Q1

package exame;

import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class "classname" {Stockmarket

   public static class "mapperclassname" extends Mapper<LongWritable, Text, Text, IntWritable>{

public void map(LongWritable key, text value, context context)

{

try { string [] str = value.to.string().split(",");

long vol = Long.parselLong(str[7]);

context.write(new Text(str[1]),)new LongWritable()vol));

}

catch(Exception e)

{

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

}

    }

}

 }

  public static class "reducerclassname"  extends Reducer<Text,IntWritable,Text,IntWritable> {

  private Intwritable result = new Intwritable();

  public void reduce (textkey,iterable<Intwritabe> values,context)

  }

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

    Configuration conf = new Configuration();

    Job job = Job.getInstance(conf, " Stokmarket");

    job.setJarByClass(Stokmarket.class);

    job.setMapperClass(mapclass.class);

    job.setReducerClass(ReduceClass.class);

    job.setNumReduceTasks(1);

    job.setMapOutputKeyClass(text.class);

    job.setMapOutputValueClass(Intwritable.class);

    job.setOutputKeyClass(text.class);

    job.setOutputValueClass(Intwritable.class);

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

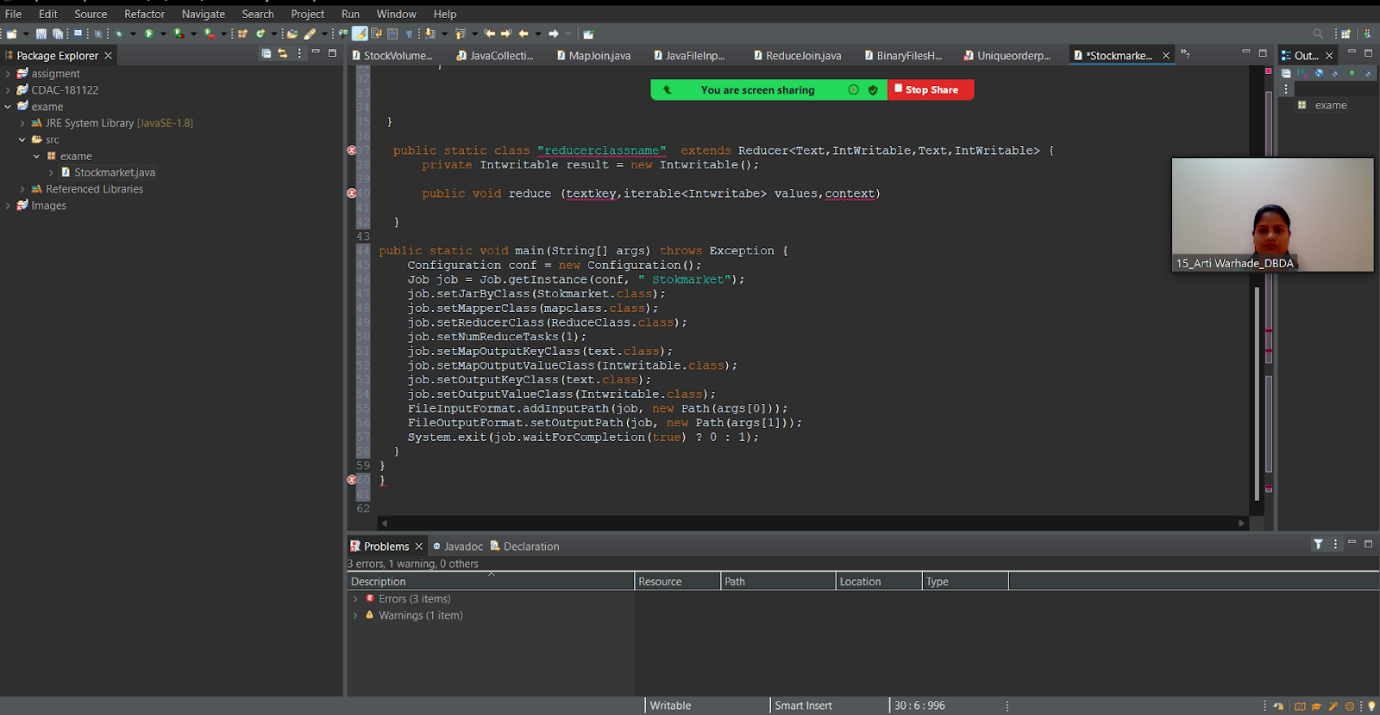
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

  }

}

}



Q2

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

Please find the sales data set.

create table customer( customer\_id string,

firstname string,

lastname string,

age int,

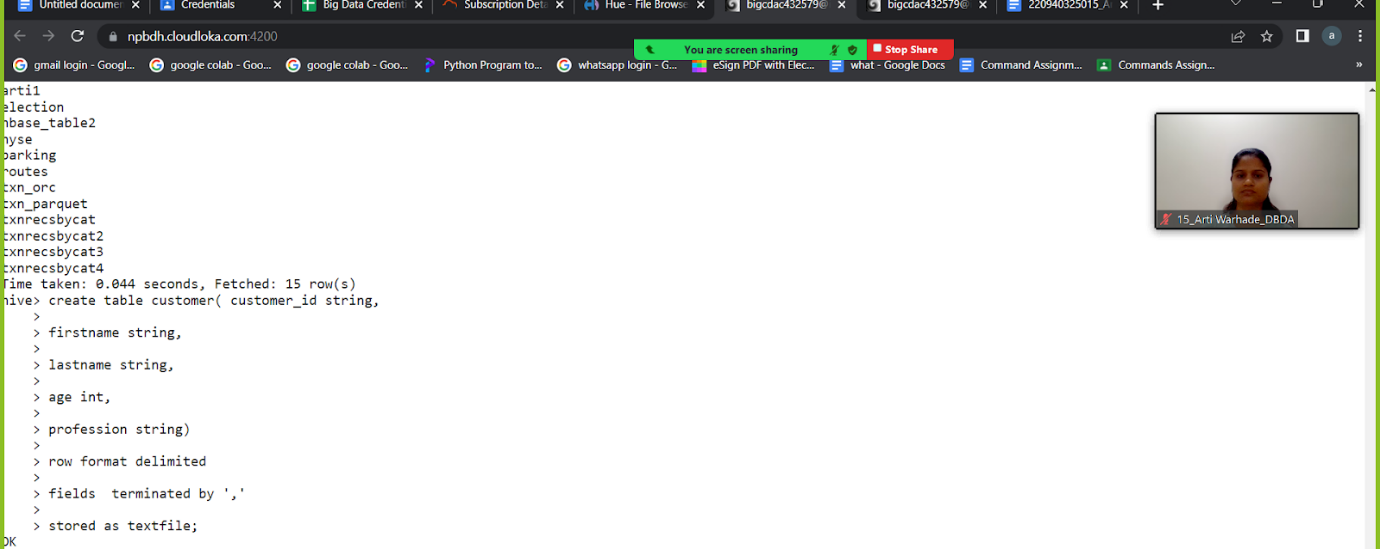
profession string)

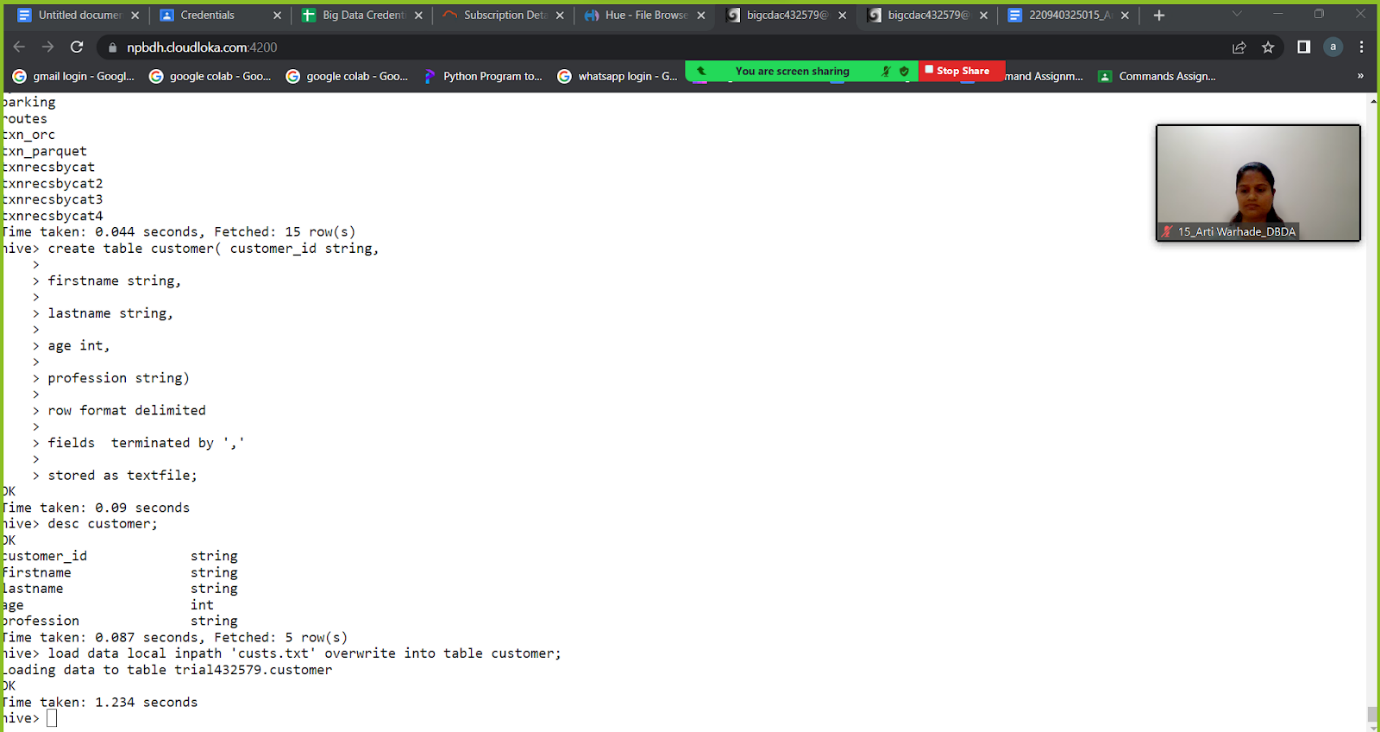
row format delimited

fields  terminated by ','

stored as textfile;







>>select count(cust\_id) from customer\_table group by profession

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

Ans:-

Create table txn1

(txn id int,

txn date string,

cust \_id int,

amount double,

category  string,

product string,

city string,

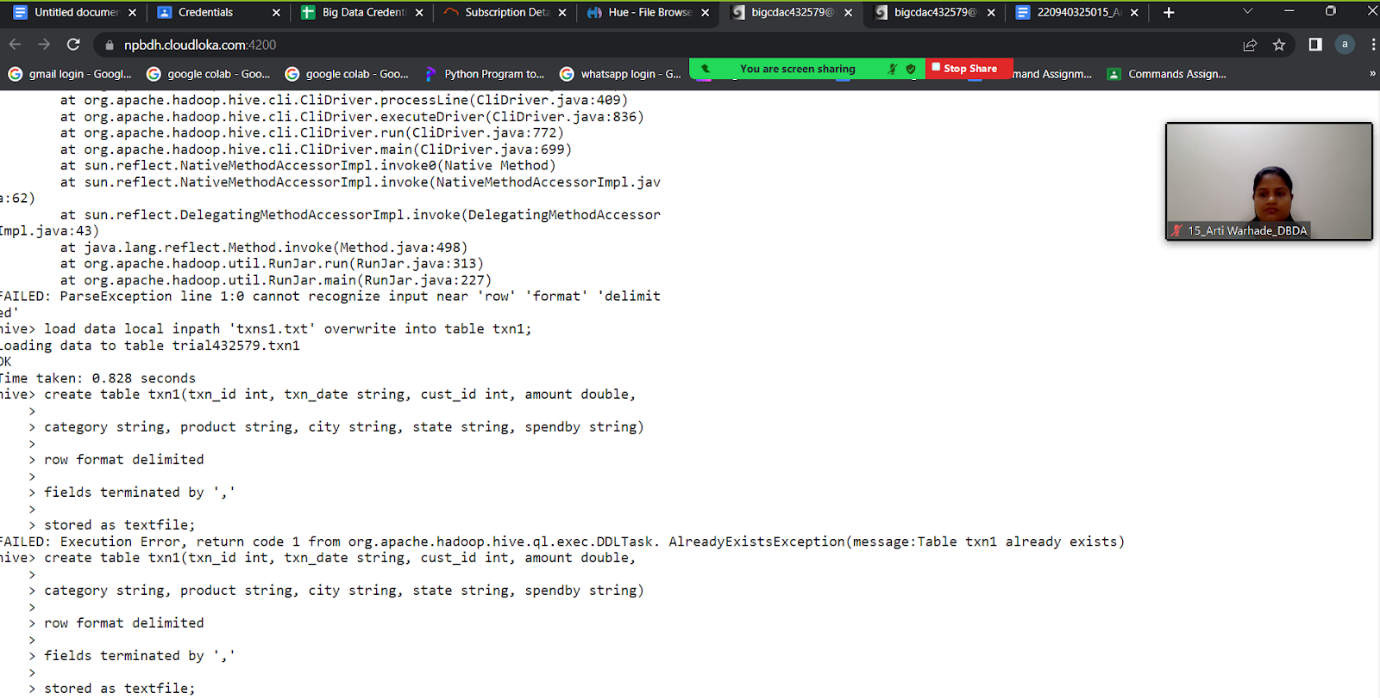
state string,

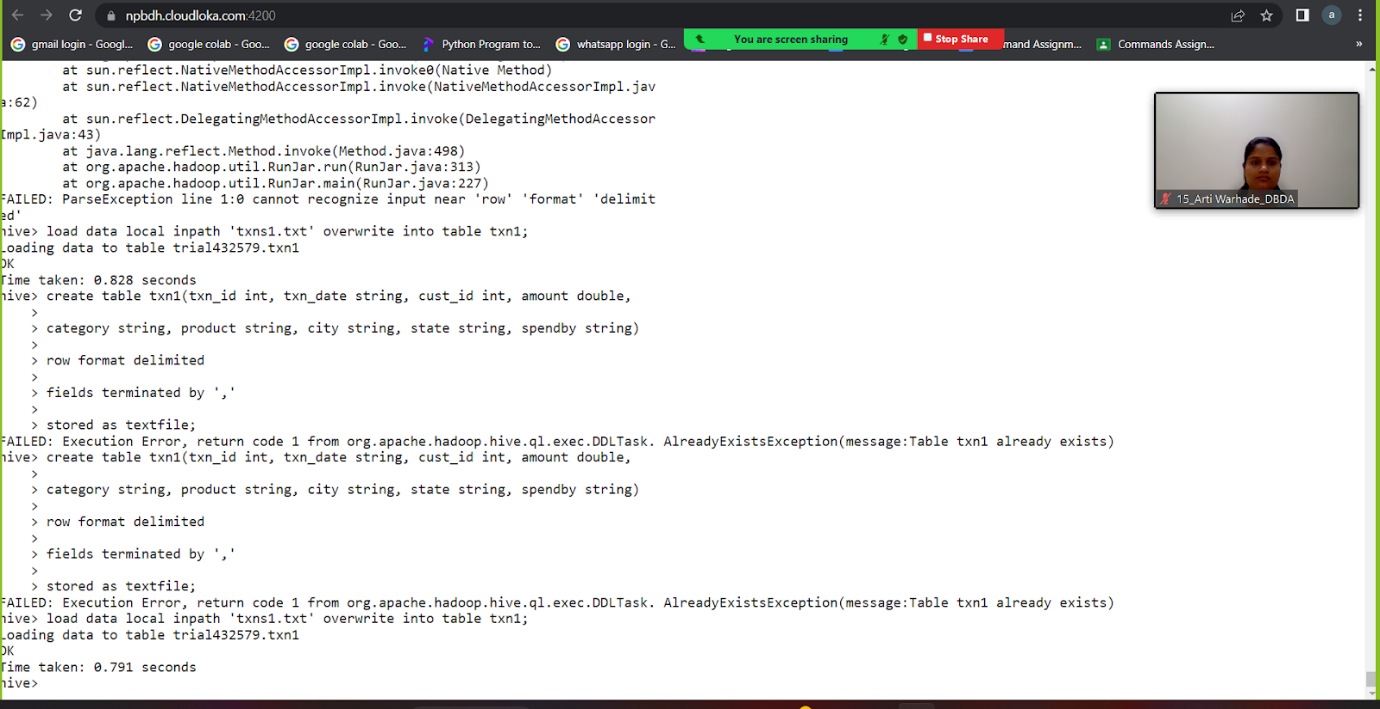
spendby string)

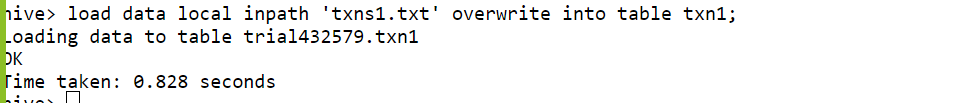
Row format delimited

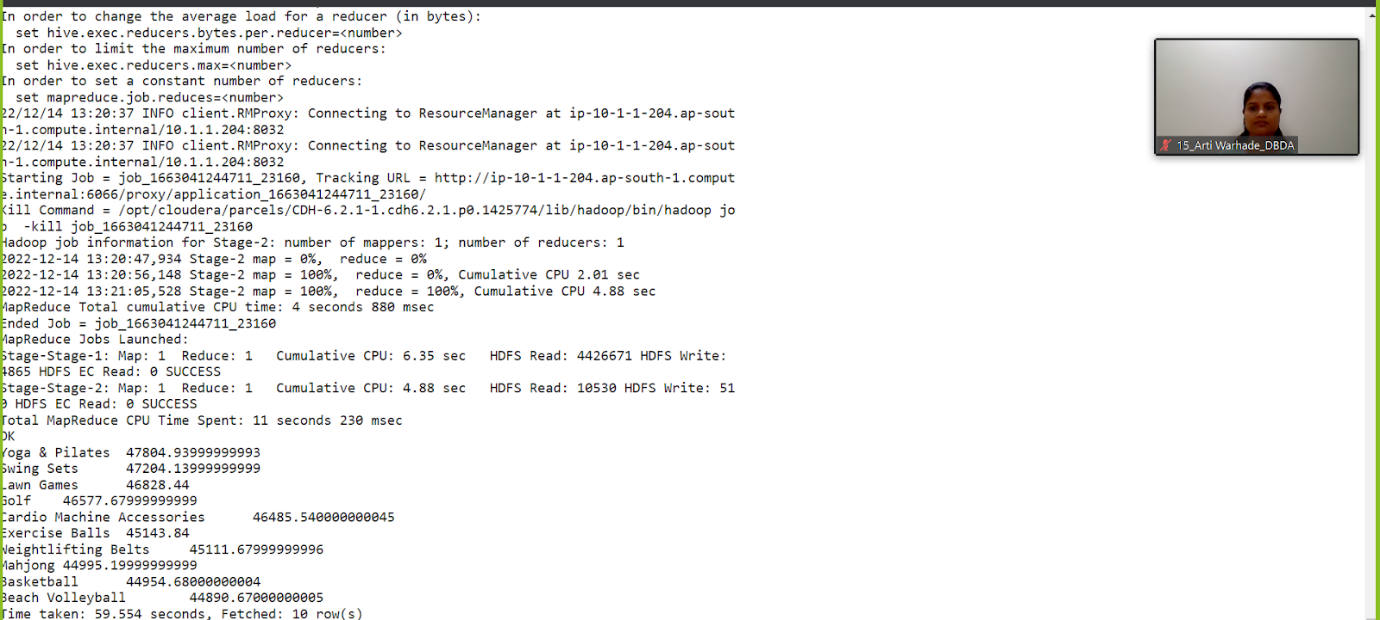
Fields terminated by  ','

Stored as textfile;









3) Write a program to create partitioned table on category

ans:-

create table txnByPartition(

txn\_id int,

txn\_date string,

cust\_id int,

amount double,

product string,

city string,

state string,

spendby string)

partitioned by (category string)

row format delimited

fields terminated by ','

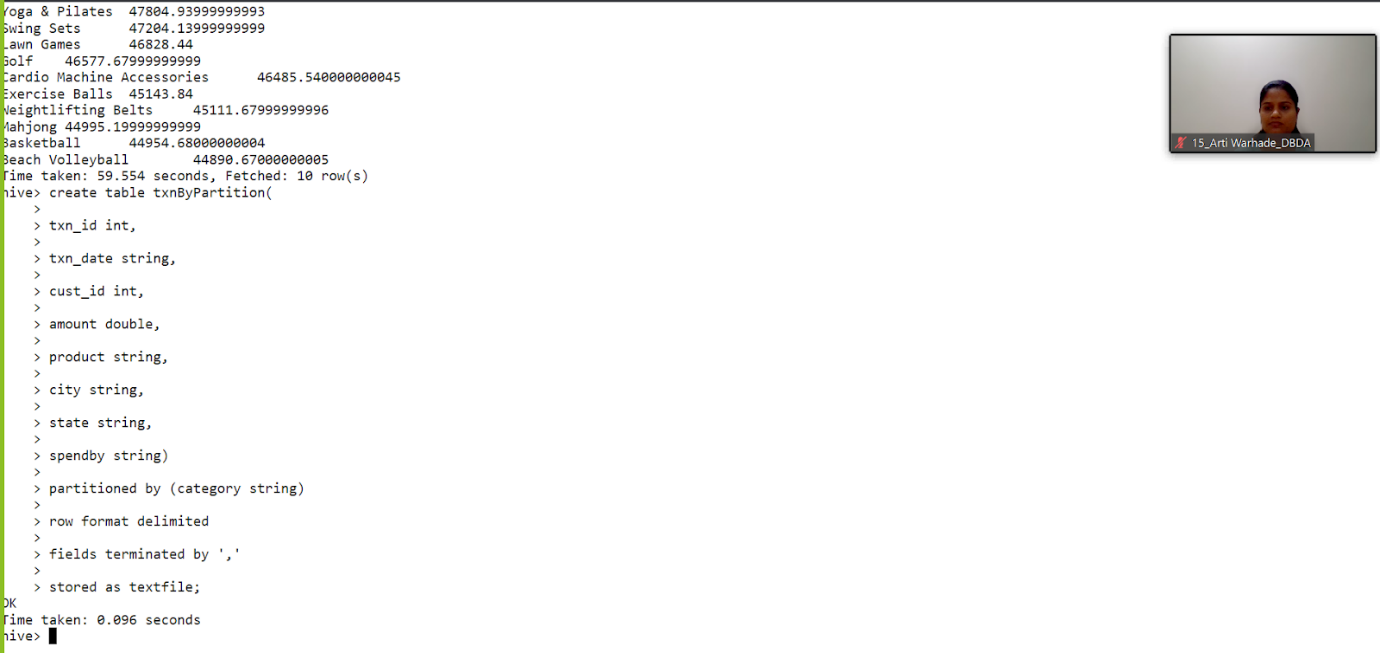
stored as textfile;

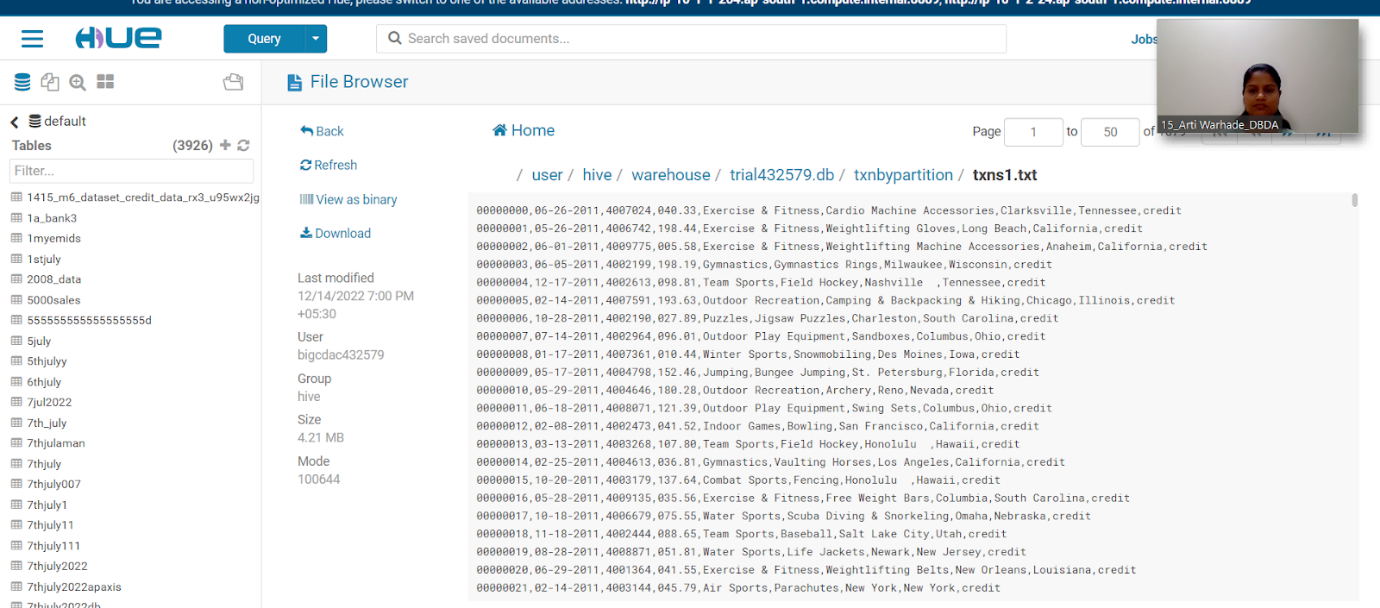
INSERT OVERWRITE TABLE txnByPartition PARTITION(category)

select txn.txn\_id, txn.txn\_date,txn.cust\_id, txn.amount,txn.product,

txn.city,txn.state, txn.spendby,

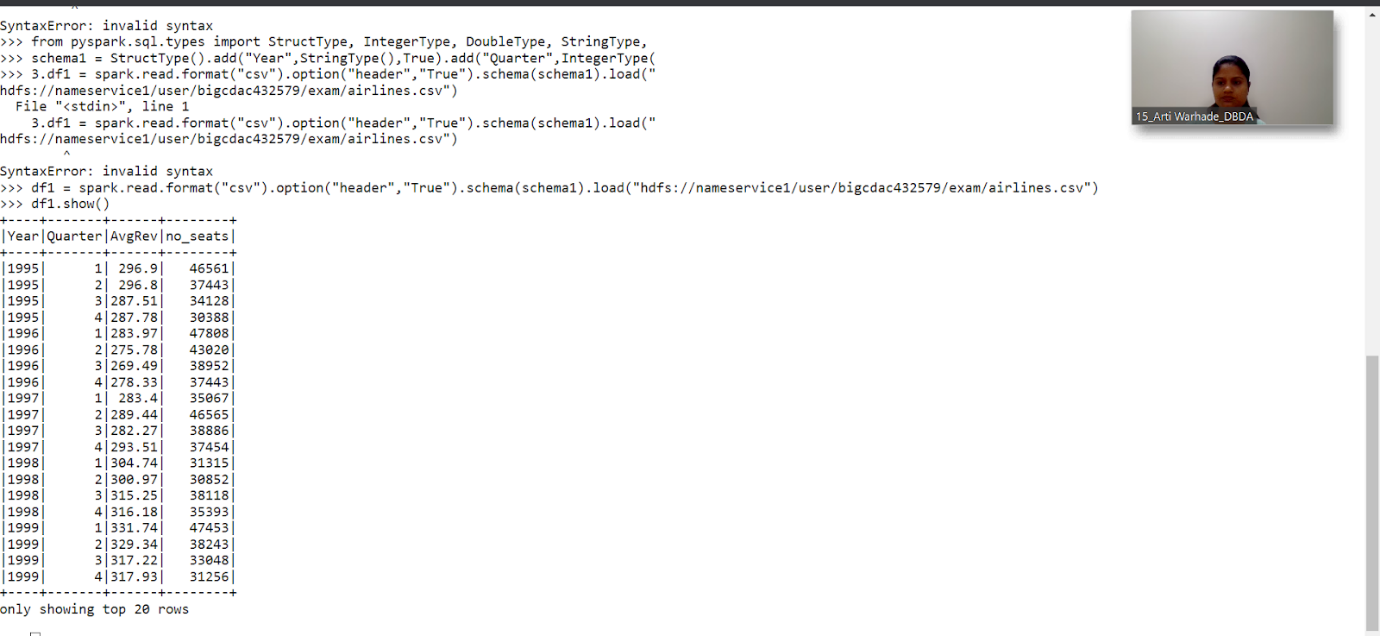
txn.category from txn1 txn DISTRIBUTE By category;

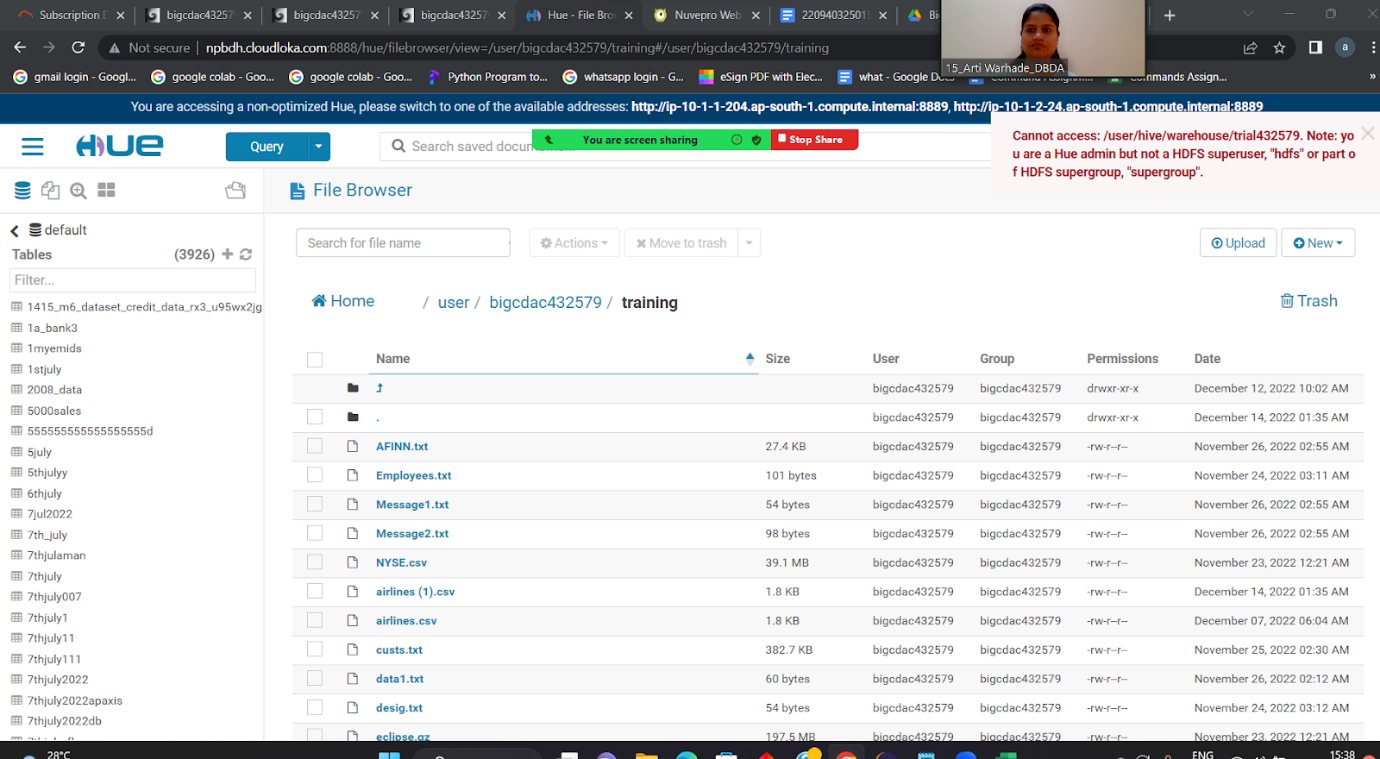
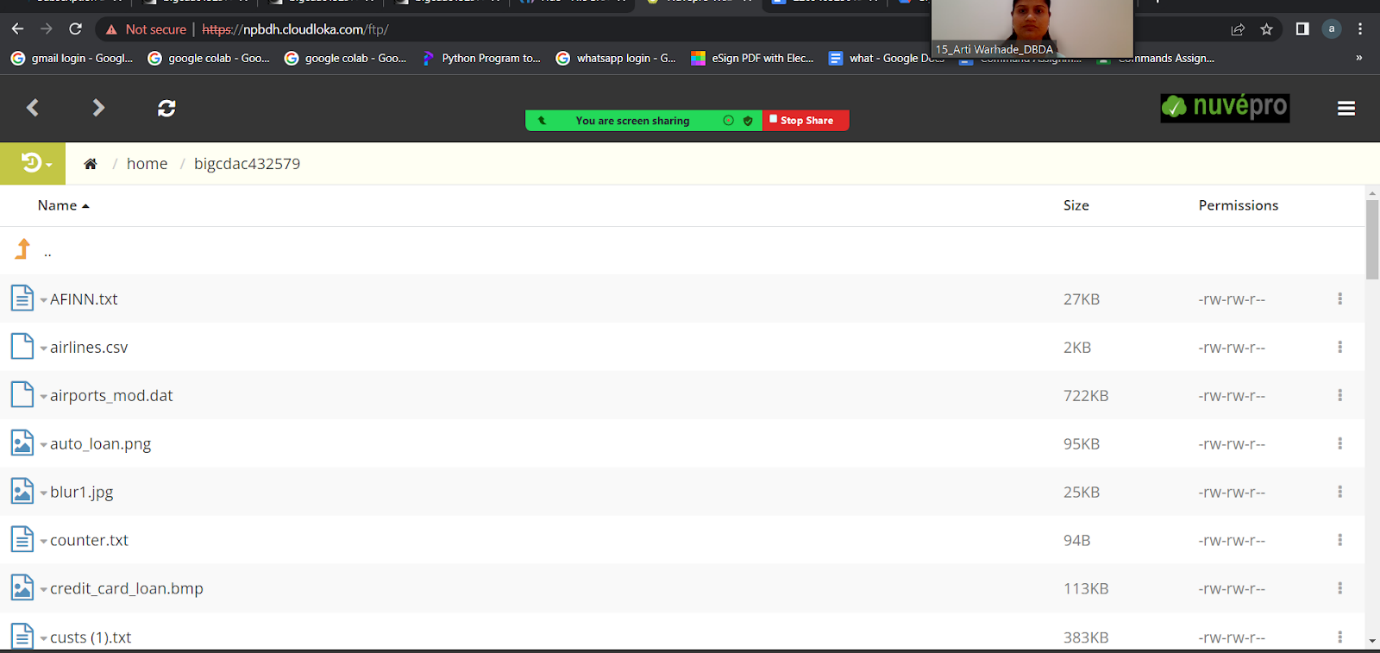




Q3

1. from pyspark.sql.types import StructType, IntegerType, DoubleType, StringType, LongType
2. schema1 = StructType().add("Year",StringType(),True).add("Quarter",IntegerType(),True).add("AvgRev",DoubleType(),True).add("no\_seats",LongType(),True)
3. df1 = spark.read.format("csv").option("header","True").schema(schema1).load("hdfs://nameservice1/user/bigcdac432537/exam/airlines.csv")
4. df1.show()
5. df1.registerTempTable("airline")

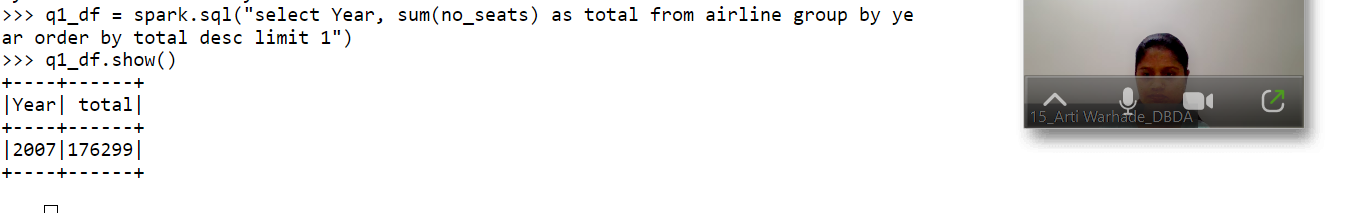




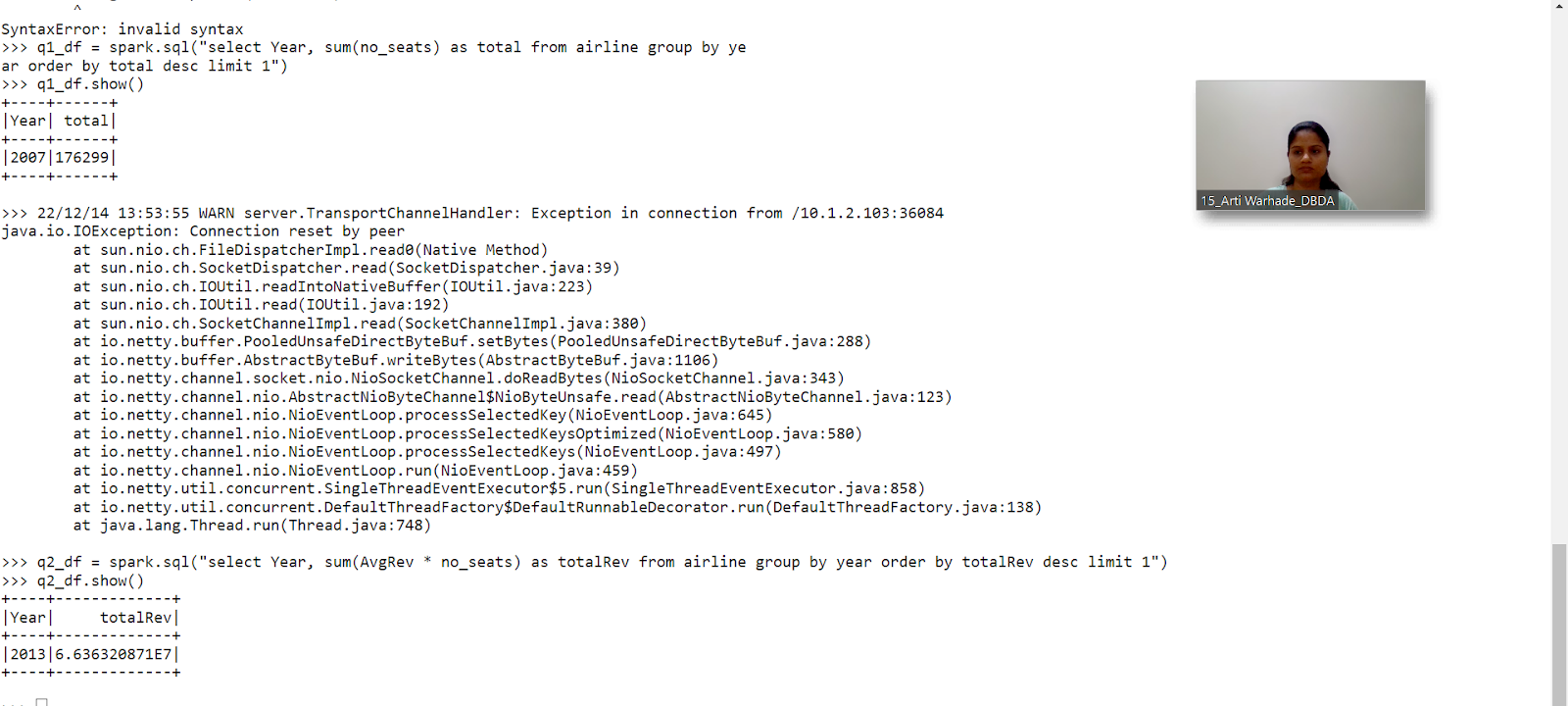
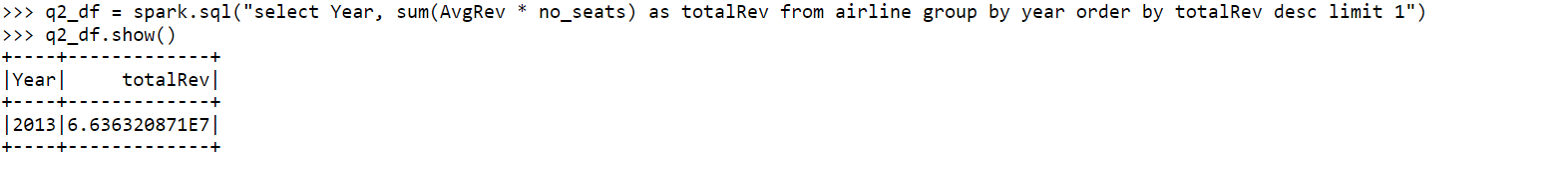
1) What was the highest number of people travelled in which

year?

Ans:-



highest number of passengers 176299 in 2017.

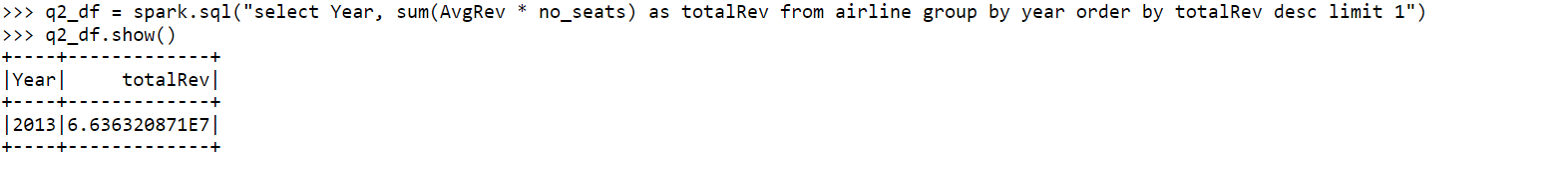


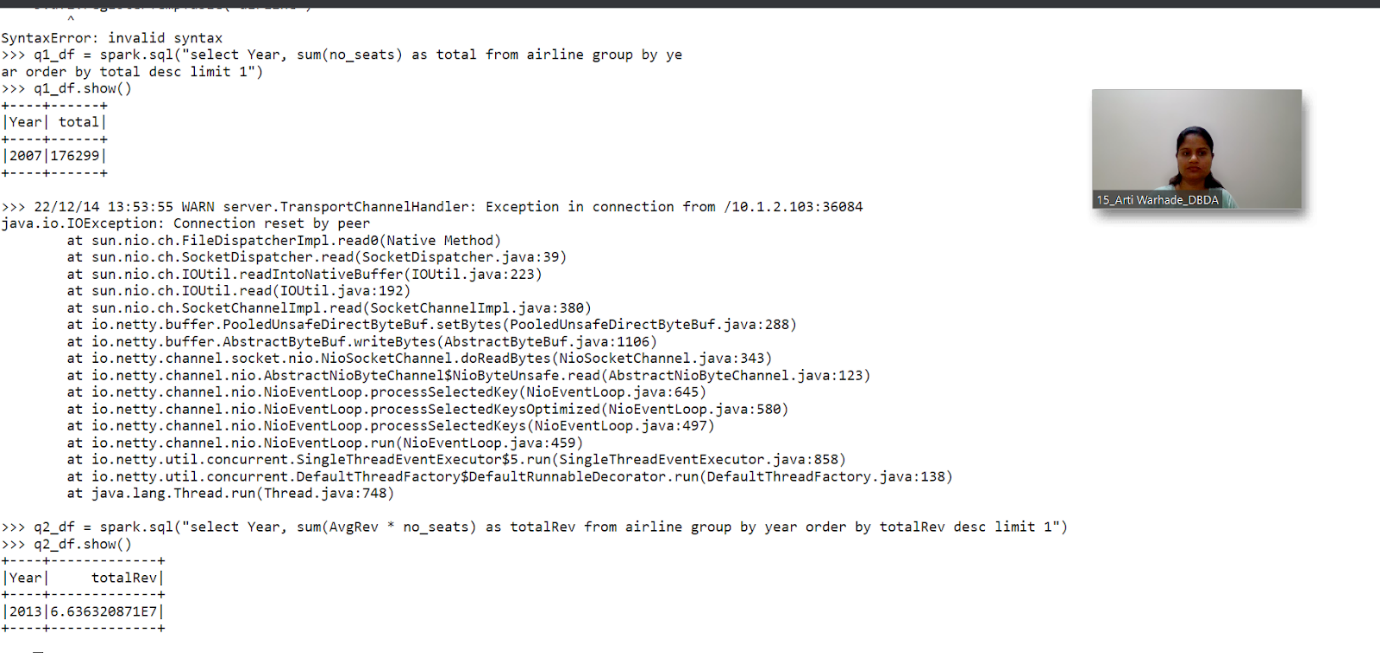
2) Identifying the highest revenue generation for which year

ans:

q2\_df = spark.sql("select Year, sum(AvgRev \* no\_seats) as totalRev from airline group by year order by totalRev desc limit 1")

q2\_show()





highest revenue 6.636320871E7  was generated in the year 2013

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

group

ans:

q3\_df = spark.sql("select Year, Quarter, sum(AvgRev \* no\_seats) as totalRev from airline group by Year, Quarter order by totalRev desc limit 5")

q3\_df.show()

