EXERCISE:2 Upload and Analyze the data set given in csv format and perform data preprocessing and visualization.

Aim:

To upload the given CSV dataset (*sales-data.csv*), perform data preprocessing, and visualize key insights using data analysis techniques.

Procedure:

1. **Upload the Dataset:**
   Import the *sales-data.csv* file into the Python environment using pandas (pd.read_csv()).
2. **Inspect the Data:**
   Display the first few rows (head()), check the number of rows and columns, and identify data types.
3. **Handle Missing Values:**
   Detect and handle missing or null values by either filling them with mean/median or removing incomplete rows.
4. **Data Cleaning and Transformation:**
   Remove duplicates, convert data types if necessary (e.g., dates to datetime), and standardize categorical values.
5. **Data Visualization:**
   Use **Matplotlib** or **Seaborn** to visualize trends such as total sales by region, product category, or month.

In [14]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
file_path='C:\\Users\kaviy\\Downloads\\sales_data.csv'
df = pd.read_csv(file_path)
print(df.head())
print(df.isnull().sum())
df['Sales'].fillna(df['Sales'].mean(), inplace=True)
df.dropna(subset=['Product', 'Quantity', 'Region'], inplace=True)
print(df.describe())
product_summary = df.groupby('Product').agg({
'Sales': 'sum',
'Quantity': 'sum'
}).reset_index()
print(product_summary)
```

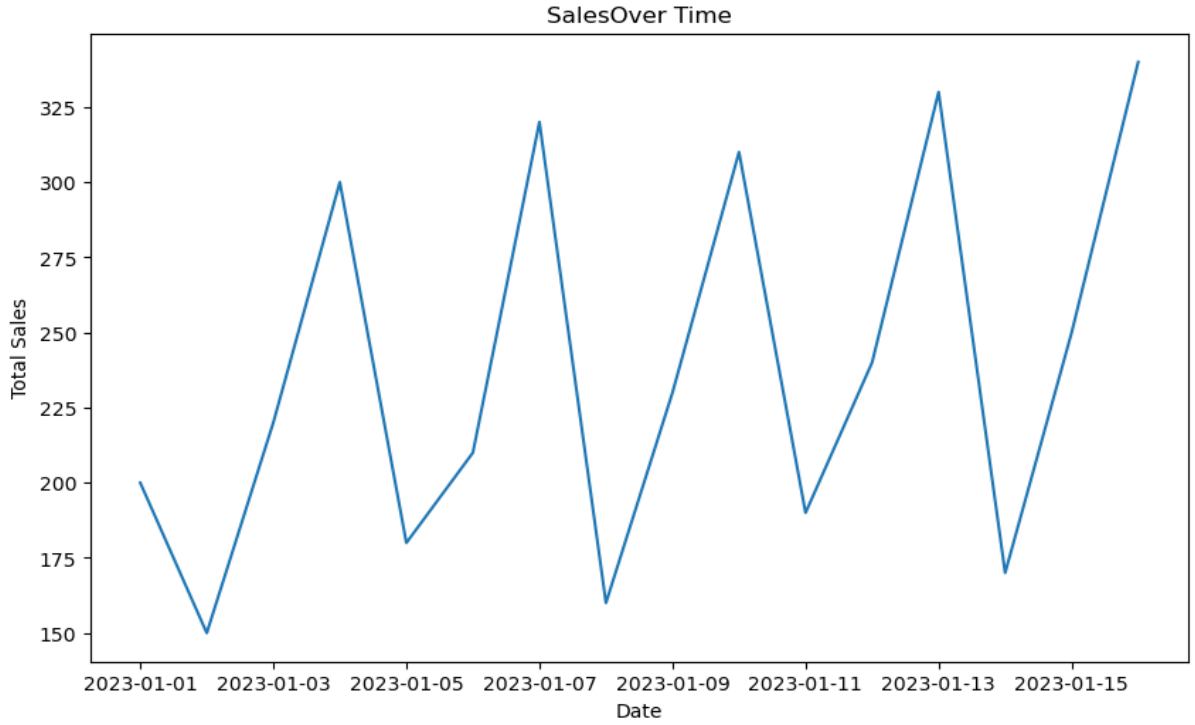
EXERCISE:2 Upload and Analyze the data set given in csv format and perform data preprocessing and visualization.

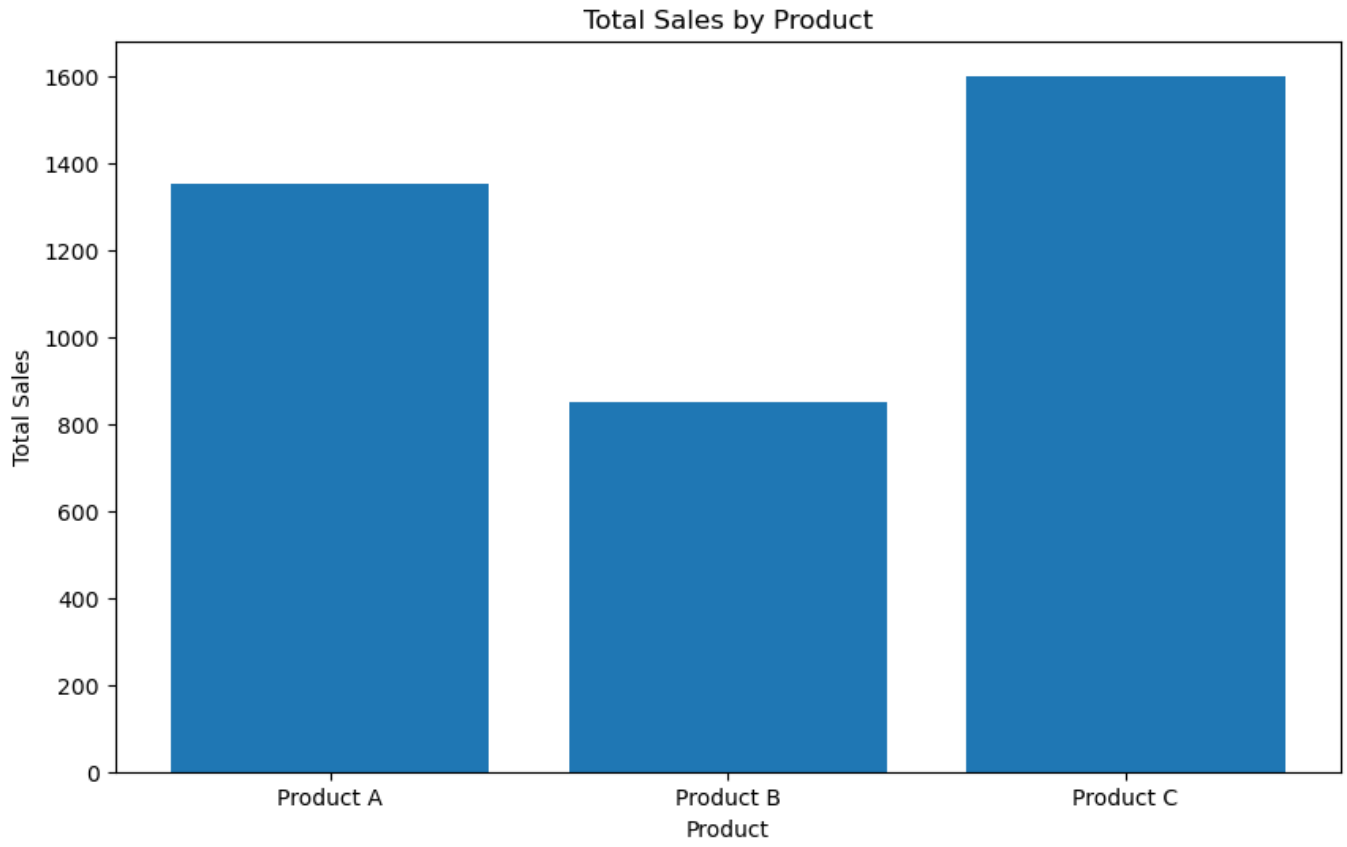
```
plt.figure(figsize=(10, 6))
plt.bar(product_summary['Product'], product_summary['Sales'])
plt.xlabel('Product')
plt.ylabel('Total Sales')
plt.title('Total Sales by Product')
plt.show()
df['Date'] = df['Date'] = pd.to_datetime(df['Date'], dayfirst=True,
errors='coerce')
sales_over_time = df.groupby('Date').agg({'Sales': 'sum'}).reset_index()
plt.figure(figsize=(10, 6))
plt.plot(sales_over_time['Date'],sales_over_time['Sales'])
plt.xlabel('Date')
plt.ylabel('Total Sales')
plt.title('SalesOver Time')
plt.show()
pivot_table = df.pivot_table(values='Sales', index='Region',
columns='Product',
aggfunc=np.sum, fill_value=0)
print(pivot_table)
correlation_matrix = df.corr()
print(correlation_matrix)
import seaborn as sns
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

```
         Date    Product  Sales  Quantity Region
0  01-01-2023  Product A    200         4  North
1  02-01-2023  Product B    150         3  South
2  03-01-2023  Product A    220         5  North
3  04-01-2023  Product C    300         6   East
4  05-01-2023  Product B    180         4   West
Date        0
Product     0
Sales       0
Quantity    0
Region      0
dtype: int64
            Sales   Quantity
count   16.000000  16.000000
mean   237.500000   5.375000
std     64.031242   1.746425
min    150.000000   3.000000
25%    187.500000   4.000000
50%    225.000000   5.500000
75%    302.500000   7.000000
max    340.000000   8.000000
     Product  Sales  Quantity
0  Product A   1350        33
```

EXERCISE:2 Upload and Analyze the data set given in csv format and perform data preprocessing and visualization.

```
1   Product B     850         17
2   Product C    1600         36
```

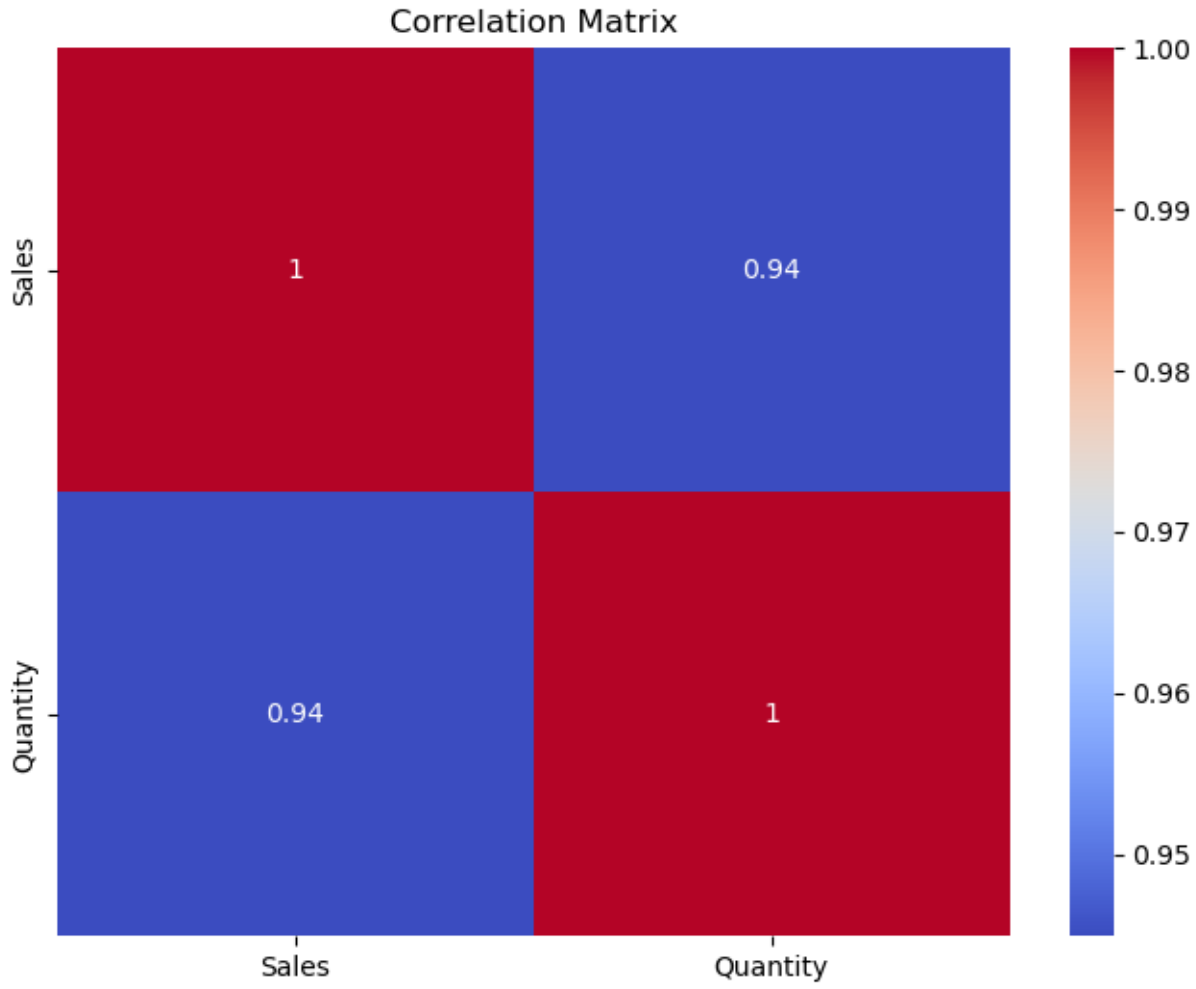





```
Product   Product A   Product B   Product C
Region
```

EXERCISE:2 Upload and Analyze the data set given in csv format and perform data preprocessing and visualization.

```
East              0          0       1600
North          1350          0          0
South             0        480          0
West              0        370          0
              Sales  Quantity
Sales      1.000000  0.944922
Quantity   0.944922  1.000000
C:\Users\kaviy\AppData\Local\Temp\ipykernel_13256\136924332.py:39:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only
valid columns or specify the value of numeric_only to silence this warning.
  correlation_matrix = df.corr()
```



RESULT:

The dataset was successfully loaded, cleaned by handling missing values, and formatted correctly for analysis.Summary statistics and visualizations revealed that certain products and regions generated higher total sales.Time series and correlation analysis showed sales trends over time and strong relationships between sales and quantity.