Name: Khushal Pareta

Student ID: 240840325031

**#HIVE**

**Question 1:**

1-> airports that are listed as both source and destination

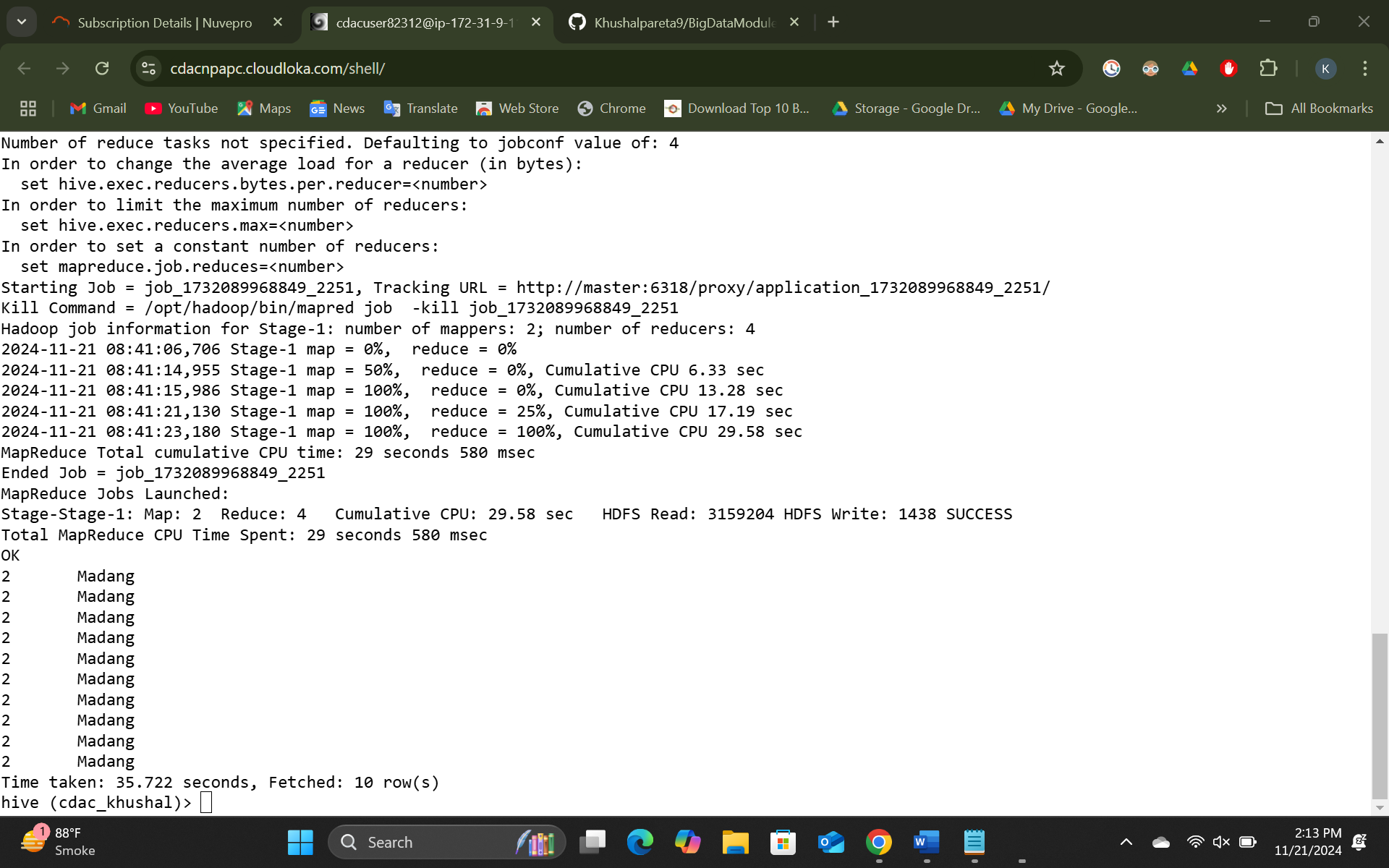
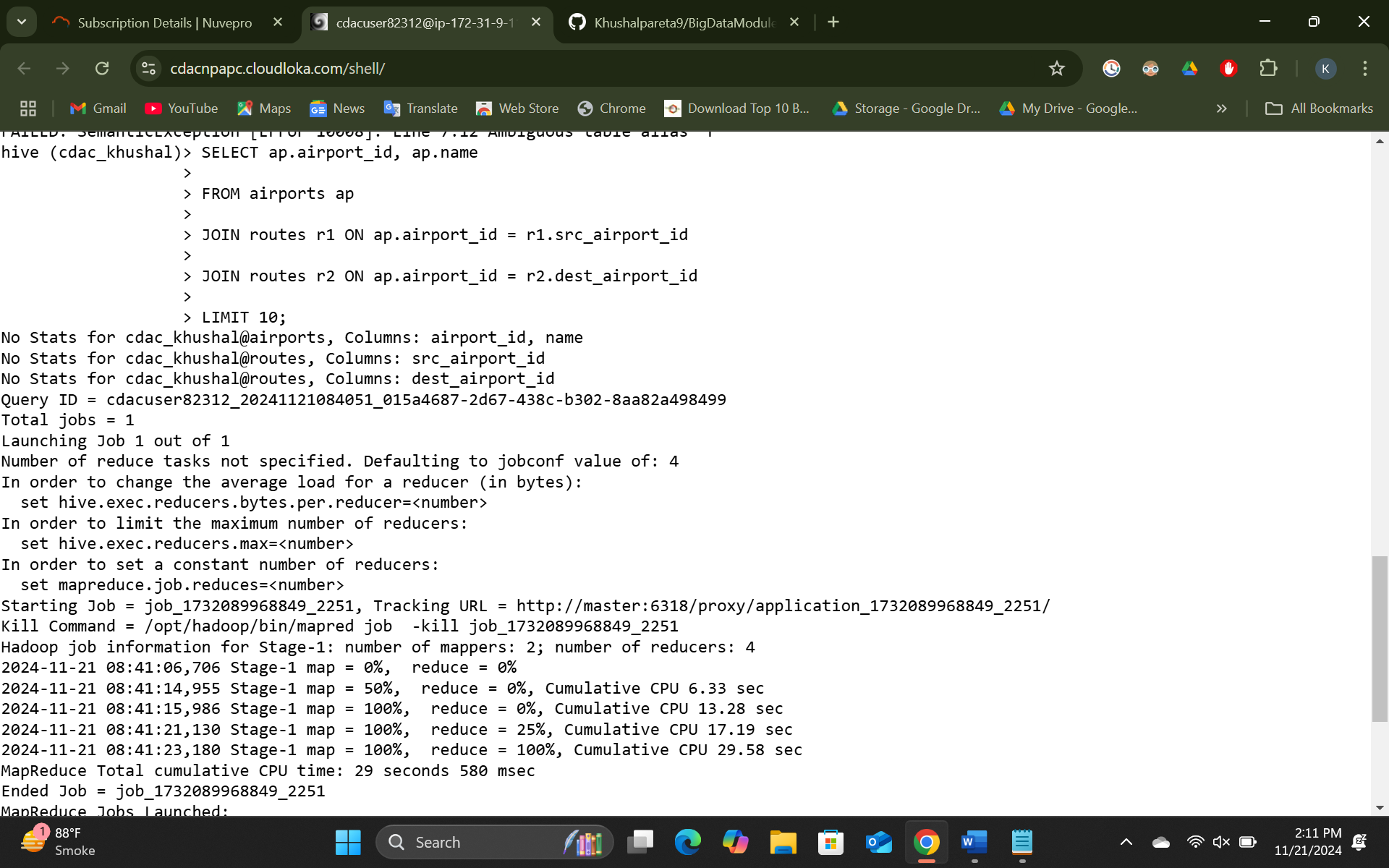
SELECT ap.airport\_id, ap.name

FROM airports ap

JOIN routes r1 ON ap.airport\_id = r1.src\_airport\_id

JOIN routes r2 ON ap.airport\_id = r2.dest\_airport\_id

LIMIT 10;



2 -> determine equipment that is used on highest number of routes

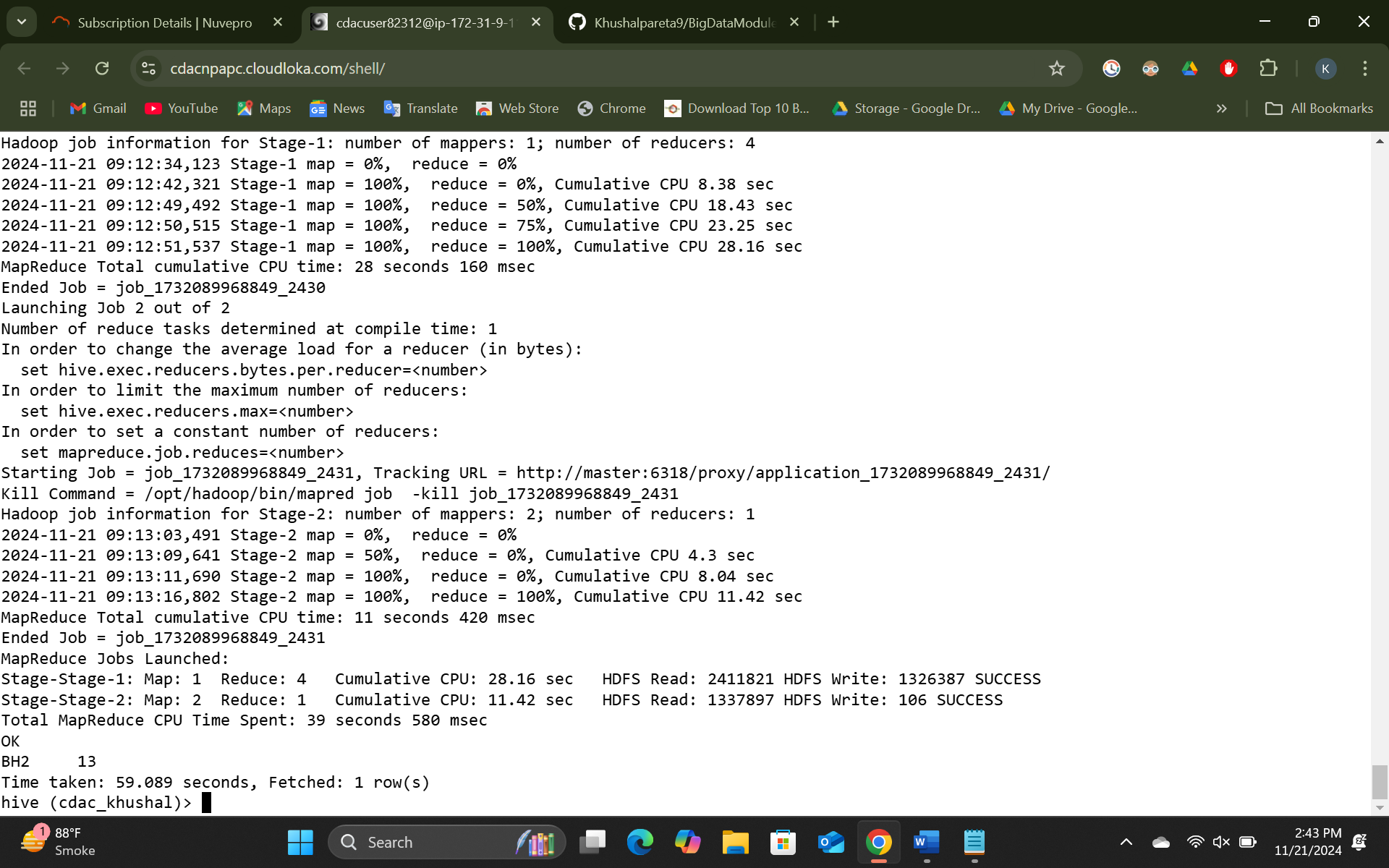
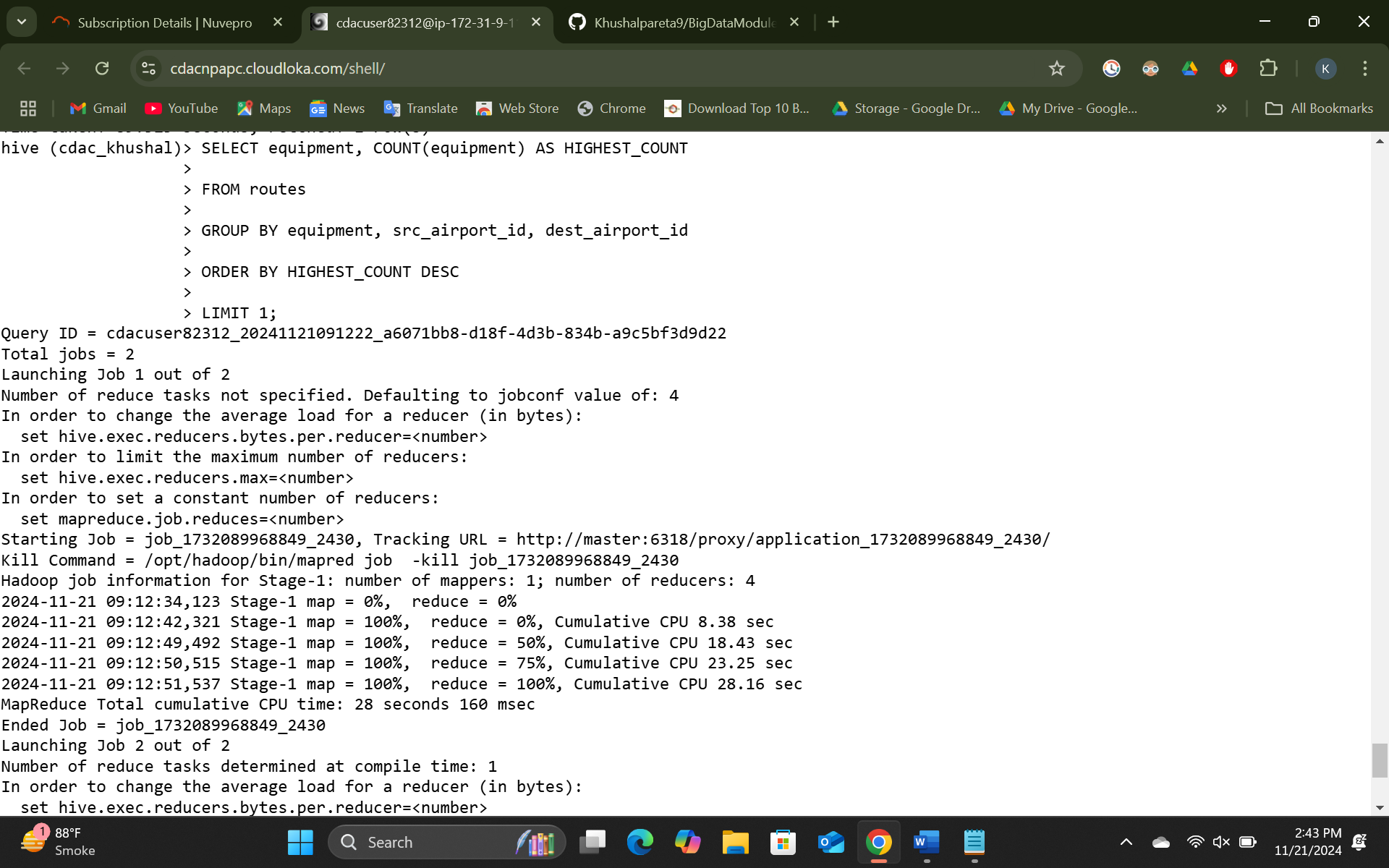
SELECT equipment, COUNT(equipment) AS HIGHEST\_COUNT

FROM routes

GROUP BY equipment, src\_airport\_id, dest\_airport\_id

ORDER BY HIGHEST\_COUNT DESC

LIMIT 1;



3 - > Airline which operates the highest number of routes and count of those routes

SELECT a.name, COUNT(a.airline\_id) AS ROUTE\_COUNT

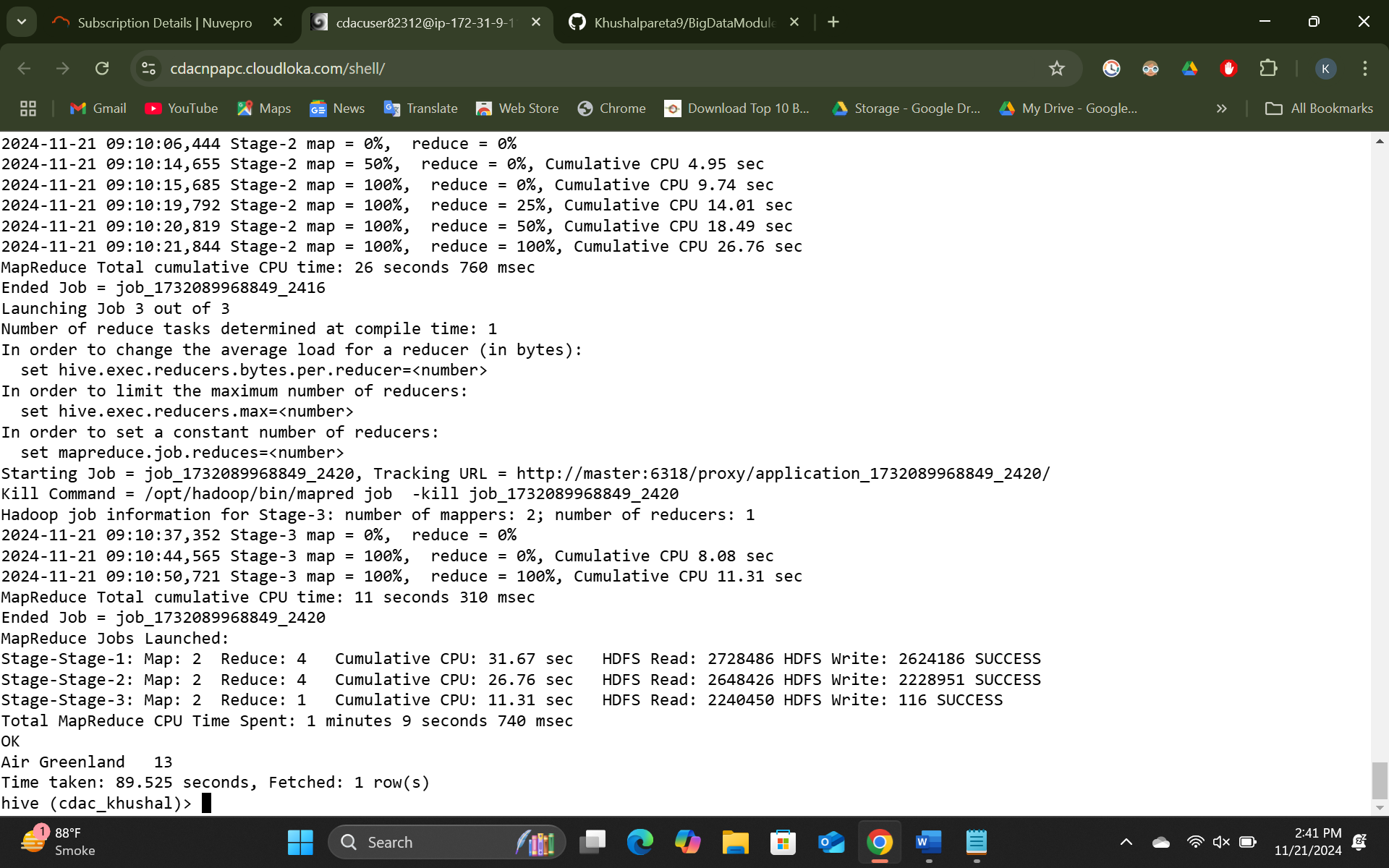
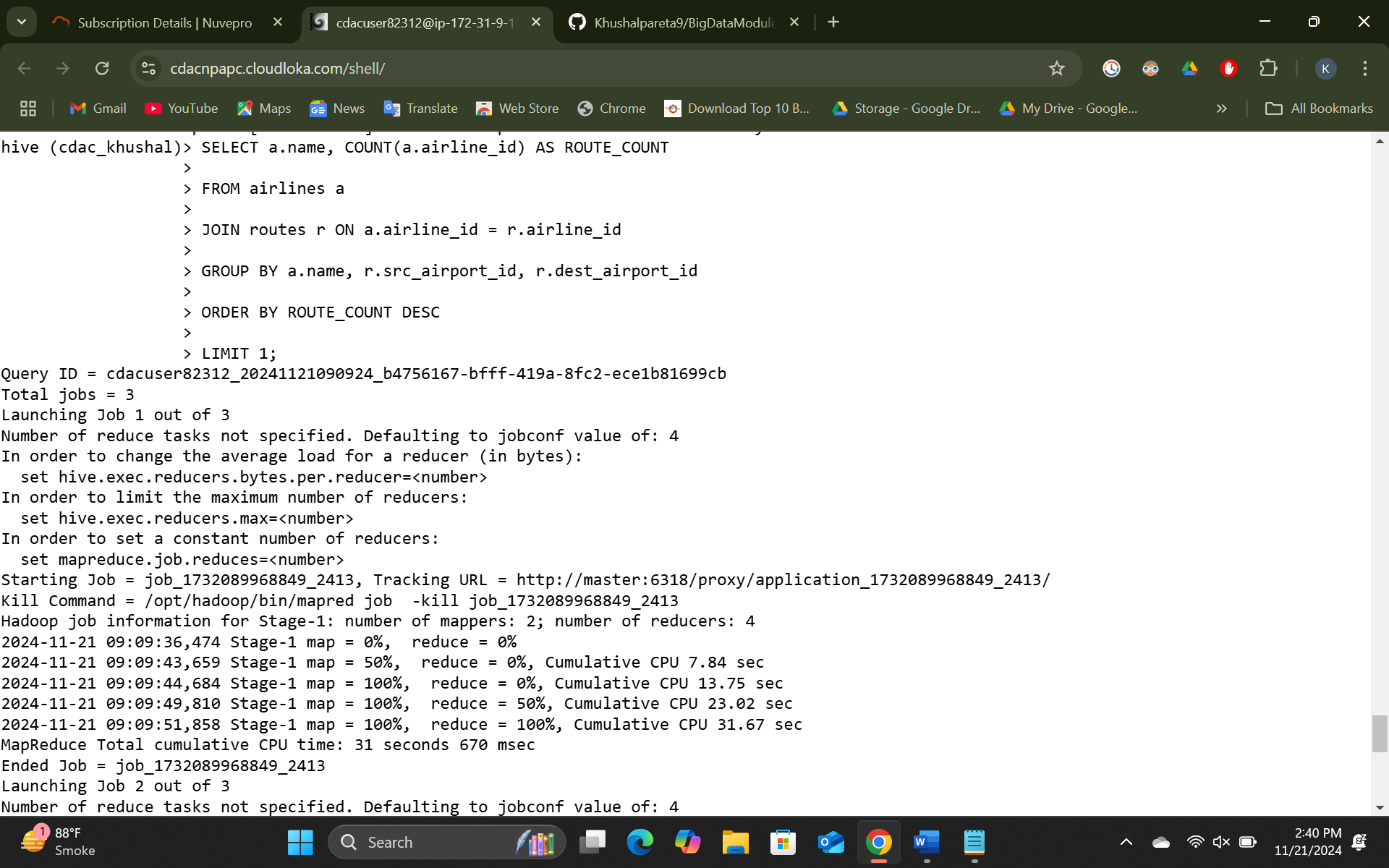
FROM airlines a

JOIN routes r ON a.airline\_id = r.airline\_id

GROUP BY a.name, r.src\_airport\_id, r.dest\_airport\_id

ORDER BY ROUTE\_COUNT DESC

LIMIT 1;

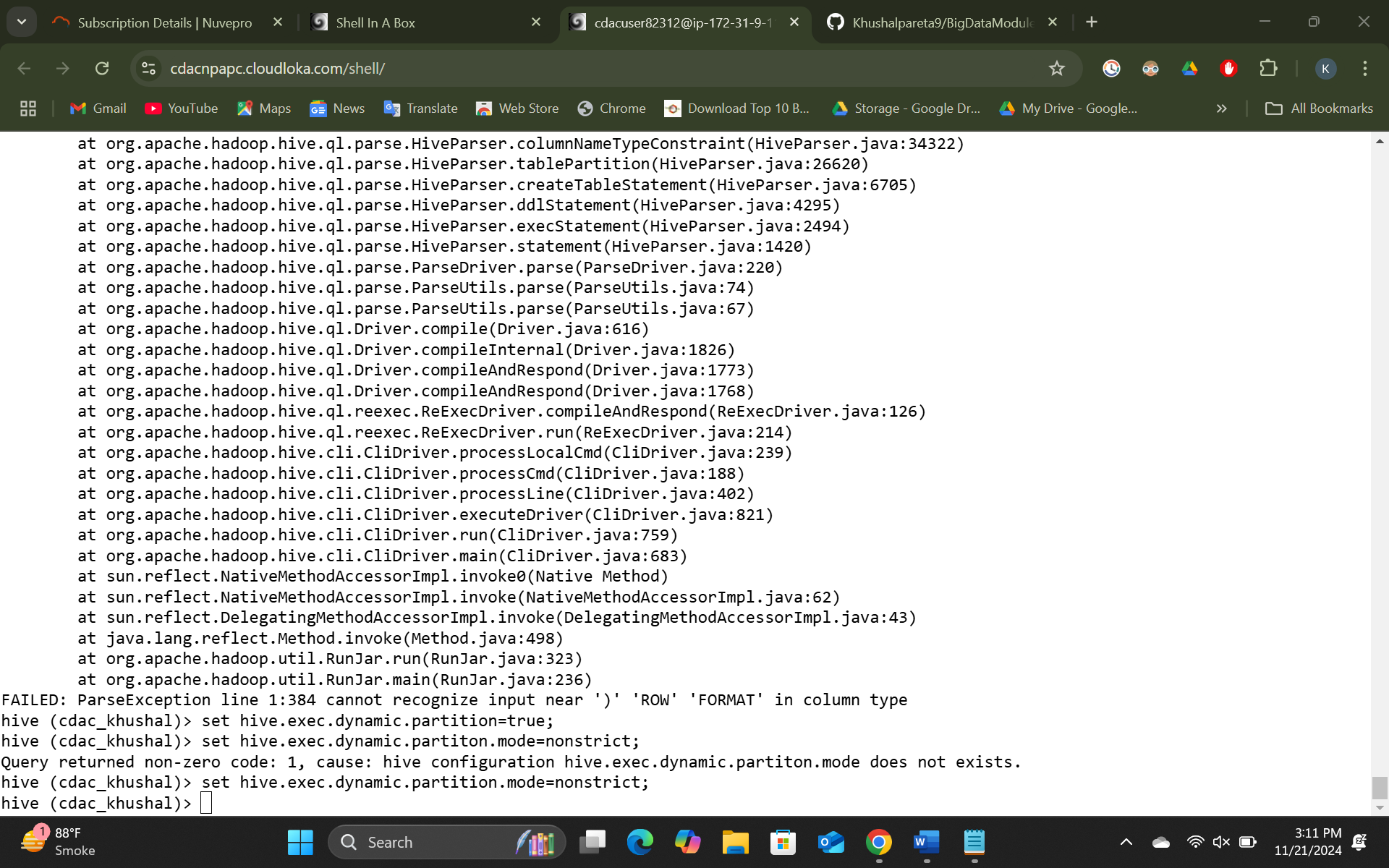


**Question 2:**

1 -> Create a partition table for the source\_airport, write a sql query to create this table and insert data into it.

SET hive.exec.dynamic.partition=true;

SET hive.exec.dynamic.partiton.mode=nonstrict;



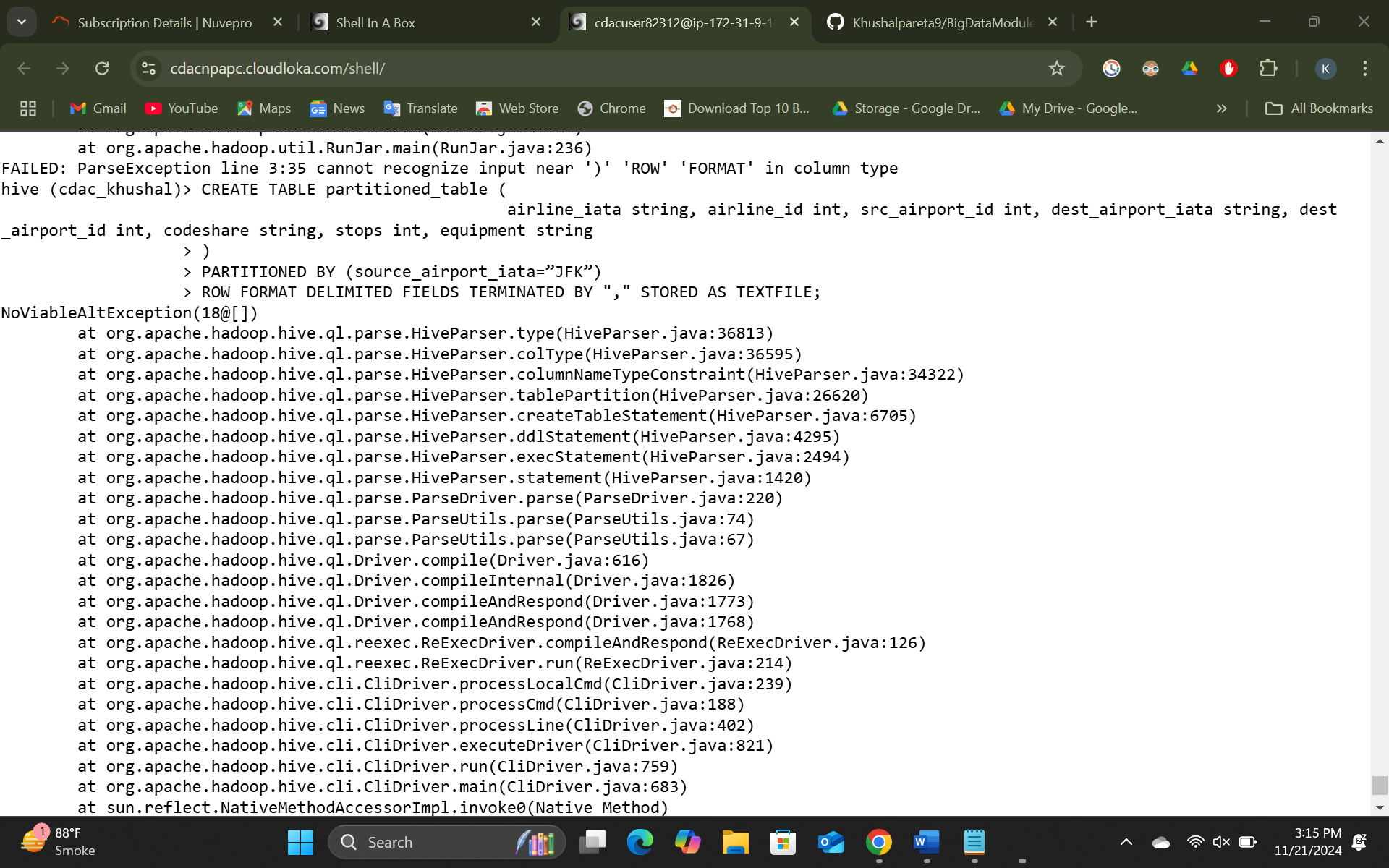
CREATE TABLE partitioned\_table (

airline\_iata string, airline\_id int, src\_airport\_id int, dest\_airport\_iata string, dest\_airport\_id int, codeshare string, stops int, equipment string

)

PARTITIONED BY (source\_airport\_iata=‘JFK’)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ STORED AS TEXTFILE;



2 ->

INSERT INTO TABLE partitioned\_table SELECT \* FROM airports WHERE src\_airport\_iata=‘JFK’

3 ->

SELECT \* FROM partitioned\_table WHERE src\_airport\_iata=‘LAX’;

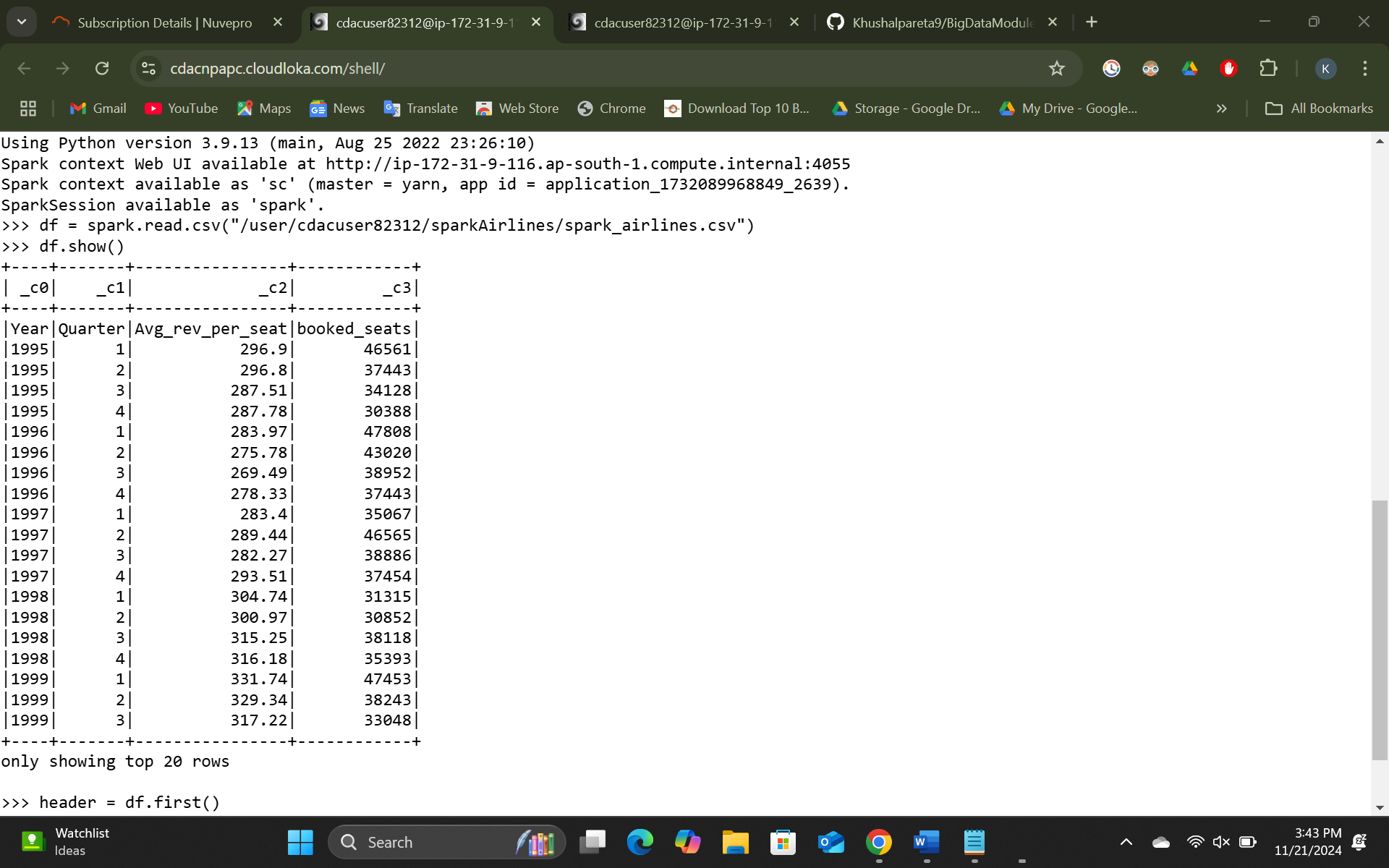
4 ->

SHOW paritions partitioned\_table

**#SPARK**

Question 1:

df = spark.read.csv(‘/user/cdacuser82312/sparkAirlines/spark\_airlines.csv’, header=True, inferSchema=True)



1 ->

From pyspark.sql.functions import sum

df.groupBy(‘Year’, ‘Quarter’).agg(count(sum(‘booked\_seats’)) > 40000).show()

2 ->

df.groupBy(‘Year’).show()

QUESTION 2:

1->

From pyspark.sql.functions import sum, avg, min

df.groupBy(‘Year’, ‘Quarter’).agg(sum(‘Avg\_rev\_per\_seat’).alias(‘TotalRevenuePerSeat’), avg(‘Avg\_rev\_per\_seat’).alias(‘AverageRevenuePerSeat’), min(‘Avg\_rev\_per\_seat’).alias(‘MinimumRevenuePerSeat’)).show()

2 ->

df.groupBy(‘Year’, ‘Quarter’).agg(count(avg\_rev\_per\_seat).alias(‘Total\_Count’) > 290)).show()

3 ->

From pyspark.sql.functions import sum

df.groupBy(‘Year’).agg(sum(‘booked\_seats’).alias(‘Total\_Booked\_Seats)).show()

4 ->

df.groupBy(‘year’).show()

5 ->

From pyspark.sql.functions import sum

df.groupBy(‘Year’).agg(sum(‘avg\_rev\_per\_seat’).alias(‘Total\_Avg\_Revenue’)).show()