Data Analysis Project Guideline

Project: Ecommerce Data Analysis

Objective:

The objective of this data analysis project is to gain insights into the ecommerce order data of a fictional company. We will analyze customer information, product details, and order history to derive meaningful conclusions that can aid business decision-making.

Here are some potential analyses we are going to perform:

Customer Analysis:
- Identify the total number of customers city wise.
- Identify the most frequent customers based on their order history.

Product Analysis:
- Determine the total number of products available by category.
- Analyze the distribution of products across sub-categories.
- Identify products with low stock levels.
- Calculate the average, maximum, and minimum prices for products.

Order Analysis:
- Calculate the top 10 orders product wise.
- Analyze the order status distribution (e.g., pending, delivered).
- Identify the most popular products based on order quantity.

Sales Analysis:
- Calculate total revenue generated from orders product wise.
- Calculate the total revenue generated from all orders
- Calculate total revenue product category wise percentage.
- Analyze the performance of different product categories in terms of sales.
- Identify the most profitable products based on the difference between original and selling prices.

Customer Order Patterns:
- Identify product names with the highest and lowest order quantities.
- Identify customers with the highest and lowest order quantities by customer name.
- Determine the most preferred payment modes.

Time-based Analysis:
- Month wise total sales.
- Month and year wise total sales
- Identify peak order date.

Geographical Analysis:
- Explore the distribution of customers across different cities.

- Analyze whether certain products or categories are more popular in a specific city.

Product Performance:

- Identify the top 10 best-selling products.
- Identify top 10 slow-moving products based on low sales.

Customer Retention:

- Analyze repeat customers and their order patterns.
- Calculate customer retention rates over time.

Payment Analysis:

- Display successful and pending payments order counts.

Dataset Overview:

The dataset consists of three main tables:

`customer`: Contains information about customers, including their ID, name, contact details, and address.

```
CREATE TABLE `customer` (
  `customer_id` varchar(10) NOT NULL,
  `name` varchar(100) NOT NULL,
  `city` varchar(65) NOT NULL,
  `email` varchar(45) NOT NULL,
  `phone_no` varchar(15) NOT NULL,
  `address` varchar(100) NOT NULL,
  `pin_code` int NOT NULL,
  PRIMARY KEY (`customer_id`)
) ;
```

`product`: Contains details about the products available for purchase, including product ID, name, category, and pricing.

```
CREATE TABLE `product` (
  `product_id` varchar(10) NOT NULL,
  `product_name` varchar(100) NOT NULL,
  `category` varchar(65) NOT NULL,
  `sub_category` varchar(45) NOT NULL,
  `original_price` double NOT NULL,
  `selling_price` double NOT NULL,
```

```
  `stock` int NOT NULL,
  PRIMARY KEY (`product_id`)
);
```

`order_details`: Captures information about customer orders, including order ID, customer ID, product ID, quantity, total price, payment mode, order date, and order status.

```
CREATE TABLE `order_details` (
  `order_id` int NOT NULL AUTO_INCREMENT,
  `customer_id` varchar(10) NOT NULL,
  `product_id` varchar(10) NOT NULL,
  `quantity` double NOT NULL,
  `total_price` double NOT NULL,
  `payment_mode` varchar(60) NOT NULL,
  `order_date` datetime DEFAULT NULL,
  `order_status` varchar(20) NOT NULL,
  PRIMARY KEY (`order_id`),
  KEY `customer_id` (`customer_id`),
  KEY `product_id` (`product_id`),
  CONSTRAINT `order_details_ibfk_1` FOREIGN KEY (`customer_id`)
REFERENCES `customer` (`customer_id`),
  CONSTRAINT `order_details_ibfk_2` FOREIGN KEY (`product_id`)
REFERENCES `product` (`product_id`)
);
```

Step to be perform:

Database Connectivity:

To perform the analysis, we'll establish a connection to the MySQL database containing the ecommerce data. We'll use the appropriate Python libraries, such as `pandas` and `mysql`, to fetch and manipulate the data directly from the database.

```
# Example Python code for connecting to MySQL database using
mysql-connector-python
import mysql.connector
import pandas as pd

# Replace 'your_username', 'your_password', 'your_host', and 'your_database' with
actual credentials
connection = mysql.connector.connect(
    user='root',
    password='',
    host='localhost',
    database='ecommerce'
)

# Create a cursor object to execute SQL queries
cursor = connection.cursor()

# Query data from the 'customer' table
cursor.execute('SELECT * FROM customer')
#After fetching data from the database we are storing it into Pandas DataFrame
customer_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in
cursor.description])

# Query data from the 'product' table
cursor.execute('SELECT * FROM product')
product_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in
cursor.description])

# Query data from the 'order_details' table
cursor.execute('SELECT * FROM order_details')
order_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in
cursor.description])

#printing first 5 records from each table
print(customer_data.head())
print(product_data.head())
print(order_data.head())
```

Output:

	customer_id	name	city	email	phone_no	address	pin_code
0	C1001	Steve	Tokyo	steve@gmail.com	4567897652	f.g.road	99
1	C1002	john	Sydney	john@gmail.com	9987234567	k.c.road	75001
2	C1003	Peter	Kanagawa	peter.parker@mail.com	9969834567	2F Ikenobecho	171
3	C1004	Jackson	Tokyo	Jackson@gmail.com	7765834567	24-2, Sendagaya	8429
4	C1005	Jack	Lake Buena Vista	Jack@gmail.com	8876345678	1520 E Buena Vista Drive	32830

	product_id	product_name	category	sub_category	original_price	selling_price	stock
0	P102	Chair	furniture	Chairs	20000.0	15000.00	10
1	P103	Laptop	Electronics	computer	60000.0	55000.00	50
2	P104	Smartphone	Electronics	phone	45000.0	40000.00	20
3	P105	Blender	Appliance	Electronics	500.0	450.00	10
4	P106	Laptop HP	Electronics	Computers	67200.0	55000.99	50

	order_id	customer_id	product_id	quantity	total_price	payment_mode	order_date	order_status
0	3	C1004	P112	1.0	1000.0	COD	2023-11-30	Pending
1	4	C1005	P102	1.0	20000.0	COD	2023-11-30	Pending
2	5	C1005	P102	1.0	20000.0	COD	2023-12-08	Delivered
3	7	C1006	P103	1.0	55000.0	COD	2023-12-15	Delivered
4	8	C1006	P102	1.0	15000.0	COD	2023-12-01	delivered

Description:

customer_data = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in cursor.description])

cursor.fetchall(): This part of the code fetches all the rows from the result set obtained from a database query using the cursor object. The fetchall() method retrieves all the rows as a list of tuples.

columns=[desc[0] for desc in cursor.description]: This part creates a list of column names for the DataFrame. It uses a list comprehension to iterate over the

cursor.description, which is a list of 7-item tuples describing the columns in the result set. The [desc[0] for desc in cursor.description] extracts the first element (column name) from each tuple in the cursor.description and creates a list of column names.

pd.DataFrame(...): This part creates a Pandas DataFrame using the pd.DataFrame() constructor. It takes the fetched data (result of the query) and the list of column names as arguments. The DataFrame is assigned to the variable customer_data.

In summary, this line of code fetches data from a database using a cursor, extracts column names from the cursor description, and creates a Pandas DataFrame with the fetched data and column names. The resulting DataFrame (customer_data) can be used for further analysis or manipulation using Pandas functionalities.

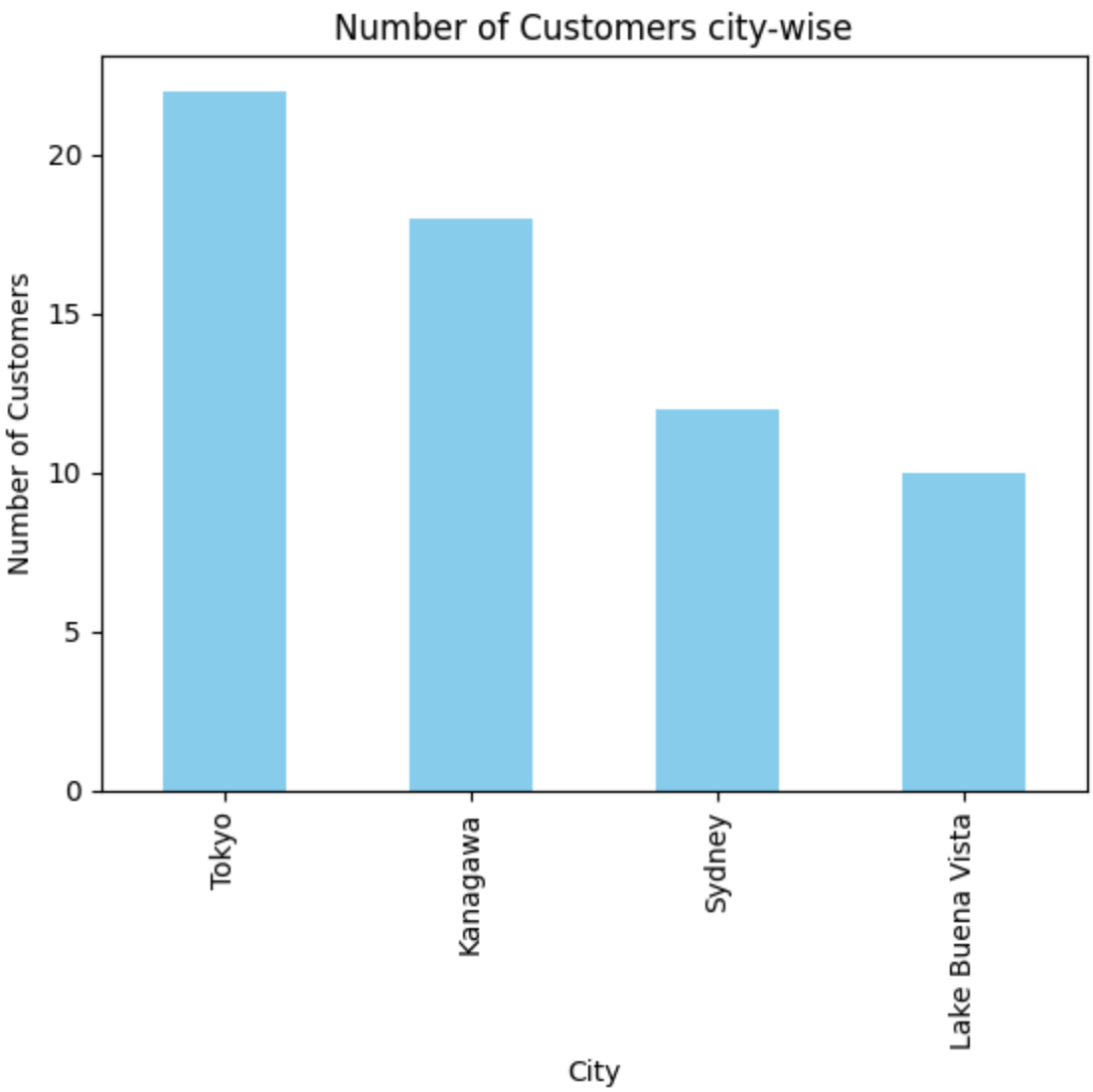
Data Cleaning:

Before proceeding with the analysis, let's perform some basic data cleaning:

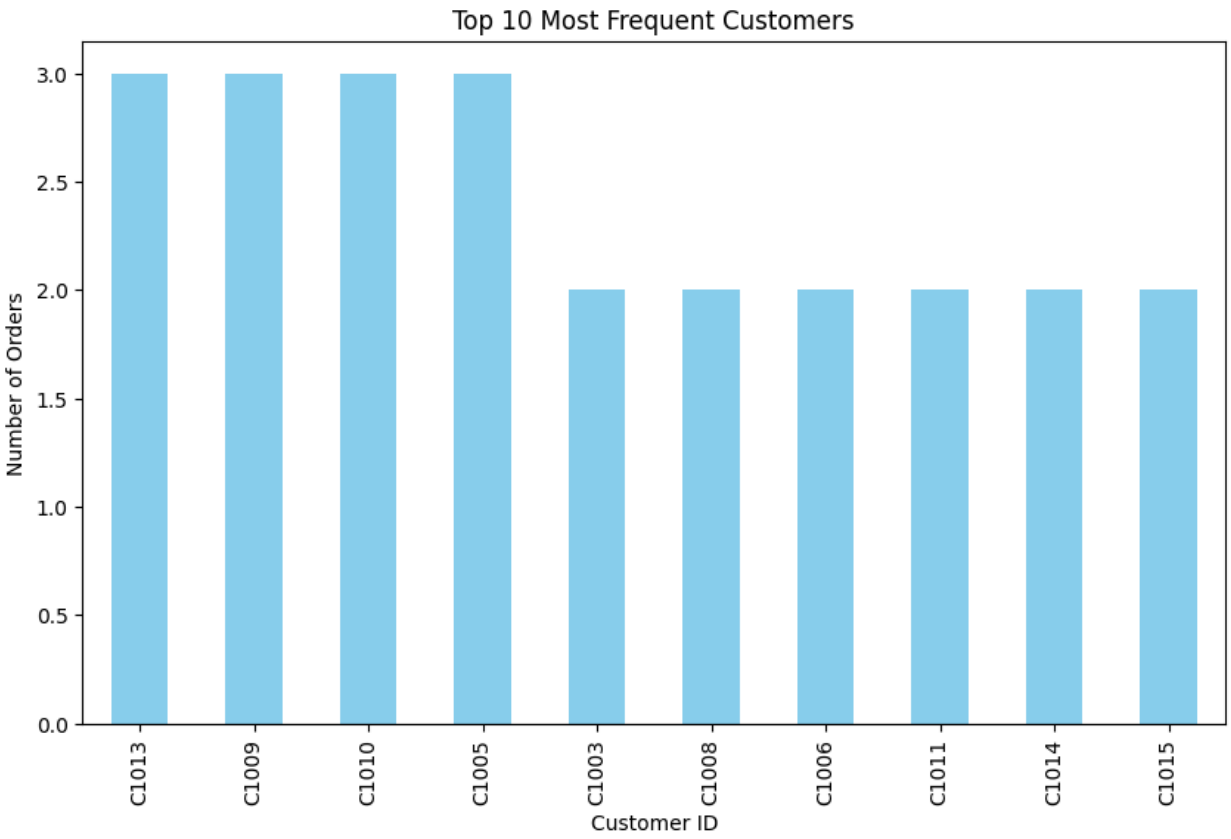
- Check for missing values in each table.
- Ensure data types are appropriate for each column.
- Handle any outliers or inconsistencies.

Exploratory Data Analysis (EDA) and Visualization:

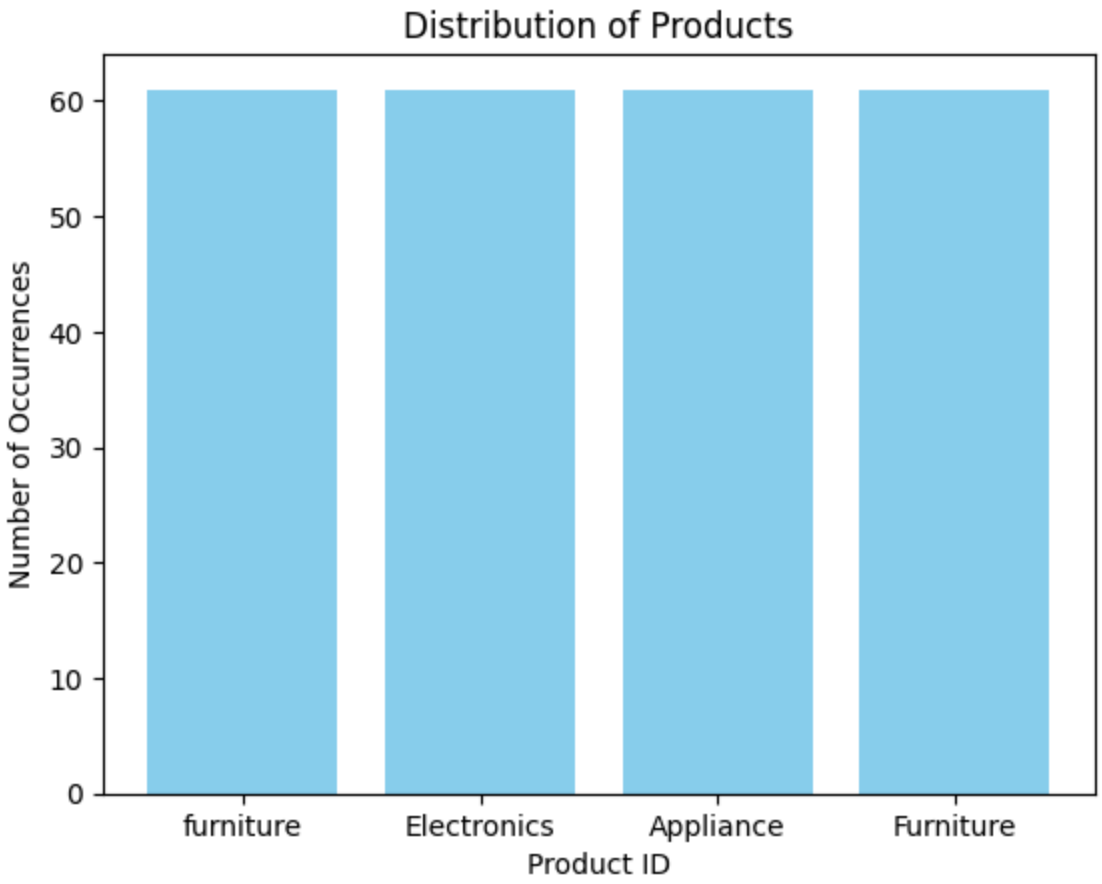
- Customer Analysis:
    - Identify the total number of customers City wise.



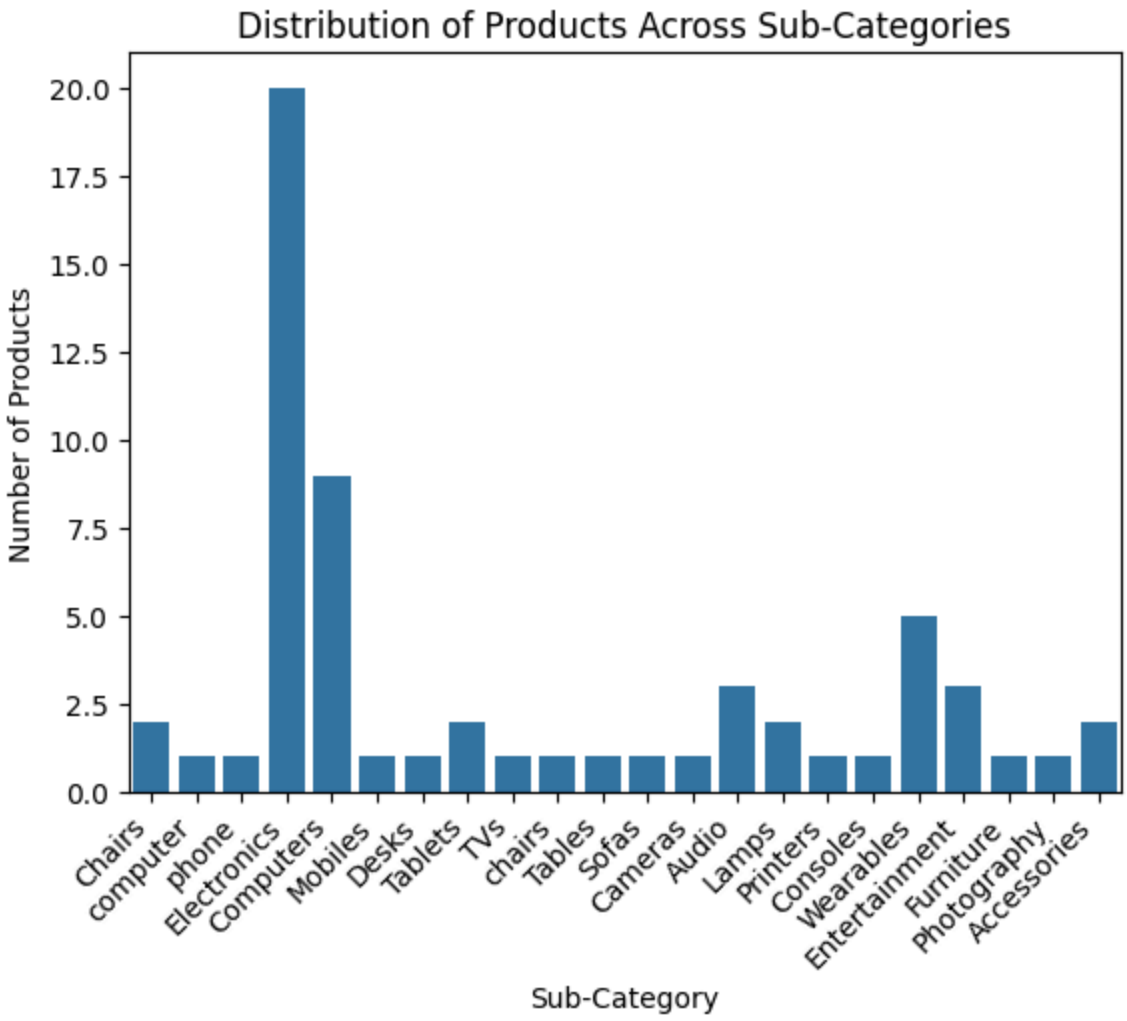
- Identify the most frequent customers based on their order history.



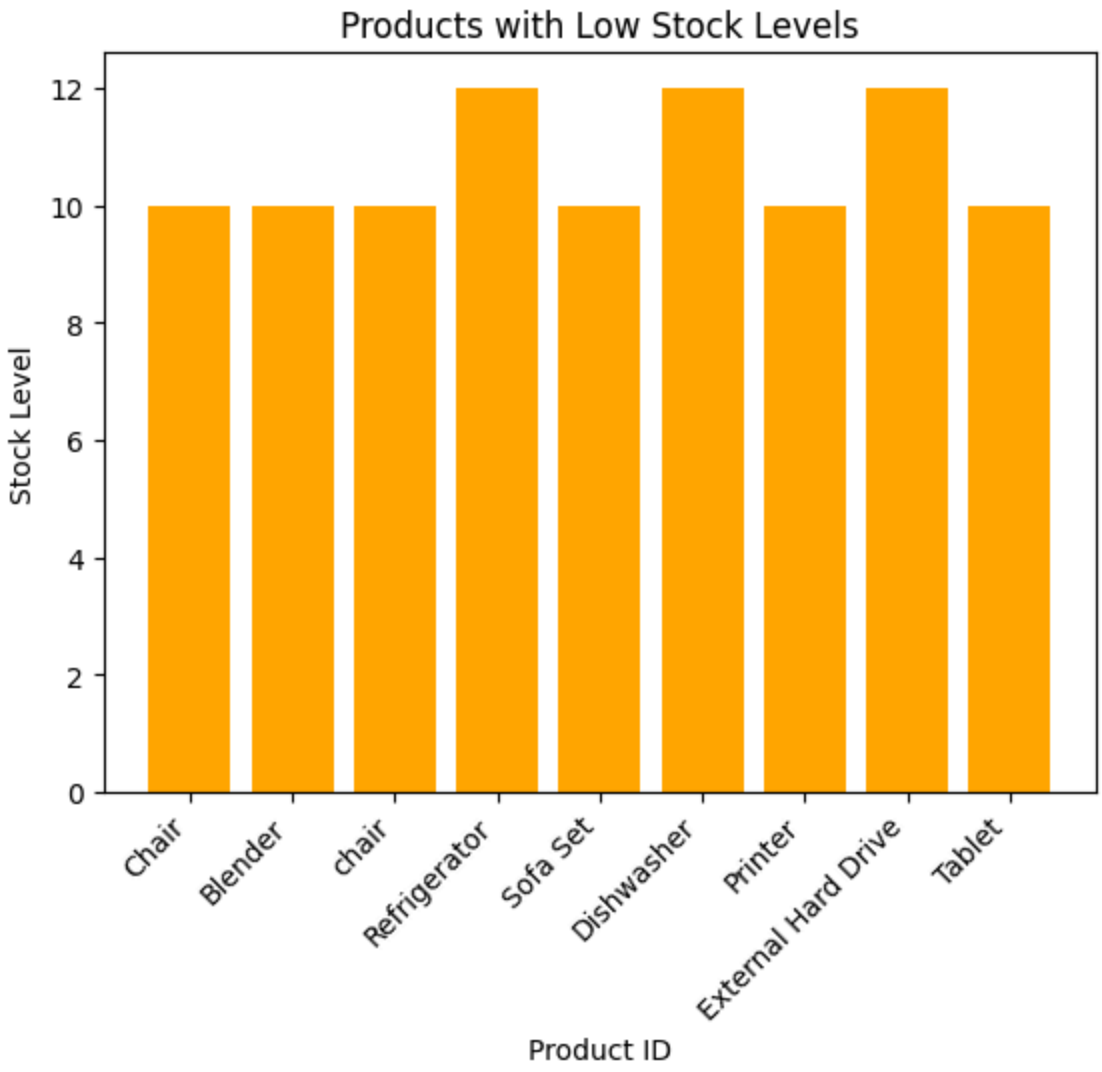
- Product Analysis:
    - Determine the total number of products available by category.



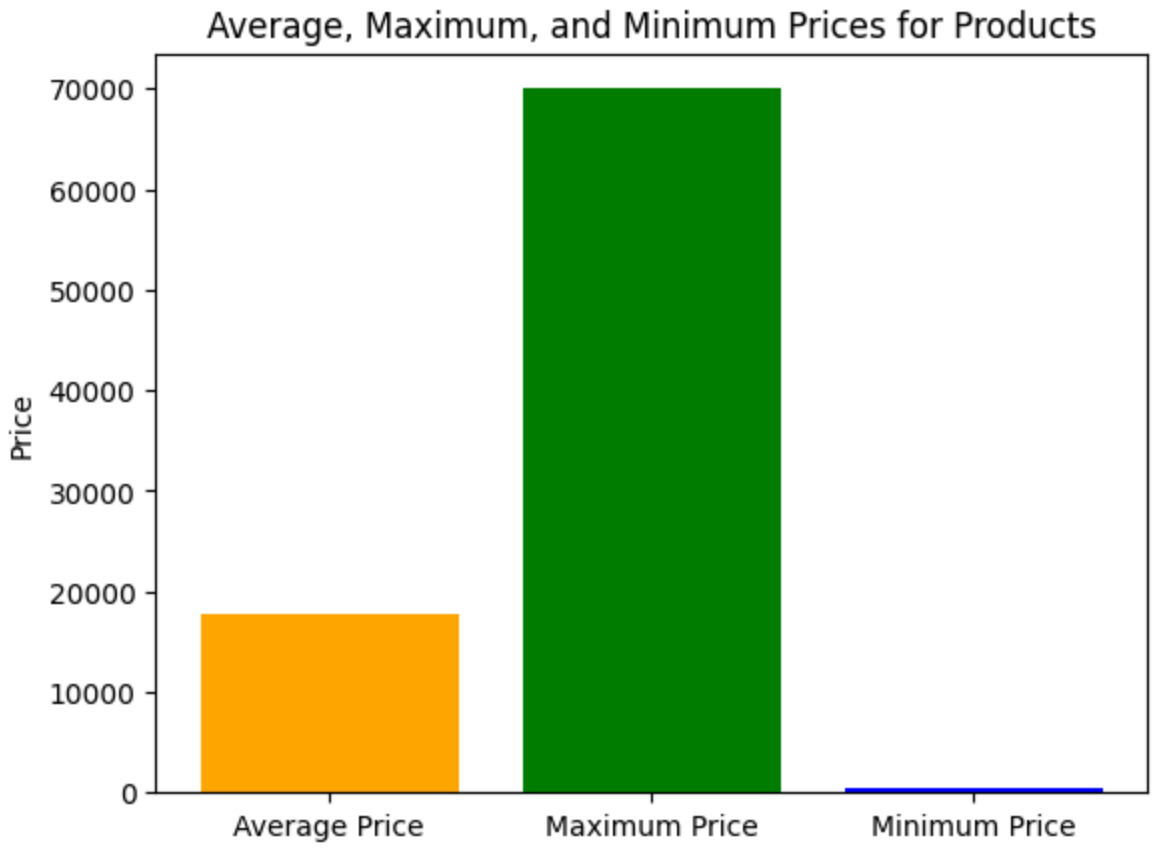
- Analyze the distribution of products across sub-categories.



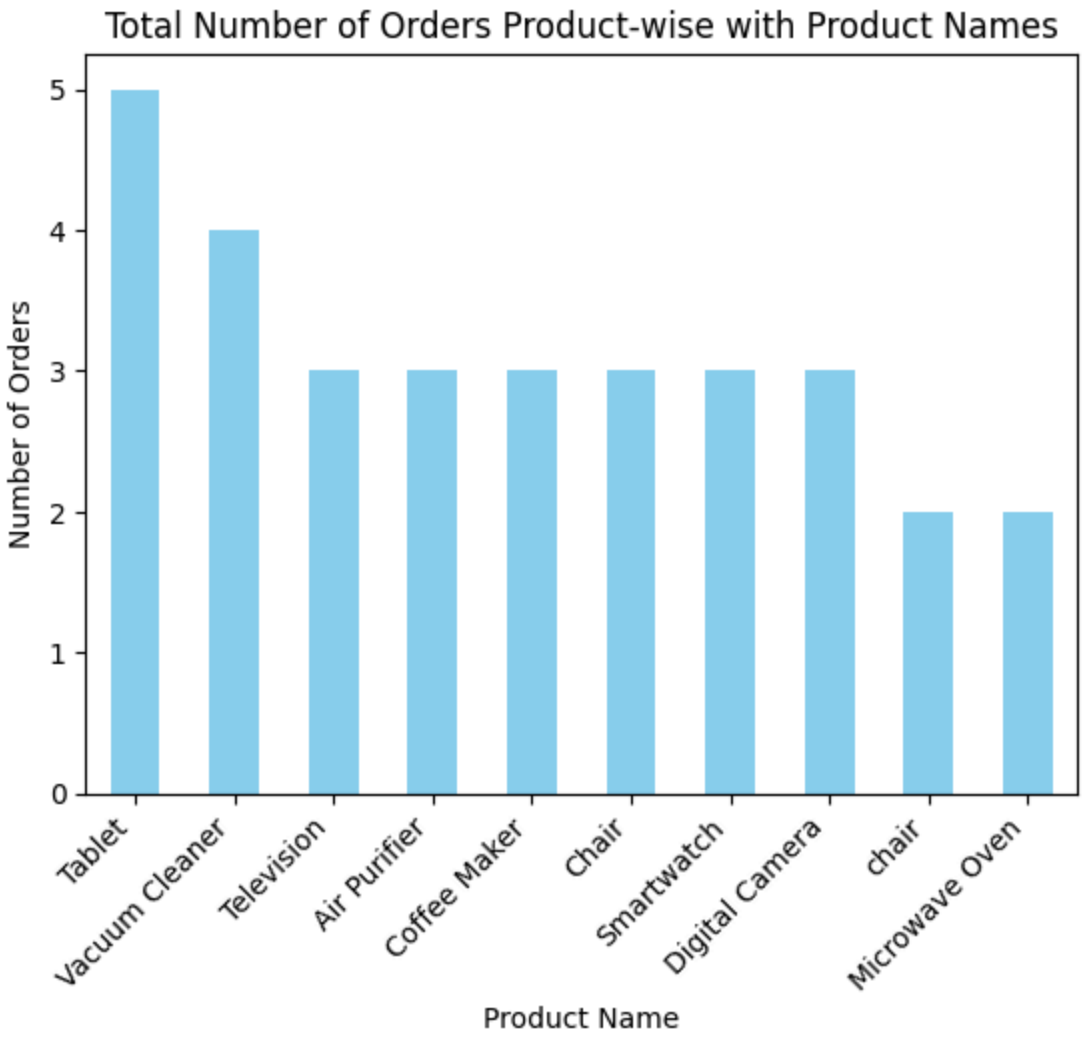
- Identify products with low stock levels.



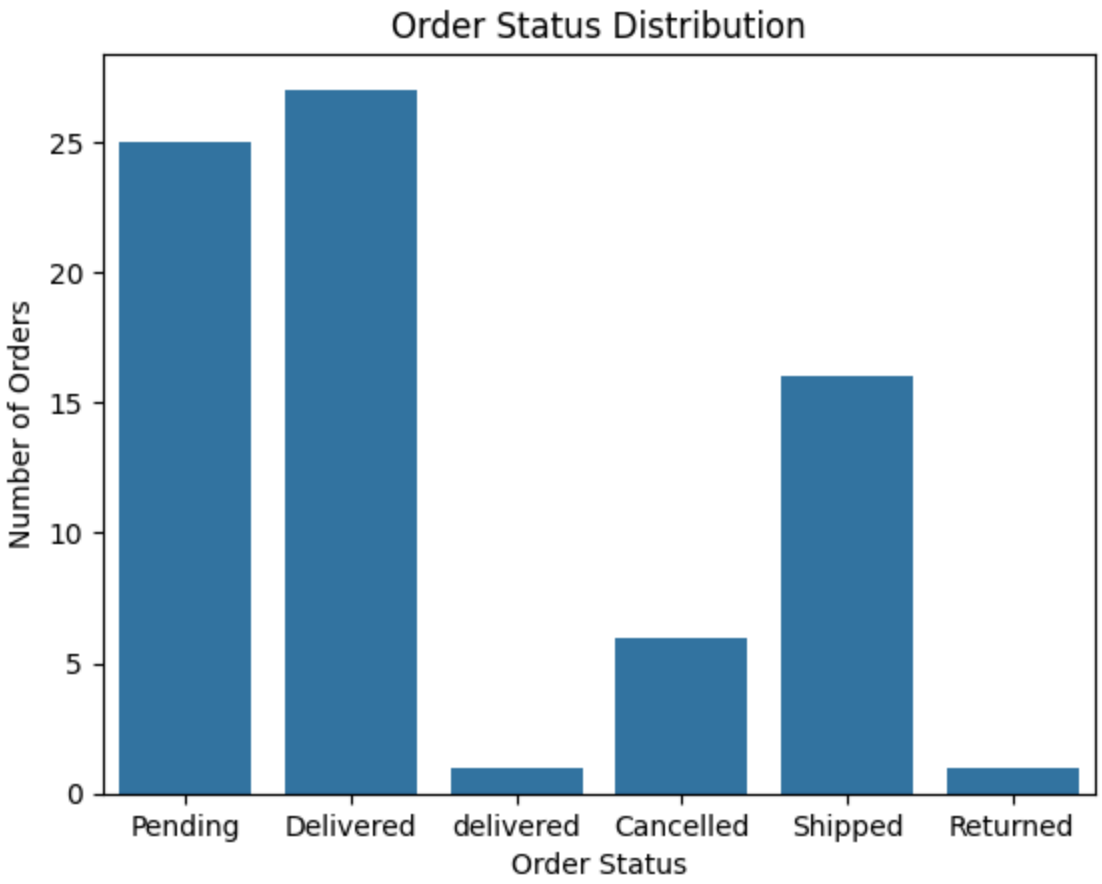
- Calculate the average, maximum, and minimum selling prices for products.



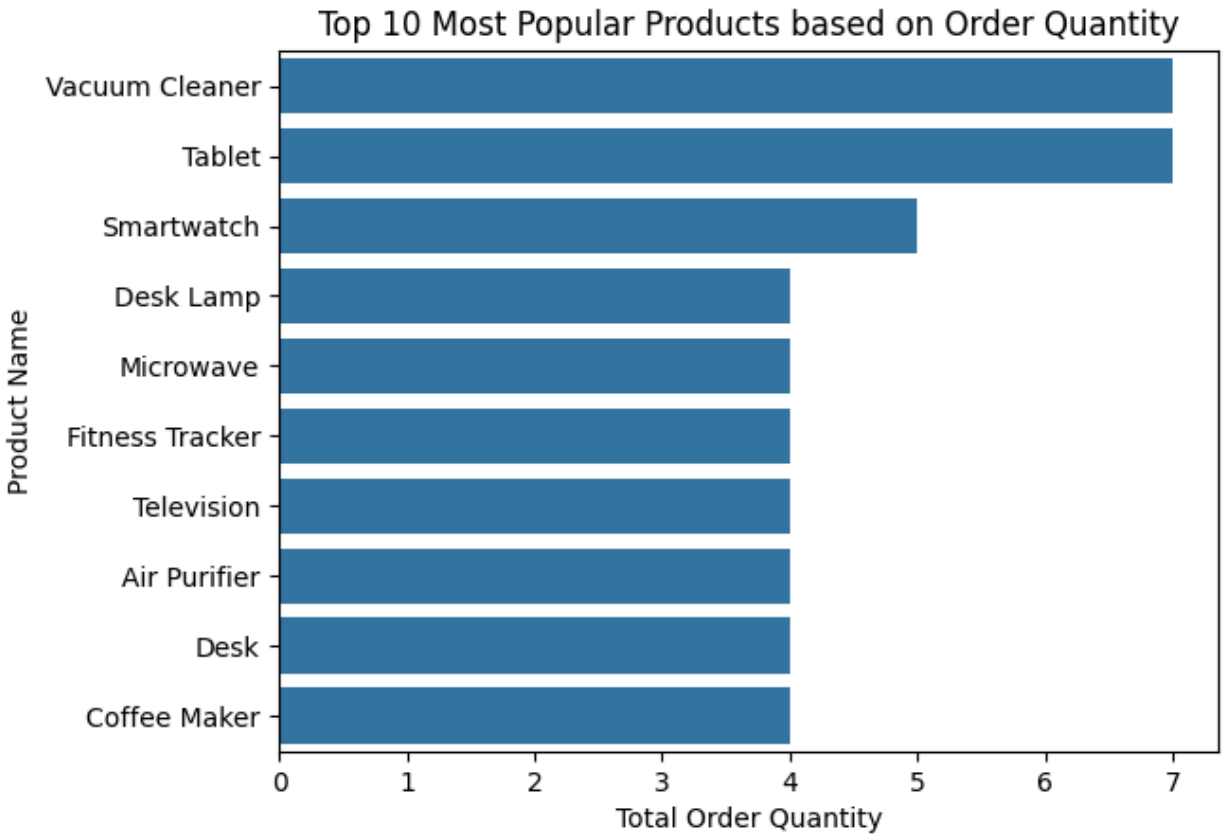
- Order Analysis:
    - Calculate the top 10 orders product wise.



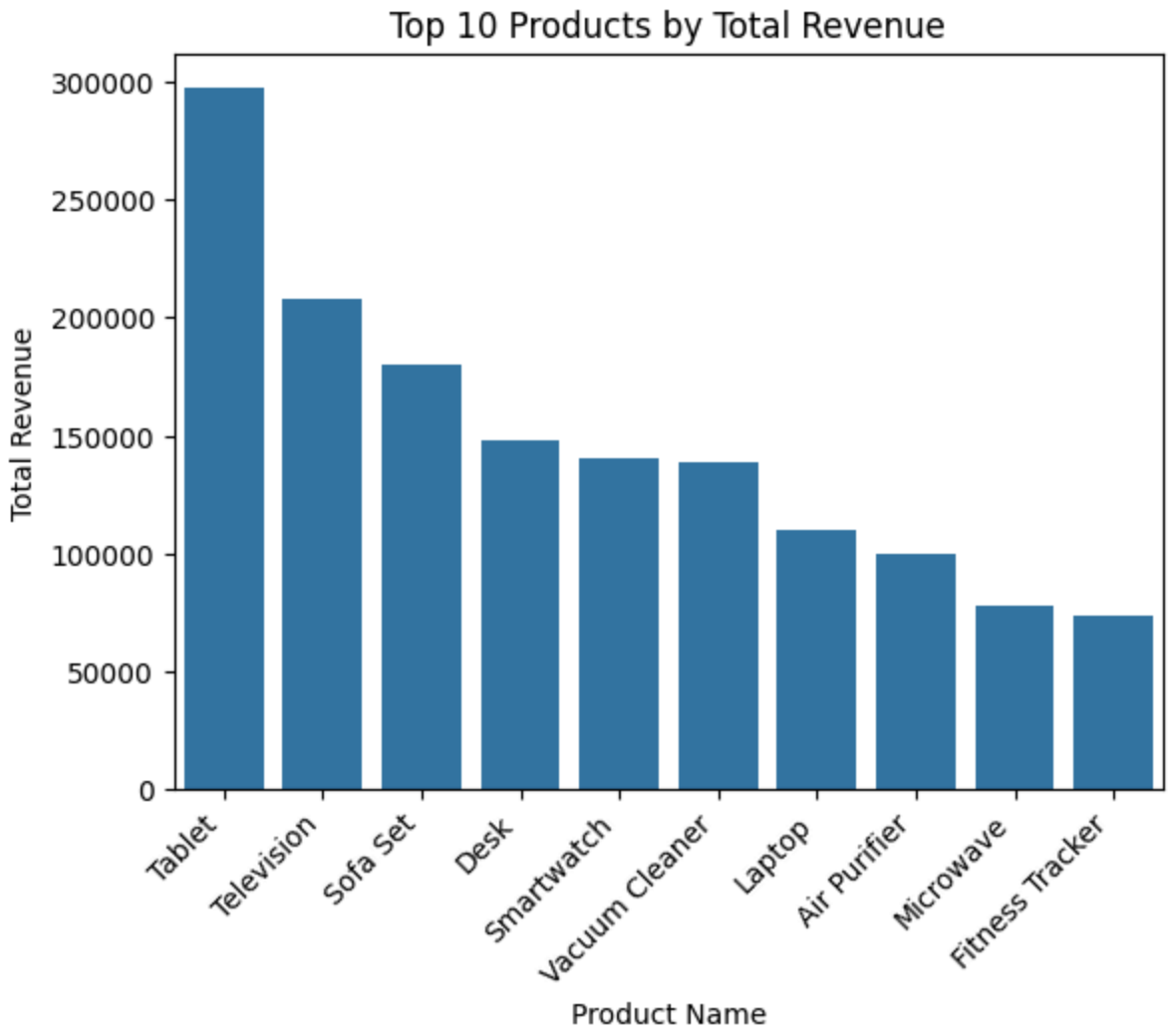
- Analyze the order status distribution (e.g., pending, delivered).



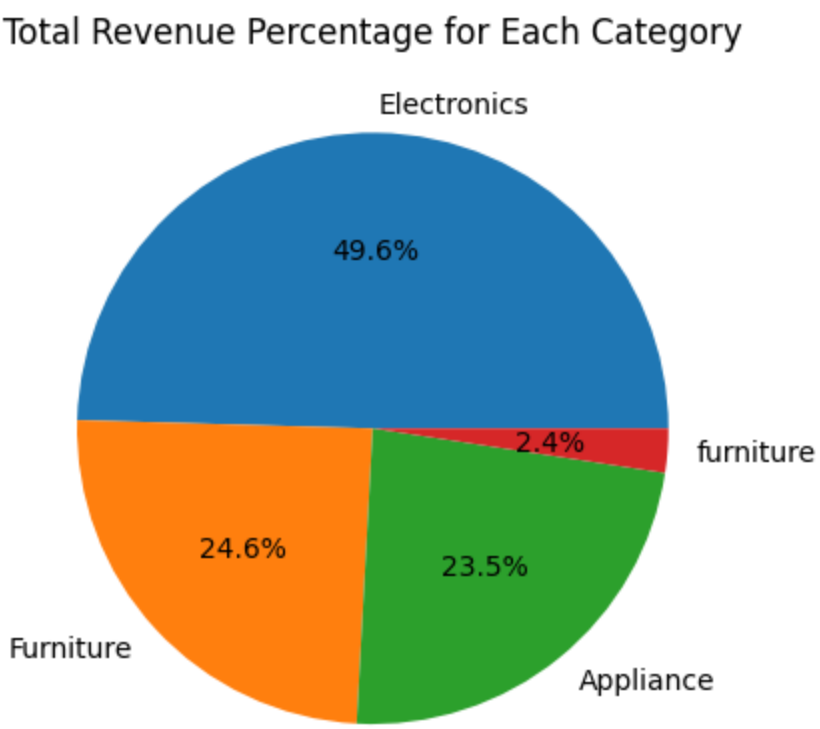
- Identify the most popular products based on order quantity.



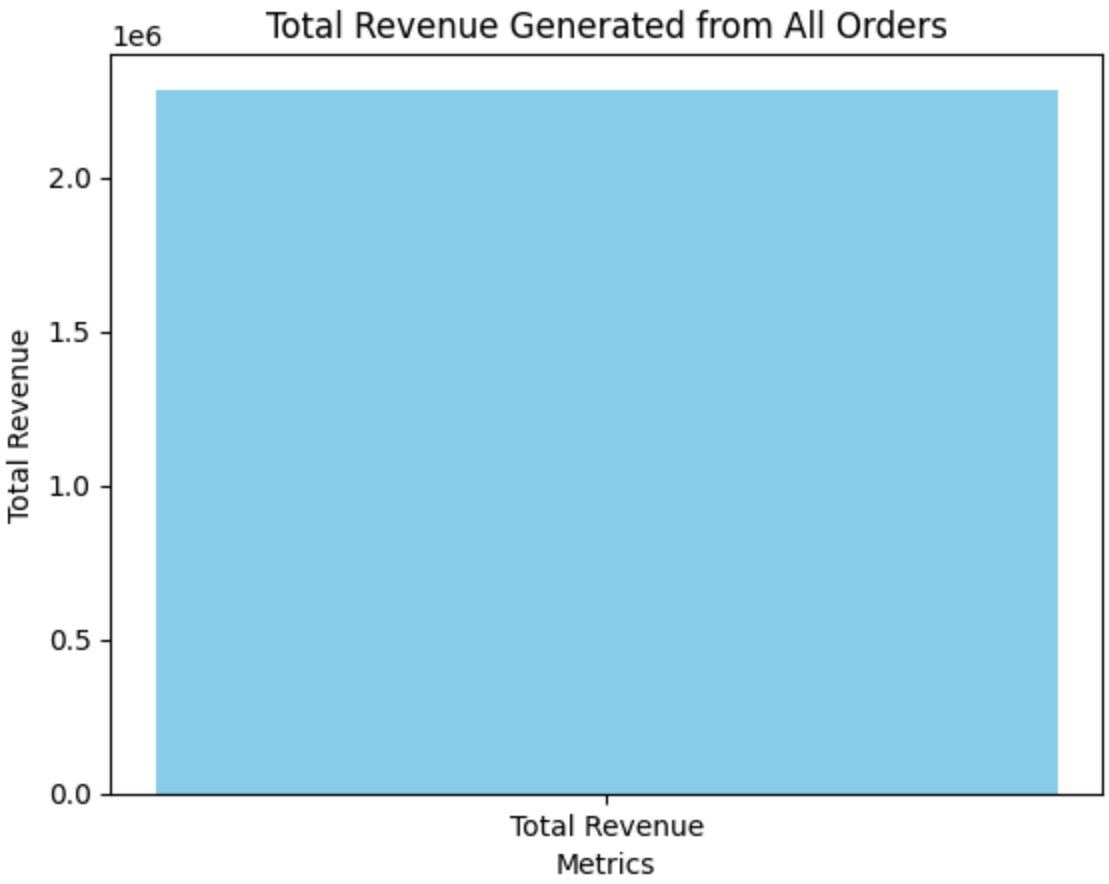
- Sales Analysis:
    - Calculate total revenue generated from orders product wise.



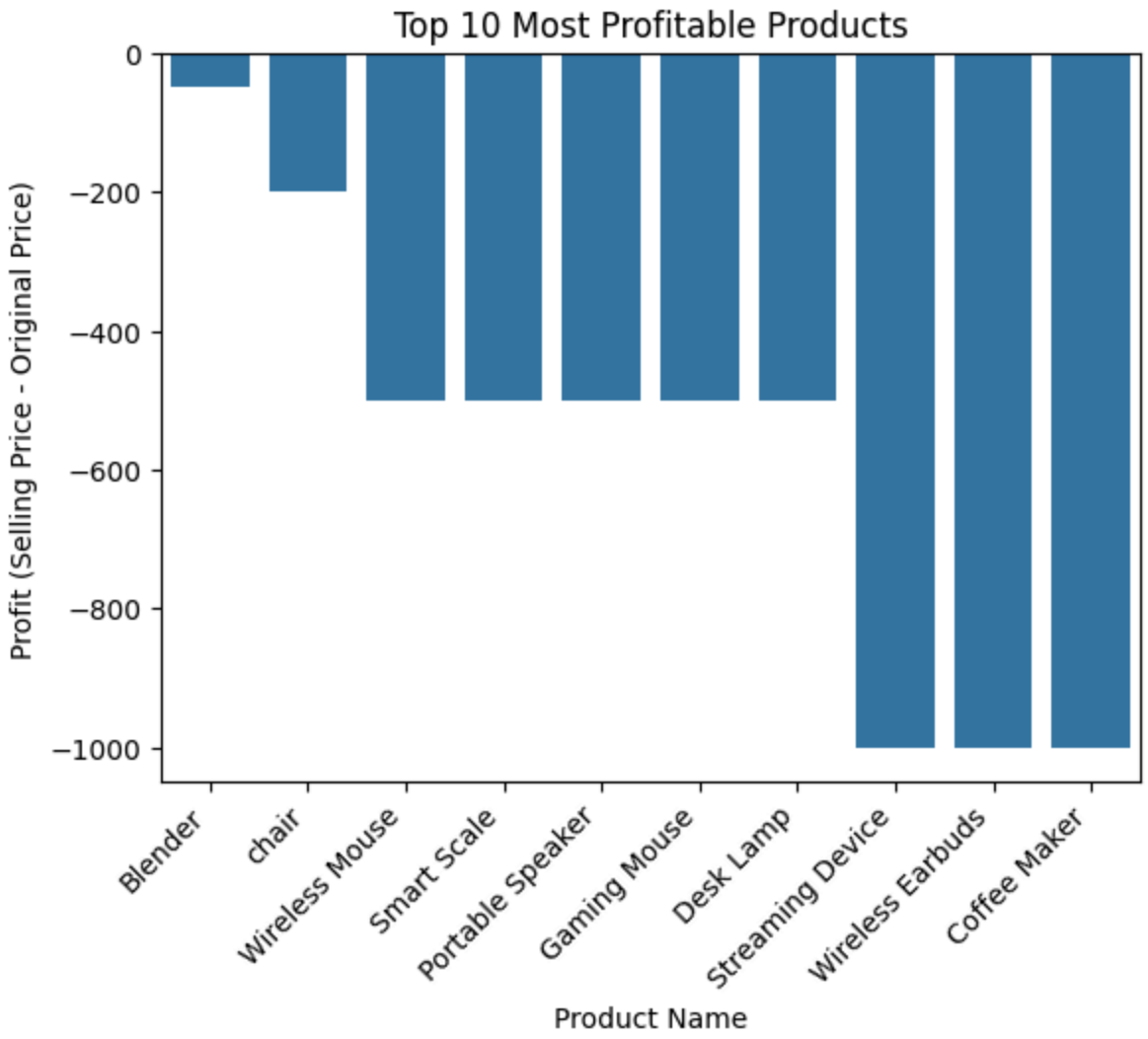
- Calculate total revenue product category wise percentage.



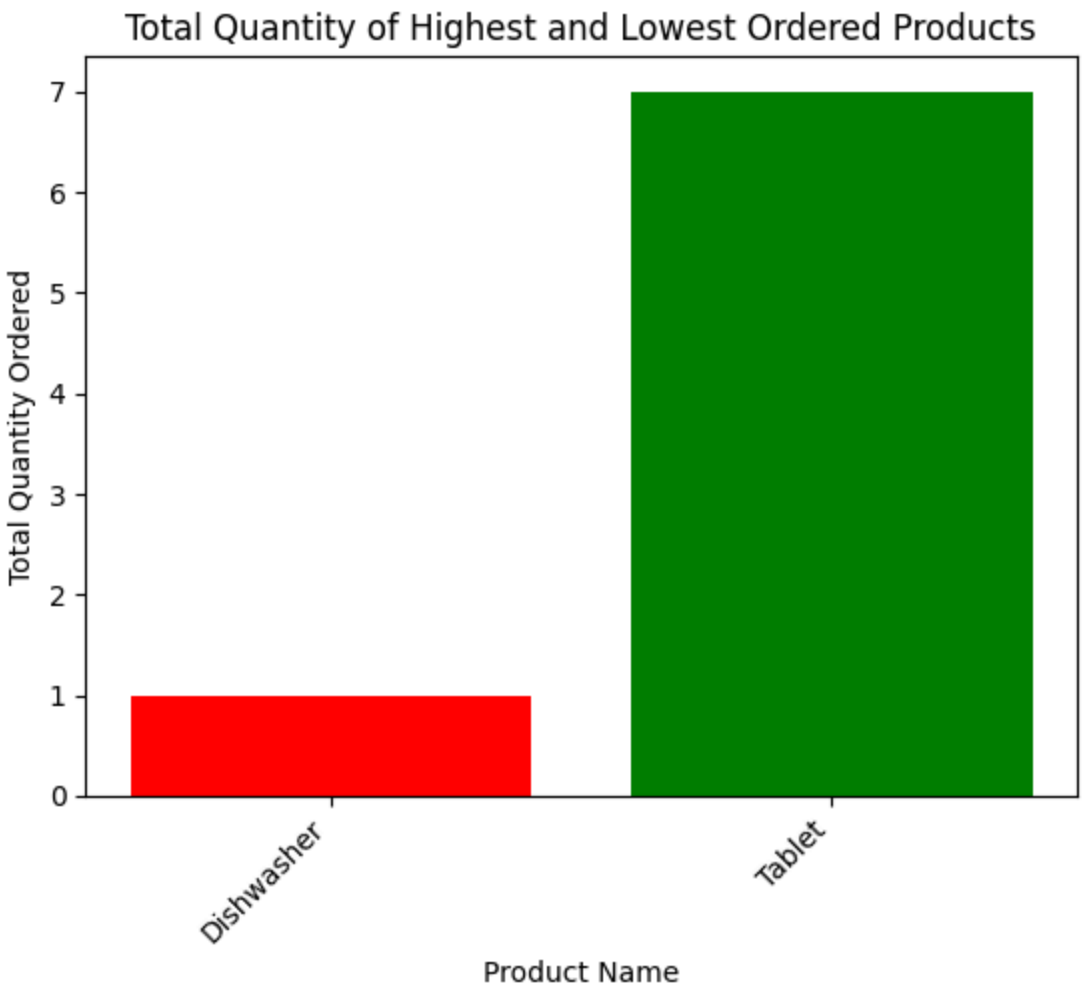
- Calculate the total revenue generated from all orders



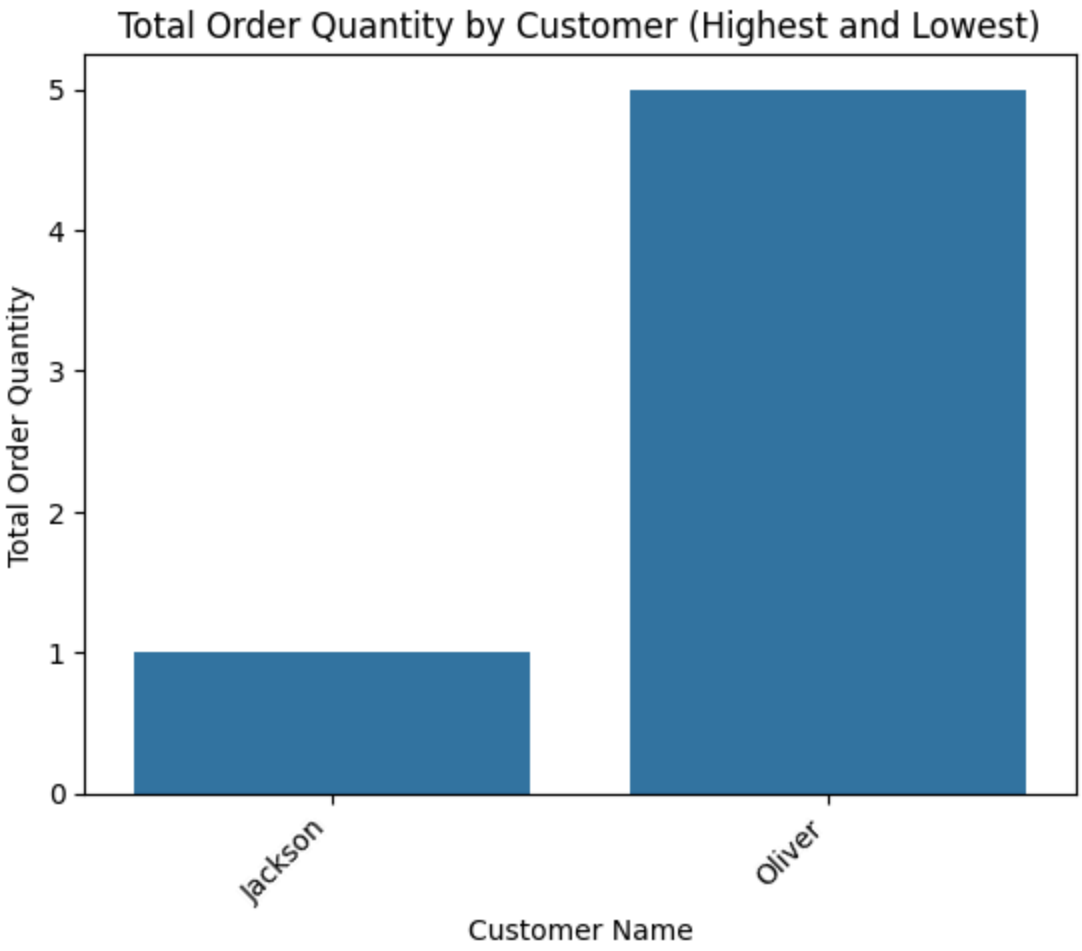
- Identify the most profitable products based on the difference between original and selling prices.



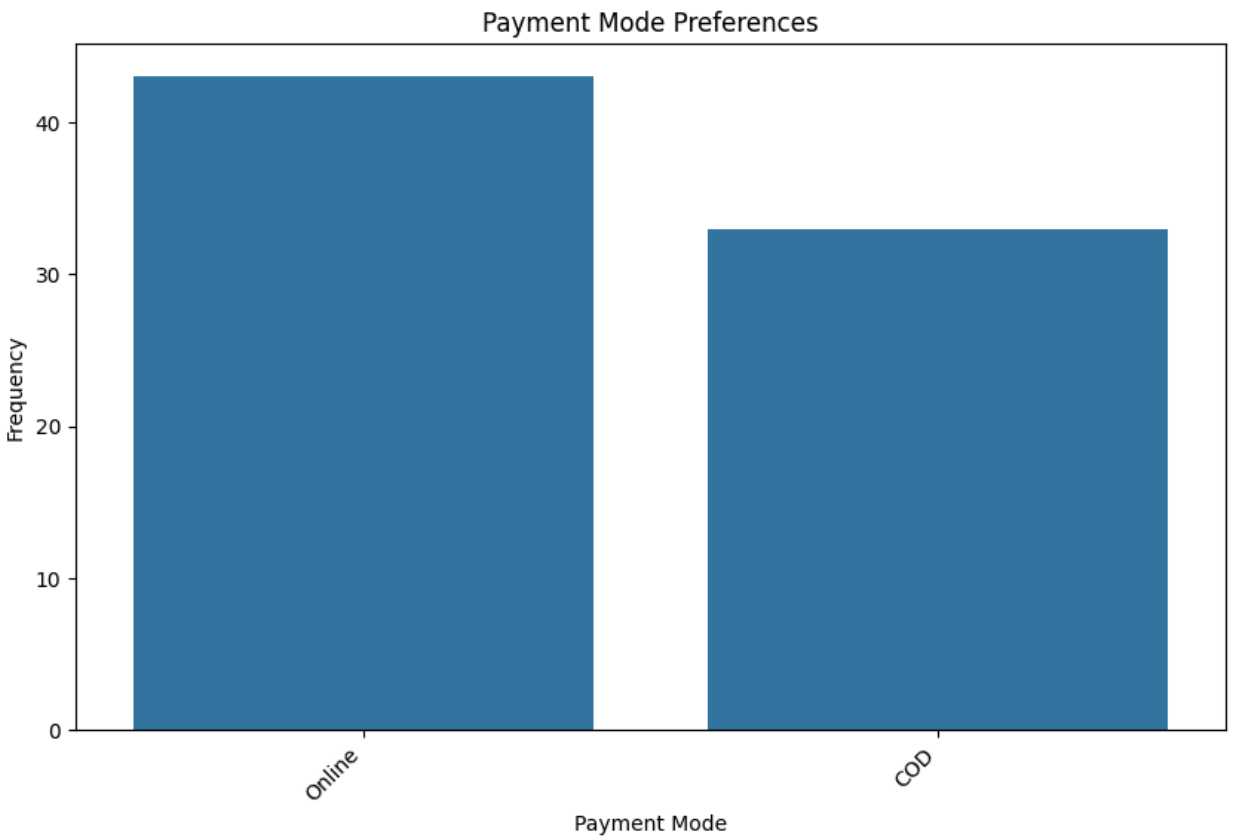
- Customer Order Patterns:
    - Identify product names with the highest and lowest order quantities.



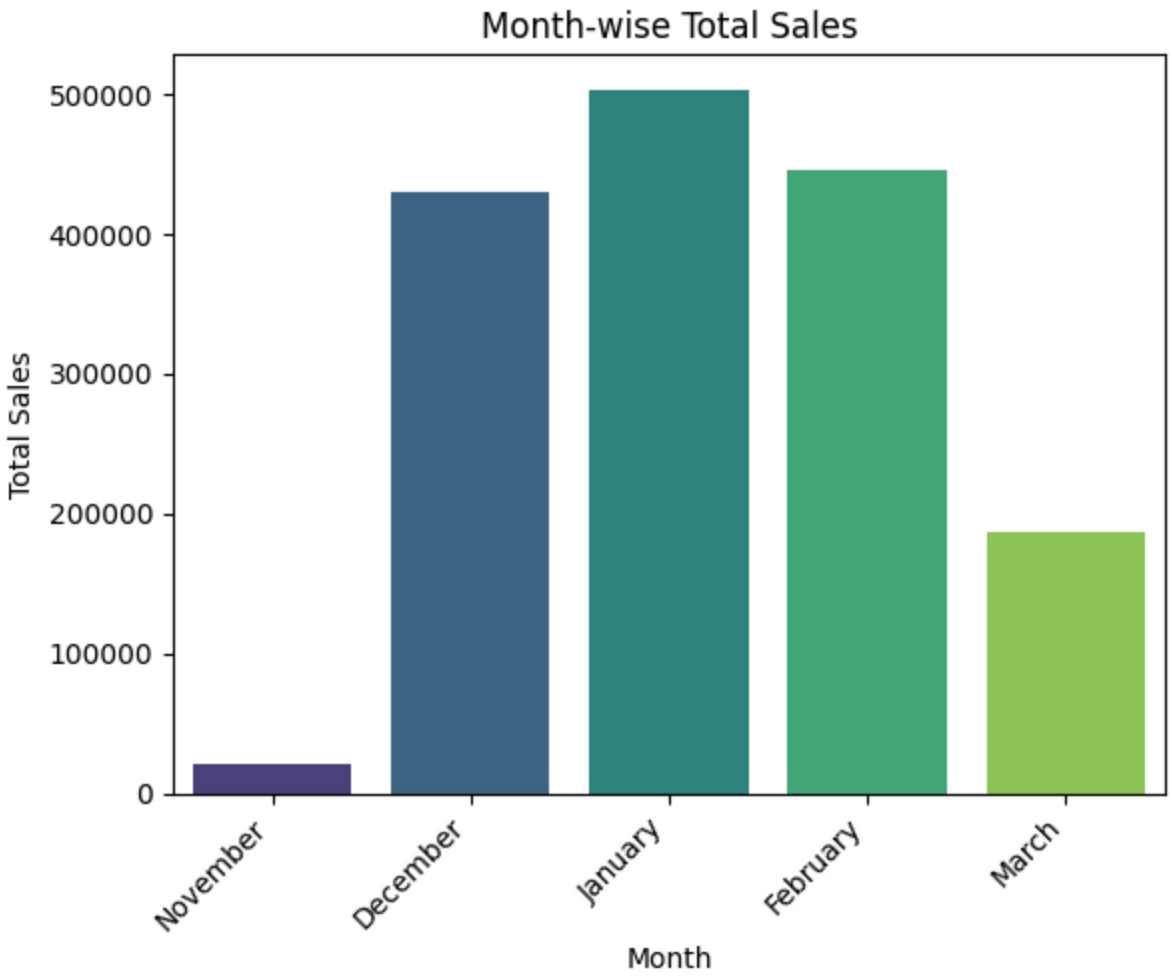
- Identify customers with the highest and lowest order quantities by customer name.



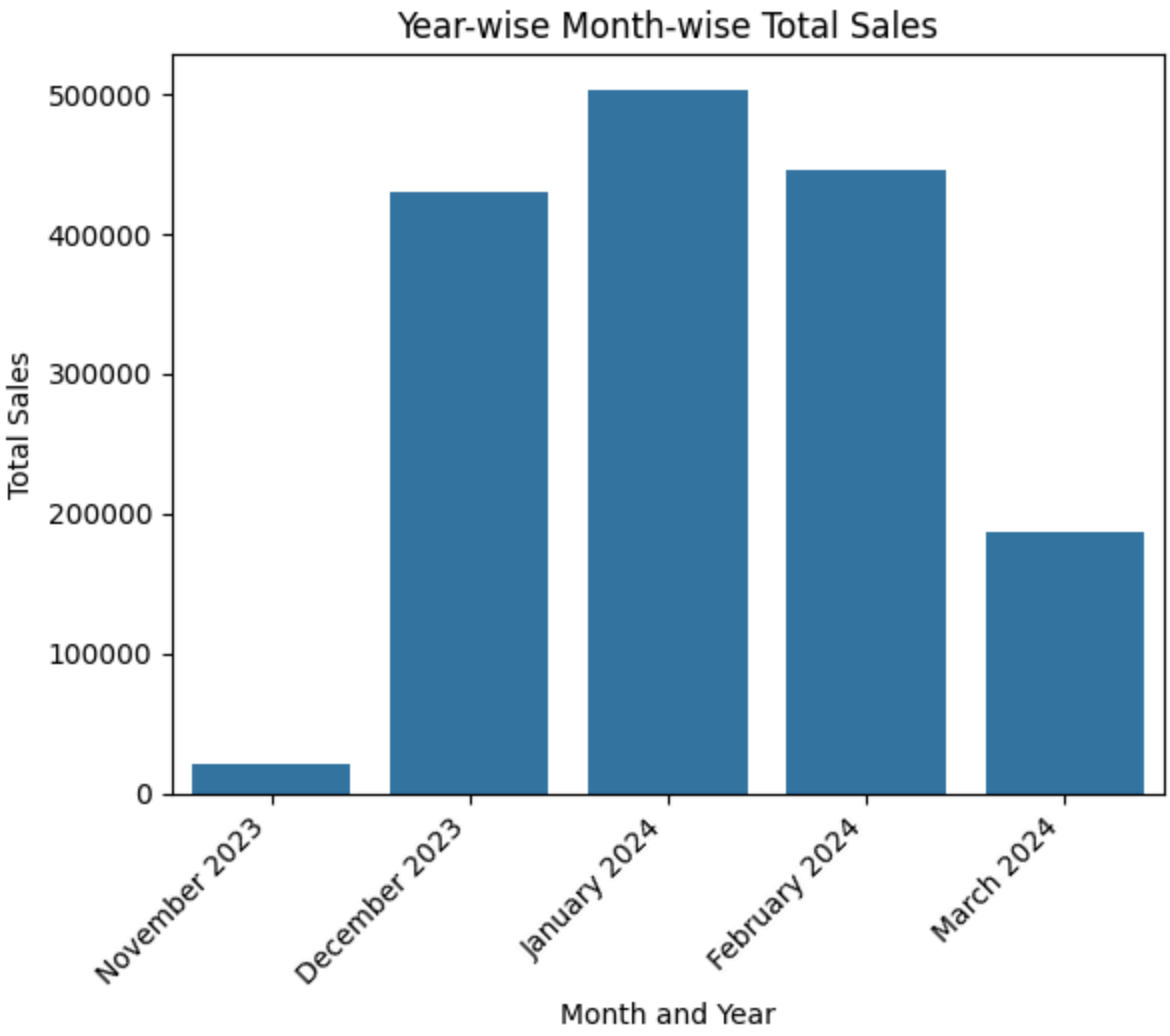
- Determine the most preferred payment modes.



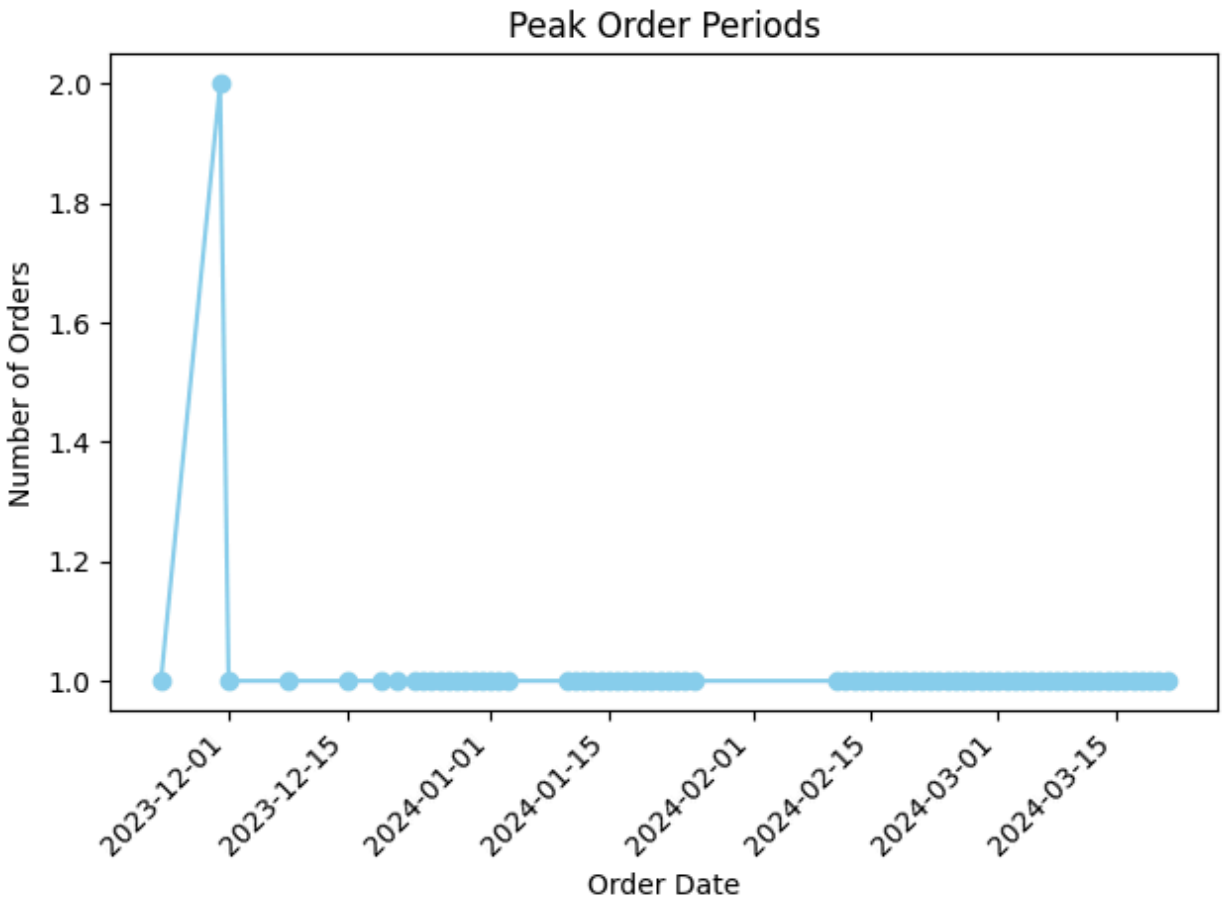
- Time-based Analysis:
    - Month wise total sales.



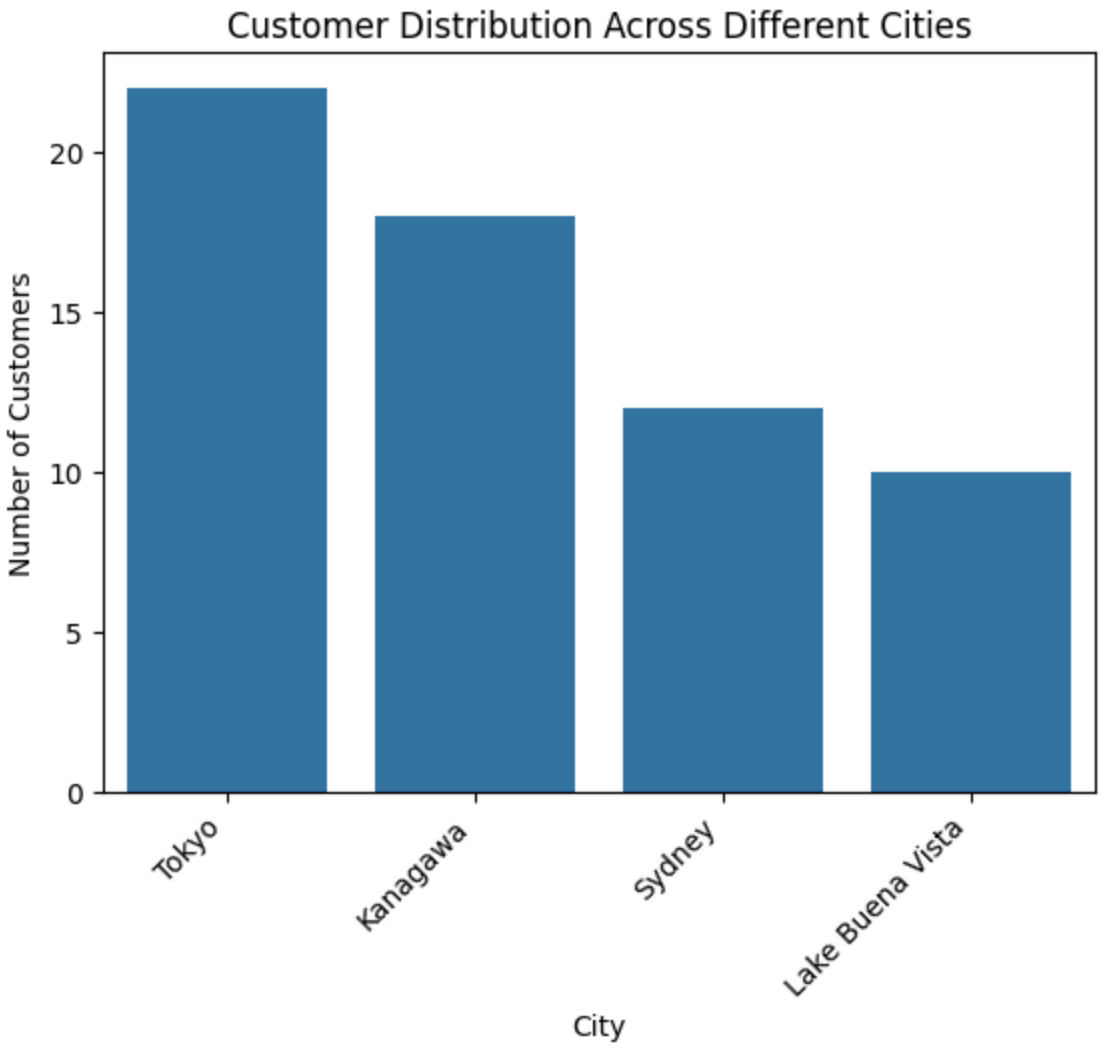
- Month and year wise total sales



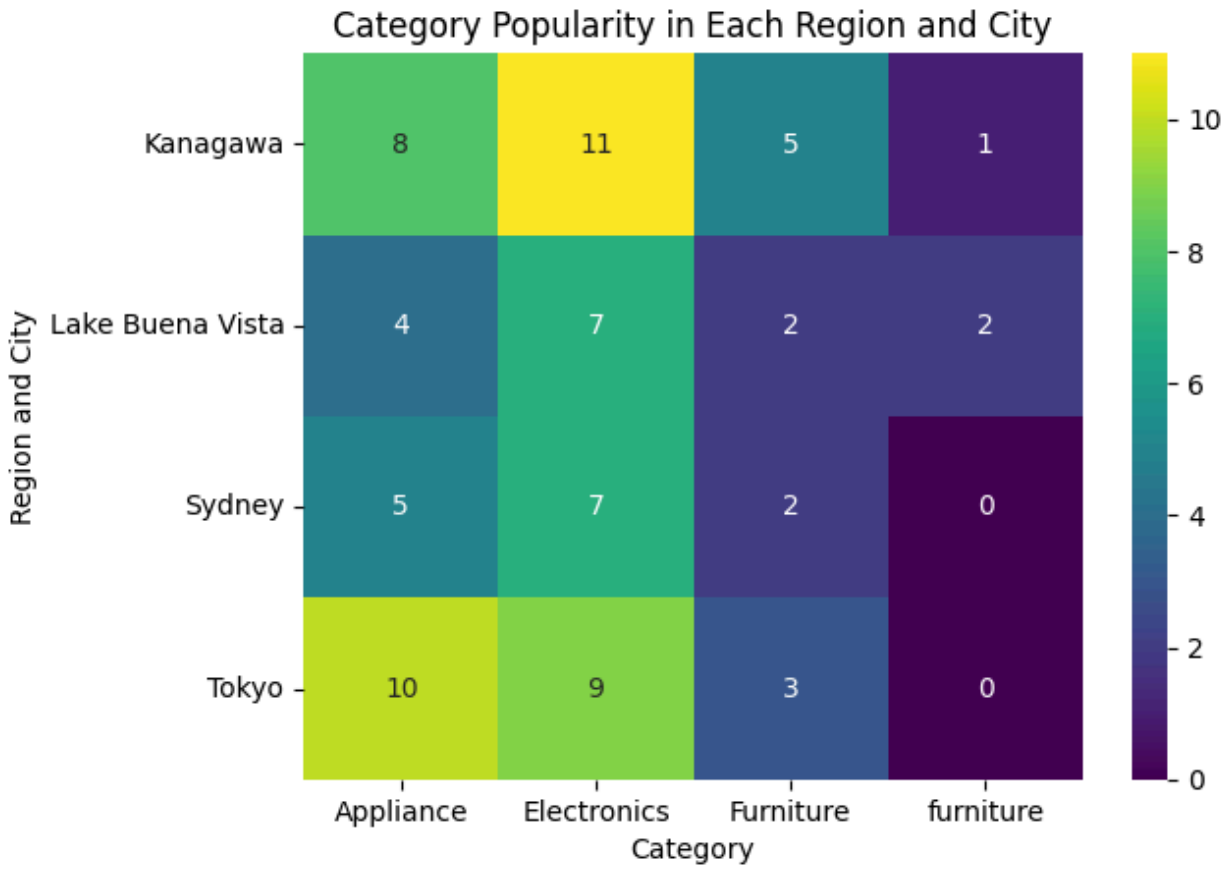
- Identify peak order date.



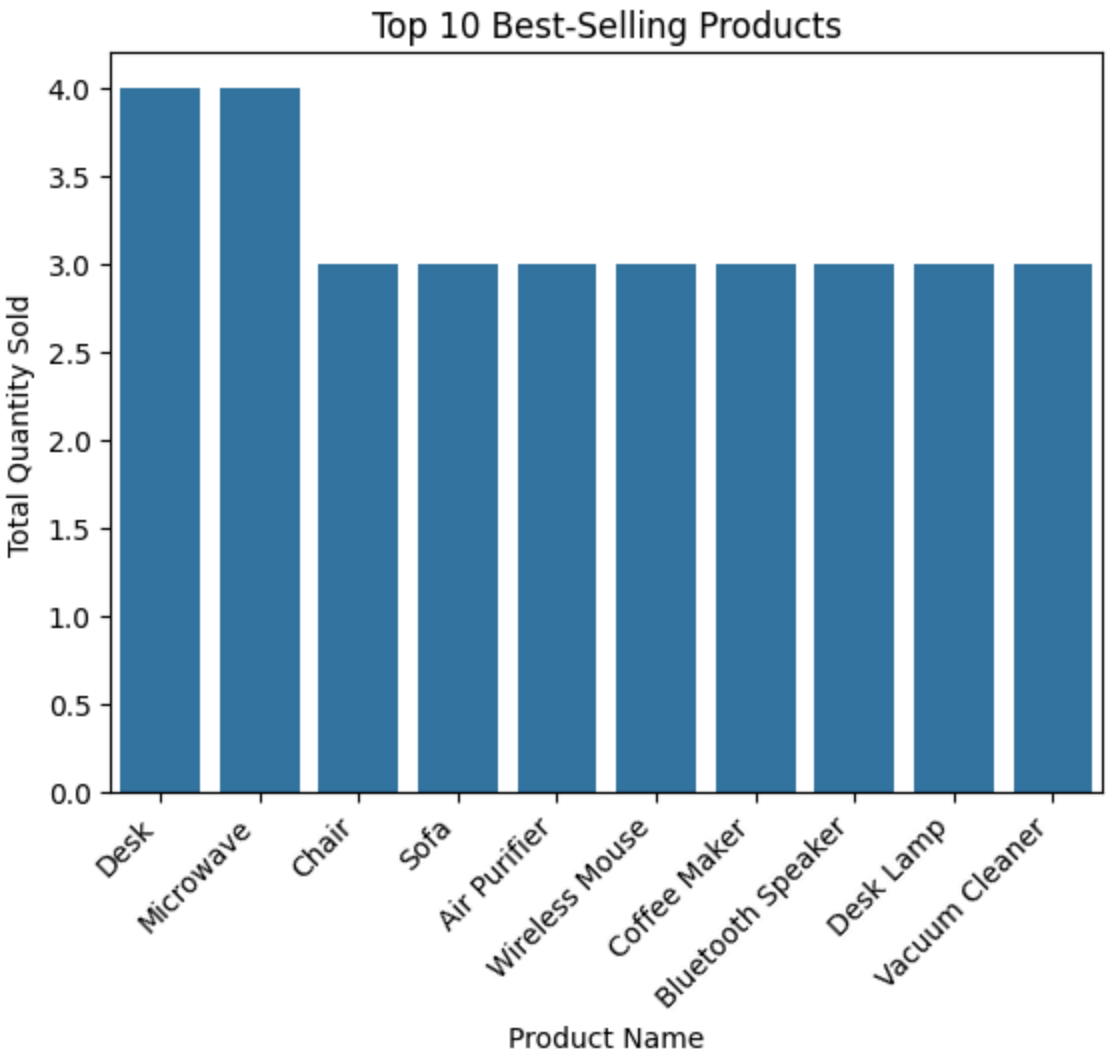
- Geographical Analysis:
    - Explore the distribution of customers across different cities.



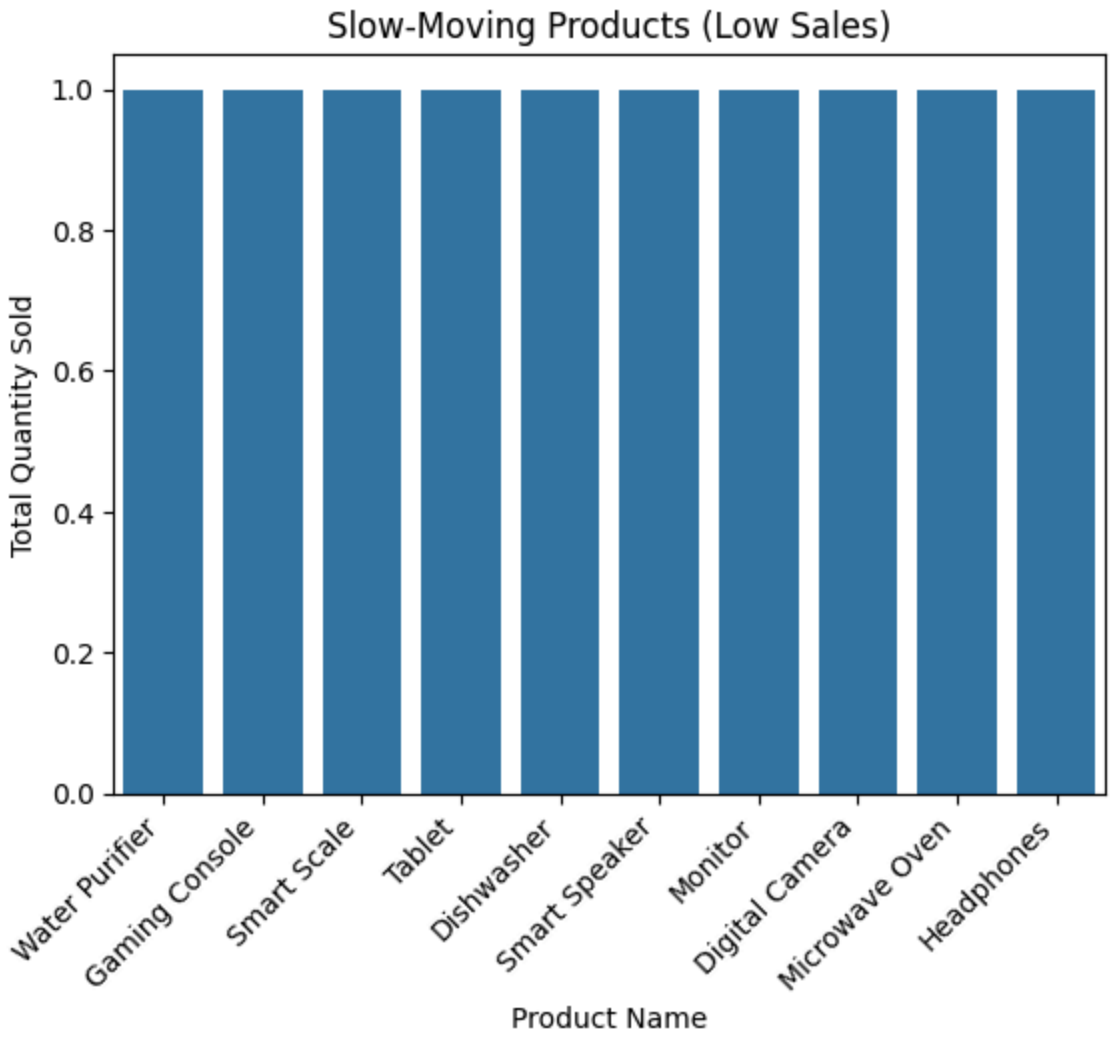
- Analyze whether certain products or categories are more popular in a specific city.



- Product Performance:
    - Identify the best-selling products.



- Identify top 10 slow-moving products based on low sales.



- Payment Analysis:
    - Display successful and pending payments order counts.

Conclusion:

Summarize the key insights derived from the analysis. Provide recommendations for business improvement based on the identified patterns and trends.

Future Work:

Suggest potential areas for further analysis or data collection that could enhance the depth of insights gained.