

Big Data Exam

```
Subscription Details | Nuvepro Web FTP | Hue - File Browser | cdacuser125@ip-172- | cdacuser125@ip-172-3 | Untitled document - C | Big Data | GitHub Dashboard
npapc.cloudloka.com/shell/
Hive Session ID = 7fb68362-0d7f-4aa0-9239-867219f0884d
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive> use cdac_vijay;
OK
Time taken: 0.338 seconds
hive> desc airport_data;
OK
airport_id      int
name            string
city            string
country         string
iata            string
icao            string
latitude        double
longitude        double
altitude        int
timezone        double
dst             string
tz             string
Time taken: 0.13 seconds, Fetched: 12 row(s)
hive> desc airlines_data;
OK
airline_id      int
name            string
alias           string
iata            string
icao            string
callsign        string
country         string
active          string
Time taken: 0.035 seconds, Fetched: 8 row(s)
hive> desc routes_data;
OK
airline_iata    string
airline_id      int
src_airport_iata string
src_airport_id  int
dest_airport_iata string
dest_airport_id int
codeshare       string
stops           int
equipment       string
Time taken: 0.036 seconds, Fetched: 9 row(s)
hive>
```

```
airport_id      int
name            string
city            string
country         string
iata            string
icao            string
latitude        double
longitude        double
altitude        int
timezone        double
dst             string
tz             string
Time taken: 0.13 seconds, Fetched: 12 row(s)
hive> desc airlines_data;
OK
airline_id      int
name            string
alias           string
iata            string
icao            string
callsign        string
country         string
active          string
```

Big Data Exam

Time taken: 0.035 seconds, Fetched: 8 row(s)

```
hive> desc routes_data;
```

OK

```
airline_iata      string
airline_id        int
src_airport_iata  string
src_airport_id    int
dest_airport_iata string
dest_airport_id   int
codeshare         string
stops             int
equipment         string
```

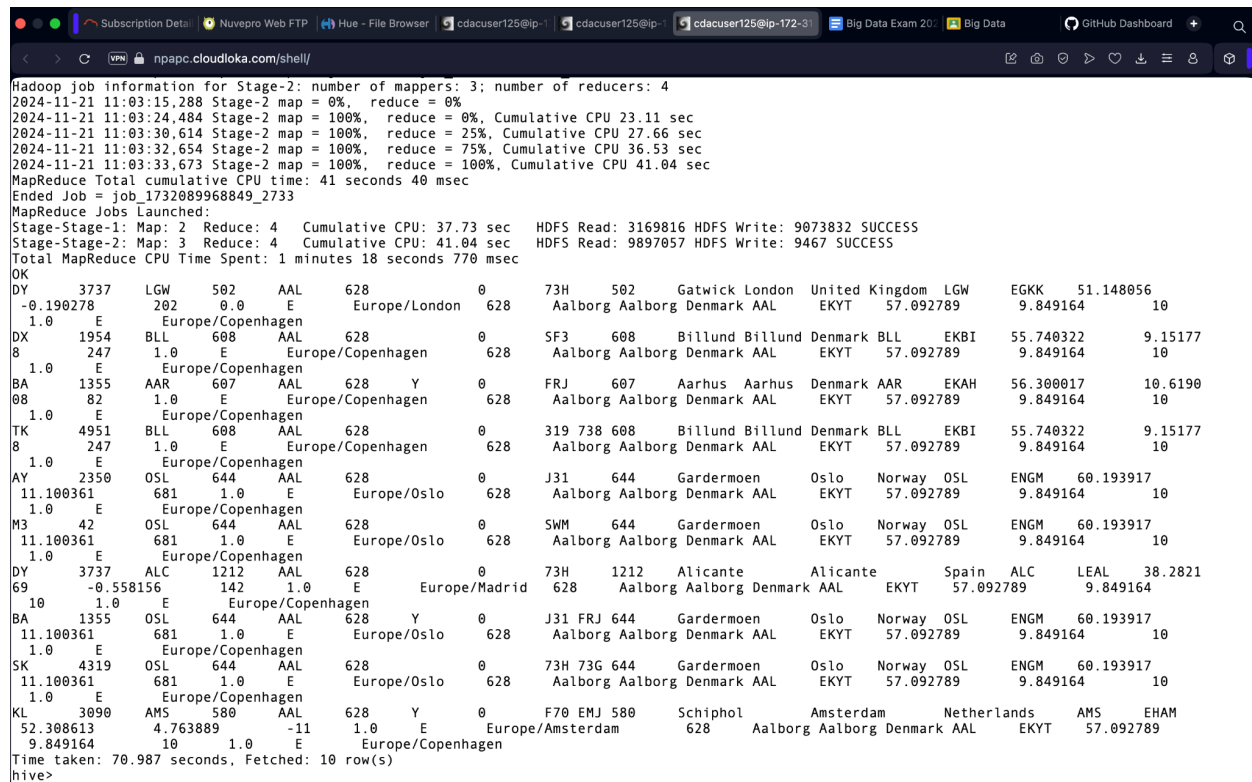
Time taken: 0.036 seconds, Fetched: 9 row(s)

Hive:

Q 1:

1)

```
select * from routes_data r join airport_data src on
src.iata=r.src_airport_iata join airport_data dest on dest.iata
=r.dest_airport_iata
where src.iata=r.src_airport_iata and src.iata !=r.dest_airport_iata
limit 10;
```



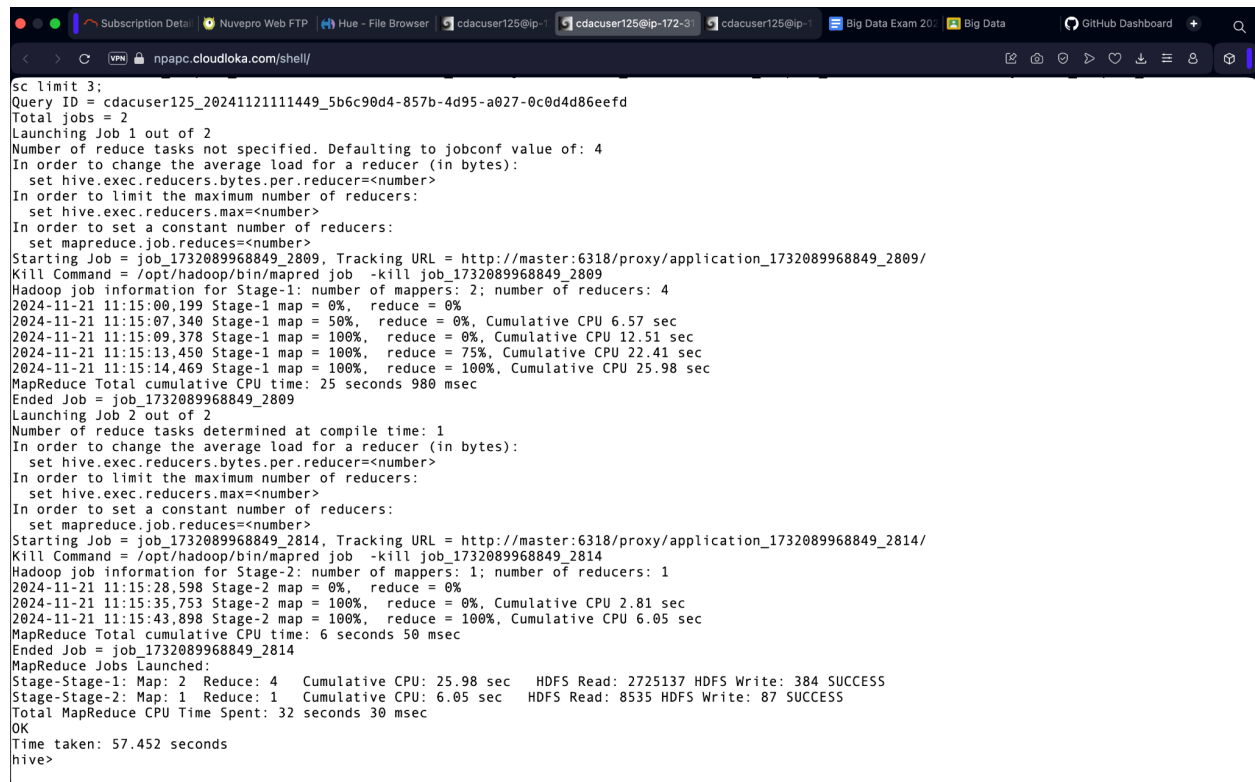
The screenshot shows a terminal window with a dark background. At the top, there are several browser tabs open, including 'Subscription Detail', 'Nuvepro Web FTP', 'Hue - File Browser', and 'cdacuser125@ip-1'. The main content of the terminal is a Hadoop job log and a Hive query result. The Hadoop job log shows the progress of a MapReduce job, including the number of mappers and reducers, the time taken for each stage, and the final status. The Hive query result shows a list of flight routes, including the airline, origin and destination airports, and the flight number. The results are displayed in a table format with multiple columns.

```
Hadoop job information for Stage-2: number of mappers: 3; number of reducers: 4
2024-11-21 11:03:15,288 Stage-2 map = 0%, reduce = 0%
2024-11-21 11:03:24,484 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 23.11 sec
2024-11-21 11:03:30,614 Stage-2 map = 100%, reduce = 25%, Cumulative CPU 27.66 sec
2024-11-21 11:03:32,654 Stage-2 map = 100%, reduce = 75%, Cumulative CPU 36.53 sec
2024-11-21 11:03:33,673 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 41.04 sec
MapReduce Total cumulative CPU time: 41 seconds 40 msec
Ended Job = job_1732089968849_2733
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2 Reduce: 4 Cumulative CPU: 37.73 sec HDFS Read: 3169816 HDFS Write: 9073832 SUCCESS
Stage-Stage-2: Map: 3 Reduce: 4 Cumulative CPU: 41.04 sec HDFS Read: 9897057 HDFS Write: 9467 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 18 seconds 770 msec
OK
DY 3737 LGW 502 AAL 628 0 73H 502 Gatwick London United Kingdom LGW EGKK 51.148056
-0.190278 202 0.0 E Europe/London 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
DX 1954 BLL 608 AAL 628 0 SF3 608 Billund Billund Denmark BLL EKBI 55.740322 9.15177
8 247 1.0 E Europe/Copenhagen 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
BA 1355 AAR 607 AAL 628 Y 0 FRJ 607 Aarhus Aarhus Denmark AAR EKAH 56.300017 10.6190
08 82 1.0 E Europe/Copenhagen 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
TK 4951 BLL 608 AAL 628 0 319 738 608 Billund Billund Denmark BLL EKBI 55.740322 9.15177
8 247 1.0 E Europe/Copenhagen 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
AY 2350 OSL 644 AAL 628 0 J31 644 Gardermoen Oslo Norway OSL ENGM 60.193917
11.100361 681 1.0 E Europe/Oslo 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
M3 42 OSL 644 AAL 628 0 SWM 644 Gardermoen Oslo Norway OSL ENGM 60.193917
11.100361 681 1.0 E Europe/Oslo 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
DY 3737 ALC 1212 AAL 628 0 73H 1212 Alicante Alicante Spain ALC LEAL 38.2821
69 -0.558156 142 1.0 E Europe/Madrid 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164
10 1.0 E Europe/Copenhagen
BA 1355 OSL 644 AAL 628 Y 0 J31 FRJ 644 Gardermoen Oslo Norway OSL ENGM 60.193917
11.100361 681 1.0 E Europe/Oslo 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
SK 4319 OSL 644 AAL 628 0 73H 73G 644 Gardermoen Oslo Norway OSL ENGM 60.193917
11.100361 681 1.0 E Europe/Oslo 628 Aalborg Aalborg Denmark AAL EKYT 57.092789 9.849164 10
1.0 E Europe/Copenhagen
KL 3090 AMS 580 AAL 628 Y 0 F70 EMJ 580 Schiphol Amsterdam Netherlands AMS EHAM
52.308613 4.763889 -11 1.0 E Europe/Amsterdam 628 Aalborg Aalborg Denmark AAL EKYT 57.092789
9.849164 10 1.0 E Europe/Copenhagen
Time taken: 70.987 seconds, Fetched: 10 row(s)
hive>
```

Big Data Exam

2)

```
select r.src_airport_iata , b.name from routes_data r join
airlines_data b on r.src_airport_iata = b.iata order by
r.src_airport_iata desc limit 3;
```



The screenshot shows a terminal window with the following content:

```
sc limit 3:
Query ID = cdacuser125_20241121111449_5b6c90d4-857b-4d95-a027-0c0d4d86eefd
Total jobs = 2
Launching Job 1 out of 2
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_2809, Tracking URL = http://master:6318/proxy/application_1732089968849_2809/
Kill Command = /opt/hadoop/bin/mapred job -kill job_1732089968849_2809
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 4
2024-11-21 11:15:00.199 Stage-1 map = 0%, reduce = 0%
2024-11-21 11:15:07.340 Stage-1 map = 50%, reduce = 0%, Cumulative CPU 6.57 sec
2024-11-21 11:15:09.378 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 12.51 sec
2024-11-21 11:15:13.450 Stage-1 map = 100%, reduce = 75%, Cumulative CPU 22.41 sec
2024-11-21 11:15:14.469 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 25.98 sec
MapReduce Total cumulative CPU time: 25 seconds 980 msec
Ended Job = job_1732089968849_2809
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
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_2814, Tracking URL = http://master:6318/proxy/application_1732089968849_2814/
Kill Command = /opt/hadoop/bin/mapred job -kill job_1732089968849_2814
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2024-11-21 11:15:28.598 Stage-2 map = 0%, reduce = 0%
2024-11-21 11:15:35.753 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 2.81 sec
2024-11-21 11:15:43.898 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 6.05 sec
MapReduce Total cumulative CPU time: 6 seconds 50 msec
Ended Job = job_1732089968849_2814
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2 Reduce: 4 Cumulative CPU: 25.98 sec HDFS Read: 2725137 HDFS Write: 384 SUCCESS
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 6.05 sec HDFS Read: 8535 HDFS Write: 87 SUCCESS
Total MapReduce CPU Time Spent: 32 seconds 30 msec
OK
Time taken: 57.452 seconds
hive>
```

3)

```
select count(equipment) from routes_data group by equipment;
OR
select distinct(equipment) from routes_data ;
```

Big Data Exam

[illegible]

Q2)

1)

Here we are performing static partition on dest_airport_iata column where the iata= KZN

Table Schema:

```
create table routes_partitioned (airline_iata string, airline_id int,
src_airport_iata string , src_airport_id int, dest_airport_id int,
codeshare string, stops int) partitioned by (dest_airport_iata
string) row format delimited fields terminated by ',' stored as
textfile;
```

Overwriting the table

```
insert overwrite table routes_partitioned
partition(dest_airport_iata) select airline_iata , airline_id ,
src_airport_iata , src_airport
_id , dest_airport_id, codeshare , stops , dest_airport_iata from
routes data where dest airport iata = 'KZN';
```

Big Data Exam

The screenshot shows the Hue File Browser interface. On the left, a sidebar lists databases and tables. The main area displays the file browser for the path: `/ user / hive / warehouse / cdac_vijay.db / routes_partitioned / dest_airport_iata=KZN / 000000_0`. The file is a Hive table partitioned by `dest_airport_iata`. The table's metadata is shown on the left, and the table's data is displayed in a grid on the right.

File Name	Last modified	User	Group	Size	Mode
01sept	21/11/2024 17:08	cdacuser125	hive	693 B	100644
1sept2024_anup					
02oct2024					
2sep2024					
2sep_abhi					
2sep_abhi_new					
2sep2024_postoffice					
3octsd					
3sept_2024					
7sep24					
9thsept					
10october					
24aug2024					
27_aug_2024_devang					
27_aug_2024_pavan					
27_aug_2024_prawn					
27_aug_2024_sumedh					
27_aug_chirag					
27aug24					
27aug24_avinash					
27aug24_nn					
27aug2024					
27aug2024_aniket					
27aug2024_barkha					
27aug2024_deadpool					
27aug2024_kuru					
27aug2024_m1					
27aug2024_man					

The table data is displayed in a grid with columns for airline, destination airport, and stops. The data is as follows:

airline	dest_airport_iata	stops
2B, 410, AER, 2965, 2990, , 0		
2B, 410, ASF, 2966, 2990, , 0		
2B, 410, CEK, 2968, 2990, , 0		
2B, 410, DME, 4029, 2990, , 0		
2B, 410, EGO, 6156, 2990, , 0		
2B, 410, LED, 2948, 2990, , 0		
2B, 410, SVX, 2975, 2990, , 0		
8Q, 3788, AYT, 1688, 2990, , 0		
FV, 641, LED, 2948, 2990, , 0		
FZ, 14485, DWC, 8076, 2990, , 0		
G9, 329, SHJ, 2191, 2990, , 0		
HY, 5281, FEG, 6149, 2990, , 0		
HY, 5281, SKD, 2981, 2990, , 0		
HY, 5281, TAS, 2983, 2990, , 0		
KC, 1006, ALA, 2908, 2990, , 0		
S7, 329, DME, 4029, 2990, , 0		
SU, 130, LED, 2948, 2990, , 0		
SU, 130, SVO, 2985, 2990, , 0		
SZ, 10122, DYU, 2979, 2990, , 0		
SZ, 10122, LBD, 6147, 2990, , 0		
TK, 4951, IST, 1701, 2990, , 0		
U6, 5234, DME, 4029, 2990, , 0		
U6, 5234, DYU, 2979, 2990, , 0		
U6, 5234, LBD, 6147, 2990, , 0		
UN, 9067, DME, 4029, 2990, , 0		
UR, 15814, UFA, 2992, 2990, , 0		
YK, 1943, FRU, 2912, 2990, , 0		
YK, 1943, OSS, 2913, 2990, , 0		

Dynamic partitioning

```
SET hive.exec.dynamic.partition=true;
```

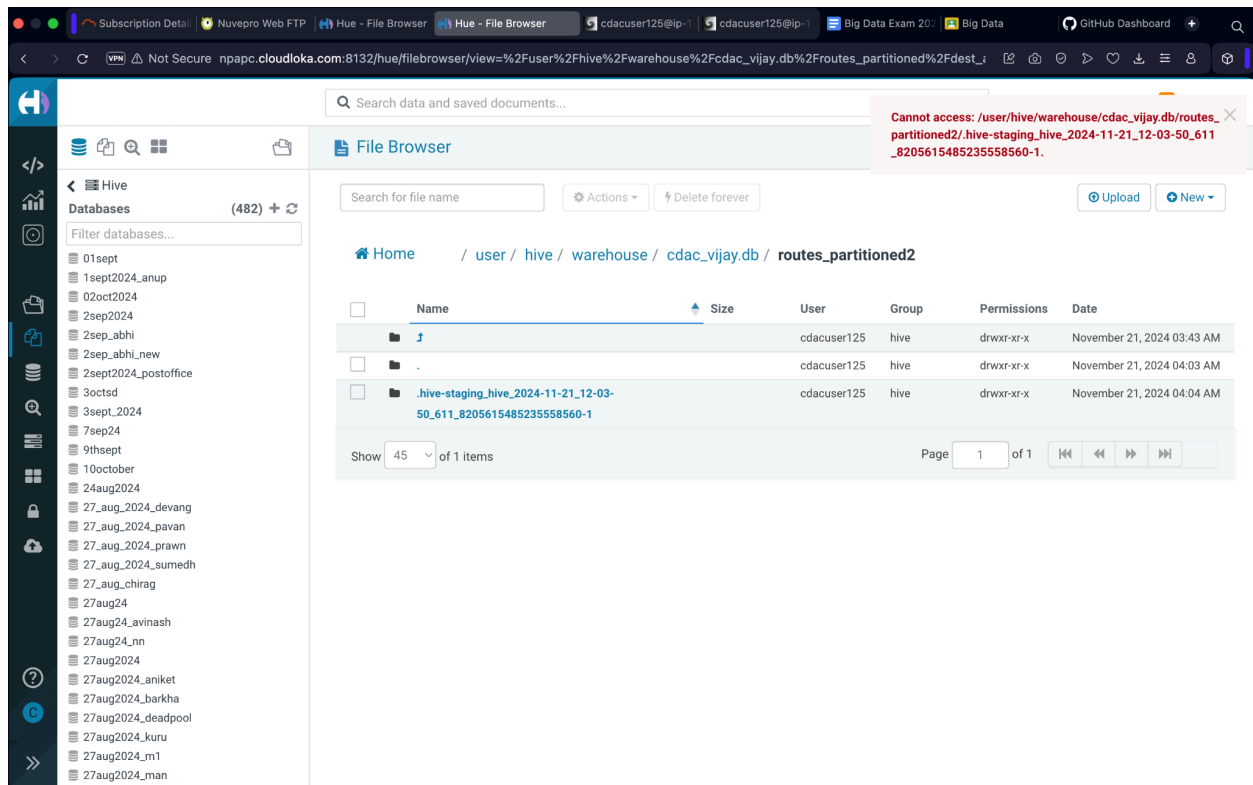
```
SET hive.exec.dynamic.partition.mode=nonstrict;
```

```
create table routes_partitioned2 (airline_iata string, airline_id
int, src_airport_iata string , src_airport_id int, dest_airport_id
int,
codeshare string, stops int) partitioned by (dest_airport_iata
string) row format delimited fields terminated by ',' stored as orc;
```

```
insert overwrite table routes_partitioned2
partition(dest_airport_iata) select airline_iata , airline_id ,
src_airport_iata , src_airport_id , dest_airport_id, codeshare ,
stops , dest_airport_iata from routes_data;
```

Big Data Exam

The shell is not handling the jobs properly partitions cannot be added



2)

Orc format support ddl commands

```
insert into routes_partitioned2 values ("hji",789,"HJK",  
786,"MGN",567,899,"HUJ",0) partition(dest_airport_iata='ORD');
```

3)

```
select * from routes_partitioned2 partition  
(dest_airport_iata='ORD');
```

4)

In hive the num of distinct dest_airport_iata are the number of partitions

```
Select distinct(dest_airport_iata),count(*) from routes_partitioned2;
```

Sir taught us the way in spark sql In spark.sql

```
df = spark.sql("select * form routes_partitioned2")  
df.show()  
df.getNumPartitions()
```

Big Data Exam

Spark:

Q1)

1)

```
df.registerTempTable("airline")
booked_seats = spark.sql("select count(*) from airline where
booked_seats between 20000 and 50000")
booked_seats.show()
```

```
+-----+
|count(1)|
+-----+
|      84|
+-----+
```

Big Data Exam

```
Subscription | Nuvepro Web | Hue - File Br | Hue - File Br | cdacuser125 | cdacuser125 | cdacuser125@ip-172- | Big Data Exam | Big data exam | Submission L | GitHub Dash |
npapc.cloudloka.com/shell/
|1998| 4| 316.18| 35393|
|1999| 1| 331.74| 47453|
|1999| 2| 329.34| 38243|
|1999| 3| 317.22| 33048|
|1999| 4| 317.93| 31256|
+-----+
only showing top 20 rows
>>> df.resisterTempTable("airline")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/opt/spark-3.1.2/python/pyspark/sql/dataframe.py", line 1643, in __getattr__
    raise AttributeError(
AttributeError: 'DataFrame' object has no attribute 'resisterTempTable'
>>> df.registerTempTable("airline")
>>> booked_seats = spark.sql("select count(*) from airlines where booked_seats between 20000 and 50000")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/opt/spark-3.1.2/python/pyspark/sql/session.py", line 723, in sql
    return DataFrame(self._jsparkSession.sql(sqlQuery), self._wrapped)
  File "/opt/spark-3.1.2/python/lib/py4j-0.10.9-src.zip/py4j/java_gateway.py", line 1304, in __call__
  File "/opt/spark-3.1.2/python/pyspark/sql/utils.py", line 117, in deco
    raise converted from None
pyspark.sql.utils.AnalysisException: cannot resolve ''booked_seats'' given input columns: [spark_catalog.default.airlines.active, spark_catalog.default.airlines.airline_id, spark_catalog.default.airlines.alias, spark_catalog.default.airlines.callsign, spark_catalog.default.airlines.country, spark_catalog.default.airlines.iata, spark_catalog.default.airlines.icao, spark_catalog.default.airlines.name]; line 1 pos 36;
'Aggregate [unresolvedalias(count(1), None)]
+- 'Filter (('booked_seats >= 20000) AND ('booked_seats <= 50000))
   +- SubqueryAlias spark_catalog.default.airlines
      +- HiveTableRelation ['default'.airlines', org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, Data Cols: [airline_id#263, name#264, alias#265, iata#266, icao#267, callsign#268, country#269, active#270], Partition Cols: []]

>>> booked_seats = spark.sql("select count(*) from airline where booked_seats between 20000 and 50000")
>>> boood_seats.show()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'boood_seats' is not defined
>>> booked_seats.show()
+-----+
|count(1)|
+-----+
|      84|
+-----+
>>>
```

2)

Q2)

```
df = spark.read.csv('/user/cdacuser125/airlines1.csv' , header=True,
inferSchema=True)
>>> df.count()
84
>>> df.printSchema()
root
|-- Year: integer (nullable = true)
|-- Quarter: integer (nullable = true)
|-- Avg_rev_per_seat: double (nullable = true)
|-- booked_seats: integer (nullable = true)

>>> df.show()
+-----+-----+-----+-----+-----+-----+
|Year|Quarter|Avg_rev_per_seat|booked_seats|
```


Big Data Exam

Year	Quarter	Avg_rev_per_seat	booked_seats
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
1996	2	275.78	43020
1996	3	269.49	38952
1996	4	278.33	37443
1997	1	283.4	35067
1997	2	289.44	46565
1997	3	282.27	38886
1997	4	293.51	37454
1998	1	304.74	31315
1998	2	300.97	30852
1998	3	315.25	38118
1998	4	316.18	35393
1999	1	331.74	47453
1999	2	329.34	38243
1999	3	317.22	33048
1999	4	317.93	31256

only showing top 20 rows

1)

```
from pyspark.sql.functions import min,max,avg
avg_seats = df.agg(avg("booked_seats").alias("avg"))
>>> avg_seats.show()
+-----+
|          avg|
+-----+
|39640.70238095238|
+-----+
```

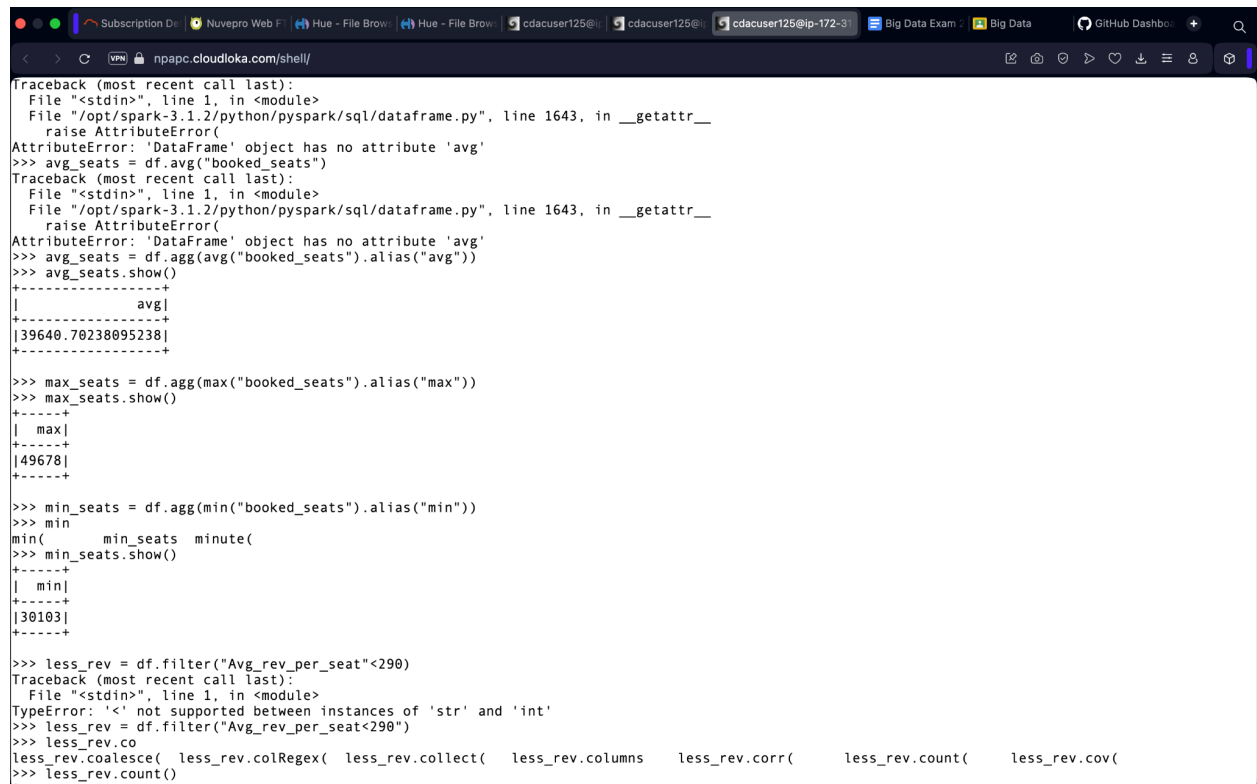
```
>>>
max_seats = df.agg(max("booked_seats").alias("max"))
>>> max_seats.show()
+-----+
|    max|
+-----+
|49678|
+-----+
```

Big Data Exam

```
>>>
min_seats = df.agg(min("booked_seats").alias("min"))
```

```
>>> min_seats.show()
```

```
+-----+
|  min|
+-----+
|30103|
+-----+
```



```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/opt/spark-3.1.2/python/pyspark/sql/dataframe.py", line 1643, in __getattr__
    raise AttributeError(
AttributeError: 'DataFrame' object has no attribute 'avg'
>>> avg_seats = df.avg("booked_seats")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/opt/spark-3.1.2/python/pyspark/sql/dataframe.py", line 1643, in __getattr__
    raise AttributeError(
AttributeError: 'DataFrame' object has no attribute 'avg'
>>> avg_seats = df.agg(avg("booked_seats").alias("avg"))
>>> avg_seats.show()
+-----+
|      avg|
+-----+
|39640.70238095238|
+-----+

>>> max_seats = df.agg(max("booked_seats").alias("max"))
>>> max_seats.show()
+-----+
|    max|
+-----+
|49678|
+-----+

>>> min_seats = df.agg(min("booked_seats").alias("min"))
>>> min_seats.show()
+-----+
|    min|
+-----+
|30103|
+-----+

>>> less_rev = df.filter("Avg_rev_per_seat<290")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: '<' not supported between instances of 'str' and 'int'
>>> less_rev = df.filter("Avg_rev_per_seat<290")
>>> less_rev.coalesce(less_rev.colRegexp(less_rev.collect(less_rev.columns, less_rev.count(less_rev.cov(less_rev.count()
```

2)

```
less_rev = df.filter("Avg_rev_per_seat<290")
```

```
>>> less_rev.count()
```

```
9
```

```
>>> less_rev.show()
```

```
+-----+-----+-----+-----+
|Year|Quarter|Avg_rev_per_seat|booked_seats|
+-----+-----+-----+-----+
|1995|      3|      287.51|      34128|
|1995|      4|      287.78|      30388|
|1996|      1|      283.97|      47808|
|1996|      2|      275.78|      43020|
|1996|      3|      269.49|      38952|
```

Big Data Exam

```
|1996|      4|      278.33|      37443|
|1997|      1|      283.4|      35067|
|1997|      2|      289.44|      46565|
|1997|      3|      282.27|      38886|
+-----+-----+-----+-----+
```

```
>>> max_seats.show()
+-----+
|_max|
+-----+
|49678|
+-----+

>>> min_seats = df.agg(min("booked_seats").alias("min"))
>>> min
min(      min_seats  minute(
>>> min_seats.show()
+-----+
|_min|
+-----+
|30103|
+-----+

>>> less_rev = df.filter("Avg_rev_per_seat"<290)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: '<' not supported between instances of 'str' and 'int'
>>> less_rev = df.filter("Avg_rev_per_seat"<290)
>>> less_rev.co
less_rev.coalesce( less_rev.colRegex( less_rev.collect( less_rev.columns less_rev.corr( less_rev.count( less_rev.cov(
>>> less_rev.count()
9
>>> less_rev.show()
+-----+-----+-----+-----+
|Year|Quarter|Avg_rev_per_seat|booked_seats|
+-----+-----+-----+-----+
|1995|      3|      287.51|      34128|
|1995|      4|      287.78|      30388|
|1996|      1|      283.97|      47808|
|1996|      2|      275.78|      43020|
|1996|      3|      269.49|      38952|
|1996|      4|      278.33|      37443|
|1997|      1|      283.4|      35067|
|1997|      2|      289.44|      46565|
|1997|      3|      282.27|      38886|
+-----+-----+-----+-----+

>>> book_seats_by_qua = df.groupBy("Quarter").agg(avg("booked_seats").alias("avg"))
>>> book_seats_by_qua.show()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
```

```
>>>
3)
abook_seats_by_qua =
df.groupBy("Quarter").agg(avg("booked_seats").alias("avg"))
>>> abook_seats_by_qua.show()
+-----+-----+-----+
|Quarter|      avg|
+-----+-----+
|      1|41607.666666666664|
|      3| 39386.23809523809|
|      4| 39111.95238095238|
|      2| 38456.95238095238|
+-----+-----+-----+
```

Big Data Exam

```
npapc.cloudloka.com/shell/

+-----+
|Year|Quarter|Avg_rev_per_seat|booked_seats|
+-----+
|1995|3|287.51|34128|
|1995|4|287.78|30388|
|1996|1|283.97|47808|
|1996|2|275.78|43020|
|1996|3|269.49|38952|
|1996|4|278.33|37443|
|1997|1|283.4|35067|
|1997|2|289.44|46565|
|1997|3|282.27|38886|
+-----+

>>> book_seats_by_qua = df.groupby("Quarter").agg(avg("booked_seats").alias("avg"))
>>> book_seats_by_qua.show()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'book_seats_by_qua' is not defined
>>> abook_seats_by_qua = df.groupby("Quarter").agg(avg("booked_seats").alias("avg"))
>>> abook_seats_by_qua.show()
+-----+
|Quarter|avg|
+-----+
|1|41607.666666666664|
|3|39386.23809523809|
|4|39111.95238095238|
|2|38456.95238095238|
+-----+

= d
>>>
>>> dist_years = df.groupby("Year").count()
>>> dist_years.show()
+-----+
|Year|count|
+-----+
|2003|4|
|2007|4|
|2015|4|
|2006|4|
|2013|4|
|1997|4|
|2014|4|
|2004|4|
+-----+
```

```
4)
>>> dist_years = df.groupby("Year").count()
>>> dist_years.show()
+-----+
|Year|count|
+-----+
|2003|4|
|2007|4|
|2015|4|
|2006|4|
|2013|4|
|1997|4|
|2014|4|
|2004|4|
|1996|4|
|1998|4|
|2012|4|
|2009|4|
|1995|4|
|2001|4|
|2005|4|
|2000|4|
```

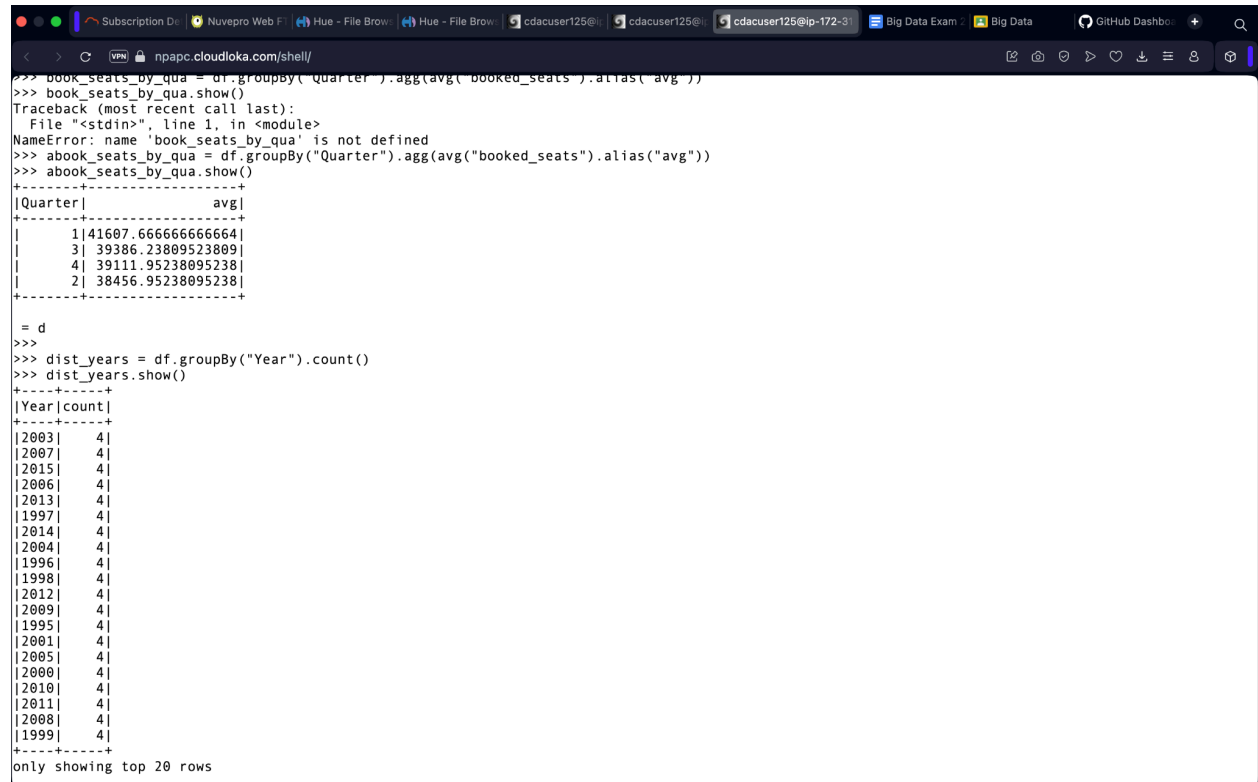
Big Data Exam

```
|2010|    4|
|2011|    4|
|2008|    4|
|1999|    4|
```

```
+-----+-----+
```

only showing top 20 rows

>>>



```
>>> book_seats_by_qua = df.groupby("Quarter").agg(avg("booked_seats").alias("avg"))
>>> book_seats_by_qua.show()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'book_seats_by_qua' is not defined
>>> abook_seats_by_qua = df.groupby("Quarter").agg(avg("booked_seats").alias("avg"))
>>> abook_seats_by_qua.show()
+-----+-----+
|Quarter|      avg|
+-----+-----+
|1|41607.666666666664|
|3| 39386.23809523809|
|4| 39111.95238095238|
|2| 38456.95238095238|
+-----+-----+

= d
>>>
>>> dist_years = df.groupby("Year").count()
>>> dist_years.show()
+-----+-----+
|Year|count|
+-----+-----+
|2003|    4|
|2007|    4|
|2015|    4|
|2006|    4|
|2013|    4|
|1997|    4|
|2014|    4|
|2004|    4|
|1996|    4|
|1998|    4|
|2012|    4|
|2009|    4|
|1995|    4|
|2001|    4|
|2005|    4|
|2000|    4|
|2010|    4|
|2011|    4|
|2008|    4|
|1999|    4|
+-----+-----+
only showing top 20 rows
```

5)

```
>>> high_rev =
df.withColumn("total_revenue", col("Avg_rev_per_seat") * col("booked_sea
ts")).groupBy("Quarter").agg(max("total_revenue")).orderBy(
"Quarter")
>>> high_rev.show()
```

```
+-----+-----+
|Quarter| max(total_revenue)|
+-----+-----+
|1|1.8572613990000002E7|
|2|      1.731616761E7|
|3|      1.81778142E7|
|4|      1.881940848E7|
+-----+-----+
```

Big Data Exam

>>>

```
Subscription Di Nuvepro Web F Hue - File Brow Hue - File Brow cdacuser125@i cdacuser125@i cdacuser125@ip-172-31 Big Data Exam Big Data GitHub Dashbo
npapc.cloudloka.com/shell/

|2015| 4|
|2006| 4|
|2013| 4|
|1997| 4|
|2014| 4|
|2004| 4|
|1996| 4|
|1998| 4|
|2012| 4|
|2009| 4|
|1995| 4|
|2001| 4|
|2005| 4|
|2000| 4|
|2010| 4|
|2011| 4|
|2008| 4|
|1999| 4|
+-----+
only showing top 20 rows

>>> high_rev = df.withColumn("total_revenue",col("Avg_rev_per_seat")*col("booked_seats")).groupBy("Quarter").agg(max("total_revenue"))
>>> high_rev.show()
+-----+
|Quarter| max(total_revenue)|
+-----+
|1|1.8572613990000002E7|
|3| 1.81778142E7|
|4| 1.881940848E7|
|2| 1.731616761E7|
+-----+

>>> high_rev = df.withColumn("total_revenue",col("Avg_rev_per_seat")*col("booked_seats")).groupBy("Quarter").agg(max("total_revenue")).orderBy(
"Quarter")
>>> high_rev.show()
+-----+
|Quarter| max(total_revenue)|
+-----+
|1|1.8572613990000002E7|
|2| 1.731616761E7|
|3| 1.81778142E7|
|4| 1.881940848E7|
+-----+

>>> █
```