Importing Libraries

In [ ]:
```
#upper heading comes from (### **Heading**) and taskbar me code ki markdown then run
```

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

In [ ]:
```
df=pd.read_csv("Northern Super Market Data CSV.csv")
df
```

Out[ ]:

	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	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	DRZ25	Soft Drinks	2015	OUT045	Tier 2	Small	Supermarket Type1	0.033970	19.60	55.1614	5.0
...	...	...	...	...	...	...	...	...	...	...	...	...
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	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

8523 rows × 12 columns

In [ ]:
```
df.head(15)
```

Out[ ]:

	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	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	DRZ25	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	NCL05	Health and Hygiene	2011	OUT010	Tier 3	Small	Grocery Store	0.098312	11.80	81.4618	5.0
7	Low Fat	NCD30	Household	2015	OUT045	Tier 2	Small	Supermarket Type1	0.028904	19.70	96.0726	5.0
8	Low Fat	FDW20	Fruits and Vegetables	2014	OUT013	Tier 3	High	Supermarket Type1	0.024129	20.75	124.1750	5.0
9	Low Fat	FDX25	Canned	2018	OUT027	Tier 3	Medium	Supermarket Type3	0.151562	NaN	181.8282	5.0

In [ ]:
```
df.tail() #5 by-default, 8522 is index number
```

Out[ ]:

	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
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	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

In [ ]:
```
df.shape #no. of rows & columns
```

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

In [ ]:
```
df.columns #all field names
```

Out[ ]:
```
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')
```

In [ ]:
```
df.dtypes #datatypes in python
```

Out[ ]:
```
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
```

In [ ]:
```
print(df['Item Fat Content'].unique()) #to check unique element in column. we are using it to clean the data
```

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

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

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

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

KPI Requirements

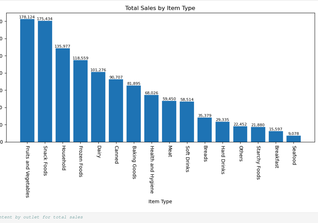
In [ ]:
```
#Total sales, average sales, No. of items sold, average ratings
Total_salesdf=df['Sales'].sum()
Average_salesdf=df['Sales'].mean()
No_of_item_soldsdf=df['Sales'].count()
Average_ratingsdf=df['Rating'].mean()

print("Total_sales = ", Total_sales)
print("Average_sales = ", Average_sales)
print("No_of_item_sold = ", No_of_item_sold)
print("Average_ratings = ", Average_ratings)

Total_sales = 1201681.1928
Average_sales =  140.9927839613163
No_of_item_sold =  8523
Average_ratings =  3.965857092573038
```

In [ ]:
```
#To round-off above expressions----------
print(f"Total Sales: ${Total_sales:,.0f}")
print(f"Total Sales: ${Total_sales:,.2f}")
print(f"Average Sales: ${Average_sales:,.2f}")
print(f"Average Sales: ${Average_sales:,.0f}")
print(f"No. of item sold: {No_of_item_sold:,}")
print(f"Average ratings: {Average_ratings:,.1f}")

Total Sales: $1,201,681.5
Total Sales: $1,201,681.19
Average Sales: $141.0
Average Sales: $141
No. of item sold: 8523
Average ratings: 4.0
```

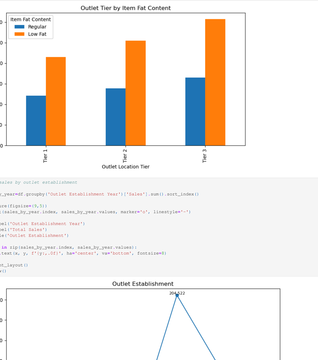
Chart's Requirements

In [ ]:
```
#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')
plt.show()
```

In [ ]:
```
#Total sales by item

sales_by_type=df.groupby('Item Type')['Sales'].sum().sort_values(ascending=False)
plt.figure(figsize=(15,6)) #if,f=horizontal,Vertical of canvas
bars=plt.bar(sales_by_type.index, sales_by_type.values)

plt.xticks(rotation=90) #90 is item type name
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) #for sale value in bars & size of bar

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



In [ ]:
```
#Fat content by outlet for total sales

grouped=df.groupby(['Outlet Location Type','Item Fat Content'])['Sales'].sum().unstack() #unstack means not overlapping
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()
```



In [ ]:
```
#Total sales by outlet establishment

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

plt.figure(figsize=(7,5))
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:,.2f}', ha='center', va='bottom', fontsize=8)

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

In [ ]:
```
#Sales by outlet size

sales_by_size=df.groupby('Outlet Size')['Sales'].sum()

plt.figure(figsize=(6,6)) #it is for size, we can remove it also.
plt.pie(sales_by_size, labels=sales_by_size.index, autopct='%1.1f%%', startangle=90)
plt.title('Sales by Outlet Size')
plt.tight_layout ()
plt.show()
```


In [ ]:
```
#Sales by outlet location type

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)) #width,height
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 () #layout fits without scroll
plt.show()
```


In [ ]:
```
```