Customer Shopping Behaviour Analysis

1. Project Overview

This project analyses customer shopping behaviour 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: 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 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.

● **Initial Exploration**: Used **df.info()** to check structure and .**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	Previous Purchases	Payment Method	Frequency of Purchases
count	3900.000000	3900.000000	3900	3900	3900	3900.000000	3900	3900	3900	3900	3863.000000	3900	3900	3900	3900	3900.000000	3900	3900
unique	NaN	NaN	2	25	4	NaN	50	4	25	4	NaN	2	6	2	2	NaN	6	7
top	NaN	NaN	Male	Blouse	Clothing	NaN	Montana	M	Olive	Spring	NaN	No	Free Shipping	No	No	NaN	PayPal	Every 3 Months
freq	NaN	NaN	2652	171	1737	NaN	96	1755	177	999	NaN	2847	675	2223	2223	NaN	677	584
mean	1950.500000	44.068462	NaN	NaN	NaN	59.764359	NaN	NaN	NaN	NaN	3.750065	NaN	NaN	NaN	NaN	25.351538	NaN	NaN
std	1125.977353	15.207589	NaN	NaN	NaN	23.685392	NaN	NaN	NaN	NaN	0.716983	NaN	NaN	NaN	NaN	14.447125	NaN	NaN
min	1.000000	18.000000	NaN	NaN	NaN	20.000000	NaN	NaN	NaN	NaN	2.500000	NaN	NaN	NaN	NaN	1.000000	NaN	NaN
25%	975.750000	31.000000	NaN	NaN	NaN	39.000000	NaN	NaN	NaN	NaN	3.100000	NaN	NaN	NaN	NaN	13.000000	NaN	NaN
50%	1950.500000	44.000000	NaN	NaN	NaN	60.000000	NaN	NaN	NaN	NaN	3.800000	NaN	NaN	NaN	NaN	25.000000	NaN	NaN
75%	2925.250000	57.000000	NaN	NaN	NaN	81.000000	NaN	NaN	NaN	NaN	4.400000	NaN	NaN	NaN	NaN	38.000000	NaN	NaN
max	3900.000000	70.000000	NaN	NaN	NaN	100.000000	NaN	NaN	NaN	NaN	5.000000	NaN	NaN	NaN	NaN	50.000000	NaN	NaN

● **Missing Data Handling**: Checked for null values 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 **SQLite** and loaded the cleaned **Data Frame** into the database for SQL analysis in DB Browser for Sqlite3.

4. Data Analysis using SQL (Business Transactions)

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

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

```
SELECT gender, sum(purchase_amount) as revenue
FROM customer_shopping_data
group by gender
```

OUTPUT :

	gender	revenue
1	Female	75191
2	Male	157890

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

```
SELECT customer_id, purchase_amount
FROM customer_shopping_data
WHERE LOWER(discount_applied) = 'yes'
  AND purchase_amount > (SELECT AVG(purchase_amount) FROM customer_shopping_data);
```

	customer_id	purchase_amount
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

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

```
SELECT item_purchased, avg(review_rating) as "Average Prodcut Rating"
FROM customer_shopping_data
GROUP BY item_purchased
ORDER BY avg(Review_rating) DESC
LIMIT 5;
```

	item_purchased	Average Prodcut Rating
1	Gloves	3.86142857142857
2	Sandals	3.844375
3	Boots	3.81875
4	Hat	3.8012987012987
5	Skirt	3.78481012658228

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

```
SELECT shipping_type,
round(AVG(purchase_amount),2)
FROM customer_shopping_data
WHERE shipping_type in ('Standard','Express')
```

GROUP BY shipping_type

	shipping_type	round(AVG(purchase_amount),2)
1	Express	60.48
2	Standard	58.46

5. Subscribers vs. Non-Subscribers – Compared average spend and total revenue across 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_shopping_data

GROUP BY subscription_status

ORDER BY total_revenue, avg_spend

	subscription_status	total_customers	avg_spend	total_revenue
1	Yes	1053	59.49	62645.0
2	No	2847	59.87	170436.0

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

SELECT item_purchased,

ROUND(100.0 * SUM(CASE WHEN UPPER(discount_applied) = 'YES' THEN 1 ELSE 0 END) / COUNT(*), 2) AS discount_rate

FROM customer_shopping_data

GROUP BY item_purchased

ORDER BY discount_rate DESC

LIMIT 5;

	item_purchased	discount_rate
1	Hat	50.0
2	Sneakers	49.66
3	Coat	49.07
4	Sweater	48.17
5	Pants	47.37

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

```
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_shopping_data
)
SELECT customer_segment, count(*) as "Number of Customers"
FROM customer_type
GROUP BY customer_segment
```

	customer_segment	Number of Customers
1	LOYAL	3116
2	NEW	83
3	RETURNING	701

8. Top 3 Products per Category – Listed the 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_shopping_data
    GROUP BY category, item_purchased
)
SELECT item_rank, category, item_purchased, total_orders
FROM item_counts
WHERE item_rank <= 3;
```

	item_rank	category	item_purchased	total_orders
1	1	Accessories	Jewelry	171
2	2	Accessories	Sunglasses	161
3	3	Accessories	Belt	161
4	1	Clothing	Pants	171
5	2	Clothing	Blouse	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

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

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

	subscription_status	repeat_buyers
1	No	2518
2	Yes	958

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

```
SELECT  age_group, SUM(purchase_amount) AS total_revenue
FROM customer_shopping_data
GROUP BY age_group
ORDER BY total_revenue desc;
```

	age_group	total_revenue
1	Young Adult	62143
2	Middle-aged	59197
3	Adult	55978
4	Senior	55763

5. Dashboard in Power BI

Finally, we built an interactive dashboard 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** – Highlight top-rated and best-selling products in campaigns.
- **Targeted Marketing** – Focus efforts on high-revenue age groups and express-shipping users.