PROOF OF CONCEPT OR POC ON E-Commerce DATASET ANALYSIS

BY-SAMIKSHA BARNWAL(099samiksha@gmail.com)

COURSE: Bachelors of Computer Applications (Big Data Analytics in association with IBM)

REG NO:12208146

ROLL NO:11

TEACHER'S NAME: M/s GURPREET KAUR

HOD NAME: MR. SARTAAJ SINGH

NAME OF UNIVERSITY: LOVELY PROFESSIONAL UNIVERSITY,PHAGWARA

- ***POC #: E-Commerce DATASET ANALYSIS***
  The POC is based on E-commerce dataset
- ***Public Dataset available at below website***

https://www.kaggle.com/datasets/carrie1/ecommerce-data

INDUSTRY:E-Commerce Industry

ACKNOWLEDGEMENT:

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our 3rd semester B.D.E project supervisor, Ms. Gurpreet Kaur, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project and especially in writing this report. Furthermore, I would also like to acknowledge with much appreciation her crucial role, in giving the permission to use all required equipment and the necessary materials to complete the task 'Analysis E-Commerce DATASET' using Hive, MS Excel and gave suggestion about the task.

TABLE OF CONTENT:

About Dataset

Context

Typically,e-commerce datasets are proprietary and consequently hard to find among publicly available data. However, The UCI Machine Learning Repository has made this dataset containing actual transactions from 2010 and 2011. The dataset is maintained on their site, where it can be found by the title "Online Retail".

Content

"This is a transnational data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail. The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers."

Acknowledgements

Per the UCI Machine Learning Repository, this data was made available by Dr Daqing Chen, Director: Public Analytics group. chend '@' lsbu.ac.uk, School of Engineering, London South Bank University, London SE1 0AA, UK.

Image from stocksnap.io.

Inspiration

Analyses for this dataset could include time series, clustering, classification and more.

DATA: Publicly available dataset with attributes like:

InvoiceNo: an 6-digit string, which is not unique
StockCode: a string of the StockNumber
Description: an string of description of product
Quantity: a integer of no of products
InvoiceDate: a string of date and time when the product has been billed
UnitPrice: a integer number of UnitPrice
CustomerID: a integer number of CustomerID
Country: a string of country

Problem statement:

Problem statement is to

1)Top 10 quantity on comparison with all dataset

2)BOTTOM 10 QUANTITY BASED ON COUNTRY

3) TOP 10 unit price in UK descending order

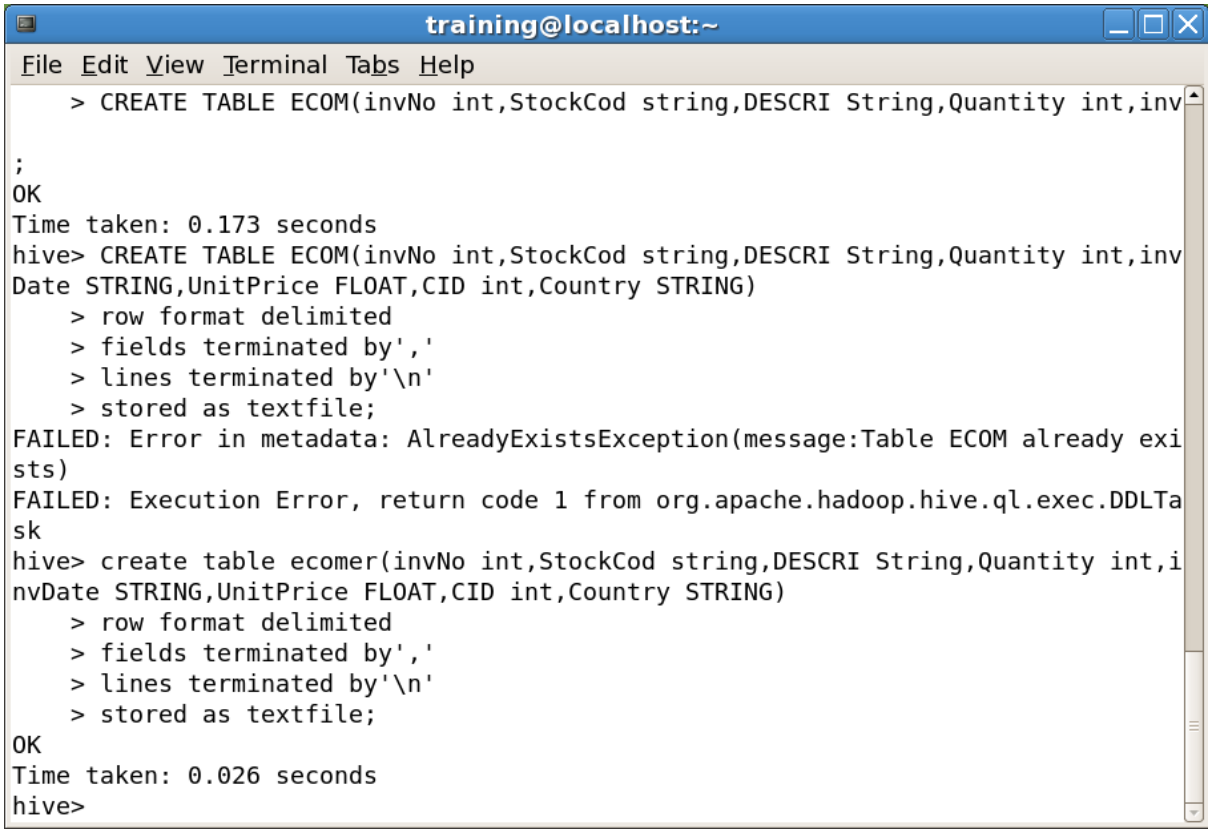
4) Find the AVERAGE UNIT PRICE IN FRANCE

5) Find the TOP 15 unitprice as per description

Shell Script:

Purpose of this shell script is to perform clean-up (delete existing output files) and execute the Hive Commands to store the resultant in Hive Tables and store result in file(CSV format).

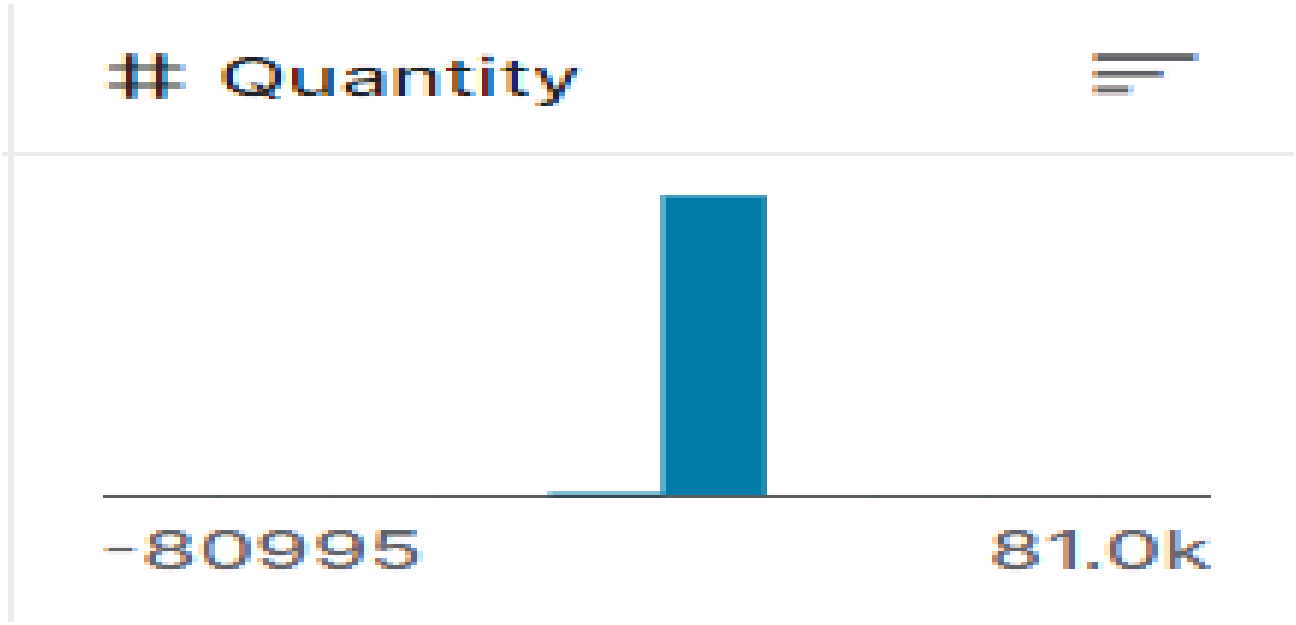
TEP 1: Syntax to create a table and then use it



1)Top 10 quantity on comparison with all dataset

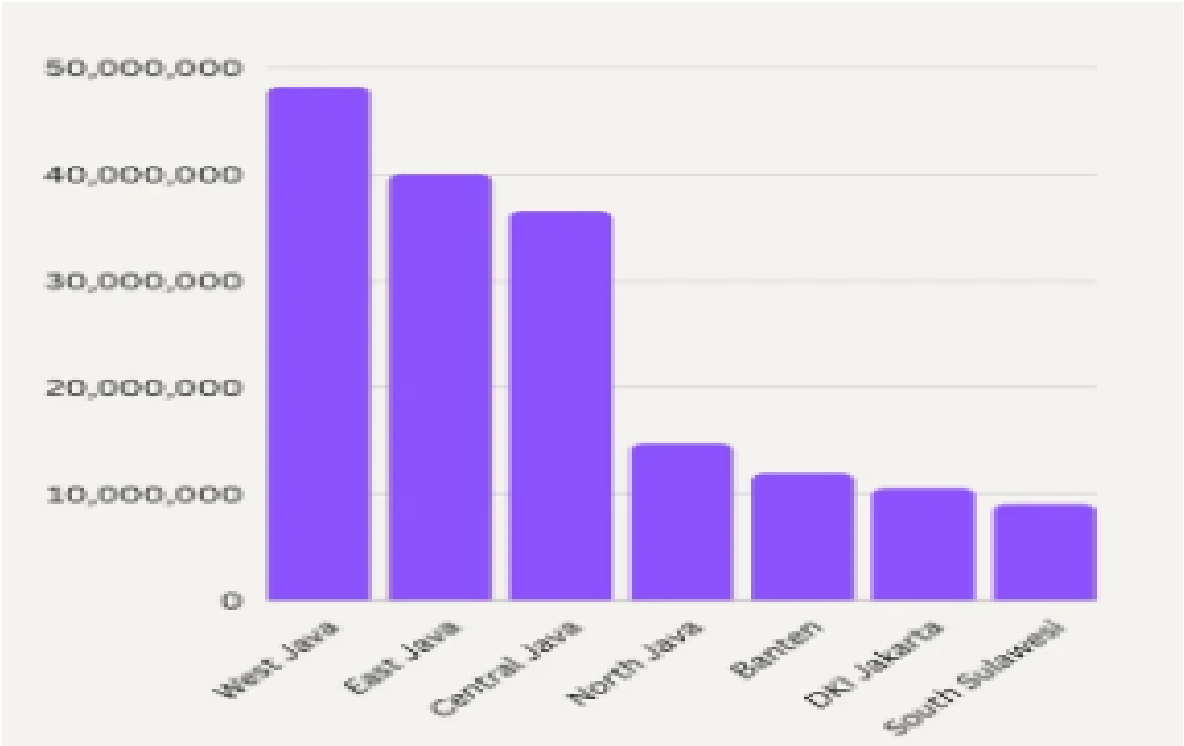
```
hive> select Quantity
    > from ecomer
    > where Quantity IS NOT NULL
    > ORDER BY Quantity ASC
    > LIMIT 10;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<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 mapred.reduce.tasks=<number>
Starting Job = job_202310230137_0009, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_20
2310230137_0009
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:8021 -kill job_2023102301
37_0009
2023-10-27 10:38:49,609 Stage-1 map = 0%,  reduce = 0%
2023-10-27 10:38:52,621 Stage-1 map = 100%,  reduce = 0%
2023-10-27 10:38:58,768 Stage-1 map = 100%,  reduce = 33%
2023-10-27 10:38:59,772 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202310230137_0009
OK
-74215
-9600
-9600
-9360
-9058
-5368
-3667
-3167
```

```
    > LIMIT 10;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 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 mapred.reduce.tasks=<number>
Starting Job = job_202310230137_0009, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_20
2310230137_0009
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:8021 -kill job_2023102301
37_0009
2023-10-27 10:38:49,609 Stage-1 map = 0%,  reduce = 0%
2023-10-27 10:38:52,621 Stage-1 map = 100%,  reduce = 0%
2023-10-27 10:38:58,768 Stage-1 map = 100%,  reduce = 33%
2023-10-27 10:38:59,772 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202310230137_0009
OK
-74215
-9600
-9600
-9360
-9058
-5368
-3667
-3167
-3114
-3100
Time taken: 11.352 seconds
```

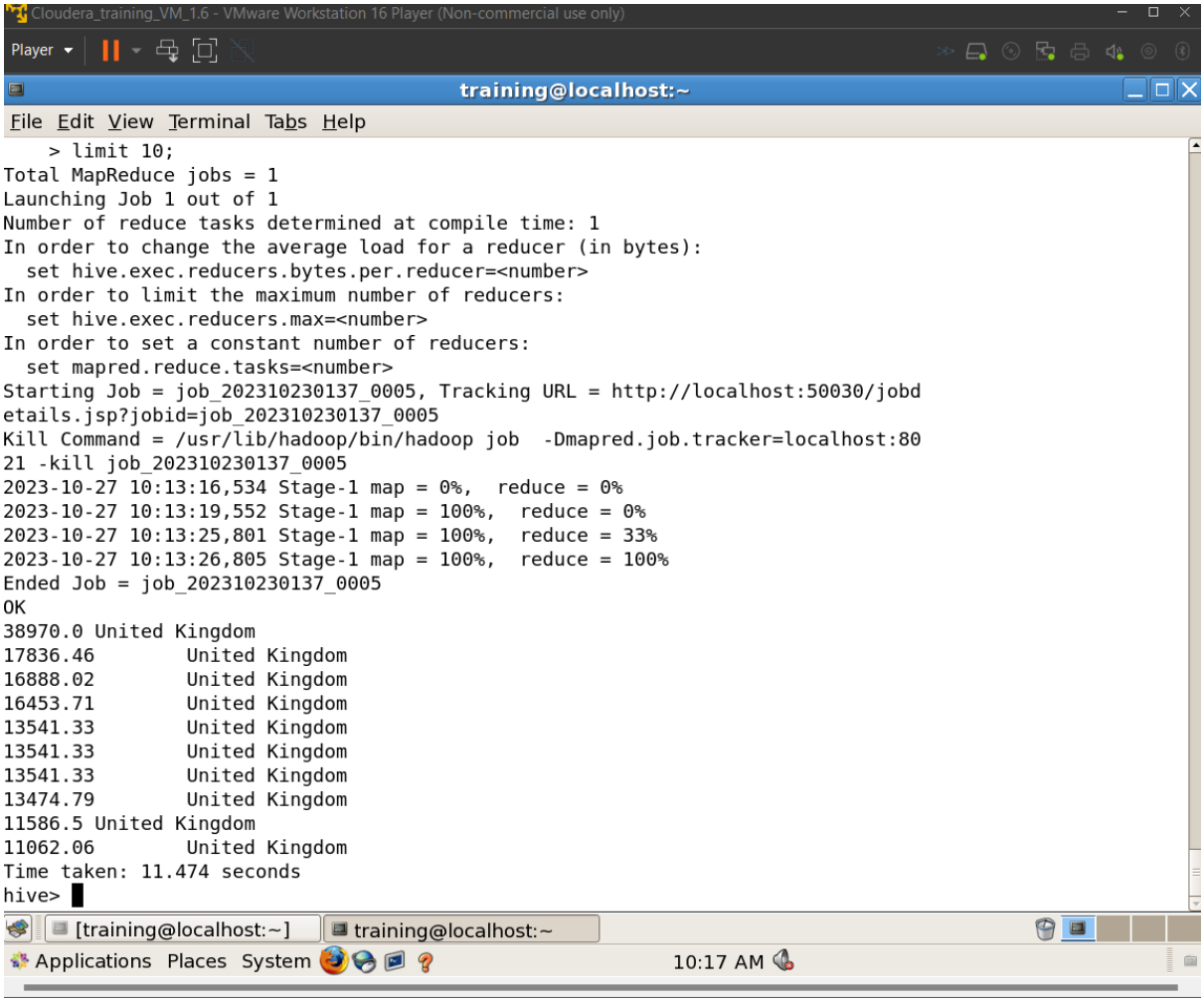


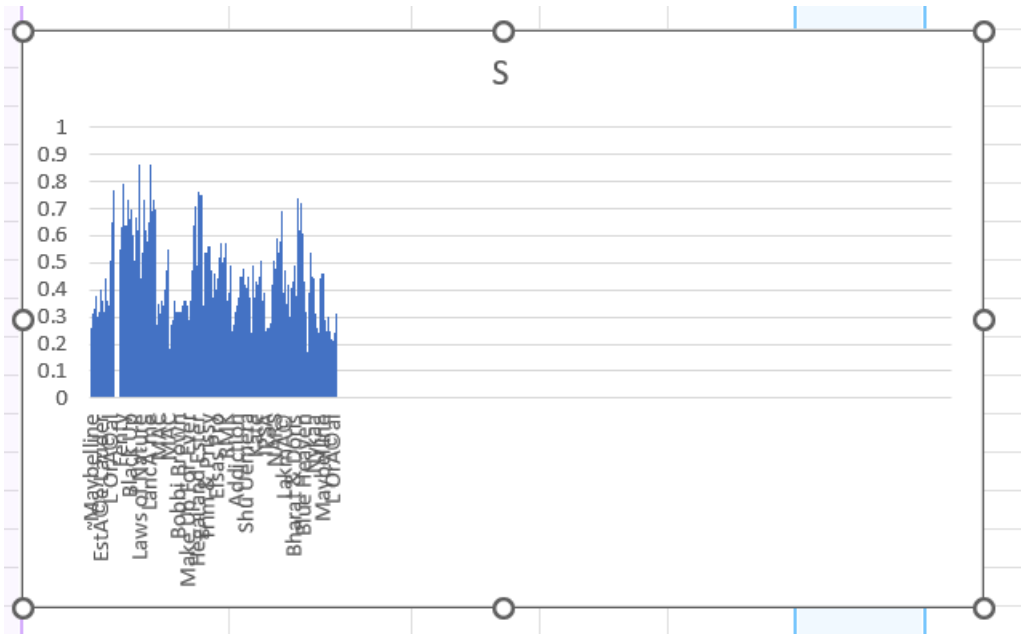
2)BOTTOM 10 QUANTITY BASED ON COUNTRY

```
hive>  select Quantity,Country from ecomer
    > order by Quantity desc
    > limit 10;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 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 mapred.reduce.tasks=<number>
Starting Job = job_202310230137_0008, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_20
2310230137_0008
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:8021 -kill job_2023102301
37_0008
2023-10-27 10:27:51,852 Stage-1 map = 0%,  reduce = 0%
2023-10-27 10:27:54,863 Stage-1 map = 100%,  reduce = 0%
2023-10-27 10:28:00,926 Stage-1 map = 100%,  reduce = 33%
2023-10-27 10:28:01,932 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202310230137_0008
OK
74215   United Kingdom
12540   United Kingdom
5568    United Kingdom
4800    United Kingdom
4300    United Kingdom
4000    United Kingdom
3906    United Kingdom
3186    United Kingdom
3114    United Kingdom
3114    United Kingdom
```



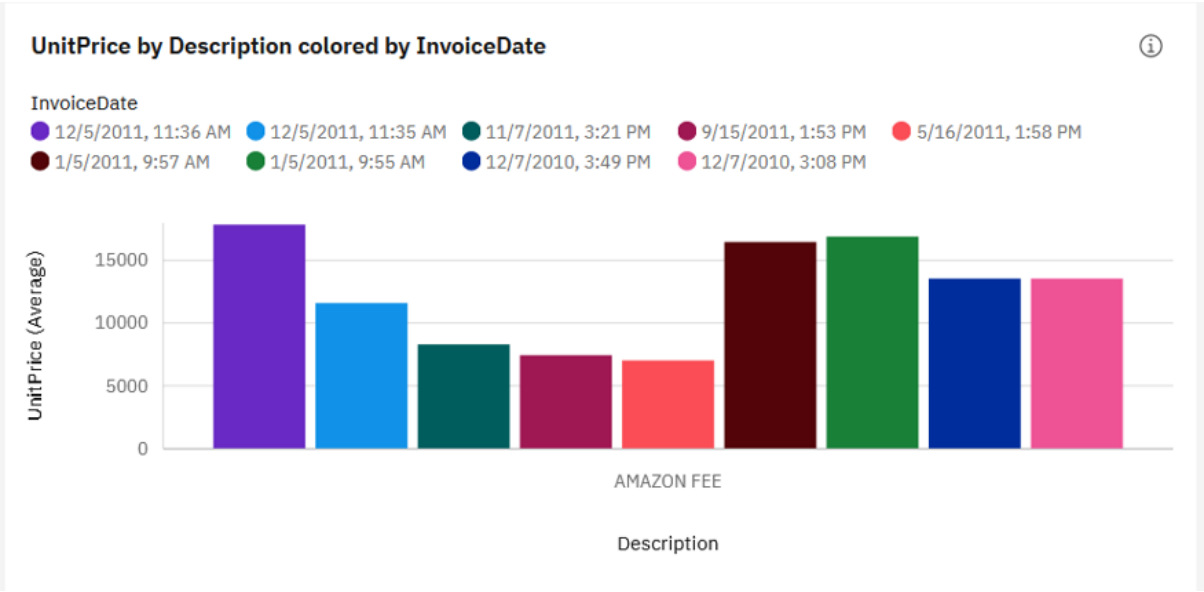
3)TOP 10 UNIT PRICE in UK IN DESC ORDER



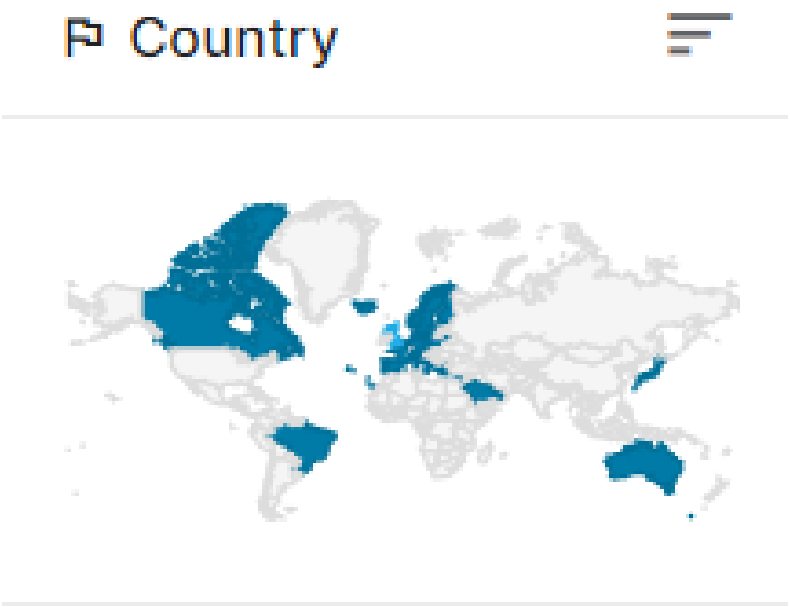


4)Find the AVERAGE UNIT PRICE IN FRANCE

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
hive> SELECT AVG(UNITPRICE) AS avg_sale_price
    > From ecomer
    > WHERE Country = 'France' AND Unitprice IS NOT NULL;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 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 mapred.reduce.tasks=<number>
Starting Job = job_202310230137_0015, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202310230137_0015
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_202310230137_0015
2023-10-27 11:52:16,182 Stage-1 map = 0%,  reduce = 0%
2023-10-27 11:52:18,189 Stage-1 map = 100%,  reduce = 0%
2023-10-27 11:52:24,210 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202310230137_0015
OK
5.0402719346113045
Time taken: 10.252 seconds
hive>
```



5)Find the TOP 15 UNITPRICE AS PER DESCRIPTION




```
File  Edit  View  Terminal  Tabs  Help
21 -kill job_202310230137_0011
2023-10-27 11:30:28,668 Stage-1 map = 0%,  reduce = 0%
2023-10-27 11:30:31,680 Stage-1 map = 100%,  reduce = 0%
2023-10-27 11:30:37,920 Stage-1 map = 100%,  reduce = 33%
2023-10-27 11:30:38,923 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202310230137_0011
OK
Manual  38970.0
AMAZON FEE      17836.46
AMAZON FEE      16888.02
AMAZON FEE      16453.71
AMAZON FEE      13541.33
AMAZON FEE      13541.33
AMAZON FEE      13541.33
AMAZON FEE      13474.79
AMAZON FEE      11586.5
Adjust bad debt 11062.06
AMAZON FEE      8286.22
POSTAGE 8142.75
POSTAGE 8142.75
AMAZON FEE      7427.97
AMAZON FEE      7006.83
Time taken: 12.236 seconds
hive>
```

REFERENCES:

1)https://www.kaggle.com/datasets/carrie1/ecommerce-data