```python
# -*- coding: utf-8 -*-
"""Aditya Chikte - Unified Mentor Data Analytics Internship [Analyzing Amazon Sales Data].ipynb

Automatically generated by Colab.

Original file is located at
    https://colab.research.google.com/drive/1uQQPiZwyvAHYPymT8hhLLt6QuCs7-IDS

# **Importing Necessary Libraries**
"""

# We'll start by importing the necessary libraries for data analysis and visualization.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

"""# **Loading the Dataset**"""

# Install the gdown library
!pip install gdown

# Import the gdown library
import gdown

# Google Drive link to the dataset
url = "https://drive.google.com/uc?id=10sofXyF6NjwN6ngLyFfiPI-CUDpeqaN_"

# Download the dataset
gdown.download(url, 'amazon_sales.csv', quiet=False)

# Load dataset into a Pandas DataFrame
df = pd.read_csv('amazon_sales.csv')

"""# **Data Cleaning and Preprocessing**

**1. Viewing Dataset**
"""

# Display the first few rows of the dataset
print(df.head())

"""**2. Performing Data Preprocessing**"""

# Check for missing values
print(df.isnull().sum())

# Drop rows with missing values
df.dropna(inplace=True)

# Convert date columns to datetime format
df['Order Date'] = pd.to_datetime(df['Order Date'])
df['Ship Date'] = pd.to_datetime(df['Ship Date'])

# Extract month and year from date columns
df['Order Month'] = df['Order Date'].dt.month
df['Order Year'] = df['Order Date'].dt.year

"""**3. Displaying the modified DataFrame**"""

# Display the first few rows of the modified DataFrame
print("First few rows after preprocessing:")
print(df.head())

"""# **Exploratory Data Analysis (EDA)**

**1. Sales Trends Analysis**
"""

# Explore sales trends month-wise, year-wise, and yearly-month-wise
plt.figure(figsize=(14, 6))

# Monthly Sales Trends
plt.subplot(1, 3, 1)
sns.lineplot(data=df, x='Order Month', y='Total Revenue', estimator=sum)
plt.title('Monthly Sales Trends')
plt.xlabel('Month')
plt.ylabel('Total Revenue')

# Yearly Sales Trends
plt.subplot(1, 3, 2)
sns.lineplot(data=df, x='Order Year', y='Total Revenue', estimator=sum)
plt.title('Yearly Sales Trends')
plt.xlabel('Year')
plt.ylabel('Total Revenue')

# Yearly-Monthly Sales Trends
plt.subplot(1, 3, 3)
sns.lineplot(data=df, x='Order Month', y='Total Revenue', hue='Order Year', estimator=sum)
plt.title('Yearly-Monthly Sales Trends')
plt.xlabel('Month')
plt.ylabel('Total Revenue')
plt.legend(title='Year')
plt.tight_layout()
plt.show()

"""**2. Relationships Visualization**"""

# Visualize relationships between attributes using scatter plots
plt.figure(figsize=(12, 6))

# Scatter plot: Units Sold vs Total Revenue
plt.subplot(1, 2, 1)
sns.scatterplot(data=df, x='Units Sold', y='Total Revenue')
plt.title('Units Sold vs Total Revenue')

# Scatter plot: Unit Price vs Total Revenue
plt.subplot(1, 2, 2)
sns.scatterplot(data=df, x='Unit Price', y='Total Revenue')
plt.title('Unit Price vs Total Revenue')
plt.tight_layout()
plt.show()

# Calculate correlation matrix
correlation_matrix = df[['Units Sold', 'Unit Price', 'Unit Cost', 'Total Revenue', 'Total Cost', 'Total Profit']].corr()

# Visualize correlation matrix using heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()

"""**3. Pair Plot Analysis**"""

# Visualize relationships between numerical attributes using pair plot
sns.pairplot(df[['Units Sold', 'Unit Price', 'Unit Cost', 'Total Revenue', 'Total Cost', 'Total Profit']])
plt.title('Pair Plot of Numerical Attributes')
plt.show()

"""**4. Categorical Variables Exploration**"""

# Explore categorical variables using count plots or bar plots
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.countplot(data=df, x='Sales Channel')
plt.title('Sales Channel Distribution')

plt.subplot(1, 2, 2)
sns.countplot(data=df, x='Order Priority')
plt.title('Order Priority Distribution')
plt.tight_layout()
plt.show()

"""# **Data Analysis**"""

# Calculate average sales per month/year
avg_monthly_sales = df.groupby('Order Month')['Total Revenue'].mean()
avg_yearly_sales = df.groupby('Order Year')['Total Revenue'].mean()

# Product categories with highest sales
top_categories = df.groupby('Item Type')['Total Revenue'].sum().nlargest(5)

# Print average monthly sales
print("Average Monthly Sales:")
print(avg_monthly_sales)

# Print average yearly sales
print("\nAverage Yearly Sales:")
print(avg_yearly_sales)

# Print top product categories by sales
print("\nTop Product Categories by Sales:")
print(top_categories)

"""# **Data Visualization**

**1. Visualizing Top Product Categories by Sales**

This section of the code generates a bar chart to visualize the top product categories by sales revenue. The bar chart provides insights into which product categories contribute the mo
"""

# Visualize top product categories by sales
plt.figure(figsize=(10, 6))
top_categories.plot(kind='bar')
plt.title('Top Product Categories by Sales')
plt.xlabel('Product Category')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

"""**2. Visualizing Distribution of Unit Price and Unit Cost**

This section of the code generates histograms to visualize the distribution of unit prices and unit costs. Histograms provide insights into the frequency distribution of values within
"""

# Visualize distribution of unit price and unit cost
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.histplot(df['Unit Price'], bins=20, kde=True)
plt.title('Distribution of Unit Price')

plt.subplot(1, 2, 2)
sns.histplot(df['Unit Cost'], bins=20, kde=True)
plt.title('Distribution of Unit Cost')
plt.tight_layout()
plt.show()

"""# **Advanced Data Visualization**"""

# Import Plotly
import plotly.graph_objects as go

# Calculate total revenue for each product category
category_revenue = df.groupby('Item Type')['Total Revenue'].sum().reset_index()

# Create an interactive bar chart
fig = go.Figure(data=[go.Bar(
    x=category_revenue['Item Type'],
    y=category_revenue['Total Revenue'],
    hovertext=category_revenue['Total Revenue'],  # Display revenue on hover
    marker_color='skyblue'  # Change color of bars
)])

# Customize the layout
fig.update_layout(
    title='Total Revenue by Product Category',
    xaxis=dict(title='Product Category'),
    yaxis=dict(title='Total Revenue'),
    plot_bgcolor='rgba(0,0,0,0)'  # Set background color to transparent
)

# Show the interactive plot
fig.show()

"""# **Conclusion and Insights**"""

print("Conclusion and Insights:")
print("- The analysis reveals various trends and relationships in the sales data.")
print("- Monthly sales exhibit seasonal variations, with higher sales during certain months.")
print("- Yearly sales have been increasing steadily over the years, indicating overall business growth.")
print("- There is a strong positive correlation between units sold and total revenue, indicating that higher sales volumes contribute to higher revenue.")
print("- The correlation matrix highlights strong correlations between units sold, total revenue, and total profit, suggesting that these variables are closely related.")
print("- Recommendations for improving sales performance:")
print("- Explore targeted marketing strategies to capitalize on peak sales months and drive sales during slower months.")
print("- Analyze pricing strategies to optimize profit margins while maintaining competitive pricing.")
print("- Consider expanding product offerings or entering new markets to further drive revenue growth.")
```