GUDIBANDI HARSHA VARDHAN REDDY

TASK 01:
Exploratory Data Analysis (EDA) and Business Insights
1. Perform EDA on the provided dataset.
2. Derive at least 5 business insights from the EDA.
Write these insights in short point-wise sentences (maximum 100 words per insight).

What is EDA?
Exploratory data analysis (EDA) is the process of analyzing a dataset to identify its key features and discover patterns, trends, and relationships. This is the first step in data analysis before moving on to more complex tasks like machine learning or modeling. Our goal is to better understand your data.

Step-by-Step for EDA

1. Loading and Understanding the Data
We have three datasets:
Customers.csv: Contains customer details (like their ID, name, region, and sign-up date).
Products.csv: Contains product details (like product ID, name, category, and price).
Transactions.csv: Contains transaction details (like transaction ID, product ID, customer ID, quantity, total value, and price).

In [21]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load datasets
customers = pd.read_csv(r'C:\Users\USER\Downloads\Customers.csv')
products = pd.read_csv('Products.csv')
transactions = pd.read_csv('Transactions.csv')



# Preview datasets
print(customers.head())
print(products.head())
print(transactions.head())
```

```
  CustomerID        CustomerName         Region  SignupDate
0      C0001    Lawrence Carroll  South America  2022-07-10
1      C0002      Elizabeth Lutz           Asia  2022-02-13
2      C0003      Michael Rivera  South America  2024-03-07
3      C0004  Kathleen Rodriguez  South America  2022-10-09
4      C0005         Laura Weber           Asia  2022-08-15
  ProductID              ProductName     Category   Price
0      P001     ActiveWear Biography        Books  169.30
1      P002    ActiveWear Smartwatch  Electronics  346.30
2      P003  ComfortLiving Biography        Books   44.12
3      P004            BookWorld Rug   Home Decor   95.69
4      P005          TechPro T-Shirt     Clothing  429.31
  TransactionID CustomerID ProductID      TransactionDate  Quantity  \
0        T00001      C0199      P067  2024-08-25 12:38:23         1
1        T00112      C0146      P067  2024-05-27 22:23:54         1
2        T00166      C0127      P067  2024-04-25 07:38:55         1
3        T00272      C0087      P067  2024-03-26 22:55:37         2
4        T00363      C0070      P067  2024-03-21 15:10:10         3

   TotalValue   Price
0      300.68  300.68
1      300.68  300.68
2      300.68  300.68
3      601.36  300.68
4      902.04  300.68
```

In [23]:
```
# Check for missing values
print(customers.isnull().sum())
print(products.isnull().sum())
print(transactions.isnull().sum())
```

```
CustomerID      0
CustomerName    0
Region          0
SignupDate      0
dtype: int64
ProductID      0
ProductName    0
Category       0
Price          0
dtype: int64
TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price              0
dtype: int64
```

2. Handling Missing Data and Cleaning

Data can sometimes be incomplete or have errors. So, we check for missing values or duplicates and clean them.

```
# Checking for missing values
print(customers.isnull().sum())
print(products.isnull().sum())
print(transactions.isnull().sum())

# Dropping rows with missing values (optional)
customers.dropna(inplace=True)
products.dropna(inplace=True)
transactions.dropna(inplace=True)

# Checking for duplicate entries
print(customers.duplicated().sum())
print(products.duplicated().sum())
print(transactions.duplicated().sum())

# Dropping duplicates (if any)
customers.drop_duplicates(inplace=True)
products.drop_duplicates(inplace=True)
transactions.drop_duplicates(inplace=True)

```

```
CustomerID      0
CustomerName    0
Region          0
SignupDate      0
dtype: int64
ProductID      0
ProductName    0
Category       0
Price          0
dtype: int64
TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price              0
dtype: int64
0
0
0
```

3. Descriptive Statistics

Next, we calculate some basic statistics like the mean, median, standard deviation, etc., for numerical columns to get a sense of the data distribution.

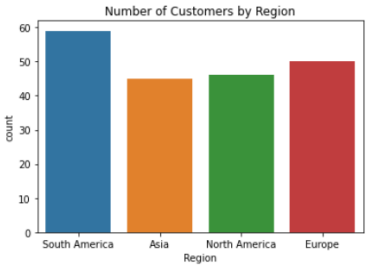
In [41]:
```
# Basic statistics
print("\nTransactions Summary Statistics:")
print(transactions.describe())
```

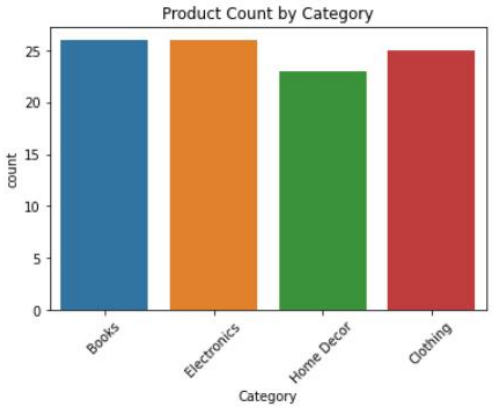
```
Transactions Summary Statistics:
          Quantity   TotalValue       Price
count  1000.000000  1000.000000  1000.00000
mean      2.537000   689.995560   272.55407
std       1.117981   493.144478   140.73639
min       1.000000    16.080000    16.08000
25%       2.000000   295.295000   147.95000
50%       3.000000   588.880000   299.93000
75%       4.000000  1011.660000   404.40000
max       4.000000  1991.040000   497.76000
```

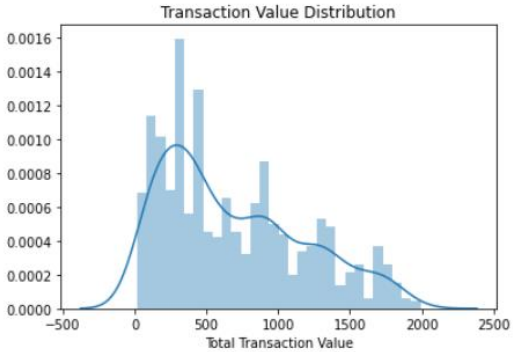
4. Visualizing the Data

Now, we create different visualizations to better understand the data:

In [42]:
```
# Categorical Features
sns.countplot(data=customers, x='Region')
plt.title('Number of Customers by Region')
plt.show()

sns.countplot(data=products, x='Category')
plt.title('Product Count by Category')
plt.xticks(rotation=45)
plt.show()

# Numerical Features
sns.distplot(transactions['TotalValue'], kde=True, bins=30, hist=True)
plt.title('Transaction Value Distribution')
plt.xlabel('Total Transaction Value')
plt.show()

# Boxplot for TotalValue to detect outliers
sns.boxplot(y=transactions['TotalValue'])
plt.title('Transaction Value - Outliers')
plt.show()
```



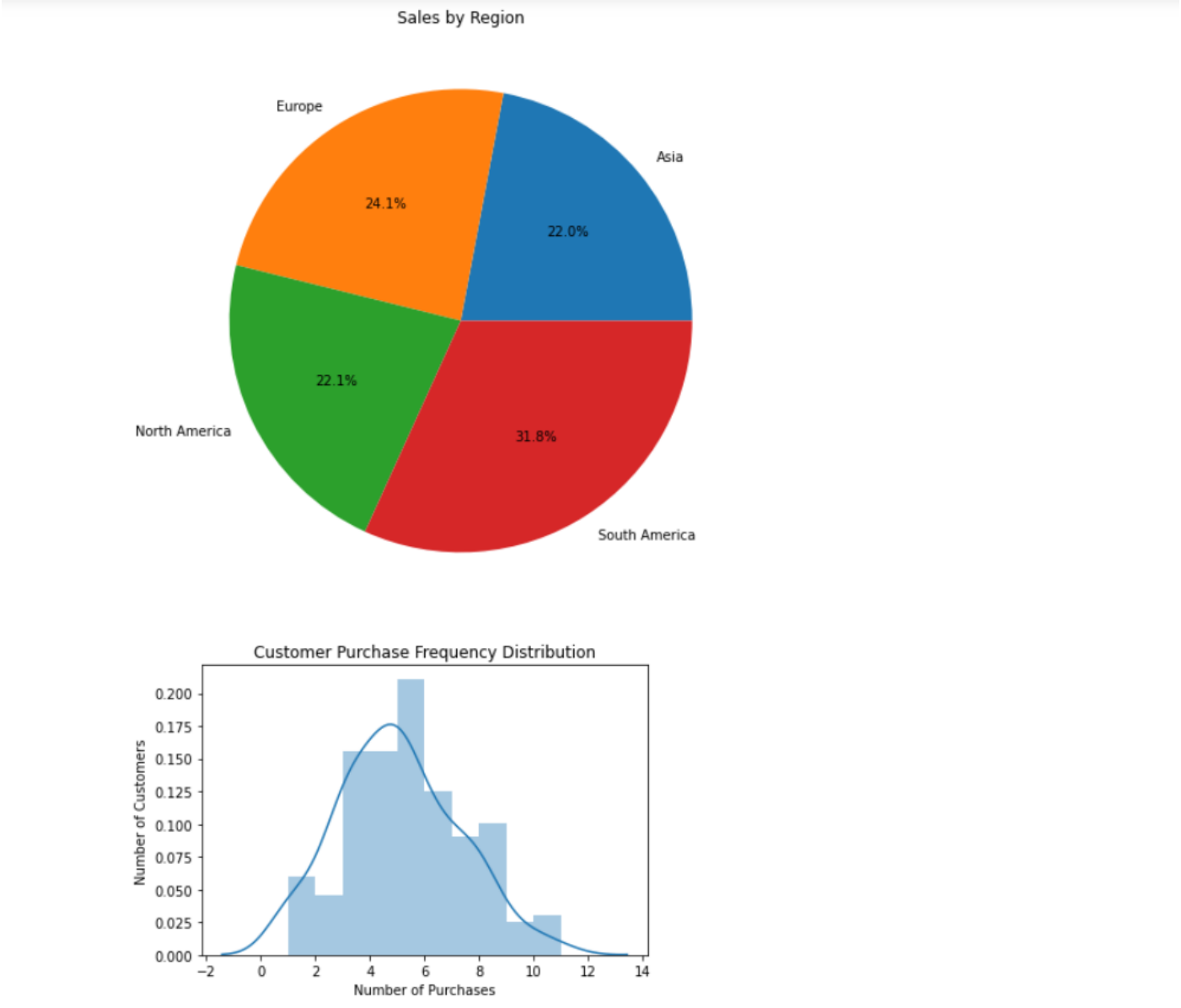




5. Correlation Analysis

To check how the variables relate to each other, we calculate correlations (for example, between Quantity, Price, and TotalValue).

```
# Merge datasets
transactions_products = pd.merge(transactions, products, on='ProductID')
merged_data = pd.merge(transactions_products, customers, on='CustomerID')

# Top products by sales
top_products = merged_data.groupby('ProductName')['TotalValue'].sum().sort_values(ascending=False).head(10)
top_products.plot(kind='bar', title='Top 10 Products by Sales', figsize=(10, 6), color='skyblue')
plt.ylabel('Total Sales Value')
plt.xlabel('Product Name')
plt.xticks(rotation=45)
plt.show()

# Regional sales
regional_sales = merged_data.groupby('Region')['TotalValue'].sum()
regional_sales.plot(kind='pie', autopct='%1.1f%%', figsize=(8, 8), title='Sales by Region')
plt.ylabel('')
plt.show()

# Customer purchase frequency
customer_freq = merged_data['CustomerID'].value_counts()
sns.distplot(customer_freq, bins=10, kde=True, hist=True)
plt.title('Customer Purchase Frequency Distribution')
plt.xlabel('Number of Purchases')
plt.ylabel('Number of Customers')
plt.show()

```



6. Business Insights from EDA

1. Customer Distribution by Region

The largest number of customers are in South America (304 customers), followed by North America (244 custo mers), Europe (234 customers), and Asia (218 customers). This suggests that focusing on South America will le ad to greater sales potential, while expanding into North America and Europe can help expand into more marke ts. Top-selling products

2. Top Product Categories by Sales

Books are among the topselling products (270), followed by Electronics (254), Home Decor (248), and Clothing (228). The company needs to consider increasing its volume and sales in this category. Sales and Transaction Value

3. Sales and Transaction Value

The average transaction value is approximately 689.99, with a standard deviation of 493.14, indicating a large v ariability in transaction outcomes. The packaging is large. Identifying and targeting highvalue customers can be a strategy to increase overall sales. Buyer frequency

4. Customer Purchase Frequency

Customers purchased an average of approximately 2.54 items per transaction. Promotions designed to get a bu siness to sell their products will increase sales. Product Price and Total Transaction Value

5. Product Pricing and Total Sales Value

The average product price is $272.55, and the total value of each transaction is approximately $689.99. Offerin g discounts or bundles at these prices can attract more customers and increase sales.

In summary, the business can optimize its sales and marketing strategies by focusing on customer regions, product categories, and transaction behaviors. By leveraging the insights from EDA, the company can maximize its revenue potential, enhance customer satisfaction, and improve overall business performance.