**Q1.**

**MapReduce**

**Problem Statement**

**Here, we have chosen the stock market dataset on which we have performed map-reduce**

**operations. Following is the structure of the data. Kindlyfind 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**

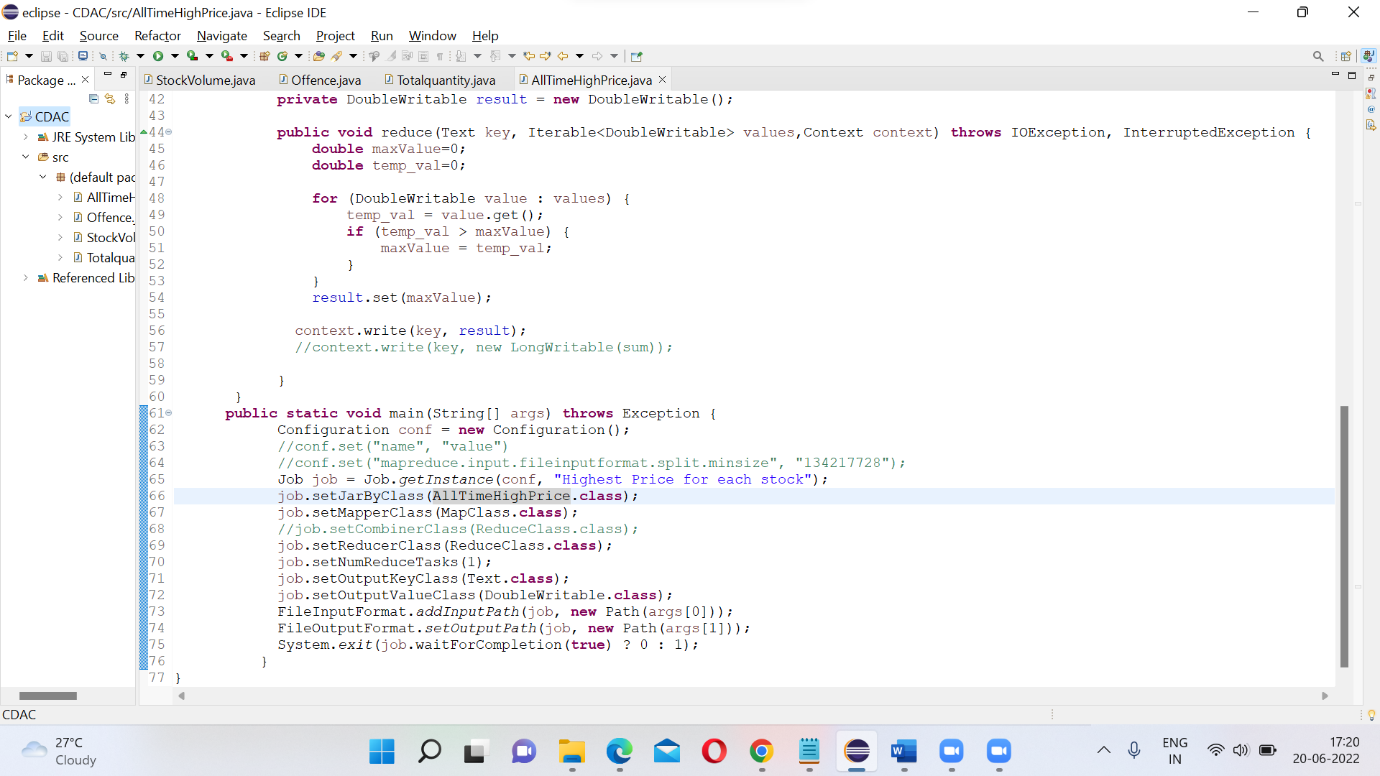
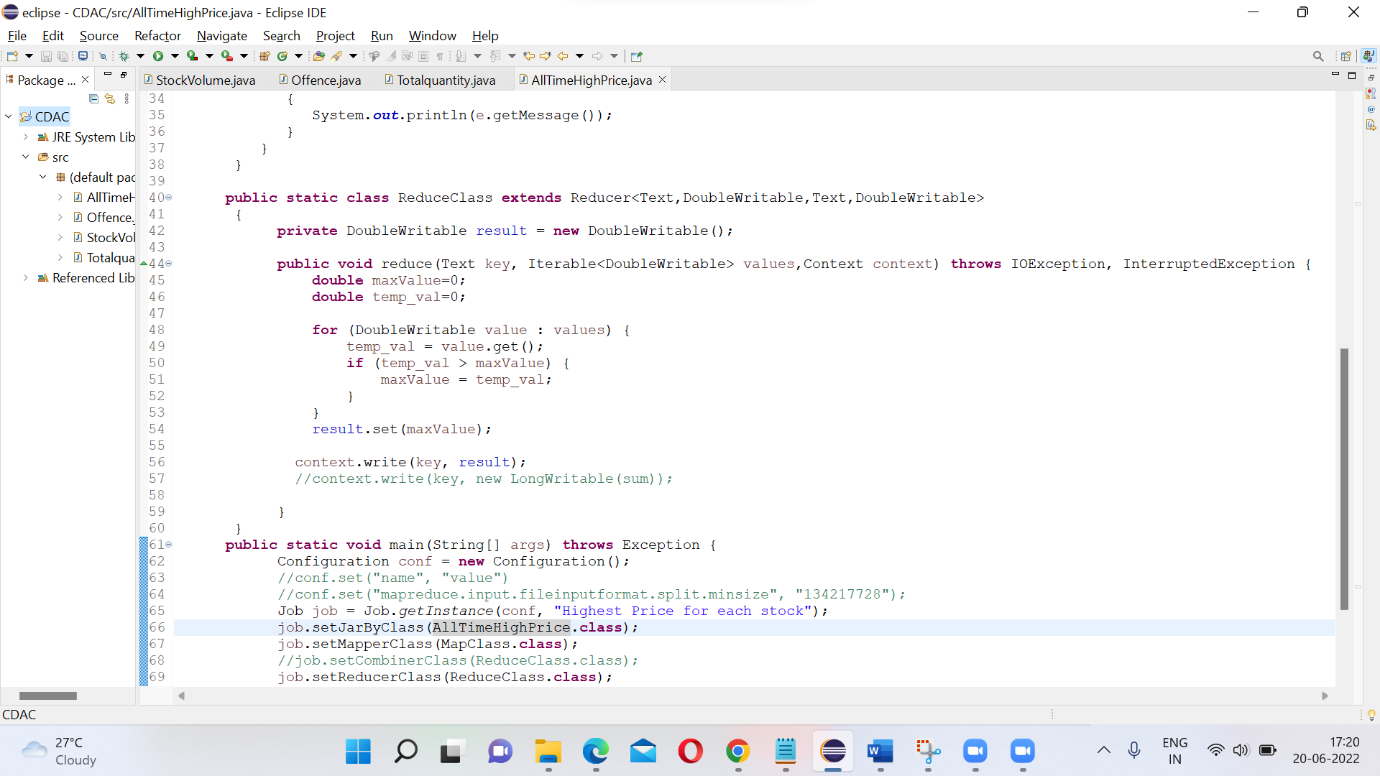
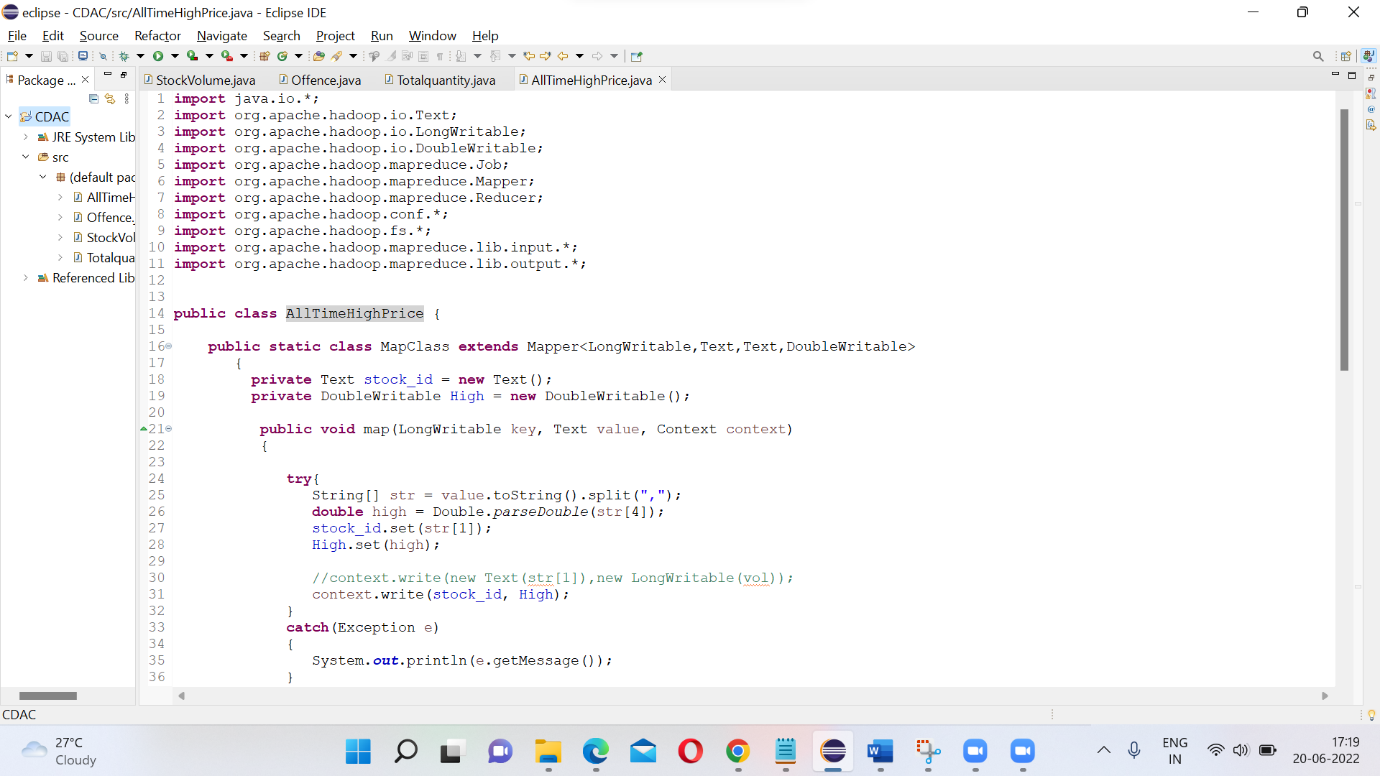
**9. Adjustment Closing price of the stock**

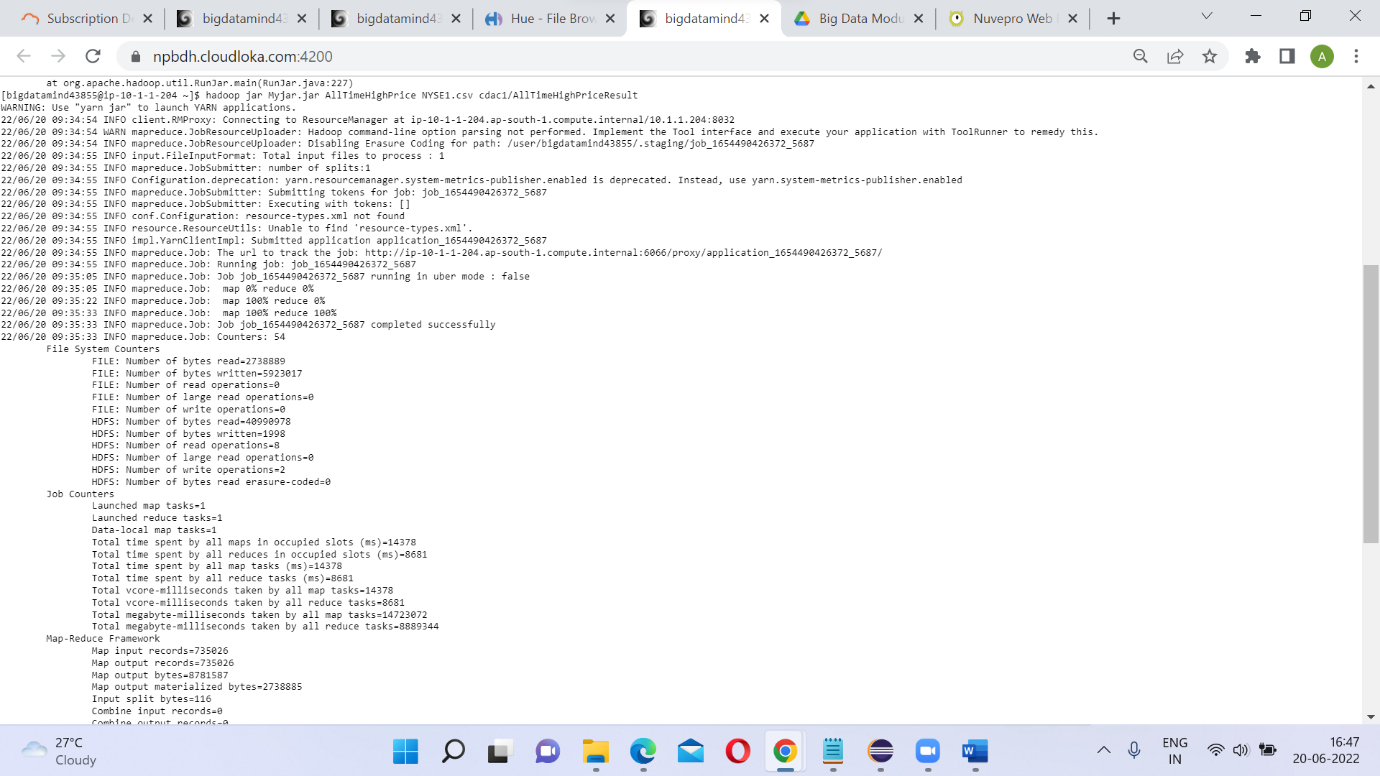
**Field Separator – comma**

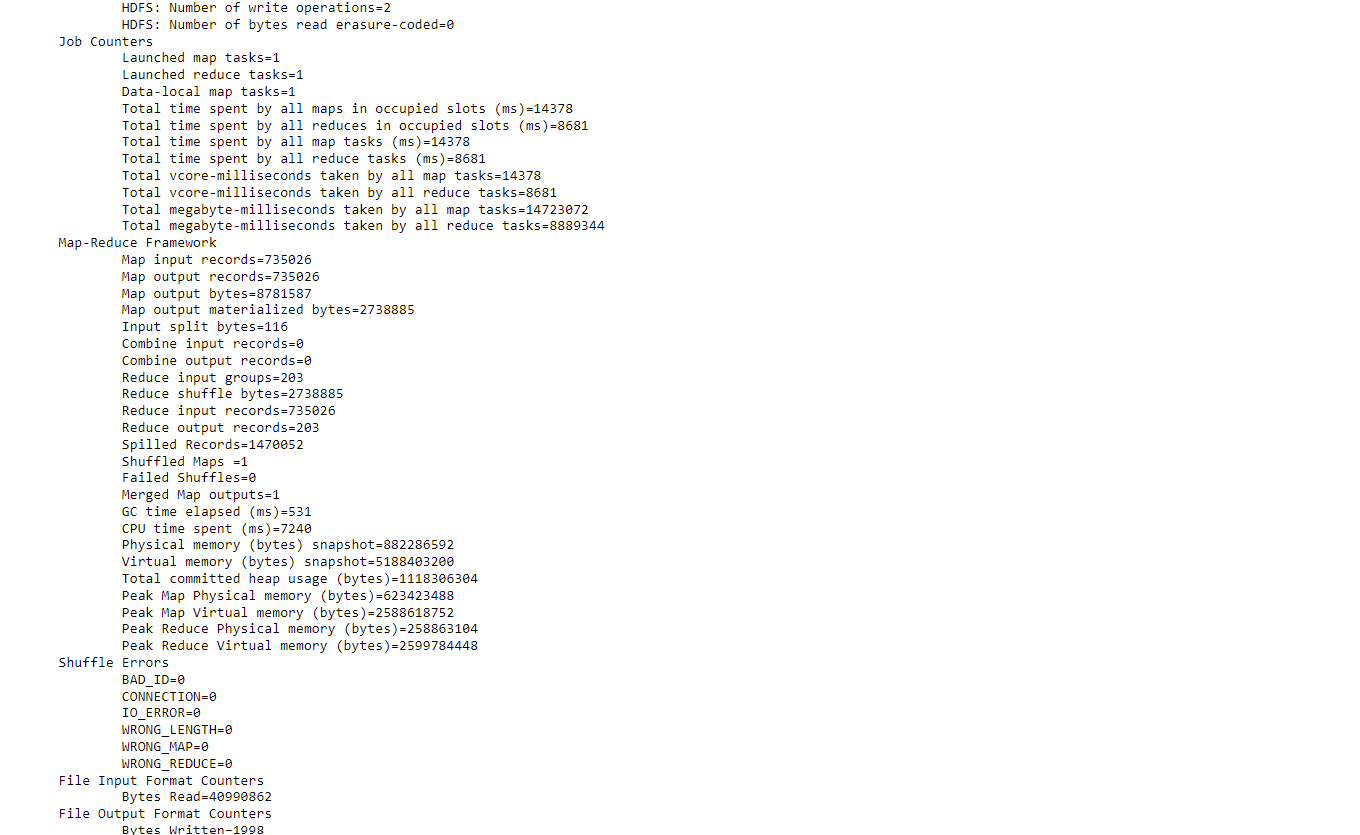
**Question 2 : Find all time High price for each stock**

[bigdatamind43855@ip-10-1-1-204 ~]$ hadoop fs -put NYSE1.csv

[bigdatamind43855@ip-10-1-1-204 ~]$ hadoop jar Myjar.jar AllTimeHighPrice NYSE1.csv cdac1/AllTimeHighPriceResult







Output:



**3.Hive**

**Please find the customer data set.**

**cust id**

**firstname**

**lastname**

**age**

**profession**

**Please find the sales data set.**

**txn id**

**txn date**

**cust id**

**amount**

**category**

**product**

**city**

**state**

**spendby**

hive (training08)> create table customers(cust\_id int, firstname string, lastname string, age int, profession string) row format delimited fields terminated by ',' stored as textfile;

OK

Time taken: 0.413 seconds

hive (training08)> desc customers;

OK

cust\_id int

firstname string

lastname string

age int

profession string

Time taken: 0.109 seconds, Fetched: 5 row(s)

load data inpath '/user/bigdatamind43855/txn

s1.txt' overwrite into table txnrecord;

hive (training08)> select \* from txnrecord limit 5;

OK

0 06-26-2011 4007024 40.33 Exercise & Fitness Cardio Mach

ine Accessories Clarksville Tennessee credit

1 05-26-2011 4006742 198.44 Exercise & Fitness Weightlifti

ng Gloves Long Beach California credit

2 06-01-2011 4009775 5.58 Exercise & Fitness Weightlifti

ng Machine Accessories Anaheim California credit

3 06-05-2011 4002199 198.19 Gymnastics Gymnastics Rings M

ilwaukee Wisconsin credit

4 12-17-2011 4002613 98.81 Team Sports Field Hockey Nas

hville Tennessee credit

Time taken: 0.271 seconds, Fetched: 5 row(s)

hive (training08)> select profession, count(\*) as totalcount from customers group by profession order by totalcount;

load data inpath '/user/bigdatamind43855/cus

ts.txt' overwrite into table customers;

hive (training08)> select \* from customers 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: 0.1 seconds, Fetched: 5 row(s)

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

select profession, count(\*) as totalcount from

customers group by profession order by totalcount;

Output

Social Worker 1

Writer 101

Artist 175

Environmental scientist 176

Carpenter 181

Dancer 185

Therapist 187

Economist 189

Real estate agent 191

Electrical engineer 192

Nurse 192

Civil engineer 193

Automotive mechanic 193

Psychologist 194

Electrician 194

Agricultural and food scientist 195

Athlete 196

Statistician 196

Judge 196

Doctor 197

Financial analyst 198

Accountant 199

Reporter 200

Secretary 200

Coach 201

Physicist 201

Farmer 201

Actor 202

Architect 203

Computer hardware engineer 204

Teacher 204

Engineering technician 204

Designer 205

Musician 205

Childcare worker 207

Veterinarian 208

Chemist 209

Police officer 210

Recreation and fitness worker 210

Pilot 211

Lawyer 212

Social worker 212

Human resources assistant 212

Pharmacist 213

Computer software engineer 216

Firefighter 217

Librarian 218

Loan officer 221

Photographer 222

Computer support specialist 222

Politician 228

Time taken: 72.964 seconds, Fetched: 51 row(s)

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

select product, sum(amount) as max\_sales from

txnrecord group by product order by max\_sales limit 10;

Output

Air Suits 19371.570000000007

Mechanical Puzzles 20381.669999999984

Disc Golf 33944.02000000002

Surfing 35069.079999999965

Tetherball 35611.92999999999

Downhill Skiing 36870.35999999996

Cricket 37061.58000000001

Archery 37088.659999999945

Fishing 37144.60999999998

Exercise Bands 37679.750000000015

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

create table txnpartition(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 txnpartition partition(category) se

lect t.txnid, t.txndate, t.custid, t.amount, t.product, t.city, t.state, t.spendby, t.category from txnrecord t distribute by category;

**QUESTION 3**

**PySpark**

**Please find the AIRLINES data set**

**Year**

**Quarter**

**Average revenue per seat**

**Total number of booked seats**

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

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

>>> schema1 = StructType().add("year",StringType(),True).add("qtr",

StringType(),True).add("arps",DoubleType(),True).add("booked\_seats",

LongType(),True)

>>> airlineDF = spark.read.format("csv").option("header","TRUE").schema

(schema1).load("/user/bigdatamind43855/airlines.csv")

>>> airlineDF.registerTempTable("airlines")

>>> highpass = spark.sql("select year, sum(booked\_seats) as totalpassengers from airlines group by year order by totalpassengers desc")

>>> highpass.show()

+----+---------------+

|year|totalpassengers|

+----+---------------+

|2007| 176299|

|2013| 173676|

|2001| 173598|

|1996| 167223|

|2008| 166897|

|2012| 166076|

|2015| 165438|

|2004| 164800|

|2010| 163741|

|2014| 159823|

|1997| 157972|

|2003| 156153|

|2000| 154376|

|2006| 153789|

|2002| 152195|

|2005| 150610|

|2009| 150308|

|1999| 150000|

|1995| 148520|

|2011| 142647|

+----+---------------+

only showing top 20 rows

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

highrev = spark.sql("select year, sum(Average\_revenue\_per\_seat\*booked\_seats)/1000000 as rev\_in\_millions from airlines group by year order by rev\_in\_millions desc limit 1")

>>> highrev.show()

+----+---------------+

|year|rev\_in\_millions|

+----+---------------+

|2013| 66.36320871|

+----+---------------+

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

**group)**

highrev = spark.sql("select (year,Quarter) as period, sum(average\_revenue\_per\_seat\*booked\_seats)/1000000 as rev\_in\_million from airlines group by period order by rev\_in\_million desc limit 1")

>>> highrev.show()

+---------+--------------+

| period|rev\_in\_million|

+---------+--------------+

|[2014, 4]| 18.81940848|

+---------+--------------+