AWS Practical 5

Rahul Agrawal (21200098)

Q1> Implement Map and Reduce functions of the matrix-matrix multiplication in Python.

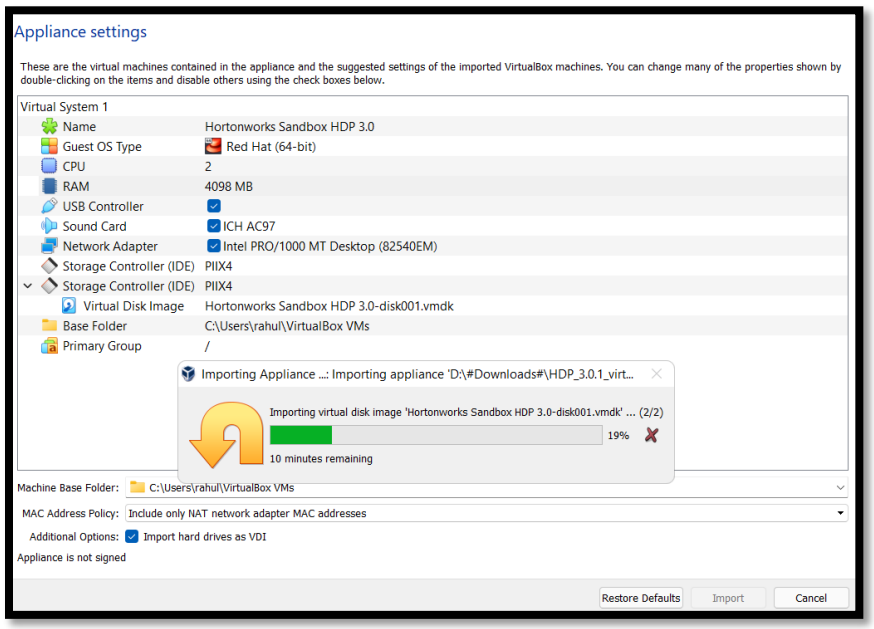
1. **Describe the setup architecture of your exercise.**

Answer ➔

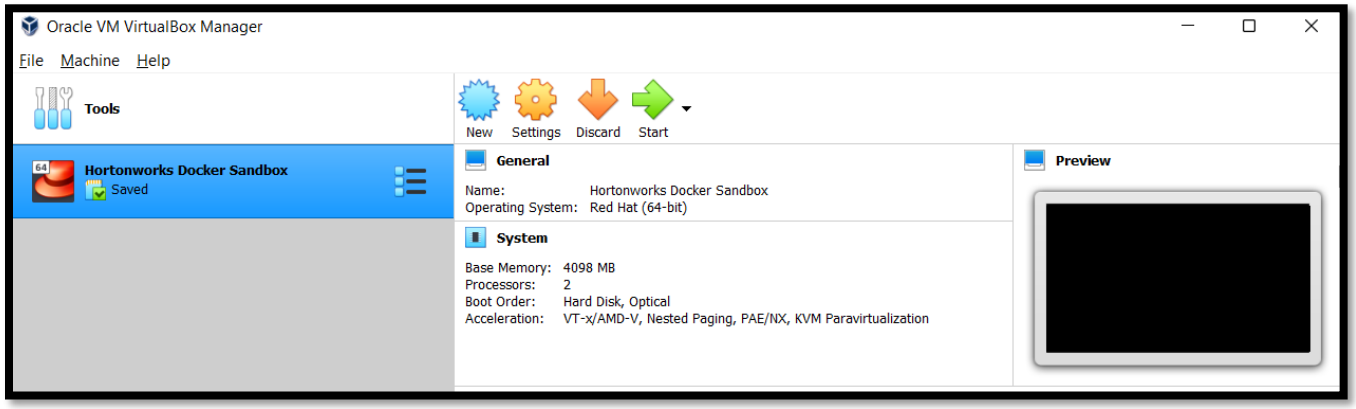
a) Download Hortonworks HDP for Oracle VirtualBox from the below link. This will download a **.ova** file. https://www.cloudera.com/downloads/hortonworks-sandbox.html

b) Download and install Oracle VirtualBox (v6.1) from here https://www.virtualbox.org .

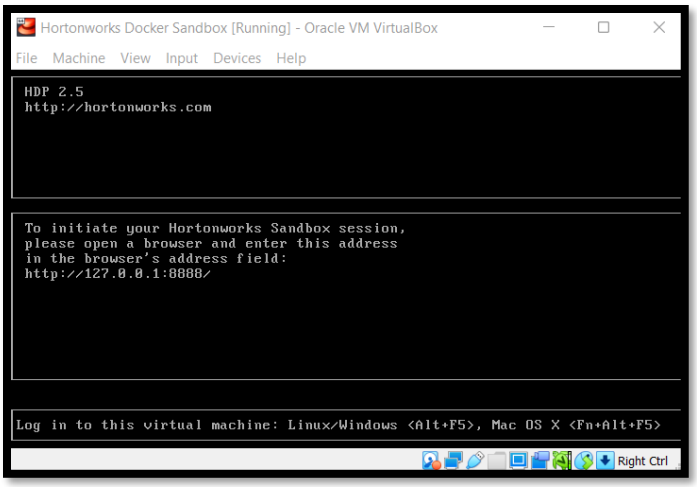
c) Open the HDP (.ova) file from VirtualBox by navigating to File->Import Appliance. Edit RAM and CPU according to laptop (I changed it to 4GB and 2 cores).



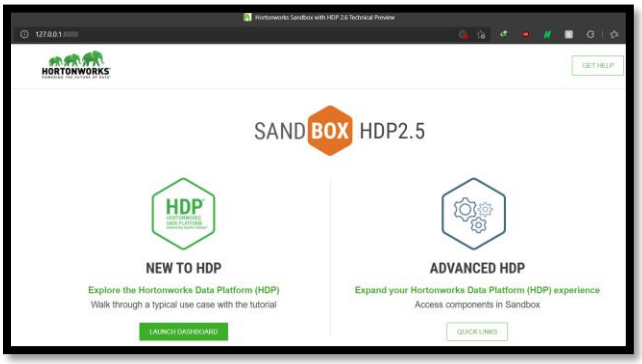
d) This will import the appliance into VirtualBox and you can see one sandbox.

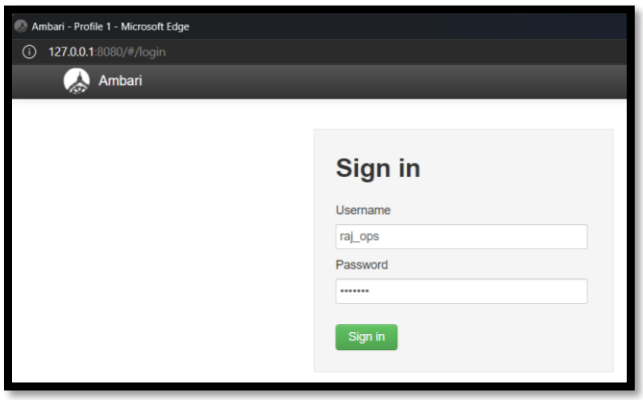


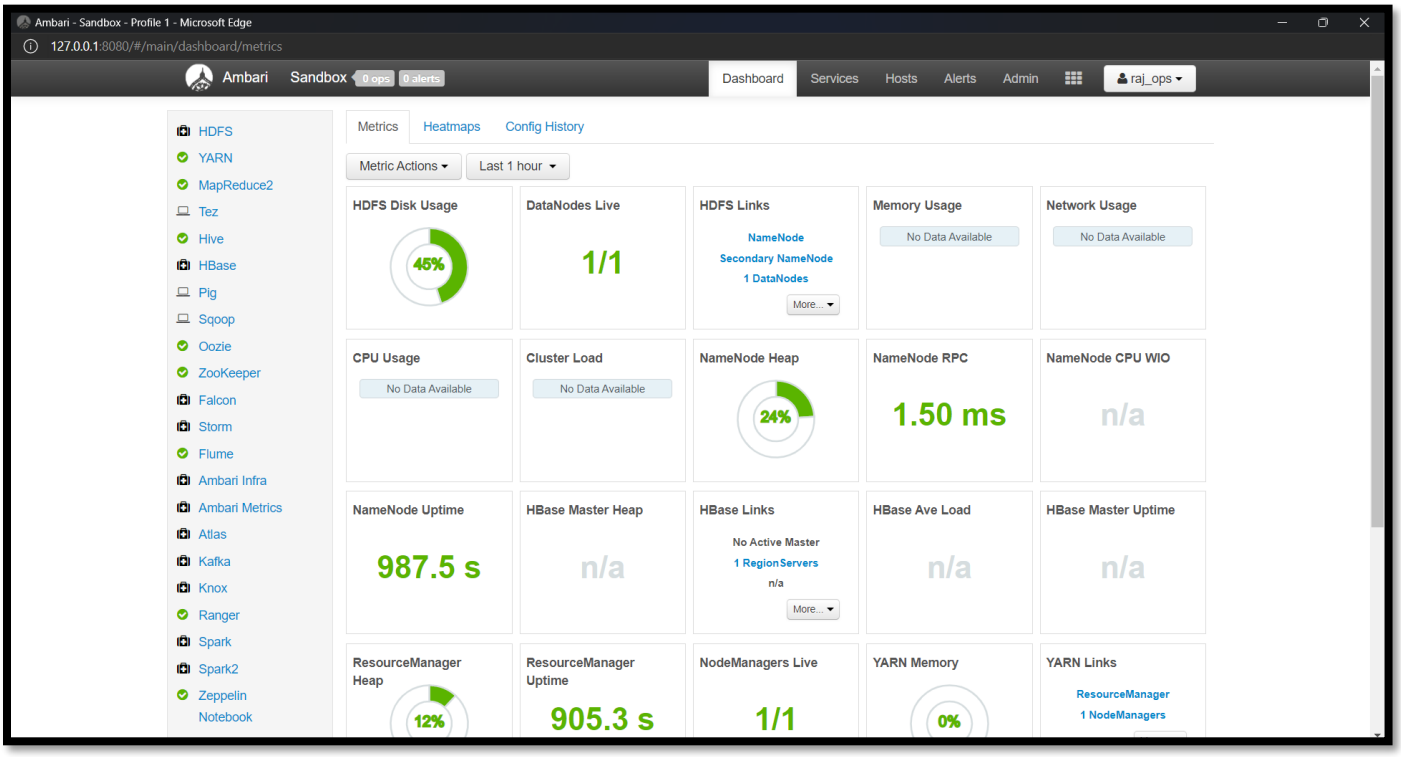
e) Go to settings and navigate to Network->Adapter 2, change this to Host-only Adapter (if not enabled). Click on the green button Start. This will open a new window and installation will start. After that Hortonworks will be hosted on http://127.0.0.1:8888



f) An interface like this will open, click on **Launch Dashboard**. Click on Advanced HDP and then under Ambari, you'll find username and password (under Cluster Operator). Use these credentials to login from the dashboard.

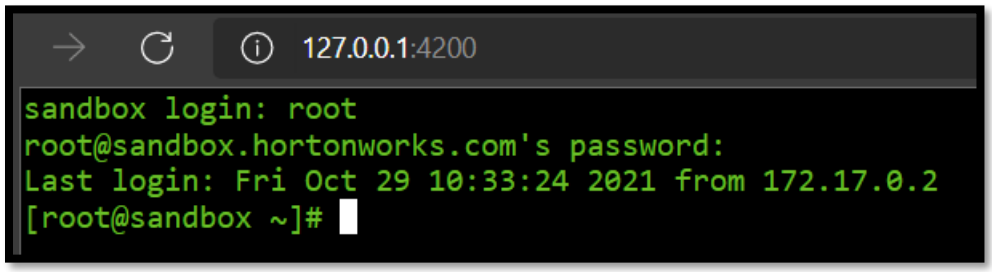






g) Go back to http://127.0.0.1:8888. Open Advanced HDP and under SSH Client you can see the IP and port of namenode. If no SSH client is installed we can use web client on http://127.0.0.1:4200

h) Open http://127.0.0.1:4200/ and login with root/hadoop. Change the password and this kind of terminal will appear.



2. Describe Mapper and Reducer function in Python.

Answer ➔

Mapper: Mapper will read the input.txt file and segregate the two matrices by assigning a key to them. A for first one and B for second one.

Reducer: Reducer will read the input from mapper and create two dictionary which will represent two matrices a and b. It will also count the number of rows and number of columns present in each matrix.

a ➔ {(0, 0): 1, (0, 1): 2, (0, 2): 3, (1, 0): 4, (1, 1): 5, (1, 2): 6, (2, 0): 3, (2, 1): 5, (2, 2): 2}

b ➔ {(0, 0): 7, (0, 1): 8, (1, 0): 9, (1, 1): 10, (2, 0): 11, (2, 1): 12}

Reducer will only proceed only if column count of a is equal to row count of b. It will iterate and perform the multiplication process. Reducer will then print the result to console.

Q2> Implement Map and Reduce functions of the matrix-matrix multiplication in Python. Describe your experience step-by-step in your own words and provide screenshots of executed MapReduce programs.

Answer ➔

a) First write this python program in txt editor

Mapper.py

```
#!/usr/bin/env python
import sys
for inputline in sys.stdin:
    matrix, row, col, val = inputline.strip().split(",")
    value = ','.join((row, col, val))
    if matrix == 'A':
        print('{0}\t{1}'.format(matrix, value))
    elif matrix == 'B':
        print('{0}\t{1}'.format(matrix, value))
```

Reducer.py

```
#!/usr/bin/env python
import sys
a, b = {}, {}
a_row, a_col, b_row, b_col = 0, 0, 0, 0
for inputstring in sys.stdin:
    inputstring = inputstring.strip()
    matrix, value = inputstring.split("\t", 1)
    val = value.split(",")
    i, j, k = int(val[0]), int(val[1]), int(val[2])
    if matrix == 'A':
        a[(i, j)] = k
        a_row = i + 1
        a_col = j + 1
    elif matrix == 'B':
        b[(i, j)] = k
        b_row = i + 1
        b_col = j + 1

# Doing multiplication
result = 0
if a_col == b_row:
    for i in range(0, a_row):
        for j in range(0, b_col):
            for k in range(0, a_col):
                result = result + a[(i, k)]*b[(k, j)]
            print("{0},{1},{2}".format(i, j, result))
            result = 0
```

b) Save these files. Input.txt file contains two matrices A and B where A is 3x3 matrix and B is 3x2 matrix.

```
A,0,0,1
A,0,1,2
A,0,2,3
A,1,0,4
A,1,1,5
A,1,2,6
A,2,0,3
A,2,1,5
A,2,2,2
B,0,0,7
B,0,1,8
B,1,0,9
B,1,1,10
B,2,0,11
B,2,1,12
```

c) Create a folder **mapred_upload** from namenode terminal and then navigate to HDP dashboard and open View Files. Thereafter upload these three files (input.txt, mapper.py, reducer.py).

hdfs dfs -mkdir /mapred_upload

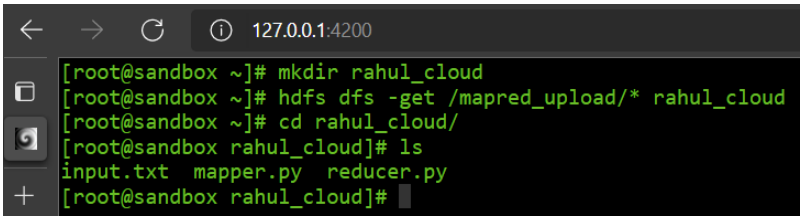
d) Open the namenode terminal and create a directory **rahul_cloud** and copy back these three uploaded files to namenode.

mkdir rahul_cloud

hdfs dfs -get /mapred_upload/* rahul_cloud

cd rahul_cloud

ls



e) Check if there is a folder in hdfs under **/user/root/mapreducetask**. If not present make directory using

hdfs dfs -mkdir /user/root

hdfs dfs -mkdir /user/root/mapreducetask

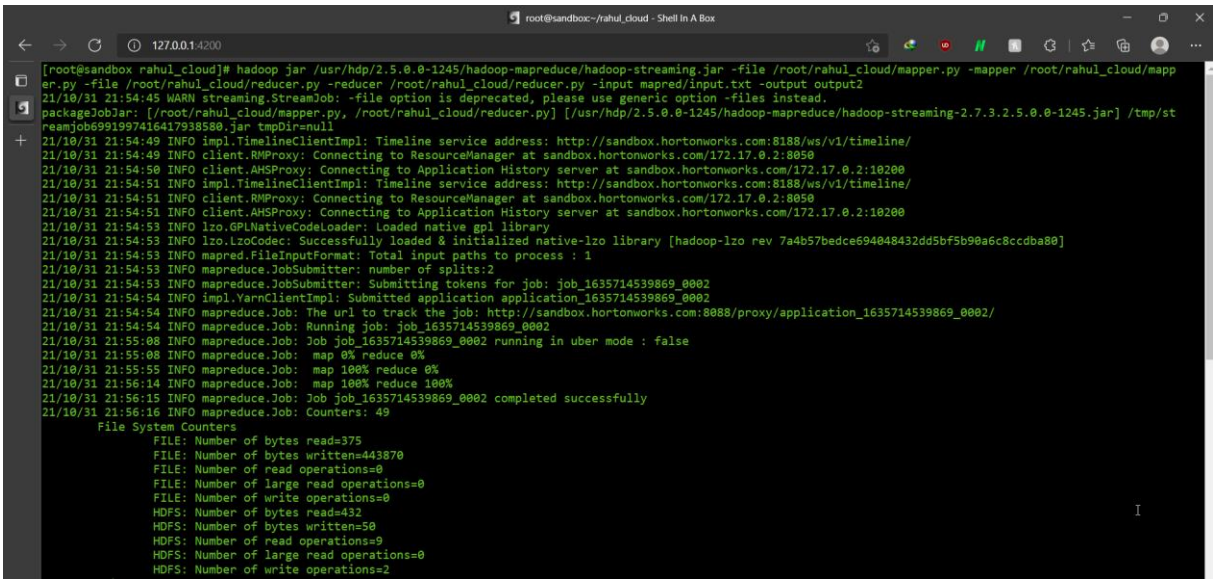
f) Copy input.txt from namenode to /user/root/mapreducetask using

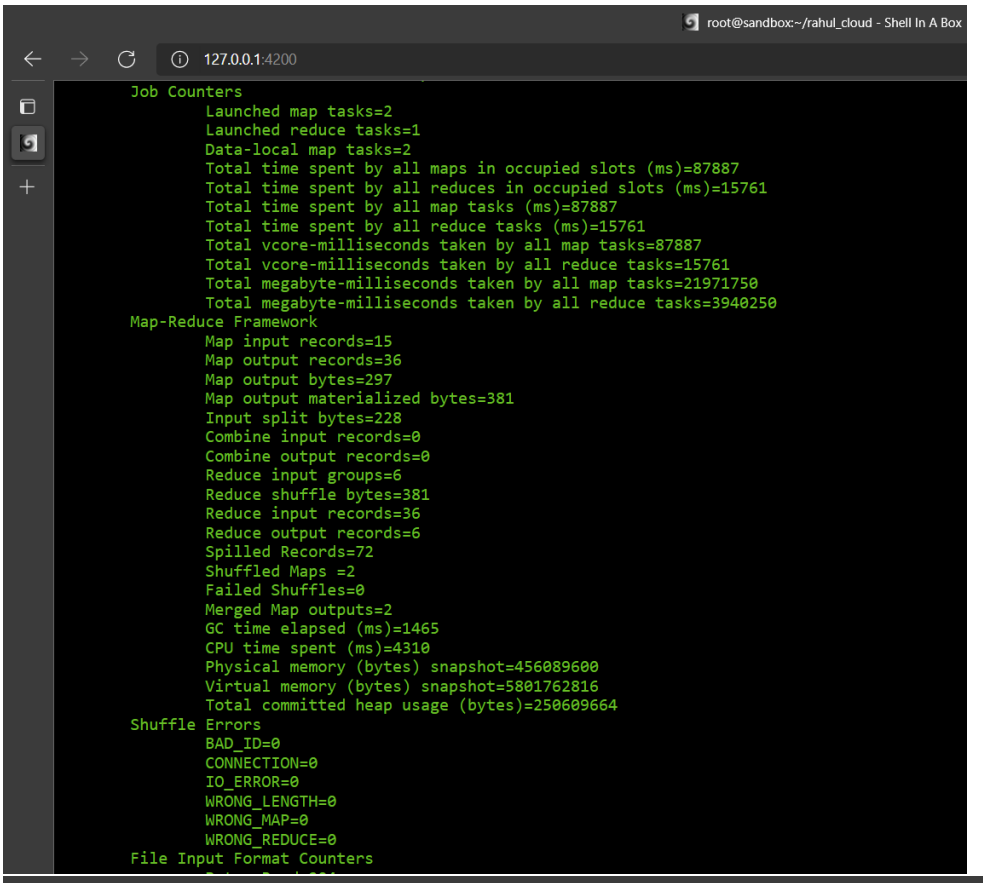
cd rahul_cloud

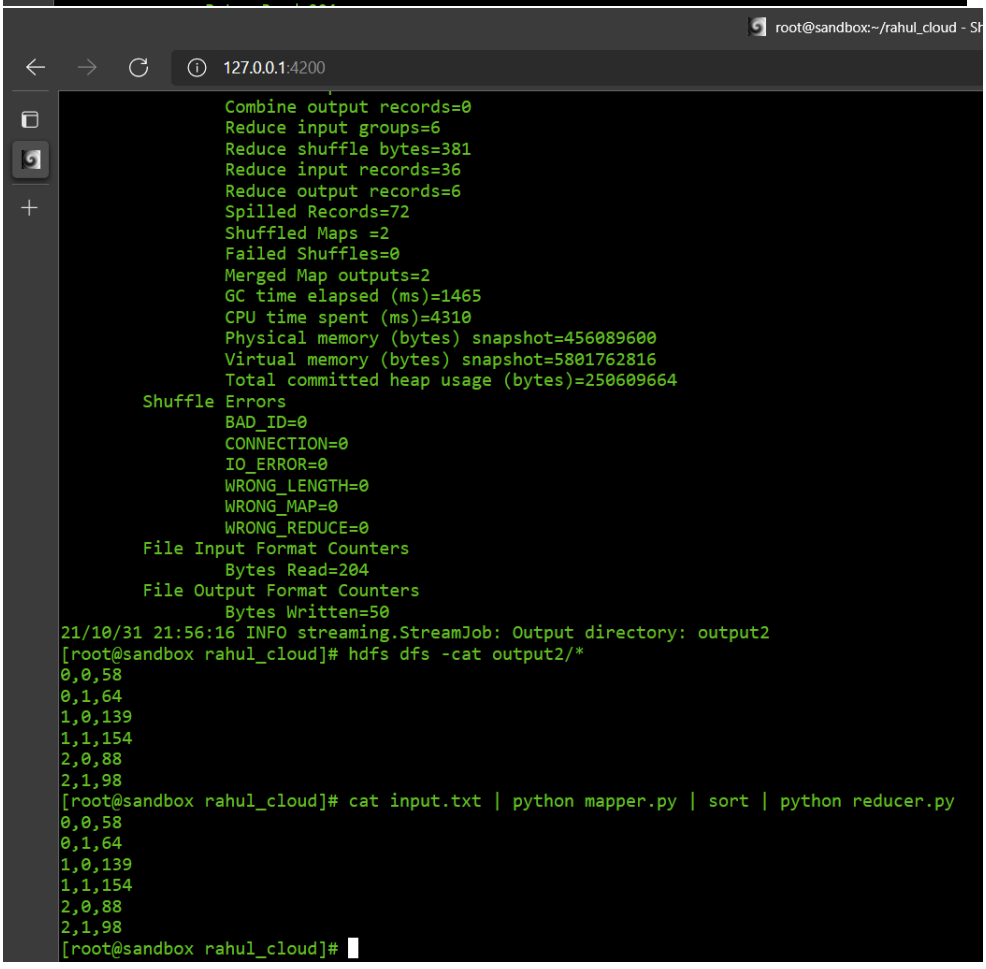
hdfs dfs -put ./input.txt mapreducetask

g) Start Hadoop MapReduce Streaming jar file with following parameters.

***hadoop jar /usr/hdp/2.5.0.0-1245/hadoop-mapreduce/hadoop-streaming.jar ***
***-file /root/rahul_cloud/mapper.py -mapper /root/rahul_cloud/mapper.py ***
***-file /root/rahul_cloud/reducer.py -reducer /root/rahul_cloud/reducer.py ***
-input mapreducetask/input.txt -output output1




root@sandbox:~/rahul_cloud - Shell In A Box
127.0.0.1:4200
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)=87887
Total time spent by all reduces in occupied slots (ms)=15761
Total time spent by all map tasks (ms)=87887
Total time spent by all reduce tasks (ms)=15761
Total vcore-milliseconds taken by all map tasks=87887
Total vcore-milliseconds taken by all reduce tasks=15761
Total megabyte-milliseconds taken by all map tasks=21971750
Total megabyte-milliseconds taken by all reduce tasks=3940250
Map-Reduce Framework
Map input records=15
Map output records=36
Map output bytes=297
Map output materialized bytes=381
Input split bytes=228
Combine input records=0
Combine output records=0
Reduce input groups=6
Reduce shuffle bytes=381
Reduce input records=36
Reduce output records=6
Spilled Records=72
Shuffled Maps =2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=1465
CPU time spent (ms)=4310
Physical memory (bytes) snapshot=456089600
Virtual memory (bytes) snapshot=5801762816
Total committed heap usage (bytes)=250609664
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters


root@sandbox:~/rahul_cloud - Sh
127.0.0.1:4200
Combine output records=0
Reduce input groups=6
Reduce shuffle bytes=381
Reduce input records=36
Reduce output records=6
Spilled Records=72
Shuffled Maps =2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=1465
CPU time spent (ms)=4310
Physical memory (bytes) snapshot=456089600
Virtual memory (bytes) snapshot=5801762816
Total committed heap usage (bytes)=250609664
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=204
File Output Format Counters
Bytes Written=50
21/10/31 21:56:16 INFO streaming.StreamJob: Output directory: output2
[root@sandbox rahul_cloud]# hdfs dfs -cat output2/*
0,0,58
0,1,64
1,0,139
1,1,154
2,0,88
2,1,98
[root@sandbox rahul_cloud]# cat input.txt | python mapper.py | sort | python reducer.py
0,0,58
0,1,64
1,0,139
1,1,154
2,0,88
2,1,98
[root@sandbox rahul_cloud]#