Customer Shopping Behaviour Analysis

1.Project Overview

This project focuses on analyzing customer shopping behavior using transactional data containing approximately 3,900 purchase records across different product categories. The objective is to explore customer spending patterns, purchasing frequency, product preferences, and subscription behavior.

By examining key variables such as purchase amount, product category, customer demographics, review ratings, and shipping type, the analysis aims to uncover meaningful insights that can support data-driven business decisions. The findings help identify high-performing products, customer segments, and revenue trends, enabling businesses to improve marketing strategies, enhance customer retention, and optimize overall performance.

2.Dataset Summary

The dataset contains transactional records of customer purchases collected for shopping behavior analysis. It includes approximately ***3,900 rows*** *and multiple features describing customer demographics, purchase details, and engagement patterns.*

Key Characteristics:

- ***Total Records:*** *~3,900 transactions*
- ***Number of Columns:*** *18–20 attributes*
- ***Data Type:*** *Structured transactional dataset*
- ***Missing Values:*** *A small number of missing values(37) observed in the review_rating column*

Main Variables Included:

- ***Customer Information:***
  - *Customer ID*
  - *Age*
  - *Gender*
  - *Location*

- ***Purchase Details:***
    - *Item Purchased*
    - *Category*
    - *Purchase Amount*
    - *Previous Purchases*
    - *Frequency of Purchases*
- ***Customer Engagement & Behavior:***
    - *Review Rating*
    - *Subscription Status*
    - *Discount Applied*
    - *Promo Code Used*
    - *Shipping Type*
    - *Payment Method*
    - *Season*
    - *Product Size and Color*

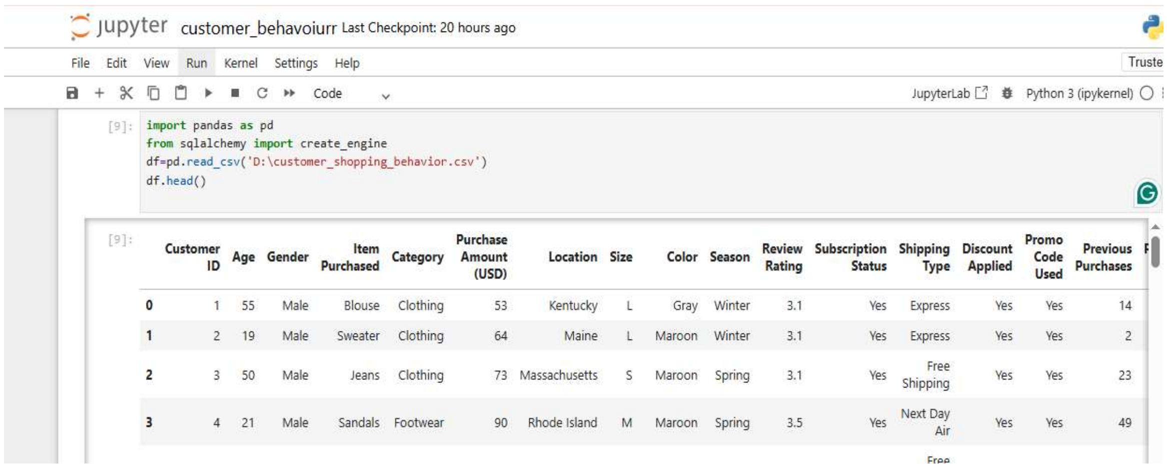
This dataset enables analysis of customer spending habits, discount effectiveness, subscription impact on revenue, product performance, and seasonal trends.

3. Exploratory Data Analysis Using Python

Exploratory Data Analysis (EDA) was performed using Python to understand the structure, distribution, and patterns within the dataset before conducting advanced analysis.

Data Inspection & Cleaning

- *Loaded dataset using panda*


Jupyter customer_behavoiurr Last Checkpoint: 20 hours ago
File Edit View Run Kernel Settings Help
Truste
Code
JupyterLab
Python 3 (ipykernel)

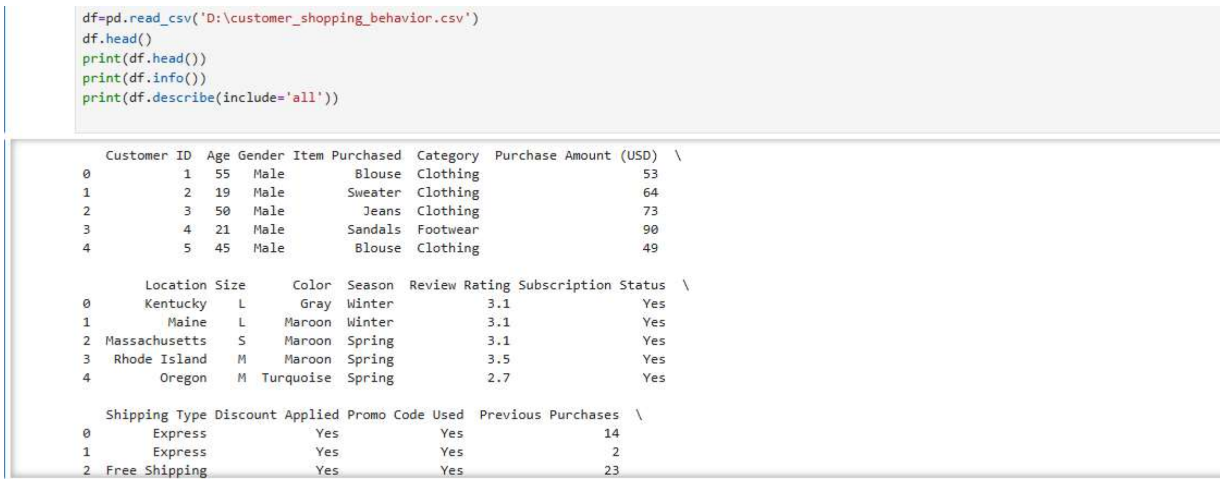
```
[9]: import pandas as pd
     from sqlalchemy import create_engine
     df=pd.read_csv('D:\customer_shopping_behavior.csv')
     df.head()
```

[9]:

	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	Promo Code Used	Previous Purchases
0	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winter	3.1	Yes	Express	Yes	Yes	14
1	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winter	3.1	Yes	Express	Yes	Yes	2
2	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Spring	3.1	Yes	Free Shipping	Yes	Yes	23
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5	Yes	Next Day Air	Yes	Yes	49
													Free			

s

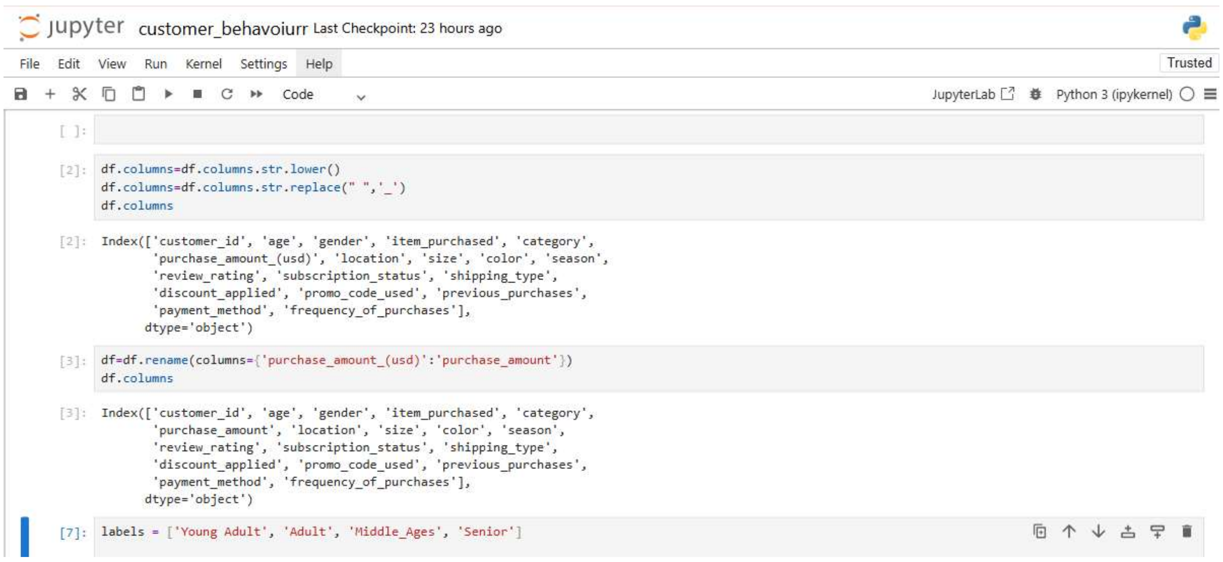
-
- *Checked data types using .info()*
- *Examined summary statistics using .describe()*



```
df=pd.read_csv('D:\customer_shopping_behavior.csv')
df.head()
print(df.head())
print(df.info())
print(df.describe(include='all'))
```

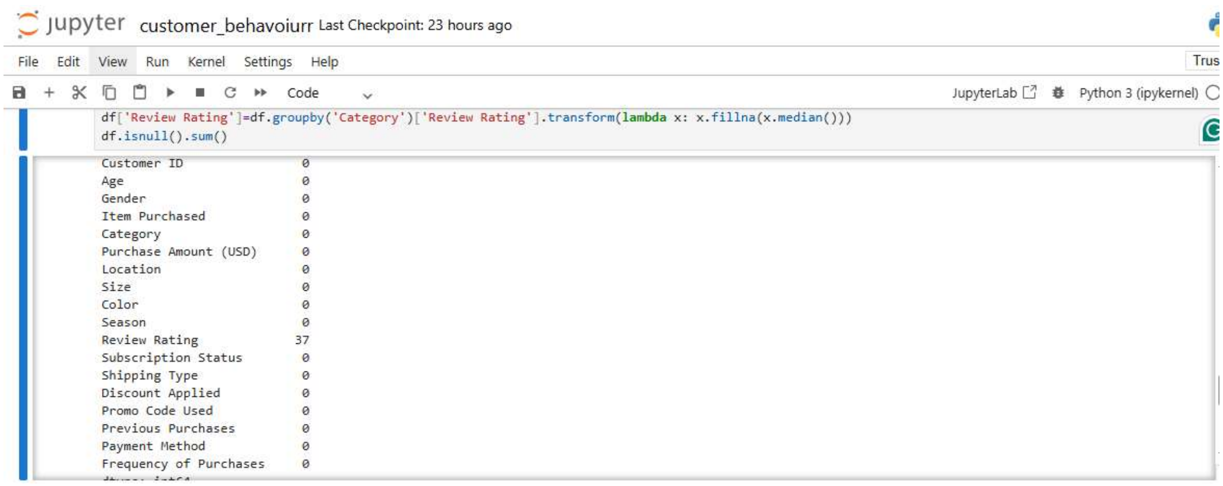
```
   Customer ID  Age Gender Item Purchased  Category  Purchase Amount (USD)  \
0            1   55   Male         Blouse  Clothing                     53   
1            2   19   Male        Sweater  Clothing                     64   
2            3   50   Male          Jeans  Clothing                     73   
3            4   21   Male        Sandals  Footwear                     90   
4            5   45   Male         Blouse  Clothing                     49   

        Location Size      Color  Season  Review Rating Subscription Status  \
0       Kentucky    L       Gray  Winter            3.1                 Yes   
1          Maine    L     Maroon  Winter            3.1                 Yes   
2  Massachusetts    S     Maroon  Spring            3.1                 Yes   
3   Rhode Island    M     Maroon  Spring            3.5                 Yes   
4         Oregon    M  Turquoise  Spring            2.7                 Yes   

   Shipping Type Discount Applied Promo Code Used  Previous Purchases  \
0        Express              Yes             Yes                  14   
1        Express              Yes             Yes                   2   
2  Free Shipping              Yes             Yes                  23   
```

- *Identified missing values using .isnull().sum()*
- *Converted numerical columns (e.g., purchase_amount) to appropriate numeric format*

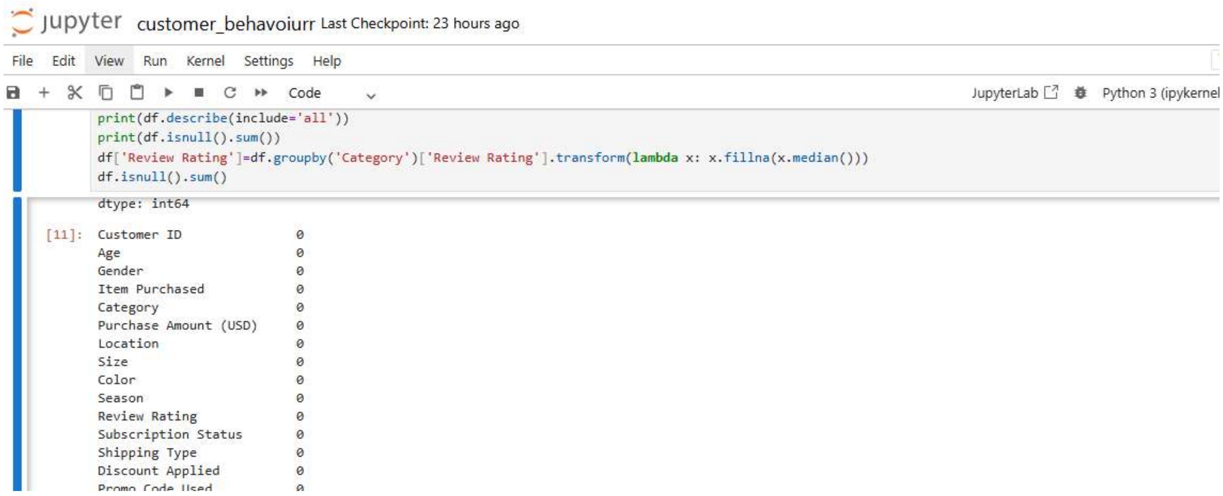


```
[2]: df.columns=df.columns.str.lower()
     df.columns=df.columns.str.replace(" ",'_')
     df.columns

[2]: Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
            'purchase_amount_(usd)', 'location', 'size', 'color', 'season',
            'review_rating', 'subscription_status', 'shipping_type',
            'discount_applied', 'promo_code_used', 'previous_purchases',
            'payment_method', 'frequency_of_purchases'],
           dtype='object')

[3]: df=df.rename(columns={'purchase_amount_(usd)':'purchase_amount'})
     df.columns

[3]: Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
            'purchase_amount', 'location', 'size', 'color', 'season',
            'review_rating', 'subscription_status', 'shipping_type',
            'discount_applied', 'promo_code_used', 'previous_purchases',
            'payment_method', 'frequency_of_purchases'],
           dtype='object')

[7]: labels = ['Young Adult', 'Adult', 'Middle_Ages', 'Senior']
```

- *Handled minor missing values in review_rating*



```
df['Review Rating']=df.groupby('Category')['Review Rating'].transform(lambda x: x.fillna(x.median()))
df.isnull().sum()

Customer ID               0
Age                       0
Gender                    0
Item Purchased            0
Category                  0
Purchase Amount (USD)     0
Location                  0
Size                      0
Color                     0
Season                    0
Review Rating            37
Subscription Status       0
Shipping Type             0
Discount Applied          0
Promo Code Used           0
Previous Purchases        0
Payment Method            0
Frequency of Purchases    0
```



```
print(df.describe(include='all'))
print(df.isnull().sum())
df['Review Rating']=df.groupby('Category')['Review Rating'].transform(lambda x: x.fillna(x.median()))
df.isnull().sum()

dtype: int64

[11]: Customer ID               0
      Age                       0
      Gender                    0
      Item Purchased            0
      Category                  0
      Purchase Amount (USD)     0
      Location                  0
      Size                      0
      Color                     0
      Season                    0
      Review Rating             0
      Subscription Status       0
      Shipping Type             0
      Discount Applied          0
      Promo Code Used           0
```

Column Standardization: Rename the column in snake cases for better readability and documentation.

Feature Engineering:

Create age_group column and frequency_purchase_days column from ages and purchase data.

Data Consistency check: Verified if discount_applied and promo_code_used were redundant drop promo_code_used.

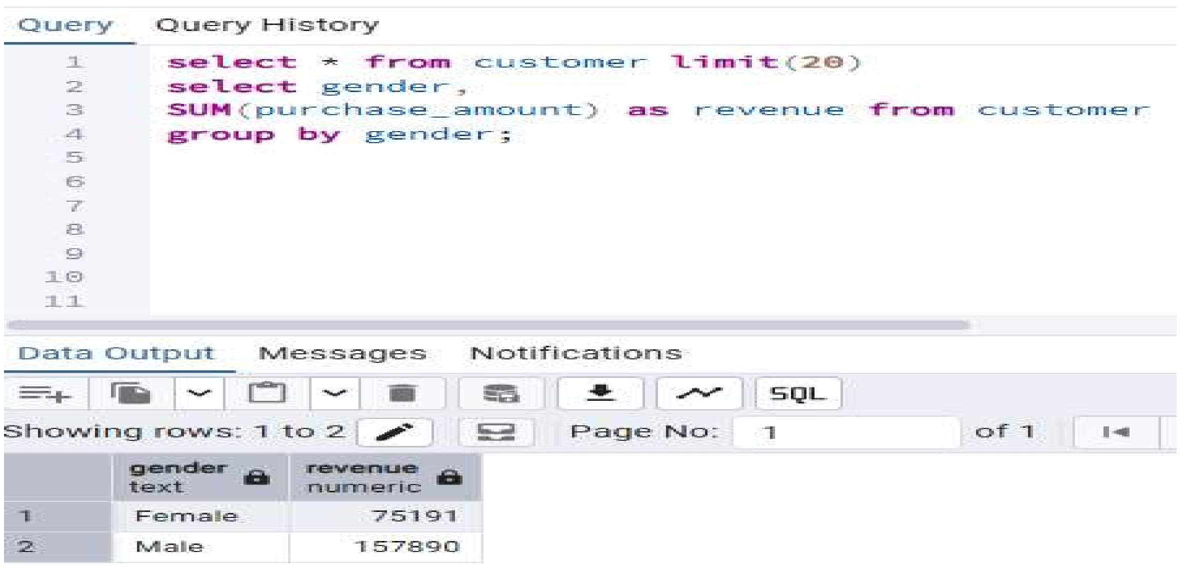
Database integration: Connect python script to PostgreSQL and loaded the cleaned data frame into database for SQL analysis

4.Data Analysis Using SQL

We performed structured analysis in PostgreSQL to answer the key business questions

1.Revenue by Genger:

Compare total revenue generated by male vs female costumes

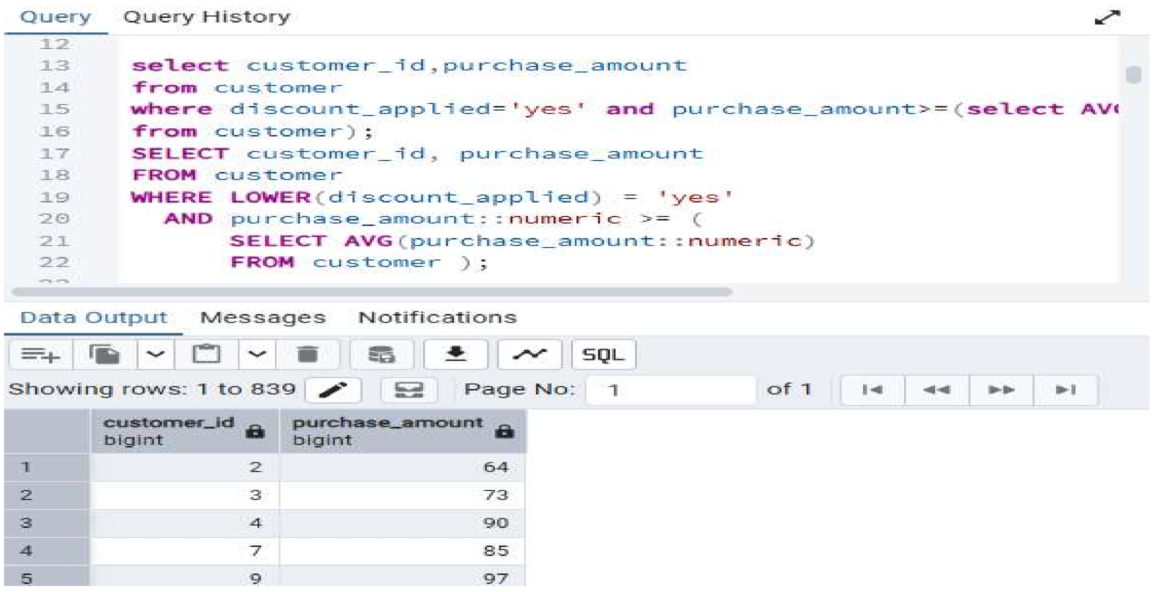

```
select * from customer limit(20)
select gender,
SUM(purchase_amount) as revenue from customer
group by gender;
```

Showing rows: 1 to 2   Page No: 1 of 1

	gender text	revenue numeric
1	Female	75191
2	Male	157890

2.High Spending Discount User:

Identifies the customers who uses discounts but still spend above the average purchase amount.



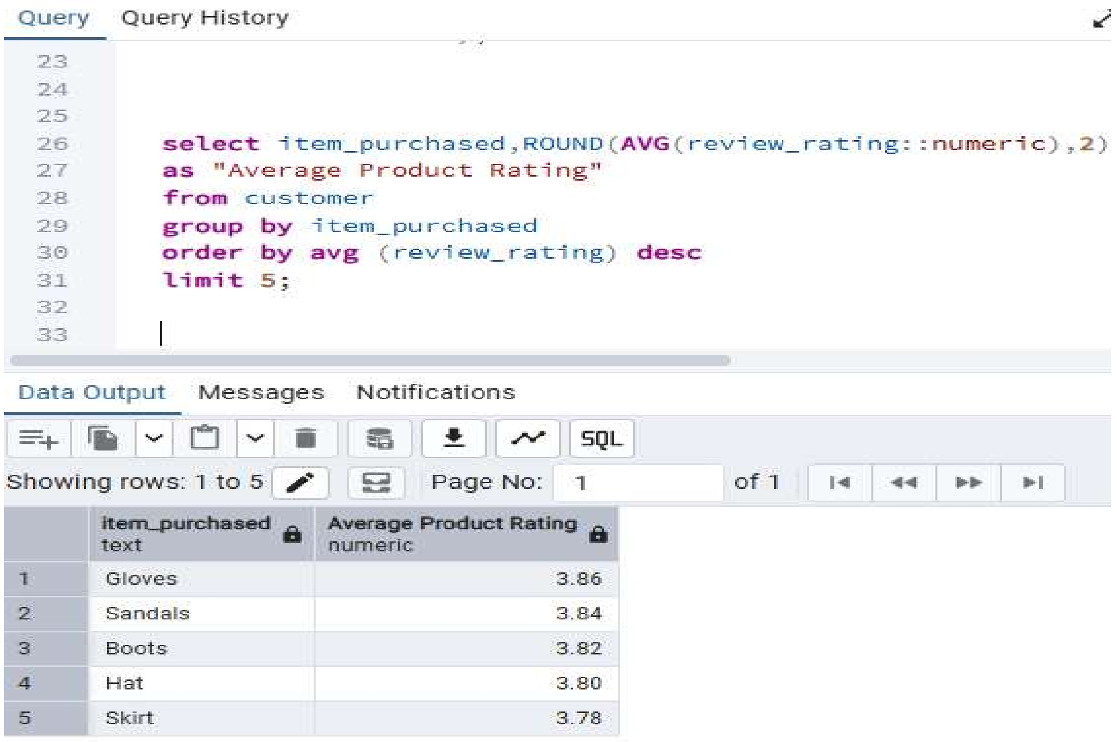
```
select customer_id,purchase_amount
from customer
where discount_applied='yes' and purchase_amount>=(select AV
from customer);
SELECT customer_id, purchase_amount
FROM customer
WHERE LOWER(discount_applied) = 'yes'
  AND purchase_amount::numeric >= (
      SELECT AVG(purchase_amount::numeric)
      FROM customer );
```

Showing rows: 1 to 839   Page No: 1 of 1

	customer_id bigint	purchase_amount bigint
1	2	64
2	3	73
3	4	90
4	7	85
5	9	97

3.Top 5 Product by Rating:

Find product with highest average review rating.



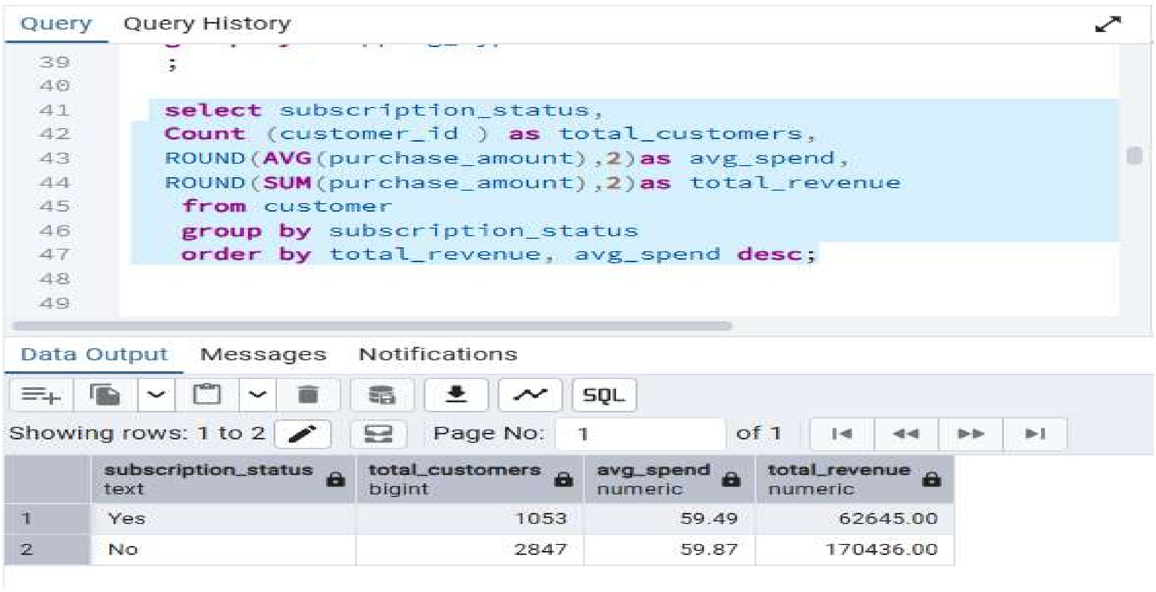
```
select item_purchased,ROUND(AVG(review_rating::numeric),2)
as "Average Product Rating"
from customer
group by item_purchased
order by avg (review_rating) desc
limit 5;
```

Showing rows: 1 to 5   Page No: 1 of 1

	item_purchased text	Average Product Rating numeric
1	Gloves	3.86
2	Sandals	3.84
3	Boots	3.82
4	Hat	3.80
5	Skirt	3.78

4.Subscribers vs non-Subscribers:

Compared average spend and total revenue across the subscription status.



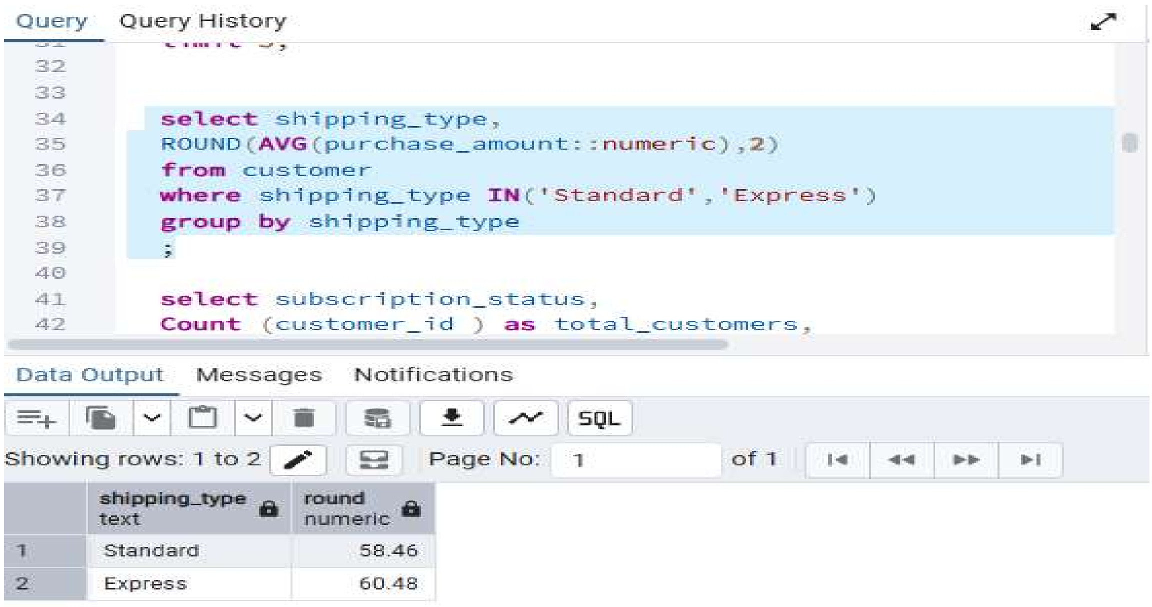
```
;

select subscription_status,
Count (customer_id ) as total_customers,
ROUND(AVG(purchase_amount),2)as avg_spend,
ROUND(SUM(purchase_amount),2)as total_revenue
 from customer
 group by subscription_status
 order by total_revenue, avg_spend desc;
```

Showing rows: 1 to 2 Page No: 1 of 1

	subscription_status text	total_customers bigint	avg_spend numeric	total_revenue numeric
1	Yes	1053	59.49	62645.00
2	No	2847	59.87	170436.00

5.Shipping Type Comparison:

Compared average purchase amount between standard and average shipping.



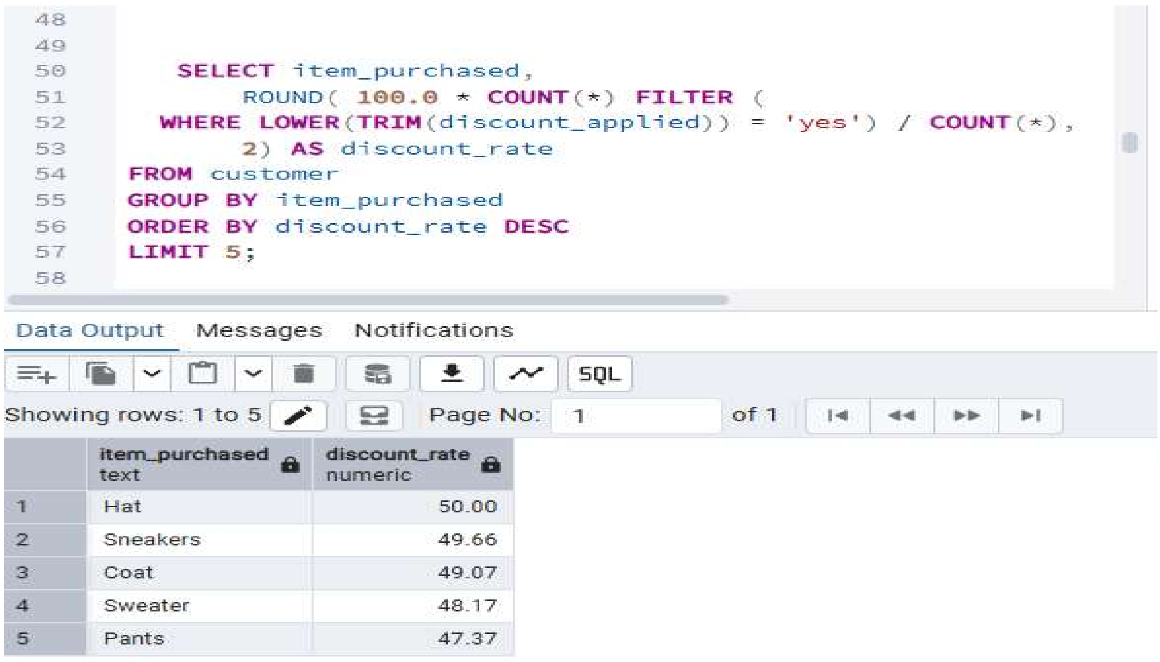
```
select shipping_type,
ROUND(AVG(purchase_amount::numeric),2)
from customer
where shipping_type IN('Standard','Express')
group by shipping_type
;

select subscription_status,
Count (customer_id ) as total_customers,
```

Showing rows: 1 to 2 Page No: 1 of 1

	shipping_type text	round numeric
1	Standard	58.46
2	Express	60.48

6.Discount Dependent Products:

Identify top 5 products highest percentage of discount purchase.



```
   SELECT item_purchased,
       ROUND( 100.0 * COUNT(*) FILTER (
  WHERE LOWER(TRIM(discount_applied)) = 'yes') / COUNT(*),
       2) AS discount_rate
FROM customer
GROUP BY item_purchased
ORDER BY discount_rate DESC
LIMIT 5;
```

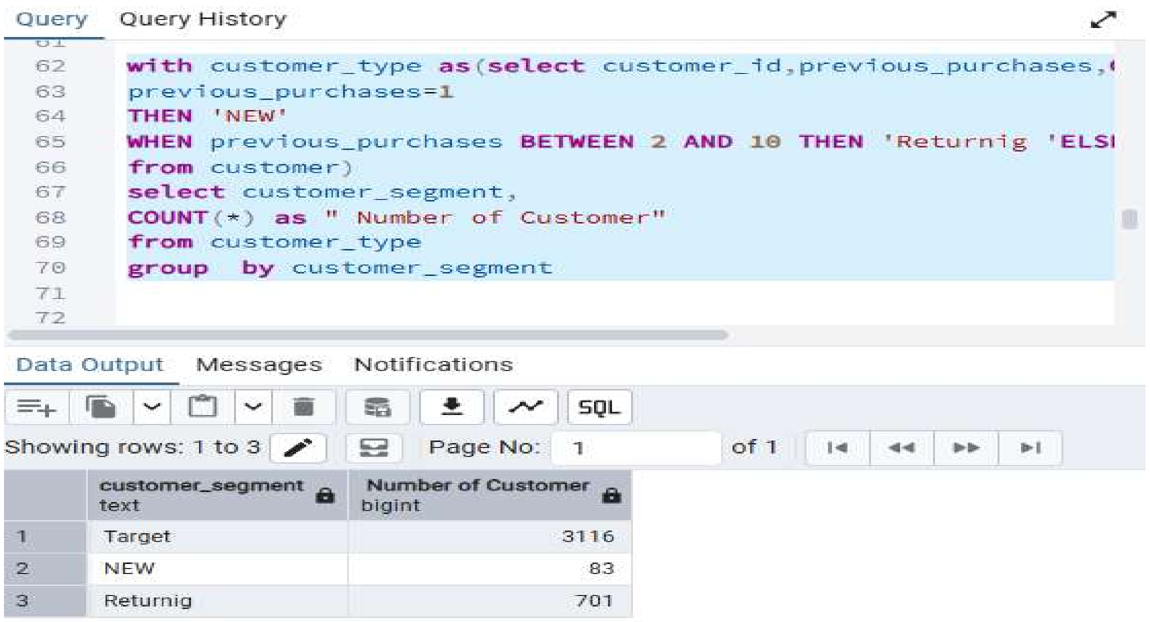
Data Output Messages Notifications

Showing rows: 1 to 5 Page No: 1 of 1

	item_purchased text	discount_rate numeric
1	Hat	50.00
2	Sneakers	49.66
3	Coat	49.07
4	Sweater	48.17
5	Pants	47.37

7.Customer Segmentation:

Classify the customers into new, returning and loyal customers, segments based on the purchase history.



Query Query History

```
with customer_type as(select customer_id,previous_purchases,
previous_purchases=1
THEN 'NEW'
WHEN previous_purchases BETWEEN 2 AND 10 THEN 'Returnig 'ELS
from customer)
select customer_segment,
COUNT(*) as " Number of Customer"
from customer_type
group  by customer_segment
```

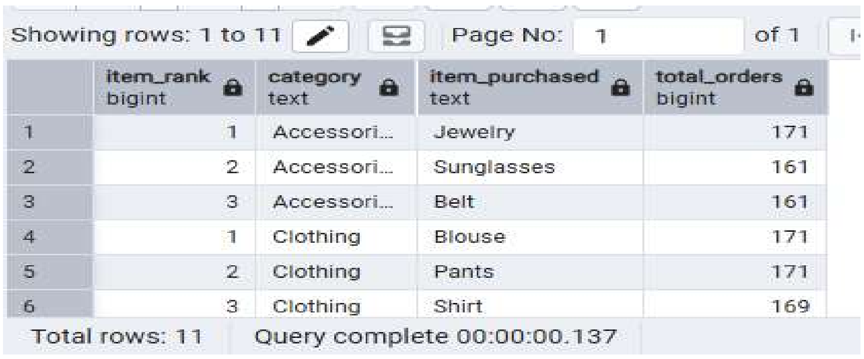
Data Output Messages Notifications

Showing rows: 1 to 3 Page No: 1 of 1

	customer_segment text	Number of Customer bigint
1	Target	3116
2	NEW	83
3	Returnig	701

7.Top 3 products per Category:

Listed most purchased products within each category.



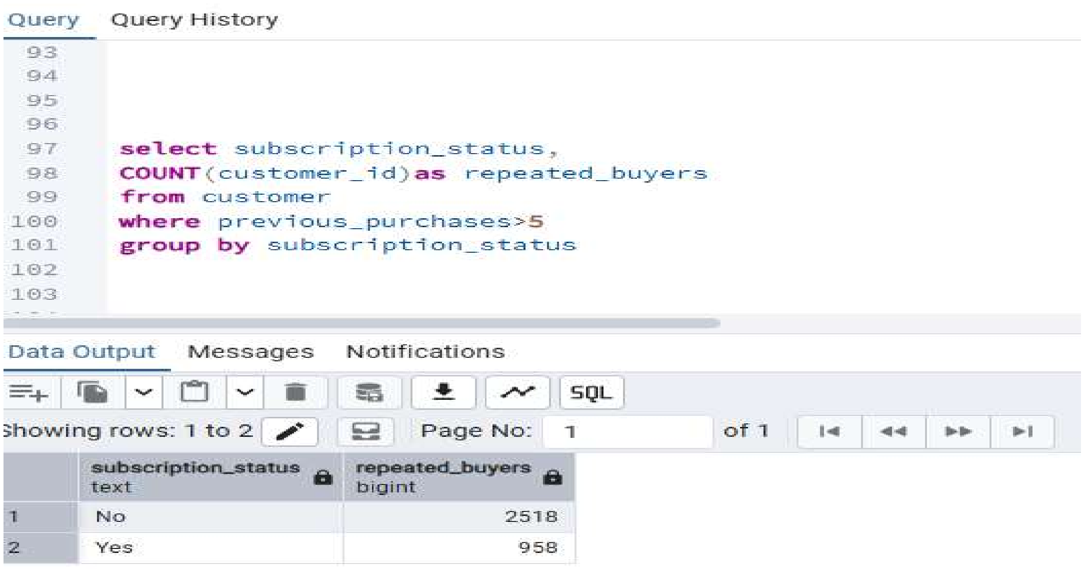
Showing rows: 1 to 11 Page No: 1 of 1

	item_rank bigint	category text	item_purchased text	total_orders bigint
1	1	Accessori...	Jewelry	171
2	2	Accessori...	Sunglasses	161
3	3	Accessori...	Belt	161
4	1	Clothing	Blouse	171
5	2	Clothing	Pants	171
6	3	Clothing	Shirt	169

Total rows: 11 Query complete 00:00:00.137

9.Repeated Buyers and Subscription:

Checked weather the customers with >5 purchases are most likely to subscribe



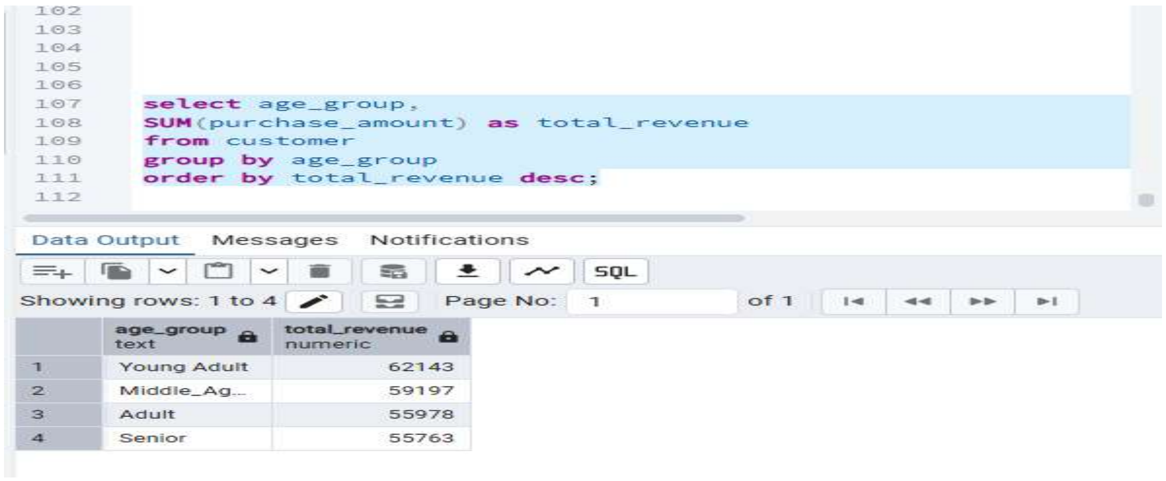
```
select subscription_status,
COUNT(customer_id)as repeated_buyers
from customer
where previous_purchases>5
group by subscription_status
```

Showing rows: 1 to 2 Page No: 1 of 1

	subscription_status text	repeated_buyers bigint
1	No	2518
2	Yes	958

10.Revenue by Age group:

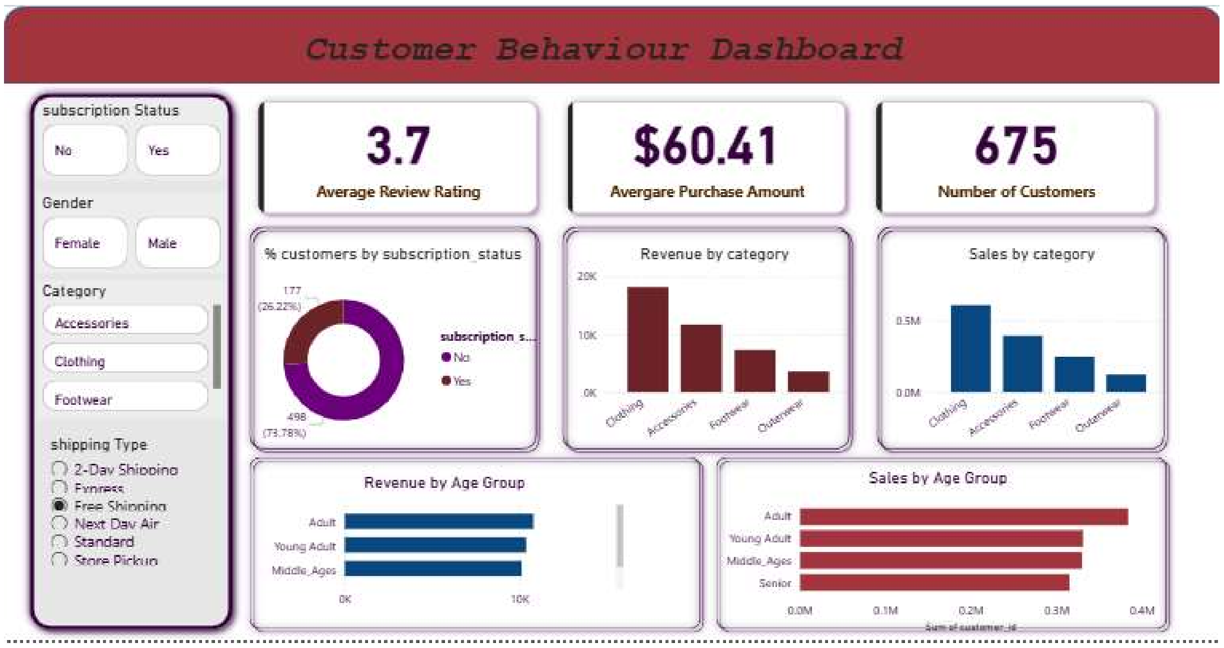
Calculate total revenue distributed by age group.



```sql
select age_group,
SUM(purchase_amount) as total_revenue
from customer
group by age_group
order by total_revenue desc;
```

	age_group text	total_revenue numeric
1	Young Adult	62143
2	Middle_Ag...	59197
3	Adult	55978
4	Senior	55763

5.Dashboard in Power BI:



5.Business Recommendation:

***1.Boost subscription**: promote the exclusive benefits for subscriptions*

***2.Customer Loyalty Program**: Reward repeated buyers to move them into the loyal customers.*

***3.Review Discount Policy**: Balance sales boost with margin control*

***4.Product Positioning**: Highlight top rated and best-selling products in campaigns*

***5. Target Marketing**: Focus efforts on high revenue age group and express shipping users*