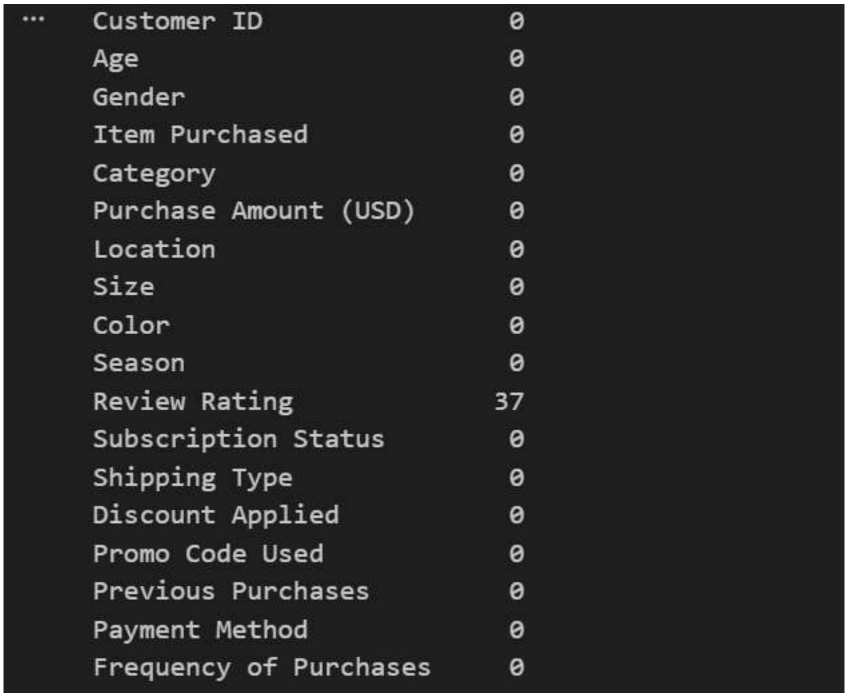
Customer Shopping Behavior Analysis

1. Project Overview

This project analyzes customer shopping behavior using transactional data from 3,900 purchases across various product categories. The goal is to uncover insights into spending patterns, customer segments, product preferences, and subscription behavior to guide strategic business decisions.

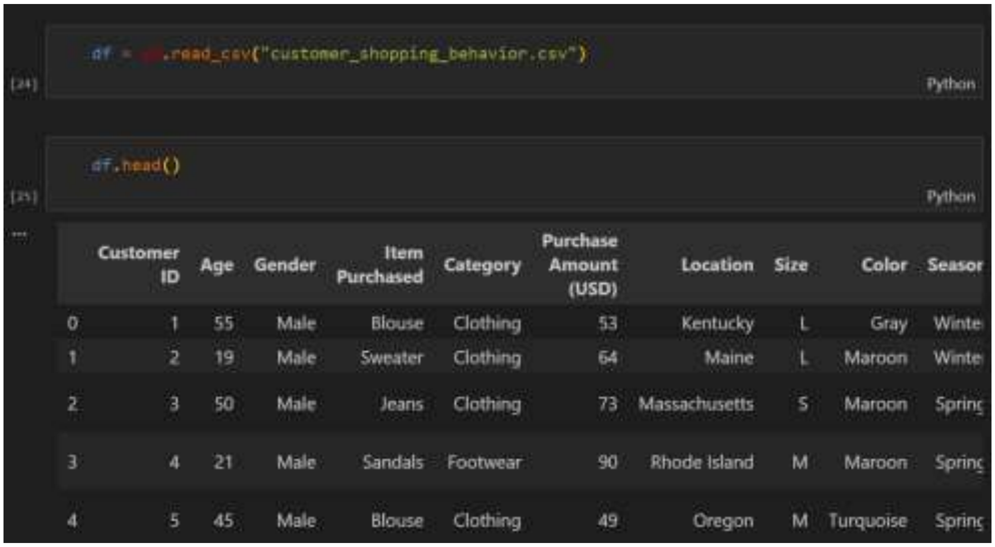
2. Dataset Summary

Rows	3,900
Columns	18
Key Features	Customer demographics (Age, Gender, Location, Subscription Status)
	Purchase details (Item Purchased, Category, Purchase Amount, Season, Size, Color)
	Shopping behavior (Discount Applied, Promo Code Used, Previous Purchases, Frequency of Purchases, Review Rating, Shipping Type)
Missing Data	37 values in the Review Rating column  ... 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

3. Exploratory Data Analysis using Python

Data preparation and cleaning steps included:

- Data Loading using pandas.

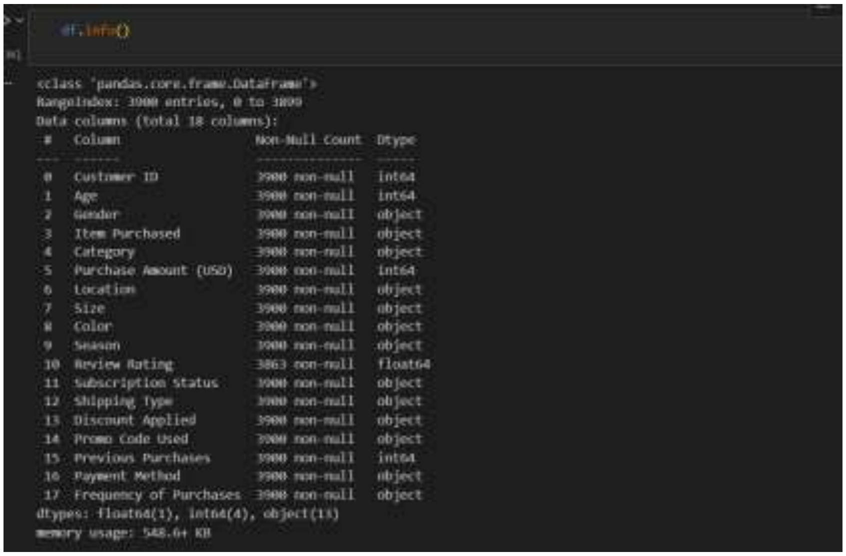


```python
df = pd.read_csv("customer_shopping_behavior.csv")
```

```python
df.head()
```

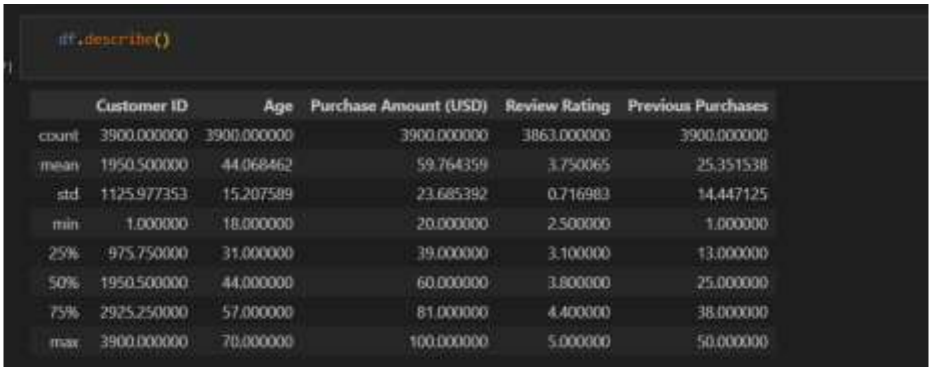
	Customer ID	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Seasor
0	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winte
1	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winte
2	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Sprinc
3	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Sprinc
4	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Sprinc

- Initial Exploration using df.info() and describe().



```python
df.info()
```

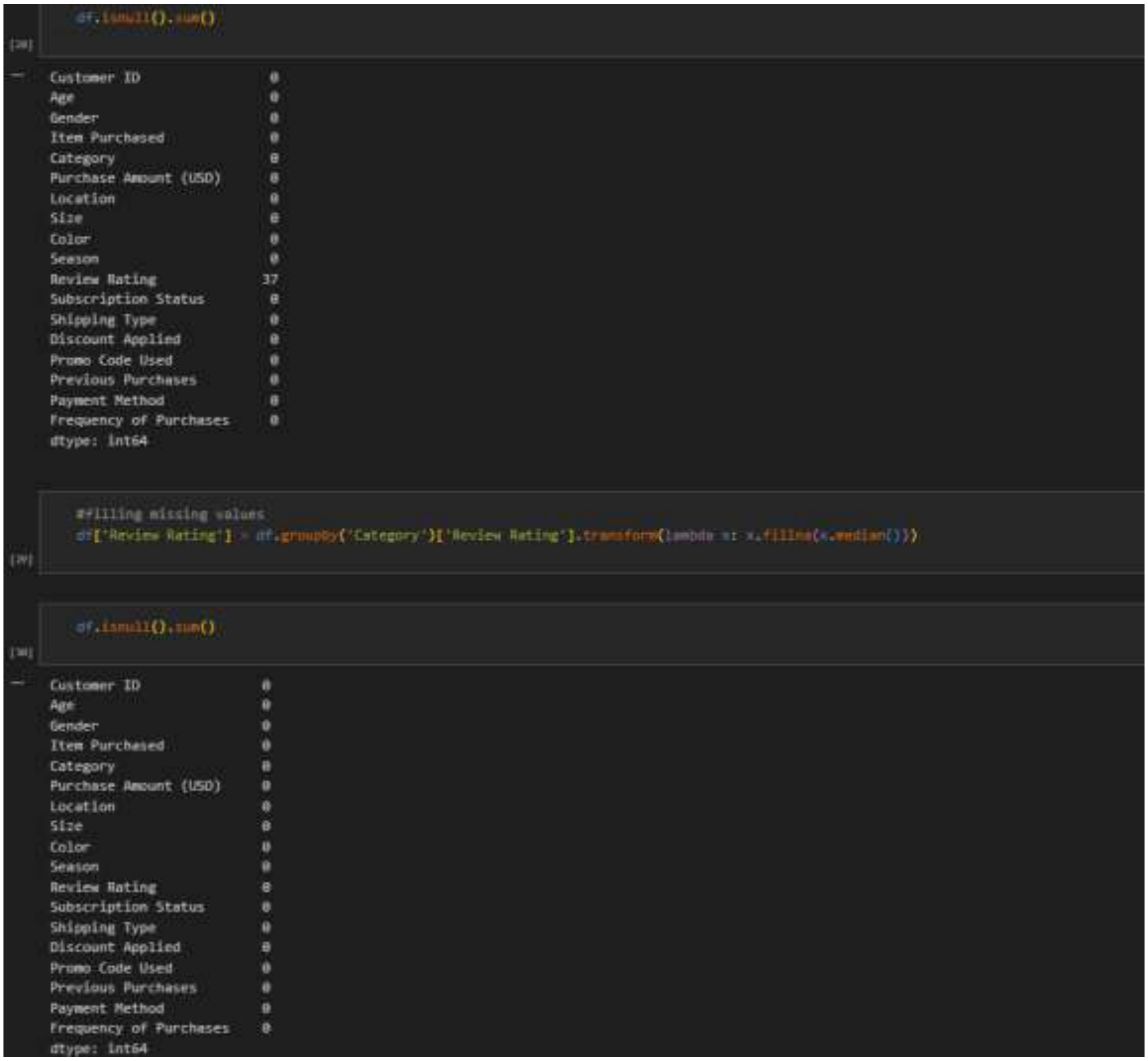
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Customer ID             3900 non-null   int64  
 1   Age                     3900 non-null   int64  
 2   Gender                  3900 non-null   object 
 3   Item Purchased          3900 non-null   object 
 4   Category                3900 non-null   object 
 5   Purchase Amount (USD)   3900 non-null   int64  
 6   Location                3900 non-null   object 
 7   Size                    3900 non-null   object 
 8   Color                   3900 non-null   object 
 9   Season                  3900 non-null   object 
 10  Review Rating           3863 non-null   float64
 11  Subscription Status     3900 non-null   object 
 12  Shipping Type           3900 non-null   object 
 13  Discount Applied        3900 non-null   object 
 14  Promo Code Used         3900 non-null   object 
 15  Previous Purchases      3900 non-null   int64  
 16  Payment Method          3900 non-null   object 
 17  Frequency of Purchases  3900 non-null   object 
dtypes: float64(1), int64(4), object(13)
memory usage: 548.6+ KB
```



```
df.describe()
```

	Customer ID	Age	Purchase Amount (USD)	Review Rating	Previous Purchases
count	3900.000000	3900.000000	3900.000000	3863.000000	3900.000000
mean	1950.500000	44.068462	59.764359	3.750065	25.351538
std	1125.977353	15.207589	23.685392	0.716983	14.447125
min	1.000000	18.000000	20.000000	2.500000	1.000000
25%	975.750000	31.000000	39.000000	3.100000	13.000000
50%	1950.500000	44.000000	60.000000	3.800000	25.000000
75%	2925.250000	57.000000	81.000000	4.400000	38.000000
max	3900.000000	70.000000	100.000000	5.000000	50.000000

- Missing Data Handling: Imputed missing Review Rating values with median per product category.



```
df.isnull().sum()
```

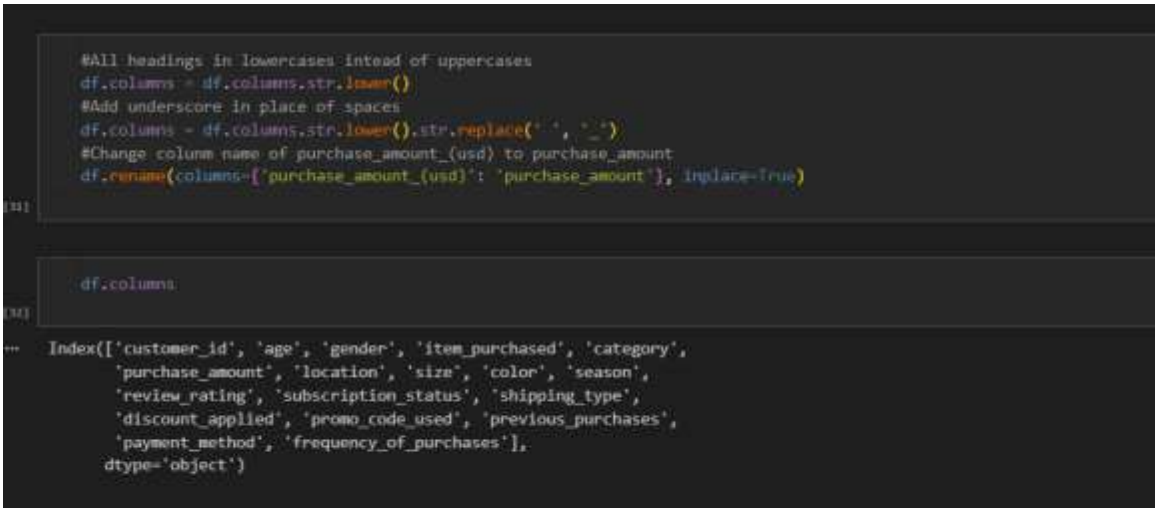
```
Customer ID                0
Age                        0
Gender                     0
Item Purchased             0
Category                   0
Purchase Amount (USD)      0
Location                   0
Size                       0
Color                      0
Season                     0
Review 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
dtype: int64
```

```
#filling missing values
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             0
Subscription Status       0
Shipping Type             0
Discount Applied          0
Promo Code Used           0
Previous Purchases        0
Payment Method            0
Frequency of Purchases    0
dtype: int64
```

- Column Standardization: Renamed columns to snake_case.

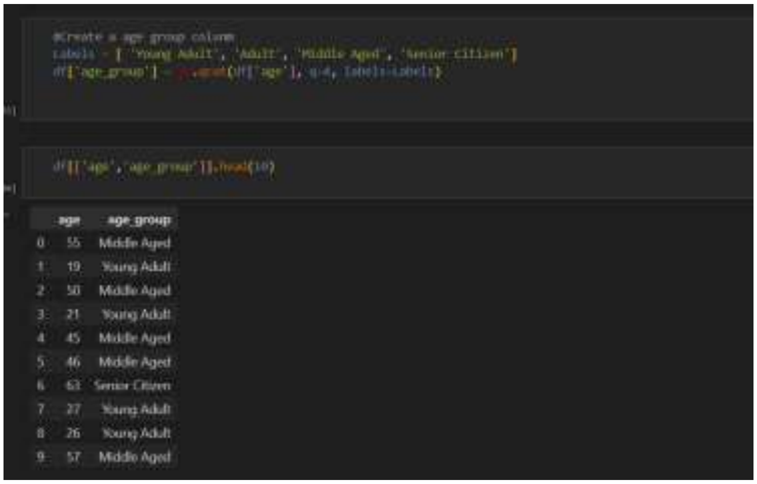


```
#All headings in lowercases intead of uppercases
df.columns = df.columns.str.lower()
#Add underscore in place of spaces
df.columns = df.columns.str.lower().str.replace(' ', '_')
#Change column name of purchase_amount_(usd) to purchase_amount
df.rename(columns={'purchase_amount_(usd)': 'purchase_amount'}, inplace=True)
```

```
df.columns
```

```
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')
```

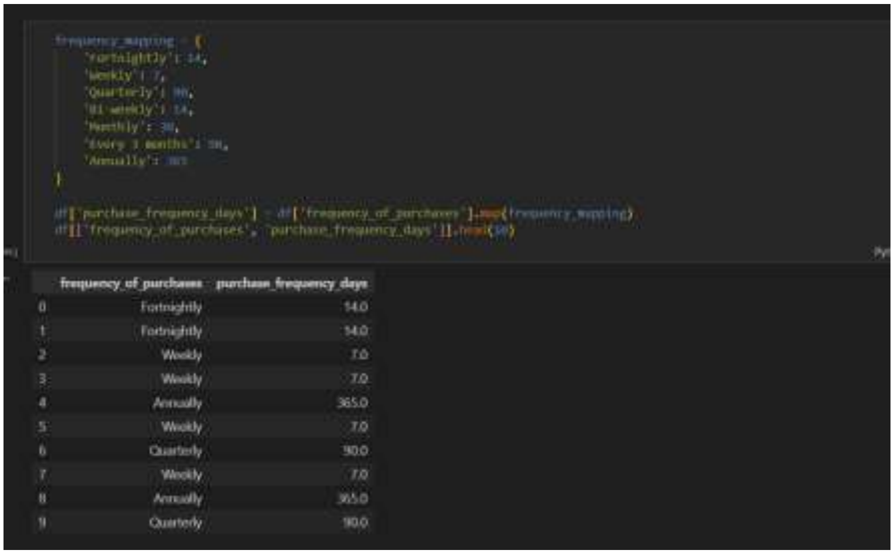
- Feature Engineering: Created age_group and purchase_frequency_days.



```
#Create a age group column
labels = [ 'Young Adult', 'Adult', 'Middle Aged', 'Senior Citizen']
df['age_group'] = pd.qcut(df['age'], q=4, labels=labels)
```

```
df[['age','age_group']].head(10)
```

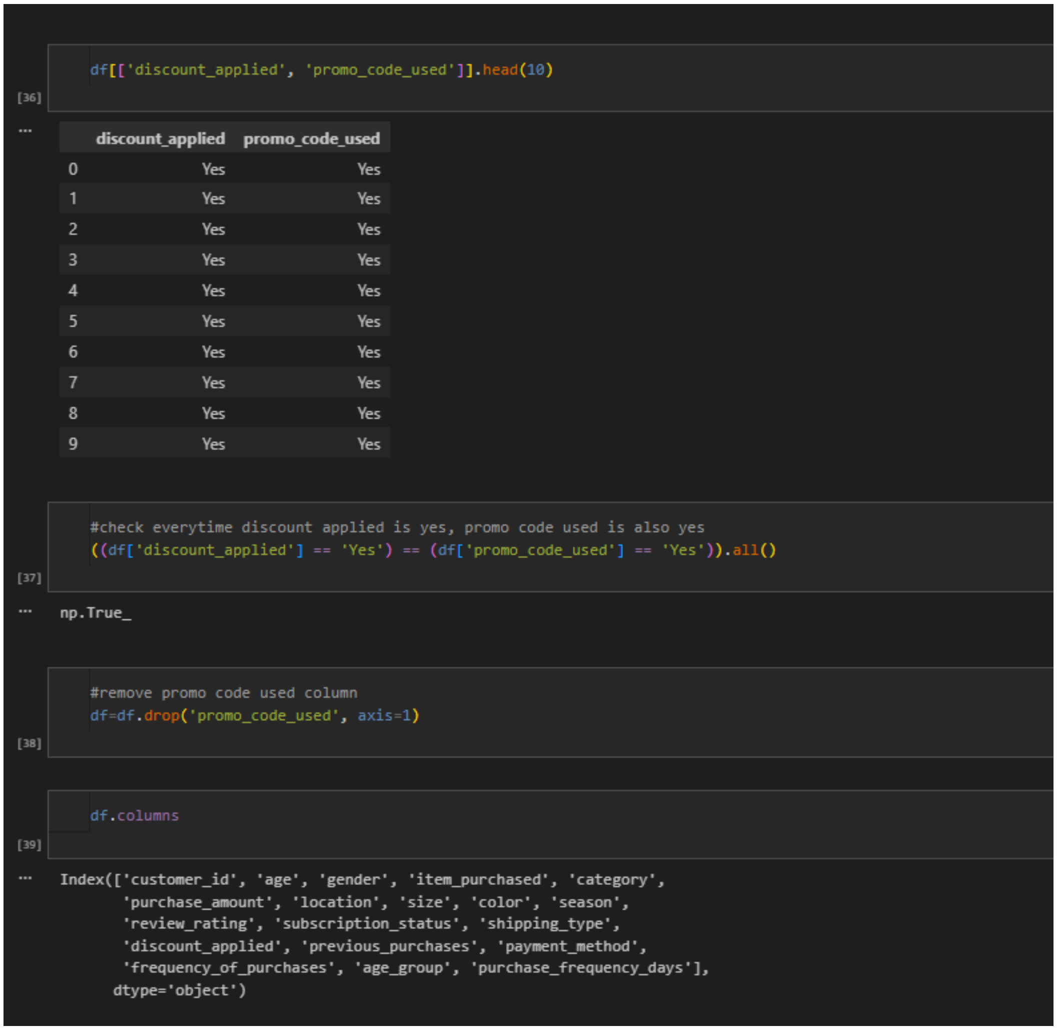
```
   age  age_group
0   55  Middle Aged
1   19  Young Adult
2   50  Middle Aged
3   21  Young Adult
4   45  Middle Aged
5   46  Middle Aged
6   63  Senior Citizen
7   27  Young Adult
8   26  Young Adult
9   57  Middle Aged
```



```
frequency_mapping = {
    'Fortnightly': 14,
    'Weekly': 7,
    'Quarterly': 90,
    'Bi-Weekly': 14,
    'Monthly': 30,
    'Every 3 months': 90,
    'Annually': 365
}

df['purchase_frequency_days'] = df['frequency_of_purchases'].map(frequency_mapping)
df[['frequency_of_purchases', 'purchase_frequency_days']].head(10)
```

```
   frequency_of_purchases  purchase_frequency_days
0             Fortnightly                     14.0
1             Fortnightly                     14.0
2                  Weekly                      7.0
3                  Weekly                      7.0
4                Annually                    365.0
5                  Weekly                      7.0
6               Quarterly                     90.0
7                  Weekly                      7.0
8                Annually                    365.0
9               Quarterly                     90.0
```

- Consistency Check: Removed redundant promo_code_used.



[36]

```
df[['discount_applied', 'promo_code_used']].head(10)
```

	discount_applied	promo_code_used
0	Yes	Yes
1	Yes	Yes
2	Yes	Yes
3	Yes	Yes
4	Yes	Yes
5	Yes	Yes
6	Yes	Yes
7	Yes	Yes
8	Yes	Yes
9	Yes	Yes

[37]

```
#check everytime discount applied is yes, promo code used is also yes
((df['discount_applied'] == 'Yes') == (df['promo_code_used'] == 'Yes')).all()
```

```
np.True_
```

[38]

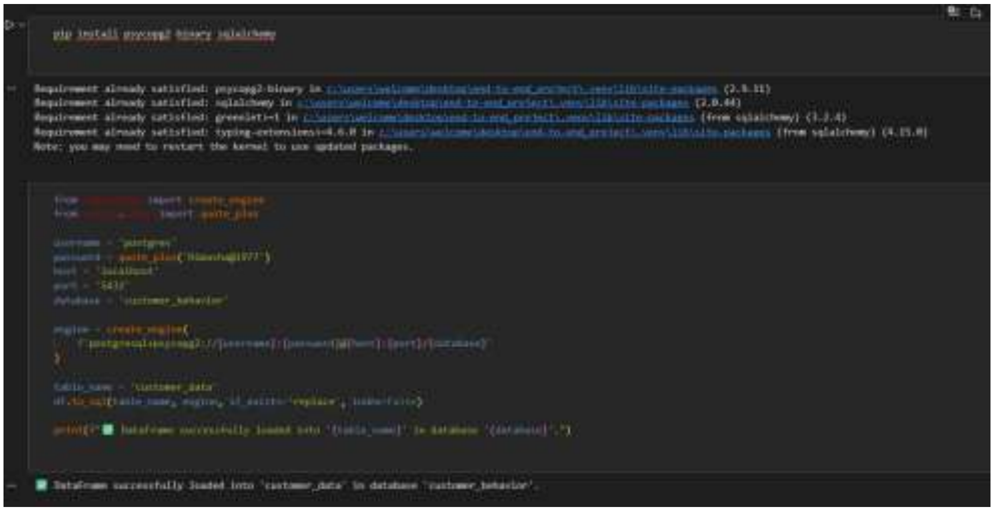
```
#remove promo code used column
df=df.drop('promo_code_used', axis=1)
```

[39]

```
df.columns
```

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

- Database Integration: Loaded the cleaned DataFrame into PostgreSQL.



```
pip install psycopg2-binary sqlalchemy
```

```
Requirement already satisfied: psycopg2-binary in c:\users\welcome\desktop\end-to-end_project1\.venv\lib\site-packages (2.9.11)
Requirement already satisfied: sqlalchemy in c:\users\welcome\desktop\end-to-end_project1\.venv\lib\site-packages (2.0.44)
Requirement already satisfied: greenlet>=1 in c:\users\welcome\desktop\end-to-end_project1\.venv\lib\site-packages (from sqlalchemy) (3.2.4)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\welcome\desktop\end-to-end_project1\.venv\lib\site-packages (from sqlalchemy) (4.15.0)
Note: you may need to restart the kernel to use updated packages.
```

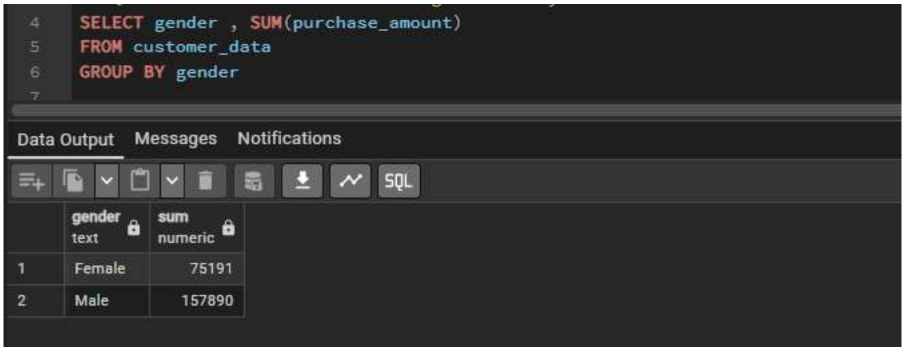
```
from sqlalchemy import create_engine
from urllib.parse import quote_plus

username = 'postgres'
password = quote_plus('Himesha@1977')
host = 'localhost'
port = '5432'
database = 'customer_behavior'

engine = create_engine(
    f'postgresql+psycopg2://{username}:{password}@{host}:{port}/{database}'
)

table_name = 'customer_data'
df.to_sql(table_name, engine, if_exists='replace', index=False)

print(f"✅ DataFrame successfully loaded into '{table_name}' in database '{database}'.")
```

```
✅ DataFrame successfully loaded into 'customer_data' in database 'customer_behavior'.
```

4. Data Analysis using SQL (Business Transactions)

Key SQL analysis:

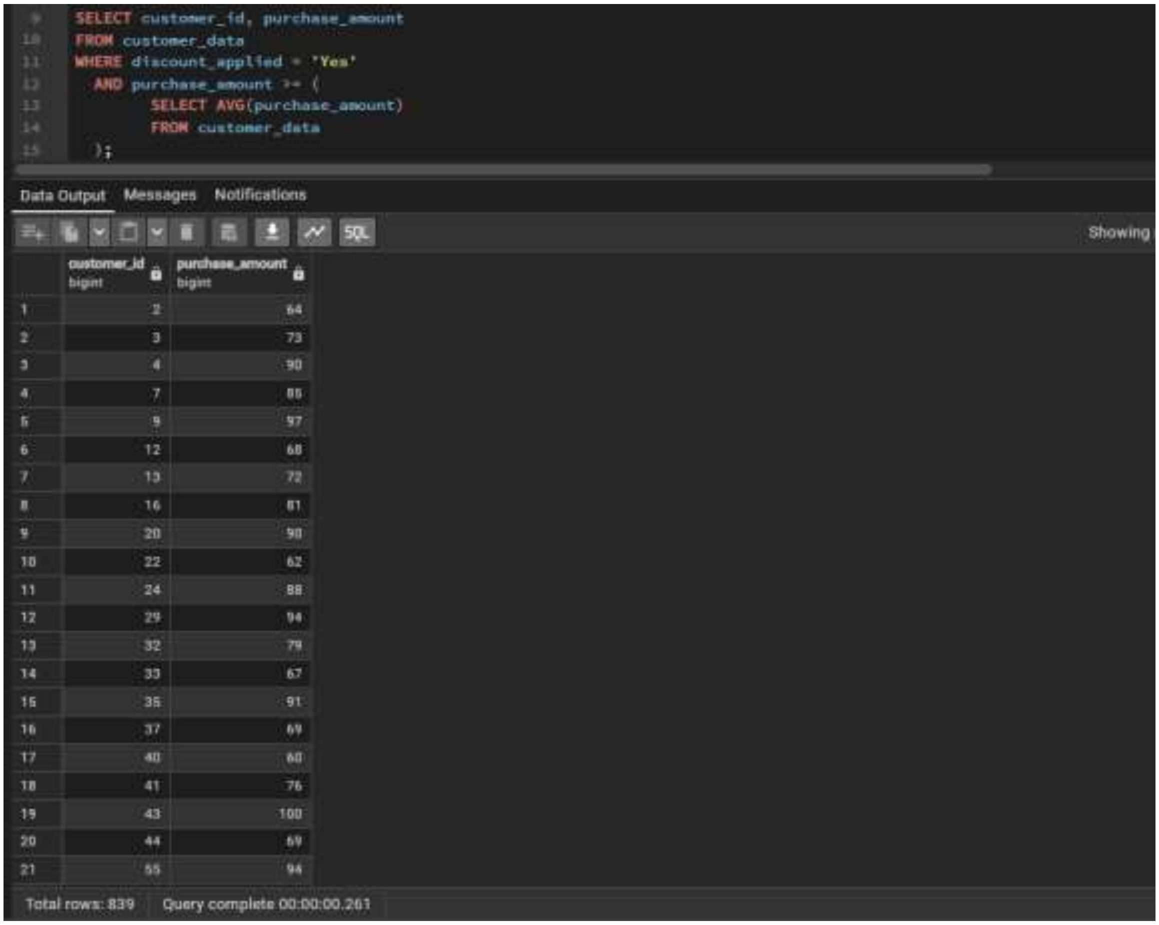
1. Revenue by Gender



```sql
SELECT gender , SUM(purchase_amount)
FROM customer_data
GROUP BY gender
```

	gender text	sum numeric
1	Female	75191
2	Male	157890

2. High-Spending Discount Users.

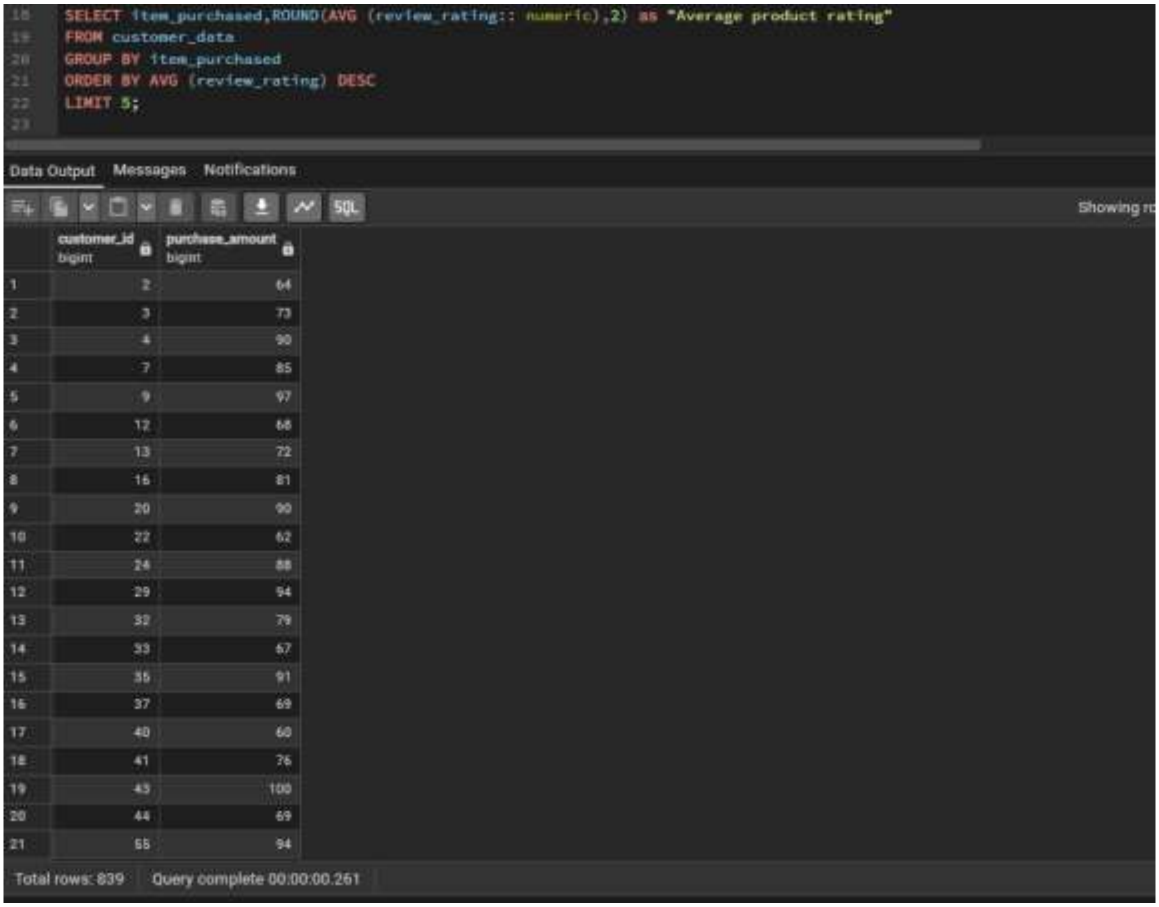


```sql
SELECT customer_id, purchase_amount
FROM customer_data
WHERE discount_applied = 'Yes'
  AND purchase_amount >= (
        SELECT AVG(purchase_amount)
        FROM customer_data
  );
```

	customer_id bigint	purchase_amount bigint
1	2	64
2	3	73
3	4	90
4	7	85
5	9	97
6	12	68
7	13	72
8	16	81
9	20	90
10	22	62
11	24	88
12	29	94
13	32	79
14	33	67
15	35	91
16	37	69
17	40	60
18	41	76
19	43	100
20	44	69
21	55	94

```
Total rows: 839    Query complete 00:00:00.261
```

3. Top 5 Products by Rating.

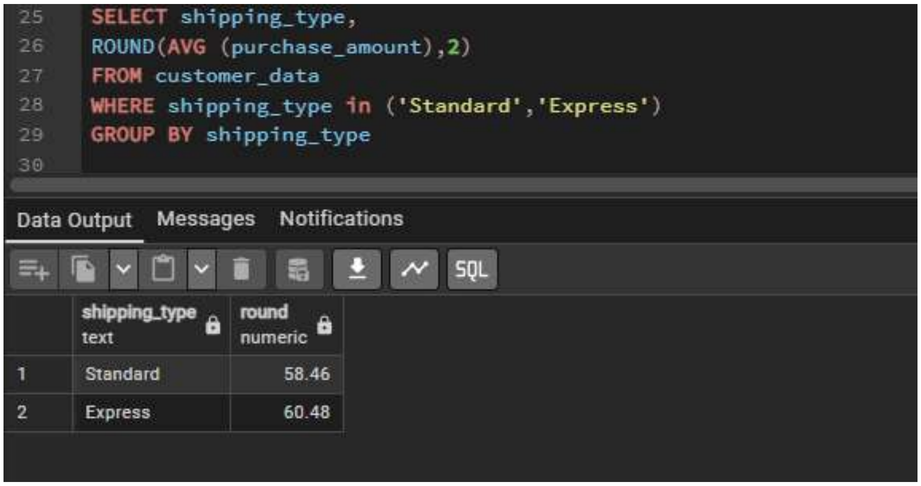


```sql
SELECT item_purchased,ROUND(AVG (review_rating:: numeric),2) as "Average product rating"
FROM customer_data
GROUP BY item_purchased
ORDER BY AVG (review_rating) DESC
LIMIT 5;
```

	customer_id bigint	purchase_amount bigint
1	2	64
2	3	73
3	4	90
4	7	85
5	9	97
6	12	68
7	13	72
8	16	81
9	20	90
10	22	62
11	24	88
12	29	94
13	32	79
14	33	67
15	35	91
16	37	69
17	40	60
18	41	76
19	43	100
20	44	69
21	55	94

```
Total rows: 839    Query complete 00:00:00.261
```

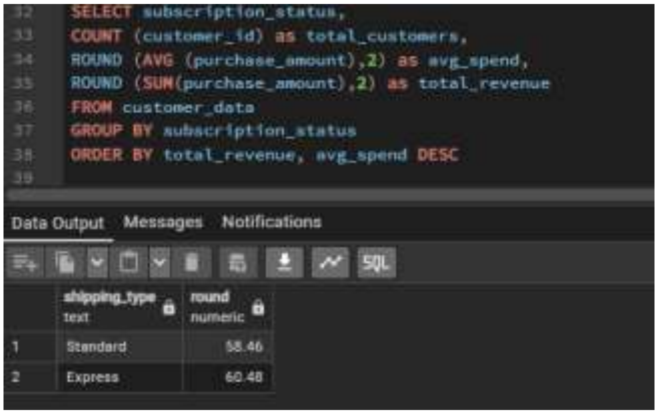
4. Shipping Type Comparison.



```sql
SELECT shipping_type,
ROUND(AVG (purchase_amount),2)
FROM customer_data
WHERE shipping_type in ('Standard','Express')
GROUP BY shipping_type
```

	shipping_type text	round numeric
1	Standard	58.46
2	Express	60.48

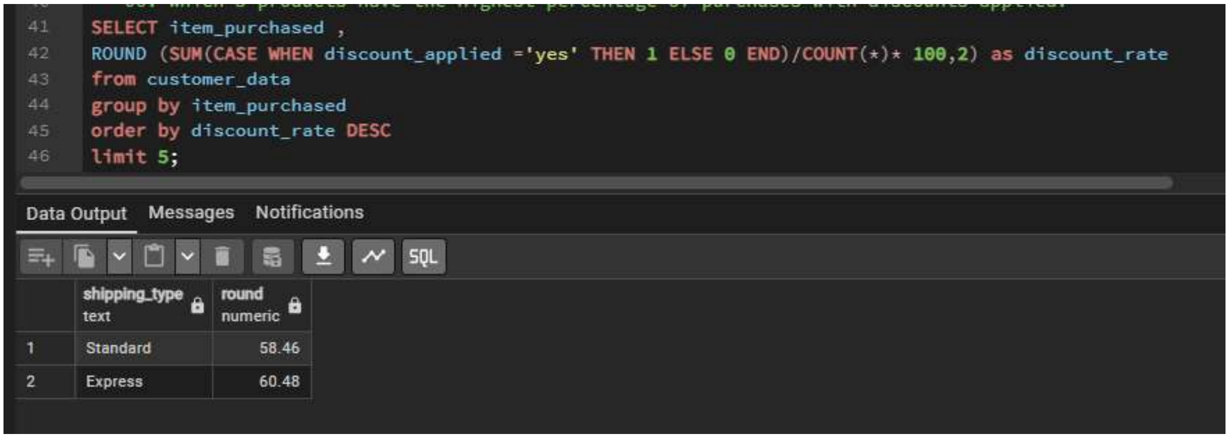
5. Subscribers vs Non-Subscribers.



```sql
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_data
GROUP BY subscription_status
ORDER BY total_revenue, avg_spend DESC
```

	shipping_type text	round numeric
1	Standard	58.46
2	Express	60.48

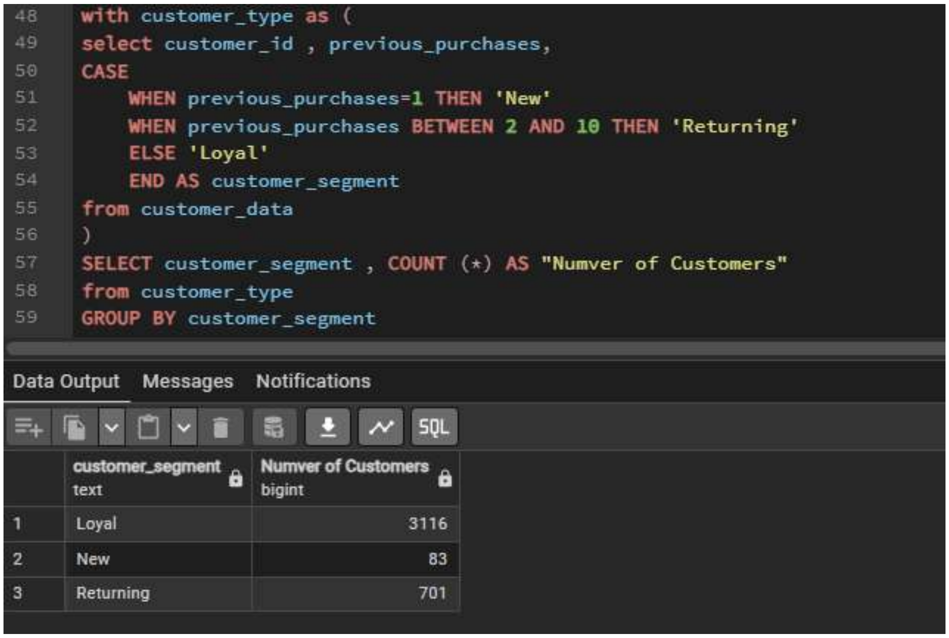
6. Discount-Dependent Products



```sql
Which 5 products have the highest percentage of purchases with discounts applied.
SELECT item_purchased ,
ROUND (SUM(CASE WHEN discount_applied ='yes' THEN 1 ELSE 0 END)/COUNT(*)* 100,2) as discount_rate
from customer_data
group by item_purchased
order by discount_rate DESC
limit 5;
```

	shipping_type text	round numeric
1	Standard	58.46
2	Express	60.48

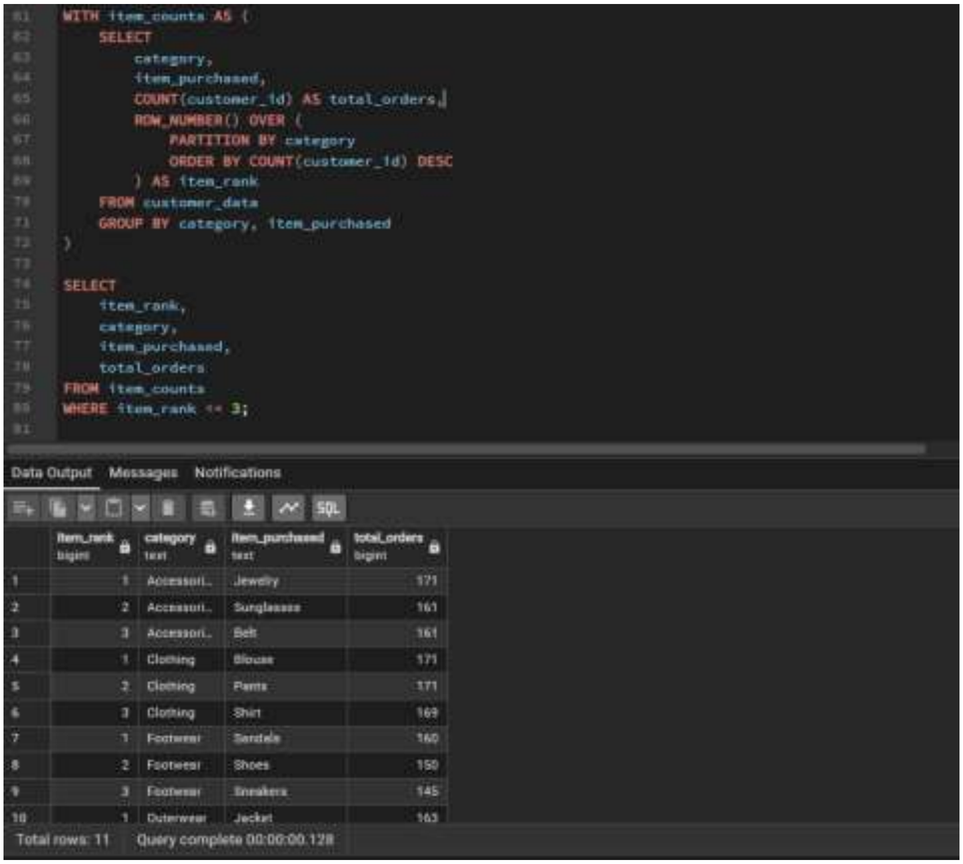
7. Customer Segmentation.



```sql
with customer_type as (
select customer_id , previous_purchases,
CASE
    WHEN previous_purchases=1 THEN 'New'
    WHEN previous_purchases BETWEEN 2 AND 10 THEN 'Returning'
    ELSE 'Loyal'
    END AS customer_segment
from customer_data
)
SELECT customer_segment , COUNT (*) AS "Numver of Customers"
from customer_type
GROUP BY customer_segment
```

	customer_segment text	Numver of Customers bigint
1	Loyal	3116
2	New	83
3	Returning	701

8. Top 3 Products per Category.

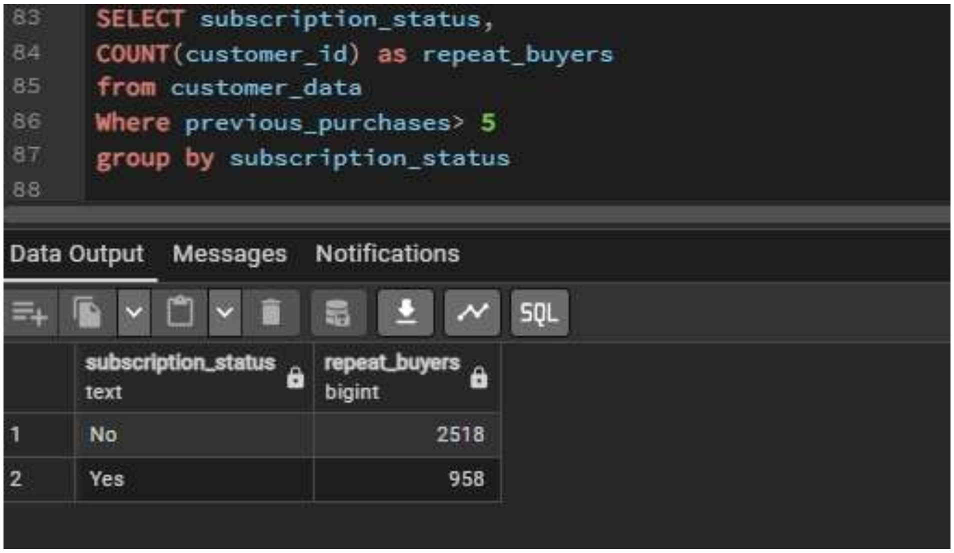


```sql
WITH item_counts AS (
    SELECT
        category,
        item_purchased,
        COUNT(customer_id) AS total_orders,
        ROW_NUMBER() OVER (
            PARTITION BY category
            ORDER BY COUNT(customer_id) DESC
        ) AS item_rank
    FROM customer_data
    GROUP BY category, item_purchased
)

SELECT
    item_rank,
    category,
    item_purchased,
    total_orders
FROM item_counts
WHERE item_rank <= 3;
```

	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
7	1	Footwear	Sandals	160
8	2	Footwear	Shoes	150
9	3	Footwear	Sneakers	145
10	1	Outerwear	Jacket	163

Total rows: 11 Query complete 00:00:00.128

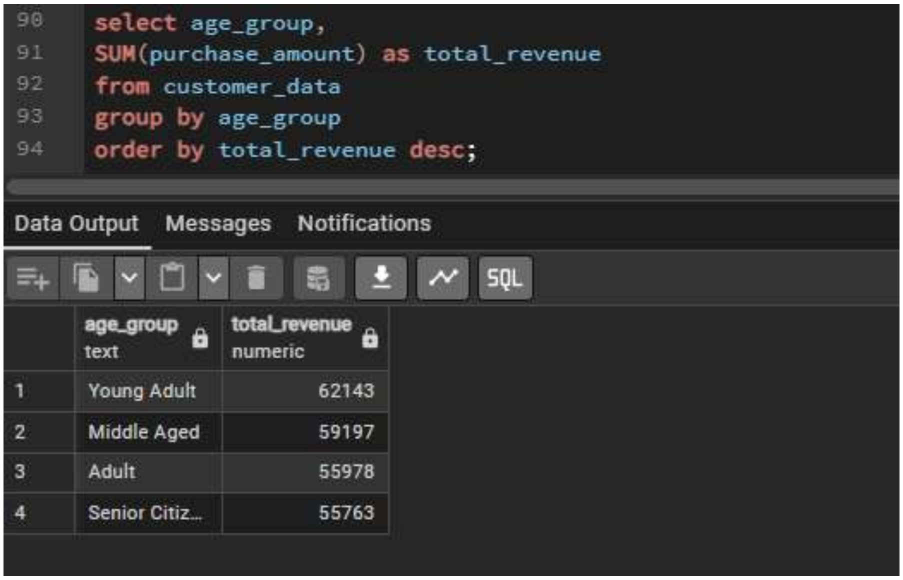
9. Repeat Buyers and Subscription Likelihood.



```sql
SELECT subscription_status,
COUNT(customer_id) as repeat_buyers
from customer_data
Where previous_purchases> 5
group by subscription_status
```

	subscription_status text	repeat_buyers bigint
1	No	2518
2	Yes	958

10. Revenue by Age Group



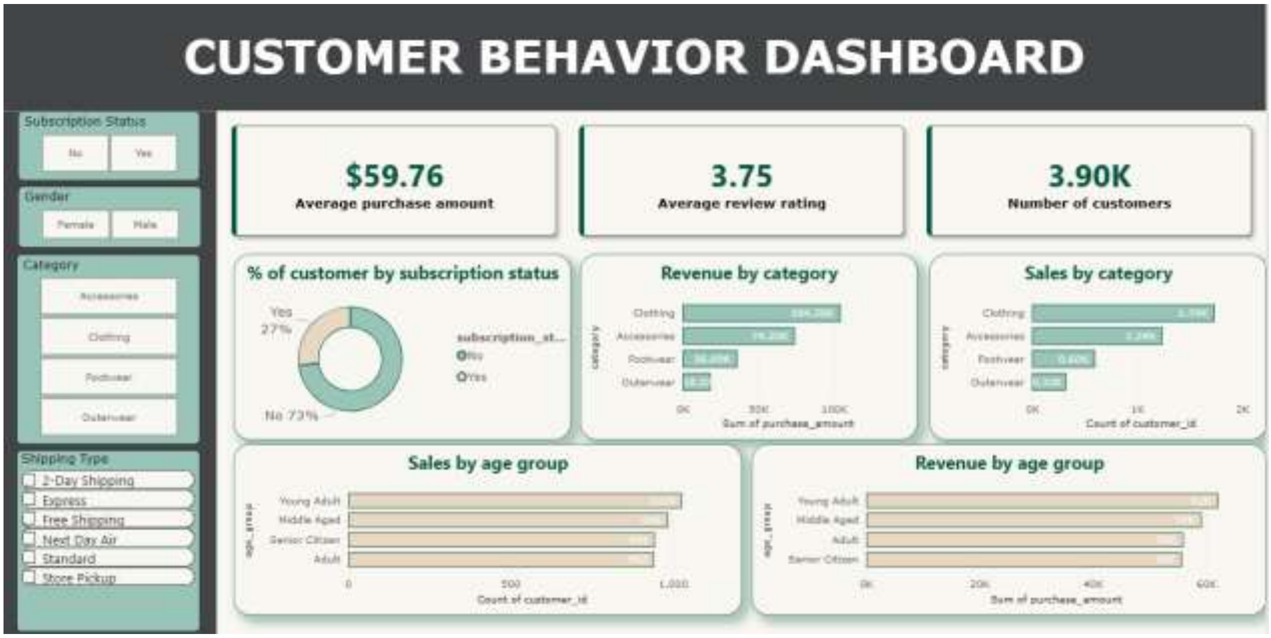
```
select age_group,
SUM(purchase_amount) as total_revenue
from customer_data
group by age_group
order by total_revenue desc;
```

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

5. Dashboard in Power BI

Interactive visuals displayed:

- Customer demographics
- Revenue patterns
- Purchase category trends
- Subscription impacts
- Seasonal behavior insights



6. Business Recommendations

- Boost Subscriptions with exclusive perks.
- Implement Customer Loyalty Programs.
- Review Discount Policies for margin protection.
- Highlight top-rated products.
- Target high-value age groups and express-shipping users.