<u>ENGINEERING CLOUD COMPUTING: ASSIGNMENT 1(Report)</u>

<u>Name:</u> Aditi Gode
<u>Username:</u> adigode

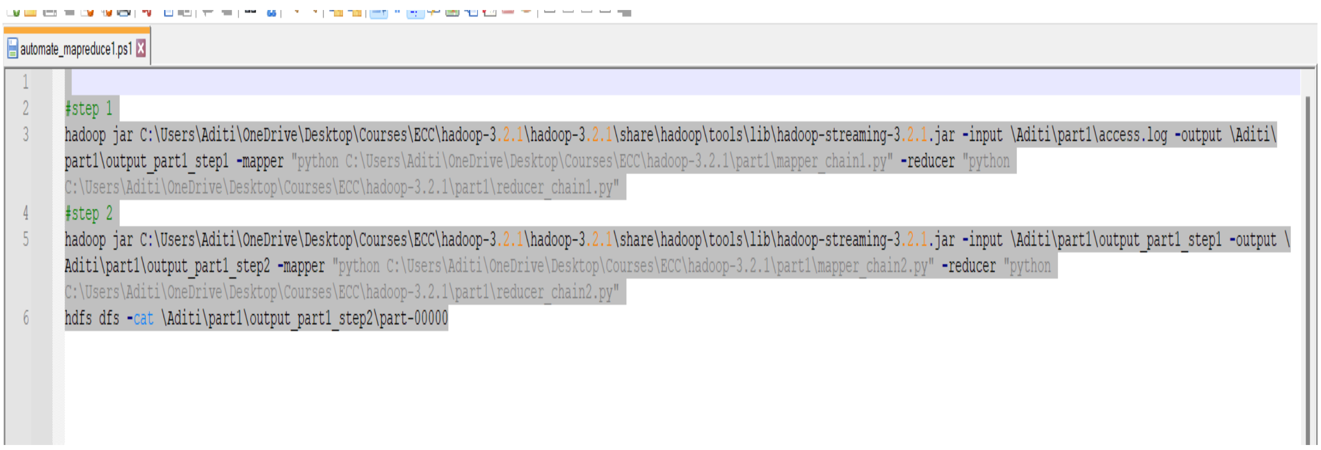
Note: I did all the tests for sample.log first, and then used the 3.2 GB access.log file. The output below is for access.log file(size:3.2 GB). I have zipped my part1, and part2 code together with this report.

<u>Part 1: Output the top-3 IP addresses with the granularity of an hour</u>

<u>Python code(mapper and reducer logic):</u>

Mapper 1 will simply take out each hour concatenated to the IP address and write a 1 next to it. Reducer 1 will add all occurrences of the same IP addresses occurring in the same hour. Mapper 2 will create a nested dictionary to store different IP addresses under each hour separately. Now, I could easily sort and take top 3 IP addresses in each hour. So, mapper 2 will find the local top 3 in each hour. Finally, reducer 2 will find the local top 3 in each hour. In the final output, we will get only top 3 IP addresses occurring in each hour.

For my windows machine, I created a script to run the two mappers and reducers and print the final output on the screen.


automate_mapreduce1.ps1

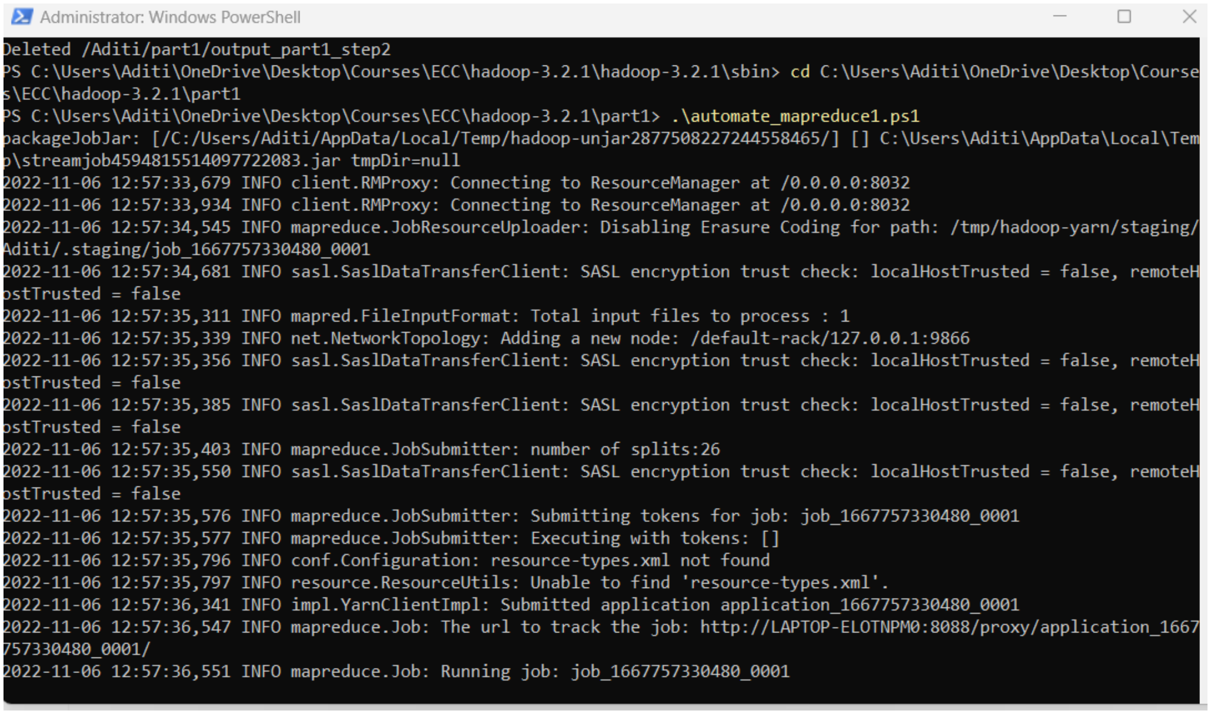
```
#step 1
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\access.log -output \Aditi\part1\output_part1_step1 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\mapper_chain1.py" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\reducer_chain1.py"
#step 2
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\output_part1_step1 -output \Aditi\part1\output_part1_step2 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\mapper_chain2.py" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\reducer_chain2.py"
hdfs dfs -cat \Aditi\part1\output_part1_step2\part-00000
```

<u>Hadoop command(because screenshot might not be clear):</u>

#round 1
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\access.log -output \Aditi\part1\output_part1_step1 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\mapper_chain1.py" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\reducer_chain1.py"

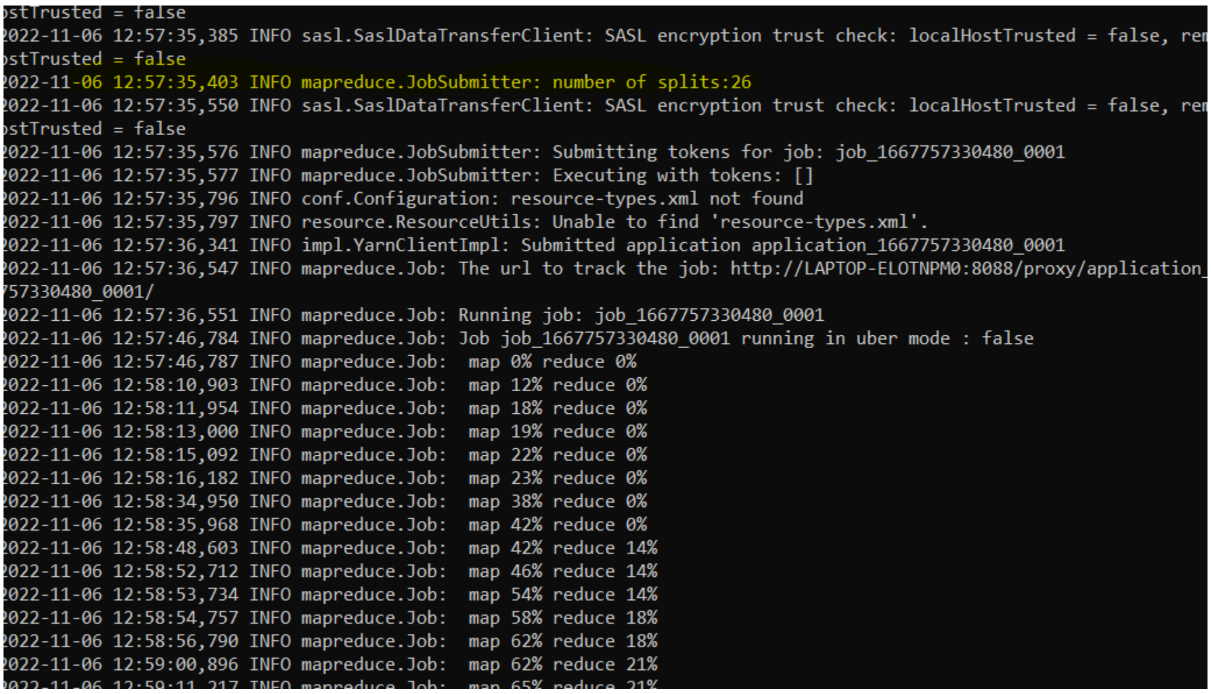
#round 2
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\output_part1_step1 -output \Aditi\part1\output_part1_step2 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\mapper_chain2.py" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1\reducer_chain2.py"

#print output on the screen
hdfs dfs -cat \Aditi\part1\output_part1_step2\part-00000


Administrator: Windows PowerShell

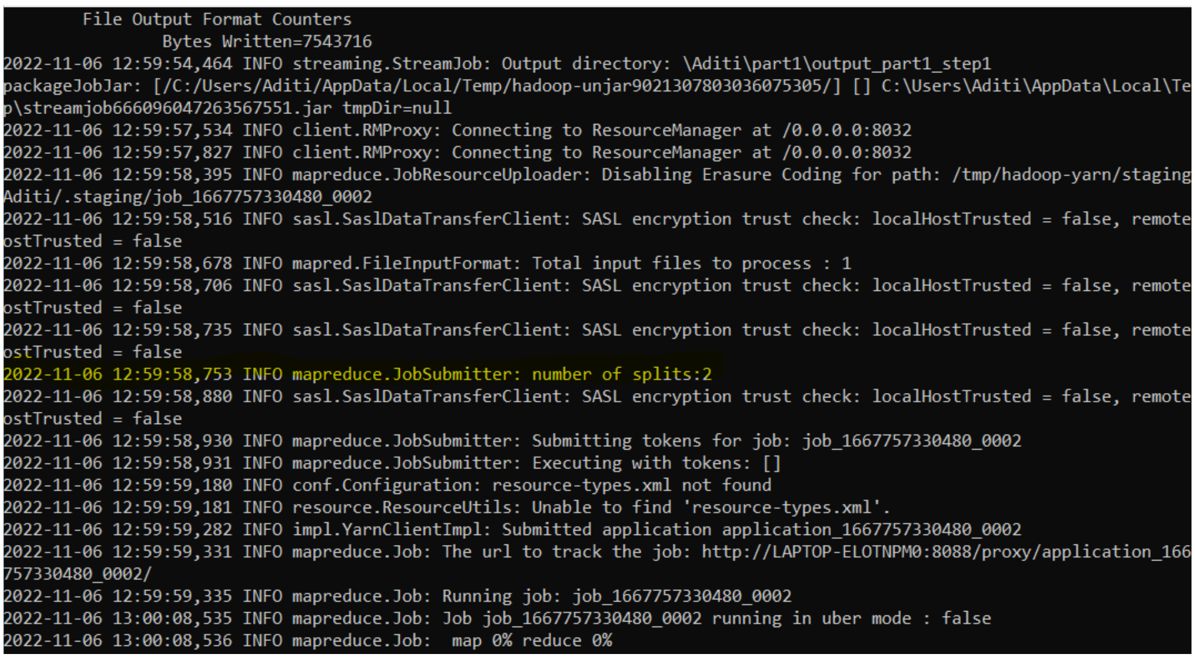
```
Deleted /Aditi/part1/output_part1_step2
PS C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\sbin> cd C:\Users\Aditi\OneDrive\Desktop\Course
s\ECC\hadoop-3.2.1\part1
PS C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part1> .\automate_mapreduce1.ps1
packageJobJar: [/C:/Users/Aditi/AppData/Local/Temp/hadoop-unjar2877508227244558465/] [] C:\Users\Aditi\AppData\Local\Tem
p\streamjob4594815514097722083.jar tmpDir=null
2022-11-06 12:57:33,679 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2022-11-06 12:57:33,934 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2022-11-06 12:57:34,545 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/
Aditi/.staging/job_1667757330480_0001
2022-11-06 12:57:34,681 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:57:35,311 INFO mapred.FileInputFormat: Total input files to process : 1
2022-11-06 12:57:35,339 INFO net.NetworkTopology: Adding a new node: /default-rack/127.0.0.1:9866
2022-11-06 12:57:35,356 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:57:35,385 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:57:35,403 INFO mapreduce.JobSubmitter: number of splits:26
2022-11-06 12:57:35,550 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:57:35,576 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1667757330480_0001
2022-11-06 12:57:35,577 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-11-06 12:57:35,796 INFO conf.Configuration: resource-types.xml not found
2022-11-06 12:57:35,797 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-11-06 12:57:36,341 INFO impl.YarnClientImpl: Submitted application application_1667757330480_0001
2022-11-06 12:57:36,547 INFO mapreduce.Job: The url to track the job: http://LAPTOP-ELOTNPM0:8088/proxy/application_1667
757330480_0001/
2022-11-06 12:57:36,551 INFO mapreduce.Job: Running job: job_1667757330480_0001
```

For the first mapper, the number of splits were 26, resulting in 26 mapper tasks



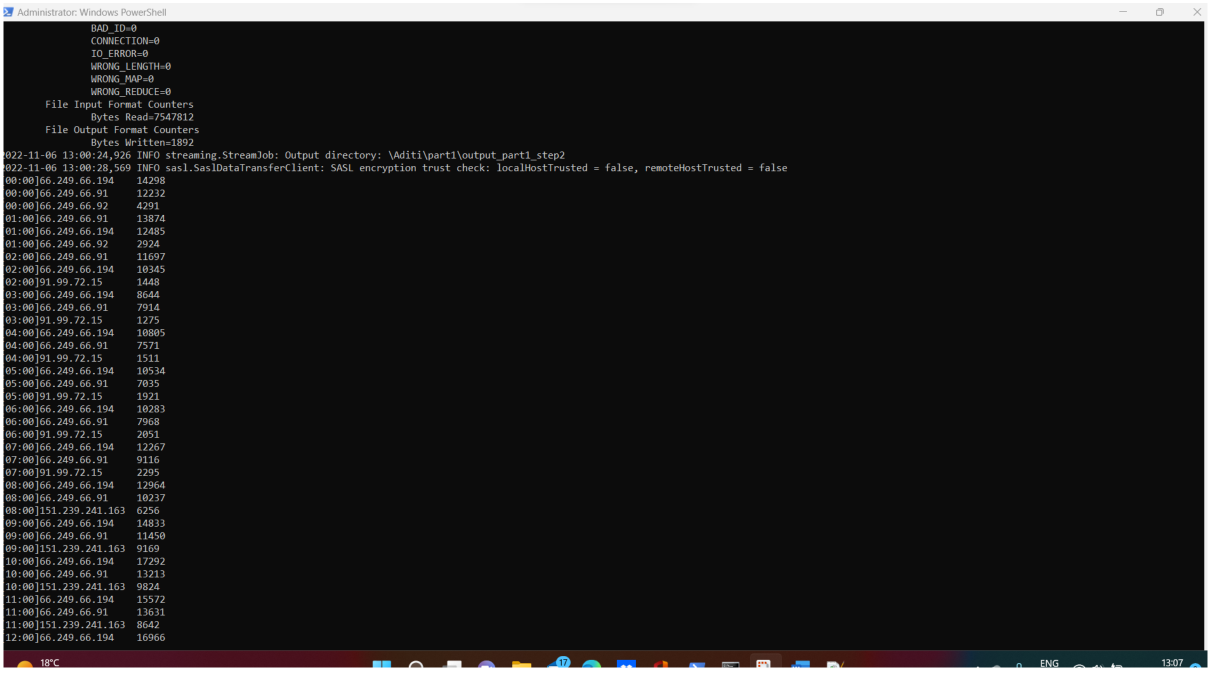
```
ostTrusted = false
2022-11-06 12:57:35,385 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, rem
ostTrusted = false
2022-11-06 12:57:35,403 INFO mapreduce.JobSubmitter: number of splits:26
2022-11-06 12:57:35,550 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, rem
ostTrusted = false
2022-11-06 12:57:35,576 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1667757330480_0001
2022-11-06 12:57:35,577 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-11-06 12:57:35,796 INFO conf.Configuration: resource-types.xml not found
2022-11-06 12:57:35,797 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-11-06 12:57:36,341 INFO impl.YarnClientImpl: Submitted application application_1667757330480_0001
2022-11-06 12:57:36,547 INFO mapreduce.Job: The url to track the job: http://LAPTOP-ELOTNPM0:8088/proxy/application_
757330480_0001/
2022-11-06 12:57:36,551 INFO mapreduce.Job: Running job: job_1667757330480_0001
2022-11-06 12:57:46,784 INFO mapreduce.Job: Job job_1667757330480_0001 running in uber mode : false
2022-11-06 12:57:46,787 INFO mapreduce.Job:  map 0% reduce 0%
2022-11-06 12:58:10,903 INFO mapreduce.Job:  map 12% reduce 0%
2022-11-06 12:58:11,954 INFO mapreduce.Job:  map 18% reduce 0%
2022-11-06 12:58:13,000 INFO mapreduce.Job:  map 19% reduce 0%
2022-11-06 12:58:15,092 INFO mapreduce.Job:  map 22% reduce 0%
2022-11-06 12:58:16,182 INFO mapreduce.Job:  map 23% reduce 0%
2022-11-06 12:58:34,950 INFO mapreduce.Job:  map 38% reduce 0%
2022-11-06 12:58:35,968 INFO mapreduce.Job:  map 42% reduce 0%
2022-11-06 12:58:48,603 INFO mapreduce.Job:  map 42% reduce 14%
2022-11-06 12:58:52,712 INFO mapreduce.Job:  map 46% reduce 14%
2022-11-06 12:58:53,734 INFO mapreduce.Job:  map 54% reduce 14%
2022-11-06 12:58:54,757 INFO mapreduce.Job:  map 58% reduce 18%
2022-11-06 12:58:56,790 INFO mapreduce.Job:  map 62% reduce 18%
2022-11-06 12:59:00,896 INFO mapreduce.Job:  map 62% reduce 21%
2022-11-06 12:59:11,217 INFO mapreduce.Job:  map 65% reduce 21%
```

For the second mapper, the number of splits were 2



```
        File Output Format Counters
                Bytes Written=7543716
2022-11-06 12:59:54,464 INFO streaming.StreamJob: Output directory: \Aditi\part1\output_part1_step1
packageJobJar: [/C:/Users/Aditi/AppData/Local/Temp/hadoop-unjar9021307803036075305/] [] C:\Users\Aditi\AppData\Local\Te
p\streamjob666096047263567551.jar tmpDir=null
2022-11-06 12:59:57,534 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2022-11-06 12:59:57,827 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2022-11-06 12:59:58,395 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging
Aditi/.staging/job_1667757330480_0002
2022-11-06 12:59:58,516 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remote
ostTrusted = false
2022-11-06 12:59:58,678 INFO mapred.FileInputFormat: Total input files to process : 1
2022-11-06 12:59:58,706 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remote
ostTrusted = false
2022-11-06 12:59:58,735 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remote
ostTrusted = false
2022-11-06 12:59:58,753 INFO mapreduce.JobSubmitter: number of splits:2
2022-11-06 12:59:58,880 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remote
ostTrusted = false
2022-11-06 12:59:58,930 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1667757330480_0002
2022-11-06 12:59:58,931 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-11-06 12:59:59,180 INFO conf.Configuration: resource-types.xml not found
2022-11-06 12:59:59,181 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-11-06 12:59:59,282 INFO impl.YarnClientImpl: Submitted application application_1667757330480_0002
2022-11-06 12:59:59,331 INFO mapreduce.Job: The url to track the job: http://LAPTOP-ELOTNPM0:8088/proxy/application_166
757330480_0002/
2022-11-06 12:59:59,335 INFO mapreduce.Job: Running job: job_1667757330480_0002
2022-11-06 13:00:08,535 INFO mapreduce.Job: Job job_1667757330480_0002 running in uber mode : false
2022-11-06 13:00:08,536 INFO mapreduce.Job:  map 0% reduce 0%
```

Output:



Administrator: Windows PowerShell

```
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=7547812
        File Output Format Counters
                Bytes Written=1892
022-11-06 13:00:24,926 INFO streaming.StreamJob: Output directory: \Aditi\part1\output_part1_step2
022-11-06 13:00:28,569 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
00:00]66.249.66.194    14298
00:00]66.249.66.91     12232
00:00]66.249.66.92     4291
01:00]66.249.66.91     13874
01:00]66.249.66.194    12485
01:00]66.249.66.92     2924
02:00]66.249.66.91     11697
02:00]66.249.66.194    10345
02:00]91.99.72.15      1448
03:00]66.249.66.194    8644
03:00]66.249.66.91     7914
03:00]91.99.72.15      1275
04:00]66.249.66.194    10805
04:00]66.249.66.91     7571
04:00]91.99.72.15      1511
05:00]66.249.66.194    10534
05:00]66.249.66.91     7035
05:00]91.99.72.15      1921
06:00]66.249.66.194    10283
06:00]66.249.66.91     7968
06:00]91.99.72.15      2051
07:00]66.249.66.194    12267
07:00]66.249.66.91     9116
07:00]91.99.72.15      2295
08:00]66.249.66.194    12964
08:00]66.249.66.91     10237
08:00]151.239.241.163  6256
09:00]66.249.66.194    14833
09:00]66.249.66.91     11450
09:00]151.239.241.163  9169
10:00]66.249.66.194    17292
10:00]66.249.66.91     13213
10:00]151.239.241.163  9824
11:00]66.249.66.194    15572
11:00]66.249.66.91     13631
11:00]151.239.241.163  8642
12:00]66.249.66.194    16966
```

As you see, I got the top 3 IP addresses in each hour.

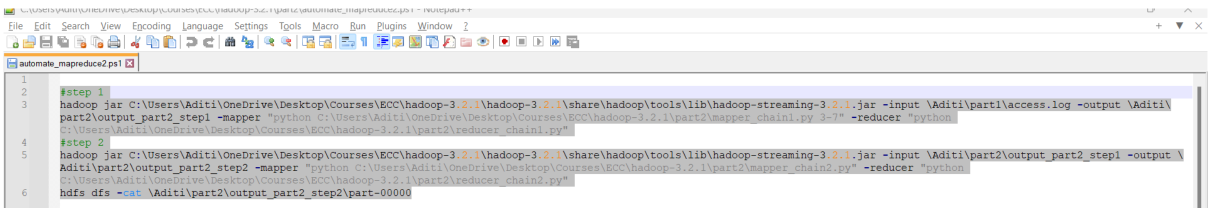
Part 2a: Make the program like a database search to only output top 3 IP addresses in the user given range

Python code(mapper and reducer logic):

In this, I added a filter to the first mapper to only include IP addresses in the selected time range(which will be given by the user) as a database search.

The second mapper just had the task of sorted the IP addresses in the time range locally to give to the reducer. Each mapper will give its local top 3 addresses to the reducer. The reducer will find the global top 3 and will write the final output to a file.

For my windows machine, I created a script to run the two mappers and reducers and print the final output on the screen.



C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\automate_mapreduce2.ps1 - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
automate_mapreduce2.ps1

```

#step 1
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\access.log -output \Aditi\
part2\output_part2_step1 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\mapper_chain1.py 3-7" -reducer "python
C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\reducer_chain1.py"
#step 2
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part2\output_part2_step1 -output \
Aditi\part2\output_part2_step2 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\mapper_chain2.py" -reducer "python
C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\reducer_chain2.py"
hdfs dfs -cat \Aditi\part2\output_part2_step2\part-00000
```

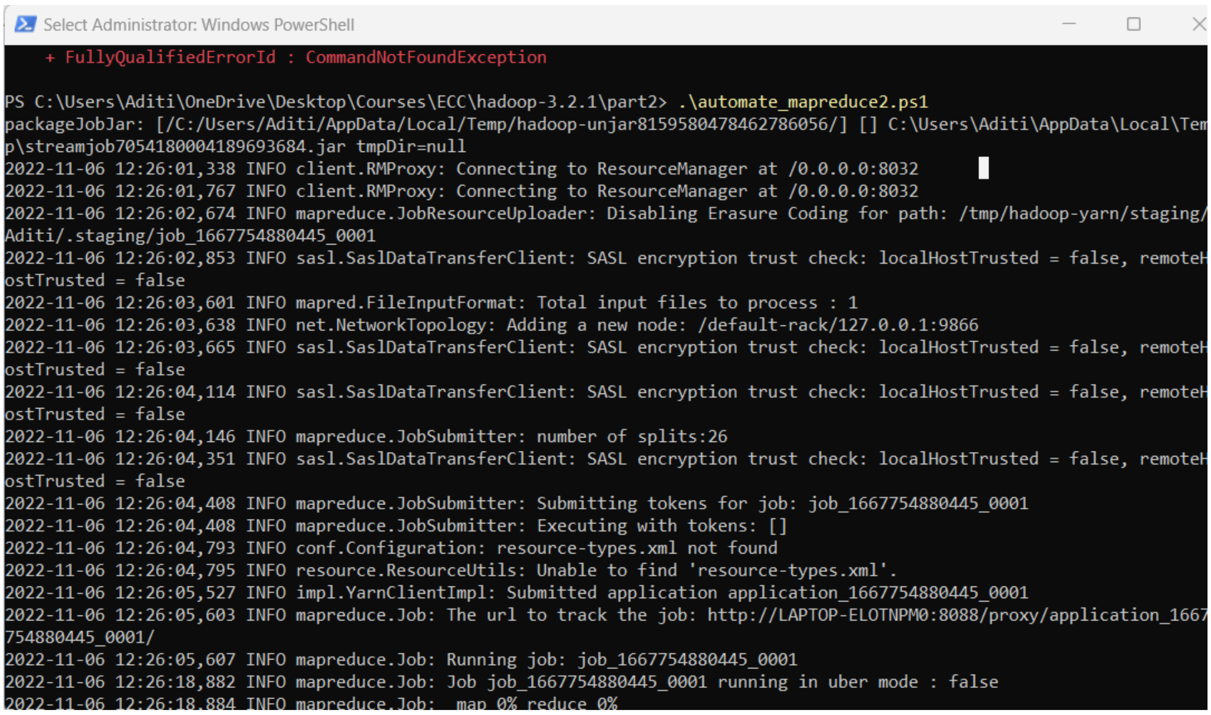
Commands:

#round 1
#Note: User gives the range in the first mapper which is passed as a command line argument to the mapper
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\access.log -output \Aditi\part2\output_part2_step1 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\mapper_chain1.py 3-7" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\reducer_chain1.py"

#round 2
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part2\output_part2_step1 -output \Aditi\part2\output_part2_step2 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\mapper_chain2.py" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2\reducer_chain2.py"

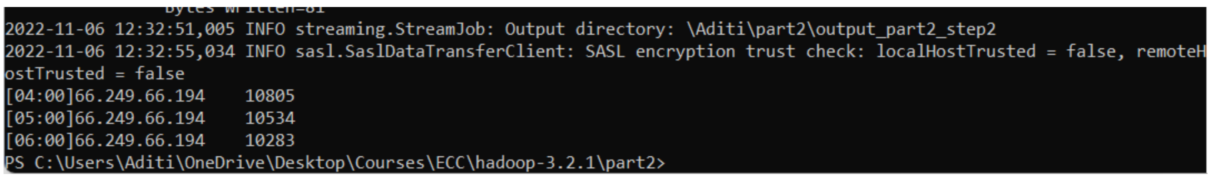
#printing the final output to the screen
hdfs dfs -cat \Aditi\part2\output_part2_step2\part-00000

Running the script on Hadoop:



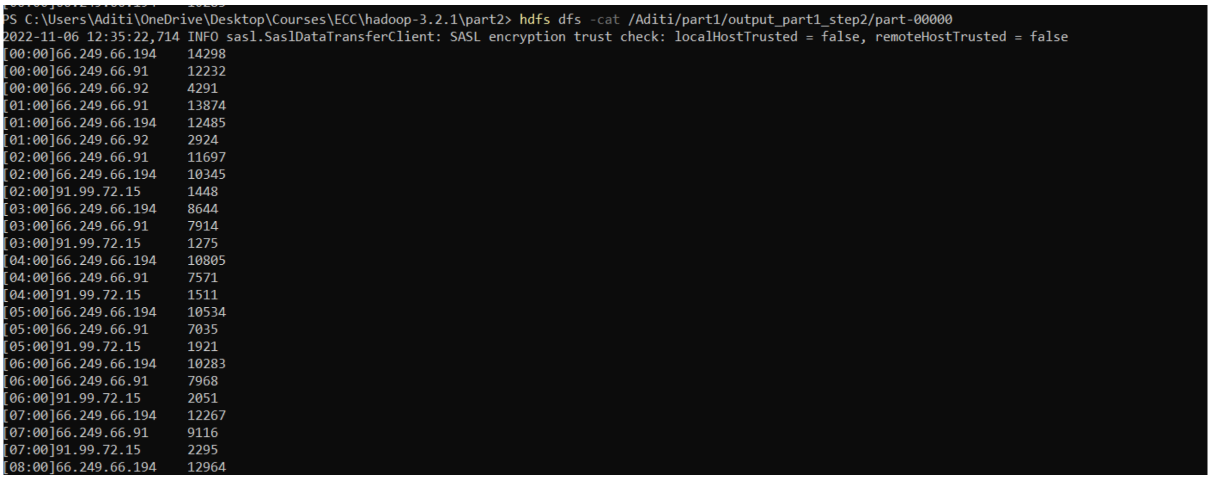
```
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2> .\automate_mapreduce2.ps1
packageJobJar: [/C:/Users/Aditi/AppData/Local/Temp/hadoop-unjar8159580478462786056/] [] C:\Users\Aditi\AppData\Local\Tem
p\streamjob7054180004189693684.jar tmpDir=null
2022-11-06 12:26:01,338 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2022-11-06 12:26:01,767 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2022-11-06 12:26:02,674 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/
Aditi/.staging/job_1667754880445_0001
2022-11-06 12:26:02,853 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:26:03,601 INFO mapred.FileInputFormat: Total input files to process : 1
2022-11-06 12:26:03,638 INFO net.NetworkTopology: Adding a new node: /default-rack/127.0.0.1:9866
2022-11-06 12:26:03,665 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:26:04,114 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:26:04,146 INFO mapreduce.JobSubmitter: number of splits:26
2022-11-06 12:26:04,351 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
2022-11-06 12:26:04,408 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1667754880445_0001
2022-11-06 12:26:04,408 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-11-06 12:26:04,793 INFO conf.Configuration: resource-types.xml not found
2022-11-06 12:26:04,795 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-11-06 12:26:05,527 INFO impl.YarnClientImpl: Submitted application application_1667754880445_0001
2022-11-06 12:26:05,603 INFO mapreduce.Job: The url to track the job: http://LAPTOP-ELOTNPM0:8088/proxy/application_1667
754880445_0001/
2022-11-06 12:26:05,607 INFO mapreduce.Job: Running job: job_1667754880445_0001
2022-11-06 12:26:18,882 INFO mapreduce.Job: Job job_1667754880445_0001 running in uber mode : false
2022-11-06 12:26:18,884 INFO mapreduce.Job:  map 0% reduce 0%
```

Output: for top 3 ip addresses in the time range 3-7 are:



```
2022-11-06 12:32:51,005 INFO streaming.StreamJob: Output directory: \Aditi\part2\output_part2_step2
2022-11-06 12:32:55,034 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteH
ostTrusted = false
[04:00]66.249.66.194    10805
[05:00]66.249.66.194    10534
[06:00]66.249.66.194    10283
PS C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2>
```

We can verify this by taking a look at the output of the part1 for access.log file and manually checking if the top 3 entries are correct.



```
PS C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\part2> hdfs dfs -cat /Aditi/part1/output_part1_step2/part-00000
2022-11-06 12:35:22,714 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
[00:00]66.249.66.194    14298
[00:00]66.249.66.91     12232
[00:00]66.249.66.92     4291
[01:00]66.249.66.91     13874
[01:00]66.249.66.194    12485
[01:00]66.249.66.92     2924
[02:00]66.249.66.91     11697
[02:00]66.249.66.194    10345
[02:00]91.99.72.15      1448
[03:00]66.249.66.194    8644
[03:00]66.249.66.91     7914
[03:00]91.99.72.15      1275
[04:00]66.249.66.194    10805
[04:00]66.249.66.91     7571
[04:00]91.99.72.15      1511
[05:00]66.249.66.194    10534
[05:00]66.249.66.91     7035
[05:00]91.99.72.15      1921
[06:00]66.249.66.194    10283
[06:00]66.249.66.91     7968
[06:00]91.99.72.15      2051
[07:00]66.249.66.194    12267
[07:00]66.249.66.91     9116
[07:00]91.99.72.15      2295
[08:00]66.249.66.194    12964
```

This proves that the database search works just fine.

Part 2b: Run it along with other examples , wordcount, sort, Grep, at the same time.

Hadoop already has example programs for wordcount, sort, and grep. I planned to use those for all of these tasks but sort requires sequence format files as input, and there are no python references available online for doing that so I just used wordcount, and grep. I used my own program to do sorting.

I ran all these tasks at the same time for testing.

Commands:

For MapReduce part2 program(script)

```
.\automate_mapreduce2.ps1
```

For sorting:

```
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-
3.2.1\share\hadoop\tools\lib\hadoop-streaming-3.2.1.jar -input \Aditi\part1\sample.log -output
\Aditi\sorting2 -mapper "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-
3.2.1\test\mapper1.py" -reducer "python C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-
3.2.1\test\reducer1.py"
```

For grep:

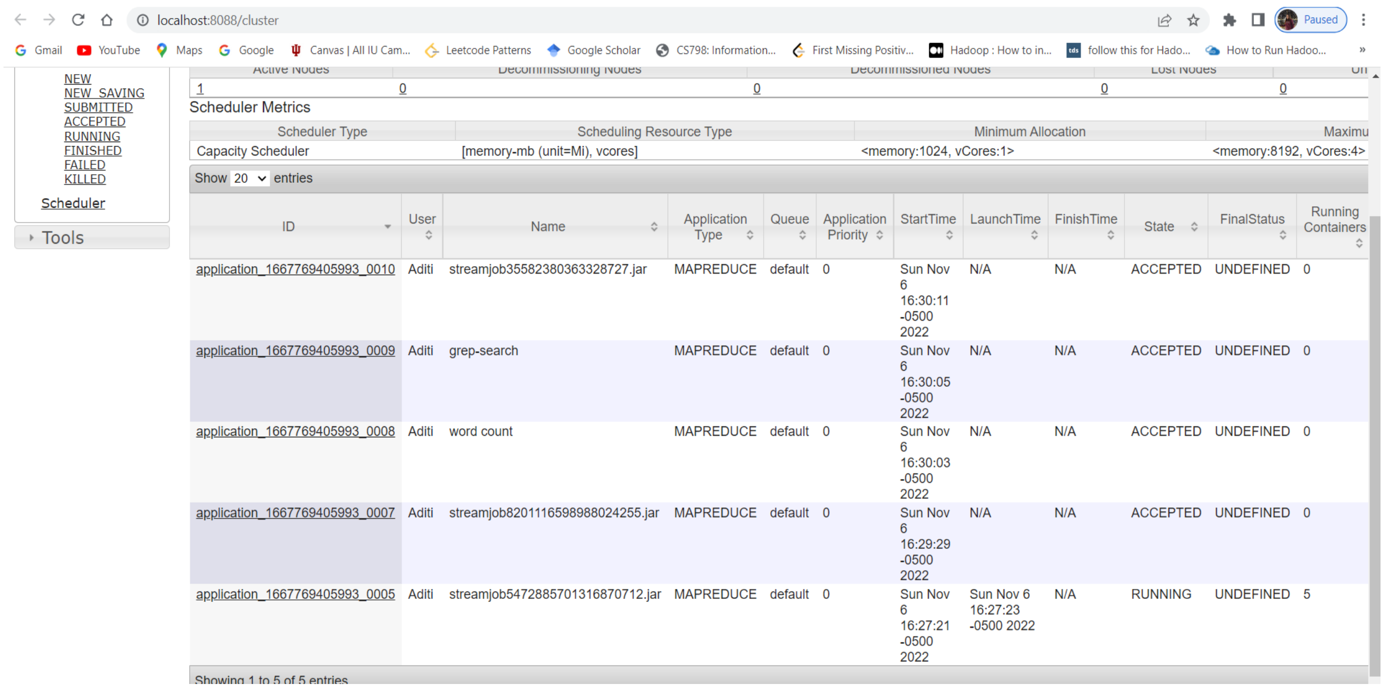
```
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-
3.2.1\share\hadoop\mapreduce\hadoop-mapreduce-examples-3.2.1.jar grep \Aditi\part2\files
/Aditi/grep2 "Hadoop"
```

For wordcount:

```
hadoop jar C:\Users\Aditi\OneDrive\Desktop\Courses\ECC\hadoop-3.2.1\hadoop-
3.2.1\share\hadoop\mapreduce\hadoop-mapreduce-examples-3.2.1.jar wordcount
\Aditi\part2\files /Aditi/worcount2
```

Part 2b: Testing fair and capacity schedulers:

1) Testing capacity scheduler



Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Un
1	0	0	0	0

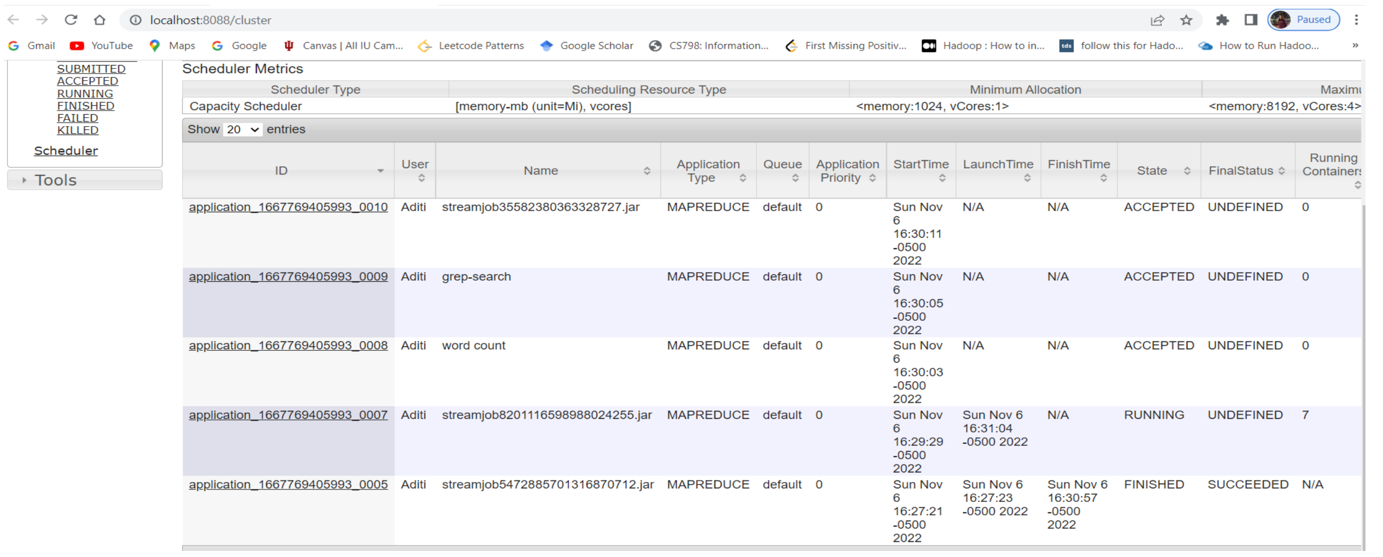
Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximu
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers
application_1667769405993_0010	Aditi	streamjob35582380363328727.jar	MAPREDUCE	default	0	Sun Nov 6 16:30:11 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0009	Aditi	grep-search	MAPREDUCE	default	0	Sun Nov 6 16:30:05 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0008	Aditi	word count	MAPREDUCE	default	0	Sun Nov 6 16:30:03 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0007	Aditi	streamjob8201116598988024255.jar	MAPREDUCE	default	0	Sun Nov 6 16:29:29 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0005	Aditi	streamjob5472885701316870712.jar	MAPREDUCE	default	0	Sun Nov 6 16:27:21 -0500 2022	Sun Nov 6 16:27:23 -0500 2022	N/A	RUNNING	UNDEFINED	5

Showing 1 to 5 of 5 entries



Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximu
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers
application_1667769405993_0010	Aditi	streamjob35582380363328727.jar	MAPREDUCE	default	0	Sun Nov 6 16:30:11 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0009	Aditi	grep-search	MAPREDUCE	default	0	Sun Nov 6 16:30:05 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0008	Aditi	word count	MAPREDUCE	default	0	Sun Nov 6 16:30:03 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667769405993_0007	Aditi	streamjob8201116598988024255.jar	MAPREDUCE	default	0	Sun Nov 6 16:29:29 -0500 2022	Sun Nov 6 16:31:04 -0500 2022	N/A	RUNNING	UNDEFINED	7
application_1667769405993_0005	Aditi	streamjob5472885701316870712.jar	MAPREDUCE	default	0	Sun Nov 6 16:27:21 -0500 2022	Sun Nov 6 16:27:23 -0500 2022	Sun Nov 6 16:30:57 -0500 2022	FINISHED	SUCCEEDED	N/A

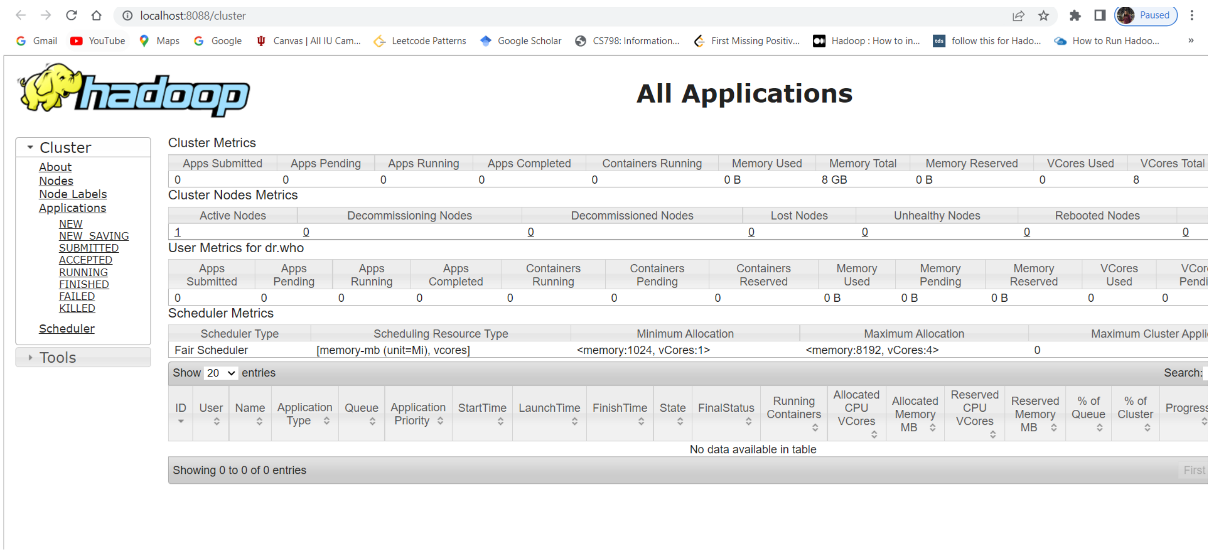
As you see, the by default scheduler is Capacity scheduler. Capacity scheduler completes one task before moving onto the second. Queues are used in capacity schedulers because a single large cluster can be shared by many groups in the organization.

However, organizations are worried about using capacity schedulers as they have to share a cluster, and SLA-critical resources will have to wait a long due to someone else using the resources in the same cluster. However, the benefits of using a shared cluster is that, anyone can use the excess resources that are not being use as there is no fixed resources allocated to anyone.

2) **Testing fair scheduler:**

I edited my yarn-site.xml file to enable fair scheduler. Reference is linked below.

No jobs running yet:



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total
0	0	0	0	0	0 B	8 GB	0 B	0	8

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	
1	0	0	0	0	0	0

User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	VCores Used	VCor Pendi
0	0	0	0	0	0	0	0 B	0 B	0 B	0	0

Scheduler Metrics

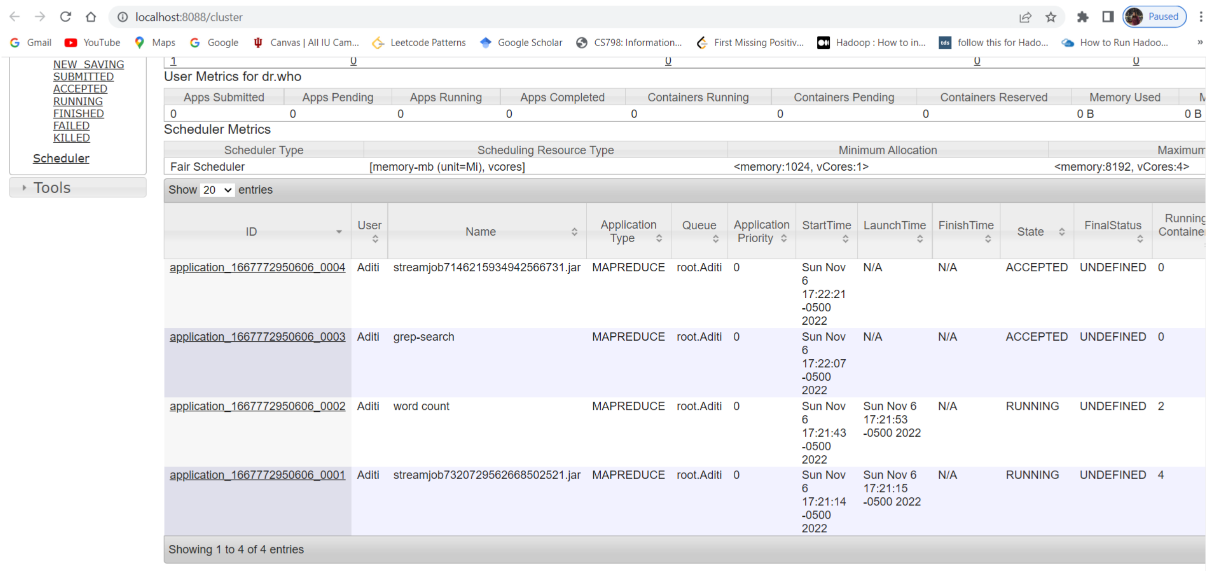
Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Appli
Fair Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries Search:

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Reserved CPU VCores	Reserved Memory MB	% of Queue	% of Cluster	Progress
No data available in table																		

Showing 0 to 0 of 0 entries First

After running part2 program, wordcount, grep, and sort at the same time



1	0	0	0	0

User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	M
0	0	0	0	0	0	0	0 B	0 B

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum
Fair Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containe
application_1667772950606_0004	Aditi	streamjob7146215934942566731.jar	MAPREDUCE	root.Aditi	0	Sun Nov 6 17:22:21 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667772950606_0003	Aditi	grep-search	MAPREDUCE	root.Aditi	0	Sun Nov 6 17:22:07 -0500 2022	N/A	N/A	ACCEPTED	UNDEFINED	0
application_1667772950606_0002	Aditi	word count	MAPREDUCE	root.Aditi	0	Sun Nov 6 17:21:43 -0500 2022	Sun Nov 6 17:21:53 -0500 2022	N/A	RUNNING	UNDEFINED	2
application_1667772950606_0001	Aditi	streamjob7320729562668502521.jar	MAPREDUCE	root.Aditi	0	Sun Nov 6 17:21:14 -0500 2022	Sun Nov 6 17:21:15 -0500 2022	N/A	RUNNING	UNDEFINED	4

Showing 1 to 4 of 4 entries

As you can see that, as soon as task 1 resources got free, task 2 was given those resources, enabling fair scheduling

Fair schedulers are such that all jobs get equal resources on average. When one single job is running, the job can use the entire cluster, but when other jobs are submitted, the resources that become available can be immediately used by the other jobs. This ensures that all jobs get equal amounts of CPU time.

Challenges:

I am just going to list all the challenges I faced while doing this assignment for your reference:

1) I decided to use Python for Hadoop, and later realized that there are no resources online to chain multiple mappers and reducers in Hadoop streaming. The resources available online for Hadoop streaming did not help much. Programming in Java has multiple sources online and it is also possible to chain mappers and reducers in the same code.
2) I wanted to use JetStream2 for this assignment but all 25 instances were occupied by other people in the class and there was no room for me.

References:

[1]https://towardsdatascience.com/installing-hadoop-3-2-1-single-node-cluster-on-windows-10-ac258dd48aef

[2]https://hadoop.apache.org/docs/r2.8.0/hadoop-streaming/HadoopStreaming.html

[3]https://prwatech.in/blog/hadoop/hadoop-fair-scheduler-tutorial/