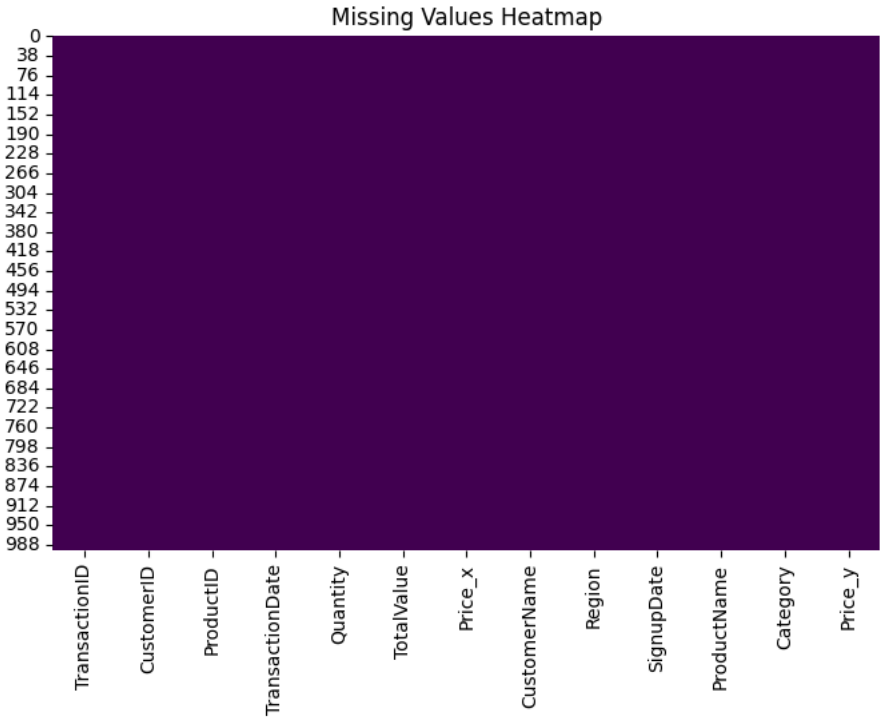
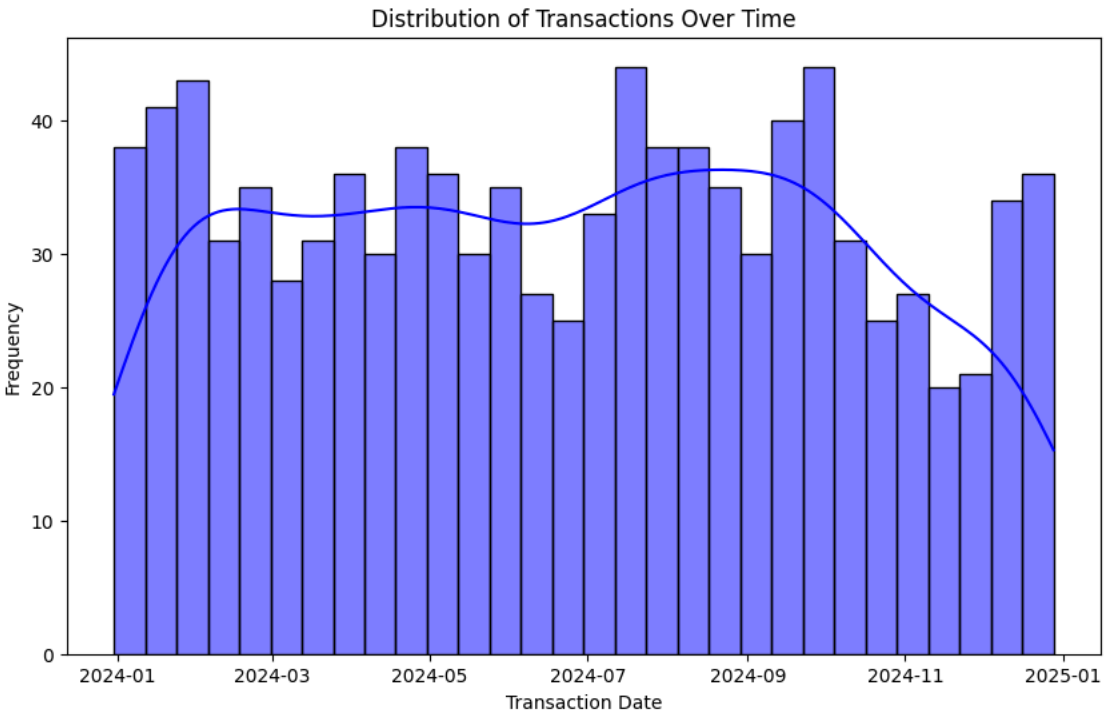
```
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')

# File paths in Google Drive
customers_path = '/content/drive/MyDrive/DataScience Intern/Customers.csv'
products_path = '/content/drive/MyDrive/DataScience Intern/Products.csv'
transactions_path = '/content/drive/MyDrive/DataScience Intern/Transactions.csv'

# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load datasets
customers = pd.read_csv(customers_path)
products = pd.read_csv(products_path)
transactions = pd.read_csv(transactions_path)

# Convert date columns to datetime
customers['SignupDate'] = pd.to_datetime(customers['SignupDate'])
transactions['TransactionDate'] = pd.to_datetime(transactions['TransactionDate'])

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

# 1. Missing values analysis
missing_values = merged_data.isnull().sum()
plt.figure(figsize=(8, 5))
sns.heatmap(merged_data.isnull(), cbar=False, cmap="viridis")
plt.title("Missing Values Heatmap")
plt.show()

# 2. Distribution of Transactions by Date
plt.figure(figsize=(10, 6))
sns.histplot(merged_data['TransactionDate'], bins=30, kde=True, color="blue")
plt.title("Distribution of Transactions Over Time")
plt.xlabel("Transaction Date")
plt.ylabel("Frequency")
plt.show()

# 3. Top Regions by Customer Count
top_regions = customers['Region'].value_counts()
plt.figure(figsize=(8, 5))
sns.barplot(x=top_regions.index, y=top_regions.values, palette="coolwarm")
plt.title("Customer Distribution by Region")
plt.xlabel("Region")
plt.ylabel("Number of Customers")
plt.xticks(rotation=45)
plt.show()

# 4. Top Products by Sales
top_products = merged_data.groupby('ProductName')['TotalValue'].sum().sort_values(ascending=False).head(10)
plt.figure(figsize=(10, 6))
sns.barplot(x=top_products.values, y=top_products.index, palette="viridis")
plt.title("Top 10 Products by Sales")
plt.xlabel("Total Sales (USD)")
plt.ylabel("Product Name")
plt.show()

# 5. Correlation Heatmap
numeric_columns = merged_data.select_dtypes(include=['float64', 'int64']).columns
plt.figure(figsize=(8, 5))
sns.heatmap(merged_data[numeric_columns].corr(), annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()

# 6. Signup Trends
customers['SignupMonth'] = customers['SignupDate'].dt.to_period('M')
signup_trends = customers['SignupMonth'].value_counts().sort_index()
plt.figure(figsize=(10, 6))
signup_trends.plot(kind='line', marker='o', color="purple")
plt.title("Signup Trends Over Time")
plt.xlabel("Month")
plt.ylabel("Number of Signups")
```

```
    plt.grid(True)
    plt.show()

    # 7. Product Category Distribution
    if 'Category' in products.columns:  # Check if the Category column exists
        category_distribution = products['Category'].value_counts()
        plt.figure(figsize=(8, 5))
        category_distribution.plot(kind='pie', autopct='%1.1f%%', startangle=90, cmap="Set2")
        plt.title("Product Category Distribution")
        plt.ylabel("")
        plt.show()
    # Task 2: Deriving Business Insights from EDA

    # 1. Insight: Customer Distribution by Region
    top_regions = customers['Region'].value_counts()
    print("Insight 1: Customer Distribution by Region")
    print(f"Most customers are from the region: {top_regions.idxmax()} with {top_regions.max()} customers.")
    print(f"Least customers are from the region: {top_regions.idxmin()} with {top_regions.min()} customers.")

    # 2. Insight: Top Products by Sales
    top_products = merged_data.groupby('ProductName')['TotalValue'].sum().sort_values(ascending=False).head(5)
    print("\nInsight 2: Top 5 Products by Sales")
    print("These products generate the highest revenue:")
    print(top_products)

    # 3. Insight: Signup Trends
    signup_trends = customers['SignupDate'].dt.to_period('M').value_counts().sort_index()
    most_signups_month = signup_trends.idxmax()
    print(f"\nInsight 3: Customer Signup Trends")
    print(f"The month with the highest signups is {most_signups_month} with {signup_trends.max()} signups.")
    print("Focusing marketing efforts in similar periods can maximize user acquisition.")

    # 4. Insight: Transaction Trends Over Time
    transaction_trends = merged_data['TransactionDate'].dt.to_period('M').value_counts().sort_index()
    most_transactions_month = transaction_trends.idxmax()
    print(f"\nInsight 4: Transaction Trends")
    print(f"The highest transaction volume occurred in {most_transactions_month} with {transaction_trends.max()} transactions.")
    print("Analyzing factors behind this spike can help replicate success in other months.")

    # 5. Insight: High Revenue Product Categories (if available)
    if 'Category' in products.columns:
        category_revenue = merged_data.groupby('Category')['TotalValue'].sum().sort_values(ascending=False)
        print("\nInsight 5: High Revenue Product Categories")
        print("Categories generating the most revenue:")
        print(category_revenue.head(3))
    else:
        print("\nInsight 5: Product Category data not available in the dataset.")
```

```
Mounted at /content/drive
```






```
<ipython-input-1-15edfb027c58>:46: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and s

  sns.barplot(x=top_regions.index, y=top_regions.values, palette="coolwarm")
```