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

In [3]:
```
df=pd.read_csv('supermart_grocery_sales.csv')
```

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

Out[5]:

| | Order ID | Customer Name | Category | Sub Category | City | Order Date | Region | Sales | Disco |
|---|---|---|---|---|---|---|---|---|---|
| 0 | OD1 | Harish | Oil & Masala | Masalas | Vellore | 11-08-2017 | North | 1254 | 0 |
| 1 | OD2 | Sudha | Beverages | Health Drinks | Krishnagiri | 11-08-2017 | South | 749 | 0 |
| 2 | OD3 | Hussain | Food Grains | Atta & Flour | Perambalur | 06-12-2017 | West | 2360 | 0 |
| 3 | OD4 | Jackson | Fruits & Veggies | Fresh Vegetables | Dharmapuri | 10-11-2016 | South | 896 | 0 |
| 4 | OD5 | Ridhesh | Food Grains | Organic Staples | Ooty | 10-11-2016 | South | 2355 | 0 |

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

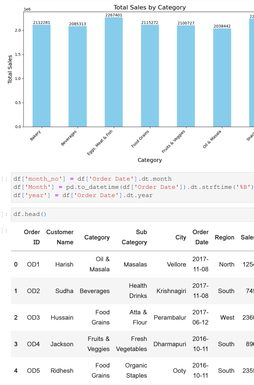
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Order ID       9994 non-null   object 
 1   Customer Name  9994 non-null   object 
 2   Category       9994 non-null   object 
 3   Sub Category   9994 non-null   object 
 4   City           9994 non-null   object 
 5   Order Date     9994 non-null   object 
 6   Region         9994 non-null   object 
 7   Sales          9994 non-null   int64  
 8   Discount       9994 non-null   float64
 9   Profit         9994 non-null   float64
 10  State          9994 non-null   object 
dtypes: float64(2), int64(1), object(8)
memory usage: 859.0+ KB
```

In [15]:
```
# Convert 'Order Date' to datetime format
df['Order Date'] = pd.to_datetime(df['Order Date'], errors='coerce')
```

In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype         
---  ------         --------------  -----         
 0   Order ID       9994 non-null   object        
 1   Customer Name  9994 non-null   object        
 2   Category       9994 non-null   object        
 3   Sub Category   9994 non-null   object        
 4   City           9994 non-null   object        
 5   Order Date     4042 non-null   datetime64[ns]
 6   Region         9994 non-null   object        
 7   Sales          9994 non-null   int64         
 8   Discount       9994 non-null   float64       
 9   Profit         9994 non-null   float64       
 10  State          9994 non-null   object        
dtypes: datetime64[ns](1), float64(2), int64(1), object(7)
memory usage: 859.0+ KB
```

In [19]:
```
# Assuming df is your DataFrame containing 'Category' and 'Sales' colu

# Grouping by 'Category' and calculating total sales for each category
Sales_category = df.groupby("Category")["Sales"].sum()

# Creating a bar plot for sales by category
plt.figure(figsize=(10, 6))  # Set the figure size
bars = Sales_category.plot(kind='bar', color='skyblue')

# Adding labels and title
plt.title('Total Sales by Category', fontsize=16)
plt.xlabel('Category', fontsize=14)
plt.ylabel('Total Sales', fontsize=14)
plt.xticks(rotation=45, ha='right')

# Adding value labels on top of the bars
for bar in bars.patches:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval, round(yval, 2), ha

# Displaying the plot
plt.tight_layout()  # Adjust layout to prevent clipping of tick-labels
plt.show()
plt.savefig('myactivities')
```



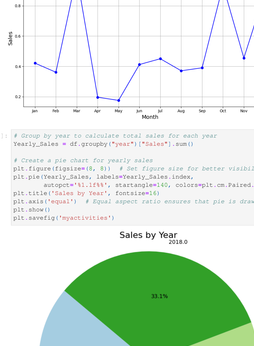

In [21]:
```
df['month_no'] = df['Order Date'].dt.month
df['Month'] = pd.to_datetime(df['Order Date']).dt.strftime('%B')
df['year'] = df['Order Date'].dt.year
```

In [23]:
```
df.head()
```

Out[23]:

| | Order ID | Customer Name | Category | Sub Category | City | Order Date | Region | Sales | Disco |
|---|---|---|---|---|---|---|---|---|---|
| 0 | OD1 | Harish | Oil & Masala | Masalas | Vellore | 2017-11-08 | North | 1254 | 0 |
| 1 | OD2 | Sudha | Beverages | Health Drinks | Krishnagiri | 2017-11-08 | South | 749 | 0 |
| 2 | OD3 | Hussain | Food Grains | Atta & Flour | Perambalur | 2017-06-12 | West | 2360 | 0 |
| 3 | OD4 | Jackson | Fruits & Veggies | Fresh Vegetables | Dharmapuri | 2016-10-11 | South | 896 | 0 |
| 4 | OD5 | Ridhesh | Food Grains | Organic Staples | Ooty | 2016-10-11 | South | 2355 | 0 |

In [25]:
```
# Assuming df is your DataFrame with 'Month' and 'Sales' columns
monthly_sales = df.groupby('Month')['Sales'].sum().reset_index()

# Sort the data by month
monthly_sales_sorted = monthly_sales.sort_values(by='Month')

# Create the line chart
plt.figure(figsize=(10, 6))
plt.plot(monthly_sales_sorted['Month'],
         monthly_sales_sorted['Sales'], marker='o', color='b')
plt.title('Sales by Month', fontsize=16)
plt.xlabel('Month', fontsize=14)
plt.ylabel('Sales', fontsize=14)
plt.xticks(monthly_sales_sorted['Month'], ['Jan', 'Feb', 'Mar',
                                           'Apr', 'May', 'Jun',
                                           'Jul', 'Aug', 'Sep',
                                           'Oct', 'Nov', 'Dec'])

plt.grid(True)
plt.tight_layout()  # Adjust layout to prevent clipping
plt.show()
plt.savefig('myactivities')
```




In [27]:
```
# Group by year to calculate total sales for each year
Yearly_Sales = df.groupby('year')['Sales'].sum()

# Create a pie chart for yearly sales
plt.figure(figsize=(8, 8))  # Set figure size for better visibility
plt.pie(Yearly_Sales, labels=Yearly_Sales.index,
        autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors
plt.title('Sales by Year', fontsize=16)
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a
plt.show()
plt.savefig('myactivities')
```


In [33]:
```
# Step 1: Extract relevant columns
city_sales = df[['City', 'Sales']]

# Step 2: Calculate total sales per city
total_sales = city_sales.groupby('City').sum()

# Step 3: Sort the cities by sales in descending order
sorted_cities = total_sales.sort_values(by='Sales', ascending=False)

# Step 4: Select the top 5 cities
top_cities = sorted_cities.head(5)

# Step 5: Plot the bar chart
plt.figure(figsize=(10, 6))  # Set figure size for better visibility
bars = plt.bar(top_cities.index, top_cities['Sales'], color='orange')

# Adding labels and title
plt.xlabel('City', fontsize=14)
plt.ylabel('Sales', fontsize=14)
plt.title('Top 5 Cities by Sales', fontsize=16)
plt.xticks(rotation=45)

# Adding data labels on top of the bars
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval, round(yval, 2),
             ha='center', va='bottom', fontsize=10)

plt.tight_layout()  # Adjust layout to prevent clipping
plt.show()
plt.savefig('myactivities')
```




In [ ]:
```
```