Data Analysis using Python- 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 [2]:

```
df = pd.read_csv(r"C:\Users\LALITHRAJ R\Downloads\BlinkIT Grocery Data.csv")
```

Sample Data

In [3]:

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

Out[3]:

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outlet Size	Outlet
0	Regular	FDX32	Fruits and Vegetables	2012	OUT049	Tier 1	Medium	Superm
1	Low Fat	NCB42	Health and Hygiene	2022	OUT018	Tier 3	Medium	Superm
2	Regular	FDR28	Frozen Foods	2016	OUT046	Tier 1	Small	Superm
3	Regular	FDL50	Canned	2014	OUT013	Tier 3	High	Superm
4	Low Fat	DRI25	Soft Drinks	2015	OUT045	Tier 2	Small	Superm
5	low fat	FDS52	Frozen Foods	2020	OUT017	Tier 2	Small	Superm
6	Low Fat	NCU05	Health and Hygiene	2011	OUT010	Tier 3	Small	Gr
7	Low Fat	NCD30	Household	2015	OUT045	Tier 2	Small	Superm
8	Low Fat	FDW20	Fruits and Vegetables	2014	OUT013	Tier 3	High	Superm
9	Low Fat	FDX25	Canned	2018	OUT027	Tier 3	Medium	Superm
10	LF	FDX21	Snack Foods	2018	OUT027	Tier 3	Medium	Superm
11	Low Fat	NCU41	Health and Hygiene	2017	OUT035	Tier 2	Small	Superm
12	Low Fat	FDL20	Fruits and Vegetables	2022	OUT018	Tier 3	Medium	Superm
13	Low Fat	NCR54	Household	2014	OUT013	Tier 3	High	Superm
14	Low Fat	FDH19	Meat	2018	OUT027	Tier 3	Medium	Superm
15	Regular	FDB57	Fruits and Vegetables	2017	OUT035	Tier 2	Small	Superm
16	Low Fat	FDO23	Breads	2022	OUT018	Tier 3	Medium	Superm
17	Low Fat	NCB07	Household	2012	OUT049	Tier 1	Medium	Superm
18	Low Fat	FDJ56	Fruits and Vegetables	2018	OUT027	Tier 3	Medium	Superm

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outlet Size	Outlet
19	Low Fat	DRN47	Hard Drinks	2022	OUT018	Tier 3	Medium	Superm

```
In [4]: df.tail(15)
```

Out[4]:

	Item Fat Content	Item Identifier	Item Type	Outlet Establishment Year	Outlet Identifier	Outlet Location Type	Outlet Size	Out
8508	Regular	FDU57	Snack Foods	2018	OUT027	Tier 3	Medium	Supe
8509	Regular	FDU58	Snack Foods	2018	OUT027	Tier 3	Medium	Supe
8510	Regular	FDX46	Snack Foods	2018	OUT027	Tier 3	Medium	Supe
8511	Regular	FDX57	Snack Foods	2018	OUT027	Tier 3	Medium	Supe
8512	Regular	FDY33	Snack Foods	2018	OUT027	Tier 3	Medium	Supe
8513	Regular	DRY23	Soft Drinks	2018	OUT027	Tier 3	Medium	Supe
8514	low fat	FDA11	Baking Goods	2018	OUT027	Tier 3	Medium	Supe
8515	low fat	FDK38	Canned	2018	OUT027	Tier 3	Medium	Supe
8516	low fat	FDO38	Canned	2018	OUT027	Tier 3	Medium	Supe
8517	low fat	FDG32	Fruits and Vegetables	2018	OUT027	Tier 3	Medium	Supe
8518	low fat	NCT53	Health and Hygiene	2018	OUT027	Tier 3	Medium	Supe
8519	low fat	FDN09	Snack Foods	2018	OUT027	Tier 3	Medium	Supe
8520	low fat	DRE13	Soft Drinks	2018	OUT027	Tier 3	Medium	Supe
8521	reg	FDT50	Dairy	2018	OUT027	Tier 3	Medium	Supe
8522	reg	FDM58	Snack Foods	2018	OUT027	Tier 3	Medium	Supe

Size of Data use numpy shape

```
In [5]: df.shape
```

```
Out[5]: (8523, 12)
```

```
In [6]: print("Size of data:",df.shape)
```

```
Size of data: (8523, 12)
```

Field info

```
In [7]: df.columns
```

```
Out[7]: 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 [8]: df.dtypes
```

```
Out[8]: 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
```

Printing unique values in the column Item Fat Content for data cleaning

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

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

Data Cleaning

```
In [10]: df['Item Fat Content']=df['Item Fat Content'].replace({'LF':'Low Fat',
                                                               'low fat': 'Low Fat',
                                                               'reg':'Regular'})
```

```
In [11]: print(df['Item Fat Content'].unique())
```

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

Business Requirements

KPI Requirements

In [12]:

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

##Avg Sales mean means avg func
avg_sales=df['Sales'].mean()

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

#Avg rating
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 Rating: {avg_rating:,.0f}")
```

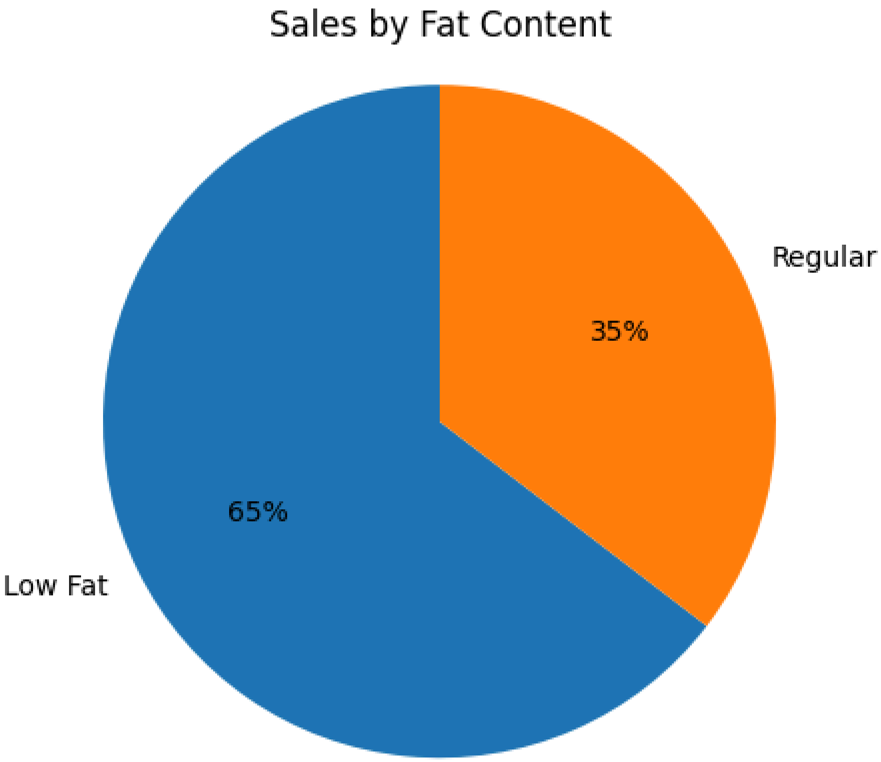
```
Total Sales: $1,201,681
Average Sales: $141.0
No of Items sold: 8,523
Average Rating: 4
```

Charts Requirement

Total sales by fat content

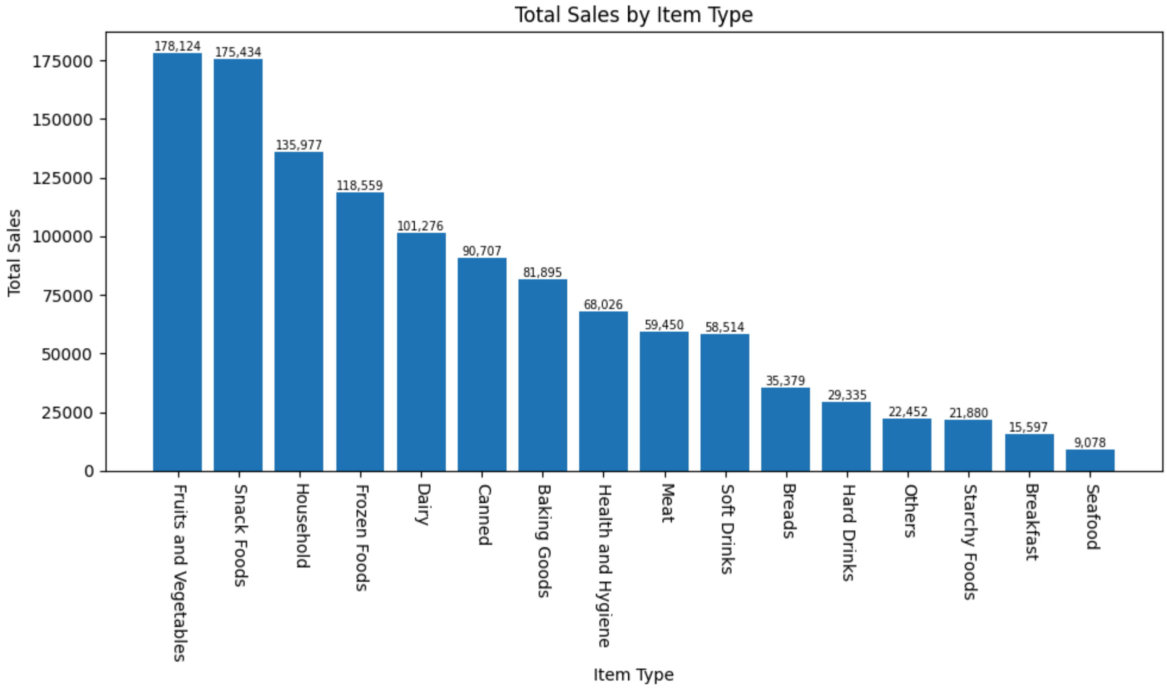
In [13]:

```
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

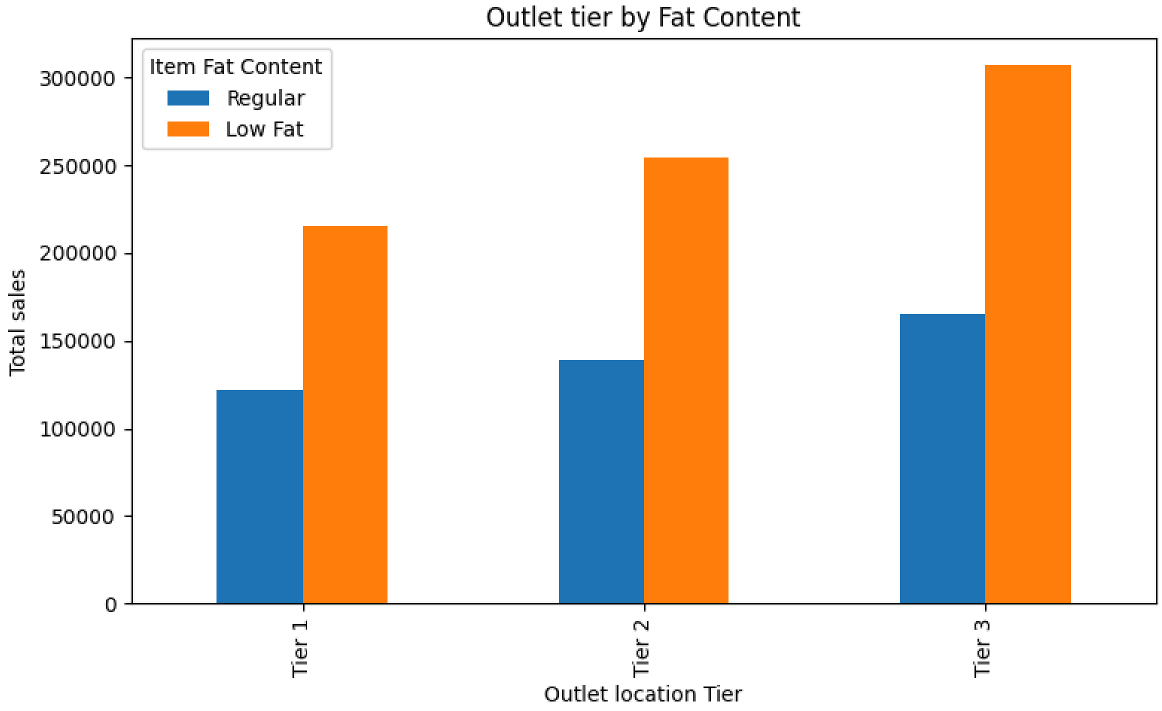
```
In [14]: sales_by_type = df.groupby('Item Type')['Sales'].sum().sort_values(ascending = F
         plt.figure(figsize=(10,6))
         bars=plt.bar(sales_by_type.index,sales_by_type.values)
         plt.xticks(rotation=-90)
         plt.xlabel("Item Type")
         plt.ylabel("Total Sales")
         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=7
                     )
         plt.tight_layout()
         plt.show()
```



Fat Content by outlet for total sales

In [15]:

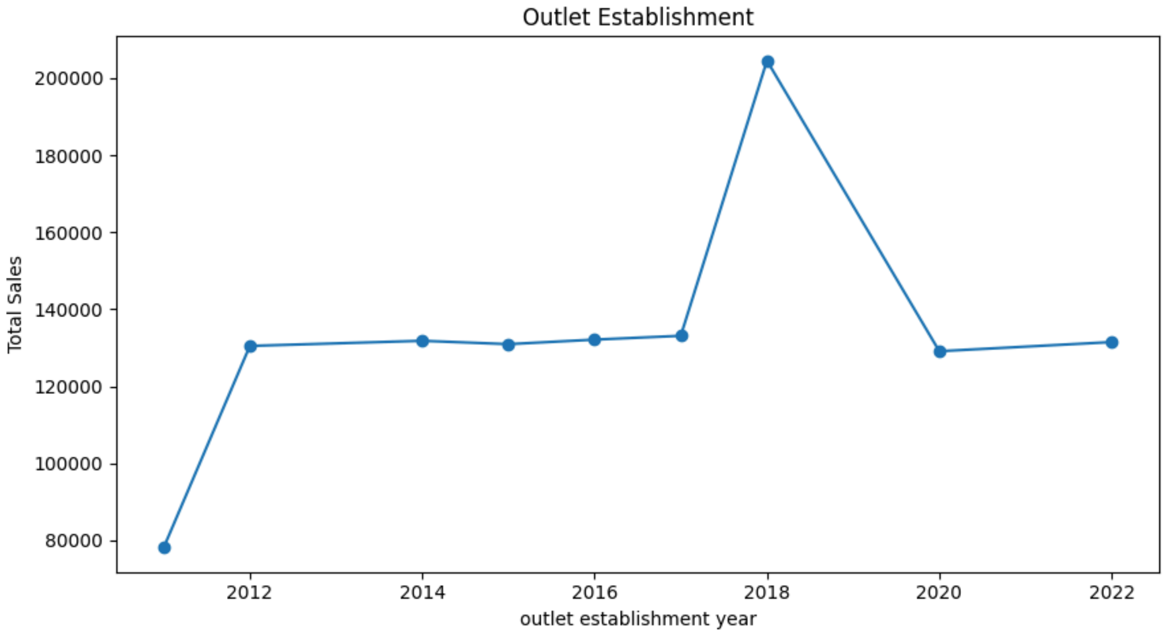
```
grouped=df.groupby(['Outlet Location Type','Item Fat Content'])['Sales'].sum().u
grouped = grouped[['Regular','Low Fat']]
ax=grouped.plot(kind='bar',figsize=(8,5),title="Outlet tier by Fat Content")
plt.xlabel('Outlet location Tier')
plt.ylabel('Total sales')
plt.legend(title='Item Fat Content')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



Total Sales by outlet establishment

In [16]:

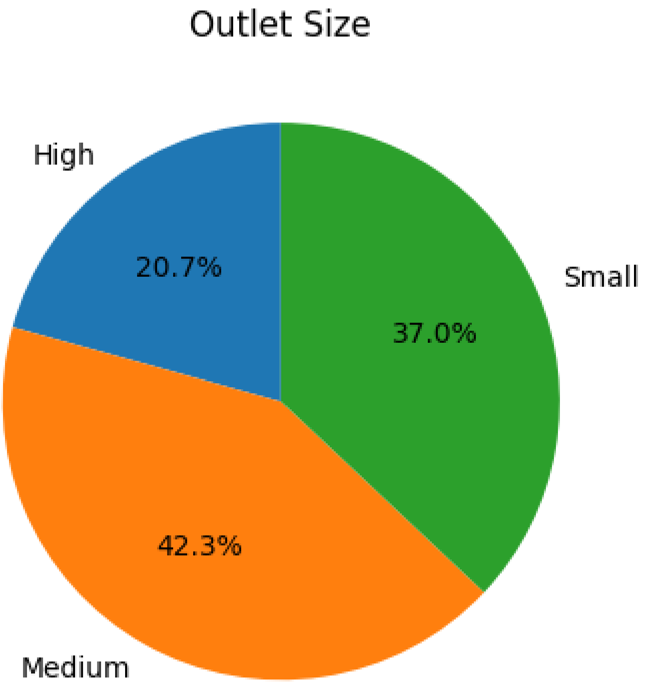
```
sales_by_year = df.groupby('Outlet Establishment Year')['Sales'].sum().sort_inde
plt.figure(figsize=(9,5))
plt.plot(sales_by_year.index,sales_by_year.values,marker='o',linestyle='-')
plt.xlabel('outlet establishment year')
plt.ylabel('Total Sales')
plt.title("Outlet Establishment")
plt.tight_layout()
plt.show()
```



Total Sales by outlet size

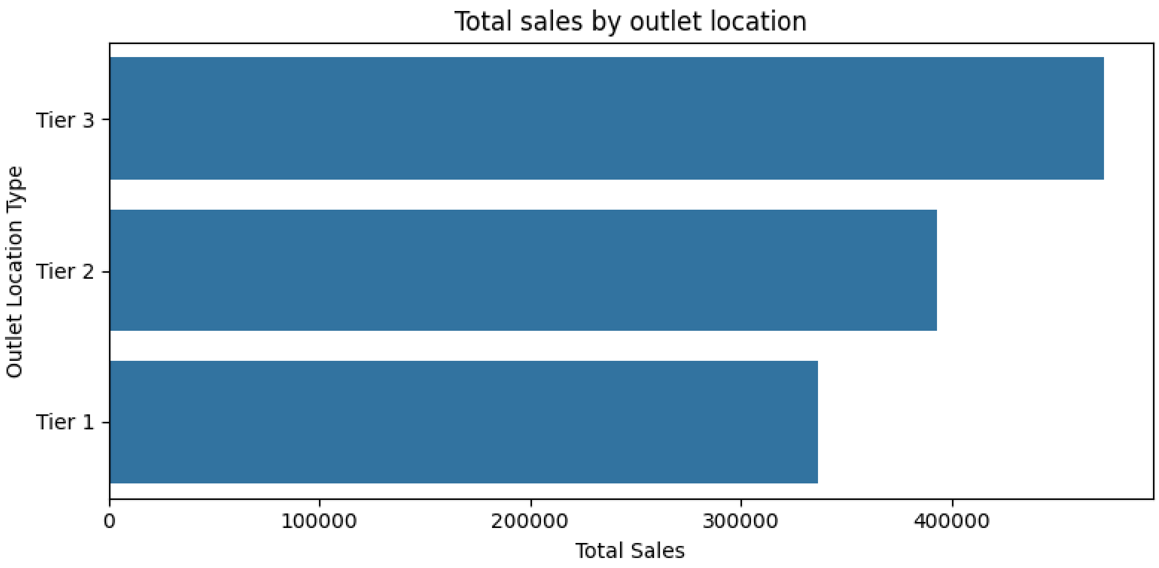
In [17]:

```
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='%.1f%%',startangle=90)
plt.title('Outlet Size')
plt.tight_layout()
plt.show()
```



Total sales by outlet location

```
In [18]: sales_by_loc = df.groupby("Outlet Location Type")['Sales'].sum().reset_index()
         sales_by_loc = sales_by_loc.sort_values('Sales',ascending=False)
         plt.figure(figsize=(8,4)) #smaller height,enough width
         ax = sns.barplot(x='Sales',y='Outlet Location Type',data=sales_by_loc)
         plt.title('Total sales by outlet location')
         plt.xlabel("Total Sales")
         plt.ylabel("Outlet Location Type")
         plt.tight_layout() #Ensures Layout fits without scroll
         plt.show()
```


Total sales by outlet location
Tier 3
Tier 2
Tier 1
Outlet Location Type
Total Sales