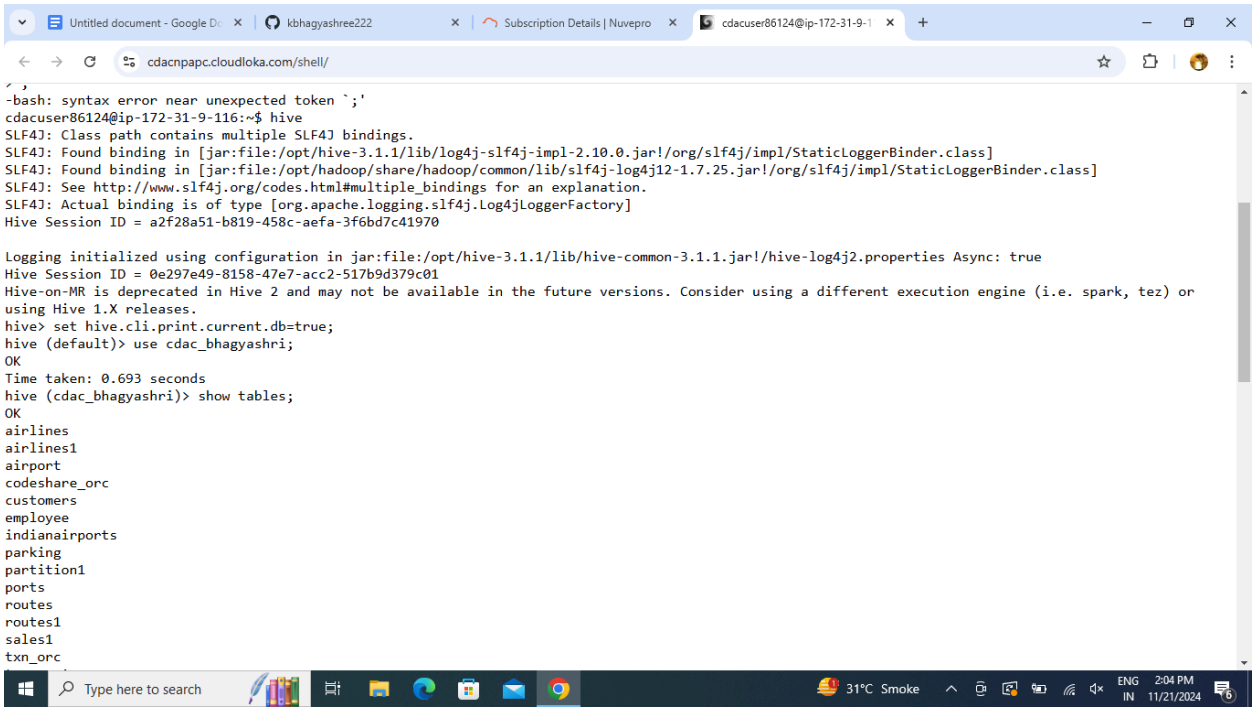
Hive
set hive.cli.print.current.db=true;
hive (default)> use cdac_bhagyashri;



Q1. find airport that are listed as both a source and destination in the routes table.

select a1.name from airport a1 join routes r on a1.airport_id=r.src_airport_id join airport a2 on a2.airport_id=r.dest_airpor
t_id where a1.airport_id=a2.airport_id;

Untitled document - Google D | kbhagyashree222 | Subscription Details | Nuvepro | cdacuser86124@ip-172-31-9-1

cdacnpapc.cloudloka.com/shell/

```
Time taken: 61.685 seconds
hive (cdac_bhagyashri)> select a1.name from airport a1 join routes r on a1.airport_id=r.src_airport_id join airport a2 on a2.airport_id=r.dest_airpor
t_id where a1.airport_id=a2.airport_id;
No Stats for cdac_bhagyashri@airport, Columns: airport_id, name
No Stats for cdac_bhagyashri@routes, Columns: dest_airport_id, src_airport_id
No Stats for cdac_bhagyashri@airport, Columns: airport_id
Query ID = cdacuser86124_20241121084129_720f2ce0-a23f-4d32-ab8d-fe30ee24b051
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_2254, Tracking URL = http://master:6318/proxy/application_1732089968849_2254/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2254
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 4
2024-11-21 08:41:41,327 Stage-1 map = 0%,  reduce = 0%
2024-11-21 08:41:47,499 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 6.47 sec
2024-11-21 08:41:49,547 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 13.03 sec
2024-11-21 08:41:53,645 Stage-1 map = 100%,  reduce = 25%, Cumulative CPU 15.69 sec
2024-11-21 08:41:54,671 Stage-1 map = 100%,  reduce = 50%, Cumulative CPU 18.13 sec
2024-11-21 08:41:55,698 Stage-1 map = 100%,  reduce = 75%, Cumulative CPU 21.74 sec
2024-11-21 08:41:56,723 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 24.37 sec
MapReduce Total cumulative CPU time: 24 seconds 370 msec
Ended Job = job_1732089968849_2254
Launching Job 2 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>
```

Type here to search | 31°C Smoke | ENG IN | 2:13 PM 11/21/2024

Untitled document - Google D | kbhagyashree222 | Subscription Details | Nuvepro | cdacuser86124@ip-172-31-9-1

cdacnpapc.cloudloka.com/shell/

```
2024-11-21 08:41:49,547 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 13.03 sec
2024-11-21 08:41:53,645 Stage-1 map = 100%,  reduce = 25%, Cumulative CPU 15.69 sec
2024-11-21 08:41:54,671 Stage-1 map = 100%,  reduce = 50%, Cumulative CPU 18.13 sec
2024-11-21 08:41:55,698 Stage-1 map = 100%,  reduce = 75%, Cumulative CPU 21.74 sec
2024-11-21 08:41:56,723 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 24.37 sec
MapReduce Total cumulative CPU time: 24 seconds 370 msec
Ended Job = job_1732089968849_2254
Launching Job 2 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_2256, Tracking URL = http://master:6318/proxy/application_1732089968849_2256/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2256
Hadoop job information for Stage-2: number of mappers: 2; number of reducers: 4
2024-11-21 08:42:09,839 Stage-2 map = 0%,  reduce = 0%
2024-11-21 08:42:17,026 Stage-2 map = 50%,  reduce = 0%, Cumulative CPU 6.31 sec
2024-11-21 08:42:18,057 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 11.38 sec
2024-11-21 08:42:24,201 Stage-2 map = 100%,  reduce = 50%, Cumulative CPU 17.46 sec
2024-11-21 08:42:25,224 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 25.14 sec
MapReduce Total cumulative CPU time: 25 seconds 140 msec
Ended Job = job_1732089968849_2256
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 4   Cumulative CPU: 24.37 sec   HDFS Read: 3151869 HDFS Write: 416 SUCCESS
Stage-Stage-2: Map: 2  Reduce: 4   Cumulative CPU: 25.14 sec   HDFS Read: 773632 HDFS Write: 369 SUCCESS
Total MapReduce CPU Time Spent: 49 seconds 510 msec
OK
Iskandar
Time taken: 59.502 seconds, Fetched: 1 row(s)
hive (cdac_bhagyashri)>
```

Type here to search | 31°C Smoke | ENG IN | 2:12 PM 11/21/2024

Q2 determine the aircraft type(equipment) that is used on the highest number of routes.

```
select r.equipment , count(r.equipment) as ecount from routes  r
group by r.equipment order by ecount desc limit 1;
```

```
321     1146
E90     1049
Time taken: 57.156 seconds, Fetched: 10 row(s)
hive (cdac_bhagyashri)> select r.equipment , count(r.equipment) as ecount from routes  r group by r.equipment order by ecount desc limit 1;
Query ID = cdacuser86124_20241121085759_fb2661c9-6815-4d61-bee1-07ee79b86b93
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_2351, Tracking URL = http://master:6318/proxy/application_1732089968849_2351/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2351
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 4
2024-11-21 08:58:09,495 Stage-1 map = 0%,  reduce = 0%
2024-11-21 08:58:17,684 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.67 sec
2024-11-21 08:58:23,829 Stage-1 map = 100%,  reduce = 50%, Cumulative CPU 12.3 sec
2024-11-21 08:58:24,855 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 19.5 sec
MapReduce Total cumulative CPU time: 19 seconds 500 msec
Ended Job = job_1732089968849_2351
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_2357, Tracking URL = http://master:6318/proxy/application_1732089968849_2357/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2357
Hadoop job information for Stage-2: number of mappers: 2; number of reducers: 1
```


```
Starting Job = job_1732089968849_2351, Tracking URL = http://master:6318/proxy/application_1732089968849_2351/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2351
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 4
2024-11-21 08:58:09,495 Stage-1 map = 0%,  reduce = 0%
2024-11-21 08:58:17,684 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.67 sec
2024-11-21 08:58:23,829 Stage-1 map = 100%,  reduce = 50%, Cumulative CPU 12.3 sec
2024-11-21 08:58:24,855 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 19.5 sec
MapReduce Total cumulative CPU time: 19 seconds 500 msec
Ended Job = job_1732089968849_2351
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_2357, Tracking URL = http://master:6318/proxy/application_1732089968849_2357/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2357
Hadoop job information for Stage-2: number of mappers: 2; number of reducers: 1
```


```
2024-11-21 08:58:38,872 Stage-2 map = 0%,  reduce = 0%
2024-11-21 08:58:46,110 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 6.23 sec
2024-11-21 08:58:52,256 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 9.55 sec
MapReduce Total cumulative CPU time: 9 seconds 550 msec
Ended Job = job_1732089968849_2357
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 4   Cumulative CPU: 19.5 sec   HDFS Read: 2408617 HDFS Write: 122522 SUCCESS
Stage-Stage-2: Map: 2  Reduce: 1   Cumulative CPU: 9.55 sec   HDFS Read: 134292 HDFS Write: 108 SUCCESS
Total MapReduce CPU Time Spent: 29 seconds 50 msec
OK
320     9180
Time taken: 57.101 seconds, Fetched: 1 row(s)
hive (cdac_bhagyashri)>
```

Q3 find the airline that operates the highest number of routes and the count of those routes.

```
select a.name, count(r.src_airport_id) as countr from airlines a join routes r on a.airline_id = r.airline_id join airport a1 on a1.airport_id=r.src_airport_id group by a.name order by countr desc limit 1;
```

```
City Connexion Airlines 332
Royal Air Maroc 239
Time taken: 113.57 seconds, Fetched: 10 row(s)
hive (cdac_bhagyashri)> select a.name, count(r.src_airport_id) as countr from airlines a join routes r on a.airline_id = r.airline_id join airport a1
 on a1.airport_id=r.src_airport_id group by a.name order by countr desc limit 1;
No Stats for cdac_bhagyashri@airlines, Columns: name, airline_id
No Stats for cdac_bhagyashri@routes, Columns: airline_id, src_airport_id
No Stats for cdac_bhagyashri@airport, Columns: airport_id
Query ID = cdacuser86124_20241121090845_4f3f2ad3-46b4-4398-a1fc-c4087d70e635
Total jobs = 4
Launching Job 1 out of 4
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_2409, Tracking URL = http://master:6318/proxy/application_1732089968849_2409/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2409
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 4
2024-11-21 09:08:56,550 Stage-1 map = 0%,  reduce = 0%
2024-11-21 09:09:03,729 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 6.41 sec
2024-11-21 09:09:04,755 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 13.86 sec
2024-11-21 09:09:09,880 Stage-1 map = 100%,  reduce = 25%, Cumulative CPU 17.87 sec
2024-11-21 09:09:10,910 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 29.53 sec
MapReduce Total cumulative CPU time: 29 seconds 530 msec
Ended Job = job_1732089968849_2409
Launching Job 2 out of 4
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>
```

```
2024-11-21 09:10:03,668 Stage-3 map = 50%,  reduce = 0%, Cumulative CPU 2.69 sec
2024-11-21 09:10:04,695 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 5.37 sec
2024-11-21 09:10:09,818 Stage-3 map = 100%,  reduce = 50%, Cumulative CPU 10.79 sec
2024-11-21 09:10:12,892 Stage-3 map = 100%,  reduce = 100%, Cumulative CPU 16.14 sec
MapReduce Total cumulative CPU time: 16 seconds 140 msec
Ended Job = job_1732089968849_2415
Launching Job 4 out of 4
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_2419, Tracking URL = http://master:6318/proxy/application_1732089968849_2419/
Kill Command = /opt/hadoop/bin/mapred job  -kill job_1732089968849_2419
Hadoop job information for Stage-4: number of mappers: 2; number of reducers: 1
2024-11-21 09:10:28,529 Stage-4 map = 0%,  reduce = 0%
2024-11-21 09:10:34,676 Stage-4 map = 50%,  reduce = 0%, Cumulative CPU 2.58 sec
2024-11-21 09:10:35,702 Stage-4 map = 100%,  reduce = 0%, Cumulative CPU 5.25 sec
2024-11-21 09:10:40,821 Stage-4 map = 100%,  reduce = 100%, Cumulative CPU 8.59 sec
MapReduce Total cumulative CPU time: 8 seconds 590 msec
Ended Job = job_1732089968849_2419
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 4   Cumulative CPU: 29.53 sec   HDFS Read: 3149905 HDFS Write: 1522672 SUCCESS
Stage-Stage-2: Map: 4  Reduce: 4   Cumulative CPU: 42.69 sec   HDFS Read: 1879121 HDFS Write: 4062 SUCCESS
Stage-Stage-3: Map: 2  Reduce: 4   Cumulative CPU: 16.14 sec   HDFS Read: 24446 HDFS Write: 4062 SUCCESS
Stage-Stage-4: Map: 2  Reduce: 1   Cumulative CPU: 8.59 sec   HDFS Read: 15873 HDFS Write: 114 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 36 seconds 950 msec
OK
Air China       1258
Time taken: 117.462 seconds, Fetched: 1 row(s)
hive (cdac_bhagyashri)>
```

Q2}
Q1.

```
set hive.exec.dynamic.partition.mode=unstrict;
set hive.exec.dynamic.partition=true;
set hive.exec.bucketing=true;

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

insert into routepart partitioned(src_airport_id) select * from
routes describe src_airport_id;
```

Q 2.

RDD:

```
dataRDD=sc.textFile("/user/cdacuser86124/airlines.csv")
>>> dataRDD.count()

header=dataRDD.first()
>>> newRDD=dataRDD.filter(lambda line: line!=header)
>>> for line in newRDD.take(5):
...     print(line)

split=newRDD.map(lambda a:
(a.split(',')[0],a.split(',')[1],a.split(',')[2],a.split(',')[3]))
>>> for line in split.take(5):
...     print(line)


seatRDD=split.map(lambda x: (int(x[3])>40000))
```

1] count the number of rows where the total no. of booked seats exceeds 40000

```
seat=split.filter(lambda x: (int(x[3])>40000))
>>> seat.take(5)
[('1995', '1', '296.9', '46561'), ('1996', '1', '283.97', '47808'),
('1996', '2', '275.78', '43020'), ('1997', '2', '289.44', '46565'),
('1999', '1',
'331.74', '47453')]
>>> seat.count()
38
```

```
    / __/__  ___ _____/ /__
   _\ \/ _ \/ _ `/ __/  '_/
  /__ / .__/\_,_/_/ /_/\_\   version 3.1.2
     /_/

Using Python version 3.9.13 (main, Aug 25 2022 23:26:10)
Spark context Web UI available at http://ip-172-31-9-116.ap-south-1.compute.internal:4046
Spark context available as 'sc' (master = yarn, app id = application_1732089968849_2487).
SparkSession available as 'spark'.
>>>
>>> for line in seatRDD.take(5):
...     print(line)
...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'seatRDD' is not defined
>>> dataRDD=sc.textFile("/user/cdacuser86124/airlines.csv")
>>> header=dataRDD.first()
>>> newRDD=dataRDD.filter(lambda line: line!=header)
>>> split=newRDD.map(lambda a: (a.split(',')[0],a.split(',')[1],a.split(',')[2],a.split(',')[3]))
>>> split.take(5)
[('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')]
>>> seatRDD=split.map(lambda x: (int(x[3])>40000))
>>> seatRDD.take(5)
[True, False, False, False, True]
>>> seat=split.filter(lambda x: (int(x[3])>40000))
>>> seat.take(5)
[('1995', '1', '296.9', '46561'), ('1996', '1', '283.97', '47808'), ('1996', '2', '275.78', '43020'), ('1997', '2', '289.44', '46565'), ('1999', '1',
 '331.74', '47453')]
>>> seat.count()
38
>>>
```

```
2] find
>>> year=split.map(lambda a: (a[0],1))
>>> rdd1=year.reduceByKey(lambda a,b: a+b)
>>> rdd1.take(10)
[('1995', 4), ('2002', 4), ('2003', 4), ('2004', 4), ('2007', 4),
('2010', 4), ('2011', 4), ('2012', 4), ('2013', 4), ('2014', 4)]
>>> rdd1.count()
21
>>> rdd2=rdd1.map(lambda y: y[0])
>>> rdd2.take(6)
['1995', '2002', '2003', '2004', '2007', '2010']
```

```
2012
2012
2013
2013
2013
2013
2014
2014
2014
2014
2015
2015
2015
2015
>>> year=split.map(lambda a: (a,1))
>>> rdd1=year.reduceByKey(lambda a,b: a+b)
>>> rdd1.take(10)
[(('1995', '3', '287.51', '34128'), 1), (('1995', '4', '287.78', '30388'), 1), (('1996', '1', '283.97', '47808'), 1), (('1996', '4', '278.33', '37443
'), 1), (('1997', '3', '282.27', '38886'), 1), (('1997', '4', '293.51', '37454'), 1), (('1998', '3', '315.25', '38118'), 1), (('1998', '4', '316.18',
 '35393'), 1), (('1999', '1', '331.74', '47453'), 1), (('1999', '3', '317.22', '33048'), 1)]
>>> year=split.map(lambda a: (a[0],1))
>>> rdd1=year.reduceByKey(lambda a,b: a+b)
>>> rdd1.take(10)
[('1995', 4), ('2002', 4), ('2003', 4), ('2004', 4), ('2007', 4), ('2010', 4), ('2011', 4), ('2012', 4), ('2013', 4), ('2014', 4)]
>>> rdd1.count()
21
>>> rdd2=rdd1.map(lambda y: y[0])
>>> rdd2.take(6)
['1995', '2002', '2003', '2004', '2007', '2010']
>>> rdd2.take(21)
['1995', '2002', '2003', '2004', '2007', '2010', '2011', '2012', '2013', '2014', '2015', '1996', '1997', '1998', '1999', '2000', '2001', '2005', '200
6', '2008', '2009']
>>>
```

Q2]

1. Min , max, avg of avg_rev per seat

```
avgRDD=split.map(lambda a: (a[3]).mean())
```

2.

```
RDD2=split.filter(lambda x: (float(x[2])>290.0))
```

```
RDD2=split.filter(lambda x: (float(x[2])>290.0))
>>> RDD2.take(5)
[('1995', '1', '296.9', '46561'), ('1995', '2', '296.8', '37443'),
('1997', '4', '293.51', '37454'), ('1998', '1', '304.74', '31315'),
('1998', '2',
'300.97', '30852')]
>>> RDD2.count()
75
```

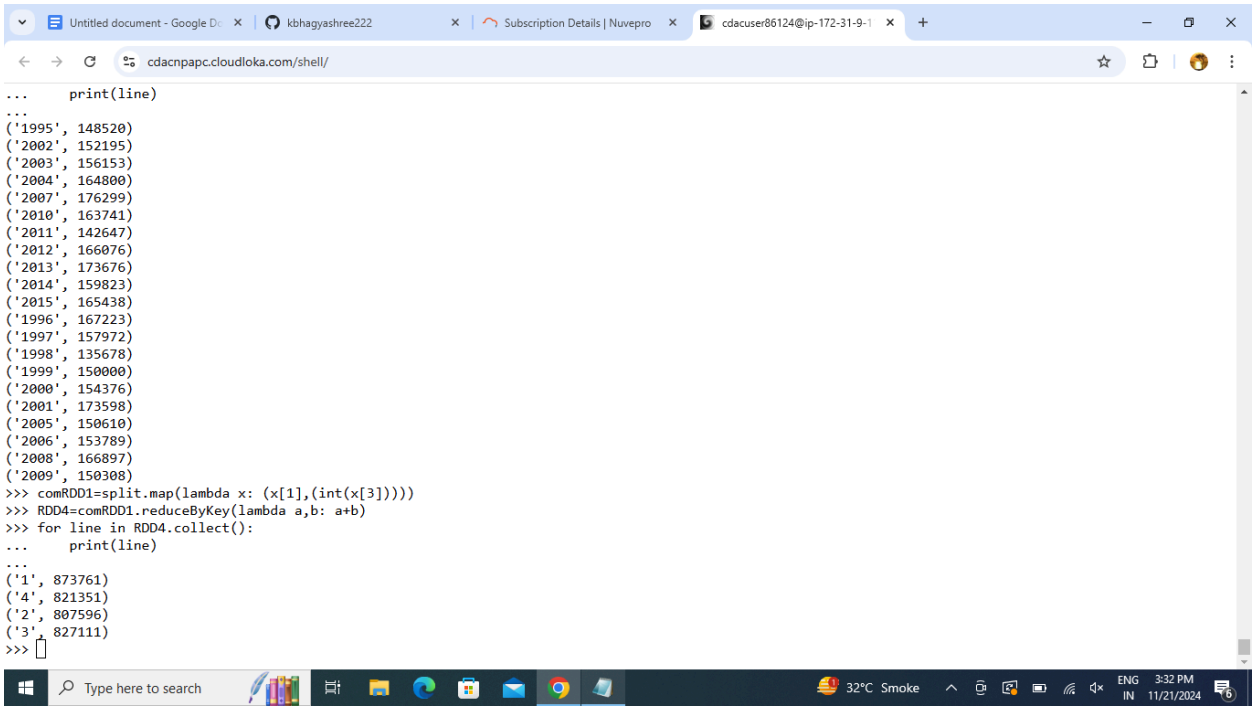
```
        at scala.collection.generic.Growable.$plus$plus$eq(Growable.scala:62)
        at scala.collection.generic.Growable.$plus$plus$eq$(Growable.scala:53)
        at scala.collection.mutable.ArrayBuffer.$plus$plus$eq(ArrayBuffer.scala:105)
        at scala.collection.mutable.ArrayBuffer.$plus$plus$eq(ArrayBuffer.scala:49)
        at scala.collection.TraversableOnce.to(TraversableOnce.scala:315)
        at scala.collection.TraversableOnce.to$(TraversableOnce.scala:313)
        at org.apache.spark.InterruptibleIterator.to(InterruptibleIterator.scala:28)
        at scala.collection.TraversableOnce.toBuffer(TraversableOnce.scala:307)
        at scala.collection.TraversableOnce.toBuffer$(TraversableOnce.scala:307)
        at org.apache.spark.InterruptibleIterator.toBuffer(InterruptibleIterator.scala:28)
        at scala.collection.TraversableOnce.toArray(TraversableOnce.scala:294)
        at scala.collection.TraversableOnce.toArray$(TraversableOnce.scala:288)
        at org.apache.spark.InterruptibleIterator.toArray(InterruptibleIterator.scala:28)
        at org.apache.spark.api.python.PythonRDD$.$anonfun$runJob$1(PythonRDD.scala:166)
        at org.apache.spark.SparkContext.$anonfun$runJob$5(SparkContext.scala:2236)
        at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
        at org.apache.spark.scheduler.Task.run(Task.scala:131)
        at org.apache.spark.executor.Executor$TaskRunner.$anonfun$run$3(Executor.scala:497)
        at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1439)
        at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:500)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
        ... 1 more

>>> avgRDD=split.map(lambda a: (a[3]).mean())
>>>
>>> RDD2=split.filter(lambda x: (float(x[2])>290.0))
>>> RDD2.take(5)
[('1995', '1', '296.9', '46561'), ('1995', '2', '296.8', '37443'), ('1997', '4', '293.51', '37454'), ('1998', '1', '304.74', '31315'), ('1998', '2',
'300.97', '30852')]
>>> RDD2.count()
75
>>>
```

3. Find the total no. booked seats for all quarter combined

```
comRDD=split.map(lambda x: (x[0],(int(x[3]))))
>>> RDD3=comRDD.reduceByKey(lambda a,b: a+b)
>>> for line in RDD3.collect():
...     print(line)
```

```
>>> RDD2=split.filter(lambda x: (float(x[2])>290.0))
>>> RDD2.take(5)
[('1995', '1', '296.9', '46561'), ('1995', '2', '296.8', '37443'), ('1997', '4', '293.51', '37454'), ('1998', '1', '304.74', '31315'), ('1998', '2',
'300.97', '30852')]
>>> RDD2.count()
75
>>> comRDD=split.map(lambda x: (x[0],(int(x[3]))))
>>> RDD3=comRDD.reduceByKey(lambda a,b: a+b)
>>> for line in RDD3.collect():
...     print(line)
...
('1995', 148520)
('2002', 152195)
('2003', 156153)
('2004', 164800)
('2007', 176299)
('2010', 163741)
('2011', 142647)
('2012', 166076)
('2013', 173676)
('2014', 159823)
('2015', 165438)
('1996', 167223)
('1997', 157972)
('1998', 135678)
('1999', 150000)
('2000', 154376)
('2001', 173598)
('2005', 150610)
('2006', 153789)
('2008', 166897)
('2009', 150308)
>>>
```

```
comRDD1=split.map(lambda x: (x[1],(int(x[3]))))
>>> RDD4=comRDD1.reduceByKey(lambda a,b: a+b)
>>> for line in RDD4.collect():
...     print(line)
...
('1', 873761)
('4', 821351)
('2', 807596)
('3', 827111)
```



4. List all distinct years

```
>>> year=split.map(lambda a: (a[0],1))
>>> rdd1=year.reduceByKey(lambda a,b: a+b)
>>> rdd1.take(10)
[('1995', 4), ('2002', 4), ('2003', 4), ('2004', 4), ('2007', 4),
('2010', 4), ('2011', 4), ('2012', 4), ('2013', 4), ('2014', 4)]
>>> rdd1.count()
21
>>> rdd2=rdd1.map(lambda y: y[0])
>>> rdd2.take(6)
['1995', '2002', '2003', '2004', '2007', '2010']
```

5.

```
revtotal=split.map(lambda x: (x[0],( (float(x[2]))*(int(x[3])) )))

rdd7=revtotal.reduceByKey(lambda a,b: a+b)

For line in rdd7.collect():
     print(line)
```

Untitled document - Google Do | kbhagyashree222 | Subscription Details | Nuvepro | cdacuser86124@ip-172-31-9-1

cdacnpapc.cloudloka.com/shell/

```
>>> dataRDD=sc.textFile("/user/cdacuser86124/airlines.csv")
>>> header=dataRDD.first()
>>> newRDD=dataRDD.filter(lambda line: line!=header)
>>> split=newRDD.map(lambda a: (a.split(',')[0],a.split(',')[1],a.split(',')[2],a.split(',')[3]))
>>> revtotal=split.map(lambda x: (x[0],( (float(x[2]))*(int(x[3])) )))
>>> rdd7=revtotal.reduceByKey(lambda a,b: a+b)
>>> rdd7.take(5)
[('1995', 43494243.22), ('2002', 47499146.5), ('2003', 49273210.83), ('2004', 50631364.949999996), ('2007', 57309216.07)]
>>> for line in rdd7.collect():
...     print(line)
...
('1995', 43494243.22)
('2002', 47499146.5)
('2003', 49273210.83)
('2004', 50631364.949999996)
('2007', 57309216.07)
('2010', 54861521.29)
('2011', 51888286.22)
('2012', 62199127.28)
('2013', 66363208.71)
('2014', 62624175.85000001)
('2015', 62378990.57)
('1996', 46358778.03)
('1997', 45385236.16)
('1998', 42035717.78)
('1999', 48757714.48)
('2000', 52342926.550000004)
('2001', 55533779.99999999)
('2005', 46376786.24)
('2006', 50437898.419999994)
('2008', 57653170.760000005)
('2009', 46746446.59)
>>>
```

Type here to search 32°C Smoke ENG IN 3:44 PM 11/21/2024