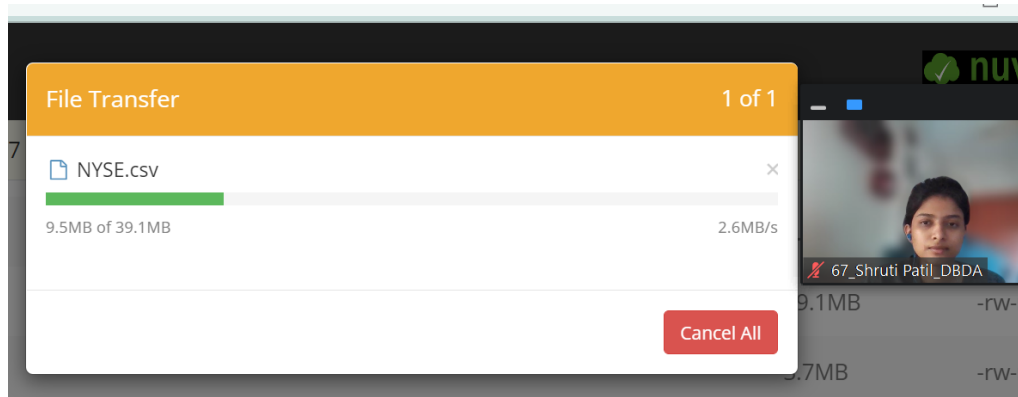
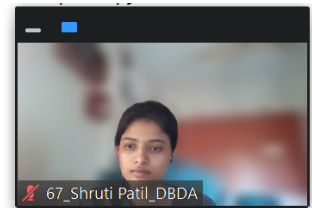


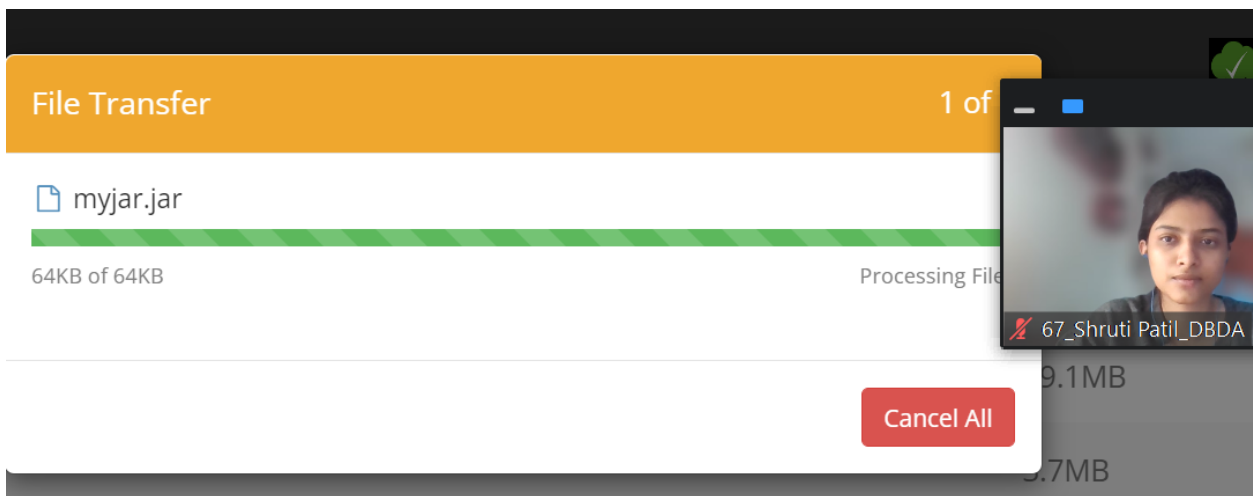
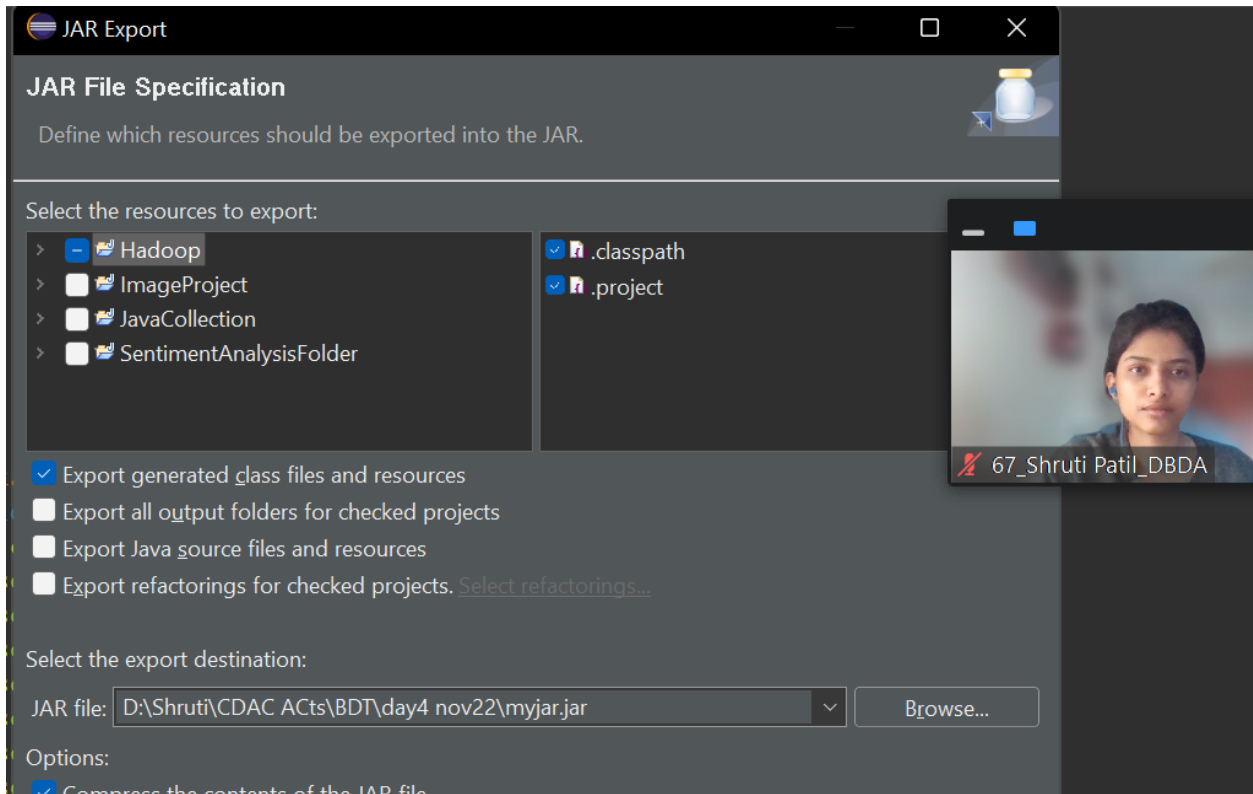
BDT Module Exam Answer Sheet

Q1. Map Reduce



```
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -mkdir exam
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -ls exam
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -put NYSE.csv training/exam
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -ls exam
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -put NYSE.csv training/exam/
put: `training/exam': File exists
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -ls exam
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -put NYSE.csv exam
[bigcdac432527@ip-10-1-1-204 ~]$ hadoop fs -ls exam
Found 1 items
-rw-r--r--  3 bigcdac432527 bigcdac432527  40990862 2022-12-14 09:00 exam/NYSE.csv
[bigcdac432527@ip-10-1-1-204 ~]$
```





```
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 Exam67 {
```

```
    public static class MapClass extends Mapper<LongWritable,Text,Text,DoubleWritable>
    {
        public void map(LongWritable key, Text value, Context context)
        {
            try{
                String[] str = value.toString().split(",");
                double high = Double.parseDouble(str[4]);
                context.write(new Text(str[1]),new DoubleWritable(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 max = 0.00;

            for (DoubleWritable val : values)
            {
                if (val.get() > max) {
                    max = val.get();
                }
            }

            result.set(max);
            context.write(key, result);
            //context.write(key, new LongWritable(sum));
        }
    }
}
```

```

    }
}
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    conf.set("mapreduce.output.textoutputformat.separator", ",");
    //conf.set("name", "value")
    conf.set("mapreduce.input.fileinputformat.split.maxsize", "28311552");
    Job job = Job.getInstance(conf, "All Time High Price for each stock");
    job.setJarByClass(Exam67.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);
}
}



```


```





hadoop jar myjar.jar Exam67 /user/bigcdac432527/exam/NYSE.csv
/user/bigcdac432527/exam/Q1output

```

Hive



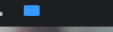
 You are screen sharing

  1  

Participants Chat



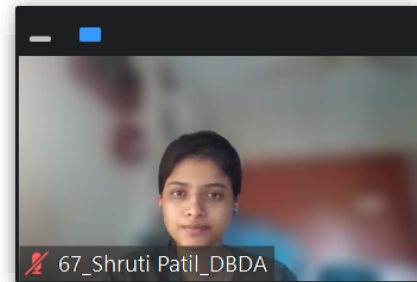
67_Shruti Patil_DBDA

A screenshot of a video call window. The window has a dark title bar with a white close button and a blue maximize button. The video feed shows a woman with dark hair and a bindi, looking directly at the camera. The background is blurred, showing some indoor elements.

```

hive> show tables;
OK
cust_exam
Time taken: 0.269 seconds, Fetched: 1 row(s)
hive> desc cust_exam;
OK
custid                bigint
firstname              string
lastname               string
age                    int
profession              string
Time taken: 0.14 seconds, Fetched: 5 row(s)
hive> select * from cust_exam limit 5;
OK
Time taken: 0.543 seconds
hive> load data local inpath 'custs.txt' overwrite into table cust_exam;
Loading data to table exam67.cust_exam
OK
Time taken: 1.487 seconds
hive> select * from cust_exam limit 5;
OK
4000001 Kristina      Chung    55      Pilot
4000002 Paige      Chen    74      Teacher
4000003 Sherri    Melton   34      Firefighter
4000004 Gretchen   Hill     66      Computer hardware engineer
4000005 Karen     Puckett 74      Lawyer
Time taken: 0.218 seconds, Fetched: 5 row(s)
hive>

```



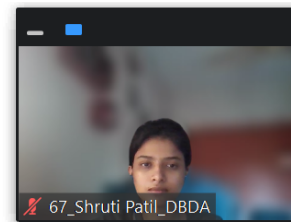
ANS 1:

select profession, count(custid) from cust_exam group by profession;

```

hive> select profession, count(custid) from cust_exam group by profession;
Query ID = bigcdac432527_20221214094127_0647927d-45be-4db8-98fc-7380133975e0
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 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>
22/12/14 09:41:29 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute
22/12/14 09:41:29 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute
Starting Job = job_1663041244711_22786, Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal
_22786/
Kill Command = /opt/cloudera/parcels/CDH-6.2.1-1.cdh6.2.1.p0.1425774/lib/hadoop/bin/hadoop job -kill j

```

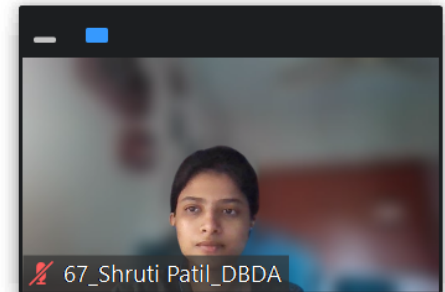


Output:

Total MapReduce CPU Time Spent: 5 seconds 950 msec

OK

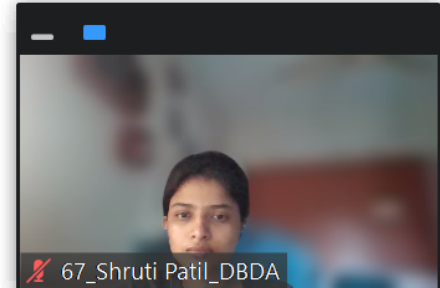
Accountant	199
Actor	202
Agricultural and food scientist	195
Architect	203
Artist	175
Athlete	196
Automotive mechanic	193
Carpenter	181
Chemist	209
Childcare worker	207
Civil engineer	193
Coach	201
Computer hardware engineer	204
Computer software engineer	216
Computer support specialist	222
Dancer	185
Designer	205
Doctor	197
Economist	189
Electrical engineer	192
Electrician	194
Engineering technician	204
Environmental scientist	176
Farmer	201
Financial analyst	198
Firefighter	217
Human resources assistant	212
Judge	196
Lawyer	212
Librarian	218



```

Librarian      218
Loan officer   221
Musician       205
Nurse    192
Pharmacist     213
Photographer   222
Physicist      201
Pilot    211
Police officer 210
Politician     228
Psychologist   194
Real estate agent      191
Recreation and fitness worker    210
Reporter       200
Secretary      200
Social Worker   1
Social worker   212
Statistician    196
Teacher 204
Therapist       187
Veterinarian    208
Writer    101
Time taken: 55.422 seconds, Fetched: 51 row(s)

```



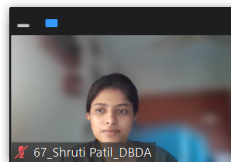
ANS2:

create table sales_exam(txn_id bigint, txn_date string, cust_id bigint, amount double, category string, product string, city string, state string, spendby string)

```

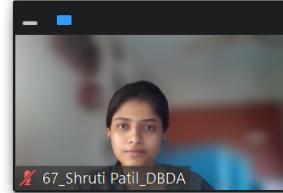
hive> create table sales_exam(txn_id bigint, txn_date string, cust_id bigint, amount double, category string, product string, city string, state string, spendby string)
> row format delimited
> fields terminated by ','
> stored as textfile;
OK
Time taken: 0.103 seconds
hive> load data local inpath 'txns1.txt' overwrite into table sales_exam;
Loading data to table exam67.sales_exam
OK
Time taken: 1.024 seconds
hive> show tables;
OK
cust_exam
sales_exam
Time taken: 0.098 seconds, Fetched: 2 row(s)
hive> desc sales_exam;
OK
txn_id          bigint
txn_date        string
cust_id         bigint
amount          double
category        string
product         string
city            string
state           string
spendby         string
Time taken: 0.167 seconds, Fetched: 9 row(s)

```




```
select product, sum(amount) as sales from sales_exam
group by product order by sales desc limit 10;
```

```
hive> select product, sum(amount) as sales from sales_exam group by product order by sales desc limit 10;
Query ID = bigcdac432527_20221214095112_443caf0a-6f06-48d1-94a3-04565956d603
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 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>
22/12/14 09:51:12 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10
22/12/14 09:51:12 INFO client.RMProxy: Connecting to ResourceManager at ip-10-1-1-204.ap-south-1.compute.internal/10
Starting Job = job_16630411244711_22831 Tracking URL = http://ip-10-1-1-204.ap-south-1.compute.internal:6066/progress
```

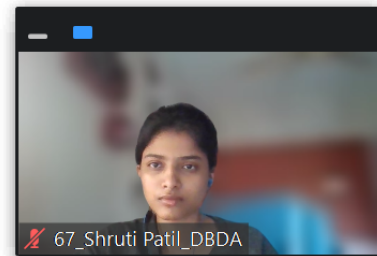


MapReduce Jobs Launched:

```
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 7.48 sec HDFS Read: 4426708 HDFS Write: 10530
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 5.74 sec HDFS Read: 10530 HDFS Write: 10530
Total MapReduce CPU Time Spent: 13 seconds 220 msec
```

OK

```
Yoga & Pilates 47804.939999999993
Swing Sets 47204.139999999999
Lawn Games 46828.44
Golf 46577.679999999999
Cardio Machine Accessories 46485.5400000000045
Exercise Balls 45143.84
Weightlifting Belts 45111.679999999996
Mahjong 44995.199999999999
Basketball 44954.680000000004
Beach Volleyball 44890.670000000005
Time taken: 188.055 seconds, Fetched: 10 row(s)
hive>
```



ANS3:

```
create table sales_exam_part(txn_id bigint, txn_date string, cust_id bigint, amount double,
product string, city string, state string, spendby string)
partitioned by (category string)
row format delimited
fields terminated by ','
stored as textfile;
```

```
hive> create table sales_exam_part(txn_id bigint, txn_date string,
cust_id bigint, amount double, product string, city string, state
string,
spendby string)
> partitioned by (category string)
> row format delimited
> fields terminated by ','
```

```
> stored as textfile;
```

OK

Time taken: 0.131 seconds

```
hive> set hive.exec.dynamic.partition.mode=nonstrict;
```

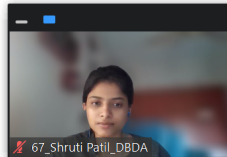
```
hive> insert overwrite table sales_exam_part partition(category)
select x.txn_id, x.txn_date, x.cust_id, x.amount, x.product, x.city,
x.stat
e, x.spendby, x.category from sales_exam x distribute by category;
```

```
hive> create table sales_exam_part(txn_id bigint, txn_date string, cust_id bigint, amount double, product string, city string, state string,
spendby string)
> partitioned by (category string)
> row format delimited
> fields terminated by ','
> stored as textfile;
```

OK

Time taken: 0.131 seconds

```
hive> insert overwrite table sales_exam_part partition(category) select x.txn_id, x.txn_date, x.cust_id, x.amount, x.product, x.city, x.stat
e, x.spendby from sales_exam x distribute by category;
FAILED: SemanticException [Error 10096]: Dynamic partition strict mode requires at least one static partition column. To turn this off set h
ive.exec.dynamic.partition.mode=nonstrict
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> insert overwrite table sales_exam_part partition(category) select x.txn_id, x.txn_date, x.cust_id, x.amount, x.product, x.city, x.stat
e, x.spendby from sales_exam x distribute by category;
FAILED: SemanticException [Error 10004]: Line 1:179 Invalid table alias or column reference 'category': (possible column names are: txn_id,
txn_date, cust_id, amount, product, city, state, spendby)
hive> insert overwrite table sales_exam_part partition(category) select x.txn_id, x.txn_date, x.cust_id, x.amount, x.product, x.city, x.stat
e, x.spendby, x.category from sales_exam x distribute by category;
Query ID = bigcdac432527_20221214100305_927e0f28-4e50-4e62-af94-4dc9dd3c2763
Total jobs = 1
Launching Job 1 out of 1
```



```
Loading data to table exam67.sales_exam_part partition (category=null)
```

Time taken to load dynamic partitions: 0.445

Time taken for adding to write entity : 0.00

MapReduce Jobs Launched:

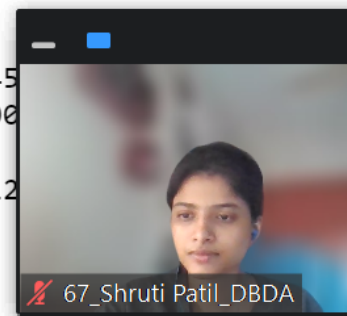
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 12


















Total MapReduce CPU Time Spent: 12 seconds 50 msec

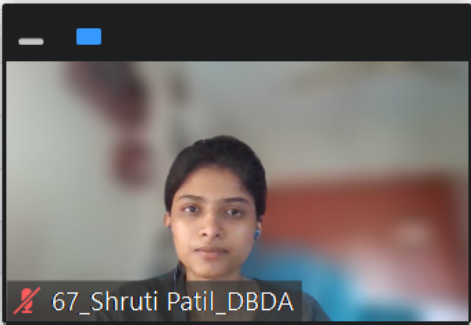
OK

Time taken: 97.049 seconds

```
hive> █
```



<input type="checkbox"/>	Name	Size	User
<input type="checkbox"/>	 ↑		bigcdac432
<input type="checkbox"/>	 .		bigcdac432
<input type="checkbox"/>	 category=Air Sports		bigcdac432
<input type="checkbox"/>	 category=Combat Sports		bigcdac432
<input type="checkbox"/>	 category=Dancing		bigcdac432
<input type="checkbox"/>	 category=Exercise & Fitness		bigcdac432
<input type="checkbox"/>	 category=Games		bigcdac432
<input type="checkbox"/>	 category=Gymnastics		bigcdac432
<input type="checkbox"/>	 category=Indoor Games		bigcdac432
<input type="checkbox"/>	 category=Jumping		bigcdac432
<input type="checkbox"/>	 category=Outdoor Play Equipment		bigcdac432
<input type="checkbox"/>	 category=Outdoor Recreation		bigcdac432
<input type="checkbox"/>	 category=Puzzles		bigcdac432
<input type="checkbox"/>	 category=Racquet Sports		bigcdac432
<input type="checkbox"/>	 category=Team Sports		bigcdac432
<input type="checkbox"/>	 category=Water Sports		bigcdac432
<input type="checkbox"/>	 category=Winter Sports		bigcdac432
Show 100 of 15 items			



PySpark

```
In [1]: > import pyspark
        > from pyspark import SparkContext

In [2]: > sc = SparkContext()

In [4]: > from pyspark.sql.types import *

In [6]: > from pyspark.sql import SparkSession

In [7]: > spark = SparkSession.builder.master("local[1]").appName("test").getOrCreate()

In [10]: > schema1 = StructType().add("year",StringType(),True).add("quarter",StringType(), True).add("AvgRev", DoubleType(), True).add(
<
>

In [11]: > data = spark.read.format("csv").option("header", "True").option("sep", "," ).schema(schema1).load("hdfs://nameservice1/user/big
<
>

In [12]: > data.registerTempTable("airline")

In [20]: > data.printSchema()
```

```
: > data.registerTempTable("airline")

: > data.printSchema()

root
 |-- year: string (nullable = true)
 |-- quarter: string (nullable = true)
 |-- AvgRev: double (nullable = true)
 |-- Booked: integer (nullable = true)
```

```
: > spark.sql("select * from airline limit 5").show()
```

```
+---+-----+-----+-----+
|year|quarter|AvgRev|Booked|
+---+-----+-----+-----+
|1995|      1| 296.9| 46561|
|1995|      2| 296.8| 37443|
|1995|      3| 287.51| 34128|
|1995|      4| 287.78| 30388|
|1996|      1| 283.97| 47808|
+---+-----+-----+-----+
```

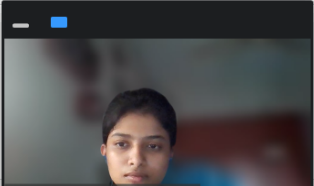
ANS1:

spark.sql("select year, sum(Booked) as total from airline group by year order by total desc limit 1").show()

```
In [23]: # What was the highest number of people travelled in which year?

spark.sql("select year, sum(Booked) as total from airline group by year order by total desc limit 1").show()
```

year	total
2007	176299



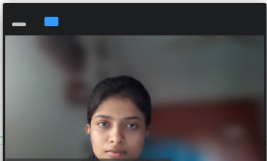
ANS 2:

spark.sql("select year, sum(AvgRev*Booked)/100 as rev from airline group by year order by rev desc limit 1").show()

```
In [24]: # Identifying the highest revenue generation for which year

spark.sql("select year, sum(AvgRev*Booked)/100 as rev from airline group by year order by rev desc limit 1").show()
```

year	rev
2013	663632.0871



ANS 3:

spark.sql("select year, quarter, sum(AvgRev*Booked)/100 as rev from airline group by year,quarter order by rev desc limit 1").show()

```
In [25]: # Identifying the highest revenue generation for which year and quarter (Common group)

spark.sql("select year, quarter, sum(AvgRev*Booked)/100 as rev from airline group by year,quarter order by rev desc limit 1").show()
```

year	quarter	rev
2014	4	188194.0848

