**EX:No.1 221501040**

**25/01/25**

**Program to implement for time series data cleaning, loading and handling times series data and pre-processing techniques.**

**Aim:** to implement for time series data cleaning, loading and handling times series data and pre-processing techniques.

**Code:**

#Import necessary libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

# Load dataset

file\_path = "og.csv"

df = pd.read\_csv(file\_path)

# Convert date columns to datetime format with correct format

df['Order Date'] = pd.to\_datetime(df['Order Date'], format='%m/%d/%Y', errors='coerce')

df['Ship Date'] = pd.to\_datetime(df['Ship Date'], format='%m/%d/%Y', errors='coerce')

# Drop rows with invalid or missing dates

df.dropna(subset=['Order Date'], inplace=True)

# Handle missing values (fill Postal Code with most frequent value)

df['Postal Code'].fillna(df['Postal Code'].mode()[0], inplace=True)

df.fillna(method='ffill', inplace=True)

# Drop unnecessary columns

df.drop(columns=['Row ID', 'Customer Name'], inplace=True)

# Set Order Date as index and sort

df.set\_index('Order Date', inplace=True)

df.sort\_index(inplace=True)

# Handle duplicates

df.drop\_duplicates(inplace=True)

# Resample sales data to monthly frequency

df\_monthly\_sales = df['Sales'].resample('M').sum()

# Normalize Sales using Min-Max Scaling

scaler = MinMaxScaler()

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

# Save cleaned data

df.to\_csv('/mnt/data/cleaned\_sales\_data.csv')

print("Data loading, cleaning, and preprocessing completed.")

# Visualizing sales trends

plt.figure(figsize=(12, 6))

sns.lineplot(x=df\_monthly\_sales.index, y=df\_monthly\_sales.values, marker='o', linestyle='-', color='b', label='Monthly Sales')

plt.title('Monthly Sales Trend', fontsize=16)

plt.xlabel('Date', fontsize=12)

plt.ylabel('Total Sales', fontsize=12)

plt.grid(True, linestyle='--', alpha=0.6)

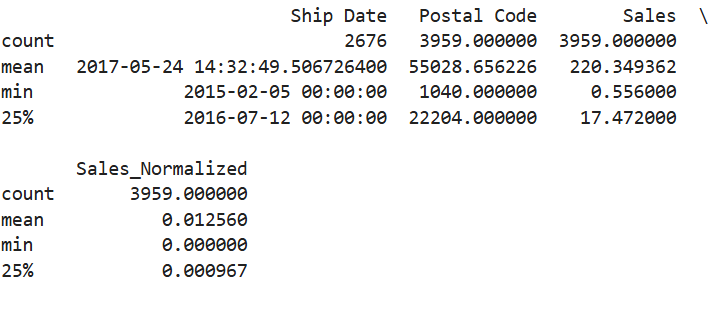
plt.xticks(rotation=45)

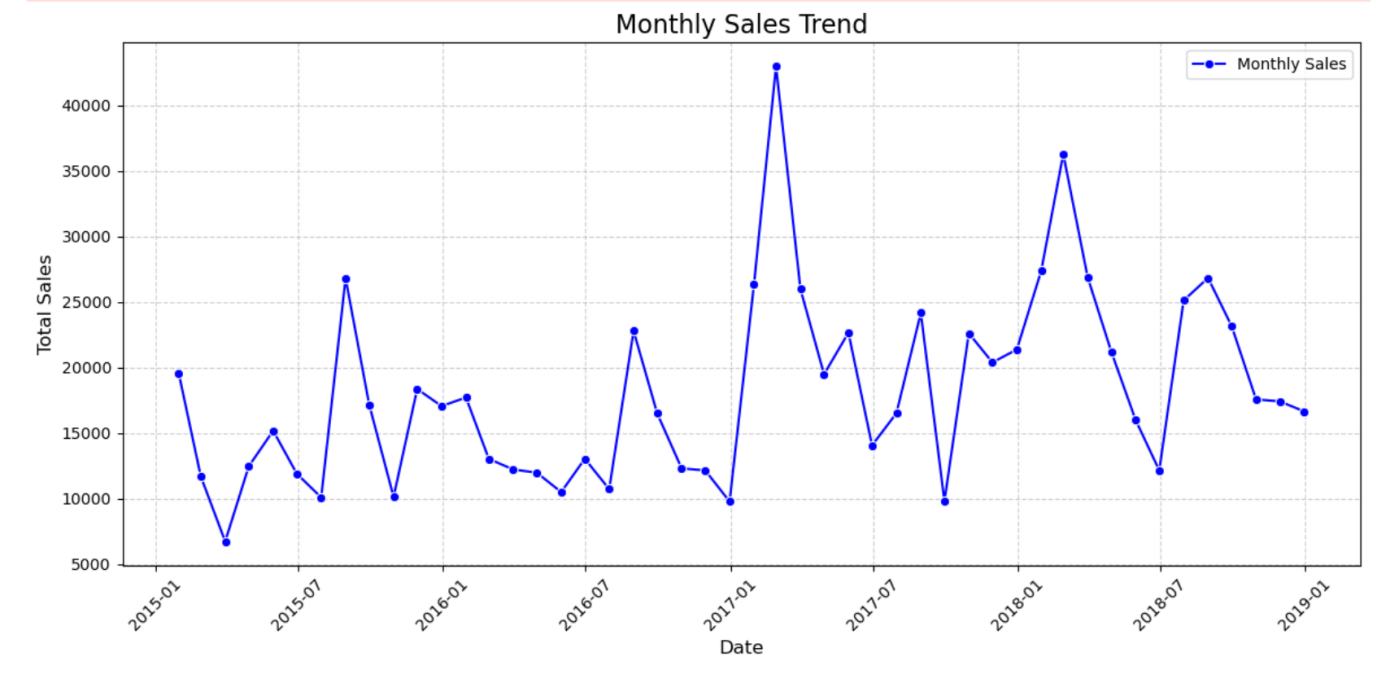
plt.legend()

plt.tight\_layout()

plt.show()

**Output:**

Summary of cleaned data  




**Result:** Thus program to implement for time series data cleaning, loading and handling times series data and pre-processing techniques.