CAPSTONE PROJECT

Panagam Mohitha

Part I – Python (50 Marks)

Attached PDF report, python notebook and HTML file along with this report

Summary:

The top product categories of the Ecommerce Company are clothing, Mobiles & Accessories, Automotive, Home Decor & Festive Needs, Computers. Allure Auto has the highest Brand Revenue among all the 2580 unique brands. Head(n), Tail(n) prints out the top n rows and the bottom n records in the data frame respectively. Shape of the data frame gives the no. of rows and columns of the data frame. Another way would be length of the axes to get the total rows and columns. dtypes.value_counts() returns the no. of variables with given data type. Drop function is used to permanently delete the columns from the data frame. ECom_Data['Brand'].unique() returns the length of the array of unique values of the brand. groupby() returns the data group by brand and mean() gives the calculated average ratings of the products for each brand.

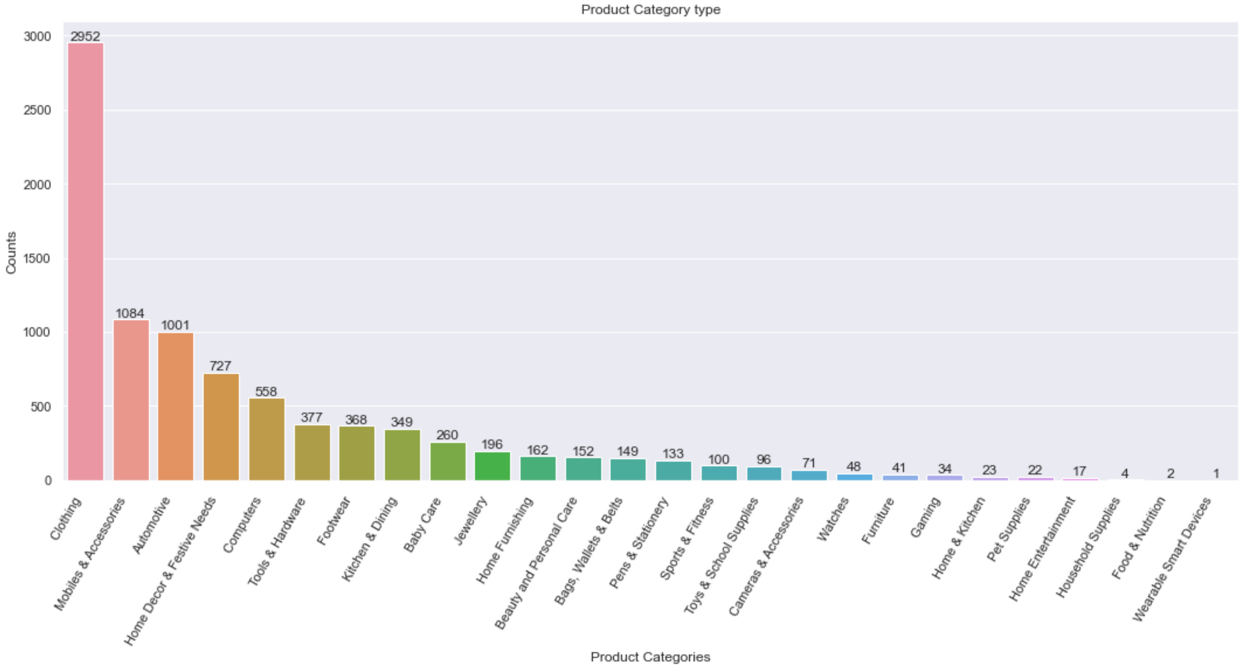


Fig.1 Count plot of Product Categories

The Retail price vs Discounted price chart shows us the prices are concentrated at the 20000 and there is gradual increase in the Retail price and also shown same as the Discounted price.

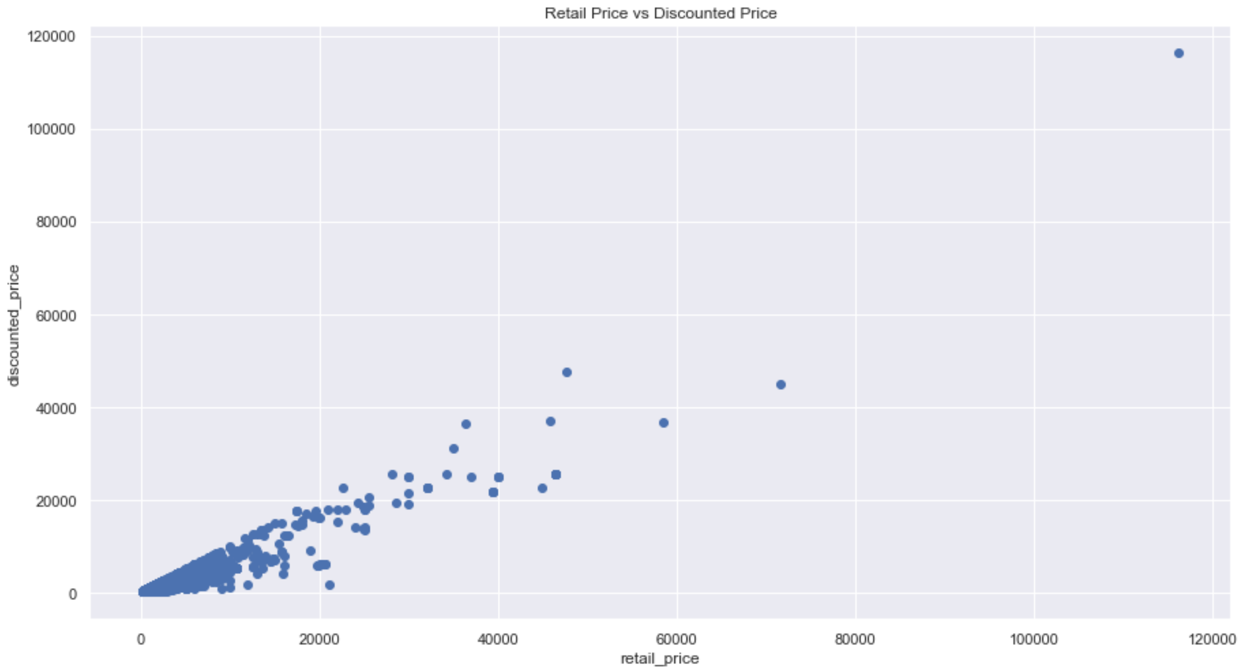


Fig.2 Scatter plot of Retail Price vs Discounted Price

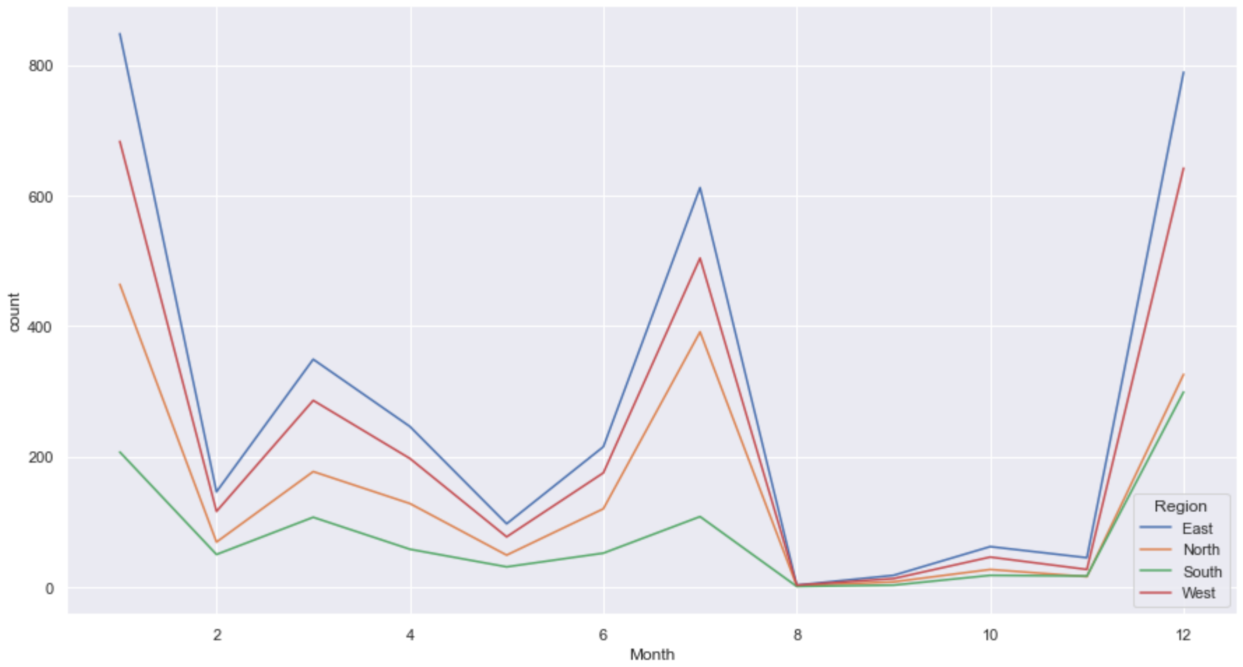


Fig.3 Line chart showing performance of Ecommerce company in different regions

January is the best performing month for all except South region. August is worst performing month for all regions. Whereas South region records December as it's best performing month

Appendix PDF of Python notebook

Capstone Project

Name: Panagam Mohitha

Problem Statement Sales Analytics

Context: You have been given the role of a business analyst for an E-Commerce company and have been asked to prepare a basic report on the data. Follow the steps below for preparation of the report.

Before you start analysing the data, it is always a good practice to see the size of the data, its features and feature types. If the data set is big, it is not possible to print out all the records.

Load the necessary libraries. Import and load the dataset with a name ECom_Data .

In [78]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
pd.options.display.max_rows = 10
import warnings
import os
import re
import copy
```

In [79]:

```
# suppress display of warnings
warnings.filterwarnings('ignore')

# display all dataframe columns
pd.options.display.max_columns = None

# to set the limit to 3 decimals
pd.options.display.float_format = '{:.3f}'.format

# display all dataframe rows
pd.options.display.max_rows = None
```

In [80]:

```
# Get the Data

ECom_Data=pd.read_csv('C:\\Users\\Mohitha Panagam\\Downloads\\Capstone_Project\\E-
```

We have read the data and stored the data in "ECom_Data" variable

Q 1. To get familiar with the data

a) Print out the first 10 and the last 10 records of the data. (2 marks)

b) How many rows and columns are present in the dataset? Use any two different methods to extract this information. (2 + 2 + 1 marks)

c) How many object data types are there? (1 mark)

d) Is there any Boolean data type? (1 mark)

Note: Use the markdown feature of Python to explain your answer.

Ans 1 a)

```
In [105…  ECom_Data.head(10)
```

Out[105]:

	Customer_uniq_id	Region	Order_Date	Expected_Delivery_Date	Delivered_Dat
0	c2d766ca982eca8304150849735ffef9	North	31-12-2021	03-01-2022	02-01-202
1	7f7036a6d550aaa89d34c77bd39a5e48	East	31-12-2021	03-01-2022	02-01-202
2	f449ec65dcbc041b6ae5e6a32717d01b	East	31-12-2021	03-01-2022	02-01-202
3	0973b37acd0c664e3de26e97e5571454	West	31-12-2021	03-01-2022	02-01-202
4	bc940ea42ee6bef5ac7cea3fb5cfbee7	West	31-12-2021	03-01-2022	02-01-202
5	ce5a6818f7707e2cb61fdcdbba61f5ad	East	25-12-2021	28-12-2021	27-12-202
6	8542703ca9e6ebdf6d742638dfb1f2ca	East	25-12-2021	28-12-2021	27-12-202
7	29c8d290caa451f97b1c32df64477a2c	North	25-12-2021	28-12-2021	27-12-202
8	4044c0ac52c1ee4b28777417651faf42	East	25-12-2021	28-12-2021	27-12-202
9	e54bc0a7c3429da2ebef0b30331fe3d2	East	25-12-2021	28-12-2021	27-12-202

In [106…

```
ECom_Data.tail(10)
```

Out[106]:

	Customer_uniq_id	Region	Order_Date	Expected_Delivery_Date	Delivered_
8917	4db03bc4ccbe216cf151b2f2b904ba3f	East	01-12-2020	04-12-2020	04-12-
8918	4d4d6aeb13fa253499d0dd45a5abd87e	West	01-12-2020	04-12-2020	04-12-
8919	ea19e7ef703293d3f6c799ca9db4642d	West	01-12-2020	04-12-2020	04-12-
8920	19a53a958992fb575acffb5d41e7ef9e	East	01-12-2020	04-12-2020	04-12-
8921	09abab80c8dfdc6f268e0a6f05a0be11	North	01-12-2020	04-12-2020	04-12-
8922	90d30478255e23621e8929ed15c2f6e4	South	01-12-2020	04-12-2020	04-12-
8923	20a73e3f41490a73ceeba5f17658db8f	West	01-12-2020	04-12-2020	04-12-
8924	5c1554cd45f9d538c2c6947dbdd59c75	East	01-12-2020	04-12-2020	04-12-
8925	6b737a4deca1ed0e56c179e66036e994	West	01-12-2020	04-12-2020	04-12-
8926	a5235ac28d3d5487f54025f9d6b57433	North	01-12-2020	04-12-2020	04-12-

Ans 1 b)

In [83]:

```
# Method 1
print("Number of Rows:", ECom_Data.shape[0])
print("Number of Columns:", ECom_Data.shape[1])
```

```
Number of Rows: 8927
Number of Columns: 17
```

In [84]:

```
# Method 2
print("Number of Rows: " + str(len(ECom_Data.axes[0])))
print("Number of Columns: " + str(len(ECom_Data.axes[1])))
```

```
Number of Rows: 8927
Number of Columns: 17
```

In [85]:

```
# Method 3
ECom_Data.info(verbose = False)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8927 entries, 0 to 8926
Columns: 17 entries, Customer_uniq_id to description
dtypes: float64(1), int64(2), object(14)
memory usage: 1.2+ MB
```

Ans 1 c)

In [86]:

```
try:
    is_object = ECom_Data.dtypes.value_counts()['object']
    print('Count of object data types: ', is_object)
except:
    print('No object type data')
```

```
No object type data
```

Ans 1 d)

In [87]:

```
try:
    is_bool = ECom_Data.dtypes.value_counts()['boolean']
    print('Count of object data types: ', is_bool)
except:
    print('No boolean type data')
```

```
No boolean type data
```

Inference:

- head(n), tail(n) prints out the top n rows and the bottom n records in the dataframe respectively
- Shape of the dataframe gives the no.of rows and columns of the data frame. Another way would be length of the axes to get the total rows and columns.
- Number of Rows: 8927
- Number of Columns: 15
- dtypes.value_counts() returns the no. of variables with given datatype
- There are 14 object data type variables and zero boolean attributes

Once you are familiar with the data, you may decide that not all features are of use to you and you may want to delete the non-informative features (columns)

Q 2. Eliminating the non-informative columns.

a) Drop the columns product_specifications and description. (2 marks)

b) Which method or function is used to permanently delete the columns mentioned in part (b)? Write the code explicitly (2 marks)

Note: Use the markdown feature of Python to explain your answer.

Ans 2 a)

In [88]:
```
ECom_Data.drop(columns=['product_specifications','description']).head(1)
```

Out[88]:

	Customer_uniq_id	Region	Order_Date	Expected_Delivery_Date	Delivered_Date
0	c2d766ca982eca8304150849735ffef9	North	31-12-2021	03-01-2022	02-01-2022

Ans 2 b)

In [89]:
```
ECom_Data = ECom_Data.drop(columns=['product_specifications','description'])
```

Inference:

- Drop function is used to permanently delete the columns from the dataframe.

The next steps in this project involves summarization of data at various levels and visualization. Such apparently simple steps are very useful to get an overall sense of the data.

Q 3. Here we summarize the data at brand level.

a) How many unique Brands are there. (1 mark)

b) Note that each brand contains multiple products. Show the average rating of the products within each Brand (2 marks)

Ans 3 a)

In [90]:
```
print('Number of unique Brands:', len(ECom_Data['Brand'].unique()))
```

```
Number of unique Brands: 2580
```

Ans 3 b)

In [107…

```
ECom_Data.groupby(['Brand'])['product_rating'].mean().reset_index()
```

Out[107]:

	Brand	product_rating
0	1OAK	1.500
1	3A AUTOCARE	3.268
2	3D Mat	3.000
3	3kFactory	2.000
4	4D	3.600
5	4D Mats	4.333
6	5 FEELINGS	4.000
7	69th Avenue	2.000
8	720 Armour	2.500
9	99Gems	2.500
10	@home	3.250
11	A A STORE	4.750
12	A R ENTERPRISES	4.000
13	A To Z Traders	1.000
14	A-maze	3.000
15	ABIDA	5.000
16	ABSTAR	5.000
17	ACCESSOREEZ	4.000
18	ACM	3.400
19	ADDICTION	1.333
20	ADIWALK	2.000
21	AHAANA FASHION	3.000
22	AJ Dezines	3.300
23	AKUP	2.591
24	ALIFS	5.000
25	ALL DAY 365	2.333
26	AM Creation	5.000
27	AMCO	2.000
28	AMZER	4.333
29	ANAHI	2.500
30	ANAND ARCHIES	4.000
31	ANASAZI	4.000
32	AND Designs	4.500
33	APG	5.000
34	APOLLO+	4.000
35	APS	3.111

	Brand	product_rating
2556	texclusive	1.000
2557	the jewelbox	4.000
2558	tiwaritraders	2.000
2559	topolino	2.000
2560	totalcare Expert	4.000
2561	tp-Link	4.000
2562	trustedsnap	5.000
2563	tryfeet	3.000
2564	umda	1.000
2565	urban monk creations	3.000
2566	vatika	5.000
2567	vency creation	2.000
2568	veronica	5.000
2569	vinay	5.000
2570	vinaya	4.333
2571	walletsnbags	1.000
2572	wallskart	3.000
2573	womaniya	1.000
2574	xpert	5.000
2575	youniqueshop	2.000
2576	zDelhi.com	5.000
2577	zaidis	4.000
2578	zasmina	4.500
2579	Tarkan	5.000

Inference:

- ECom_Data['Brand'].unique() returns the length of the array of unique values of the brand
- groupby() returns the data group by brand and mean() gives the calculated average ratings of the products for each brand

Q 4. Next we study the main categories of the products.

a) Create an appropriate plot to show the count of items ordered for each product_main_category. (5 marks).

Hint: Create a bar chart titled "Product Category type" where product names are on x-axis and counts are on y-axis.

Note: Both axis labels, i.e. the names of the product categories and counts must be clearly legible.

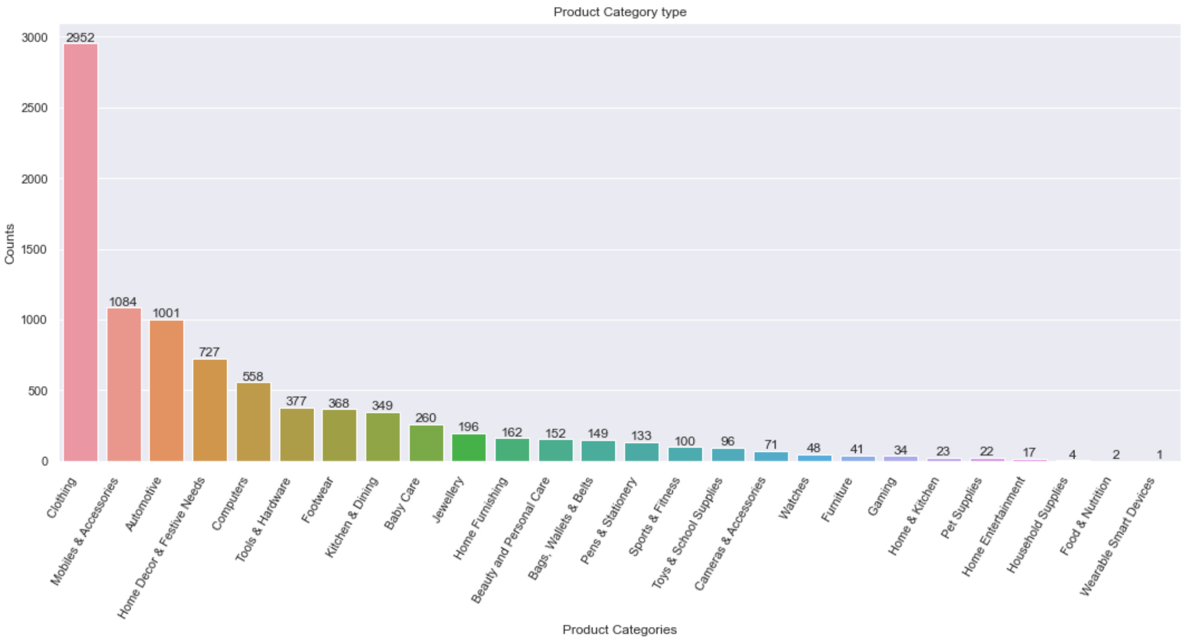
b) From the plot identify for which two product_main_category(s) the maximum and the minimum orders were placed. (2 marks)

c) Write code to print out the top 5 product_main_category(s) in descending order? (2 marks)

Ans 4 a)

In [108…

```
plt.figure(figsize=(15,8))
cp = sns.countplot(x ='product_main_category', data = ECom_Data, order = ECom_Data
cp.set_xticklabels(cp.get_xticklabels(), rotation=60, ha="right")
cp.bar_label(cp.containers[0])
plt.tight_layout()
plt.xlabel("Product Categories")
plt.ylabel("Counts")
plt.title("Product Category type")
plt.show()
```



Ans 4 b)

From the above graph,

- clothing has the maximum number of orders placed and
- Wearables Smart Devices have the minimum orders were placed

Ans 4 c)

In [93]:

```
Product_category_sort = ECom_Data.groupby(['product_main_category'])['Customer_uni
.count().reset_index().sort_values(by=['Customer_uniq_id'], ascending=False).renam
```

In [94]:

```
Product_category_sort.head(5)
```

Out[94]:

	product_main_category	orders
5	Clothing	2952
18	Mobiles & Accessories	1084
0	Automotive	1001
12	Home Decor & Festive Needs	727
6	Computers	558

In E-commerce, both the retailers (here brands) and the company have to make profit to sustain in the business. The E-Commerce company has the following rule for computing their own revenue, which depends on how much each retailer can sell.

The company charges each retailer (Brand)

(i) 25% on the final price (discounted price) of total orders if the number of orders placed is greater than 600

(ii) 15% on the final price (discounted price) of total orders if the number of orders placed is greater than 350 but less than or equal to 600

(iii) 10% on the final price (discounted price) of total orders if the number of orders placed is greater than 100 but less than or equal to 350

(iv) Otherwise, 5% on the final price (discounted price) of total orders

Q 5. Find the net revenue generated by the E-Commerce company over all orders placed. (6 marks)

Ans 5

In [95]:

```
def total_payment(retailer_sale):
    if retailer_sale.orders > 600:
        Ecom_charges = retailer_sale.discounted_price *0.25
    elif retailer_sale.orders > 350 and retailer_sale.orders <= 600 :
        Ecom_charges = retailer_sale.discounted_price *0.15
    elif retailer_sale.orders > 100 and retailer_sale.orders <= 350 :
        Ecom_charges = retailer_sale.discounted_price *0.10
    else:
         Ecom_charges = retailer_sale.discounted_price *0.05
    return Ecom_charges
```

In [96]:

```
net_revenue = ECom_Data.groupby(['Brand']).agg({'Customer_uniq_id':'count','discou
                                .rename({'Customer_uniq_id':'orders'},axis=1)
net_revenue['Ecom_charges'] = net_revenue.apply(total_payment, axis = 1)
print('Total revenue of E-commerce company', round(net_revenue['Ecom_charges'].sum
```

```
Total revenue of E-commerce company 615446.95
```

Now you need to find the revenue for each retailer (Brand)

Q6.Calculate the BrandRevenue for each brand and list the top 14 brands having maximum revenue in descending order (5 marks)

Hint: Brand revenue is different from the E-Commerce company revenue.

Ans 6

In [97]:

```
net_revenue['brand_profit'] = net_revenue['discounted_price'] - net_revenue['Ecom_
net_revenue_brand = net_revenue[['Brand','brand_profit']]
net_revenue_brand.sort_values(by=['brand_profit'],ascending=False).reset_index(dro
```

Out[97]:

	Brand	brand_profit
0	Allure Auto	564926.150
1	GAGA	300694.000
2	Slim	244959.300
3	DailyObjects	218376.000
4	Diviniti	181279.000
5	Enthopia	147775.500
6	Regular	137232.000
7	Asus	125705.900
8	TheLostPuppy	123453.900
9	Springwel	112706.100
10	Audeze	110477.400
11	Lal Haveli	93802.050
12	FabHomeDecor	86054.800
13	Netgear	83484.100

Let us now investigate multiple features for each product to determine any pattern.

Q 7. Compare prices for each product.

a) Draw boxplots of retail_price & discount_price. (2 marks)

b) Are there any outliers? (Yes/No) (1mark)

c) Create a scatterplot of retail_price (x-axis) and discounted_price (y-axis) (2 marks)

In [98]:

```
ECom_Data.head(1)
```

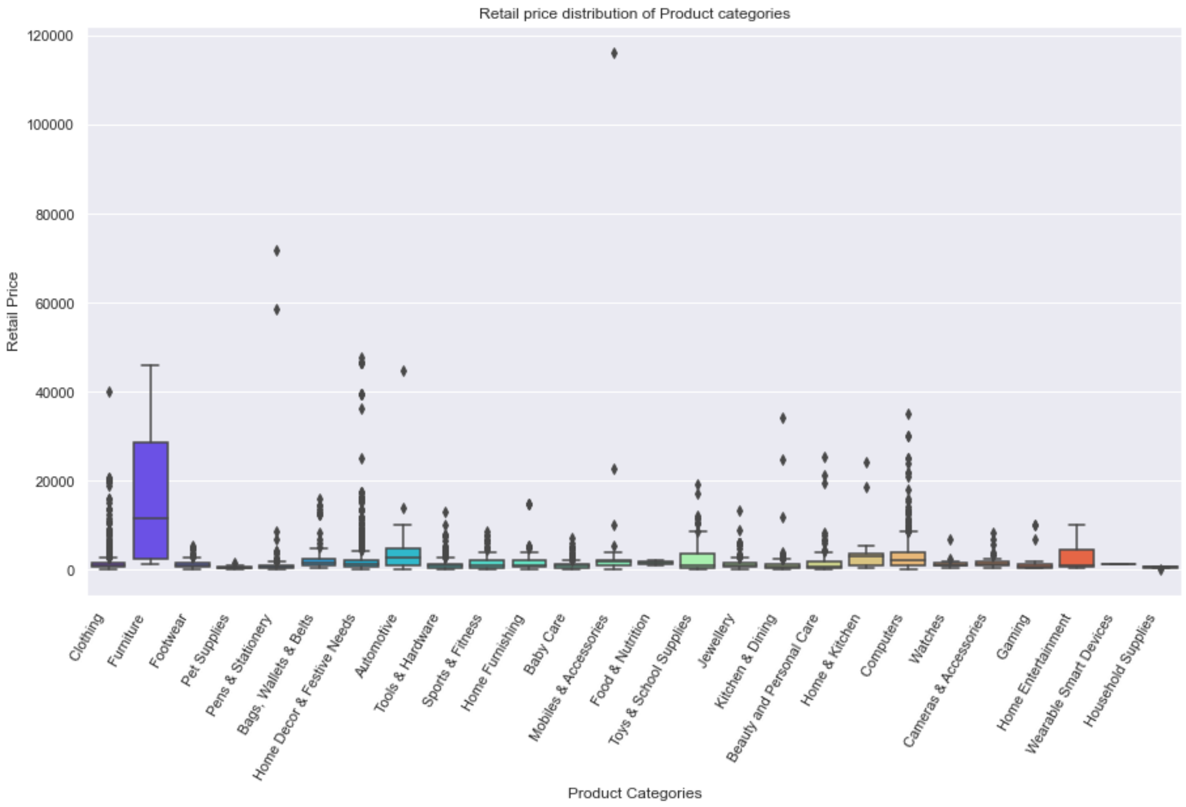
Out[98]:

	Customer_uniq_id	Region	Order_Date	Expected_Delivery_Date	Delivered_Date
0	c2d766ca982eca8304150849735ffef9	North	31-12-2021	03-01-2022	02-01-2022

Ans 7 a)

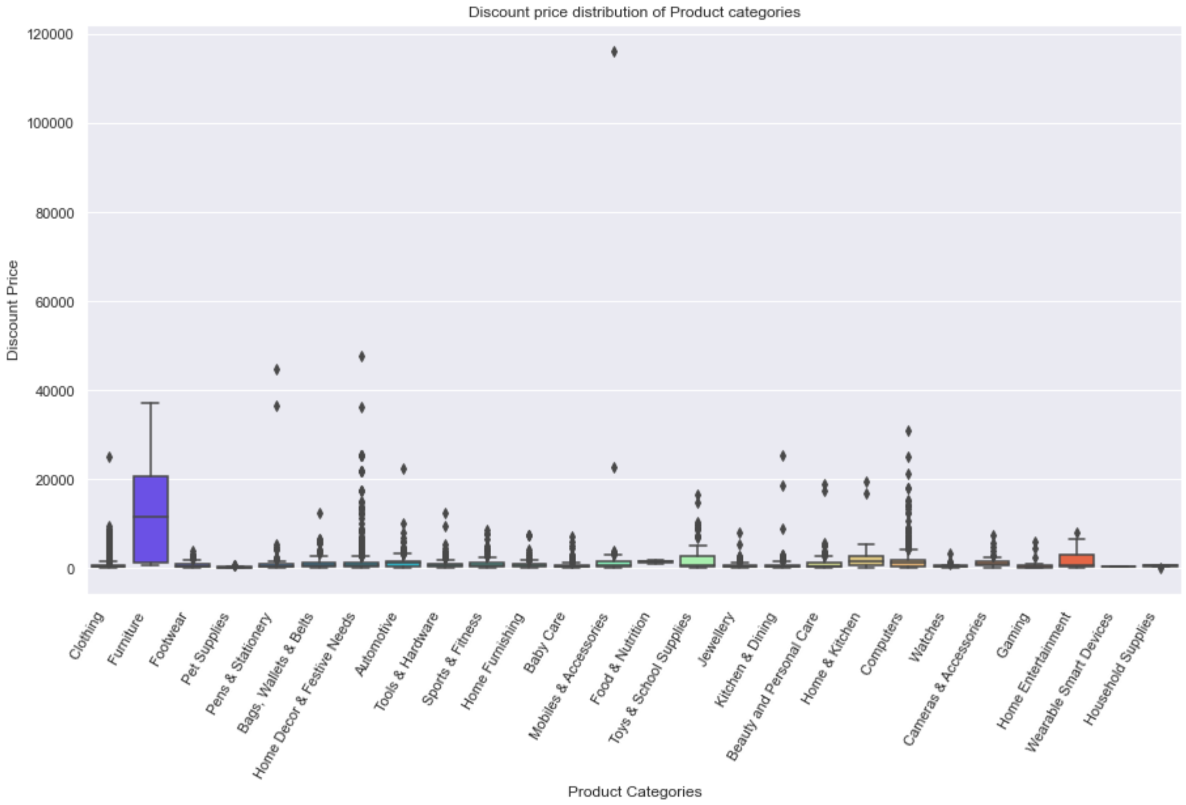
In [99]:

```
plt.figure(figsize=(15,8))
bp = sns.boxplot(x='product_main_category',y='retail_price',data=ECom_Data, palett
bp.set_xticklabels(bp.get_xticklabels(), rotation=60, ha="right")
plt.xlabel("Product Categories")
plt.ylabel("Retail Price")
plt.title("Retail price distribution of Product categories")
plt.show()
```


Retail price distribution of Product categories

In [100…

```
plt.figure(figsize=(15,8))
bp = sns.boxplot(x='product_main_category',y='discounted_price',data=ECom_Data, pa
bp.set_xticklabels(bp.get_xticklabels(), rotation=60, ha="right")
plt.xlabel("Product Categories")
plt.ylabel("Discount Price")
plt.title("Discount price distribution of Product categories")
plt.show()
```


Discount price distribution of Product categories

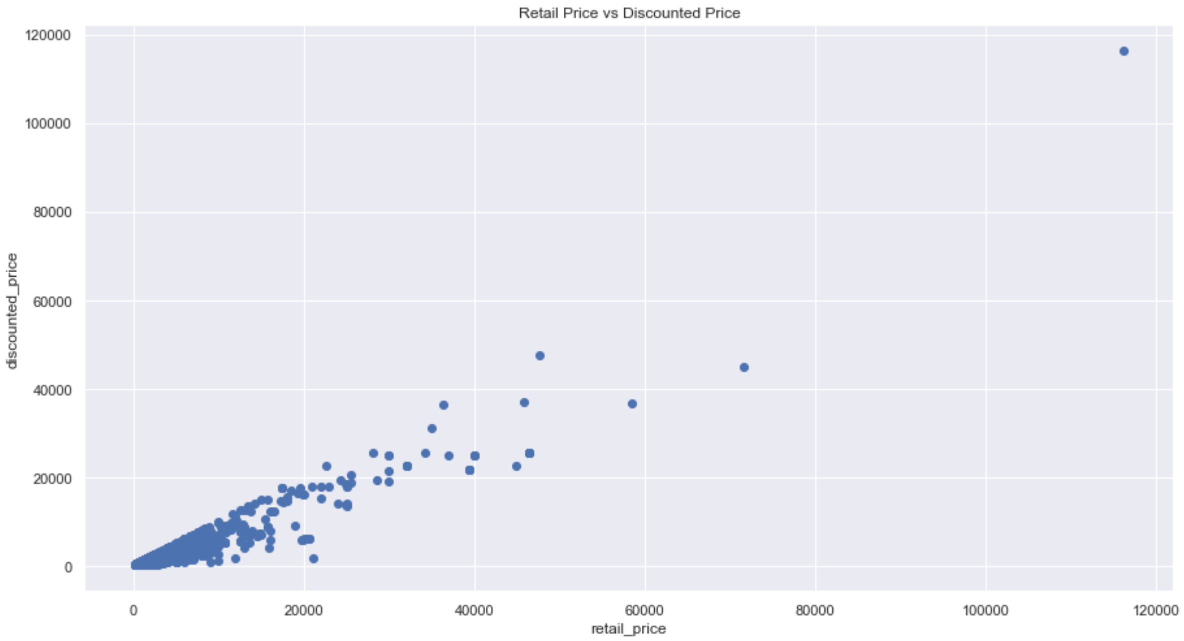
Ans 7 b)

Yes, as per the above charts, it is clear that both the Retail price and Discounted price have outliers.

Ans 7 c)

In [101…

```
plt.figure(figsize=(15,8))
plt.scatter(ECom_Data['retail_price'], ECom_Data['discounted_price'])
plt.title("Retail Price vs Discounted Price")
plt.xlabel("retail_price")
plt.ylabel("discounted_price")
plt.show()
```



The next steps will enable to study brand-level information. </font>

Q 8. Create a new dataframe to include the brand specific information as stated:

i. total number of orders placed per brand

ii. total retail_price per brand

iii. total discount_price per brand, and

iv. total BrandRevenue generated per brand.

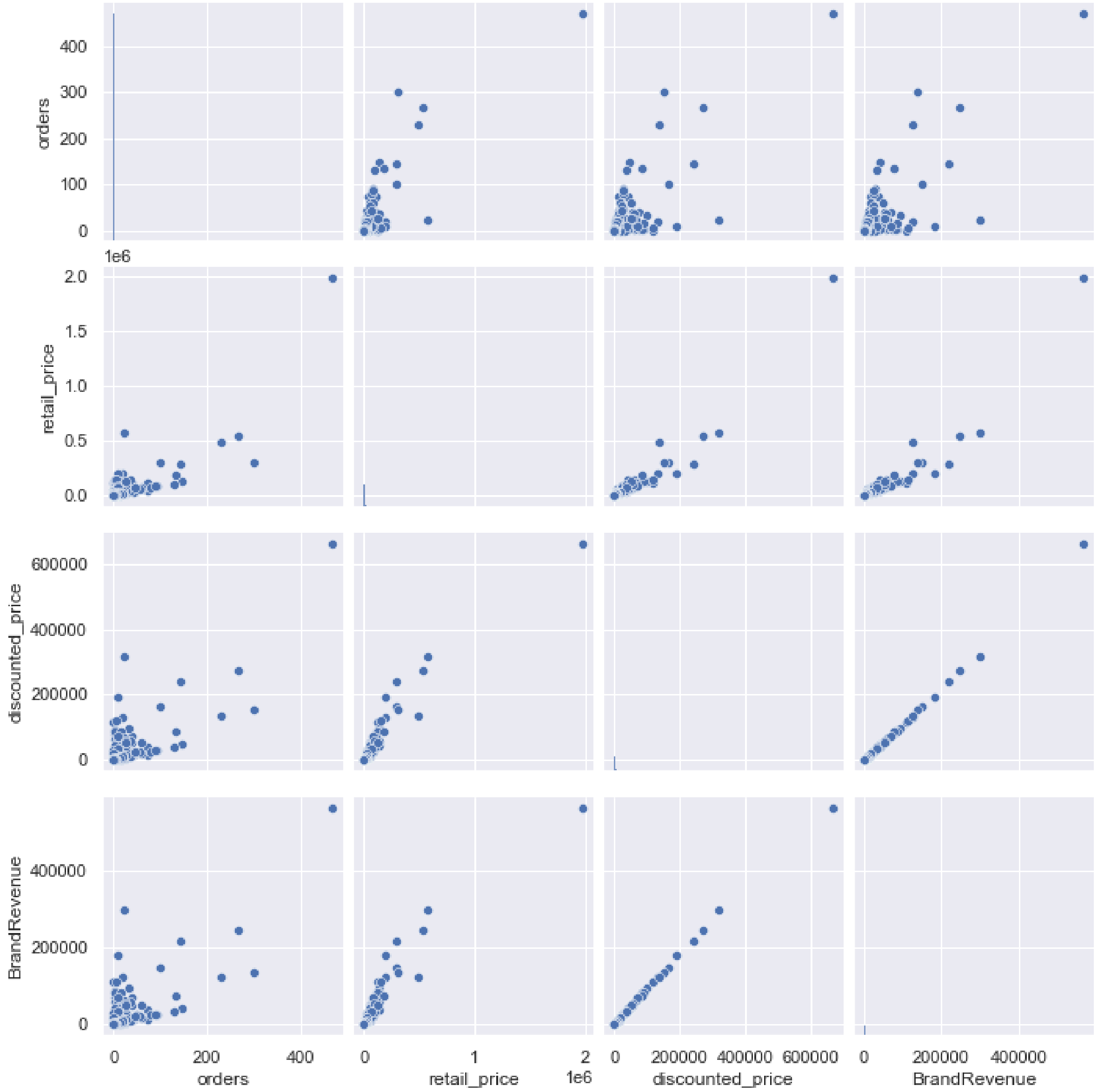
Also, draw a pairplot using these four features. (3 marks)

Ans 8

```
In [102…  net_revenue_brand_info = ECom_Data.groupby(['Brand']).agg({'Customer_uniq_id':'cou
                                                   .rename({'Customer_uniq_id':'orders'},axis=1)
          net_revenue_brand_info['Ecom_charges'] = net_revenue_brand_info.apply(total_paymen
          net_revenue_brand_info['BrandRevenue'] = net_revenue_brand_info['discounted_price'
          net_revenue_brand_info = net_revenue_brand_info[['Brand','orders','retail_price','
```

```
In [103…  sns.pairplot(net_revenue_brand_info[['orders','retail_price','discounted_price','Br
```

```
Out[103]: <seaborn.axisgrid.PairGrid at 0x1fdcc7e0c40>
```



The E-Commerce company operate in multiple regions. It is important to understand its performance in each region.

Q 9.Compare performance regionwise

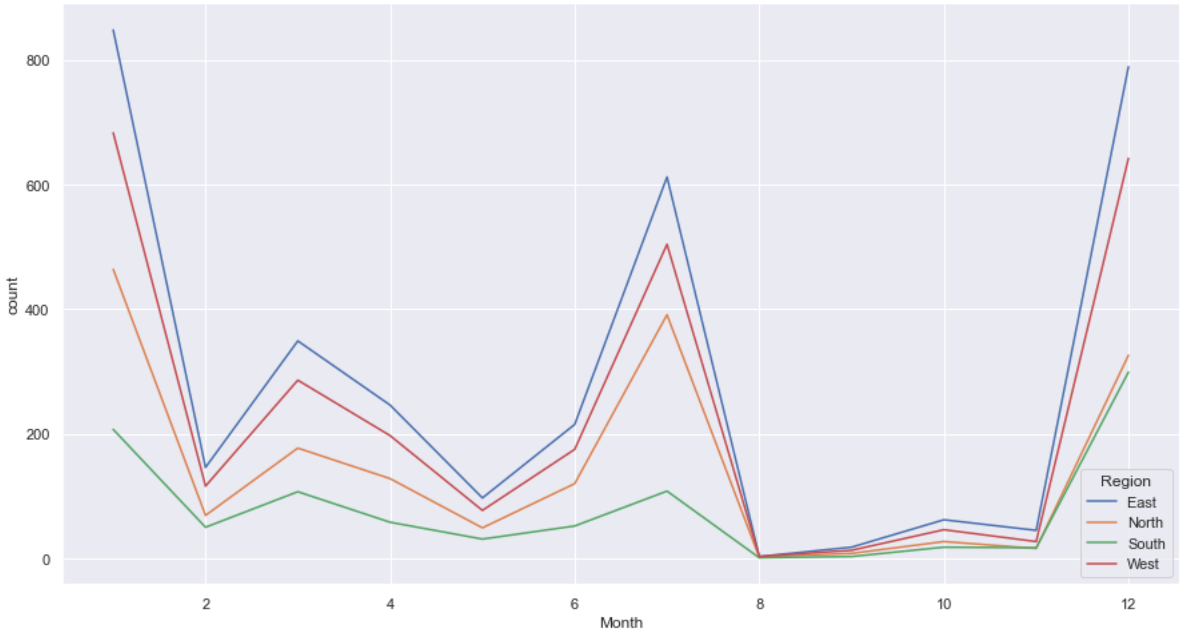
a) Draw a lineplot for the monthly Revenue of ECom Company for each region separately. (4 marks)

b) Identify the best and the worst performing months for each region. (2 marks)

Ans 9 a)

In [104…

```
data_region = copy.deepcopy(ECom_Data)
data_region['Month'] = pd.to_datetime(data_region['Order_Date']).dt.month
data_region = data_region.groupby(['Month','Region'])['Order_Date'].count().reset_
plt.figure(figsize=(15,8))
sns.lineplot(data = data_region, x = 'Month', y = 'count', hue = 'Region')
plt.show()
```



Ans 9 b)

January is the best performing month for all except South region. August is worst performing month for all regions. Whereas South region records December as it's best performing month

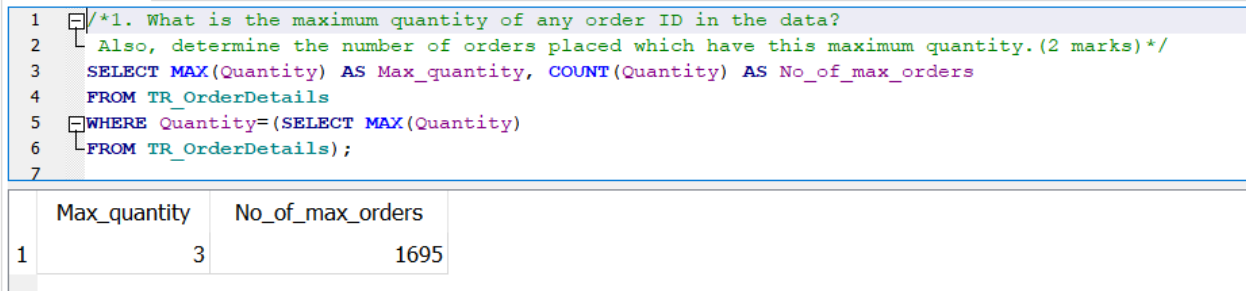
Congratulations! You have learnt how to approach a complex data and extract information out of it.

Part II – SQL (40 Marks)

Attached SQL Queries along with this report.

Panagam Mohitha

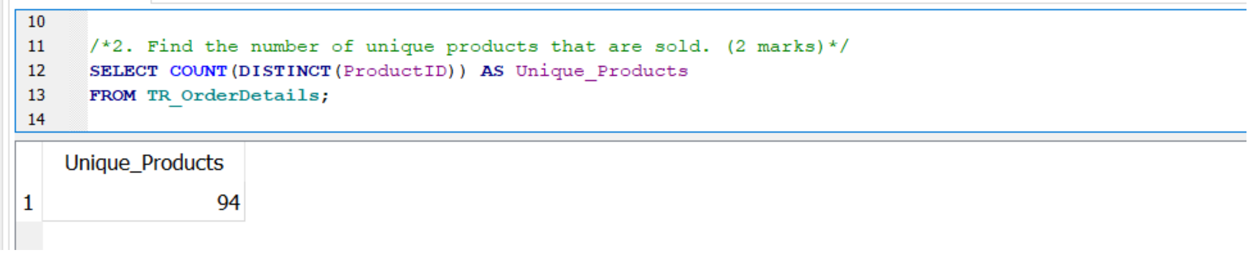
1. What is the maximum quantity of any order ID in the data? Also, determine the number of orders placed which have this maximum quantity.(2 marks)



```
/*1. What is the maximum quantity of any order ID in the data?
 Also, determine the number of orders placed which have this maximum quantity.(2 marks)*/
SELECT MAX(Quantity) AS Max_quantity, COUNT(Quantity) AS No_of_max_orders
FROM TR_OrderDetails
WHERE Quantity=(SELECT MAX(Quantity)
FROM TR_OrderDetails);
```

	Max_quantity	No_of_max_orders
1	3	1695

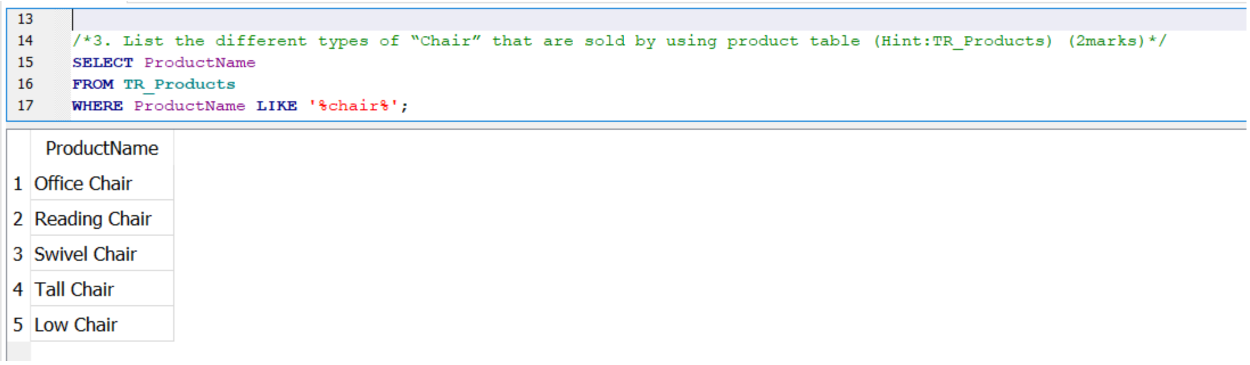
2. Find the number of unique products that are sold. (2 marks)



```
/*2. Find the number of unique products that are sold. (2 marks)*/
SELECT COUNT(DISTINCT(ProductID)) AS Unique_Products
FROM TR_OrderDetails;
```

	Unique_Products
1	94

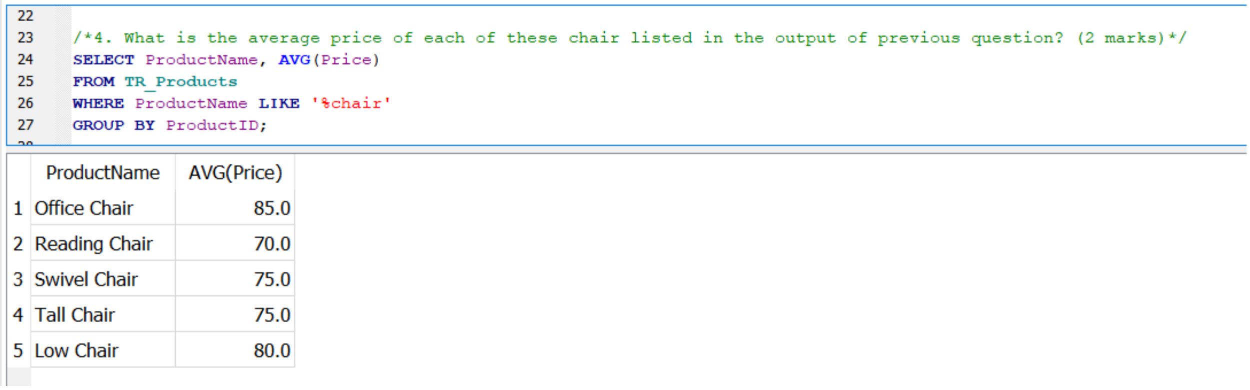
3. List the different types of "Chair" that are sold by using product table (Hint:TR_Products) (2marks)



```
/*3. List the different types of "Chair" that are sold by using product table (Hint:TR_Products) (2marks)*/
SELECT ProductName
FROM TR_Products
WHERE ProductName LIKE '%chair%';
```

	ProductName
1	Office Chair
2	Reading Chair
3	Swivel Chair
4	Tall Chair
5	Low Chair

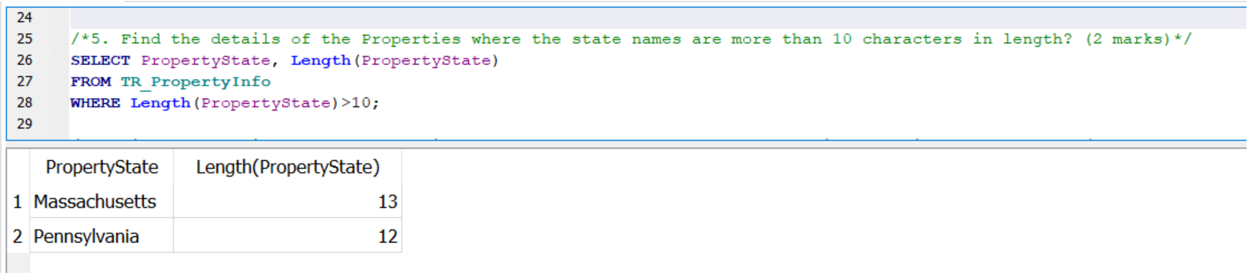
4. What is the average price of each of these chair listed in the output of previous question? (2 marks)



```
/*4. What is the average price of each of these chair listed in the output of previous question? (2 marks)*/
SELECT ProductName, AVG(Price)
FROM TR_Products
WHERE ProductName LIKE '%chair'
GROUP BY ProductID;
```

	ProductName	AVG(Price)
1	Office Chair	85.0
2	Reading Chair	70.0
3	Swivel Chair	75.0
4	Tall Chair	75.0
5	Low Chair	80.0

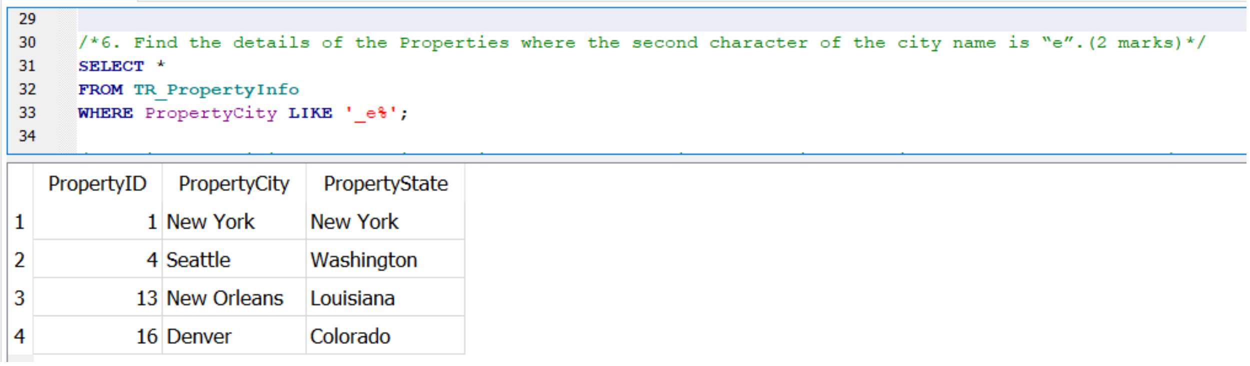
5. Find the details of the Properties where the state names are more than 10 characters in length? (2 marks)



```
/*5. Find the details of the Properties where the state names are more than 10 characters in length? (2 marks)*/
SELECT PropertyState, Length(PropertyState)
FROM TR_PropertyInfo
WHERE Length(PropertyState)>10;
```

	PropertyState	Length(PropertyState)
1	Massachusetts	13
2	Pennsylvania	12

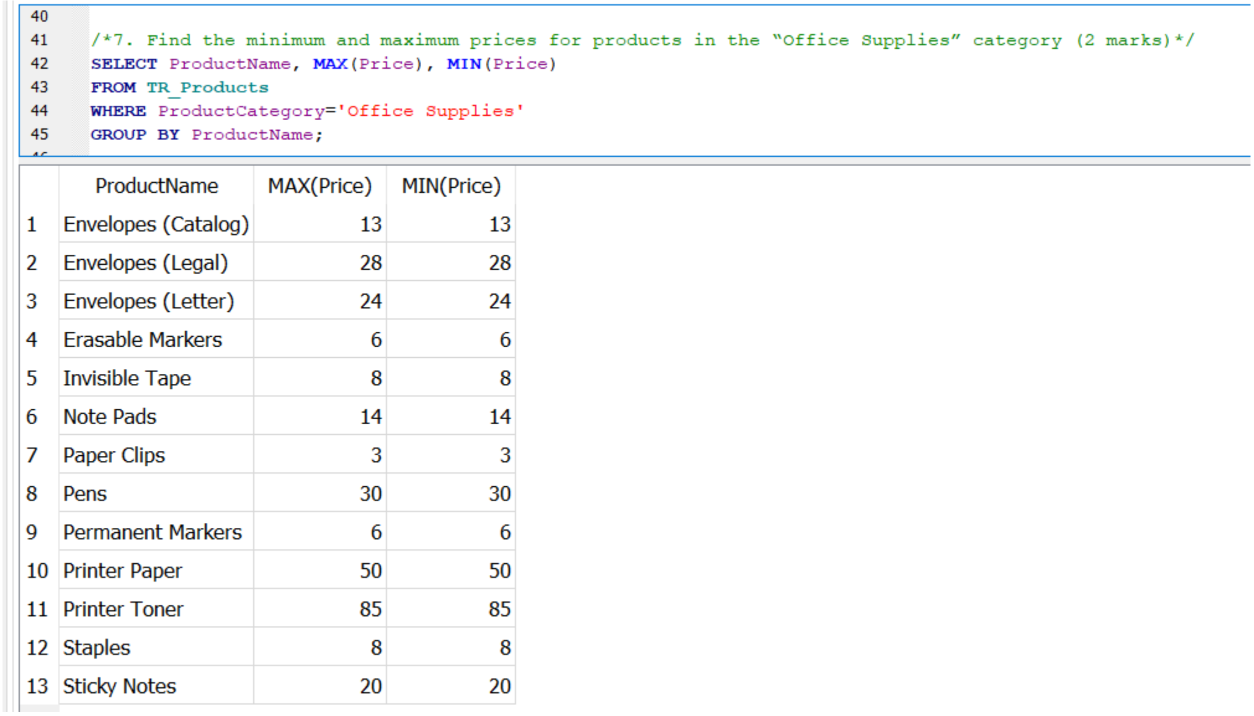
6. Find the details of the Properties where the second character of the city name is “e”.(2 marks)



```
/*6. Find the details of the Properties where the second character of the city name is “e”.(2 marks)*/
SELECT *
FROM TR_PropertyInfo
WHERE PropertyCity LIKE '_e%';
```

	PropertyID	PropertyCity	PropertyState
1	1	New York	New York
2	4	Seattle	Washington
3	13	New Orleans	Louisiana
4	16	Denver	Colorado

7. Find the minimum and maximum prices for products in the “Office Supplies” category (2 marks)



```
/*7. Find the minimum and maximum prices for products in the “Office Supplies” category (2 marks)*/
SELECT ProductName, MAX(Price), MIN(Price)
FROM TR_Products
WHERE ProductCategory='Office Supplies'
GROUP BY ProductName;
```

	ProductName	MAX(Price)	MIN(Price)
1	Envelopes (Catalog)	13	13
2	Envelopes (Legal)	28	28
3	Envelopes (Letter)	24	24
4	Erasable Markers	6	6
5	Invisible Tape	8	8
6	Note Pads	14	14
7	Paper Clips	3	3
8	Pens	30	30
9	Permanent Markers	6	6
10	Printer Paper	50	50
11	Printer Toner	85	85
12	Staples	8	8
13	Sticky Notes	20	20

8. What is the purpose of using GROUP BY in SQL? (Hint: This is a theoretical question and needs to be explained with an clear example other than the application given in this project) (2 marks)

The Group By statement is used for organizing similar data into groups. The GROUP BY Statement in SQL is used to arrange identical data into groups with the help of some functions. i.e if a particular column has same values in different rows then it will arrange these rows in a group.

For example, the GROUP BY query will be used to count the number of employees in each department, or to get the department wise total salaries.

Emp_id	Emp_Name	Salary	Dept
1001	Nishanth	30000	Sales
1002	Sumanth	25000	Marketing
1003	Amisha	25000	Marketing
1004	Durga	20000	Accounting
1005	Revati	30000	Accounting
1006	Joseph	25000	Accounting

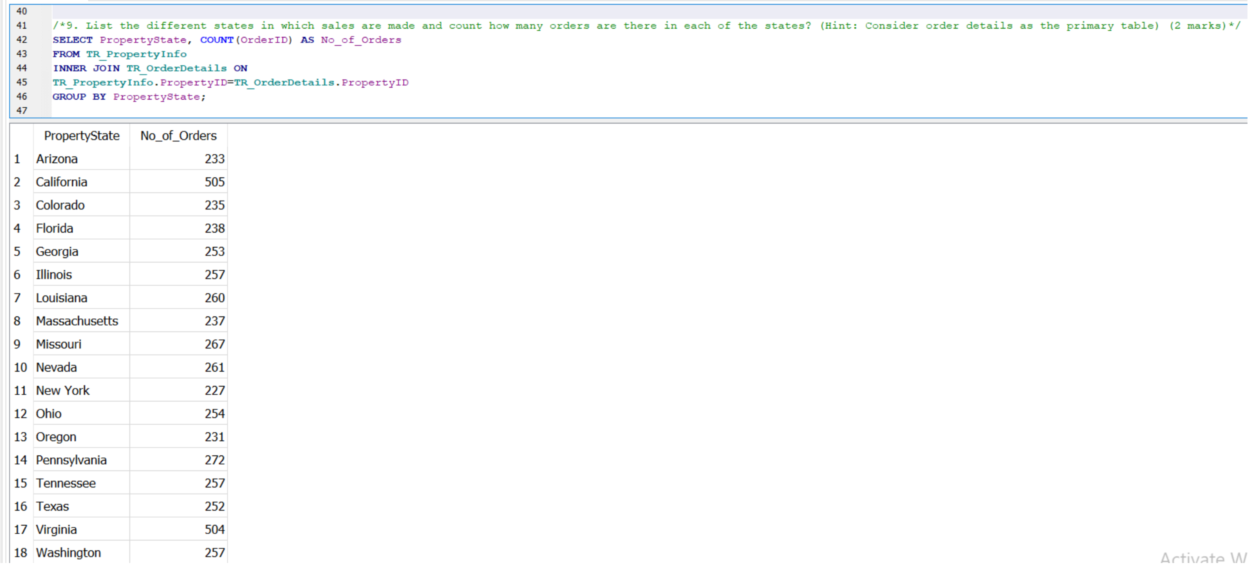
Consider the Query:

```
SELECT Dept, SUM(Salary)
 FROM Employee
GROUP BY Dept;
```

OUTPUT of above query is:

Dept	SUM(Salary)
Sales	30000
Marketing	50000
Accounting	75000

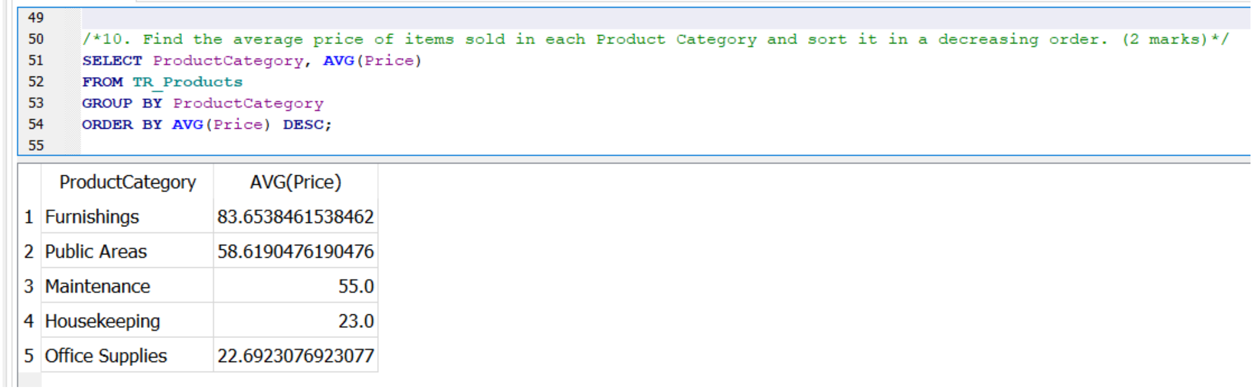
9. List the different states in which sales are made and count how many orders are there in each of the states? (Hint: Consider order details as the primary table) (2 marks)



```
/*9. List the different states in which sales are made and count how many orders are there in each of the states? (Hint: Consider order details as the primary table) (2 marks)*/
SELECT PropertyState, COUNT(OrderID) AS No_of_Orders
FROM TR_PropertyInfo
INNER JOIN TR_OrderDetails ON
TR_PropertyInfo.PropertyID=TR_OrderDetails.PropertyID
GROUP BY PropertyState;
```

	PropertyState	No_of_Orders
1	Arizona	233
2	California	505
3	Colorado	235
4	Florida	238
5	Georgia	253
6	Illinois	257
7	Louisiana	260
8	Massachusetts	237
9	Missouri	267
10	Nevada	261
11	New York	227
12	Ohio	254
13	Oregon	231
14	Pennsylvania	272
15	Tennessee	257
16	Texas	252
17	Virginia	504
18	Washington	257

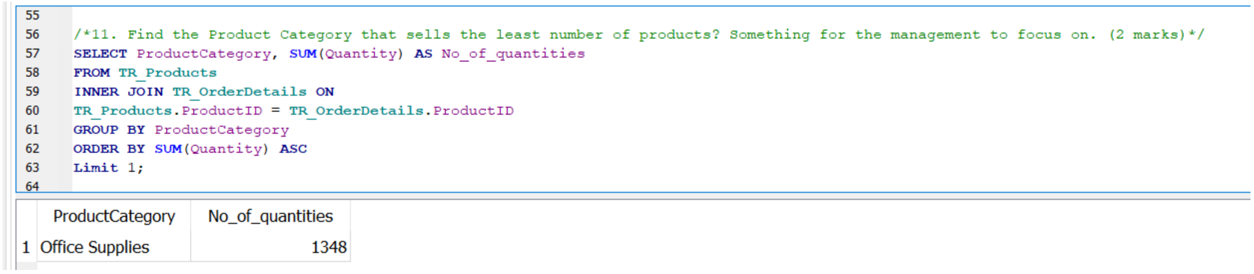
10. Find the average price of items sold in each Product Category and sort it in a decreasing order. (2 marks)



```
/*10. Find the average price of items sold in each Product Category and sort it in a decreasing order. (2 marks)*/
SELECT ProductCategory, AVG(Price)
FROM TR_Products
GROUP BY ProductCategory
ORDER BY AVG(Price) DESC;
```

	ProductCategory	AVG(Price)
1	Furnishings	83.6538461538462
2	Public Areas	58.6190476190476
3	Maintenance	55.0
4	Housekeeping	23.0
5	Office Supplies	22.6923076923077

11. Find the Product Category that sells the least number of products? Something for the management to focus on. (2 marks)



```
/*11. Find the Product Category that sells the least number of products? Something for the management to focus on. (2 marks)*/
SELECT ProductCategory, SUM(Quantity) AS No_of_quantities
FROM TR_Products
INNER JOIN TR_OrderDetails ON
TR_Products.ProductID = TR_OrderDetails.ProductID
GROUP BY ProductCategory
ORDER BY SUM(Quantity) ASC
Limit 1;
```

	ProductCategory	No_of_quantities
1	Office Supplies	1348

12. What is the difference between a WHERE v/s HAVING clause in SQL? (Hint: This is a theoretical question and needs to be explained with an clear example other than the application given in this project) (2 marks)

The main difference between them is that the WHERE clause is used to specify a condition for filtering records before any groupings are made, while the HAVING clause is used to specify a condition for filtering values from a group.

WHERE Clause is used to filter the records from the table or used while joining more than one table. Only those records will be extracted who are satisfying the specified condition in WHERE clause. HAVING Clause is used to filter the records from the groups based on the given condition in the HAVING Clause. Those groups who will satisfy the given condition will appear in the final result.

Let us consider below table “Employee”

Emp_id	**Emp_Name**	**Salary**
1001	Nishanth	30000
1002	Sumanth	25000
1003	Amisha	25000
1004	Durga	20000
1005	Revati	30000
1006	Joseph	25000

Consider the Query:

```
SELECT Emp_Name, Salary

 FROM Employee

WHERE Salary >=25000
```

OUTPUT of above query is :

Emp_Name	**Salary**
Nishanth	30000
Sumanth	25000
Amisha	25000
Revati	30000
Joseph	25000

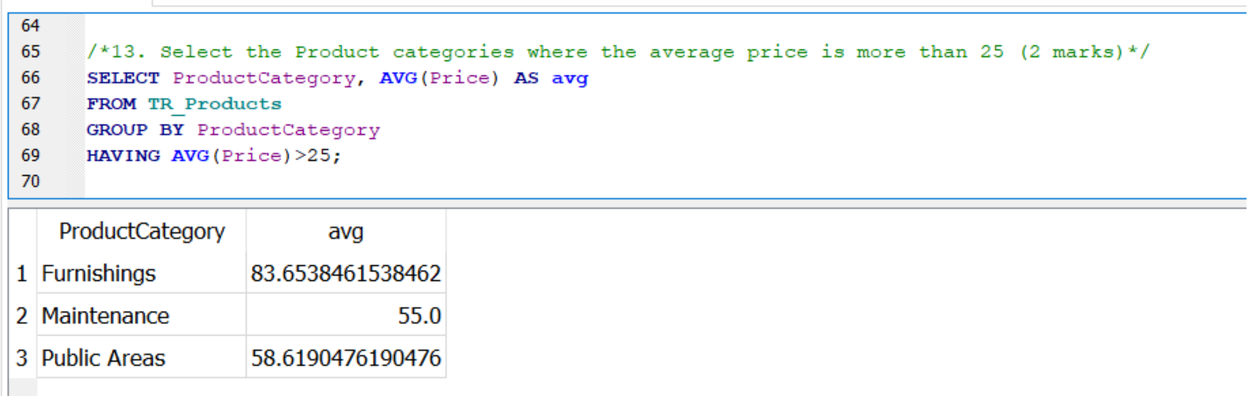
Let us consider Employee table mentioned above and apply having clause on it:

```
SELECT Salary, COUNT(Emp_id) AS No_of_Employees

FROM Employee

GROUP BY Salary

HAVING COUNT(Emp_id)>1001
```

OUTPUT of the above query is:

Salary	**No_of_Employees**
30000	2
25000	3
20000	1

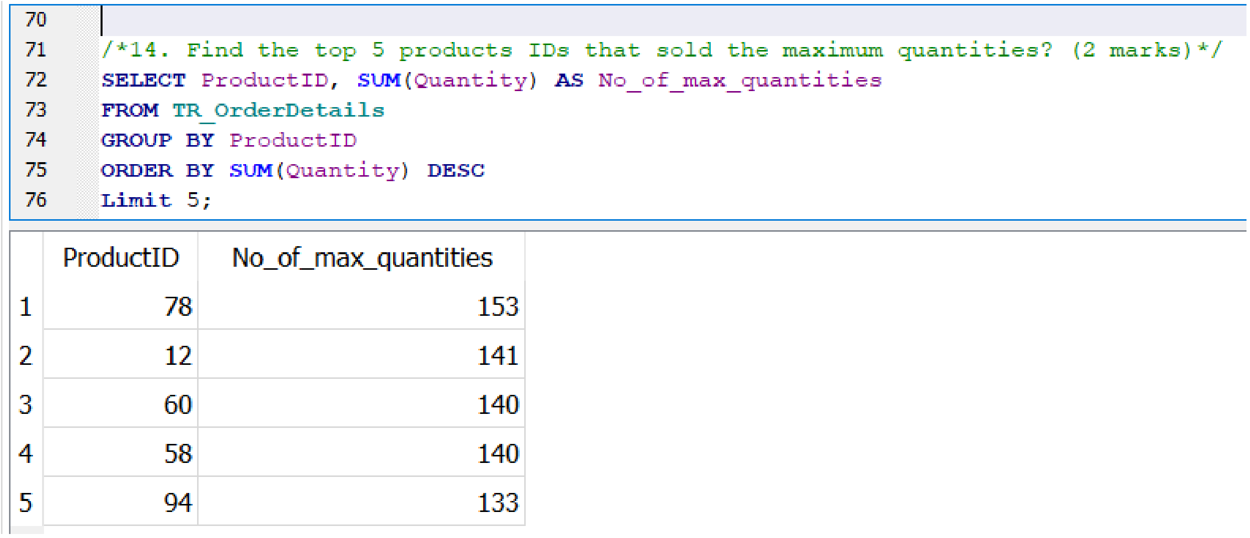
13. Select the Product categories where the average price is more than 25 (2 marks)



```
/*13. Select the Product categories where the average price is more than 25 (2 marks)*/
SELECT ProductCategory, AVG(Price) AS avg
FROM TR_Products
GROUP BY ProductCategory
HAVING AVG(Price)>25;
```

	ProductCategory	avg
1	Furnishings	83.6538461538462
2	Maintenance	55.0
3	Public Areas	58.6190476190476

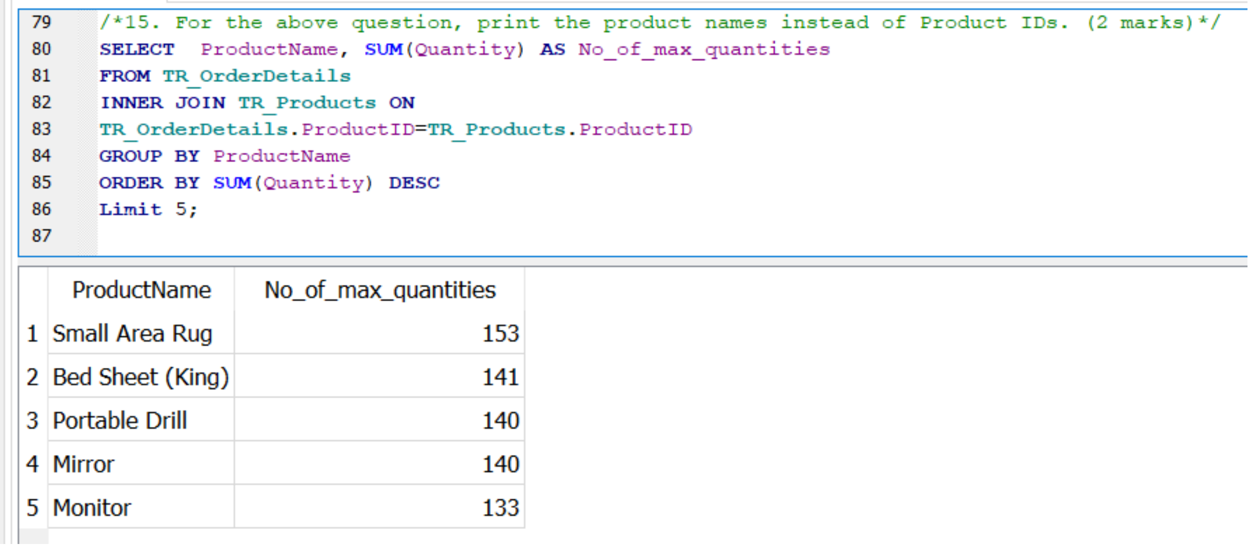
14. Find the top 5 products IDs that sold the maximum quantities? (2 marks)



```
/*14. Find the top 5 products IDs that sold the maximum quantities? (2 marks)*/
SELECT ProductID, SUM(Quantity) AS No_of_max_quantities
FROM TR_OrderDetails
GROUP BY ProductID
ORDER BY SUM(Quantity) DESC
Limit 5;
```

	ProductID	No_of_max_quantities
1	78	153
2	12	141
3	60	140
4	58	140
5	94	133

15. For the above question, print the product names instead of Product IDs. (2 marks)



```sql
/*15. For the above question, print the product names instead of Product IDs. (2 marks)*/
SELECT  ProductName, SUM(Quantity) AS No_of_max_quantities
FROM TR_OrderDetails
INNER JOIN TR_Products ON
TR_OrderDetails.ProductID=TR_Products.ProductID
GROUP BY ProductName
ORDER BY SUM(Quantity) DESC
Limit 5;
```

	ProductName	No_of_max_quantities
1	Small Area Rug	153
2	Bed Sheet (King)	141
3	Portable Drill	140
4	Mirror	140
5	Monitor	133

16. Mention the different types of joins in SQL? Give simple examples of each. Also represent them using Venn diagrams (Hint: This is a theoretical question, the explanation needs to be in detail along with an example other than the one given in this project) (2 marks)

Here are the different types of the JOINs in SQL:

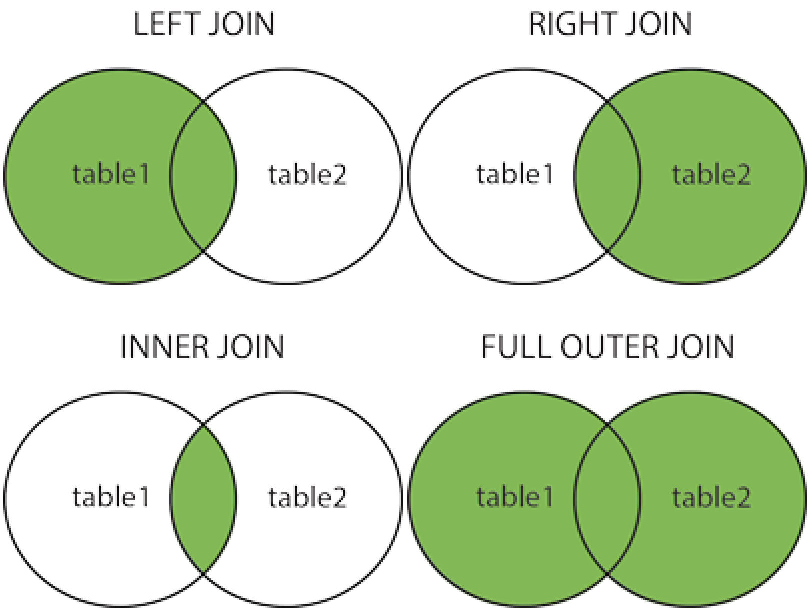
- (INNER) JOIN: Returns records that have matching values in both tables
- LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
- RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
- FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table

TABLE 1		TABLE 2	
Emp_id	**Emp_Name**	**Emp_id**	**Salary**
1001	Nishanth	1002	30000
1002	Sumanth	1003	25000
1003	Amisha	1005	25000
1004	Durga	1007	20000
1005	Revati	1008	30000

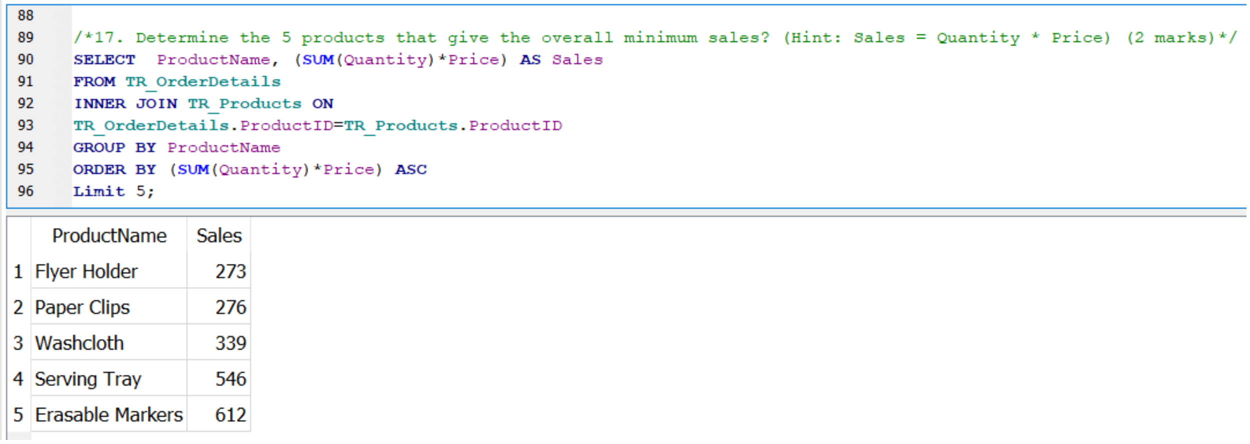
Output of the join queries of Table 1 & Table 2 are:

INNER JOIN		**FULL OUTER JOIN**	
Emp_Name	**Salary**	**Emp_Name**	**Salary**
Sumanth	30000	Sumanth	30000
Amisha	25000	Amisha	25000
Revati	25000	Revati	25000
		Nishanth	NULL
		Durga	NULL
		NULL	20000
		NULL	30000

LEFT JOIN		**RIGHT JOIN**	
Emp_Name	**Salary**	**Emp_Name**	**Salary**
Sumanth	30000	Sumanth	30000
Amisha	25000	Amisha	25000
Revati	25000	Revati	25000
Nishanth	NULL	NULL	20000
Durga	NULL	NULL	30000



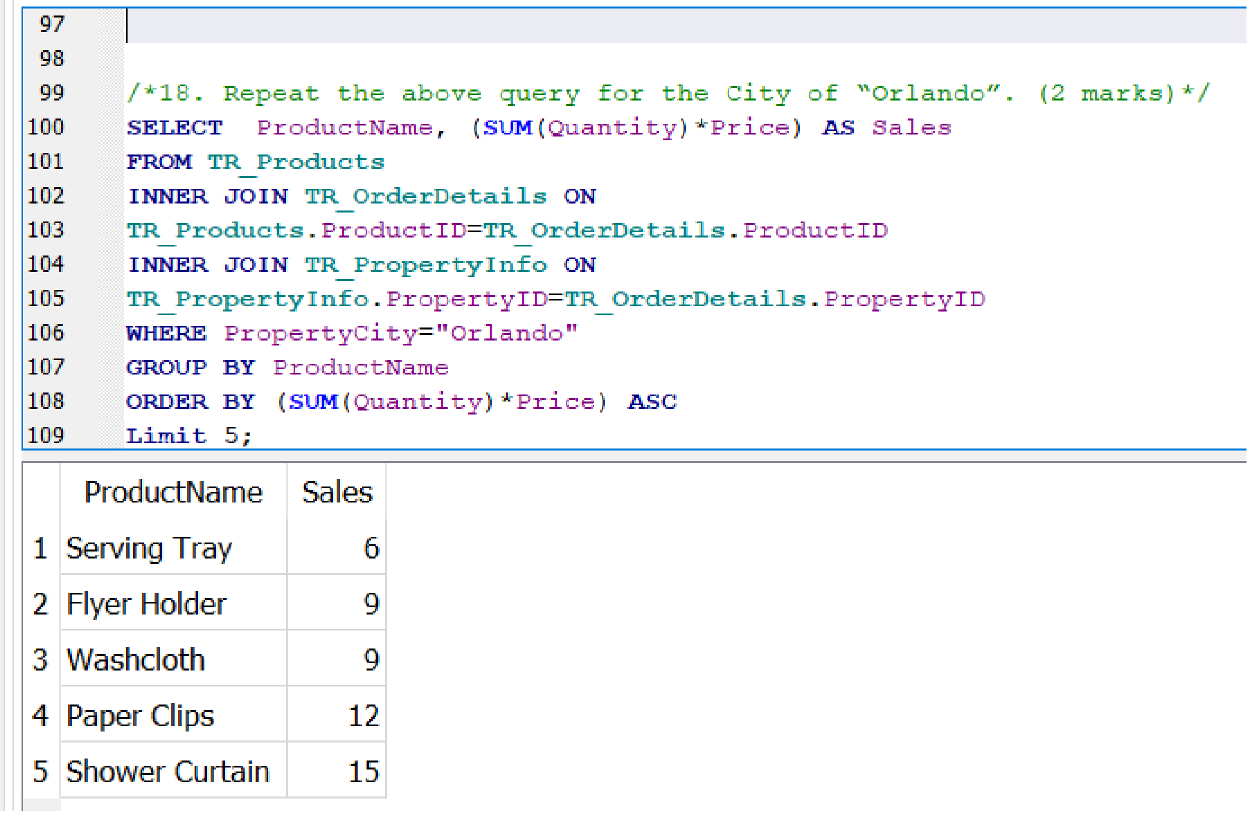
17. Determine the 5 products that give the overall minimum sales? (Hint: Sales = Quantity * Price) (2 marks)



```

/*17. Determine the 5 products that give the overall minimum sales? (Hint: Sales = Quantity * Price) (2 marks)*/
SELECT  ProductName, (SUM(Quantity)*Price) AS Sales
FROM TR_OrderDetails
INNER JOIN TR_Products ON
TR_OrderDetails.ProductID=TR_Products.ProductID
GROUP BY ProductName
ORDER BY (SUM(Quantity)*Price) ASC
Limit 5;
```

	ProductName	Sales
1	Flyer Holder	273
2	Paper Clips	276
3	Washcloth	339
4	Serving Tray	546
5	Erasable Markers	612

18. Repeat the above query for the City of “Orlando”. (2 marks)



```


/*18. Repeat the above query for the City of “Orlando”. (2 marks)*/
SELECT  ProductName, (SUM(Quantity)*Price) AS Sales
FROM TR_Products
INNER JOIN TR_OrderDetails ON
TR_Products.ProductID=TR_OrderDetails.ProductID
INNER JOIN TR_PropertyInfo ON
TR_PropertyInfo.PropertyID=TR_OrderDetails.PropertyID
WHERE PropertyCity="Orlando"
GROUP BY ProductName
ORDER BY (SUM(Quantity)*Price) ASC
Limit 5;
```

	ProductName	Sales
1	Serving Tray	6
2	Flyer Holder	9
3	Washcloth	9
4	Paper Clips	12
5	Shower Curtain	15

19. What is the difference between Drop, Truncate and Delete? Explain with examples. (2 marks)

- The **DELETE** command deletes one or more existing records from the table in the database. We can restore any deleted row or multiple rows from the database using the ROLLBACK command.
- The **DROP** Command drops the complete table from the database. We cannot get the complete table deleted from the database using the ROLLBACK command.
- The **TRUNCATE** Command deletes all the rows from the existing table, leaving the row with the column names. We cannot restore all the deleted rows from the database using the ROLLBACK command.

DROP Example

Let us consider a table Student in the database.

```
DROP TABLE Employee;
```

The above SQL Query removes all the records and drops the table Employee by removing it permanently from the database. The table removed using the DROP command can be retrieved back after execution.

DELETE Example

Let's consider the table Student with the following data.

Emp_id	Emp_Name	Salary
1001	Nishanth	30000
1002	Sumanth	25000
1003	Amisha	25000
1004	Durga	20000
1005	Revati	30000
1006	Joseph	25000

Let us understand the working of the DELETE command with the help of an example.

```
DELETE FROM Employee WHERE Salary =20000;
```

The table Employee contains multiple rows. The above SQL Query helps us filter the appropriate row that needs to be deleted by the user. The WHERE clause helps us find the row with the student whose Salary is 20000. The DELETE command deletes the row returned by the WHERE clause. In the above SQL query, the DELETE command deletes the record of Employee Durga from the table whose Salary is 20000.

The resulting table after the execution of the above SQL Query is:

Emp_id	Emp_Name	Salary
1001	Nishanth	30000
1002	Sumanth	25000
1003	Amisha	25000
1005	Revati	30000
1006	Joseph	25000

TRUNCATE Example

Let's consider the table Employee to understand the working of the TRUNCATE command in a better way.

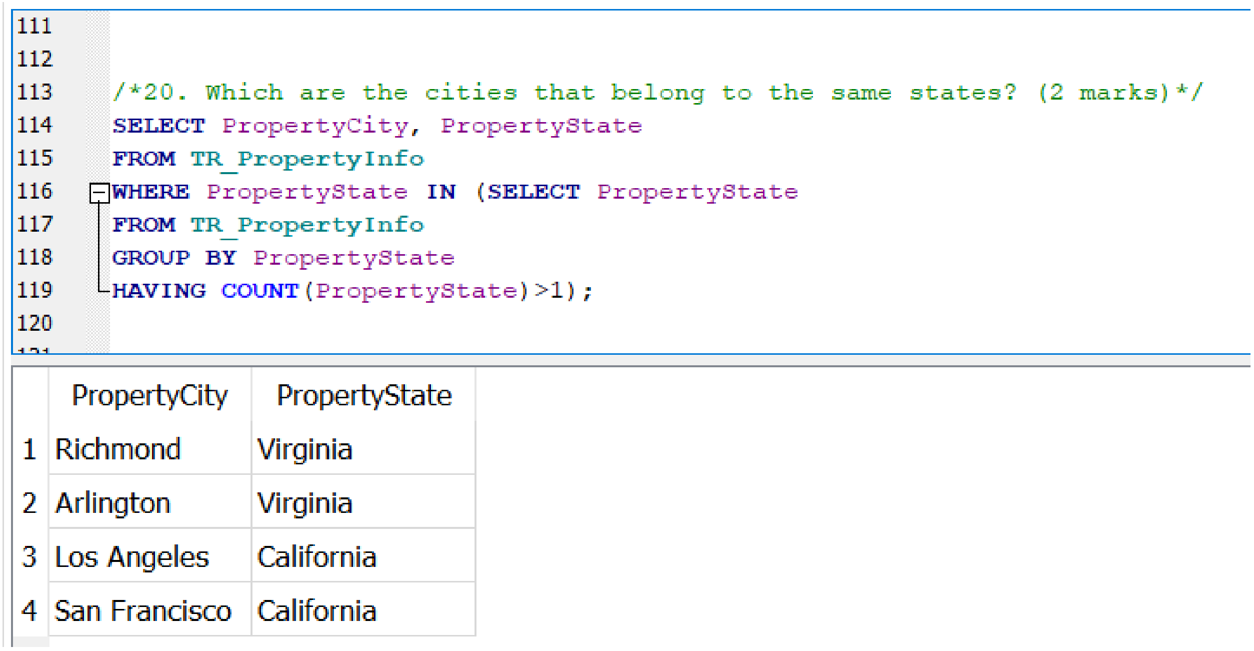
TRUNCATE TABLE Employee;

The execution of the above SQL Query deletes all the records from the Employee table, leaving only the column names. It only removes the records from the table but doesn't remove the table from the database.

The resulting table after the execution of the above SQL Query is:

Emp_id	Emp_Name	Salary

20. Which are the cities that belong to the same states? (2 marks)



```sql
/*20. Which are the cities that belong to the same states? (2 marks)*/
SELECT PropertyCity, PropertyState
FROM TR_PropertyInfo
WHERE PropertyState IN (SELECT PropertyState
FROM TR_PropertyInfo
GROUP BY PropertyState
HAVING COUNT(PropertyState)>1);
```

	PropertyCity	PropertyState
1	Richmond	Virginia
2	Arlington	Virginia
3	Los Angeles	California
4	San Francisco	California

Appendix PDF of SQL Queries

```sql
/*1. What is the maximum quantity of any order ID in the data?
 Also, determine the number of orders placed which have this maximum quantity.(2 marks)*/
SELECT MAX(Quantity) AS Max_quantity, COUNT(Quantity) AS No_of_max_orders
FROM TR_OrderDetails
WHERE Quantity=(SELECT MAX(Quantity)
FROM TR_OrderDetails);

/*2. Find the number of unique products that are sold. (2 marks)*/
SELECT COUNT(DISTINCT(ProductID)) AS Unique_Products
FROM TR_OrderDetails;

/*3. List the different types of “Chair” that are sold by using product table (Hint:TR_Products) (2marks)*/
SELECT ProductName
FROM TR_Products
WHERE ProductName LIKE '%chair%';

/*4. What is the average price of each of these chair listed in the output of previous question? (2 marks)*/
SELECT ProductName, AVG(Price)
FROM TR_Products
WHERE ProductName LIKE '%chair'
GROUP BY ProductID;

/*5. Find the details of the Properties where the state names are more than 10 characters in length? (2 marks)*/
SELECT PropertyState, Length(PropertyState)
FROM TR_PropertyInfo
WHERE Length(PropertyState)>10;

/*6. Find the details of the Properties where the second character of the city name is “e”.(2 marks)*/
SELECT *
FROM TR_PropertyInfo
WHERE PropertyCity LIKE '_e%';

/*7. Find the minimum and maximum prices for products in the “Office Supplies” category (2 marks)*/
SELECT ProductName, MAX(Price), MIN(Price)
FROM TR_Products
WHERE ProductCategory='Office Supplies'
GROUP BY ProductName;

/*9. List the different states in which sales are made and count how many orders are there in each of the states? (Hint: Consider order details as the primary table) (2 marks)*/
SELECT PropertyState, COUNT(OrderID) AS No_of_Orders
FROM TR_PropertyInfo
INNER JOIN TR_OrderDetails ON
TR_PropertyInfo.PropertyID=TR_OrderDetails.PropertyID
GROUP BY PropertyState;

/*10. Find the average price of items sold in each Product Category and sort it in a decreasing order. (2 marks)*/
SELECT ProductCategory, AVG(Price)
FROM TR_Products
GROUP BY ProductCategory
ORDER BY AVG(Price) DESC;

/*11. Find the Product Category that sells the least number of products? Something for the management to focus on. (2 marks)*/
SELECT ProductCategory, SUM(Quantity) AS No_of_quantities
FROM TR_Products
INNER JOIN TR_OrderDetails ON
TR_Products.ProductID = TR_OrderDetails.ProductID
GROUP BY ProductCategory
ORDER BY SUM(Quantity) ASC
Limit 1;

/*13. Select the Product categories where the average price is more than 25 (2 marks)*/
SELECT ProductCategory, AVG(Price) AS avg
FROM TR_Products
GROUP BY ProductCategory
HAVING AVG(Price)>25;
```

```sql

/*14. Find the top 5 products IDs that sold the maximum quantities? (2 marks)*/
SELECT ProductID, SUM(Quantity) AS No_of_max_quantities
FROM TR_OrderDetails
GROUP BY ProductID
ORDER BY SUM(Quantity) DESC
Limit 5;

/*15. For the above question, print the product names instead of Product IDs. (2 marks)*/
SELECT  ProductName, SUM(Quantity) AS No_of_max_quantities
FROM TR_OrderDetails
INNER JOIN TR_Products ON
TR_OrderDetails.ProductID=TR_Products.ProductID
GROUP BY ProductName
ORDER BY SUM(Quantity) DESC
Limit 5;

/*17. Determine the 5 products that give the overall minimum sales? (Hint: Sales = Quantity * Price) (2 marks)*/
SELECT  ProductName, (SUM(Quantity)*Price) AS Sales
FROM TR_OrderDetails
INNER JOIN TR_Products ON
TR_OrderDetails.ProductID=TR_Products.ProductID
GROUP BY ProductName
ORDER BY (SUM(Quantity)*Price) ASC
Limit 5;

/*18. Repeat the above query for the City of “Orlando”. (2 marks)*/
SELECT  ProductName, (SUM(Quantity)*Price) AS Sales
FROM TR_Products
INNER JOIN TR_OrderDetails ON
TR_Products.ProductID=TR_OrderDetails.ProductID
INNER JOIN TR_PropertyInfo ON
TR_PropertyInfo.PropertyID=TR_OrderDetails.PropertyID
WHERE PropertyCity="Orlando"
GROUP BY ProductName
ORDER BY (SUM(Quantity)*Price) ASC
Limit 5;

/*20. Which are the cities that belong to the same states? (2 marks)*/
SELECT PropertyCity, PropertyState
FROM TR_PropertyInfo
WHERE PropertyState IN (SELECT PropertyState
FROM TR_PropertyInfo
GROUP BY PropertyState
HAVING COUNT(PropertyState)>1);

```