DATA ANALYSIS PYTHON PROJET - 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("C:/Users/EMO/Desktop/Blinkit_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 Type	Item Visibility	Item Weight	Sales	Rating
0	Regular	FDX23	Fruits and Vegetables	2012	OUT049	Tier 1	Medium	Supermarket Type1	0.100014	15.10	145.4786	5.0
1	Low Fat	NCB42	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	DRZ5	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	0.005935	8.89	102.4016	5.0
6	Low Fat	FDA41	Health and Hygiene	2011	OUT010	Tier 3	Small	Grocery Store	0.098312	11.60	81.4618	5.0
7	Low Fat	NCO30	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
10	LF	FDX21	Snack Foods	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.084555	NaN	109.8912	5.0
11	Low Fat	NCU41	Health and Hygiene	2017	OUT026	Tier 2	Small	Supermarket Type1	0.052045	16.85	192.1846	5.0
12	Low Fat	FDL29	Fruits and Vegetables	2022	OUT018	Tier 3	Medium	Supermarket Type2	0.126938	17.10	112.3866	5.0
13	Low Fat	NCR56	Household	2014	OUT013	Tier 3	High	Supermarket Type1	0.090487	16.35	195.2110	5.0
14	Low Fat	FDH19	Meat	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.032928	NaN	173.1738	5.0
15	Regular	FDB57	Fruits and Vegetables	2017	OUT035	Tier 2	Small	Supermarket Type1	0.018802	20.25	222.1772	5.0
16	Low Fat	FDO23	Breads	2022	OUT018	Tier 3	Medium	Supermarket Type2	0.147024	17.85	93.7436	5.0
17	Low Fat	NCB07	Household	2012	OUT049	Tier 1	Medium	Supermarket Type1	0.077626	19.20	187.6110	5.0
18	Low Fat	FDJ58	Fruits and Vegetables	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.182515	NaN	96.7700	5.0
19	Low Fat	DRN47	Hard Drinks	2022	OUT018	Tier 3	Medium	Supermarket Type2	0.016895	12.10	178.5660	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	0.073496	NaN	78.9966	4.0
8517	low fat	FDO22	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	164.8526	4.0
8519	low fat	FDN09	Snack Foods	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.034706	NaN	241.6826	4.0
8520	low fat	DRE13	Soft Drinks	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.027571	NaN	86.6198	4.0
8521	reg	FDT50	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)
```

Field information

In [6]:
```
df.columns
```

Out[6]:
```
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 [7]:
```
df.dtypes
```

Out[7]:
```
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 [8]:
```
print(df['Item Fat Content'].unique())
```

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

In [9]:
```
df['Item Fat Content'] = df['Item Fat Content'].replace({'LF':'Low Fat','low fat':'Low Fat','reg':'Regular'})
```

In [10]:
```
print(df['Item Fat Content'].unique())
```

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

BUSINESS REQUIREMENT

KPI's Requirement

In [11]:
```
# 1.  Total Sales : The overall revenue generated from all items sold.
total_sales = df['Sales'].sum()

# 2.  Average Sales : The average revenue per sale.
avg_sales = df['Sales'].mean()

# 3.  Number of Items : The total count of different items sold.
no_of_items_sold = df['Sales'].count()

# 4.  Average Rating : The average customer rating for item sold.
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
NO. Of Items Sold: 8,523
Average Ratings: 4.0
```

CHARTS Requirement

In [12]:
```
# 1. Total Sales by Fat Content :
#   Objective: Analyze the impact of fat content on total sales.
#   Additional KPI Metrics: Assess how other KPIs (Average Sales, Number of items, Average Ratings) vary with 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')
plt.show()
```

Total Sales by Item Type

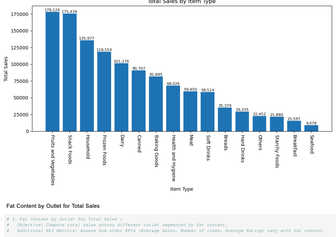
In [13]:
```
# 2. Total Sales by Item Type :
#   Objective: Identify the performance of different item types in terms of total sales.
#   Additional KPI Metrics: Assess how other KPIs (Average Sales, Number of items, Average Ratings) vary with fat content.

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

plt.figure(figsize=(11,6))
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:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(),
             #f'{bar.get_height():,.0f}', ha='center',va='bottom',fontsize=8)
    width, height =bar.get_width(), bar.get_height()
    x,y = bar.get_xy()
    plt.text(x + width / 2,y + height,f'{bar.get_height():,.0f}',ha='center',va= 'bottom',fontsize= 8)
plt.tight_layout()
plt.show()
```

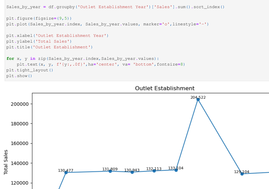


Fat Content by Outlet for Total Sales

In [14]:
```
# 3. Fat Content by Outlet for Total Sales :
#   Objective: Compare total sales across different outlet segmented by fat content.
#   Additional KPI Metrics: Assess how other KPIs (Average Sales, Number of items, Average Ratings) vary with fat content.

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')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.xlabel('Outlet Location Tier')
plt.ylabel('Total Sales')
plt.legend(title='Item Fat Content')
plt.tight_layout()
plt.show()
```



Total Sales Outlet Establishment

In [15]:
```
# 4.  Total Sales by Outlet Establishment :
# Objective: Evaluate how the age or type of outlet establishment influences total sales.


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 [16]:
```
# 5. Percentage of Sales by Outlet Size :
#   Objective: Analyze the correlation between outlet size and total sales.

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

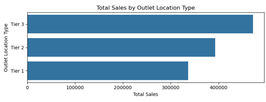
In [17]:
```
# 6.  Sales by Outlet Location :
#   Objective: Assess the geographic distribution of sales across different locations.


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=(9,5))  #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()
```


Total Sales by Outlet Location Type
Outlet Location Type
Tier 3
Tier 2
Tier 1
0
100000
200000
300000
400000
Total Sales