PROJECT : Big Mart Sales Prediction

Import required libraries

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/Train.csv')
df
```

Out[ ]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	FDA15	9.300	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.1380
1	DRC01	5.920	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
2	FDN15	17.500	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.2700
3	FDX07	19.200	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	Tier 3	Grocery Store	732.3800
4	NCD19	8.930	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052
...	...	...	...	...	...	...	...	...	...	...	...	...
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	High	Tier 3	Supermarket Type1	2778.3834
8519	FDS36	8.380	Regular	0.046982	Baking Goods	108.1570	OUT045	2002	NaN	Tier 2	Supermarket Type1	549.2850
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	Small	Tier 2	Supermarket Type1	1193.1136
8521	FDN46	7.210	Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Medium	Tier 3	Supermarket Type2	1845.5976
8522	DRG01	14.800	Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	Small	Tier 1	Supermarket Type1	765.6700

8523 rows × 12 columns

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

Out[ ]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.1380
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.2700
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	Tier 3	Grocery Store	732.3800
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052

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

Out[ ]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	High	Tier 3	Supermarket Type1	2778.3834
8519	FDS36	8.380	Regular	0.046982	Baking Goods	108.1570	OUT045	2002	NaN	Tier 2	Supermarket Type1	549.2850
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	Small	Tier 2	Supermarket Type1	1193.1136
8521	FDN46	7.210	Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Medium	Tier 3	Supermarket Type2	1845.5976
8522	DRG01	14.800	Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	Small	Tier 1	Supermarket Type1	765.6700

Statistical measures of the dataset

In [ ]:
```
df.describe()
```

Out[ ]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	7060.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857645	0.066132	140.992782	1997.831867	2181.288914
std	4.643456	0.051598	62.275067	8.371760	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.773750	0.026989	93.826500	1987.000000	834.247400
50%	12.600000	0.053931	143.012800	1999.000000	1794.331000
75%	16.850000	0.094585	185.643700	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

In [ ]:
```
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_Identifier            8523 non-null   object 
 1   Item_Weight                7060 non-null   float64
 2   Item_Fat_Content           8523 non-null   object 
 3   Item_Visibility            8523 non-null   float64
 4   Item_Type                  8523 non-null   object 
 5   Item_MRP                   8523 non-null   float64
 6   Outlet_Identifier          8523 non-null   object 
 7   Outlet_Establishment_Year  8523 non-null   int64  
 8   Outlet_Size                6113 non-null   object 
 9   Outlet_Location_Type       8523 non-null   object 
 10  Outlet_Type                8523 non-null   object 
 11  Item_Outlet_Sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

In [ ]:
```
df.dtypes
```

Out[ ]:
```
Item_Identifier              object
Item_Weight                 float64
Item_Fat_Content             object
Item_Visibility             float64
Item_Type                    object
Item_MRP                    float64
Outlet_Identifier            object
Outlet_Establishment_Year     int64
Outlet_Size                  object
Outlet_Location_Type         object
Outlet_Type                  object
Item_Outlet_Sales           float64
dtype: object
```

Checking missing values

In [ ]:
```
df.isna().sum()
```

Out[ ]:
```
Item_Identifier               0
Item_Weight                1463
Item_Fat_Content              0
Item_Visibility               0
Item_Type                     0
Item_MRP                      0
Outlet_Identifier             0
Outlet_Establishment_Year     0
Outlet_Size                2410
Outlet_Location_Type          0
Outlet_Type                   0
Item_Outlet_Sales             0
dtype: int64
```

Fill the missing value

In [ ]:
```
df['Item_Weight']=df['Item_Weight'].fillna(df['Item_Weight'].mean())
df['Outlet_Size']=df['Outlet_Size'].fillna(df['Outlet_Size'].mode()[0])
```

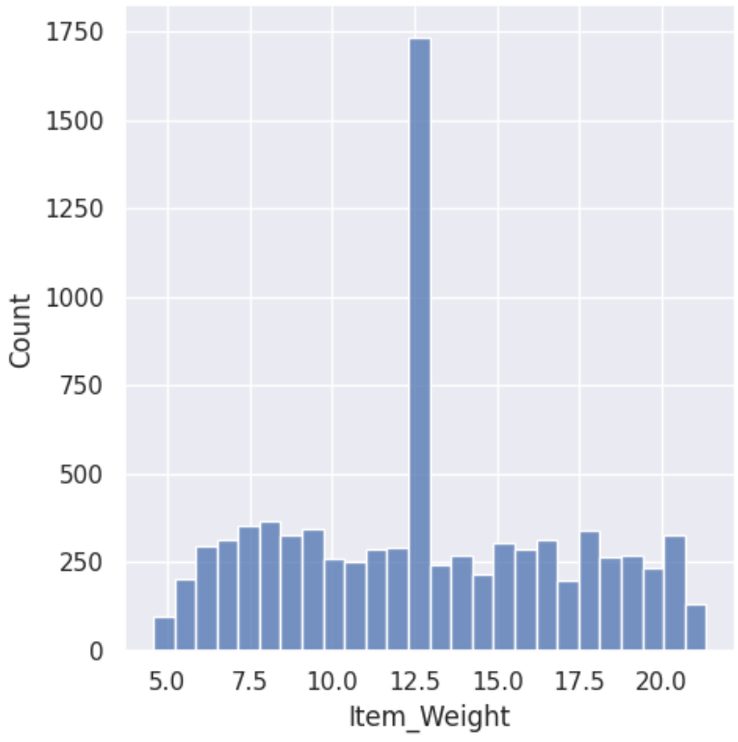
In [ ]:
```
df.isna().sum()
```

Out[ ]:
```
Item_Identifier              0
Item_Weight                  0
Item_Fat_Content             0
Item_Visibility              0
Item_Type                    0
Item_MRP                     0
Outlet_Identifier            0
Outlet_Establishment_Year    0
Outlet_Size                  0
Outlet_Location_Type         0
Outlet_Type                  0
Item_Outlet_Sales            0
dtype: int64
```

Distribution of Item Weight

In [ ]:
```
sns.set()
plt.figure(figsize=(6,6))
sns.displot(df['Item_Weight'])
plt.show()
```

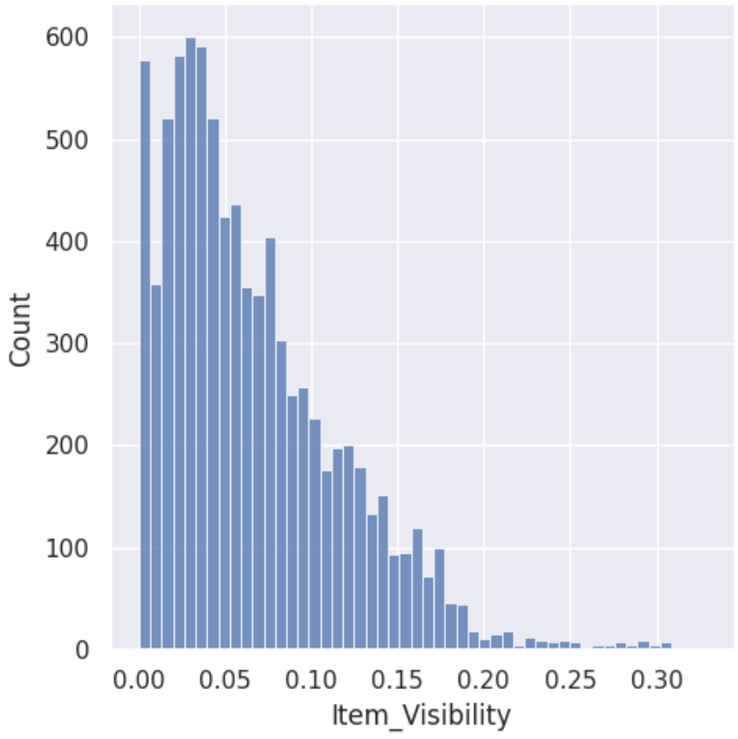
```
<Figure size 600x600 with 0 Axes>
```



Distribution of Item Visibility

In [ ]:
```
plt.figure(figsize=(6,6))
sns.displot(df['Item_Visibility'])
plt.show()
```

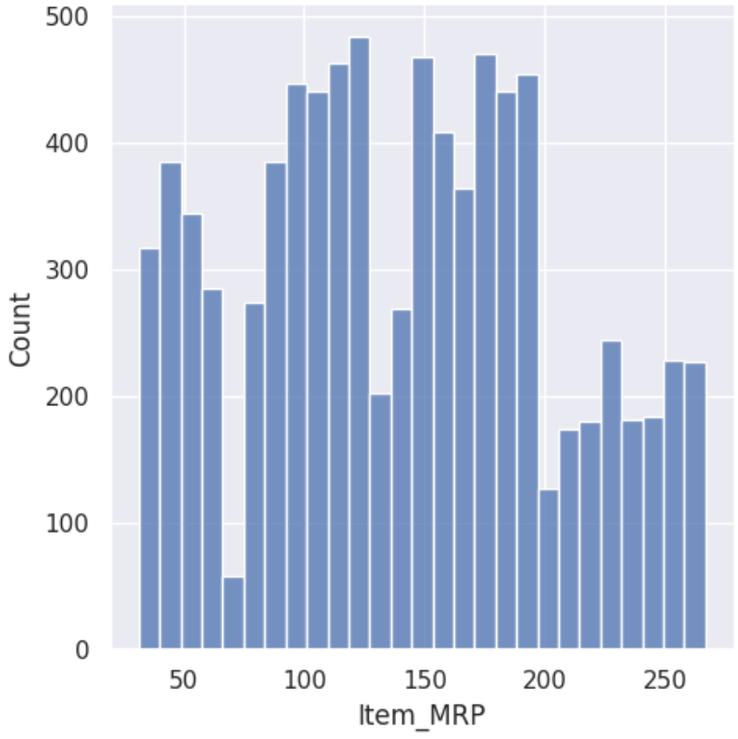
```
<Figure size 600x600 with 0 Axes>
```



Distribution of Item MRP

In [ ]:
```
plt.figure(figsize=(6,6))
sns.displot(df['Item_MRP'])
plt.show()
```

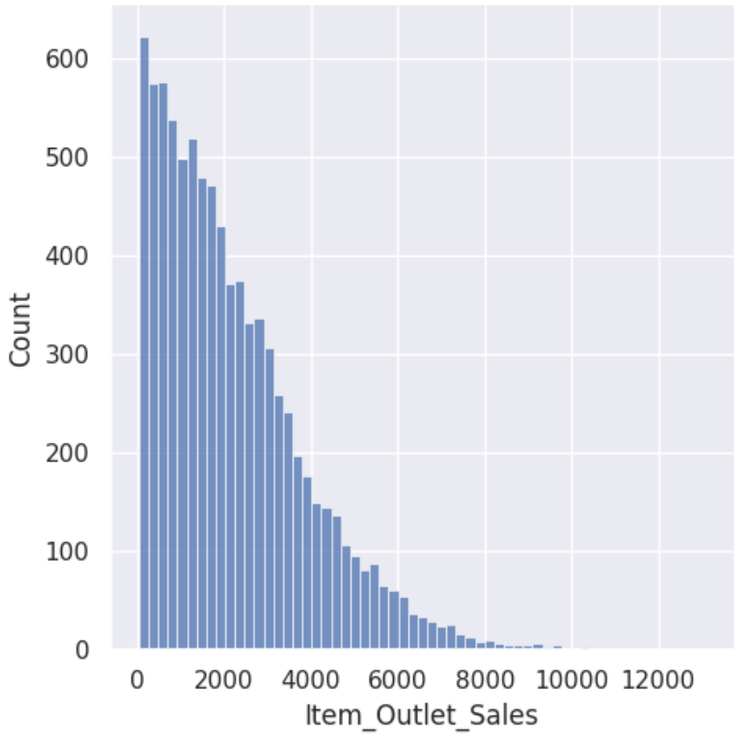
```
<Figure size 600x600 with 0 Axes>
```



Distribution of Item Outlet Sales

In [ ]:
```
plt.figure(figsize=(6,6))
sns.displot(df['Item_Outlet_Sales'])
plt.show()
```

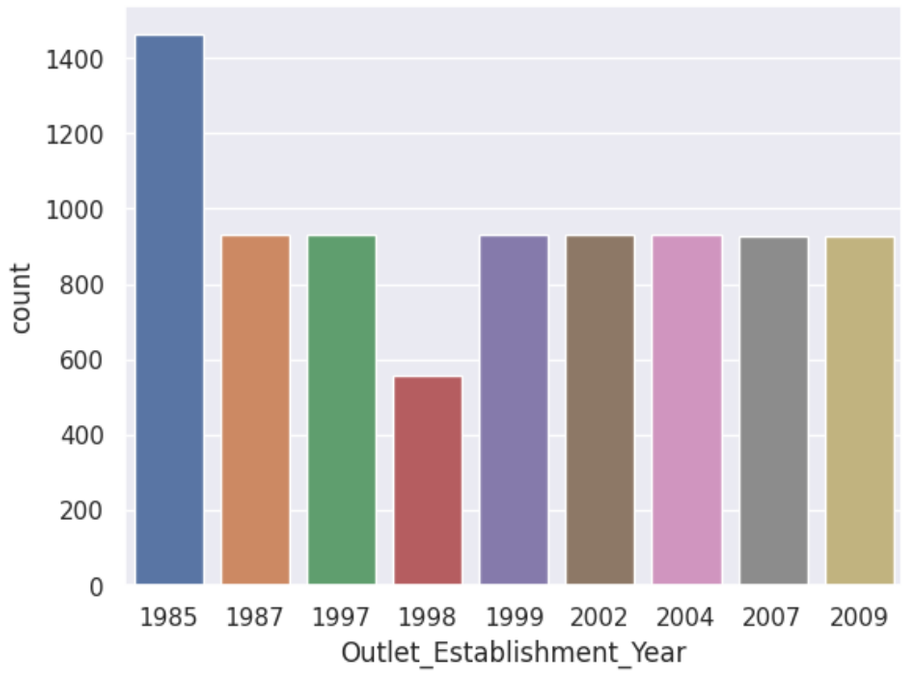
```
<Figure size 600x600 with 0 Axes>
```



Values of Outlet Establishment Year

In [ ]:
```
sns.countplot(x='Outlet_Establishment_Year',data=df)
```

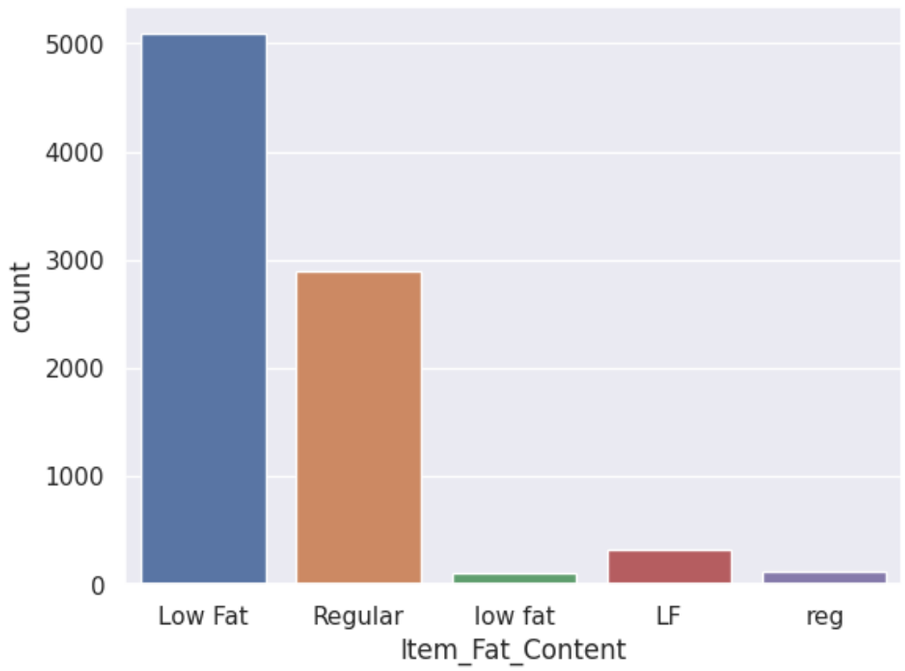
Out[ ]: <Axes: xlabel='Outlet_Establishment_Year', ylabel='count'>



Values of Item_Fat_Content

In [ ]:
```
sns.countplot(x='Item_Fat_Content',data=df)
```

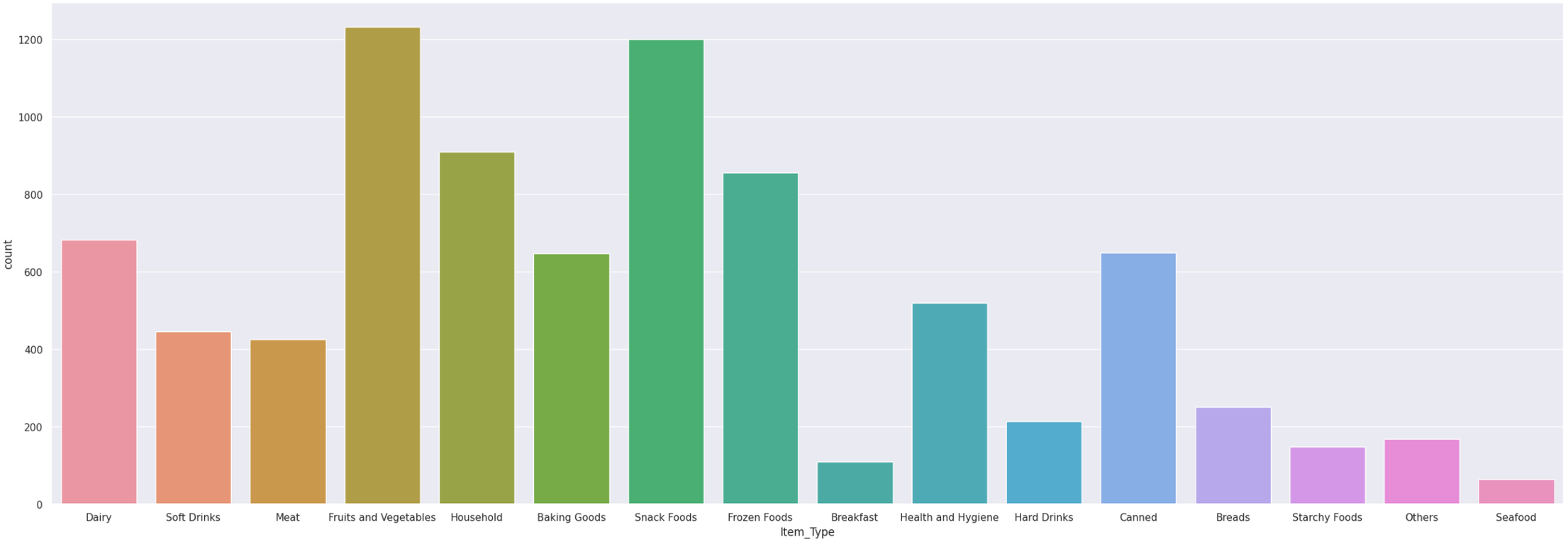
Out[ ]: <Axes: xlabel='Item_Fat_Content', ylabel='count'>



Values of Item_Type

In [ ]:
```
plt.figure(figsize=(30,10))
sns.countplot(x='Item_Type',data=df)
```

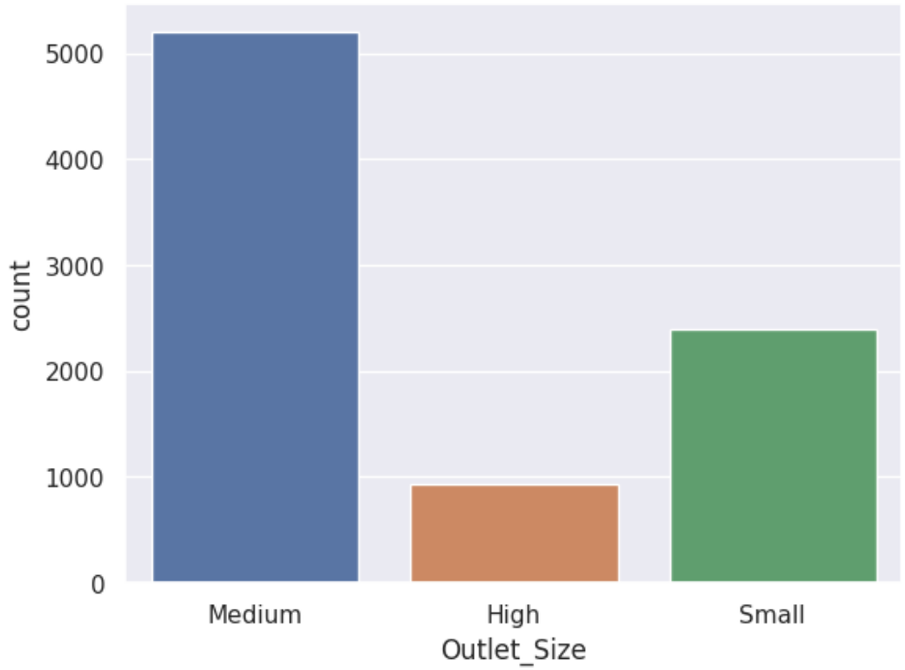
Out[ ]: <Axes: xlabel='Item_Type', ylabel='count'>



Values of Outlet_Size

In [ ]:
```
sns.countplot(x='Outlet_Size',data=df)
```

Out[ ]:
```
<Axes: xlabel='Outlet_Size', ylabel='count'>
```



In [ ]:
```
df['Item_Fat_Content'].value_counts()
```

Out[ ]:
```
Low Fat    5089
Regular    2889
LF          316
reg         117
low fat     112
Name: Item_Fat_Content, dtype: int64
```

In [ ]:
```
df['Item_Fat_Content'].replace({'LF':'Low Fat','low fat':'Low Fat','reg':'Regular'},inplace=True)
```

In [ ]:
```
df['Item_Fat_Content'].value_counts()
```

Out[ ]:
```
Low Fat    5517
Regular    3006
Name: Item_Fat_Content, dtype: int64
```

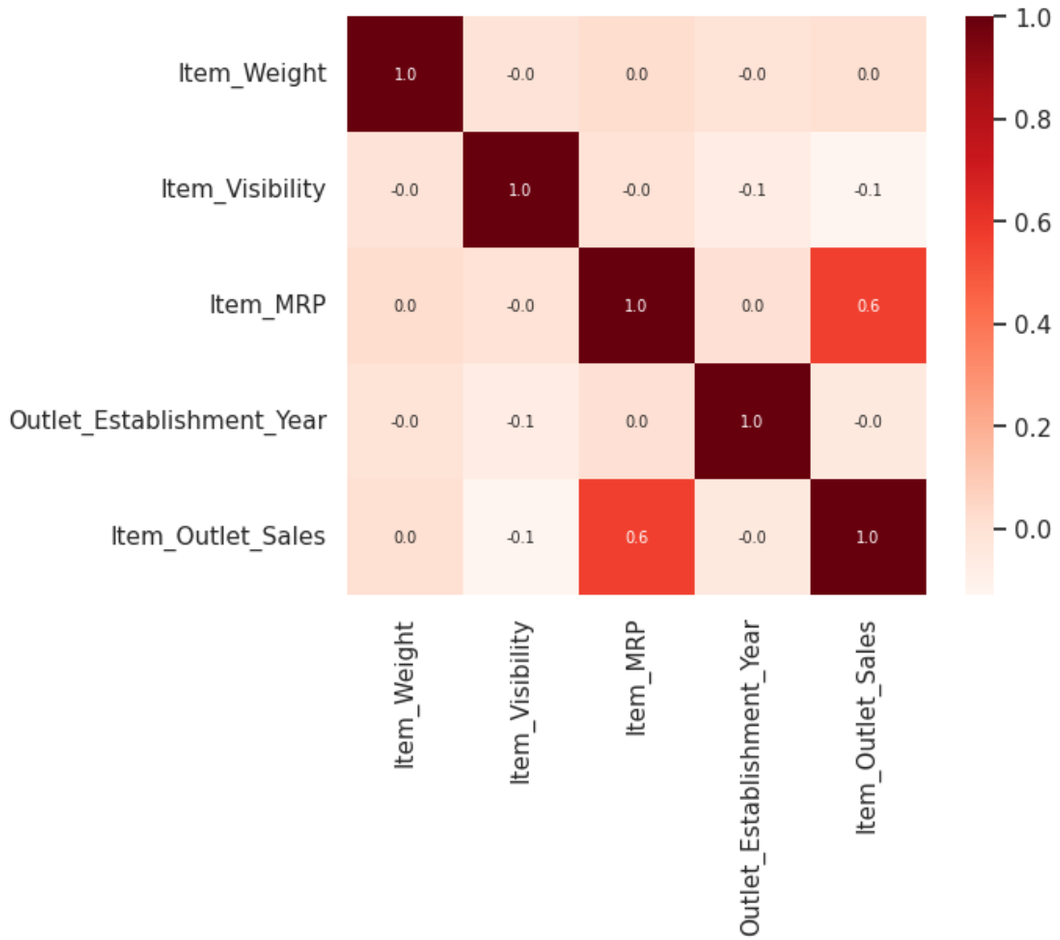
Display Correlation of the data

In [ ]:
```
corr=df.corr()
```

```
<ipython-input-21-0014364bc22a>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  corr=df.corr()
```

In [ ]:
```
sns.heatmap(corr,cbar=True,square=True,fmt='.1f',annot=True,annot_kws={'size':7},cmap='Reds')
```

Out[ ]:
```
<Axes: >
```



Remove unwanted column

In [ ]:
```
df.drop(['Item_Identifier','Item_Visibility','Outlet_Identifier','Outlet_Location_Type'],axis=1,inplace=True)
df.head()
```

Out[ ]:

	Item_Weight	Item_Fat_Content	Item_Type	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Type	Item_Outlet_Sales
0	9.30	Low Fat	Dairy	249.8092	1999	Medium	Supermarket Type1	3735.1380
1	5.92	Regular	Soft Drinks	48.2692	2009	Medium	Supermarket Type2	443.4228
2	17.50	Low Fat	Meat	141.6180	1999	Medium	Supermarket Type1	2097.2700
3	19.20	Regular	Fruits and Vegetables	182.0950	1998	Medium	Grocery Store	732.3800
4	8.93	Low Fat	Household	53.8614	1987	High	Supermarket Type1	994.7052

Encoding columns using LabelEncoder

In [ ]:
```
col=['Item_Fat_Content','Item_Type','Outlet_Size','Outlet_Type']
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
for i in col:
  df[i]=le.fit_transform(df[i])
```

```
df.head()
```

Out[ ]:

	Item_Weight	Item_Fat_Content	Item_Type	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Type	Item_Outlet_Sales
0	9.30	0	4	249.8092	1999	1	1	3735.1380
1	5.92	1	14	48.2692	2009	1	2	443.4228
2	17.50	0	10	141.6180	1999	1	1	2097.2700
3	19.20	1	6	182.0950	1998	1	0	732.3800
4	8.93	0	9	53.8614	1987	0	1	994.7052

Separating the Input features and target

In [ ]:
```
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

Repeat steps in test data

In [ ]:
```
df1=pd.read_csv('/content/Test.csv')
df1
```

Out[ ]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type
0	FDW58	20.750	Low Fat	0.007565	Snack Foods	107.8622	OUT049	1999	Medium	Tier 1	Supermarket Type1
1	FDW14	8.300	reg	0.038428	Dairy	87.3198	OUT017	2007	NaN	Tier 2	Supermarket Type1
2	NCN55	14.600	Low Fat	0.099575	Others	241.7538	OUT010	1998	NaN	Tier 3	Grocery Store
3	FDQ58	7.315	Low Fat	0.015388	Snack Foods	155.0340	OUT017	2007	NaN	Tier 2	Supermarket Type1
4	FDY38	NaN	Regular	0.118599	Dairy	234.2300	OUT027	1985	Medium	Tier 3	Supermarket Type3
...	...	...	...	...	...	...	...	...	...	...	...
5676	FDB58	10.500	Regular	0.013496	Snack Foods	141.3154	OUT046	1997	Small	Tier 1	Supermarket Type1
5677	FDD47	7.600	Regular	0.142991	Starchy Foods	169.1448	OUT018	2009	Medium	Tier 3	Supermarket Type2
5678	NCO17	10.000	Low Fat	0.073529	Health and Hygiene	118.7440	OUT045	2002	NaN	Tier 2	Supermarket Type1
5679	FDJ26	15.300	Regular	0.000000	Canned	214.6218	OUT017	2007	NaN	Tier 2	Supermarket Type1
5680	FDU37	9.500	Regular	0.104720	Canned	79.7960	OUT045	2002	NaN	Tier 2	Supermarket Type1

5681 rows × 11 columns

In [ ]:
```
df1.head()
```

Out[ ]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type
0	FDW58	20.750	Low Fat	0.007565	Snack Foods	107.8622	OUT049	1999	Medium	Tier 1	Supermarket Type1
1	FDW14	8.300	reg	0.038428	Dairy	87.3198	OUT017	2007	NaN	Tier 2	Supermarket Type1
2	NCN55	14.600	Low Fat	0.099575	Others	241.7538	OUT010	1998	NaN	Tier 3	Grocery Store
3	FDQ58	7.315	Low Fat	0.015388	Snack Foods	155.0340	OUT017	2007	NaN	Tier 2	Supermarket Type1
4	FDY38	NaN	Regular	0.118599	Dairy	234.2300	OUT027	1985	Medium	Tier 3	Supermarket Type3

In [ ]:
```
df1.tail()
```

Out[ ]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type
5676	FDB58	10.5	Regular	0.013496	Snack Foods	141.3154	OUT046	1997	Small	Tier 1	Supermarket Type1
5677	FDD47	7.6	Regular	0.142991	Starchy Foods	169.1448	OUT018	2009	Medium	Tier 3	Supermarket Type2
5678	NCO17	10.0	Low Fat	0.073529	Health and Hygiene	118.7440	OUT045	2002	NaN	Tier 2	Supermarket Type1
5679	FDJ26	15.3	Regular	0.000000	Canned	214.6218	OUT017	2007	NaN	Tier 2	Supermarket Type1
5680	FDU37	9.5	Regular	0.104720	Canned	79.7960	OUT045	2002	NaN	Tier 2	Supermarket Type1

In [ ]:
```
df1.describe()
```

Out[ ]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year
count	4705.000000	5681.000000	5681.000000	5681.000000
mean	12.695633	0.065684	141.023273	1997.828903
std	4.664849	0.051252	61.809091	8.372256
min	4.555000	0.000000	31.990000	1985.000000
25%	8.645000	0.027047	94.412000	1987.000000
50%	12.500000	0.054154	141.415400	1999.000000
75%	16.700000	0.093463	186.026600	2004.000000
max	21.350000	0.323637	266.588400	2009.000000

In [ ]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5681 entries, 0 to 5680
Data columns (total 11 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item_Identifier            5681 non-null   object
 1   Item_Weight                4705 non-null   float64
 2   Item_Fat_Content           5681 non-null   object
 3   Item_Visibility            5681 non-null   float64
 4   Item_Type                  5681 non-null   object
 5   Item_MRP                   5681 non-null   float64
 6   Outlet_Identifier          5681 non-null   object
 7   Outlet_Establishment_Year  5681 non-null   int64
 8   Outlet_Size                4075 non-null   object
 9   Outlet_Location_Type       5681 non-null   object
 10  Outlet_Type                5681 non-null   object
dtypes: float64(3), int64(1), object(7)
memory usage: 488.3+ KB
```

In [ ]:
```
df1.dtypes
```

Out[ ]:
```
Item_Identifier              object
Item_Weight                 float64
Item_Fat_Content             object
Item_Visibility             float64
Item_Type                    object
Item_MRP                    float64
Outlet_Identifier            object
Outlet_Establishment_Year     int64
Outlet_Size                  object
Outlet_Location_Type         object
Outlet_Type                  object
dtype: object
```

In [ ]:
```
df1.isna().sum()
```

Out[ ]:
```
Item_Identifier               0
Item_Weight                 976
Item_Fat_Content              0
Item_Visibility               0
Item_Type                     0
Item_MRP                      0
Outlet_Identifier             0
Outlet_Establishment_Year     0
Outlet_Size                1606
Outlet_Location_Type          0
Outlet_Type                   0
dtype: int64
```

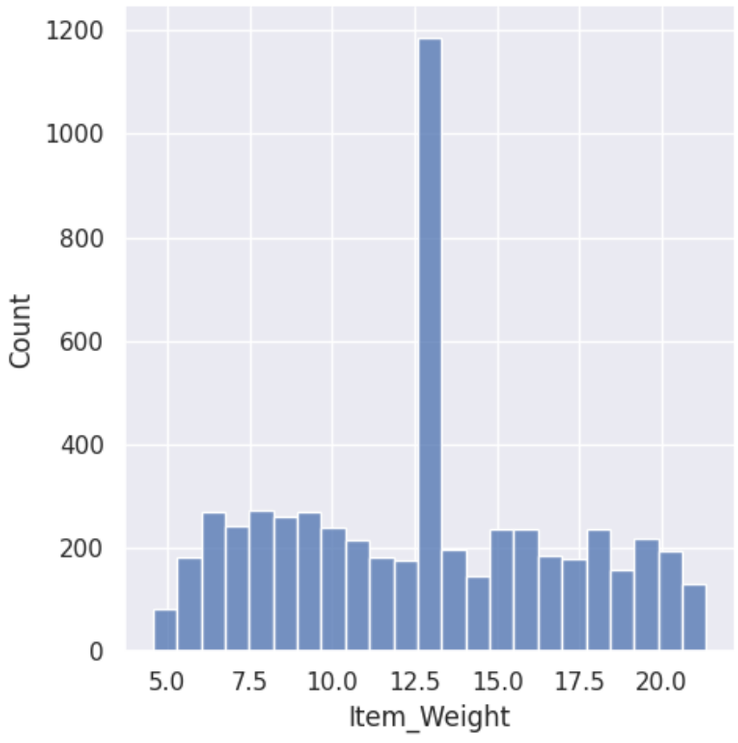
In [ ]:
```
df1['Item_Weight']=df1['Item_Weight'].fillna(df1['Item_Weight'].mean())
df1['Outlet_Size']=df1['Outlet_Size'].fillna(df1['Outlet_Size'].mode()[0])
```

In [ ]:
```
df1.isna().sum()
```

Out[ ]:
```
Item_Identifier              0
Item_Weight                  0
Item_Fat_Content             0
Item_Visibility              0
Item_Type                    0
Item_MRP                     0
Outlet_Identifier            0
Outlet_Establishment_Year    0
Outlet_Size                  0
Outlet_Location_Type         0
Outlet_Type                  0
dtype: int64
```

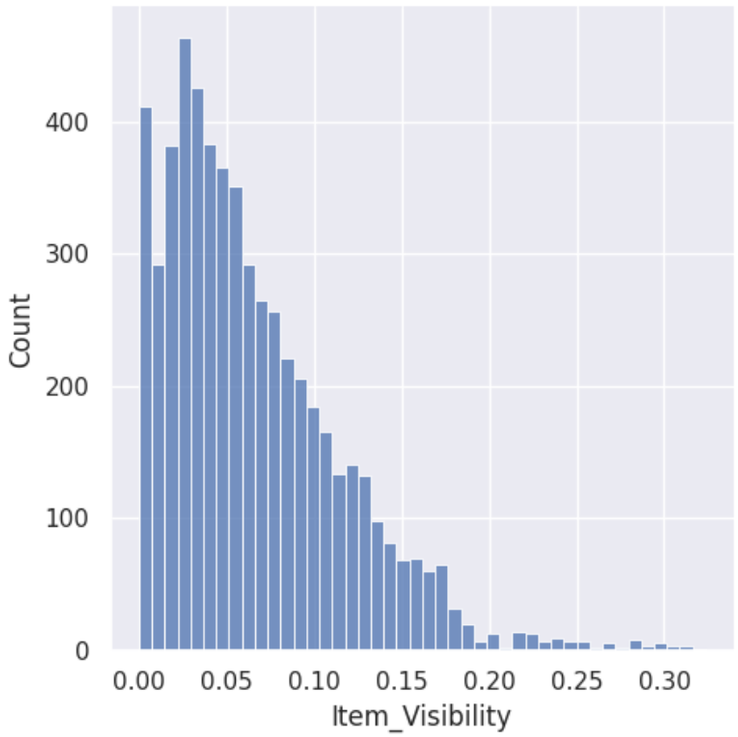
In [ ]:
```
sns.set()
plt.figure(figsize=(6,6))
sns.displot(df1['Item_Weight'])
plt.show()
```

```
<Figure size 600x600 with 0 Axes>
```



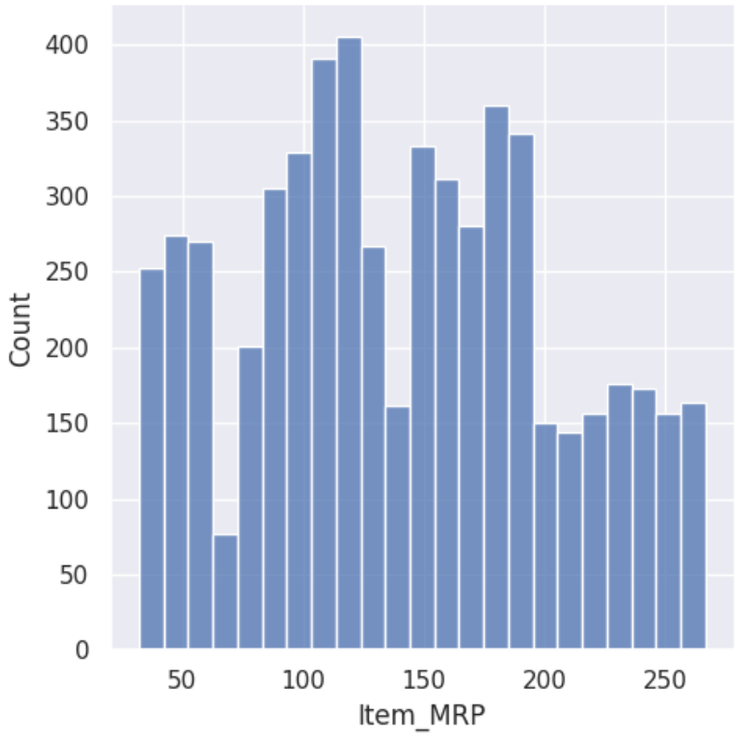
In [ ]:
```
plt.figure(figsize=(6,6))
sns.displot(df1['Item_Visibility'])
plt.show()
```

```
<Figure size 600x600 with 0 Axes>
```



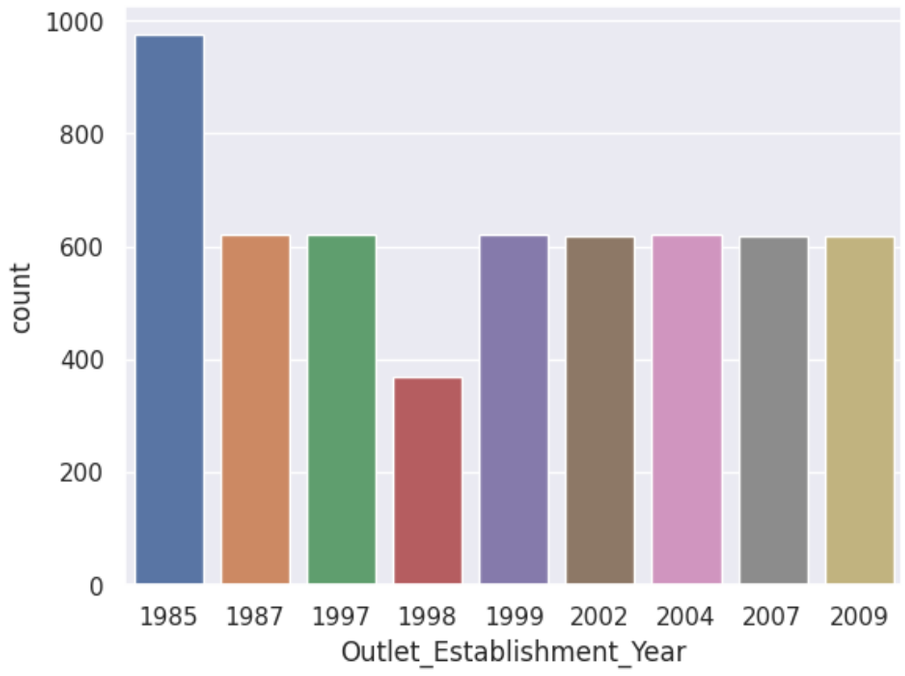
In [ ]:
```
plt.figure(figsize=(6,6))
sns.displot(df1['Item_MRP'])
plt.show()
```

```
<Figure size 600x600 with 0 Axes>
```



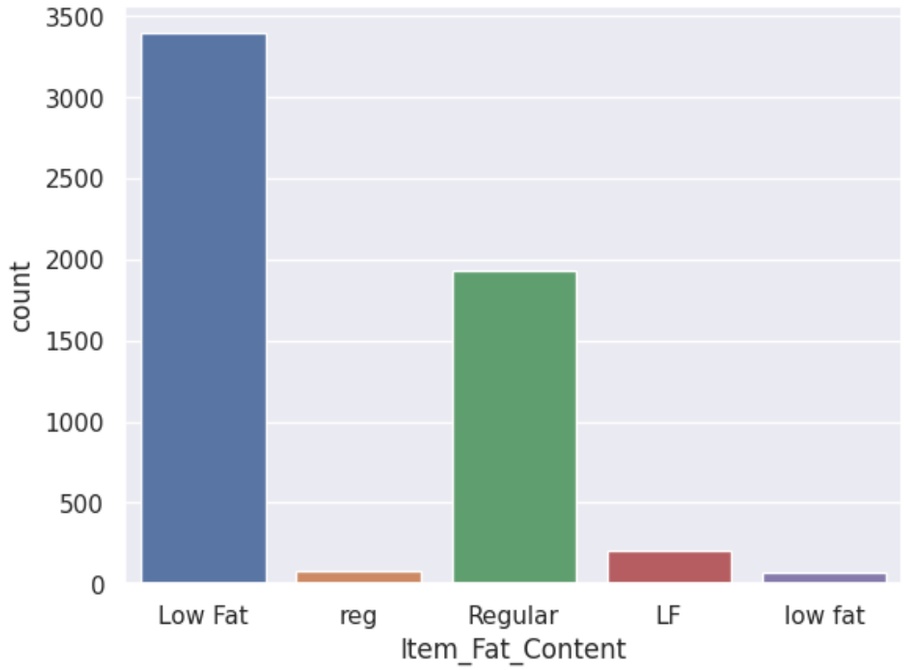
In [ ]:
```
sns.countplot(x='Outlet_Establishment_Year',data=df1)
```

Out[ ]: <Axes: xlabel='Outlet_Establishment_Year', ylabel='count'>



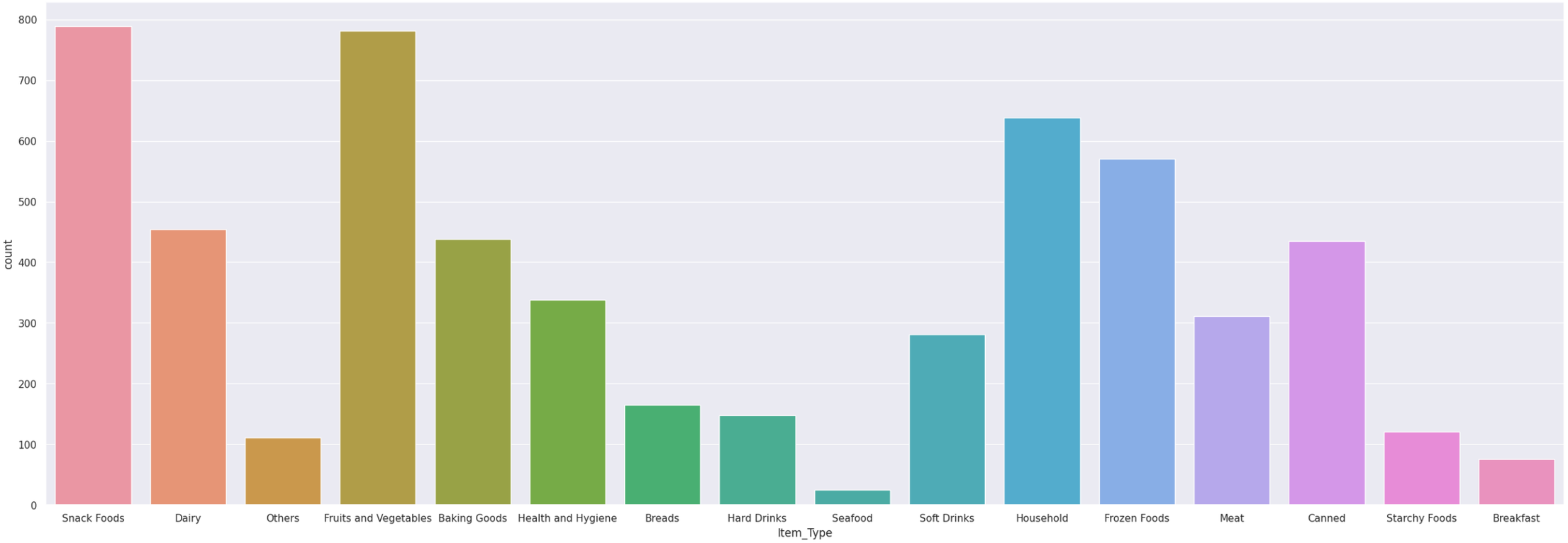
In [ ]:
```
sns.countplot(x='Item_Fat_Content',data=df1)
```

Out[ ]: <Axes: xlabel='Item_Fat_Content', ylabel='count'>



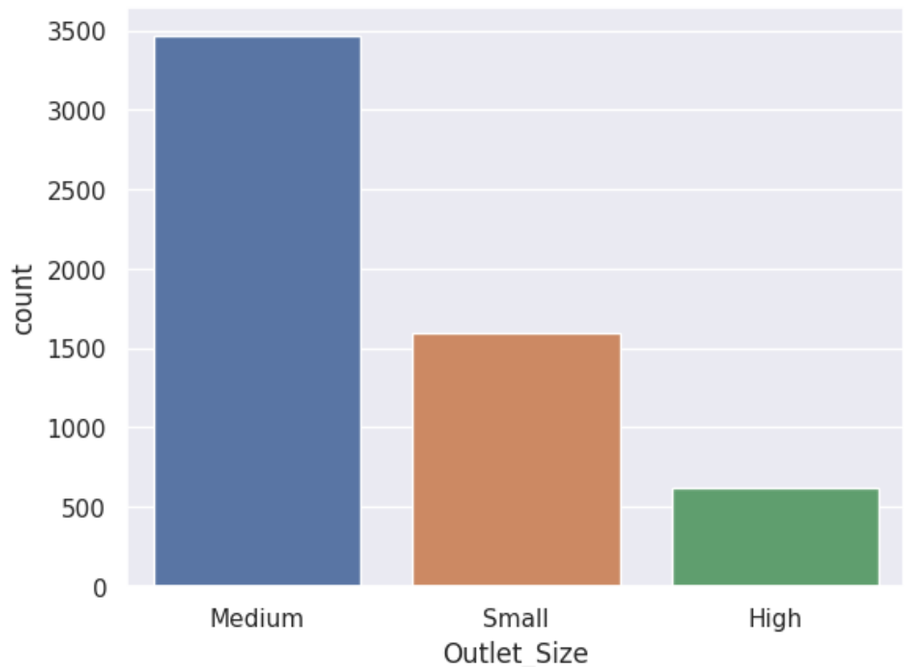
In [ ]:
```
plt.figure(figsize=(30,10))
sns.countplot(x='Item_Type',data=df1)
```

Out[ ]: <Axes: xlabel='Item_Type', ylabel='count'>



In [ ]:
```
sns.countplot(x='Outlet_Size',data=df1)
```

Out[ ]: <Axes: xlabel='Outlet_Size', ylabel='count'>



In [ ]:
```
df1['Item_Fat_Content'].value_counts()
```

Out[ ]:
```
Low Fat    3396
Regular    1935
LF          206
reg          78
low fat      66
Name: Item_Fat_Content, dtype: int64
```

In [ ]:
```
df1['Item_Fat_Content'].replace({'LF':'Low Fat','low fat':'Low Fat','reg':'Regular'},inplace=True)
```

In [ ]:
```
df1['Item_Fat_Content'].value_counts()
```

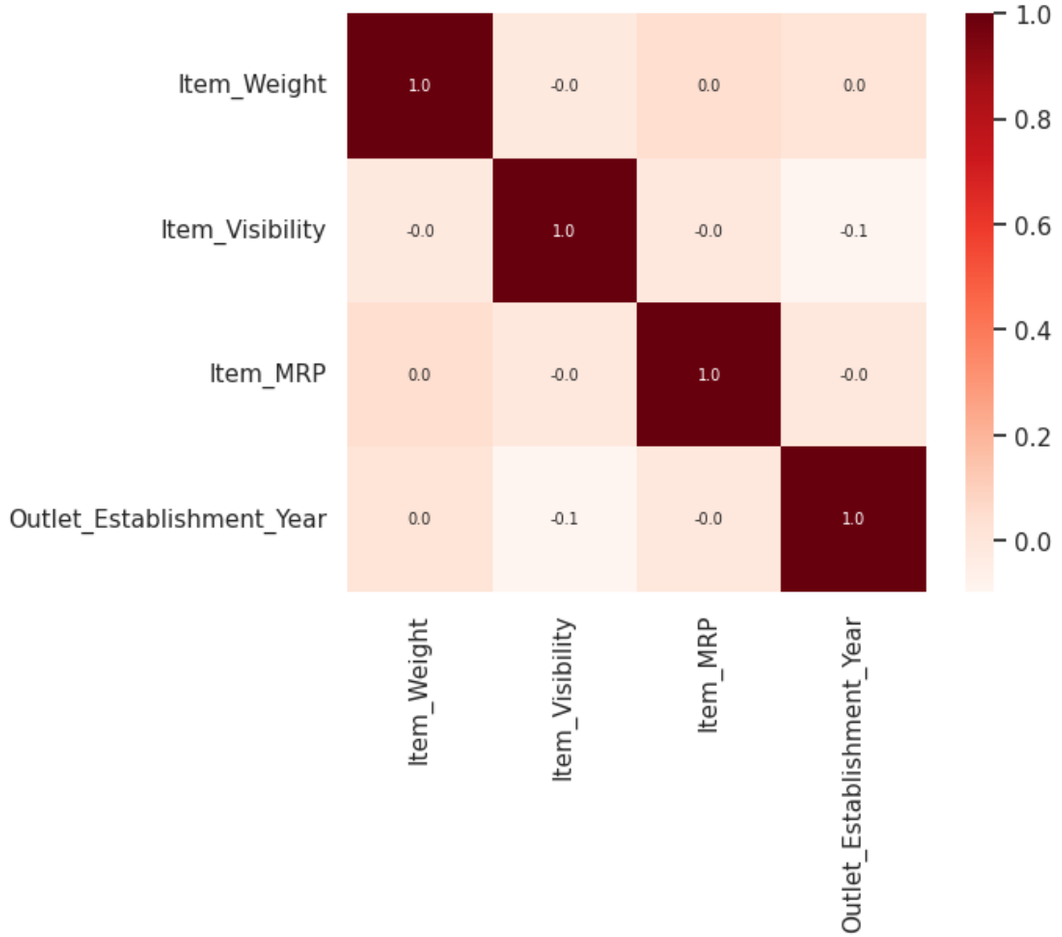
Out[ ]:
```
Low Fat    3668
Regular    2013
Name: Item_Fat_Content, dtype: int64
```

In [ ]:
```
corr=df1.corr()
```

```
<ipython-input-45-fa62c0265c31>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  corr=df1.corr()
```

In [ ]:
```
sns.heatmap(corr,cbar=True,square=True,fmt='.1f',annot=True,annot_kws={'size':7},cmap='Reds')
```

Out[ ]:
```
<Axes: >
```



In [ ]:
```
df1.drop(['Item_Identifier','Item_Visibility','Outlet_Identifier','Outlet_Location_Type'],axis=1,inplace=True)
df1.head()
```

Out[ ]:

	Item_Weight	Item_Fat_Content	Item_Type	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Type
0	20.750000	Low Fat	Snack Foods	107.8622	1999	Medium	Supermarket Type1
1	8.300000	Regular	Dairy	87.3198	2007	Medium	Supermarket Type1
2	14.600000	Low Fat	Others	241.7538	1998	Medium	Grocery Store
3	7.315000	Low Fat	Snack Foods	155.0340	2007	Medium	Supermarket Type1
4	12.695633	Regular	Dairy	234.2300	1985	Medium	Supermarket Type3

In [ ]:
```
col=['Item_Fat_Content','Item_Type','Outlet_Size','Outlet_Type']
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
for i in col:
  df1[i]=le.fit_transform(df1[i])
df1.head()
```

Out[ ]:

	Item_Weight	Item_Fat_Content	Item_Type	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Type
0	20.750000	0	13	107.8622	1999	1	1
1	8.300000	1	4	87.3198	2007	1	1
2	14.600000	0	11	241.7538	1998	1	0
3	7.315000	0	13	155.0340	2007	1	1
4	12.695633	1	4	234.2300	1985	1	3

Model creation using

- LinearRegression
- DecisionTreeRegressor
- RandomForestRegressor
- XGBRegressor

LinearRegression

In [ ]:
```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x,y)
y_pred=model.predict(df1)
y_pred
```

Out[ ]:
```
array([1470.24359658, 1211.77796228, 2703.10752783, ..., 1649.20733998,
       3189.12700876, 1095.27431328])
```

DecisionTreeRegressor

In [ ]:
```
from sklearn.tree import DecisionTreeRegressor
dec=DecisionTreeRegressor()
dec.fit(x,y)
y_pred1=dec.predict(df1)
y_pred1
```

Out[ ]:
```
array([1965.4416, 1142.5128,  759.012 , ...,  964.0784, 4060.7142,
       1038.648 ])
```

RandomForestRegressor

In [ ]:
```
from sklearn.ensemble import RandomForestRegressor
rfg=RandomForestRegressor()
rfg.fit(x,y)
y_pred2=rfg.predict(df1)
y_pred2
```

Out[ ]:
```
array([1848.806756, 1197.328114,  637.243838, ..., 1600.703044,
       3690.82901 , 1274.061564])
```

XGBRegressor

In [ ]:
```
from xgboost import XGBRegressor
xgb=XGBRegressor()
xgb.fit(x,y)
y_pred3=xgb.predict(df1)
y_pred3
```

Out[ ]:
```
array([1346.2289, 1431.7036,  624.7257, ..., 1797.3999, 3206.701 ,
       1233.9525], dtype=float32)
```

In [ ]:

In [ ]:

In [ ]: