Blinkit Sales Analysis - EDA

Import all the necessary libraries

In [33]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Import Raw Data

In [34]:
```
df = pd.read_csv("C:/Users/91911/Desktop/eda python/blinkit_data.csv")
```

Sample Data

In [35]:
```
df.head(20)
```

Out[35]:

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outle Size
0	Regular	FDX32	Fruits and Vegetables	2012	OUT049	Tier 1	Mediun
1	Low Fat	NCB42	Health and Hygiene	2022	OUT018	Tier 3	Mediun
2	Regular	FDR28	Frozen Foods	2010	OUT046	Tier 1	Smal
3	Regular	FDL50	Canned	2000	OUT013	Tier 3	High
4	Low Fat	DRI25	Soft Drinks	2015	OUT045	Tier 2	Smal
5	low fat	FDS52	Frozen Foods	2020	OUT017	Tier 2	Smal
6	Low Fat	NCU05	Health and Hygiene	2011	OUT010	Tier 3	Smal
7	Low Fat	NCD30	Household	2015	OUT045	Tier 2	Smal
8	Low Fat	FDW20	Fruits and Vegetables	2000	OUT013	Tier 3	High
9	Low Fat	FDX25	Canned	1998	OUT027	Tier 3	Mediun
10	LF	FDX21	Snack Foods	1998	OUT027	Tier 3	Mediun
11	Low Fat	NCU41	Health and Hygiene	2017	OUT035	Tier 2	Smal
12	Low Fat	FDL20	Fruits and Vegetables	2022	OUT018	Tier 3	Mediun
13	Low Fat	NCR54	Household	2000	OUT013	Tier 3	High
14	Low Fat	FDH19	Meat	1998	OUT027	Tier 3	Mediun
15	Regular	FDB57	Fruits and Vegetables	2017	OUT035	Tier 2	Smal
16	Low Fat	FDO23	Breads	2022	OUT018	Tier 3	Mediun
17	Low Fat	NCB07	Household	2012	OUT049	Tier 1	Mediun
18	Low Fat	FDJ56	Fruits and Vegetables	1998	OUT027	Tier 3	Mediun
19	Low Fat	DRN47	Hard Drinks	2022	OUT018	Tier 3	Mediun

In [36]:
```
df.dtypes
```

```
Out[36]: 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
```

Size of Data

```
In [37]: print("size of data is :" , df.shape)
```

```
size of data is : (8523, 12)
```

Data Cleaning

```
In [38]: print(df['Item Fat Content'].unique())
```

```
['Regular' 'Low Fat' 'low fat' 'LF' 'reg']
```

```
In [39]: df['Item Fat Content'] = df['Item Fat Content'].replace({'low fat' :'Low Fat
```

```
In [40]: print(df['Item Fat Content'].unique())
```

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

KPI Requirements :-

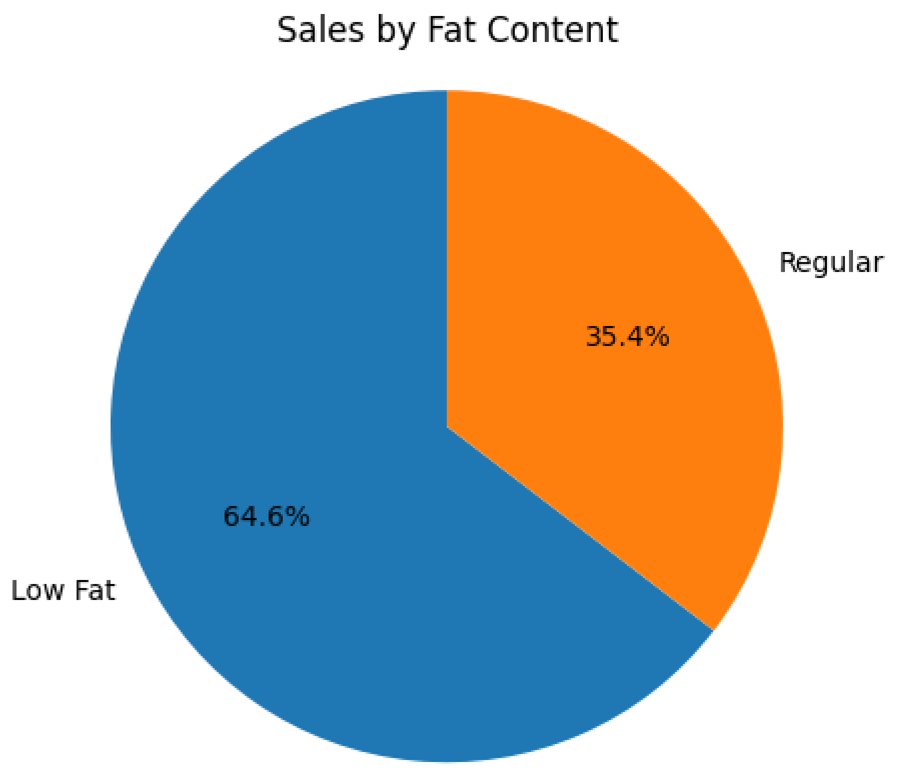
```
In [41]: # Total Sales
         total_sales = df['Sales'].sum()

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

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

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

         # Display
         print(f"Total Sales: ${total_sales:,.0f}")
         print(f"Average Sales: ${avg_sales:,.0f}")
         print(f"No of Items Sold: {no_of_items_sold:,.0f}")
         print(f"Average Ratings: {avg_ratings:,.0f}")
```

```
Total Sales: $1,201,681
Average Sales: $141
No of Items Sold: 8,523
Average Ratings: 4
```

Charts Requirements

Total Sales by Fat content

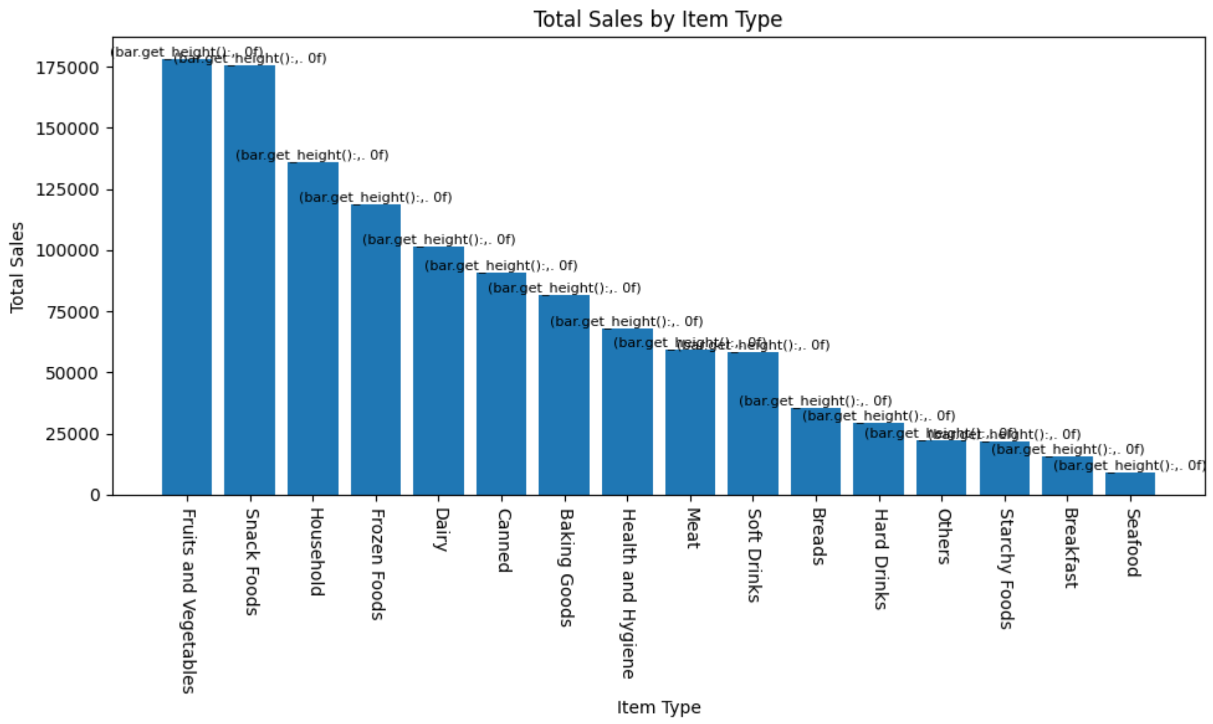
In [42]:
```python
# Total Sales by Fat Content
sales_by_fat = df.groupby('Item Fat Content')['Sales'].sum()

plt.pie(sales_by_fat,
        labels=sales_by_fat.index,
        autopct='%.1f%%',
        startangle=90)

plt.title('Sales by Fat Content')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circl
plt.show()
```



Sales By Item Type

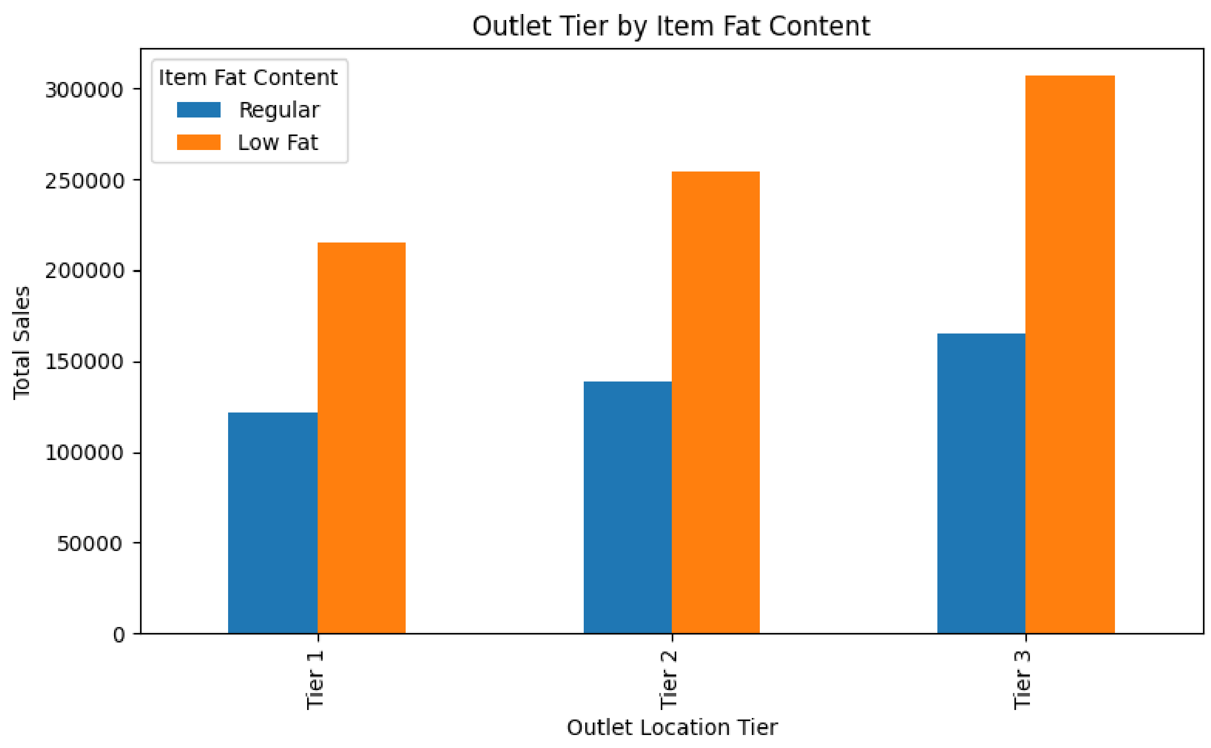
In [43]:
```python
sales_by_type = df.groupby('Item Type') ['Sales'].sum().sort_values(ascendir
plt.figure(figsize=(10, 6))
bars = plt.bar(sales_by_type.index, sales_by_type.values)
plt.xticks(rotation=-90)
plt.xlabel('Item Type')
plt.ylabel('Total Sales')
plt.title('Total Sales by Item Type')
```

```
for bar in bars:
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(),
         f'(bar.get_height():,. 0f)', ha='center', va='bottom', fontsize=8)
plt.tight_layout()
plt.show()
```



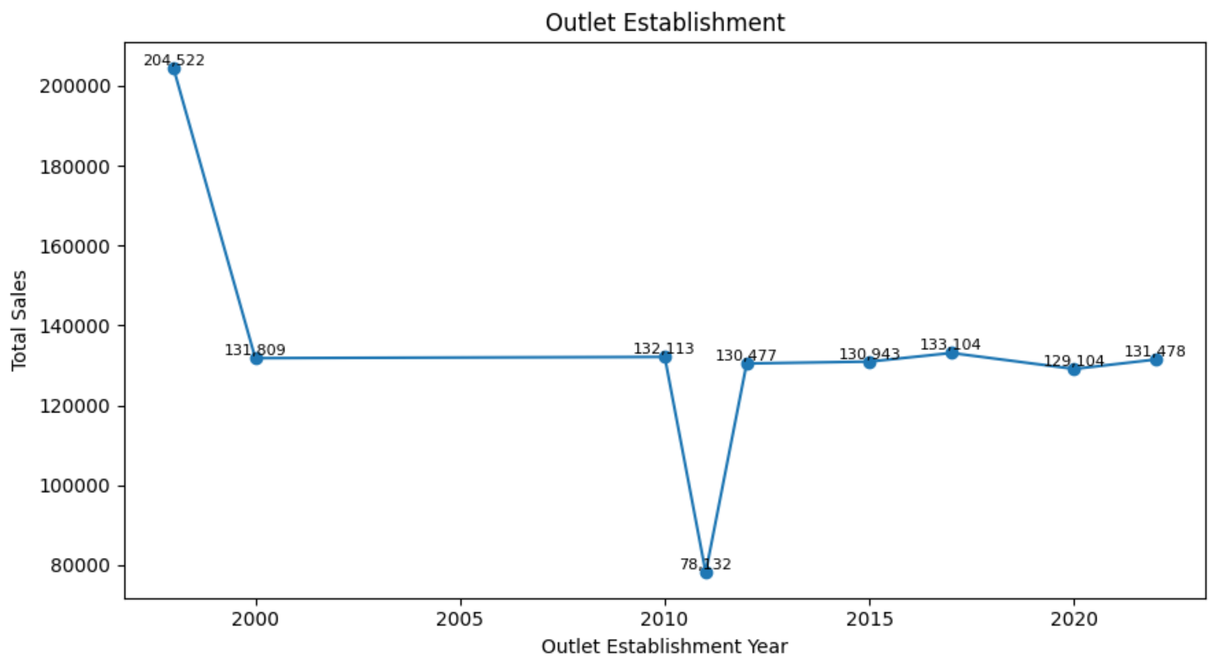
Total Sales By Outlet (Fat Content)

In [44]:
```
grouped = df.groupby(['Outlet Location Type', 'Item Fat Content'])['Sales'].
grouped = grouped[['Regular', 'Low Fat']]  # Optional filtering order

ax = grouped.plot(kind='bar', figsize=(8, 5), title='Outlet Tier by Item Fat
plt.xlabel('Outlet Location Tier')
plt.ylabel('Total Sales')
plt.legend(title='Item Fat Content')
plt.tight_layout()
plt.show()
```



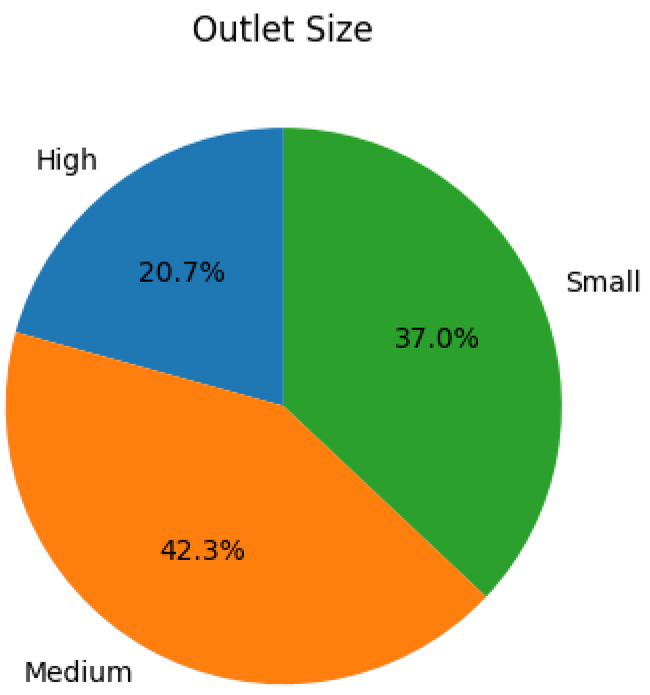
Total Sales By Outlet Establishment Year

```
In [45]: sales_by_year = df.groupby('Outlet Establishment Year')['Sales'].sum().sort_

         plt.figure(figsize=(9, 5))
         plt.plot(sales_by_year.index, sales_by_year.values, marker='o', linestyle='-

         plt.xlabel('Outlet Establishment Year')
         plt.ylabel('Total Sales')
         plt.title('Outlet Establishment')

         for x, y in zip(sales_by_year.index, sales_by_year.values):
             plt.text(x, y, f'{y:,.0f}', ha='center', va='bottom', fontsize=8)

         plt.tight_layout()
         plt.show()
```



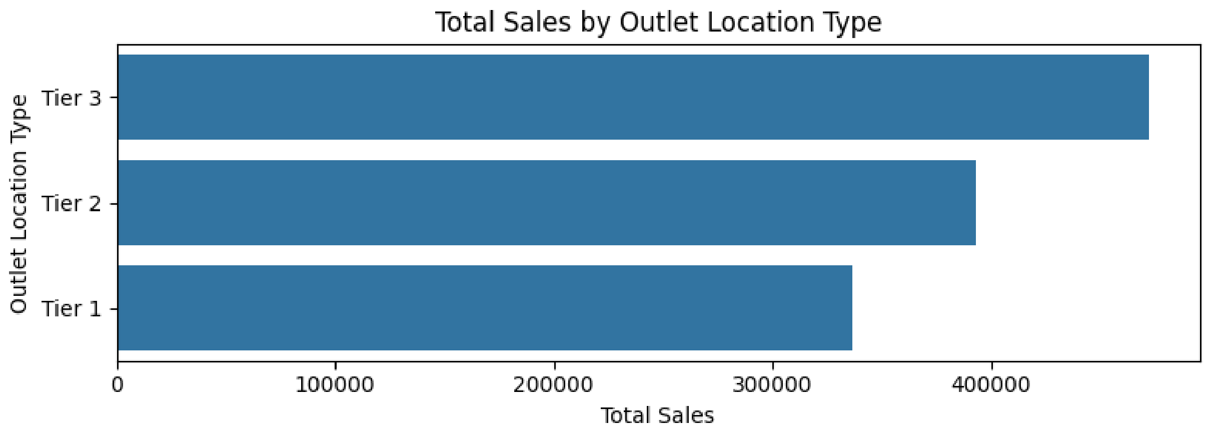
Sales By Outlet Size

In [46]:
```
sales_by_size = df.groupby('Outlet Size')['Sales'].sum()

plt.figure(figsize=(4, 4))
plt.pie(sales_by_size,
        labels=sales_by_size.index,
        autopct='%1.1f%%',
        startangle=90)

plt.title('Outlet Size')
plt.tight_layout()
plt.show()
```



Sales By Location

In [47]:
```
sales_by_location = df.groupby('Outlet Location Type')['Sales'].sum().reset_
sales_by_location = sales_by_location.sort_values('Sales', ascending=False)

plt.figure(figsize=(8, 3))  # Smaller height, enough width
ax = sns.barplot(x='Sales', y='Outlet Location Type', data=sales_by_locatior

plt.title('Total Sales by Outlet Location Type')
plt.xlabel('Total Sales')
plt.ylabel('Outlet Location Type')
plt.tight_layout()  # Ensures layout fits without scroll
plt.show()
```



In [ ]: