Sales Analysis

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
data = pd.read_csv("Supplies Data.csv")
```

In [3]:
```
data.head(3)
```

Out[3]:

	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

In [4]:
```
data.isnull().sum()
```

Out[4]:
```
OrderDate     0
Region        0
Rep           0
Item          0
Units         0
Unit Price    0
dtype: int64
```

1. Sales Analysis:

In [5]:
```
item_category = (data['Item'].unique())
print(item_category)
```

```
['Pen Set' 'Binder' 'Pencil' 'Desk' 'Pen']
```

In [6]:
```
df = pd.DataFrame(data)
```

In [7]:
```
df['Total Sales'] = df['Units']* df['Unit Price']
```

What are the total sales for each product category?

In [8]:
```
total_sales_by_category = df.groupby('Item')['Total Sales'].sum().reset_i
total_sales_by_category['S.No.'] = total_sales_by_category.index + 1
total_sales_by_category = total_sales_by_category[['S.No.', 'Item', 'Tota
print(total_sales_by_category.to_string(index=False))
```

```
S.No.    Item  Total Sales
    1  Binder      9577.65
    2    Desk      1700.00
    3     Pen      2045.22
    4 Pen Set      4169.87
    5  Pencil      2135.14
```

In [9]:
```
df["Item"].value_counts()
```

Out[9]:
```
Binder     15
Pencil     13
Pen Set     7
Pen         5
Desk        3
Name: Item, dtype: int64
```

In [10]:
```
total_units_by_category = df.groupby('Item')['Units'].sum().reset_index()
# total_units_by_category
total_units_by_category['S.No.'] = total_units_by_category.index + 1
total_units_by_category = total_units_by_category[['S.No.', 'Item', 'Unit

print(total_units_by_category.to_string(index=False))
```

```
S.No.    Item  Units
    1  Binder    722
    2    Desk     10
    3     Pen    278
    4 Pen Set    395
    5  Pencil    716
```

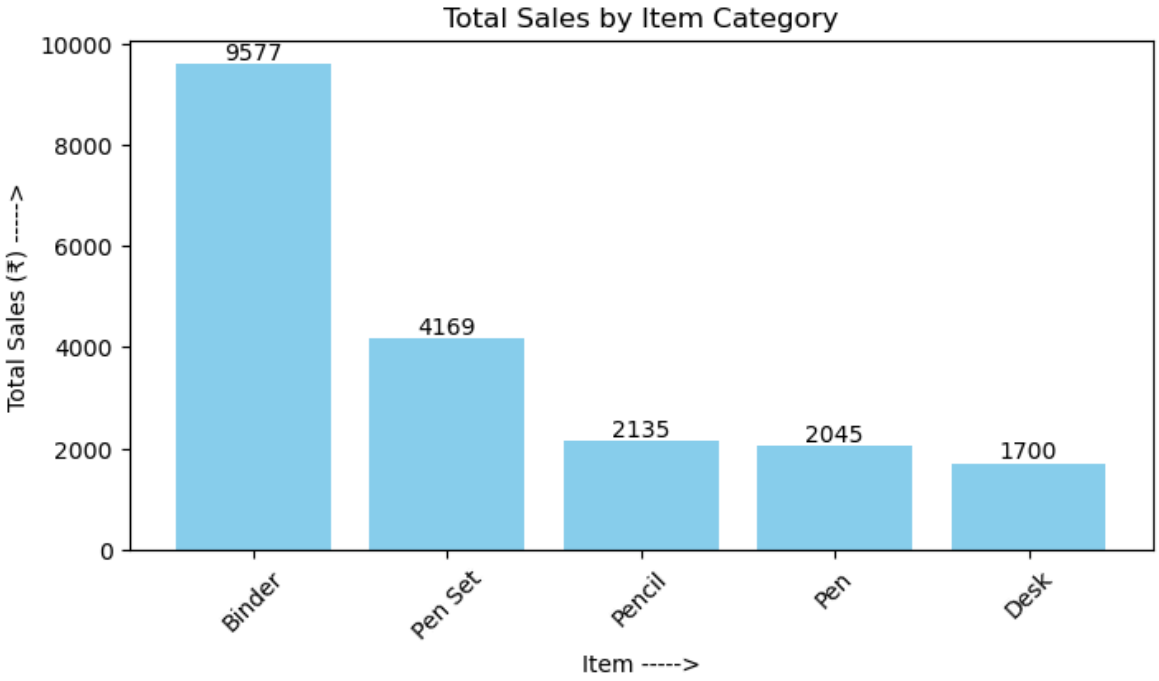
Which product category has the highest sales?

In [11]:
```
total_sales_by_category.max()
```

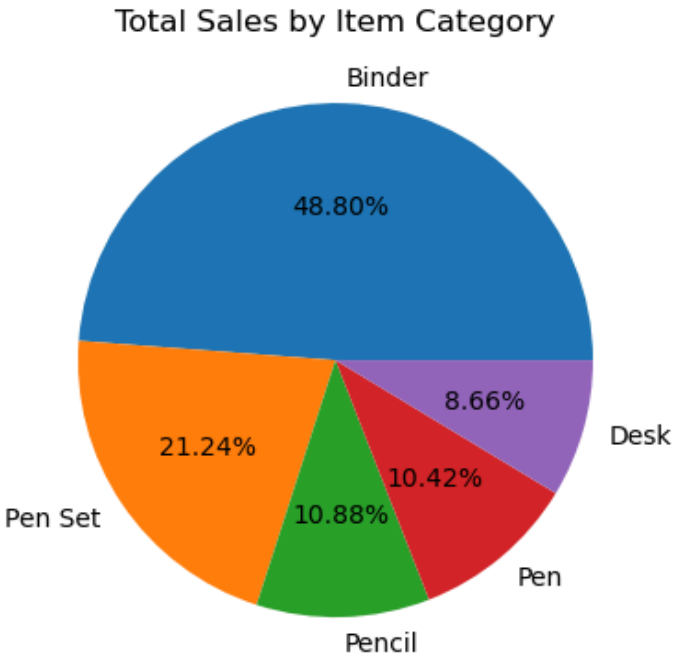
Out[11]:
```
S.No.               5
Item           Pencil
Total Sales    9577.65
dtype: object
```

Identify the top 10 best-selling products.

In [12]:
```
plt.figure(figsize=(8, 4))
total_sales_by_category = total_sales_by_category.sort_values(by='Total S
bars = plt.bar(total_sales_by_category['Item'], total_sales_by_category['
plt.xlabel('Item ----->')
plt.ylabel('Total Sales (₹) ----->')
plt.title('Total Sales by Item Category')
plt.xticks(rotation=45)
for bar in bars:
    totalsale = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, totalsale, int(totalsale),

plt.show()
```



In [13]:
```
plt.figure(figsize=(8, 4))
plt.pie(total_sales_by_category['Total Sales'], labels=total_sales_by_cat
plt.title('Total Sales by Item Category\n')
plt.axis('equal')

plt.show()
```



2. Customer Analysis:

Who are the top 10 customers by sales?

In [14]:
```
data['Rep'].unique()
```

Out[14]:
```
array(['Richard', 'Nick', 'Morgan', 'Susan', 'Matthew', 'James', 'Smit
h',
       'Bill', 'Thomas', 'Rachel', 'Alex'], dtype=object)
```

What is the total number of unique customers?

In [15]:
```
df['Rep'].nunique()
```

Out[15]:
```
11
```

In [16]:
```
total_sales_by_customer = df.groupby('Rep')['Total Sales'].sum().reset_in
total_sales_by_customer
```

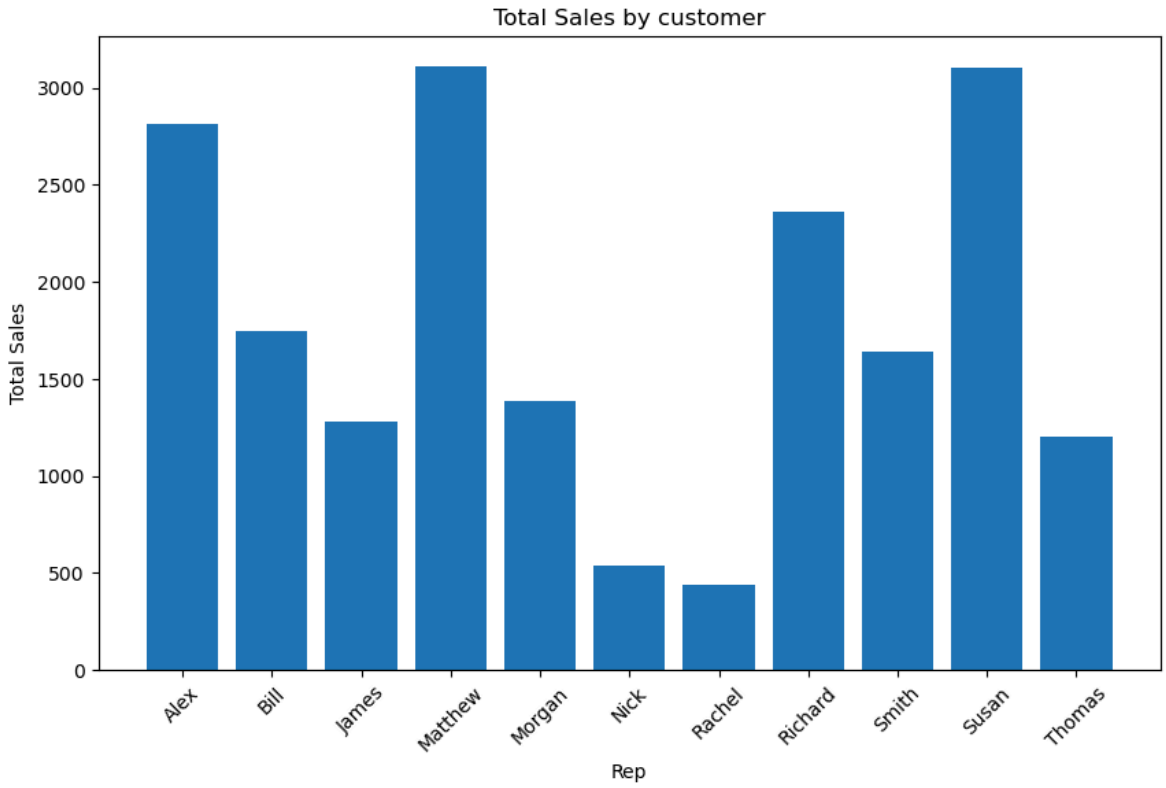
Out[16]:

	Rep	Total Sales
0	Alex	2812.19
1	Bill	1749.87
2	James	1283.61
3	Matthew	3109.44
4	Morgan	1387.77
5	Nick	536.75
6	Rachel	438.37
7	Richard	2363.04
8	Smith	1641.43
9	Susan	3102.30
10	Thomas	1203.11

In [17]:
```
total_sales_by_customer.max()
```

Out[17]:
```
Rep             Thomas
Total Sales    3109.44
dtype: object
```

In [18]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.bar(total_sales_by_customer['Rep'], total_sales_by_customer['Total Sa
plt.xlabel('Rep')
plt.ylabel('Total Sales')
plt.title('Total Sales by customer')
plt.xticks(rotation=45)
plt.show()
```



In [19]:
```
total_sales_by_customer_sorted = total_sales_by_customer.sort_values(by='
total_sales_by_customer_sorted
```

Out[19]:

	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
6	Rachel	438.37

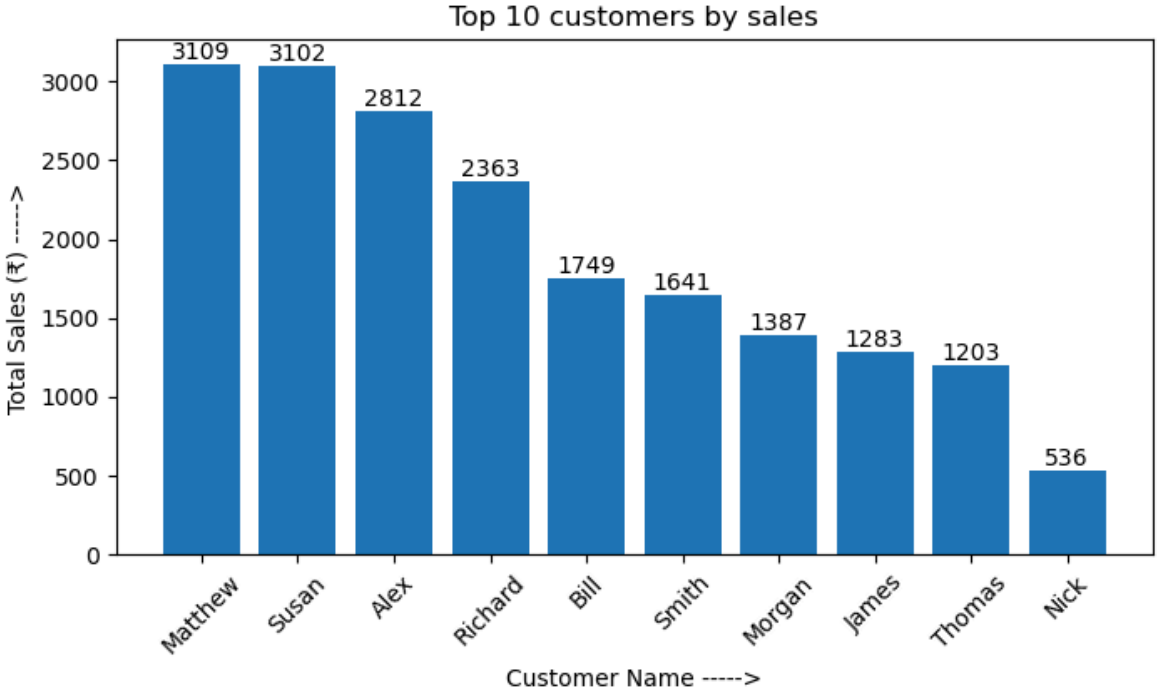
Who are the top 10 customers by sales?

In [20]:
```
top_10_customers = total_sales_by_customer_sorted.head(10)
top_10_customers
```

Out[20]:

	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

In [21]:
```
plt.figure(figsize=(8, 4))
bars = plt.bar(top_10_customers['Rep'],top_10_customers['Total Sales'], )
plt.xticks(rotation=45)
plt.xlabel('Customer Name ----->')
plt.ylabel('Total Sales (₹) ----->')
plt.title('Top 10 customers by sales')
plt.xticks(rotation=45)
for bar in bars:
    totalsale = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, totalsale, int(totalsale),
plt.show()
```



Analyze customer purchase frequency.

In [22]:
```
freq_customer = df["Rep"].value_counts()
freq_customer
```

Out[22]:
```
Richard    8
Bill       5
Alex       5
Matthew    4
James      4
Rachel     4
Morgan     3
Susan      3
Smith      3
Nick       2
Thomas     2
Name: Rep, dtype: int64
```

3. Time Series Analysis:

What are the monthly sales trends over the past year?

In [23]:
```
data.head(1)
```

Out[23]:

	OrderDate	Region	Rep	Item	Units	Unit Price	Total Sales
0	04-Jul-14	East	Richard	Pen Set	62	4.99	309.38

In [24]:
```
# Convert 'OrderDate' to datetime
data['OrderDate'] = pd.to_datetime(data['OrderDate'])
```

In [25]:
```
# Extract the month and year from the date
data['YearMonth'] = data['OrderDate'].dt.to_period('M')
```

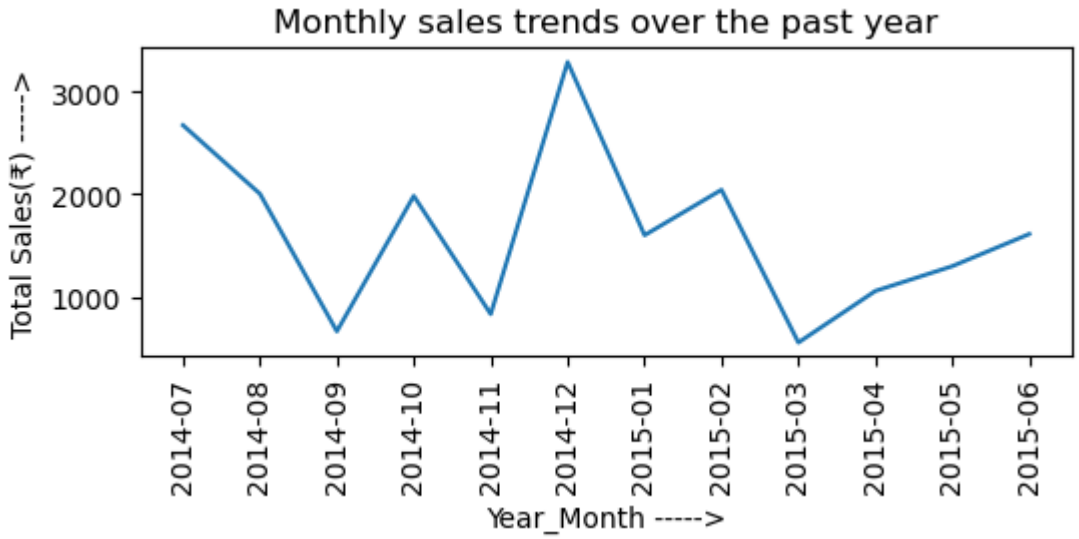
In [26]:
```
monthly_sales = data.groupby('YearMonth')['Total Sales'].sum().reset_inde
monthly_sales
```

Out[26]:

	YearMonth	Total Sales
0	2014-07	2673.23
1	2014-08	2005.55
2	2014-09	666.11
3	2014-10	1984.57
4	2014-11	833.78
5	2014-12	3288.47
6	2015-01	1602.09
7	2015-02	2044.33
8	2015-03	556.87
9	2015-04	1059.03
10	2015-05	1300.35
11	2015-06	1613.50

In [27]:
```
plt.figure(figsize=(6, 2))
plt.plot(monthly_sales["YearMonth"].astype(str),monthly_sales["Total Sale
plt.xticks(rotation = 90)
plt.xticks(rotation=45)
plt.xlabel('Year_Month ----->')
plt.ylabel('Total Sales(₹) ----->')
plt.title('Monthly sales trends over the past year')
plt.xticks(rotation=90)
```

Out[27]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 [Text(0, 0, '2014-07'),
  Text(1, 0, '2014-08'),
  Text(2, 0, '2014-09'),
  Text(3, 0, '2014-10'),
  Text(4, 0, '2014-11'),
  Text(5, 0, '2014-12'),
  Text(6, 0, '2015-01'),
  Text(7, 0, '2015-02'),
  Text(8, 0, '2015-03'),
  Text(9, 0, '2015-04'),
  Text(10, 0, '2015-05'),
  Text(11, 0, '2015-06')])
```



Identify any seasonal patterns in the sales data.

In [28]:
```
df.head(3)
```

Out[28]:

	OrderDate	Region	Rep	Item	Units	Unit Price	Total Sales	YearMonth
0	2014-07-04	East	Richard	Pen Set	62	4.99	309.38	2014-07
1	2014-07-12	East	Nick	Binder	29	1.99	57.71	2014-07
2	2014-07-21	Central	Morgan	Pen Set	55	12.49	686.95	2014-07

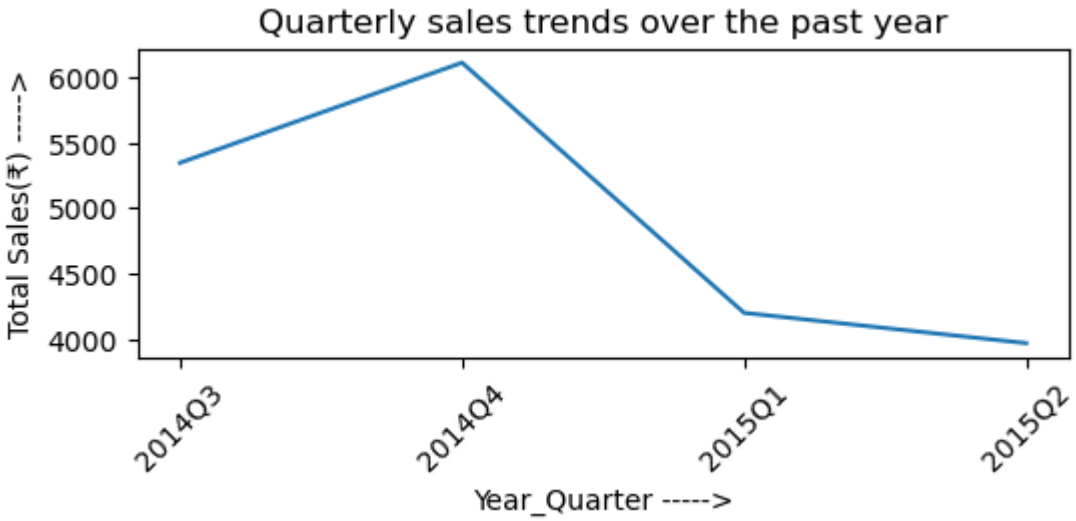
In [29]:
```
data['YearQuarter'] = data['OrderDate'].dt.to_period('Q')
monthly_sales = data.groupby('YearQuarter')['Total Sales'].sum().reset_in
monthly_sales
```

Out[29]:

	YearQuarter	Total Sales
0	2014Q3	5344.89
1	2014Q4	6106.82
2	2015Q1	4203.29
3	2015Q2	3972.88

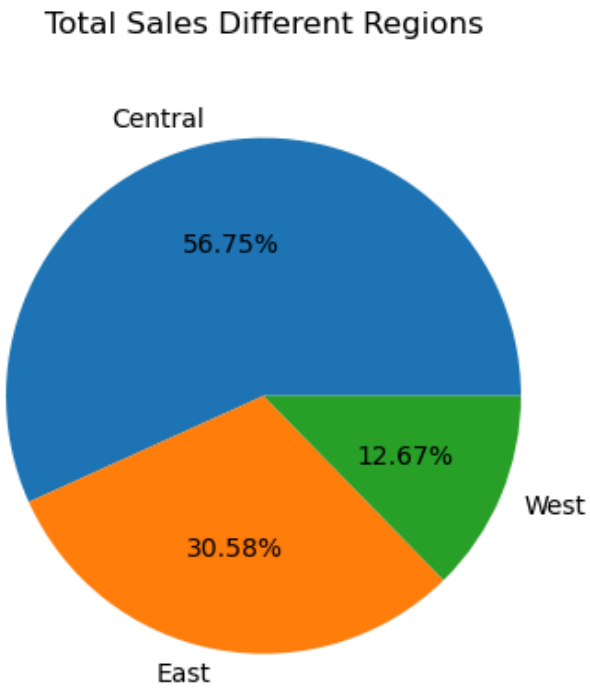
In [30]:
```
plt.figure(figsize=(6, 2))
plt.plot(monthly_sales["YearQuarter"].astype(str),monthly_sales["Total Sa
plt.xticks(rotation = 90)
plt.xticks(rotation=45)
plt.xlabel('Year_Quarter ----->')
plt.ylabel('Total Sales(₹) ----->')
plt.title('Quarterly sales trends over the past year')
plt.xticks(rotation=45)
```

Out[30]:
```
([0, 1, 2, 3],
 [Text(0, 0, '2014Q3'),
  Text(1, 0, '2014Q4'),
  Text(2, 0, '2015Q1'),
  Text(3, 0, '2015Q2')])
```

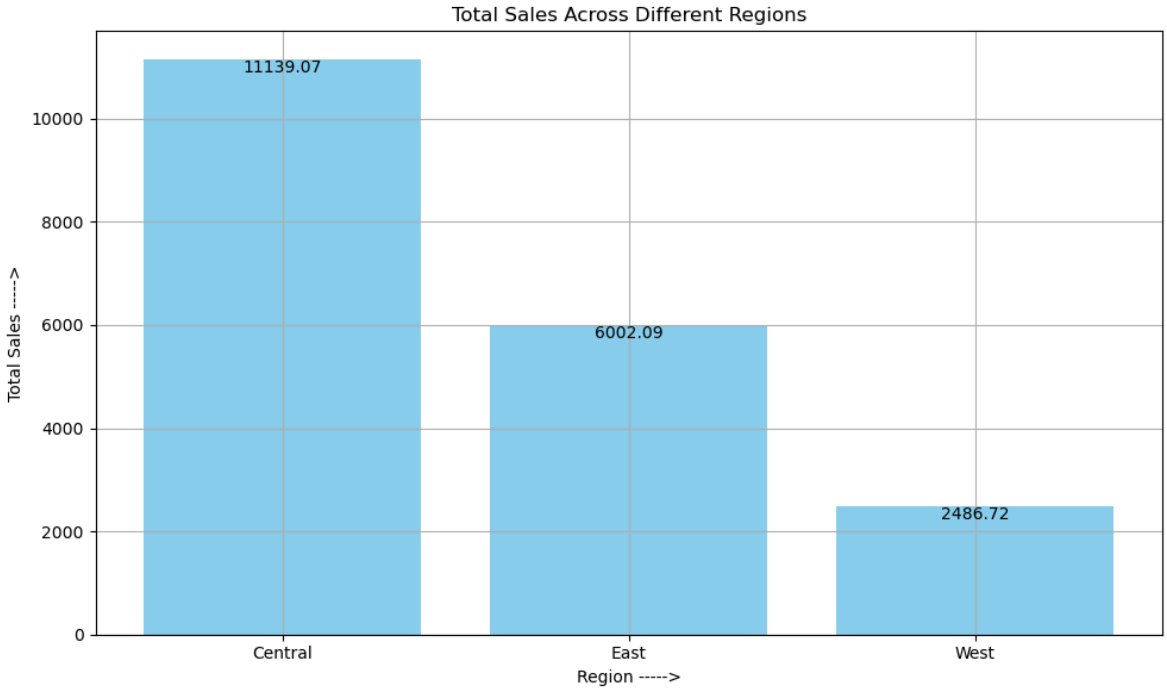

Quarterly sales trends over the past year
Total Sales(₹) ----->
6000
5500
5000
4500
4000
2014Q3
2014Q4
2015Q1
2015Q2
Year_Quarter ----->
```

In [31]:
```
for seasonal patterns we should have at least 2 years data
```

# 4. Geographical Analysis:

# - Which regions generate the most sales?

In [32]:
```
data.head(1)
```

Out[32]:

| | OrderDate | Region | Rep | Item | Units | Unit Price | Total Sales | YearMonth | YearQuarter |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 2014-07-04 | East | Richard | Pen Set | 62 | 4.99 | 309.38 | 2014-07 | 2014Q3 |

In [33]:
```
region_sales = data.groupby('Region')['Total Sales'].sum().reset_index()
region_sales
```

Out[33]:

| | Region | Total Sales |
|---|---|---|
| **0** | Central | 11139.07 |
| **1** | East | 6002.09 |
| **2** | West | 2486.72 |

In [34]:
```
region_sales.max()
```

Out[34]:
```
Region   West
Total Sales   11139.07
dtype: object
```
```

What are the sales trends across different regions?

In [35]:
```
plt.figure(figsize=(8, 4))
plt.pie(region_sales["Total Sales"],labels=region_sales['Region'], autopc

plt.title('Total Sales Different Regions\n\n')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a ci

plt.show()
```



In [36]:
```
# Aggregate sales data by region
region_sales = data.groupby('Region')['Total Sales'].sum().reset_index()

# Plotting the sales trends across different regions
plt.figure(figsize=(10, 6))
bars = plt.bar(region_sales['Region'], region_sales['Total Sales'], color

for bar, value in zip(bars, region_sales['Total Sales']):
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height(), value, ha


plt.xlabel('Region ----->')
plt.ylabel('Total Sales ----->')
plt.title('Total Sales Across Different Regions')
plt.grid(True)
plt.tight_layout()
plt.show()
```



5. Profit Analysis:

What is the total profit for each product category?

In [37]:
```
data.head(2)
```

Out[37]:

	OrderDate	Region	Rep	Item	Units	Unit Price	Total Sales	YearMonth	YearQuart
0	2014-07-04	East	Richard	Pen Set	62	4.99	309.38	2014-07	2014C
1	2014-07-12	East	Nick	Binder	29	1.99	57.71	2014-07	2014C

In [38]:
```
avg_price = df.groupby('Item')['Unit Price'].mean().reset_index()
avg_price
```

Out[38]:

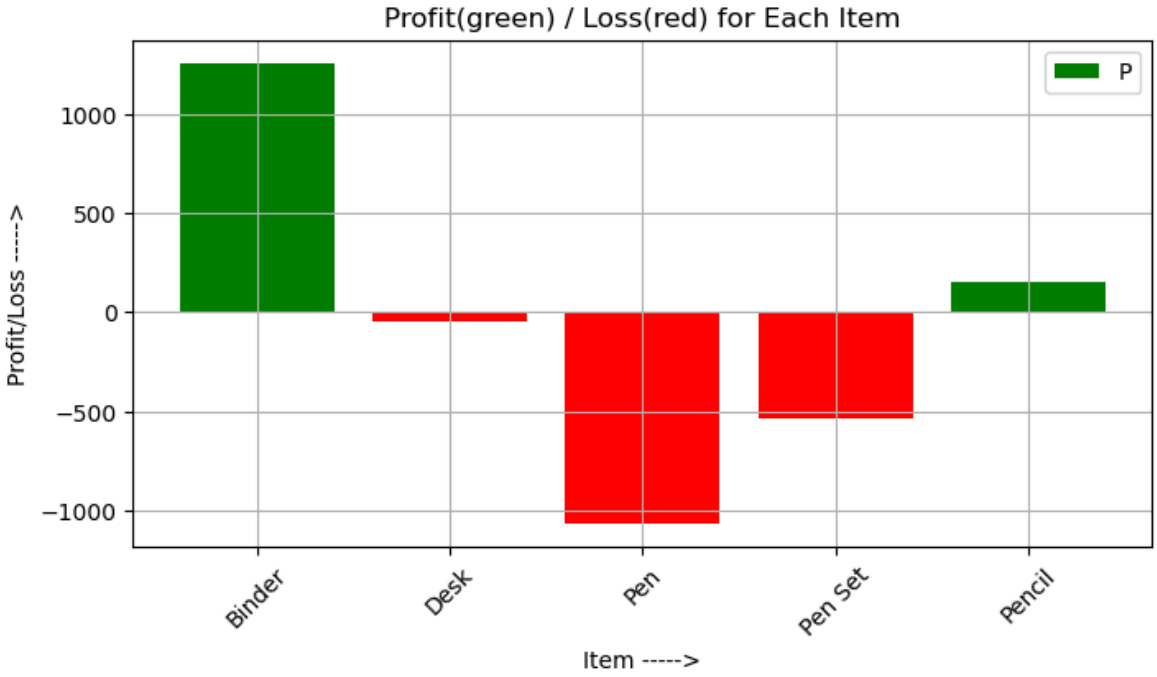
	Item	Unit Price
0	Binder	11.524000
1	Desk	175.000000
2	Pen	11.190000
3	Pen Set	11.912857
4	Pencil	2.774615

In [39]:
```
grouped_data = df.groupby('Item').agg({'Total Sales': 'sum', 'Units': 'su
# grouped_data['Average Unit Price'] = grouped_data['Total Sales'] / grou

# # Adding serial numbers for better readability
grouped_data['S.No.'] = grouped_data.index + 1

merged_data = pd.merge(grouped_data, avg_price, on='Item')

# Calculate the total cost by multiplying the average unit price by the u
merged_data['Total Cost'] = merged_data['Unit Price'] * merged_data['Unit

# Reordering columns for better readability
merged_data = merged_data[['S.No.', 'Item', 'Total Sales', 'Units', 'Unit
# Calculate profit/loss by subtracting Total Cost from Total Sales
merged_data['Profit/Loss'] = merged_data['Total Sales'] - merged_data['To

# Reordering columns for better readability
merged_data = merged_data[['S.No.', 'Item', 'Total Sales', 'Units', 'Unit

print(merged_data.to_string(index=False))
```

```
S.No.     Item  Total Sales  Units  Unit Price  Total Cost  Profit/Loss
    1   Binder      9577.65    722   11.524000 8320.328000  1257.322000
    2     Desk      1700.00     10  175.000000 1750.000000   -50.000000
    3      Pen      2045.22    278   11.190000 3110.820000 -1065.600000
    4  Pen Set      4169.87    395   11.912857 4705.578571  -535.708571
    5   Pencil      2135.14    716    2.774615 1986.624615   148.515385
```

In [40]:
```
# Determine colors based on profit/loss values
colors = ['green' if pl >= 0 else 'red' for pl in merged_data['Profit/Los
```

In [41]:
```
plt.figure(figsize=(8, 4))
bars = plt.bar(merged_data['Item'] ,merged_data['Profit/Loss'],color = co

# Adding labels with values on top of each bar
# for bar, value in zip(bars, merged_data['Profit/Loss']):
#     plt.text(bar.get_x() + bar.get_width()/2, bar.get_height(), value,

plt.xlabel('Item ----->')
plt.ylabel('Profit/Loss ----->')
plt.title('Profit(green) / Loss(red) for Each Item')
plt.legend('Profit')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```


Profit(green) / Loss(red) for Each Item
P
1000
500
0
−500
−1000
Profit/Loss ----->
Binder
Desk
Pen
Pen Set
Pencil
Item ----->
In [ ]: