Decoding Customer DNA – Retail Behavior Analytics

Problem Statement:

A leading retail company wants to better understand its customers' shopping behavior in order to improve sales, customer satisfaction, and long-term loyalty. The management team has noticed changes in purchasing patterns across demographics, product categories, and sales channels (online vs. offline). They are particularly interested in uncovering which factors, such as discounts, reviews, seasons, or payment preferences, drive consumer decisions and repeat purchases.

You are tasked with analyzing the company's consumer behavior dataset to answer the following; overarching business question:
"How can the company leverage consumer shopping data to identify trends, improve customer engagement, and optimize marketing and product strategies?"

Deliverables:

I. **Data Preparation & Modeling (Python):** Clean and transform the raw dataset for analysis.

II. **Data Analysis (SQL):** Organize the data into a structured format, simulate business transactions, and run queries to extract insights on customer segments, loyalty, and purchase drivers.

III. **Visualization & Insights (Power BI):** Build an interactive dashboard that highlights key patterns and trends, enabling stakeholders to make data-driven decisions.

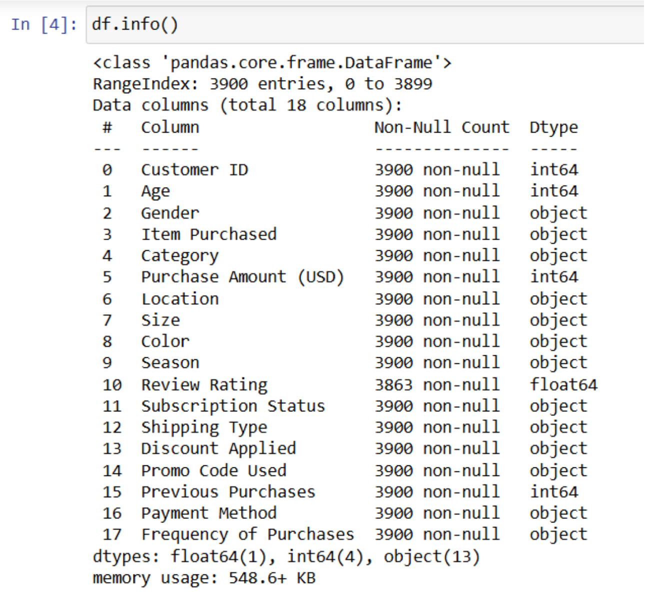
IV. **Report and Presentation:** Write a clear project report summarizing your key findings and business recommendations. Prepare a presentation that visually communicates insights and actionable recommendations to stakeholders.

V. **GitHub Repository:** Include all Python scripts, SQL queries, and dashboard files in a well-structured repository.

1) Data Preparation in Python:

Started off by importing the dataset and performing Exploratory Data Analysis (EDA), which revealed the following insights:

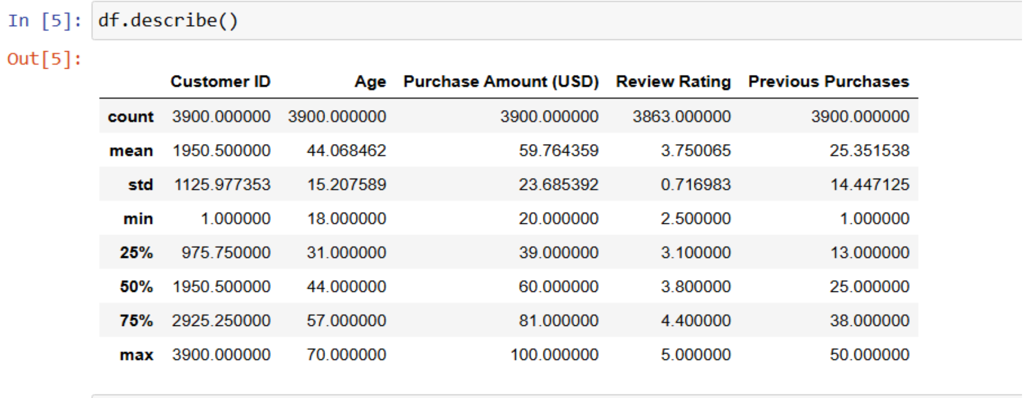
a) The data types of the columns were fine. Only the ‘Review Rating’ Column had null values. I filled the null values with the Median values for the specific categories.



```
In [4]: 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
```

b) The description of numerical columns:



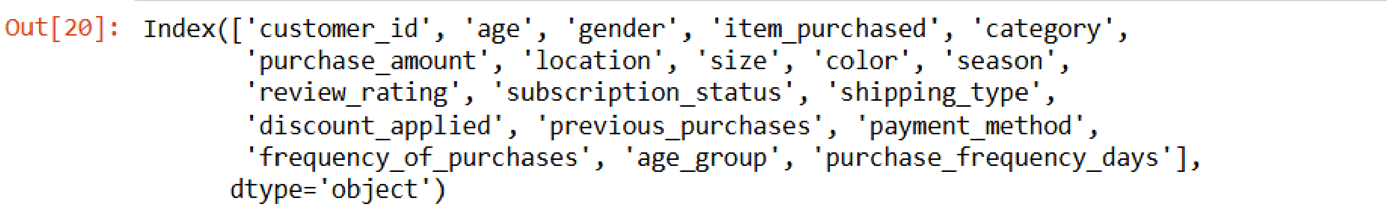
```
In [5]: df.describe()
Out[5]:
```

	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

c) Added two features for clearer analysis.

One which maps ages into groups of young adults, adults, middle-aged, and seniors, and the other one labels the purchase frequency days into fortnightly, weekly, monthly, quarterly, and yearly.

d) After all the preprocessing and feature engineering, here is the list of columns which will later be imported into pgAdmin4 for further analysis using SQL.

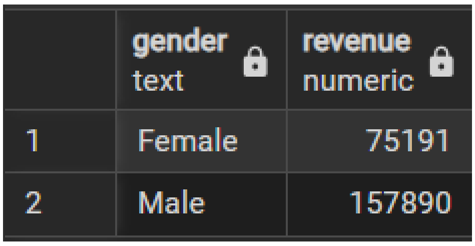


```
Out[20]: Index(['customer_id', 'age', 'gender', 'item_purchased', 'category',
               'purchase_amount', 'location', 'size', 'color', 'season',
               'review_rating', 'subscription_status', 'shipping_type',
               'discount_applied', 'previous_purchases', 'payment_method',
               'frequency_of_purchases', 'age_group', 'purchase_frequency_days'],
              dtype='object')
```

2) Data Analysis (SQL):

-- Q1. What is the total revenue generated by Male and Female customers?

```
SELECT gender, SUM(purchase_amount) as Revenue
FROM customer
GROUP BY gender
```

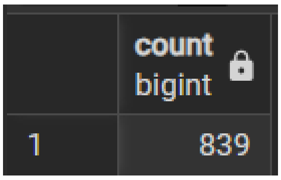


	gender text	revenue numeric
1	Female	75191
2	Male	157890

We can observe that the sample we have shows male dominance.

--Q2. Which customers used a discount but still spent more than the average purchase amount?

```
SELECT COUNT(customer_id) FROM customer
WHERE discount_applied = 'Yes'
AND purchase_amount > (SELECT AVG(purchase_amount) FROM customer)
```

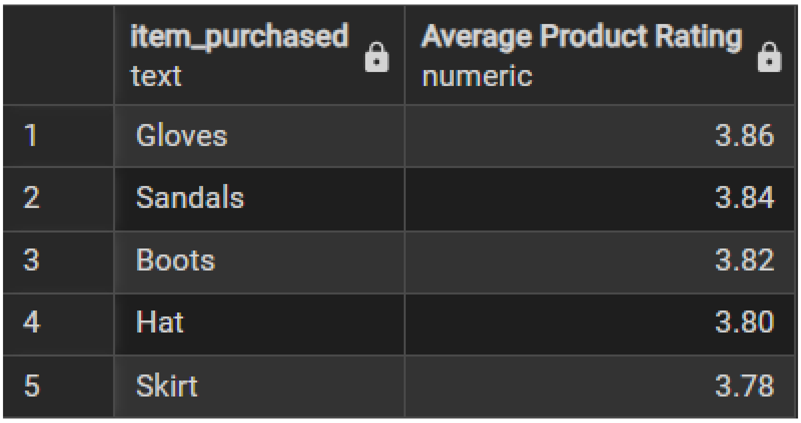


	count bigint
1	839

839 customers used the discount code, but they still spent more than average!

-- Q3. Which are the top 5 products with the highest average review rating?

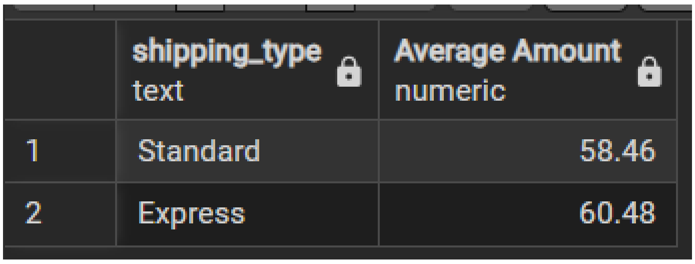
```
SELECT item_purchased,
ROUND(AVG(review_rating::numeric),2) AS "Average Product Rating"
FROM customer
GROUP BY item_purchased
ORDER BY AVG(review_rating) DESC
LIMIT 5
```



	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

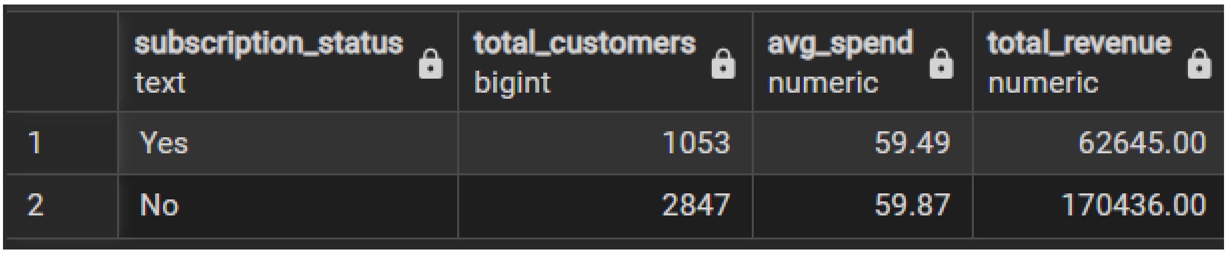
We can observe top-selling products.

```
--Q4. Compare the average Purchase Amounts between Standard and Express Shipping.
SELECT shipping_type,
ROUND(AVG(purchase_amount),2) AS "Average Amount"
FROM customer
WHERE shipping_type IN ('Standard','Express')
GROUP BY shipping_type;
```



	shipping_type text	Average Amount numeric
1	Standard	58.46
2	Express	60.48

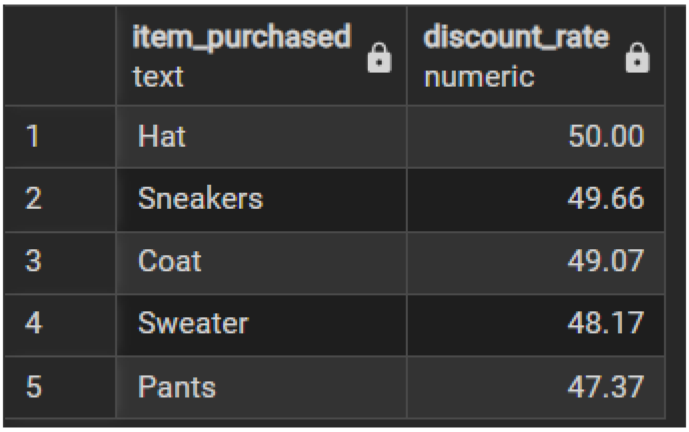
```
--Q5. Do subscribed customers spend more? Compare average spend and total revenue
--between subscribers and non-subscribers.
SELECT subscription_status,
    COUNT(customer_id) AS total_customers,
    ROUND(AVG(purchase_amount),2) AS avg_spend,
    ROUND(SUM(purchase_amount),2) AS total_revenue
FROM customer
GROUP BY subscription_status
ORDER BY total_revenue,avg_spend DESC;
```



	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

Surprisingly, non-subscribers spend more!

```
--Q6. Which 5 products have the highest percentage of purchases with discounts applied?
SELECT item_purchased,
    ROUND(100.0 * SUM(CASE WHEN discount_applied = 'Yes' THEN 1 ELSE 0
END)/COUNT(*),2) AS discount_rate
FROM customer
GROUP BY item_purchased
ORDER BY discount_rate DESC
LIMIT 5;
```

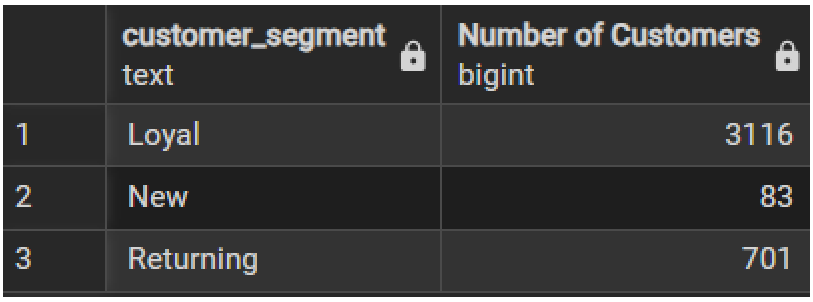


	item_purchased text	discount_rate numeric
1	Hat	50.00
2	Sneakers	49.66
3	Coat	49.07
4	Sweater	48.17
5	Pants	47.37

People love buying these products at a cheaper price.

```
--Q7. Segment customers into New, Returning, and Loyal based on their total
-- number of previous purchases, and show the count of each segment.
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)

select customer_segment,count(*) AS "Number of Customers"
from customer_type
group by customer_segment;
```

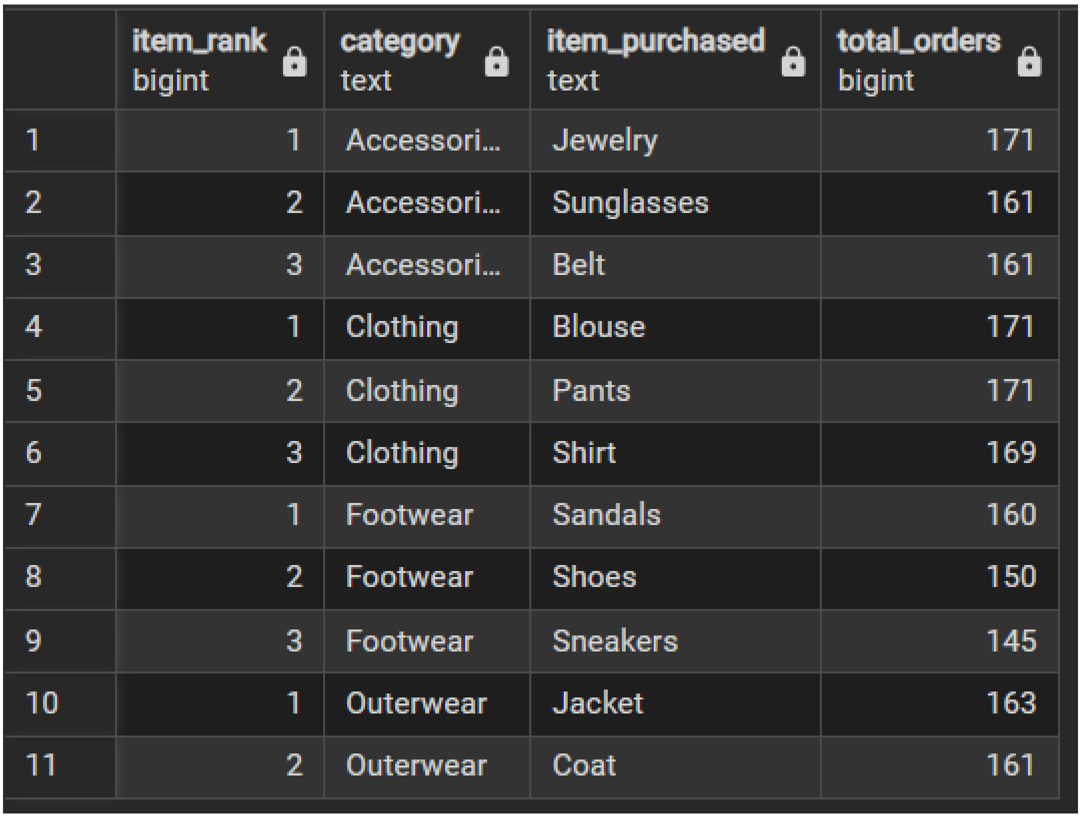


	customer_segment text	Number of Customers bigint
1	Loyal	3116
2	New	83
3	Returning	701

A very few new customers lately.

```
--Q8. What are the top 3 most purchased products within each category?
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
   GROUP BY category, item_purchased
)
```

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

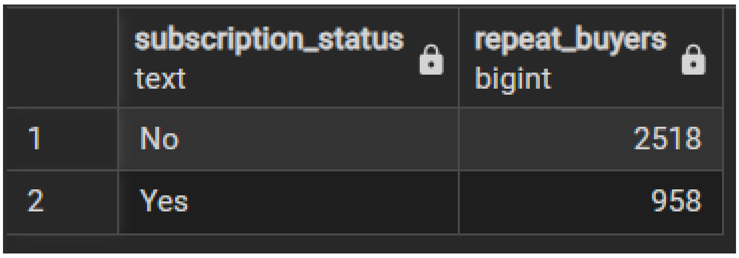


	item_rank bigint	category text	item_purchased text	total_orders bigint
1	1	Accessori...	Jewelry	171
2	2	Accessori...	Sunglasses	161
3	3	Accessori...	Belt	161
4	1	Clothing	Blouse	171
5	2	Clothing	Pants	171
6	3	Clothing	Shirt	169
7	1	Footwear	Sandals	160
8	2	Footwear	Shoes	150
9	3	Footwear	Sneakers	145
10	1	Outerwear	Jacket	163
11	2	Outerwear	Coat	161

Popular products from each of the categories.

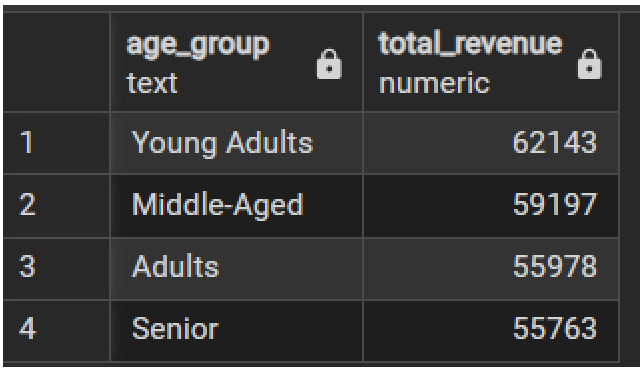
--Q9. Are customers who are repeat buyers (more than 5 previous purchases) also likely to subscribe?

```
SELECT subscription_status,
       COUNT(customer_id) AS repeat_buyers
FROM customer
WHERE previous_purchases > 5
GROUP BY subscription_status;
```



	subscription_status text	repeat_buyers bigint
1	No	2518
2	Yes	958

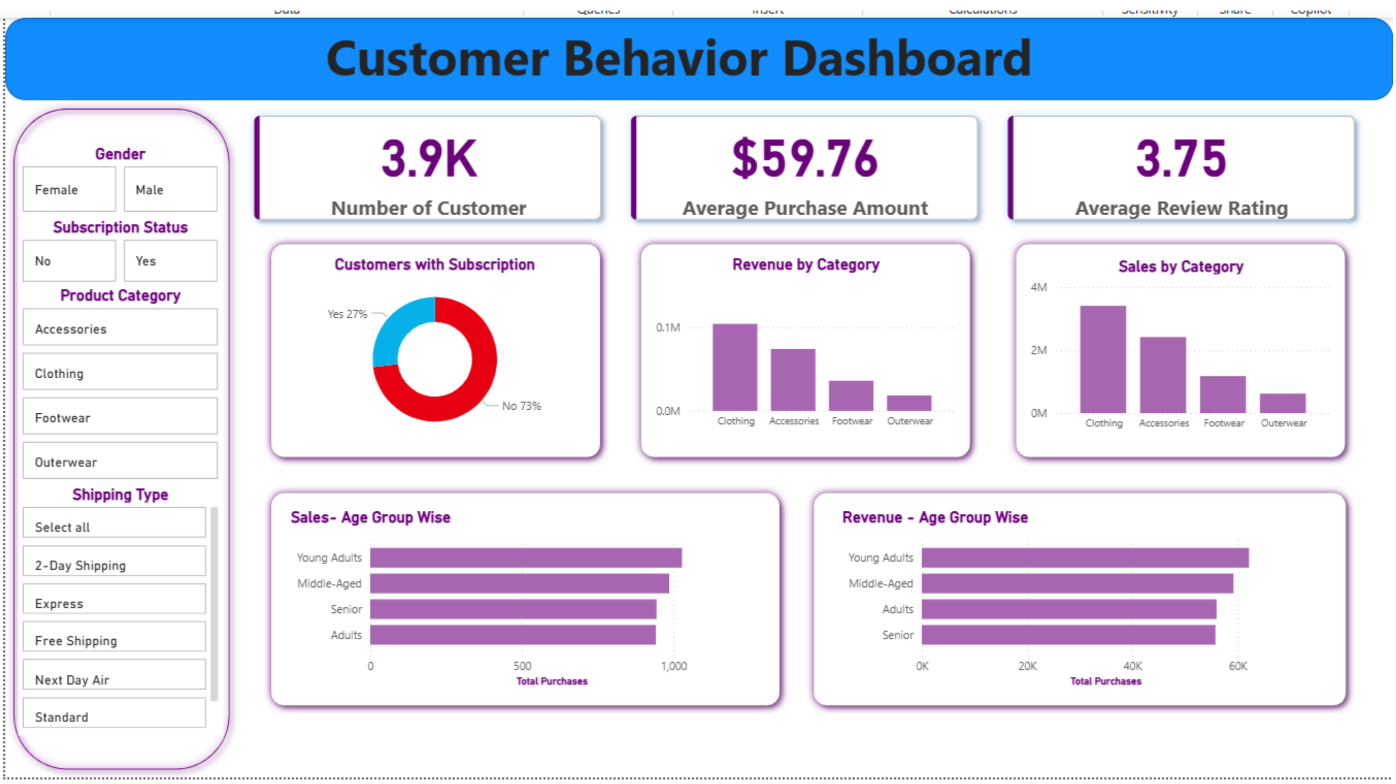
```
--Q10. What is the revenue contribution of each age group?
SELECT
    age_group,
    SUM(purchase_amount) AS total_revenue
FROM customer
GROUP BY age_group
ORDER BY total_revenue desc;
```



	age_group text	total_revenue numeric
1	Young Adults	62143
2	Middle-Aged	59197
3	Adults	55978
4	Senior	55763

Revenue declines as the age goes up.

3) Visualizing it through Power BI:



Customer Behavior Dashboard

3.9K
Number of Customer

$59.76
Average Purchase Amount

3.75
Average Review Rating

Gender
- Female
- Male

Subscription Status
- No
- Yes

Product Category
- Accessories
- Clothing
- Footwear
- Outerwear

Shipping Type
- Select all
- 2-Day Shipping
- Express
- Free Shipping
- Next Day Air
- Standard