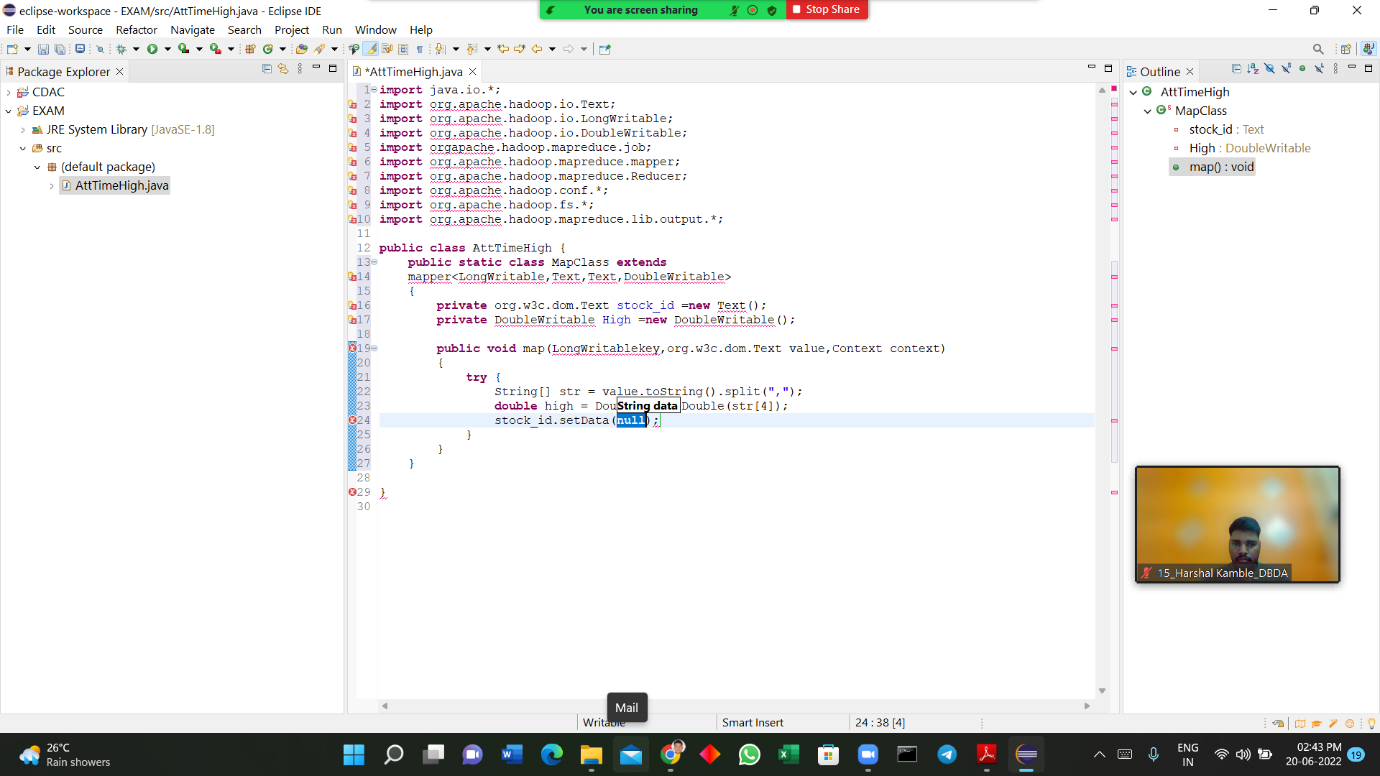
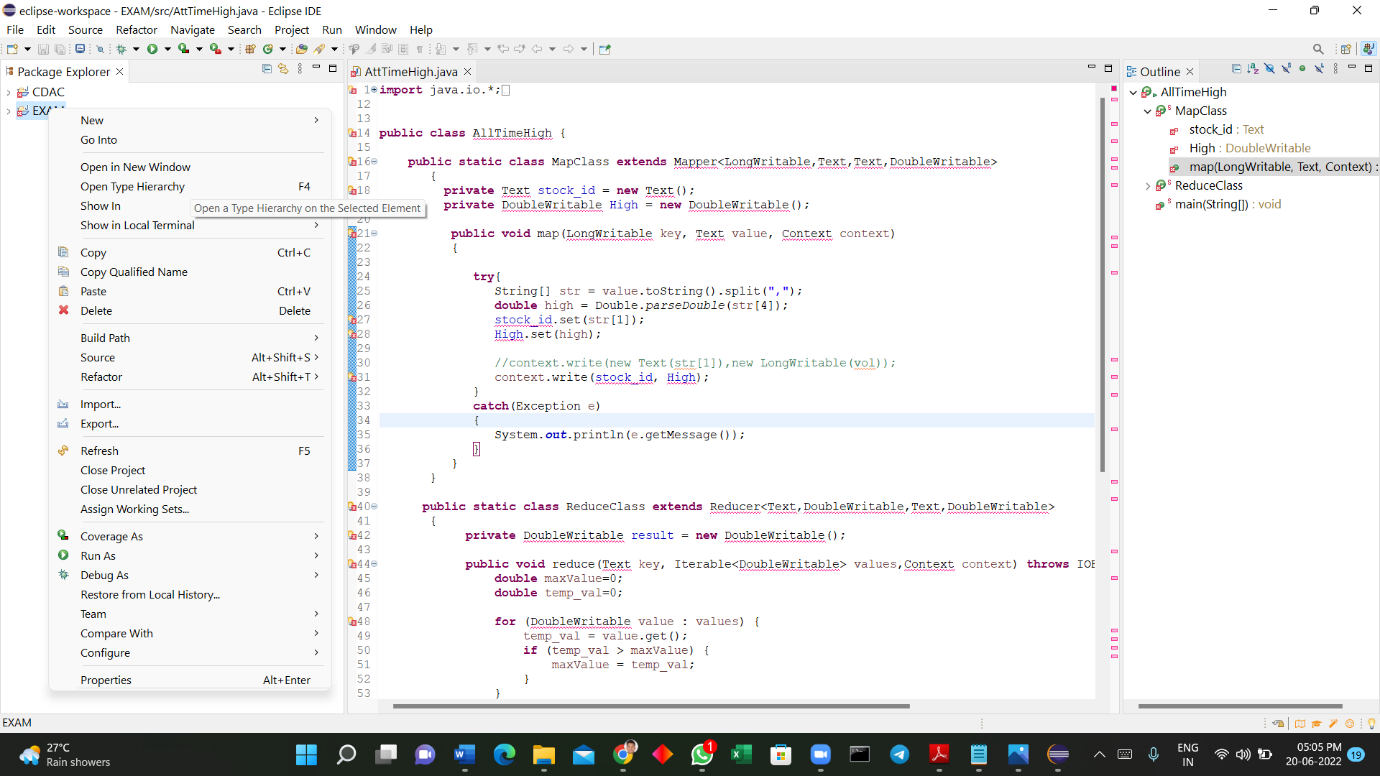
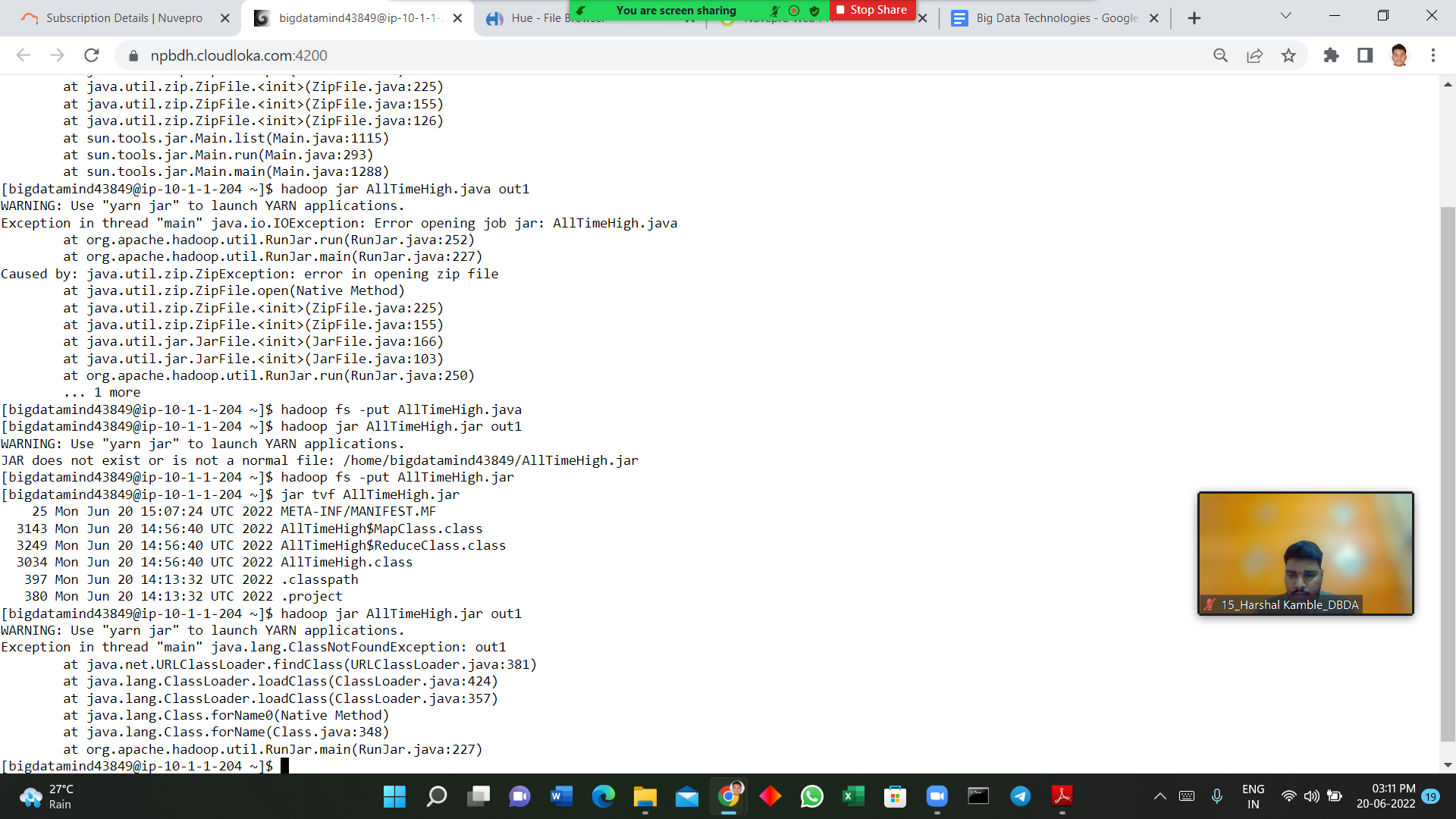
**Question 1**

Q 1. Map Reduce : Find all time High price for each stock

hadoop jar myjar.jar AllTimeHigh CDAC/NYSE.csv exam/mr







**Question No 2**

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

create table customer (custno int , firstname string, lastname string , age int , profession string) row format delimited fields terminated by ','stored a

s textfile;

load data local inpath 'custs.txt' overwrite into table customer ;

select profession , count(custno) from customer group by profession ;

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

> create table txn(txnno int , txndate string , custno int, amount double , category string, product string , city string, state string, spendby string)

> row format delimited fields terminated by ',' stored as textfile;

load data local inpath 'txns1.txt' overwrite into table txn;

select product , sum(amount) total from txn group by product order by total desc limit 10 ;

**3) Write a program to create partiioned table on category**

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

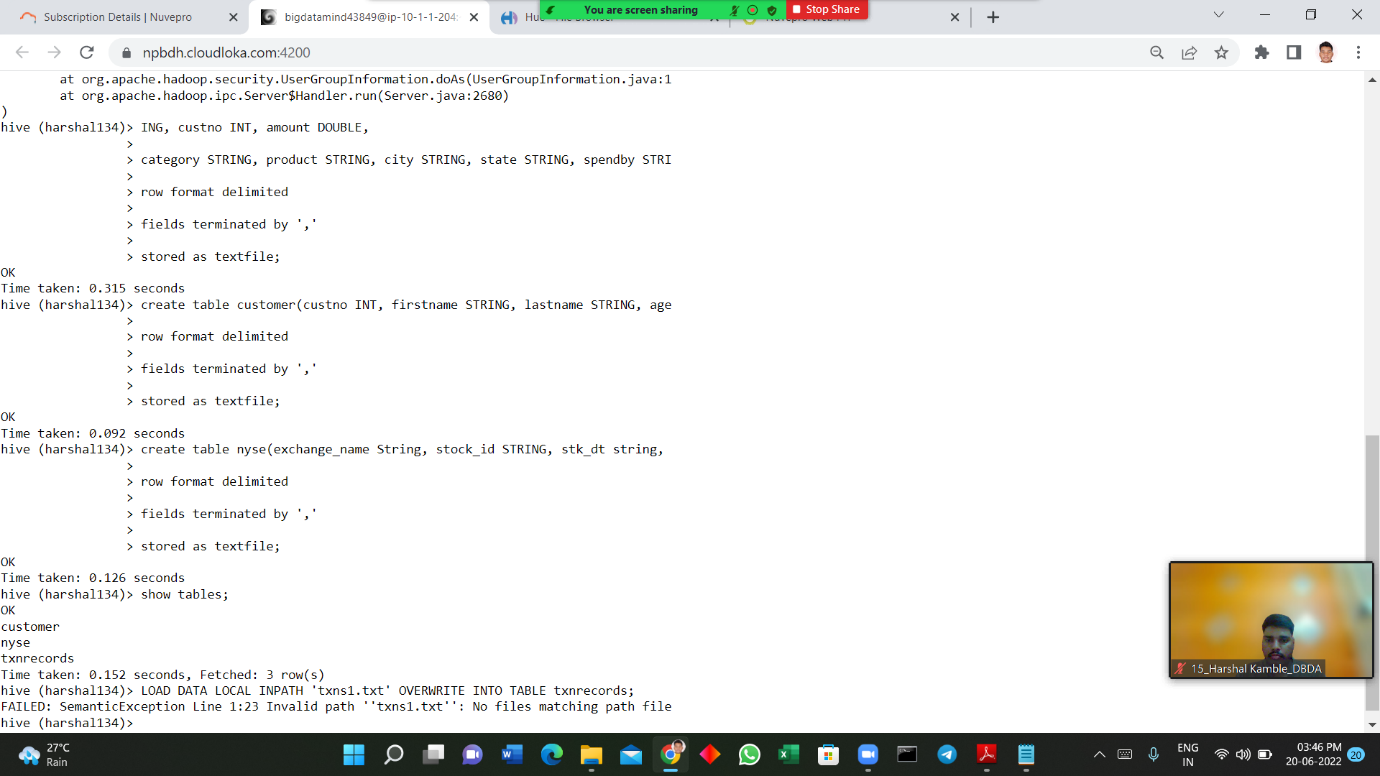
set hive.exec.dynamic.partition = true;

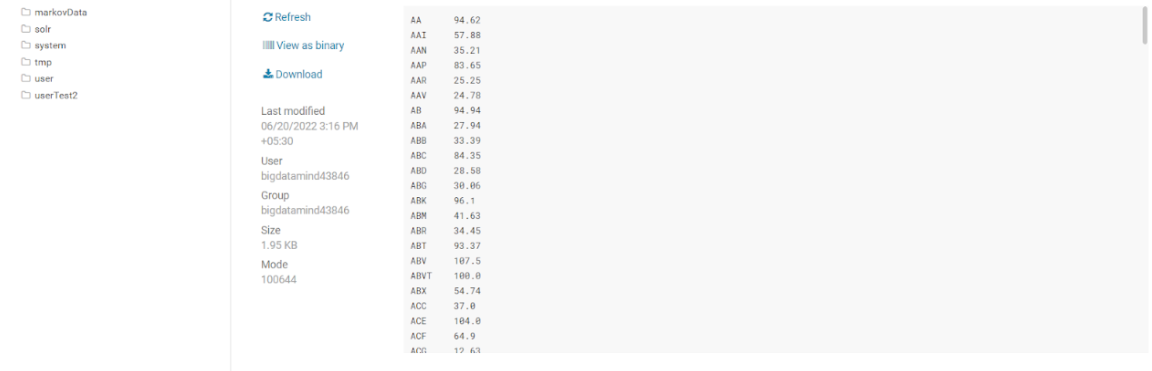
> create table txnbycat(txnno int, txndate string, custno int, amount double, product string, city string, state string, spendby string)

> partitioned by (category string)

> row format delimited fields terminated by ',' stored as textfile;

insert overwrite table txnbycat partition(category) select txn.txnno, txn.txndate, txn.custno, txn.amount, txn.product, txn.city, txn.state, txn.spendby ,txn.category from txn distribute by category;





**Question 3**

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

rdd = sc.textFile("/user/bigdatamind43850/airlines.csv")

>>> header = rdd.first()

>>> rdd2 = rdd.filter(lambda a : a != header)

>>> rdd3 = rdd2.map(lambda a : a.encode("ascii","ignore"))

>>> arrayrdd = rdd3.map(lambda a : a.split(","))

>>> kvrdd = arrayrdd.map(lambda a : (a[0],int(a[3])))

>> counts = kvrdd.reduceByKey(lambda a,b : a+b)

>>> sort = counts.sortBy(lambda a : -a[1])

>>> for i in sort.take(1):

... print(i)

...

('2007', 176299)

**2) Identifying the highest revenue generation for which year**

kvrdd = arrayrdd.map(lambda a :(a[0],float(a[2])\*int(a[3])))

counts = kvrdd.reduceByKey(lambda a,b : a+b)

>>> sort = counts.sortBy(lambda a : -a[1])

>>> for i in sort.take(1):

... print(i)

...

('2013', 66363208.71)

