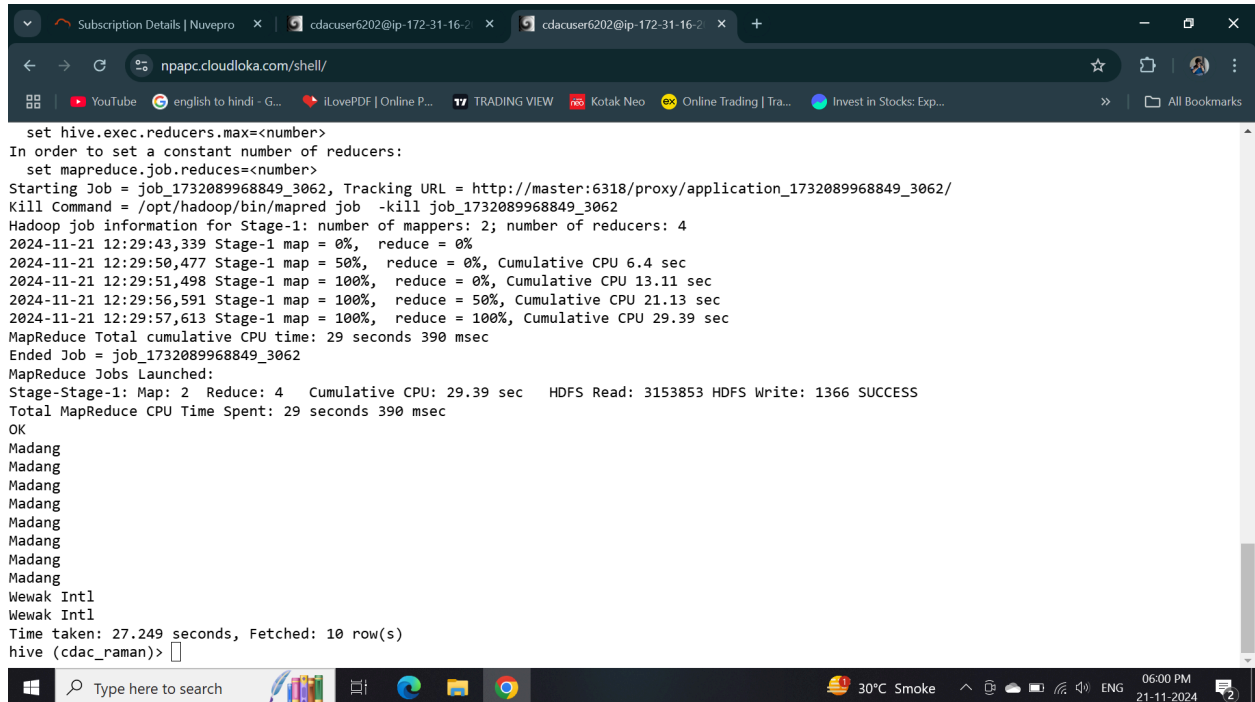


Hive

Question 1 .

```
1.select a.name from airport a join routes r on
a.airport_id=r.src_airport_id where src_airport_id != dest_airport_id
limit
10 ;
```



The screenshot shows a terminal window with a web browser interface at the top. The browser tabs include 'Subscription Details | Nuvepro', 'cdacuser6202@ip-172-31-16-2', and 'cdacuser6202@ip-172-31-16-2'. The address bar shows 'npapc.cloudloka.com/shell/'. The terminal output displays the following text:

```
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reducers=<number>
Starting Job = job_1732089968849_3062, Tracking URL = http://master:6318/proxy/application_1732089968849_3062/
Kill Command = /opt/hadoop/bin/mapred job -kill job_1732089968849_3062
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 4
2024-11-21 12:29:43,339 Stage-1 map = 0%, reduce = 0%
2024-11-21 12:29:50,477 Stage-1 map = 50%, reduce = 0%, Cumulative CPU 6.4 sec
2024-11-21 12:29:51,498 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 13.11 sec
2024-11-21 12:29:56,591 Stage-1 map = 100%, reduce = 50%, Cumulative CPU 21.13 sec
2024-11-21 12:29:57,613 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 29.39 sec
MapReduce Total cumulative CPU time: 29 seconds 390 msec
Ended Job = job_1732089968849_3062
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2 Reduce: 4 Cumulative CPU: 29.39 sec HDFS Read: 3153853 HDFS Write: 1366 SUCCESS
Total MapReduce CPU Time Spent: 29 seconds 390 msec
OK
Madang
Madang
Madang
Madang
Madang
Madang
Madang
Madang
Wewak Intl
Wewak Intl
Time taken: 27.249 seconds, Fetched: 10 row(s)
hive (cdac_raman)>
```

```
2. select air.name from airlines air join routes r
air.airline_id=r.airline_id where src_airport_id = dest_airport_id
limit
1;
```

```
Subscription Details | Nuvepro x cdacuser6202@ip-172-31-16-2 x cdacuser6202@ip-172-31-16-2 x +
npapc.cloudloka.com/shell/
hive (cdac_raman)> select air.name from airlines air join routes r on air.airline_id=r.airline_id where src_airport_id = dest_airport_id 1
imit 10;
Query ID = cdacuser6202_20241121123334_2becf501-87f2-41cf-9c03-9b213c57c58c
Total jobs = 1
Launching Job 1 out of 1
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_3070, Tracking URL = http://master:6318/proxy/application_1732089968849_3070/
Kill Command = /opt/hadoop/bin/mapred job -kill job_1732089968849_3070
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 4
2024-11-21 12:33:45,409 Stage-1 map = 0%, reduce = 0%
2024-11-21 12:33:53,571 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 12.59 sec
2024-11-21 12:33:59,695 Stage-1 map = 100%, reduce = 50%, Cumulative CPU 19.44 sec
2024-11-21 12:34:00,717 Stage-1 map = 100%, reduce = 75%, Cumulative CPU 22.99 sec
2024-11-21 12:34:02,752 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 26.31 sec
MapReduce Total cumulative CPU time: 26 seconds 310 msec
Ended Job = job_1732089968849_3070
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2 Reduce: 4 Cumulative CPU: 26.31 sec HDFS Read: 2729520 HDFS Write: 377 SUCCESS
Total MapReduce CPU Time Spent: 26 seconds 310 msec
OK
Illinois Airways
Time taken: 33.311 seconds, Fetched: 1 row(s)
hive (cdac_raman)>
```

3.

```
select count(distinct(airline_id)) from routes;
```

```
Subscription Details | Nuvepro x cdacuser6202@ip-172-31-16-2 x cdacuser6202@ip-172-31-16-2 x +
npapc.cloudloka.com/shell/
OK
10121
Time taken: 1.341 seconds, Fetched: 1 row(s)
hive (cdac_raman)> select count(distinct(airline_id)) from routes;
Query ID = cdacuser6202_20241121122348_58f2a825-46a1-4467-a73d-90b5040de013
Total jobs = 1
Launching Job 1 out of 1
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_3051, Tracking URL = http://master:6318/proxy/application_1732089968849_3051/
Kill Command = /opt/hadoop/bin/mapred job -kill job_1732089968849_3051
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-11-21 12:24:03,939 Stage-1 map = 0%, reduce = 0%
2024-11-21 12:24:10,068 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.99 sec
2024-11-21 12:24:16,197 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 7.24 sec
MapReduce Total cumulative CPU time: 7 seconds 240 msec
Ended Job = job_1732089968849_3051
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 7.24 sec HDFS Read: 2385309 HDFS Write: 103 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 240 msec
OK
540
Time taken: 31.541 seconds, Fetched: 1 row(s)
hive (cdac_raman)>
```

Spark

Question 1 (b)

```
data = sc.textFile("/user/cdacuser6202/airlines.csv")
>>> data.count()
header = data.first()
>>> head_fil = data.filter(lambda line : line != header)
>>> head_fil.take(2)
dataa = head_fil.map(lambda x :
(x.split(",")[0],x.split(",")[1],float(x.split(",")[2]),x.split(",")[
3]))
>>> dataa1 = dataa.map(lambda x : (x[0],x[1]))
dataa1 = dataa.map(lambda x : (x[0],x[1]))
dataa1.take(10)
for line in dataa1.take(10):
...     print(line)
```

```
Spark context available as 'sc' (master = yarn, app id = application_1732089968849_2726).
SparkSession available as 'spark'.
>>> data = sc.textFile("/user/cdacuser6202/airlines.csv")
>>> data.count()
85
>>> header = data.first()
>>> head_fil = data.filter(lambda line : line != header)
>>> head_fil.take(2)
['1995,1,296.9,46561', '1995,2,296.8,37443']
>>> dataa = head_fil.map(lambda x : (x.split(",")[0],x.split(",")[1],float(x.split(",")[2]),x.split(",")[3]))
>>> dataa1 = dataa.map(lambda x : (x[0],x[1]))
>>> dataa2 = dataa1.reduceByKey(lambda a,b : a+b).sortByKey(lambda a : -a[1])
>>> dataa2.take(5)
[('1995', '1234'), ('1996', '1234'), ('1997', '1234'), ('1998', '1234'), ('1999', '1234')]
>>> dataa2 = dataa1.reduceByKey(lambda a,b : a+b)
>>> dataa2.take(5)
[('1995', '1234'), ('2002', '1234'), ('2003', '1234'), ('2004', '1234'), ('2007', '1234')]
>>> for line in dataa2.take(5):
...     File "<stdin>", line 1
...     for line in dataa2.take(5)
...         ^
SyntaxError: invalid syntax
>>> for line in dataa2.take(5):
...     print(line)
...
('1995', '1234')
('2002', '1234')
('2003', '1234')
('2004', '1234')
('2007', '1234')
>>>
SyntaxError: invalid syntax
>>> for line in dataa2.take(5):
...     print(line)
...
('1995', '1234')
('2002', '1234')
('2003', '1234')
('2004', '1234')
('2007', '1234')
>>> dataa1 = dataa.map(lambda x : (x[0],x[1]))
>>> dataa1
PythonRDD[22] at RDD at PythonRDD.scala:53
>>> dataa1.take(10)
[('1995', '1'), ('1995', '2'), ('1995', '3'), ('1995', '4'), ('1996', '1'), ('1996', '2'), ('1996', '3'), ('1996', '4'), ('1997', '1'), ('1997', '2')]
>>> for line in dataa1.take(10):
...     print(line)
...
('1995', '1')
('1995', '2')
('1995', '3')
('1995', '4')
('1996', '1')
('1996', '2')
('1996', '3')
('1996', '4')
('1997', '1')
('1997', '2')
>>>
```

Question 2

1.

```
min = dataa.map(lambda x : ( x[0]+" "+x[1],x[3]))
>>> minn = min.reduceByKey(lambda a,b : a+b)
```

```
minnn = minn.sortBy(lambda a : a[1])
minnn.take(1)
```

```

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$(TraversableOnce.scala:288)
at org.apache.spark.InterruptibleIterator.toArray(InterruptibleIterator.scala:28)
at org.apache.spark.rdd.RDD.$anonfun$collect$2(RDD.scala:1030)
at org.apache.spark.SparkContext.$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

>>> minnn = minn.sortBy(lambda a : a[1])
>>> minnn.take(10)
[('2000 4', '30103'), ('1995 4', '30388'), ('2011 1', '30562'), ('1998 2', '30852'), ('2004 3', '30877'), ('1999 4', '31256'), ('1998 1', '31315'), ('2005 1', '32003'), ('2002 4', '32406'), ('2009 2', '32491')]
>>> minnn.take(1)
[('2000 4', '30103')]
>>>

```

```
max = minn.sortBy(lambda a : a[0])
>>> max.take(5)
[('1995 1', '46561'), ('1995 2', '37443'), ('1995 3', '34128'),
 ('1995 4', '30388'), ('1996 1', '47808')]
>>> max.take(1)
[('1995 1', '46561')]
```

```
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.rdd.RDD.$anonfun$collect$2(RDD.scala:1030)
at org.apache.spark.SparkContext.$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

>>> max = minn.sortBy(lambda a : a[])
File "<stdin>", line 1
max = minn.sortBy(lambda a : a[])
^
SyntaxError: invalid syntax
>>> max = minn.sortBy(lambda a : a[0])
>>> max.take(5)
[('1995 1', '46561'), ('1995 2', '37443'), ('1995 3', '34128'), ('1995 4', '30388'), ('1996 1', '47808')]
>>> max.take(1)
[('1995 1', '46561')]
>>>
```

2.

```
at org.apache.spark.InterruptibleIterator.foreach(InterruptibleIterator.scala:28)
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.rdd.RDD.$anonfun$collect$2(RDD.scala:1030)
at org.apache.spark.SparkContext.$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

>>> min = dataa.map(lambda x : (x[3] < '290')).count()
>>> min
84
>>>
```

```
3 min = dataa.map(lambda x : ( x[0]+" "+x[1],x[3]))
>>> minn = min.reduceByKey(lambda a,b : a+b)
>>> minn.take(5)
for line in minn.take(5):
```

```
...     print(line)
```

5

```
data2 = data1.map(lambda x : (x[0]+' '+x[1],x[2]))
data3 = data2.reduceByKey(lambda a,b : a+b)
data4 = data3.sortBy(lambda a : -a[1])
data4.take(1)
```

```
Subscription Details | Nuvepro x cdacuser6202@ip-172-31-16-2 x cdacuser6202@ip-172-31-16-2 x +
npapc.cloudloka.com/shell/
at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:458)
at org.apache.spark.shuffle.sort.BypassMergeSortShuffleWriter.write(BypassMergeSortShuffleWriter.java:132)
at org.apache.spark.shuffle.ShuffleWriteProcessor.write(ShuffleWriteProcessor.scala:59)
at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:99)
at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:52)
at org.apache.spark.scheduler.Task.run(Task.scala:131)
at org.apache.spark.executor.Executor$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

>>> data2 = data1.map(lambda x : (x[1]+" "+x[1]))
>>> data2 = data1.map(lambda x : (x[0]+" "+x[1],x[2]))
>>> data3 = data2.reduceByKey(lambda a,b : a+b)
>>> data4 = data3.sortBy(lambda a : a[1])
>>> data4.tail(5)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'PipelinedRDD' object has no attribute 'tail'
>>> data4.take(5)
[('1996 3', 269.49), ('1996 2', 275.78), ('1996 4', 278.33), ('1997 3', 282.27), ('1997 1', 283.4)]
>>> data4 = data3.sortBy(lambda a :-a[1])
>>> data4.take(5)
[('2014 3', 396.37), ('2014 2', 395.62), ('2014 4', 392.66), ('2013 3', 390.04), ('2015 1', 388.32)]
>>> data4.take(1)
[('2014 3', 396.37)]
>>>
```