In [1]:

```
import pandas as pd
df=pd.read_csv("Blinkit Grocery Data.csv",sep=",")
df
```

Out[1]:

	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	DRI25	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	241.6928	4.0
8520	low fat	DRE15	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 [2]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:

```
df.shape
```

Out[4]:

```
(8523, 12)
```

In [6]:

```
df.head()
```

Out[6]:

	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	DRI25	Soft Drinks	2015	OUT045	Tier 2	Small	Supermarket Type1	0.033970	19.60	55.1614	5.0

In [7]:

```
df.columns.values
```

Out[7]:

```
array(['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 [11]:

```
df.info()
#missing values in item weight
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item Fat Content           8523 non-null   object
 1   Item Identifier            8523 non-null   object
 2   Item Type                  8523 non-null   object
 3   Outlet Establishment Year  8523 non-null   int64
 4   Outlet Identifier          8523 non-null   object
 5   Outlet Location Type       8523 non-null   object
 6   Outlet Size                8523 non-null   object
 7   Outlet Type                8523 non-null   object
 8   Item Visibility            8523 non-null   float64
 9   Item Weight                7060 non-null   float64
 10  Sales                      8523 non-null   float64
 11  Rating                     8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

In [12]:

```
df.isnull().sum()
```

Out[12]:

```
Item Fat Content                0
Item Identifier                 0
Item Type                       0
Outlet Establishment Year       0
Outlet Identifier               0
Outlet Location Type            0
Outlet Size                     0
Outlet Type                     0
Item Visibility                 0
Item Weight                  1463
Sales                           0
Rating                          0
dtype: int64
```

In [18]:

```
df['Item Weight'].fillna(df['Item Weight'].mean),inplace=True)
#replace the missing value by filling method with help of mean
```

In [20]:

```
df.isnull().sum()
```

Out[20]:

```
Item Fat Content                0
Item Identifier                 0
Item Type                       0
Outlet Establishment Year       0
Outlet Identifier               0
Outlet Location Type            0
Outlet Size                     0
Outlet Type                     0
Item Visibility                 0
Item Weight                     0
Sales                           0
Rating                          0
dtype: int64
```

In [21]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item Fat Content           8523 non-null   object
 1   Item Identifier            8523 non-null   object
 2   Item Type                  8523 non-null   object
 3   Outlet Establishment Year  8523 non-null   int64
 4   Outlet Identifier          8523 non-null   object
 5   Outlet Location Type       8523 non-null   object
 6   Outlet Size                8523 non-null   object
 7   Outlet Type                8523 non-null   object
 8   Item Visibility            8523 non-null   float64
 9   Item Weight                8523 non-null   float64
 10  Sales                      8523 non-null   float64
 11  Rating                     8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

In [22]:

```
df.describe()
#provides a snapshot of the dataset
```

Out[22]:

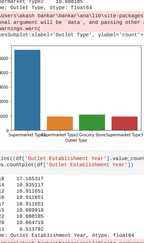
	Outlet Establishment Year	Item Visibility	Item Weight	Sales	Rating
count	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	2014.450546	0.066132	12.857645	140.992783	3.965857
std	3.189396	0.051598	4.226124	62.275067	0.605651
min	2011.000000	0.000000	4.555000	31.290000	1.000000
25%	2014.000000	0.026989	9.310000	93.826500	4.000000
50%	2016.000000	0.053931	12.857645	143.012800	4.000000
75%	2018.000000	0.094565	16.000000	185.643700	4.200000
max	2022.000000	0.328391	21.350000	266.888400	5.000000

In [29]:

```
print(df['Outlet Type'].value_counts()/8523)*100)
sns.countplot(df['Outlet Type'])
```

```
Supermarket Type1    65.434706
Grocery Store        12.708793
Supermarket Type3    10.979316
Supermarket Type2    10.877153
Name: Outlet Type, dtype: float64
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='Outlet Type', ylabel='count'>
```

Out[29]:

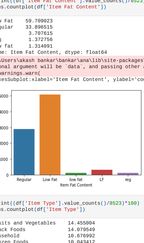


In [30]:

```
print((df['Outlet Establishment Year'].value_counts()/8523)*100)
sns.countplot(df['Outlet Establishment Year'])
```

```
2018    17.165317
2014    10.935117
2012    10.911651
2013    10.911651
2017    10.911651
2015    10.899918
2022    10.888185
2020    10.864719
2011     6.511792
Name: Outlet Establishment Year, dtype: float64
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='Outlet Establishment Year', ylabel='count'>
```

Out[30]:



In [35]:

```
df.head()
```

Out[35]:

	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	DRI25	Soft Drinks	2015	OUT045	Tier 2	Small	Supermarket Type1	0.033970	19.60	55.1614	5.0

In [36]:

```
print((df['Item Fat Content'].value_counts()/8523)*100)
sns.countplot(df['Item Fat Content'])
```

```
Low Fat    59.709023
Regular    33.896515
LF          3.707015
reg         1.172791
low fat     1.762753
Name: Item Fat Content, dtype: float64
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='Item Fat Content', ylabel='count'>
```

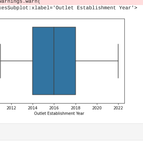
Out[36]:

In [42]:

```
print((df['Item Type'].value_counts()/8523)*100)
sns.countplot(df['Item Type'])
```

```
Fruits and Vegetables    14.456904
Snack Foods              14.078549
Household                10.676992
Frozen Foods             10.496772
Dairy                     8.601877
Canned                    7.614690
Baking Goods              7.602957
Health and Hygiene        6.101138
Soft Drinks               5.221166
Meat                      4.906507
Breads                    2.944972
Hard Drinks               2.510853
Others                    1.902870
Starchy Foods             1.736476
Breakfast                 1.055497
Seafood                   0.750899
Name: Item Type, dtype: float64
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='Item Type', ylabel='count'>
```

Out[42]:



In [45]:

```
print((df['Outlet Location Type'].value_counts()/8523)*100)
sns.countplot(df['Outlet Location Type'])
```

```
Tier 3    39.305409
Tier 2    32.476288
Tier 1    28.218303
Name: Outlet Location Type, dtype: float64
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='Outlet Location Type', ylabel='count'>
```

Out[45]:

In [46]:

```
sns.distplot(df['Outlet Establishment Year'])
print(df['Outlet Establishment Year'].skew())
print(df['Outlet Establishment Year'].kurt())
```

```
0.06192073080990190
-0.81299399794653?
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[46]:

In [48]:

```
sns.boxplot(df['Outlet Establishment Year'])
```

```
C:\Users\akash bankar\bankar\ana\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
<AxesSubplot:xlabel='Outlet Establishment Year'>
```

Out[48]:

In [ ]: