DATA ANALYSIS PYTHON PROJECT - BLINKIT ANALYSIS

Import Libraries

In [1]:

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

Import Raw Data

In [4]:

```
df = pd.read_csv(r"C:\Users\TEMP USER\Downloads\blinkit_data.csv")
```

Sample Data

In [6]:

```
df.head(20)
```

Out[6]:

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

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outlet Size	Outlet Typ
17	Low Fat	NCB07	Household	2012	OUT049	Tier 1	Medium	Supermark Type
18	Low Fat	FDJ56	Fruits and Vegetables	1998	OUT027	Tier 3	Medium	Supermark Type
19	Low Fat	DRN47	Hard Drinks	2022	OUT018	Tier 3	Medium	Supermark Type

```
In [8]: df.tail(20)
```

Out[8]:

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outlet Size	Outlet T
8503	Regular	FDR22	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8504	Regular	FDS09	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8505	Regular	FDS34	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8506	Regular	FDU09	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8507	Regular	FDU33	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8508	Regular	FDU57	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8509	Regular	FDU58	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8510	Regular	FDX46	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8511	Regular	FDX57	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8512	Regular	FDY33	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8513	Regular	DRY23	Soft Drinks	1998	OUT027	Tier 3	Medium	Superma Ty
8514	low fat	FDA11	Baking Goods	1998	OUT027	Tier 3	Medium	Superma Ty
8515	low fat	FDK38	Canned	1998	OUT027	Tier 3	Medium	Superma Ty
8516	low fat	FDO38	Canned	1998	OUT027	Tier 3	Medium	Superma Ty
8517	low fat	FDG32	Fruits and Vegetables	1998	OUT027	Tier 3	Medium	Superma Ty
8518	low fat	NCT53	Health and Hygiene	1998	OUT027	Tier 3	Medium	Superma Ty
8519	low fat	FDN09	Snack Foods	1998	OUT027	Tier 3	Medium	Superma Ty
8520	low fat	DRE13	Soft Drinks	1998	OUT027	Tier 3	Medium	Superma Ty

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outlet Size	Outlet T
8521	reg	FDT50	Dairy	1998	OUT027	Tier 3	Medium	Superma T
8522	reg	FDM58	Snack Foods	1998	OUT027	Tier 3	Medium	Superma T

Size of Data

```
In [11]: print("size of Data:", df.shape)
```

```
size of Data: (8523, 12)
```

Field info

```
In [12]: df.columns
```

```
Out[12]: 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 Types

```
In [13]: df.dtypes
```

```
Out[13]: 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

```
In [14]: print(df['Item Fat Content'].unique())
```

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

```
In [15]: df['Item Fat Content'] = df['Item Fat Content'] .replace({'LF': 'Low Fat',
                                                                    'low fat': 'Low Fat',
                                                          'reg': 'Regular'
                                                         })
```

In [16]:
```
print(df['Item Fat Content'].unique())
```

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

BUSINESS REQUIREMENTS

KPI's REQUIREMENTS

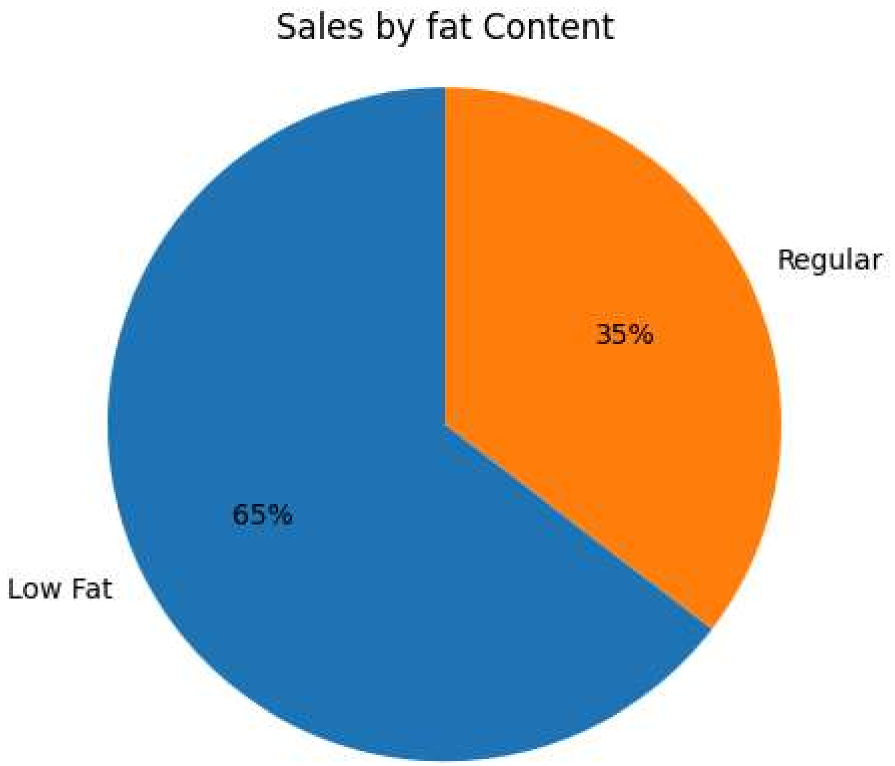
In [28]:
```
#Total Sales
total_sales = df['Sales'].sum()

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

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

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

#Display

print(f"Total sales: ${total_sales:,.0f}")
print(f"Average sales: ${avg_sales:,.1f}")
print(f"No_of_Items_sold: {no_of_items_sold:,.0f}")
print(f"Average Ratings: {avg_rating:,.1f}")
```

```
Total sales: $1,201,681
Average sales: $141.0
No_of_Items_sold: 8,523
Average Ratings: 4.0
```

CHARTS REQUIREMENTS

Total Sales by Fat Content:

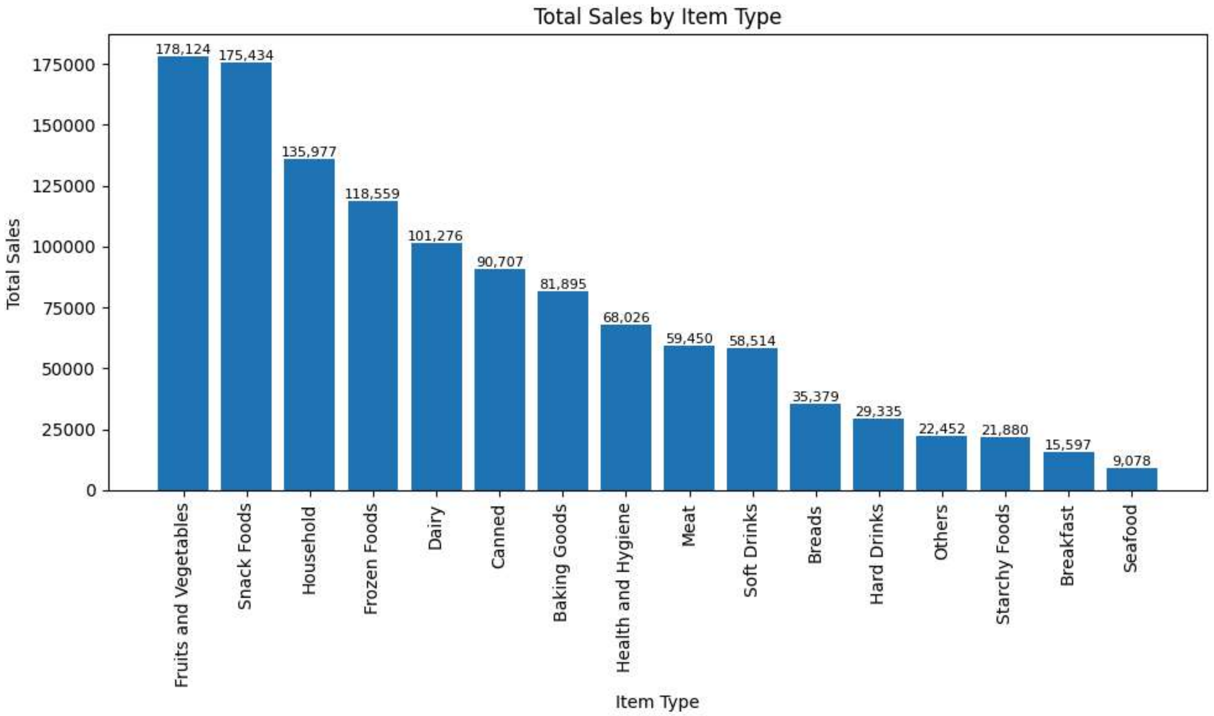
In [31]:
```
Sales_by_fat = df.groupby('Item Fat Content')['Sales'].sum()

plt.pie(Sales_by_fat, labels =Sales_by_fat.index,
                           autopct ='%.0f%%',
                           startangle = 90)

plt.title('Sales by fat Content')
plt.axis('equal')
plt.show()
```



Total Sales by Item Type

In [36]:

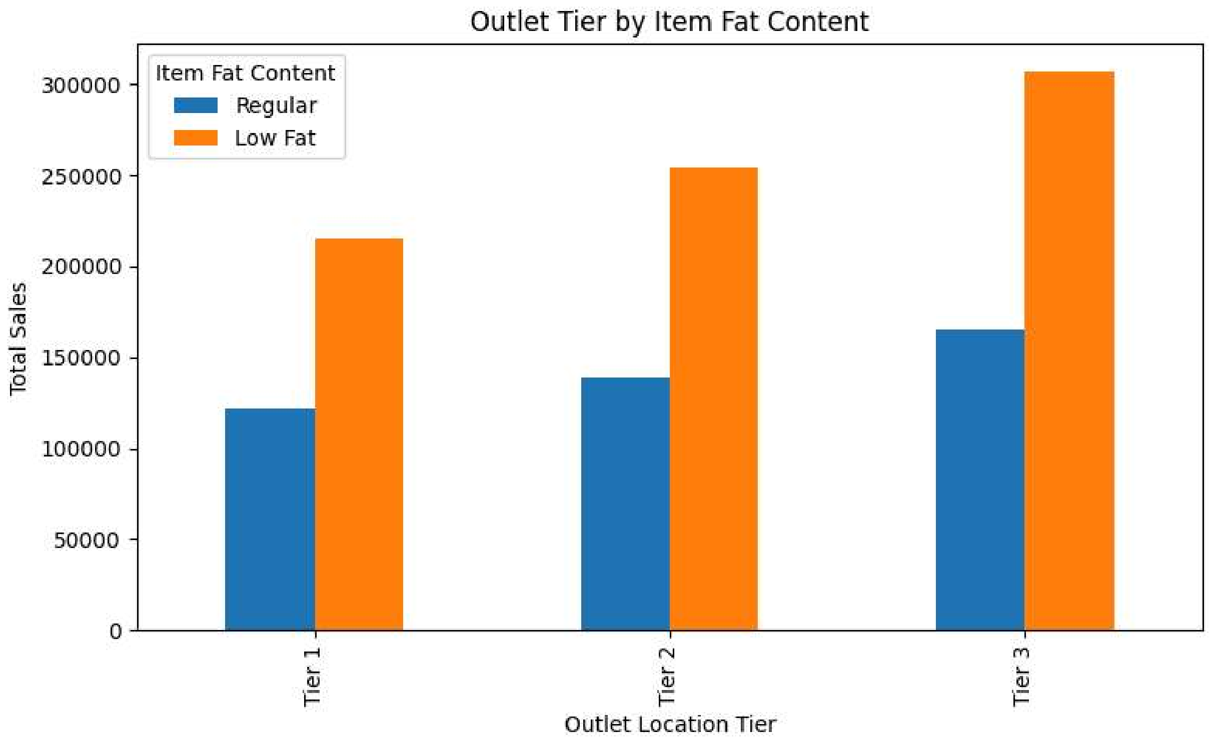
```
sales_by_type = df.groupby('Item Type')['Sales'].sum().sort_values(ascending=False)

plt.figure(figsize=(10, 6))  # Ensure reasonable size
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')

# Annotate bars
for bar in bars:
    plt.text(
        bar.get_x() + bar.get_width() / 2,
        bar.get_height(),
        f'{bar.get_height():,.0f}',  # CORRECTED f-string
        ha='center',
        va='bottom',
        fontsize=8
    )

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



Fat Content by Outlet for Total Sales

In [38]:

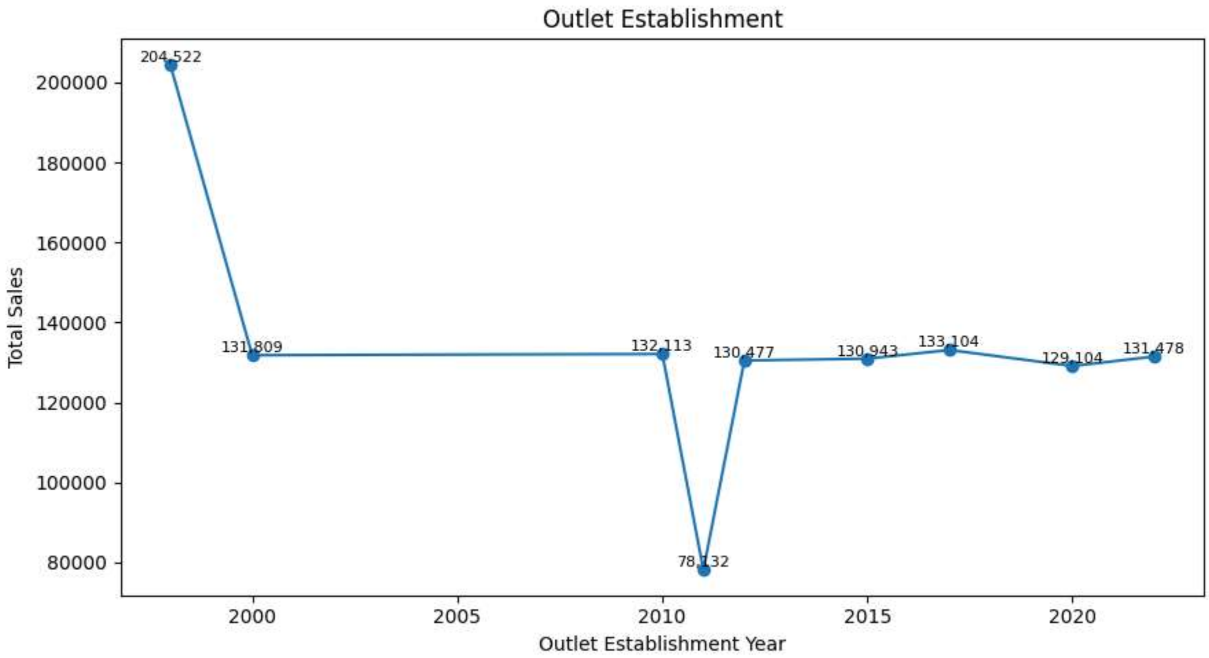
```
grouped = df.groupby(['Outlet Location Type', 'Item Fat Content'])['Sales'].sum().u
grouped = grouped[['Regular', 'Low Fat']]

ax = grouped.plot(kind='bar', figsize=(8, 5), title='Outlet Tier by Item Fat Conten
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

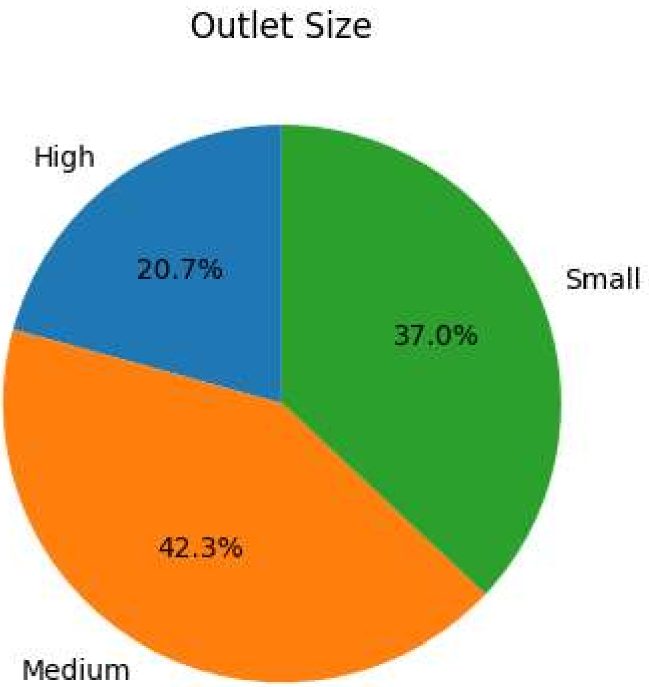
In [39]:

```python
sales_by_year = df.groupby('Outlet Establishment Year')['Sales'].sum().sort_index()

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

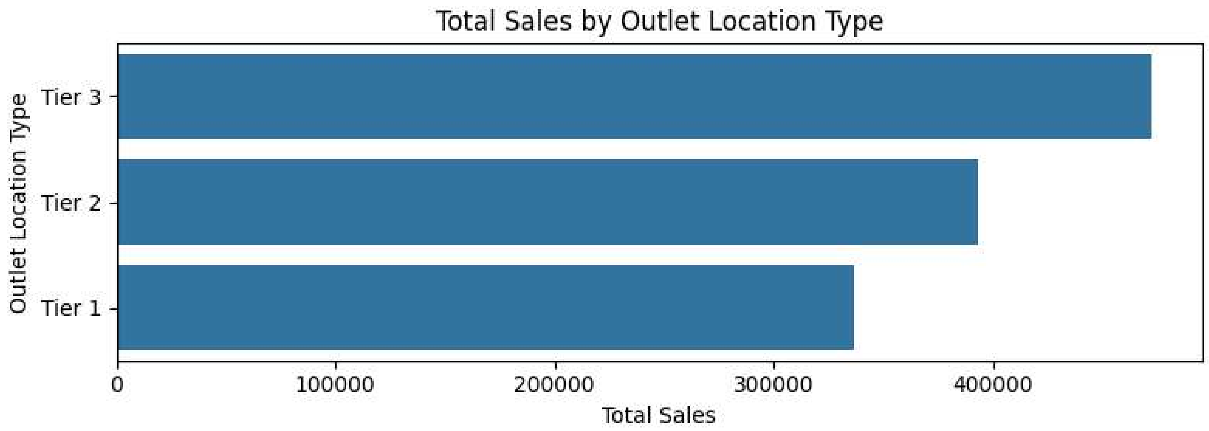
In [40]:
```
sales_by_size = df.groupby('Outlet Size')['Sales'].sum()

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 Outlet Location

In [41]:

```
sales_by_location = df.groupby('Outlet Location Type')['Sales'].sum().reset_index()
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_location)

plt.title('Total Sales by Outlet Location Type')
plt.xlabel('Total Sales')
plt.ylabel('Outlet Location Type')

plt.tight_layout()  # Ensure Layout fits without scroll
plt.show()
```



In [ ]: