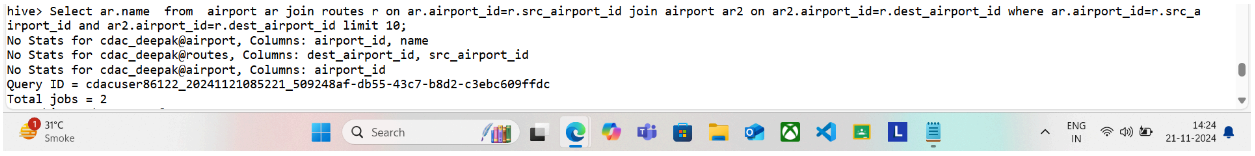
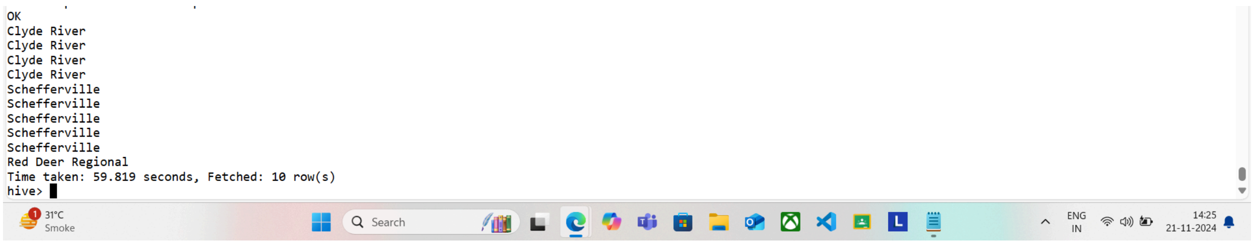
name-deepakBourai
Roll no- 240840325021

q1.

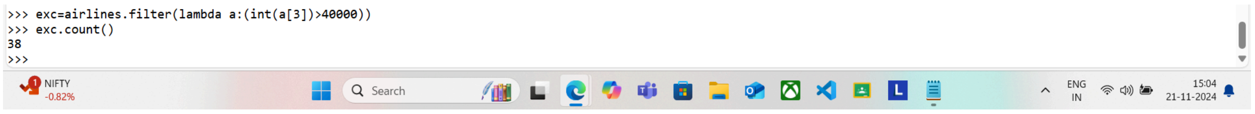


```
hive> Select ar.name  from  airport ar join routes r on ar.airport_id=r.src_airport_id join airport ar2 on ar2.airport_id=r.dest_airport_id where ar.airport_id=r.src_a
irport_id and ar2.airport_id=r.dest_airport_id limit 10;
No Stats for cdac_deepak@airport, Columns: airport_id, name
No Stats for cdac_deepak@routes, Columns: dest_airport_id, src_airport_id
No Stats for cdac_deepak@airport, Columns: airport_id
Query ID = cdacuser86122_20241121085221_509248af-db55-43c7-b8d2-c3ebc609ffdc
Total jobs = 2
```



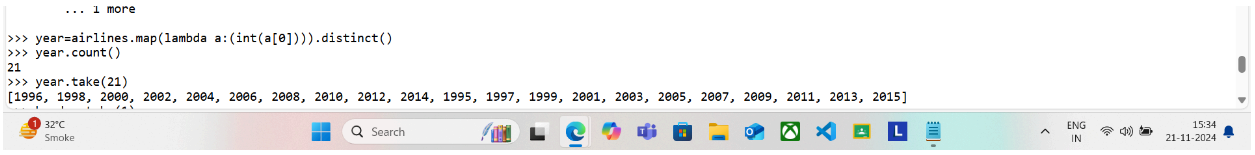
```
OK
Clyde River
Clyde River
Clyde River
Clyde River
Schefferville
Schefferville
Schefferville
Schefferville
Schefferville
Red Deer Regional
Time taken: 59.819 seconds, Fetched: 10 row(s)
hive>
```

Pyspark.
Q1
prt-1



```
>>> exc=airlines.filter(lambda a:(int(a[3])>40000))
>>> exc.count()
38
>>>
```

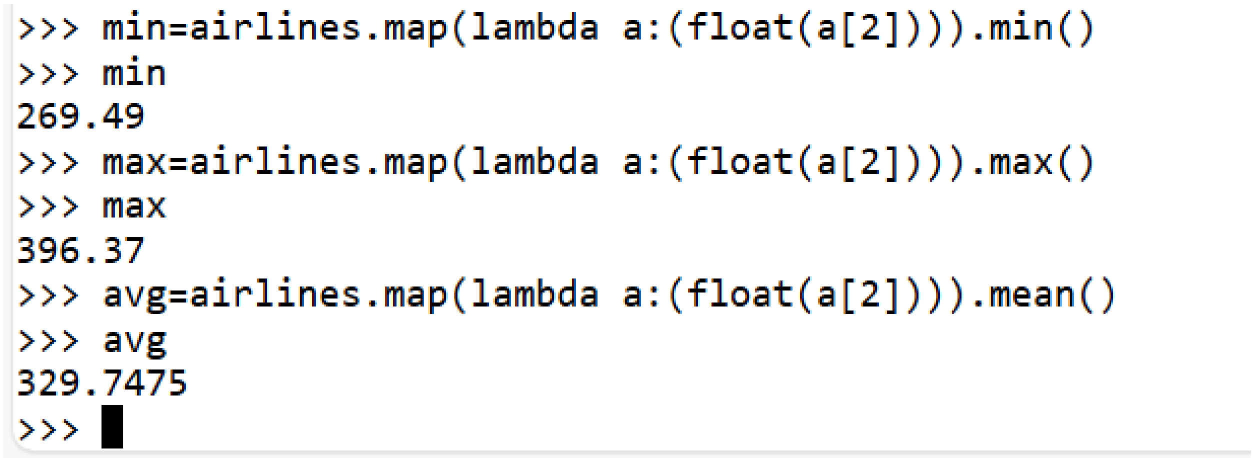
prt-2



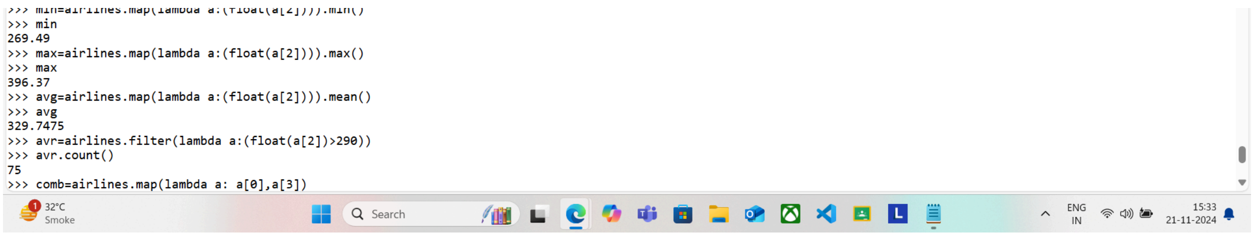
```
        ... 1 more

>>> year=airlines.map(lambda a:(int(a[0]))).distinct()
>>> year.count()
21
>>> year.take(21)
[1996, 1998, 2000, 2002, 2004, 2006, 2008, 2010, 2012, 2014, 1995, 1997, 1999, 2001, 2003, 2005, 2007, 2009, 2011, 2013, 2015]
```

Q2.
Part-1

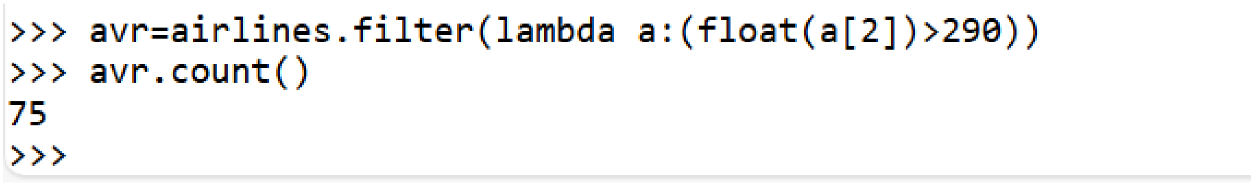


```
>>> min=airlines.map(lambda a:(float(a[2]))).min()
>>> min
269.49
>>> max=airlines.map(lambda a:(float(a[2]))).max()
>>> max
396.37
>>> avg=airlines.map(lambda a:(float(a[2]))).mean()
>>> avg
329.7475
>>> 
```

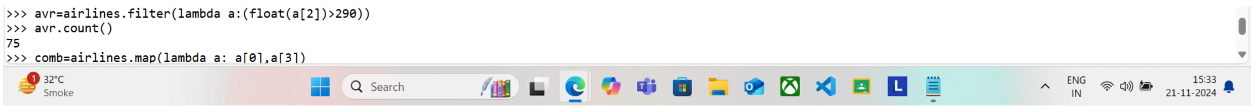


```
>>> min=airlines.map(lambda a:(float(a[2]))).min()
>>> min
269.49
>>> max=airlines.map(lambda a:(float(a[2]))).max()
>>> max
396.37
>>> avg=airlines.map(lambda a:(float(a[2]))).mean()
>>> avg
329.7475
>>> avr=airlines.filter(lambda a:(float(a[2])>290))
>>> avr.count()
75
>>> comb=airlines.map(lambda a: a[0],a[3])
```

Part.2

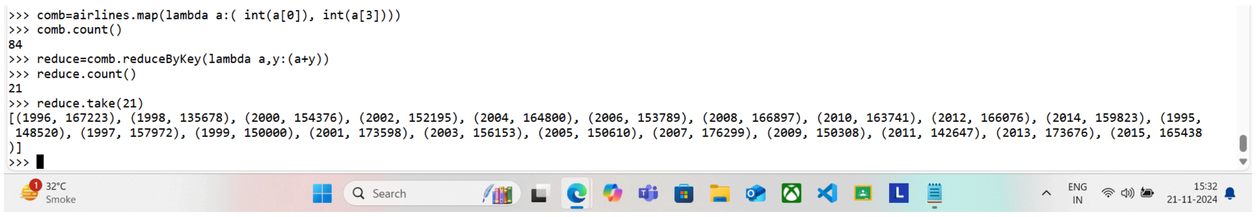


```
>>> avr=airlines.filter(lambda a:(float(a[2])>290))
>>> avr.count()
75
>>> 
```



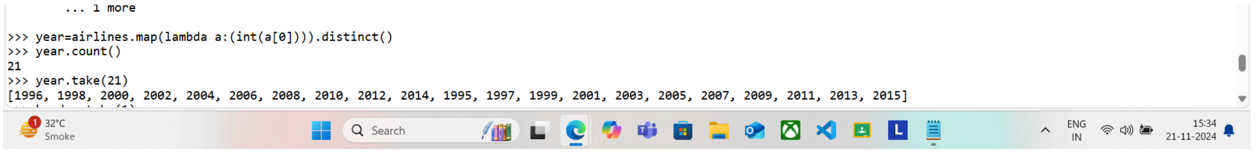
```
>>> avr=airlines.filter(lambda a:(float(a[2])>290))
>>> avr.count()
75
>>> comb=airlines.map(lambda a: a[0],a[3])
```

Part.3



```
>>> comb=airlines.map(lambda a:( int(a[0]), int(a[3])))
>>> comb.count()
84
>>> reduce=comb.reduceByKey(lambda a,y:(a+y))
>>> reduce.count()
21
>>> reduce.take(21)
[(1996, 167223), (1998, 135678), (2000, 154376), (2002, 152195), (2004, 164800), (2006, 153789), (2008, 166897), (2010, 163741), (2012, 166076), (2014, 159823), (1995,
 148520), (1997, 157972), (1999, 150000), (2001, 173598), (2003, 156153), (2005, 150610), (2007, 176299), (2009, 150308), (2011, 142647), (2013, 173676), (2015, 165438
)]
>>> 
```

Part.4



```
        ... 1 more

>>> year=airlines.map(lambda a:(int(a[0]))).distinct()
>>> year.count()
21
>>> year.take(21)
[1996, 1998, 2000, 2002, 2004, 2006, 2008, 2010, 2012, 2014, 1995, 1997, 1999, 2001, 2003, 2005, 2007, 2009, 2011, 2013, 2015]
```

Part5

Session was closed .so can’t able to print

comb=airlines.map(lambda a:((int(a[0])),(float(a[2]))x(int(a[3]))))

Now my new rdd have year and revenue,
I need to reduce the key bcs year is quarter wise.

Reduceb=comb.map(lambda a,b: (a+b))

reduceb.count()
reduced.take(21)