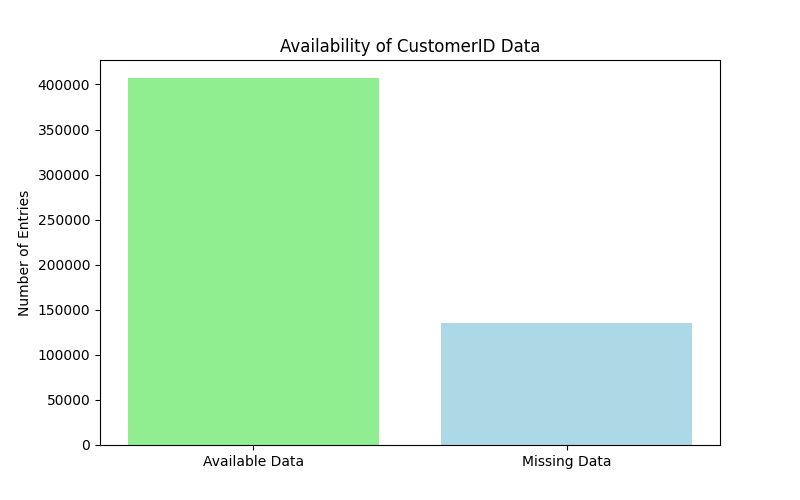
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
# Load the dataset  
  
df = pd.read\_excel(r"C:\Users\prabh\OneDrive\Desktop\Python Project\Online Retail.xlsx")  
  
# Summary of the DataFrame including column names, data types, and non-null counts.  
print(df.info())  
print("Descriptive Statistics: \n", df.describe())  
print("First 5 rows: \n", df.head())  
print("Last 5 rows: \n", df.tail())  
print("Shape of the DataFrame: \n", df.shape)  
  
# Check for null values per attribute  
print("Null values per attribute: \n", df.isnull().sum())  
  
# Remove duplicate rows  
df\_cleaned = df.drop\_duplicates()  
  
# Drop columns that contain all missing values  
df\_cleaned.dropna(axis=1, how='all', inplace=True)  
  
# Fill missing values in 'CustomerID' with "Unknown"  
df\_cleaned['CustomerID'] = df\_cleaned['CustomerID'].fillna("Unknown")  
print(df\_cleaned.info())  
  
# 1. Analyze the completeness of CustomerID data  
plt.figure(figsize=(8, 5))  
missing\_customer = df["CustomerID"].isnull().sum()  
available\_customer = df["CustomerID"].notnull().sum()  
plt.bar(["Available Data", "Missing Data"], [available\_customer, missing\_customer], color=["lightgreen", "lightblue"])  
plt.title("Availability of CustomerID Data")  
plt.ylabel("Number of Entries")  
plt.show()  
  
# 2. Distribution of Quantity  
plt.figure(figsize=(8, 5))  
sns.histplot(df\_cleaned["Quantity"], bins=30, kde=True, color="green")  
plt.title("Distribution of Quantity")  
plt.xlabel("Quantity")  
plt.ylabel("Frequency")  
plt.show()  
  
# 3. Proportion of Orders by Country  
country\_orders = df\_cleaned["Country"].value\_counts()  
plt.figure(figsize=(10, 5))  
sns.barplot(x=country\_orders.index, y=country\_orders.values, palette="Set2")  
plt.title("Proportion of Orders by Country")  
plt.xlabel("Country")  
plt.ylabel("Number of Orders")  
plt.xticks(rotation=90)  
plt.show()  
  
# 4. Average UnitPrice by Country  
avg\_price\_country = df\_cleaned.groupby("Country")["UnitPrice"].mean().sort\_values(ascending=False)  
plt.figure(figsize=(12, 6))  
sns.barplot(x=avg\_price\_country.index, y=avg\_price\_country.values, palette="viridis")  
plt.title("Average UnitPrice by Country")  
plt.xlabel("Country")  
plt.ylabel("Average Unit Price")  
plt.xticks(rotation=90)  
plt.show()  
  
# 5. Quantity vs Unit Price  
plt.figure(figsize=(8, 5))  
sns.scatterplot(x=df\_cleaned["Quantity"], y=df\_cleaned["UnitPrice"], color='blue', alpha=0.5)  
plt.title("Quantity vs Unit Price")  
plt.xlabel("Quantity")  
plt.ylabel("Unit Price")  
plt.show()  
  
# 6. Top 10 Most Purchased Products by Quantity  
top\_products = df\_cleaned.groupby("Description")["Quantity"].sum().nlargest(10)  
top\_products.plot(kind="bar", figsize=(12, 6), color="coral")  
plt.title("Top 10 Most Purchased Products by Quantity")  
plt.xlabel("Product Description")  
plt.ylabel("Total Quantity Purchased")  
plt.xticks(rotation=45)  
plt.show()  
  
# 7. Count of Orders by CustomerID  
customer\_orders = df\_cleaned["CustomerID"].value\_counts().head(10)  
plt.figure(figsize=(12, 6))  
sns.barplot(x=customer\_orders.index, y=customer\_orders.values, palette="Blues")  
plt.title("Top 10 Customers by Number of Orders")  
plt.xlabel("CustomerID")  
plt.ylabel("Number of Orders")  
plt.xticks(rotation=90)  
plt.show()  
  
# 8. Sales by Month  
df\_cleaned['InvoiceDate'] = pd.to\_datetime(df\_cleaned['InvoiceDate'], format='%d-%m-%Y %H:%M')  
df\_cleaned['Month'] = df\_cleaned['InvoiceDate'].dt.month  
monthly\_sales = df\_cleaned.groupby("Month")["UnitPrice"].sum()  
plt.figure(figsize=(8, 5))  
monthly\_sales.plot(kind="bar", color="teal")  
plt.title("Sales by Month")  
plt.xlabel("Month")  
plt.ylabel("Total Sales")  
plt.xticks(rotation=0)  
plt.show()  
  
# 9. Heatmap of Correlation Between Numerical Features  
numerical\_df = df\_cleaned.select\_dtypes(include=["number"])  
correlation\_matrix = numerical\_df.corr()  
plt.figure(figsize=(12, 6))  
sns.heatmap(correlation\_matrix, annot=True, fmt=".2f", cmap="coolwarm", linewidths=0.5)  
plt.title("Correlation Heatmap of Numerical Features")  
plt.show()  
  
# 10. Detect Outliers Using Z-score  
from scipy.stats import zscore  
columns = [  
 'Quantity',  
 'UnitPrice'  
]  
df\_zscore = df\_cleaned[columns].apply(zscore)  
threshold = 3  
outliers = df\_cleaned[(np.abs(df\_zscore) > threshold).any(axis=1)]  
print("Outliers detected using Z-score:")  
print(outliers)

A green bar graph with numbers

AI-generated content may be incorrect.