Customer Shopping Behaviour Analysis

1. Project Overview

This project analyses the shopping behaviour of the customers 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 behaviour to guide strategic business decisions.

2. Dataset Summary

- Rows : 3900
- Columns : 18
- Key Features :
  - Customer demographics (Age, Gender, Location, Subscription Status)
  - Purchase details (Item Purchased, Category, Purchase Amount, Season, Size, Color)
  - Shopping behaviour (Discount Applied, Promo Code Used, Previous Purchases, Frequency Of Purchases, Review Rating, Shipping Type)
  - Missing data : 37 values in Review Rating column

3. Exploratory Data Analysis using Python

We began with data preparation and cleaning in Python :

- **Data Loading :** Imported the dataset using pandas.
  (Dataset is imported in dataframe named df**)**

- **Initial Exploration :** Used df.info() to check structure and df.describe() for summary statistics.

	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
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000	3900	3900	3900	3900
unique	NaN	NaN	2	25	4	NaN	50	4	25	4	NaN	2	6	2	2
top	NaN	NaN	Male	Blouse	Clothing	NaN	Montana	M	Olive	Spring	NaN	No	Free Shipping	No	No
freq	NaN	NaN	2652	171	1737	NaN	96	1755	177	999	NaN	2847	675	2223	2223
mean	1950.500000	44.068462	NaN	NaN	NaN	59.764359	NaN	NaN	NaN	NaN	3.750065	NaN	NaN	NaN	NaN
std	1125.977353	15.207589	NaN	NaN	NaN	23.685392	NaN	NaN	NaN	NaN	0.716983	NaN	NaN	NaN	NaN
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	NaN	NaN	NaN	NaN	2.500000	NaN	NaN	NaN	NaN
25%	975.750000	31.000000	NaN	NaN	NaN	39.000000	NaN	NaN	NaN	NaN	3.100000	NaN	NaN	NaN	NaN
50%	1950.500000	44.000000	NaN	NaN	NaN	60.000000	NaN	NaN	NaN	NaN	3.800000	NaN	NaN	NaN	NaN
75%	2925.250000	57.000000	NaN	NaN	NaN	81.000000	NaN	NaN	NaN	NaN	4.400000	NaN	NaN	NaN	NaN
max	3900.000000	70.000000	NaN	NaN	NaN	100.000000	NaN	NaN	NaN	NaN	5.000000	NaN	NaN	NaN	NaN

```
 #   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
```

- **Missing Data Handling :** Checked for null values by using df.isnull().sum() and imputed missing values in the Review Rating column using the median rating of each product category.

- **Column Standardization :** Renamed columns to **snake case** for better readability and documentation.

- **Feature Engineering :**

    - Created **age_group** column by binning customer ages.

    - Created **purchase_frequency_days** column from purchase data.

- **Data Consistency Check :** Verified if discount_applied and promo_code_used were redundant; dropped promo_code_used.

- **Database Integration** : Connected Python script to MySQL and loaded the cleaned DataFrame into the database for SQL analysis.

4. Data Analysis using SQL

We performed structured analysis in MySQL to answer key business questions :

1. **Revenue by Gender** – Compared total revenue generated by male vs female customers.

```
mysql> select gender,SUM(purchase_amount) as revenue from customer group by gender;
+--------+---------+
| gender | revenue |
+--------+---------+
| Male   |  157890 |
| Female |   75191 |
+--------+---------+
2 rows in set (0.02 sec)
```

2. **High - Spending Discount Users** – Identified customers who used discounts but still spent above the average purchase amount.

```
mysql> select customer_id,purchase_amount from customer where discount_applied ="Yes" and purchase_amount  >= (select AVG(purchase_amount) from customer);
+-------------+-----------------+
| customer_id | purchase_amount |
+-------------+-----------------+
|           2 |              64 |
|           3 |              73 |
|           4 |              90 |
|           7 |              85 |
|           9 |              97 |
|          12 |              68 |
|          13 |              72 |
|          16 |              81 |
|          20 |              90 |
|          22 |              62 |
|          24 |              88 |
|          29 |              94 |
|          32 |              79 |
|          33 |              67 |
|          35 |              91 |
|          37 |              69 |
|          40 |              60 |
|          41 |              76 |
|          43 |             100 |
|          44 |              69 |
|          55 |              94 |
```

3. **Top 5 Products by Rating** – Found products with the highest average review ratings.

```
mysql> Select item_purchased,ROUND(AVG(review_rating),2) as "Average Product Rating" from customer group by item_purchased order by avg(review_rating) desc
limit 5;
+---------------+------------------------+
| item_purchased | Average Product Rating |
+---------------+------------------------+
| Gloves        |                   3.86 |
| Sandals       |                   3.84 |
| Boots         |                   3.82 |
| Hat           |                    3.8 |
| Skirt         |                   3.78 |
+---------------+------------------------+
```

4. **Shipping Type Comparison** – Compared average purchase amounts between Standard vs Express shipping.

```
mysql> select shipping_type,AVG(purchase_amount) from customer where shipping_type in ("Standard","Express") group by shipping_type;
+---------------+----------------------+
| shipping_type | AVG(purchase_amount) |
+---------------+----------------------+
| Express       |              60.4752 |
| Standard      |              58.4602 |
+---------------+----------------------+
```

5. **Subscribers vs Non-Subscribers** – Compared average spend and total revenue across subscription status.

```
mysql> select subscription_status,COUNT(customer_id),Round(AVG(purchase_Amount),2) as Average_amount,SUM(purchase_Amount) as Total_revenue from customer gro
up by subscription_status;
+-------------------+--------------------+----------------+---------------+
| subscription_status | COUNT(customer_id) | Average_amount | Total_revenue |
+-------------------+--------------------+----------------+---------------+
| Yes               |               1053 |          59.49 |         62645 |
| No                |               2847 |          59.87 |        170436 |
+-------------------+--------------------+----------------+---------------+
```

6. **Discount-Dependent Products** – Identifies 5 products with the highest percentage of discounted purchases.

```
mysql> select item_purchased,ROUND(100 * 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 | discount_rate |
+----------------+---------------+
| Hat            |         50.00 |
| Sneakers       |         49.66 |
| Coat           |         49.07 |
| Sweater        |         48.17 |
| Pants          |         47.37 |
+----------------+---------------+
```

7. **Customer Segmentation** – Classified customers into New, Returning and Loyal segments based on purchase history.

```
mysql> with customer_type as (select customer_id,previous_purchases, CASE WHEN previous_purchases = 1 THEN "New" WHEN previous_purchases BETWEEN 10 AND 20 T
HEN "Returning" ELSE "Loyal" END as customer_segment FROM customer) Select customer_segment,count(*) as "Number of Customer" from customer_type group by cus
tomer_segment order by customer_segment DESC;
+-----------------+--------------------+
| customer_segment | Number of Customer |
+-----------------+--------------------+
| Returning       |                853 |
| New             |                 83 |
| Loyal           |               2964 |
+-----------------+--------------------+
```

8. **Top 3 Products per Category** – Listed the most purchased products with each category.

```
mysql> with item_counts as (select category,item_purchased,COUNT(customer_id) as total_orders,ROW_NUMBER() over(partition by category order by count(custome
r_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_r
ank <= 3;
+-----------+-------------+----------------+--------------+
| item_rank | category    | item_purchased | total_orders |
+-----------+-------------+----------------+--------------+
|         1 | Accessories | Jewelry        |          171 |
|         2 | Accessories | Sunglasses     |          161 |
|         3 | Accessories | Belt           |          161 |
|         1 | Clothing    | Blouse         |          171 |
|         2 | Clothing    | Pants          |          171 |
|         3 | Clothing    | Shirt          |          169 |
|         1 | Footwear    | Sandals        |          160 |
|         2 | Footwear    | Shoes          |          150 |
|         3 | Footwear    | Sneakers       |          145 |
|         1 | Outerwear   | Jacket         |          163 |
|         2 | Outerwear   | Coat           |          161 |
+-----------+-------------+----------------+--------------+
```

9. **Repeat Buyers & Subscriptions** – Checked whether customers with > 5 purchases are more likely to subscribe.

```
mysql> select subscription_status,count(customer_id) as repeat_buyers from customer where previous_purchases > 5 group by subscription_status;
+-------------------+---------------+
| subscription_status | repeat_buyers |
+-------------------+---------------+
| Yes               |           958 |
| No                |          2518 |
+-------------------+---------------+
```

10. **Revenue by Age Group** – Calculated total revenue contribution of each age group.

```
mysql> select age_group,SUM(purchase_amount) as Total_Revenue from customer group by age_group order by Total_Revenue DESC;
+-------------+---------------+
| age_group   | Total_Revenue |
+-------------+---------------+
| Young Adult |         62143 |
| Middle-aged |         59197 |
| Adult       |         55978 |
| Senior      |         55763 |
+-------------+---------------+
```

5. Dashboard in Power BI

The dashboard is created in Power BI to present insights visually.

6. Business Recommendations

- **Boost Subscriptions –** Promote exclusive benefits for subscribers.
- **Customer Loyalty Programs –** Reward repeat buyers to move them into the “Loyal” segment.
- **Review Discount Policy –** Balance sales boosts with margin control.
- **Product Positioning –** Highlights top-rated and best selling products in campaigns.
- **Targeted Marketed –** Focus efforts on high-revenue age groups and express-shipping users.