import pandas as pd

# Read the CSV file into a DataFrame

df = pd.read\_csv('sales\_data.csv')

# Check for missing data

missing\_data = df.isna().sum()

print("Missing Data:")

print(missing\_data)

# Decide on how to handle missing or incorrect data

# For simplicity, let's drop rows with missing data

df.dropna(inplace=True)

# Data normalization

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

df[['Unit', 'Sales']] = scaler.fit\_transform(df[['Unit', 'Sales']])

# Convert 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Extract hour from the 'Time' column

df['Hour'] = df['Time'].apply(lambda x: x.split('-')[0])

# Convert 'Hour' column to categorical type

df['Hour'] = pd.Categorical(df['Hour'], categories=['Morning', 'Afternoon', 'Evening', 'Night'], ordered=True)

# Descriptive statistical analysis

descriptive\_stats = df[['Unit', 'Sales']].describe()

# Determine highest and lowest sales groups and states

highest\_sales\_group = df.groupby('Group')['Sales'].sum().idxmax()

lowest\_sales\_group = df.groupby('Group')['Sales'].sum().idxmin()

highest\_sales\_state = df.groupby('State')['Sales'].sum().idxmax()

lowest\_sales\_state = df.groupby('State')['Sales'].sum().idxmin()

# Generate weekly, monthly, and quarterly reports

weekly\_report = df.groupby(df['Date'].dt.week)['Sales'].sum()

monthly\_report = df.groupby(df['Date'].dt.month)['Sales'].sum()

quarterly\_report = df.groupby(df['Date'].dt.quarter)['Sales'].sum()

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Read the CSV file into a DataFrame

df = pd.read\_csv('sales\_data.csv')

# Convert 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Time-of-the-day analysis

sns.boxplot(x='Time', y='Sales', data=df)

plt.title('Time-of-the-Day Analysis')

plt.xlabel('Time of the Day')

plt.ylabel('Sales')

plt.show()

# State-wise sales analysis for different groups

plt.figure(figsize=(10, 6))

sns.barplot(x='State', y='Sales', hue='Group', data=df)

plt.title('State-wise Sales Analysis for Different Groups')

plt.xlabel('State')

plt.ylabel('Sales')

plt.xticks(rotation=45)

plt.legend(title='Group')

plt.show()

# Group-wise sales analysis across different states

plt.figure(figsize=(10, 6))

sns.barplot(x='Group', y='Sales', hue='State', data=df)

plt.title('Group-wise Sales Analysis Across Different States')

plt.xlabel('Group')

plt.ylabel('Sales')

plt.legend(title='State')

plt.show()

import pandas as pd

# Read the CSV file into a DataFrame

df = pd.read\_csv('sales\_data.csv')

# Check for missing data

missing\_data = df.isna().sum()

print("Missing Data:")

print(missing\_data)

# Decide on how to handle missing or incorrect data

# For simplicity, let's drop rows with missing data

df.dropna(inplace=True)

# Data normalization

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

df[['Unit', 'Sales']] = scaler.fit\_transform(df[['Unit', 'Sales']])

# Convert 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Extract hour from the 'Time' column

df['Hour'] = df['Time'].apply(lambda x: x.split('-')[0])

# Convert 'Hour' column to categorical type

df['Hour'] = pd.Categorical(df['Hour'], categories=['Morning', 'Afternoon', 'Evening', 'Night'], ordered=True)

# Descriptive statistical analysis

descriptive\_stats = df[['Unit', 'Sales']].describe()

# Determine highest and lowest sales groups and states

highest\_sales\_group = df.groupby('Group')['Sales'].sum().idxmax()

lowest\_sales\_group = df.groupby('Group')['Sales'].sum().idxmin()

highest\_sales\_state = df.groupby('State')['Sales'].sum().idxmax()

lowest\_sales\_state = df.groupby('State')['Sales'].sum().idxmin()

# Generate weekly, monthly, and quarterly reports

weekly\_report = df.groupby(df['Date'].dt.week)['Sales'].sum()

monthly\_report = df.groupby(df['Date'].dt.month)['Sales'].sum()

quarterly\_report = df.groupby(df['Date'].dt.quarter)['Sales'].sum()

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Read the CSV file into a DataFrame

df = pd.read\_csv('sales\_data.csv')

# Convert 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Time-of-the-day analysis

plt.figure(figsize=(10, 6))

sns.boxplot(x='Time', y='Sales', data=df, palette='pastel') # Use a pastel color palette

plt.title('Sales Distribution Across Different Times of the Day', fontsize=16)

plt.xlabel('Time of the Day', fontsize=14)

plt.ylabel('Sales', fontsize=14)

plt.xticks(fontsize=12)

plt.yticks(fontsize=12)

plt.grid(True, linestyle='--', alpha=0.7) # Add grid lines for better readability

plt.tight\_layout() # Adjust layout to prevent overlapping

plt.show()

# State-wise sales analysis for different groups

plt.figure(figsize=(12, 8))

sns.barplot(x='State', y='Sales', hue='Group', data=df, palette='Set2') # Use a Set2 color palette

plt.title('State-wise Sales Analysis for Different Groups', fontsize=16)

plt.xlabel('State', fontsize=14)

plt.ylabel('Sales', fontsize=14)

plt.xticks(rotation=45, fontsize=12)

plt.yticks(fontsize=12)

plt.legend(title='Group', fontsize=12)

plt.tight\_layout()

plt.show()

# Group-wise sales analysis across different states

plt.figure(figsize=(12, 8))

sns.barplot(x='Group', y='Sales', hue='State', data=df, palette='colorblind') # Use a colorblind-friendly palette

plt.title('Group-wise Sales Analysis Across Different States', fontsize=16)

plt.xlabel('Group', fontsize=14)

plt.ylabel('Sales', fontsize=14)

plt.xticks(fontsize=12)

plt.yticks(fontsize=12)

plt.legend(title='State', fontsize=12, bbox\_to\_anchor=(1, 1)) # Adjust legend position

plt.tight\_layout()

plt.show()

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Read the CSV file into a DataFrame

df = pd.read\_csv('sales\_data.csv')

# Convert 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Filter data for the fourth quarter (October to December)

fourth\_quarter\_data = df[(df['Date'].dt.quarter == 4) & (df['Date'].dt.year == 2023)]

# Analyze sales across Australia for the fourth quarter

sales\_across\_australia = fourth\_quarter\_data.groupby('Date')['Sales'].sum()

# Plotting sales trend across Australia for the fourth quarter

plt.figure(figsize=(10, 6))

sns.lineplot(x=sales\_across\_australia.index, y=sales\_across\_australia.values)

plt.title('Sales Trend Across Australia (Fourth Quarter)')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.xticks(rotation=45)

plt.grid(True)

plt.show()

# Analyze sales state by state for the fourth quarter

statewise\_sales\_q4 = fourth\_quarter\_data.groupby('State')['Sales'].sum().sort\_values(ascending=False)

# Plotting state-wise sales for the fourth quarter

plt.figure(figsize=(10, 6))

sns.barplot(x=statewise\_sales\_q4.index, y=statewise\_sales\_q4.values, palette='viridis')

plt.title('State-wise Sales Analysis for the Fourth Quarter')

plt.xlabel('State')

plt.ylabel('Sales')

plt.xticks(rotation=45)

plt.show()

# Determine top performing and bottom performing states

top\_performing\_state\_q4 = statewise\_sales\_q4.idxmax()

bottom\_performing\_state\_q4 = statewise\_sales\_q4.idxmin()

print("Top performing state in the fourth quarter:", top\_performing\_state\_q4)

print("Bottom performing state in the fourth quarter:", bottom\_performing\_state\_q4)