DATA ANALYSIS PYTHON PROJECT - BLINKIT ANALYSIS

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sn

df=pd.read_csv("C:\\Users\\hp\\Downloads\\BlinkIT Grocery Data.csv")

df.head(5)
```

```
  Item Fat Content Item Identifier              Item Type  \
0          Regular           FDX32  Fruits and Vegetables
1          Low Fat           NCB42    Health and Hygiene
2          Regular           FDR28          Frozen Foods
3          Regular           FDL50                Canned
4          Low Fat           DRI25           Soft Drinks

   Outlet Establishment Year Outlet Identifier Outlet Location Type  \
0                       2012            OUT049               Tier 1
1                       2022            OUT018               Tier 3
2                       2016            OUT046               Tier 1
3                       2014            OUT013               Tier 3
4                       2015            OUT045               Tier 2

  Outlet Size        Outlet Type  Item Visibility  Item Weight
Sales  \
0      Medium  Supermarket Type1         0.100014        15.10
145.4786
1      Medium  Supermarket Type2         0.008596        11.80
115.3492
2       Small  Supermarket Type1         0.025896        13.85
165.0210
3        High  Supermarket Type1         0.042278        12.15
126.5046
4       Small  Supermarket Type1         0.033970        19.60
55.1614

   Rating
0     5.0
1     5.0
2     5.0
3     5.0
4     5.0
```

```
df.tail()
```

```
     Item Fat Content Item Identifier              Item Type  \
8518          Low Fat           NCT53  Health and Hygiene
8519          Low Fat           FDN09         Snack Foods
```

```
8520          Low Fat           DRE13         Soft Drinks
8521          Regular           FDT50               Dairy
8522          Regular           FDM58         Snack Foods

      Outlet Establishment Year Outlet Identifier Outlet Location Type
\
8518                       2018            OUT027                Tier 3

8519                       2018            OUT027                Tier 3

8520                       2018            OUT027                Tier 3

8521                       2018            OUT027                Tier 3

8522                       2018            OUT027                Tier 3


     Outlet Size        Outlet Type  Item Visibility  Item Weight
Sales  \
8518      Medium  Supermarket Type3         0.000000          NaN
164.5526
8519      Medium  Supermarket Type3         0.034706          NaN
241.6828
8520      Medium  Supermarket Type3         0.027571          NaN
86.6198
8521      Medium  Supermarket Type3         0.107715          NaN
97.8752
8522      Medium  Supermarket Type3         0.000000          NaN
112.2544

      Rating
8518     4.0
8519     4.0
8520     4.0
8521     4.0
8522     4.0

df.describe()

       Outlet Establishment Year  Item Visibility  Item Weight
Sales  \
count                8523.000000      8523.000000  7060.000000
8523.000000
mean                 2016.450546         0.066132    12.857645
140.992783
std                     3.189396         0.051598     4.643456
62.275067
min                  2011.000000         0.000000     4.555000
31.290000
25%                  2014.000000         0.026989     8.773750
```

```
93.826500
50%                  2016.000000         0.053931    12.600000
143.012800
75%                  2018.000000         0.094585    16.850000
185.643700
max                  2022.000000         0.328391    21.350000
266.888400

            Rating
count  8523.000000
mean      3.965857
std       0.605651
min       1.000000
25%       4.000000
50%       4.000000
75%       4.200000
max       5.000000
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item Fat Content           8523 non-null   object
 1   Item Identifier            8523 non-null   object
 2   Item Type                  8523 non-null   object
 3   Outlet Establishment Year  8523 non-null   int64
 4   Outlet Identifier          8523 non-null   object
 5   Outlet Location Type       8523 non-null   object
 6   Outlet Size                8523 non-null   object
 7   Outlet Type                8523 non-null   object
 8   Item Visibility            8523 non-null   float64
 9   Item Weight                7060 non-null   float64
 10  Sales                      8523 non-null   float64
 11  Rating                     8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

Size of Data

```
df.shape
```

```
(8523, 12)
```

Field Information

```
df.columns
```

```
Index(['Item Fat Content', 'Item Identifier', 'Item Type',
       'Outlet Establishment Year', 'Outlet Identifier',
       'Outlet Location Type', 'Outlet Size', 'Outlet Type', 'Item
Visibility',
       'Item Weight', 'Sales', 'Rating'],
      dtype='object')
```

Data Type

```
df.dtypes
```

```
Item Fat Content              object
Item Identifier               object
Item Type                     object
Outlet Establishment Year      int64
Outlet Identifier             object
Outlet Location Type          object
Outlet Size                   object
Outlet Type                   object
Item Visibility              float64
Item Weight                  float64
Sales                        float64
Rating                       float64
dtype: object
```

Data Cleaning

```
print(df['Item Fat Content'].unique())
```

```
['Regular' 'Low Fat']
```

BUSINESS REQUIREMENTS

KPI'S REQUIREMENT

```
# Total Sales
Total_sales = df['Sales'].sum()

# Average Sales
Avg_sales = df['Sales'].mean()

# No of Items Sold
No_of_items_sold = df['Sales'].count()

# Average Rating
Avg_rating = df['Rating'].mean()

# Display

print(f"Total Sales: ${Total_sales:,.1f}")
```

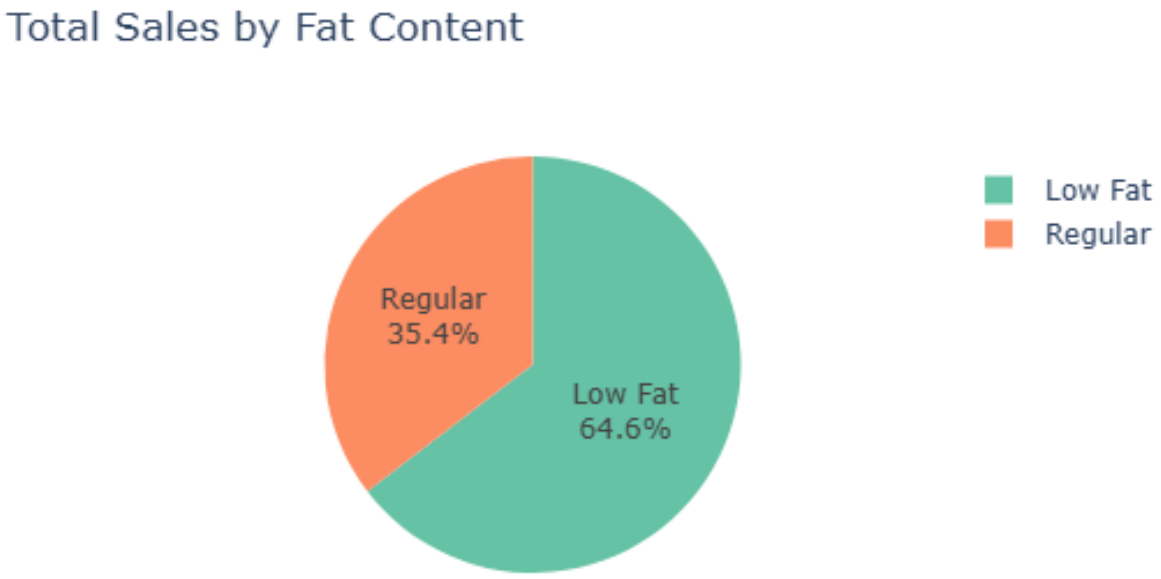
```
print(f"Average Sales: ${Avg_sales:,.0f}")

print(f"Number of Items Sold: {No_of_items_sold:,.0f}")

print(f"Average Rating: {Avg_rating:,.1f}")
```

```
Total Sales: $1,201,681.5
Average Sales: $141
Number of Items Sold: 8,523
Average Rating: 4.0
```

CHART REQUIREMENT

1. Total Sales by Fat Content

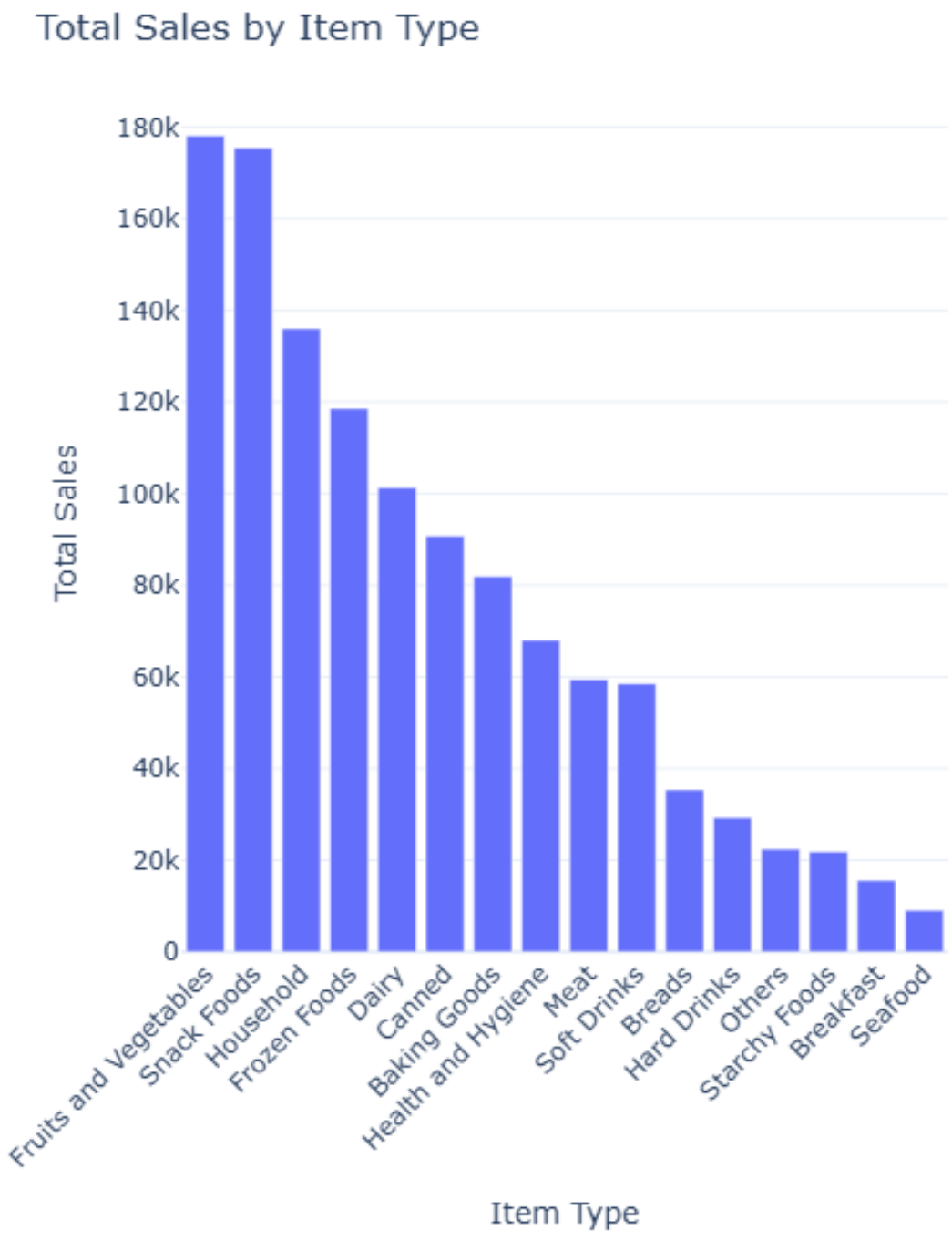
```
import plotly.express as px

# Grouping sales by 'Item Fat Content'
Sales_by_fat = df.groupby("Item Fat Content")
['Sales'].sum().reset_index()

# Create the pie chart
fig = px.pie(Sales_by_fat,
             names="Item Fat Content",
             values="Sales",
             title="Total Sales by Fat Content",
             hole=0,        # For donut chart, set hole > 0
             color_discrete_sequence=px.colors.qualitative.Set2)

fig.update_traces(textinfo='percent+label')

# Show the plot
fig.show()
```



2. Total Sales by Item Type

```
import plotly.express as px
import pandas as pd

# Group and sort data
Sales_by_Item = df.groupby("Item Type")
['Sales'].sum().sort_values(ascending=False).reset_index()

# Create Plotly bar chart
fig = px.bar(Sales_by_Item,
             x='Item Type',
             y='Sales',
             text='Sales',
             title='Total Sales by Item Type',
             labels={'Sales': 'Total Sales', 'Item Type': 'Item
Type'},
             template='plotly_white')

# Customize text position
fig.update_traces(texttemplate='%{text:.0f}', textposition='outside')

# Update layout for better spacing
fig.update_layout(xaxis_tickangle=-45,
                  uniformtext_minsize=8,
```

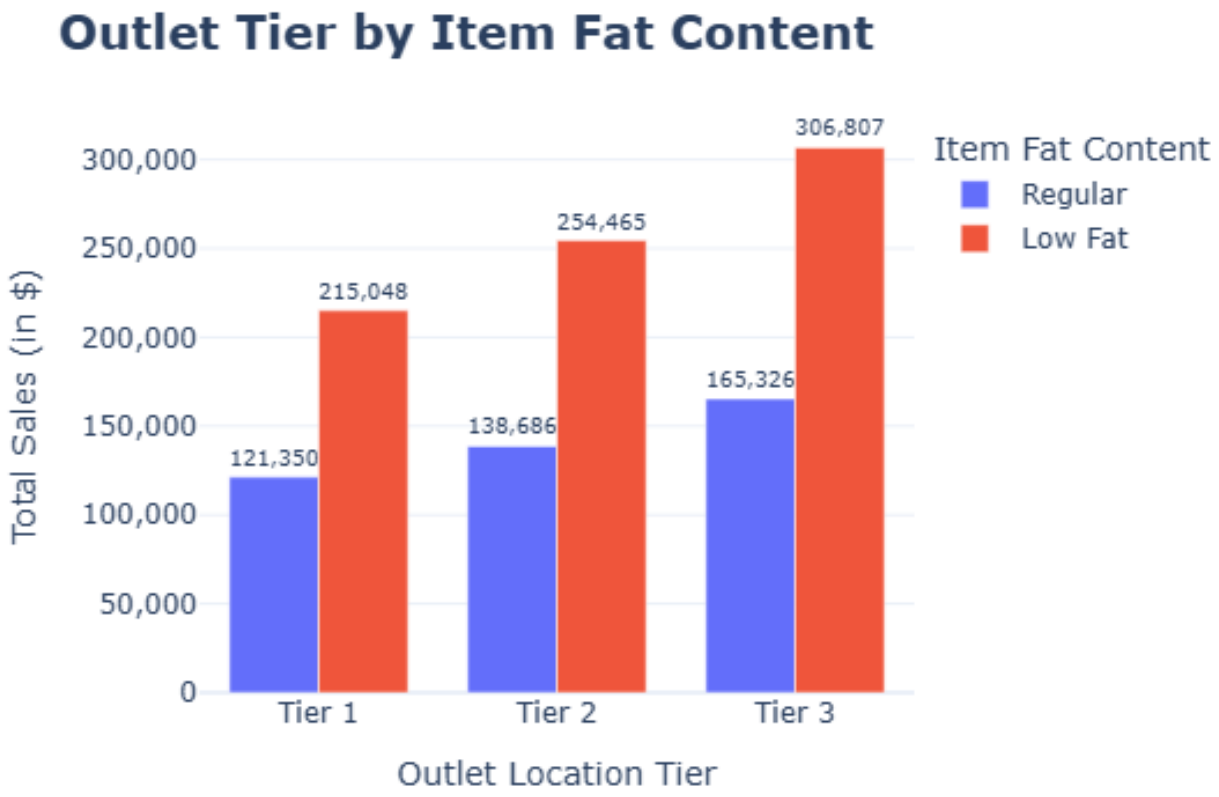
```
                  uniformtext_mode='hide',
                  height=600,
                  margin=dict(t=50, b=100))

# Show the plot
fig.show()
```



3. Fat Content by Outlet for Total Sales

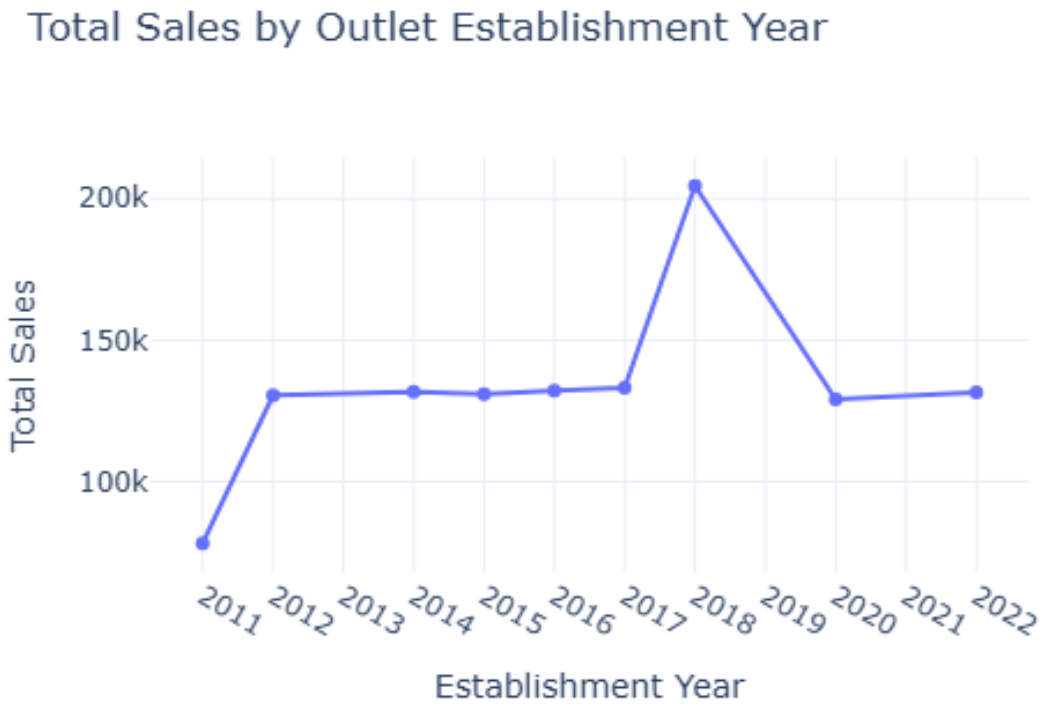
```
# Grouping and reshaping
grouped = df.groupby(['Outlet Location Type', 'Item Fat Content'])
['Sales'].sum().unstack()
grouped = grouped[['Regular', 'Low Fat']]  # Maintain order

# Melt for Plotly
grouped_reset = grouped.reset_index().melt(id_vars='Outlet Location
Type',
                                           var_name='Item Fat
Content',
                                           value_name='Total Sales')

# Plotly bar chart
fig = px.bar(grouped_reset,
             x='Outlet Location Type',
             y='Total Sales',
             color='Item Fat Content',
             barmode='group',
             text='Total Sales',
             title='<b>Outlet Tier by Item Fat Content</b>',
             labels={
                 'Outlet Location Type': 'Outlet Location Tier',
                 'Total Sales': 'Total Sales (in $)',
                 'Item Fat Content': 'Fat Content'
             },
             color_discrete_map={
                 'Low Fat': '#EF553B',  # orange-red
                 'Regular': '#636EFA'  # Plotly blue
             },
             template='plotly_white')

# Formatting numbers with commas and positioning
fig.update_traces(
    texttemplate='%{text:,.0f}',
    textposition='outside'
)

# Layout adjustments
fig.update_layout(
    xaxis_title='Outlet Location Tier',
    yaxis_title='Total Sales (in $)',
    legend_title='Item Fat Content',
    uniformtext_minsize=8,
    uniformtext_mode='show',
    bargap=0.25,
    yaxis_tickformat=',',
    title_font=dict(size=20),
    margin=dict(l=40, r=40, t=60, b=40)
```

```
)

fig.show()
```



4. Total Sales by Outlet Establishment

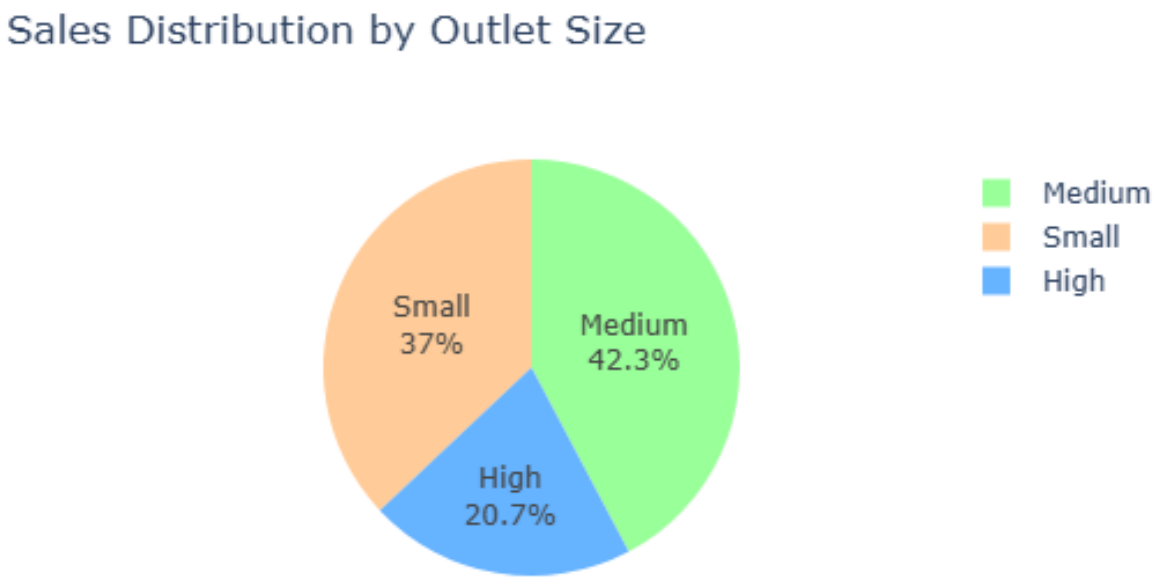
```
# Group by Establishment Year and calculate total sales
est_sales = df.groupby('Outlet Establishment Year')
['Sales'].sum().sort_index().reset_index()

# Plot with Plotly Express
fig = px.line(est_sales, x='Outlet Establishment Year', y='Sales',
              title='Total Sales by Outlet Establishment Year',
              markers=True,
              labels={'Outlet Establishment Year': 'Establishment
Year', 'Sales': 'Total Sales'})

fig.update_layout(
    xaxis=dict(tickmode='linear'),
    yaxis=dict(title='Total Sales'),
    template='plotly_white'
)

fig.show()
```

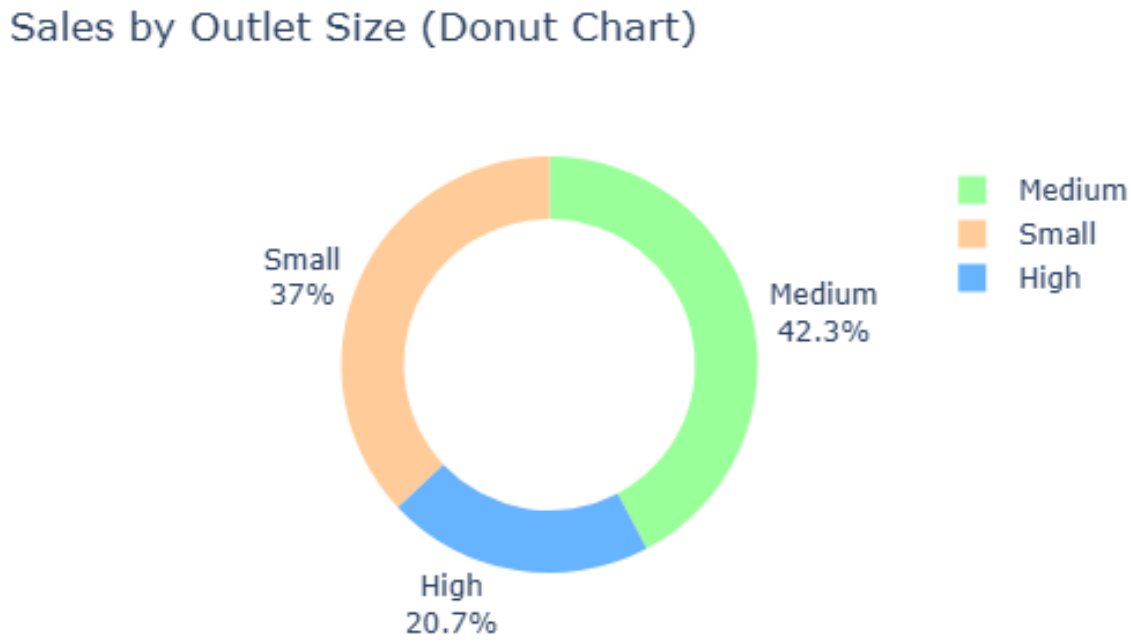


5. Sales by Outlet Size

```
import plotly.graph_objects as go

# Group by Outlet Size
size_sales = df.groupby('Outlet Size')['Sales'].sum()

# Pie Chart
fig = go.Figure(data=[go.Pie(
    labels=size_sales.index,
    values=size_sales.values,
    textinfo='percent+label',
    marker=dict(colors=['#66b3ff', '#99ff99', '#ffcc99']),
    hole=0  # 0 = pie chart (no hole)
)])

fig.update_layout(title_text='Sales Distribution by Outlet Size')
fig.show()

# Donut Chart
fig = go.Figure(data=[go.Pie(
    labels=size_sales.index,
    values=size_sales.values,
    textinfo='percent+label',
    marker=dict(colors=['#66b3ff', '#99ff99', '#ffcc99']),
```

```
    hole=0.7  # >0 = donut chart
)])

fig.update_layout(title_text='Sales by Outlet Size (Donut Chart)')
fig.show()
```





6. Sales by Outlet Location

```
# Group by Outlet Location
location_sales = df.groupby('Outlet Location Type')
['Sales'].sum().sort_values(ascending=True)

# Plotly horizontal bar chart
fig = go.Figure(go.Bar(
    x=location_sales.values,
    y=location_sales.index,
    orientation='h',
    marker=dict(color='#6A5ACD'),
    text=[f'{int(val):,}' for val in location_sales.values],
    textposition='inside',
    textfont=dict(size=12, color='black')
))

# Update layout
fig.update_layout(
    title='Sales by Outlet Location',
    xaxis_title='Total Sales',
    yaxis_title='Outlet Location Type',
    template='plotly_white',
    margin=dict(l=100, r=30, t=50, b=30)
)
```

```
fig.show()
```


Sales by Outlet Location
Outlet Location Type
Tier 3
Tier 2
Tier 1
472,133
393,150
336,397
0
100k
200k
300k
400k
Total Sales