: 1 })
City_1
college> db.Student.createIndex({ Age: 1 })
Age_1
college> db.Student.createIndex({ Certification: 1 })
Certification_1
college> db.Student.createIndex({ FirstName: 1, LastName: 1 })
FirstName_1_LastName_1
college> db.Student.createIndex({ City: 1, Certification: 1 })
City_1_Certification_1
college> db.Student.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { Age: 1 }, name: 'Age_1' },
  { v: 2, key: { Certification: 1 }, name: 'Certification_1' },
  {
    v: 2,
    key: { FirstName: 1, LastName: 1 },
    name: 'FirstName_1_LastName_1'
  },
  {
    v: 2,
    key: { City: 1, Certification: 1 },
    name: 'City_1_Certification_1'
  }
]
college>
```




localhost:27017
student
localhost:27017 > college > student
Documents 4
Aggregations
Schema
Indexes 1
Validation
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
1 - 20 of 30
My Queries
Performance
Databases
admin
college
student
config
local
myDB_486
My_collection_486


localhost:27017
student
localhost:27017 > college > student
Documents 4
Aggregations
Schema
Indexes 1
Validation
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
1 - 20 of 30
My Queries
Performance
Databases
admin
college
student
config
local
myDB_486
My_collection_486


MongoDB Compass - localhost:27017/college.student
Connect Edit View Collection Help
localhost:27017
My Queries
Performance
Databases
Search
admin
college
student
config
local
myDB_486
My_collection_486
student
local
localhost:27017 > college > student
Documents 4
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
1 - 20 of 30
Pincode : "400031"
Certification : "Python Development"
Pan Number : "ASC21345"
_id: ObjectId('69b31d0f30d3eee5698228fd')
firstName : "Rahul"
MiddleName : "Ayush"
LastName : "Verma"
Age : "24"
Roll No : "23MCA01"
City : "Kalyan"
Street : "Ambivali"
Pincode : "421102"
Certification : "NLP"
Pan Number : "DFG21345"
_id: ObjectId('69b31d0f30d3eee5698228fe')
firstName : "Rohan"
MiddleName : "Kumar"
LastName : "Verma"
Age : "22"
Roll No : "23MCA02"
City : "Navi Mumbai"
Street : "Andheri"
Pincode : "400058"
Certification : "Machine Learning"
Pan Number : "HJK3K678"
_id: ObjectId('69b31d0f30d3eee5698228ff')
firstName : "Priya"
MiddleName : "Anil"
LastName : "Sharma"
Age : "23"
Roll No : "23MCA03"
City : "Pune"
Street : "Baner"
Pincode : "411045"
Certification : "Data Science"

Practical No. :- 5

Aim:- Hive: Hive Data Types, Create Database & Table in Hive, Hive Partitioning, Hive Built-In Operators, Hive Built-In Functions, Hive Views and Indexes, HiveQL : Select Where, Select OrderBy, Select GroupBy, Select Joins.

Theory:-

1. Hive Data Types

Hive supports **Primitive Data Types** (e.g., INT, STRING, FLOAT, BOOLEAN) for basic data handling and **Complex Data Types** (e.g., ARRAY, MAP, STRUCT) for hierarchical or semi-structured data.

2. Create Database & Table

Databases in Hive are created using CREATE DATABASE <db_name>;, and tables are defined using CREATE TABLE with specified columns and data types. Tables can store data in various file formats like TEXTFILE or ORC.

3. Hive Partitioning

Partitioning divides data into directories based on column values, improving query performance by reducing the data scanned. Partitions are declared during table creation using PARTITIONED BY.

4. Hive Built-In Operators

Includes **Arithmetic** (+, -, *, /), **Relational** (=, !=, >), and **Logical** (AND, OR, NOT) operators for performing computations and filtering data.

5. Hive Built-In Functions

Functions include **String Functions** (CONCAT, LOWER), **Aggregate Functions** (SUM, AVG), and **Date Functions** (NOW, DATEDIFF) for data manipulation.

6. Hive Views and Indexes

Views provide a virtual table for reusable queries, while indexes optimize data retrieval by creating quick lookup structures.

7. HiveQL

HiveQL supports SQL-like operations such as SELECT WHERE, ORDER BY, GROUP BY, and JOINS for querying and manipulating datasets.

Code:-

view_and_indexes.sql:-

```
--Create view
CREATE VIEW mydb.costly_employee_vw AS
SELECT e.empfname, e.job, e.salary
FROM mydb.employee e
WHERE e.salary > 2000
AND e.job NOT LIKE 'M%';

--Drop the view
DROP VIEW mydb.costly_employee_vw;

--index creation
CREATE INDEX index_emp
ON TABLE employee(empcode)
AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler';

--DROP the index
DROP Index index_emp ON mydb.employee;
```

Student_data_all.txt:-

```
1,TONY,STARK,JAVA
2,TIM,ADOLF,HTML
3,KIM,JARVIS,EXCEL
4,SAM,MILES,HTML
5,KEVIN,HILL,EXCEL
6,CONNIE,SMITH,HADOOP
7,ALFRED,KINSLEY,PYTHON
8,PAUL,TIMOTHY,HTML
9,JOHN,ASGHAR,JAVA
10,ROSE,SUMMERS,DBA
11,ANDREW,FAULKNER,HADOOP
12,KAREN,MATTHEWS,JAVA
13,WENDY,SHAWN,HTML
14,BELLA,SWAN,EXCEL
15,MADII,HIMBURY,HADOOP
16,ATHENA,WILSON,HADOOP
```

Partitioning.sql:-

```
--Create a normal table to hold all the students data. This is like a staging table.
--hadoop fs -mkdir /user/sohan/input/student
--hadoop fs -copyFromLocal /home/sohan/hive_examples/student_data_all.txt /user/sohan/input/student

CREATE EXTERNAL TABLE mydb.student_info_all
(
roll    INT,
stfname     STRING,
stlname     STRING,
course  STRING
```

```
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION '/user/sohan/input/student'
;

--Create a partitioned table
CREATE TABLE  mydb.student
(
roll      INT,
stfname     STRING,
stlname     STRING
)
PARTITIONED BY (course STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
;

.....Static Partitioning
--ensure below properties:
set hive.exec.dynamic.partition=false;
set hive.exec.dynamic.partition.mode=strict;

INSERT OVERWRITE TABLE mydb.student
PARTITION(course='HTML')
SELECT roll, stfname, stlname
FROM mydb.student_info_all
where course = 'HTML';

INSERT OVERWRITE TABLE mydb.student
PARTITION(course='HADOOP')
SELECT roll, stfname, stlname
FROM mydb.student_info_all
where course = 'HADOOP';

SELECT * FROM mydb.student;

SHOW  PARTITIONS mydb.student;

INSERT OVERWRITE TABLE mydb.student
PARTITION(course)
SELECT roll, stfname, stlname
FROM mydb.student_info_all
where course = 'PYTHON';

ALTER TABLE  mydb.student
ADD PARTITION  (course='PYTHON');

ALTER TABLE mydb.student DROP PARTITION (course='PYTHON');

SHOW PARTITIONS; --also check hdfs location

ALTER TABLE mydb.student DROP PARTITION (course='HADOOP');
ALTER TABLE mydb.student DROP PARTITION (course='HTML');

DESC FORMATTED  mydb.student;
```

- ---------- Dynamic partitioning:

```
set hive.exec.dynamic.partition=true;
set hive.exec.dynamic.partition.mode=nonstrict;

INSERT OVERWRITE TABLE mydb.student
PARTITION(course)
SELECT roll, stfname, stlname, course
FROM mydb.student_info_all;

SELECT * FROM mydb.student;

SHOW PARTITIONS mydb.student;

SHOW PARTITIONS mydb.student PARTITION (course=HADOOP);
```

Employee.sql :-

```
--Start hive using:
--beeline -u jdbc:hive2://
----------------------------------------
--List all the databases
SHOW DATABASES;

--creating a db
CREATE DATABASE mydb123;

--deleting an existing db
DROP DATABASE mydb123;

--create and use a db
CREATE DATABASE mydb;
USE mydb;

--List all the tables in selected database
SHOW TABLES;

--Create a Hive Managed table
CREATE TABLE mydb.employee
(
empcode     INT,
empfname    STRING,
emplname    STRING,
job         STRING,
manager     STRING,
hiredate    STRING,
salary      INT,
commission INT,
deptcode    INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
;
```

```
--(Manually)PLACE THE FILE employee_data.txt IN THE BELOW MENTIONED LOCAL
DIRECTORY.

--EXECUTE LOAD COMMAND
LOAD DATA LOCAL INPATH '/home/sohan/hive_examples/employee_data.txt'
INTO TABLE mydb.employee;

--List all the records in a table
SELECT * FROM mydb.employee;

--(Manually)Check if the data in employee_data.txt (/home) matches with employee_data.txt (hive
warehouse directory).

--Get the Table definition
SHOW CREATE TABLE mydb.employee; --find hive warehouse directory location

--Get the table structure
DESCRIBE mydb.employee;

--Insert into an existing table
INSERT INTO mydb.employee
VALUES (9861,'JENNIFER','HUETTE','ANALYST',7839,'1996-07-01',5000,100,50);

SELECT * FROM mydb.employee; -- to check if record got inserted.

--(Manually)Check if the record is visible in a separate newly created file under table directory on hdfs
(inside hive warehouse directory).

--SELECT WHERE
--Display the FULL NAME and SALARY drawn by an analyst working in dept no 20
SELECT empfname||" "||emplname AS empfullname, salary
FROM mydb.employee
WHERE job = 'ANALYST' AND deptcode = 20;

--SELECT ORDER BY
--Display all the MANAGERS in order where the one who served the most appears first.
SELECT empfname||" "||emplname AS empfullname, hiredate
FROM mydb.employee
WHERE job = 'MANAGER'
ORDER BY hiredate asc;

--SELECT GROUP BY
--Prepare a list of all the jobs and their respective number of employees working for that job.
SELECT job, COUNT(empfname)
FROM mydb.employee
GROUP BY job;
```

Employee_dateConversion.sql:-

```
--Create a Hive Managed table
CREATE TABLE  mydb.employee
(
empcode      INT,
empfname     STRING,
emplname     STRING,
job          STRING,
manager      STRING,
hiredate     STRING,
salary       INT,
commission INT,
deptcode     INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
;

--Create another staging table to hold all the data in String format.
CREATE  TABLE  mydb.employee_string
(
empcode      STRING,
empfname     STRING,
emplname     STRING,
job          STRING,
manager      STRING,
hiredate     STRING,
salary       STRING,
commission STRING,
deptcode     STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
;

--Load Staging table with file data.
LOAD DATA LOCAL INPATH '/home/sohan/Hive_examples/employee_data.txt'
INTO TABLE mydb.employee_string;

--Insert the data with date conversion into main table
INSERT OVERWRITE TABLE mydb.employee
SELECT empcode,
empfname,
emplname,
job,
manager,
from_unixtime(unix_timestamp(hiredate, 'yyyy-MM-dd')) hiredate,
salary,
commission,
deptcode
FROM mydb.employee_string;
```

employee_data.txt :-

```
9369,TONY,STARK,SOFTWAREENGINEER,7902,1980-12-17,2800,0,20
9499,TIM,ADOLF,SALESMAN,7698,1981-02-20,1600,300,30
9566,KIM,JARVIS,MANAGER,7839,1981-04-02,3570,0,20
9654,SAM,MILES,SALESMAN,7698,1981-09-28,1250,1400,30
9782,KEVIN,HILL,MANAGER,7839,1981-06-09,2940,0,10
9788,CONNIE,SMITH,ANALYST,7566,1982-12-09,3000,0,20
9839,ALFRED,KINSLEY,PRESIDENT,7566,1981-11-17,5000,0,10
9844,PAUL,TIMOTHY,SALESMAN,7698,1981-09-08,1500,0,30
9876,JOHN,ASGHAR,SOFTWAREENGINEER,7788,1983-01-12,3100,0,20
9900,ROSE,SUMMERS,TECHNICALLEAD,7698,1981-12-03,2950,0,20
9902,ANDREW,FAULKNER,ANALYST,7566,1981-12-03,3000,0,10
9934,KAREN,MATTHEWS,SOFTWAREENGINEER,7782,1982-01-23,3300,0,20
9591,WENDY,SHAWN,SALESMAN,7698,1981-02-22,500,0,30
9698,BELLA,SWAN,MANAGER,7839,1981-05-01,3420,0,30
9777,MADII,HIMBURY,ANALYST,7839,1981-05-01,2000,200,
9860,ATHENA,WILSON,ANALYST,7839,1992-06-21,7000,100,50
```

Dept_with_Joins.sql :-

```
--hdfs dfs -mkdir -p /user/sohan/input/dept
--hdfs dfs -copyFromLocal hive_examples/dept_data.txt /user/sohan/input/dept/
--hdfs dfs -cat /user/sohan/input/dept/dept_data.txt

--create external data
CREATE EXTERNAL TABLE mydb.department
(
  deptcode INT,
  deptname STRING,
  location STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION '/user/sohan/input/dept'
;

--Check dept data
SELECT * FROM mydb.department;

--Inner Join
SELECT e.empfname, e.deptcode, d.deptname, d.location
FROM mydb.employee e
INNER JOIN mydb.department d
ON e.deptcode = d.deptcode;

--Left Join
SELECT e.empfname, e.deptcode, d.deptname, d.location
FROM mydb.employee e
LEFT JOIN mydb.department d
```

```
ON e.deptcode = d.deptcode;

--Right Join
SELECT e.empfname, e.deptcode, d.deptname, d.location
FROM mydb.employee e
RIGHT JOIN mydb.department d
ON e.deptcode = d.deptcode;

--Full Outer Join
SELECT e.empfname, e.deptcode, d.deptname, d.location
FROM (SELECT empfname, deptcode from mydb.employee where job = 'ANALYST') e
FULL OUTER JOIN mydb.department d
ON e.deptcode = d.deptcode;
```

dept_data.txt :-

```
10,FINANCE,EDINBURGH
20,SOFTWARE,PADDINGTON
30,SALES,MAIDSTONE
40,MARKETING,DARLINGTON
50,ADMIN,BIRMINGHAM
```

Output:-

```
hive> create database mydb;
OK
Time taken: 0.04 seconds
hive> show databases;
OK
default
demo
mydb
Time taken: 0.041 seconds, Fetched: 3 row(s)
hive> CREATE TABLE mydb.employee
    > (
    > empcode      INT,
    > empfname     STRING,
    > emplname     STRING,
    > job          STRING,
    > manager      STRING,
    > hiredate     STRING,
    > salary       INT,
    > commission   INT,
    > deptcode     INT
    > )
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > ;
```

Practical No. :- 6

Aim:- Pig: Pig Latin Basic, Pig Data Types, Download the data, Create your Script, Save and Execute the Script, Pig Operations : Diagnostic.

Theory:-

1. Pig Latin Basic

Pig Latin is a high-level data flow language for processing large datasets in Hadoop. It provides a simple way to express data transformations and is designed for ease of use, focusing on data processing rather than programming logic.

2. Pig Data Types

Pig supports several data types:

- **Primitive Types**: INT, LONG, FLOAT, DOUBLE, CHARARRAY, BYTEARRAY.
- **Complex Types**:
  - TUPLE (ordered set of elements),
  - BAG (unordered collection of tuples),
  - MAP (key-value pairs).

These data types help represent structured and semi-structured data.

3. Download the Data

Pig works with data stored in HDFS or local files. Data can be downloaded or transferred to HDFS before processing in Pig scripts.

4. Create Your Script

Pig scripts are written in .pig files and contain a sequence of statements to load, transform, and store data.

5. Save and Execute the Script

Save the script and execute it using the pig command-line interface, which runs the script on Hadoop's distributed environment.

6. Pig Operations: Diagnostic

Diagnostic operations, like DESCRIBE, EXPLAIN, and DUMP, help in understanding the structure of data, viewing execution plans, and debugging Pig scripts.

Code:-

```
employee = LOAD '/home/cloudera/BDAV/employee_data.txt'
USING PigStorage(',') AS (empcode:int,empfname:chararray,
emplname:chararray,job:chararray,manager:chararray,hiredate:chararray,
salary:int,commission:int, deptcode:int);
```

```
grunt> emp_details = FOREACH employee generate CONCAT(empfname,emplname) AS FullName, SUBSTRING(job,0,3) AS jobs;
grunt> dump emp_details;
2024-09-12 04:34:54,948 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: UN
KNOWN
```

```
employee = LOAD '/home/cloudera/BDAV/employee_data.txt' USING
PigStorage(',') AS (empcode:int, empfname:chararray, emplname:chararray,
job:chararray, manager:chararray, hiredate:chararray, salary:int, commission:int,
deptcode:int);
grouped_by_dept = GROUP employee BY deptcode;
max_salary = FOREACH grouped_by_dept GENERATE group AS deptcode,
MAX(employee.salary) AS max_salary;
DUMP max_salary;
```

```
employee = LOAD '/home/cloudera/BDAV/employee_data.txt' USING PigStorage(',')
AS (empcode:int, empfname:chararray, emplname:chararray, job:chararray,
manager:chararray, hiredate:chararray, salary:int, commission:int, deptcode:int);
dept_30_employees = FILTER employee BY
deptcode == 30; DUMP dept_30_employees;
```

```
lines = LOAD '/path/to/textfile.txt' USING TextLoader() AS
(line:chararray); words = FOREACH lines GENERATE
FLATTEN(TOKENIZE(line)) AS word; word_groups =
GROUP words BY word;
word_count = FOREACH word_groups GENERATE group AS word,
COUNT(words) AS count; DUMP word_count;
```

Output:-

```
grunt> sorted_employee = ORDER employee BY empcode;
grunt> dump sorted_employee
```

```
2024-09-12 05:42:55,456 [main] INFO  org.apache.pig.backend.hadoop.ex
(9369,TONY,STARK,SOFTWAREENGINEER,7902,1980-12-17,2800,0,20)
(9499,TIM,ADOLF,SALESMAN,7698,1981-02-20,1600,300,30)
(9566,KIM,JARVIS,MANAGER,7839,1981-04-02,3570,0,20)
(9591,WENDY,SHAWN,SALESMAN,7698,1981-02-22,500,0,30)
(9654,SAM,MILES,SALESMAN,7698,1981-09-28,1250,1400,30)
(9698,BELLA,SWAN,MANAGER,7839,1981-05-01,3420,0,30)
(9777,MADII,HIMBURY,ANALYST,7839,1981-05-01,2000,200,)
(9782,KEVIN,HILL,MANAGER,7839,1981-06-09,2940,0,10)
(9788,CONNIE,SMITH,ANALYST,7566,1982-12-09,3000,0,20)
(9839,ALFRED,KINSLEY,PRESIDENT,7566,1981-11-17,5000,0,10)
(9844,PAUL,TIMOTHY,SALESMAN,7698,1981-09-08,1500,0,30)
(9860,ATHENA,WILSON,ANALYST,7839,1992-06-21,7000,100,50)
(9876,JOHN,ASGHAR,SOFTWAREENGINEER,7788,1983-01-12,3100,0,20)
(9900,ROSE,SUMMERS,TECHNICALLEAD,7698,1981-12-03,2950,0,20)
(9902,ANDREW,FAULKNER,ANALYST,7566,1981-12-03,3000,0,10)
(9934,KAREN,MATTHEWS,SOFTWAREENGINEER,7782,1982-01-23,3300,0,20)
grunt>
```

```
s to process : 1
(TONYSTARK,SOF)
(TIMADOLF,SAL)
(KIMJARVIS,MAN)
(SAMMILES,SAL)
(KEVINHILL,MAN)
(CONNIESMITH,ANA)
(ALFREDKINSLEY,PRE)
(PAULTIMOTHY,SAL)
(JOHNASGHAR,SOF)
(ROSESUMMERS,TEC)
(ANDREWFAULKNER,ANA)
(KARENMATTHEWS,SOF)
(WENDYSHAWN,SAL)
(BELLASWAN,MAN)
(MADIIHIMBURY,ANA)
(ATHENAWILSON,ANA)
grunt>
```

```
(10,5000)
(20,3570)
(30,3420)
(50,7000)
(,2000)
grunt>
```

```
(9499,TIM,ADOLF,SALESMAN,7698,1981-02-20,1600,300,30)
(9654,SAM,MILES,SALESMAN,7698,1981-09-28,1250,1400,30)
(9844,PAUL,TIMOTHY,SALESMAN,7698,1981-09-08,1500,0,30)
(9591,WENDY,SHAWN,SALESMAN,7698,1981-02-22,500,0,30)
(9698,BELLA,SWAN,MANAGER,7839,1981-05-01,3420,0,30)
```

```
(A,1)
(I,1)
(be,2)
(of,1)
(or,1)
(to,4)
(and,7)
(for,1)
(how,1)
(job,1)
(key,1)
(the,7)
(Take,1)
(been,1)
(feel,1)
(here,1)
(main,1)
(this,1)
(time,1)
(very,1)
(with,1)
(your,4)
(about,2)
(learn,1)
(them.,1)
(I’ve,1)
(duties,1)
(growth,1)
(relate,1)
(skills,1)
(values,1)
(aligned,1)
(company,1)
(culture,1)
(discuss,1)
```

Practical No. :- 7

Aim:- Spark: Downloading Data Set and Processing it Spark,

Theory:-

1. Downloading Dataset

To start processing data with Spark, the first step is to download or access a dataset. Common sources include public repositories (like Kaggle or UCI), HDFS (Hadoop Distributed File System), local files, or cloud storage (e.g., Amazon S3, Azure Blob Storage). The dataset can be in formats such as CSV, JSON, Parquet, or text files.

2. Loading Data into Spark

Once the dataset is obtained, it can be loaded into Spark using its APIs. In PySpark (Python), you can load data using:

- spark.read.csv('path_to_file') for CSV files.
- spark.read.json('path_to_file') for JSON files.
- spark.read.parquet('path_to_file') for Parquet files. This allows Spark to load data into **DataFrames** or **RDDs**.

3. Data Processing in Spark

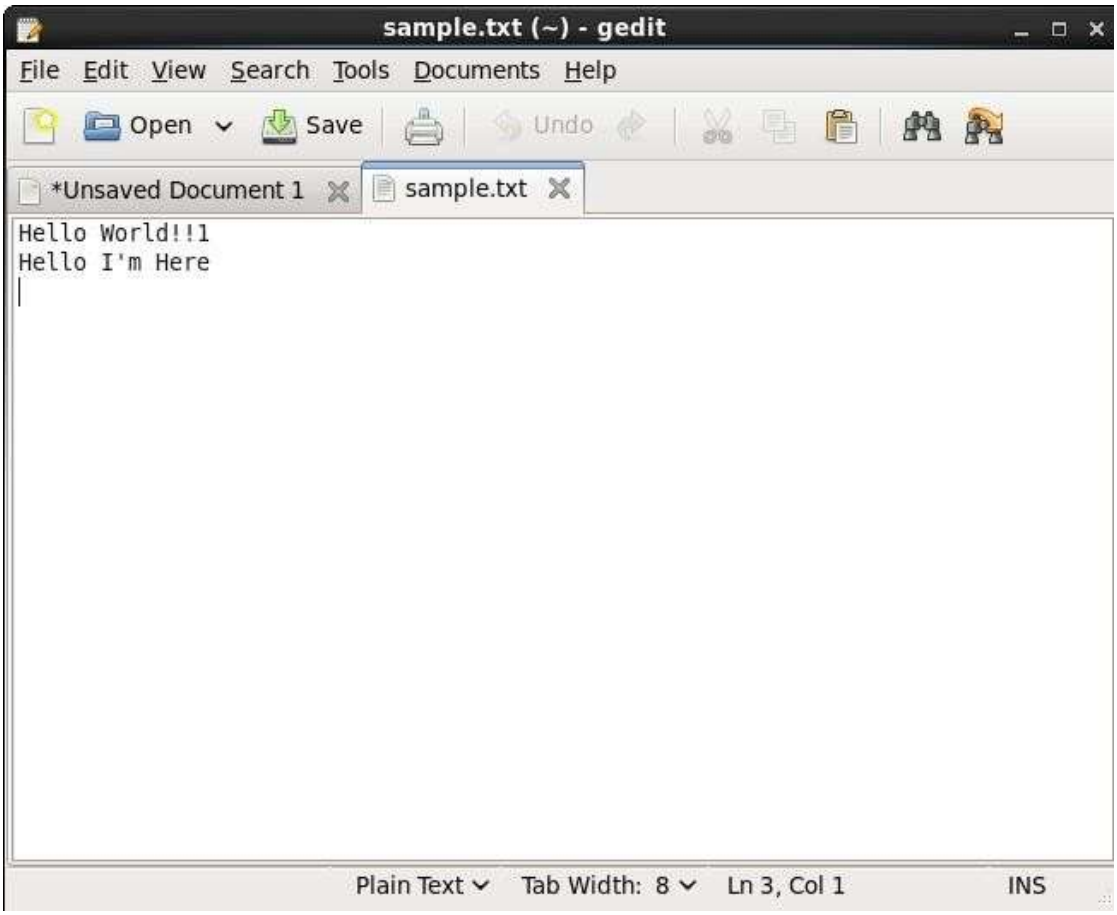
After loading the data, Spark provides powerful operations for data manipulation:

- **DataFrames** allow SQL-like operations, such as select(), filter(), and groupBy().
- **RDDs** support low-level transformations like map(), flatMap(), and reduceByKey().
- **Actions** like collect(), count(), and save() trigger computation and results can be stored or analyzed further.

Code:-

```
String dataPath = " hdfs://localhost:8020/user/cloudera/input/sample.txt ";
Dataset<Row> df = spark.read()
        .option("header", "true")
        .option("inferSchema", "true")
        .csv(dataPath);
df.printSchema();
df.show(5);
```

Output:-



Practical No. :- 8

Aim:- Spark: Word Count in Apache Spark.

Theory:-

The **Word Count** program in Apache Spark is a common example used to demonstrate basic data processing tasks in Spark. It counts the occurrence of each word in a text dataset.

1. Input Data

The input dataset can be a text file stored in HDFS, a local file system, or a cloud storage service. Each line of the text file typically contains words separated by spaces or other delimiters.

2. Loading Data into Spark

The text file is loaded into Spark using spark.read.text() or spark.sparkContext.textFile() in PySpark. This creates an RDD (Resilient Distributed Dataset) where each element represents a line from the file.

3. Transformation

Spark processes the data through transformations:

- **FlatMap**: Splits each line into words.
- **Map**: Pairs each word with the value 1 (indicating one occurrence).
- **ReduceByKey**: Aggregates the word counts by summing the values for each key (word).

4. Action

The action collect() is used to retrieve the word count results from all partitions to the driver node.

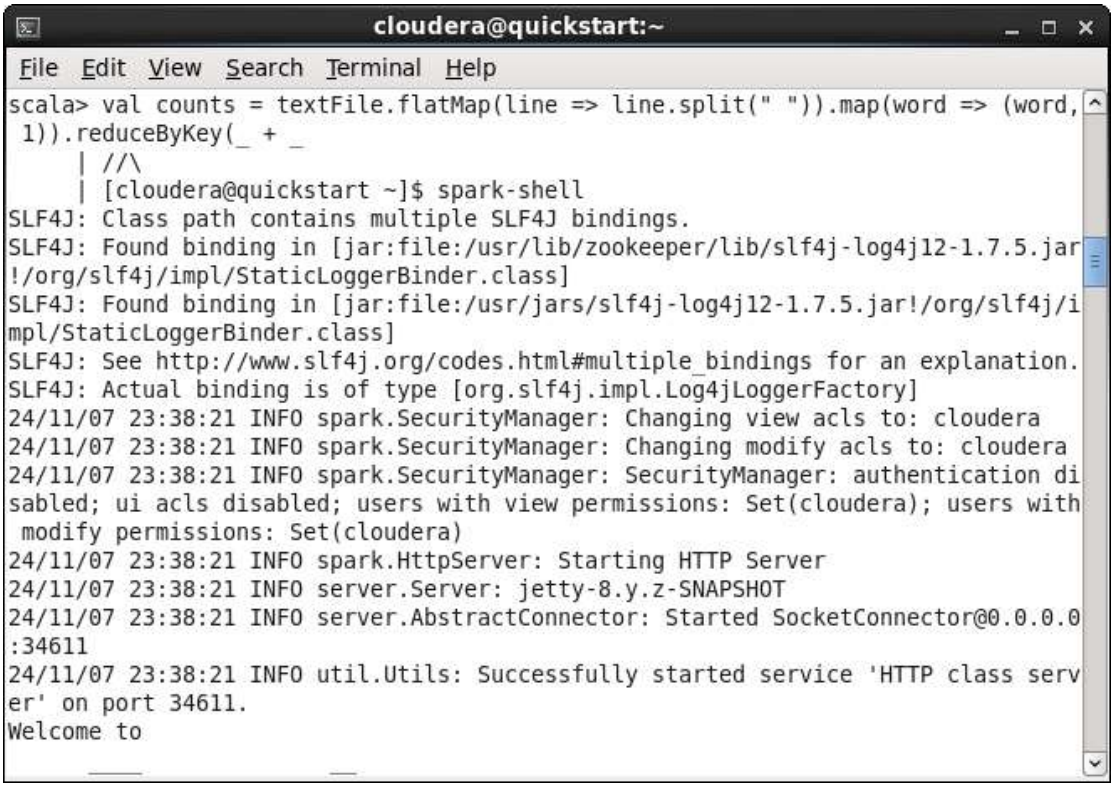
5. Output

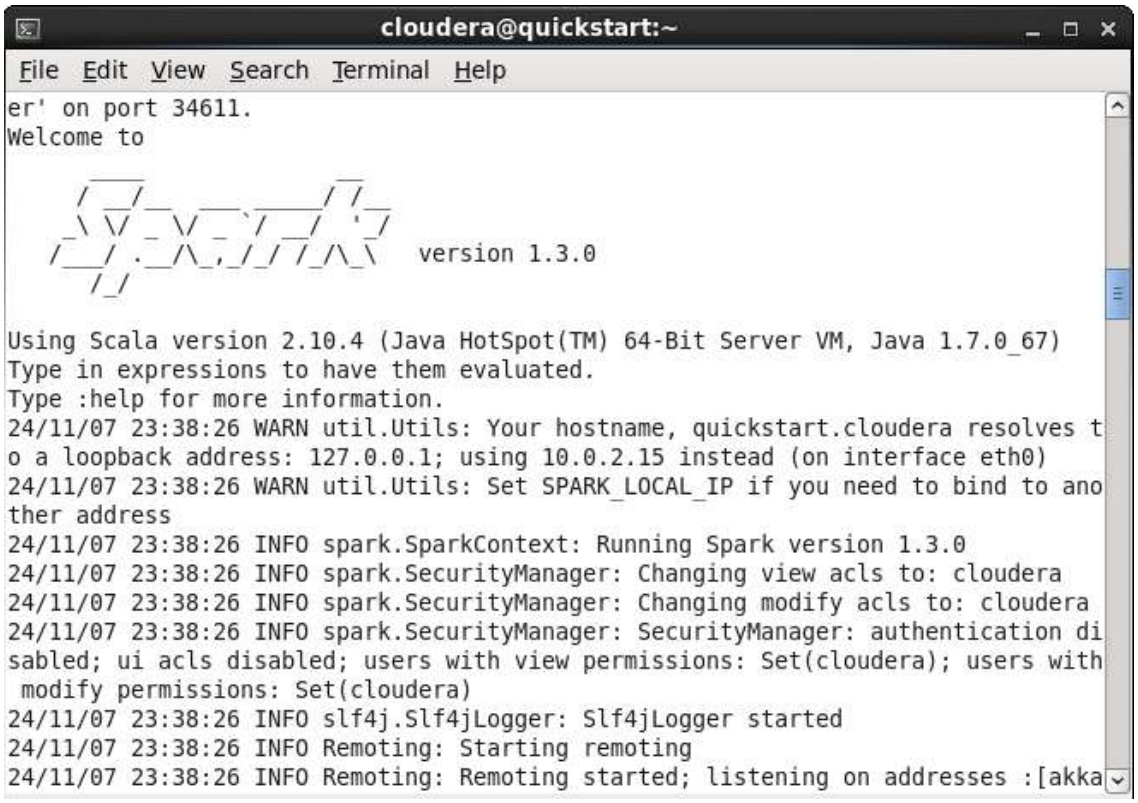
The result is a list of tuples with each word and its corresponding count, which can be printed or saved into a file.

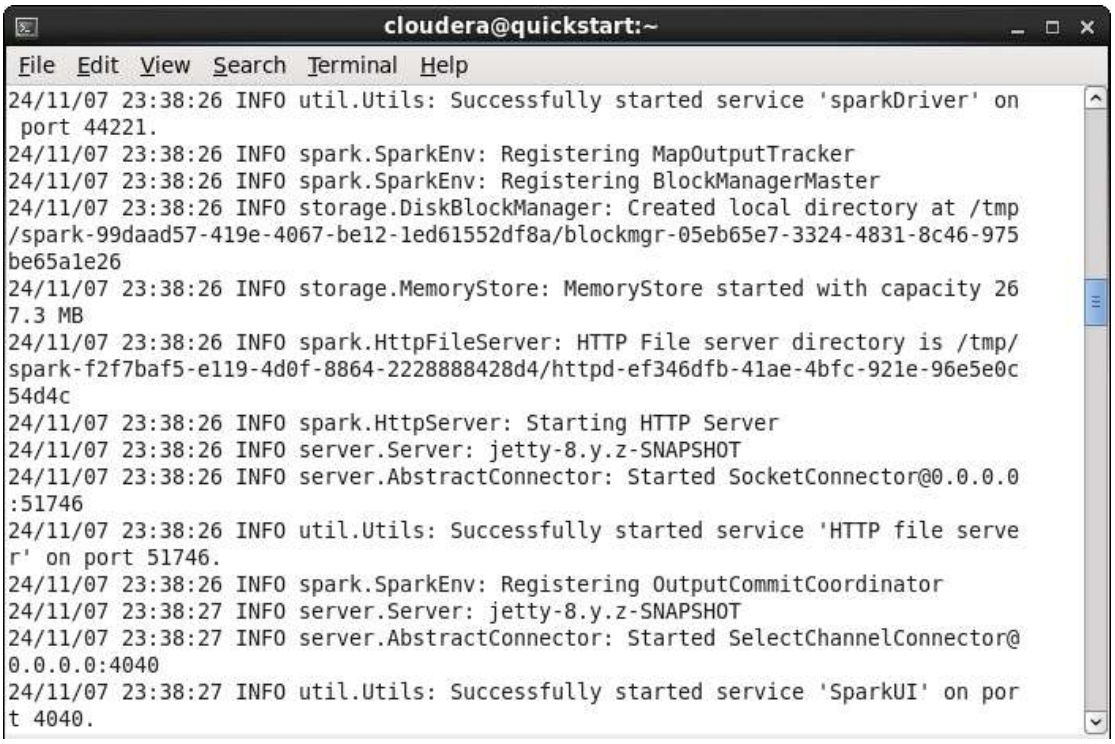
Code:-

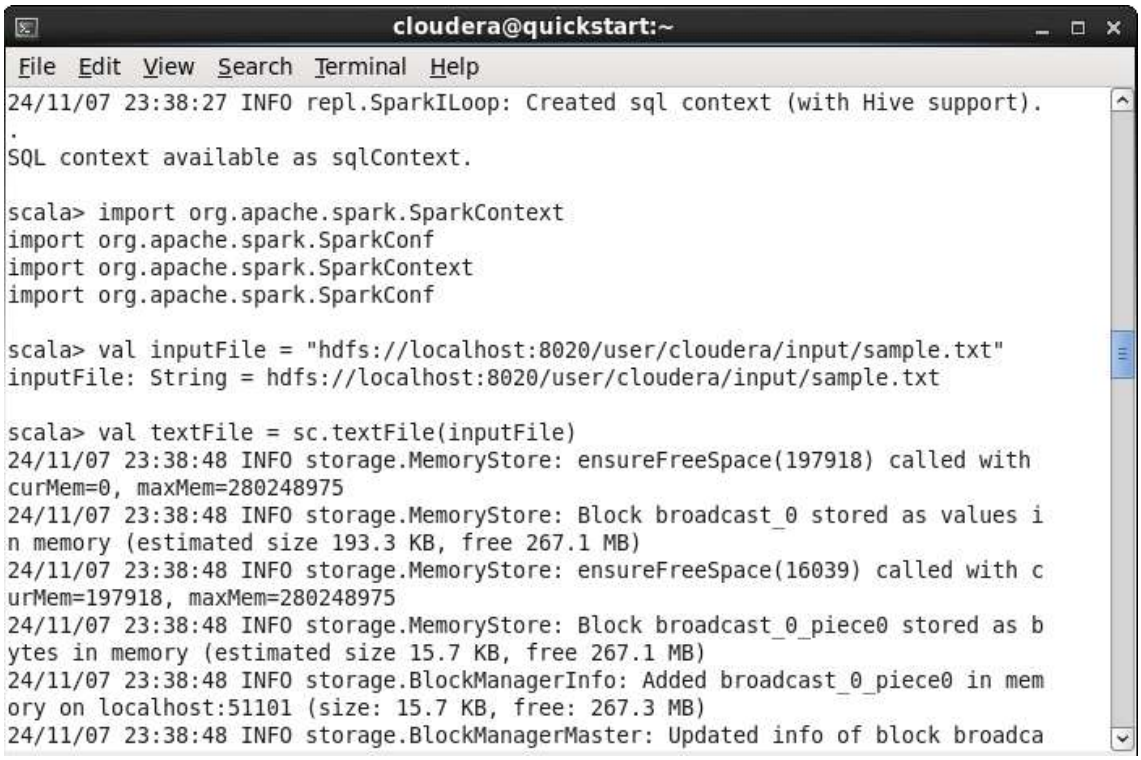
```
import org.apache.spark.SparkContext
import org.apache.spark.SparkConf

val conf = new SparkConf().setAppName("WordCountApp5")
sc = new SparkContext(conf)

val inputFile = "hdfs://localhost:8020/user/cloudera/input/sample.txt"
val outputFile = "hdfs://localhost:8020/user/cloudera/output/wordcount6"

val textFile = sc.textFile(inputFile)
val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)
counts.collect().foreach(println)
counts.take(10).foreach(println)
val outputFile = "hdfs://localhost:8020/user/cloudera/output/wordcount6"

counts.saveAsTextFile(outputFile)
println("Word Count Completed")
```

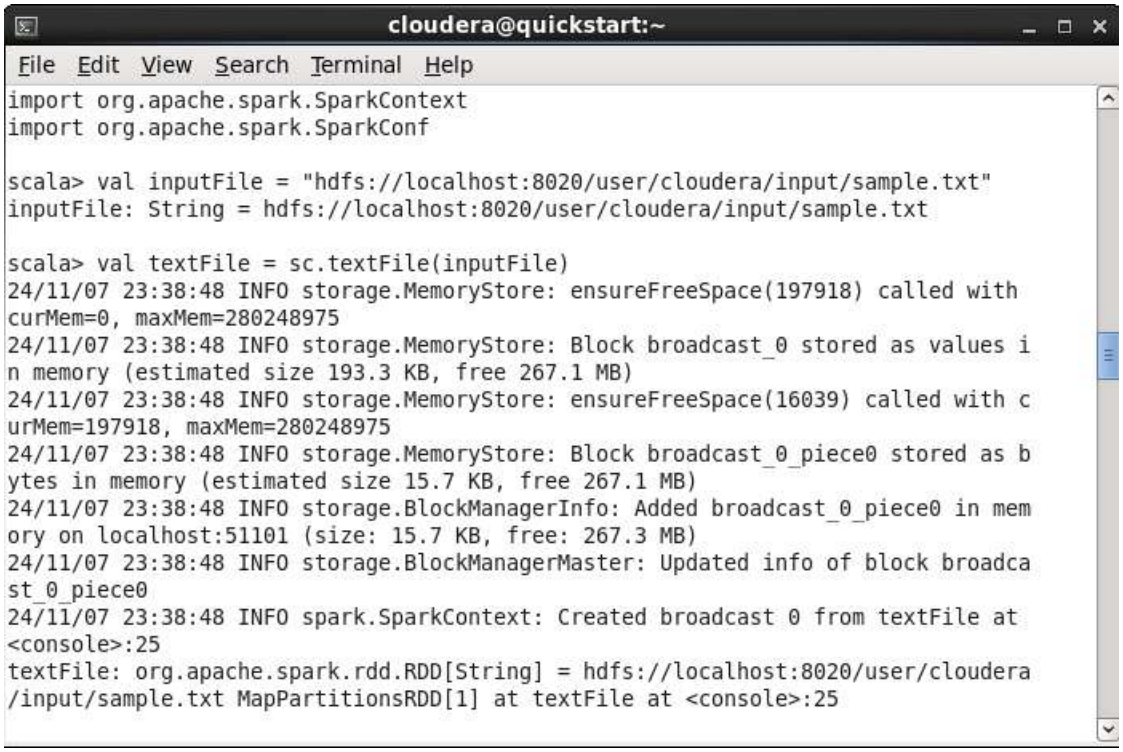
Output:-

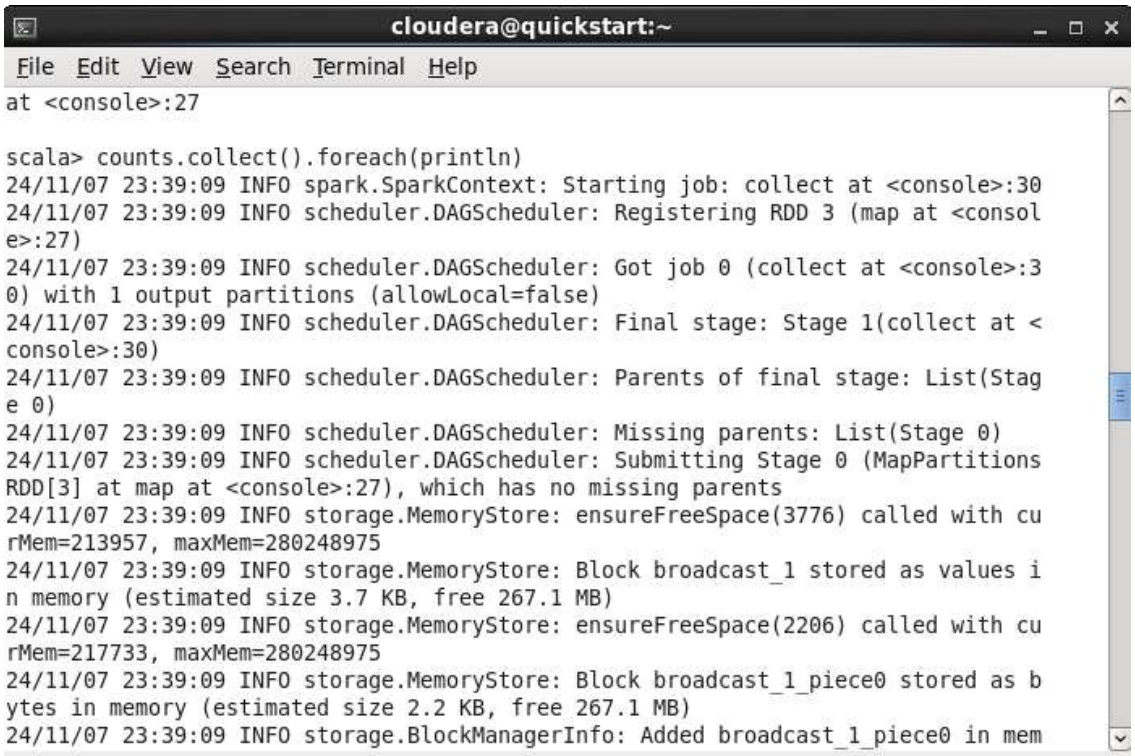

cloudera@quickstart:~
File Edit View Search Terminal Help
scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word,
1)).reduceByKey(_ + _
| //\
| [cloudera@quickstart ~]$ spark-shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/jars/slf4j-log4j12-1.7.5.jar!/org/slf4j/i
mpl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
24/11/07 23:38:21 INFO spark.SecurityManager: Changing view acls to: cloudera
24/11/07 23:38:21 INFO spark.SecurityManager: Changing modify acls to: cloudera
24/11/07 23:38:21 INFO spark.SecurityManager: SecurityManager: authentication di
sabled; ui acls disabled; users with view permissions: Set(cloudera); users with
modify permissions: Set(cloudera)
24/11/07 23:38:21 INFO spark.HttpServer: Starting HTTP Server
24/11/07 23:38:21 INFO server.Server: jetty-8.y.z-SNAPSHOT
24/11/07 23:38:21 INFO server.AbstractConnector: Started SocketConnector@0.0.0.0
:34611
24/11/07 23:38:21 INFO util.Utils: Successfully started service 'HTTP class serv
er' on port 34611.
Welcome to

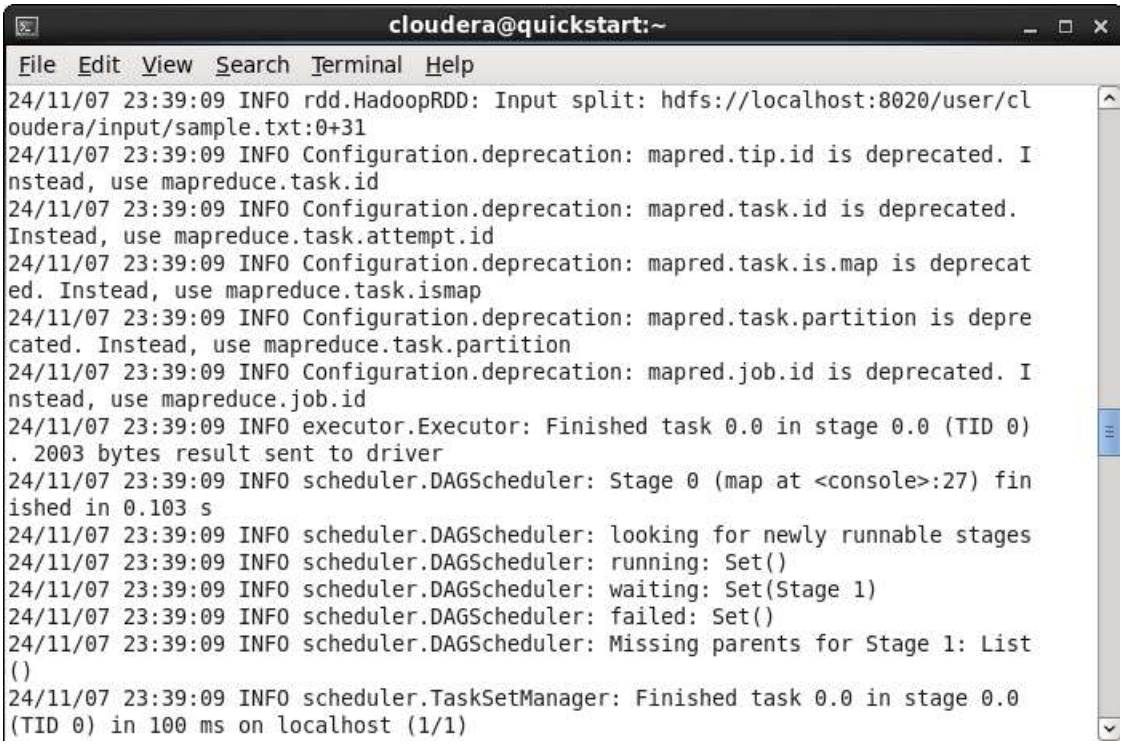

cloudera@quickstart:~
File Edit View Search Terminal Help
er' on port 34611.
Welcome to
version 1.3.0
Using Scala version 2.10.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_67)
Type in expressions to have them evaluated.
Type :help for more information.
24/11/07 23:38:26 WARN util.Utils: Your hostname, quickstart.cloudera resolves t
o a loopback address: 127.0.0.1; using 10.0.2.15 instead (on interface eth0)
24/11/07 23:38:26 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to ano
ther address
24/11/07 23:38:26 INFO spark.SparkContext: Running Spark version 1.3.0
24/11/07 23:38:26 INFO spark.SecurityManager: Changing view acls to: cloudera
24/11/07 23:38:26 INFO spark.SecurityManager: Changing modify acls to: cloudera
24/11/07 23:38:26 INFO spark.SecurityManager: SecurityManager: authentication di
sabled; ui acls disabled; users with view permissions: Set(cloudera); users with
modify permissions: Set(cloudera)
24/11/07 23:38:26 INFO slf4j.Slf4jLogger: Slf4jLogger started
24/11/07 23:38:26 INFO Remoting: Starting remoting
24/11/07 23:38:26 INFO Remoting: Remoting started; listening on addresses :[akka

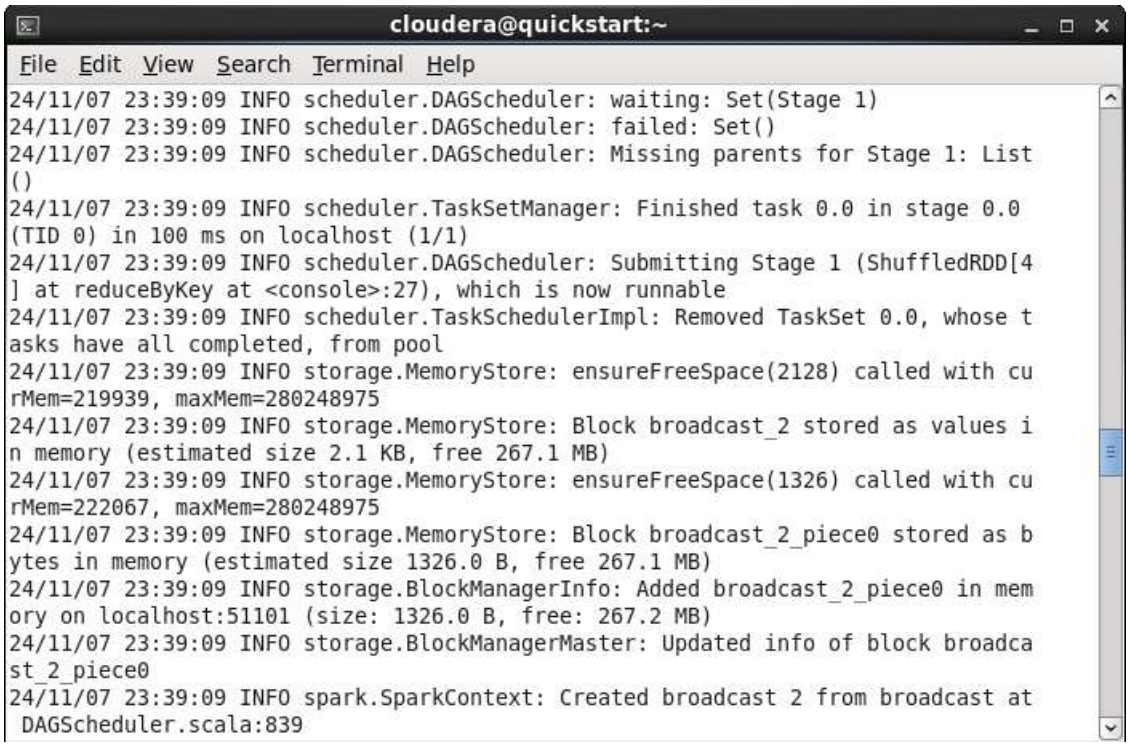

cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:38:26 INFO util.Utils: Successfully started service 'sparkDriver' on
port 44221.
24/11/07 23:38:26 INFO spark.SparkEnv: Registering MapOutputTracker
24/11/07 23:38:26 INFO spark.SparkEnv: Registering BlockManagerMaster
24/11/07 23:38:26 INFO storage.DiskBlockManager: Created local directory at /tmp
/spark-99daad57-419e-4067-be12-1ed61552df8a/blockmgr-05eb65e7-3324-4831-8c46-975
be65a1e26
24/11/07 23:38:26 INFO storage.MemoryStore: MemoryStore started with capacity 26
7.3 MB
24/11/07 23:38:26 INFO spark.HttpFileServer: HTTP File server directory is /tmp/
spark-f2f7baf5-e119-4d0f-8864-2228888428d4/httpd-ef346dfb-41ae-4bfc-921e-96e5e0c
54d4c
24/11/07 23:38:26 INFO spark.HttpServer: Starting HTTP Server
24/11/07 23:38:26 INFO server.Server: jetty-8.y.z-SNAPSHOT
24/11/07 23:38:26 INFO server.AbstractConnector: Started SocketConnector@0.0.0.0
:51746
24/11/07 23:38:26 INFO util.Utils: Successfully started service 'HTTP file serve
r' on port 51746.
24/11/07 23:38:26 INFO spark.SparkEnv: Registering OutputCommitCoordinator
24/11/07 23:38:27 INFO server.Server: jetty-8.y.z-SNAPSHOT
24/11/07 23:38:27 INFO server.AbstractConnector: Started SelectChannelConnector@
0.0.0.0:4040
24/11/07 23:38:27 INFO util.Utils: Successfully started service 'SparkUI' on por
t 4040.

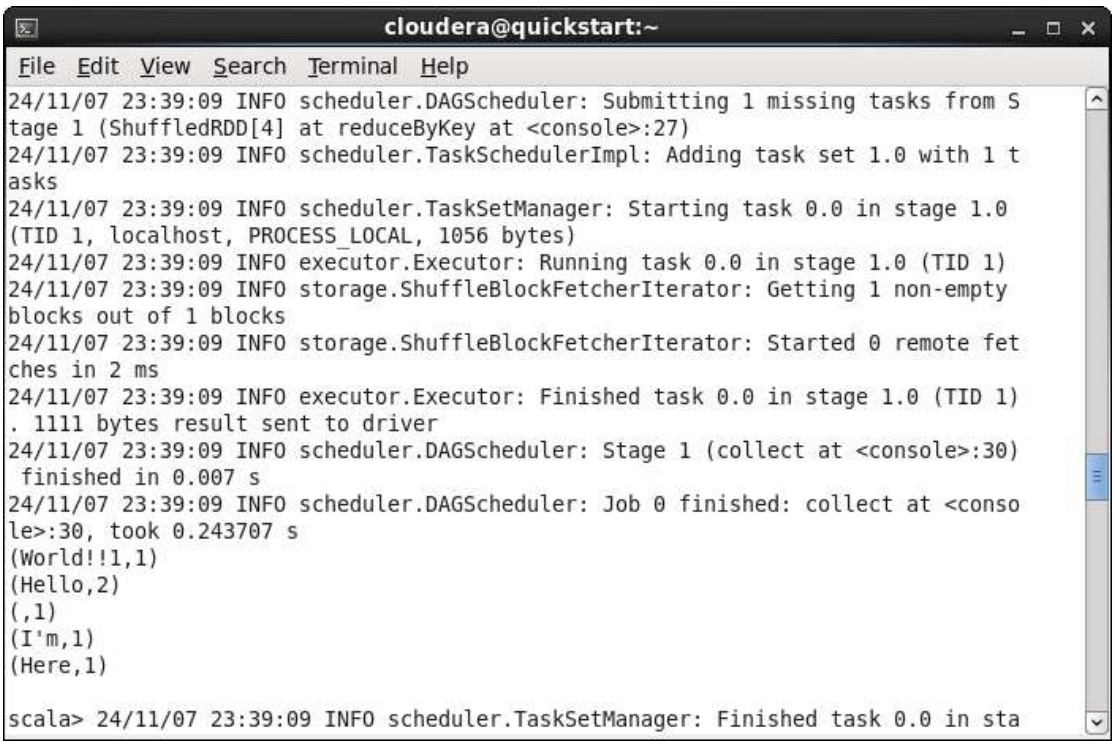

cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:38:27 INFO repl.SparkILoop: Created sql context (with Hive support).
.
SQL context available as sqlContext.
scala> import org.apache.spark.SparkContext
import org.apache.spark.SparkConf
import org.apache.spark.SparkContext
import org.apache.spark.SparkConf
scala> val inputFile = "hdfs://localhost:8020/user/cloudera/input/sample.txt"
inputFile: String = hdfs://localhost:8020/user/cloudera/input/sample.txt
scala> val textFile = sc.textFile(inputFile)
24/11/07 23:38:48 INFO storage.MemoryStore: ensureFreeSpace(197918) called with
curMem=0, maxMem=280248975
24/11/07 23:38:48 INFO storage.MemoryStore: Block broadcast_0 stored as values i
n memory (estimated size 193.3 KB, free 267.1 MB)
24/11/07 23:38:48 INFO storage.MemoryStore: ensureFreeSpace(16039) called with c
urMem=197918, maxMem=280248975
24/11/07 23:38:48 INFO storage.MemoryStore: Block broadcast_0_piece0 stored as b
ytes in memory (estimated size 15.7 KB, free 267.1 MB)
24/11/07 23:38:48 INFO storage.BlockManagerInfo: Added broadcast_0_piece0 in mem
ory on localhost:51101 (size: 15.7 KB, free: 267.3 MB)
24/11/07 23:38:48 INFO storage.BlockManagerMaster: Updated info of block broadca

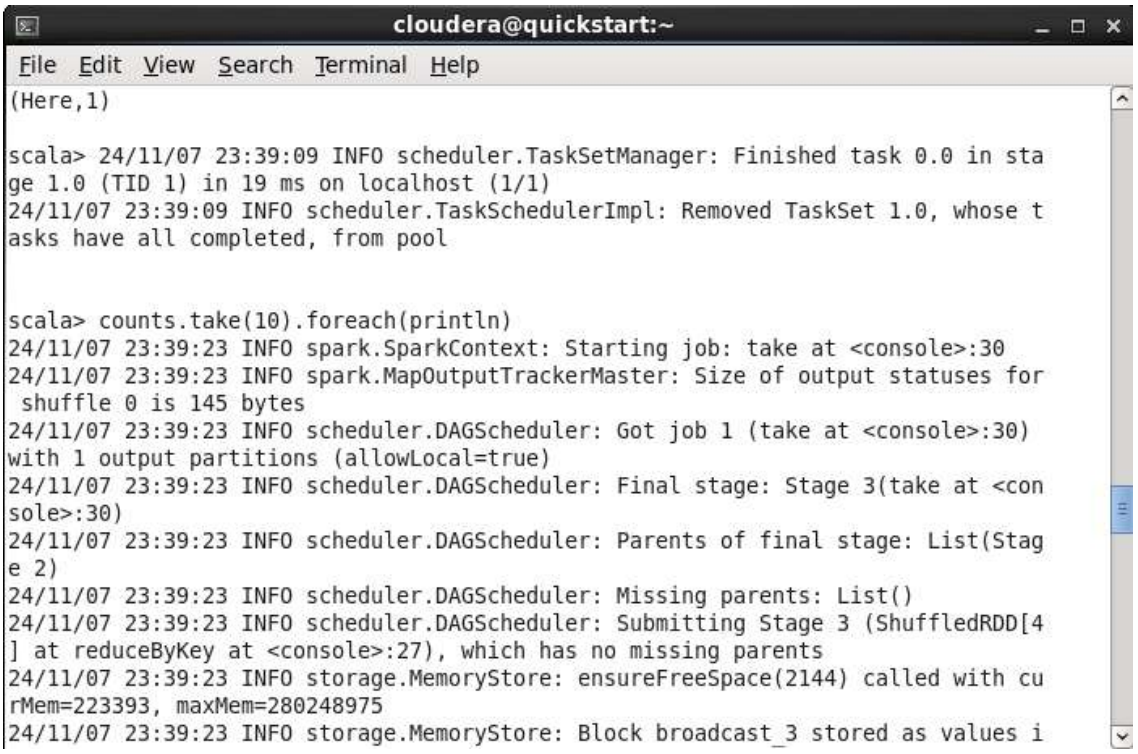

cloudera@quickstart:~
File Edit View Search Terminal Help
import org.apache.spark.SparkContext
import org.apache.spark.SparkConf
scala> val inputFile = "hdfs://localhost:8020/user/cloudera/input/sample.txt"
inputFile: String = hdfs://localhost:8020/user/cloudera/input/sample.txt
scala> val textFile = sc.textFile(inputFile)
24/11/07 23:38:48 INFO storage.MemoryStore: ensureFreeSpace(197918) called with
curMem=0, maxMem=280248975
24/11/07 23:38:48 INFO storage.MemoryStore: Block broadcast_0 stored as values i
n memory (estimated size 193.3 KB, free 267.1 MB)
24/11/07 23:38:48 INFO storage.MemoryStore: ensureFreeSpace(16039) called with c
urMem=197918, maxMem=280248975
24/11/07 23:38:48 INFO storage.MemoryStore: Block broadcast_0_piece0 stored as b
ytes in memory (estimated size 15.7 KB, free 267.1 MB)
24/11/07 23:38:48 INFO storage.BlockManagerInfo: Added broadcast_0_piece0 in mem
ory on localhost:51101 (size: 15.7 KB, free: 267.3 MB)
24/11/07 23:38:48 INFO storage.BlockManagerMaster: Updated info of block broadca
st_0_piece0
24/11/07 23:38:48 INFO spark.SparkContext: Created broadcast 0 from textFile at
<console>:25
textFile: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/cloudera
/input/sample.txt MapPartitionsRDD[1] at textFile at <console>:25

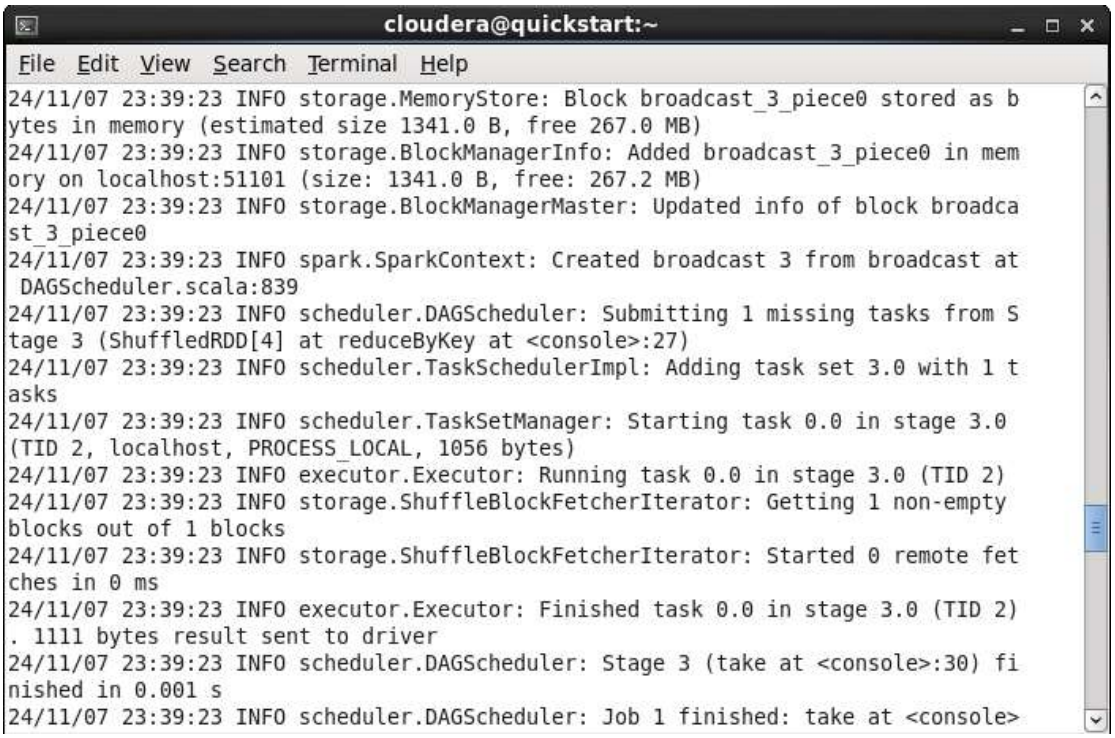

cloudera@quickstart:~
File Edit View Search Terminal Help
at <console>:27
scala> counts.collect().foreach(println)
24/11/07 23:39:09 INFO spark.SparkContext: Starting job: collect at <console>:30
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Registering RDD 3 (map at <consol
e>:27)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Got job 0 (collect at <console>:3
0) with 1 output partitions (allowLocal=false)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Final stage: Stage 1(collect at <
console>:30)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Parents of final stage: List(Stag
e 0)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Missing parents: List(Stage 0)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Submitting Stage 0 (MapPartitions
RDD[3] at map at <console>:27), which has no missing parents
24/11/07 23:39:09 INFO storage.MemoryStore: ensureFreeSpace(3776) called with cu
rMem=213957, maxMem=280248975
24/11/07 23:39:09 INFO storage.MemoryStore: Block broadcast_1 stored as values i
n memory (estimated size 3.7 KB, free 267.1 MB)
24/11/07 23:39:09 INFO storage.MemoryStore: ensureFreeSpace(2206) called with cu
rMem=217733, maxMem=280248975
24/11/07 23:39:09 INFO storage.MemoryStore: Block broadcast_1_piece0 stored as b
ytes in memory (estimated size 2.2 KB, free 267.1 MB)
24/11/07 23:39:09 INFO storage.BlockManagerInfo: Added broadcast_1_piece0 in mem

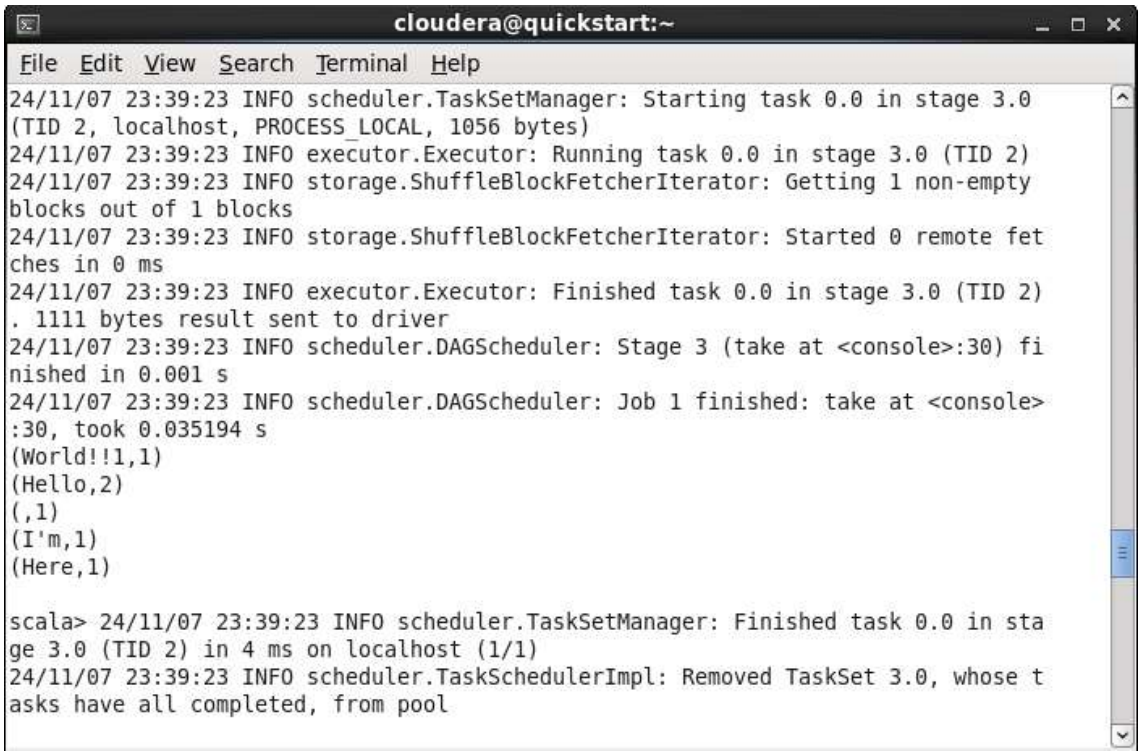

cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:39:09 INFO rdd.HadoopRDD: Input split: hdfs://localhost:8020/user/cl
oudera/input/sample.txt:0+31
24/11/07 23:39:09 INFO Configuration.deprecation: mapred.tip.id is deprecated. I
nstead, use mapreduce.task.id
24/11/07 23:39:09 INFO Configuration.deprecation: mapred.task.id is deprecated.
Instead, use mapreduce.task.attempt.id
24/11/07 23:39:09 INFO Configuration.deprecation: mapred.task.is.map is deprecat
ed. Instead, use mapreduce.task.ismap
24/11/07 23:39:09 INFO Configuration.deprecation: mapred.task.partition is depre
cated. Instead, use mapreduce.task.partition
24/11/07 23:39:09 INFO Configuration.deprecation: mapred.job.id is deprecated. I
nstead, use mapreduce.job.id
24/11/07 23:39:09 INFO executor.Executor: Finished task 0.0 in stage 0.0 (TID 0)
. 2003 bytes result sent to driver
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Stage 0 (map at <console>:27) fin
ished in 0.103 s
24/11/07 23:39:09 INFO scheduler.DAGScheduler: looking for newly runnable stages
24/11/07 23:39:09 INFO scheduler.DAGScheduler: running: Set()
24/11/07 23:39:09 INFO scheduler.DAGScheduler: waiting: Set(Stage 1)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: failed: Set()
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Missing parents for Stage 1: List
()
24/11/07 23:39:09 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 0.0
(TID 0) in 100 ms on localhost (1/1)

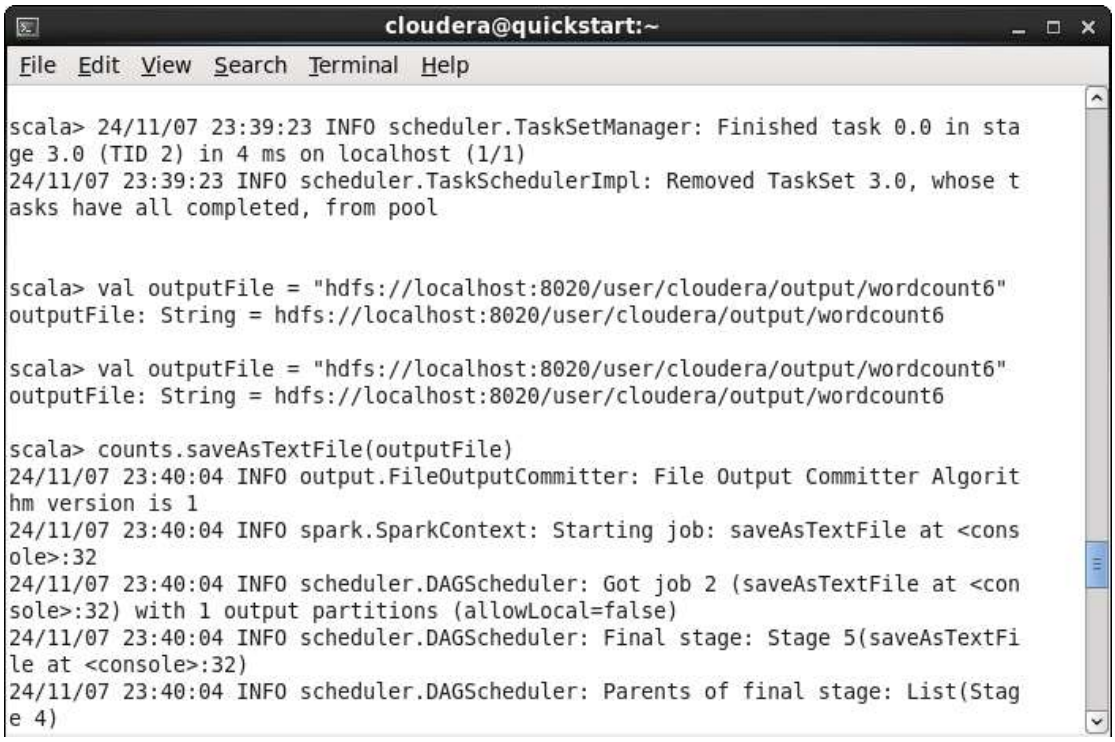

cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:39:09 INFO scheduler.DAGScheduler: waiting: Set(Stage 1)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: failed: Set()
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Missing parents for Stage 1: List
()
24/11/07 23:39:09 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 0.0
(TID 0) in 100 ms on localhost (1/1)
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Submitting Stage 1 (ShuffledRDD[4
] at reduceByKey at <console>:27), which is now runnable
24/11/07 23:39:09 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 0.0, whose t
asks have all completed, from pool
24/11/07 23:39:09 INFO storage.MemoryStore: ensureFreeSpace(2128) called with cu
rMem=219939, maxMem=280248975
24/11/07 23:39:09 INFO storage.MemoryStore: Block broadcast_2 stored as values i
n memory (estimated size 2.1 KB, free 267.1 MB)
24/11/07 23:39:09 INFO storage.MemoryStore: ensureFreeSpace(1326) called with cu
rMem=222067, maxMem=280248975
24/11/07 23:39:09 INFO storage.MemoryStore: Block broadcast_2_piece0 stored as b
ytes in memory (estimated size 1326.0 B, free 267.1 MB)
24/11/07 23:39:09 INFO storage.BlockManagerInfo: Added broadcast_2_piece0 in mem
ory on localhost:51101 (size: 1326.0 B, free: 267.2 MB)
24/11/07 23:39:09 INFO storage.BlockManagerMaster: Updated info of block broadca
st_2_piece0
24/11/07 23:39:09 INFO spark.SparkContext: Created broadcast 2 from broadcast at
DAGScheduler.scala:839

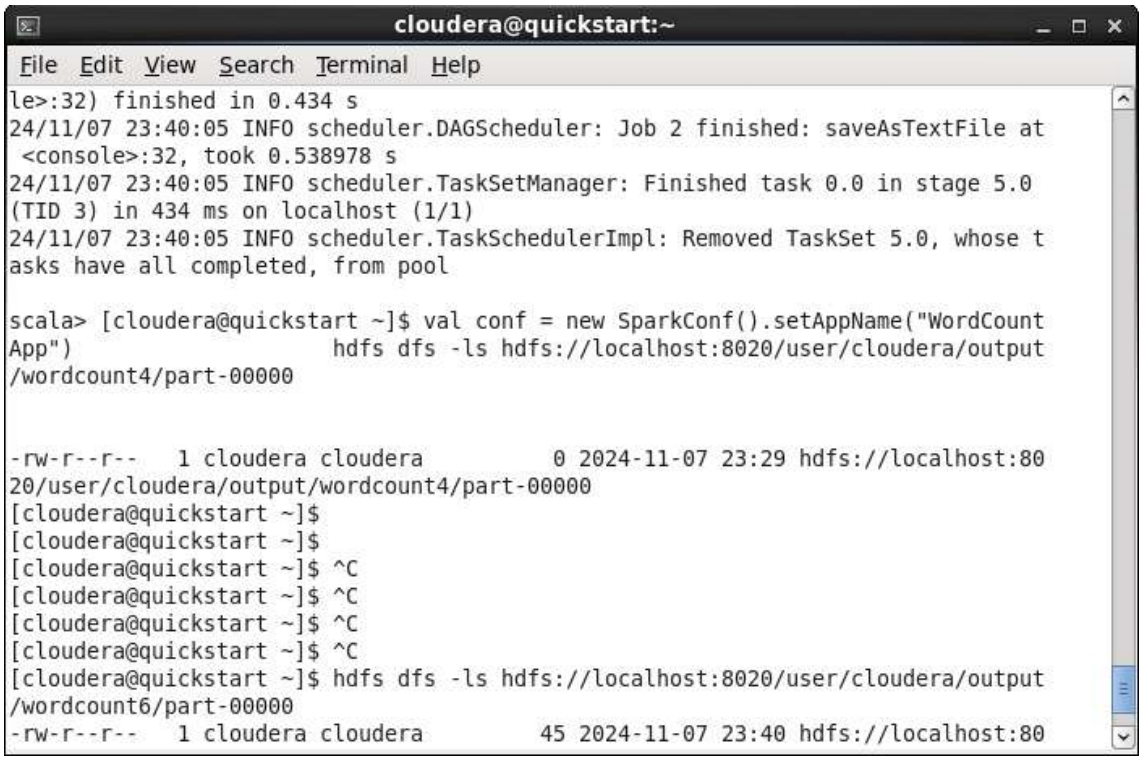

cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Submitting 1 missing tasks from S
tage 1 (ShuffledRDD[4] at reduceByKey at <console>:27)
24/11/07 23:39:09 INFO scheduler.TaskSchedulerImpl: Adding task set 1.0 with 1 t
asks
24/11/07 23:39:09 INFO scheduler.TaskSetManager: Starting task 0.0 in stage 1.0
(TID 1, localhost, PROCESS_LOCAL, 1056 bytes)
24/11/07 23:39:09 INFO executor.Executor: Running task 0.0 in stage 1.0 (TID 1)
24/11/07 23:39:09 INFO storage.ShuffleBlockFetcherIterator: Getting 1 non-empty
blocks out of 1 blocks
24/11/07 23:39:09 INFO storage.ShuffleBlockFetcherIterator: Started 0 remote fet
ches in 2 ms
24/11/07 23:39:09 INFO executor.Executor: Finished task 0.0 in stage 1.0 (TID 1)
. 1111 bytes result sent to driver
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Stage 1 (collect at <console>:30)
finished in 0.007 s
24/11/07 23:39:09 INFO scheduler.DAGScheduler: Job 0 finished: collect at <conso
le>:30, took 0.243707 s
(World!!1,1)
(Hello,2)
(,1)
(I'm,1)
(Here,1)
scala> 24/11/07 23:39:09 INFO scheduler.TaskSetManager: Finished task 0.0 in sta

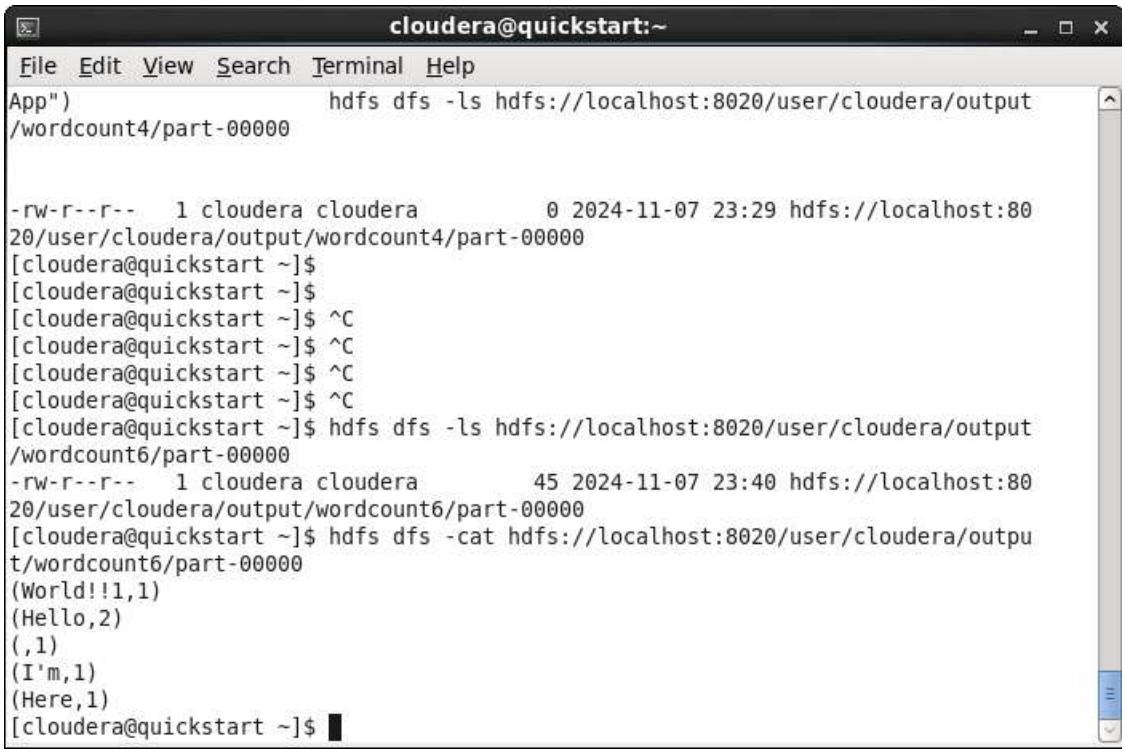

cloudera@quickstart:~
File Edit View Search Terminal Help
(Here,1)
scala> 24/11/07 23:39:09 INFO scheduler.TaskSetManager: Finished task 0.0 in sta
ge 1.0 (TID 1) in 19 ms on localhost (1/1)
24/11/07 23:39:09 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 1.0, whose t
asks have all completed, from pool
scala> counts.take(10).foreach(println)
24/11/07 23:39:23 INFO spark.SparkContext: Starting job: take at <console>:30
24/11/07 23:39:23 INFO spark.MapOutputTrackerMaster: Size of output statuses for
shuffle 0 is 145 bytes
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Got job 1 (take at <console>:30)
with 1 output partitions (allowLocal=true)
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Final stage: Stage 3(take at <con
sole>:30)
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Parents of final stage: List(Stag
e 2)
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Missing parents: List()
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Submitting Stage 3 (ShuffledRDD[4
] at reduceByKey at <console>:27), which has no missing parents
24/11/07 23:39:23 INFO storage.MemoryStore: ensureFreeSpace(2144) called with cu
rMem=223393, maxMem=280248975
24/11/07 23:39:23 INFO storage.MemoryStore: Block broadcast_3 stored as values i


cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:39:23 INFO storage.MemoryStore: Block broadcast_3_piece0 stored as b
ytes in memory (estimated size 1341.0 B, free 267.0 MB)
24/11/07 23:39:23 INFO storage.BlockManagerInfo: Added broadcast_3_piece0 in mem
ory on localhost:51101 (size: 1341.0 B, free: 267.2 MB)
24/11/07 23:39:23 INFO storage.BlockManagerMaster: Updated info of block broadca
st_3_piece0
24/11/07 23:39:23 INFO spark.SparkContext: Created broadcast 3 from broadcast at
DAGScheduler.scala:839
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Submitting 1 missing tasks from S
tage 3 (ShuffledRDD[4] at reduceByKey at <console>:27)
24/11/07 23:39:23 INFO scheduler.TaskSchedulerImpl: Adding task set 3.0 with 1 t
asks
24/11/07 23:39:23 INFO scheduler.TaskSetManager: Starting task 0.0 in stage 3.0
(TID 2, localhost, PROCESS_LOCAL, 1056 bytes)
24/11/07 23:39:23 INFO executor.Executor: Running task 0.0 in stage 3.0 (TID 2)
24/11/07 23:39:23 INFO storage.ShuffleBlockFetcherIterator: Getting 1 non-empty
blocks out of 1 blocks
24/11/07 23:39:23 INFO storage.ShuffleBlockFetcherIterator: Started 0 remote fet
ches in 0 ms
24/11/07 23:39:23 INFO executor.Executor: Finished task 0.0 in stage 3.0 (TID 2)
. 1111 bytes result sent to driver
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Stage 3 (take at <console>:30) fi
nished in 0.001 s
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Job 1 finished: take at <console>


cloudera@quickstart:~
File Edit View Search Terminal Help
24/11/07 23:39:23 INFO scheduler.TaskSetManager: Starting task 0.0 in stage 3.0
(TID 2, localhost, PROCESS_LOCAL, 1056 bytes)
24/11/07 23:39:23 INFO executor.Executor: Running task 0.0 in stage 3.0 (TID 2)
24/11/07 23:39:23 INFO storage.ShuffleBlockFetcherIterator: Getting 1 non-empty
blocks out of 1 blocks
24/11/07 23:39:23 INFO storage.ShuffleBlockFetcherIterator: Started 0 remote fet
ches in 0 ms
24/11/07 23:39:23 INFO executor.Executor: Finished task 0.0 in stage 3.0 (TID 2)
. 1111 bytes result sent to driver
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Stage 3 (take at <console>:30) fi
nished in 0.001 s
24/11/07 23:39:23 INFO scheduler.DAGScheduler: Job 1 finished: take at <console>
:30, took 0.035194 s
(World!!1,1)
(Hello,2)
(,1)
(I'm,1)
(Here,1)
scala> 24/11/07 23:39:23 INFO scheduler.TaskSetManager: Finished task 0.0 in sta
ge 3.0 (TID 2) in 4 ms on localhost (1/1)
24/11/07 23:39:23 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 3.0, whose t
asks have all completed, from pool


cloudera@quickstart:~
File Edit View Search Terminal Help
scala> 24/11/07 23:39:23 INFO scheduler.TaskSetManager: Finished task 0.0 in sta
ge 3.0 (TID 2) in 4 ms on localhost (1/1)
24/11/07 23:39:23 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 3.0, whose t
asks have all completed, from pool
scala> val outputFile = "hdfs://localhost:8020/user/cloudera/output/wordcount6"
outputFile: String = hdfs://localhost:8020/user/cloudera/output/wordcount6
scala> val outputFile = "hdfs://localhost:8020/user/cloudera/output/wordcount6"
outputFile: String = hdfs://localhost:8020/user/cloudera/output/wordcount6
scala> counts.saveAsTextFile(outputFile)
24/11/07 23:40:04 INFO output.FileOutputCommitter: File Output Committer Algorit
hm version is 1
24/11/07 23:40:04 INFO spark.SparkContext: Starting job: saveAsTextFile at <cons
ole>:32
24/11/07 23:40:04 INFO scheduler.DAGScheduler: Got job 2 (saveAsTextFile at <con
sole>:32) with 1 output partitions (allowLocal=false)
24/11/07 23:40:04 INFO scheduler.DAGScheduler: Final stage: Stage 5(saveAsTextFi
le at <console>:32)
24/11/07 23:40:04 INFO scheduler.DAGScheduler: Parents of final stage: List(Stag
e 4)


cloudera@quickstart:~
File Edit View Search Terminal Help
le>:32) finished in 0.434 s
24/11/07 23:40:05 INFO scheduler.DAGScheduler: Job 2 finished: saveAsTextFile at
<console>:32, took 0.538978 s
24/11/07 23:40:05 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 5.0
(TID 3) in 434 ms on localhost (1/1)
24/11/07 23:40:05 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 5.0, whose t
asks have all completed, from pool
scala> [cloudera@quickstart ~]$ val conf = new SparkConf().setAppName("WordCount
App") hdfs dfs -ls hdfs://localhost:8020/user/cloudera/output
/wordcount4/part-00000
-rw-r--r-- 1 cloudera cloudera 0 2024-11-07 23:29 hdfs://localhost:80
20/user/cloudera/output/wordcount4/part-00000
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ hdfs dfs -ls hdfs://localhost:8020/user/cloudera/output
/wordcount6/part-00000
-rw-r--r-- 1 cloudera cloudera 45 2024-11-07 23:40 hdfs://localhost:80


cloudera@quickstart:~
File Edit View Search Terminal Help
App") hdfs dfs -ls hdfs://localhost:8020/user/cloudera/output
/wordcount4/part-00000
-rw-r--r-- 1 cloudera cloudera 0 2024-11-07 23:29 hdfs://localhost:80
20/user/cloudera/output/wordcount4/part-00000
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ ^C
[cloudera@quickstart ~]$ hdfs dfs -ls hdfs://localhost:8020/user/cloudera/output
/wordcount6/part-00000
-rw-r--r-- 1 cloudera cloudera 45 2024-11-07 23:40 hdfs://localhost:80
20/user/cloudera/output/wordcount6/part-00000
[cloudera@quickstart ~]$ hdfs dfs -cat hdfs://localhost:8020/user/cloudera/outpu
t/wordcount6/part-00000
(World!!1,1)
(Hello,2)
(,1)
(I'm,1)
(Here,1)
[cloudera@quickstart ~]$

Practical No. :- 9

Aim:- Visualization using Tableau: Tool Overview, Importing Data, Analyzing with Charts, Creating Dashboards, Working with maps, Telling Stories with tableau.

Theory:-

1. Tool Overview

Tableau is a powerful data visualization tool that allows users to create interactive and shareable dashboards. It provides drag-and-drop features, making it accessible for both beginners and experienced analysts. Tableau connects to various data sources, including databases, spreadsheets, and cloud services.

2. Importing Data

Data can be imported into Tableau from multiple sources such as Excel, CSV files, SQL databases, Google Sheets, and more. Once connected, Tableau automatically identifies fields and data types, which users can then begin to analyze.

3. Analyzing with Charts

Tableau offers various chart types, including bar, line, scatter, pie, and histograms. Users can drag fields to rows and columns to visualize data, using dimensions (qualitative data) and measures (quantitative data). Filters and calculated fields further refine the analysis.

4. Creating Dashboards

Dashboards combine multiple visualizations (charts, maps, etc.) into a single interface. Users can arrange and interact with these elements to display key metrics in one view.

5. Working with Maps

Tableau supports geographic data visualization through maps, where users can plot data points based on location (latitude, longitude).

6. Telling Stories

Tableau allows creating "Stories" by combining sheets and dashboards in a sequence to narrate data-driven insights visually. These can be shared and presented interactively.

Output:-

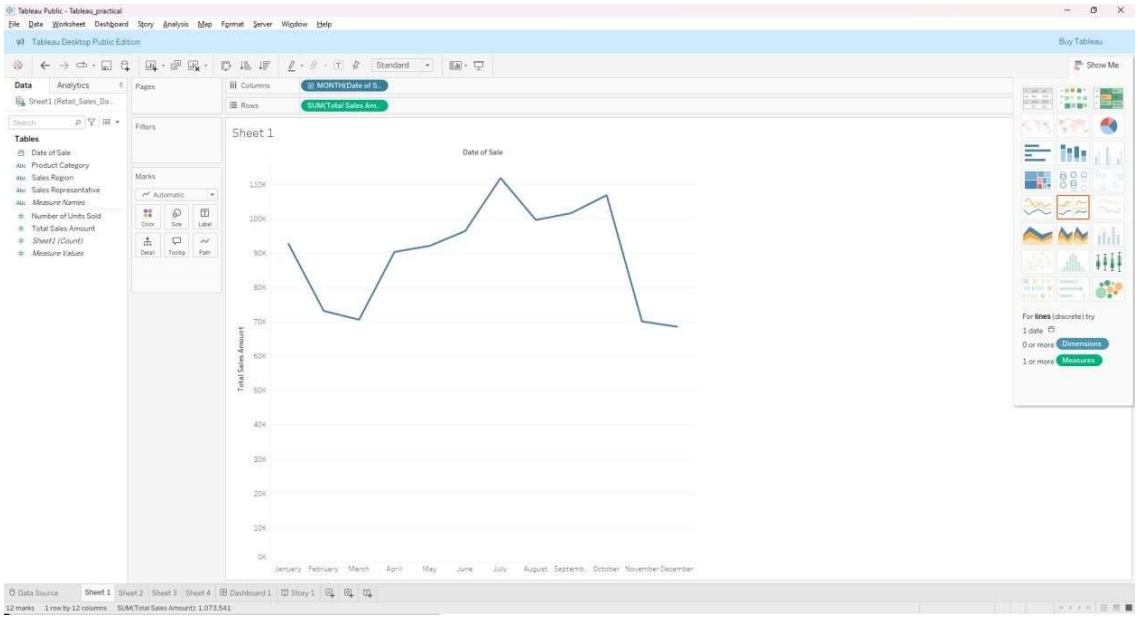


Tableau Public - Tableau_practical
Sheet 1
Date of Sale
Total Sales Amount

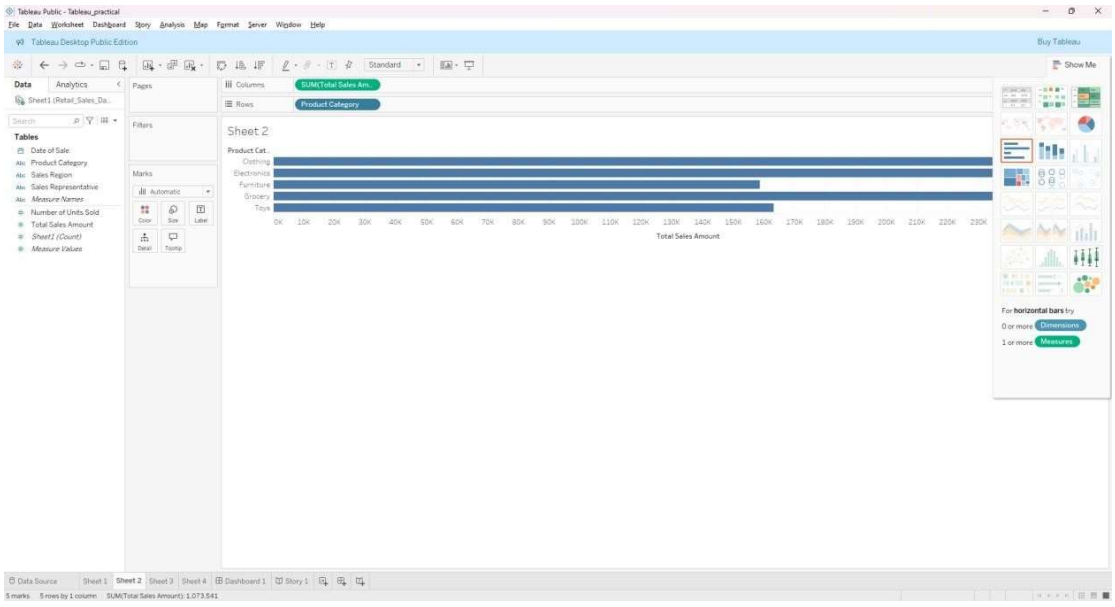


Tableau Public - Tableau_practical
Sheet 2
Product Cat..
Clothing
Electronics
Furniture
Grocery
Toys
Total Sales Amount

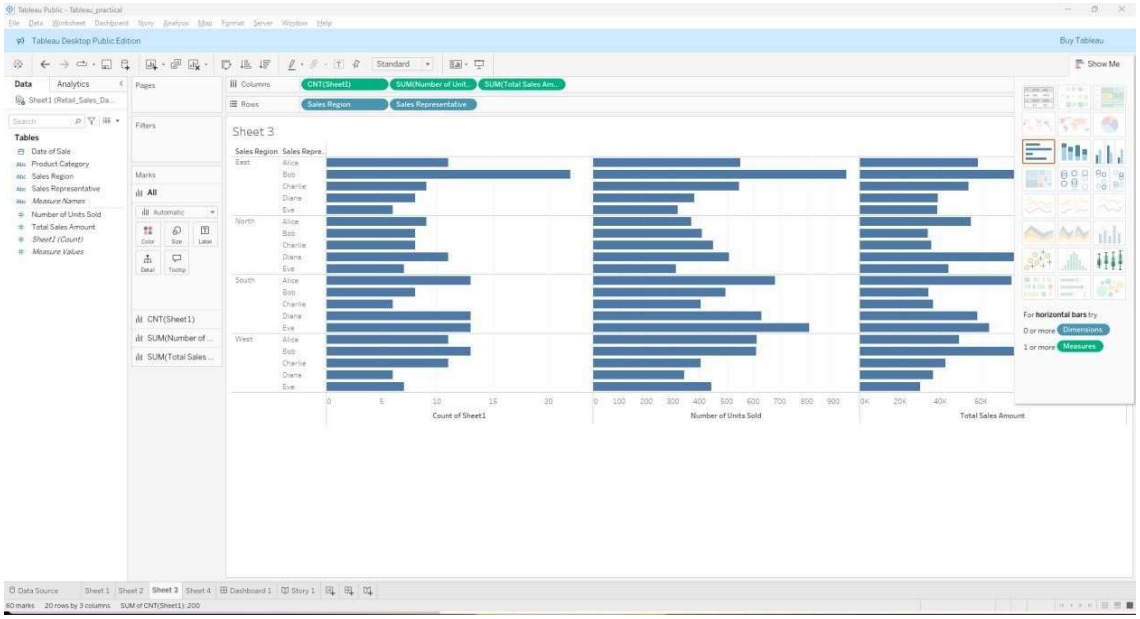


Tableau Desktop Public Edition
Sheet 3
Count of Sheet1
Number of Units Sold
Total Sales Amount

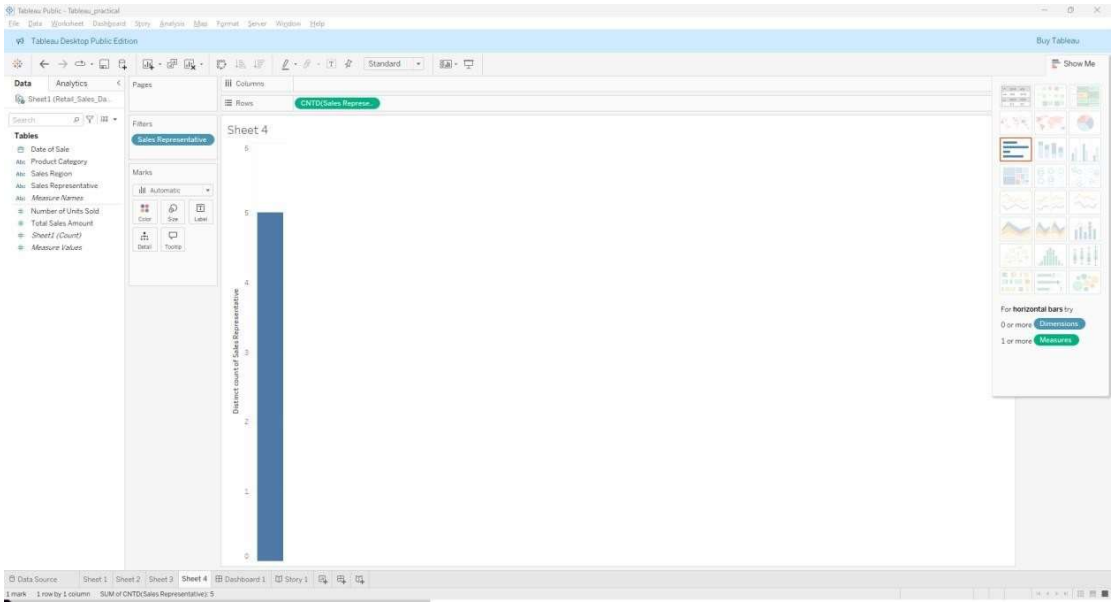


Tableau Desktop Public Edition
Sheet 4
Distinct count of Sales Representative

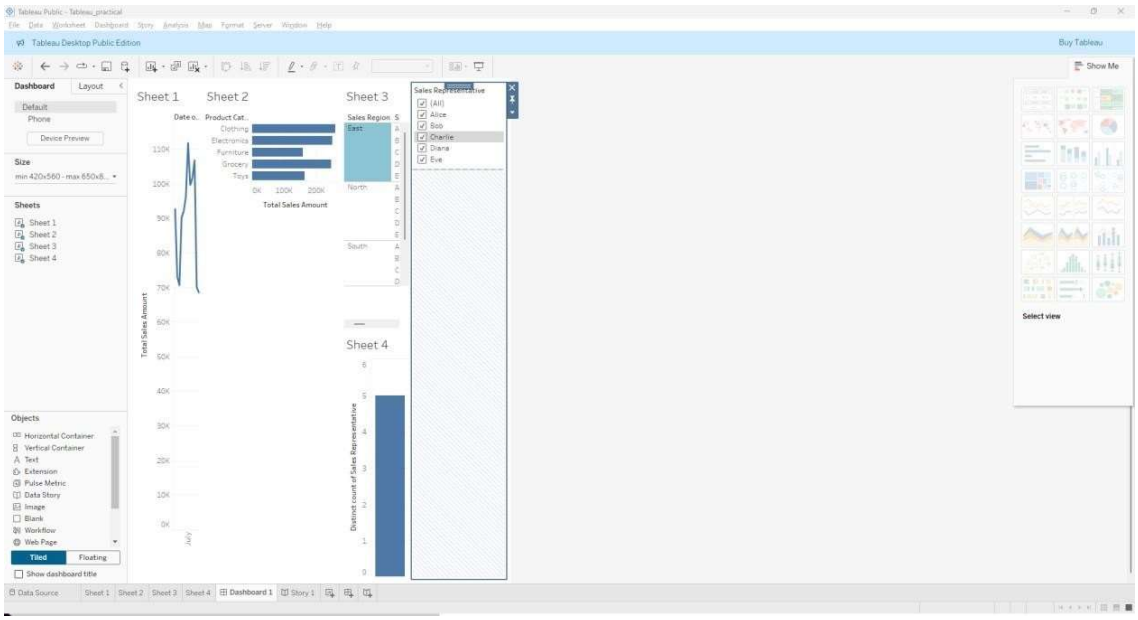


Tableau Desktop Public Edition
Buy Tableau
Show Me
Dashboard
Layout
Sheet 1
Sheet 2
Sheet 3
Sheet 4
Objects
Tiled
Floating
Select view

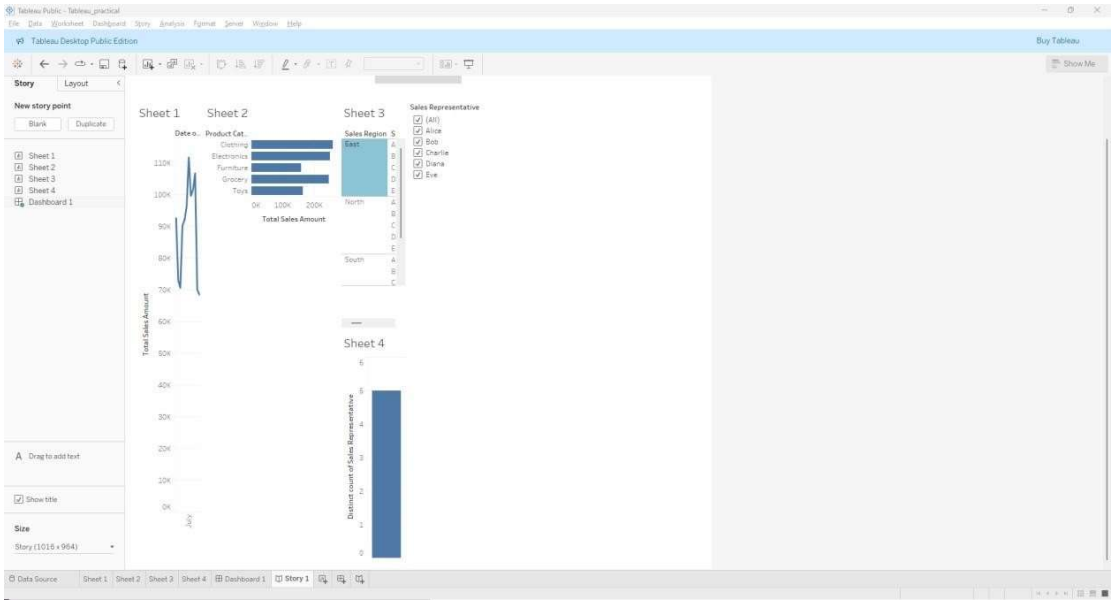


Tableau Desktop Public Edition
Buy Tableau
Show Me
Story
Layout
New story point
Blank
Duplicate
Sheet 1
Sheet 2
Sheet 3
Sheet 4
Dashboard 1
Drag to add text
Show title
Size