

Blinkit Analysis Project

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

In [2]: df = pd.read_csv("C:/Users/ACER/Desktop/Project Analysis/Blinkit Analysis/Raw Data/BlinkIT_data.csv")

In [3]: df.head(10)

Out[3]:
```

	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
0	Regular	FDX32	Fruits and Vegetables	2012	OUT049	Tier 1	Medium	Supermarket Type1	0.100014	15.10	145.4786	5.0
1	Low Fat	NC842	Health and Hygiene	2022	OUT018	Tier 3	Medium	Supermarket Type2	0.008596	11.80	115.3492	5.0
2	Regular	FDR28	Frozen Foods	2016	OUT046	Tier 1	Small	Supermarket Type1	0.025896	13.85	165.0210	5.0
3	Regular	FDL50	Canned	2014	OUT013	Tier 3	High	Supermarket Type1	0.042278	12.15	126.5046	5.0
4	Low Fat	DRl25	Soft Drinks	2015	OUT045	Tier 2	Small	Supermarket Type1	0.033970	19.60	55.1614	5.0
5	low fat	FDS52	Frozen Foods	2020	OUT017	Tier 2	Small	Supermarket Type1	0.005505	8.89	102.4016	5.0
6	Low Fat	NCU05	Health and Hygiene	2011	OUT010	Tier 3	Small	Grocery Store	0.098312	11.80	81.4618	5.0
7	Low Fat	NC030	Household	2015	OUT045	Tier 2	Small	Supermarket Type1	0.026904	19.70	96.0726	5.0
8	Low Fat	FDW20	Fruits and Vegetables	2014	OUT013	Tier 3	High	Supermarket Type1	0.024129	20.75	124.1730	5.0
9	Low Fat	FDX25	Canned	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.101562	NaN	181.9292	5.0

```
In [4]: df.tail(10)

Out[4]:
```

	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
8513	Regular	DRY23	Soft Drinks	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.108568	NaN	42.9112	4.0
8514	low fat	FDA11	Baking Goods	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.043029	NaN	94.7436	4.0
8515	low fat	FDK38	Canned	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.053032	NaN	149.1734	4.0
8516	low fat	FDO38	Canned	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.072486	NaN	78.9986	4.0
8517	low fat	FDG32	Fruits and Vegetables	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.175143	NaN	222.3772	4.0
8518	low fat	NCT53	Health and Hygiene	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.000000	NaN	164.5526	4.0
8519	low fat	FDN09	Snack Foods	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.034706	NaN	241.6828	4.0
8520	low fat	DRE13	Soft Drinks	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.027571	NaN	86.6198	4.0
8521	reg	FDI50	Dairy	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.107715	NaN	97.8752	4.0
8522	reg	FDM58	Snack Foods	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.000000	NaN	112.2544	4.0

Size of Data

```
In [5]: print("Size of data:",df.shape)
Size of data: (8523, 12)

In [6]: df.dtypes

Out[6]:
```

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 [7]: print(df['Item Fat Content'].unique())
['Regular' 'Low Fat' 'low fat' 'LF' 'reg']

In [8]: df['Item Fat Content']=df['Item Fat Content'].replace(['low fat':'Low Fat','LF':'Low Fat','reg':'Regular'])

In [9]: print(df['Item Fat Content'].unique())
['Regular' 'Low Fat']
```

KPI-Business Requirements

```
In [10]: #Total Sales
Total_Sales = df['Sales'].sum()

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

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

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

In [11]: print(f"Total Sales: ${Total_Sales :,.1f}")

print(f"Average Sales: ${Average_Sales :,.1f}")

print("No of Items:",No_of_Items)

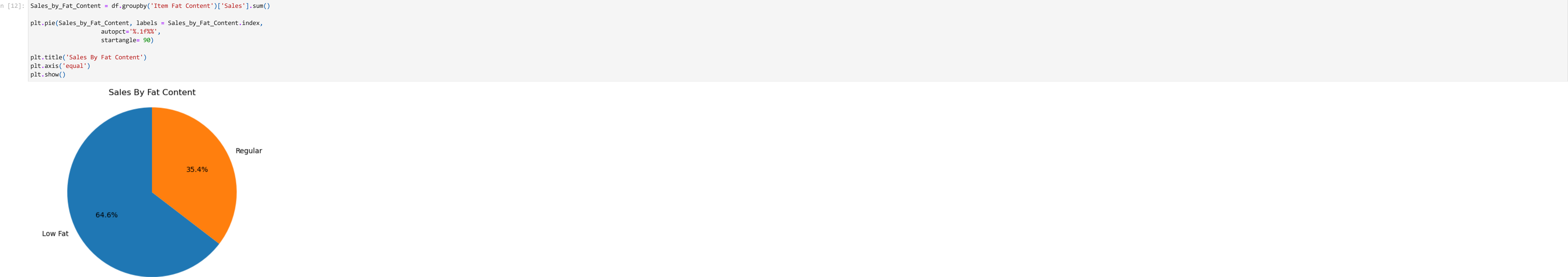
print("Average Rating:",Average_Rating)

#Rounded Rating
print(f"Average Rating: (Average_Rating :,.1f)")

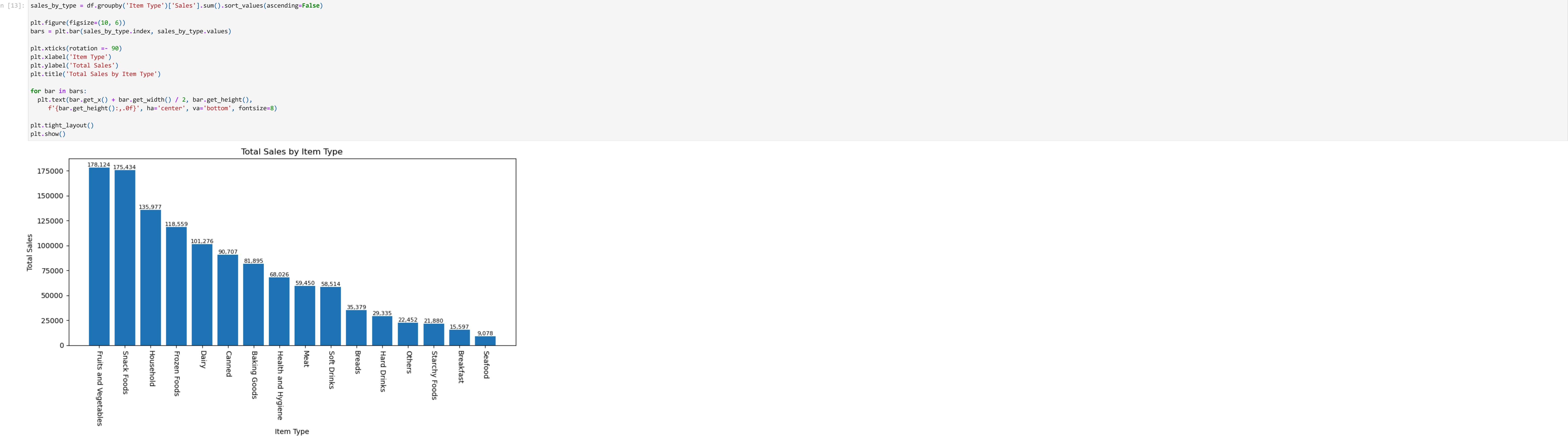
Total Sales: $1,201,681.5
Average Sales: $141.0
No of Items: 8523
Average Rating: 3.965857092573038
Average Rating: 4.0
```

Charts

Total Sales by Fat Content



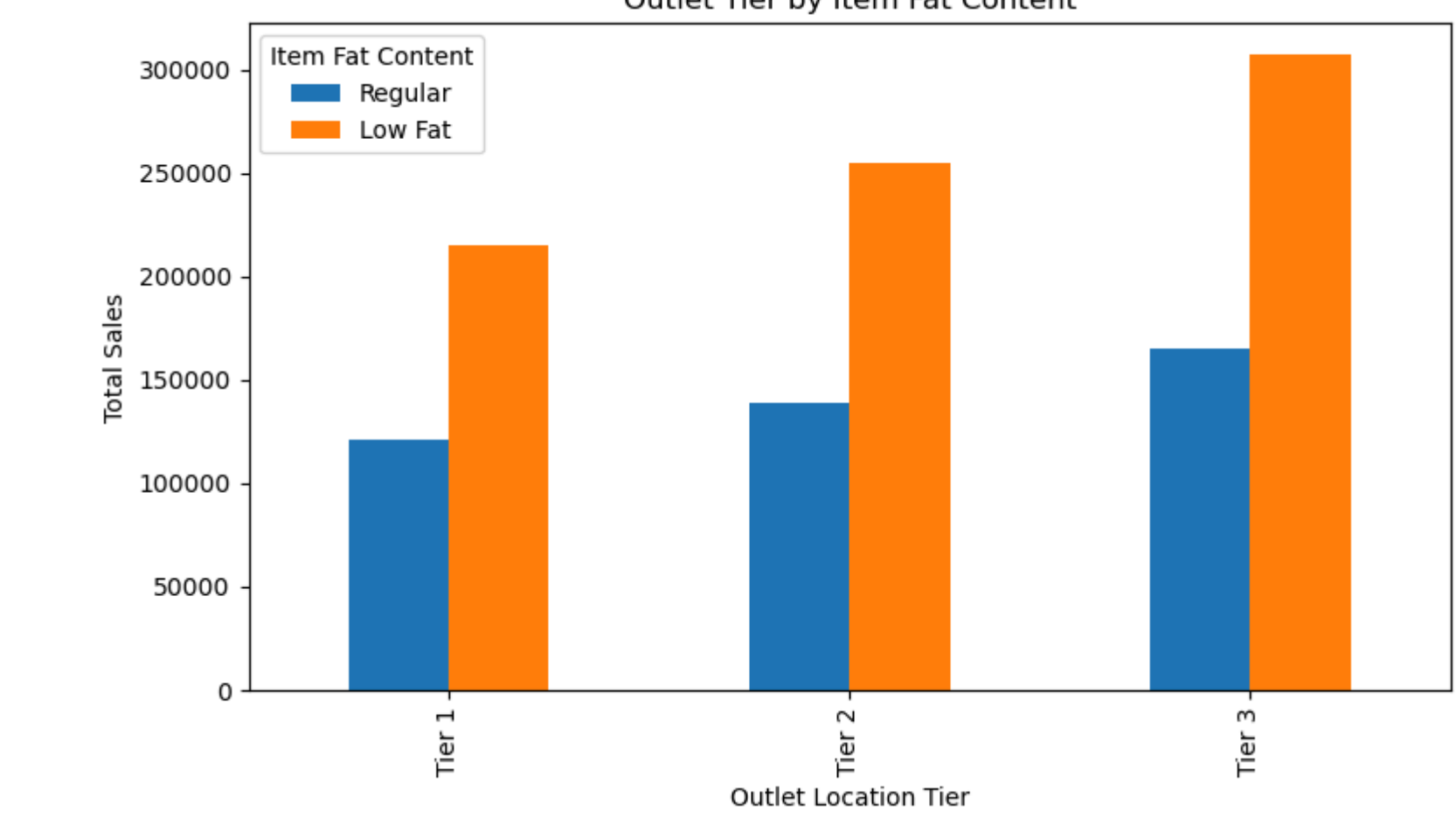
Sales by Item Type



Fat Content by Outlet for Sales

```
In [14]: grouped = df.groupby(['Outlet Location Type', 'Item Fat Content'])['Sales'].sum().unstack()
grouped = grouped[['Regular', 'Low Fat']]
ax = grouped.plot(kind='bar', figsize=(8, 5), title='Outlet Tier by Item Fat Content')
plt.xlabel('Outlet Location Tier')
plt.ylabel('Total Sales')
plt.legend(title='Item Fat Content')
plt.tight_layout()
plt.show()
```

Outlet Tier by Item Fat Content



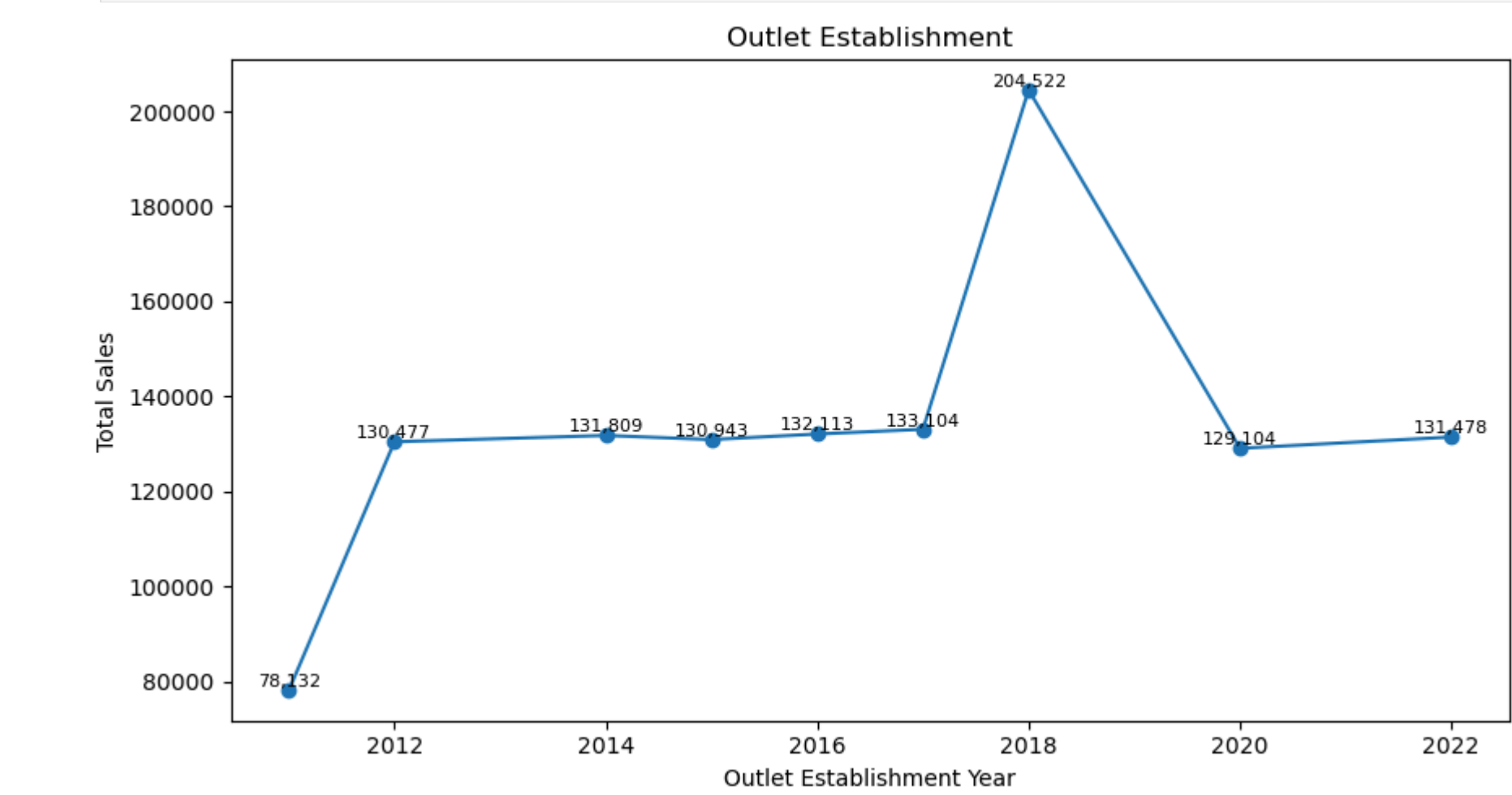
Sales by Outlet Establishment Year

```
In [15]: 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:,8f}', ha='center', va='bottom', fontsize=8)

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

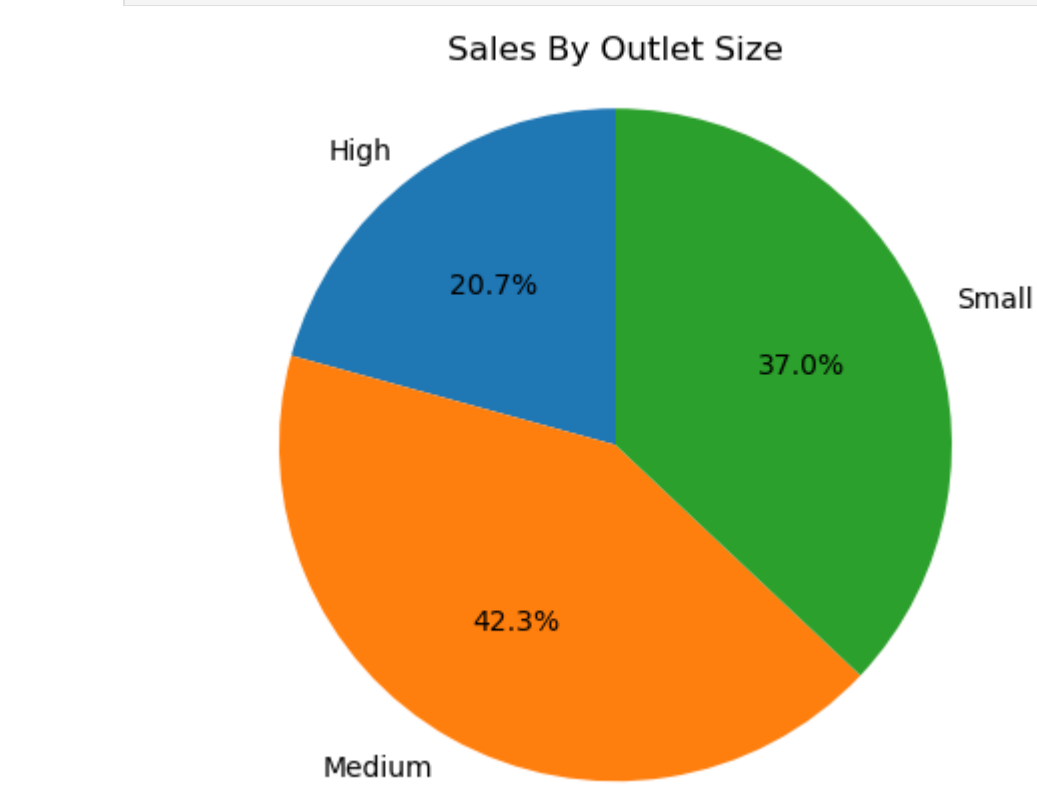


Sales by Outlet Size

```
In [16]: Sales_by_Size = df.groupby('Outlet Size')['Sales'].sum()

plt.pie(Sales_by_Size, labels = Sales_by_Size.index,
        autopct='%1.1f%%',
        startangle= 90)

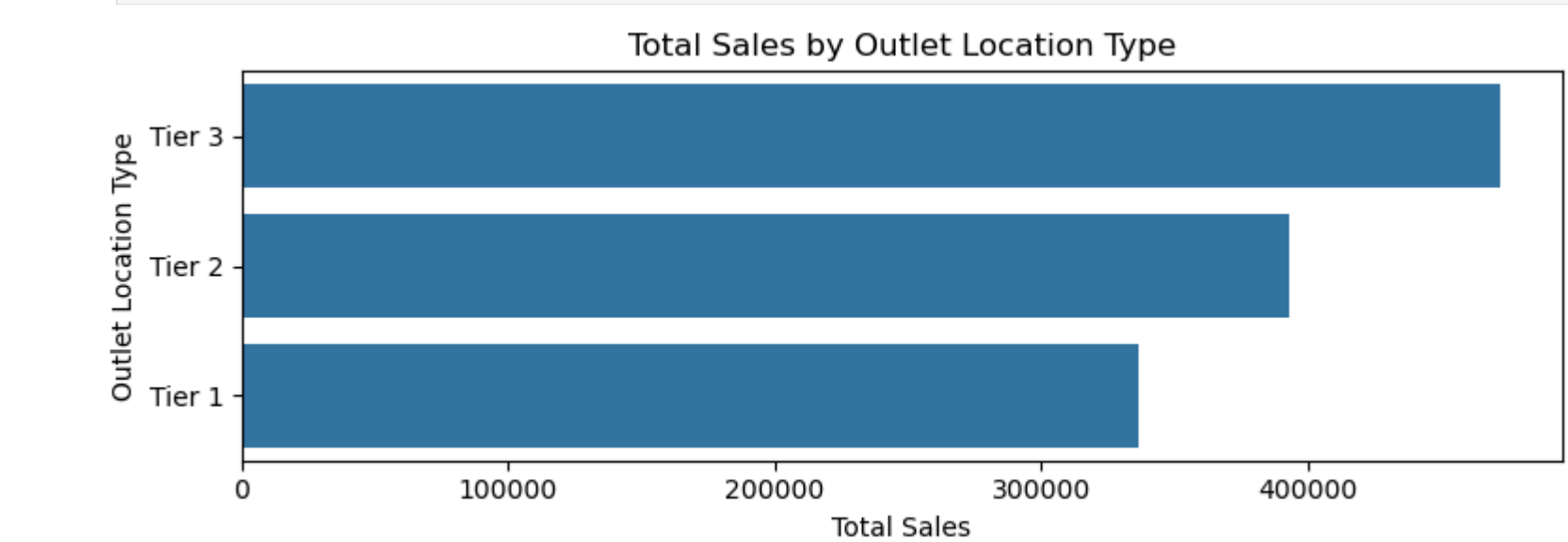
plt.title('Sales By Outlet Size')
plt.axis('equal')
plt.show()
```



Sales by Outlet Location

```
In [17]: 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() # Ensures layout fits without scroll
plt.show()
```



In []: