In [3]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [4]:
```
df=pd.read_csv(r'F:\DS assignment\DS-Assignment Dataset and instructions\P2- Of
```

In [5]:
```
df.head()
```

Out[5]:

	OrderDate	Region	Rep	Item	Units	Unit Price
0	04-Jul-14	East	Richard	Pen Set	62	4.99
1	12-Jul-14	East	Nick	Binder	29	1.99
2	21-Jul-14	Central	Morgan	Pen Set	55	12.49
3	29-Jul-14	East	Susan	Binder	81	19.99
4	07-Aug-14	Central	Matthew	Pen Set	42	23.95

In [7]:
```
df.tail()
```

Out[7]:

	OrderDate	Region	Rep	Item	Units	Unit Price
38	22-May-15	West	Thomas	Pencil	32	1.99
39	31-May-15	Central	Bill	Binder	80	8.99
40	08-Jun-15	East	Richard	Binder	60	8.99
41	17-Jun-15	Central	Matthew	Desk	5	125.00
42	25-Jun-15	Central	Morgan	Pencil	90	4.99

In [18]:
```
df
```

Out[18]:

	OrderDate	Region	Rep	Item	Units	Unit Price
0	04-Jul-14	East	Richard	Pen Set	62	4.99
1	12-Jul-14	East	Nick	Binder	29	1.99
2	21-Jul-14	Central	Morgan	Pen Set	55	12.49
3	29-Jul-14	East	Susan	Binder	81	19.99
4	07-Aug-14	Central	Matthew	Pen Set	42	23.95
5	15-Aug-14	East	Richard	Pencil	35	4.99
6	24-Aug-14	West	James	Desk	3	275.00
7	01-Sep-14	Central	Smith	Desk	2	125.00
8	10-Sep-14	Central	Bill	Pencil	7	1.29
9	18-Sep-14	East	Richard	Pen Set	16	15.99
10	27-Sep-14	West	James	Pen	76	1.99
11	05-Oct-14	Central	Morgan	Binder	28	8.99
12	14-Oct-14	West	Thomas	Binder	57	19.99
13	22-Oct-14	East	Richard	Pen	64	8.99
14	31-Oct-14	Central	Rachel	Pencil	14	1.29
15	08-Nov-14	East	Susan	Pen	15	19.99
16	17-Nov-14	Central	Alex	Binder	11	4.99
17	25-Nov-14	Central	Matthew	Pen Set	96	4.99
18	04-Dec-14	Central	Alex	Binder	94	19.99
19	12-Dec-14	Central	Smith	Pencil	67	1.29
20	21-Dec-14	Central	Rachel	Binder	28	4.99
21	29-Dec-14	East	Susan	Pen Set	74	15.99
22	06-Jan-15	East	Richard	Pencil	95	1.99
23	15-Jan-15	Central	Bill	Binder	46	8.99
24	23-Jan-15	Central	Matthew	Binder	50	19.99
25	01-Feb-15	Central	Smith	Binder	87	15.00
26	09-Feb-15	Central	Alex	Pencil	36	4.99
27	18-Feb-15	East	Richard	Binder	4	4.99
28	26-Feb-15	Central	Bill	Pen	27	19.99
29	07-Mar-15	West	James	Binder	7	19.99
30	15-Mar-15	West	James	Pencil	56	2.99
31	24-Mar-15	Central	Alex	Pen Set	50	4.99
32	01-Apr-15	East	Richard	Binder	60	4.99
33	10-Apr-15	Central	Rachel	Pencil	66	1.99
34	18-Apr-15	Central	Rachel	Pencil	75	1.99
35	27-Apr-15	East	Nick	Pen	96	4.99

	OrderDate	Region	Rep	Item	Units	Unit Price
36	05-May-15	Central	Alex	Pencil	90	4.99
37	14-May-15	Central	Bill	Pencil	53	1.29
38	22-May-15	West	Thomas	Pencil	32	1.99
39	31-May-15	Central	Bill	Binder	80	8.99
40	08-Jun-15	East	Richard	Binder	60	8.99
41	17-Jun-15	Central	Matthew	Desk	5	125.00
42	25-Jun-15	Central	Morgan	Pencil	90	4.99

In [8]:
```
df.shape
```

Out[8]:
```
(43, 6)
```

In [9]:
```
#columns present in dataset
df.columns
```

Out[9]:
```
Index(['OrderDate', 'Region', 'Rep', 'Item', 'Units', 'Unit Price'], dtype='o
bject')
```

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43 entries, 0 to 42
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   OrderDate   43 non-null     object
 1   Region      43 non-null     object
 2   Rep         43 non-null     object
 3   Item        43 non-null     object
 4   Units       43 non-null     int64
 5   Unit Price  43 non-null     float64
dtypes: float64(1), int64(1), object(4)
memory usage: 2.1+ KB
```

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
OrderDate     0
Region        0
Rep           0
Item          0
Units         0
Unit Price    0
dtype: int64
```

In [12]:
```
df.describe()
```

Out[12]:

	Units	Unit Price
count	43.000000	43.000000
mean	49.325581	20.308605
std	30.078248	47.345118
min	2.000000	1.290000
25%	27.500000	3.990000
50%	53.000000	4.990000
75%	74.500000	17.990000
max	96.000000	275.000000

1.Sales Analysis:

Q.What are the total sales for each product category?

In [14]:
```
df['OrderDate'].max()
```

Out[14]:
```
'31-Oct-14'
```

In [19]:
```
# Calculate the total sales for each row
df['Total Sales'] = df['Units'] * df['Unit Price']

# Group by 'Item' and sum the 'Total Sales'
total_sales_by_item = df.groupby('Item')['Total Sales'].sum().reset_index()

# Display the result
print(total_sales_by_item)
```

```
      Item  Total Sales
0   Binder      9577.65
1     Desk      1700.00
2      Pen      2045.22
3  Pen Set      4169.87
4   Pencil      2135.14
```

Q. Which product category has the highest sales?

In [20]:
```
max_sales_item = total_sales_by_item.loc[total_sales_by_item['Total Sales'].id

# Display the result
print(f"The product category with the highest sales is '{max_sales_item['Item']
```

```
The product category with the highest sales is 'Binder' with total sales of
$9577.65.
```

In [21]:
```
# Sort the items by 'Total Sales' in descending order
total_sales_by_item = total_sales_by_item.sort_values(by='Total Sales', ascendi

# Select the top 10 best-selling products
top_10_best_selling = total_sales_by_item.head(10)

# Display the result
print("Top 10 Best-Selling Products:")
print(top_10_best_selling)
```

```
Top 10 Best-Selling Products:
      Item  Total Sales
0   Binder      9577.65
3  Pen Set      4169.87
4   Pencil      2135.14
2      Pen      2045.22
1     Desk      1700.00
```

2. Customer Analysis:

Q. Who are the top 10 customers by sales

In [24]:
```
# Group by 'Rep' and sum the 'Total Sales'
total_sales_by_rep = df.groupby('Rep')['Total Sales'].sum().reset_index()

# Sort the reps by 'Total Sales' in descending order
total_sales_by_rep = total_sales_by_rep.sort_values(by='Total Sales', ascending

# Select the top 10 customers by sales
top_10_customers = total_sales_by_rep.head(10)

# Display the result
print("Top 10 Customers by Sales:")
print(top_10_customers)
```

```
Top 10 Customers by Sales:
        Rep  Total Sales
3   Matthew      3109.44
9     Susan      3102.30
0      Alex      2812.19
7   Richard      2363.04
1      Bill      1749.87
8     Smith      1641.43
4    Morgan      1387.77
2     James      1283.61
10   Thomas      1203.11
5      Nick       536.75
```

Q.What is the total number of unique customers?

In [25]:
```
# Calculate the total number of unique customers
total_unique_customers = df['Rep'].nunique()

total_unique_customers
```

Out[25]: 11

Q. Analyze customer purchase frequency.

In [26]:
```
# Calculate customer purchase frequency
customer_purchase_frequency = df['Rep'].value_counts()

customer_purchase_frequency
```

Out[26]:
```
Rep
Richard    8
Bill       5
Alex       5
Matthew    4
James      4
Rachel     4
Morgan     3
Susan      3
Smith      3
Nick       2
Thomas     2
Name: count, dtype: int64
```

3. Time Series Analysis:

Q.What are the monthly sales trends over the past year

In [28]:
```
# Ensure 'OrderDate' is in datetime format
df['OrderDate'] = pd.to_datetime(df['OrderDate'])

# Extract month and year from 'OrderDate'
df['Month'] = df['OrderDate'].dt.strftime('%Y-%m')

# Group by 'Month' and sum the 'Total Sales'
monthly_sales_trends = df.groupby('Month')['Total Sales'].sum()

monthly_sales_trends
```

```
C:\Users\NEHA\AppData\Local\Temp\ipykernel_18732\2656106448.py:2: UserWarnin
g: Could not infer format, so each element will be parsed individually, falli
ng back to `dateutil`. To ensure parsing is consistent and as-expected, pleas
e specify a format.
  df['OrderDate'] = pd.to_datetime(df['OrderDate'])
```

Out[28]:
```
Month
2014-07    2673.23
2014-08    2005.55
2014-09     666.11
2014-10    1984.57
2014-11     833.78
2014-12    3288.47
2015-01    1602.09
2015-02    2044.33
2015-03     556.87
2015-04    1059.03
2015-05    1300.35
2015-06    1613.50
Name: Total Sales, dtype: float64
```

Q.Identify any seasonal patterns in the sales data.

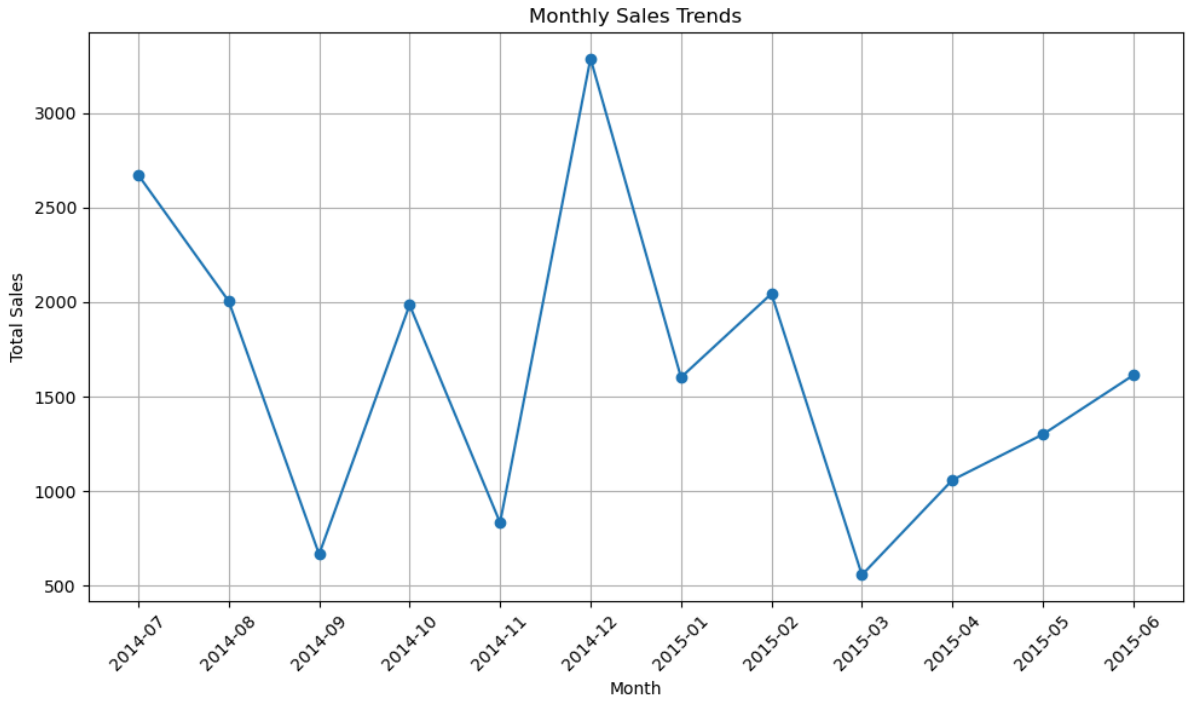
In [29]:
```
import matplotlib.pyplot as plt

# Ensure 'OrderDate' is in datetime format
df['OrderDate'] = pd.to_datetime(df['OrderDate'])

# Extract month and year from 'OrderDate'
df['Month'] = df['OrderDate'].dt.strftime('%Y-%m')

# Group by 'Month' and sum the 'Total Sales'
monthly_sales_trends = df.groupby('Month')['Total Sales'].sum()

# Plotting the monthly sales trends
plt.figure(figsize=(10, 6))
plt.plot(monthly_sales_trends.index, monthly_sales_trends.values, marker='o', l
plt.title('Monthly Sales Trends')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```



4. Geographical Analysis:

Q.Which regions generate the most sales?

In [31]:
```
# Group by 'Region' and sum the 'Total Sales'
sales_by_region = df.groupby('Region')['Total Sales'].sum()

sales_by_region
```

Out[31]:
```
Region
Central    11139.07
East        6002.09
West        2486.72
Name: Total Sales, dtype: float64
```

Q. What are the sales trends across different regions?

5. Profit Analysis:

Q.What is the total profit for each product category?

In [47]:
```
# Calculate Total Cost and Total Profit for each row
df['Total Cost'] = df['Units'] * df['Unit Price']
df['Total Profit'] = df['Total Sales'] - df['Total Cost']

# Group by 'Item' and sum the 'Total Profit'
total_profit_per_category = df.groupby('Item')['Total Profit'].sum().reset_inde

# Display the result
print("Total Profit for Each Product Category:")
print(total_profit_per_category)
```

```
Total Profit for Each Product Category:
      Item  Total Profit
0   Binder           0.0
1     Desk           0.0
2      Pen           0.0
3  Pen Set           0.0
4   Pencil           0.0
```

Q. Identify the top 10 most profitable products.

In [48]:
```
# Group by 'Item' and sum the 'Total Profit'
total_profit_per_product = df.groupby('Item')['Total Profit'].sum().reset_index

# Sort by 'Total Profit' in descending order
top_10_profitable_products = total_profit_per_product.sort_values(by='Total Pro

# Display the result
print("Top 10 Most Profitable Products:")
print(top_10_profitable_products)
```

```
Top 10 Most Profitable Products:
      Item  Total Profit
0   Binder           0.0
1     Desk           0.0
2      Pen           0.0
3  Pen Set           0.0
4   Pencil           0.0
```