Big Data Technologies CSP-554

Project Report Document

Vaishnavi Prasanna Shetty
A20519894
vprasannashetty@hawk.iit.edu
https://github.com/VFA22SCM94P/CSP554

Project Name: Bigdata File Filter and Convertor

Abstract:

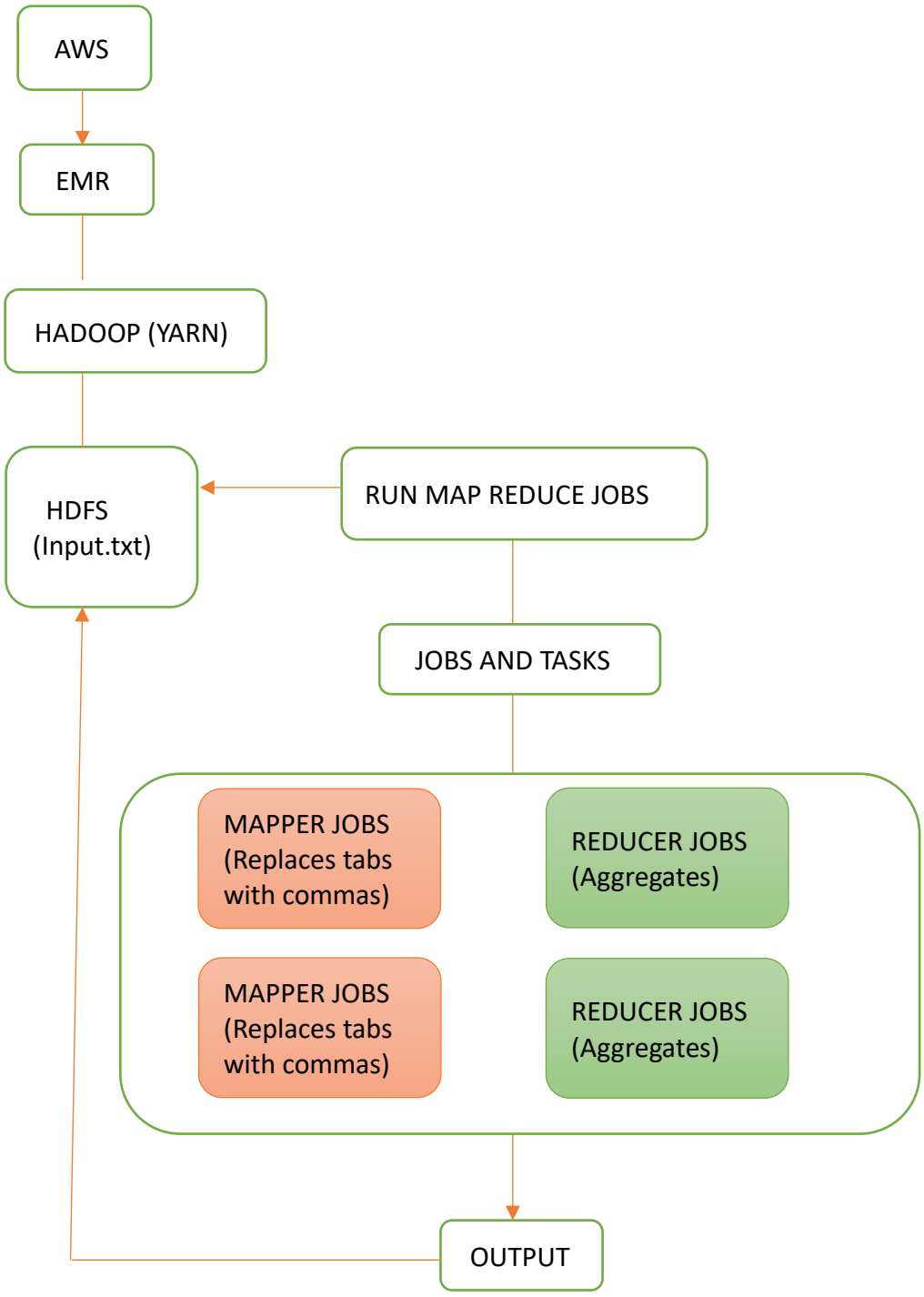
The work involves contribution towards the existing bigdata-file-viewer project by Eugene-Mark which is a cross platform desktop application to view the common bigdata binary format like Parquet, Avro.

The main objective of this project was to develop a program that converts the file format from Comma Separated Value (CSV) to Tab Separated Value (TSV) format. Map Reduce concept is utilized to provide the deliverables.

Overview:

The program has Mapper and Reducer jobs which gets the file from HDFS and modifies the file format from Comma Separated Values to Tab Separated Values. The file is fetched from HDFS location. The program mainly checks three important things before the mapper job is run which is confirming whether the given file is not empty, making sure that the file consists of commas prior running mapper and reducer jobs to avoid map reduce job runs.

Architecture:



Design:

There are 4 major components:

- Hadoop Distributed File System (HDFS): The input file is placed in HDFS.
- AWS Elastic MapReduce (EMR): This is used for creating the cluster for running the MapReduce jobs.
- MapReduce Job: This replaces the commas with the tabs for the input file.

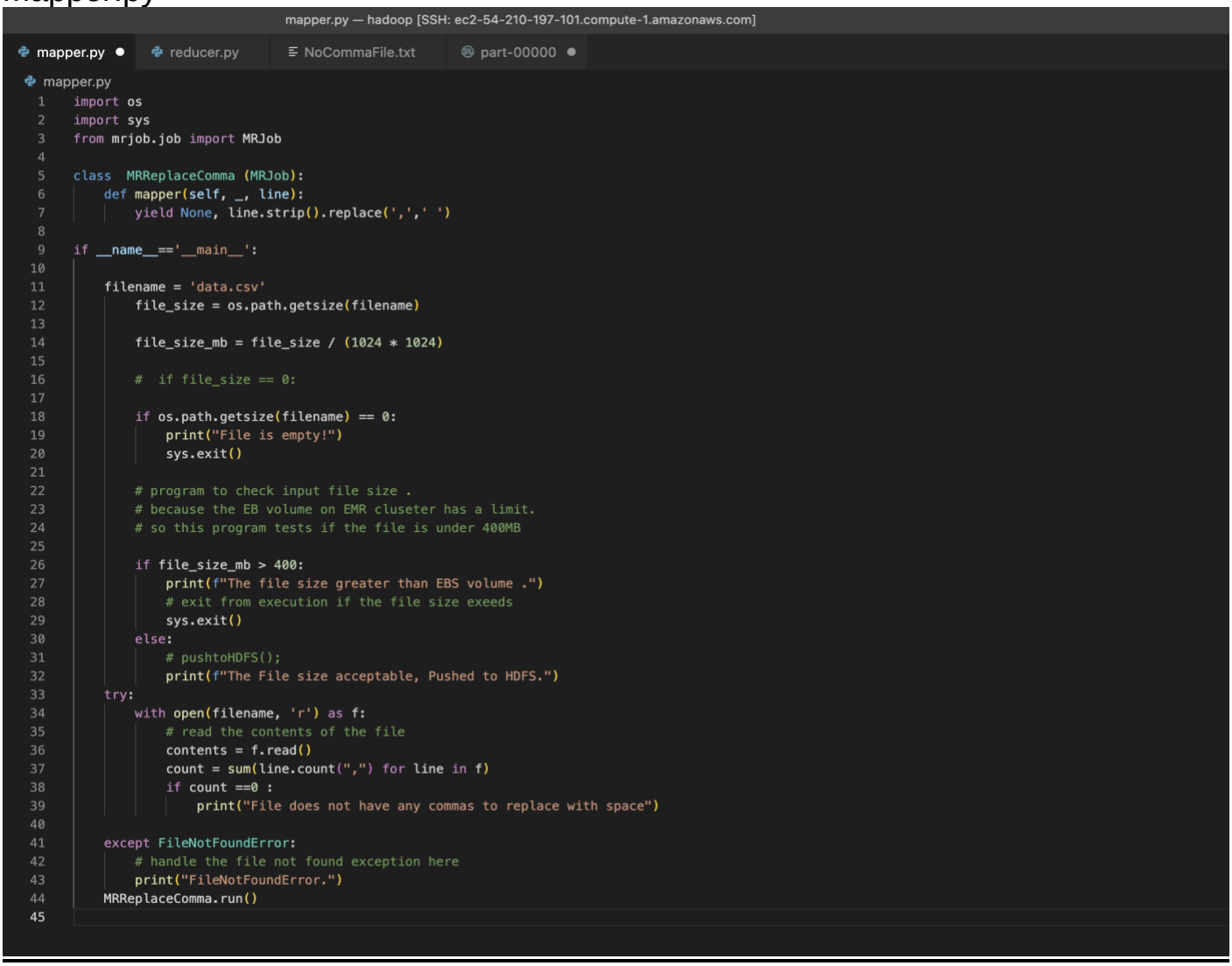
MapReduce Job:

Mapper:
The mapper reads the input data from HDFS and processes it concurrently throughout the cluster during the map phase. The input data for the mapper is provided in the form of key-value pairs, where the key is the record's input file byte offset and the value is the actual record. Each entry is processed by the mapper by substituting tab spaces for commas using the replace() method.

Reducer:
The output from the mapper is delivered to the reducer, which processes it concurrently across the cluster during the reduce phase. Key-value pairs are provided to the reducer as input data, with the key being the dummy value produced by the mapper and the value being the updated record. The output of the reducer is written to HDFS as a tab-separated file after combining all the updated records together.

Code:

mapper.py



reducer.py

File generation code

filegen.py > ...

```python
import csv
import random

with open('data.csv', 'w') as csvfile:
    writer = csv.writer(csvfile)
    for i in range(100000):
        row = [random.random() for j in range(10)]
        writer.writerow(row)

```

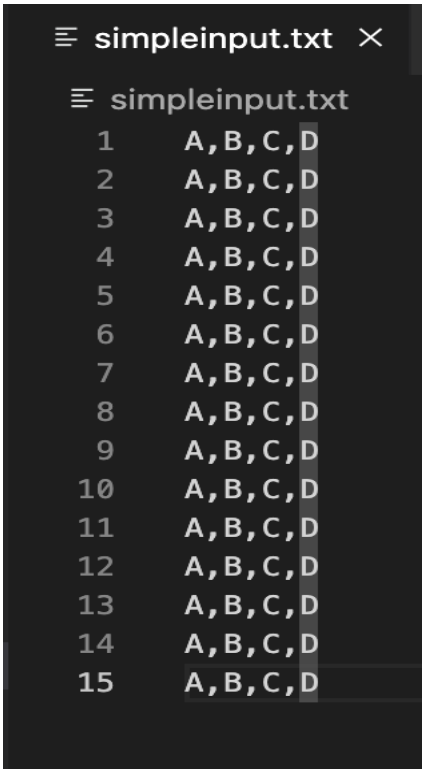
Code to generate file without any comma in it.

nocomma.py > ...

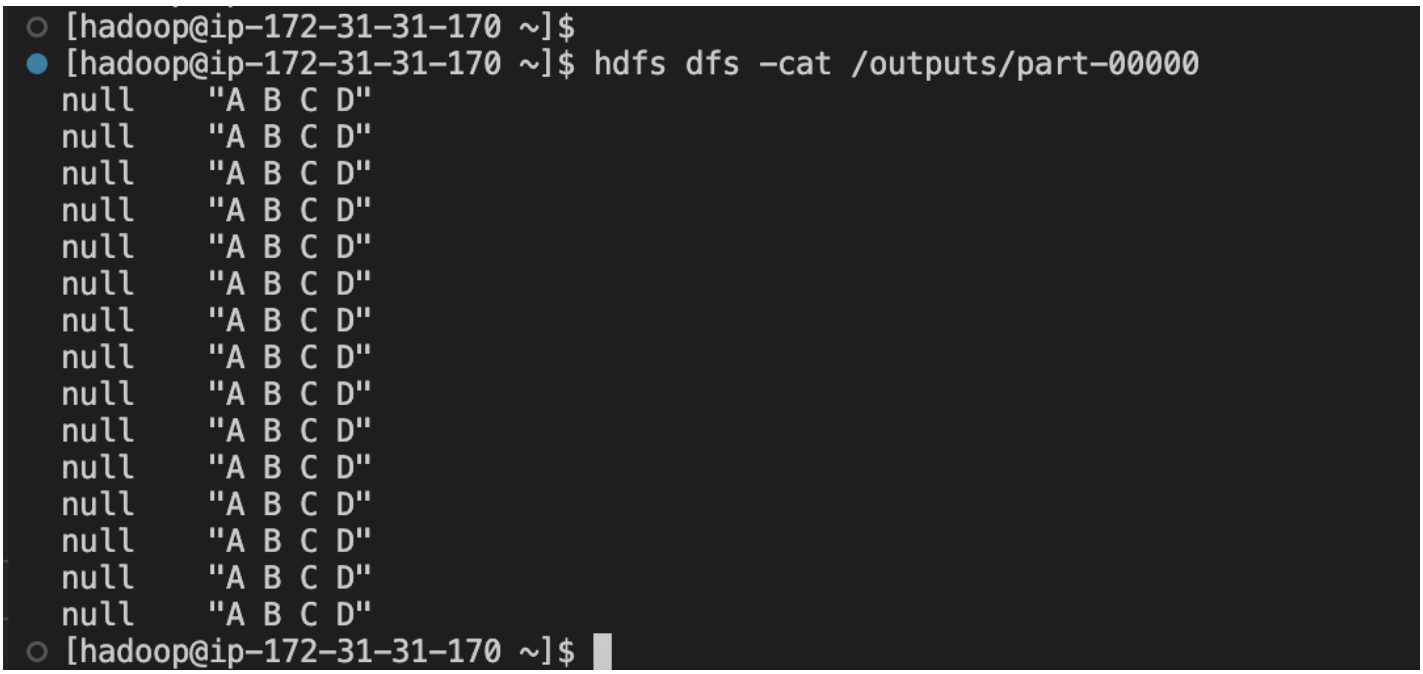
```python
import random

filename = 'NoCommaFile.txt'

# data without commas
data = ''.join(random.choices('12345', k=10000))

# Write the data to the file
with open(filename, 'w') as f:
    f.write(data)

print(f"The file created")

```

Functioning of the code on the small data sample

Input file

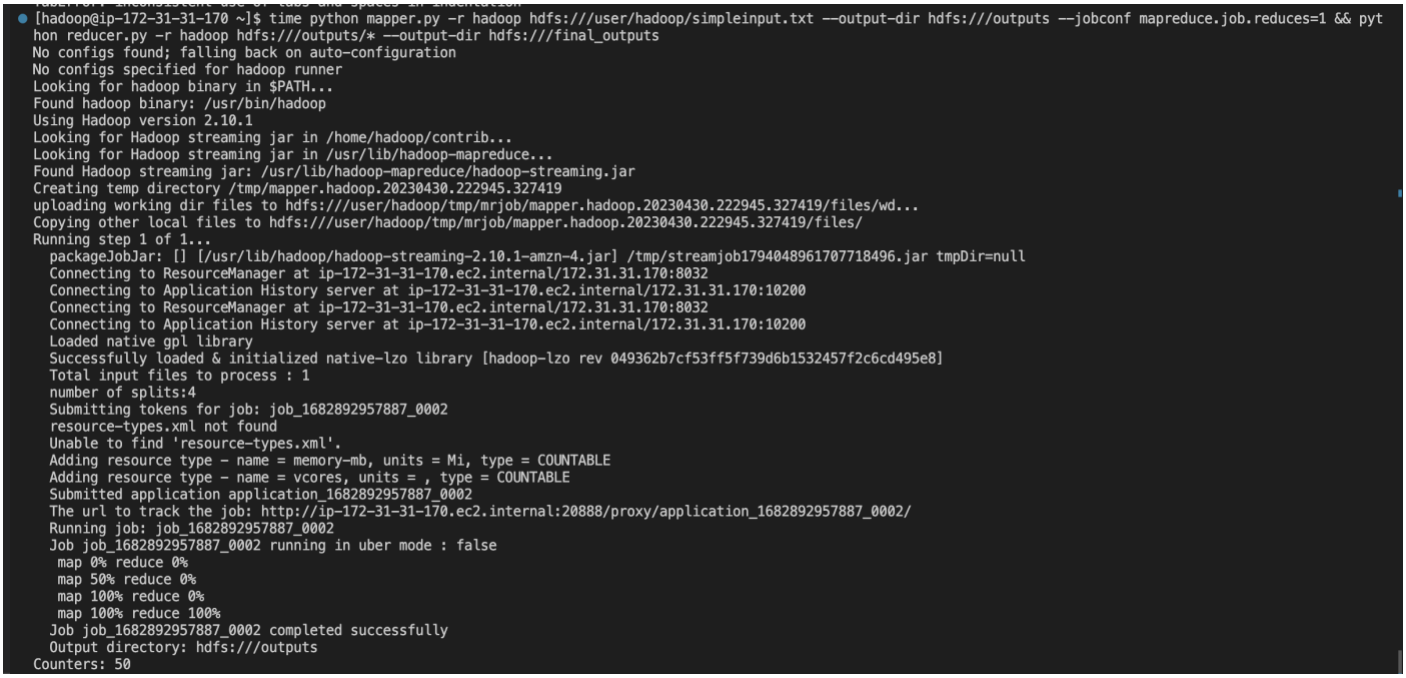


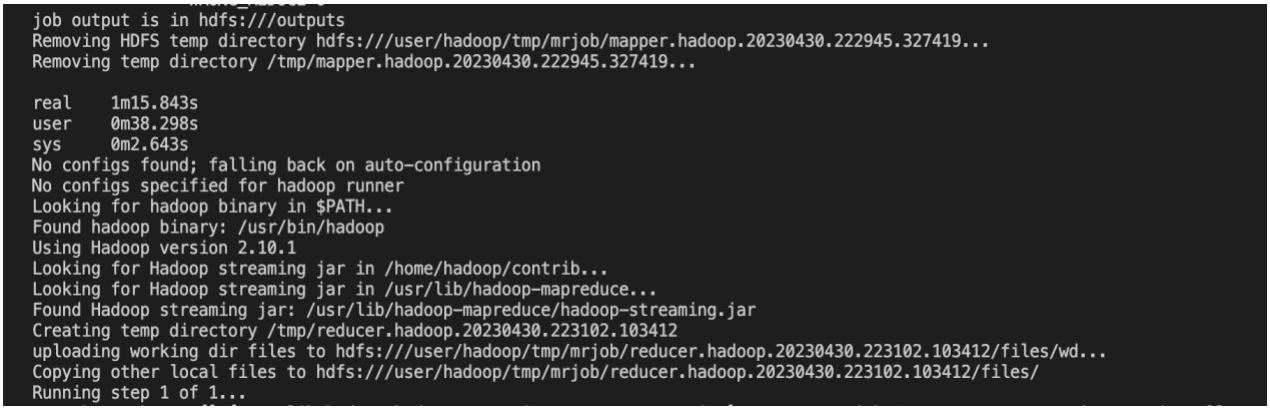
Output:

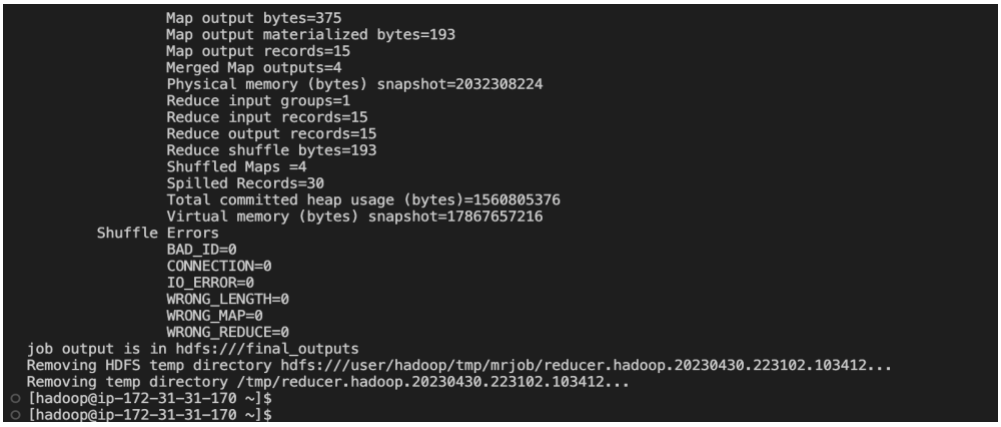


Job run screenshots:

Runtime: 38s

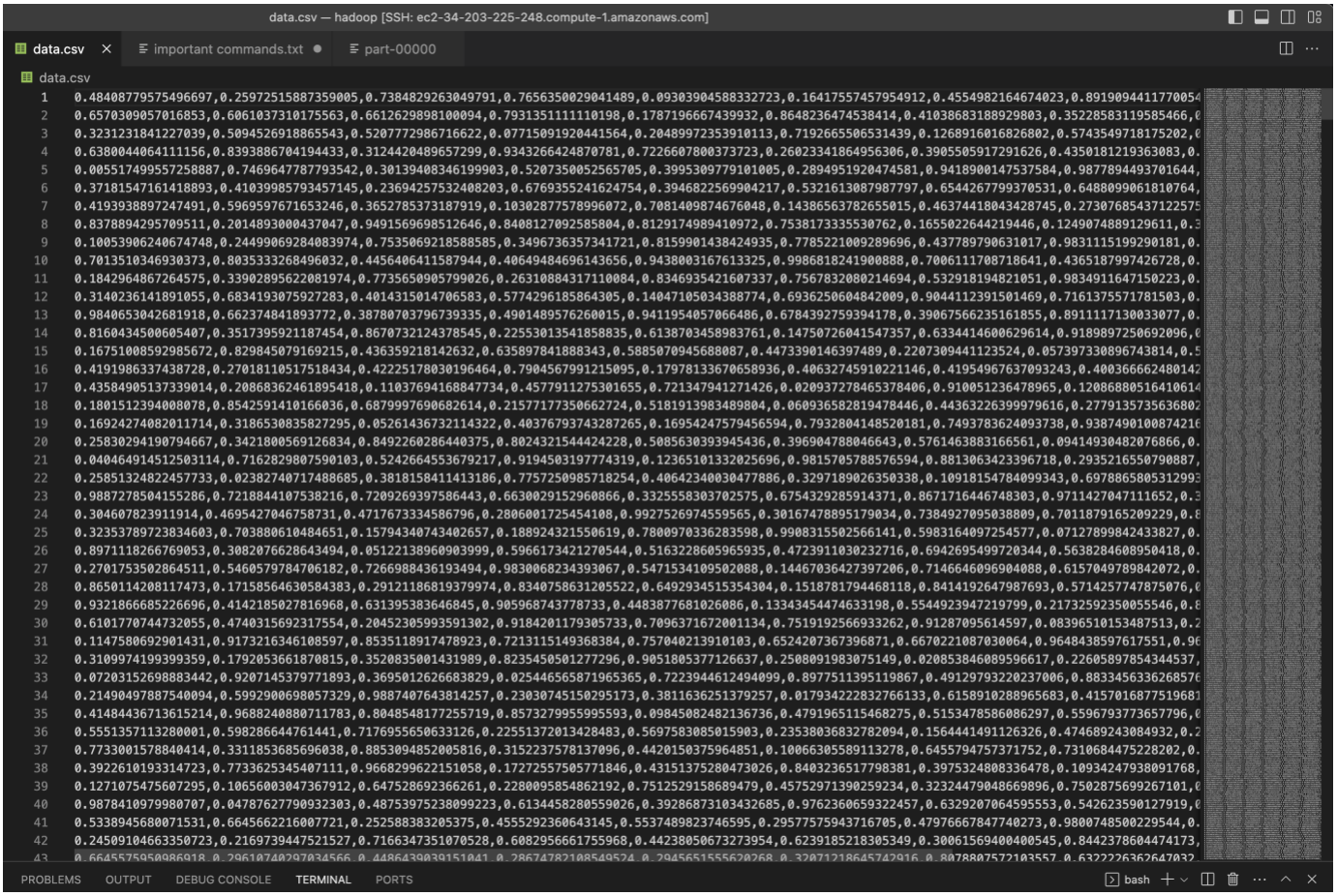




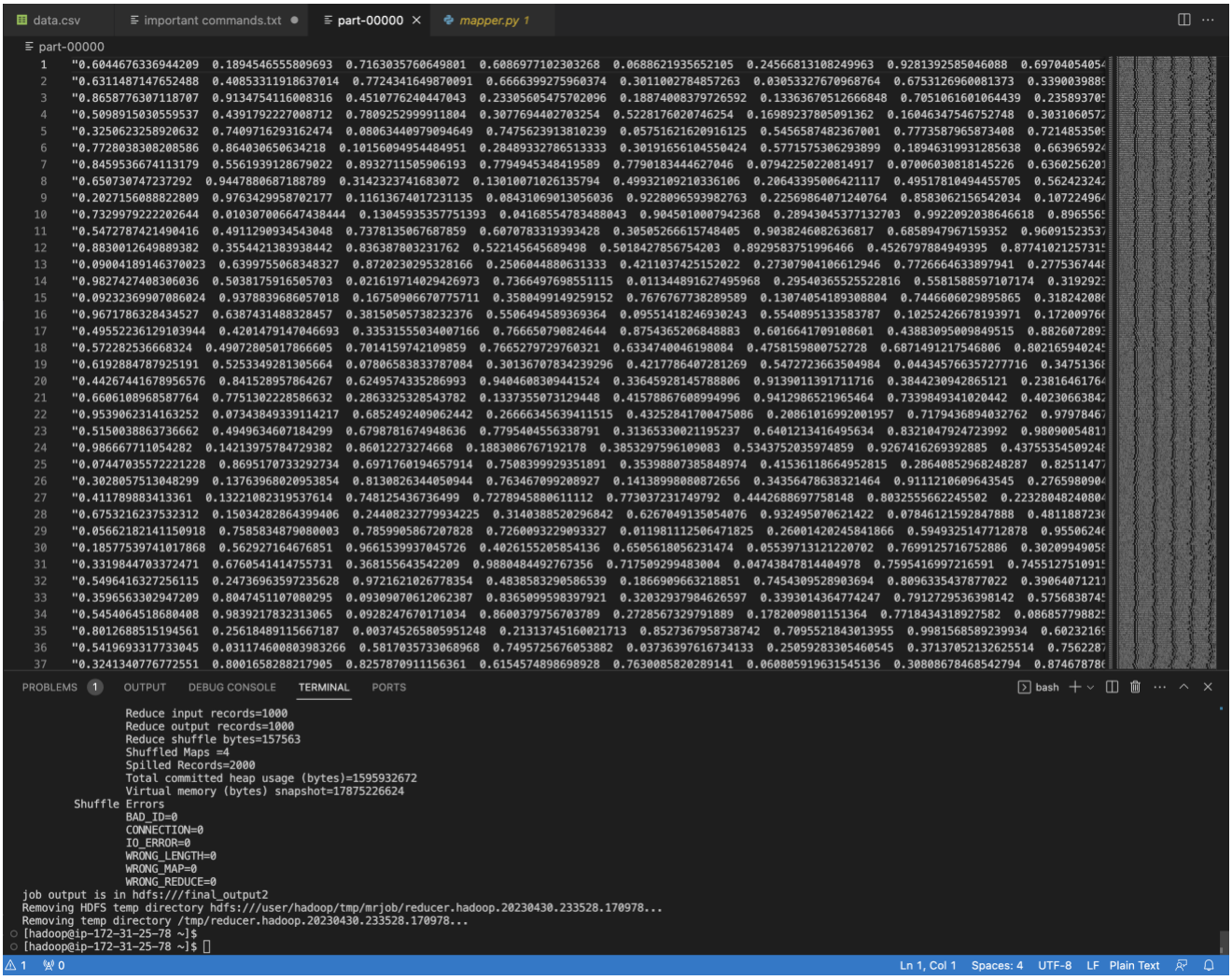


Executing the random generated file

Input data

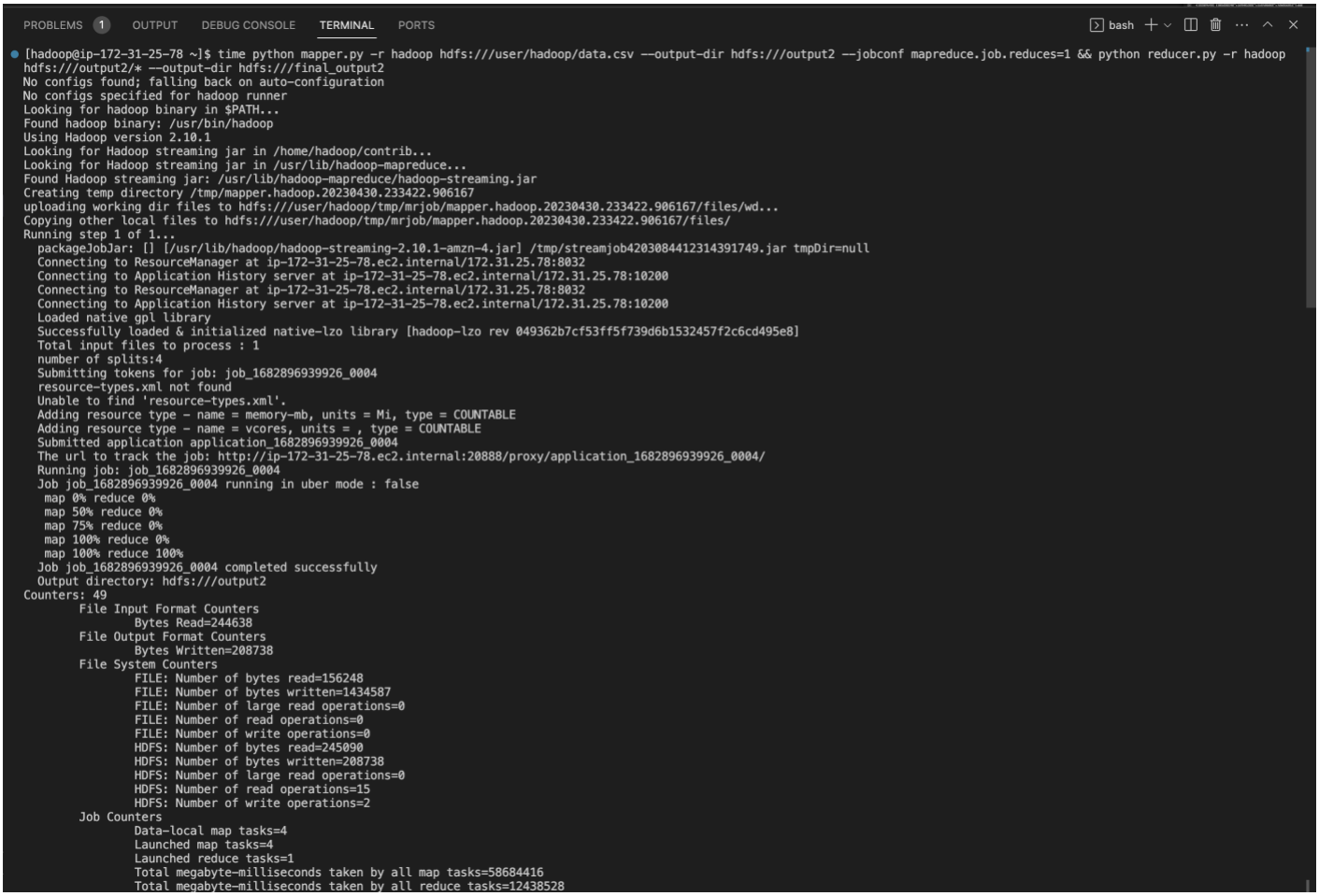


Output data


data.csv
important commands.txt
part-00000
mapper.py 1
part-00000
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash
Reduce input records=1000
Reduce output records=1000
Reduce shuffle bytes=157563
Shuffled Maps =4
Spilled Records=2000
Total committed heap usage (bytes)=1595932672
Virtual memory (bytes) snapshot=17875226624
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
job output is in hdfs:///final_output2
Removing HDFS temp directory hdfs:///user/hadoop/tmp/mrjob/reducer.hadoop.20230430.233528.170978...
Removing temp directory /tmp/reducer.hadoop.20230430.233528.170978...
[hadoop@ip-172-31-25-78 ~]$
[hadoop@ip-172-31-25-78 ~]$
Ln 1, Col 1
Spaces: 4
UTF-8
LF
Plain Text

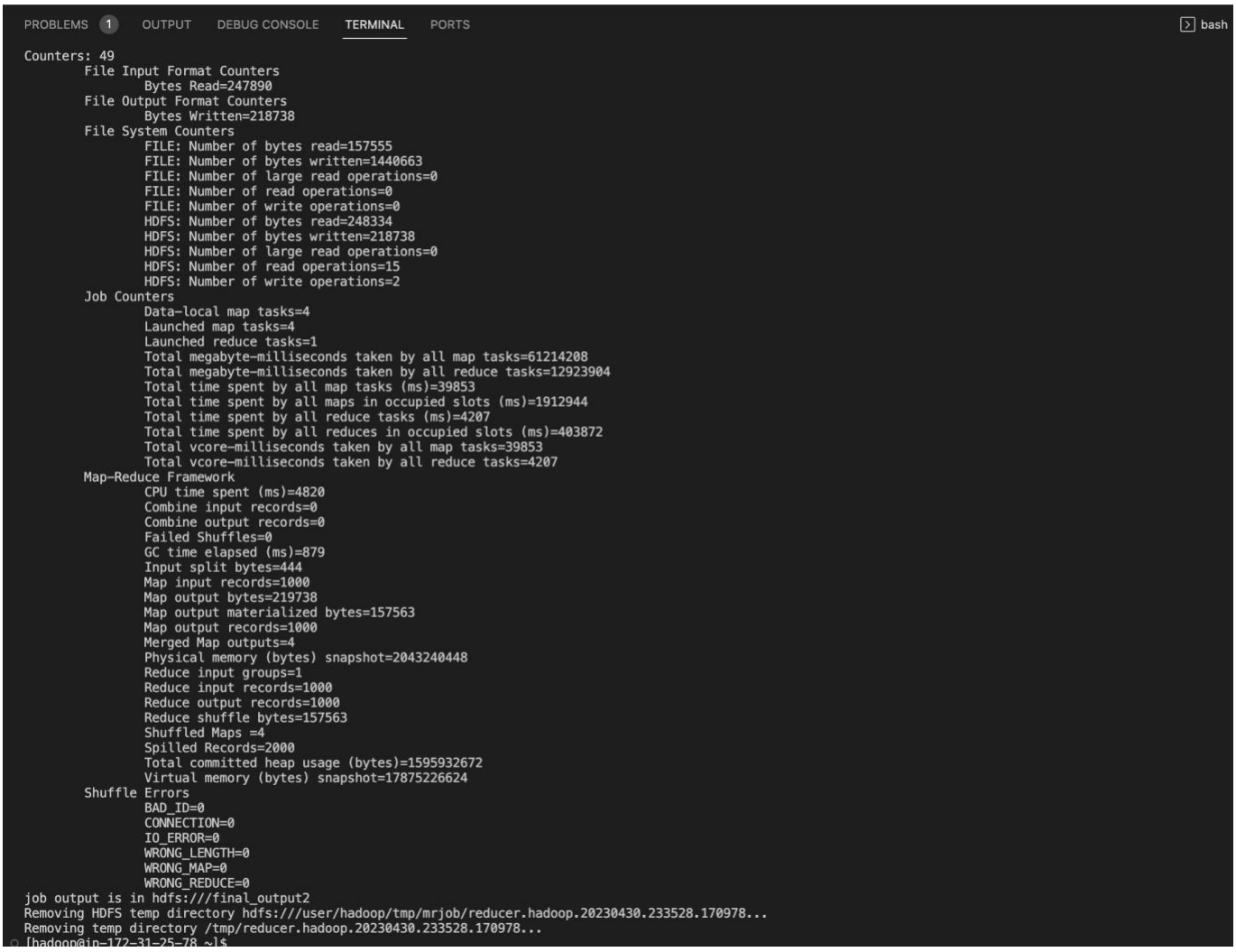
Job run screenshots.

Execution time: 38s



```
Job job_1682896939926_0004 running in uber mode : false
 map 0% reduce 0%
 map 50% reduce 0%
 map 75% reduce 0%
 map 100% reduce 0%
 map 100% reduce 100%
Job job_1682896939926_0004 completed successfully
Output directory: hdfs:///output2
Counters: 49
        File Input Format Counters
                Bytes Read=244638
        File Output Format Counters
                Bytes Written=208738
        File System Counters
                FILE: Number of bytes read=156248
                FILE: Number of bytes written=1434587
                FILE: Number of large read operations=0
                FILE: Number of read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=245090
                HDFS: Number of bytes written=208738
                HDFS: Number of large read operations=0
                HDFS: Number of read operations=15
                HDFS: Number of write operations=2
        Job Counters
                Data-local map tasks=4
                Launched map tasks=4
                Launched reduce tasks=1
                Total megabyte-milliseconds taken by all map tasks=58684416
                Total megabyte-milliseconds taken by all reduce tasks=12438528
                Total time spent by all map tasks (ms)=38206
                Total time spent by all maps in occupied slots (ms)=1833888
                Total time spent by all reduce tasks (ms)=4049
                Total time spent by all reduces in occupied slots (ms)=388704
                Total vcore-milliseconds taken by all map tasks=38206
                Total vcore-milliseconds taken by all reduce tasks=4049
        Map-Reduce Framework
                CPU time spent (ms)=4330
                Combine input records=0
```

```
        Map-Reduce Framework
                CPU time spent (ms)=4330
                Combine input records=0
                Combine output records=0
                Failed Shuffles=0
                GC time elapsed (ms)=916
                Input split bytes=452
                Map input records=1000
                Map output bytes=209738
                Map output materialized bytes=156759
                Map output records=1000
                Merged Map outputs=4
                Physical memory (bytes) snapshot=2073980928
                Reduce input groups=1
                Reduce input records=1000
                Reduce output records=1000
                Reduce shuffle bytes=156759
                Shuffled Maps =4
                Spilled Records=2000
                Total committed heap usage (bytes)=1670905856
                Virtual memory (bytes) snapshot=17882116096
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
job output is in hdfs:///output2
Removing HDFS temp directory hdfs:///user/hadoop/tmp/mrjob/mapper.hadoop.20230430.233422.906167...
Removing temp directory /tmp/mapper.hadoop.20230430.233422.906167...

real    1m5.387s
user    0m38.330s
sys     0m2.611s
```

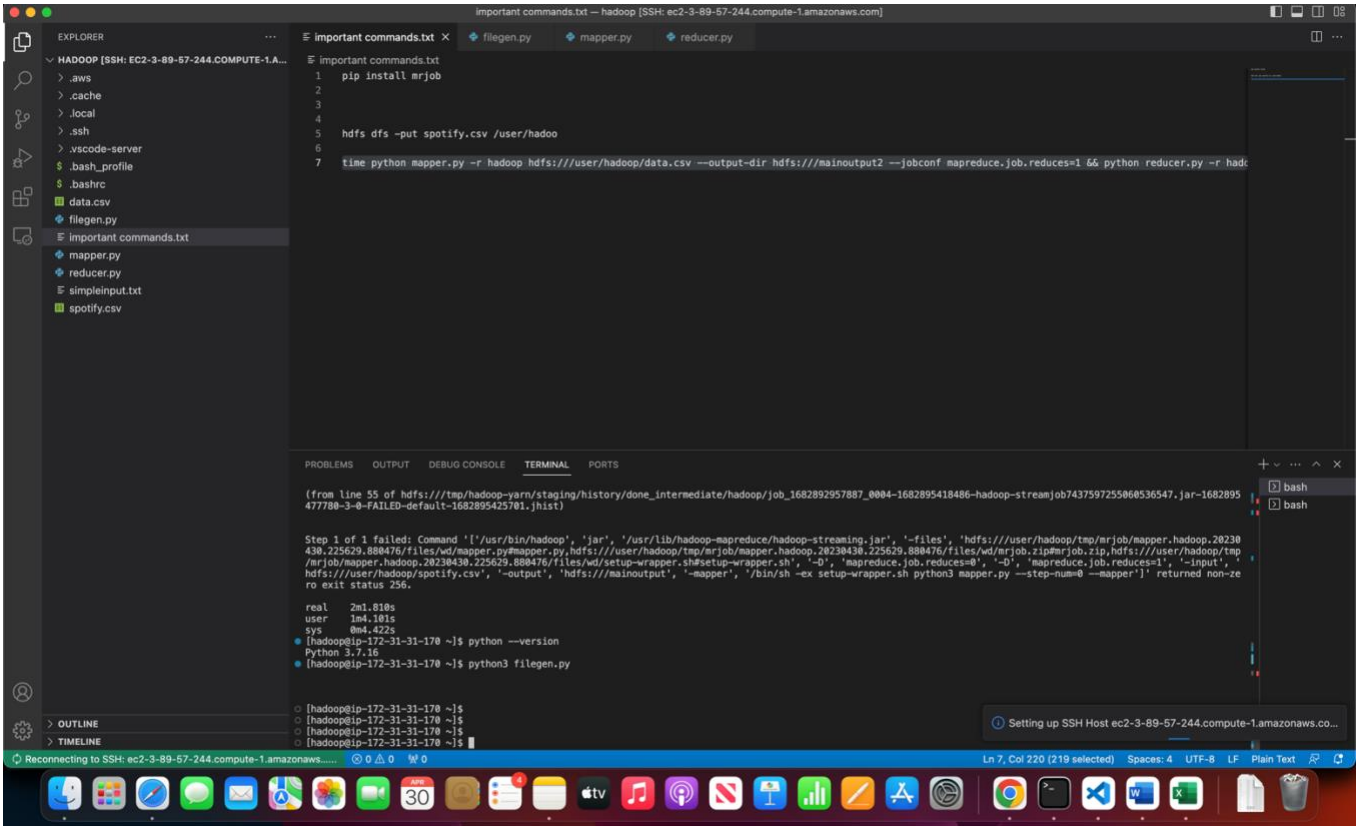


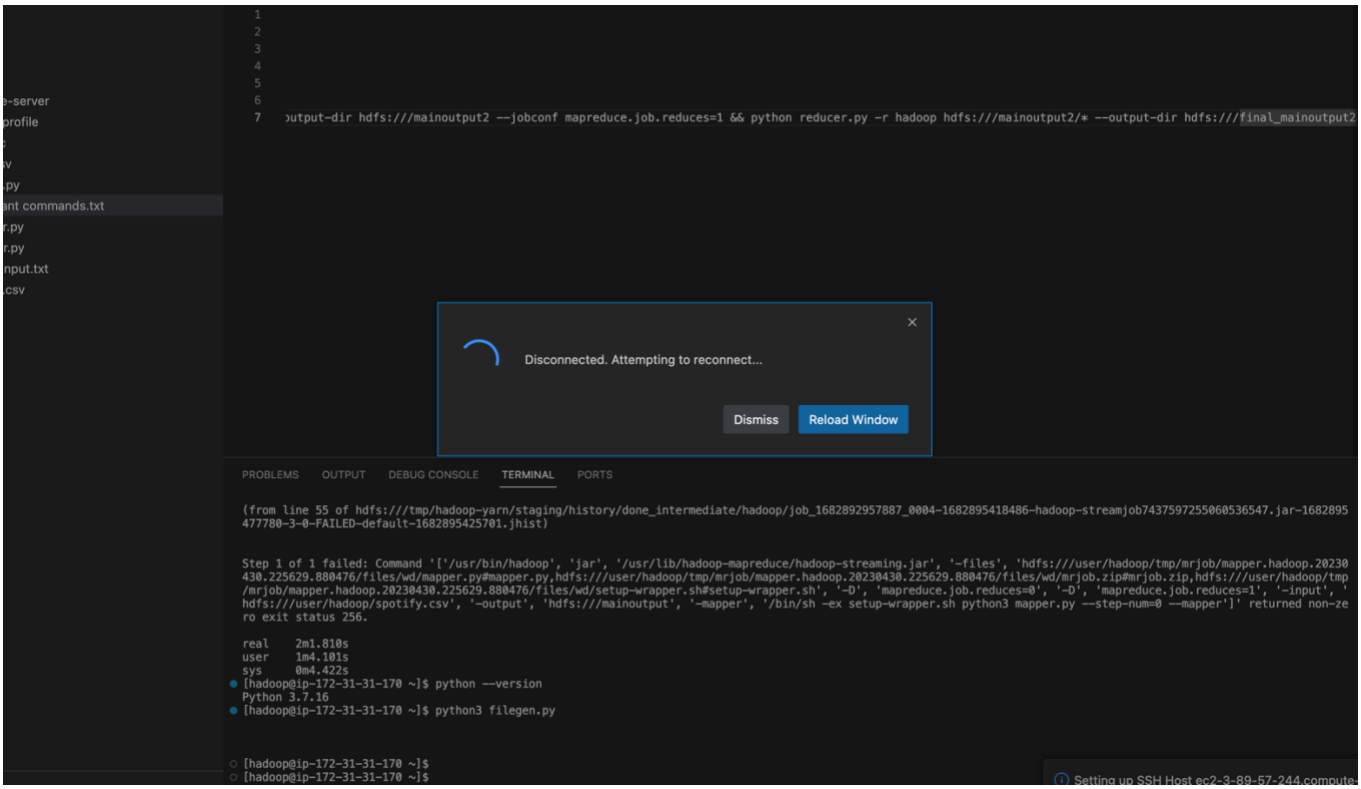
Executing the Spotify dataset from Kaggle

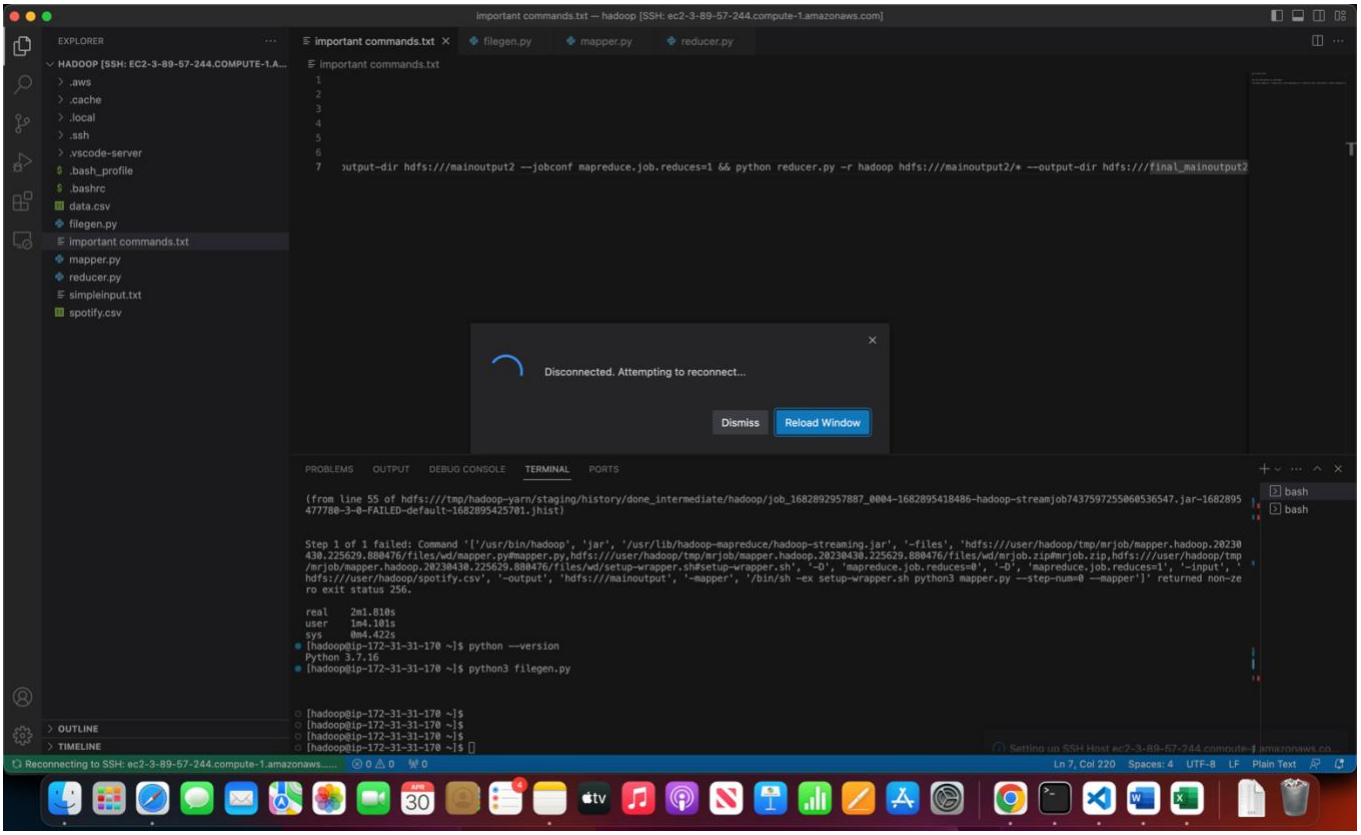
The EMR cluster starts reconnecting for the bigger size files. Soon after the cluster encounters it, it starts disconnecting and exits from backend because the file that was just being uploaded filled up the complete memory on the instance, though the cluster is still in the waiting status in the AWS console.

Therefore, I used random file generator option to create a file within the EC2 to opt for smaller size which could be stored on the EMR cluster without causing issues(Like the spotify dataset which was about 6GB which filled up the EBS volume and caused system reboot)

Please find the below screenshots for your reference.







Test cases:

Case 1: Empty file check.

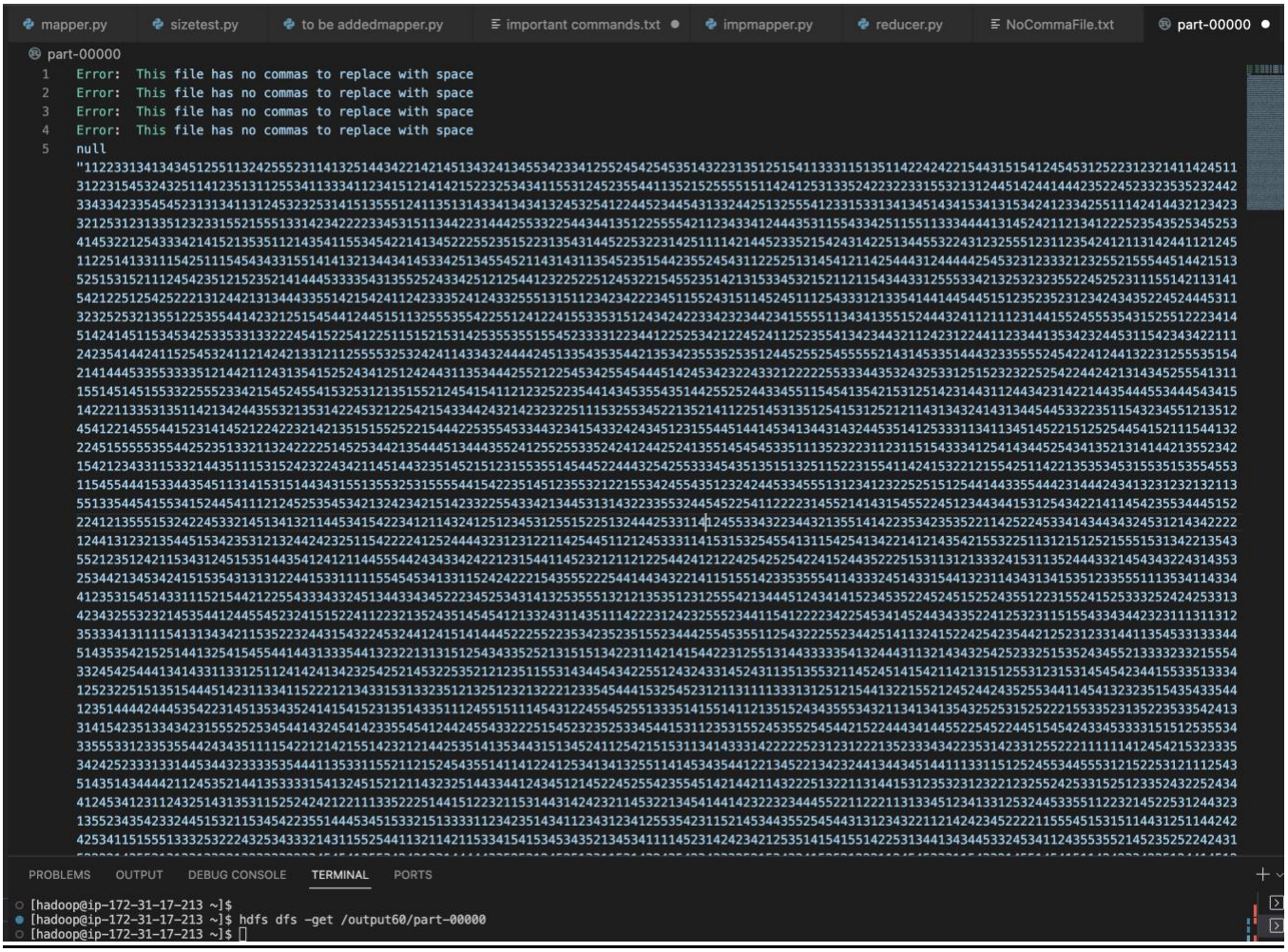
```
[hadoop@ip-172-31-17-213 ~]$ time python mapper.py -r hadoop hdfs:///user/hadoop/data.csv --output-dir hdfs:///output5 --jobconf mapreduce.job.reduces=1 && python reducer.py -r hadoop
 hdfs:///output5/* --output-dir hdfs:///final_output5
File is empty!

real    0m0.212s
user    0m0.174s
sys     0m0.024s
No configs found; falling back on auto-configuration
No configs specified for hadoop runner
Looking for hadoop binary in $PATH...
Found hadoop binary: /usr/bin/hadoop
```

Case 2: File size check to avoid EMR reboot.

```
[hadoop@ip-172-31-17-213 ~]$ time python mapper.py -r hadoop hdfs:///user/hadoop/empty.csv --output-dir hdfs:///output --jobconf mapreduce.job.reduces=1 && python reducer.py -r hadoop
 hdfs:///output/* --output-dir hdfs:///final_output
The File size acceptable, Pushed to HDFS.
No configs found; falling back on auto-configuration
No configs specified for hadoop runner
Looking for hadoop binary in $PATH...
Found hadoop binary: /usr/bin/hadoop
Using Hadoop version 2.10.1
Looking for Hadoop streaming jar in /home/hadoop/contrib...
Looking for Hadoop streaming jar in /usr/lib/hadoop-mapreduce...
```

Case 3: There no commas within the given file.

My test case fails here. The additional code that I tried to add does not seem to get the count. However, normally if there are no commas then there won't be any replace, and new file generated content will be same as old file.
In the absence of additional code for this part, the job runs successfully without making any modifications. The below screenshots reflects the run with additional code.



Conclusion:

I was able to successfully convert the file format from Comma Separated Values (CSV) to Tab Separated Values (TSV). I was able to rectify majority of the issues that was reflecting in the draft file such as /t was reflecting in the output file rather than actual tab. Currently I am successfully able to export the file where the contents are separated by tabs (instead of /t).

Reference:

- https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
- https://www.tutorialspoint.com/hadoop/hadoop_mapreduce.htm
- https://medium.com/@dhammikasamankumara/what-is-hadoop-distributed-file-system-hdfs-36a3503f9c60#:~:text=Hadoop%20distributed%20file%20System%20(HDFS)%20splits%20the%20large%20data%20files,result%20in%20data%20being%20unavailable