INDEX

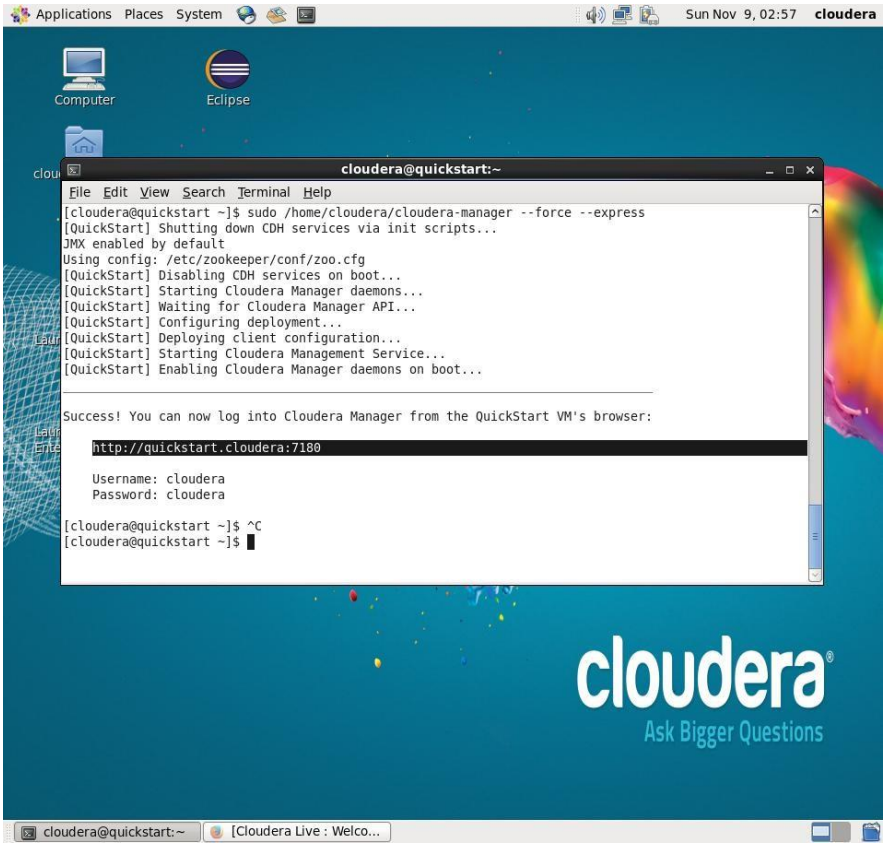
SR NO	NAME OF THE EXPERIMENT	DATE	FACULTY SIGN	CO
1	**HDFS:** List of Commands (mkdir, touchz, copy from local/put, copy to local/get, move from local, cp, rmr, du, dus, stat)			**CO1**
2	**Map Reduce:** 1. Write a program in Map Reduce for Word Count operation. 2. Write a program in Map Reduce for Union operation. 3. Write a program in Map Reduce for Intersection operation. 4. Write a program in Map Reduce for Matrix Multiplication.			**CO2**
3	**MongoDB:** 1. Installation 2. Sample Database Creation 3. Query the Sample Database using MongoDB querying commands a. Create Collection b. Insert Document c. Query Document d. Delete Document e. Indexing			**CO3**
4	**Hive:** 1. Hive Data Types 2. Create Database & Table in Hive 3. Hive Partitioning 4. Hive Built-In Operators 5. Hive Built-In Functions 6. Hive Views			**CO4**

<table>
<tr><td></td><td>7. HiveQL: Select Where, Select OrderBy, Select GroupBy, Select Joins</td><td></td><td></td><td></td></tr>
<tr><td>5</td><td>Pig:
1. Pig Latin Basic
2. Pig Data Types,
3. Download the data
4. Create your Script
5. Save and Execute the Script
6. Pig Operations: Diagnostic Operators, Grouping and Joining, Combining & Splitting, Filtering, Sorting</td><td></td><td></td><td>CO4</td></tr>
<tr><td>6</td><td>Spark:
<table>
<tr><th>Apache Spark Commands in Scala</th><th>Pair RDD (Key-Value RDD) Operations</th></tr>
<tr><td></td><td></td></tr>
<tr><td>Start Spark Shell</td><td>Create Pair RDD</td></tr>
<tr><td>Create RDD from Collection</td><td>reduceByKey</td></tr>
<tr><td>Read Text File into RDD (from HDFS)</td><td>groupByKey — With Output Viewing</td></tr>
<tr><td>Map Transformation</td><td>mapValues</td></tr>
<tr><td>Filter Transformation</td><td>sortByKey</td></tr>
<tr><td>Reduce Action</td><td>join</td></tr>
<tr><td>Collect Action</td><td>cogroup</td></tr>
<tr><td>Save RDD to HDFS</td><td>aggregateByKey</td></tr>
<tr><td>Cache/Persist RDD</td><td>foldByKey</td></tr>
<tr><td></td><td>Read from Local File</td></tr>
</table></td><td></td><td></td><td>CO5</td></tr>
<tr><td>7</td><td>Visualization using Tableau:
Tableau: Tool Overview, Importing Data, Analyzing with Charts, Creating Dashboards, working with maps</td><td></td><td></td><td>CO6</td></tr>
</table>

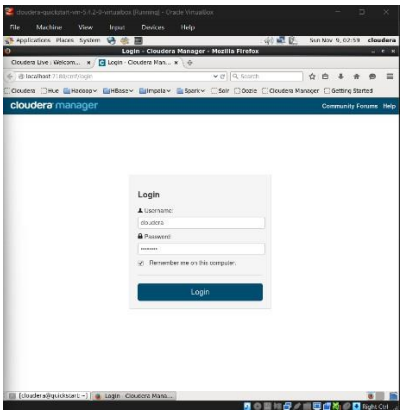
Practical No:1 HDFS

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

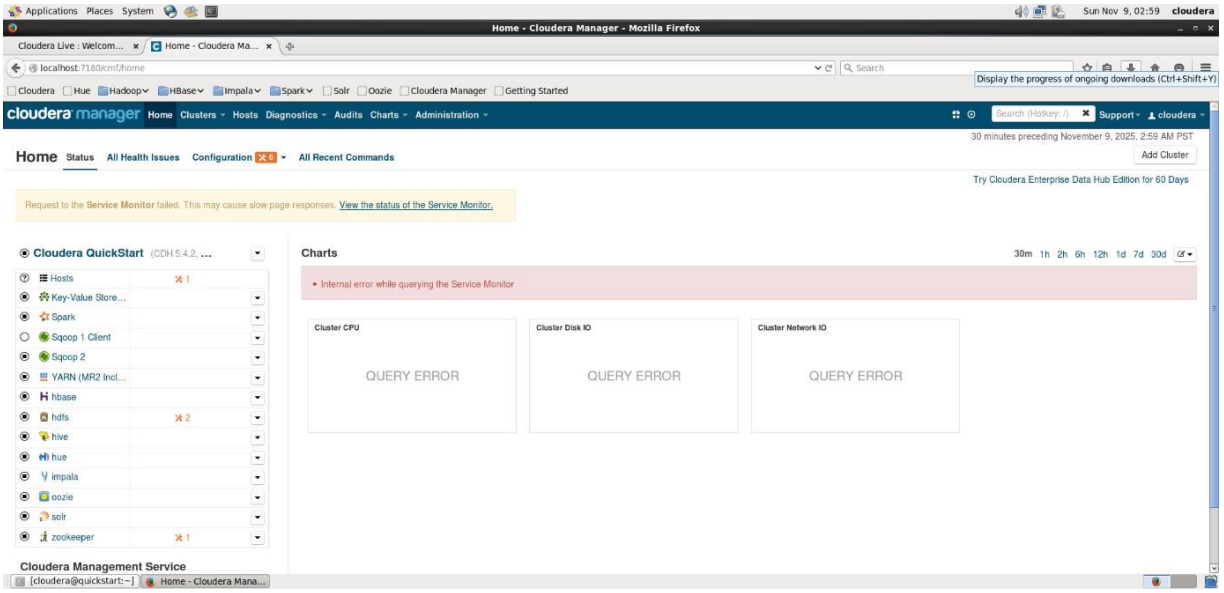
Step 1: Start Cloudera Manager in your Cloudera VM.



Step 2: Enter valid Login credentials – username and password -> will navigate to cloudera manager dashboard.

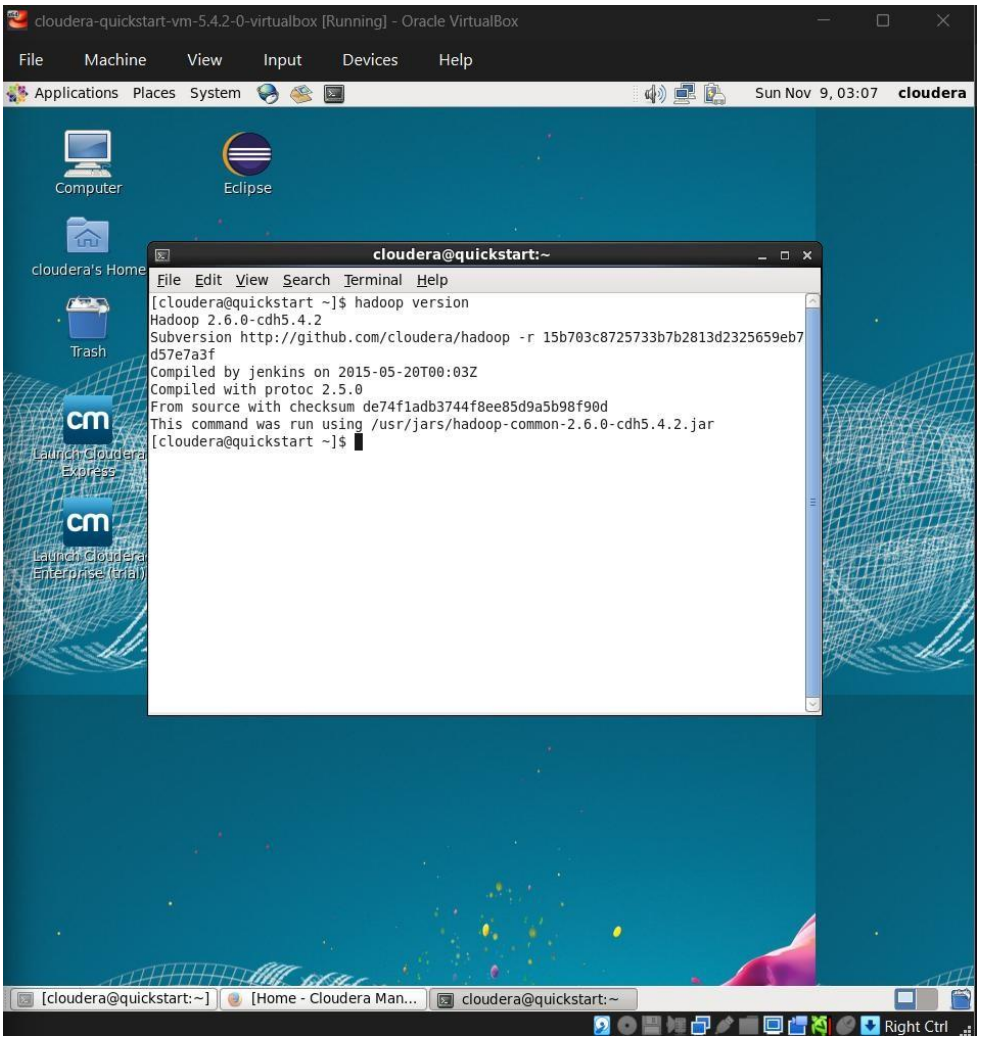


Step 3: Activate HDFS, Map Reduce, Hbase and YARN



Make sure that that should be green tick, in order to perform commands

Step 4: Verify active services by checking Hadoop version.

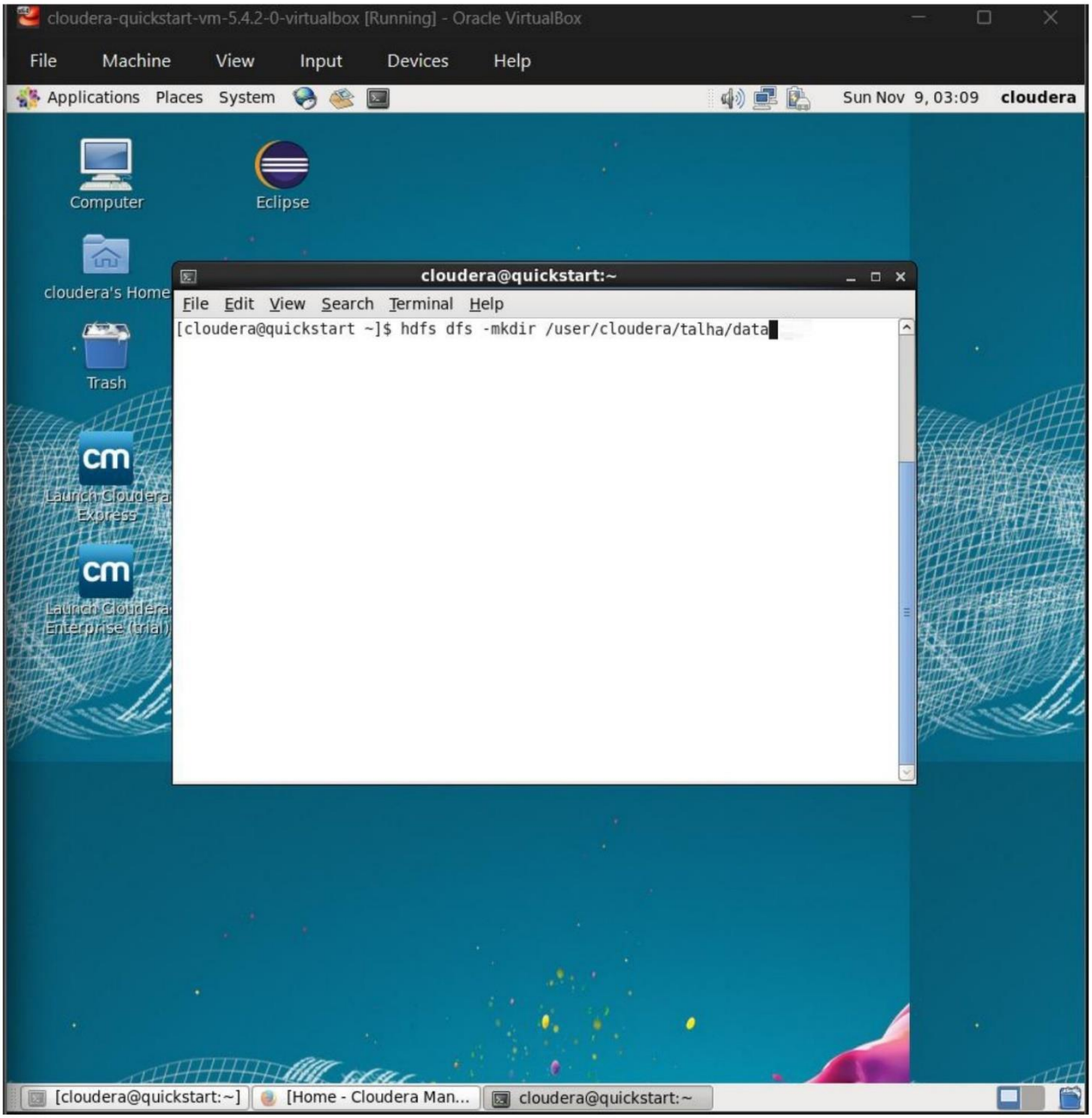


List of Commands to Perform in the terminal.

1. mkdir

Creates a new directory in HDFS.

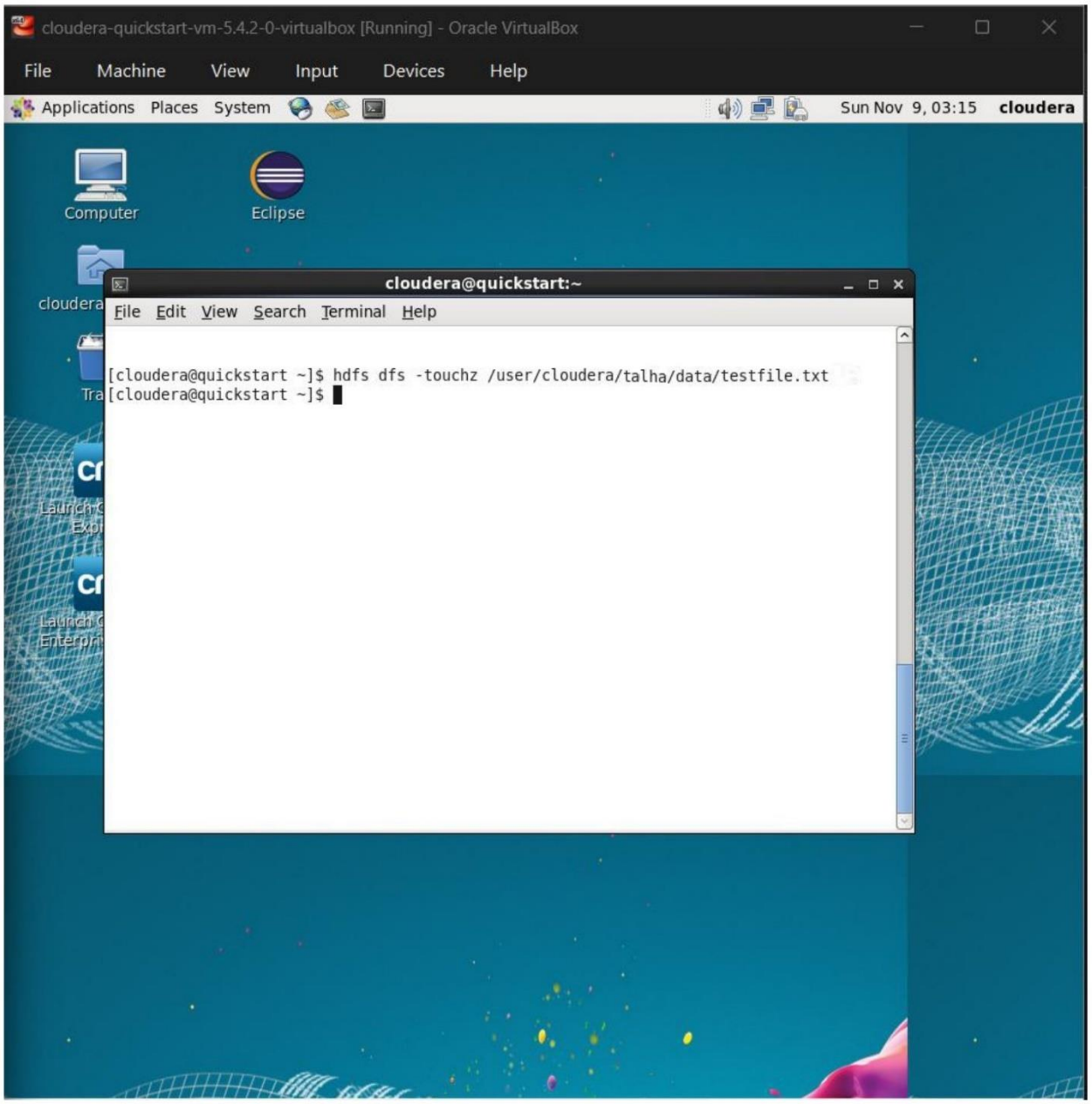
hdfs dfs -mkdir /user/cloudera/talha/data



2. touchz

Creates an empty file in HDFS.

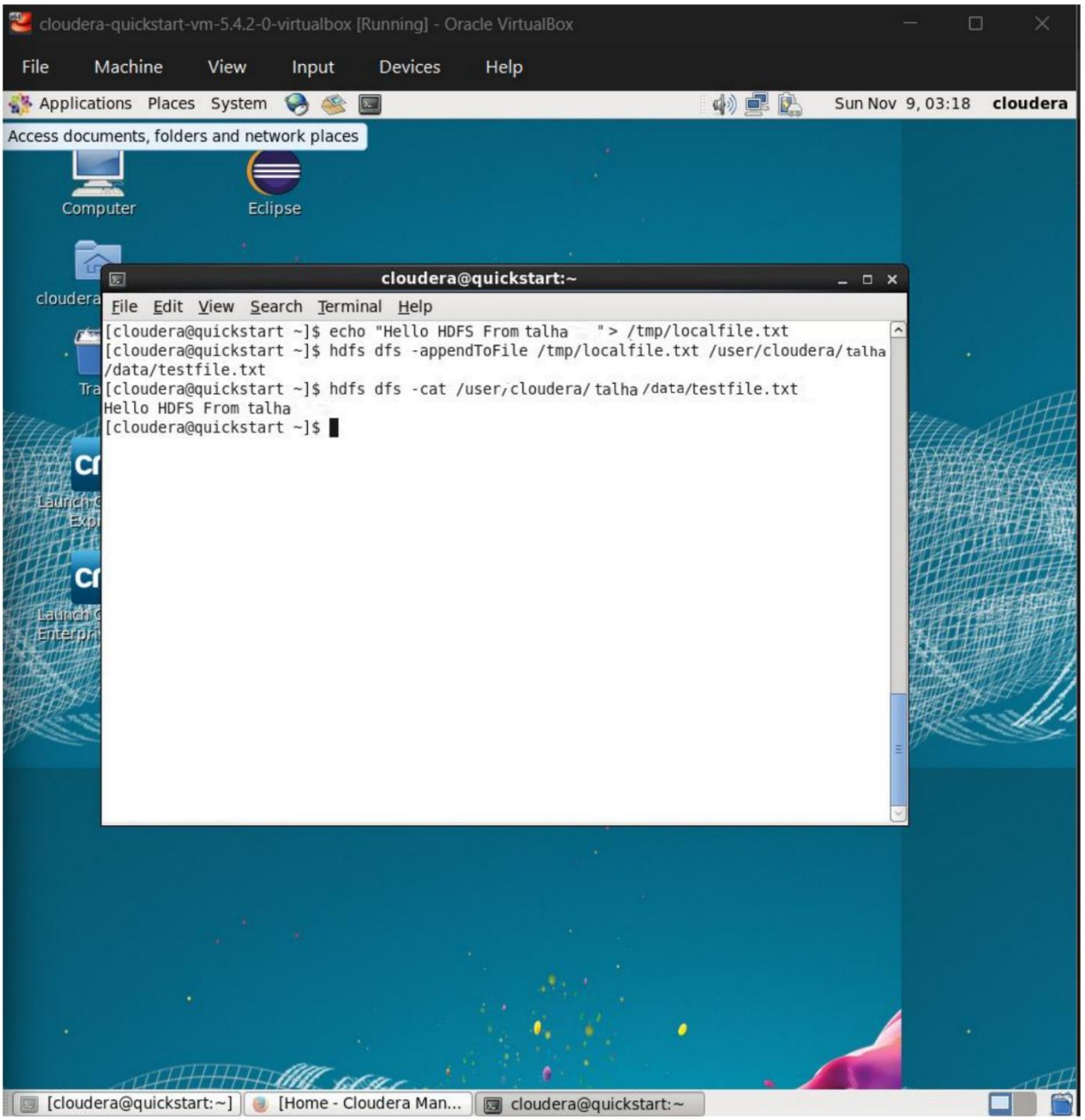
hdfs dfs -touchz /user/cloudera/talha/data/testfile.txt



3. appendToFile

Appends local file content to an HDFS file.

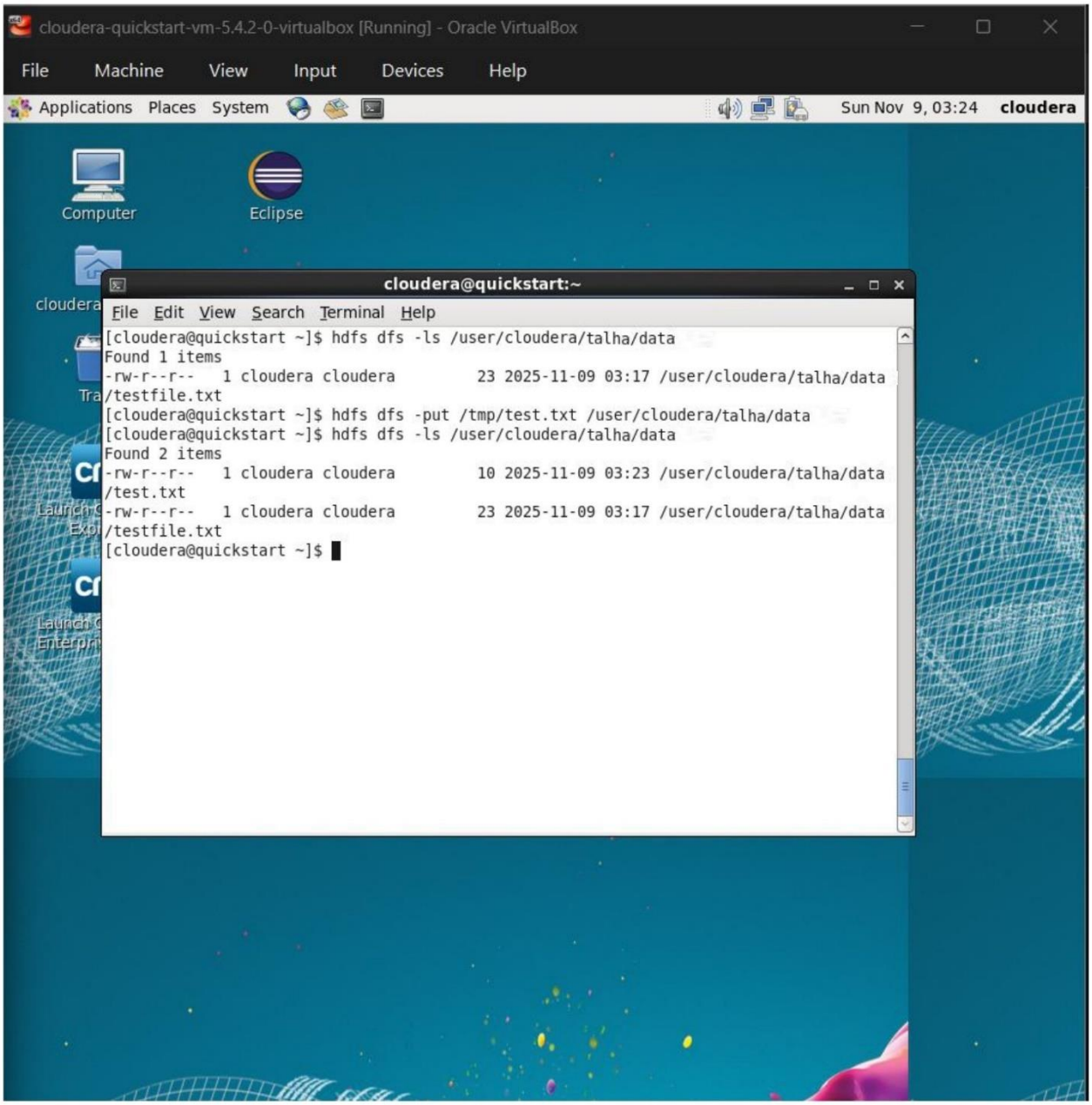
hdfs dfs -appendToFile /tmp/localfile.txt /user/cloudera/talha/data/testfile.txt



4. put

Copies file from local system to HDFS.

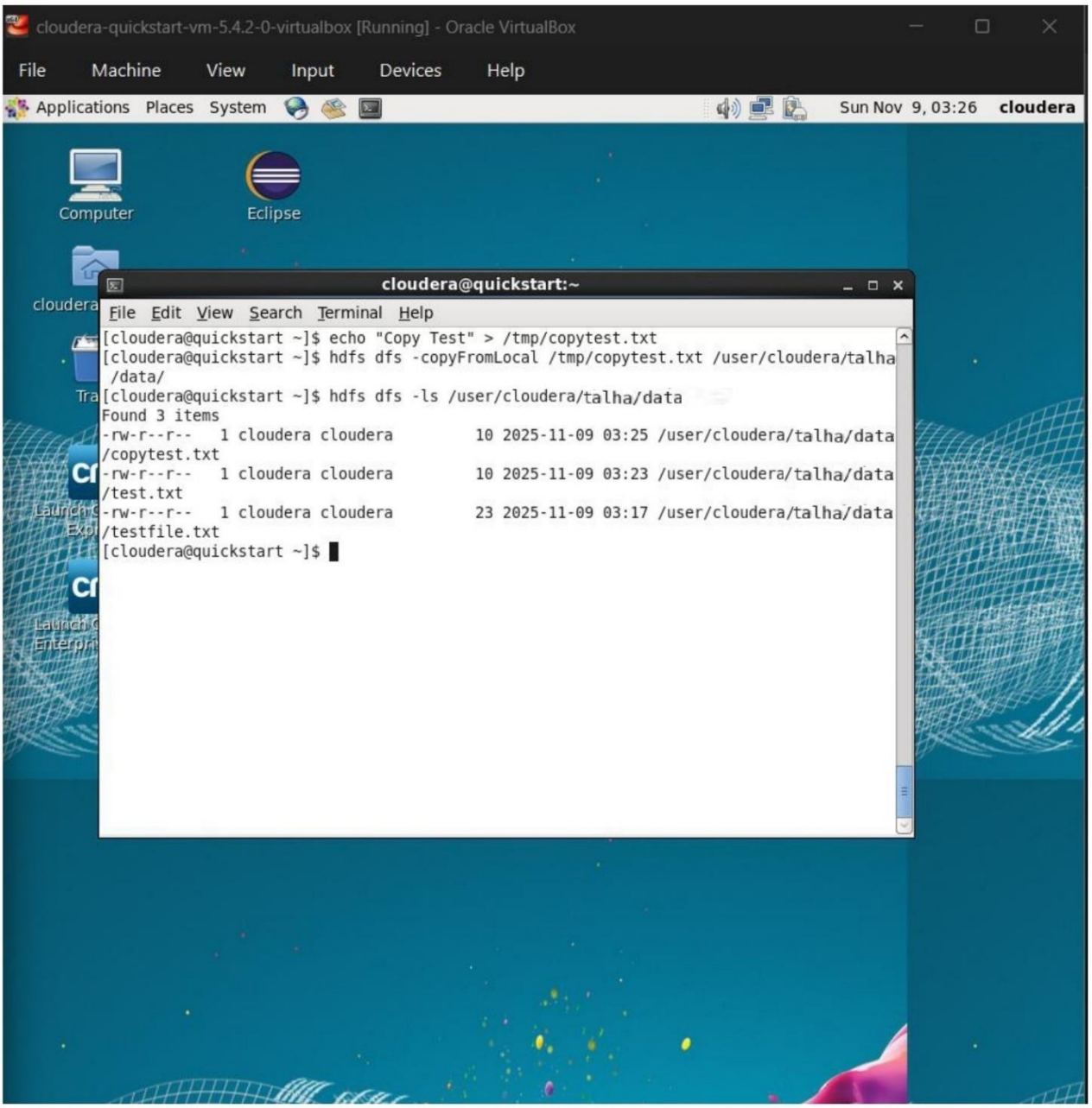
hdfs dfs -put /tmp/test.txt /user/cloudera/talha/data/



5. copyFromLocal

Alternate command to copy file from local to HDFS.

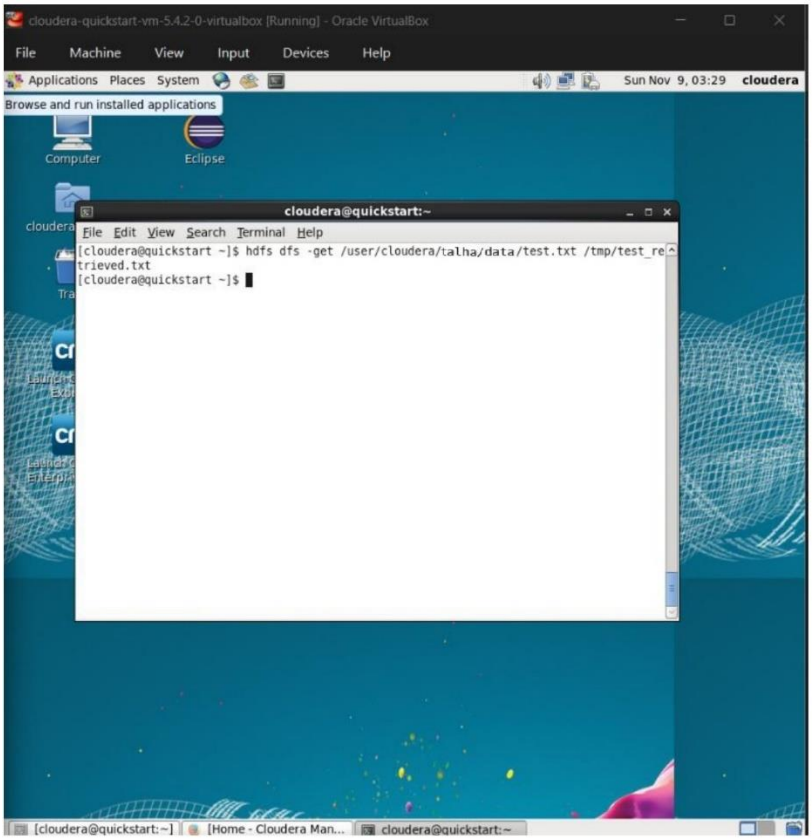
hdfs dfs -copyFromLocal /tmp/copytest.txt /user/cloudera/talha/data/

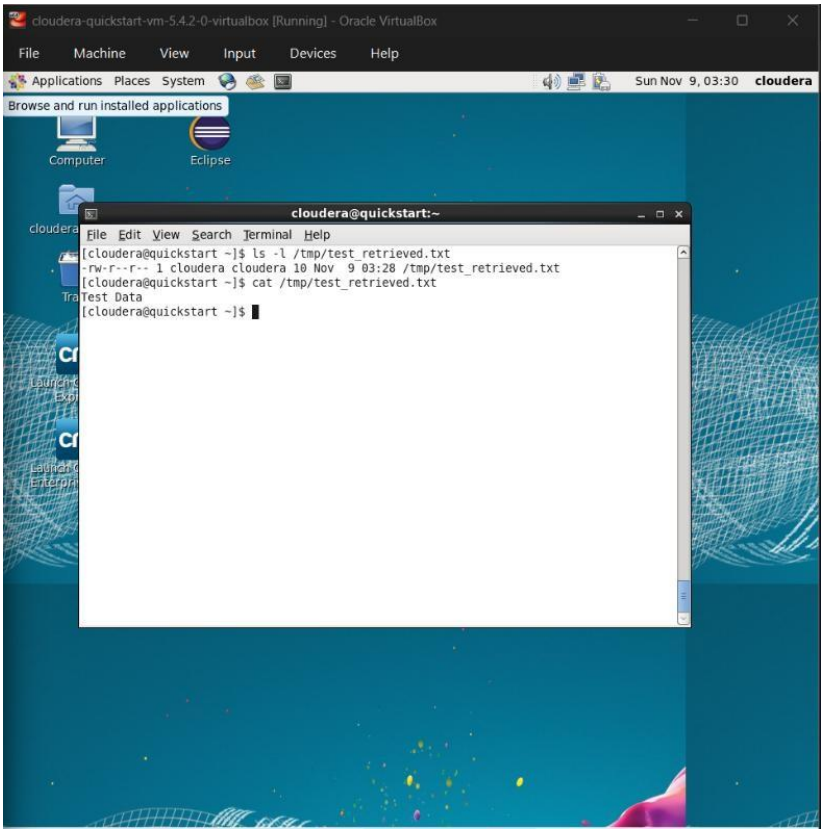


6. get

Copies file from HDFS to local system.

hdfs dfs -get /user/cloudera/talha/data/test.txt /tmp/test_retrieved.txt

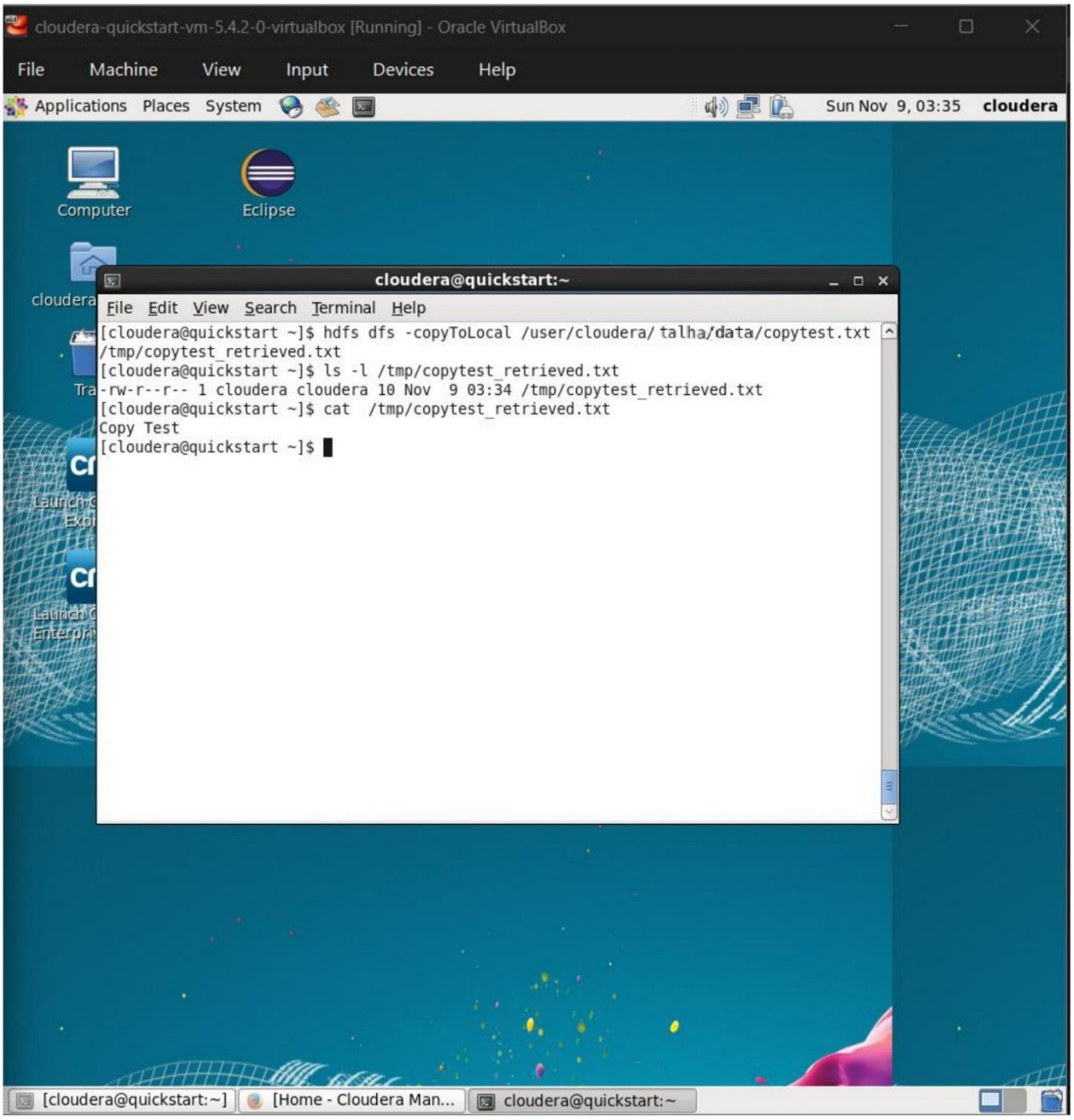

cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 03:29 cloudera
Browse and run installed applications
Computer
Eclipse
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hdfs dfs -get /user/cloudera/talha/data/test.txt /tmp/test_retrieved.txt
[cloudera@quickstart ~]$
[cloudera@quickstart:~] [Home - Cloudera Man... cloudera@quickstart:~


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 03:30 cloudera
Browse and run installed applications
Computer
Eclipse
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ ls -l /tmp/test_retrieved.txt
-rw-r--r-- 1 cloudera cloudera 10 Nov 9 03:28 /tmp/test_retrieved.txt
[cloudera@quickstart ~]$ cat /tmp/test_retrieved.txt
Test Data
[cloudera@quickstart ~]$

7. copyToLocal

Alternate command to copy file from HDFS to local.

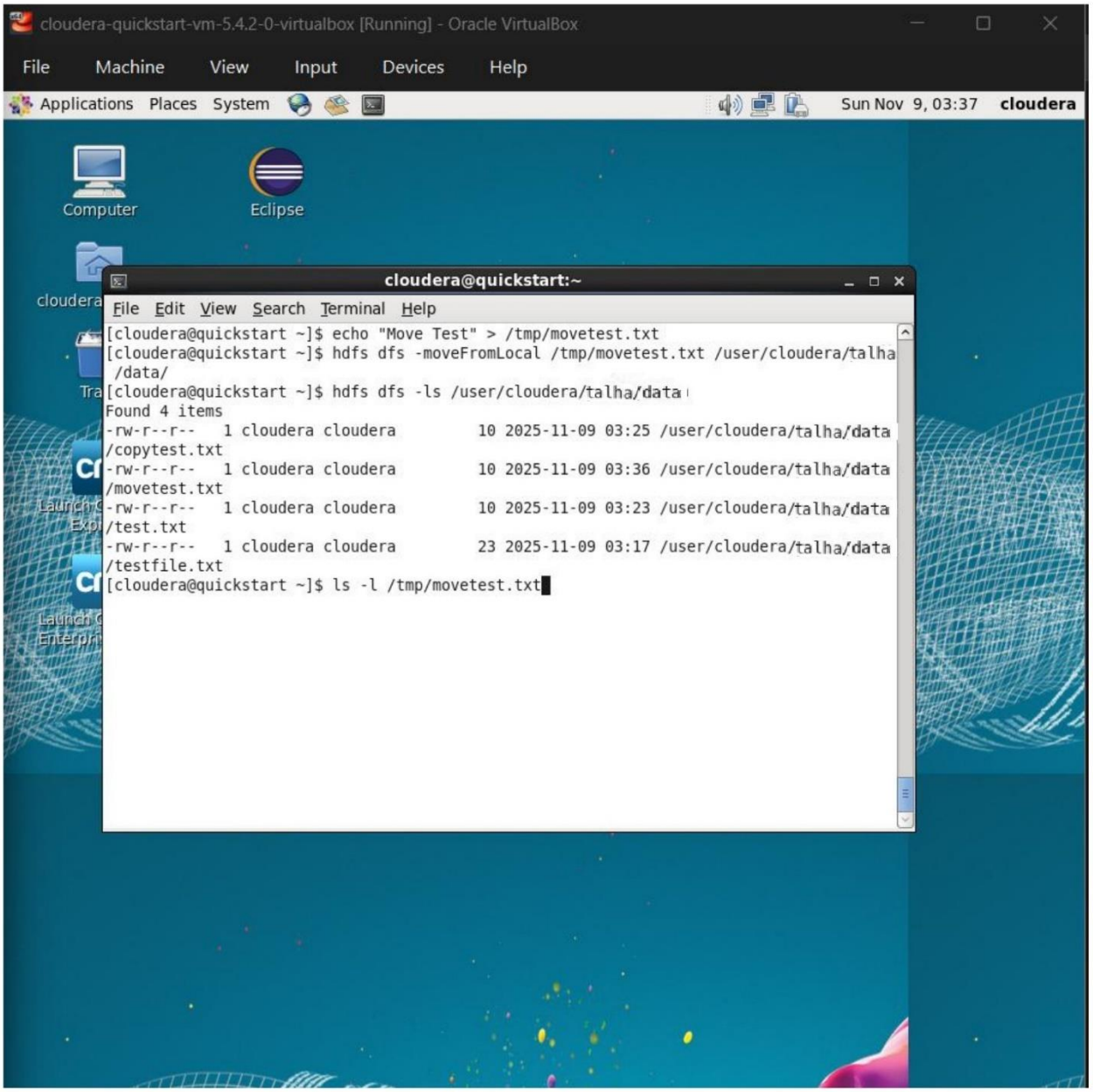
hdfs dfs -copyToLocal /user/cloudera/talha/data/copytest.txt /tmp/copytest_retrieved.txt



8. moveFromLocal

Moves a local file to HDFS (deletes local copy).

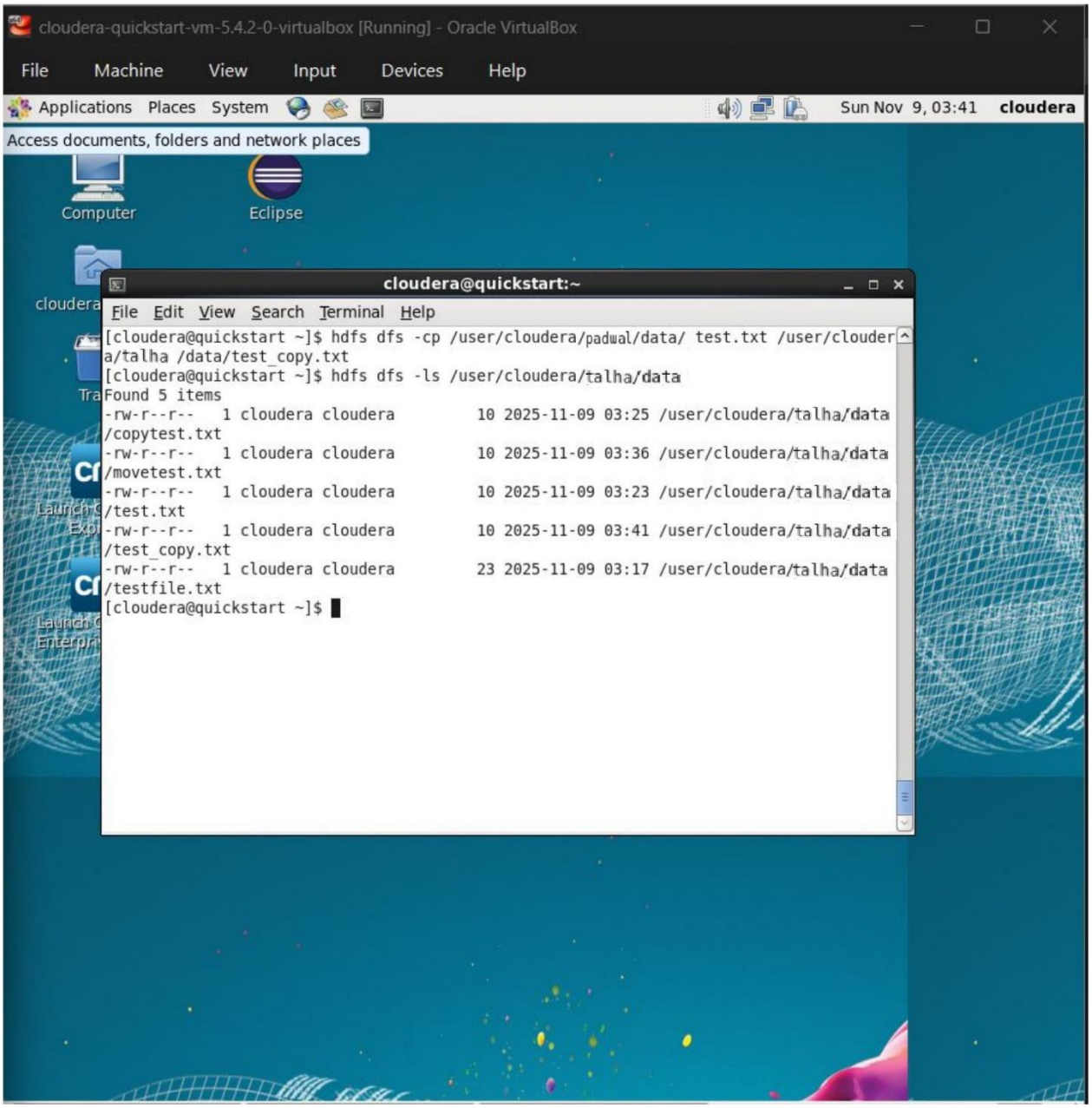
hdfs dfs -moveFromLocal /tmp/movetest.txt /user/cloudera/talha/data/



9. cp

Copies a file within HDFS directories.

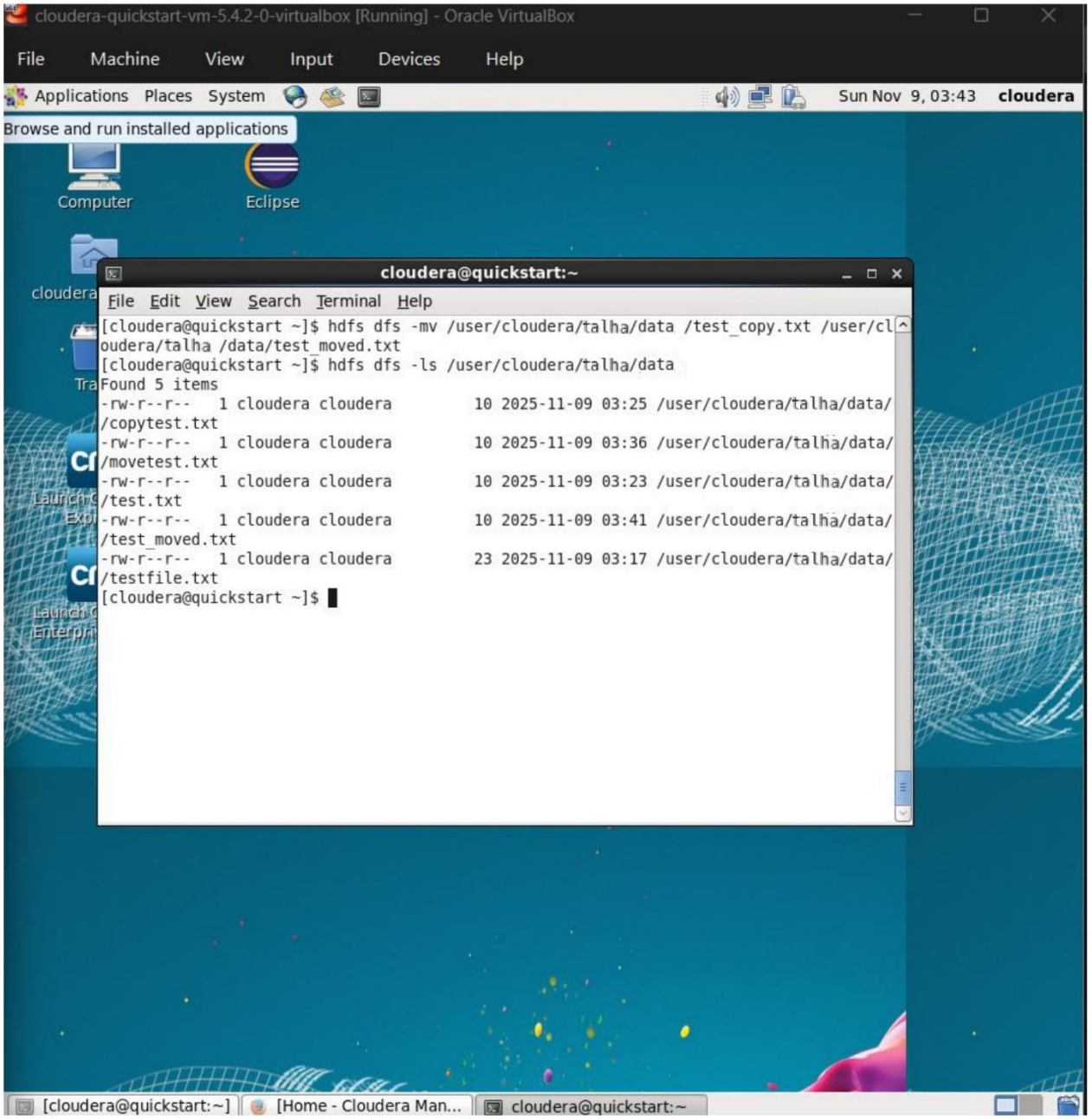
hdfs dfs -cp /user/cloudera/talha/data/test.txt /user/cloudera/talha/data/test_copy.txt



10. mv

Moves or renames a file within HDFS.

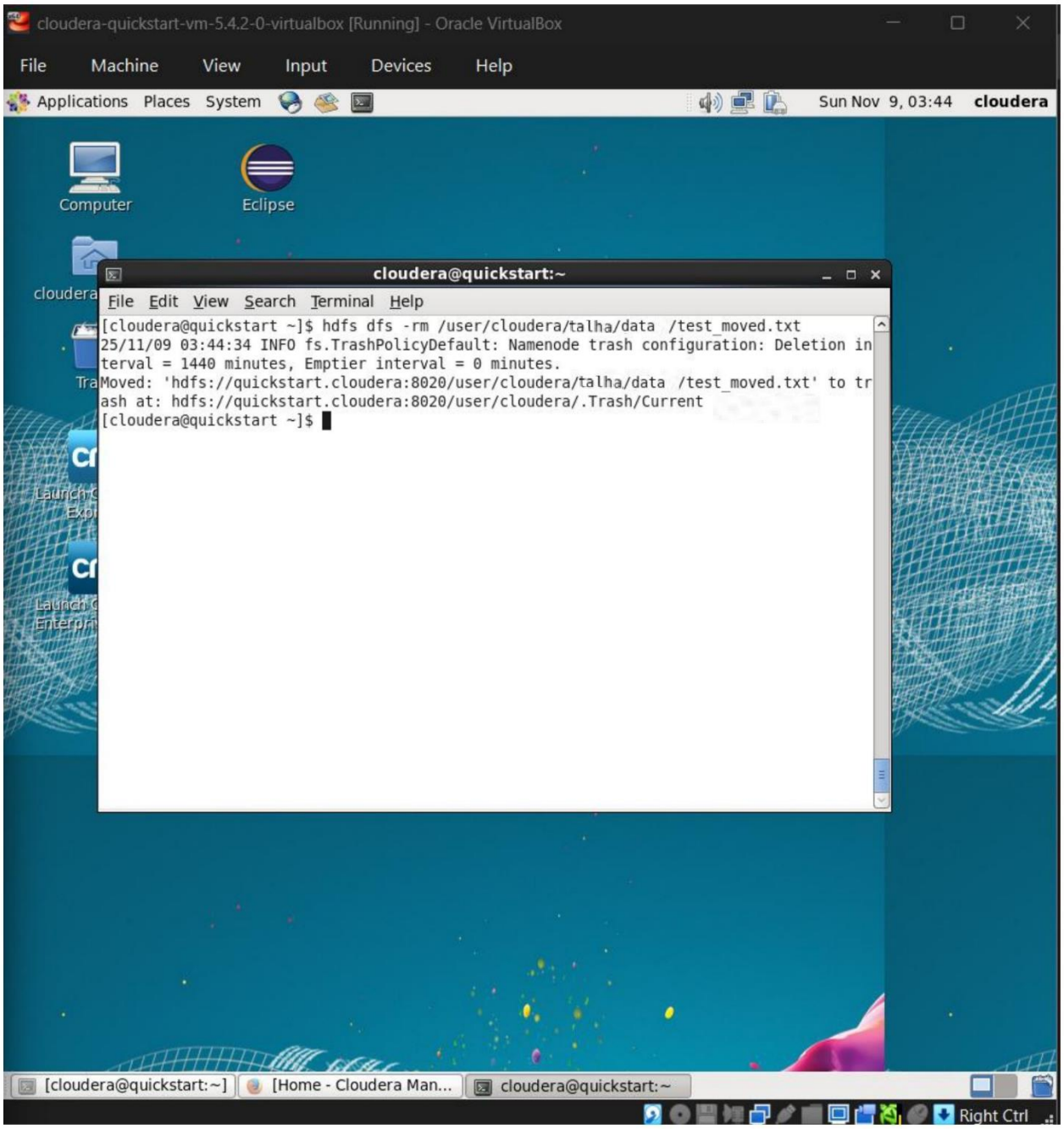
hdfs dfs -mv /user/cloudera/talha/data/test_copy.txt /user/cloudera/talha/data/test_moved.txt



11. rm

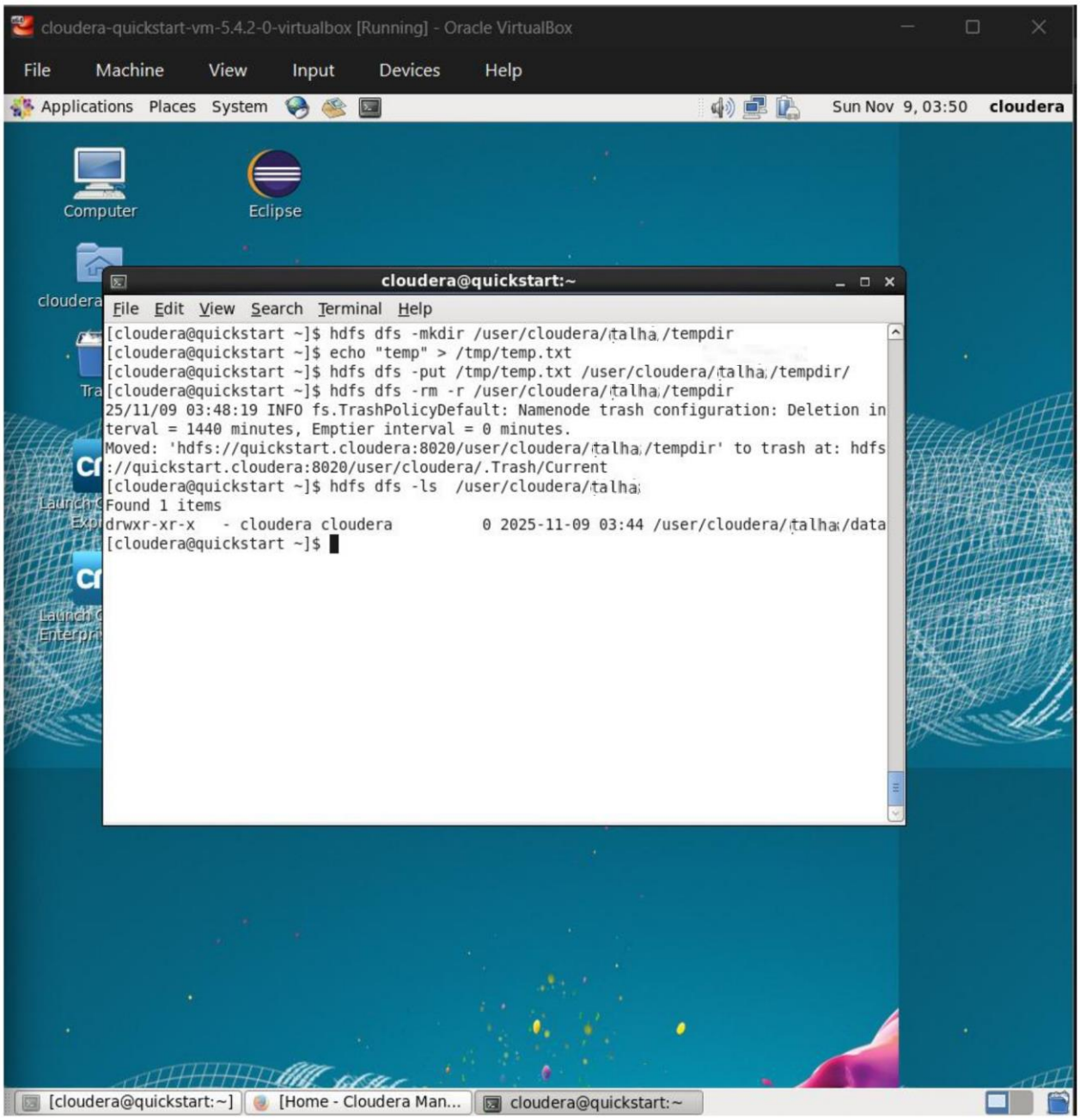
Deletes a specific file from HDFS.

hdfs dfs -rm /user/cloudera/talha/data/test_moved.txt



12. rm -r

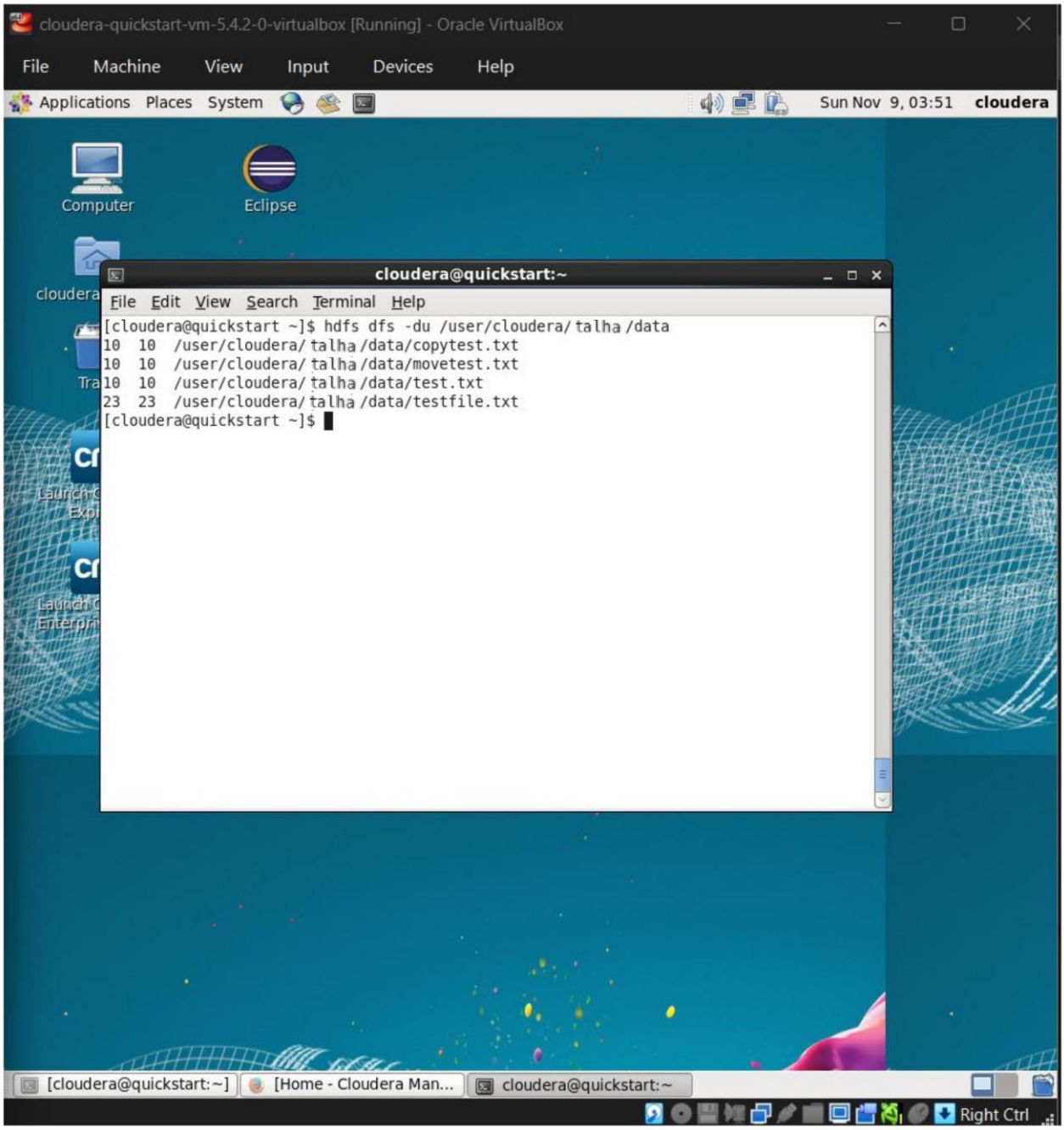
Deletes a directory recursively from HDFS.



13. du

Displays disk usage for each file in a directory.

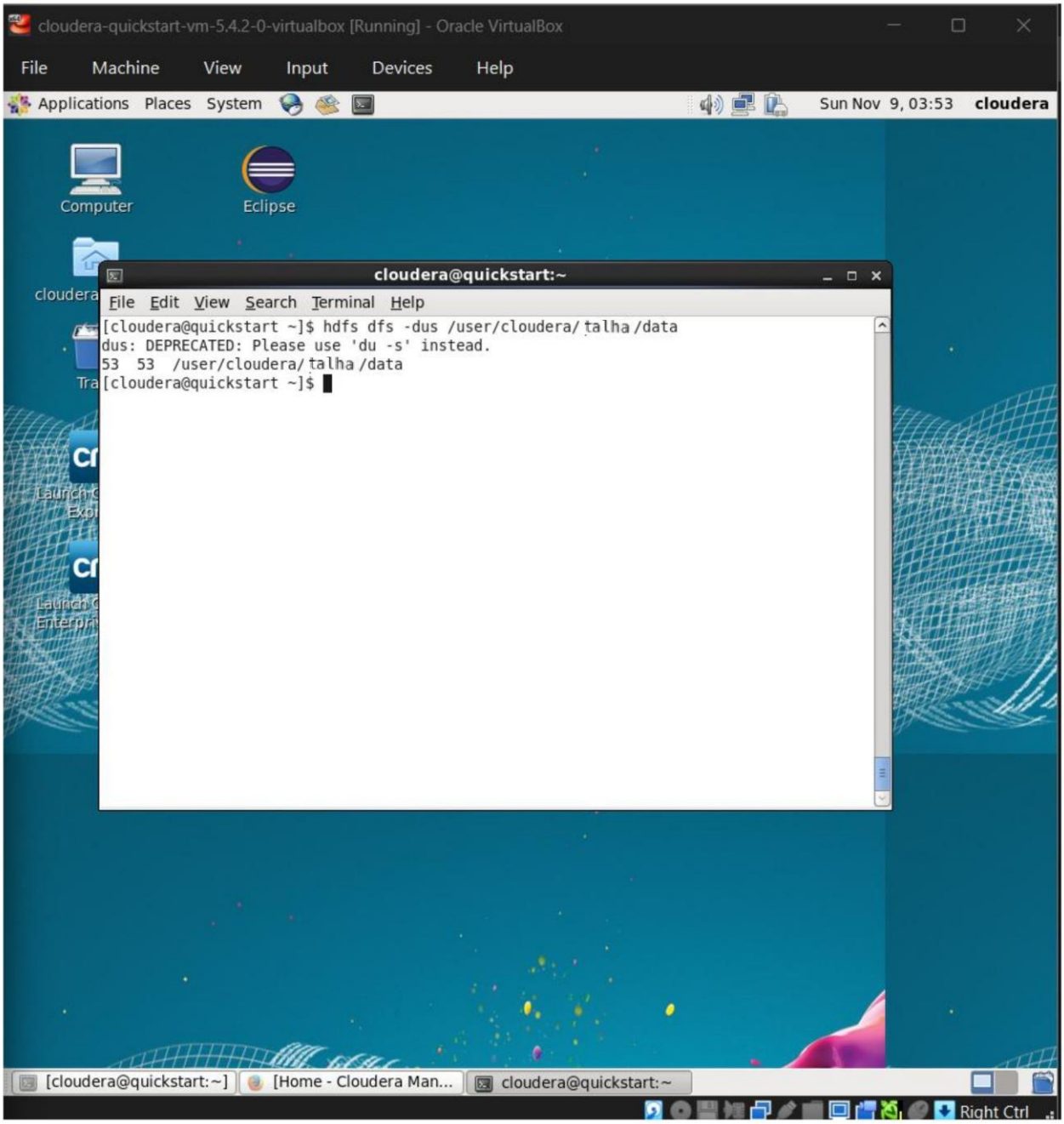
hdfs dfs -du /user/cloudera/talha/data



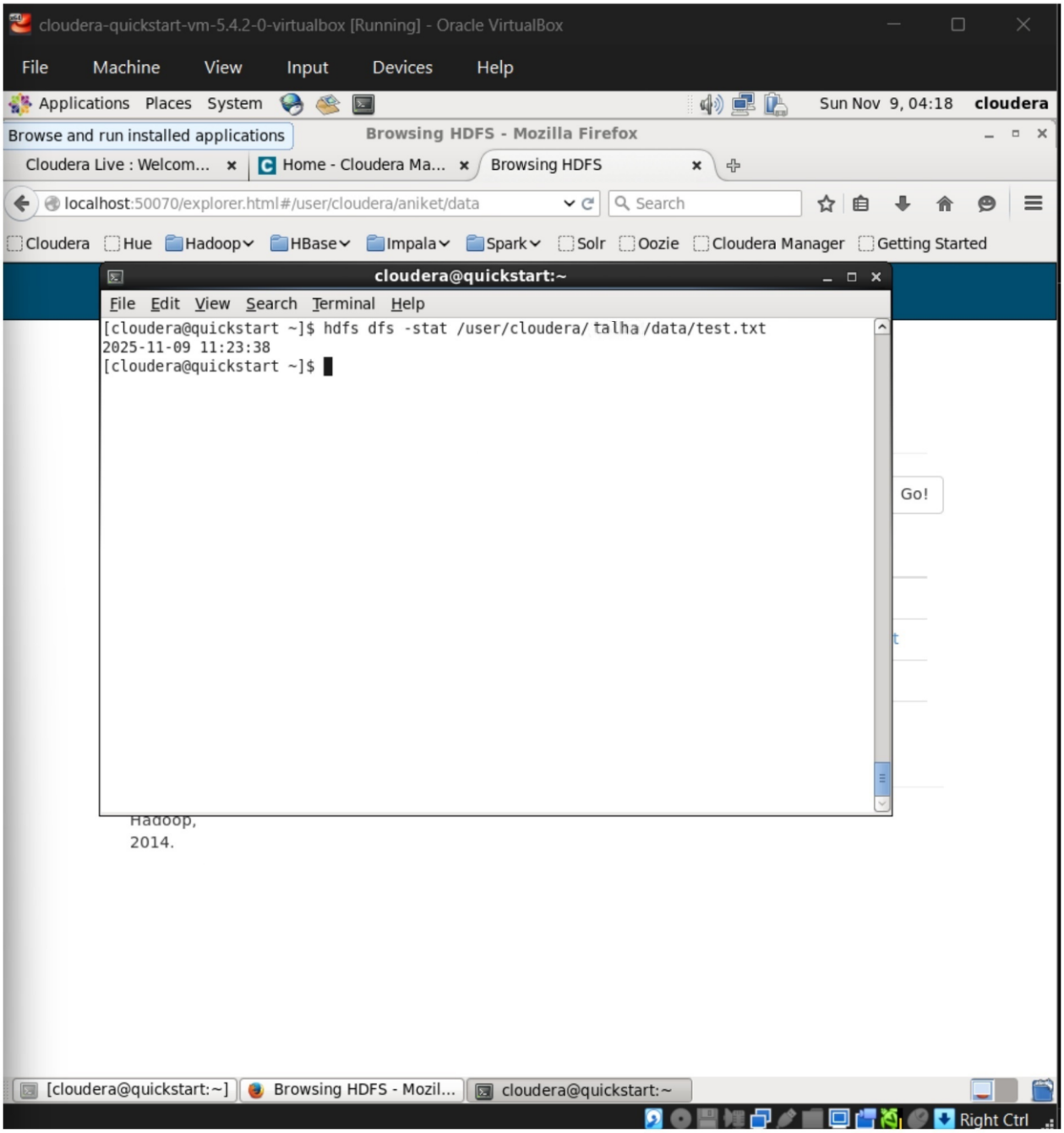
14. dus

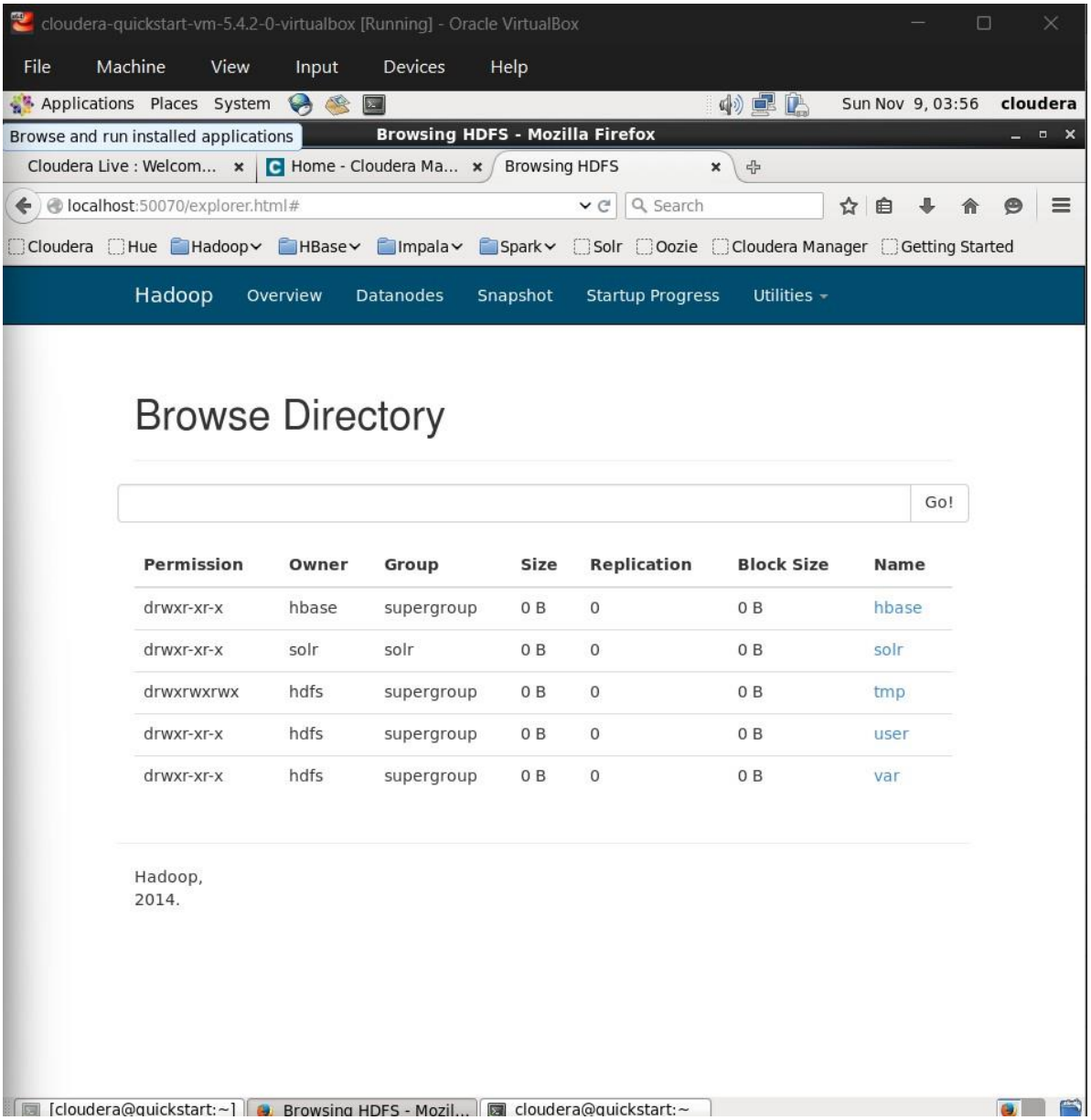
Displays total disk usage summary for a directory.

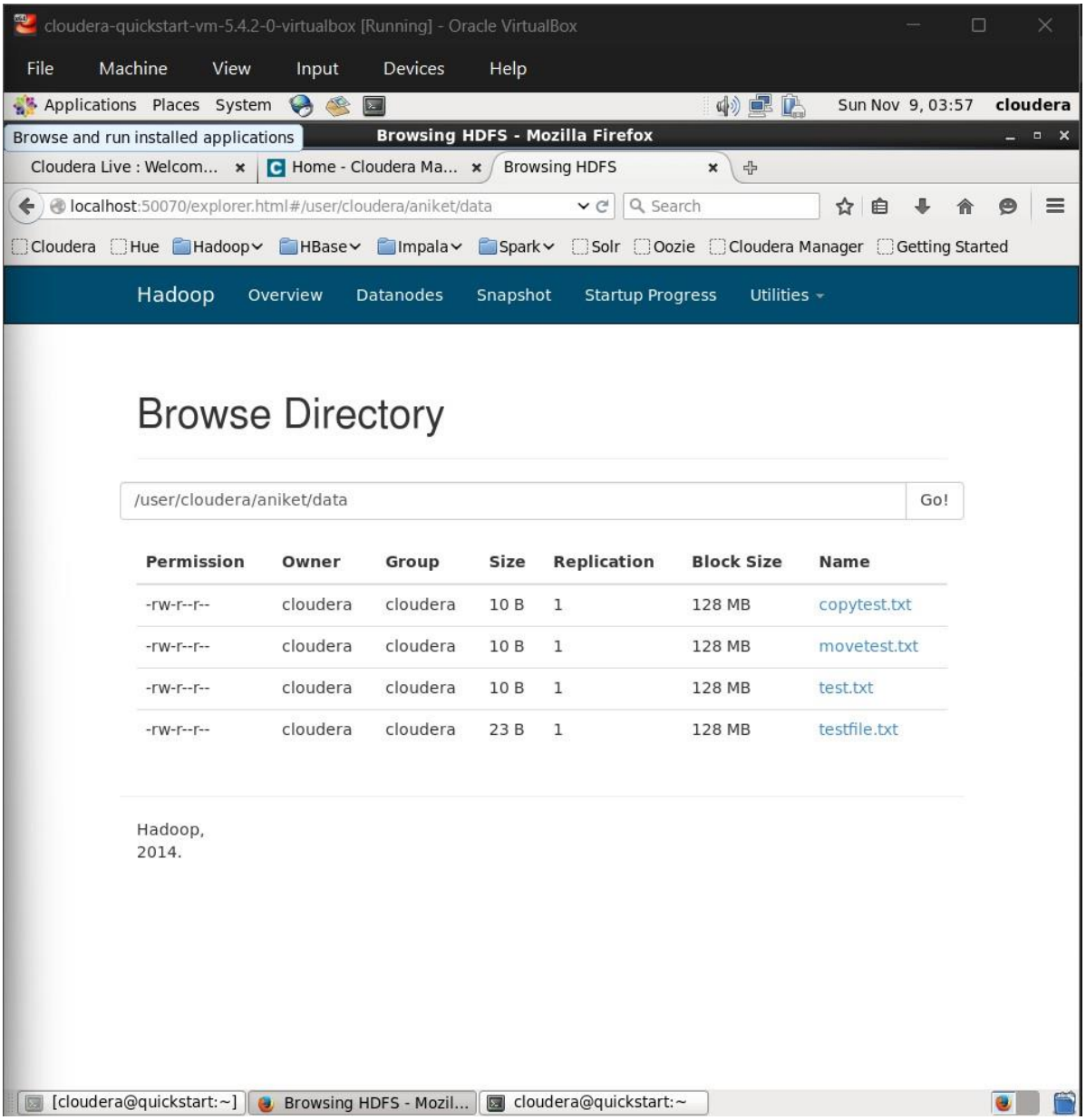
hdfs dfs -dus /user/cloudera/talha/data



15. Displays detailed **status information** about a file or directory in HDFS


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 04:18 cloudera
Browse and run installed applications
Browsing HDFS - Mozilla Firefox
Cloudera Live : Welcom...
Home - Cloudera Ma...
Browsing HDFS
localhost:50070/explorer.html#/user/cloudera/aniket/data
Search
Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hdfs dfs -stat /user/cloudera/ talha /data/test.txt
2025-11-09 11:23:38
[cloudera@quickstart ~]$
Go!
Hadoop,
2014.
[cloudera@quickstart:~]
Browsing HDFS - Mozil...
cloudera@quickstart:~
Right Ctrl


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 03:56 cloudera
Browse and run installed applications
Browsing HDFS - Mozilla Firefox
Cloudera Live : Welcom...
Home - Cloudera Ma...
Browsing HDFS
localhost:50070/explorer.html#
Search
Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started
Hadoop Overview Datanodes Snapshot Startup Progress Utilities
Browse Directory
Go!
Permission Owner Group Size Replication Block Size Name
drwxr-xr-x hbase supergroup 0 B 0 0 B hbase
drwxr-xr-x solr solr 0 B 0 0 B solr
drwxrwxrwx hdfs supergroup 0 B 0 0 B tmp
drwxr-xr-x hdfs supergroup 0 B 0 0 B user
drwxr-xr-x hdfs supergroup 0 B 0 0 B var
Hadoop,
2014.
[cloudera@quickstart:~]
Browsing HDFS - Mozil...
cloudera@quickstart:~


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 03:57 cloudera
Browse and run installed applications
Browsing HDFS - Mozilla Firefox
Cloudera Live : Welcom...
Home - Cloudera Ma...
Browsing HDFS
localhost:50070/explorer.html#/user/cloudera/aniket/data
Search
Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started
Hadoop Overview Datanodes Snapshot Startup Progress Utilities
Browse Directory
/user/cloudera/aniket/data
Go!
Permission Owner Group Size Replication Block Size Name
-rw-r--r-- cloudera cloudera 10 B 1 128 MB copytest.txt
-rw-r--r-- cloudera cloudera 10 B 1 128 MB movetest.txt
-rw-r--r-- cloudera cloudera 10 B 1 128 MB test.txt
-rw-r--r-- cloudera cloudera 23 B 1 128 MB testfile.txt
Hadoop,
2014.
[cloudera@quickstart:~]
Browsing HDFS - Mozil...
cloudera@quickstart:~

Practical 2: Map Reduce

1. Write a program in Map Reduce for Word Count operation.

Steps:

Step 1: Open virtual box and then start cloudera quickstart

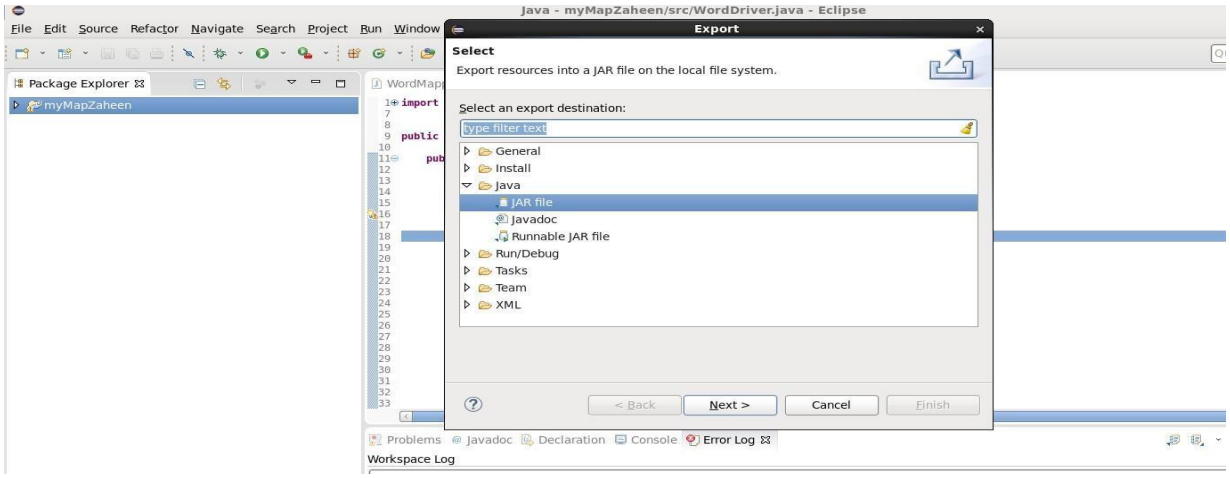
Step 2: Open eclipse present on the cloudera desktop

Step 3: Create java project
File->New-> Java Project
Give project name:
WordCount Click Next

Step 4: Add Hadoop libraries to project
Select Libraries tab->click on Add External Jars File System->user->lib->Hadoop
Select all library(jar) files->ok Again click on Add External Jars
File System->user->lib->Hadoop->client
Select all library(jar) files from client->ok->finish

Step 5: Write java code for word count
Right click on src folder of project
WordCount New->class
Write class name
WordMapper Click Finish
Write the code for WordMapper
Same way create classes SumReducer and WordCountDriver

Step 6: Export the project as jar
Right click on project WordCount and select Export>> Java>>JAR file>>Next



Select the export destination-Click browse-give file name wordcount.jar

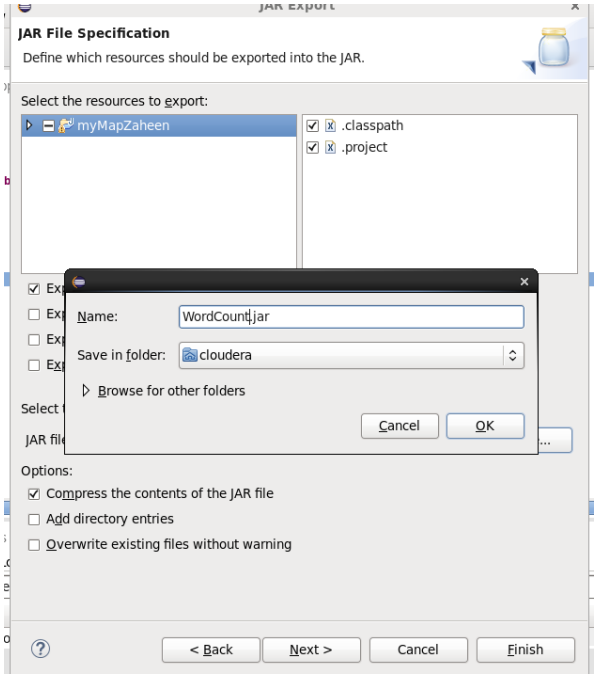
Click ok>>Finish>>ok

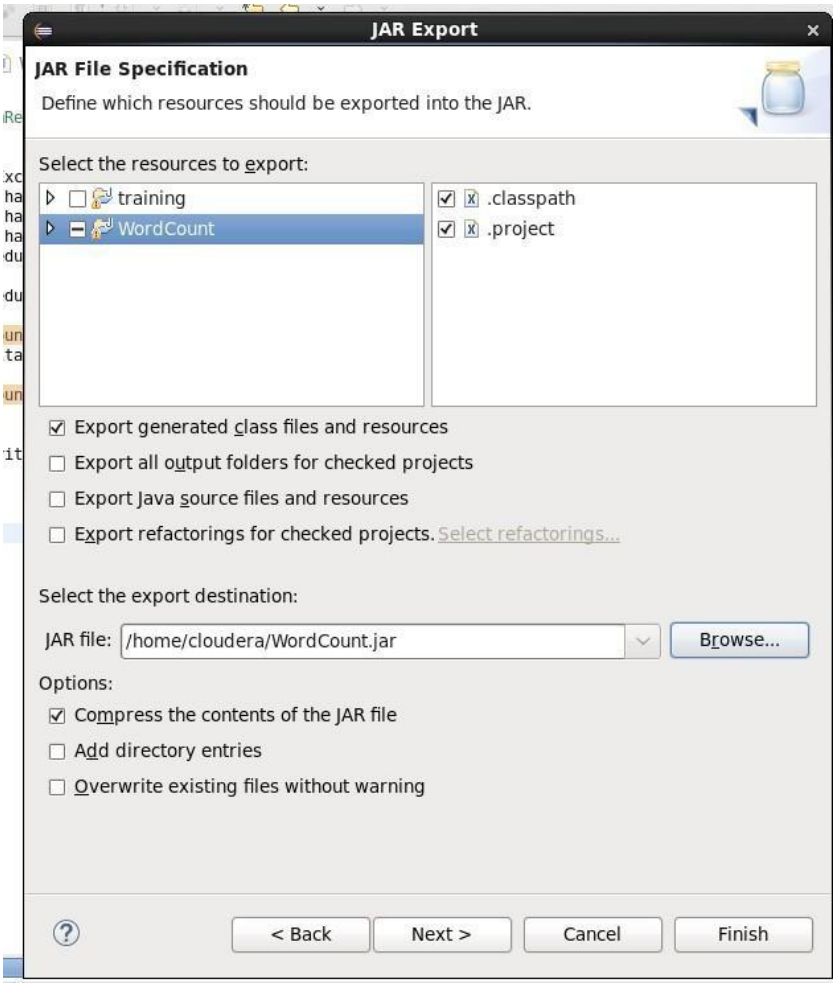
Verify the jar file

Verify the jar file through command line open terminal give command
ls

Step 7: Move the jar file to the Hadoop file

system **hdfs dfs -put wordcount.jar /user/cloudera hdfs dfs -ls**


JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
myMapZaheen
.classpath
.project
Name:
WordCount.jar
Save in folder:
cloudera
Browse for other folders
Cancel
OK
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Cancel
Finish


JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
training
WordCount
.classpath
.project
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file: /home/cloudera/WordCount.jar
Browse...
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Cancel
Finish

Step 8: Create the input file for the MapReduce

program Command: **cat > myInputFile.txt**
Welcome to the NMITD
MCA I am persuing MCA
in NMITD
Enter data in input file and press enter and ctrl z
Command: **cat myInputFile.txt**

Step 9: Move the input file to the Hadoop file system

hdfs dfs -put myInputFile.txt
/user/cloudera hdfs dfs -ls
hdfs dfs –ls / (here / indicates root directory of Hadoop file system(hdfs)) step 10: Run mapreduce program on Hadoop

syntax: hadoop jar jarfilename.jar classname inputfilename.txt outputfoldername
command: **hadoop jar wordcount.jar WordCountDriver myInputFile.txt myOutput**

Start Yarn if not started and showing connection refused

```
[cloudera@quickstart ~]$ hadoop jar Wordcount.jar newjava.WordCountDriver myInpu
tFile.txt myOutput
25/08/06 22:00:15 INFO client.RMProxy: Connecting to ResourceManager at quicksta
rt.cloudera/127.0.0.1:8032
25/08/06 22:00:15 WARN mapreduce.JobResourceUploader: Hadoop command-line option
 parsing not performed. Implement the Tool interface and execute your applicatio
n with ToolRunner to remedy this.
25/08/06 22:00:16 INFO input.FileInputFormat: Total input paths to process : 1
25/08/06 22:00:16 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1281)
        at java.lang.Thread.join(Thread.java:1355)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DF
SOutputStream.java:967)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutpu
tStream.java:705)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStre
am.java:894)
25/08/06 22:00:16 INFO mapreduce.JobSubmitter: number of splits:1
25/08/06 22:00:16 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_17
54542791586_0001
25/08/06 22:00:17 INFO impl.YarnClientImpl: Submitted application application_17
54542791586_0001
25/08/06 22:00:17 INFO mapreduce.Job: The url to track the job: http://quickstar
t.cloudera:8088/proxy/application_1754542791586_0001/
25/08/06 22:00:17 INFO mapreduce.Job: Running job: job_1754542791586_0001
25/08/06 22:00:27 INFO mapreduce.Job: Job job_1754542791586_0001 running in uber
 mode : false
25/08/06 22:00:27 INFO mapreduce.Job:  map 0% reduce 0%
25/08/06 22:00:42 INFO mapreduce.Job:  map 100% reduce 0%
25/08/06 22:00:56 INFO mapreduce.Job:  map 100% reduce 100%
25/08/06 22:00:57 INFO mapreduce.Job: Job job_1754542791586_0001 completed succe
ssfully
25/08/06 22:00:57 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=144
                FILE: Number of bytes written=295053
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=232
                HDFS: Number of bytes written=74
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
```

```
        File System Counters
                FILE: Number of bytes read=144
                FILE: Number of bytes written=295053
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=232
                HDFS: Number of bytes written=74
                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)=6788608
                Total time spent by all reduces in occupied slots (ms)=5730304
                Total time spent by all map tasks (ms)=13259
                Total time spent by all reduce tasks (ms)=11192
                Total vcore-milliseconds taken by all map tasks=13259
                Total vcore-milliseconds taken by all reduce tasks=11192
                Total megabyte-milliseconds taken by all map tasks=6788608
                Total megabyte-milliseconds taken by all reduce tasks=5730304
        Map-Reduce Framework
                Map input records=4
                Map output records=21
                Map output bytes=186
                Map output materialized bytes=140
                Input split bytes=126
                Combine input records=0
                Combine output records=0
                Reduce input groups=10
                Reduce shuffle bytes=140
                Reduce input records=21
                Reduce output records=10
                Spilled Records=42
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=795
                CPU time spent (ms)=1520
                Physical memory (bytes) snapshot=240349184
                Virtual memory (bytes) snapshot=1408757760
                Total committed heap usage (bytes)=101449728
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
```

```
        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=106
        File Output Format Counters
                Bytes Written=74
[cloudera@quickstart ~]$ hdfs dfs -ls /user/cloudera/myOutput
Found 2 items
-rw-r--r--   1 cloudera cloudera          0 2025-08-06 22:00 /user/cloudera/myOu
tput/_SUCCESS
-rw-r--r--   1 cloudera cloudera         74 2025-08-06 22:00 /user/cloudera/myOu
tput/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat /user/cloudera/myOutput/part-r-00000

I       2
MCA     4
MITD    3
MITWelcome      1
Welcome 1
am      2
in      2
persuing        2
the     2
to      2
[cloudera@quickstart ~]$
```

Step 11: view output directory
hdfs dfs –ls
hdfs dfs -ls /user/cloudera/myOutput

step 12: view the output file
hdfs dfs -cat /user/cloudera/myOutput/part-r-00000

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));
            3
              3
        3
3
```

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)
                                                                                throw
s IOException, InterruptedException {
                int wordCount = 0;
                for (IntWritable value : values) {
                        wordCount += value.get();
                3
                context.write(key, new IntWritable(wordCount));
        3
3
```

WordCountDriver.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 WordCountDirver {
   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);
                3
                Job job = new Job();
                job.setJarByClass(WordCountDriver.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);
```

```
        3
3
```

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

Code:

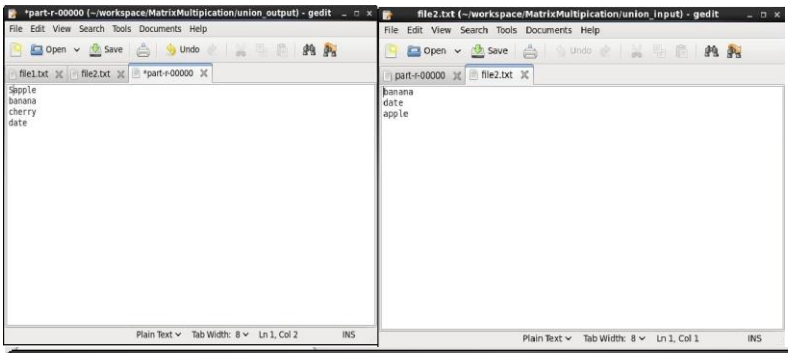
```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import
org.apache.hadoop.mapreduce.Job;
import
org.apache.hadoop.mapreduce.Mapper;
import
org.apache.hadoop.mapreduce.Reducer;
import
org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import java.io.IOException;
public class Union {
    // Mapper: Just emits each line as key
  public static class UnionMapper extends Mapper<Object, Text, Text,
    Text> { private static final Text empty = new Text("");
    public void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {
      context.write(value, empty);
    3
  3
// Reducer: Writes out each unique key once (union)
```

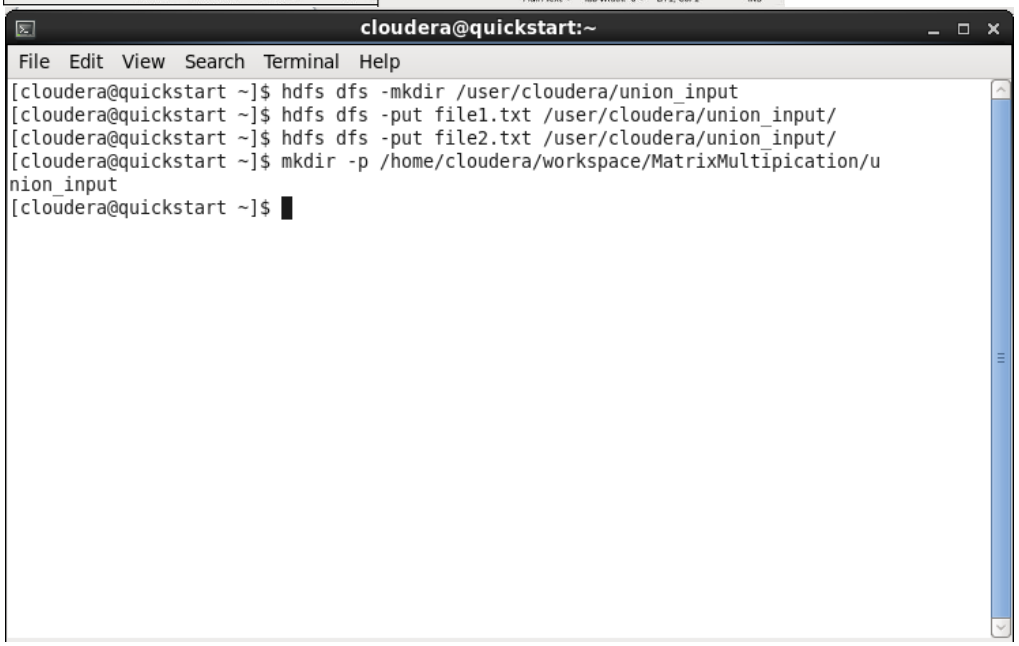
```
public static class UnionReducer extends Reducer<Text, Text, Text,
Text> { public void reduce(Text key, Iterable<Text> values, Context
context)
        throws IOException, InterruptedException {
    context.write(key, null);
3
3
public static void main(String[] args) throws Exception {
// Check for input arguments
if (args.length < 2) {
    System.err.println("Usage: Union <input path> <output
    path>"); System.exit(-1);
3
Configuration conf = new Configuration();
Job job = Job.getInstance(conf, "Union Operation");
job.setJarByClass(Union.class);
job.setMapperClass(UnionMapper.class);
job.setReducerClass(UnionReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
3
3
```

Steps to execute:

Step 1: Create two sample files in cloudera.

Step 2: Prepare Local Input File. In your Cloudera terminal

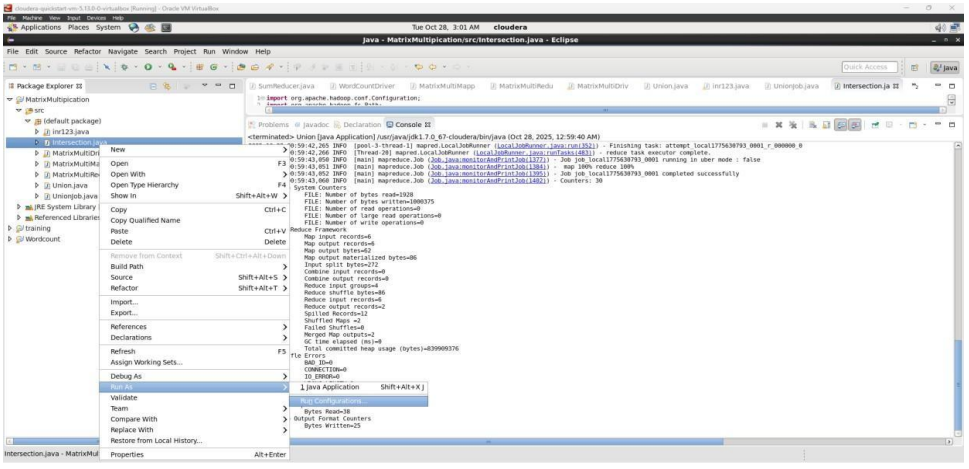




Step 3: Run Configuration in Eclipse

Right-click on Intersection.java → Run As → Run

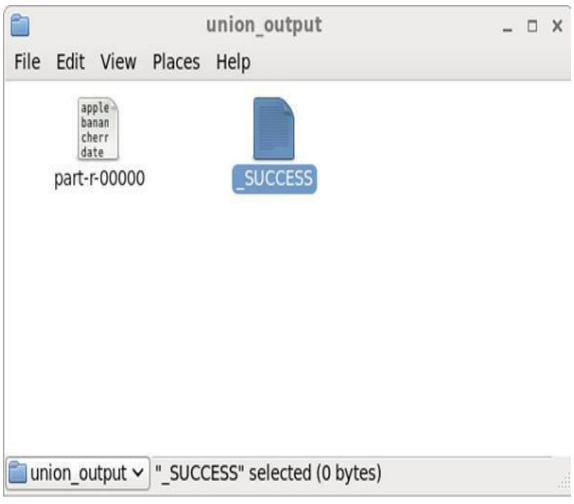
Configurations… In Arguments → Program arguments, enter:

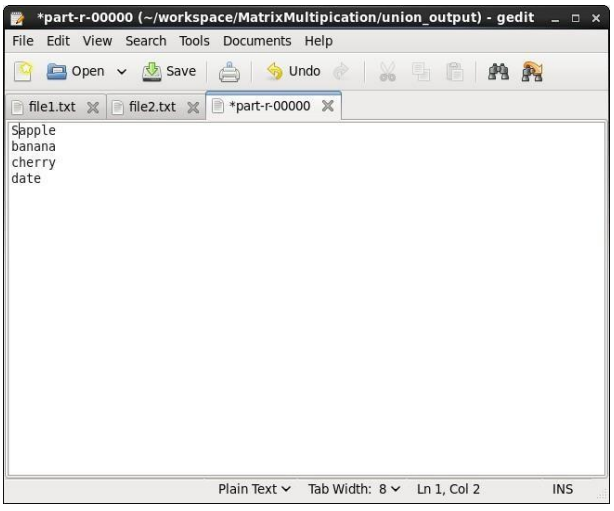


Output:

```
Problems  Javadoc  Declaration  Console
<terminated> Union [Java Application] /usr/java/jdk1.7.0_67-cloudera/bin/java (Oct 28, 2025, 12:44:00 AM)
2025-10-28 00:44:01,614 WARN  [main] util.NativeCodeLoader (NativeCodeLoader.java:<clinit>(62)) - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2025-10-28 00:44:01,964 INFO  [main] Configuration.deprecation (Configuration.java:warnOnceIfDeprecated(1204)) - session.id is deprecated. Instead, use dfs.metrics.session-id
2025-10-28 00:44:01,965 INFO  [main] jvm.JvmMetrics (JvmMetrics.java:init(76)) - Initializing JVM Metrics with processName=JobTracker, sessionId=
2025-10-28 00:44:02,093 WARN  [main] mapreduce.JobResourceUploader (JobResourceUploader.java:uploadFiles(64)) - Hadoop command-line option parsing not performed. Implement the Tool interface and execut
2025-10-28 00:44:02,098 WARN  [main] mapreduce.JobResourceUploader (JobResourceUploader.java:uploadFiles(171)) - No job jar file set.  User classes may not be found. See Job or Job#setJar(String).
2025-10-28 00:44:02,109 INFO  [main] input.FileInputFormat (FileInputFormat.java:listStatus(283)) - Total input paths to process : 2
2025-10-28 00:44:02,154 INFO  [main] mapreduce.JobSubmitter (JobSubmitter.java:submitJobInternal(202)) - number of splits:2
2025-10-28 00:44:02,282 INFO  [main] mapreduce.JobSubmitter (JobSubmitter.java:printTokens(291)) - Submitting tokens for job: job_local1773454917_0001
2025-10-28 00:44:02,435 INFO  [main] mapreduce.Job (Job.java:submit(1311)) - The url to track the job: http://localhost:8080/
2025-10-28 00:44:02,436 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1356)) - Running job: job_local1773454917_0001
2025-10-28 00:44:02,436 INFO  [Thread-20] mapred.LocalJobRunner (LocalJobRunner.java:createOutputCommitter(498)) - OutputCommitter set in config null
2025-10-28 00:44:02,439 INFO  [Thread-20] output.FileOutputCommitter (FileOutputCommitter.java:<init>(124)) - File Output Committer Algorithm version is 1
2025-10-28 00:44:02,439 INFO  [Thread-20] output.FileOutputCommitter (FileOutputCommitter.java:<init>(139)) - FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cl
2025-10-28 00:44:02,440 INFO  [Thread-20] mapred.LocalJobRunner (LocalJobRunner.java:createOutputCommitter(516)) - OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2025-10-28 00:44:02,466 INFO  [Thread-20] mapred.LocalJobRunner (LocalJobRunner.java:runTasks(475)) - Waiting for map tasks
2025-10-28 00:44:02,466 INFO  [LocalJobRunner Map Task Executor #0] mapred.LocalJobRunner (LocalJobRunner.java:run(251)) - Starting task: attempt_local1773454917_0001_m_000000_0
2025-10-28 00:44:02,484 INFO  [LocalJobRunner Map Task Executor #0] output.FileOutputCommitter (FileOutputCommitter.java:<init>(124)) - File Output Committer Algorithm version is 1
2025-10-28 00:44:02,484 INFO  [LocalJobRunner Map Task Executor #0] output.FileOutputCommitter (FileOutputCommitter.java:<init>(139)) - FileOutputCommitter skip cleanup _temporary folders under output
2025-10-28 00:44:02,494 INFO  [LocalJobRunner Map Task Executor #0] mapred.Task (Task.java:initialize(631)) - Using ResourceCalculatorProcessTree : [ ]
2025-10-28 00:44:02,499 INFO  [LocalJobRunner Map Task Executor #0] mapred.MapTask (MapTask.java:runNewMapper(762)) - Processing split: file:/home/cloudera/workspace/MatrixMultipication/union_input/fil
2025-10-28 00:44:02,705 INFO  [LocalJobRunner Map Task Executor #0] mapred.MapTask (MapTask.java:setEquator(1212)) - (EQUATOR) 0 kvi 26214396(104857584)
2025-10-28 00:44:03,441 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1384)) -  map 100% reduce 100%
2025-10-28 00:44:03,442 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1395)) - Job job_local1773454917_0001 completed successfully
2025-10-28 00:44:03,450 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1402)) - Counters: 30
        File System Counters
                FILE: Number of bytes read=1092
                FILE: Number of bytes written=1000096
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
        Map-Reduce Framework
                Map input records=6
                Map output records=6
                Map output bytes=44
                Map output materialized bytes=68
                Input split bytes=272
                Combine input records=0
                Combine output records=0
                Reduce input groups=4
                Reduce shuffle bytes=68
                Reduce input records=6
                Reduce output records=4
                Spilled Records=12
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=0
```

A new output file is created which have all the elements of both the files.





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

Code:

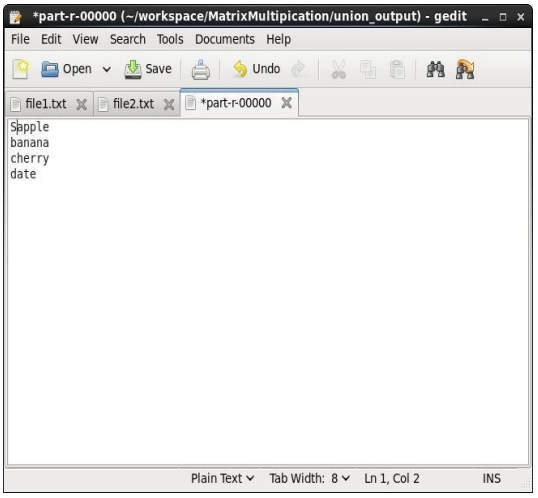
```
Import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
importorg.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import java.io.IOException;

public class Intersection {

  // Mapper: emit each line as a key,
  value = 1 public static class
  IntersectionMapper
       extends Mapper<Object, Text, Text, IntWritable> {

    private final static IntWritable one = new IntWritable(1);

    public void map(Object key, Text value, Context
         context) throws IOException, InterruptedException
         {
      context.write(value, one);
    3
  3
```

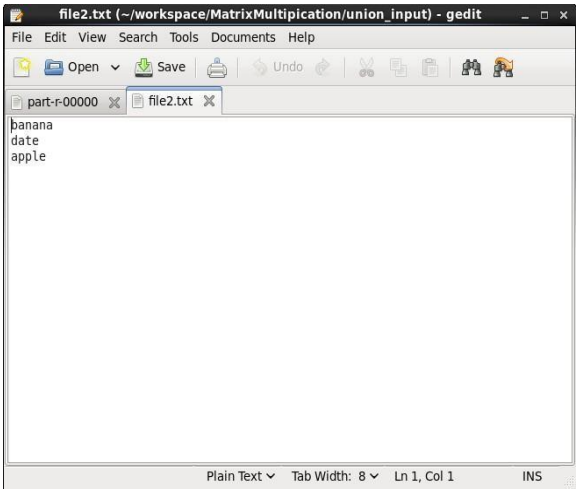
```
// Reducer: count how many times each key (line) appears

public static class IntersectionReducer
        extends Reducer<Text, IntWritable, Text, Text> {

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

        int count = 0;
        for (IntWritable val : values) {
            count += val.get();
        3

        // Assuming there are 2 files → intersection = lines appearing at least twice
        if (count >= 2) {
            context.write(key, null);
        3
    3
3

public static void main(String[] args) throws Exception {
    if (args.length < 2) {
        System.err.println("Usage: Intersection <input path> <output
        path>"); System.exit(-1);
    3

    Configuration conf = new Configuration();

    // Run in local mode for testing (optional)
    conf.set("fs.defaultFS", "file:///");
```

```
        conf.set("mapreduce.framework.name",
        "local");
        Job job = Job.getInstance(conf, "Intersection Operation");
        job.setJarByClass(Intersection.class);
        job.setMapperClass(IntersectionMapper.class);
        job.setReducerClass(IntersectionReducer.class)
        ;
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

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

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    3
3
```

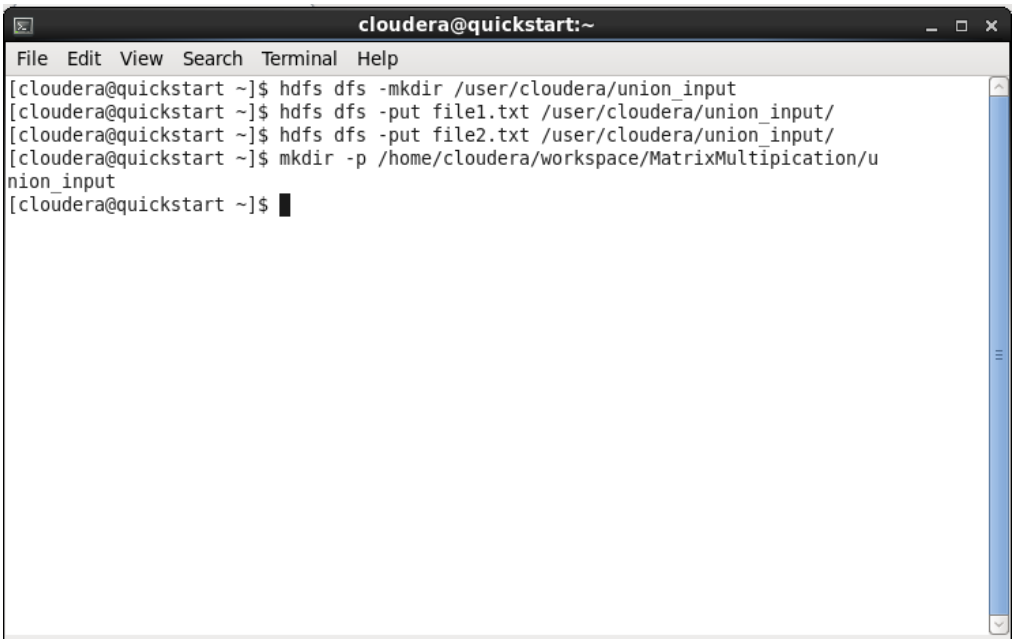
Steps for further implementation:

Step 1: Create two sample files in cloudera.





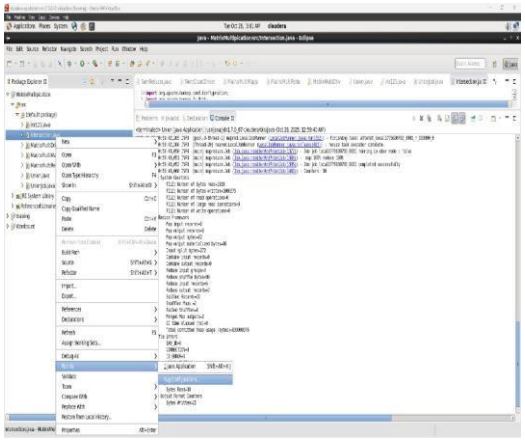
Step 2: Prepare Local Input File. In your Cloudera terminal

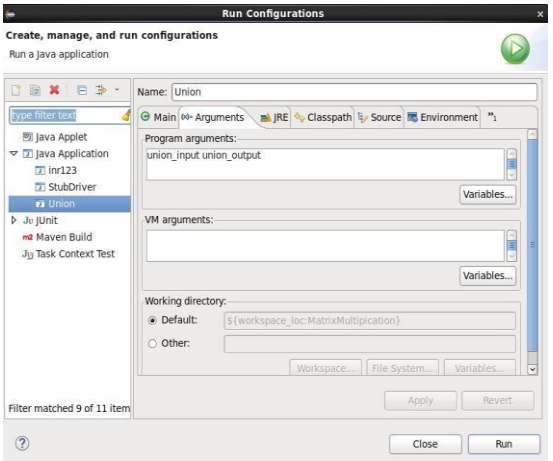


Step 3: Run Configuration in Eclipse

Right-click on Intersection.java → Run As → Run

Configurations… In Arguments → Program arguments, enter:

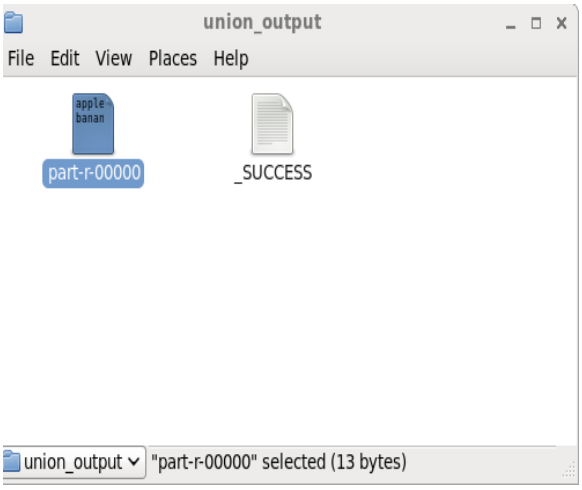


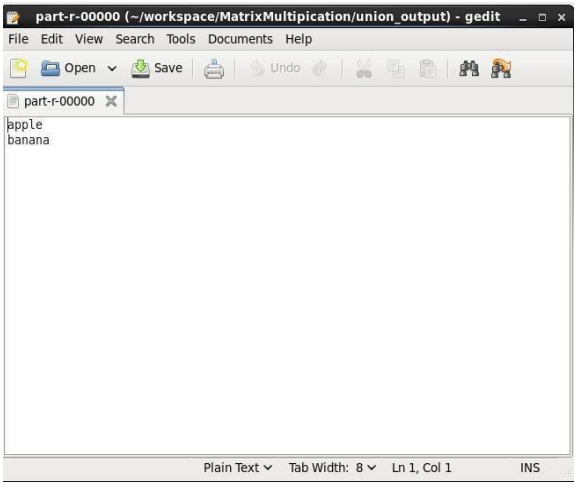


Output:

```
Problems  Javadoc  Declaration  Console
<terminated> Union [Java Application] /usr/java/jdk1.7.0_67-cloudera/bin/java (Oct 28, 2025, 12:44:00 AM)
2025-10-28 00:44:01,614 WARN  [main] util.NativeCodeLoader (NativeCodeLoader.java:<clinit>(62)) - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2025-10-28 00:44:01,964 INFO  [main] Configuration.deprecation (Configuration.java:warnOnceIfDeprecated(1204)) - session.id is deprecated. Instead, use dfs.metrics.session-id
2025-10-28 00:44:01,965 INFO  [main] jvm.JvmMetrics (JvmMetrics.java:init(76)) - Initializing JVM Metrics with processName=JobTracker, sessionId=
2025-10-28 00:44:02,093 WARN  [main] mapreduce.JobResourceUploader (JobResourceUploader.java:uploadFiles(64)) - Hadoop command-line option parsing not performed. Implement the Tool interface and execut
2025-10-28 00:44:02,098 WARN  [main] mapreduce.JobResourceUploader (JobResourceUploader.java:uploadFiles(171)) - No job jar file set.  User classes may not be found. See Job or Job#setJar(String).
2025-10-28 00:44:02,109 INFO  [main] input.FileInputFormat (FileInputFormat.java:listStatus(283)) - Total input paths to process : 2
2025-10-28 00:44:02,154 INFO  [main] mapreduce.JobSubmitter (JobSubmitter.java:submitJobInternal(202)) - number of splits:2
2025-10-28 00:44:02,282 INFO  [main] mapreduce.JobSubmitter (JobSubmitter.java:printTokens(291)) - Submitting tokens for job: job_local1773454917_0001
2025-10-28 00:44:02,435 INFO  [main] mapreduce.Job (Job.java:submit(1311)) - The url to track the job: http://localhost:8080/
2025-10-28 00:44:02,436 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1356)) - Running job: job_local1773454917_0001
2025-10-28 00:44:02,436 INFO  [Thread-20] mapred.LocalJobRunner (LocalJobRunner.java:createOutputCommitter(498)) - OutputCommitter set in config null
2025-10-28 00:44:02,439 INFO  [Thread-20] output.FileOutputCommitter (FileOutputCommitter.java:<init>(124)) - File Output Committer Algorithm version is 1
2025-10-28 00:44:02,439 INFO  [Thread-20] output.FileOutputCommitter (FileOutputCommitter.java:<init>(139)) - FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cl
2025-10-28 00:44:02,440 INFO  [Thread-20] mapred.LocalJobRunner (LocalJobRunner.java:createOutputCommitter(516)) - OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2025-10-28 00:44:02,466 INFO  [Thread-20] mapred.LocalJobRunner (LocalJobRunner.java:runTasks(475)) - Waiting for map tasks
2025-10-28 00:44:02,466 INFO  [LocalJobRunner Map Task Executor #0] mapred.LocalJobRunner (LocalJobRunner.java:run(251)) - Starting task: attempt_local1773454917_0001_m_000000_0
2025-10-28 00:44:02,484 INFO  [LocalJobRunner Map Task Executor #0] output.FileOutputCommitter (FileOutputCommitter.java:<init>(124)) - File Output Committer Algorithm version is 1
2025-10-28 00:44:02,484 INFO  [LocalJobRunner Map Task Executor #0] output.FileOutputCommitter (FileOutputCommitter.java:<init>(139)) - FileOutputCommitter skip cleanup _temporary folders under output
2025-10-28 00:44:02,494 INFO  [LocalJobRunner Map Task Executor #0] mapred.Task (Task.java:initialize(631)) -  Using ResourceCalculatorProcessTree : [ ]
2025-10-28 00:44:02,499 INFO  [LocalJobRunner Map Task Executor #0] mapred.MapTask (MapTask.java:runNewMapper(762)) - Processing split: file:/home/cloudera/workspace/MatrixMultipication/union_input/fil
2025-10-28 00:44:02,705 INFO  [LocalJobRunner Map Task Executor #0] mapred.MapTask (MapTask.java:setEquator(1212)) - (EQUATOR) 0 kvi 26214396(104857584)
2025-10-28 00:44:03,441 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1384)) -  map 100% reduce 100%
2025-10-28 00:44:03,442 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1395)) - Job job_local1773454917_0001 completed successfully
2025-10-28 00:44:03,450 INFO  [main] mapreduce.Job (Job.java:monitorAndPrintJob(1402)) - Counters: 30
        File System Counters
                FILE: Number of bytes read=1892
                FILE: Number of bytes written=1000096
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
        Map-Reduce Framework
                Map input records=6
                Map output records=6
                Map output bytes=44
                Map output materialized bytes=68
                Input split bytes=272
                Combine input records=0
                Combine output records=0
                Reduce input groups=4
                Reduce shuffle bytes=68
                Reduce input records=6
                Reduce output records=4
                Spilled Records=12
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=0
```

A new output file is created which have all the common elements of the files.





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

Steps:

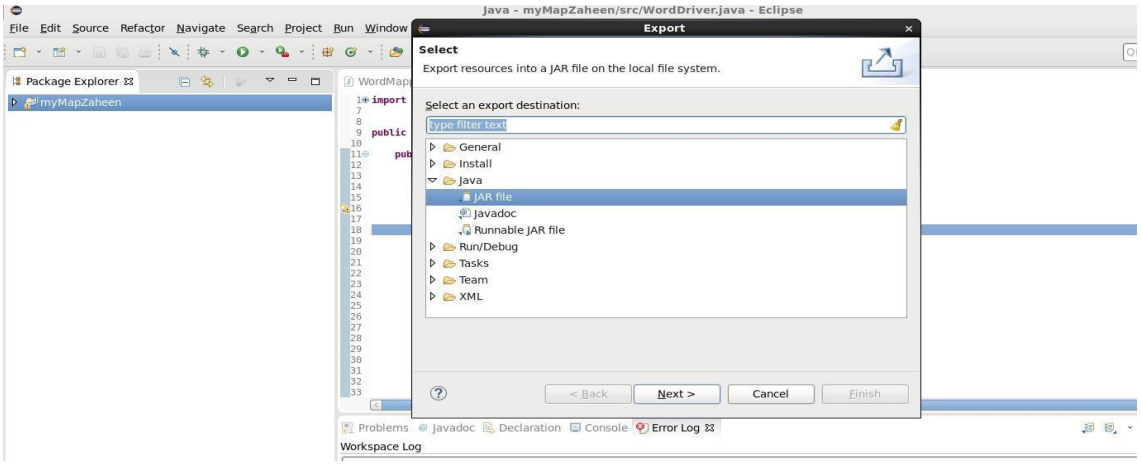
Step 1: Open virtual box and then start cloudera quickstart

Step 2: Open eclipse present on the cloudera desktop

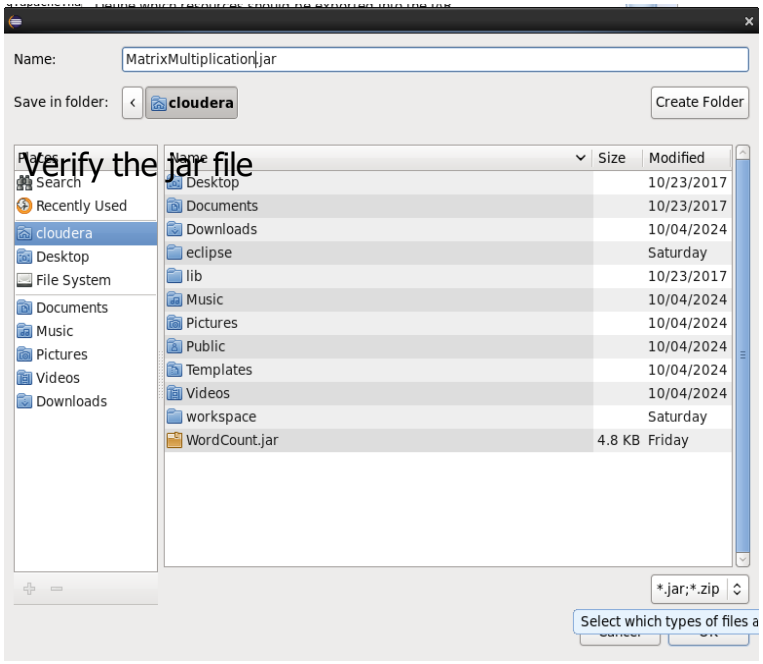
Step 3: Create java project
File->New-> Java Project
Give project name: MatrixMultiplication
Click Next

Step 4: Add Hadoop libraries to project
Select Libraries tab->click on Add Externa
Jars File System->user->lib->Hadoop
Select all library(jar) files->ok
Again, click on Add External
Jars
File System->user->lib->Hadoop->client
Select all library(jar) files from client->ok->finish

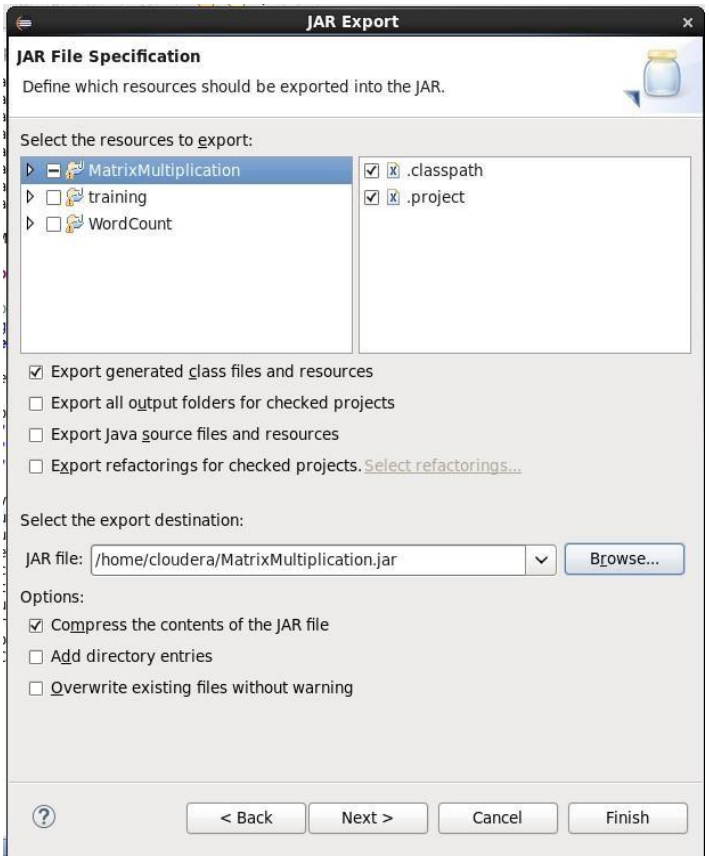
Step 5: Write java code for matrix multiplication using mapreduce Right click on src folder of project MatrixMultiplication New->class
Write class name MatrixMultiplicationMapper
Click Finish
Write the code for MatrixMultiplicationMapper
Same way create classes MatrixMultiplicationReducer and MatrixMultiplicationDriver Step 6: Export the project as jar
Right click on project MatrixMultiplication and select Export>> Java>>JAR file>>Next



Select the export destination-Click browse-give file name matrixmultiplication.jar Click ok>>Finish>>ok



Verify the jar file



Verify the jar file through command line open terminal give command
ls
step 7: Move the jar file to the Hadoop file system
hdfs dfs -put matrixmultiplication.jar /user/cloudera
hdfs dfs -ls

Step 8: Create the input file for the MapReduce program First input file
Command: **cat > myMMatrix.txt**
Enter data in input file
M,0,0,1
M,0,1,2
M,1,0,3
M,1,1,4
and press enter and ctrl z

Command: **cat myMMatrix.txt**
Second input file
Command: **cat > myNMatrix.txt**
Enter data in input file
N,0,0,3
N,0,1,6
N,1,0,4
N,1,1,2
and press enter and ctrl z
Command: **cat myNMatrix.txt**

Step 9: Create the input directory and move input files into it.

hdfs dfs –mkdir /user/cloudera/matrixInput
hdfs dfs -put myMMatrix.txt /user/cloudera/matrixInput/ hdfs dfs -put myNMatrix.txt /user/cloudera/matrixInput/ hdfs dfs -ls

step 10: Run mapreduce program on Hadoop
syntax: hadoop jar jarfilename.jar classname inputfoldername outputfoldername
command: **hadoop jar matrixmultiplication.jar MatrixMultiplicationDriver matrixInput matrixOutput**

step 11: view output directory
hdfs dfs –ls
hdfs dfs -ls /user/cloudera/matrixOutput
step 12: view the output file
hdfs dfs -cat /user/cloudera/matrixOutput/part-r-00000

Code:

MatrixMultiplicationMapper.java

```
import java.io.IOException;

import

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

public class MatrixMultiplicationMapper 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);
                3
       3 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);
```

```
                    3
                3
                3
3
```

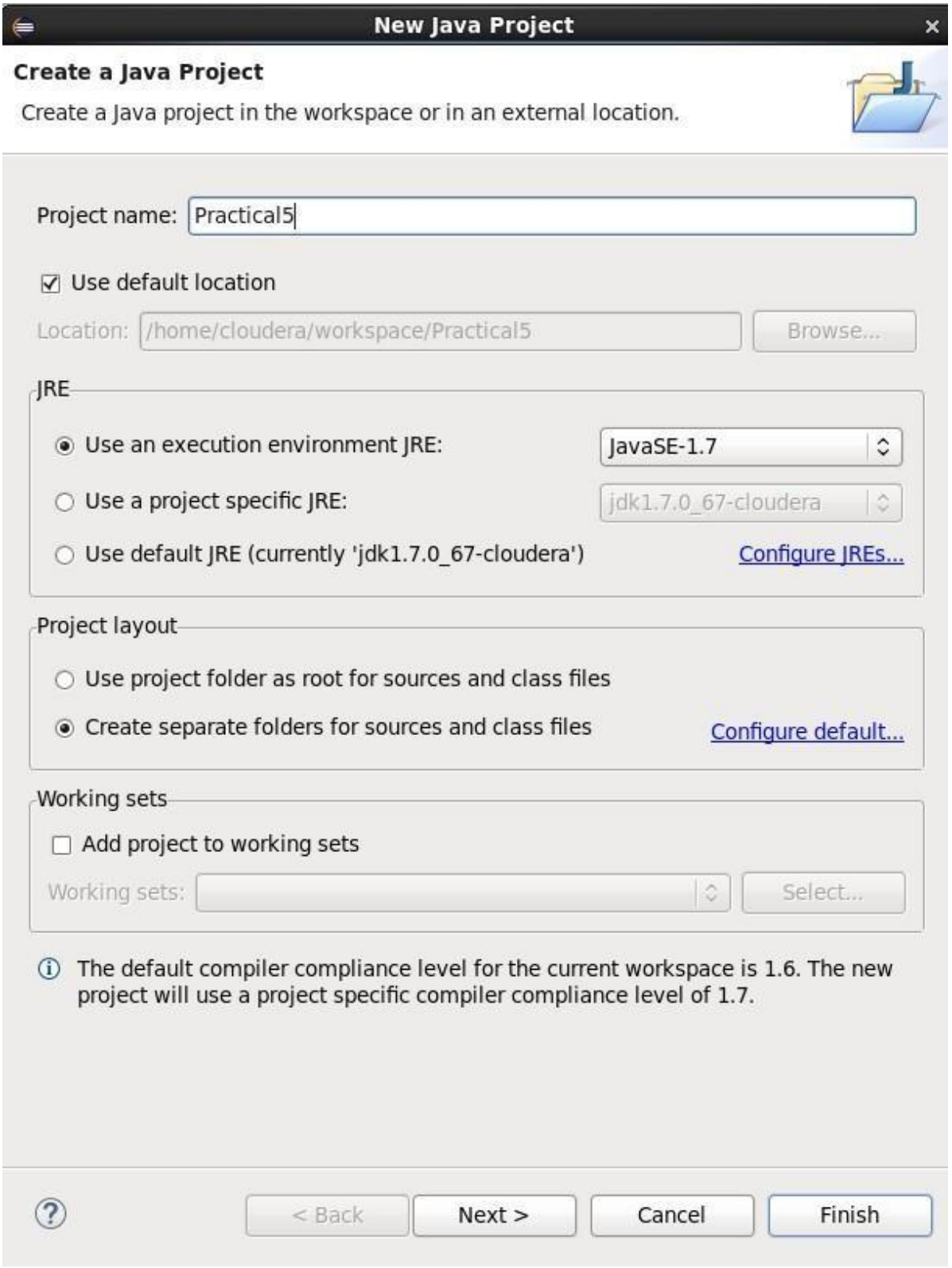
MatrixMultiplicationReducer.java

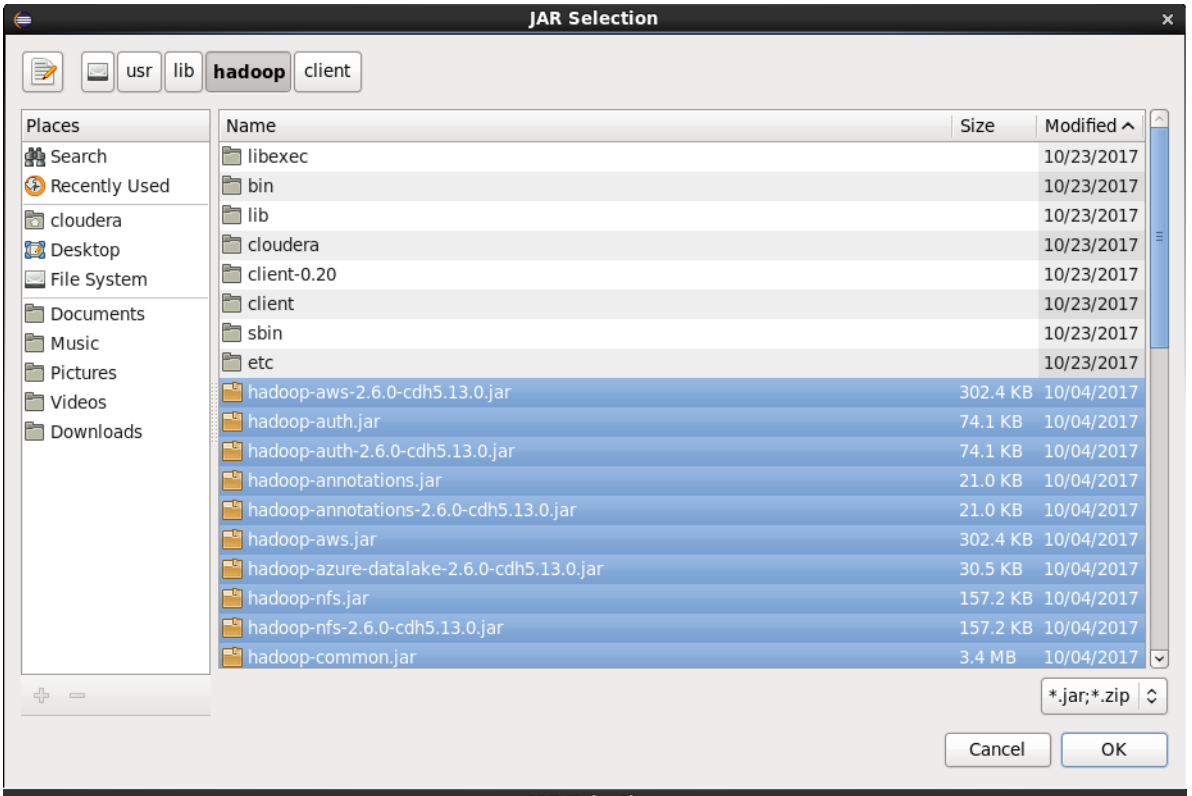
```
import org.apache.hadoop.io.Text;
import java.io.IOException;
import java.util.HashMap;

public class MatrixMultiplicationReducer 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]));
                3 else {
hashB.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
                3
        3
        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;
        3
        if (result != 0.0f) {
                context.write(null,
new Text(key.toString() + "," + Float.toString(result)));
        3
        3
3
```

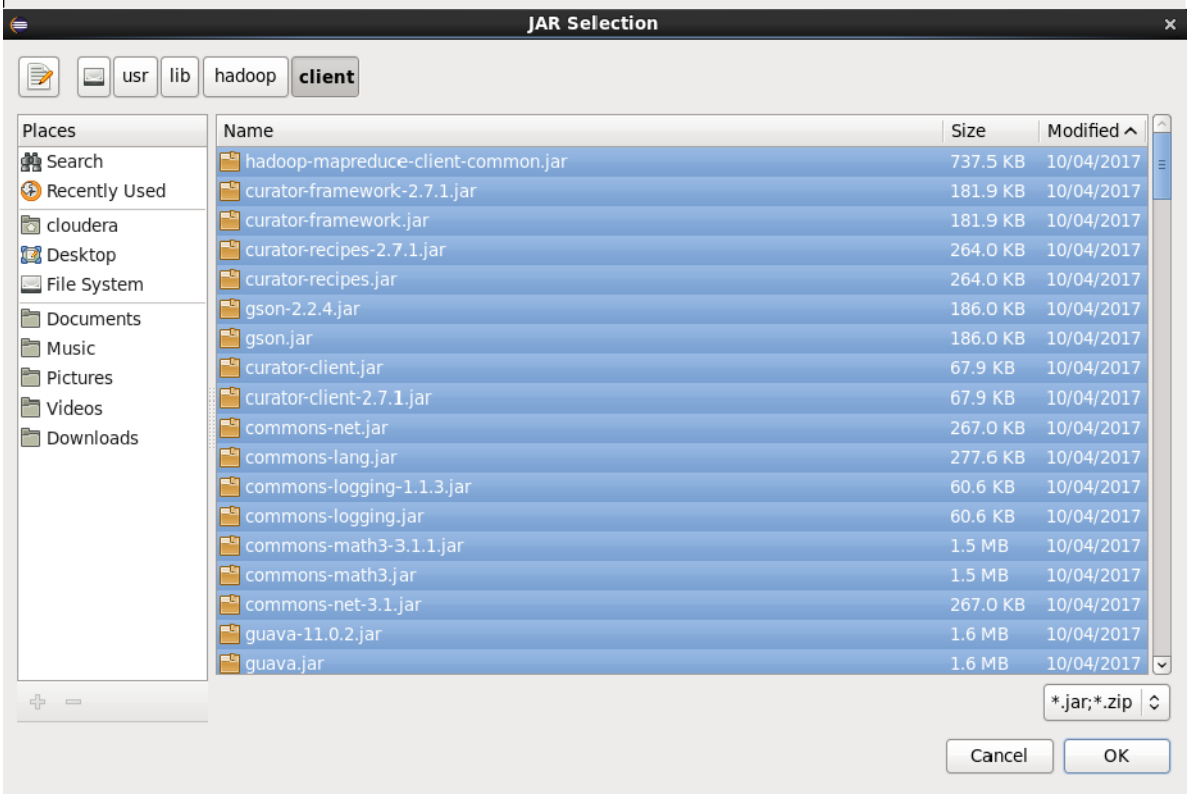
MatrixMultiplicationDriver.java

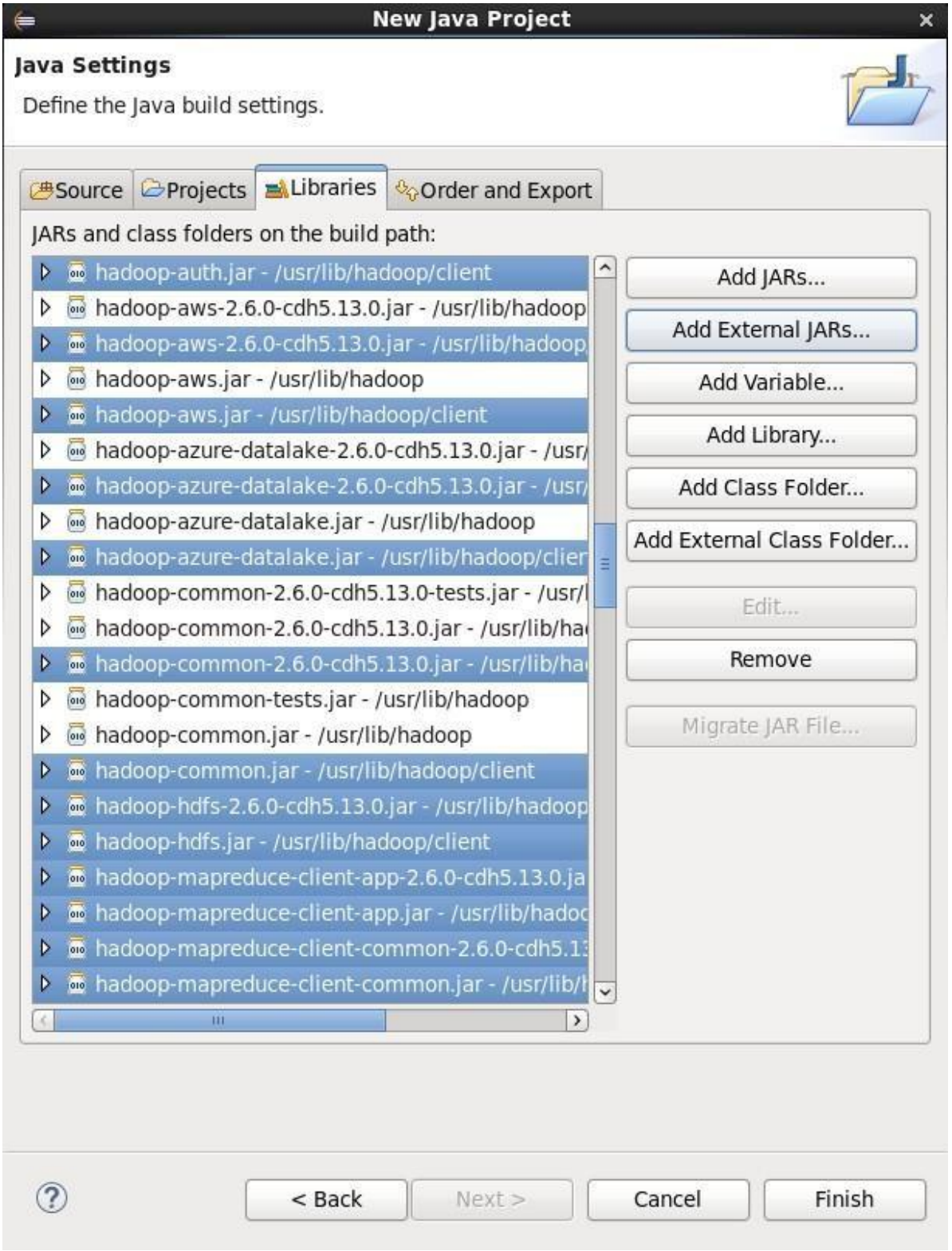
```
import org.apache.hadoop.conf.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class MatrixMultiplicationDriver {

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

// TODO Auto-generated method stub
                        if (args.length != 2) {
                        System.err.println("Usage:
                        MatrixMultiply <in_dir>
                        <out_dir>"); System.exit(2);
                        3
                       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(MatrixMultiplicationDriver.class);
                        job.setOutputKeyClass(Text.class);
                        job.setOutputValueClass(Text.class);

                        job.setMapperClass(MatrixMultiplicationMapper.class);
                        job.setReducerClass(MatrixMultiplicationReducer.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);
                3

3
```

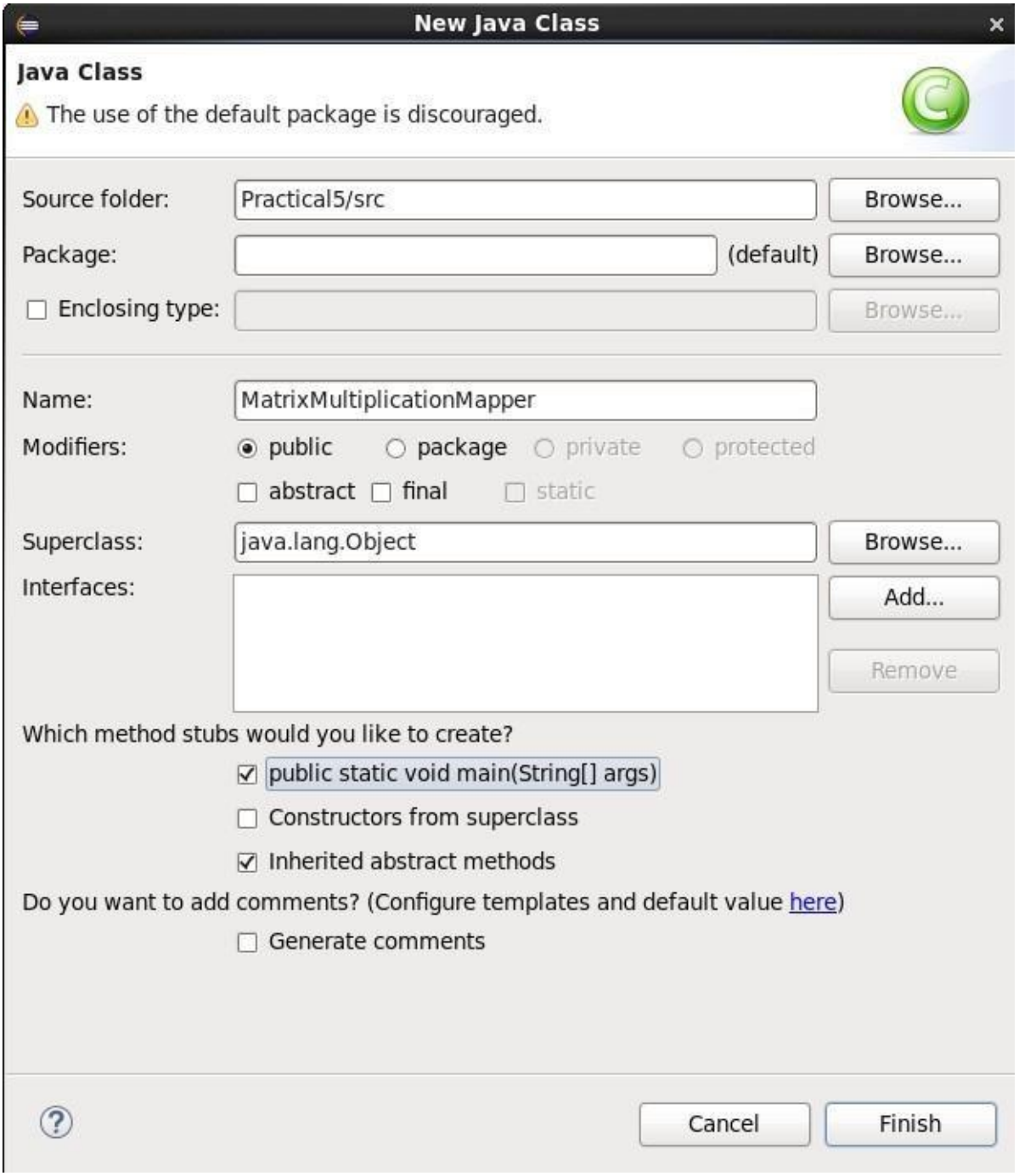
Output:

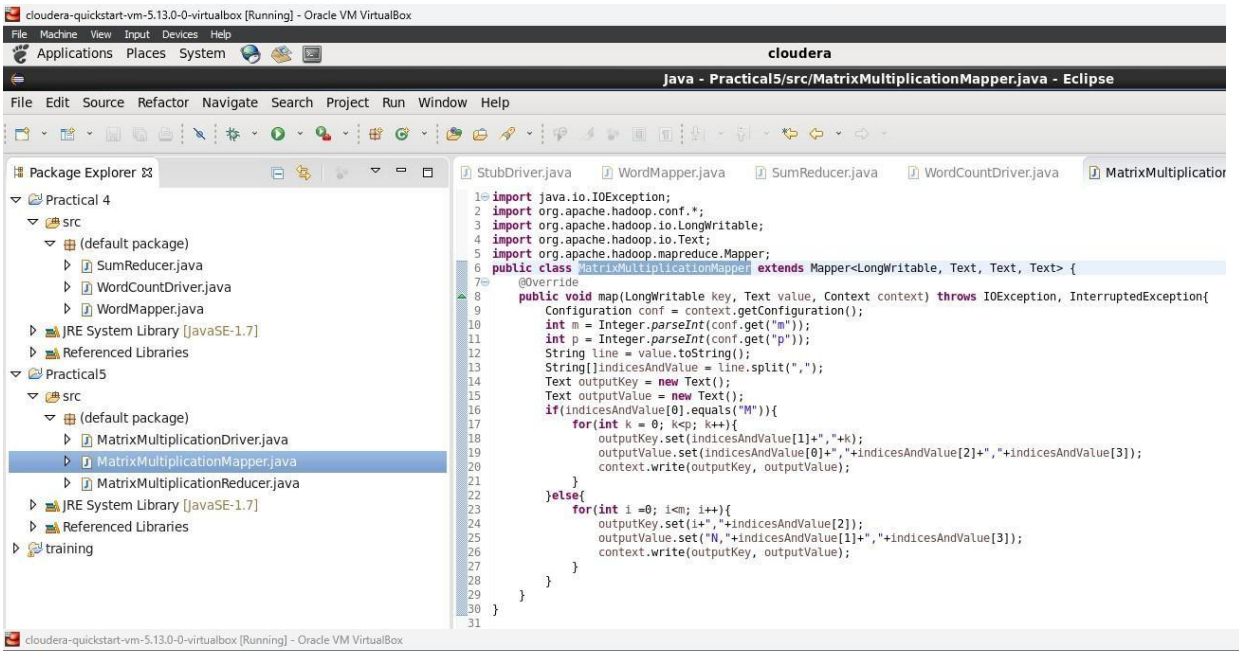

New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: Practical5
Use default location
Location: /home/cloudera/workspace/Practical5
Browse...
JRE
Use an execution environment JRE:
JavaSE-1.7
Use a project specific JRE:
jdk1.7.0_67-cloudera
Use default JRE (currently 'jdk1.7.0_67-cloudera')
Configure JREs...
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
Working sets
Add project to working sets
Working sets:
Select...
The default compiler compliance level for the current workspace is 1.6. The new project will use a project specific compiler compliance level of 1.7.
< Back
Next >
Cancel
Finish

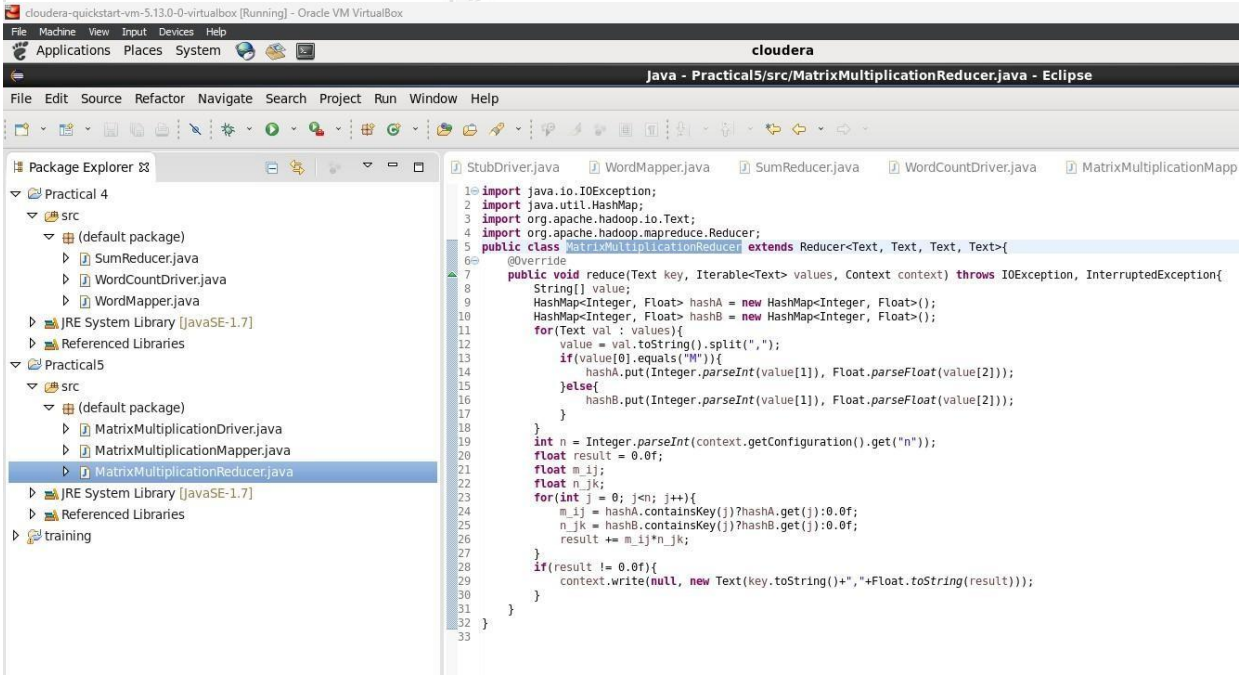

JAR Selection
usr
lib
hadoop
client
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
libexec 10/23/2017
bin 10/23/2017
lib 10/23/2017
cloudera 10/23/2017
client-0.20 10/23/2017
client 10/23/2017
sbin 10/23/2017
etc 10/23/2017
hadoop-aws-2.6.0-cdh5.13.0.jar 302.4 KB 10/04/2017
hadoop-auth.jar 74.1 KB 10/04/2017
hadoop-auth-2.6.0-cdh5.13.0.jar 74.1 KB 10/04/2017
hadoop-annotations.jar 21.0 KB 10/04/2017
hadoop-annotations-2.6.0-cdh5.13.0.jar 21.0 KB 10/04/2017
hadoop-aws.jar 302.4 KB 10/04/2017
hadoop-azure-datalake-2.6.0-cdh5.13.0.jar 30.5 KB 10/04/2017
hadoop-nfs.jar 157.2 KB 10/04/2017
hadoop-nfs-2.6.0-cdh5.13.0.jar 157.2 KB 10/04/2017
hadoop-common.jar 3.4 MB 10/04/2017
.jar;.zip
Cancel
OK

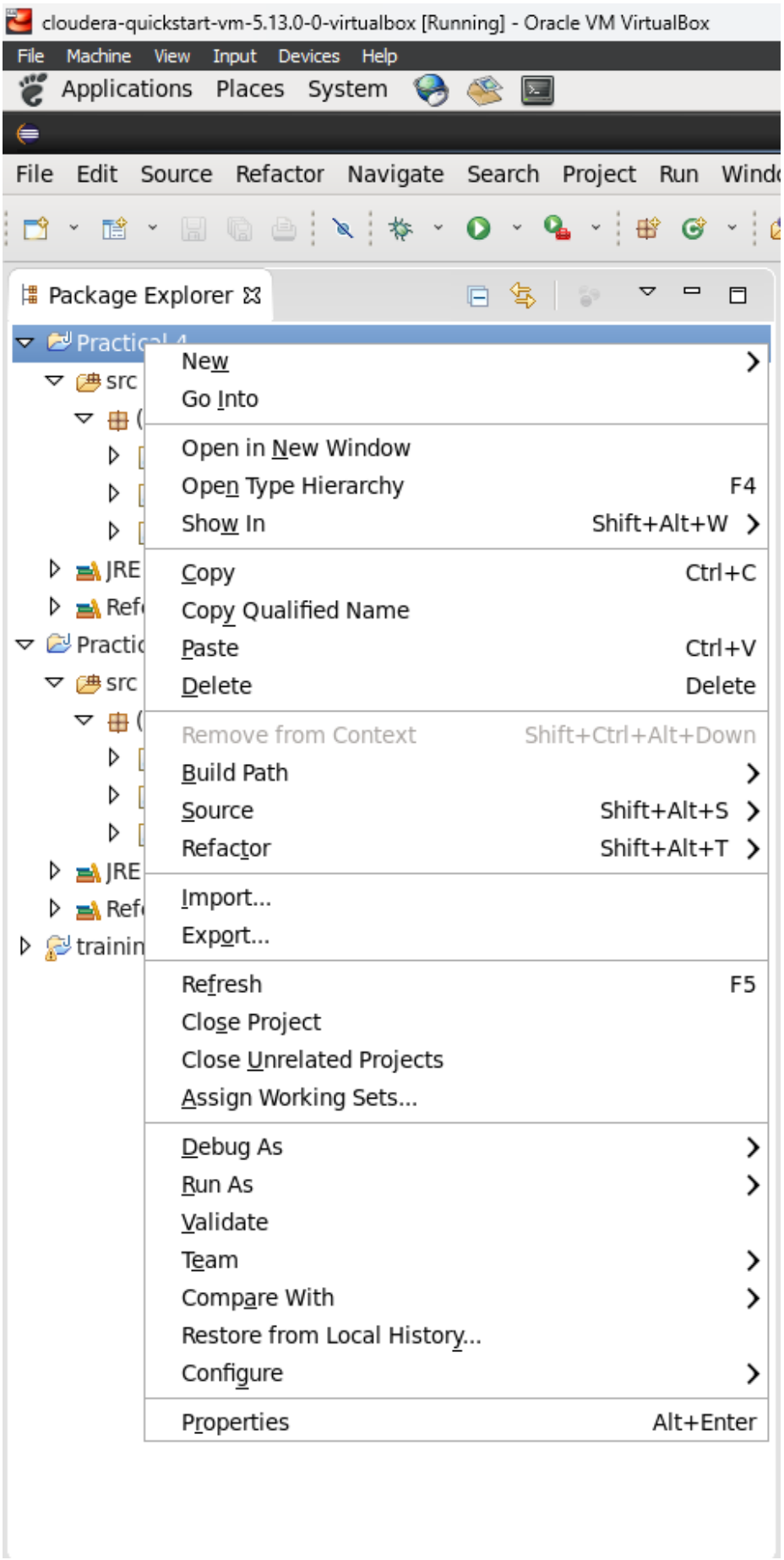

JAR Selection
usr
lib
hadoop
client
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
hadoop-mapreduce-client-common.jar 737.5 KB 10/04/2017
curator-framework-2.7.1.jar 181.9 KB 10/04/2017
curator-framework.jar 181.9 KB 10/04/2017
curator-recipes-2.7.1.jar 264.0 KB 10/04/2017
curator-recipes.jar 264.0 KB 10/04/2017
gson-2.2.4.jar 186.0 KB 10/04/2017
gson.jar 186.0 KB 10/04/2017
curator-client.jar 67.9 KB 10/04/2017
curator-client-2.7.1.jar 67.9 KB 10/04/2017
commons-net.jar 267.0 KB 10/04/2017
commons-lang.jar 277.6 KB 10/04/2017
commons-logging-1.1.3.jar 60.6 KB 10/04/2017
commons-logging.jar 60.6 KB 10/04/2017
commons-math3-3.1.1.jar 1.5 MB 10/04/2017
commons-math3.jar 1.5 MB 10/04/2017
commons-net-3.1.jar 267.0 KB 10/04/2017
guava-11.0.2.jar 1.6 MB 10/04/2017
guava.jar 1.6 MB 10/04/2017
.jar;.zip
Cancel
OK

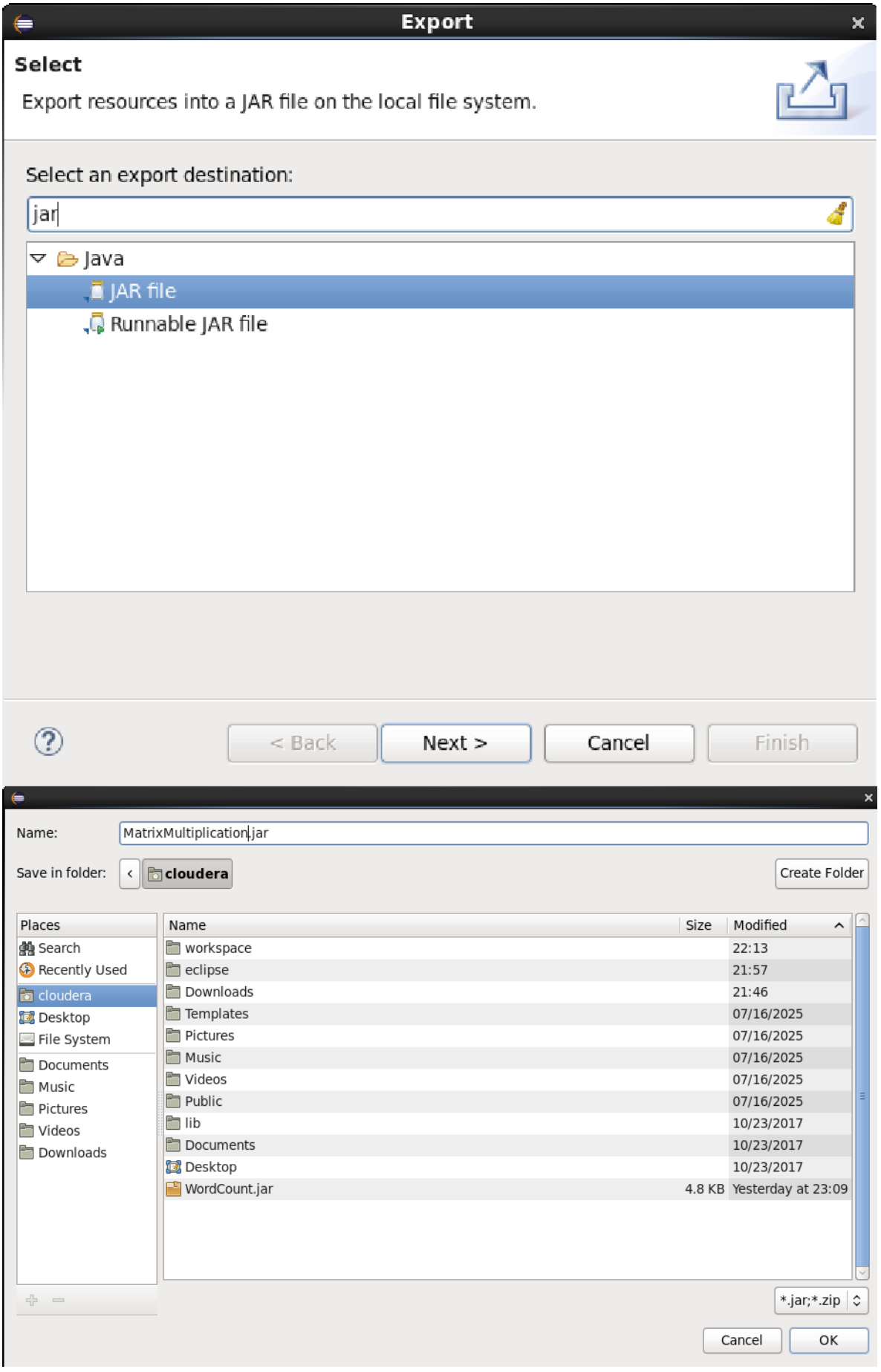

New Java Project
Java Settings
Define the Java build settings.
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
hadoop-auth.jar - /usr/lib/hadoop/client
hadoop-aws-2.6.0-cdh5.13.0.jar - /usr/lib/hadoop
hadoop-aws-2.6.0-cdh5.13.0.jar - /usr/lib/hadoop
hadoop-aws.jar - /usr/lib/hadoop
hadoop-aws.jar - /usr/lib/hadoop/client
hadoop-azure-datalake-2.6.0-cdh5.13.0.jar - /usr/
hadoop-azure-datalake-2.6.0-cdh5.13.0.jar - /usr/
hadoop-azure-datalake.jar - /usr/lib/hadoop
hadoop-azure-datalake.jar - /usr/lib/hadoop/clier
hadoop-common-2.6.0-cdh5.13.0-tests.jar - /usr/l
hadoop-common-2.6.0-cdh5.13.0.jar - /usr/lib/ha
hadoop-common-2.6.0-cdh5.13.0.jar - /usr/lib/ha
hadoop-common-tests.jar - /usr/lib/hadoop
hadoop-common.jar - /usr/lib/hadoop
hadoop-common.jar - /usr/lib/hadoop/client
hadoop-hdfs-2.6.0-cdh5.13.0.jar - /usr/lib/hadoop
hadoop-hdfs.jar - /usr/lib/hadoop/client
hadoop-mapreduce-client-app-2.6.0-cdh5.13.0.ja
hadoop-mapreduce-client-app.jar - /usr/lib/hadoc
hadoop-mapreduce-client-common-2.6.0-cdh5.13
hadoop-mapreduce-client-common.jar - /usr/lib/h
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
Migrate JAR File...
< Back
Next >
Cancel
Finish

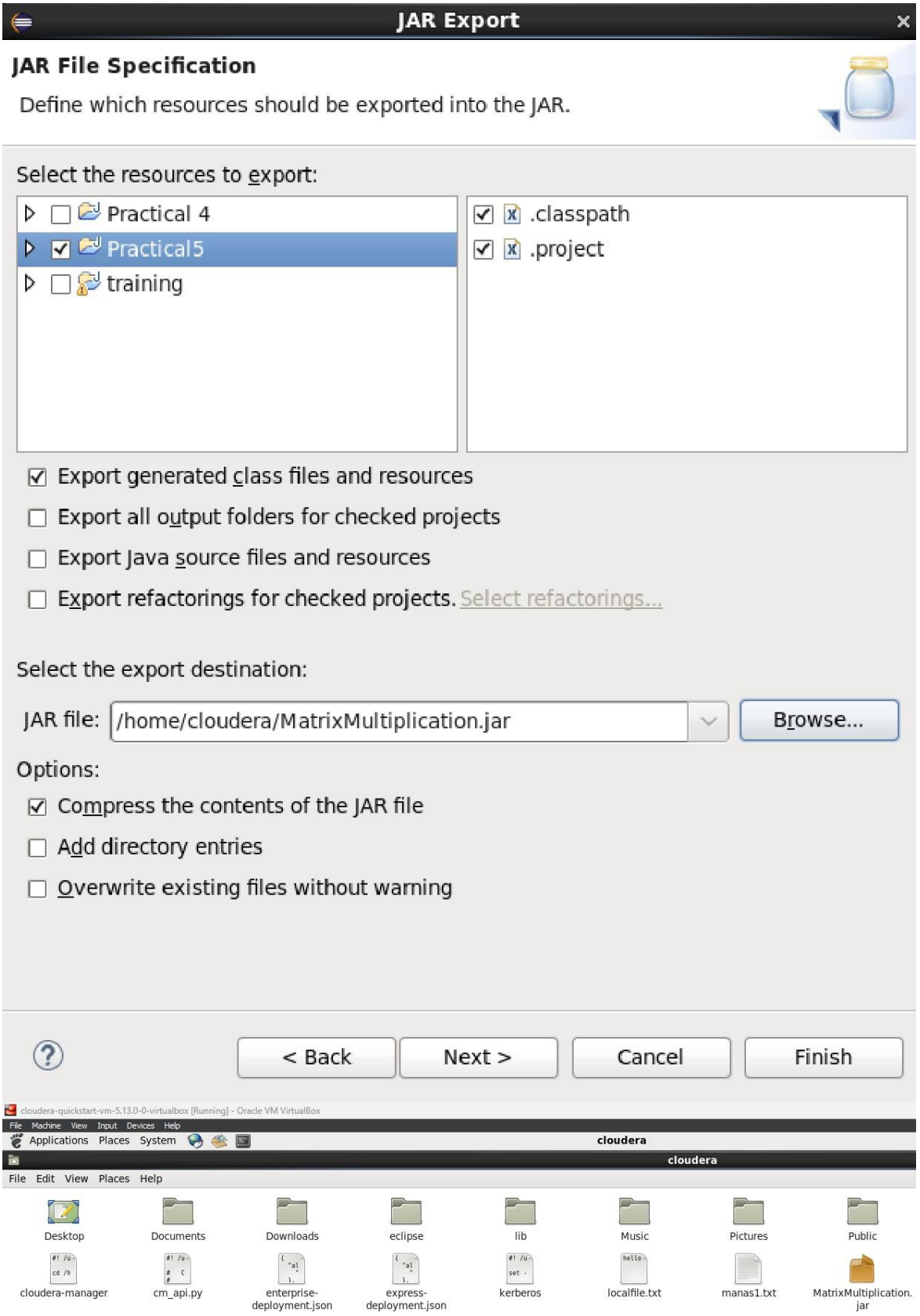

New Java Class
Java Class
The use of the default package is discouraged.
Source folder:
Practical5/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
Name:
MatrixMultiplicationMapper
Modifiers:
public
package
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Cancel
Finish

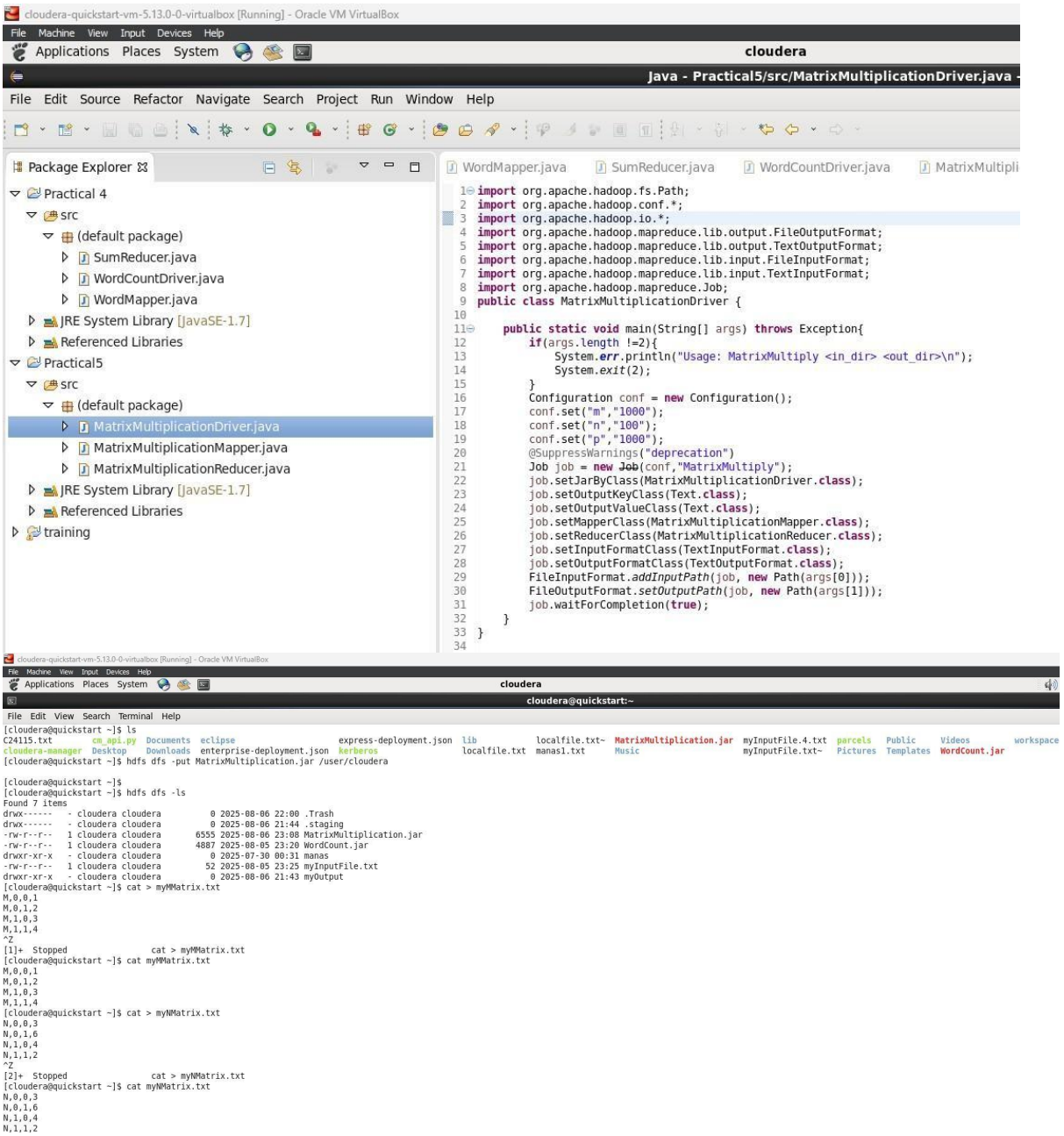

cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
Applications Places System
cloudera
Java - Practical5/src/MatrixMultiplicationMapper.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Practical 4
src
(default package)
SumReducer.java
WordCountDriver.java
WordMapper.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
Practical5
src
(default package)
MatrixMultiplicationDriver.java
MatrixMultiplicationMapper.java
MatrixMultiplicationReducer.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
training
StubDriver.java
WordMapper.java
SumReducer.java
WordCountDriver.java
MatrixMultiplicatior
import java.io.IOException;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MatrixMultiplicationMapper 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();
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);
outputValue.set(indicesAndValue[0]+","+indicesAndValue[2]+","+indicesAndValue[3]);
context.write(outputKey, outputValue);
}
}else{
for(int i =0; i<m; i++){
outputKey.set(i+","+indicesAndValue[2]);
outputValue.set("N,"+indicesAndValue[1]+","+indicesAndValue[3]);
context.write(outputKey, outputValue);
}
}
}
}


cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
Applications Places System
cloudera
Java - Practical5/src/MatrixMultiplicationReducer.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Practical 4
src
(default package)
SumReducer.java
WordCountDriver.java
WordMapper.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
Practical5
src
(default package)
MatrixMultiplicationDriver.java
MatrixMultiplicationMapper.java
MatrixMultiplicationReducer.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
training
StubDriver.java
WordMapper.java
SumReducer.java
WordCountDriver.java
MatrixMultiplicationMapp
import java.io.IOException;
import java.util.HashMap;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class MatrixMultiplicationReducer extends Reducer<Text, Text, Text, Text>{
@Override
public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException{
String[] value;
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)));
}
}
}


cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
File Edit Source Refactor Navigate Search Project Run
Package Explorer
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Shift+Alt+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Shift+Ctrl+Alt+Down
Build Path
Source Shift+Alt+S
Refactor Shift+Alt+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Assign Working Sets...
Debug As
Run As
Validate
Team
Compare With
Restore from Local History...
Configure
Properties Alt+Enter

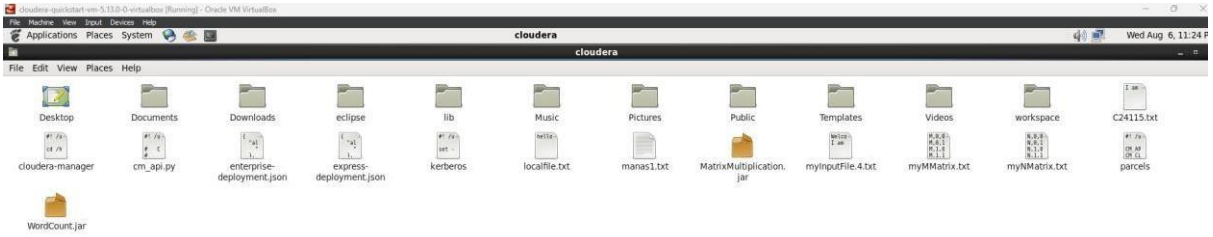

Export
Select
Export resources into a JAR file on the local file system.
Select an export destination:
jar
Java
JAR file
Runnable JAR file
< Back
Next >
Cancel
Finish
Name:
MatrixMultiplication.jar
Save in folder:
cloudera
Create Folder
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
workspace 22:13
eclipse 21:57
Downloads 21:46
Templates 07/16/2025
Pictures 07/16/2025
Music 07/16/2025
Videos 07/16/2025
Public 07/16/2025
lib 10/23/2017
Documents 10/23/2017
Desktop 10/23/2017
WordCount.jar 4.8 KB Yesterday at 23:09
.jar;.zip
Cancel
OK

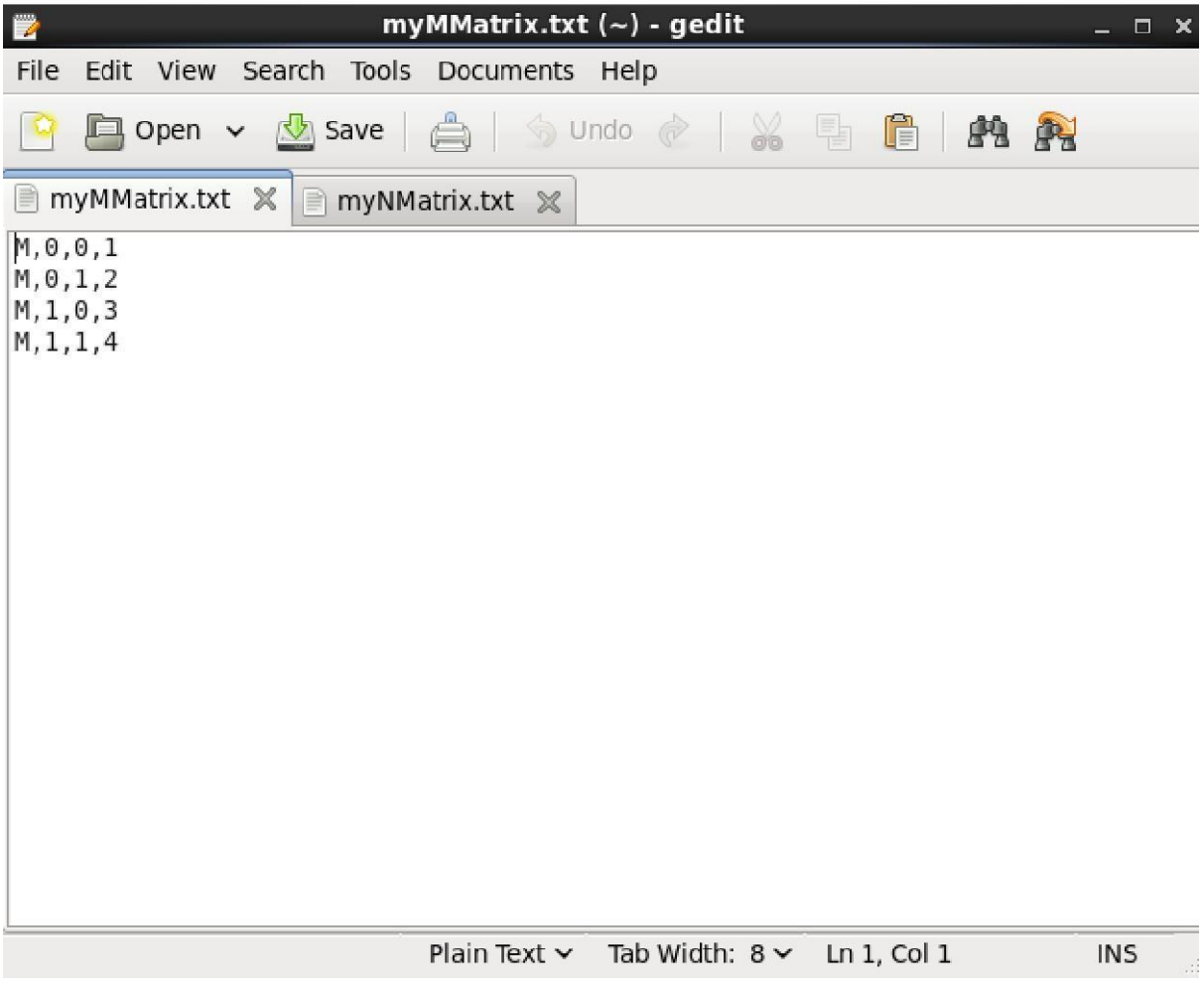

JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
Practical 4
Practical5
training
.classpath
.project
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file: /home/cloudera/MatrixMultiplication.jar
Browse...
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Cancel
Finish
cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera
cloudera
File Edit View Places Help
Desktop
Documents
Downloads
eclipse
lib
Music
Pictures
Public
cloudera-manager
cm_api.py
enterprise-deployment.json
express-deployment.json
kerberos
localfile.txt
manas1.txt
MatrixMultiplication.jar

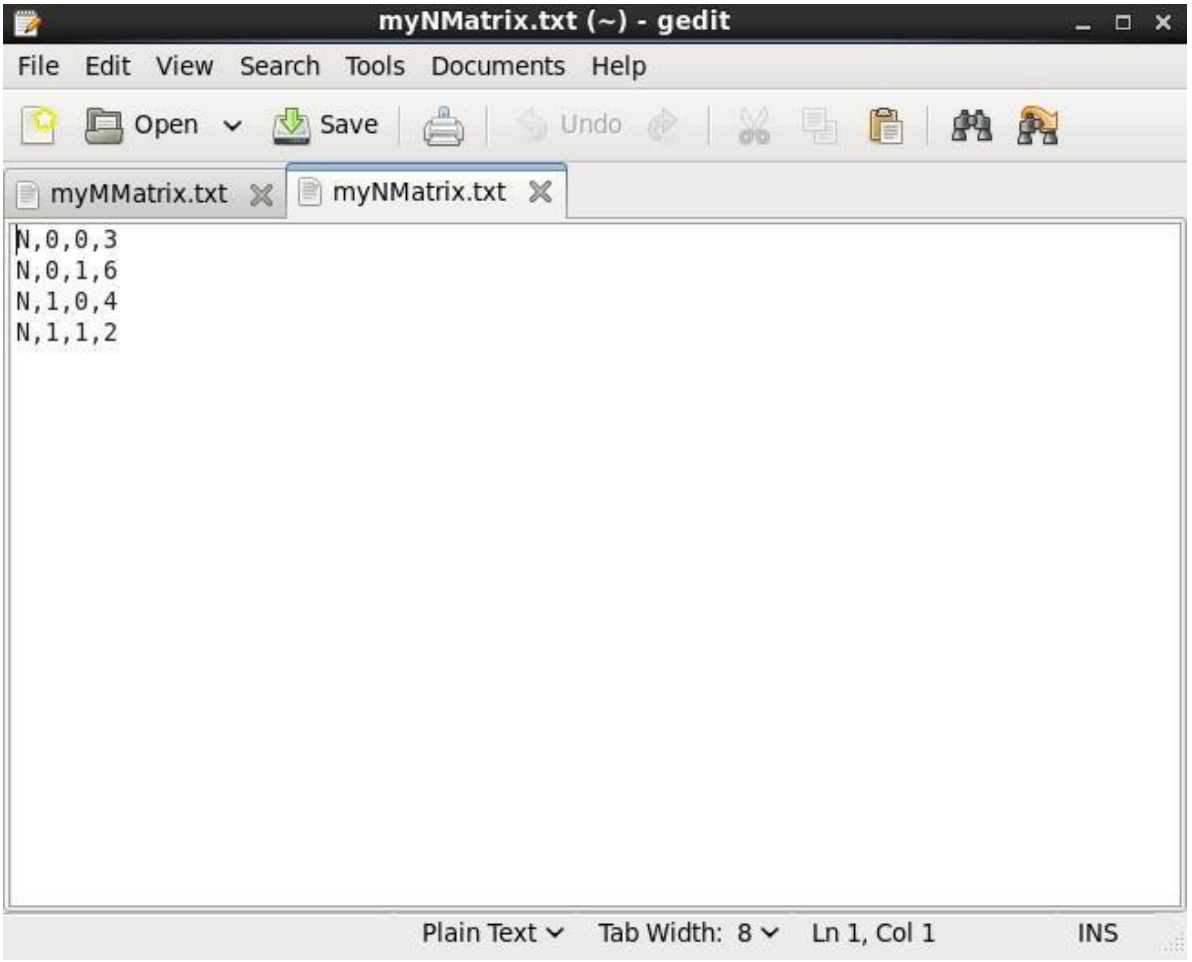

cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera
Java - Practical5/src/MatrixMultiplicationDriver.java -
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Practical 4
src
(default package)
SumReducer.java
WordCountDriver.java
WordMapper.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
Practical5
src
(default package)
MatrixMultiplicationDriver.java
MatrixMultiplicationMapper.java
MatrixMultiplicationReducer.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
training
WordMapper.java
SumReducer.java
WordCountDriver.java
MatrixMultipli
1 import org.apache.hadoop.fs.Path;
2 import org.apache.hadoop.conf.*;
3 import org.apache.hadoop.io.*;
4 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
5 import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
6 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
7 import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
8 import org.apache.hadoop.mapreduce.Job;
9 public class MatrixMultiplicationDriver {
10
11 public static void main(String[] args) throws Exception{
12 if(args.length !=2){
13 System.err.println("Usage: MatrixMultiply <in_dir> <out_dir>\n");
14 System.exit(2);
15 }
16 Configuration conf = new Configuration();
17 conf.set("m","1000");
18 conf.set("n","100");
19 conf.set("p","1000");
20 @SuppressWarnings("deprecation")
21 Job job = new Job(conf,"MatrixMultiply");
22 job.setJarByClass(MatrixMultiplicationDriver.class);
23 job.setOutputKeyClass(Text.class);
24 job.setOutputValueClass(Text.class);
25 job.setMapperClass(MatrixMultiplicationMapper.class);
26 job.setReducerClass(MatrixMultiplicationReducer.class);
27 job.setInputFormatClass(TextInputFormat.class);
28 job.setOutputFormatClass(TextOutputFormat.class);
29 FileInputFormat.addInputPath(job, new Path(args[0]));
30 FileOutputFormat.setOutputPath(job, new Path(args[1]));
31 job.waitForCompletion(true);
32 }
33 }
34

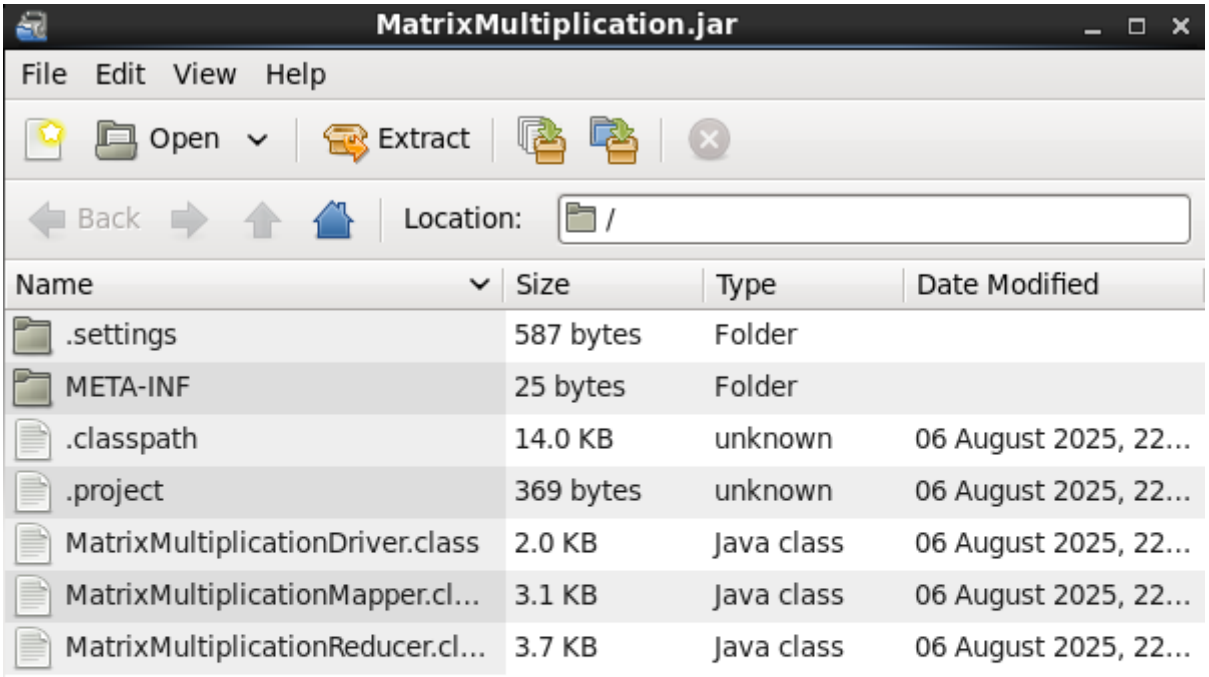
cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ ls
C24115.txt cm_api.py Documents eclipse express-deployment.json lib localfile.txt~ MatrixMultiplication.jar myInputFile.4.txt parcels Public Videos workspace
cloudera-manager Desktop Downloads enterprise-deployment.json kerberos localfile.txt manas1.txt Music myInputFile.txt~ Pictures Templates WordCount.jar
[cloudera@quickstart ~]$ hdfs dfs -put MatrixMultiplication.jar /user/cloudera
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$ hdfs dfs -ls
Found 7 items
drwx------ - cloudera cloudera 0 2025-08-06 22:00 .Trash
drwx------ - cloudera cloudera 0 2025-08-06 21:44 .staging
-rw-r--r-- 1 cloudera cloudera 6555 2025-08-06 23:08 MatrixMultiplication.jar
-rw-r--r-- 1 cloudera cloudera 4887 2025-08-05 23:20 WordCount.jar
drwxr-xr-x - cloudera cloudera 0 2025-07-30 00:31 manas
-rw-r--r-- 1 cloudera cloudera 52 2025-08-05 23:25 myInputFile.txt
drwxr-xr-x - cloudera cloudera 0 2025-08-06 21:43 myOutput
[cloudera@quickstart ~]$ cat > myMMatrix.txt
M,0,0,1
M,0,1,2
M,1,0,3
M,1,1,4
^Z
[1]+ Stopped cat > myMMatrix.txt
[cloudera@quickstart ~]$ cat myMMatrix.txt
M,0,0,1
M,0,1,2
M,1,0,3
M,1,1,4
[cloudera@quickstart ~]$ cat > myNMatrix.txt
N,0,0,3
N,0,1,6
N,1,0,4
N,1,1,2
^Z
[2]+ Stopped cat > myNMatrix.txt
[cloudera@quickstart ~]$ cat myNMatrix.txt
N,0,0,3
N,0,1,6
N,1,0,4
N,1,1,2

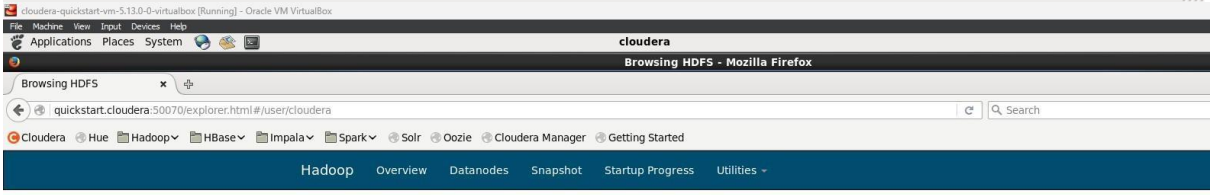
```
[cloudera@quickstart ~]$ hdfs dfs -mkdir /user/cloudera/matrixInput
[cloudera@quickstart ~]$ hdfs dfs -put myMMatrix.txt /user/cloudera/matrixInput/
[cloudera@quickstart ~]$ hdfs dfs -put myNMatrix.txt /user/cloudera/matrixInput/
[cloudera@quickstart ~]$ hdfs dfs -ls
Found 8 items
drwx------   - cloudera cloudera          0 2025-08-06 22:00 .Trash
drwx------   - cloudera cloudera          0 2025-08-06 21:44 .staging
-rw-r--r--   1 cloudera cloudera       6555 2025-08-06 23:08 MatrixMultiplication.jar
-rw-r--r--   1 cloudera cloudera       4887 2025-08-05 23:20 WordCount.jar
drwxr-xr-x   - cloudera cloudera          0 2025-07-30 00:31 manas
drwxr-xr-x   - cloudera cloudera          0 2025-08-06 23:14 matrixInput
-rw-r--r--   1 cloudera cloudera         52 2025-08-05 23:25 myInputFile.txt
drwxr-xr-x   - cloudera cloudera          0 2025-08-06 21:43 myOutput
[cloudera@quickstart ~]$ hadoop jar MatrixMultiplication.jar MatrixMultiplicationDriver matrixInput matrixOutput
25/08/06 23:15:43 INFO client.RMProxy: Connecting to ResourceManager at quickstart.cloudera/10.0.2.15:8032
25/08/06 23:15:44 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
25/08/06 23:15:45 INFO input.FileInputFormat: Total input paths to process : 2
25/08/06 23:15:45 INFO mapreduce.JobSubmitter: number of splits:2
25/08/06 23:15:45 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1754541476929_0002
25/08/06 23:15:46 INFO impl.YarnClientImpl: Submitted application application_1754541476929_0002
25/08/06 23:15:46 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1754541476929_0002/
25/08/06 23:15:46 INFO mapreduce.Job: Running job: job_1754541476929_0002
25/08/06 23:16:03 INFO mapreduce.Job: Job job_1754541476929_0002 running in uber mode : false
25/08/06 23:16:03 INFO mapreduce.Job:  map 0% reduce 0%
25/08/06 23:16:23 INFO mapreduce.Job:  map 100% reduce 0%
25/08/06 23:16:34 INFO mapreduce.Job:  map 100% reduce 100%
25/08/06 23:16:35 INFO mapreduce.Job: Job job_1754541476929_0002 completed successfully
25/08/06 23:16:35 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=31169
                FILE: Number of bytes written=505790
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=336
                HDFS: Number of bytes written=36
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=17917440
                Total time spent by all reduces in occupied slots (ms)=3418112
                Total time spent by all map tasks (ms)=34995
```






myNMatrix.txt (~) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
myMMatrix.txt
myNMatrix.txt
N,0,0,3
N,0,1,6
N,1,0,4
N,1,1,2
Plain Text
Tab Width: 8
Ln 1, Col 1
INS



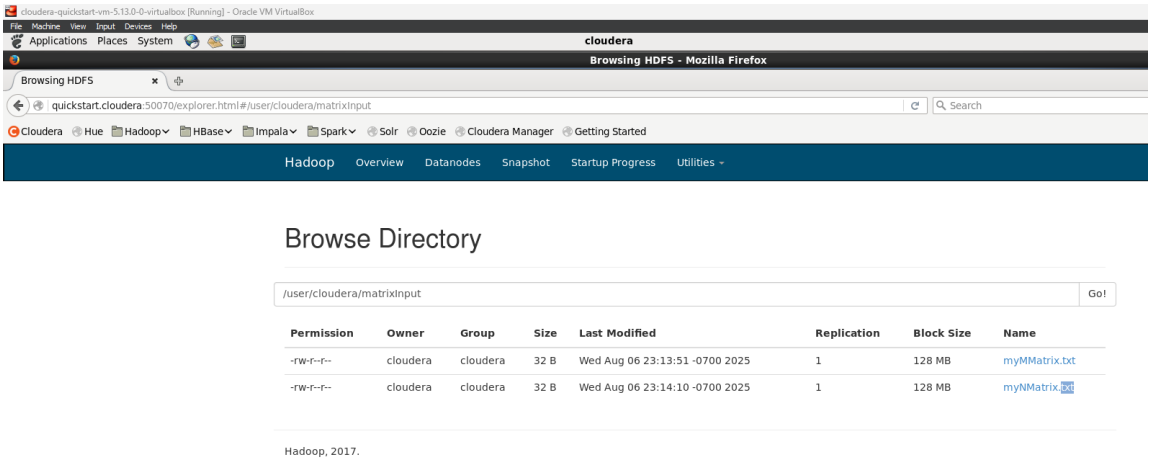


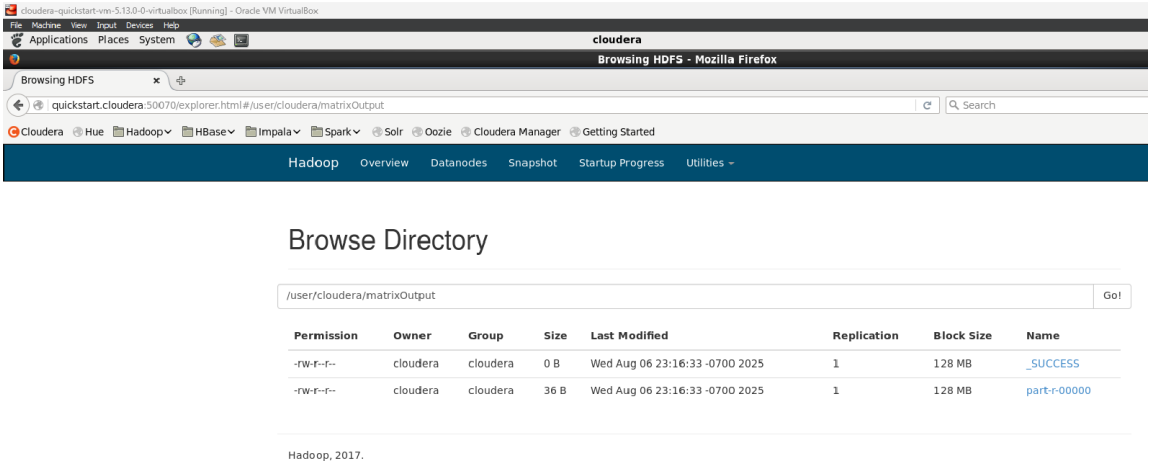
Browse Directory

/user/cloudera Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwx------	cloudera	cloudera	0 B	Wed Aug 06 22:00:01 -0700 2025	0	0 B	.Trash
drwx------	cloudera	cloudera	0 B	Wed Aug 06 23:16:35 -0700 2025	0	0 B	.staging
-rw-r--r--	cloudera	cloudera	6.4 KB	Wed Aug 06 23:08:52 -0700 2025	1	128 MB	MatrixMultiplication.jar
-rw-r--r--	cloudera	cloudera	4.77 KB	Tue Aug 05 23:20:44 -0700 2025	1	128 MB	WordCount.jar
drwxr-xr-x	cloudera	cloudera	0 B	Wed Jul 30 00:31:25 -0700 2025	0	0 B	manas
drwxr-xr-x	cloudera	cloudera	0 B	Wed Aug 06 23:14:10 -0700 2025	0	0 B	matrixInput
drwxr-xr-x	cloudera	cloudera	0 B	Wed Aug 06 23:16:33 -0700 2025	0	0 B	matrixOutput
-rw-r--r--	cloudera	cloudera	52 B	Tue Aug 05 23:25:23 -0700 2025	1	128 MB	myInputFile.txt
drwxr-xr-x	cloudera	cloudera	0 B	Wed Aug 06 21:43:57 -0700 2025	0	0 B	myOutput

Hadoop, 2017.


cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera
Browsing HDFS - Mozilla Firefox
Browsing HDFS
quickstart.cloudera:50070/explorer.html#/user/cloudera/matrixInput
Search
Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started
Hadoop Overview Datanodes Snapshot Startup Progress Utilities
Browse Directory
/user/cloudera/matrixInput
Go!
Permission Owner Group Size Last Modified Replication Block Size Name
-rw-r--r-- cloudera cloudera 32 B Wed Aug 06 23:13:51 -0700 2025 1 128 MB myMMatrix.txt
-rw-r--r-- cloudera cloudera 32 B Wed Aug 06 23:14:10 -0700 2025 1 128 MB myNMatrix.txt
Hadoop, 2017.


cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera
Browsing HDFS - Mozilla Firefox
Browsing HDFS
quickstart.cloudera:50070/explorer.html#/user/cloudera/matrixOutput
Search
Cloudera Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager Getting Started
Hadoop Overview Datanodes Snapshot Startup Progress Utilities
Browse Directory
/user/cloudera/matrixOutput
Go!
Permission Owner Group Size Last Modified Replication Block Size Name
-rw-r--r-- cloudera cloudera 0 B Wed Aug 06 23:16:33 -0700 2025 1 128 MB _SUCCESS
-rw-r--r-- cloudera cloudera 36 B Wed Aug 06 23:16:33 -0700 2025 1 128 MB part-r-00000
Hadoop, 2017.

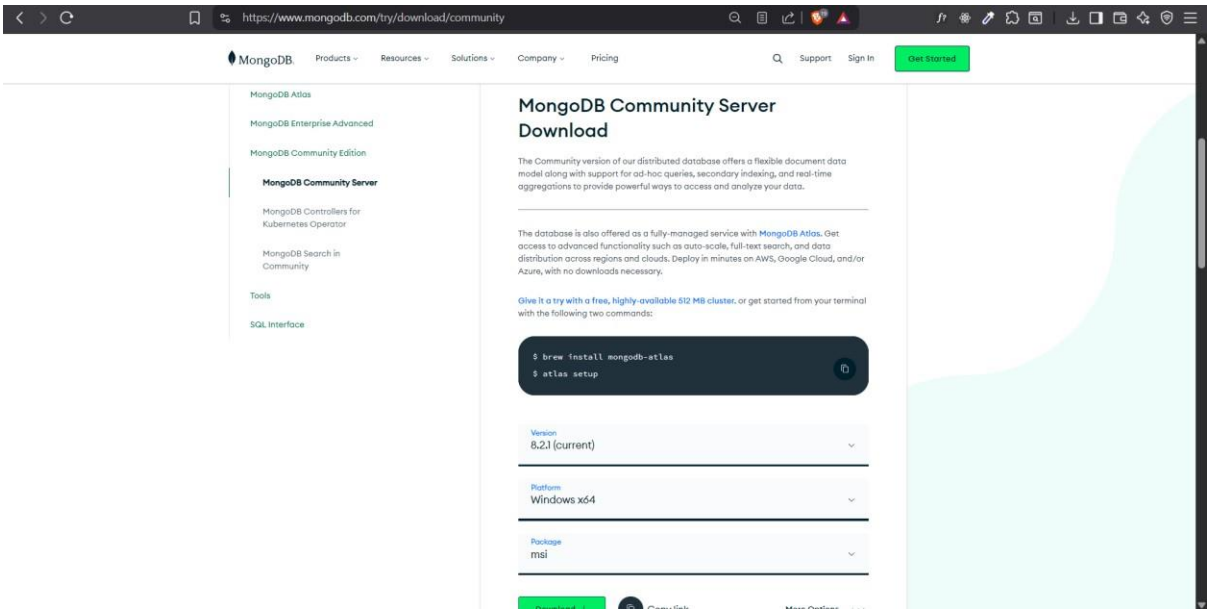
```
                Total time spent by all reduce tasks (ms)=6676
                Total vcore-milliseconds taken by all map tasks=34995
                Total vcore-milliseconds taken by all reduce tasks=6676
                Total megabyte-milliseconds taken by all map tasks=17917440
                Total megabyte-milliseconds taken by all reduce tasks=3418112
        Map-Reduce Framework
                Map input records=8
                Map output records=8000
                Map output bytes=95120
                Map output materialized bytes=31054
                Input split bytes=272
                Combine input records=0
                Combine output records=0
                Reduce input groups=3996
                Reduce shuffle bytes=31054
                Reduce input records=8000
                Reduce output records=4
                Spilled Records=16000
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=451
                CPU time spent (ms)=1730
                Physical memory (bytes) snapshot=379375616
                Virtual memory (bytes) snapshot=2108305408
                Total committed heap usage (bytes)=152174592
        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=64
        File Output Format Counters
                Bytes Written=36
[cloudera@quickstart ~]$ hdfs dfs -ls
Found 9 items
drwx------   - cloudera cloudera          0 2025-08-06 22:00 .Trash
drwx------   - cloudera cloudera          0 2025-08-06 23:16 .staging
-rw-r--r--   1 cloudera cloudera       6555 2025-08-06 23:08 MatrixMultiplication.jar
-rw-r--r--   1 cloudera cloudera       4887 2025-08-05 23:20 WordCount.jar
drwxr-xr-x   - cloudera cloudera          0 2025-07-30 00:31 manas
drwxr-xr-x   - cloudera cloudera          0 2025-08-06 23:14 matrixInput
drwxr-xr-x   - cloudera cloudera          0 2025-08-06 23:16 matrixOutput
-rw-r--r--   1 cloudera cloudera         52 2025-08-05 23:25 myInputFile.txt
drwxr-xr-x   - cloudera cloudera          0 2025-08-06 21:43 myOutput
[cloudera@quickstart ~]$ hdfs dfs -ls /user/cloudera/matrixOutput
Found 2 items
-rw-r--r--   1 cloudera cloudera          0 2025-08-06 23:16 /user/cloudera/matrixOutput/_SUCCESS
-rw-r--r--   1 cloudera cloudera         36 2025-08-06 23:16 /user/cloudera/matrixOutput/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat /user/cloudera/matrixOutput/part-r-00000
0,0,11.0
0,1,10.0
1,0,25.0
1,1,26.0
[cloudera@quickstart ~]$
```

Practical No 3 : MongoDB

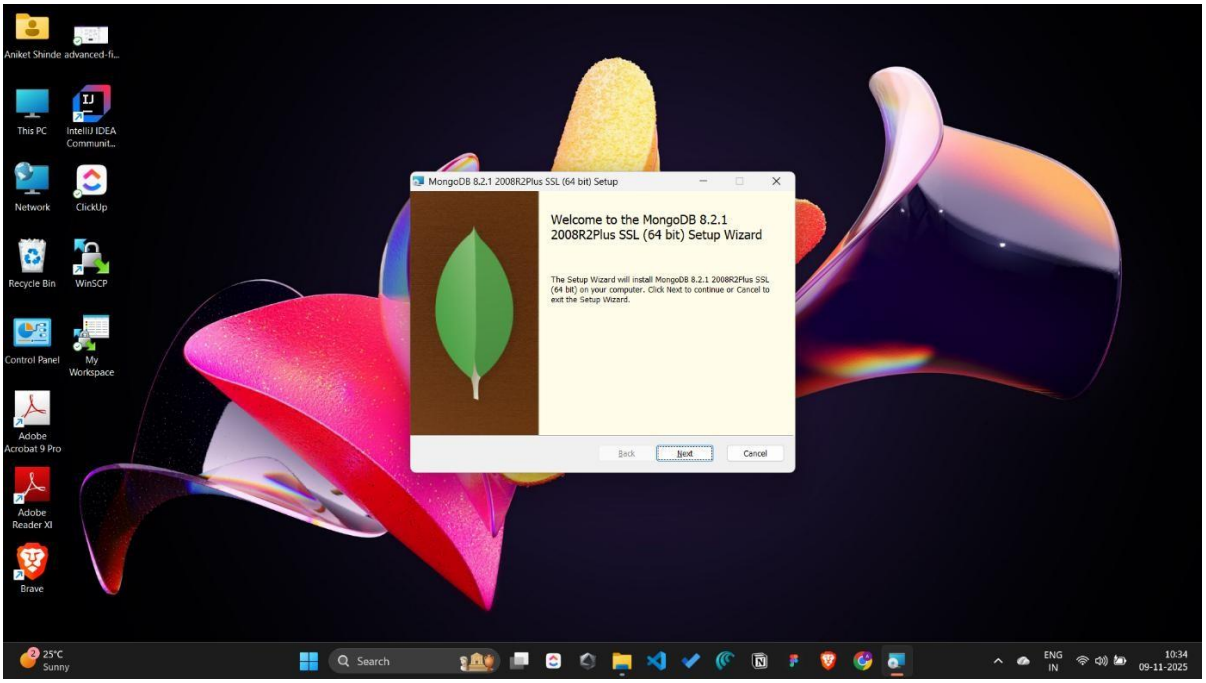
1. Installation

Installing MongoDB on Windows

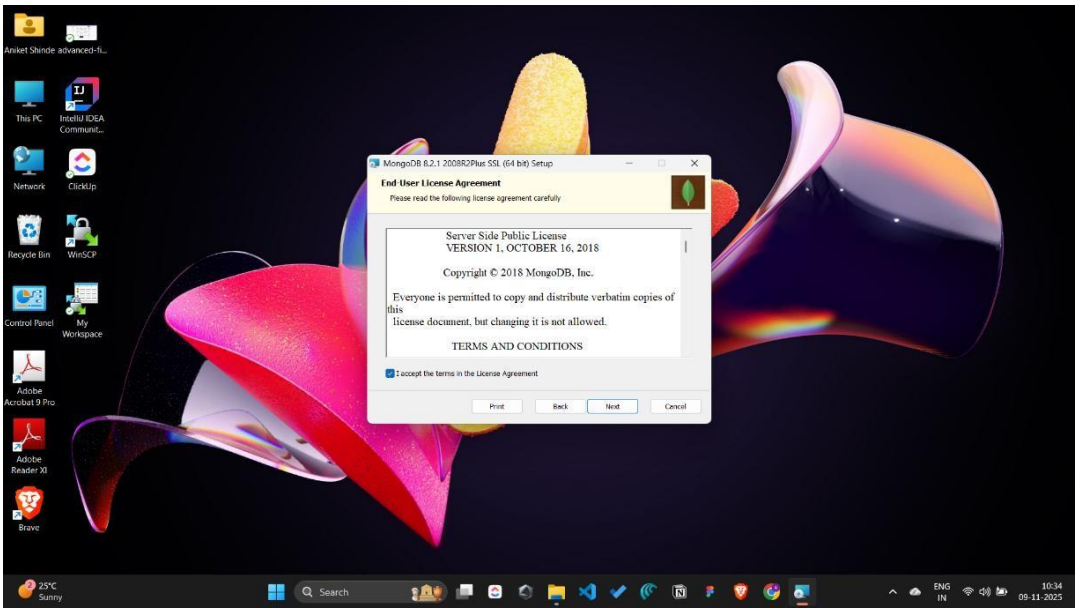
Download MongoDB Community Server. We will install **the 64-bit version** for Windows.



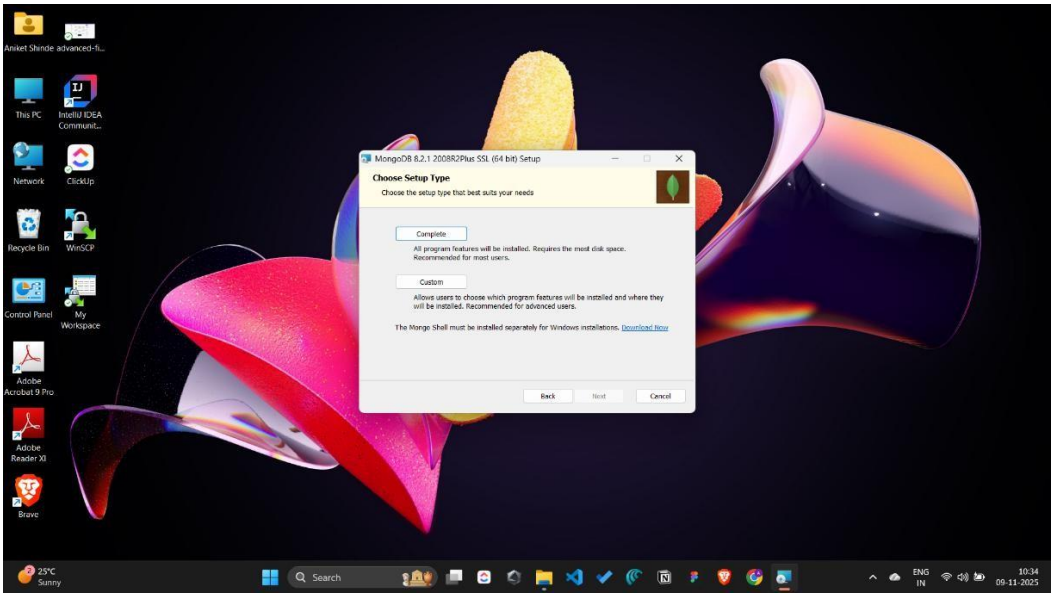
2. Click on Setup. Once download is complete open the MSI file. Click Next in the start-up screen.



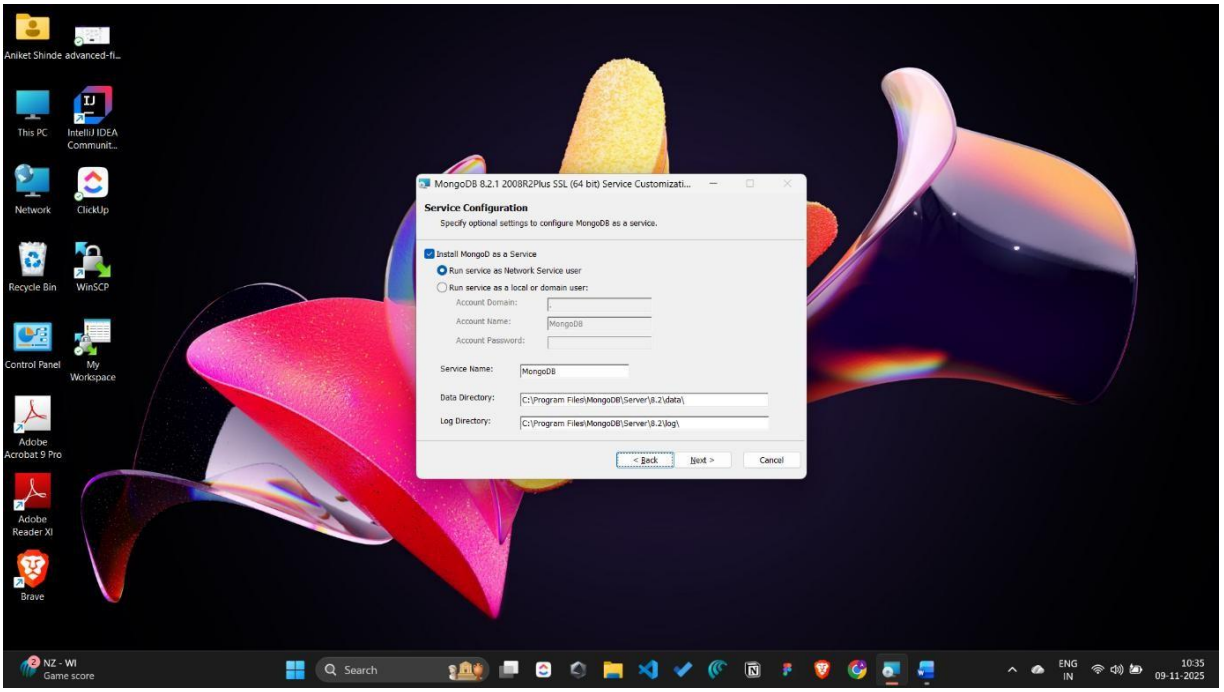
3. Accept the End-User License Agreement & Click Next.



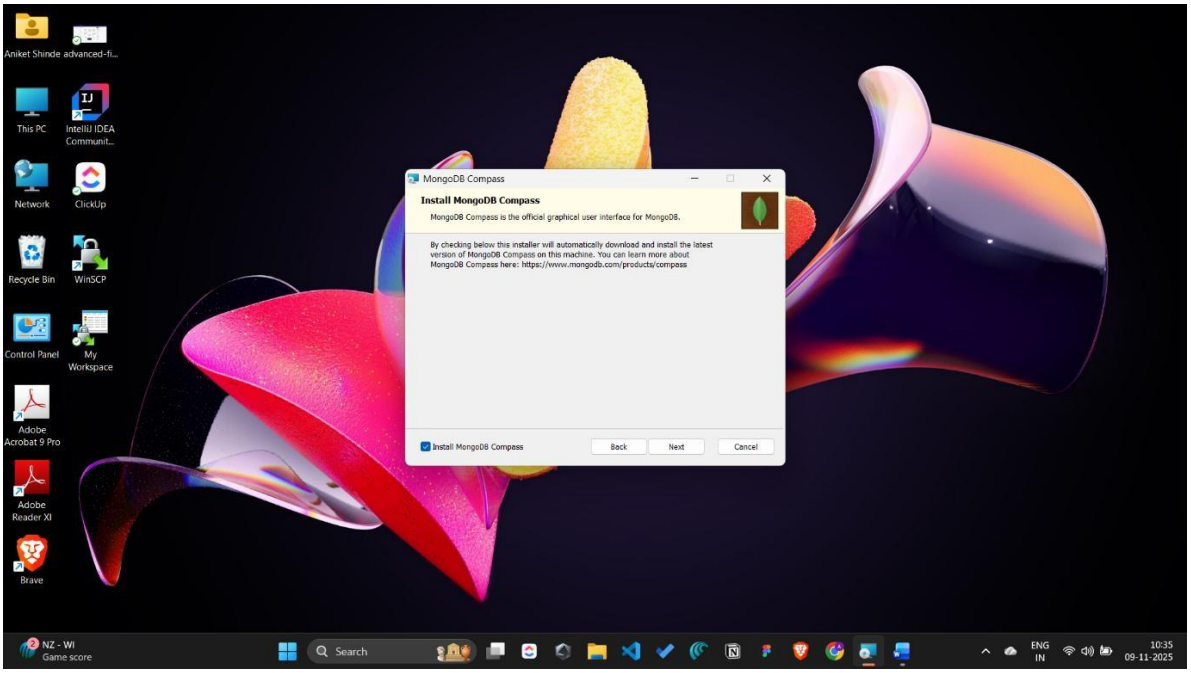
4. Click on the "complete" button to install all of the components. The custom option can be used to install selective components or if you want to change the location of the installation.



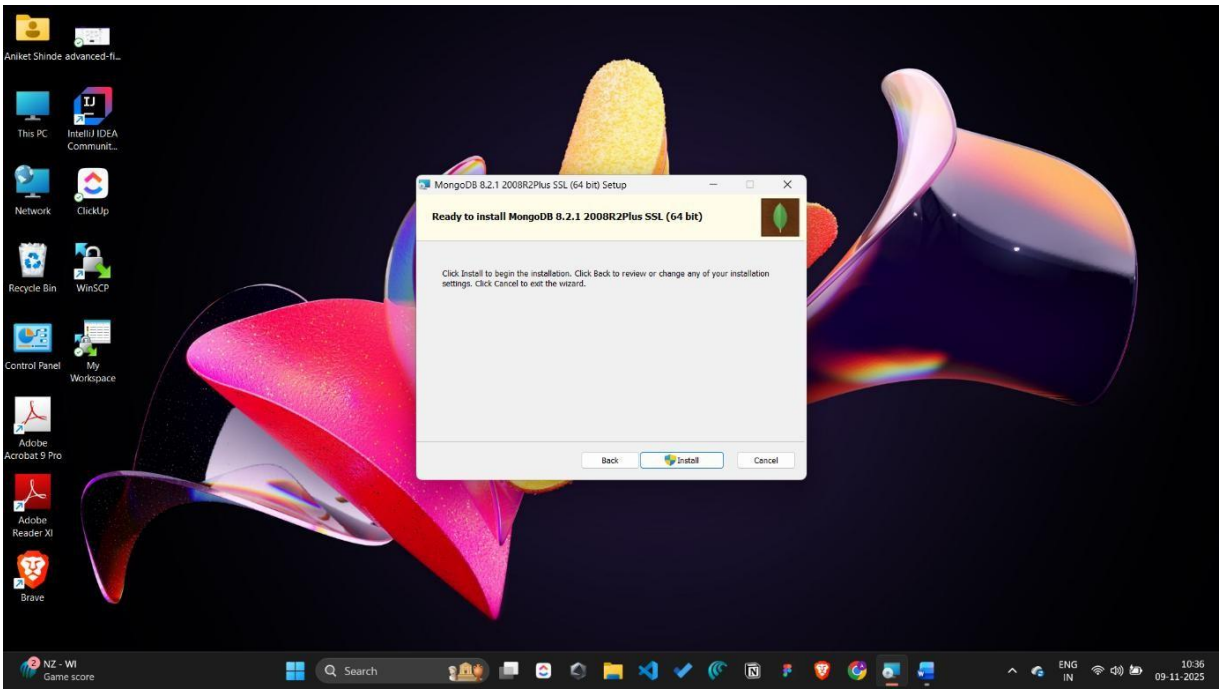
5. Service Configuration: Select "Run service as Network Service user" --> Click Next.



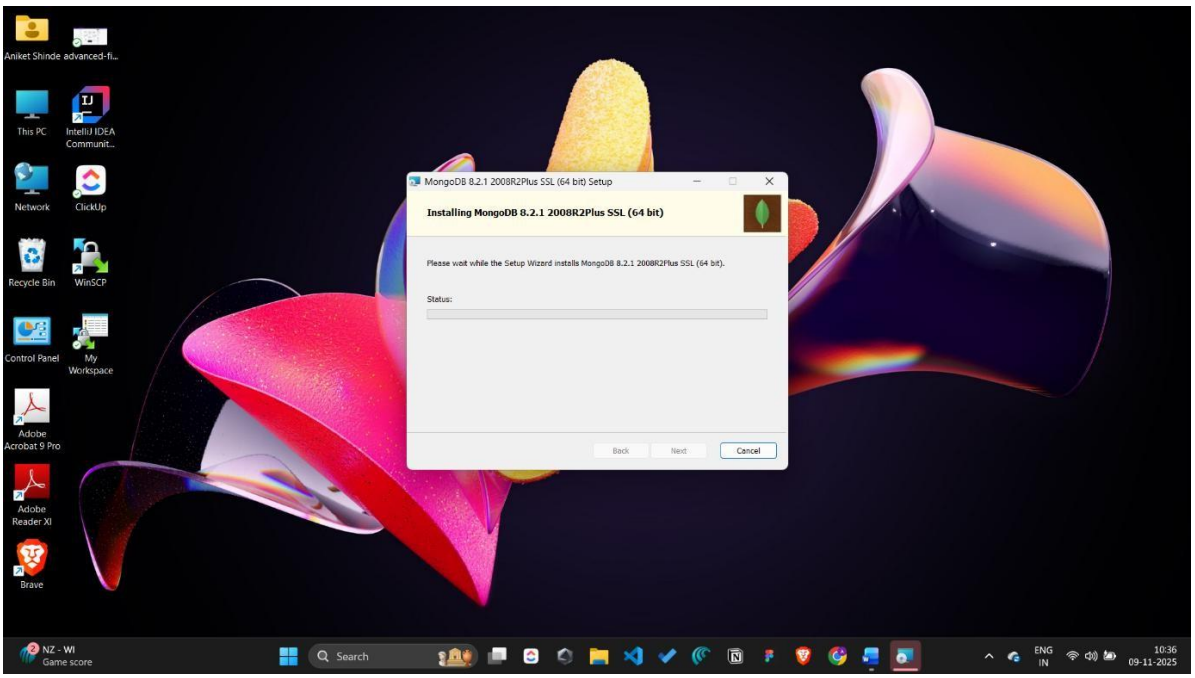
6. Install Mongodb Compass GUI (Optional).



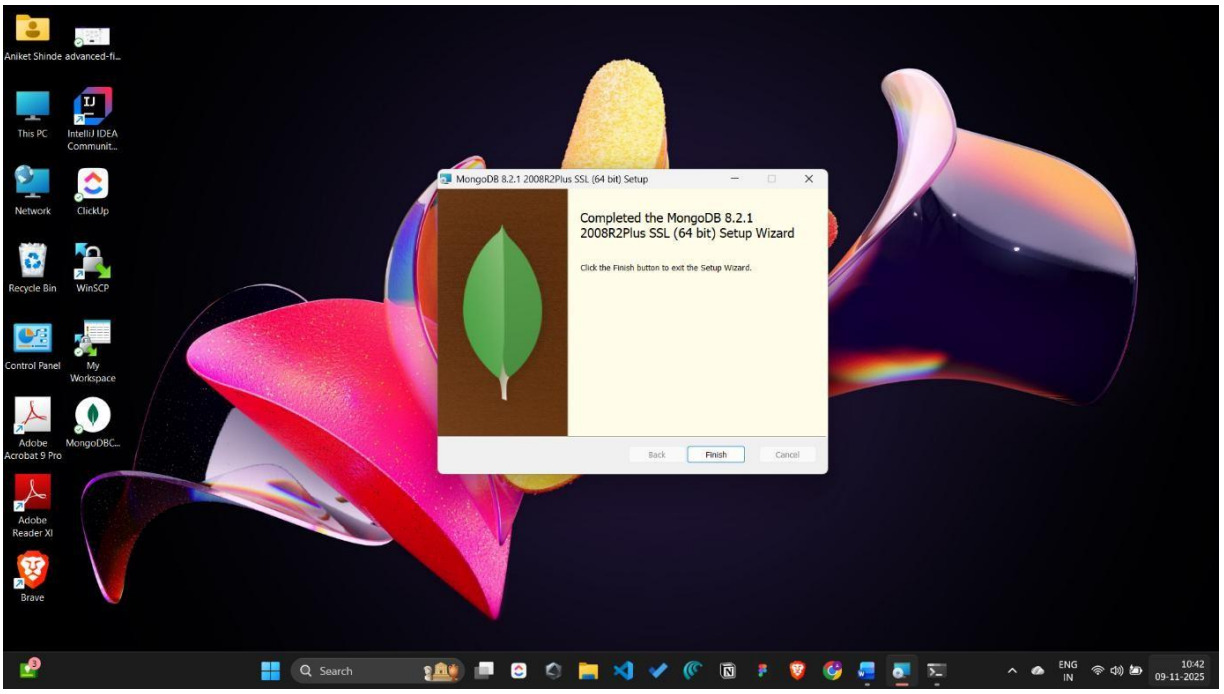
7. Click on the Install button to start the installation.



8. Installation begins. Click Next once completed.



9. Once complete the installation, Click on the Finish button.



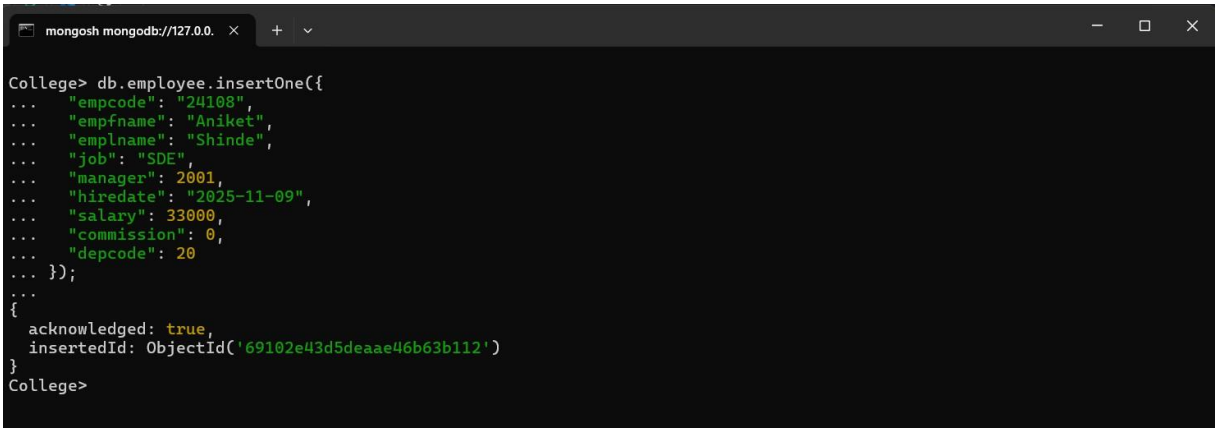
---

2. Sample Database Creation

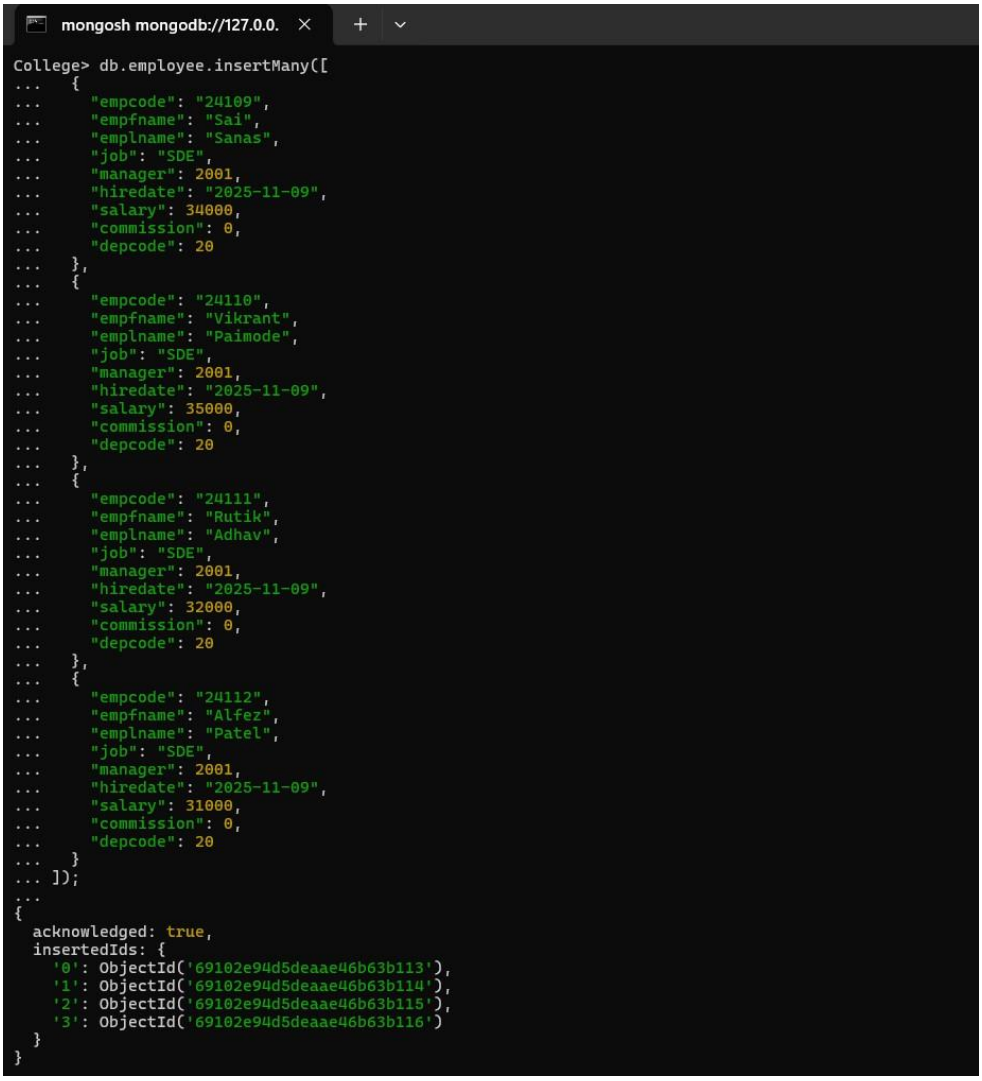
1. Now create the DB – College and use College

3. Query the Sample Database using MongoDB querying commands

1. Create a query & insert a record with insertOne()



2. Now, try to insert more records with - InsertMany()



3. To view all inserted records use query - find()

```
College> db.employee.find()
[
  {
    _id: ObjectId('69102e43d5deaae46b63b112'),
    empcode: '24108',
    empfname: 'Aniket',
    emplname: 'Shinde',
    job: 'SDE',
    manager: 2001,
    hiredate: '2025-11-09',
    salary: 33000,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b113'),
    empcode: '24109',
    empfname: 'Sai',
    emplname: 'Sanas',
    job: 'SDE-I',
    manager: 2001,
    hiredate: '2025-10-15',
    salary: 34500,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b114'),
    empcode: '24110',
    empfname: 'Vikrant',
    emplname: 'Paimode',
    job: 'SDE-II',
    manager: 2001,
    hiredate: '2025-09-28',
    salary: 36200,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b115'),
    empcode: '24111',
    empfname: 'Rutik',
    emplname: 'Adhav',
    job: 'Backend Engineer',
```

```
    manager: 2001,
    hiredate: '2025-08-10',
    salary: 33400,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b116'),
    empcode: '24112',
    empfname: 'Alfez',
    emplname: 'Patel',
    job: 'Frontend Engineer',
    manager: 2001,
    hiredate: '2025-07-25',
    salary: 31250,
    commission: 0,
    depcode: 20
  }
]
College>
```

4. To find one record with empname as Talha – use findOne()

```
  }
]
College> db.employee.findOne({empfname: "Aniket"})
{
  _id: ObjectId('69102e43d5deaae46b63b112'),
  empcode: '24108',
  empfname: 'Aniket',
  emplname: 'Shinde',
  job: 'SDE',
  manager: 2001,
  hiredate: '2025-11-09',
  salary: 33000,
  commission: 0,
  depcode: 20
}
College>
```

5. To find employee with job role as Backend Engineer– use find()

```
College> db.employee.find({ job: "Backend Engineer" });
[
  {
    _id: ObjectId('69102e94d5deaae46b63b115'),
    empcode: '24111',
    empfname: 'Rutik',
    emplname: 'Adhav',
    job: 'Backend Engineer',
    manager: 2001,
    hiredate: '2025-08-10',
    salary: 33400,
    commission: 0,
    depcode: 20
  }
]
College>
```

6. Finding all the records where salary is greater than 33000.

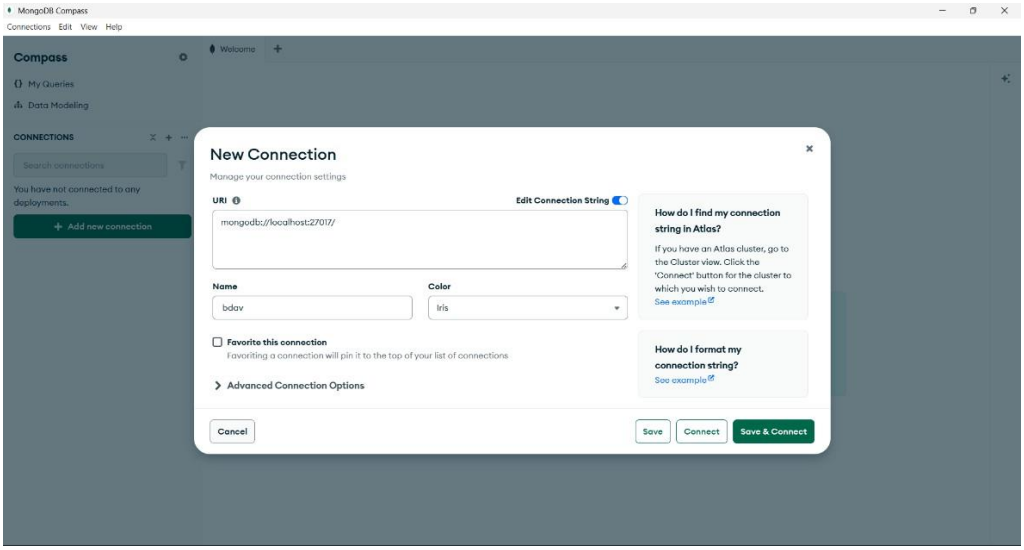
```
College> db.employee.find({ salary: { $gt: 33000 } });
[
  {
    _id: ObjectId('69102e94d5deaae46b63b113'),
    empcode: '24109',
    empfname: 'Sai',
    emplname: 'Sanas',
    job: 'SDE-I',
    manager: 2001,
    hiredate: '2025-10-15',
    salary: 34500,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b114'),
    empcode: '24110',
    empfname: 'Vikrant',
    emplname: 'Paimode',
    job: 'SDE-II',
    manager: 2001,
    hiredate: '2025-09-28',
    salary: 36200,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b115'),
    empcode: '24111',
    empfname: 'Rutik',
    emplname: 'Adhav',
    job: 'Backend Engineer',
    manager: 2001,
    hiredate: '2025-08-10',
    salary: 33400,
    commission: 0,
    depcode: 20
  }
]
College>
```

7. Finding the records with salary is > 34000 and depcode is 20.

```
College> db.employee.find({salary: { $gt: 34000 },depcode: 20});
[
  {
    _id: ObjectId('69102e94d5deaae46b63b113'),
    empcode: '24109',
    empfname: 'Sai',
    emplname: 'Sanas',
    job: 'SDE-I',
    manager: 2001,
    hiredate: '2025-10-15',
    salary: 34500,
    commission: 0,
    depcode: 20
  },
  {
    _id: ObjectId('69102e94d5deaae46b63b114'),
    empcode: '24110',
    empfname: 'Vikrant',
    emplname: 'Paimode',
    job: 'SDE-II',
    manager: 2001,
    hiredate: '2025-09-28',
    salary: 36200,
    commission: 0,
    depcode: 20
  }
]
College>
```

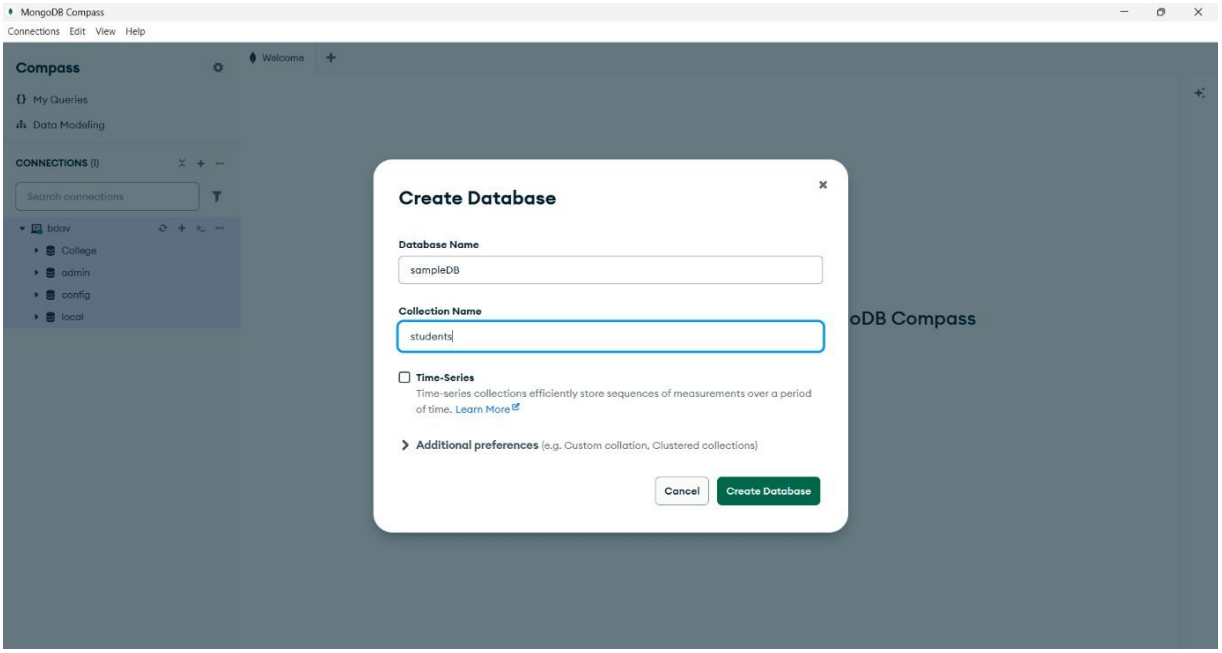
4. Operations on the collection with MongoDB Compass GUI.

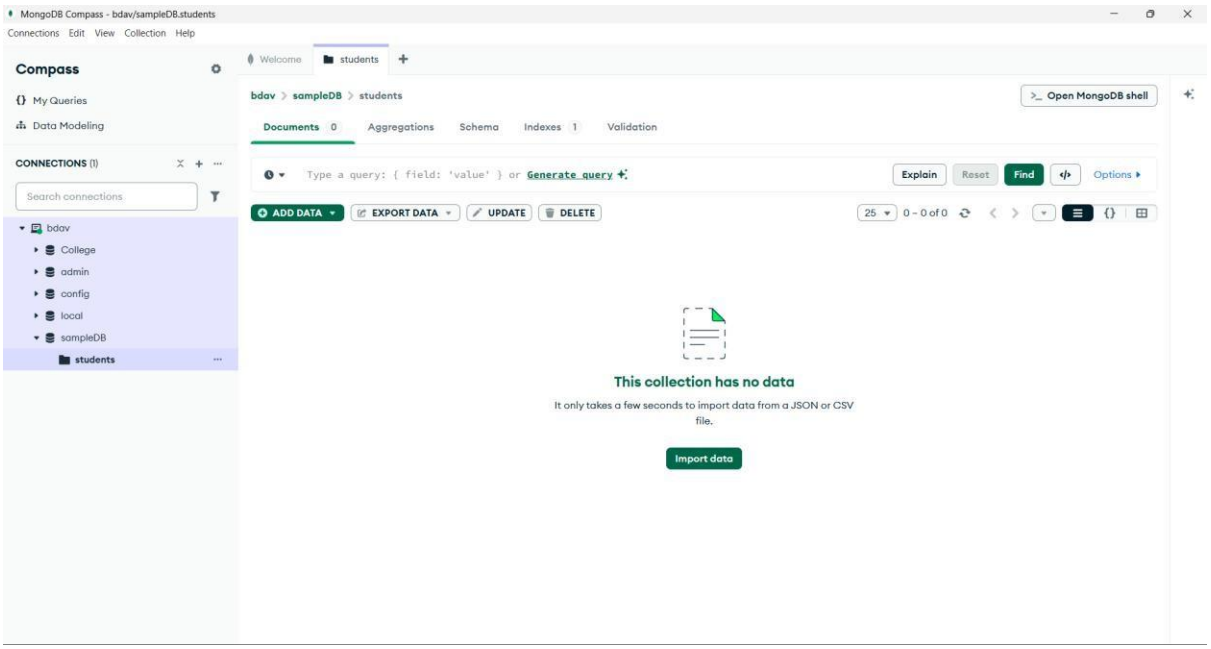
For that we have to open MongoDB Compass & create the new connection with following values and make sure to click on Save and connect.



A. Create Collection:

1.Create a database name as Students.

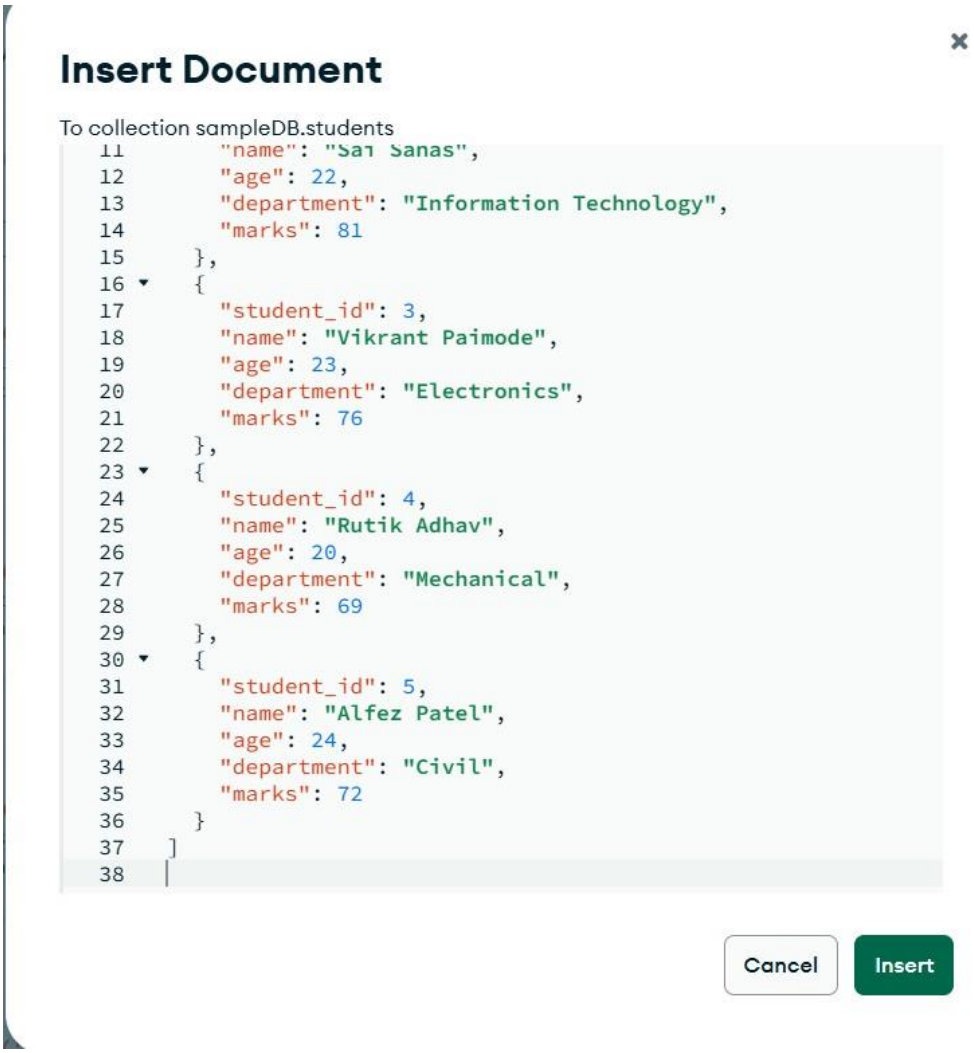


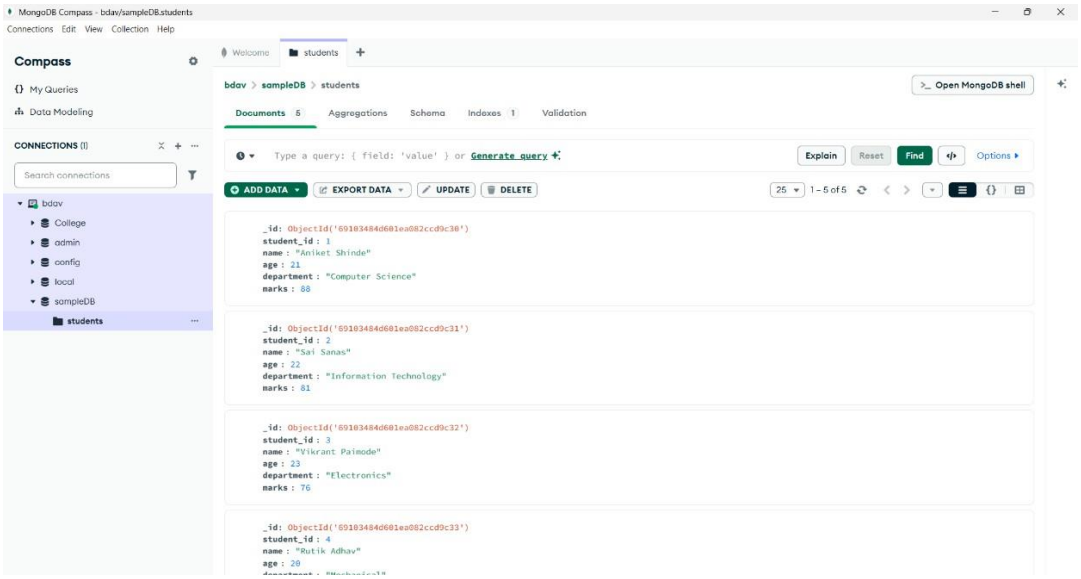


B. Insert Document

1. click on the add data -> and click on the insert document.

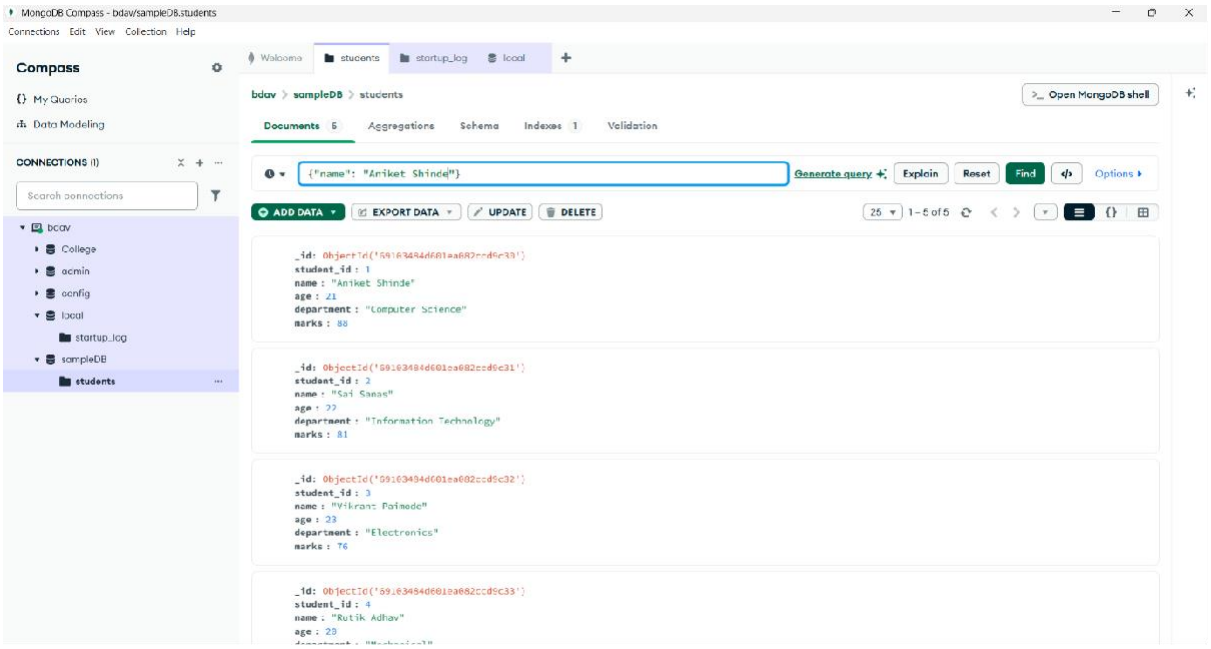
1. Insert Multiple values in that document:

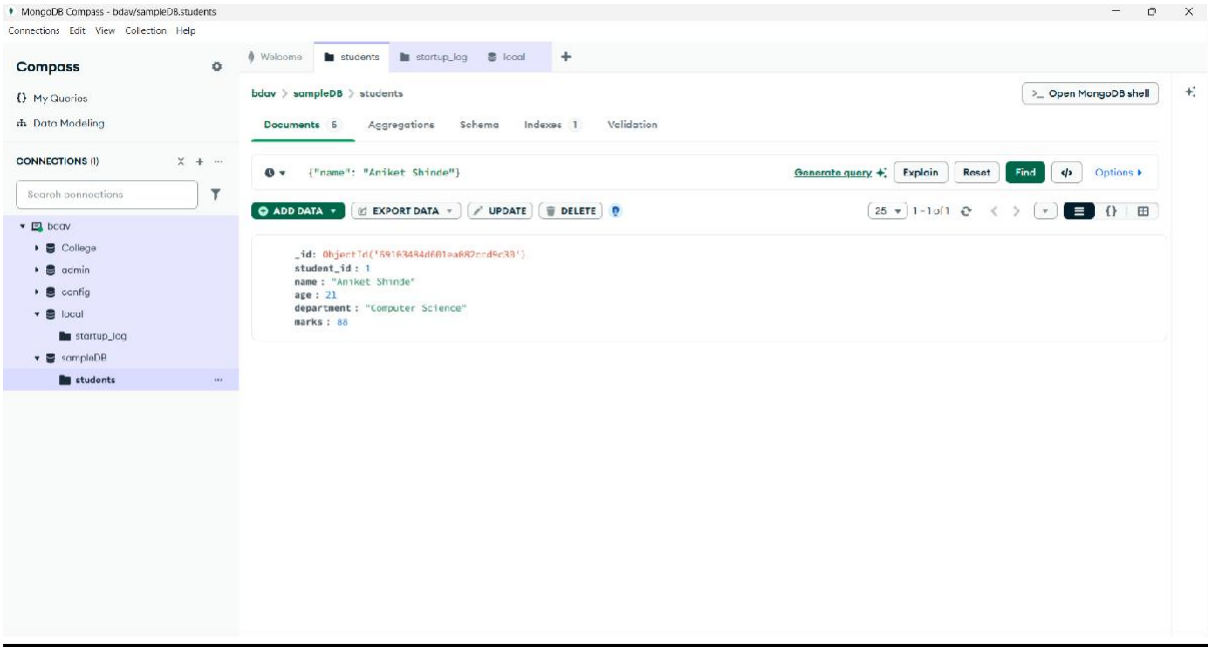




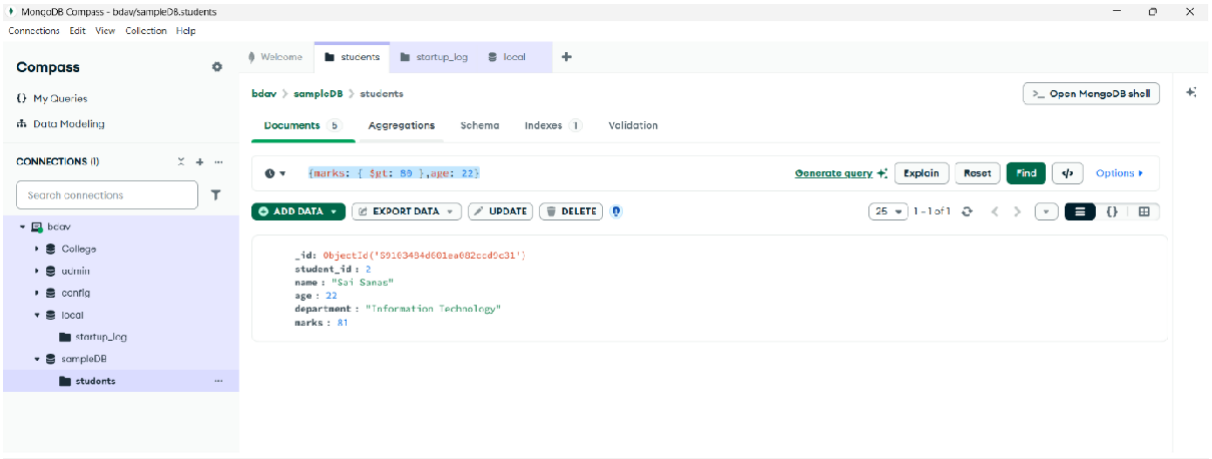
C. Query Document

Query the document for finding the name : "Talha Shinde"

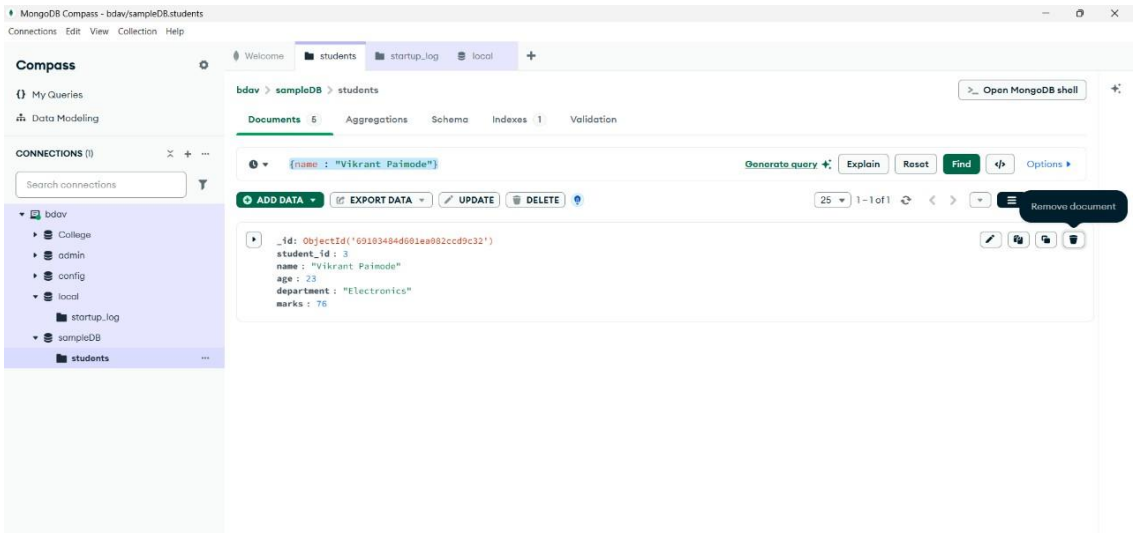


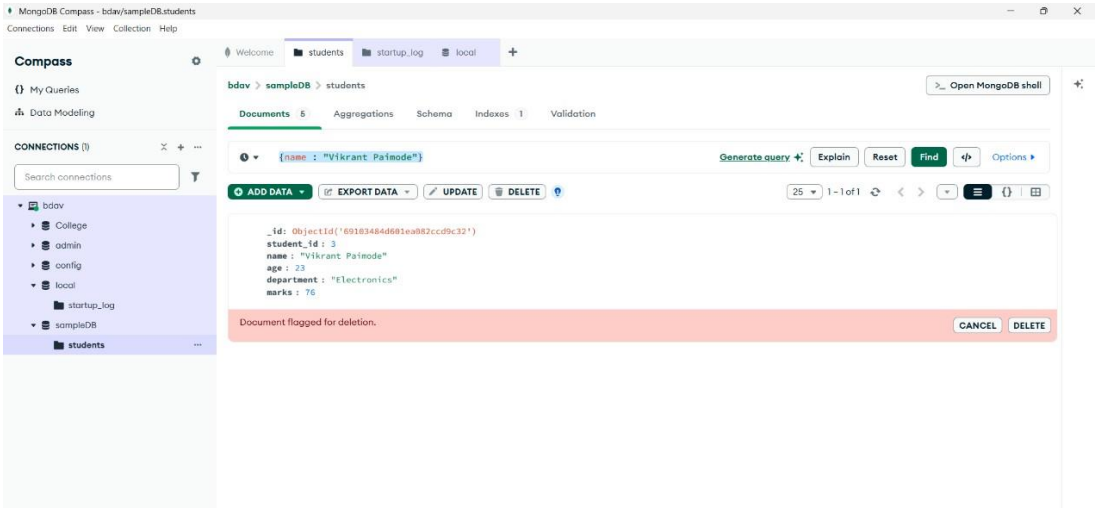


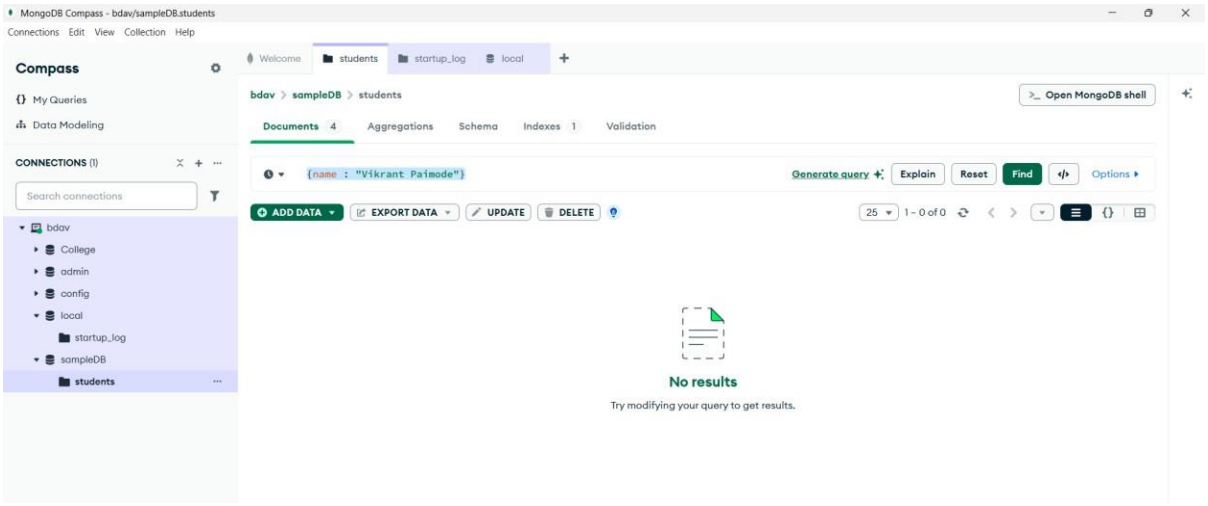
Query the document and find the Marks greater than 80 and age :20



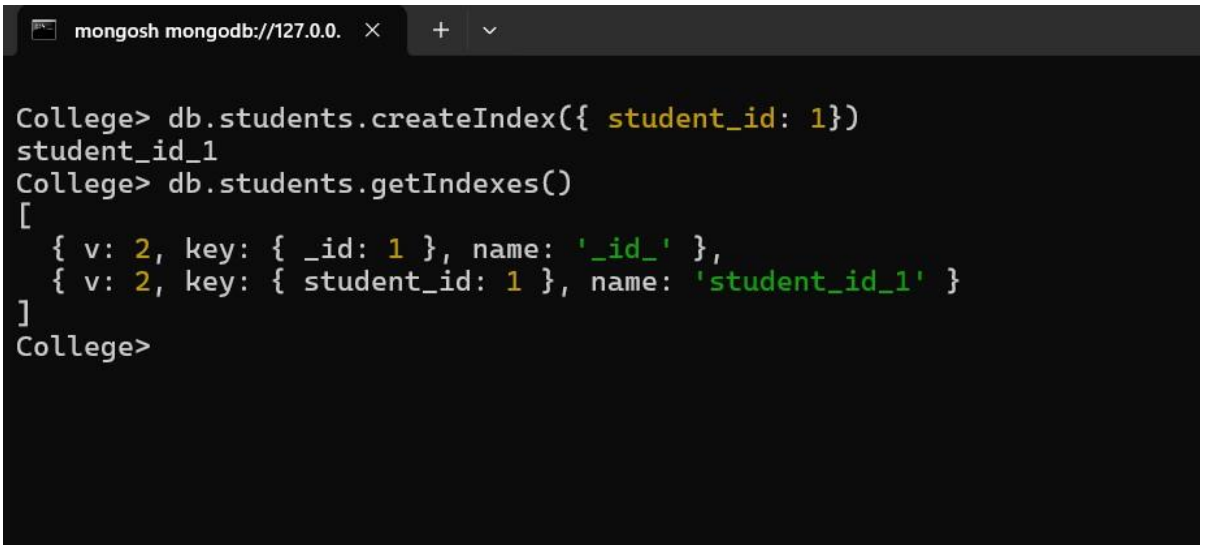
D. Delete Document






MongoDB Compass - bdav/sampleDB.students
Connections Edit View Collection Help
Compass
My Queries
Data Modeling
CONNECTIONS (1)
Search connections
bdav
College
admin
config
local
startup_log
sampleDB
students
Welcome
students
startup_log
local
bdav > sampleDB > students
Open MongoDB shell
Documents 4
Aggregations
Schema
Indexes 1
Validation
{name : "Vikrant Paimode"}
Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 0 of 0
No results
Try modifying your query to get results.

E. Indexing


mongosh mongodb://127.0.0.
College> db.students.createIndex({ student_id: 1})
student_id_1
College> db.students.getIndexes()
[
{ v: 2, key: { _id: 1 }, name: '_id_' },
{ v: 2, key: { student_id: 1 }, name: 'student_id_1' }
]
College>

Practical No 4: Hive

1. Hive Data Types

Primitive Data Types:

- **Numeric Types:**
  - TINYINT: 1-byte signed integer.
  - SMALLINT: 2-byte signed integer.
  - INT / INTEGER: 4-byte signed integer.
  - BIGINT: 8-byte signed integer.
  - FLOAT: 4-byte single-precision floating-point number.
  - DOUBLE: 8-byte double-precision floating-point number.
  - DECIMAL: Fixed-point decimal numbers with user-defined precision and scale.

- **Date/Time Types:**
  - DATE: Represents a date in 'YYYY-MM-DD' format.
  - TIMESTAMP: Represents a point in time with date and time, including optional

    nanosecond precision.
  - INTERVAL: Represents a period of time.

- **String Types:**
  - STRING: Variable-length character string.
  - VARCHAR(n): Variable-length character string with a maximum length of 'n'.
  - CHAR(n): Fixed-length character string with a length of 'n', padded with spaces if

    shorter.

- Miscellaneous Types:
    - BOOLEAN: Stores TRUE or FALSE values.
    - BINARY: Stores a sequence of bytes.

- Complex Data Types:
    - ARRAY<data_type>:

    An ordered collection of elements of the same data_type. Similar to a list in other programming languages.

    - MAP<key_data_type, value_data_type>:

    An associative array (key-value pairs) where keys are unique and of key_data_type, and values are of value_data_type

    - STRUCT<col_name : data_type, ...>:

    A record type that groups together fields (columns) of potentially different data types, similar to a struct in C or a record in other systems.

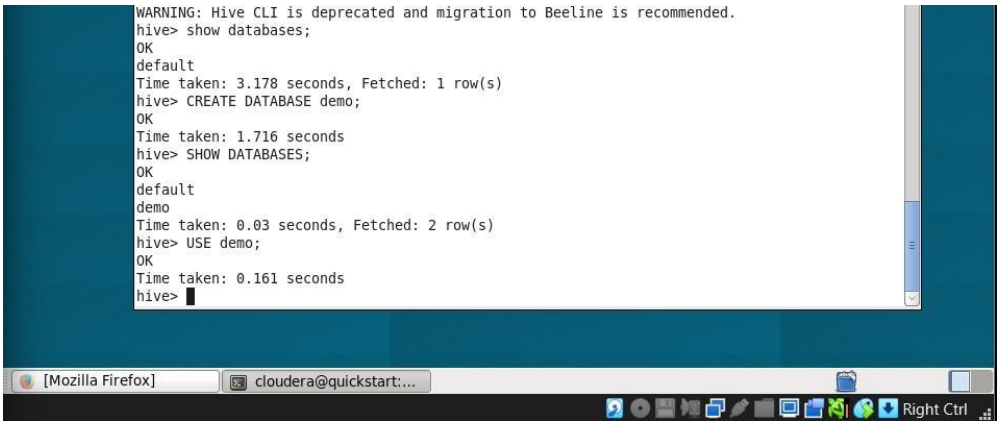
    - UNIONTYPE<data_type1, data_type2, ...>:

    A type that can hold a value of one of several specified data types at any given time.

2. Create Database & Table in Hive

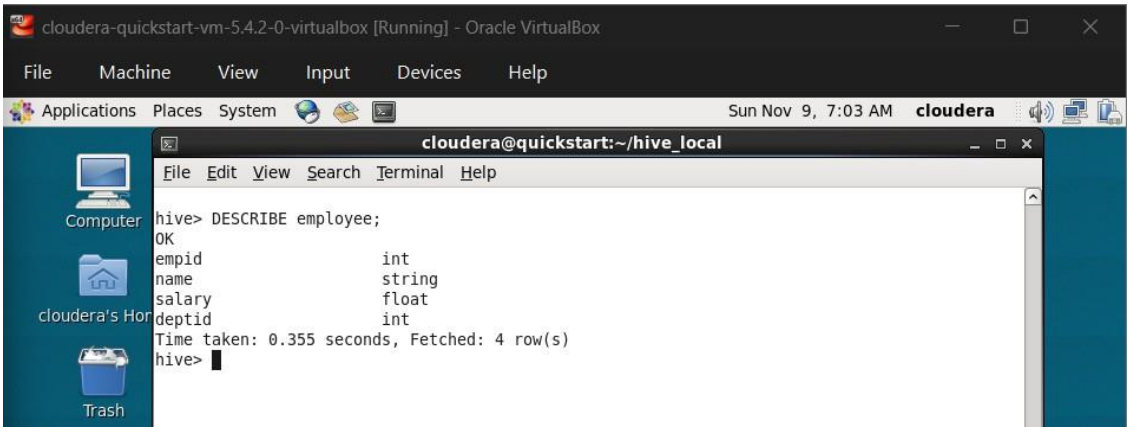
a. show databases - To check default database provided by Hive

b. create database:- To create a new database



c. Create table employee

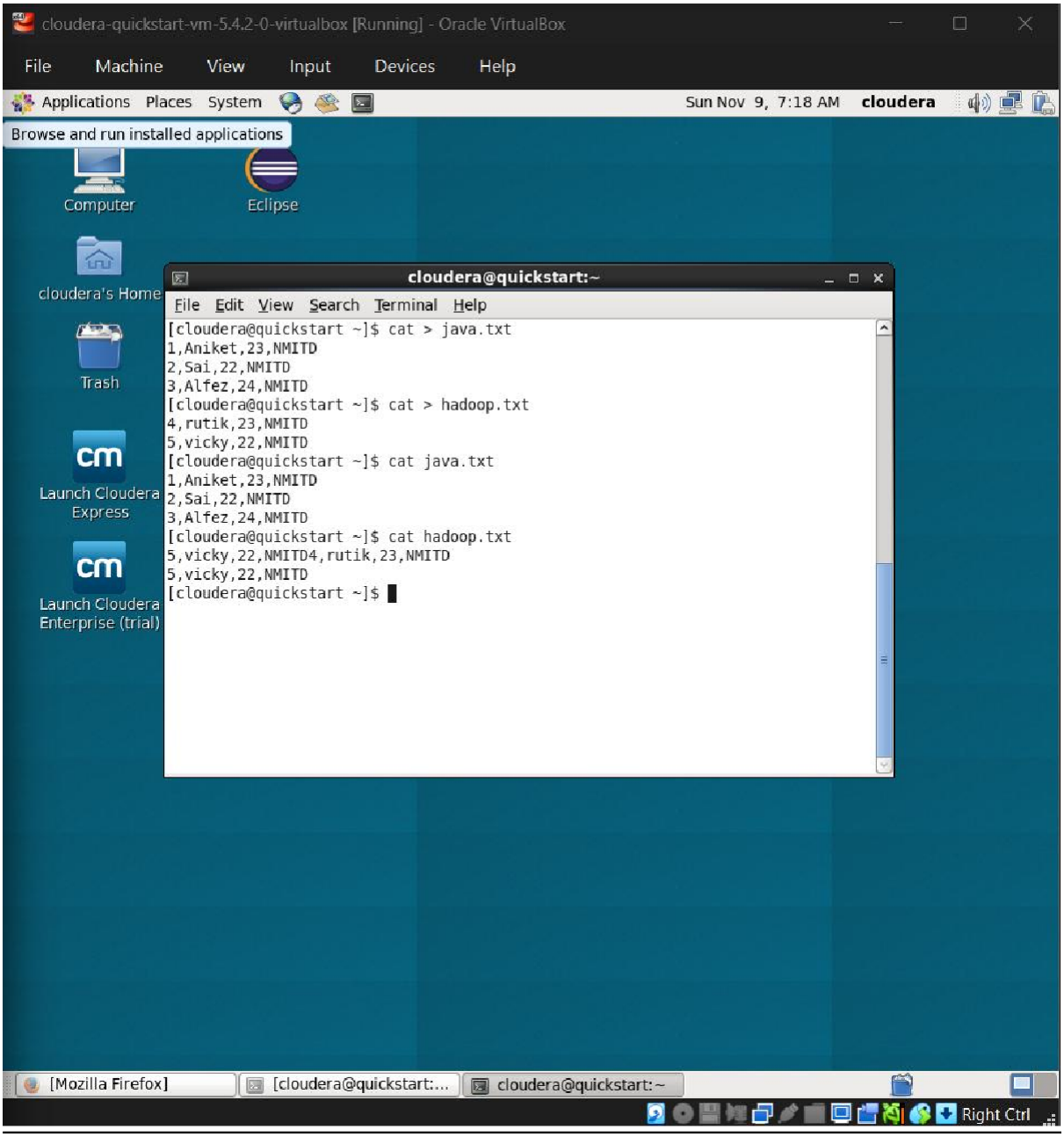
D. Describe the table employee



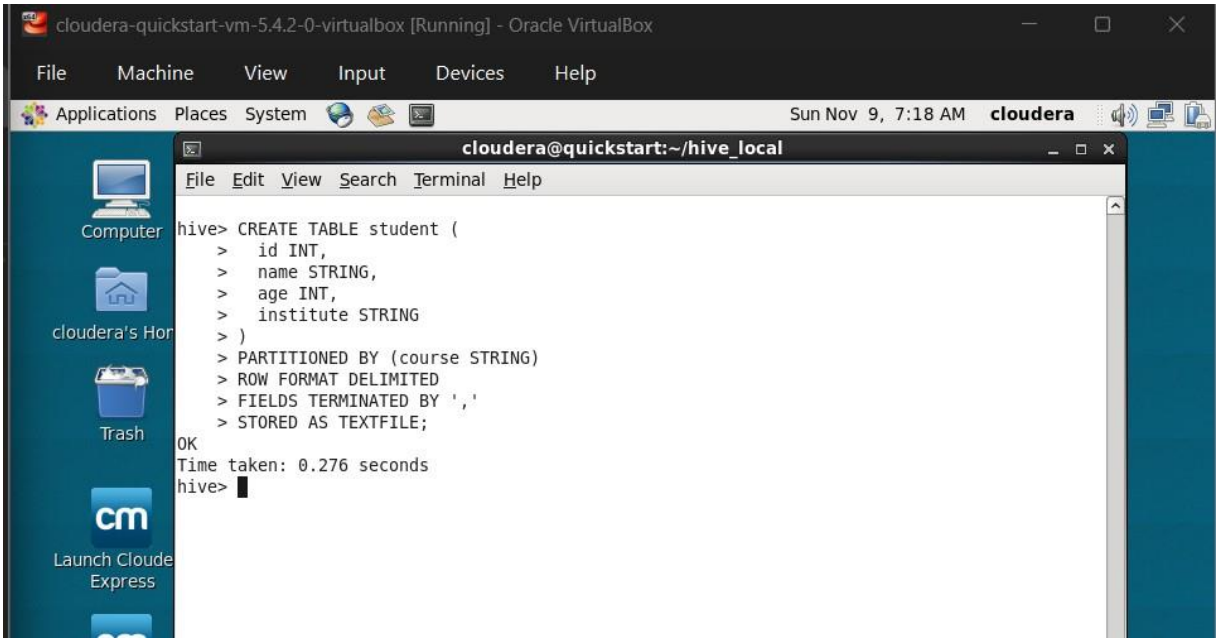
3. Hive Partitioning

A. Static Partition

To learn how to store Hive table data in **separate directory partitions** (for faster queries and organization).

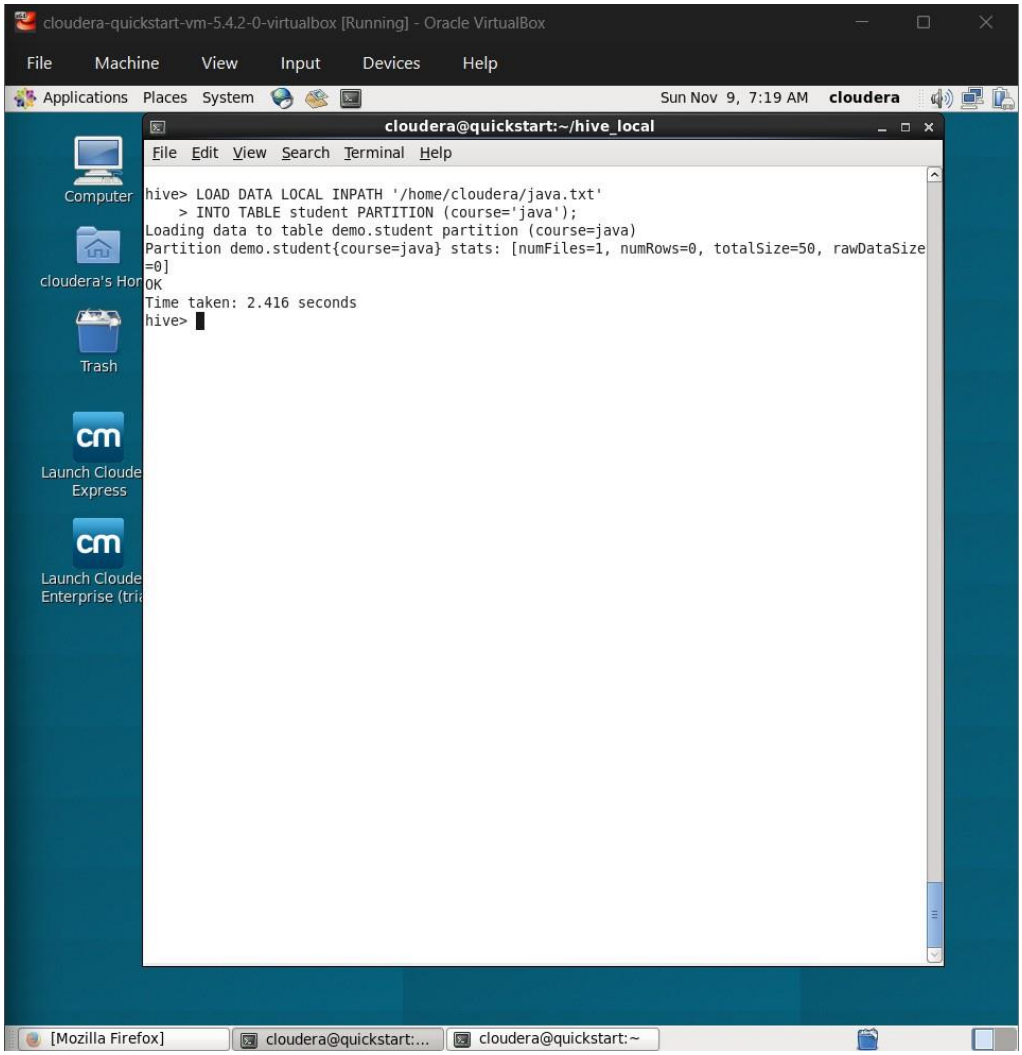


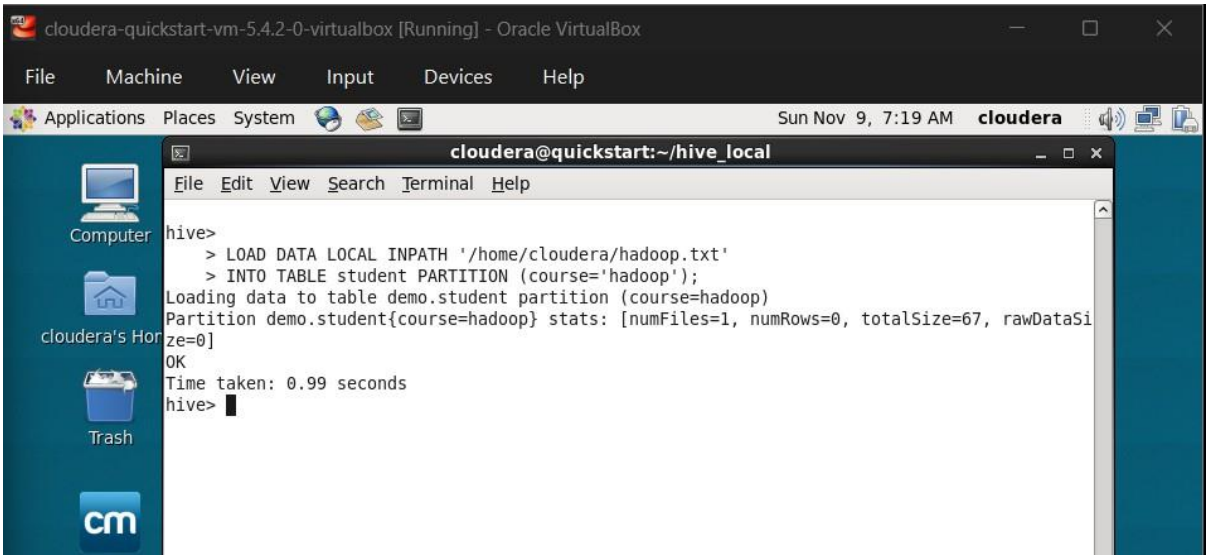
a. For this we are creating table student –



b. Load data statically into partitions

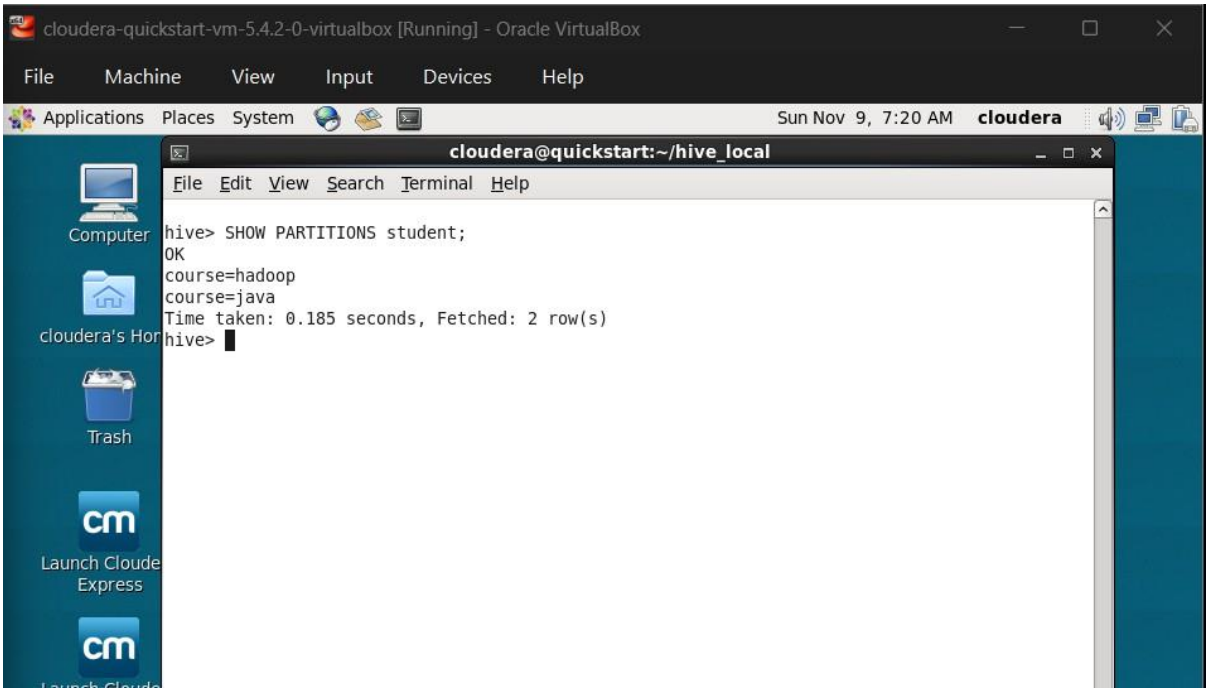
Now load each file as a separate partition:

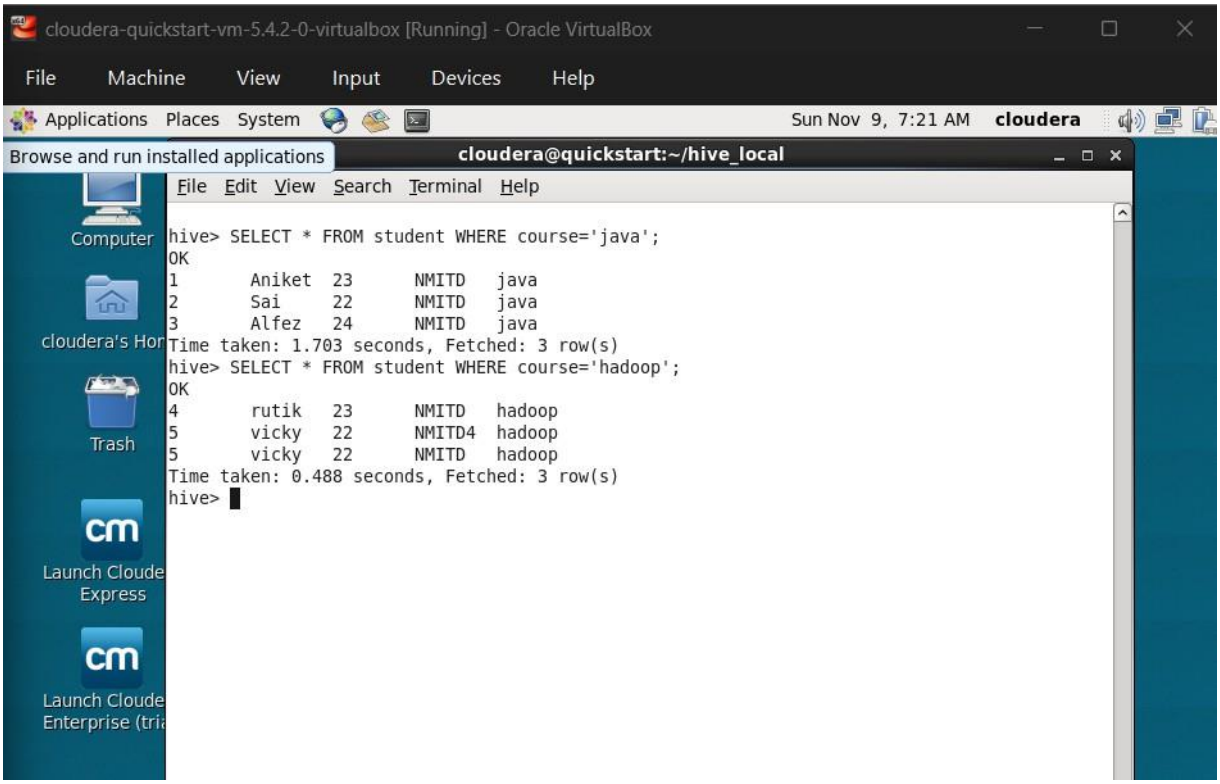




C. Verify partition structure

Check which partitions exist:



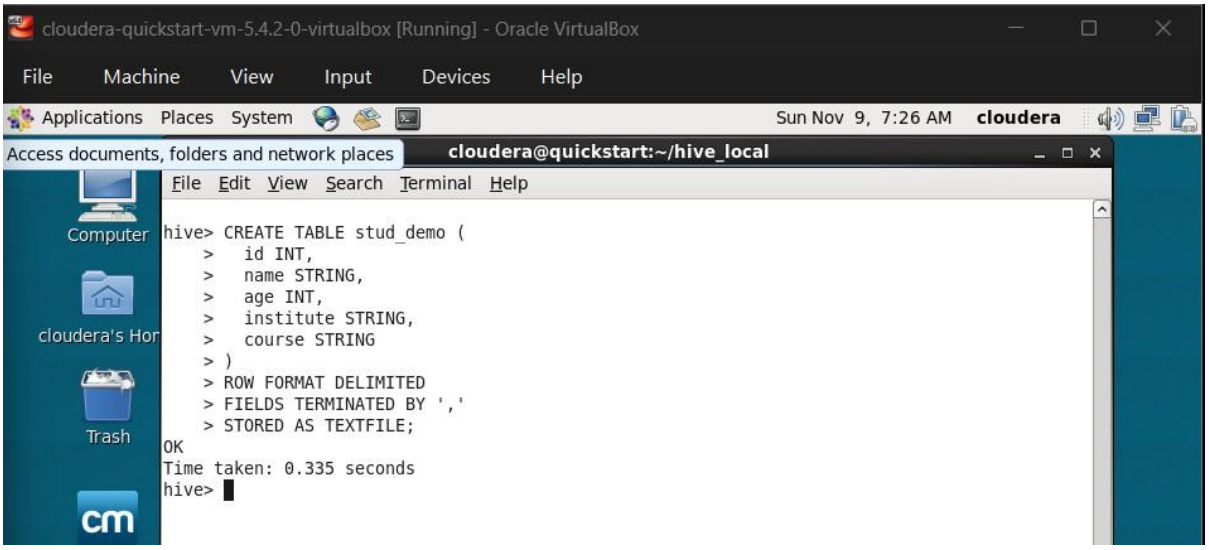


B. Dynamic Partitioning

To automatically create Hive table partitions while inserting data — instead of specifying each partition value manually.

a. Create the Staging Table

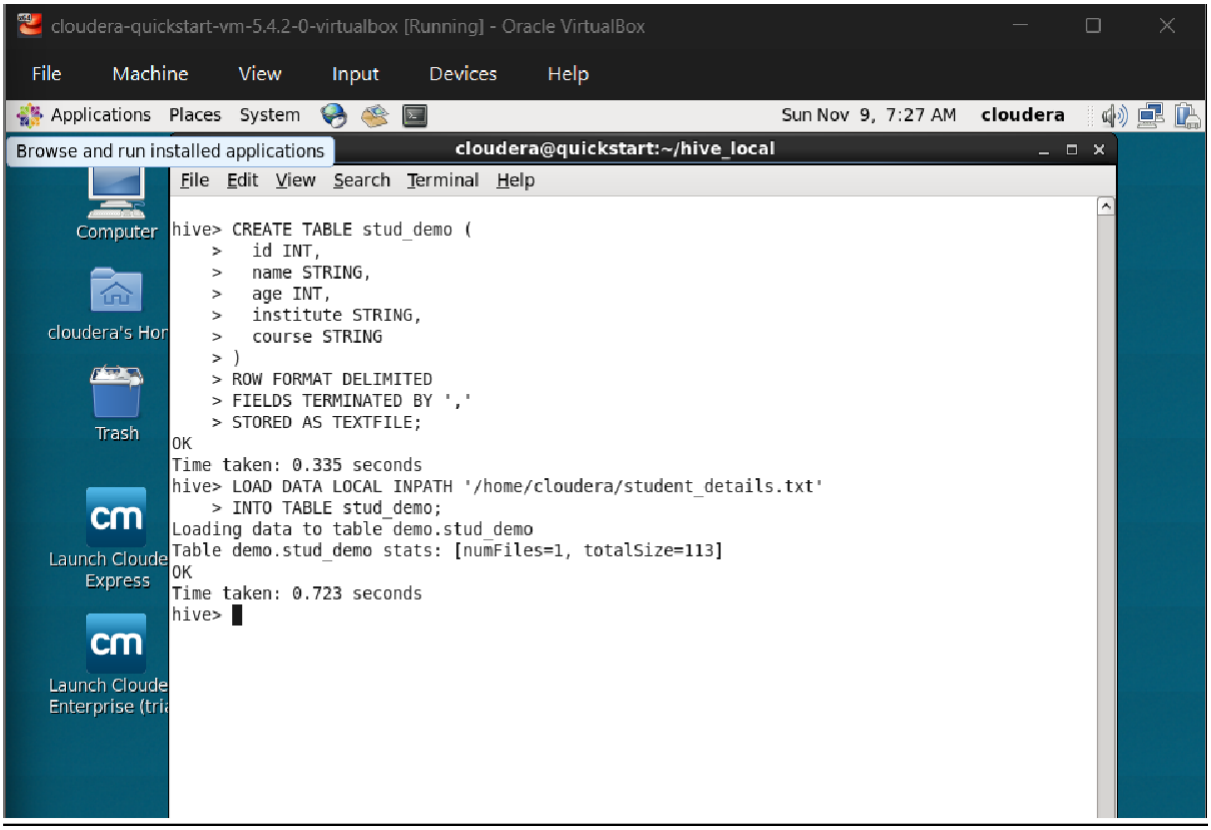
We'll first create a *non-partitioned* staging table (stud_demo) to hold all data (both Java and Hadoop students).



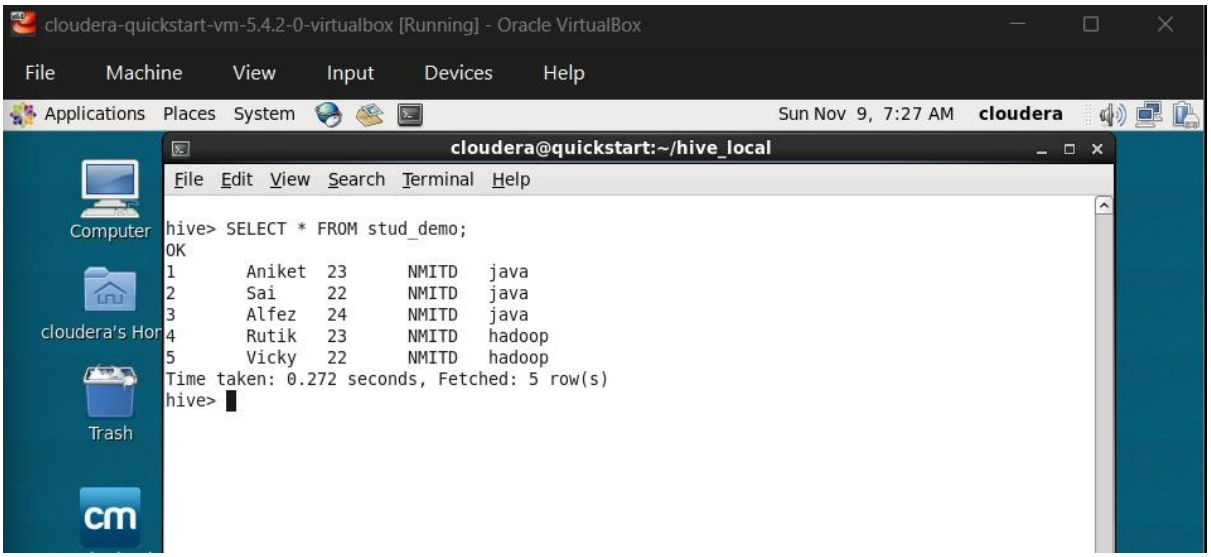
b. Prepare Combined Input File

In your **Linux terminal**, create one combined file that includes both course values:

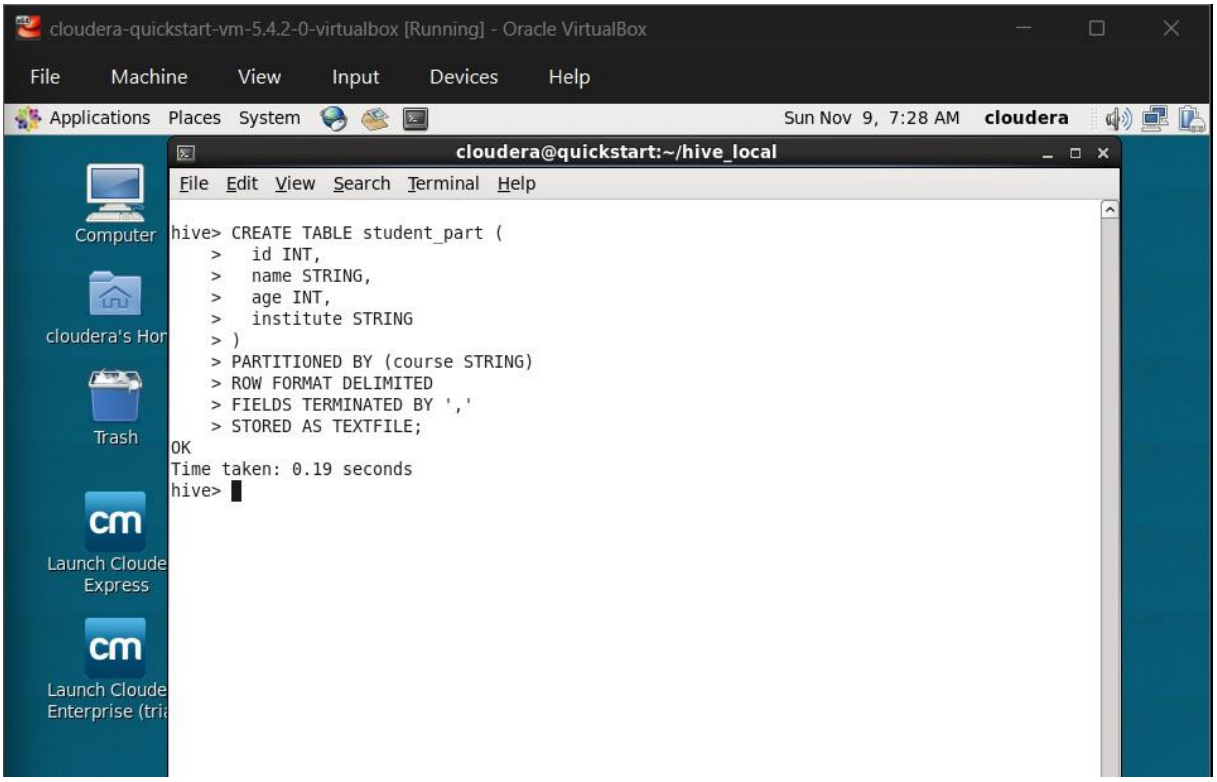
C. Load Data into the Staging Table



D. Show full stud_demo table :

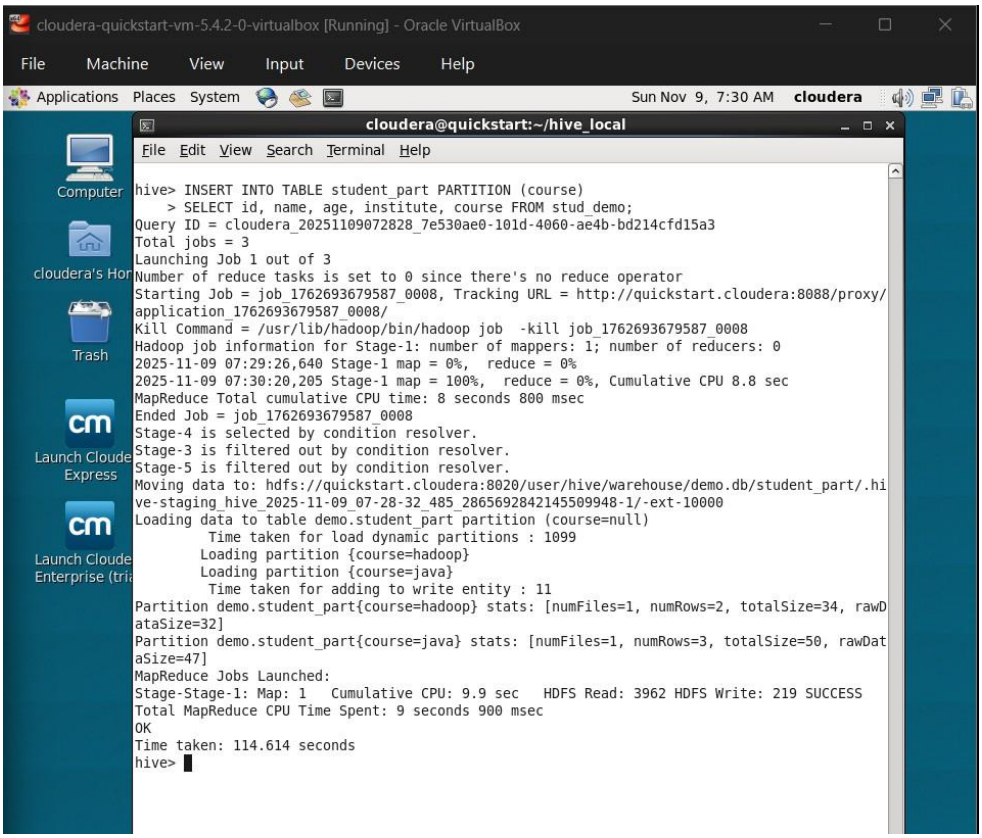


E. Create the Partitioned Target Table



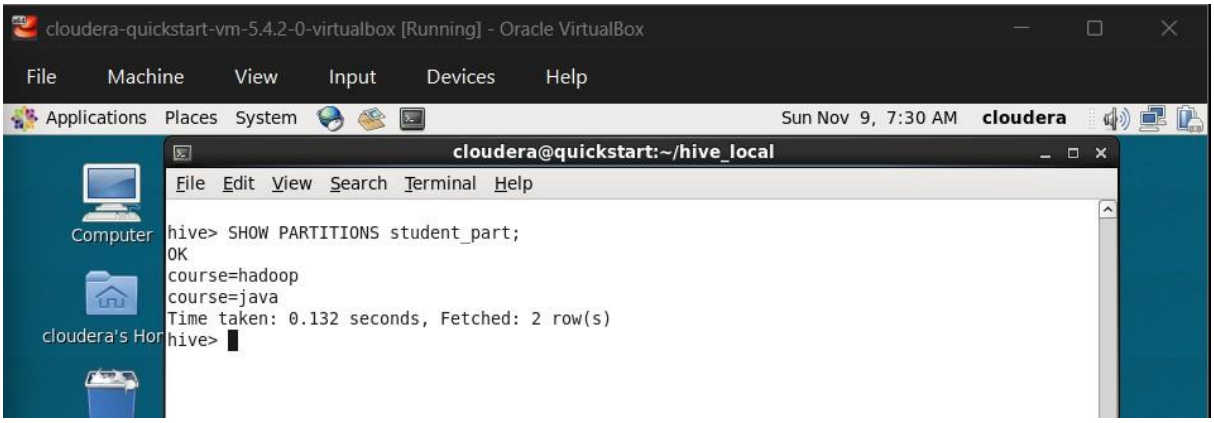
F. Insert Data Dynamically

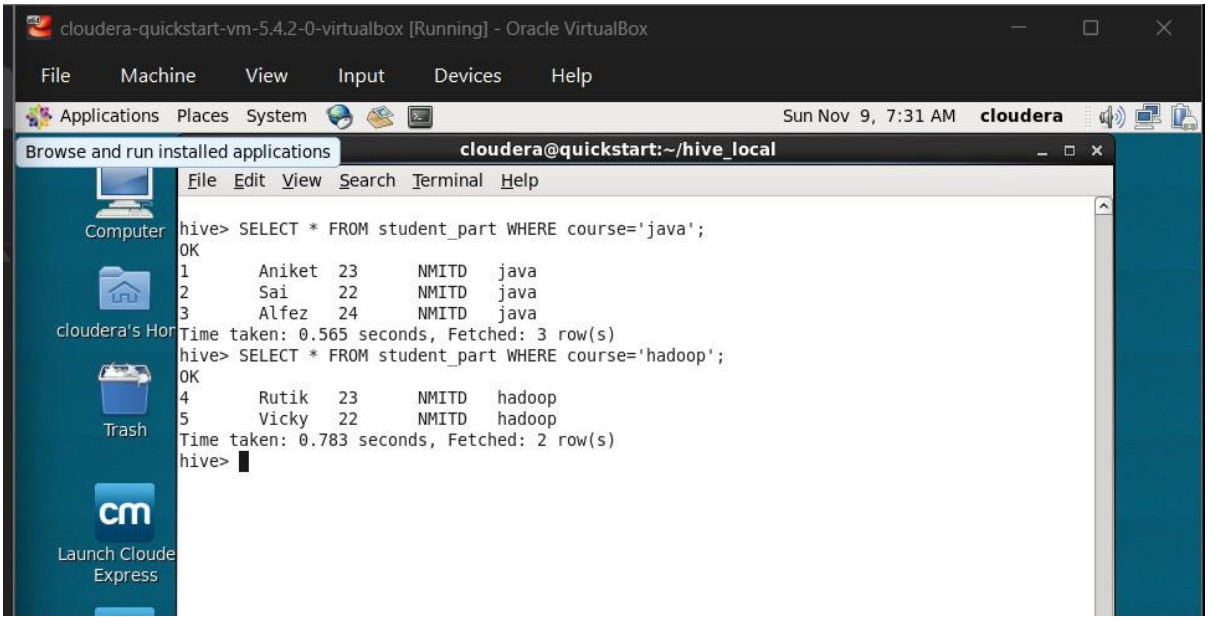
Now execute the dynamic insert statement.

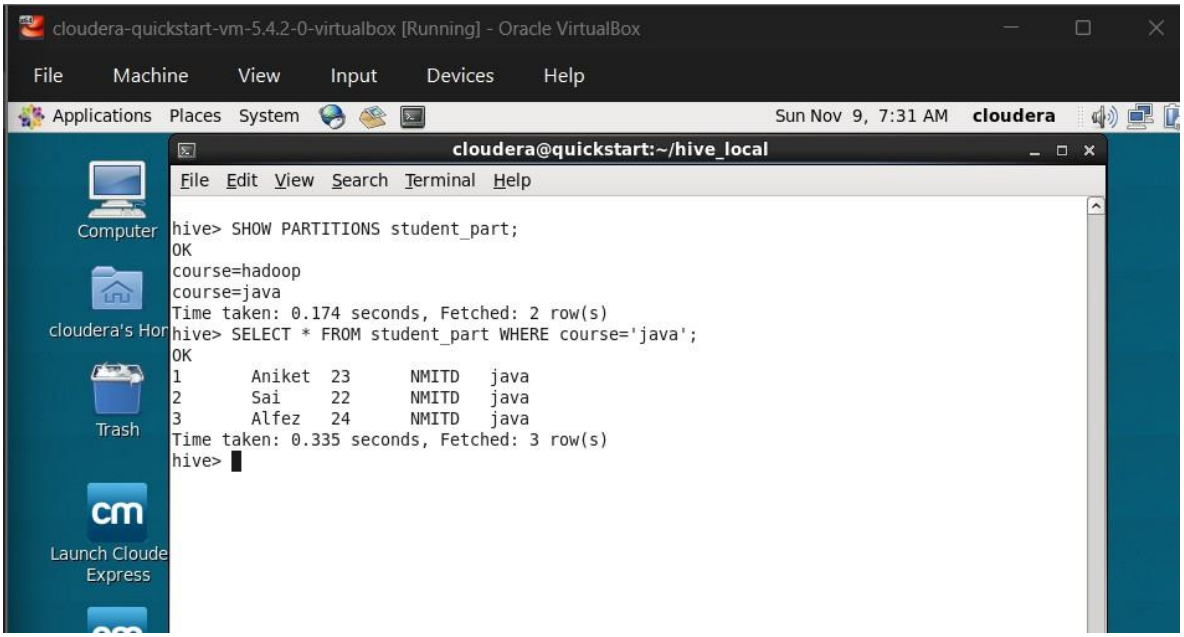


G. Verify Dynamic Partitions

List partitions.





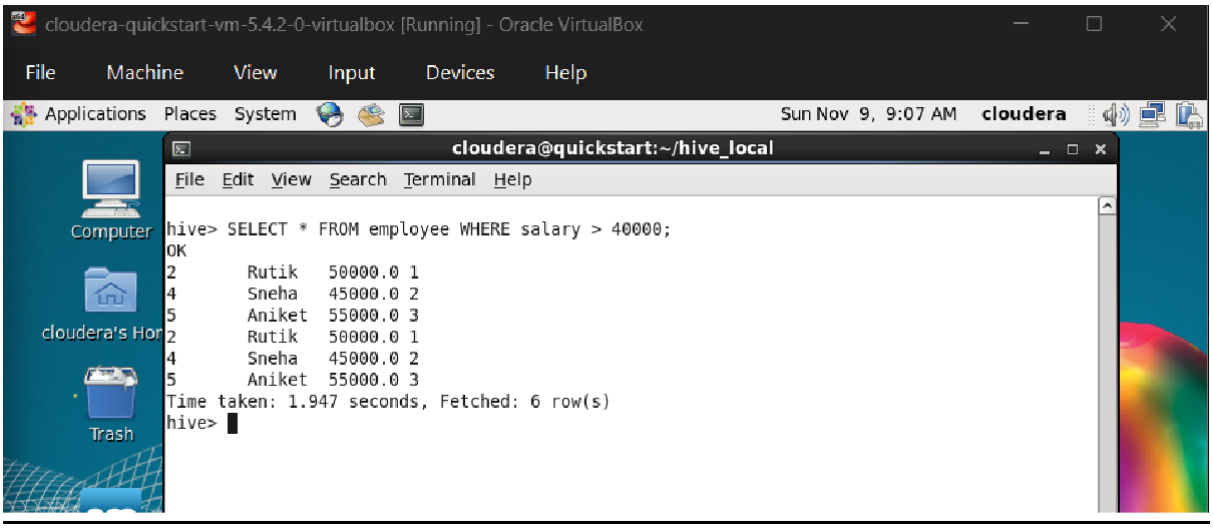


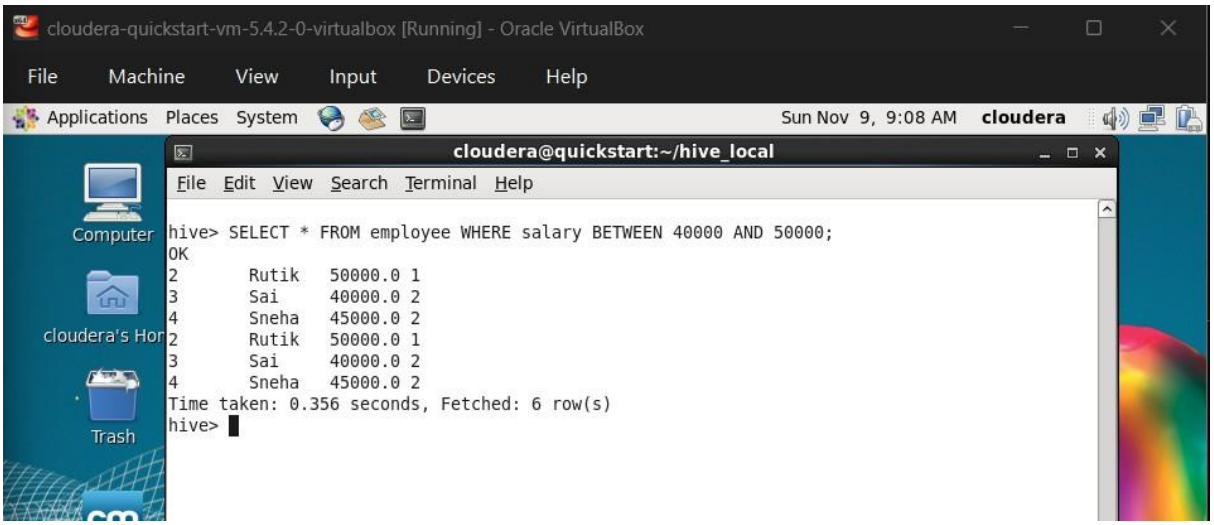
4. Hive Built-In Operators

To demonstrate the use of **Hive's built-in operators** for comparison, arithmetic, and logical operations on table data.

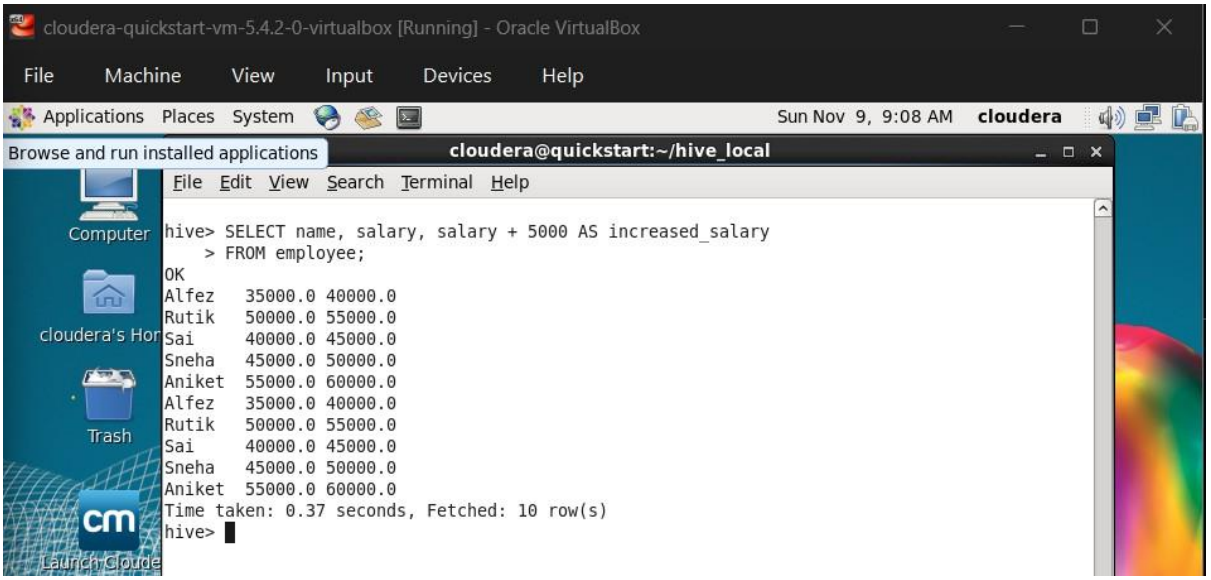
A. Comparison Operators

We'll use your existing 'employee' table for this.

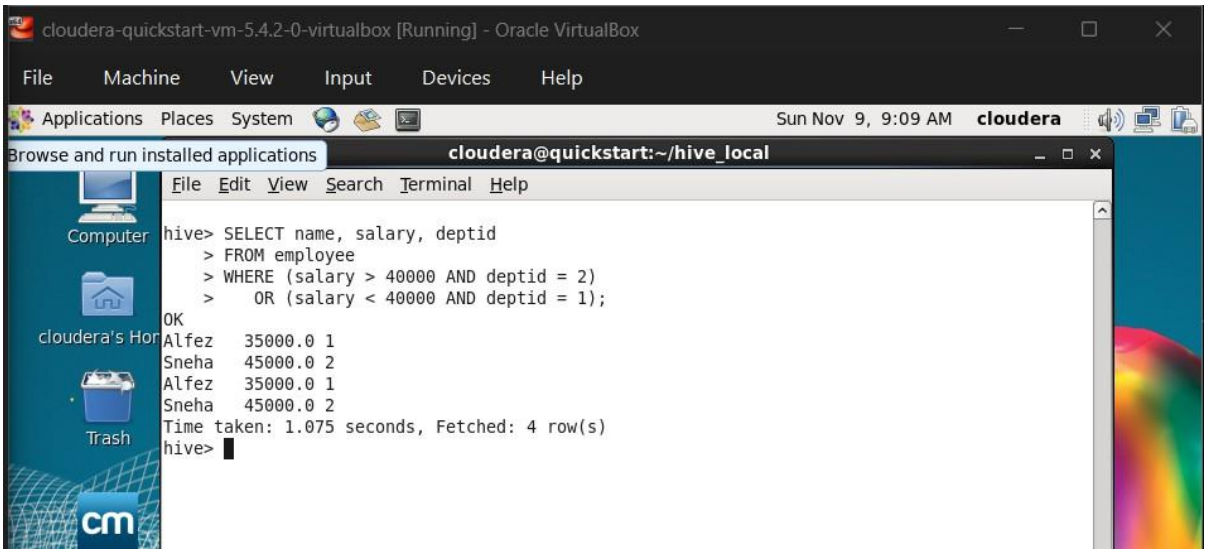




B. Arithmetic Operators

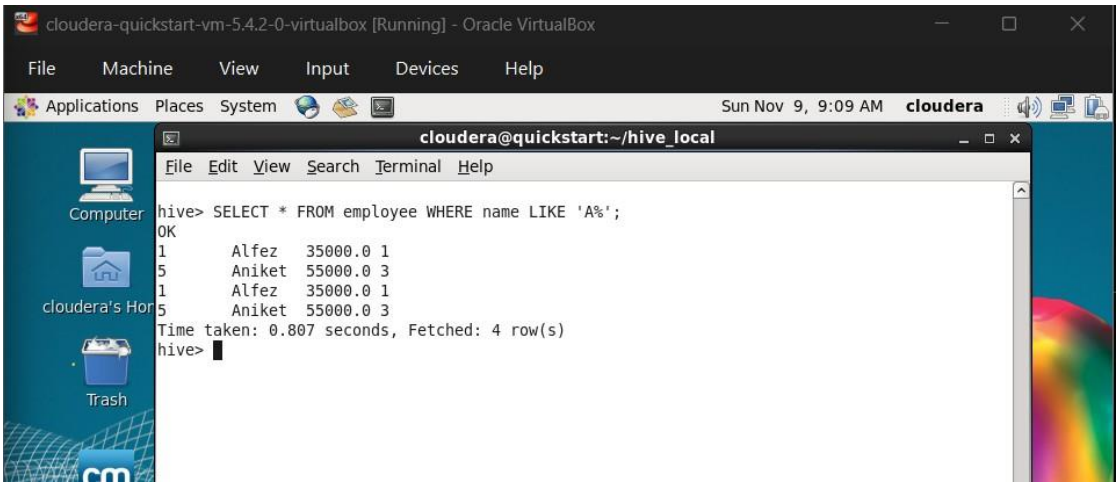


C. Logical Operators

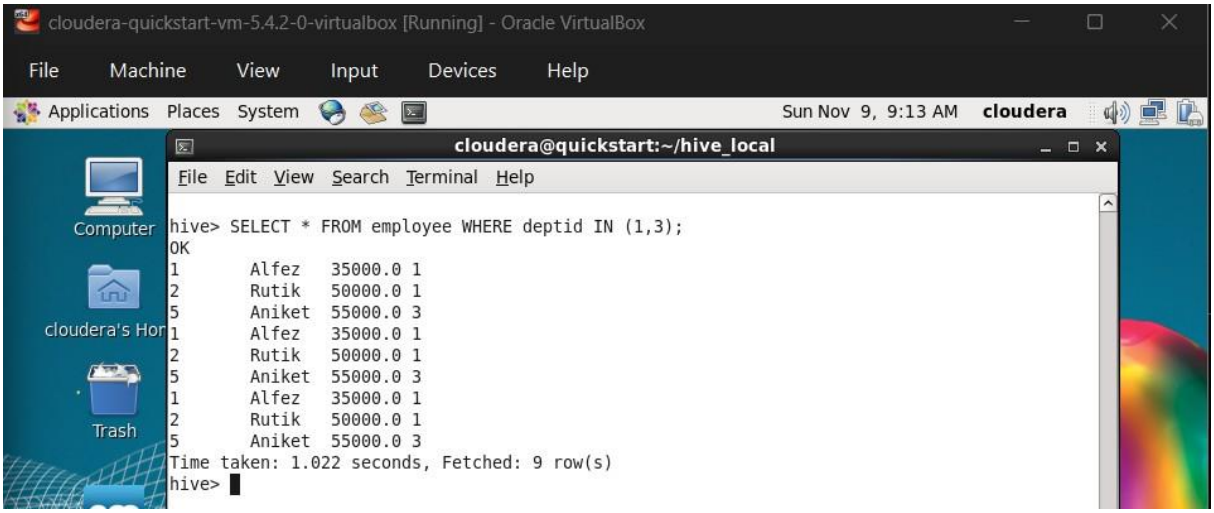


D. Special Operator Examples

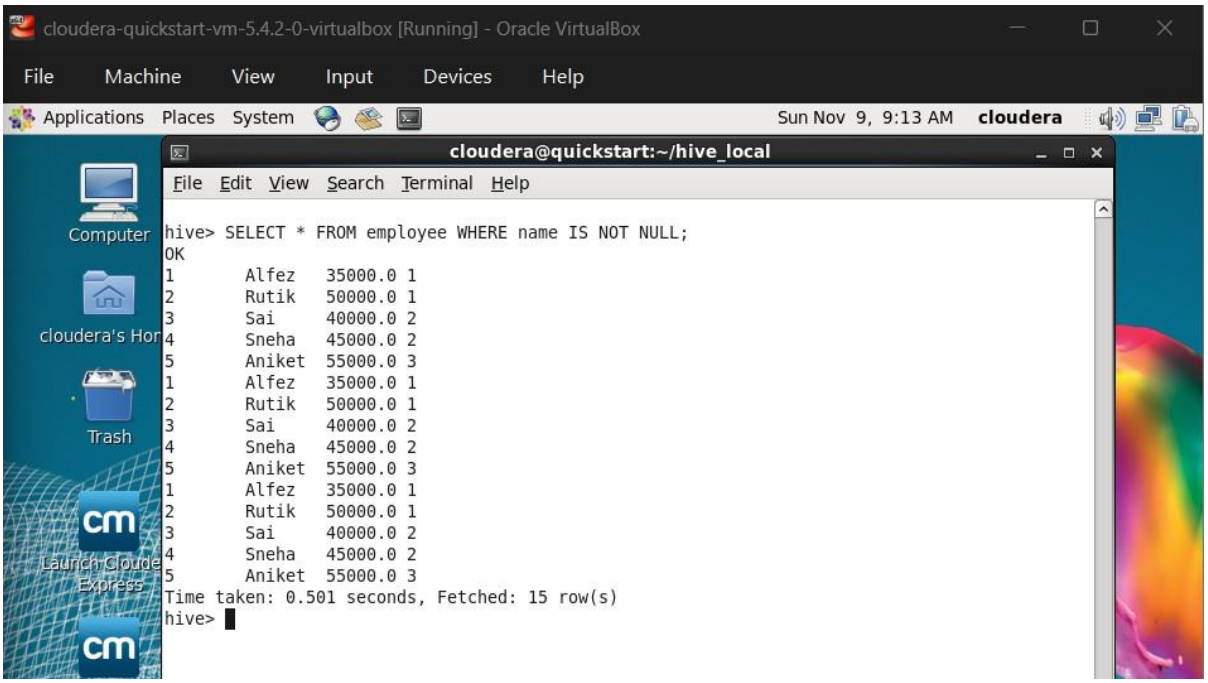
(a) LIKE — pattern matching



(b) IN / NOT IN



(c) IS NULL / IS NOT NULL

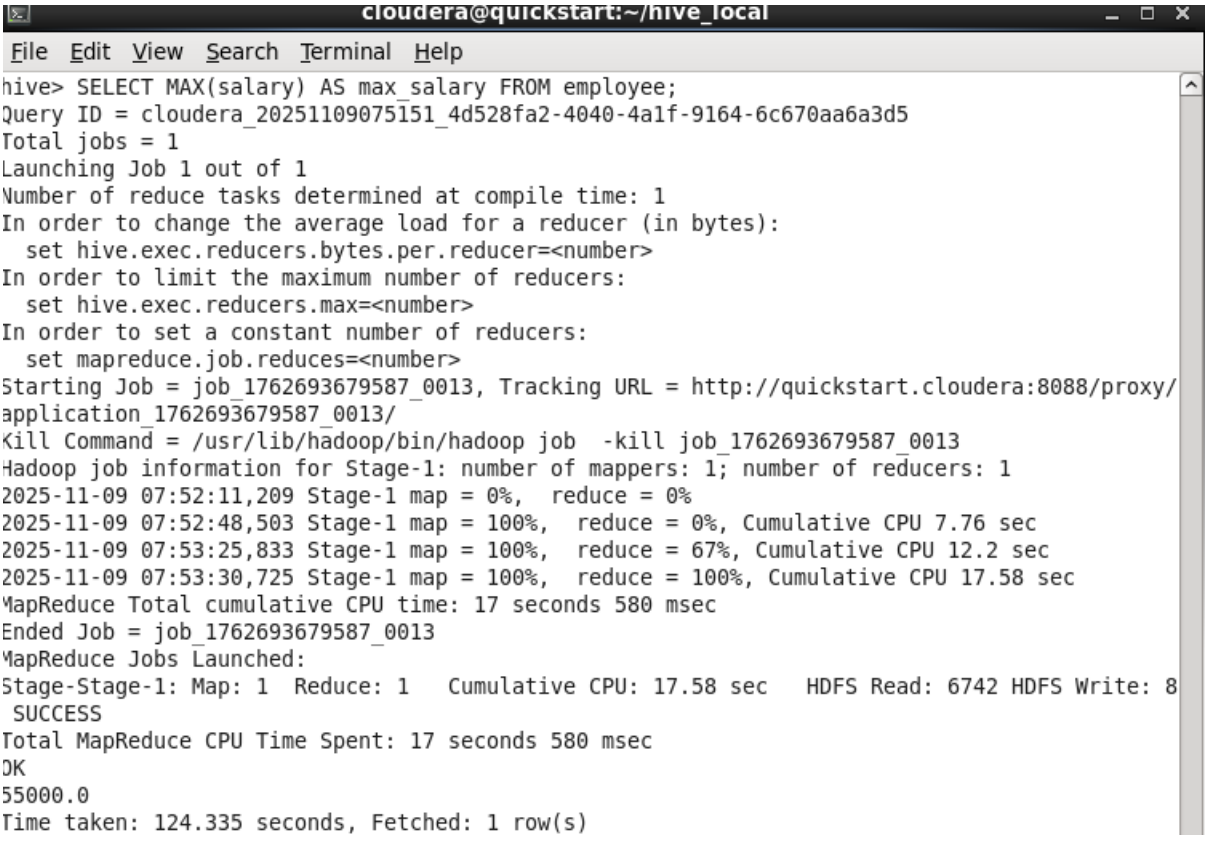


5. Hive Built-In Functions

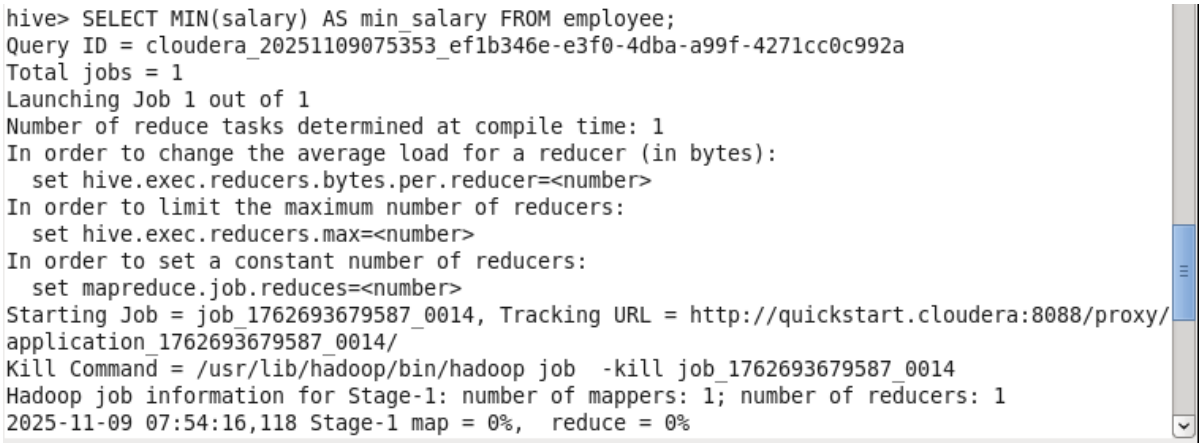
To demonstrate the use of Hive built-in functions — both aggregate functions (like MAX, MIN, AVG) and string manipulation functions (like UPPER, LOWER, LENGTH, SUBSTR).

A. Aggregate Functions

SELECT MAX(salary) AS max_salary FROM employee;



B. SELECT MIN(salary) AS min_salary FROM employee;

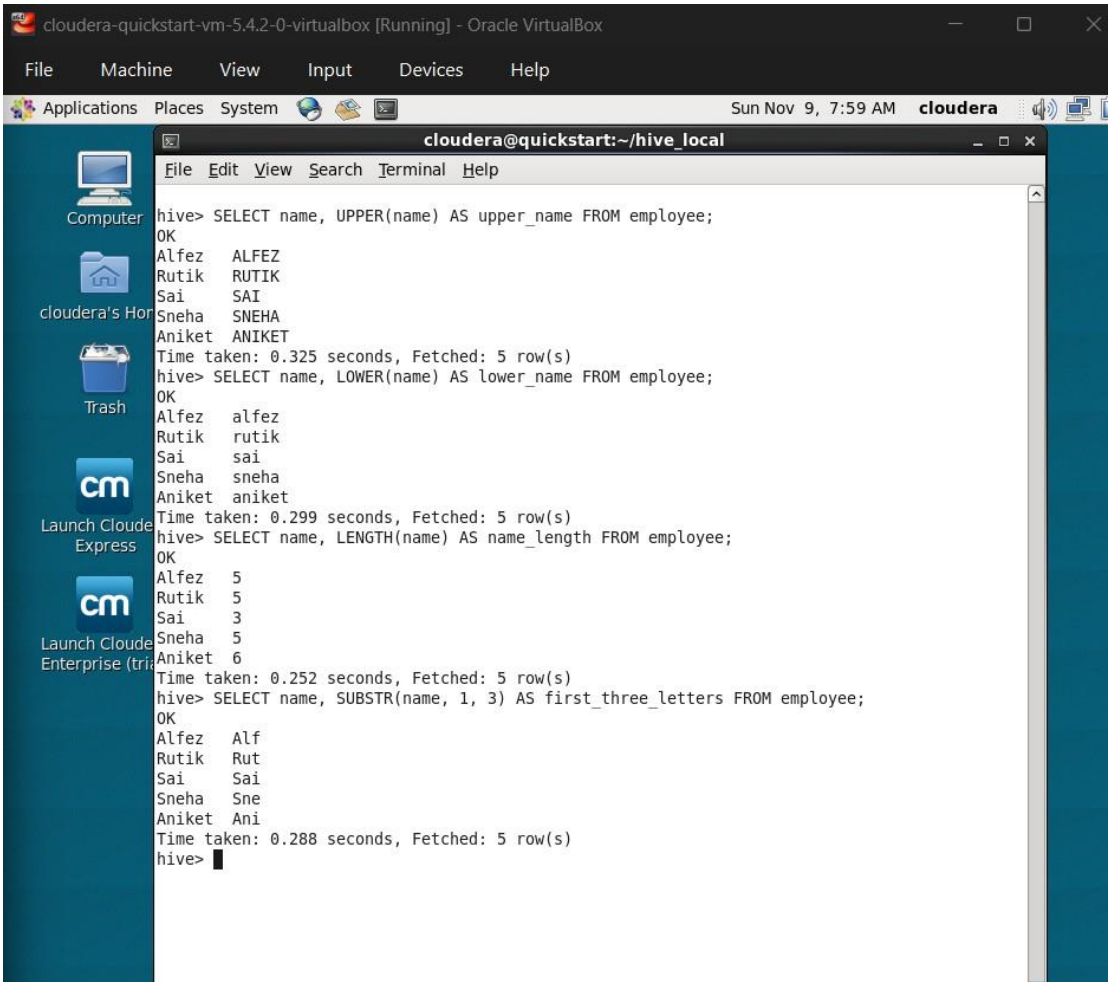


C. SELECT AVG(salary) AS avg_salary FROM employee;

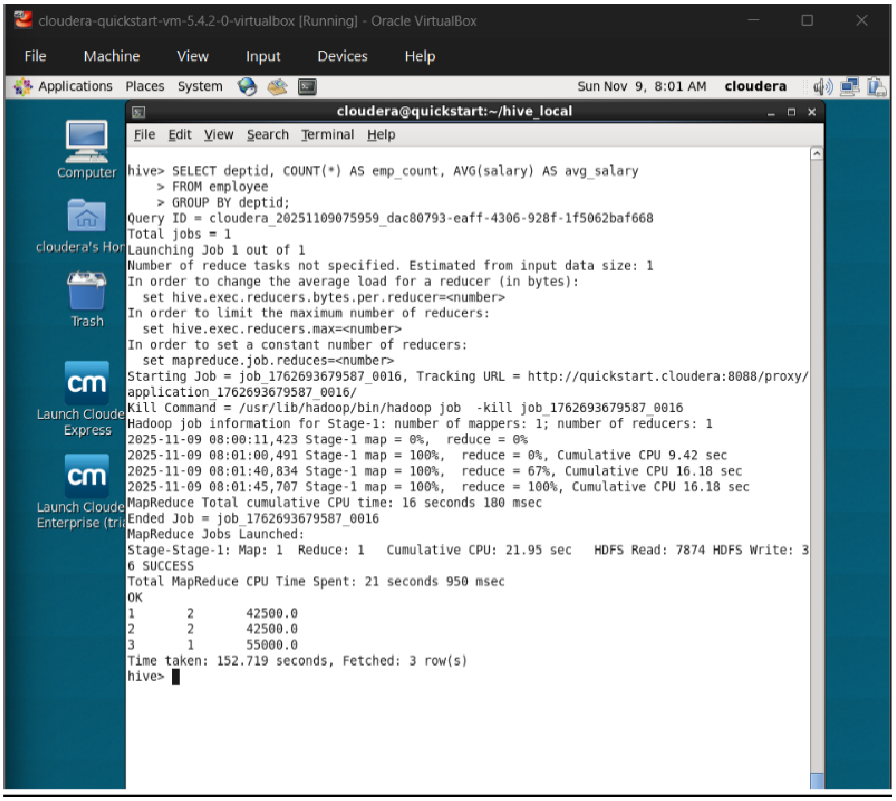
```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 18.62 sec   HDFS Read: 6749 HDFS Write: 8
 SUCCESS
Total MapReduce CPU Time Spent: 18 seconds 620 msec
OK
35000.0
Time taken: 116.092 seconds, Fetched: 1 row(s)
hive> SELECT AVG(salary) AS avg_salary FROM employee;
Query ID = cloudera_20251109075555_c907aad7-f1b2-408a-9429-efbd991e2eb4
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1762693679587_0015, Tracking URL = http://quickstart.cloudera:8088/proxy/
application_1762693679587_0015/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1762693679587_0015
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2025-11-09 07:56:23,417 Stage-1 map = 0%,  reduce = 0%
2025-11-09 07:57:01,849 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 7.94 sec
2025-11-09 07:57:33,105 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 13.68 sec
2025-11-09 07:57:37,676 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 18.06 sec
MapReduce Total cumulative CPU time: 18 seconds 60 msec
Ended Job = job_1762693679587_0015
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 18.06 sec   HDFS Read: 7116 HDFS Write: 8
 SUCCESS
Total MapReduce CPU Time Spent: 18 seconds 60 msec
OK
45000.0
Time taken: 128.181 seconds, Fetched: 1 row(s)
hive> S
```

B. String Functions

Hive provides many string manipulation utilities.
Let's use your existing employee table to test them:



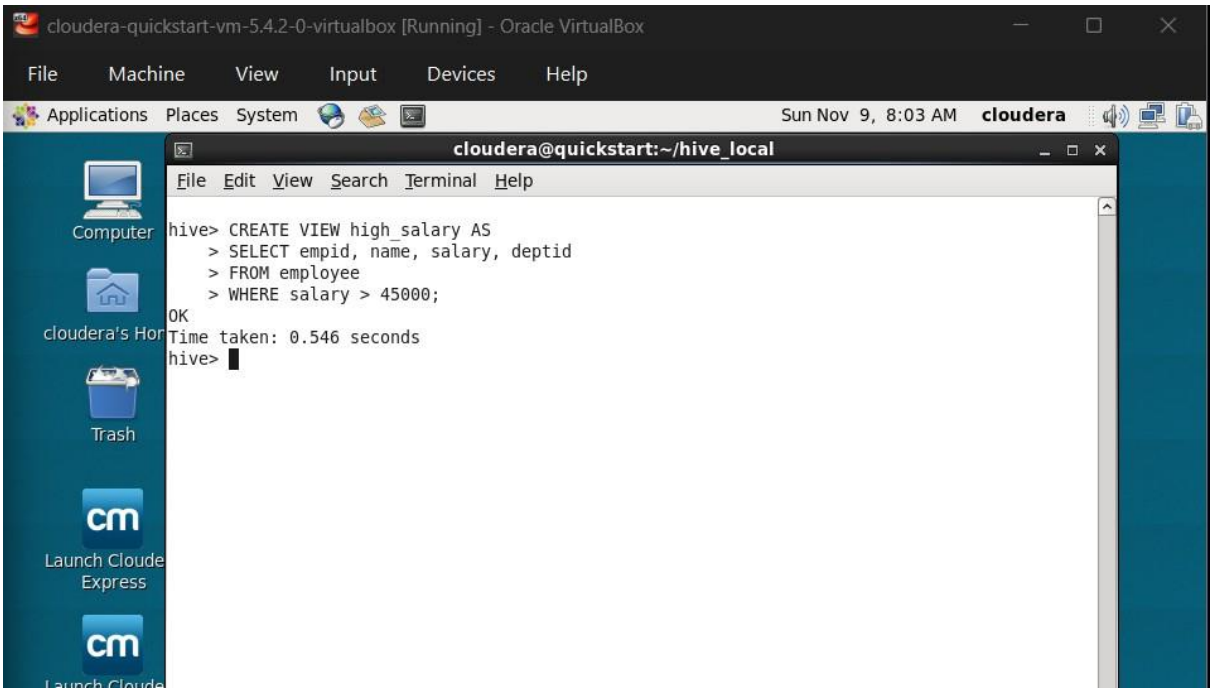
C. Combining Aggregate & Group



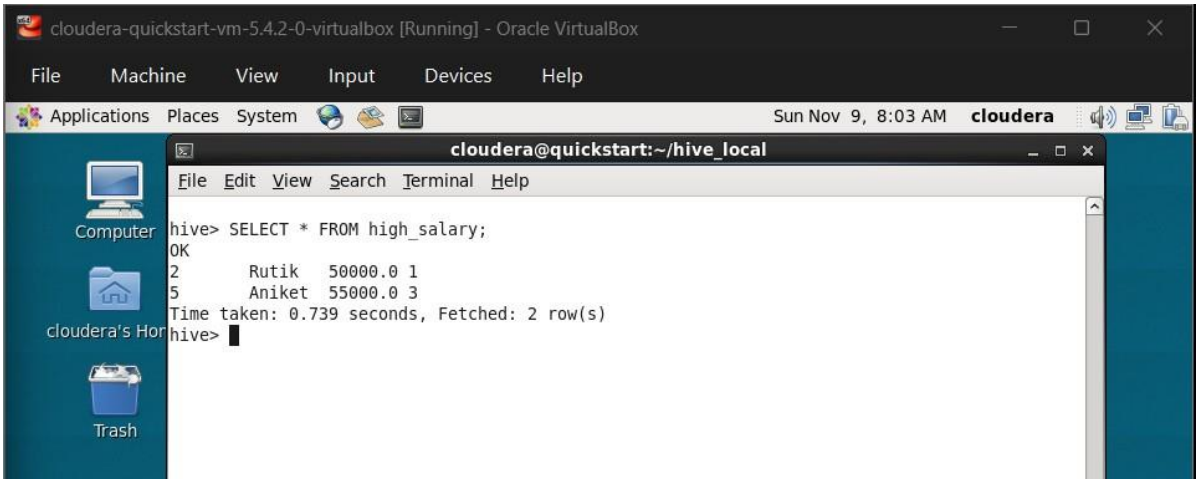
6. Hive Views

Create and query views (virtual tables) & Understand indexes in Hive (used for query optimization, optional based on VM support).

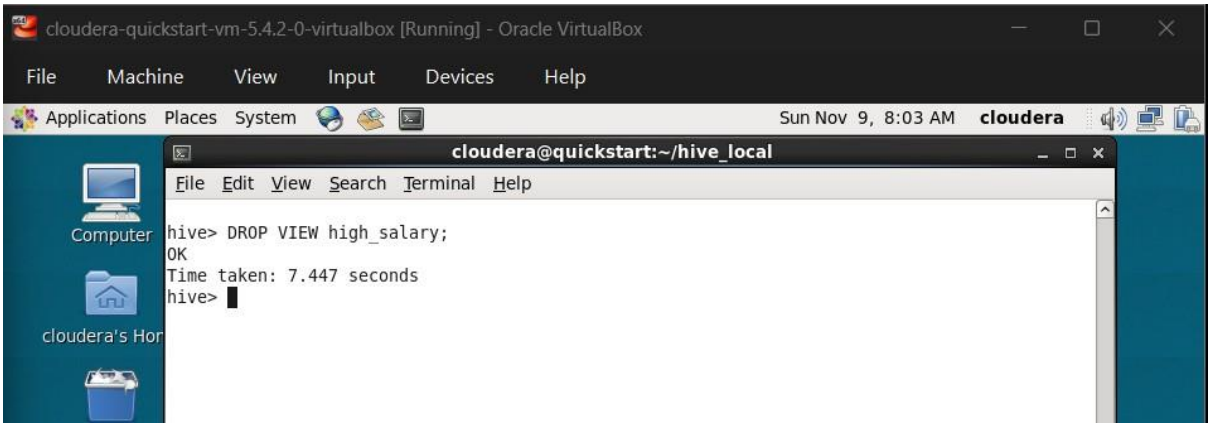
A. Create a View



B. Query the view

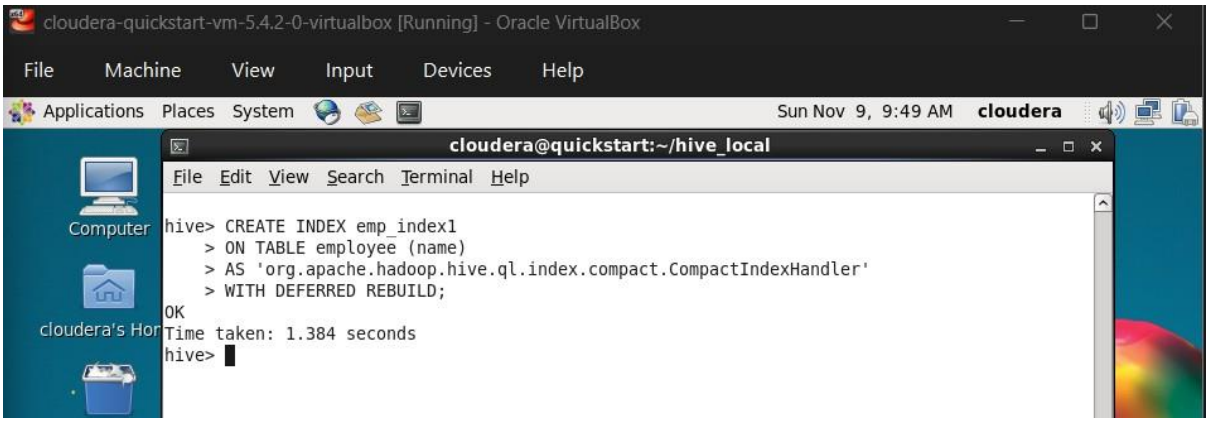


C. Drop the View

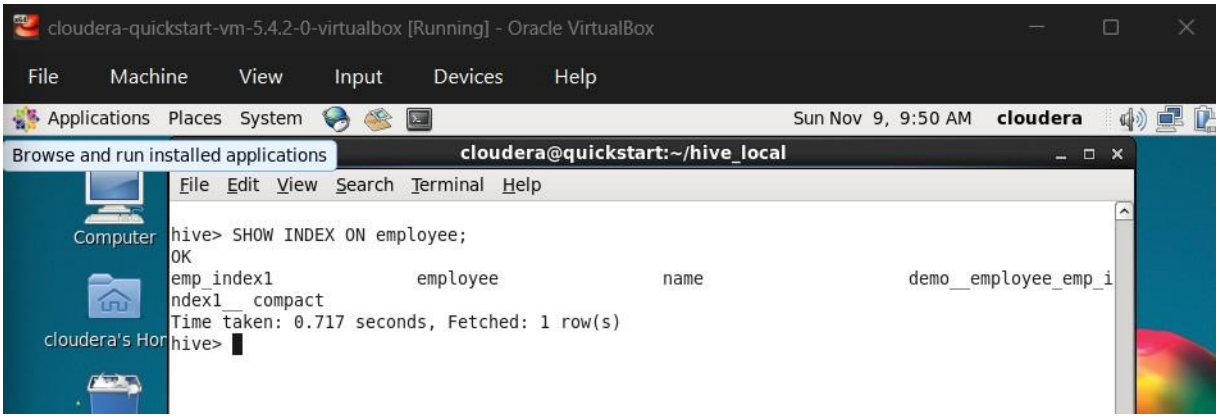


D. INDEXES

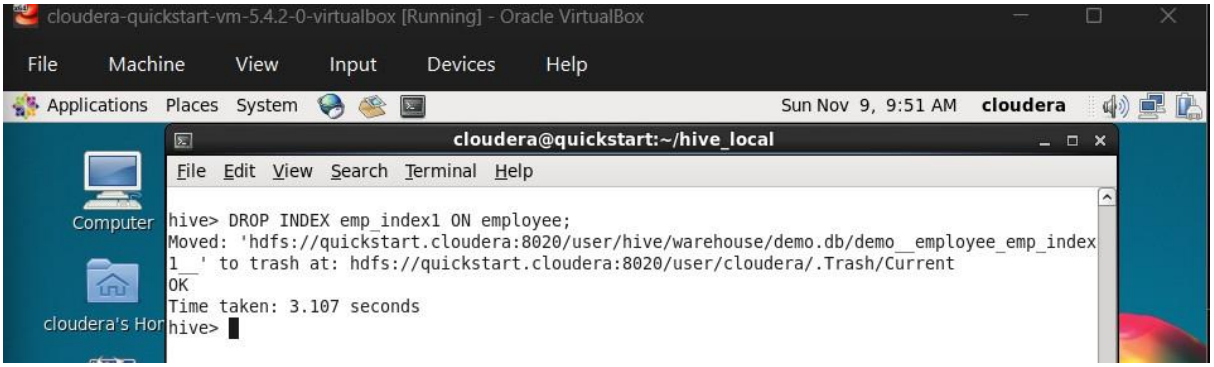
Create an Index - This command creates a metadata index on the name column — intended to speed up queries that filter by name.
The WITH DEFERRED REBUILD part tells Hive to create the index structure later.



Show Existing Indexes



Drop the Index

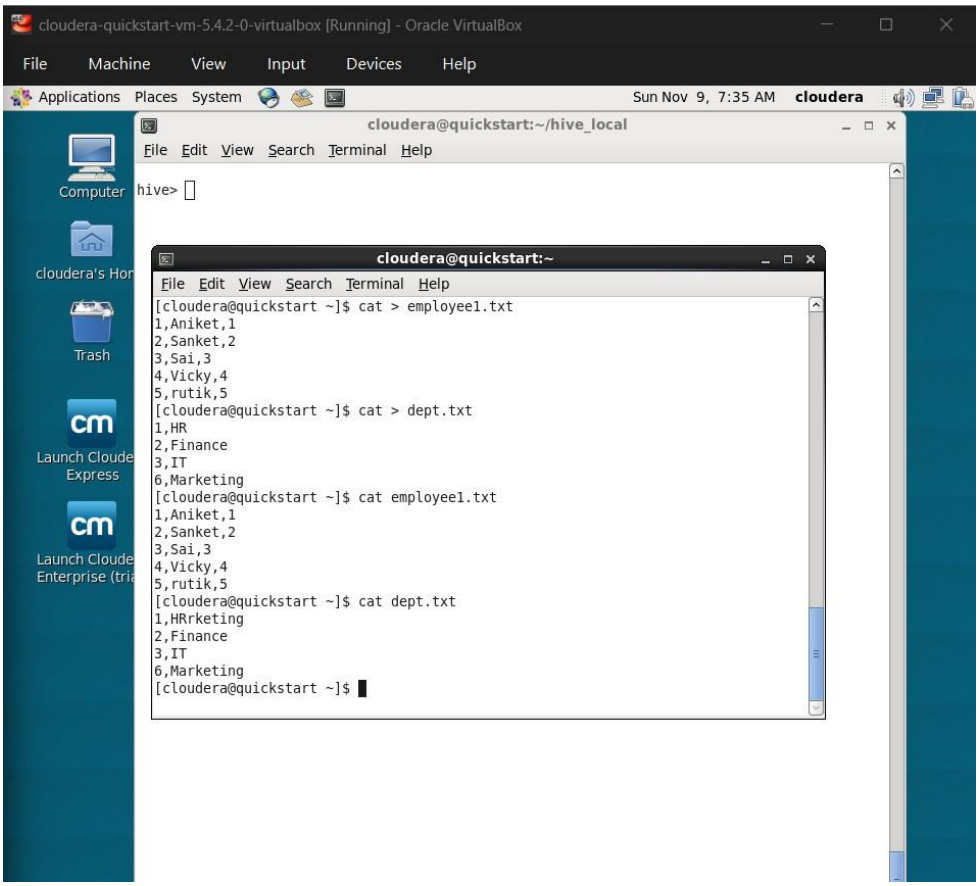


---

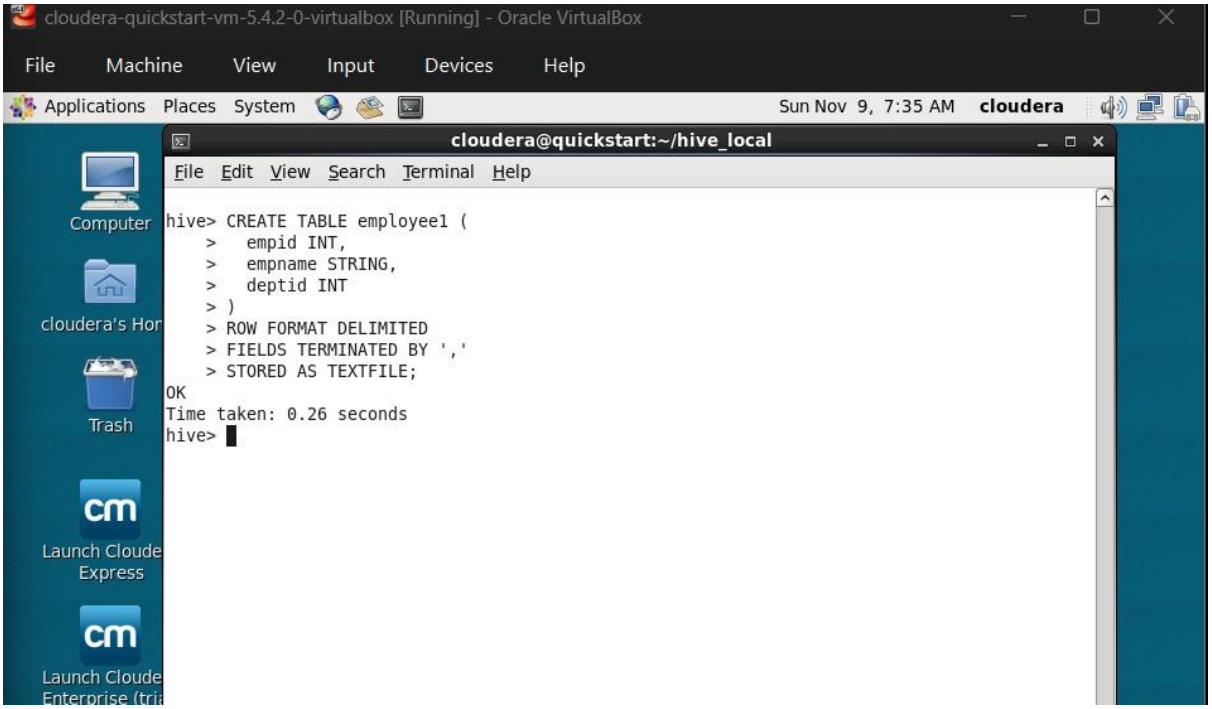
7. HiveQL: Select Where, Select OrderBy, Select GroupBy, Select Joins

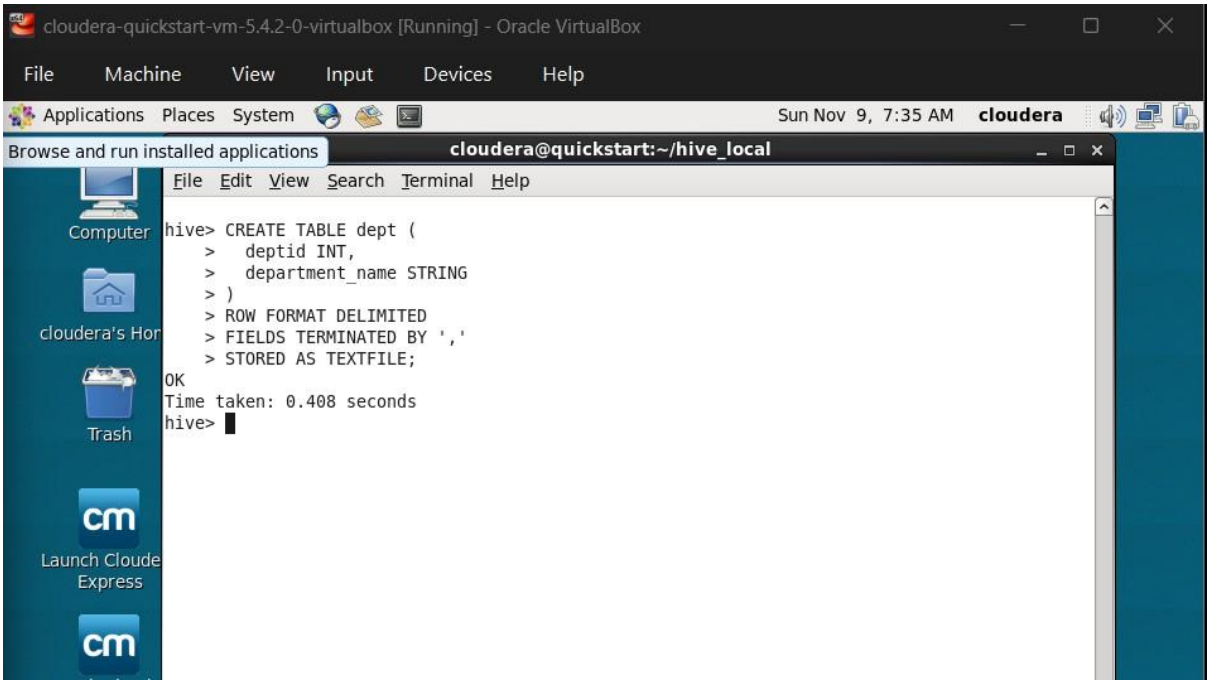
A. Create Sample Input Files

In your **Linux terminal**, create two CSV files:

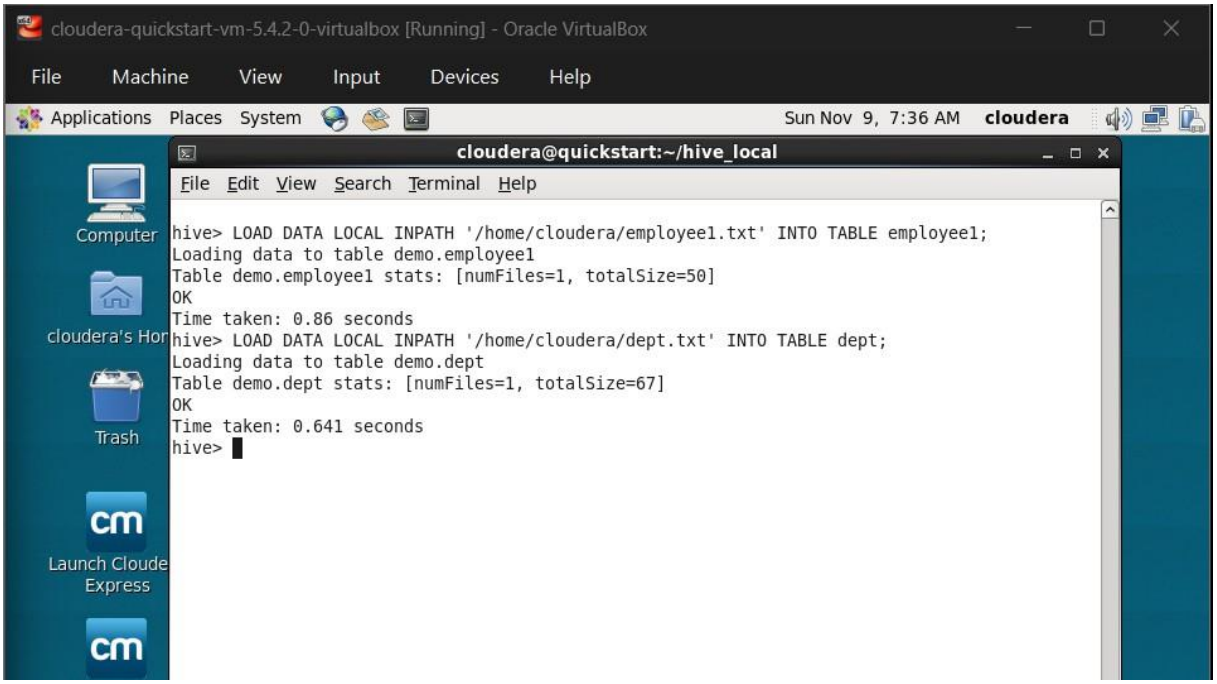


B. Create the Tables in Hive





C. Load Data into the Tables

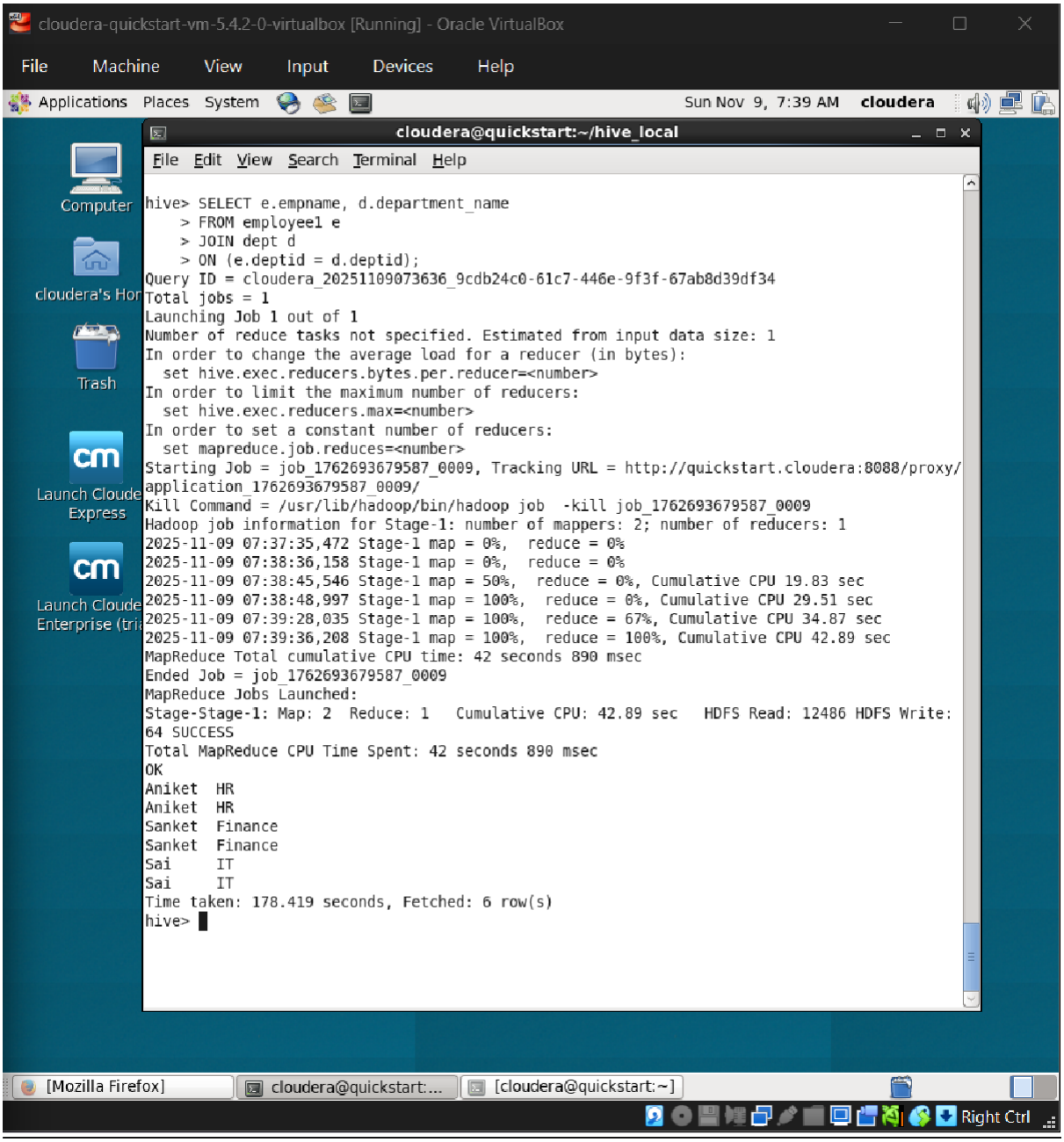


D. Disable Auto Join Optimization

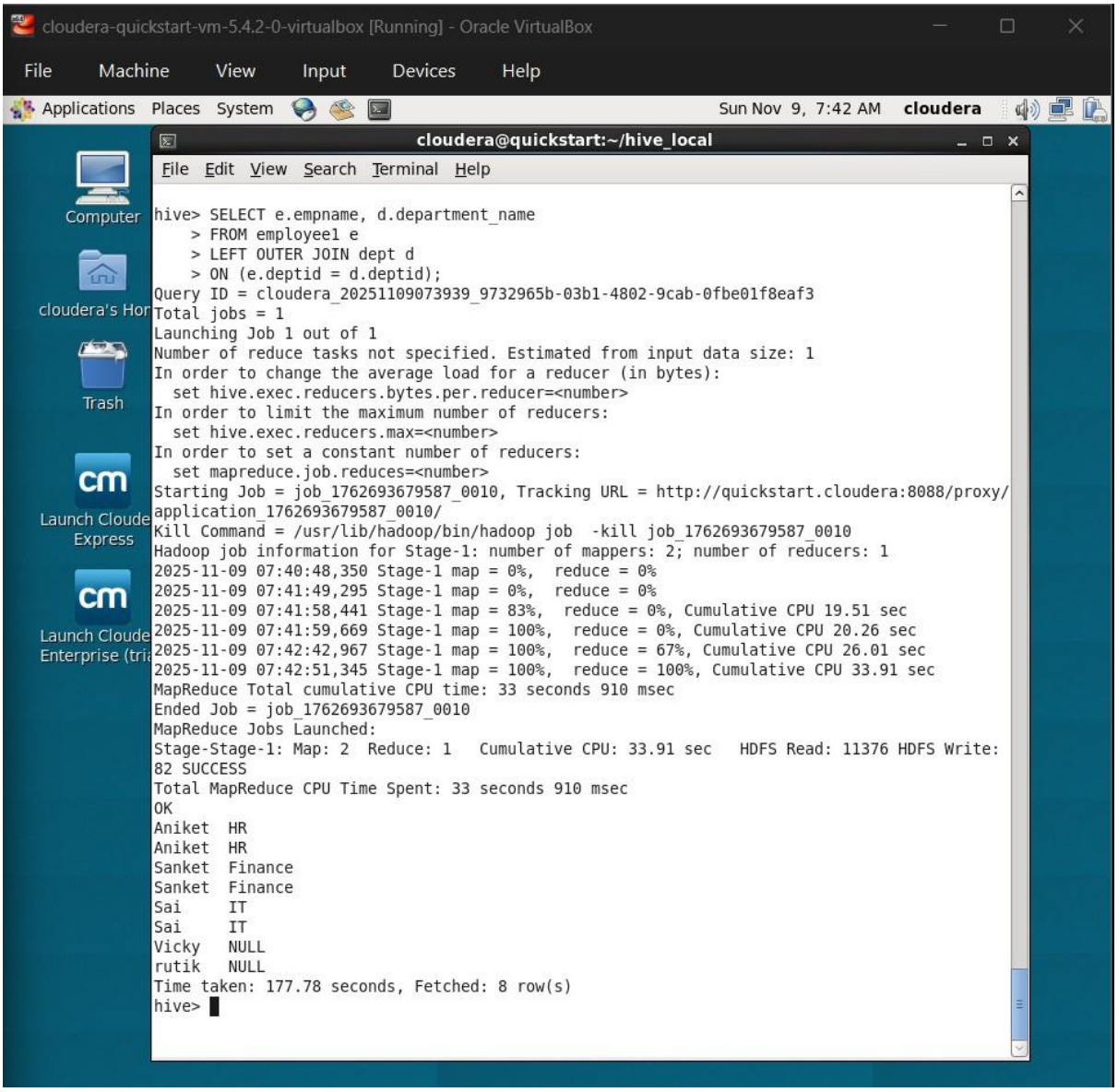
This prevents Hive from converting your join into a map-side broadcast join, which can skip reducers for small tables.
Disabling it ensures you see a MapReduce job plan.

hive.auto.convert.join=false

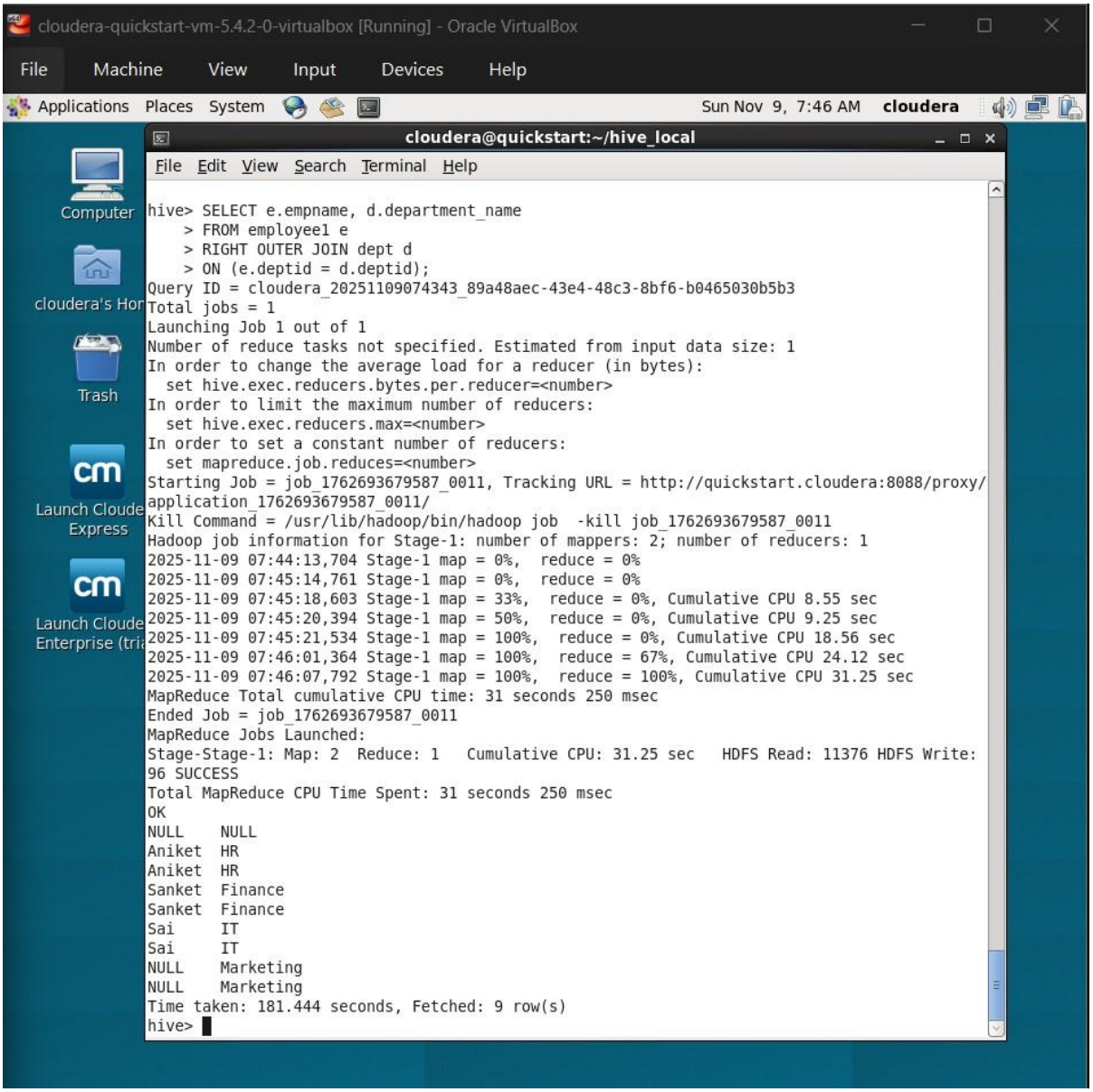
E. INNER JOIN



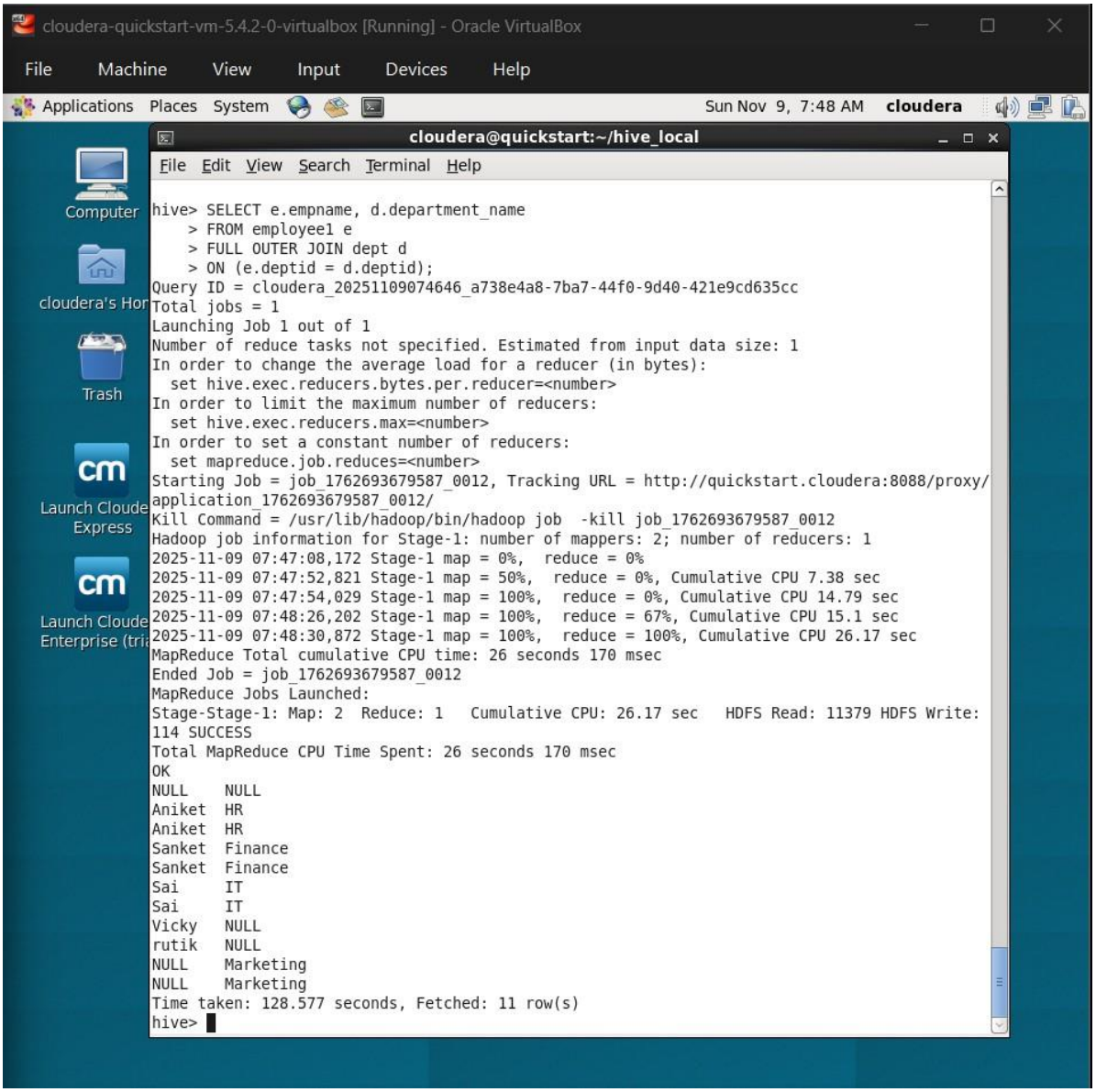
F. LEFT OUTER JOIN

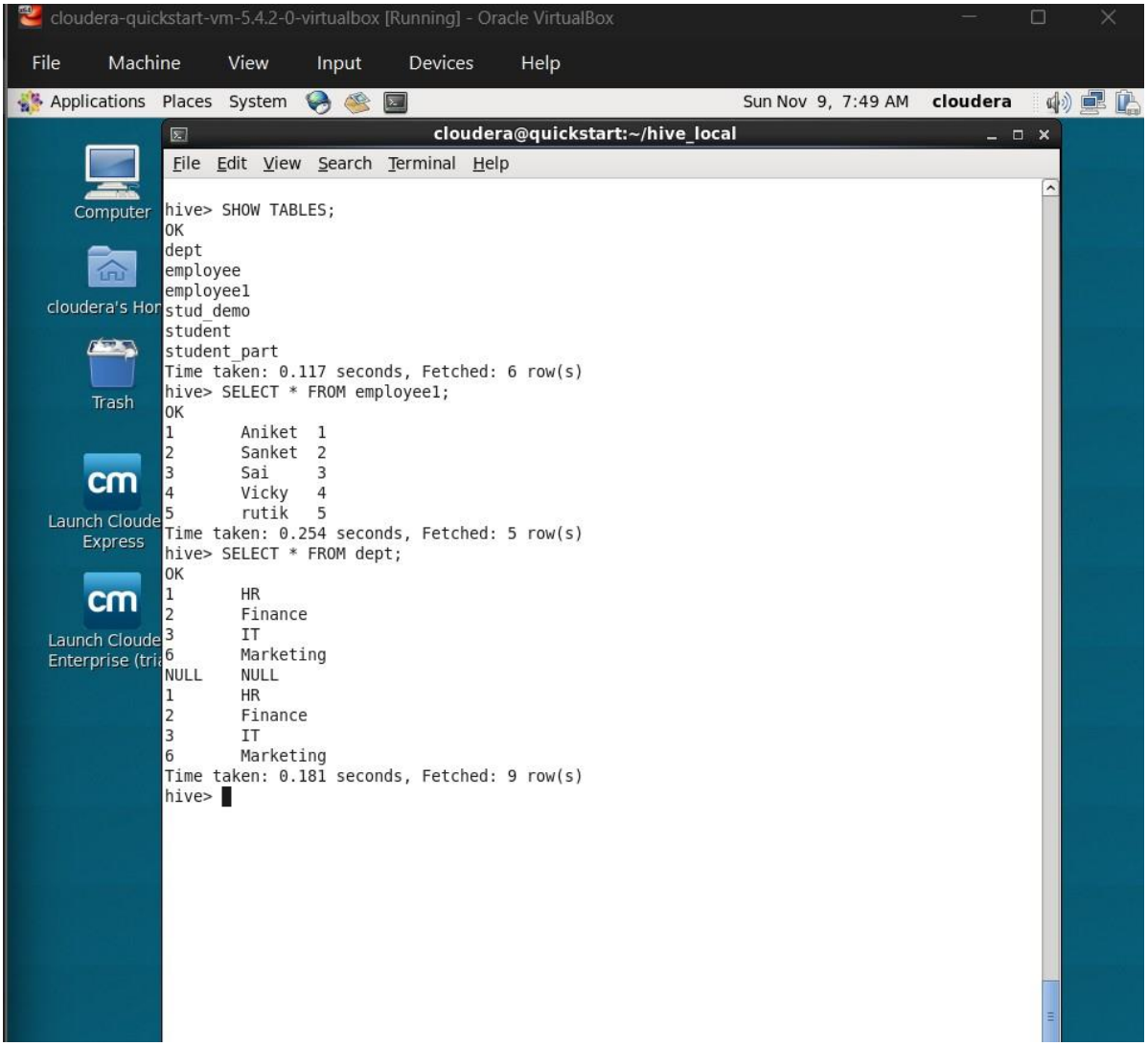


G. RIGHT OUTER JOIN



H. FULL OUTER JOIN


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 7:48 AM cloudera
cloudera@quickstart:~/hive_local
File Edit View Search Terminal Help
Computer
cloudera's Hor
Trash
Launch Cloude Express
Launch Cloude Enterprise (tri
hive> SELECT e.empname, d.department_name
> FROM employee1 e
> FULL OUTER JOIN dept d
> ON (e.deptid = d.deptid);
Query ID = cloudera_20251109074646_a738e4a8-7ba7-44f0-9d40-421e9cd635cc
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
Starting Job = job_1762693679587_0012, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1762693679587_0012/
Kill Command = /usr/lib/hadoop/bin/hadoop job -kill job_1762693679587_0012
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2025-11-09 07:47:08,172 Stage-1 map = 0%, reduce = 0%
2025-11-09 07:47:52,821 Stage-1 map = 50%, reduce = 0%, Cumulative CPU 7.38 sec
2025-11-09 07:47:54,029 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 14.79 sec
2025-11-09 07:48:26,202 Stage-1 map = 100%, reduce = 67%, Cumulative CPU 15.1 sec
2025-11-09 07:48:30,872 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 26.17 sec
MapReduce Total cumulative CPU time: 26 seconds 170 msec
Ended Job = job_1762693679587_0012
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2 Reduce: 1 Cumulative CPU: 26.17 sec HDFS Read: 11379 HDFS Write: 114 SUCCESS
Total MapReduce CPU Time Spent: 26 seconds 170 msec
OK
NULL NULL
Aniket HR
Aniket HR
Sanket Finance
Sanket Finance
Sai IT
Sai IT
Vicky NULL
rutik NULL
NULL Marketing
NULL Marketing
Time taken: 128.577 seconds, Fetched: 11 row(s)
hive>

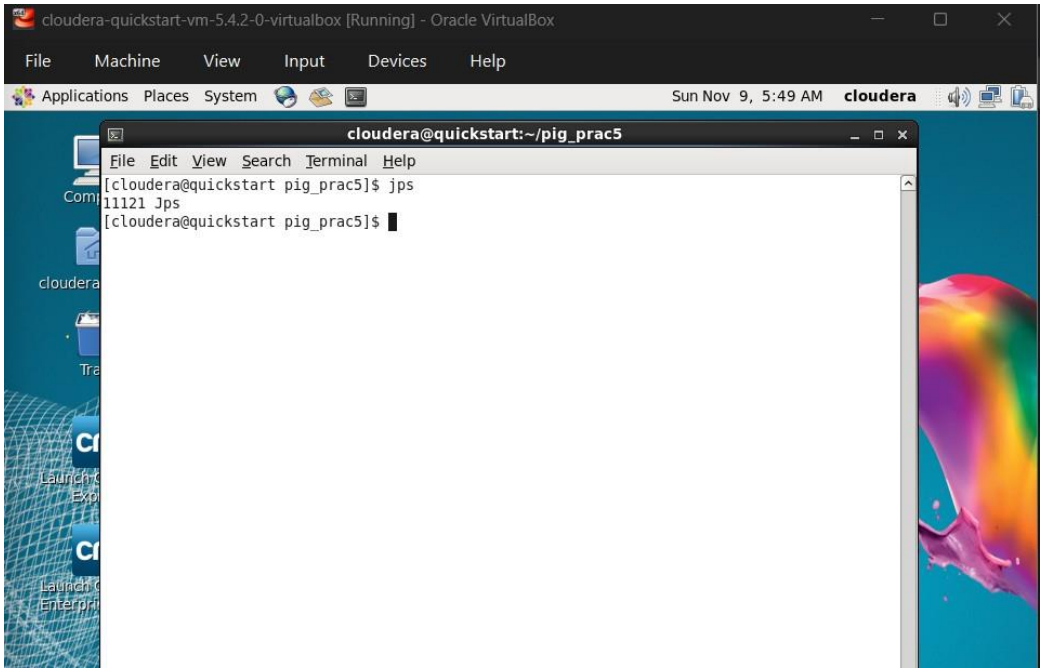

cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 7:49 AM cloudera
cloudera@quickstart:~/hive_local
File Edit View Search Terminal Help
Computer
cloudera's Hor
Trash
Launch Cloude Express
Launch Cloude Enterprise (tri
hive> SHOW TABLES;
OK
dept
employee
employee1
stud_demo
student
student_part
Time taken: 0.117 seconds, Fetched: 6 row(s)
hive> SELECT * FROM employee1;
OK
1 Aniket 1
2 Sanket 2
3 Sai 3
4 Vicky 4
5 rutik 5
Time taken: 0.254 seconds, Fetched: 5 row(s)
hive> SELECT * FROM dept;
OK
1 HR
2 Finance
3 IT
6 Marketing
NULL NULL
1 HR
2 Finance
3 IT
6 Marketing
Time taken: 0.181 seconds, Fetched: 9 row(s)
hive>

Practical No 5 : Pig

1. Start the Cloudera Quickstart VM

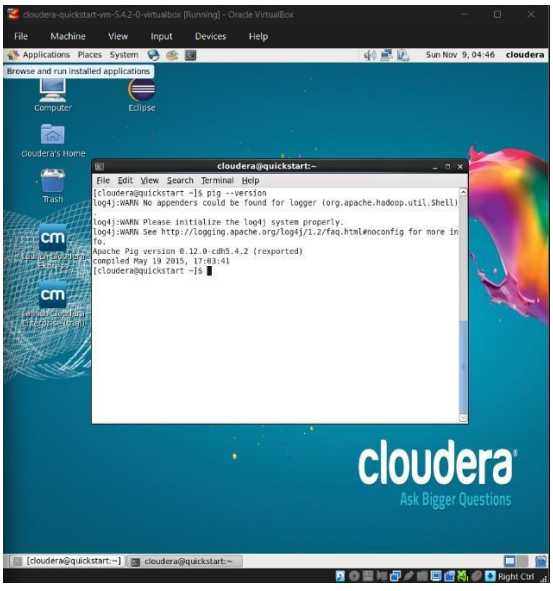
Launch the VirtualBox and start the Cloudera Quickstart virtual machine.

Confirm that Hadoop daemons (NameNode, DataNode, etc.) are running.



2: Verify Pig Installation

Check Pig version and environment

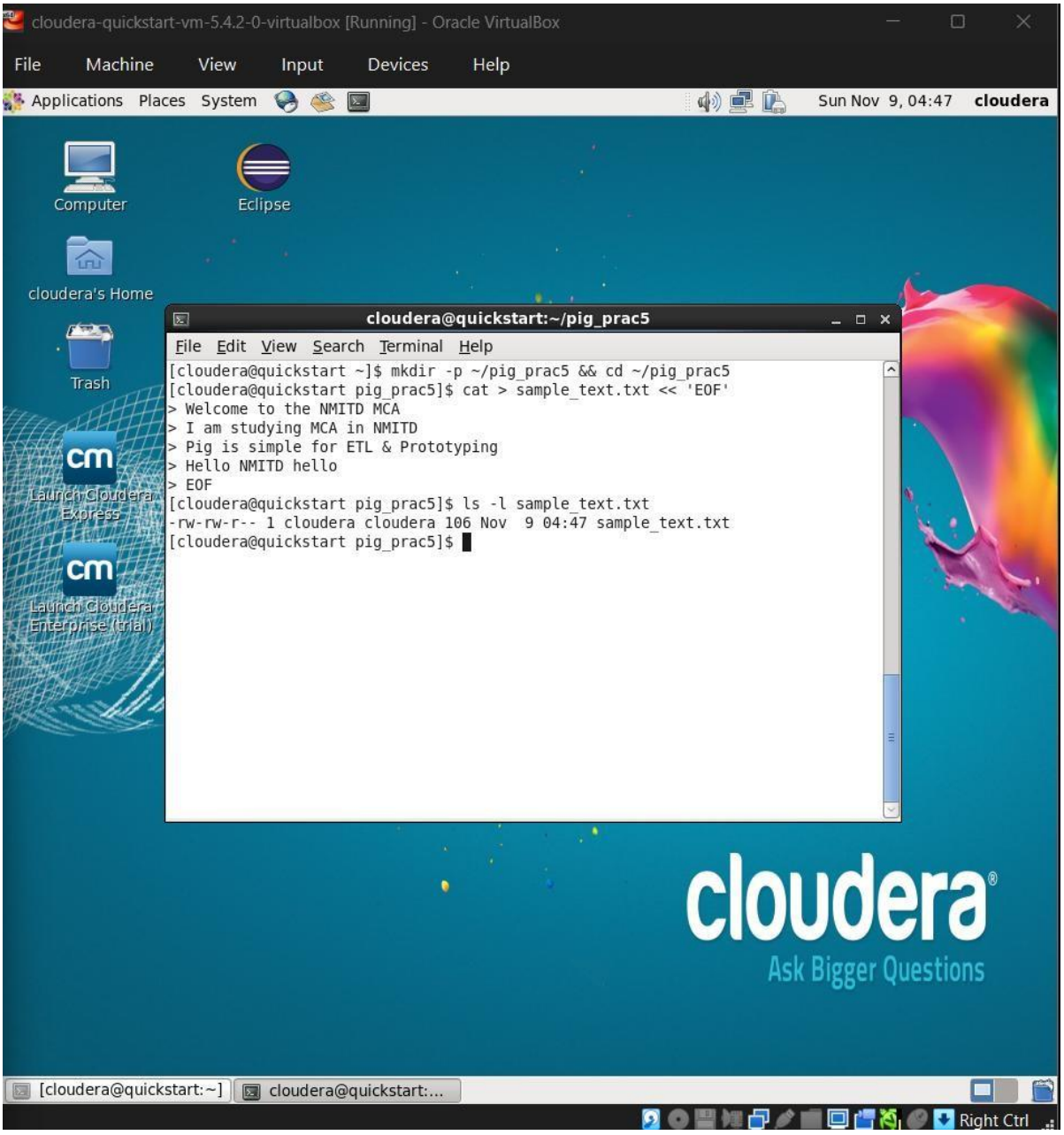


3: Create Working Directory and Dataset

Create a working directory:

Create a sample text file:

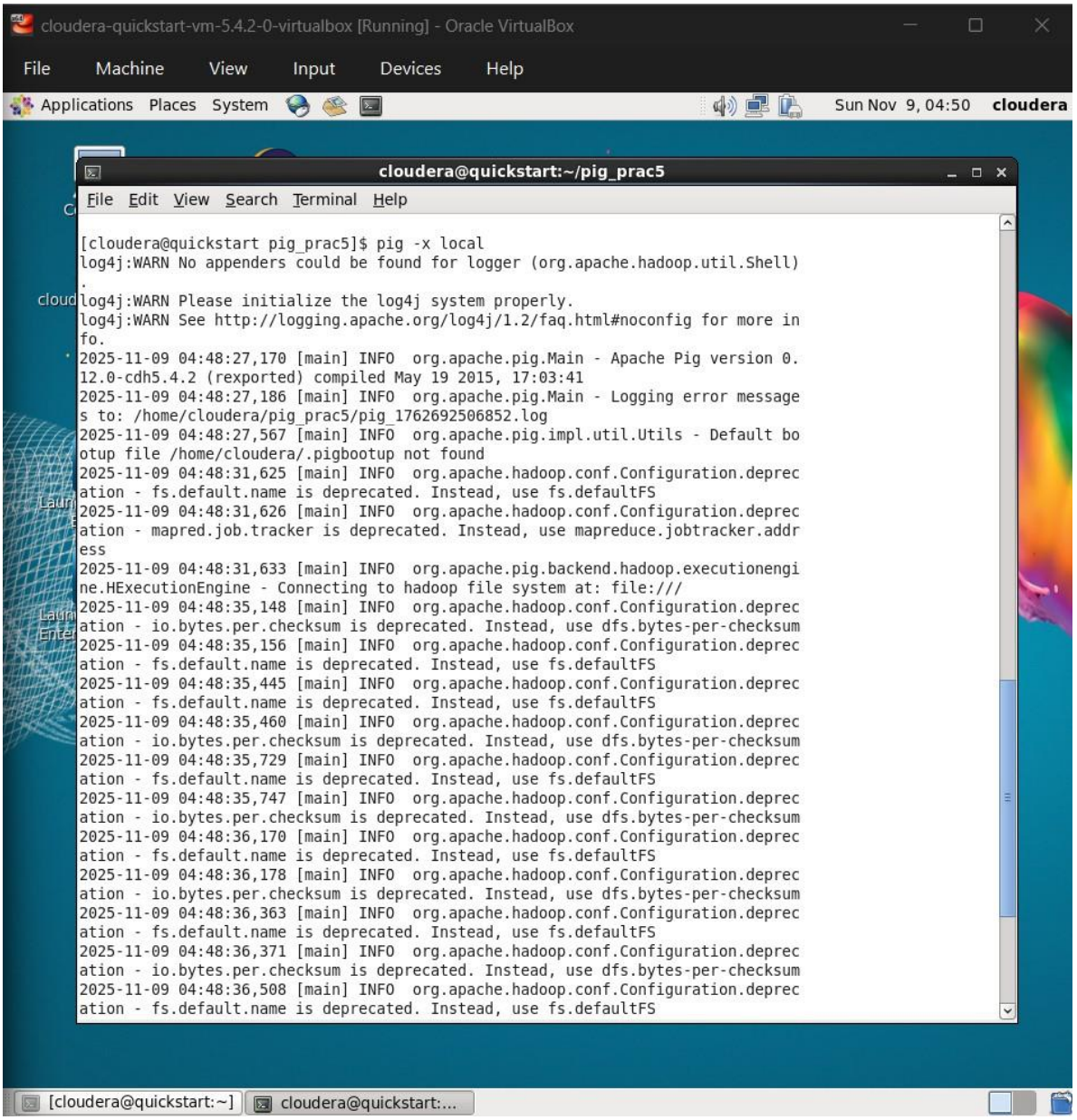
Verify the file

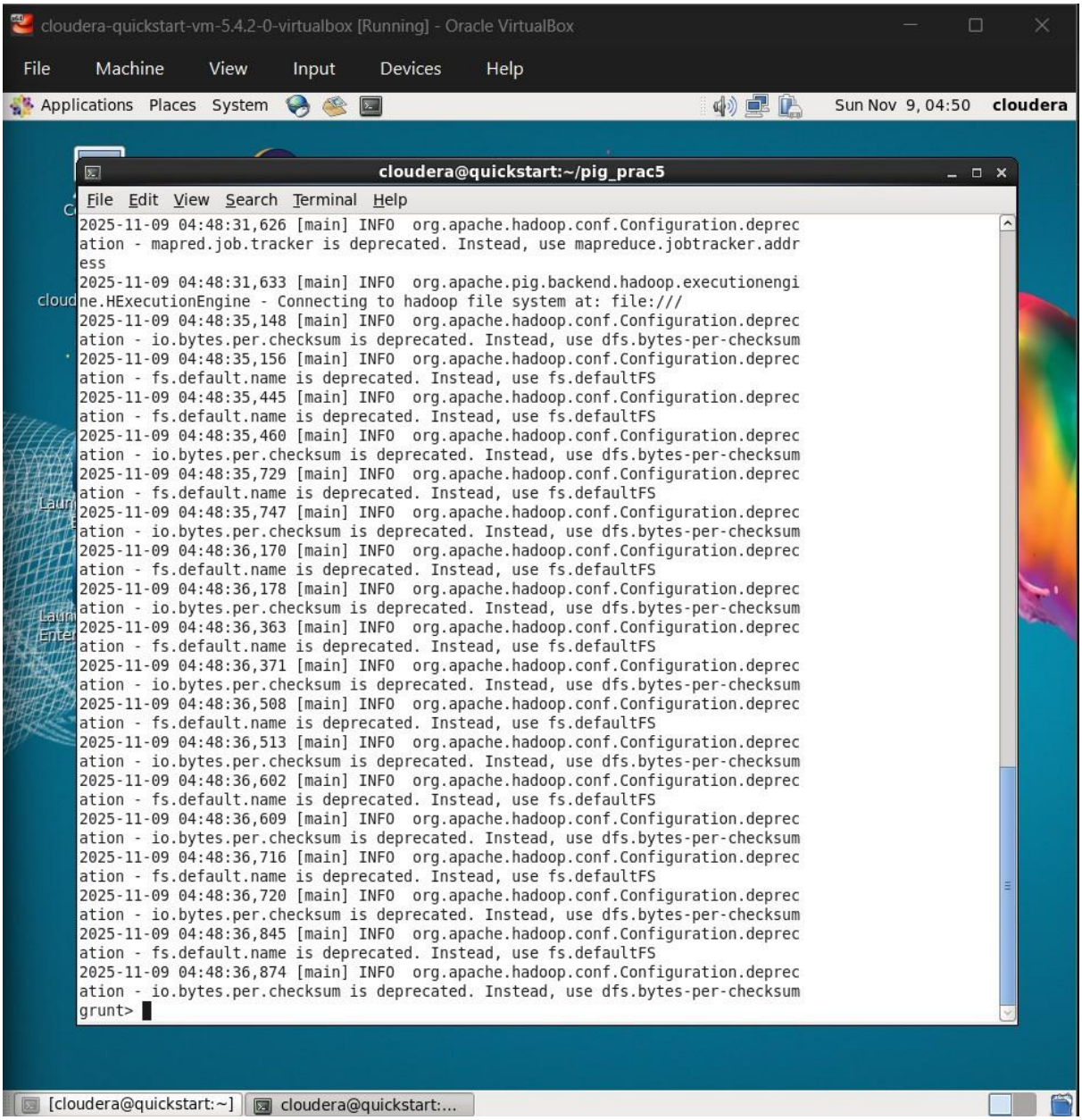


Step 4: Run Pig in Local Mode

Start Pig shell:

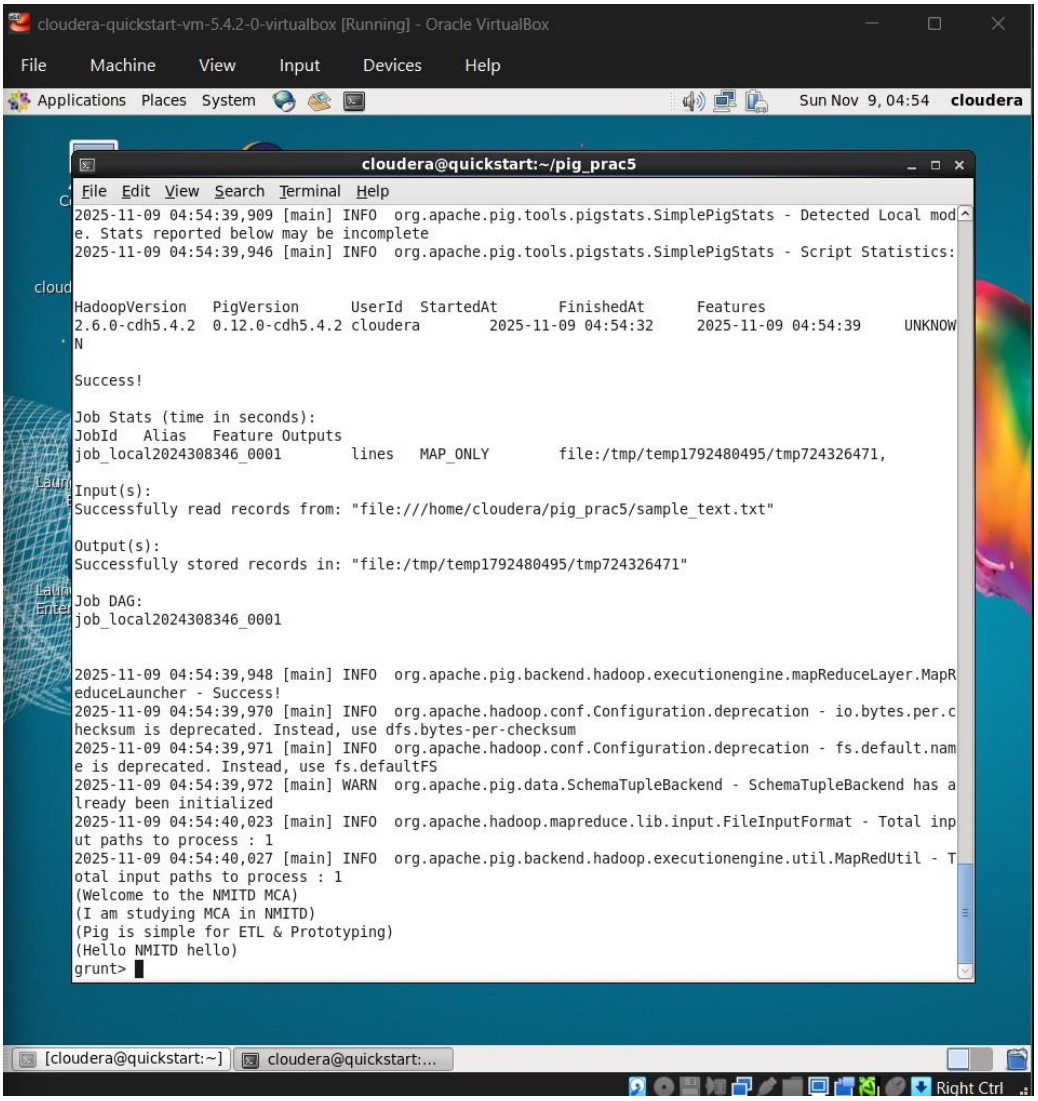
Launch Pig Grunt shell to test basic commands locally.





Load and view data:

```
ation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
grunt> lines = LOAD 'sample_text.txt' AS (line:chararray);
grunt> DUMP lines;
2025-11-09 04:54:29,083 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in
```

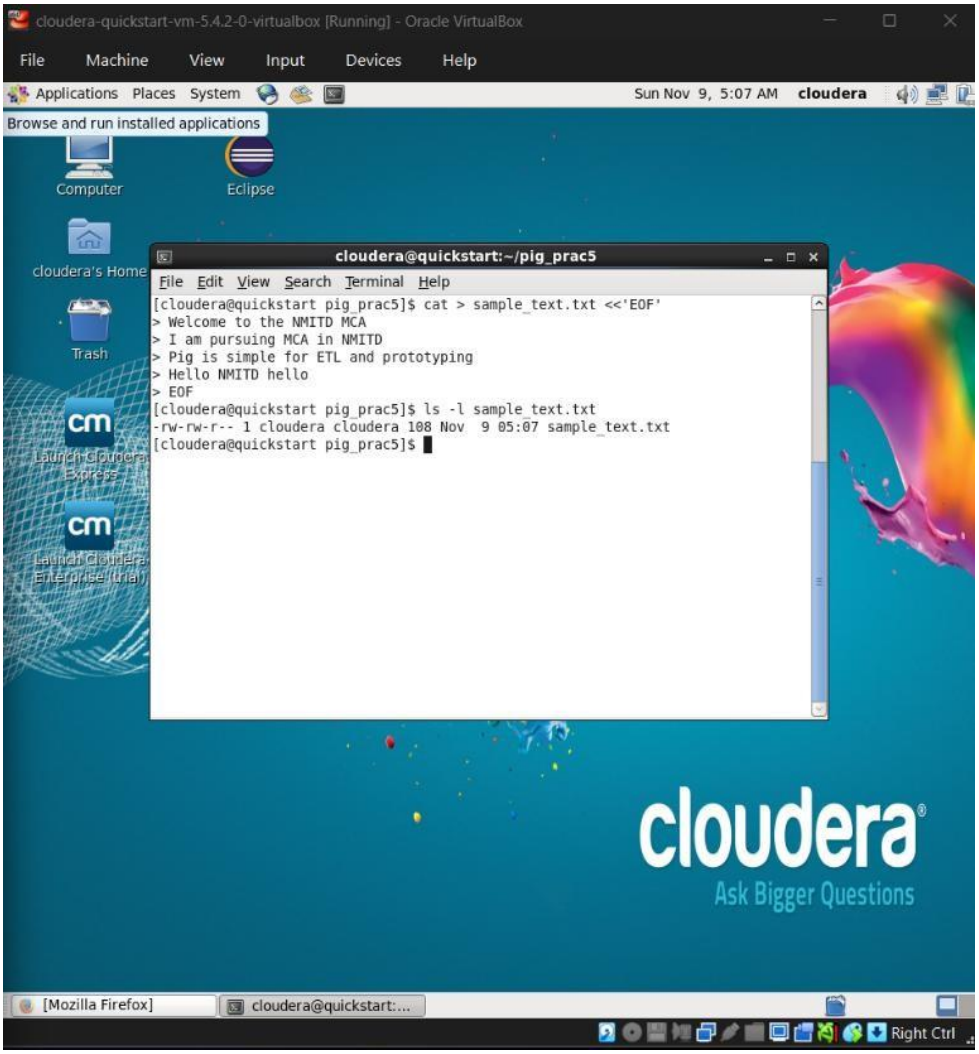


5: Write and Execute WordCount Script in Local Mode

Exit Grunt shell using:

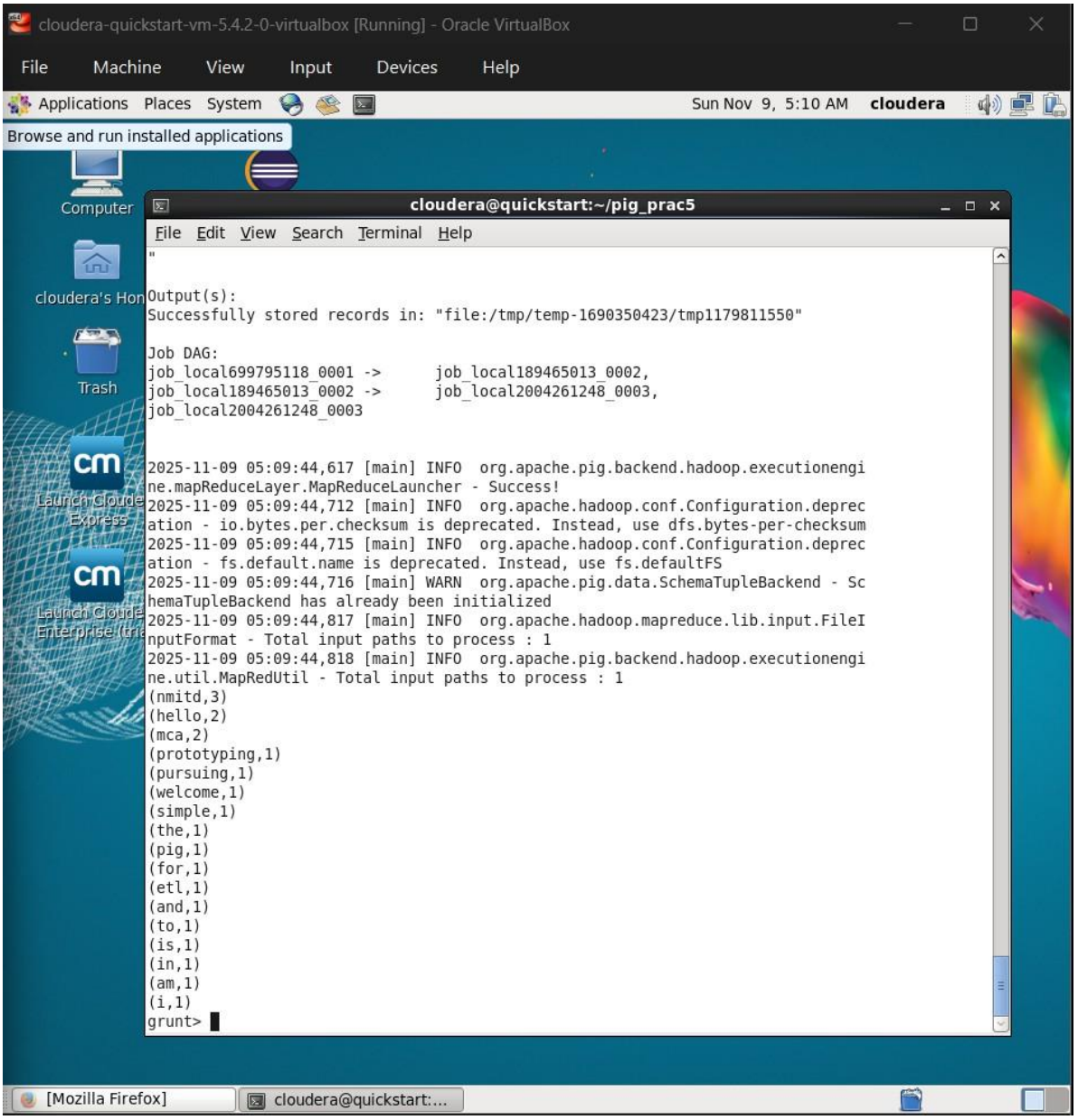
quite;

Create script file (Write Pig Latin script for word count in file):



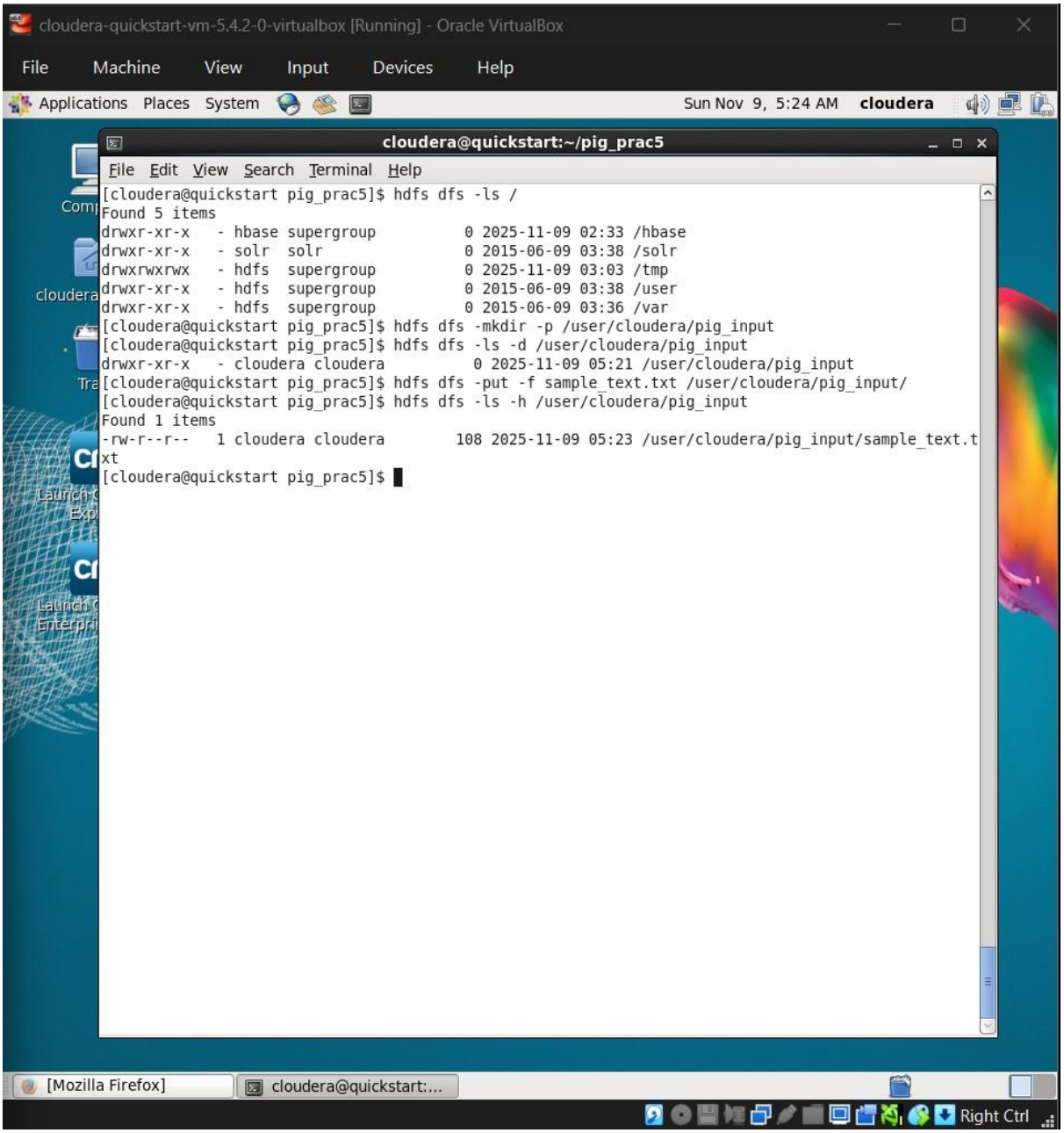
Run that file: pig -x local wordcount_local.pig

Output :

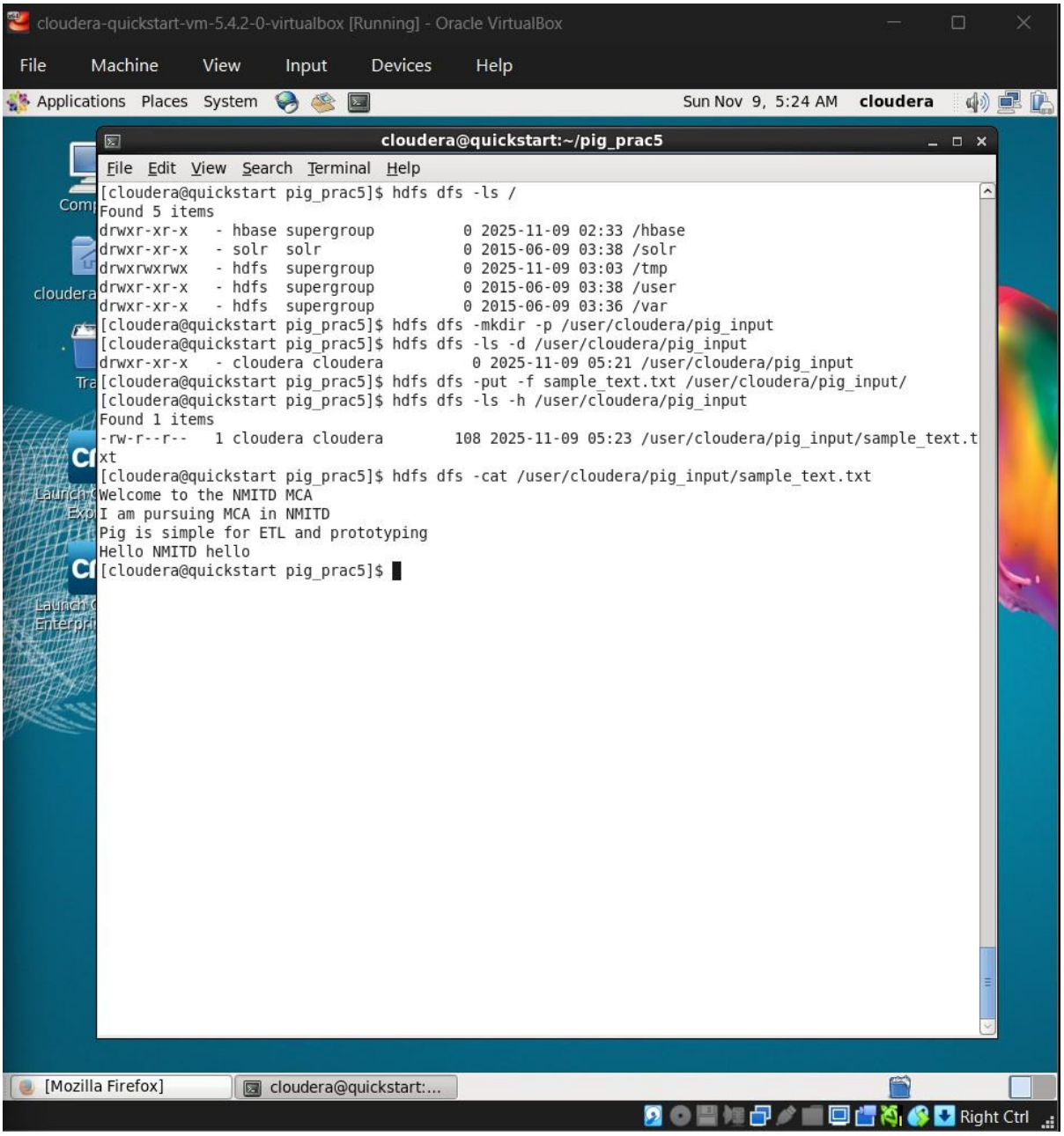


6. Prepare Data in HDFS (for MapReduce Mode)

Create directory and upload file to HDFS for MapReduce:


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 5:24 AM cloudera
cloudera@quickstart:~/pig_prac5
File Edit View Search Terminal Help
[cloudera@quickstart pig_prac5]$ hdfs dfs -ls /
Found 5 items
drwxr-xr-x - hbase supergroup 0 2025-11-09 02:33 /hbase
drwxr-xr-x - solr solr 0 2015-06-09 03:38 /solr
drwxrwxrwx - hdfs supergroup 0 2025-11-09 03:03 /tmp
drwxr-xr-x - hdfs supergroup 0 2015-06-09 03:38 /user
drwxr-xr-x - hdfs supergroup 0 2015-06-09 03:36 /var
[cloudera@quickstart pig_prac5]$ hdfs dfs -mkdir -p /user/cloudera/pig_input
[cloudera@quickstart pig_prac5]$ hdfs dfs -ls -d /user/cloudera/pig_input
drwxr-xr-x - cloudera cloudera 0 2025-11-09 05:21 /user/cloudera/pig_input
[cloudera@quickstart pig_prac5]$ hdfs dfs -put -f sample_text.txt /user/cloudera/pig_input/
[cloudera@quickstart pig_prac5]$ hdfs dfs -ls -h /user/cloudera/pig_input
Found 1 items
-rw-r--r-- 1 cloudera cloudera 108 2025-11-09 05:23 /user/cloudera/pig_input/sample_text.t
xt
[cloudera@quickstart pig_prac5]$
[Mozilla Firefox]
cloudera@quickstart:...
Right Ctrl

Output :

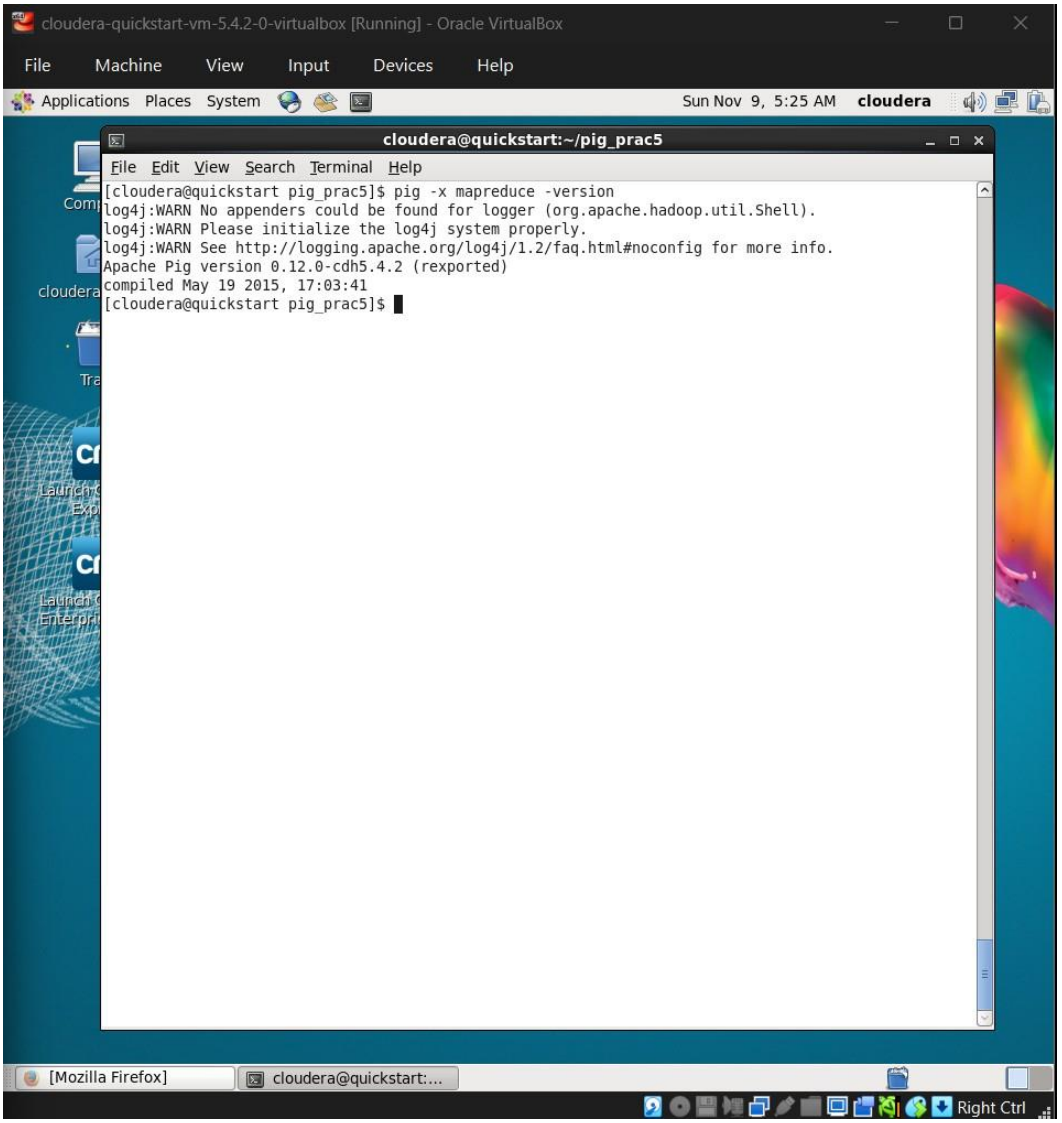


7. Verify Pig MapReduce Mode

Check Pig's MapReduce engine connectivity:

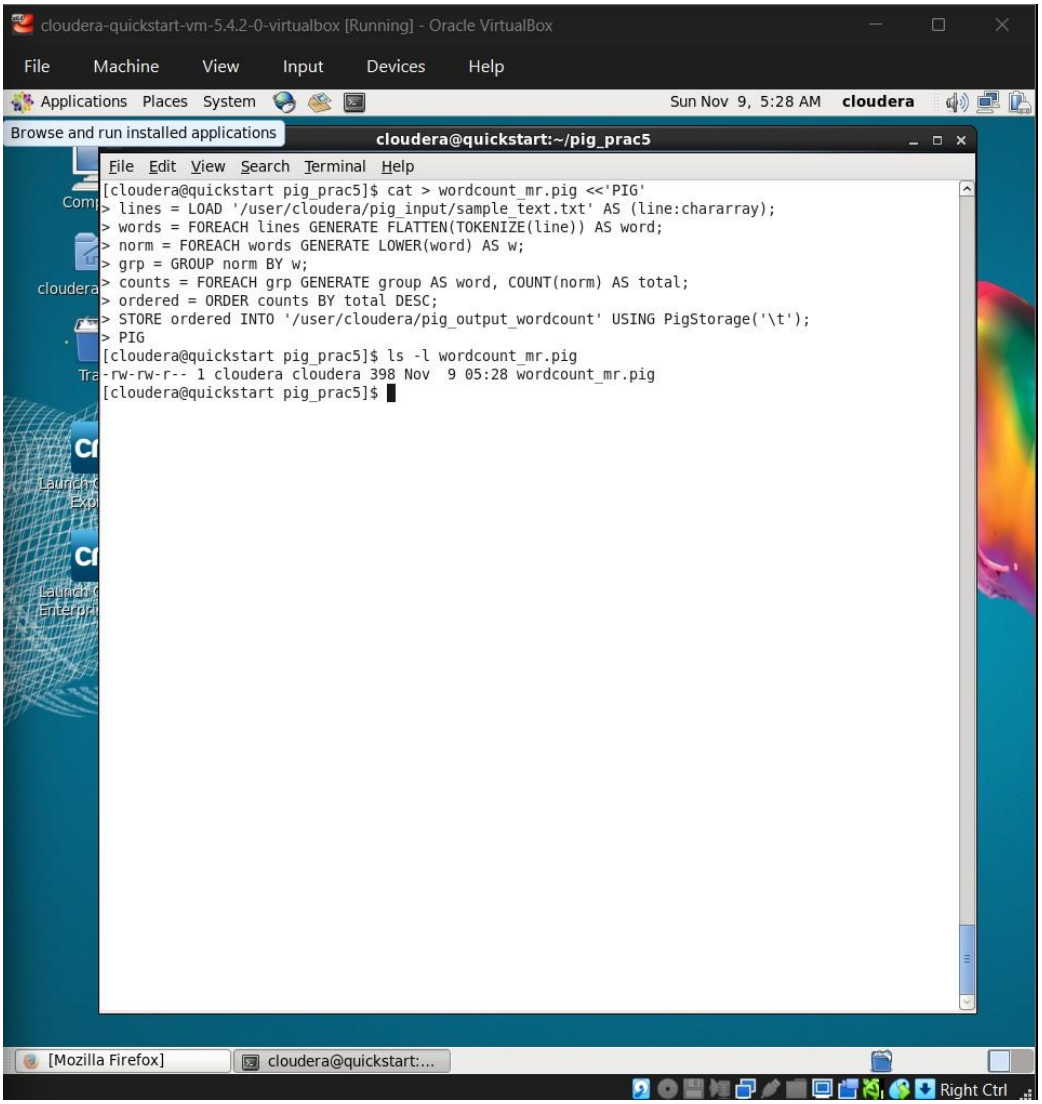
- pig -x mapreduce -version

Output :



8. Create Pig WordCount Script for HDFS

Create the Pig script:



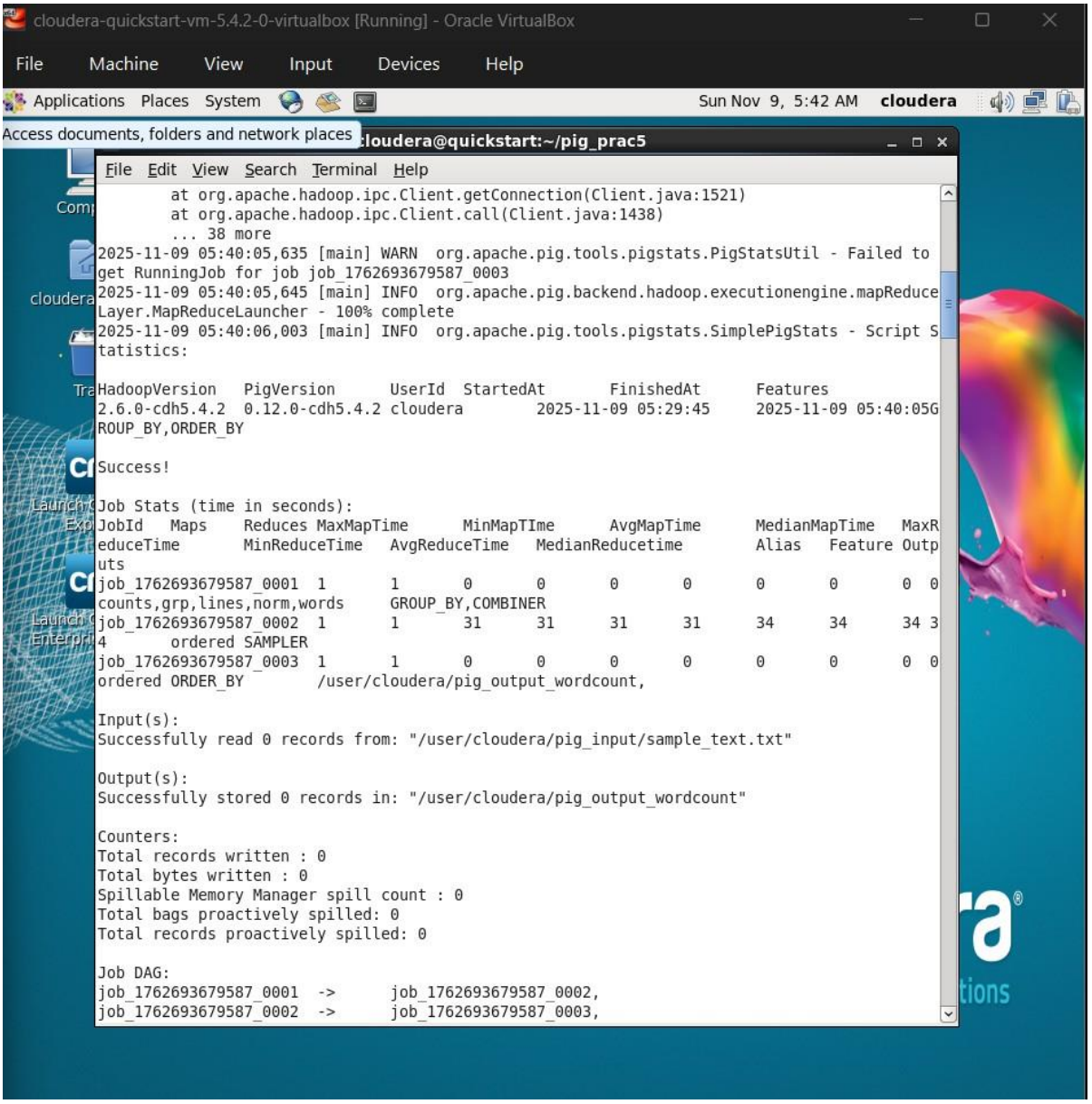
9. Run Pig in MapReduce Mode

Execute the script:

pig wordcount_mr.pig

Pig launches MapReduce jobs & output path displayed.

Output :



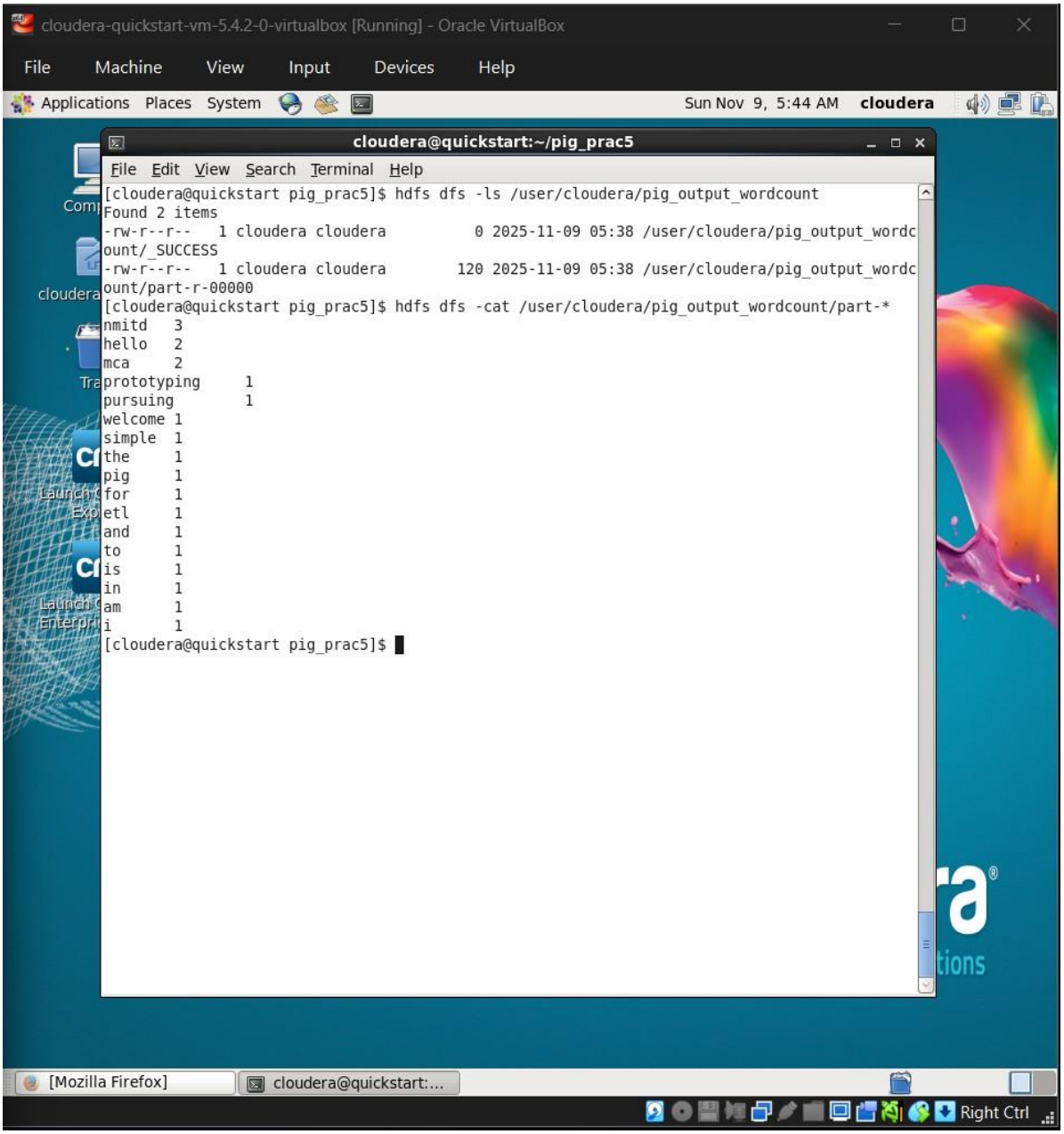
10. Verify the Output in HDFS

Check output directory:

hdfs dfs -ls /user/cloudera/pig_output_wordcount

hdfs dfs -cat /user/cloudera/pig_output_wordcount/part-r-*

Output:


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Nov 9, 5:44 AM cloudera
cloudera@quickstart:~/pig_prac5
File Edit View Search Terminal Help
[cloudera@quickstart pig_prac5]$ hdfs dfs -ls /user/cloudera/pig_output_wordcount
Found 2 items
-rw-r--r-- 1 cloudera cloudera 0 2025-11-09 05:38 /user/cloudera/pig_output_wordcount/_SUCCESS
-rw-r--r-- 1 cloudera cloudera 120 2025-11-09 05:38 /user/cloudera/pig_output_wordcount/part-r-00000
[cloudera@quickstart pig_prac5]$ hdfs dfs -cat /user/cloudera/pig_output_wordcount/part-*
nmitd 3
hello 2
mca 2
prototyping 1
pursuing 1
welcome 1
simple 1
the 1
pig 1
for 1
etl 1
and 1
to 1
is 1
in 1
am 1
i 1
[cloudera@quickstart pig_prac5]$
[Mozilla Firefox]
cloudera@quickstart:...
Right Ctrl

Practical 6 No - Spark

Apache Spark Commands in Scala	**Pair RDD (Key-Value RDD) Operations**
Start Spark Shell	Create Pair RDD
Create RDD from Collection	reduceByKey
Read Text File into RDD (from HDFS)	groupByKey — With Output Viewing
Map Transformation	mapValues
Filter Transformation	sortByKey
Reduce Action	join
Collect Action	cogroup
Save RDD to HDFS	aggregateByKey
Cache/Persist RDD	foldByKey
	Read from Local File

1. Start Spark Shell

spark-shell --master local[*] (You can type spark-shell)

Starts the interactive Spark shell using all available CPU cores.

2. Create RDD from Collection

val rdd = sc.parallelize(Seq(1, 2, 3, 4, 5))

Creates an RDD from an in-memory Scala collection.

If there is a error like this

```
scala> val rdd=sc.parallelize(Seq(1,2,3,4,5))
java.lang.IllegalStateException: Cannot call methods on a stopped SparkContext.
This stopped SparkContext was created at:

org.apache.spark.SparkContext.<init>(SparkContext.scala:84)
org.apache.spark.repl.SparkILoop.createSparkContext(SparkILoop.scala:1022)
$iwC$$iwC.<init>(<console>:15)
$iwC.<init>(<console>:25)
<init>(<console>:27)
.<init>(<console>:31)
.<clinit>(<console>)
.<init>(<console>:7)
.<clinit>(<console>)
$print(<console>)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:606)
org.apache.spark.repl.SparkIMain$ReadEvalPrint.call(SparkIMain.scala:1045)
org.apache.spark.repl.SparkIMain$Request.loadAndRun(SparkIMain.scala:1326)
org.apache.spark.repl.SparkIMain.loadAndRunReq$1(SparkIMain.scala:821)
org.apache.spark.repl.SparkIMain.interpret(SparkIMain.scala:852)
org.apache.spark.repl.SparkIMain.interpret(SparkIMain.scala:800)
org.apache.spark.repl.SparkILoop.reallyInterpret$1(SparkILoop.scala:857)

The currently active SparkContext was created at:

(No active SparkContext.)
```

You need to **create a new** `SparkContext`, since the old one is stopped and can't be used again.

✅ **Step-by-step in** `spark-shell` **or** `Scala REPL`:

1. **Stop the existing SparkContext explicitly (if not already stopped):**

```scala
sc.stop()
```

2. **Import necessary Spark classes:**

```scala
import org.apache.spark.{SparkConf, SparkContext}
```

3. **Create a new SparkConf and SparkContext:**

```scala
val conf = new SparkConf().setAppName("NewApp").setMaster("local[*]")
val sc = new SparkContext(conf)
```

4. **Now re-run your command:**

```scala
val rdd = sc.parallelize(Seq(1,2,3,4,5))
```

val rdd = sc.parallelize(Seq(1,2,3,4,5))

```
scala> val rdd = sc.parallelize(Seq(1,2,3,4,5))
rdd: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:21
```

3. Read Text File into RDD (from HDFS)

val rdd = sc.textFile("hdfs:///user/cloudera/file.txt")

Reads a text file from HDFS. Each line becomes an RDD element.

How to Read/View an RDD

Task	Command	Explanation
Print all elements	rdd.collect().foreach(println)	Prints each element. Use only if RDD is small.
Print first 5 lines	rdd.take(5).foreach(println)	Prints first 5 elements. Useful for previews.
Count number of lines	rdd.count()	Shows total number of records.
Print with line numbers	rdd.zipWithIndex().collect().foreach { case (line, i) => println(s"$i: $line") }	Shows each line with its index.

```
scala> result.collect()
[Stage 0:>                                                          (0 + 0) / 2]25/08/23 02:39:39 WARN
; check your cluster UI to ensure that workers are registered and have sufficient resources
res0: Array[Int] = Array(2, 4, 6, 8, 10)

scala>
```

4. Map Transformation

val result = rdd.map(x => x * 2)
Applies a function to every element in the RDD.

```
scala> val result=rdd.map(x=>x*2)
result: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[1] at map at <console>:23

scala> result.collect()
[Stage 0:>                                                          (0 + 0) / 2]25/08/23 02:39:39 WARN cluster.
; check your cluster UI to ensure that workers are registered and have sufficient resources
res0: Array[Int] = Array(2, 4, 6, 8, 10)

scala>
```

5. Filter Transformation

val result = rdd.filter(x => x > 3)
Keeps only the elements that satisfy the condition.

```
scala> val result=rdd.filter(x=>x>3)
result: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[2] at filter at <console>:23

scala> result.collect()
res1: Array[Int] = Array(4, 5)

scala> val sum=rdd.reduce((a,b)=>a+b)
sum: Int = 15
```

6. Reduce Action

val sum = rdd.reduce((a, b) => a + b)
Combines all elements using the specified function.

```
scala> val pairs = sc.parallelize(Seq(("j", 1), ("a", 2), ("j", 3)))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[3] at paralleli
ze at <console>:21

scala> pairs.reduceByKey(_ + _).collect().foreach(println)
(j,4)
(a,2)

scala>
```

7. Collect Action

rdd.collect()
Returns all RDD elements as an array to the driver.

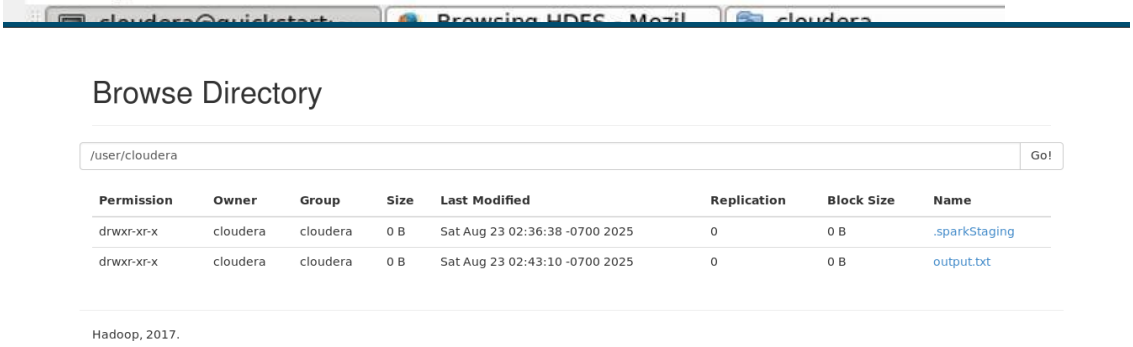
```
scala> val rdd = sc.parallelize(Seq(1,2,3,4,5))
rdd: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:21

scala> val result=rdd.map(x=>x*2)
result: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[1] at map at <console>:23

scala> result.collect()
[Stage 0:>                                                          (0 + 0) / 2]25/08/23 02:39:39 WARN cluster.
; check your cluster UI to ensure that workers are registered and have sufficient resources
res0: Array[Int] = Array(2, 4, 6, 8, 10)
```

8. Save RDD to HDFS

rdd.saveAsTextFile("hdfs:///user/cloudera/out
put")

9. Saves RDD to the given HDFS directory.

```
scala> rdd.saveAsTextFile("hdfs:///user/cloudera/output.txt")

scala>
```



10. Cache/Persist RDD

rdd.cache()

Stores the RDD in memory for faster reuse.

```
scala> rdd.cache()
res4: rdd.type = ParallelCollectionRDD[0] at parallelize at <console>:21

scala>
```

Pair RDD (Key-Value RDD) Operations

11. Create Pair RDD

val pairs = sc.parallelize(Seq(("a", 1), ("b", 2), ("a", 3)))

```
scala> val pairs=sc.parallelize(Seq(("a",1),("b",2),("a",3)))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[7] at parallelize at <console>:21
```

```
scala> val pairs=sc.parallelize(Seq(("a",1),("b",2),("c",3)))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[5] at parallelize at <console>:21
```

12. reduceByKey

pairs.reduceByKey(_ + _).collect().foreach(println)

Adds values with the same key. Efficient for aggregations.

```
scala> pairs.reduceByKey(_+_).collect().foreach(println)
(b,2)
(a,1)
(c,3)

scala> pairs.reduceByKey(_+_).collect().foreach(println)
(b,2)
(a,4)
```

13. groupByKey — With Output Viewing

```
val grouped = pairs.groupByKey()
// View output clearly
grouped.collect().foreach { case (key, values) =>
  println(s"$key -> ${values.mkString(", ")3")
3
```

Explanation:

Groups values by key. mkString makes the Iterable readable.

Output Example:
a -> 1, 3
b -> 2

```
scala> val grouped = pairs.groupByKey()

grouped.collect().foreach { case (key, values) =>
  println(s"$key -> ${values.mkString(", ")}")
}
j -> 1, 3
a -> 2
grouped: org.apache.spark.rdd.RDD[(String, Iterable[Int])] = ShuffledRDD[5] at groupBy
Key at <console>:23
```

14. mapValues

```
pairs.mapValues(x => x + 1).collect().foreach(println)
```

Applies a function to the value part only.

```
scala> pairs.mapValues(x=>x+1).collect().foreach(println)
(a,2)
(b,3)
(a,4)
```

15. sortByKey

```
pairs.sortByKey().collect().foreach(print
ln)
```

Sorts RDD by key in ascending order.

```
scala> pairs.sortByKey().collect().foreach(println)
(a,1)
(a,3)
(b,2)
```

16. Join

val rdd1 = sc.parallelize(Seq(("a", 1), ("b", 2)))
val rdd2 = sc.parallelize(Seq(("a", 100), ("b",
200))) val joined = rdd1.join(rdd2)
joined.collect().foreach(println)

Performs an inner join on two key-value RDDs.

```
scala> val rdd1=sc.parallelize(Seq(("a",1),("b",2)))
rdd1: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[14] at parallelize at <console>:21

scala> val rdd2=sc.parallelize(Seq(("a",100),("b",200)))
rdd2: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[15] at parallelize at <console>:21

scala> val joined=rdd1.join(rdd2)
joined: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[18] at join at <console>:25

scala> joined.collect().foreach(println)
(b,(2,200))
(a,(1,100))

scala>
```

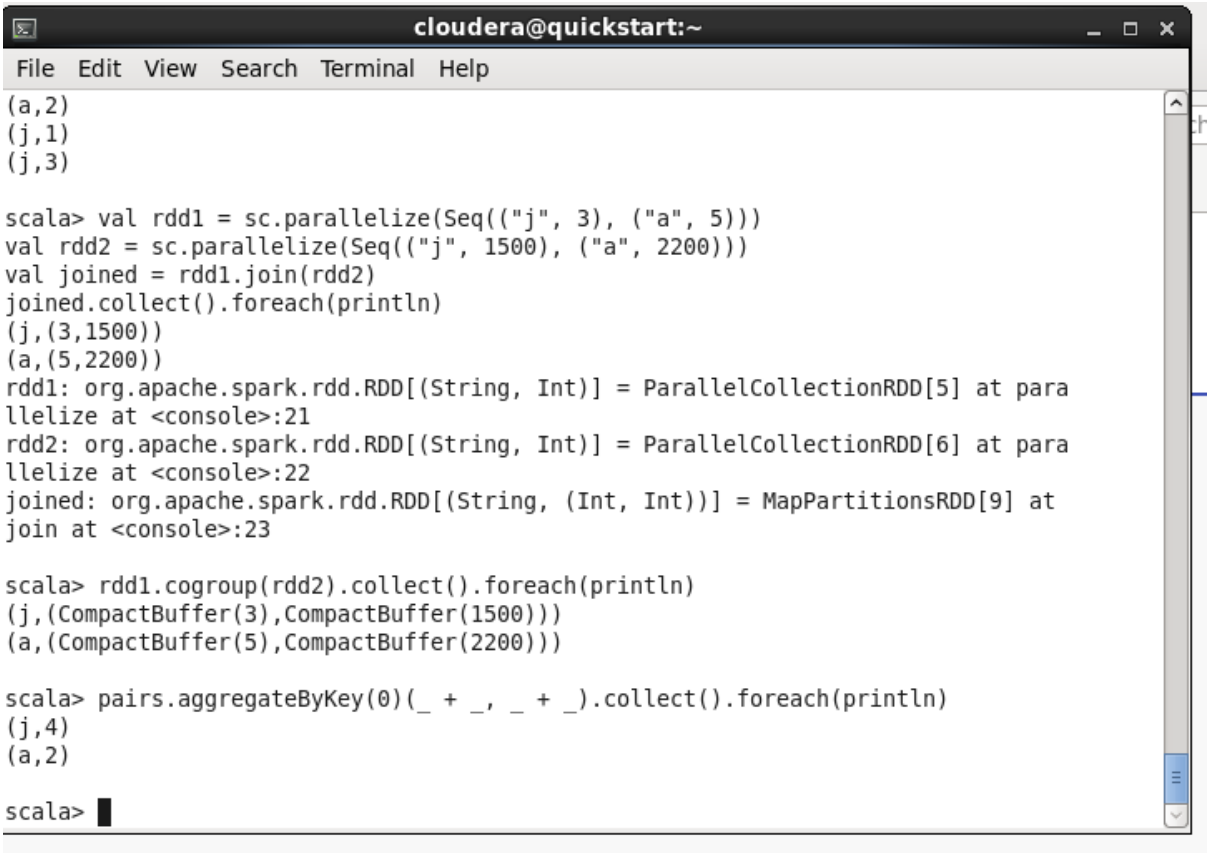
17. Cogroup

rdd1.cogroup(rdd2).collect().foreach(println)
Groups values from both RDDs into tuples of lists for each key.

```
scala> rdd1.cogroup(rdd2).collect().foreach(println)
(j,(CompactBuffer(3),CompactBuffer(1500)))
(a,(CompactBuffer(5),CompactBuffer(2200)))

scala>
```

18. aggregateByKey

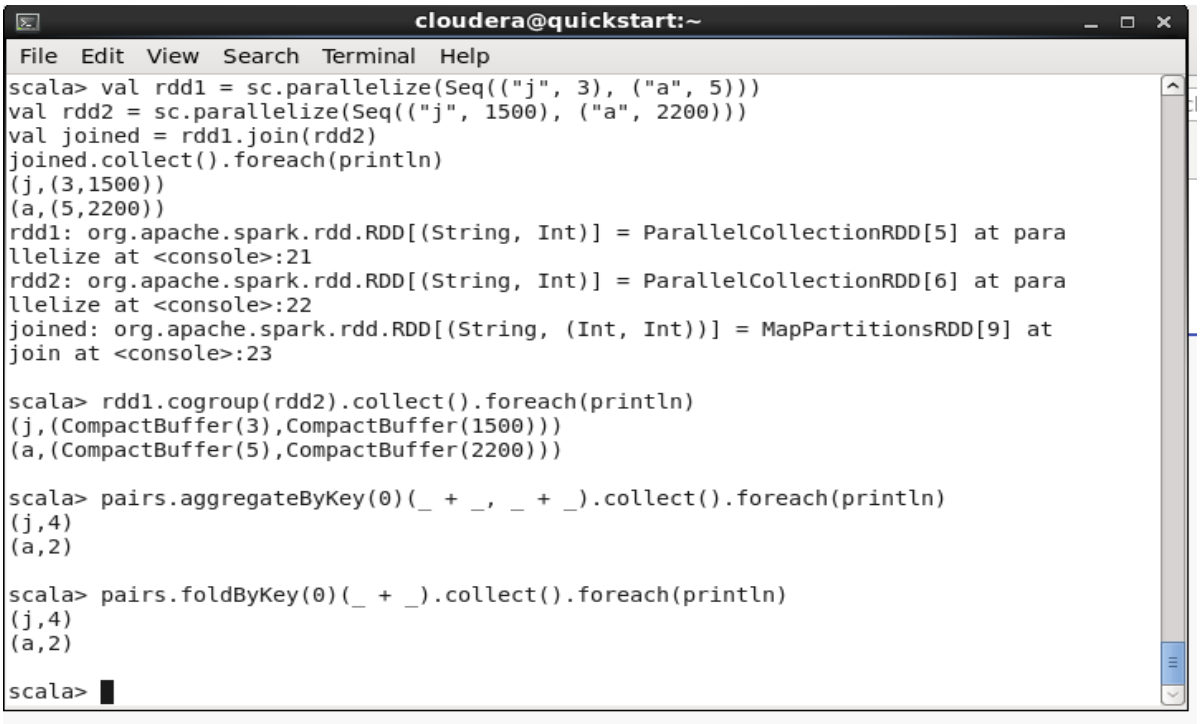
pairs.aggregateByKey(0)(_ + _, _ + _).collect().foreach(println)
Combines values for each key using custom logic for intra- and inter-partition aggregation.



19. foldByKey

pairs.foldByKey(0)(_ + _).collect().foreach(println)

Aggregates values with a neutral zero value using a single function.



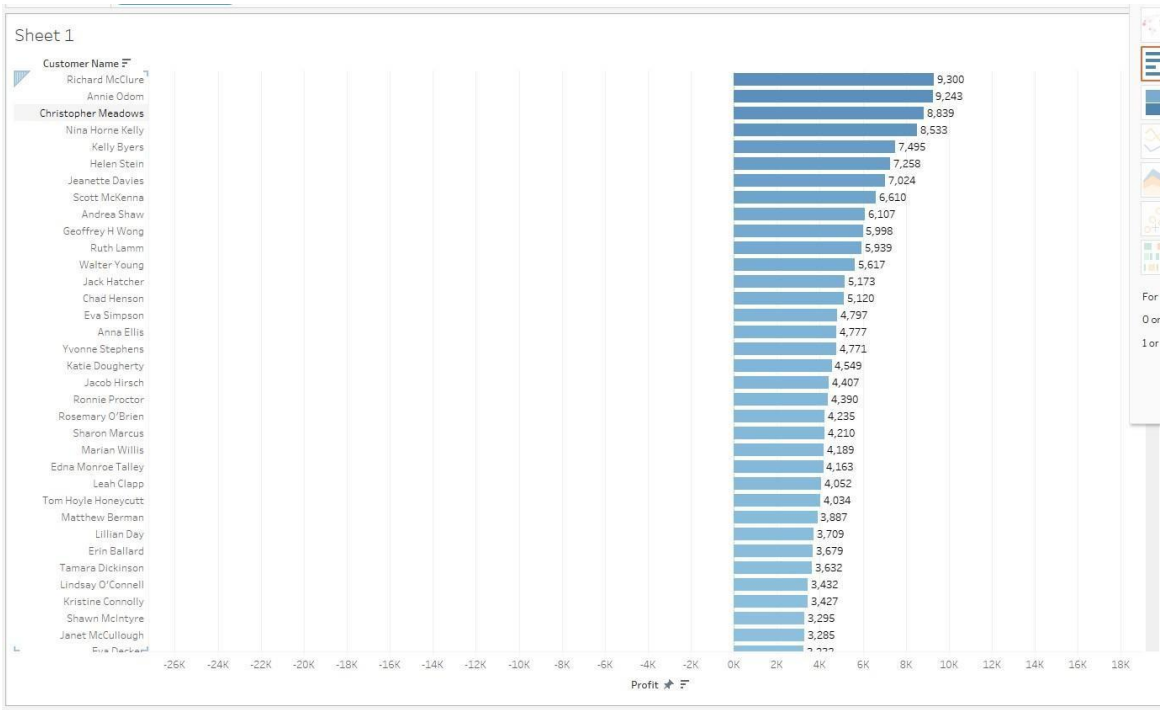
Practical 7 No - Visualization using Tableau:

Tableau: Tool Overview, Importing Data, Analyzing with Charts, Creating Dashboards, working with maps

Analysis Operations

Q1. Find the customer with the highest overall profit. What is his/her profit ratio?

- Import superstoreus2015.xlsx dataset.
- Create Calculated Field:
- Profit Ratio = (SUM([Profit]) / SUM([Sales]))
- Drag Customer Name to Rows, Profit to Columns. Sort by Profit.
- Show Profit Ratio in Marks label.




Filters
Marks
Automatic
Color
Size
Label
Detail
Tooltip
SUM(Profit)
SUM(Profit)
Columns
SUM(Profit)
Rows
Customer Name

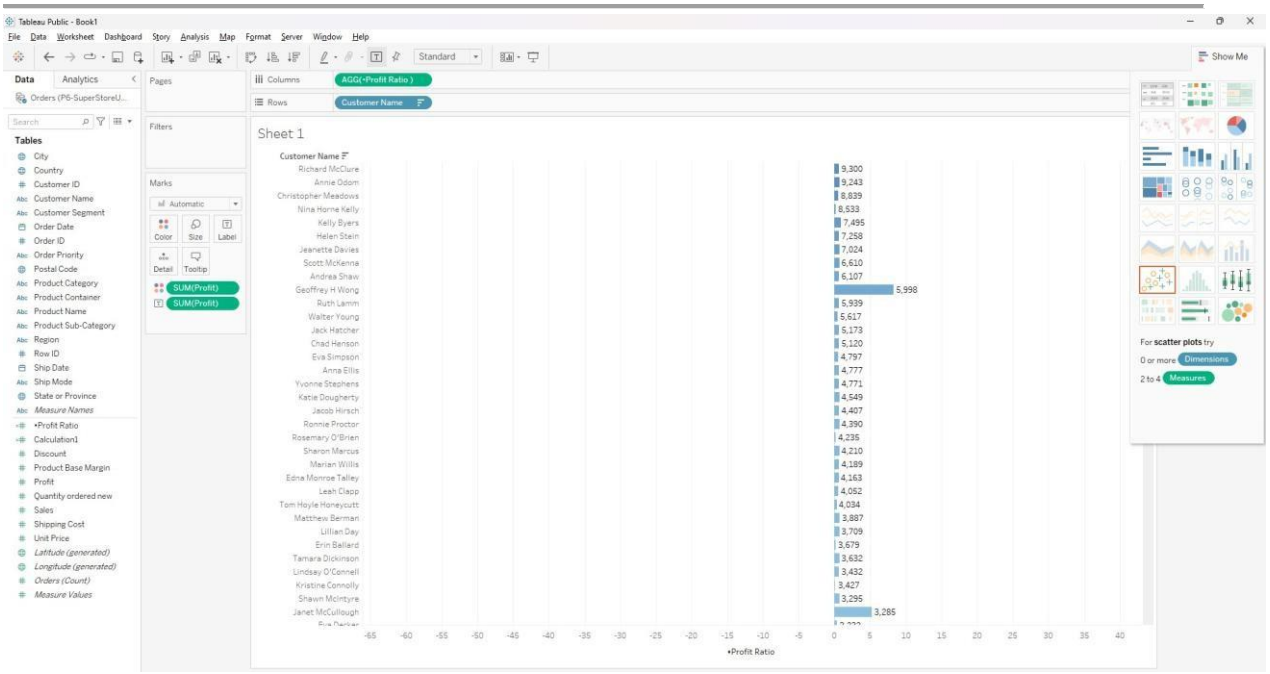
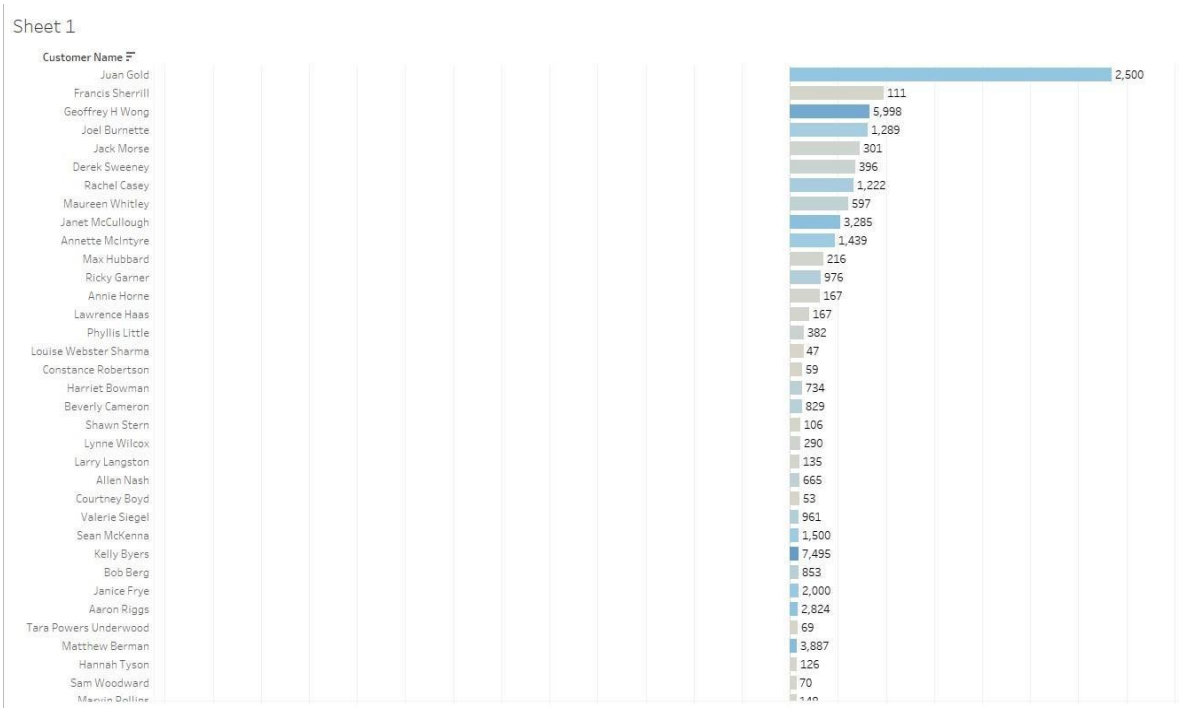


Tableau Public - Book1
Sheet 1
Customer Name
Richard McClure
Annie Odom
Christopher Meadows
Nina Horne Kelly
Kelly Byers
Helen Stein
Jeanette Davies
Scott McKenna
Andrea Shaw
Geoffrey H Wong
Ruth Lamm
Walter Young
Jack Hatcher
Chad Henson
Eva Simpson
Anna Ellis
Yvonne Stephens
Katie Dougherty
Jacob Hirsch
Ronnie Proctor
Rosemary O'Brien
Sharon Marcus
Marian Willis
Edna Monroe Talley
Leah Clapp
Tom Hoyle Honeycutt
Matthew Berman
Lillian Day
Erin Ballard
Tamara Dickinson
Lindsay O'Connell
Kristine Connolly
Shawn McIntyre
Janet McCullough
9,300
9,243
8,839
8,533
7,495
7,258
7,024
6,610
6,107
5,998
5,939
5,617
5,173
5,120
4,797
4,777
4,771
4,549
4,407
4,390
4,235
4,210
4,189
4,163
4,052
4,034
3,887
3,709
3,679
3,632
3,432
3,427
3,295
3,285
•Profit Ratio


Sheet 1
Customer Name
Juan Gold 2,500
Francis Sherrill 111
Geoffrey H Wong 5,998
Joel Burnette 1,289
Jack Morse 301
Derek Sweeney 396
Rachel Casey 1,222
Maureen Whitley 597
Janet McCullough 3,285
Annette McIntyre 1,439
Max Hubbard 216
Ricky Garner 976
Annie Horne 167
Lawrence Haas 167
Phyllis Little 382
Louise Webster Sharma 47
Constance Robertson 59
Harriet Bowman 734
Beverly Cameron 829
Shawn Stern 106
Lynne Wilcox 290
Larry Langston 135
Allen Nash 665
Courtney Boyd 53
Valerie Siegel 961
Sean McKenna 1,500
Kelly Byers 7,495
Bob Berg 853
Janice Frye 2,000
Aaron Riggs 2,824
Tara Powers Underwood 69
Matthew Berman 3,887
Hannah Tyson 126
Sam Woodward 70

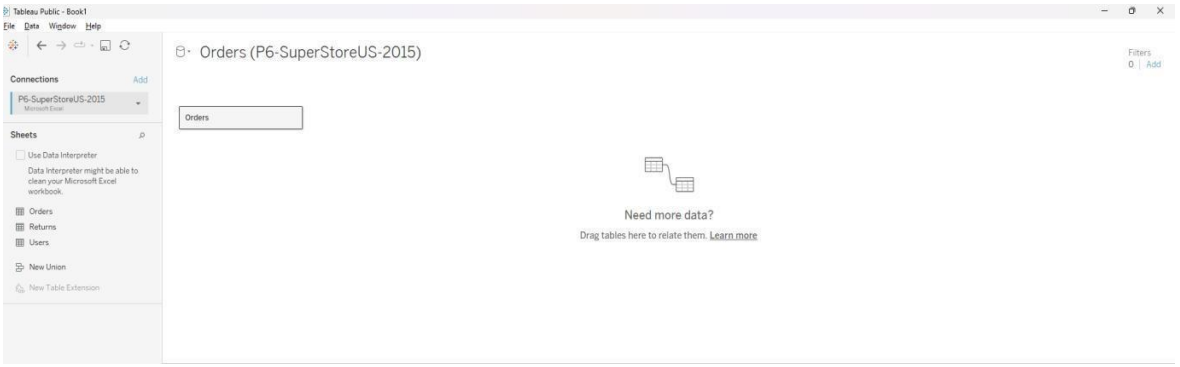
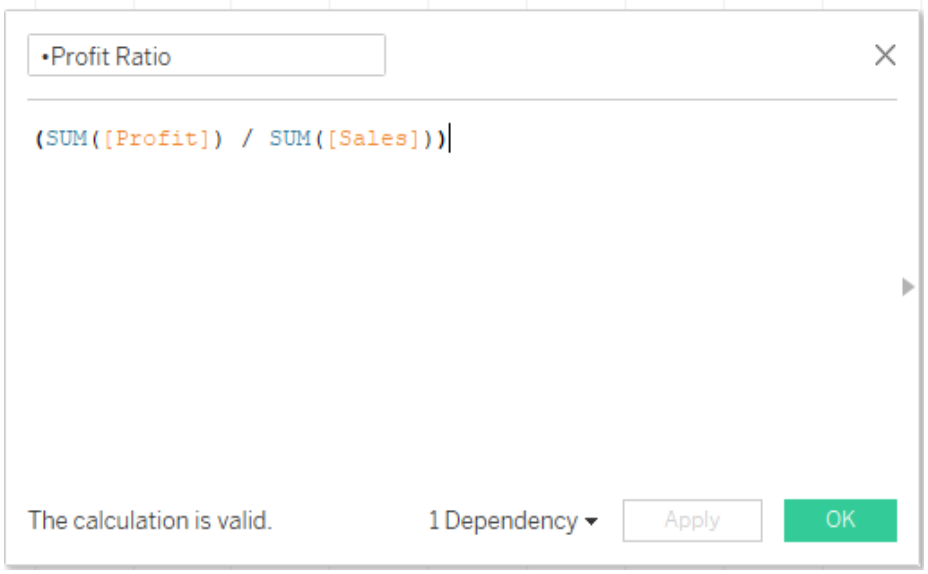
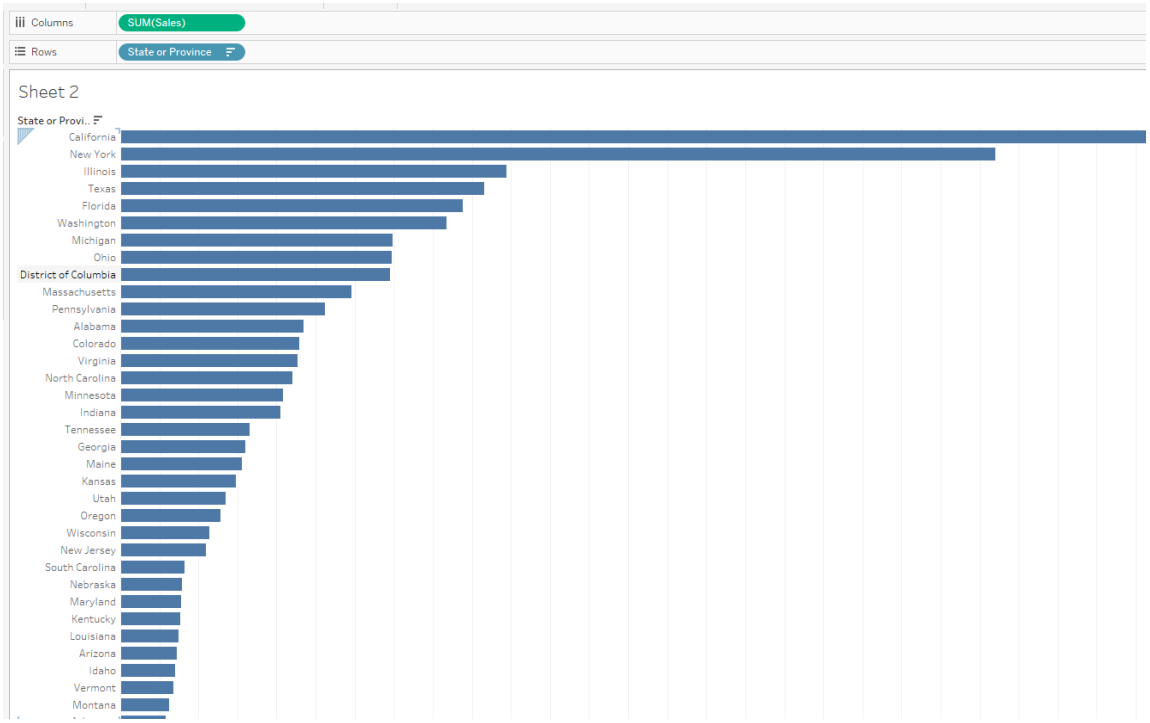


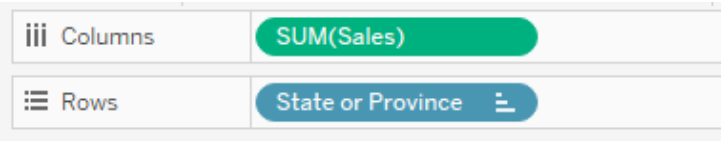
Tableau Public - Book1
File Data Window Help
Connections
Add
P6-SuperStoreUS-2015
Sheets
Use Data Interpreter
Data Interpreter might be able to clean your Microsoft Excel workbook.
Orders
Returns
Users
New Union
New Table Extension
Orders (P6-SuperStoreUS-2015)
Filters
0 | Add
Orders
Need more data?
Drag tables here to relate them. Learn more


•Profit Ratio
(SUM([Profit]) / SUM([Sales]))
The calculation is valid.
1 Dependency
Apply
OK

Q2. Which state has the highest Sales (Sum)? What is the total Sales for that state?

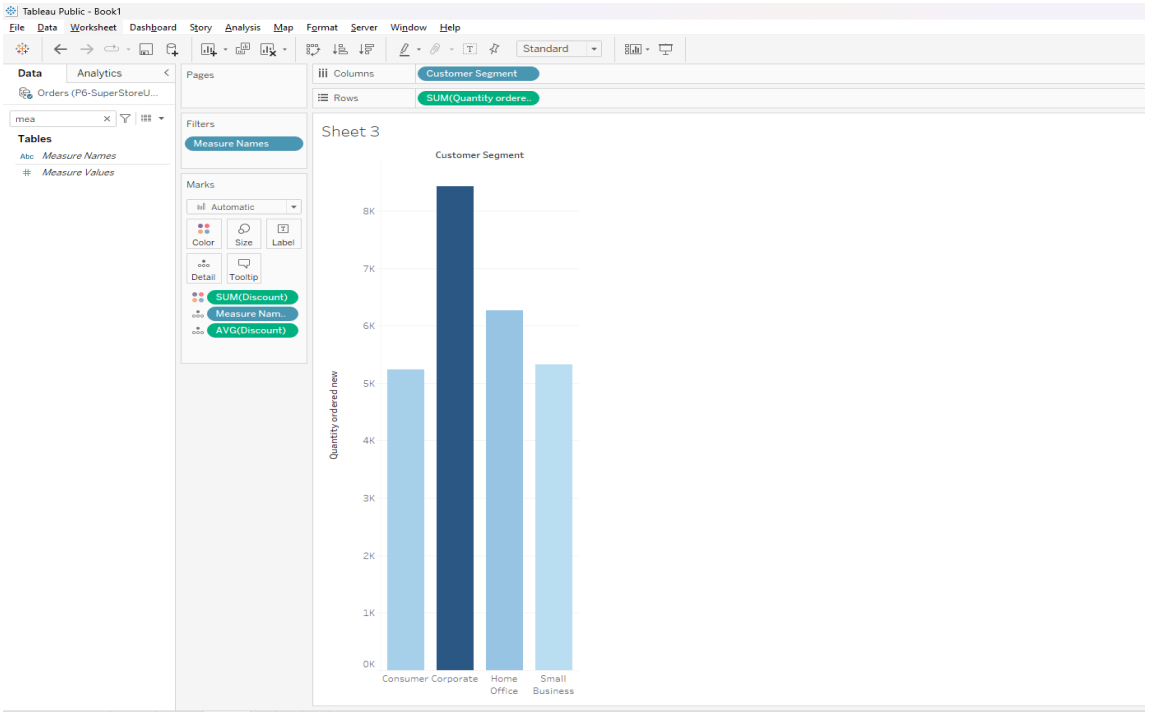
- Drag State to Rows, Sales to Columns.
- Sort descending.

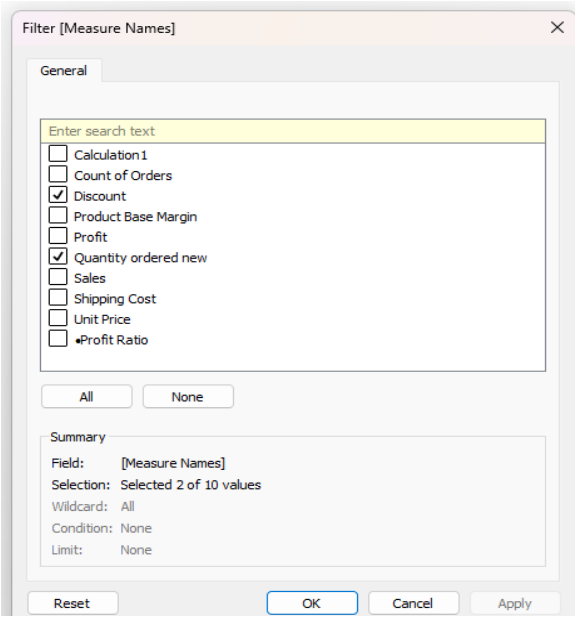




Q3. Which customer segment has both the highest order quantity and average discount rate?

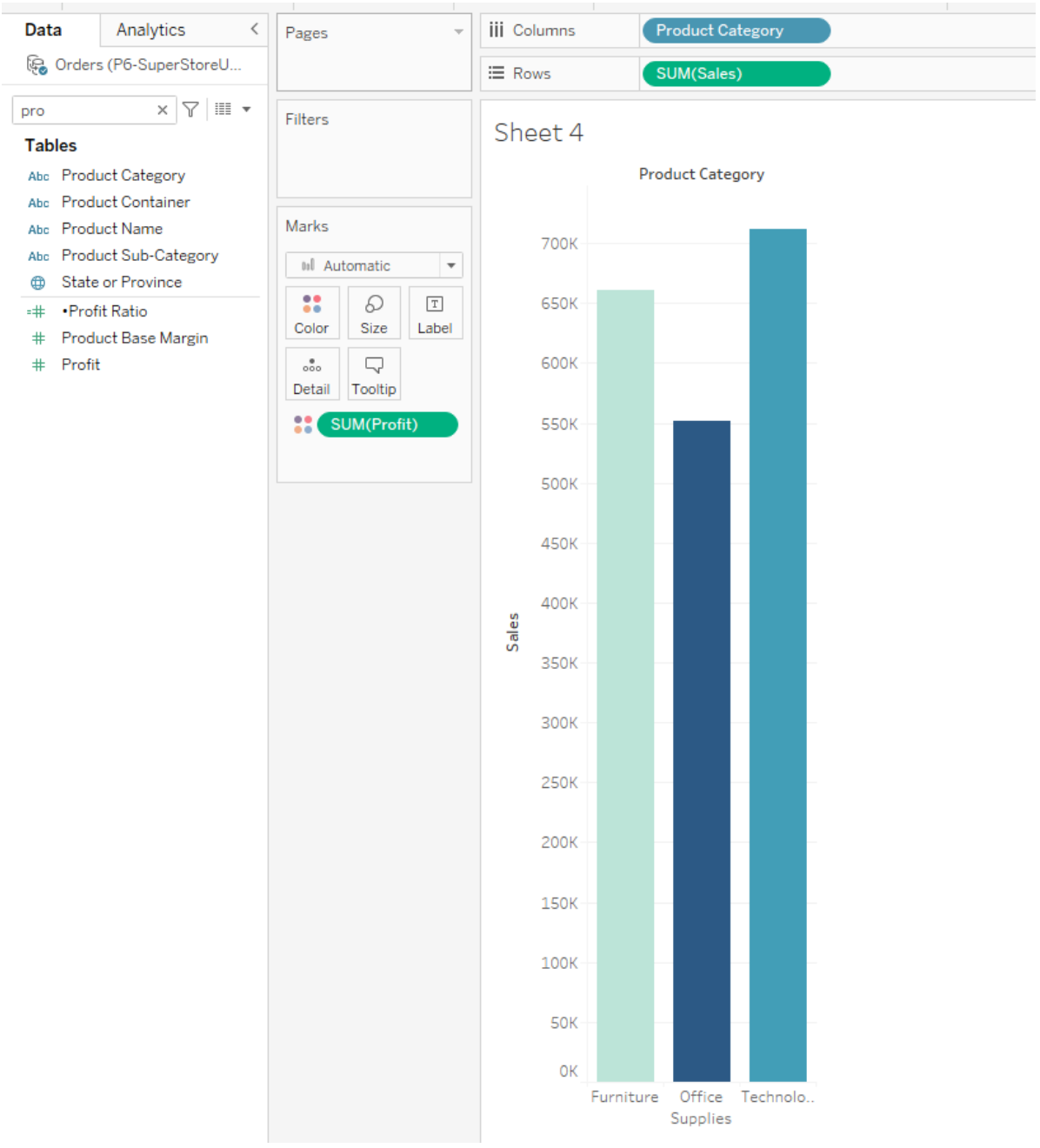
- Drag Segment to Rows.
- Add Order Quantity and Discount to Measures.
- Analyze visually.





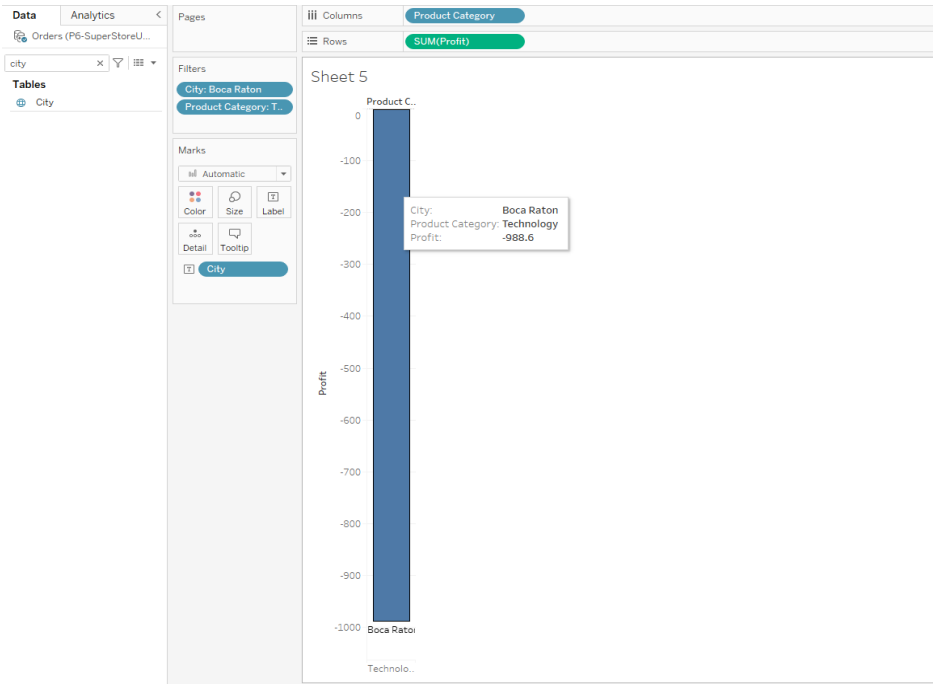
Q4. Which Product Category has the highest total Sales? Which Product Category has the worst Profit?

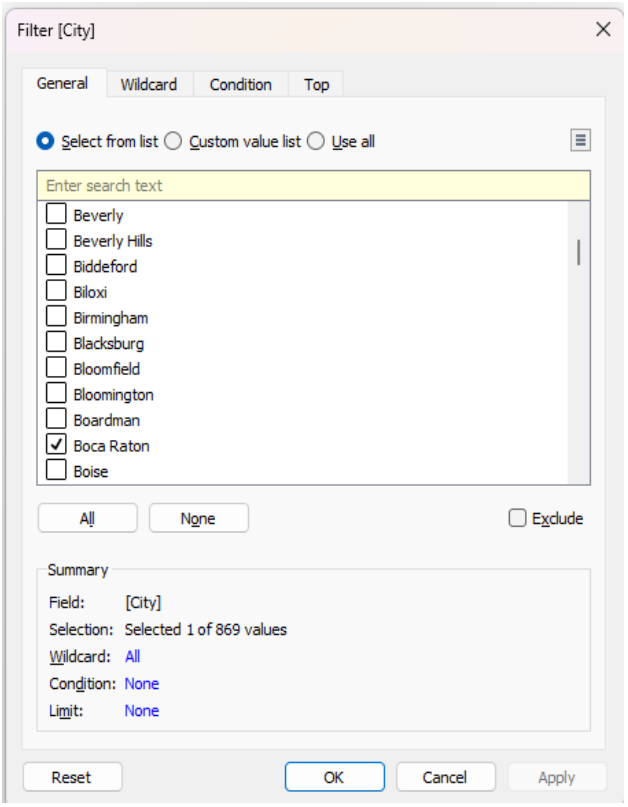
- Create a Bar Chart with Category vs Sales.
- Add Profit as color scale.

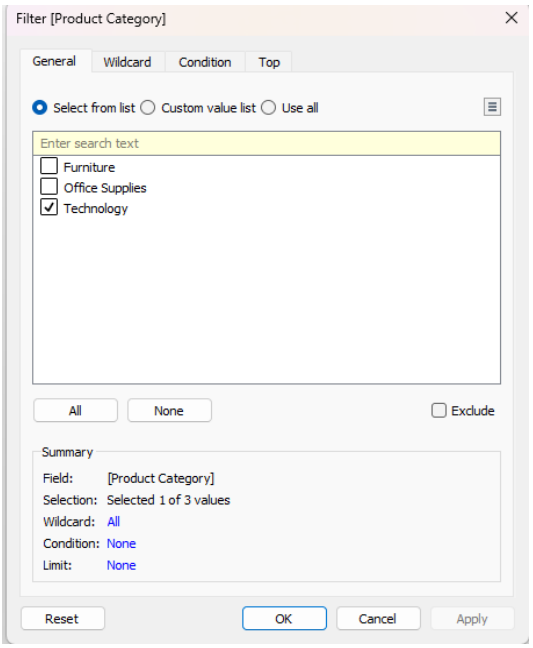


Q5. What was the Profit on Technology in Boca Raton (City)?

- Add City filter.
- Filter for "Boca Raton".
- Show Profit.

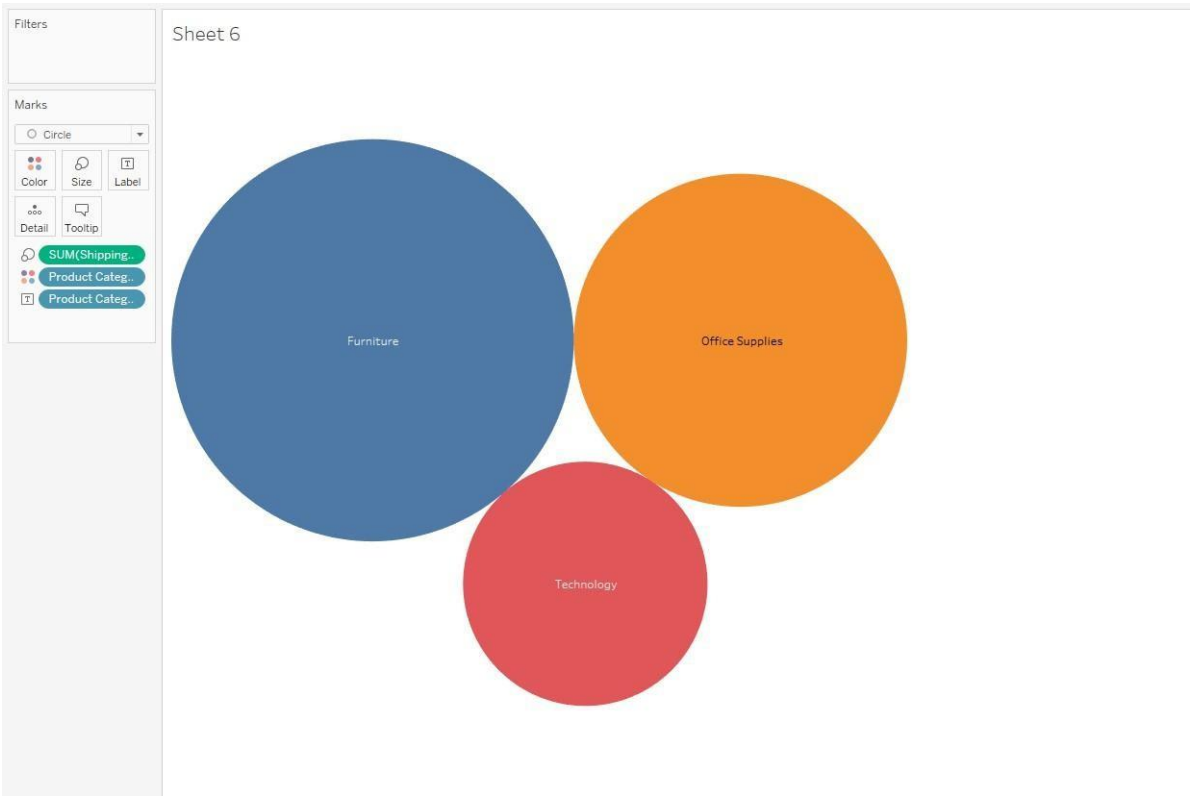






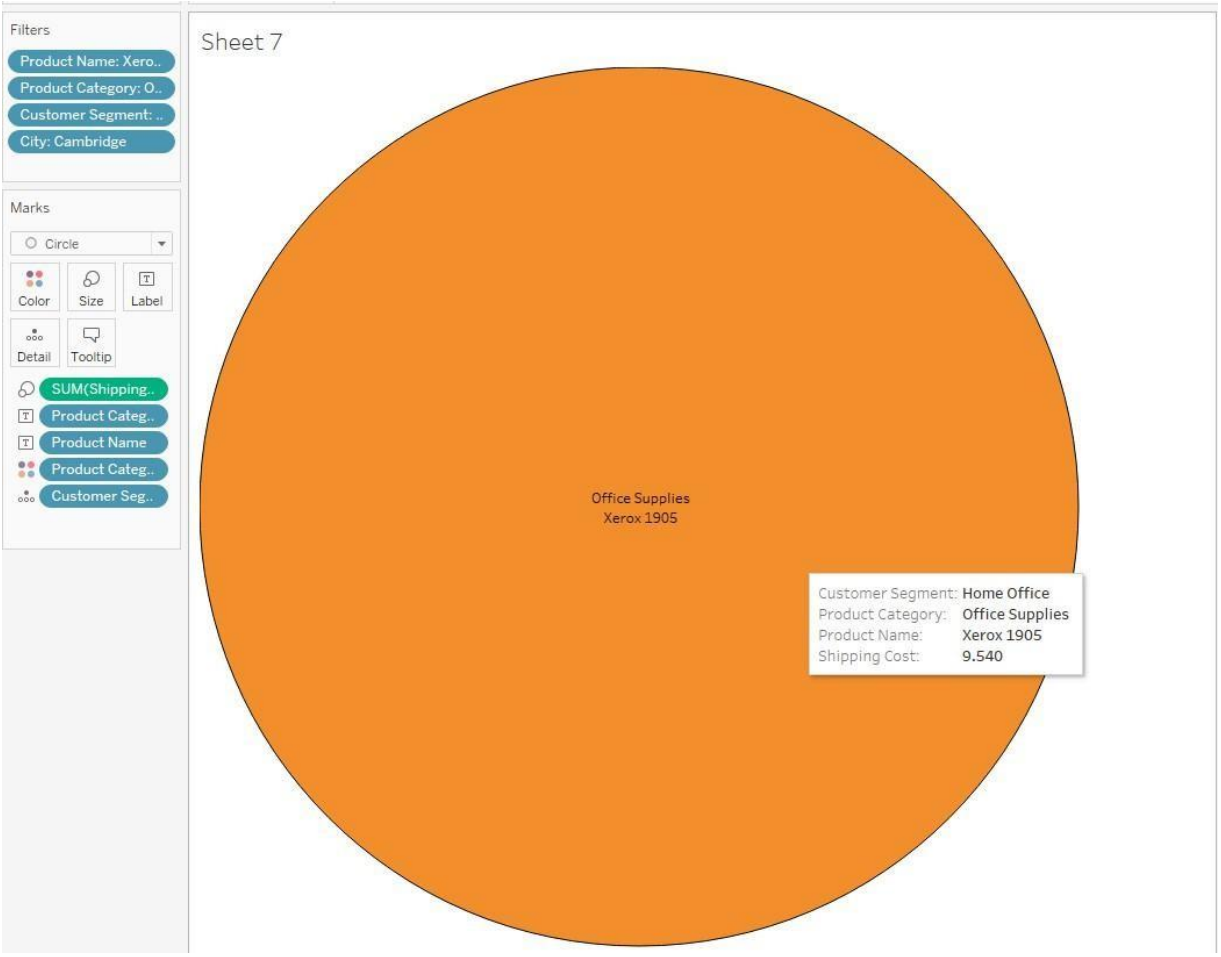
Q6. Which Product Department has the highest Shipping Costs?

- Create **Packed Bubble Chart** with Department.
- Use Shipping Cost as bubble size.



Q7. What was the shipping cost of Office Supplies for Xerox 1905 in the Home Customer Segment in Cambridge?

- Apply filters: Product → Xerox 1905, Category → Office Supplies, Segment → Home, City
  → Cambridge.



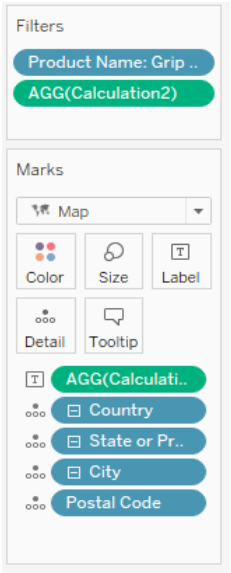
Preparing Maps

Dataset: **Superstore**

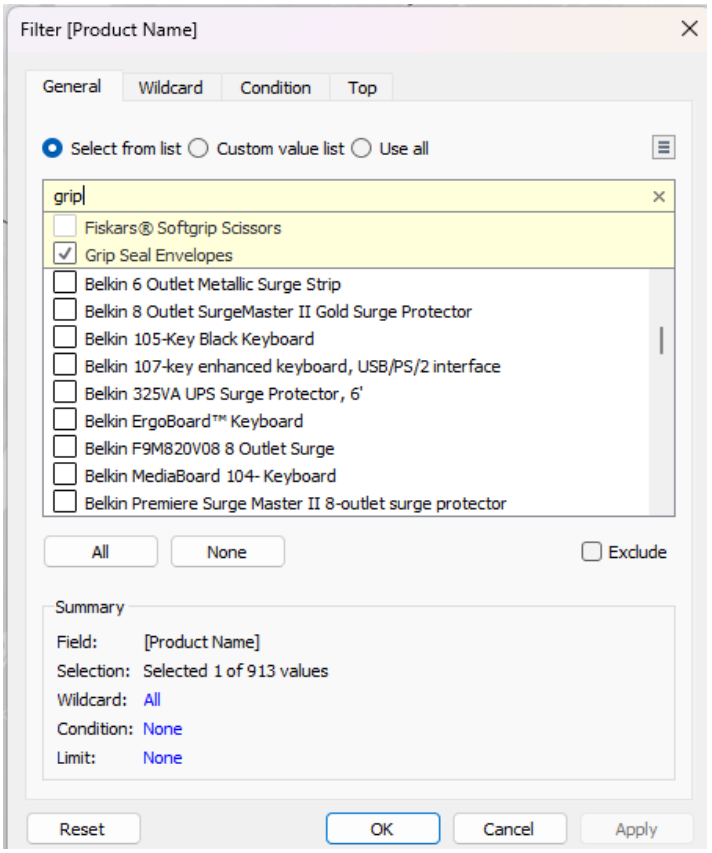
1. Create Geographic Hierarchy (Country → State → City → Postal Code)

1. In the **Data Pane**, right-click on Country→ select **Hierarchy** → **Create Hierarchy**.
2. Name it (e.g., *Geography*).
3. Drag State, City, and Postal Codeinto the hierarchy (below Country in order).
   - Now you can drill down from Country → State → City → Postal Code.


-1.0887
-0.0903


Filters
Product Name: Grip ..
AGG(Calculation2)
Marks
Map
Color
Size
Label
Detail
Tooltip
AGG(Calculati..
Country
State or Pr..
City
Postal Code

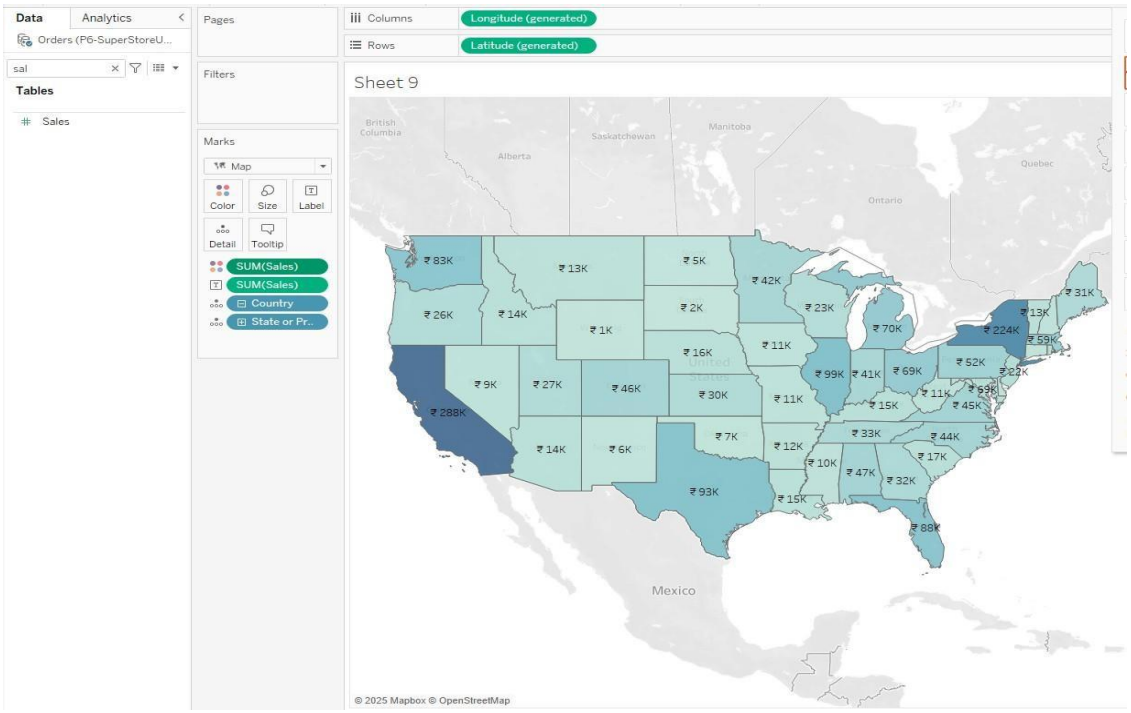

Calculation2
(SUM([Profit])/SUM([Sales]))
The calculation is valid.
1 Dependency
Apply
OK

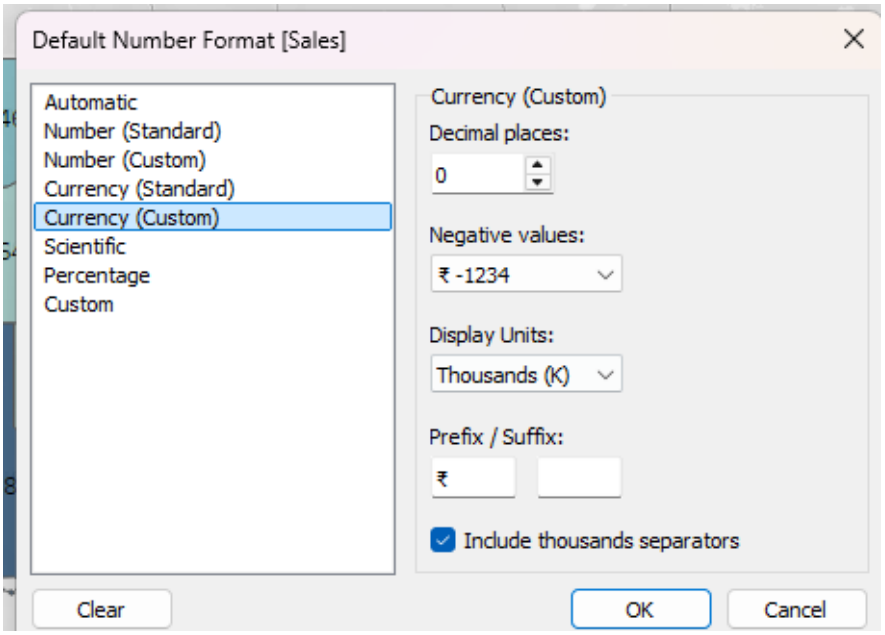


---

2. Build a Basic Map of Sales by State

1. Double-click on State→ Tableau will create a map.
2. From **Measures**, drag Sales→ drop it on **Color** (Marks card).
   - Now each state is colored by its Sales.
3. Drag Salesagain → drop it on **Label** (so values appear on the map).
4. Right-click on Sales→ **Default Properties** → **Number Format** → **Currency / Custom (No decimals, in K)**.


Data
Analytics
Orders (P6-SuperStoreU...
sal
Tables
Sales
Pages
Filters
Marks
Map
Color
Size
Label
Detail
Tooltip
SUM(Sales)
SUM(Sales)
Country
State or Pr..
Columns
Longitude (generated)
Rows
Latitude (generated)
Sheet 9
British Columbia
Alberta
Saskatchewan
Manitoba
Ontario
Quebec
Mexico
₹ 83K
₹ 26K
₹ 13K
₹ 14K
₹ 1K
₹ 5K
₹ 2K
₹ 16K
₹ 42K
₹ 23K
₹ 11K
₹ 9K
₹ 27K
₹ 46K
₹ 30K
₹ 11K
₹ 288K
₹ 14K
₹ 6K
₹ 7K
₹ 12K
₹ 93K
₹ 15K
₹ 10K
₹ 47K
₹ 32K
₹ 88K
₹ 99K
₹ 41K
₹ 69K
₹ 70K
₹ 52K
₹ 22K
₹ 11K
₹ 15K
₹ 45K
₹ 33K
₹ 44K
₹ 17K
₹ 224K
₹ 13K
₹ 59K
₹ 31K
© 2025 Mapbox © OpenStreetMap

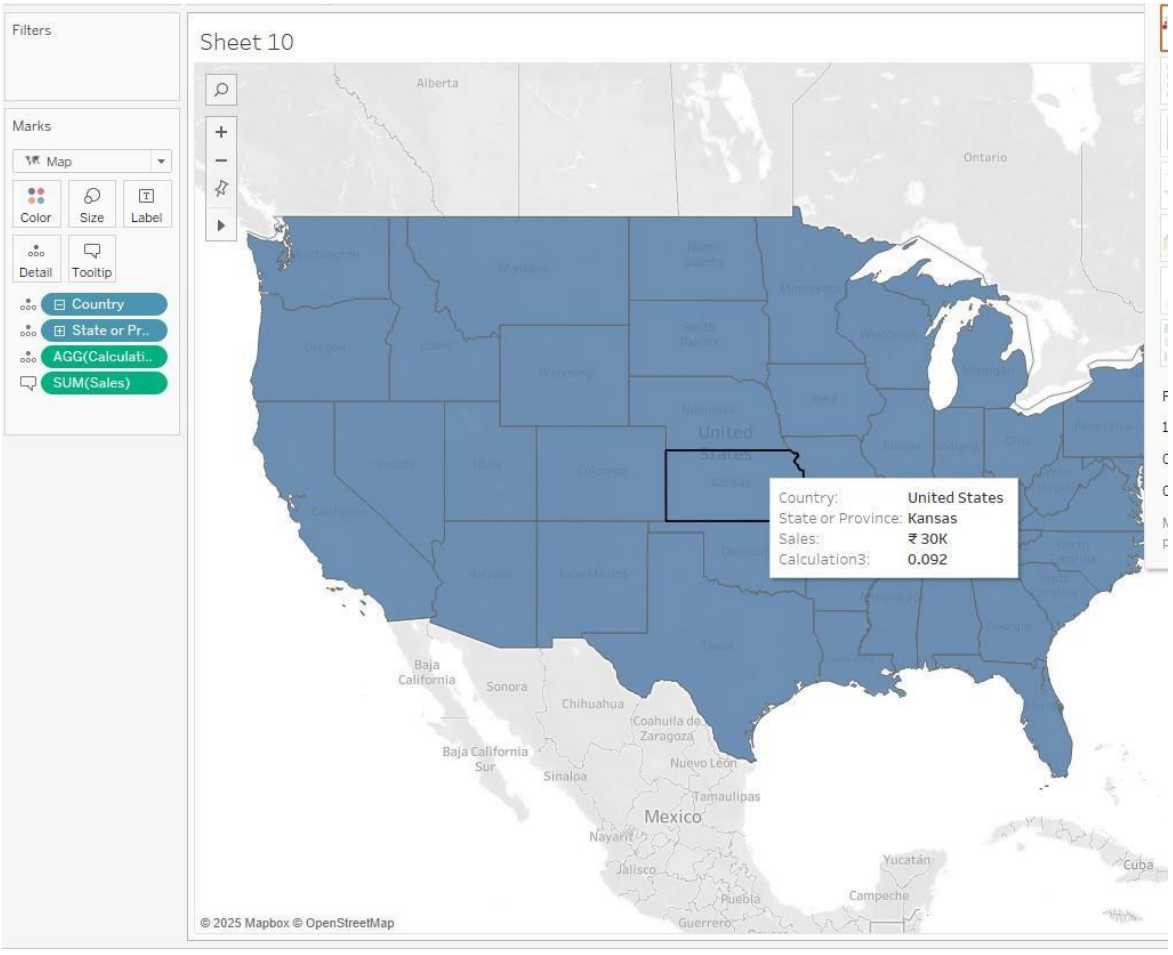

Default Number Format [Sales]
Automatic
Number (Standard)
Number (Custom)
Currency (Standard)
Currency (Custom)
Scientific
Percentage
Custom
Currency (Custom)
Decimal places:
0
Negative values:
₹ -1234
Display Units:
Thousands (K)
Prefix / Suffix:
₹
Include thousands separators
Clear
OK
Cancel

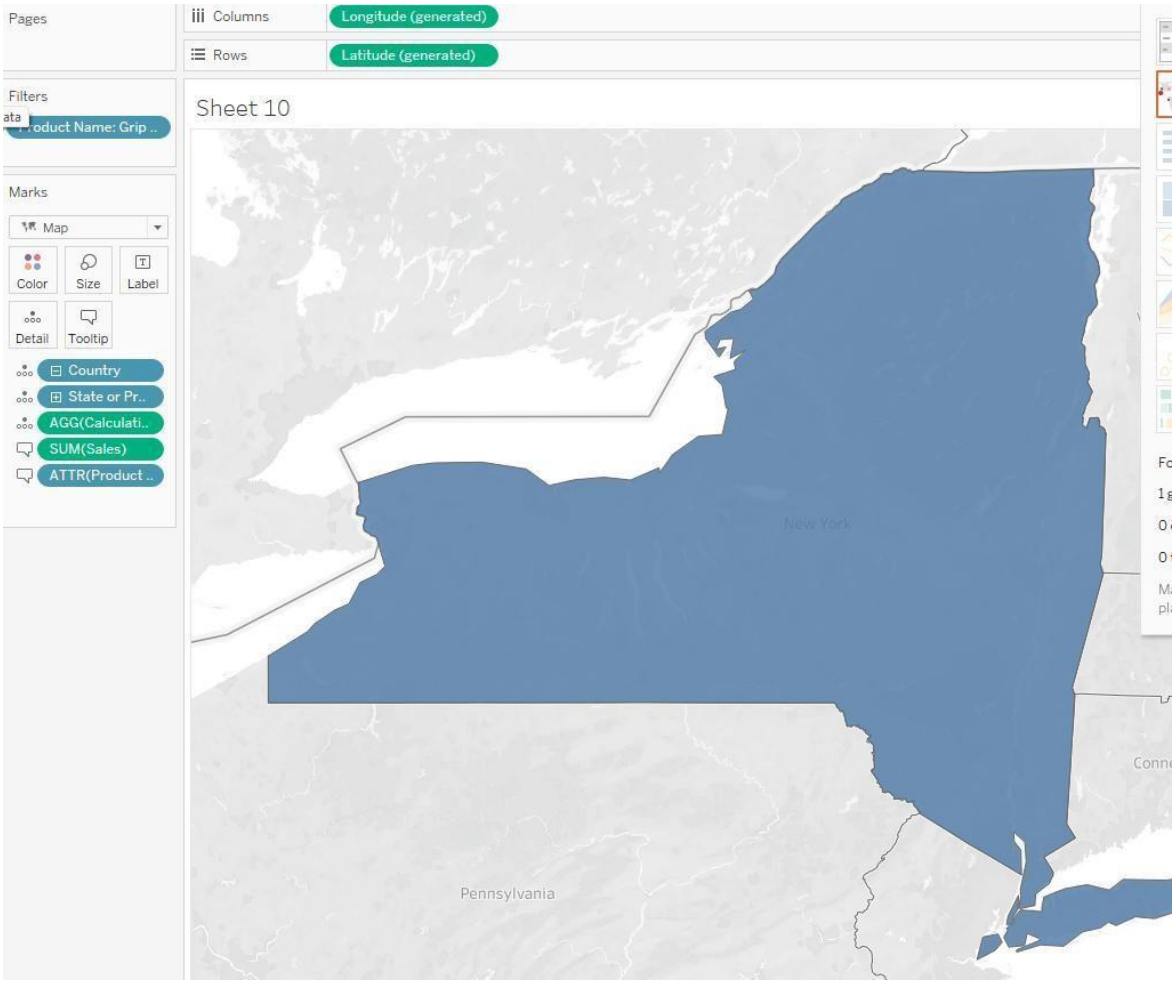
3. Add Profit Ratio as Tooltip

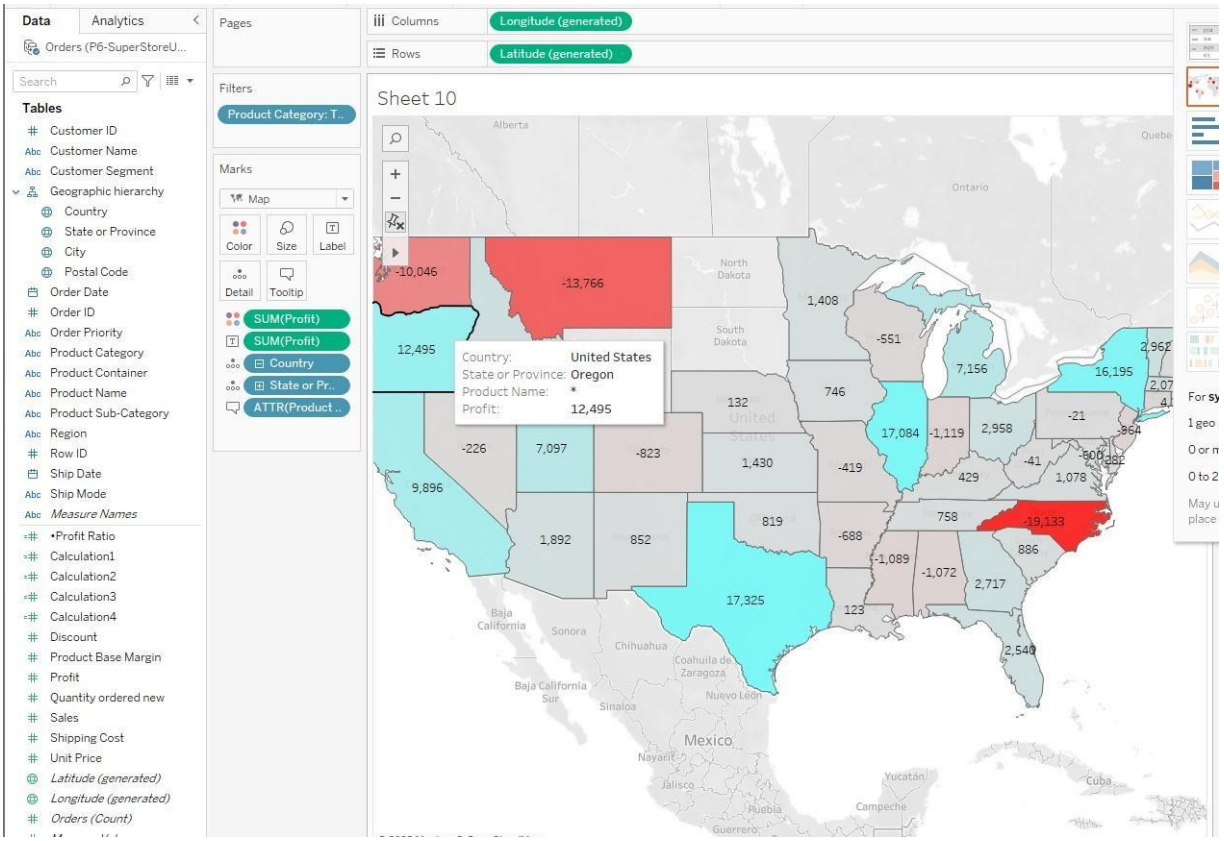
1. Create a calculated field:
2. Profit Ratio = SUM([Profit]) / SUM([Sales])
3. Drag this field onto the **Tooltip** shelf (Marks card).
4. Now when you hover over a state, you will see both Sales and Profit Ratio.

- Show Profit Ratio of each state as tooltip on map
- Show Profit ratio for Grip Envelop products
- Identify Unprofitable States in Technology Surrounded by Profitable Ones (e.g., Nevada)

1. Drag Categoryto **Filters** → select only **Technology**.
2. Drag Profitto **Color** (Marks card).
   - Positive profit → blue/green, Negative profit → red.
3. Add Profitalso to **Label** if you want exact values.
4. Look at the map → Nevada will appear **red (loss)** but surrounded by **green states (profit)**.




Pages
Columns
Longitude (generated)
Rows
Latitude (generated)
Filters
Product Name: Grip ..
Marks
Map
Color
Size
Label
Detail
Tooltip
Country
State or Pr..
AGG(Calculati..
SUM(Sales)
ATTR(Product ..
Sheet 10
New York
Pennsylvania

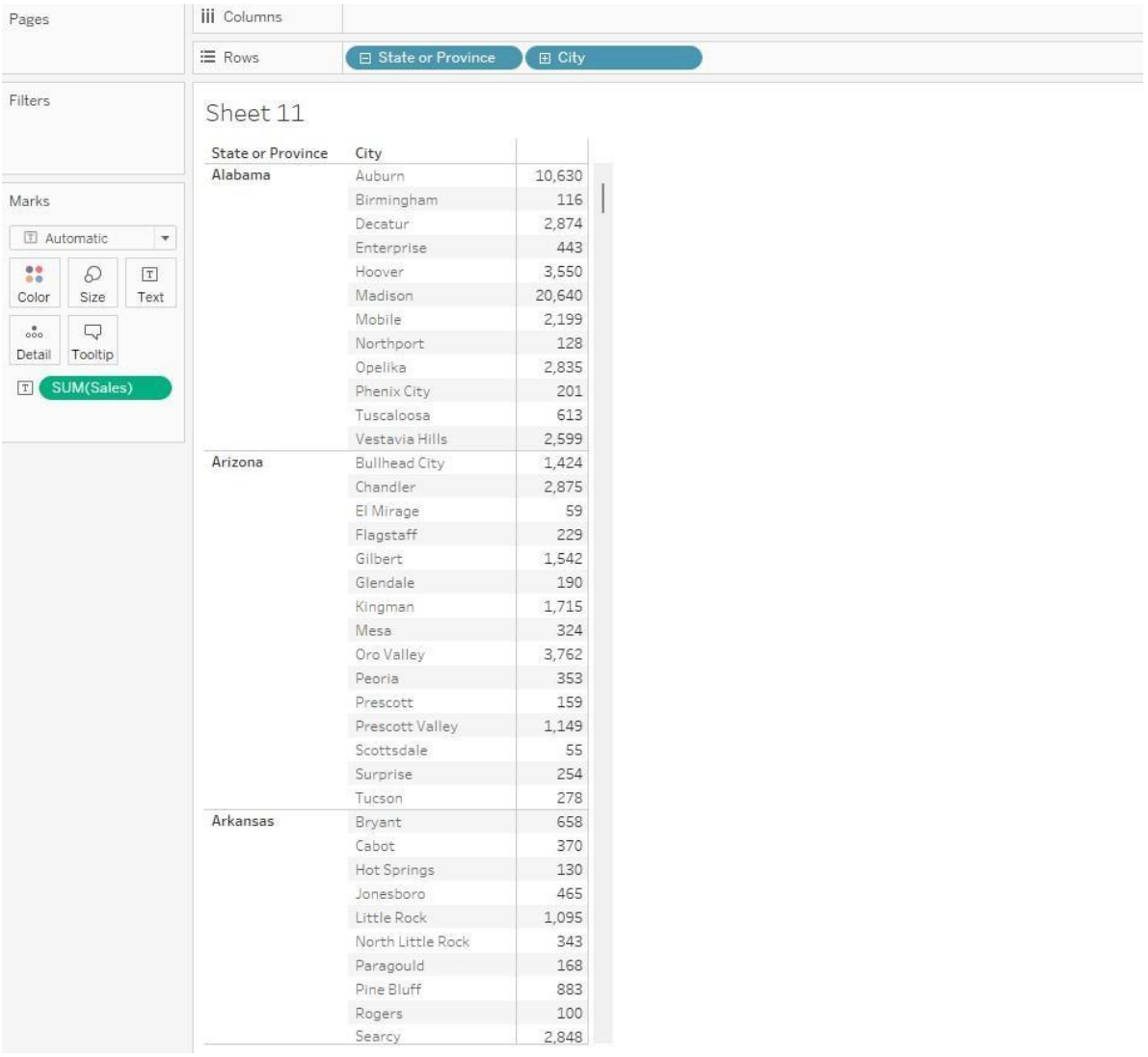


Preparing Reports

Dataset: **Superstore**

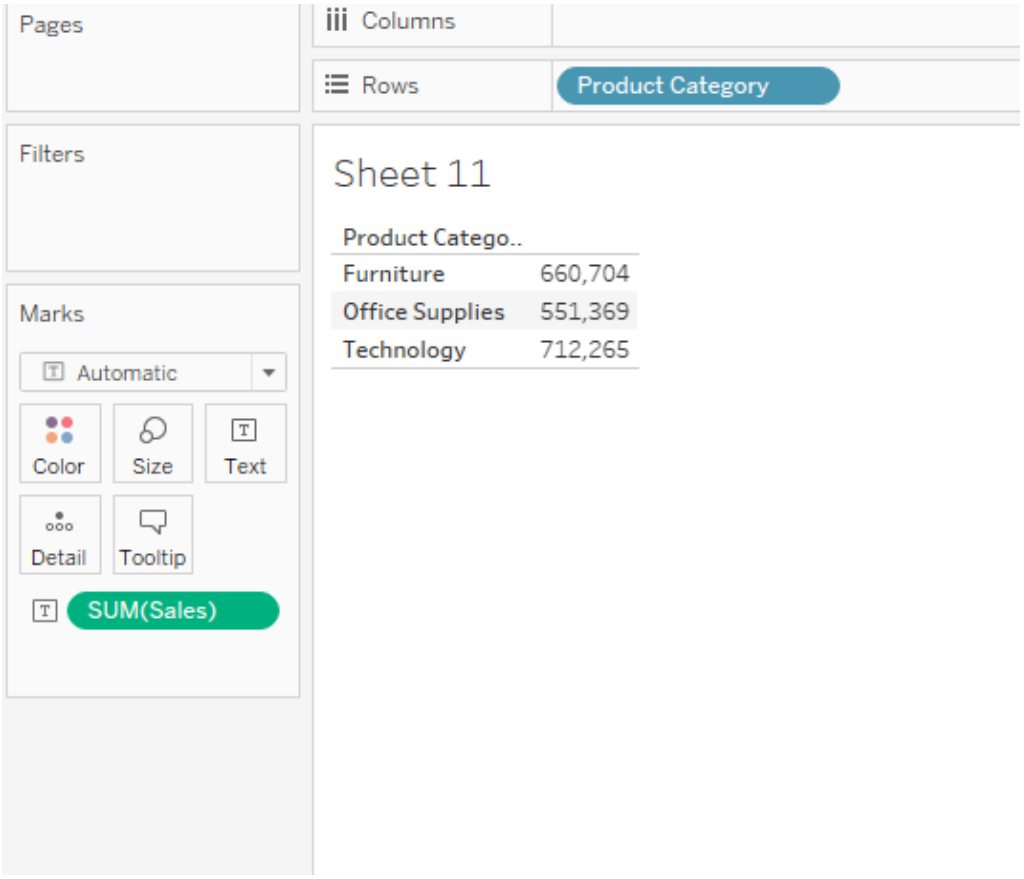
- Report: Product category-wise sales.
- Report: Region-wise product sales.
- Report: State-wise sales.
- Example Question: *What is % of total Sales for Home Office segment in 2015?*
- Example Question: *Find Top 10 Product Names by Sales in each region. Which product is ranked#2 in Central & West in 2015?*

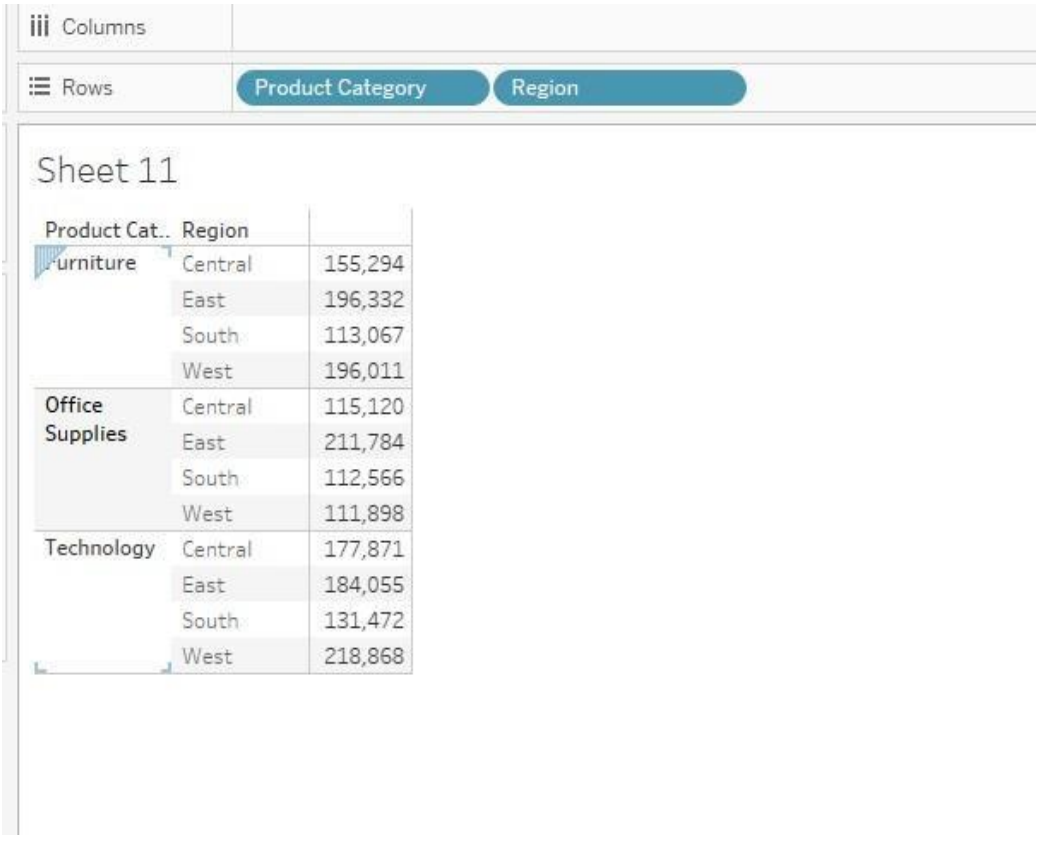
Conclusion: Successfully performed Visualization in Tableau with charts, dashboards, stories, maps, and reports.



Sheet 11

State or Province	City	
Alabama	Auburn	10,630
	Birmingham	116
	Decatur	2,874
	Enterprise	443
	Hoover	3,550
	Madison	20,640
	Mobile	2,199
	Northport	128
	Opelika	2,835
	Phenix City	201
	Tuscaloosa	613
	Vestavia Hills	2,599
Arizona	Bullhead City	1,424
	Chandler	2,875
	El Mirage	59
	Flagstaff	229
	Gilbert	1,542
	Glendale	190
	Kingman	1,715
	Mesa	324
	Oro Valley	3,762
	Peoria	353
	Prescott	159
	Prescott Valley	1,149
	Scottsdale	55
	Surprise	254
	Tucson	278
Arkansas	Bryant	658
	Cabot	370
	Hot Springs	130
	Jonesboro	465
	Little Rock	1,095
	North Little Rock	343
	Paragould	168
	Pine Bluff	883
	Rogers	100
	Searcy	2,848


Pages
Columns
Rows
Product Category
Filters
Sheet 11
Product Catego..
Furniture 660,704
Office Supplies 551,369
Technology 712,265
Marks
Automatic
Color
Size
Text
Detail
Tooltip
SUM(Sales)


Columns
Rows
Product Category
Region
Sheet 11
Product Cat.. Region
urniture Central 155,294
East 196,332
South 113,067
West 196,011
Office Supplies Central 115,120
East 211,784
South 112,566
West 111,898
Technology Central 177,871
East 184,055
South 131,472
West 218,868

Sheet 11

Product Cat..	Region	State or Province	
Furniture	Central	Illinois	20,376
		Indiana	15,094
		Iowa	2,342
		Kansas	17,074
		Michigan	24,884
		Minnesota	25,998
		Missouri	1,907
		Nebraska	5,835
		North Dakota	68
		Oklahoma	3,401
		South Dakota	131
		Texas	33,026
		Wisconsin	5,160
	East	Connecticut	3,664
		District of Columbia	9,595
		Maine	16,230
		Maryland	8,796
		Massachusetts	16,248
		New Hampshire	4,458
		New Jersey	10,933
		New York	72,852
		Ohio	20,665
		Pennsylvania	19,850
		Vermont	3,116
		West Virginia	9,928
	South	Alabama	4,547
		Arkansas	3,112
		Florida	46,019
		Georgia	12,053
		Kentucky	5,888
		Louisiana	2,860
		Mississippi	2,795
		North Carolina	11,657
		South Carolina	3,106
		Tennessee	7,146
		Virginia	13,883
	West	Arizona	8,055