**BIG DATA MODULE EXAM**

**220940325006\_Ajay\_Fatpure**

**---------------------------------------------------------------------------------------------------------------------------**

Hive

Please find the customer data set.

cust id

firstname

lastname

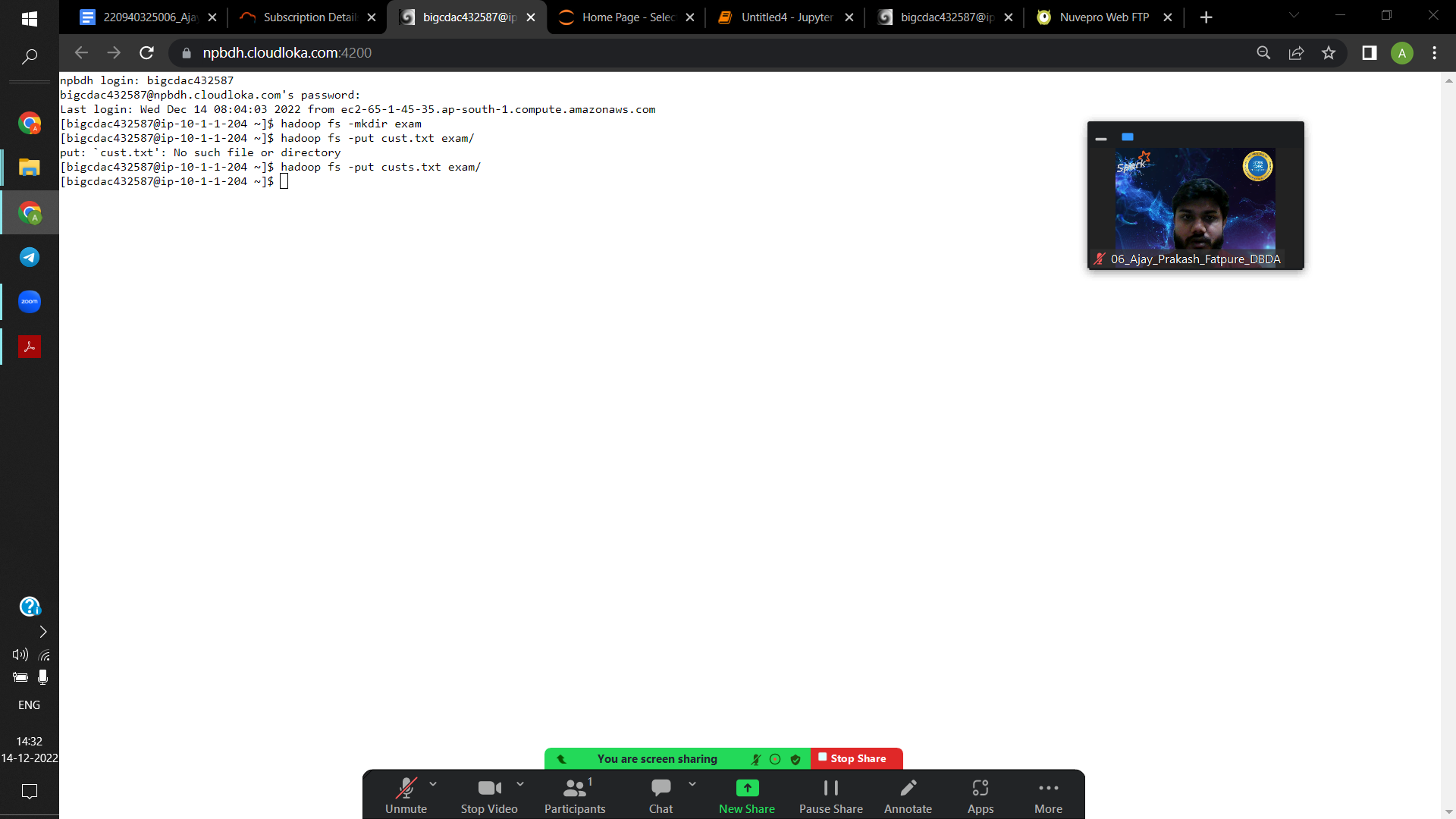
age

profession

1) Write a program to find the count of customers for each profession.

**Hadoop fs - mkdir exam**

**Hadoop fs -put cust.txt exam/**

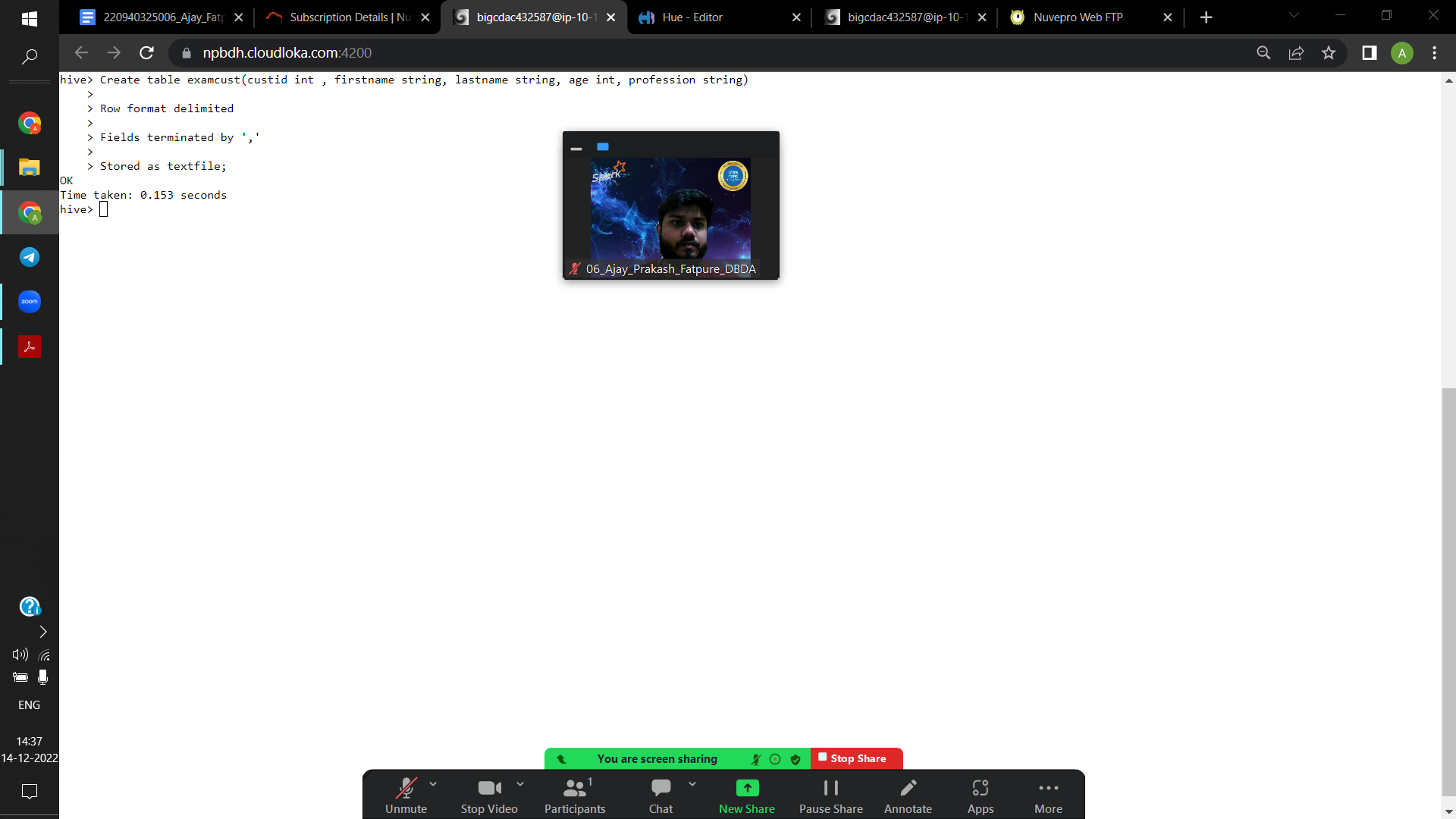


**Create table examcust(custid int , firstname string, lastname string, age int, profession string)**

**Row format delimited**

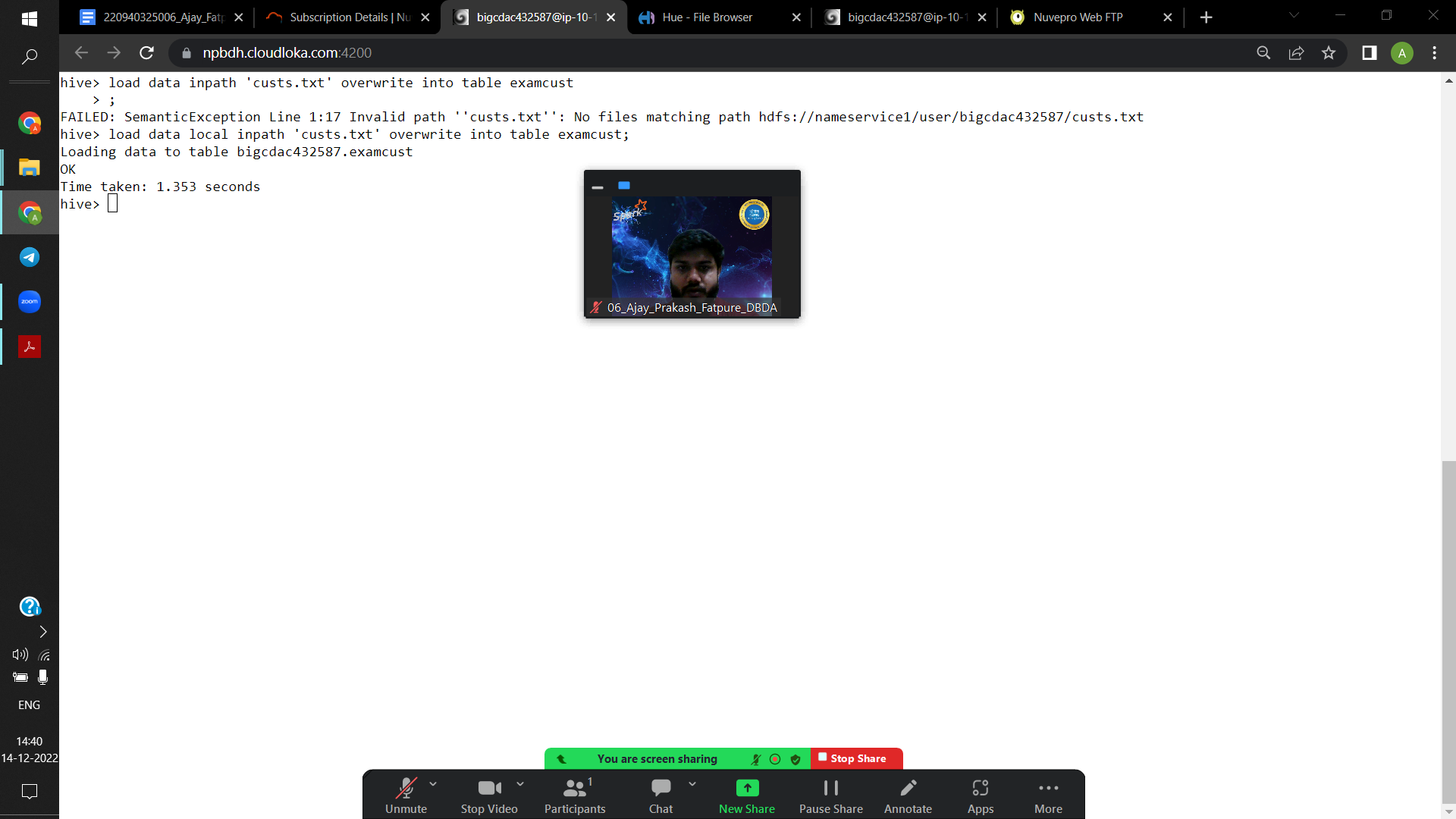
**Fields terminated by ‘,’**

**Stored as textfile;**

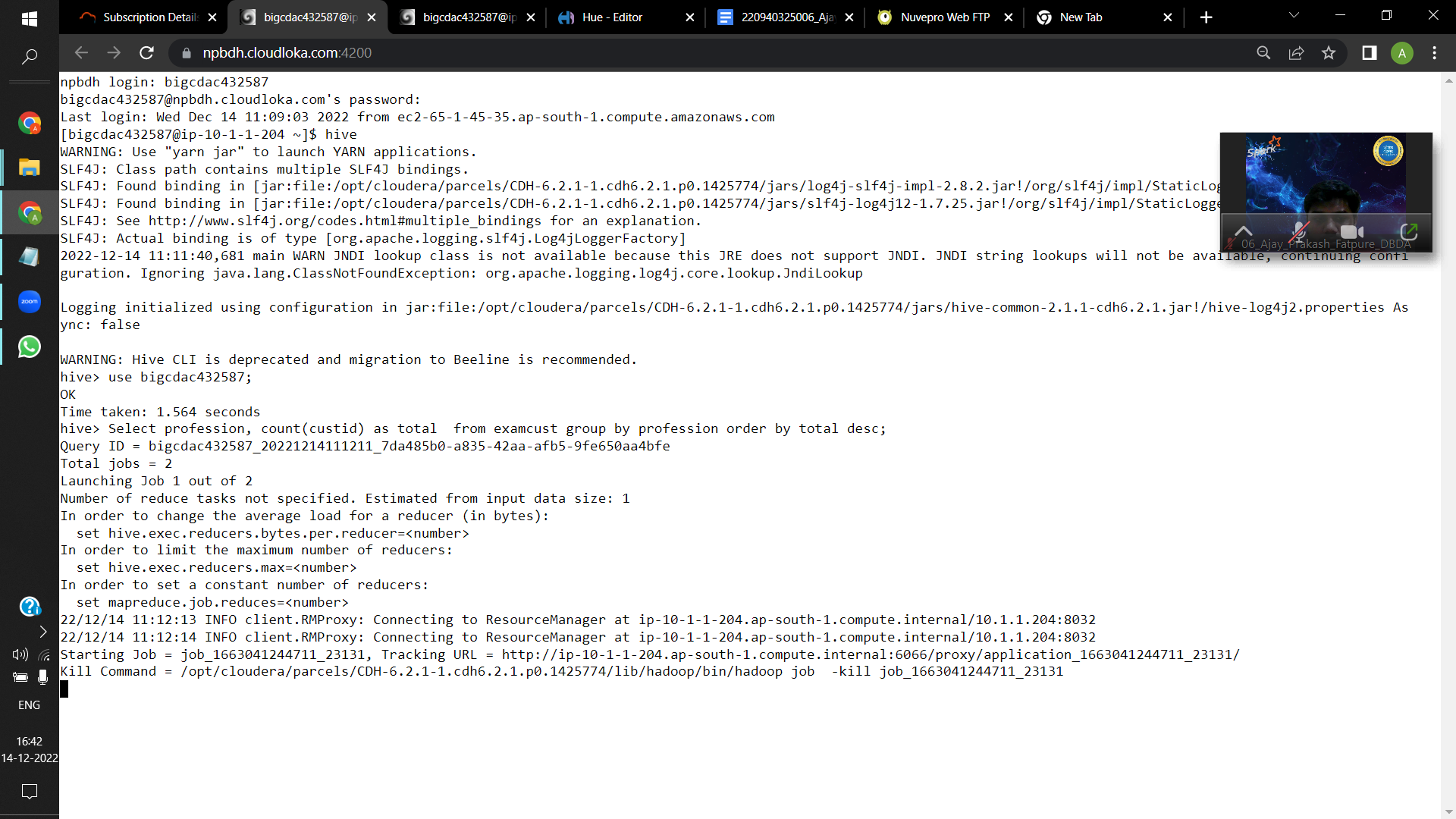




**Load data local inpath’custs.txt’ overwrite into table examcust;**



**Select profession, count(profession) as total from examcust group by profession order by total desc;**

****

=====================================================================

Please find the sales data set.

txn id

txn date

cust id

amount

category

product

city

state

Spendby

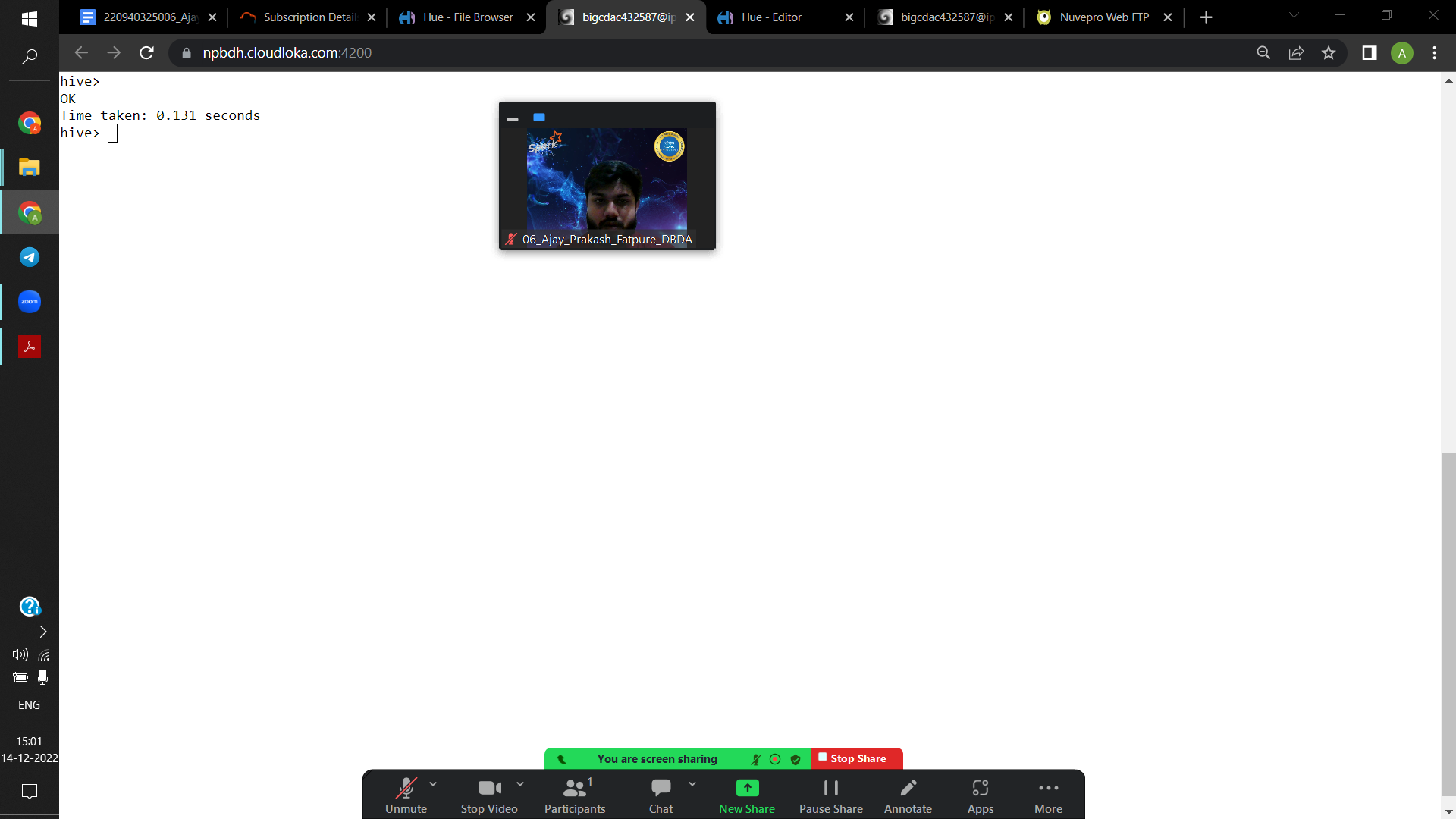
2) Write a program to find the top 10 products sales wise

**Create table saletable(txnid int, txndate string, custid int, amount double, category string, product string, city string, state string, spendby string)**

**Row format delimited**

**Fields terminated by ‘,’**

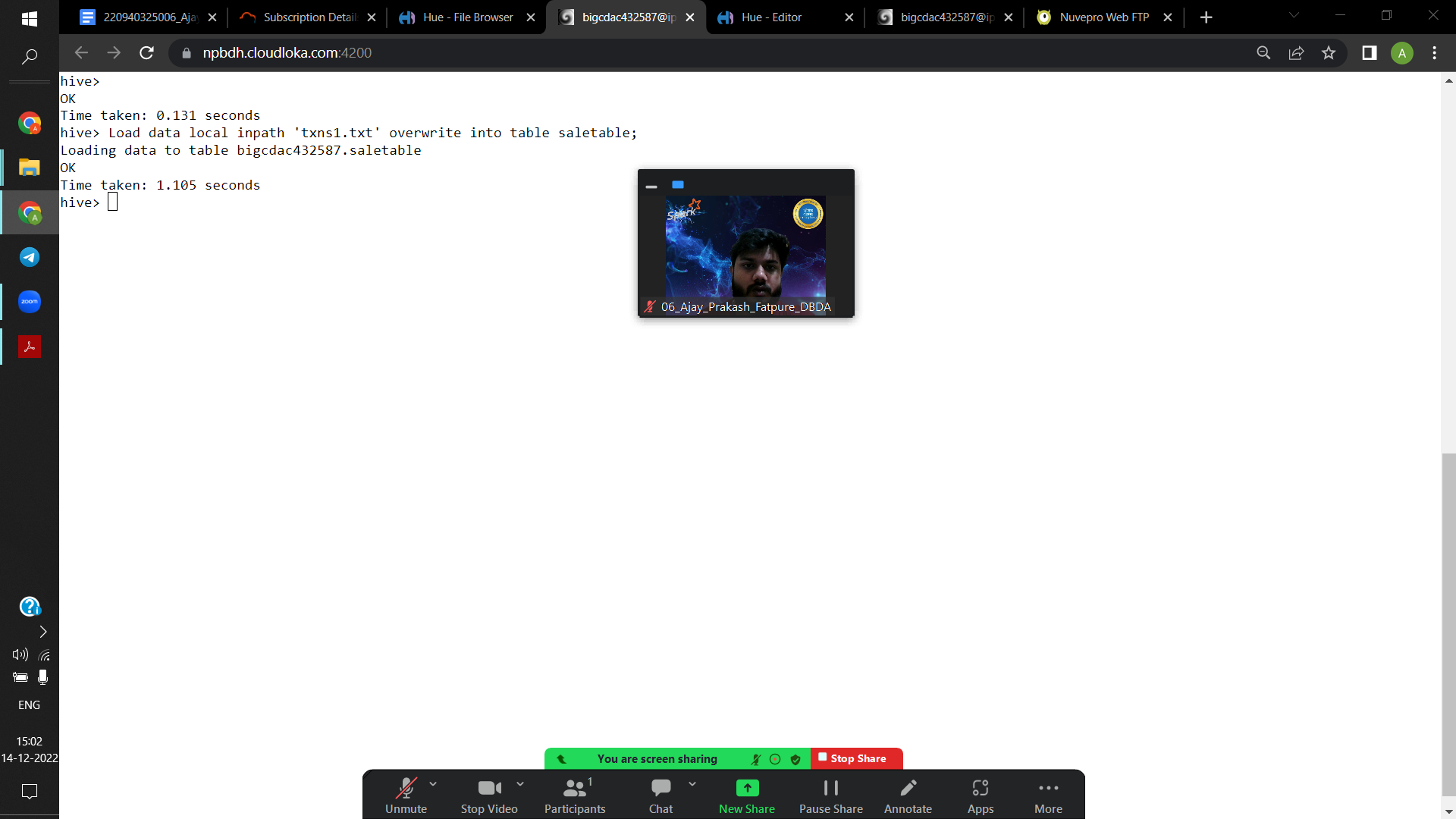
**Stored as textfile;**



**Desc formatted saletable;**



**Load data local inpath’txns1.txt’ overwrite into table saletable;**



**Select product , count(custid) as total from saletable group by product order by total desc limit 10;**



3) Write a program to create partiioned table on category

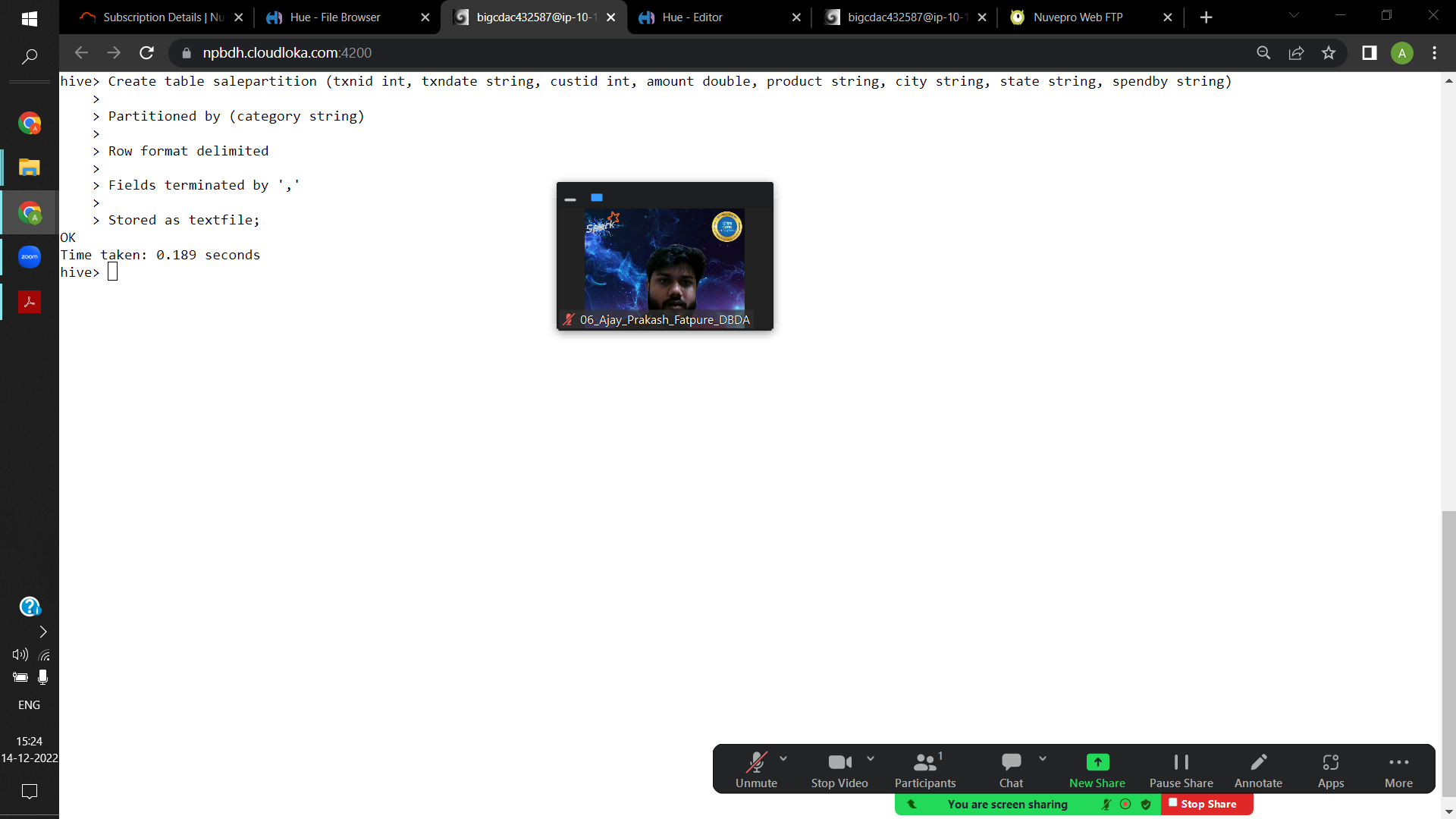
**Create table salepartition (txnid int, txndate string, custid int, amount double, product string, city string, state string, spendby string)**

**Partitioned by (category string)**

**Row format delimited**

**Fields terminated by ‘,’**

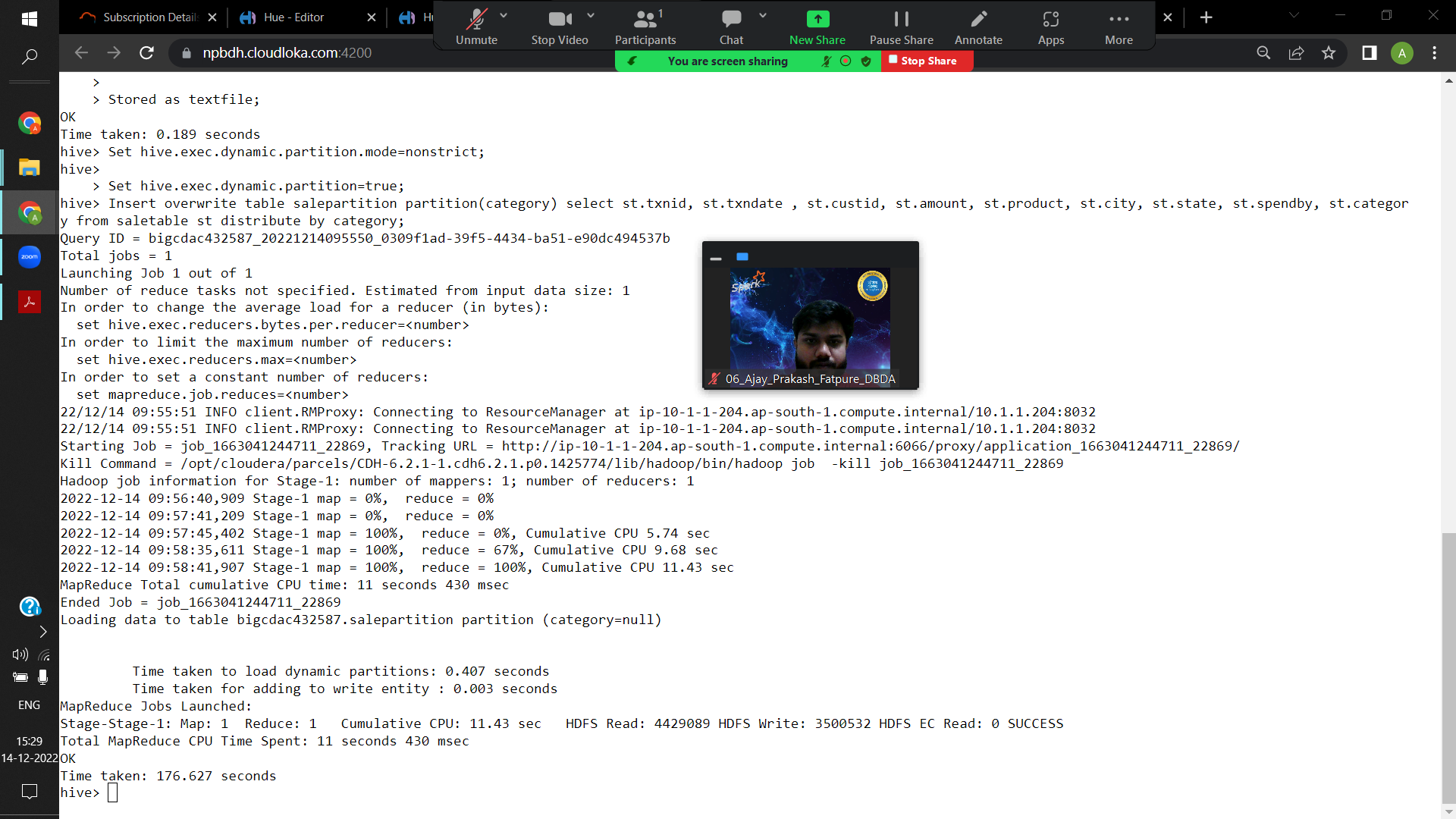
**Stored as textfile;**

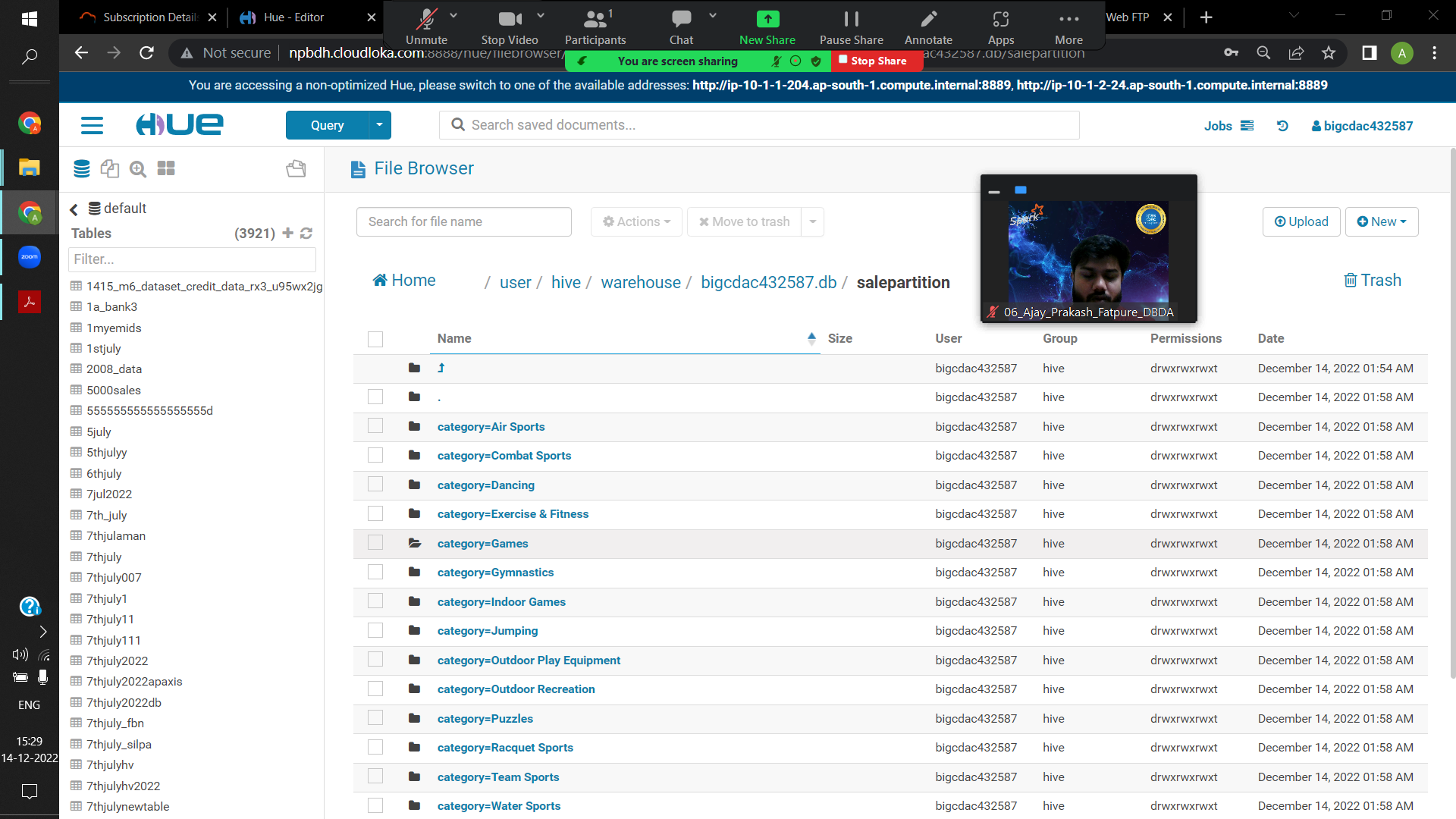


**Set hive.exec.dynamic.partition.mode=nonstrict;**

**Set hive.exec.dynamic.partition=true;**

**Insert overwrite table salepartition partition(category) select st.txnid, st.txndate , st.custid, st.amount, st.product, st.city, st.state, st.spendby, st.category from saletable st distribute by category;**





QUESTION 3 [15 marks]

PySpark

Please find the AIRLINES data set

Year

Quarter

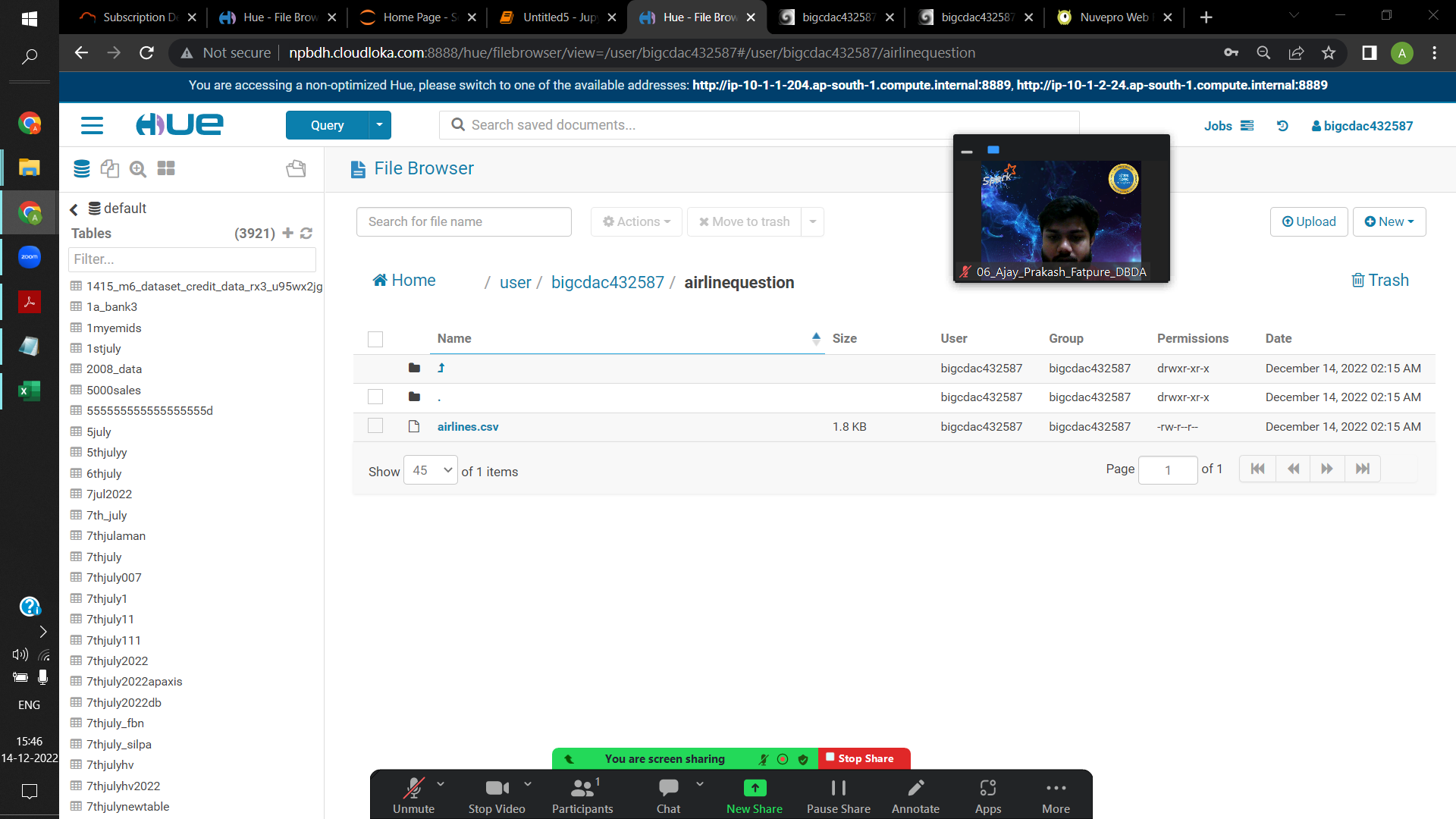
Average revenue per seat

Total number of booked seats

**Hadoop fs -mkdir airlinequestion**

**Hadoop fs -put airlines.csv airlinequestion/**





**from pyspark.context import SparkContext**

**from pyspark.sql.session import SparkSession**

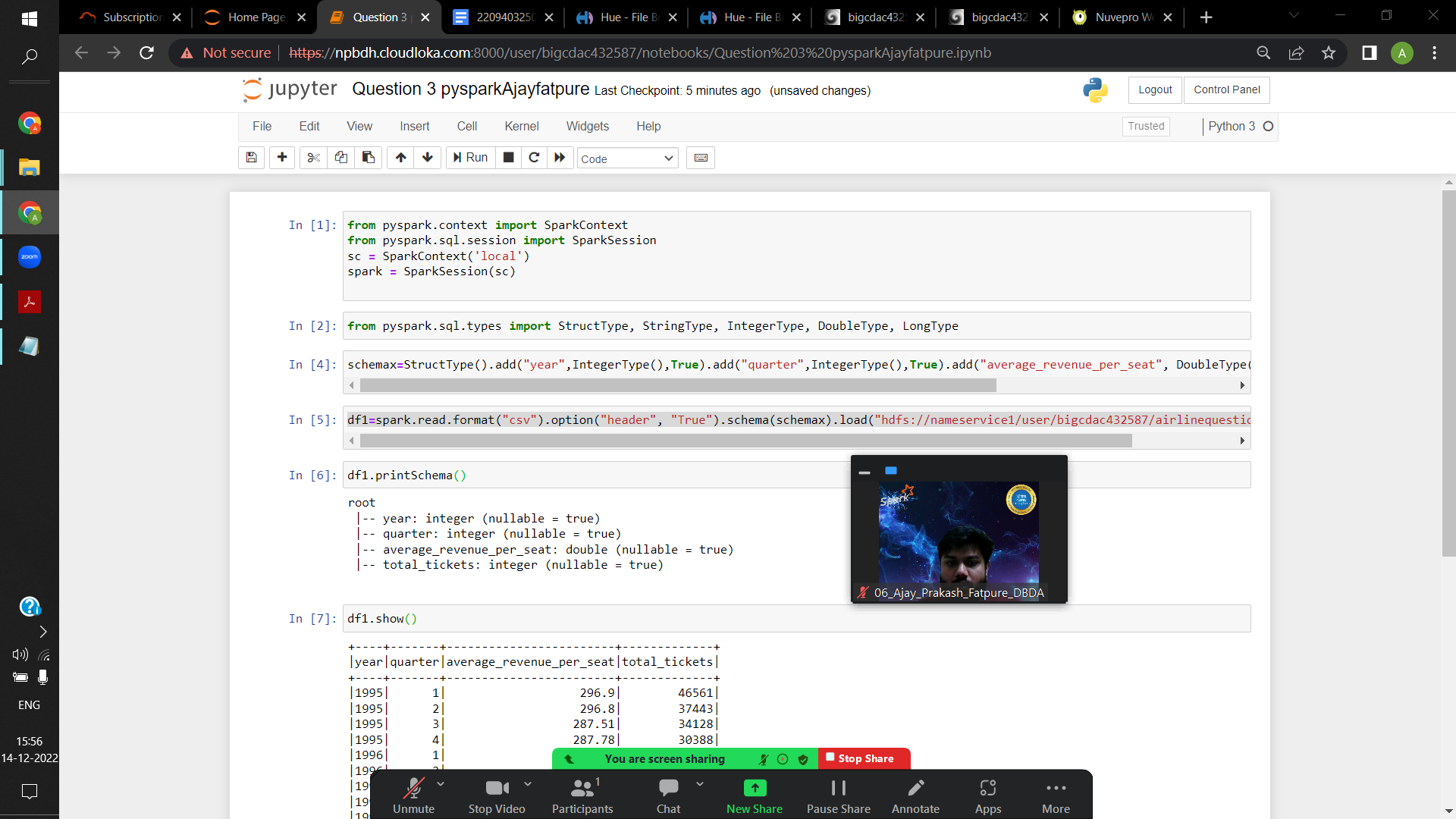
**sc = SparkContext('local')**

**spark = SparkSession(sc)**

**from pyspark.sql.types import StructType, StringType, IntegerType, DoubleType, LongType**

**schemax=StructType().add("year", IntegerType(),True).add("quarter", IntegerType(),True).add("Average\_revenue\_per\_seat", DoubleType(),True).add("total\_tickets", IntegerType(),True)**

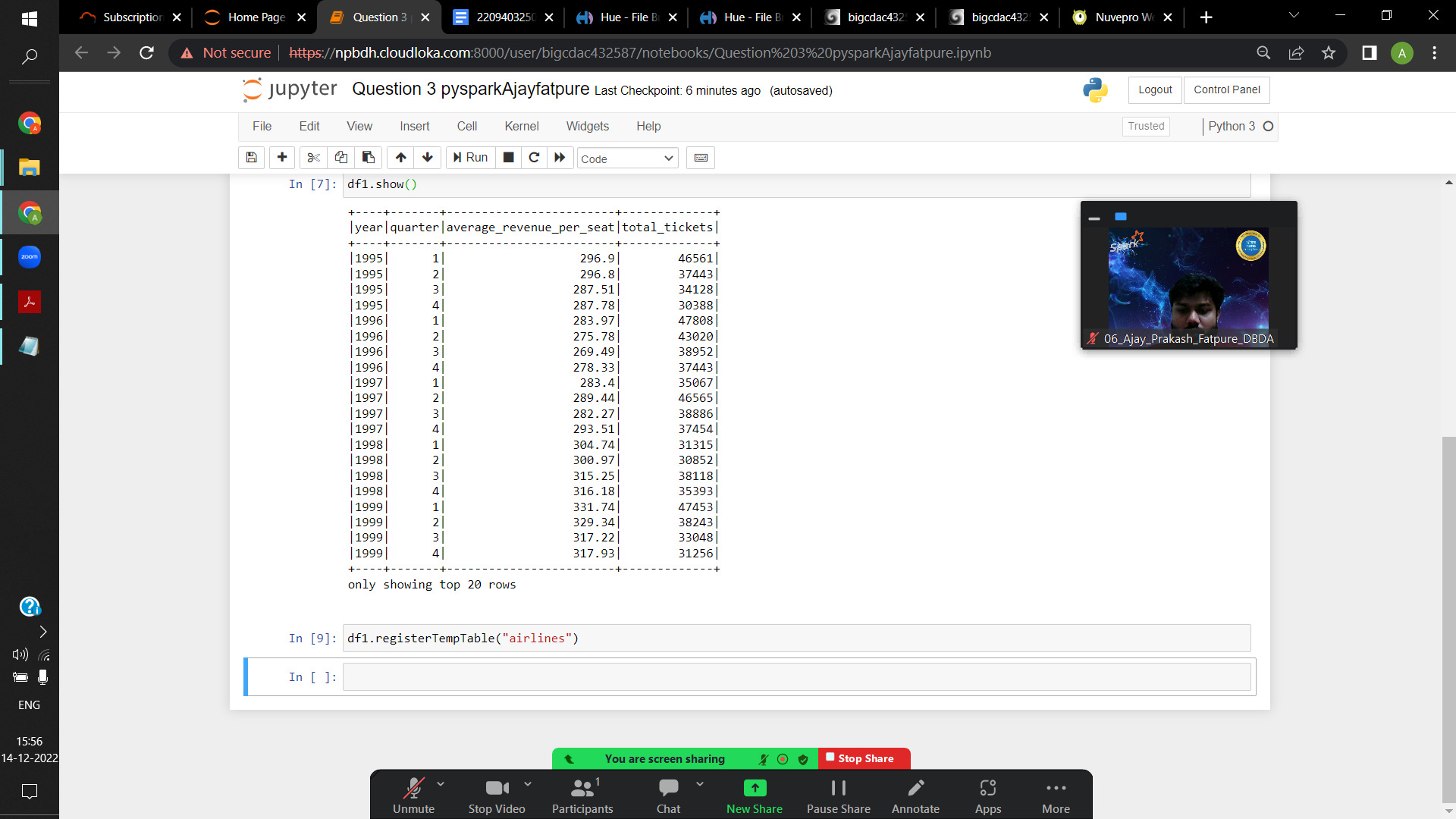
**df1=spark.read.format("csv").option("header", "True").schema(schemax).load("hdfs://nameservice1/user/bigcdac432587/airlinequestion/airlines.csv")**



**df1.printSchema()**

**df1.show()**

**df1.registerTempTable(“airlines”)**

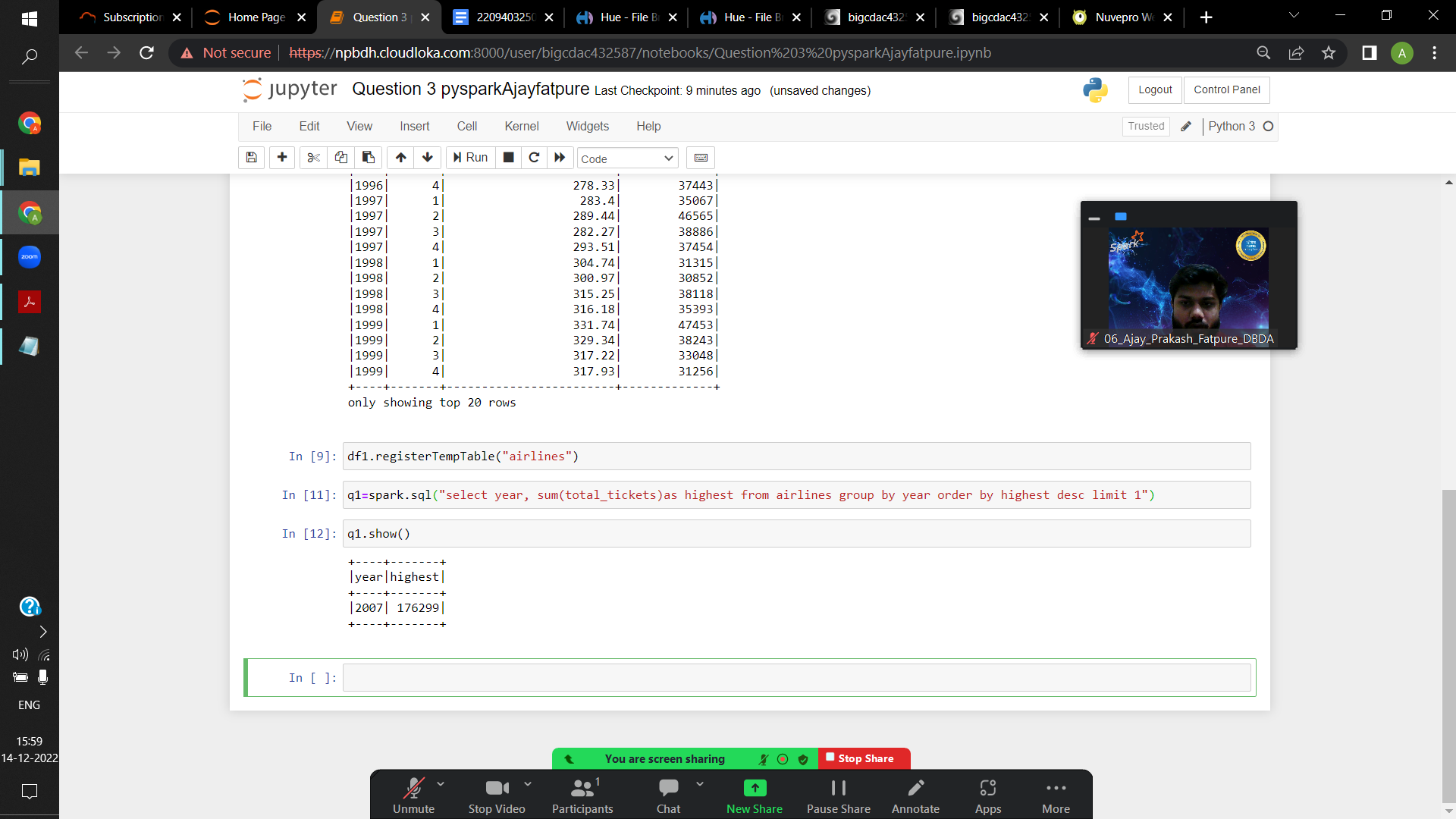


1) What was the highest number of people travelled in which

year?

**q1=spark.sql("select year, sum(total\_tickets)as highest from airlines group by year order by highest desc limit 1")**

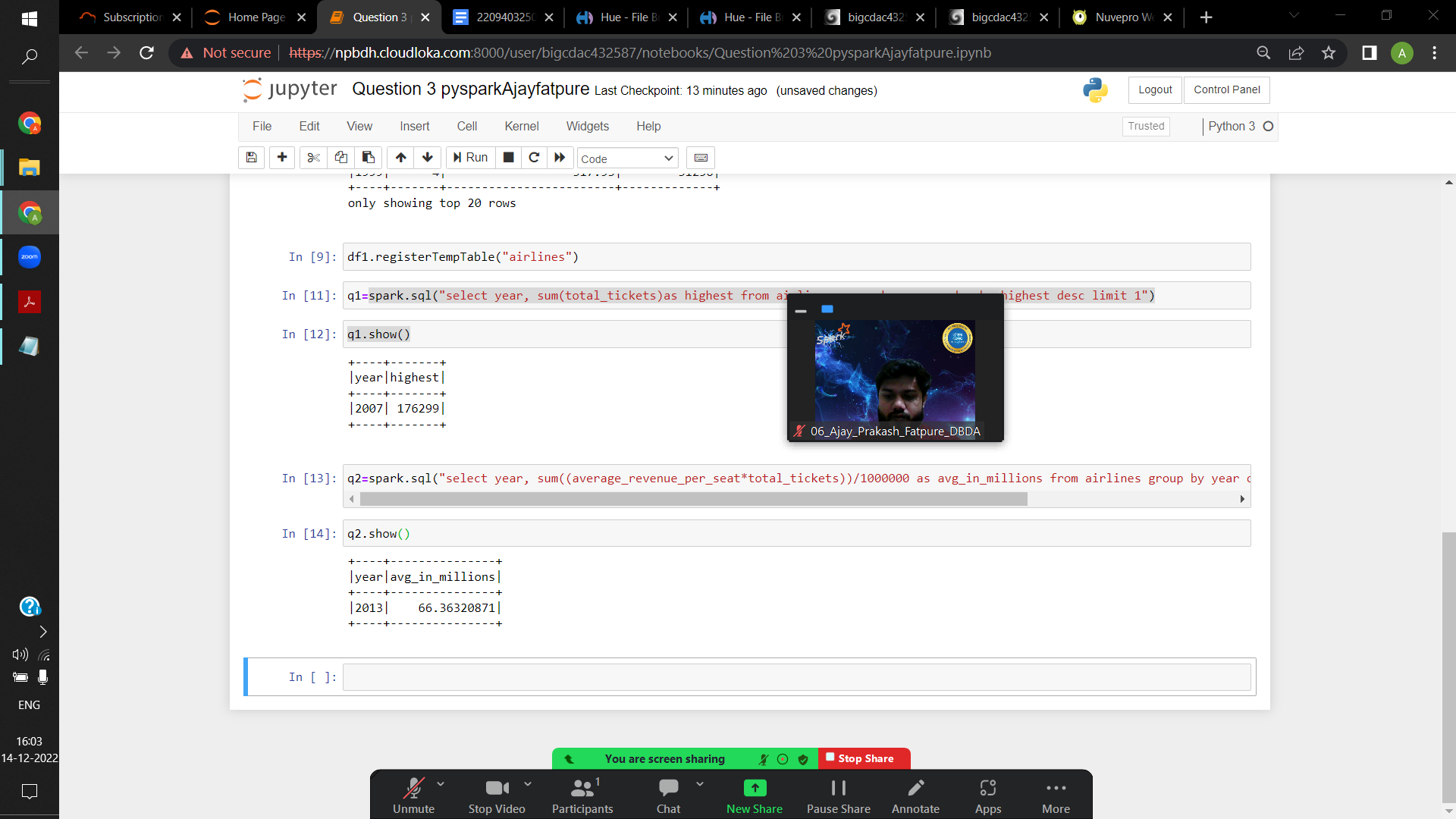
**q1.show()**



2) Identifying the highest revenue generation for which year

**q2=spark.sql("select year, sum((average\_revenue\_per\_seat\*total\_tickets))/1000000 as avg\_in\_millions from airlines group by year order by avg\_in\_millions desc limit 1 ")**

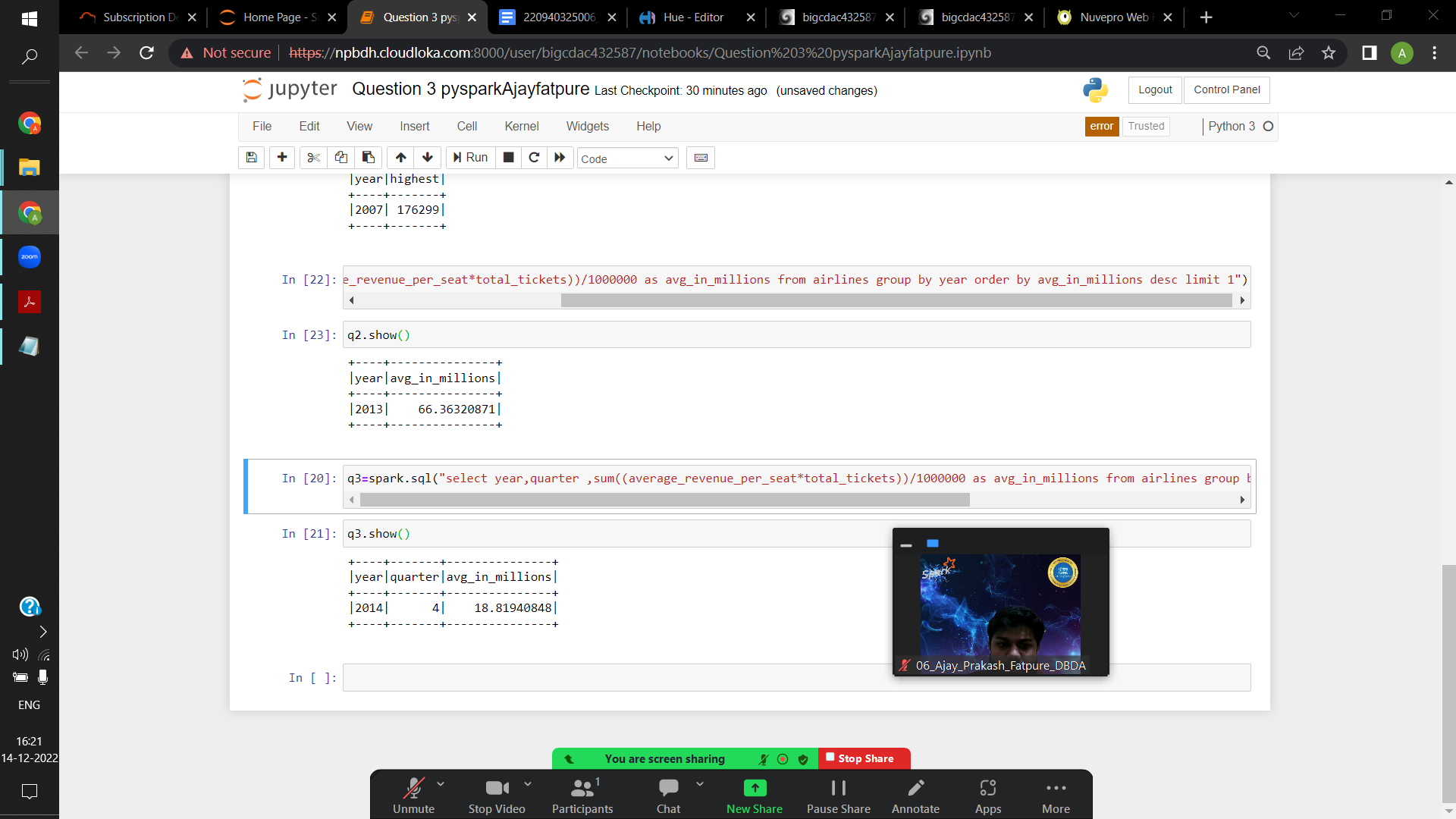
**q2.show()**



3) Identifying the highest revenue generation for which year and quarter (Common group)

**q3=spark.sql("select year,quarter ,sum((average\_revenue\_per\_seat\*total\_tickets))/1000000 as avg\_in\_millions from airlines group by year,quarter order by avg\_in\_millions desc limit 1 ")**

**q3.show()**



**Q1.**

**MapReduce**

**Problem Statement [10 marks]**

Here, we have chosen the stock market dataset on which we have performed map-reduce operations. Following is the structure of the data. Kindly Find the solutions to the questions below.

Data Structure

1. Exchange Name

2 Stock symbol

3. Transaction date

4. Opening price of the stock

5. Intra day high price of the stock

6. Intra day low price of the stock

7. Closing price of the stock

8. Total Volume of the stock on the particular day

9. Adjustment Closing price of the stock

Field Separator – comma

import java.io.\*;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.DoubleWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.conf.\*;

import org.apache.hadoop.fs.\*;

import org.apache.hadoop.mapreduce.lib.input.\*;

import org.apache.hadoop.mapreduce.lib.output.\*;

public class AllTimeHigh {

public static class MapClass extends Mapper<LongWritable,Text,Text,DoubleWritable>

{

private Text stock\_id = new Text();

private DoubleWritable High = new DoubleWritable();

public void map(LongWritable key, Text value, Context context)

{

try{

String[] str = value.toString().split(",");

double high = Double.parseDouble(str[4]);

stock\_id.set(str[1]);

High.set(high);

context.write(stock\_id, High);

}

catch(Exception e)

{

System.out.println(e.getMessage());

}

}

}

public static class ReduceClass extends Reducer<Text,DoubleWritable,Text,DoubleWritable>

{

private DoubleWritable result = new DoubleWritable();

public void reduce(Text key, Iterable<DoubleWritable> values,Context context) throws IOException, InterruptedException {

double maxValue=0;

double tempval=0;

for (DoubleWritable value : values) {

tempval = value.get();

if (tempval > maxValue) {

maxValue = tempval;

}

}

result.set(maxValue);

context.write(key, result);

}

}

public static void main(String[] args) throws Exception {

Configuration conf = new Configuration();

//conf.set("name", "value")

//conf.set("mapreduce.input.fileinputformat.split.minsize", "134217728");

Job job = Job.getInstance(conf, "Highest Price for each stock");

job.setJarByClass(AllTimeHigh.class);

job.setMapperClass(MapClass.class);

//job.setCombinerClass(ReduceClass.class);

job.setReducerClass(ReduceClass.class);

job.setNumReduceTasks(1);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(DoubleWritable.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}