Analysis of Customer Shopping Behavior and Purchasing Patterns

1. Project Overview

This project utilizes Python, SQL and Power BI to analyze a dataset focused on consumer shopping habits. The primary objective is to identify factors that influence purchasing decisions, segment the customer base, and uncover trends in product popularity and payment preferences. The insights derived from the dataset are intended to support data-driven decision-making for marketing strategies and inventory management.

2. Dataset Summary

Rows: 3900
Columns: 18
Customer Demographics : Age, Location , Subscription Status
Purchase Details: Item Purchased , Category , Purchase amount, Season, Size, Colour
Shopping Behavior: Discount Applied, Promo Code used, Previous Purchases, Frequency of purchases , Review Rating , Shipping Type
Missing Rows: Review Rating: 37 Rows

3. Exploratory Data Analysis using Python (Pandas)

a. Loading data : loaded data using pandas.

```
df = pd.read_csv('customer_shopping_behavior.csv')
```

b. Exploration: Used `df.info()` to check structure and `df.describe(include = 'all')` to check summary statistic.

	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

c. **Missing Values:** checked null values using [df.isnull().sum()] and replaced null in Review Rating by median of review rating by each category.

```
df['Review Rating'] = df.groupby('Category')['Review Rating'].transform('median')
```

d. **Column Name:** changed Column name in snake case for readability and documentation.

```
df.columns = df.columns.str.lower()
df.columns = df.columns.str.replace(' ','_')
df = df.rename(columns={'purchase_amount_(usd)':'purchase_amount'})
```

e. **Feature engineering:**
   - Created age_group column by using age.
   - Created `purchase_frequency_days` column from `frequency_of_purchases`.

f. **Data Consistency Check :** Verified the discount_applied and `promo_code_used` redundant: dropped `promo_code_used`.

g. **Database integration:** Connected python script to MySQL Workbench and loaded the cleaned DataFrame to database for SQL analysis.

4. Data Analysis using SQL:

Performed a structred analysis in MySQL to answer key business Problem.

- **Revenue by gender:**

```
SELECT gender , SUM(purchase_amount) as revenue
FROM customers
GROUP BY gender ;
```

gender	revenue
Male	157890
Female	75191

- **High Spending User:**

```
SELECT customer_id , purchase_amount
from customers
where discount_applied = 'Yes' and purchase_amount >= (select avg(purchase_amount) from customers);
```

- **Top 5 product by Rating:**

```
select item_purchased ,avg(review_rating) as Avg_rating
from customers
group by item_purchased
order by avg(review_rating) desc
limit 5 ;
```

item_purchased	Avg_rating
Gloves	3.8000000000000056
Backpack	3.8000000000000047
Boots	3.8000000000000043
Sneakers	3.800000000000004
Shoes	3.8000000000000025

- **Shipping Type:**

```
select shipping_type, round(avg(purchase_amount),2) as avg_price
from customers
where shipping_type in ('Standard','Express')
group by shipping_type ;
```

	shipping_type	avg_price
▶	Express	60.48
	Standard	58.46

- **Subscriber vs Non-subscriber:**

```
select subscription_status , count(customer_id) as total_customers,
round(avg(purchase_amount),2) as avg_spending , round(sum(purchase_amount),2) as
total_revenue
from customers
group by subscription_status ;
```

	subscription_status	total_customers	avg_spending	total_revenue
▶	Yes	1053	59.49	62645
	No	2847	59.87	170436

- **Top 5 highest purchased product with Discount:**

```
select item_purchased ,
round(100*sum(case when discount_applied = 'Yes' then 1 else 0 end)/count(*),2) as
discount_rate
from customers
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

- **Customer Segmentation:**

```
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'
when previous_purchases > 10 then 'Loyal'
end as customer_segment from customers )
select customer_segment,count(*) from customer_type
group by customer_segment ;
```

	customer_segment	count(*)
▶	Loyal	3116
	Returning	701
	New	83

- **Top 3 product by category:**

```
with category_item as (select category , item_purchased , count(customer_id) as
total_quantity,
row_number() over(partition by category order by count(customer_id) desc ) as item_rank
from customers
group by category,item_purchased)
select item_rank,category , item_purchased ,  total_quantity from category_item
having item_rank <= 3;
```

item_rank	category	item_purchased	total_quantity
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

- **Repeat buyers and Subscription:**

```
select subscription_status , count(customer_id) as repeat_buyers
from customers
where previous_purchases > 5
group by subscription_status ;
```

subscription_status	repeat_buyers
Yes	958
No	2518

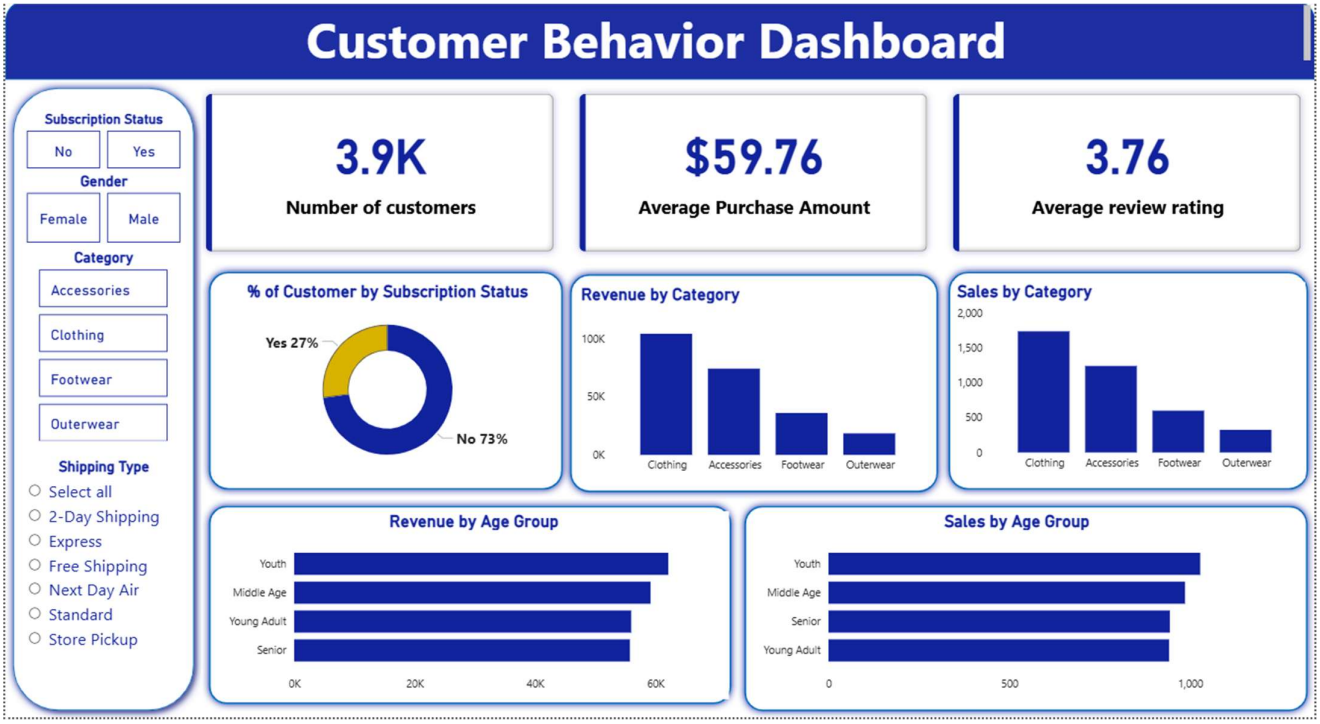
- **Revenue by Age group:**

```
select age_group ,sum(purchase_amount) as total_revenue
from customers
group by age_group
order by total_revenue desc ;
```

age_group	total_revenue
Youth	62143
Middle Age	59197
Young Adult	55978
Senior	55763

5. Dashboard using Power BI:

Created a interactive dashboard using Power BI.



6. Business Recommendation:

- **Revamp the Subscription Value Proposition:** Conduct a survey to understand why the subscription isn't attractive. Consider adding "Express Shipping" (which has a higher average purchase price of $60.48) as a free perk for subscribers to increase adoption and average order value.
- **Aggressively Target the Male Demographic:** Male customers generated $157,890 in revenue, which is more than double the revenue generated by Female customers ($75,191). Shift marketing budget to target male audiences. Highlight top-performing categories for men in ad creatives, as this segment is currently your strongest financial driver.
- **Shift Focus to New Customer Acquisition:** The business has mastered retention but is failing to attract new buyers. Launch "New Customer" exclusive promotions or referral programs specifically designed to boost the "New" segment number from 83.
- **Optimize Inventory for "Youth" and "Clothing":** Ensure inventory planning prioritizes trend-focused clothing items (specifically Blouses and Pants, which are top rankers) to cater to the high-spending Youth demographic.

- **Re-evaluate Discount Strategies on Accessories:** Since "Accessories" (like Jewellery and Belts) are top-ranked items by quantity, try bundling these high-interest items with full-price clothing rather than discounting them individually. This can protect margins while still moving volume.