Abhay Patil Big Data Exam

ID : 240840325002

HIVE : (20 mrks)

Q1.

A]

```
Time taken: 99.038 seconds, Fetched: 3254 row(s)
hive (abyyyairproject)> select distinct(ap.name)
                      >
                      > from airport ap
                      >
                      > join routes r1
                      >
                      > on r1.src_id = ap.airport_id
                      >
                      > join routes r2
                      >
                      > on r2.dest_id = ap.airport_id
                      >
                      > where r1.src_id = r2.dest_id
                      >
                      > limit 10;
No Stats for abyyyairproject@airport, Columns: airport_id, name
No Stats for abyyyairproject@routes, Columns: src_id
No Stats for abyyyairproject@routes, Columns: dest_id
Query ID = cdacuser90115_20241121084703_a2b7d650-23c7-43fe-a0d1-0a427154ea4c
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks not specified. Defaulting to jobconf value of: 4
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_1732089968849_2277, Tracking URL = http://master:6318/proxy/application_1732089968849_2277/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2277
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 4
2024-11-21 08:47:17,477 Stage-1 map = 0%,  reduce = 0%
2024-11-21 08:47:26,704 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 6.05 sec
2024-11-21 08:47:27,728 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 13.06 sec
2024-11-21 08:47:32,841 Stage-1 map = 100%,  reduce = 50%, Cumulative CPU 20.71 sec
2024-11-21 08:47:34,887 Stage-1 map = 100%,  reduce = 75%, Cumulative CPU 24.41 sec
2024-11-21 08:47:35,900 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 28.38 sec
```

```
2024-11-21 08:48:08,518 Stage-2 map = 100%,  reduce = 50%, Cumulative CPU 26.69 sec
2024-11-21 08:48:10,565 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 34.7 sec
MapReduce Total cumulative CPU time: 34 seconds 700 msec
Ended Job = job_1732089968849_2282
Launching Job 3 out of 3
Number of reduce tasks not specified. Defaulting to jobconf value of: 4
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_1732089968849_2285, Tracking URL = http://master:6318/proxy/application_1732089968849_2285/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2285
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 4
2024-11-21 08:48:21,992 Stage-3 map = 0%,  reduce = 0%
2024-11-21 08:48:28,143 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 3.1 sec
2024-11-21 08:48:34,278 Stage-3 map = 100%,  reduce = 25%, Cumulative CPU 6.22 sec
2024-11-21 08:48:36,322 Stage-3 map = 100%,  reduce = 100%, Cumulative CPU 15.67 sec
MapReduce Total cumulative CPU time: 15 seconds 670 msec
Ended Job = job_1732089968849_2285
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 4   Cumulative CPU: 28.38 sec   HDFS Read: 3148619 HDFS Write: 2543075 SUCCESS
Stage-Stage-2: Map: 3  Reduce: 4   Cumulative CPU: 34.7 sec   HDFS Read: 4954022 HDFS Write: 105775 SUCCESS
Stage-Stage-3: Map: 1  Reduce: 4   Cumulative CPU: 15.67 sec   HDFS Read: 124648 HDFS Write: 587 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 18 seconds 750 msec
OK
Aarhus
Abakan
Abbotsford
A Coruna
Abadan
Abdul Rachman Saleh
Abel Santamaria
Aalborg
Aasiaat
Aberdeen Regional Airport
Time taken: 95.573 seconds, Fetched: 10 row(s)
hive (abyyyairproject)>
```

select distinct(ap.name)

```
from airport ap
join routes r1
on r1.src_id = ap.airport_id
join routes r2
on r2.dest_id = ap.airport_id
where r1.src_id = r2.dest_id
limit 10;
```

B]

```
with (select max(equipment) from routes) as man_count,
(select equipment, count(equipment)
from routes
group by equipment
having count(equipment) = max_count);
```

“Can't get the output due to crashing of the hive terminal.”

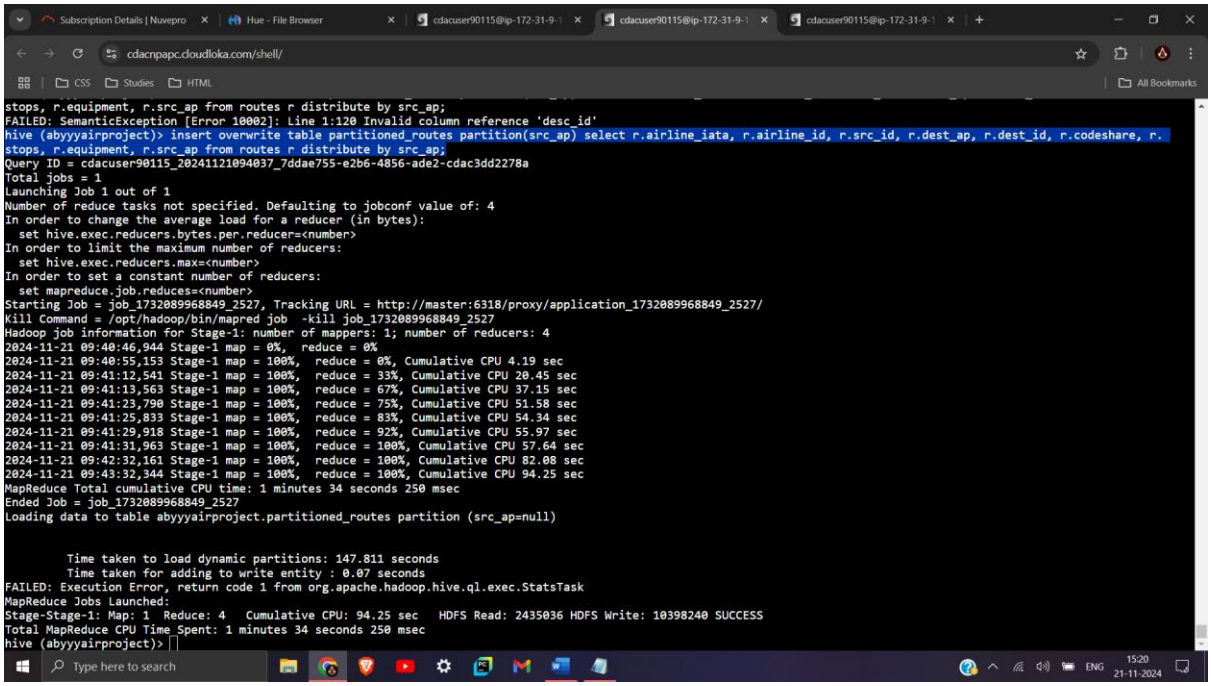
C]

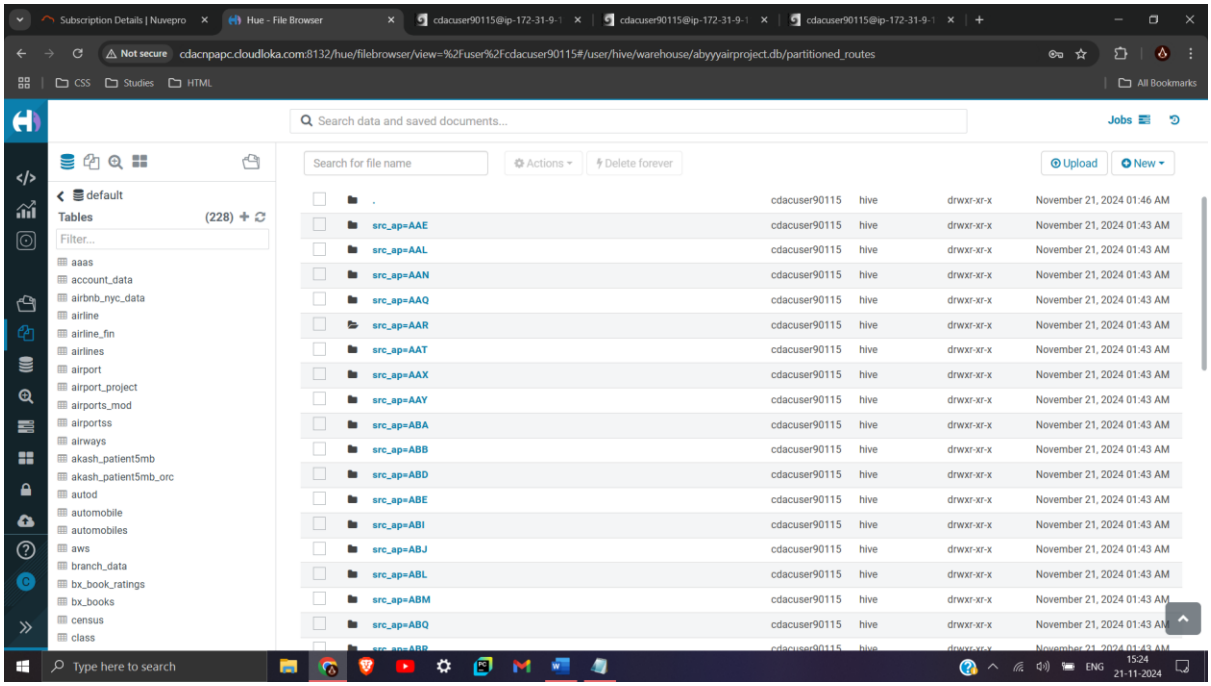
```
select a.name, count(*)
from airlines a
join routes r
on a.airline_id = r.airline_id
group by a.name
having max(count(*)) = count(*);
```

“Can't get the output due to crashing of the hive terminal.”

Q2.

A]





create table partitioned_routes (airline_iata string, airline_id int, src_id int, dest_ap string, dest_id int,

codeshare string, stops int, equipment string)

partitioned by (src_ap string)

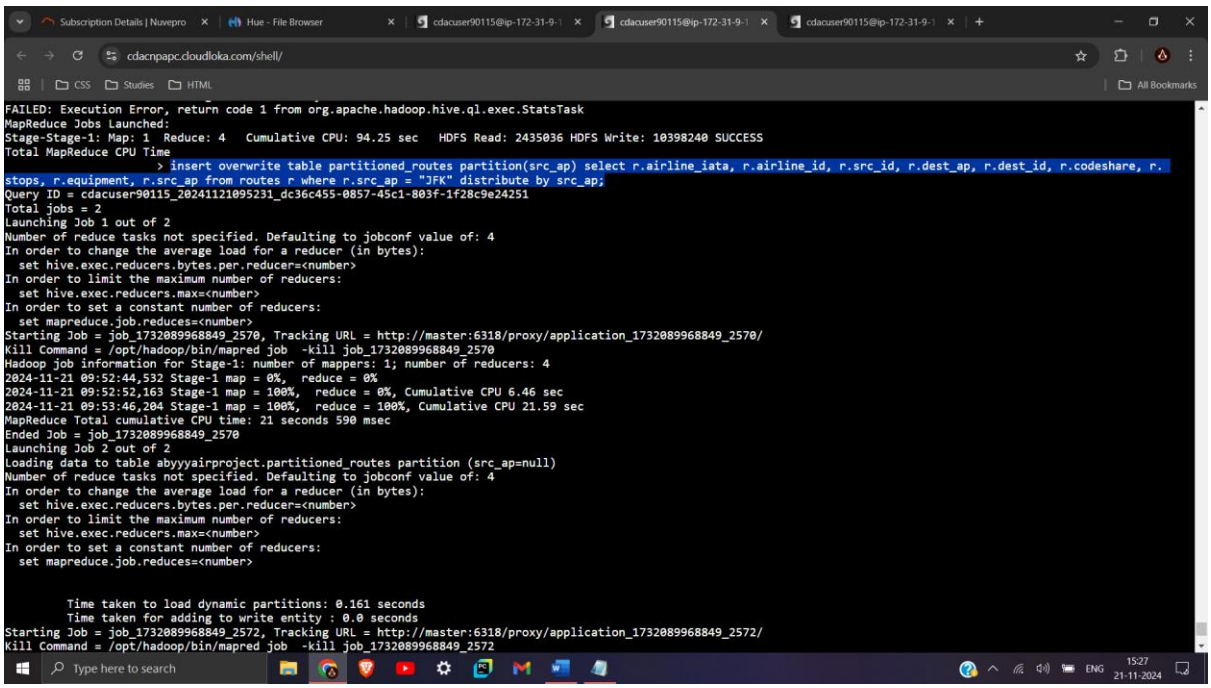
row format delimited

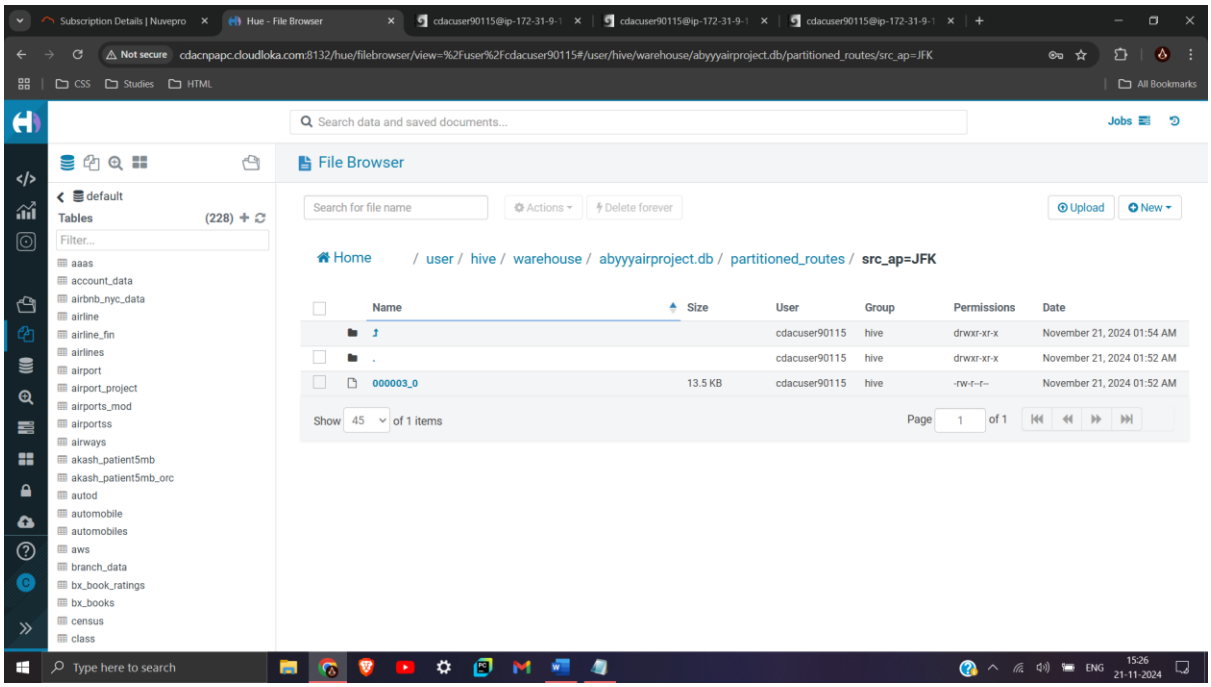
fields terminated by ','

stored as textfile;

insert overwrite table partitioned_routes partition(src_ap) select r.airline_iata, r.airline_id, r.src_id, r.dest_ap, r.dest_id, r.codeshare, r.stops, r.equipment, r.src_ap from routes r distribute by src_ap;

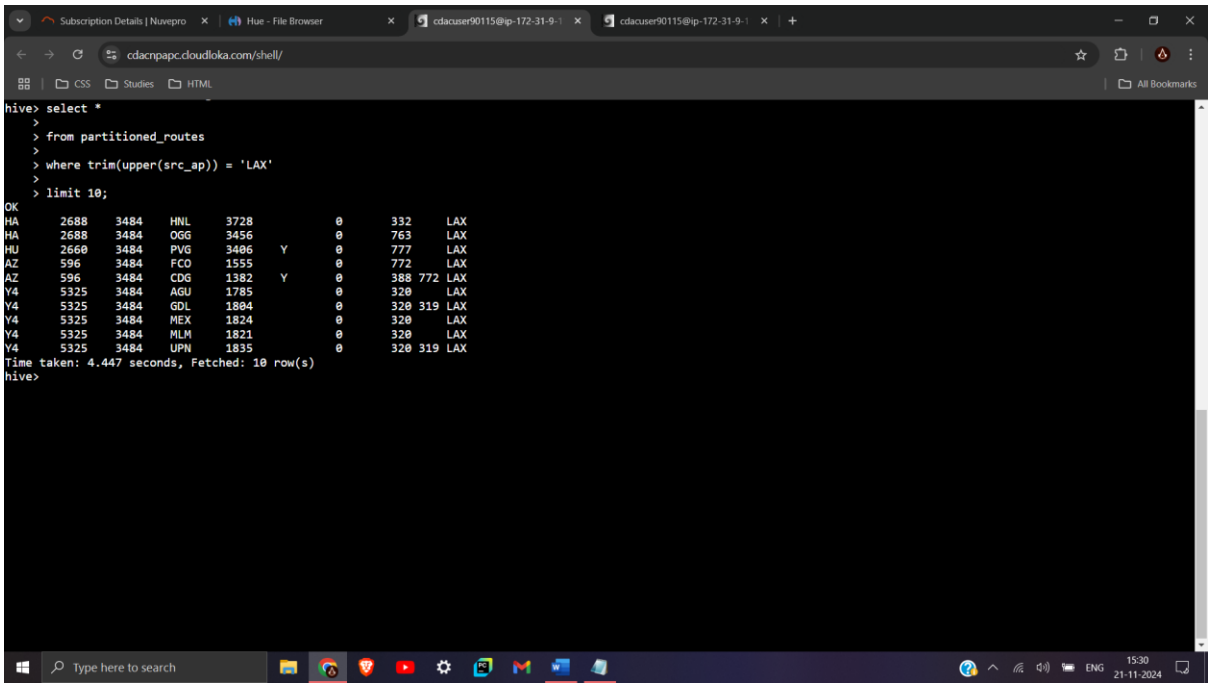
B]





insert overwrite table partitioned_routes partition(src_ap) select r.airline_iata, r.airline_id, r.src_id, r.dest_ap, r.dest_id, r.codeshare, r.stops, r.equipment, r.src_ap from routes r where r.src_ap = "JFK" distribute by src_ap;

C]



select *

from partitioned_routes

where trim(upper(src_ap)) = 'LAX'

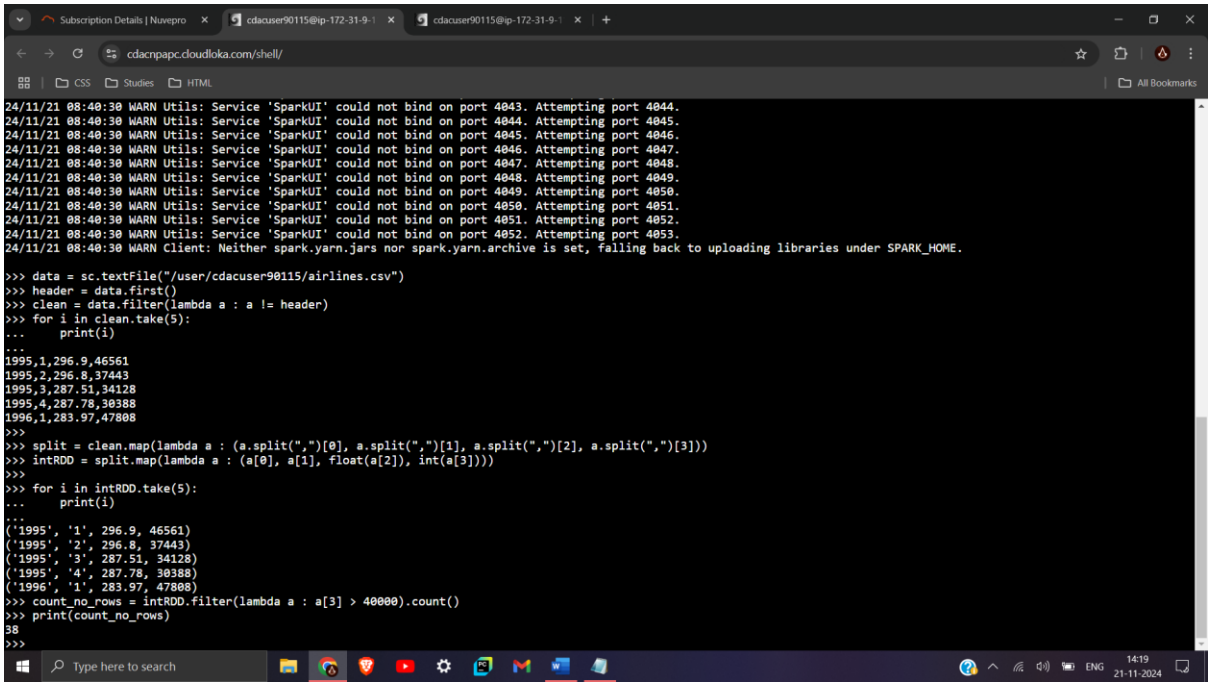
limit 10;

D]

SPARK : (20 mrks)

Q1.

A]



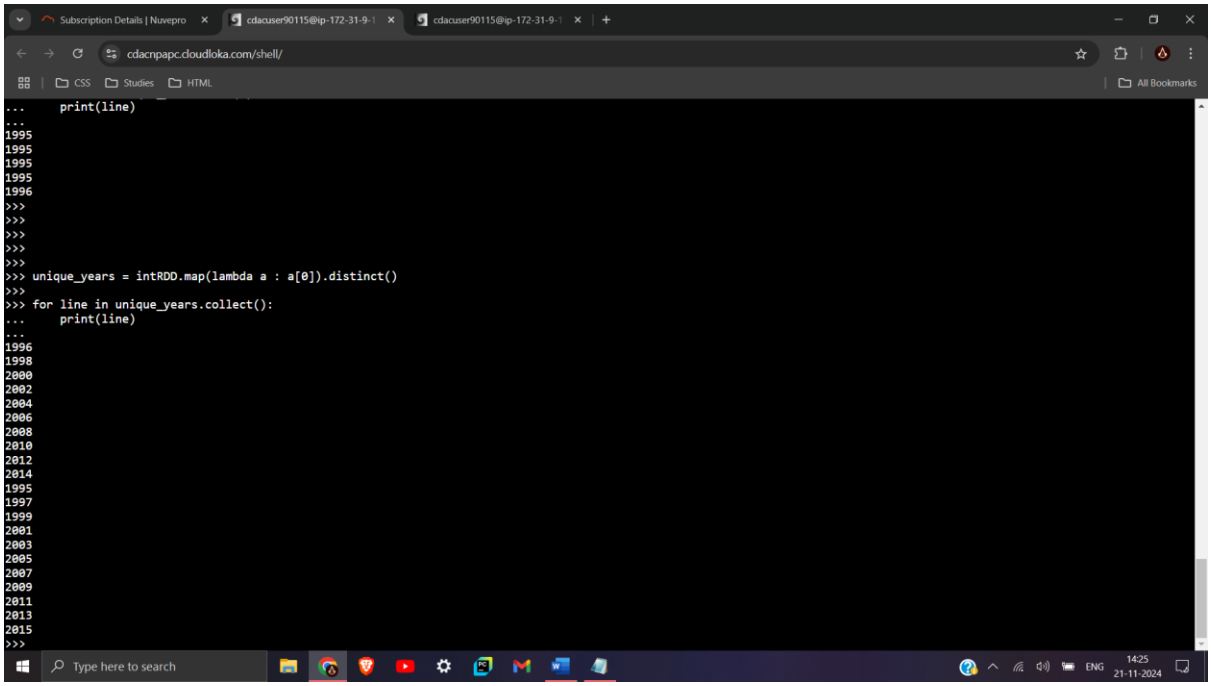
```
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4043. Attempting port 4044.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4044. Attempting port 4045.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4045. Attempting port 4046.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4046. Attempting port 4047.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4047. Attempting port 4048.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4048. Attempting port 4049.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4049. Attempting port 4050.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4050. Attempting port 4051.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4051. Attempting port 4052.
24/11/21 08:40:30 WARN Utils: Service 'SparkUI' could not bind on port 4052. Attempting port 4053.
24/11/21 08:40:30 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.

>>> data = sc.textFile("/user/cdacuser90115/airlines.csv")
>>> header = data.first()
>>> clean = data.filter(lambda a : a != header)
>>> for i in clean.take(5):
...     print(i)
...
1995,1,296.9,46561
1995,2,296.8,37443
1995,3,287.51,34128
1995,4,287.78,30388
1996,1,283.97,47808
>>>
>>> split = clean.map(lambda a : (a.split(",")[0], a.split(",")[1], a.split(",")[2], a.split(",")[3]))
>>> intRDD = split.map(lambda a : (a[0], a[1], float(a[2]), int(a[3])))
>>>
>>> for i in intRDD.take(5):
...     print(i)
...
('1995', '1', 296.9, 46561)
('1995', '2', 296.8, 37443)
('1995', '3', 287.51, 34128)
('1995', '4', 287.78, 30388)
('1996', '1', 283.97, 47808)
>>> count_no_rows = intRDD.filter(lambda a : a[3] > 40000).count()
>>> print(count_no_rows)
38
>>>
```

>>> count_no_rows = intRDD.filter(lambda a : a[3] > 40000).count()

>>> print(count_no_rows)

B]



```
...     print(line)
...
1995
1995
1995
1995
1996
>>>
>>>
>>>
>>>
>>>
>>> unique_years = intRDD.map(lambda a : a[0]).distinct()
>>>
>>> for line in unique_years.collect():
...     print(line)
...
1996
1998
2000
2002
2004
2006
2008
2010
2012
2014
1995
1997
1999
2001
2003
2005
2007
2009
2011
2013
2015
>>>
```

>>> unique_years = intRDD.map(lambda a : a[0]).distinct()

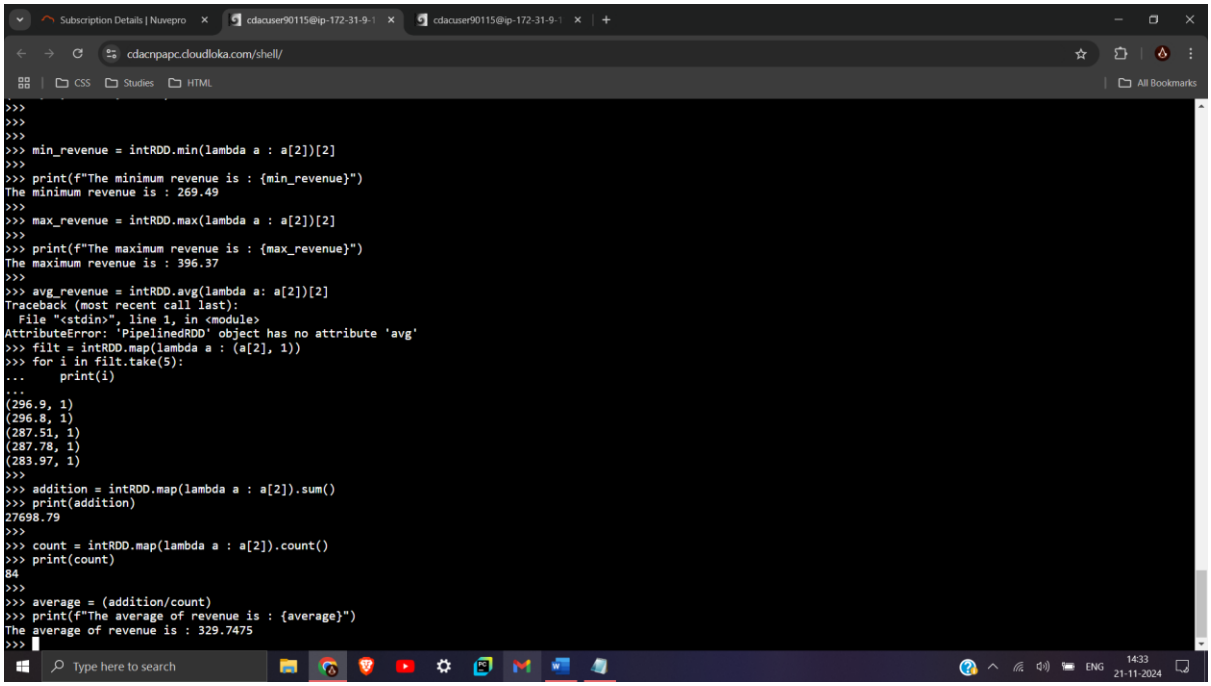
>>>

>>> for line in unique_years.collect():

... print(line)

Q2.

A]



>>> min_revenue = intRDD.min(lambda a : a[2])[2]

>>>

>>> print(f"The minimum revenue is : {min_revenue}")

The minimum revenue is : 269.49

>>>

>>> max_revenue = intRDD.max(lambda a : a[2])[2]

>>>

>>> print(f"The maximum revenue is : {max_revenue}")

The maximum revenue is : 396.37

>>> addition = intRDD.map(lambda a : a[2]).sum()

>>> print(addition)

27698.79

>>>

>>> count = intRDD.map(lambda a : a[2]).count()

>>> print(count)

84

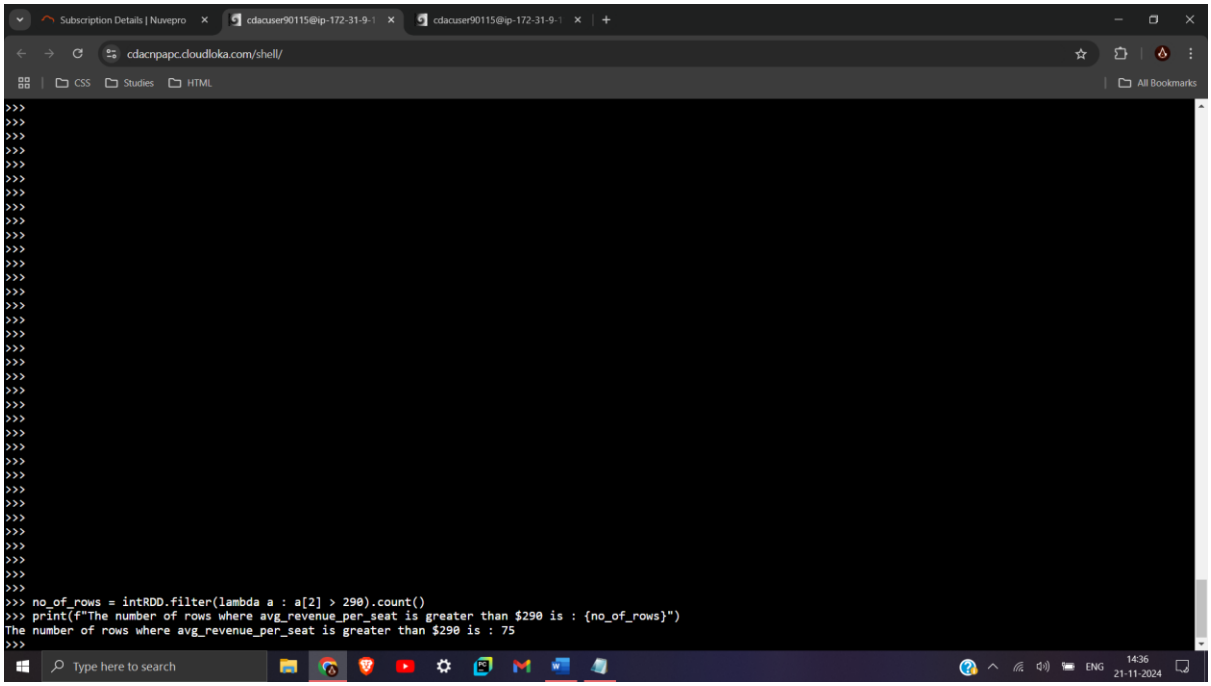
>>>

>>> average = (addition/count)

>>> print(f"The average of revenue is : {average}")

The average of revenue is : 329.7475

B]

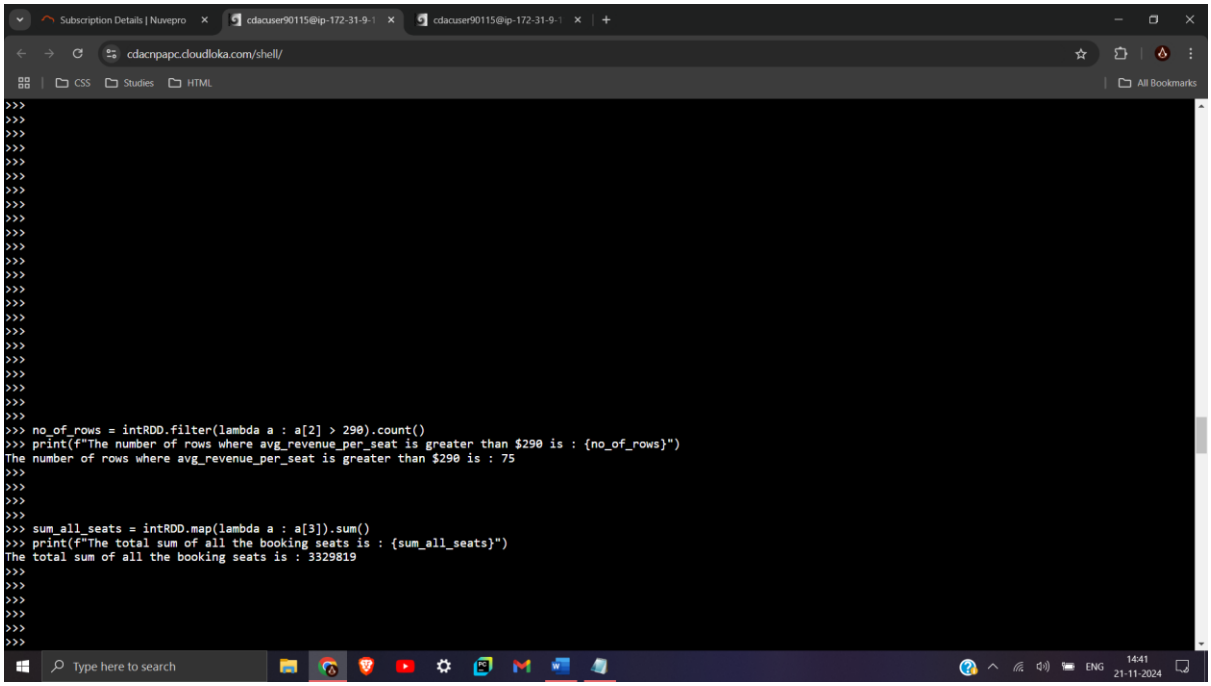


>>> no_of_rows = intRDD.filter(lambda a : a[2] > 290).count()

>>> print(f"The number of rows where avg_revenue_per_seat is greater than $290 is : {no_of_rows}")

The number of rows where avg_revenue_per_seat is greater than $290 is : 75

C]

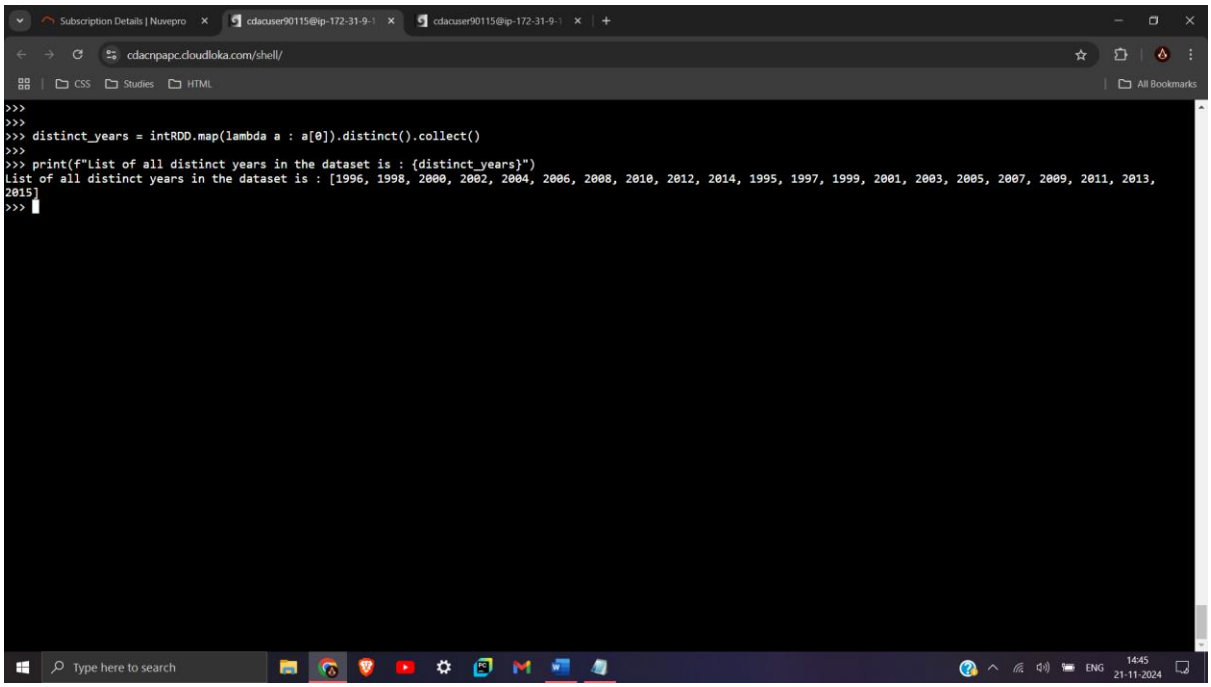


>>> sum_all_seats = intRDD.map(lambda a : a[3]).sum()

>>> print(f"The total sum of all the booking seats is : {sum_all_seats}")

The total sum of all the booking seats is : 3329819

D]



>>> distinct_years = intRDD.map(lambda a : a[0]).distinct().collect()

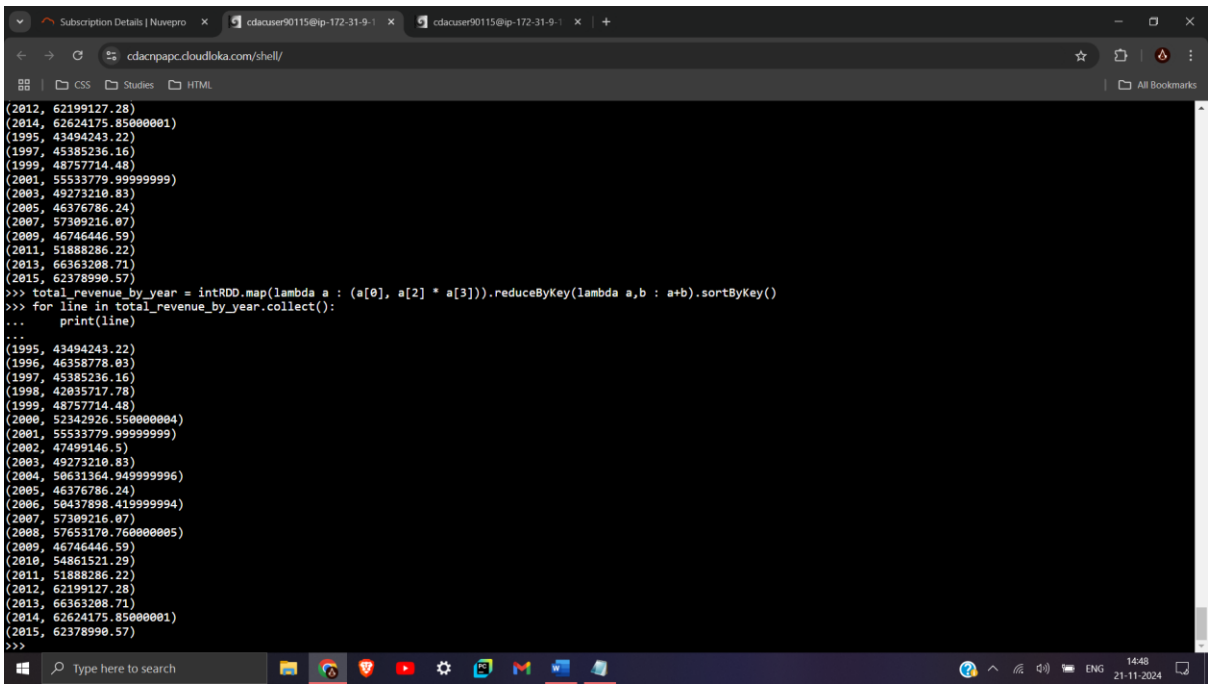
>>>

>>> print(f"List of all distinct years in the dataset is : {distinct_years}")

List of all distinct years in the dataset is : [1996, 1998, 2000, 2002, 2004, 2006, 2008, 2010, 2012, 2014, 1995, 1997, 1999, 2001, 2003, 2005, 2007, 2009, 2011, 2013,

2015]

E]



>>> total_revenue_by_year = intRDD.map(lambda a : (a[0], a[2] * a[3])).reduceByKey(lambda a,b : a+b).sortByKey()

>>> for line in total_revenue_by_year.collect():

... print(line)

...