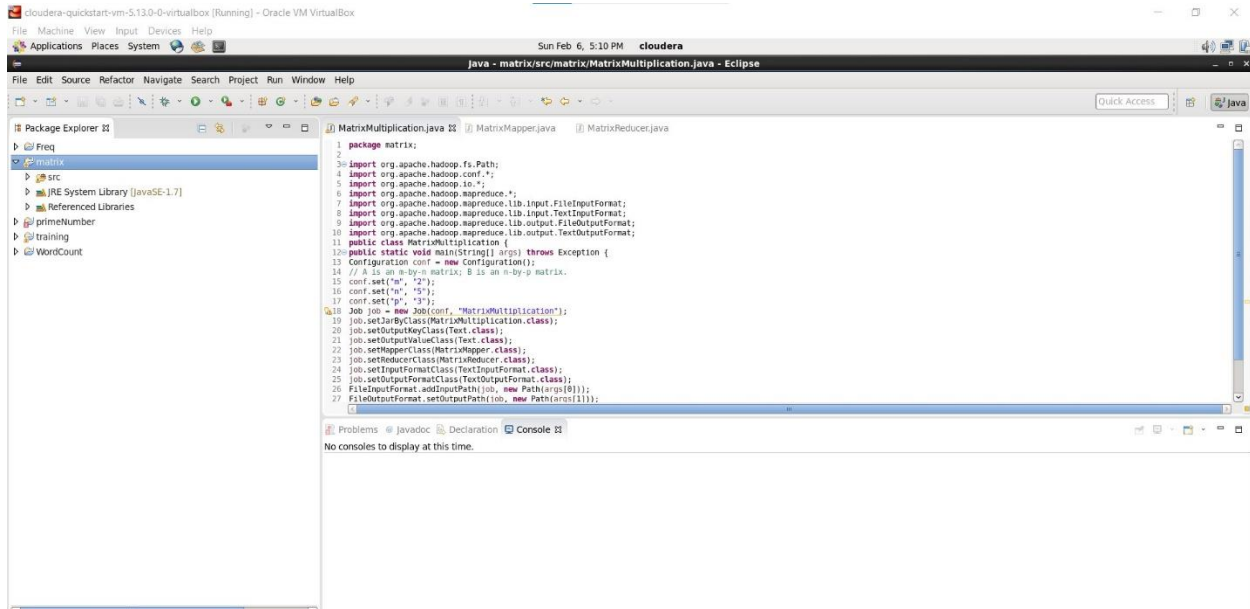
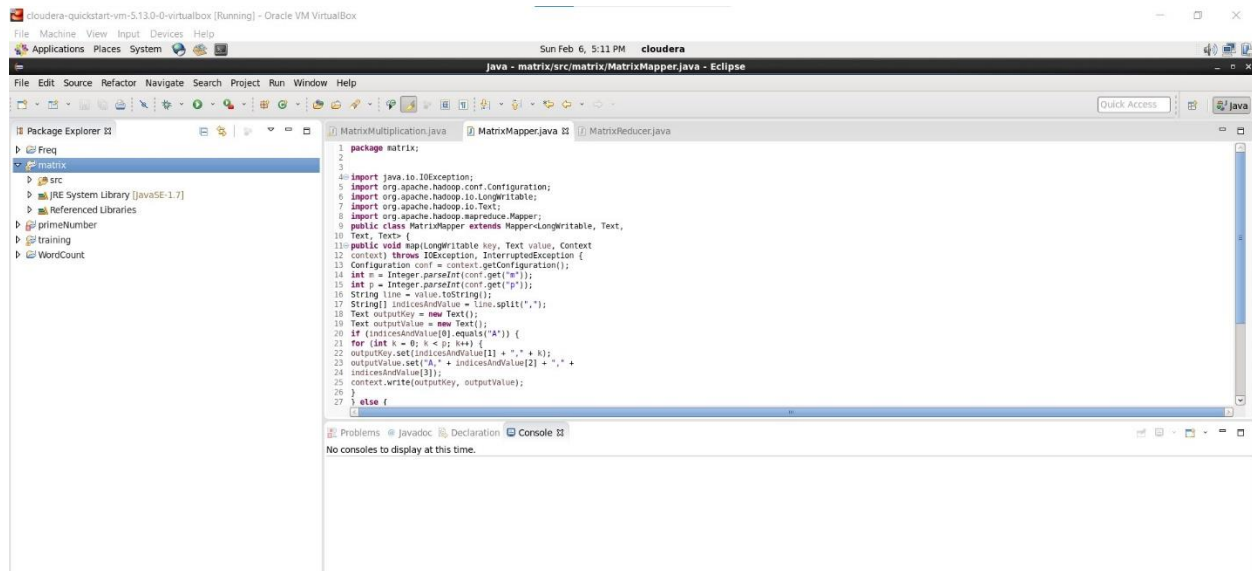


ICE-3

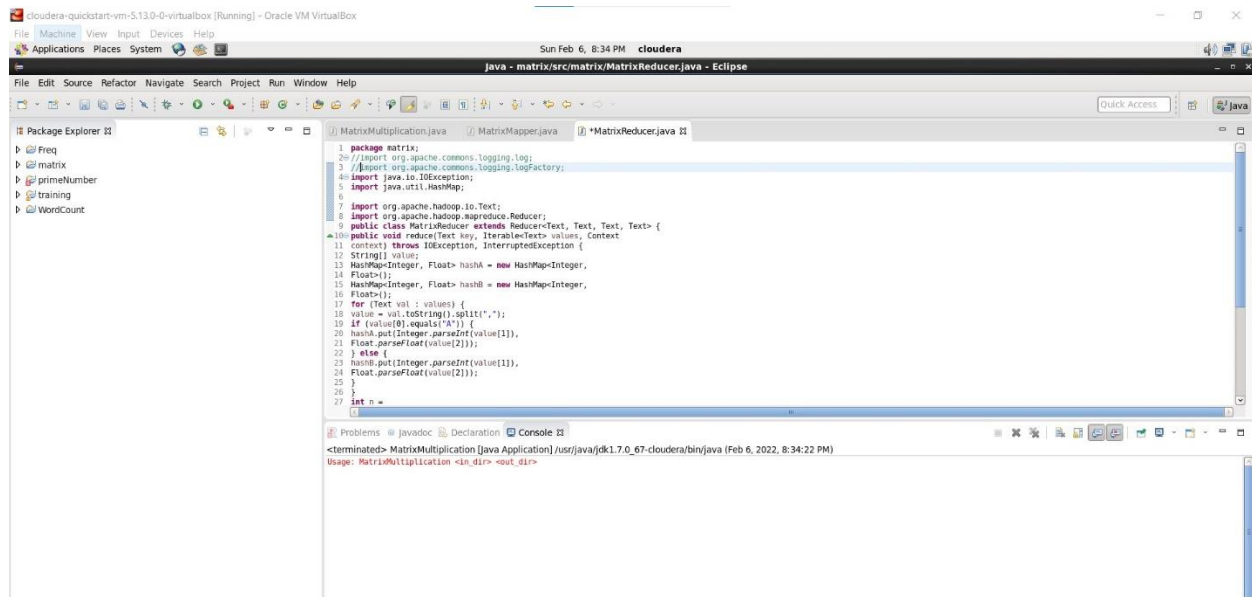
Matrix Multiplication using Mapreduce



- Firstly I have created a project named matrix and imported all the required external libraries.
- Then I have created a class named MatrixMultiplication and defined the configuration for the matrices.
- Then I have set the jobs for all classes like Mapper, Reducer, TextInput, TextOutput and also set the paths for input and output.

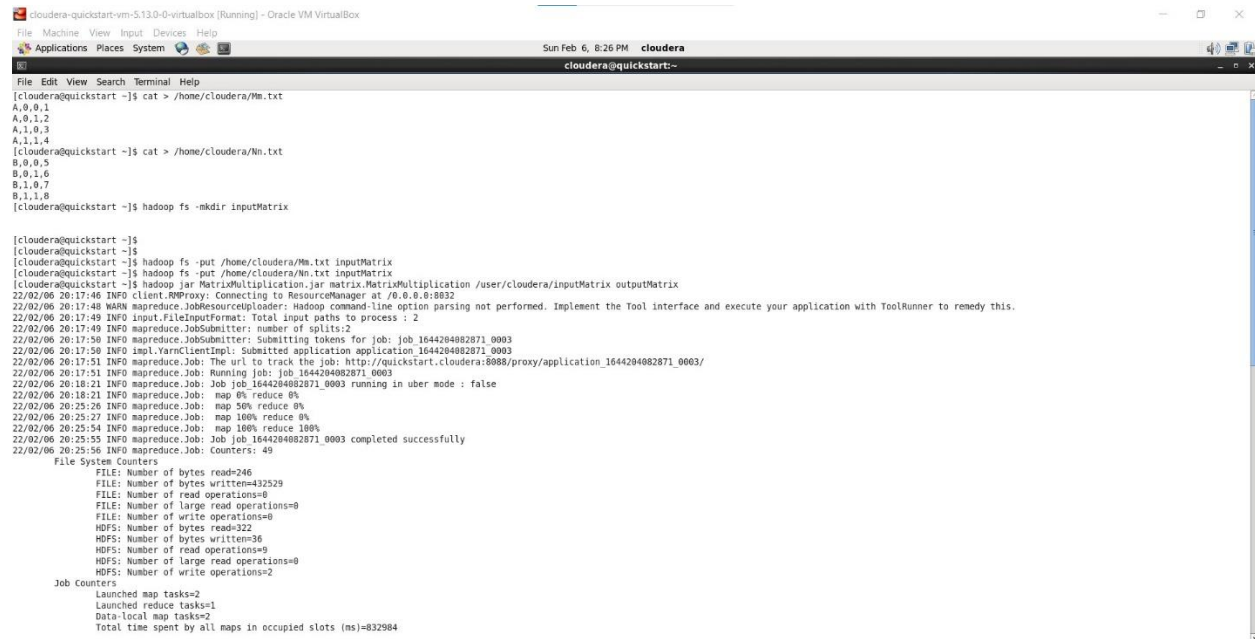


- Next I created mapper class named MatrixMapper which will extend the mapper class.
- We defined A as a m by n matrix and B as a n by p matrix.
- Then we defined map function which produces key value pairs for each element a_{ij} of A matrix. The key value pairs are produced in form of $(i,k),(A,j, a_{ij})$ where k depends on the columns of B matrix. i and j are referred as rows and columns and (A,j, a_{ij}) is the value.
- Then we defined map function which produces key value pairs for each element b_{ij} of B matrix. The key value pairs are produced in form of $(i,k),(B,j, b_{ij})$ where k depends on the rows of A matrix. i and j are referred as rows and columns and (B,j, b_{ij}) is the value.
- The map function finally returns all key values that each key (i,k) has a combination of values of A and B matrices. This output is fed as input to the reducer class which performs multiplication operation,



```
1 package matrix;
2 //import org.apache.commons.logging.Log;
3 //import org.apache.commons.logging.LogFactory;
4 import java.io.IOException;
5 import java.util.HashMap;
6
7 import org.apache.hadoop.io.Text;
8 import org.apache.hadoop.mapreduce.Reducer;
9 public class MatrixReducer extends Reducer<Text, Text, Text, Text> {
10 public void reduce(Text key, Iterable<Text> values, Context
11 context) throws IOException, InterruptedException {
12 String[] value;
13 HashMap<Integer, Float> hashA = new HashMap<Integer,
14 Float>();
15 HashMap<Integer, Float> hashB = new HashMap<Integer,
16 Float>();
17 for (Text val : values) {
18 value = val.toString().split(",");
19 if (value[0].equals("A")) {
20 hashA.put(Integer.parseInt(value[1]),
21 Float.parseFloat(value[2]));
22 } else {
23 hashB.put(Integer.parseInt(value[1]),
24 Float.parseFloat(value[2]));
25 }
26 }
27 int n =
```

- Then we define reducer class as MatrixReducer and then we define hashmap for storing key-value pairs.
- The reducer function takes key (i,k) and sorts the values that start with A by j in list_A and sorts values of B matrix in list_B.
- Then we define for loop which iterates and multiplies a(i,j) rows and b(j,k) columns with jth value of both lists and sums up the value and stores in result.
- Finally the reducer function returns result in the form of key value pair where key is (i,k) and value is the sum of product. This gives the final result matrix.



```

cloudera-quickstart-vm-5130-0-virtualbox (Running) - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Sun Feb 6, 8:26 PM cloudera
cloudera@quickstart:~$
cloudera@quickstart:~$ cat > /home/cloudera/Mm.txt
A,0,0,1
A,0,1,2
A,1,0,3
A,1,1,4
cloudera@quickstart:~$ cat > /home/cloudera/Nn.txt
B,0,0,5
B,0,1,6
B,1,0,7
B,1,1,8
cloudera@quickstart:~$ hadoop fs -mkdir inputMatrix

cloudera@quickstart:~$
cloudera@quickstart:~$
cloudera@quickstart:~$ hadoop fs -put /home/cloudera/Mm.txt inputMatrix
cloudera@quickstart:~$ hadoop fs -put /home/cloudera/Nn.txt inputMatrix
cloudera@quickstart:~$ hadoop jar MatrixMultiplication.jar matrix.MatrixMultiplication /user/cloudera/inputMatrix outputMatrix
22/02/06 20:17:46 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
22/02/06 20:17:48 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
22/02/06 20:17:49 INFO Input.FileInputFormat: Total input paths to process : 2
22/02/06 20:17:49 INFO mapreduce.JobSubmitter: number of splits:2
22/02/06 20:17:50 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1644284082871_0003
22/02/06 20:17:50 INFO impl.YarnClientImpl: Submitted application application_1644284082871_0003
22/02/06 20:17:51 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1644284082871_0003/
22/02/06 20:17:51 INFO mapreduce.Job: Running job: job_1644284082871_0003
22/02/06 20:18:21 INFO mapreduce.Job: Job job_1644284082871_0003 running in uber mode : false
22/02/06 20:18:21 INFO mapreduce.Job:  map 0% reduce 0%
22/02/06 20:25:26 INFO mapreduce.Job:  map 50% reduce 0%
22/02/06 20:25:27 INFO mapreduce.Job:  map 100% reduce 0%
22/02/06 20:25:54 INFO mapreduce.Job:  map 100% reduce 100%
22/02/06 20:25:55 INFO mapreduce.Job: Job job_1644284082871_0003 completed successfully
22/02/06 20:25:56 INFO mapreduce.Job: Counters: 49
File System Counters
  FILE: Number of bytes read=246
  FILE: Number of bytes written=432529
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=322
  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)=832984

```

- Then I created a folder Mm.txt to store the input for the A matrix and gave the input as follows,
A,0,0,1
A,0,1,2
A,1,0,3
A,1,1,4
- Then I created a folder Nn.txt to store the input for the B matrix and gave the input as follows,
B,0,0,5
B,0,1,6
B,1,0,7
B,1,1,8
- Then I have copied files from Mm.txt and Nn.txt into inputMatrix using **put** command.
- Then I executed the command for mapreduce job and it has executed as follows,

```

cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
cloudera@quickstart:~$
File Edit View Search Terminal Help
Total time spent by all reduces in occupied slots (ms)=23973
Total time spent by all map tasks (ms)=832984
Total time spent by all reduce tasks (ms)=23973
Total vcore-milliseconds taken by all map tasks=832984
Total vcore-milliseconds taken by all reduce tasks=23973
Total megabyte-milliseconds taken by all map tasks=852975616
Total megabyte-milliseconds taken by all reduce tasks=24548352
Map-Reduce Framework
Map Input records=0
Map output records=20
Map output bytes=200
Map output materialized bytes=252
Input split bytes=250
Combine input records=0
Combine output records=0
Reduce input groups=4
Reduce shuffle bytes=252
Reduce input records=20
Reduce output records=4
Spilled Records=40
Shuffled Maps =2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=329
CPU time spent (ms)=320
Physical memory (bytes) snapshot=545783880
Virtual memory (bytes) snapshot=451942080
Total committed heap usage (bytes)=391979088
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 ~]$ hadoop fs -ls outputMatrix
Found 2 items
-rw-r--r-- 1 cloudera cloudera 0 2022-02-06 20:25 outputMatrix/_SUCCESS
-rw-r--r-- 1 cloudera cloudera 36 2022-02-06 20:25 outputMatrix/part-r-00000
[cloudera@quickstart ~]$ hadoop fs -cat outputMatrix/part-r-00000
0,0,19.0
0,1,22.0
1,0,43.0
1,1,50.0
[cloudera@quickstart ~]$

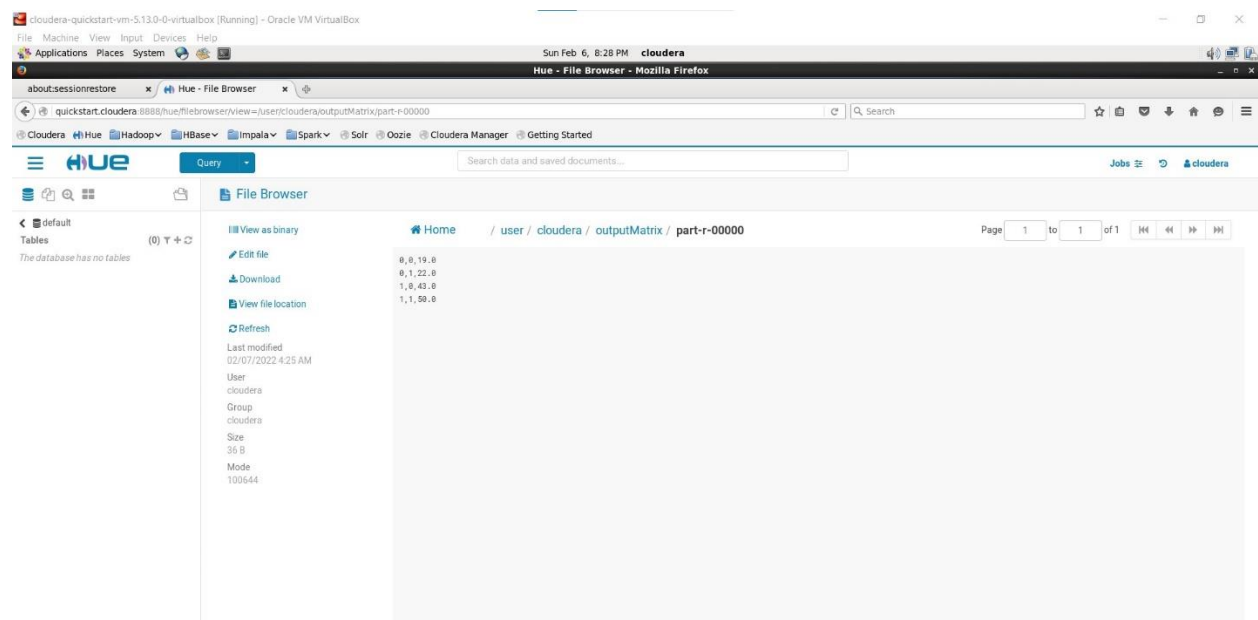
```

- Then I have listed the files in the outputMatrix using ls command
- Finally I have used cat command to display the output of matrix multiplication and the result is correct as follows,

```

0,0,19.0
0,1,22.0
1,0,43.0
1,1,50.0

```



- Then I have visualized the output file using Hue.