In [151…
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from scipy import stats
```

Store - the store number

Date - the week of sales

Weekly_Sales - sales for the given store

Holiday_Flag - whether the week is a special holiday week 1 – Holiday week 0 – Non-holiday week

Temperature - Temperature on the day of sale

Fuel_Price - Cost of fuel in the region

CPI – Prevailing consumer price index

Unemployment - Prevailing unemployment rate

Super Bowl: 12-Feb-10, 11-Feb-11, 10-Feb-12, 8-Feb-13 [2010-02-12, 2011-02-11, 2012-02-10, 2013-02-08] Labour Day: 10-Sep-10, 9-Sep-11, 7-Sep-12, 6-Sep-13 [2010-09-10, 2011-09-09, 2012-09-07, 2013-09-06] Thanksgiving: 26-Nov-10, 25-Nov-11, 23-Nov-12, 29-Nov-13 [2010-11-26, 2011-11-25, 2012-11-23, 2013-11-29] Christmas: 31-Dec-10, 30-Dec-11, 28-Dec-12, 27-Dec-13 [2010-12-31, 2011-12-30, 2010-12-28, 2013-12-27]

[2010-02-12, 2011-02-11, 2012-02-10, 2013-02-08,2010-09-10, 2011-09-09, 2012-09-07, 2013-09-06, 2010-11-26, 2011-11-25, 2012-11-23, 2013-11-29,2010-12-31, 2011-12-30, 2010-12-28, 2013-12-27]

Basics Statistics tasks

which store has maximum sales

Which store has maximum standard deviation i.e., the sales vary a lot. Also, find out the coefficient of mean to standard deviation

Which store/s has good quarterly growth rate in Q3'2012

Some holidays have a negative impact on sales. Find out holidays which have higher sales than the mean sales in non-holiday season for all stores together

Provide a monthly and semester view of sales in units and give insights

In [6]:
```
df = pd.read_csv(r'D:\whatsapp downloads\walmart_store_sales.csv')
df.head()
```

Out[6]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	8.106
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	8.106
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	8.106
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	8.106
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	8.106

In [7]:
```
#which store has maximum sales
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6435 entries, 0 to 6434
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Store         6435 non-null   int64  
 1   Date          6435 non-null   object 
 2   Weekly_Sales  6435 non-null   float64
 3   Holiday_Flag  6435 non-null   int64  
 4   Temperature   6435 non-null   float64
 5   Fuel_Price    6435 non-null   float64
 6   CPI           6435 non-null   float64
 7   Unemployment  6435 non-null   float64
dtypes: float64(5), int64(2), object(1)
memory usage: 402.3+ KB
```

In [8]:
```
df.describe()
```

Out[8]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
count	6435.000000	6.435000e+03	6435.000000	6435.000000	6435.000000	6435.000000	6435.000000
mean	23.000000	1.046965e+06	0.069930	60.663782	3.358607	171.578394	7.999151
std	12.988182	5.643666e+05	0.255049	18.444933	0.459020	39.356712	1.875885
min	1.000000	2.099862e+05	0.000000	-2.060000	2.472000	126.064000	3.879000
25%	12.000000	5.533501e+05	0.000000	47.460000	2.933000	131.735000	6.891000
50%	23.000000	9.607460e+05	0.000000	62.670000	3.445000	182.616521	7.874000
75%	34.000000	1.420159e+06	0.000000	74.940000	3.735000	212.743293	8.622000
max	45.000000	3.818686e+06	1.000000	100.140000	4.468000	227.232807	14.313000

In [9]:
```
df.sort_values(by='Weekly_Sales',ascending=False).groupby('Store').sum('Weekly_Sales')
```

Out[9]:

Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
1	2.224028e+08	10	9767.87	460.417	30887.555523	1088.290
2	2.753824e+08	10	9754.94	460.417	30837.422420	1090.210
3	5.758674e+07	10	10215.09	460.417	31372.988971	1026.309
4	2.995440e+08	10	8902.23	460.027	18401.192733	852.951
5	4.547569e+07	10	9925.65	460.417	30968.878137	900.243
6	2.237561e+08	10	9967.10	460.417	31110.107182	944.787
7	8.159828e+07	10	5680.00	463.543	27693.986741	1227.760
8	1.299512e+08	10	8939.50	460.417	31379.780750	871.134
9	7.778922e+07	10	9691.85	460.417	31406.616557	872.283
10	2.716177e+08	10	10330.49	511.357	18401.192733	1195.904
11	1.939628e+08	10	10364.75	460.417	31372.988971	1026.309
12	1.442872e+08	10	10047.58	515.718	18401.192733	1875.657
13	2.865177e+08	10	7678.69	469.919	18401.192733	1001.261
14	2.889999e+08	10	8264.11	488.718	26638.851959	1236.771
15	8.913368e+07	10	7412.24	511.696	19318.242848	1143.464
16	7.425243e+07	10	6439.30	463.543	27693.986741	926.353
17	1.277821e+08	10	6633.37	469.919	18401.192733	936.565
18	1.551147e+08	10	7632.09	492.169	19318.242848	1263.877
19	2.066349e+08	10	7478.19	511.696	19318.242848	1143.464
20	3.013978e+08	10	7929.55	488.718	29892.452680	1054.112
21	1.081179e+08	10	9845.21	460.417	30837.422420	1090.210
22	1.470756e+08	10	7850.29	492.169	19878.613542	1153.920
23	1.987506e+08	10	6979.13	492.169	19318.242848	685.830
24	1.940160e+08	10	7726.29	511.696	19318.242848	1207.923
25	1.010612e+08	10	7455.79	488.718	29892.452680	1054.112
26	1.434164e+08	10	6243.13	492.169	19318.242848	1125.706
27	2.538559e+08	10	8195.49	511.696	19878.613542	1144.250
28	1.892637e+08	10	10047.58	515.718	18401.192733	1875.657
29	7.714155e+07	10	7850.29	492.169	19318.242848	1402.313
30	6.271689e+07	10	9845.21	460.417	30837.422420	1090.210
31	1.996139e+08	10	9845.21	460.417	30837.422420	1090.210
32	1.668192e+08	10	7542.90	463.543	27693.986741	1227.760
33	3.716022e+07	10	10972.13	511.357	18401.192733	1220.241
34	1.382498e+08	10	8364.91	460.027	18401.192733	1420.677
35	1.315207e+08	10	8195.49	488.718	19878.613542	1256.766
36	5.341221e+07	10	10175.93	458.201	30706.256907	1125.274
37	7.420274e+07	10	10175.93	460.417	30706.256907	1125.274
38	5.515963e+07	10	10047.58	515.718	18401.192733	1875.657
39	2.074455e+08	10	10095.42	460.417	30706.256907	1125.274
40	1.378703e+08	10	6817.46	492.169	19318.242848	685.830
41	1.813419e+08	10	6922.68	463.543	27693.986741	997.193
42	7.956575e+07	10	10330.49	511.357	18401.192733	1195.904
43	9.056544e+07	10	9849.51	460.417	29706.128216	1420.677
44	4.329309e+07	10	7678.69	469.919	18401.192733	963.194
45	1.123953e+08	10	8264.11	488.718	26638.851959	1236.771

In [14]:
```
df['Weekly_Sales'].sum().max()
```

Out[14]:
```
6737218987.11
```

In [17]:
```
max_Sales.idxmax()
```

Out[17]:
```
20
```

In [18]:
```
max_Sales=df.groupby('Store')['Weekly_Sales'].sum()
```

```
max_Sales
```

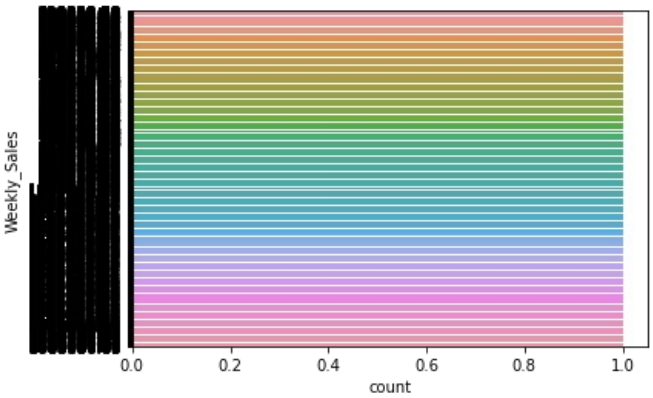
Out[18]:
```
Store
1     2.224028e+08
2     2.753824e+08
3     5.758674e+07
4     2.995440e+08
5     4.547569e+07
6     2.237561e+08
7     8.159828e+07
8     1.299512e+08
9     7.778922e+07
10    2.716177e+08
11    1.939628e+08
12    1.442872e+08
13    2.865177e+08
14    2.889999e+08
15    8.913368e+07
16    7.425243e+07
17    1.277821e+08
18    1.551147e+08
19    2.066349e+08
20    3.013978e+08
21    1.081179e+08
22    1.470756e+08
23    1.987506e+08
24    1.940160e+08
25    1.010612e+08
26    1.434164e+08
27    2.538559e+08
28    1.892637e+08
29    7.714155e+07
30    6.271689e+07
31    1.996139e+08
32    1.668192e+08
33    3.716022e+07
34    1.382498e+08
35    1.315207e+08
36    5.341221e+07
37    7.420274e+07
38    5.515963e+07
39    2.074455e+08
40    1.378703e+08
41    1.813419e+08
42    7.956575e+07
43    9.056544e+07
44    4.329309e+07
45    1.123953e+08
Name: Weekly_Sales, dtype: float64
```

In [19]:
```
max_Sales=df.groupby('Store')['Weekly_Sales'].sum()
max_Sales.idxmax()
```

Out[19]:
```
20
```

In [35]:
```
sns.countplot(y='Weekly_Sales',data=df)
plt.figure(figsize=(10,10))
```

Out[35]:
```
<Figure size 720x720 with 0 Axes>
```

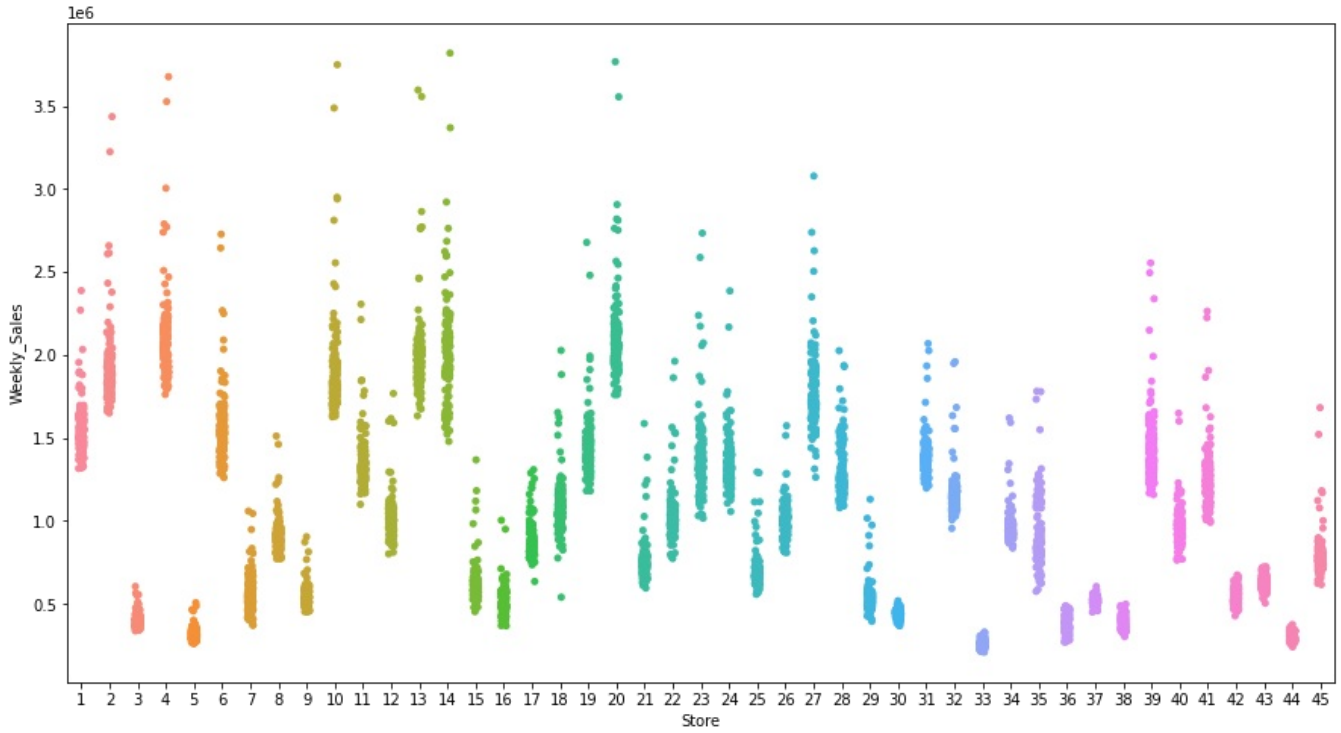



```
<Figure size 720x720 with 0 Axes>
```

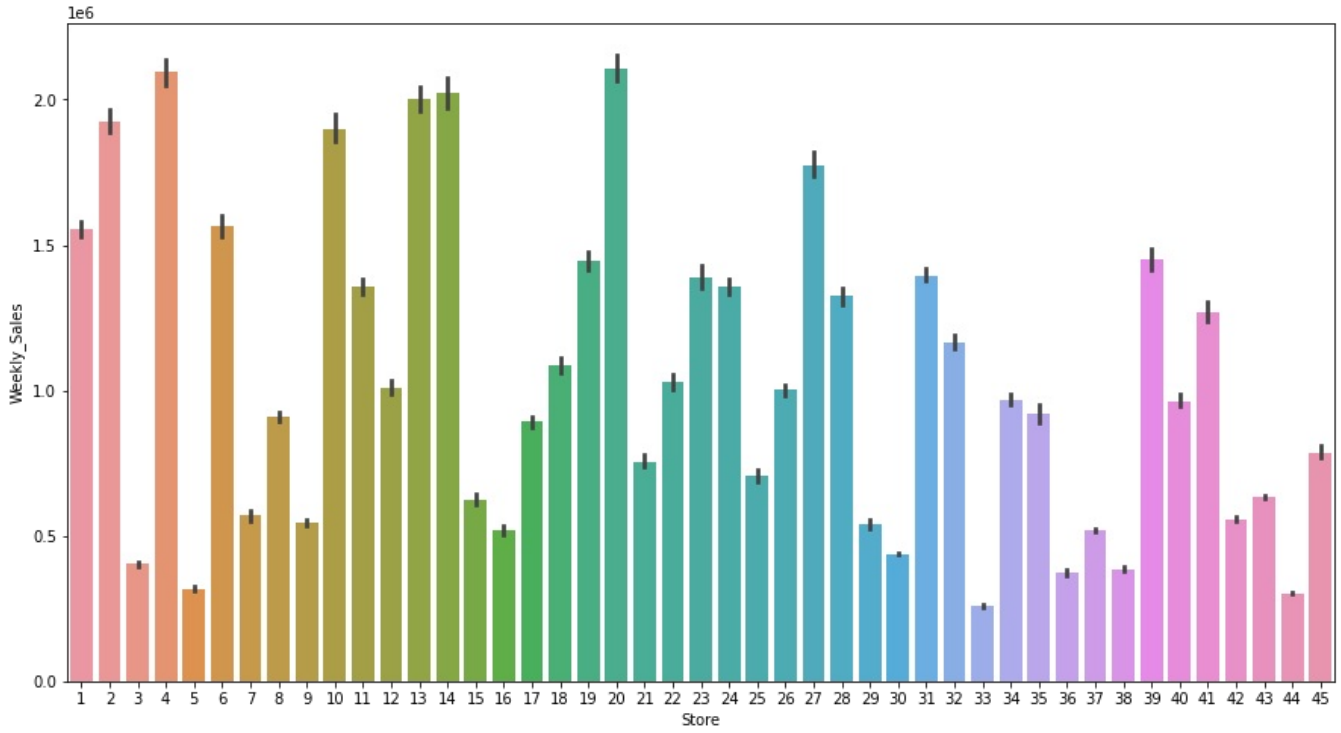
In [33]:
```
plt.figure(figsize = (15,8))
sns.stripplot('Store','Weekly_Sales',data=df)
```

```
C:\Users\Dell\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable
s as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing othe
r arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[33]:
```
<AxesSubplot:xlabel='Store', ylabel='Weekly_Sales'>
```

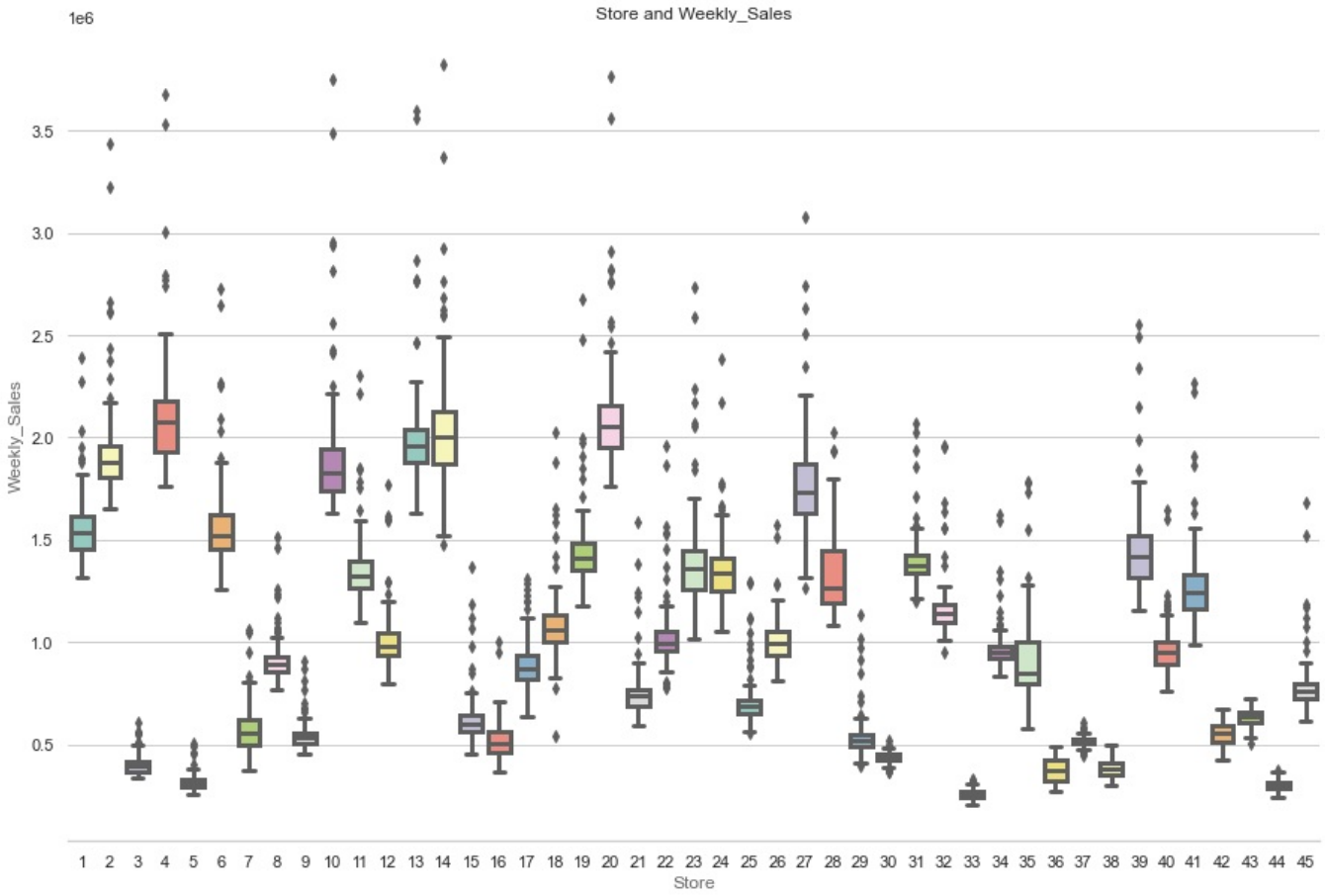


In [21]:
```
plt.figure(figsize = (15,8))
ax = sns.barplot(x="Store", y="Weekly_Sales", data=df)
```



In [51]:
```
sns.set(style="whitegrid")
f, ax = plt.subplots(figsize=(15,10))
sns.boxplot(x="Store",y="Weekly_Sales",data=df, palette="Set3", linewidth=3, ax=ax)
sns.despine(left=True)
ax.set_title("Store and Weekly_Sales")
ax.set_xlabel("Store", alpha=0.7)
ax.set_ylabel("Weekly_Sales", alpha=0.7)
```

Out[51]:
```
Text(0, 0.5, 'Weekly_Sales')
```



Which store has maximum standard deviation i.e., the sales vary a lot. Also, find out the coefficient of mean to standard deviation

In [24]:
```
max_Sales=df.groupby('Store')['Weekly_Sales'].std()

max_Sales.idxmax()
```

Out[24]:
```
14
```

coefficent of varaince for whole data = std\mean of a column

In [56]:
```
df.describe()
```

Out[56]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month
count	6435.000000	6.435000e+03	6435.000000	6435.000000	6435.000000	6435.000000	6435.000000	6435.000000	6435.000000
mean	23.000000	1.046965e+06	0.069930	60.663782	3.358607	171.578394	7.999151	2010.965035	6.475524
std	12.988182	5.643666e+05	0.255049	18.444933	0.459020	39.356712	1.875885	0.797019	3.321797
min	1.000000	2.099862e+05	0.000000	-2.060000	2.472000	126.064000	3.879000	2010.000000	1.000000
25%	12.000000	5.533501e+05	0.000000	47.460000	2.933000	131.735000	6.891000	2010.000000	4.000000
50%	23.000000	9.607460e+05	0.000000	62.670000	3.445000	182.616521	7.874000	2011.000000	6.000000
75%	34.000000	1.420159e+06	0.000000	74.940000	3.735000	212.743293	8.622000	2012.000000	9.000000
max	45.000000	3.818686e+06	1.000000	100.140000	4.468000	227.232807	14.313000	2012.000000	12.000000

In [58]:
```
coefficent_of_variance=(5.643666e+05/1.046965e+06)*100
print(coefficent_of_variance,'%')
```

```
53.90501115128012 %
```

In [59]:
```
df_Store14=df.loc[df['Store'] == 14]
```

In [60]:
```
df_Store14
```

Out[60]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month
1859	14	2010-05-02	2623469.95	0	27.31	2.784	181.871190	8.992	2010	5
1860	14	2010-12-02	1704218.84	1	27.73	2.773	181.982317	8.992	2010	12
1861	14	2010-02-19	2204556.70	0	31.27	2.745	182.034782	8.992	2010	2
1862	14	2010-02-26	2095591.63	0	34.89	2.754	182.077469	8.992	2010	2
1863	14	2010-05-03	2237544.75	0	37.13	2.777	182.120157	8.992	2010	5
...	...	...	...	...	...	...	...	...	...	...
1997	14	2012-09-28	1522512.20	0	64.88	3.997	192.013558	8.684	2012	9
1998	14	2012-05-10	1687592.16	0	64.89	3.985	192.170412	8.667	2012	5
1999	14	2012-12-10	1639585.61	0	54.47	4.000	192.327265	8.667	2012	12
2000	14	2012-10-19	1590274.72	0	56.47	3.969	192.330854	8.667	2012	10
2001	14	2012-10-26	1704357.62	0	58.85	3.882	192.308899	8.667	2012	10

143 rows × 10 columns

In [122…

```
#coefficent of varaince for store 14 =
df_Store14.describe()
```

Out[122]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month
count	143.0	1.430000e+02	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000	143.000000
mean	14.0	2.020978e+06	0.069930	57.790979	3.417608	186.285678	8.648748	2010.965035	6.475524
std	0.0	3.175699e+05	0.255926	16.271612	0.443029	3.594820	0.151460	0.799759	3.333213
min	14.0	1.479515e+06	0.000000	24.050000	2.699000	181.646815	8.424000	2010.000000	1.000000
25%	14.0	1.873298e+06	0.000000	45.585000	2.921000	182.619515	8.523000	2010.000000	4.000000
50%	14.0	2.004330e+06	0.000000	58.850000	3.541000	185.937438	8.625000	2011.000000	6.000000
75%	14.0	2.125780e+06	0.000000	72.585000	3.809000	189.924736	8.724000	2012.000000	9.000000
max	14.0	3.818686e+06	1.000000	82.990000	4.066000	192.330854	8.992000	2012.000000	12.000000

In [38]:

```
coefficent_of_varaince_store14 = (3.175699e+05/2.020978e+06)*100
print(coefficent_of_varaince_store14,'%')
```

```
15.71367427057593 %
```

Which store/s has good quarterly growth rate in Q3'2012

In [39]:

```
df.head()
```

Out[39]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	8.106
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	8.106
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	8.106
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	8.106
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	8.106

In [40]:

```
df['Date']=pd.to_datetime(df['Date'])
```

```
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '19-02-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '26-02-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '19-03-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '26-03-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '16-04-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '23-04-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '30-04-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '14-05-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
```

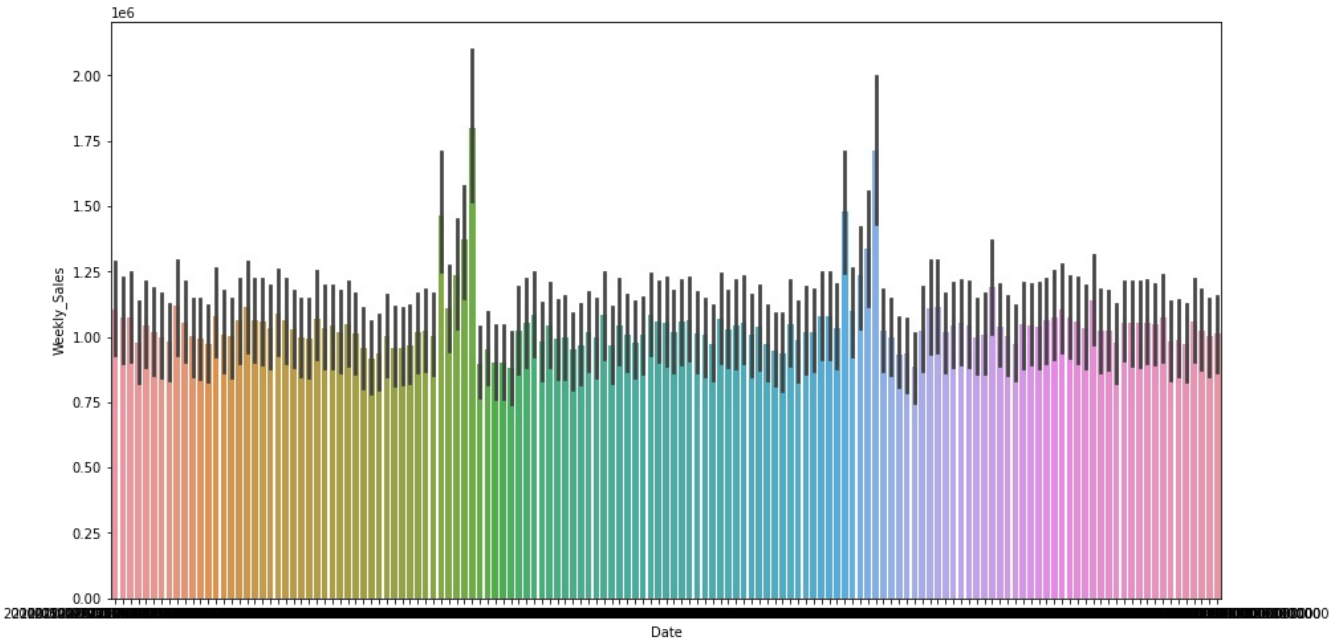
```
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '21-05-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '28-05-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '18-06-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '25-06-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '16-07-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '23-07-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '30-07-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '13-08-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '20-08-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '27-08-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '17-09-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '24-09-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '15-10-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '22-10-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '29-10-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '19-11-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '26-11-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '17-12-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '24-12-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '31-12-2010
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '14-01-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '21-01-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '28-01-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '18-02-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '25-02-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '18-03-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '25-03-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '15-04-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '22-04-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '29-04-2011
```

```
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '13-05-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '20-05-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '27-05-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '17-06-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '24-06-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '15-07-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '22-07-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '29-07-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '19-08-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '26-08-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '16-09-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '23-09-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '30-09-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '14-10-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '21-10-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '28-10-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '18-11-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '25-11-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '16-12-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '23-12-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '30-12-2011
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '13-01-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '20-01-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '27-01-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '17-02-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '24-02-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '16-03-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '23-03-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '30-03-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
```

```
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '13-04-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '20-04-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '27-04-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '18-05-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '25-05-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '15-06-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '22-06-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '29-06-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '13-07-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '20-07-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '27-07-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '17-08-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '24-08-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '31-08-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '14-09-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '21-09-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '28-09-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '19-10-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell\anaconda3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning: Parsing '26-10-2012
' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
```

In [127…

```
plt.figure(figsize = (15,8))
ax = sns.barplot(x="Date", y="Weekly_Sales", data=df)
```



In [41]:

```
df['Date'].unique()
```

In [41]:
```
df['Date'].unique()
```

Out[41]:
```
array(['2010-05-02T00:00:00.000000000', '2010-12-02T00:00:00.000000000',
       '2010-02-19T00:00:00.000000000', '2010-02-26T00:00:00.000000000',
       '2010-05-03T00:00:00.000000000', '2010-12-03T00:00:00.000000000',
       '2010-03-19T00:00:00.000000000', '2010-03-26T00:00:00.000000000',
       '2010-02-04T00:00:00.000000000', '2010-09-04T00:00:00.000000000',
       '2010-04-16T00:00:00.000000000', '2010-04-23T00:00:00.000000000',
       '2010-04-30T00:00:00.000000000', '2010-07-05T00:00:00.000000000',
       '2010-05-14T00:00:00.000000000', '2010-05-21T00:00:00.000000000',
       '2010-05-28T00:00:00.000000000', '2010-04-06T00:00:00.000000000',
       '2010-11-06T00:00:00.000000000', '2010-06-18T00:00:00.000000000',
       '2010-06-25T00:00:00.000000000', '2010-02-07T00:00:00.000000000',
       '2010-09-07T00:00:00.000000000', '2010-07-16T00:00:00.000000000',
       '2010-07-23T00:00:00.000000000', '2010-07-30T00:00:00.000000000',
       '2010-06-08T00:00:00.000000000', '2010-08-13T00:00:00.000000000',
       '2010-08-20T00:00:00.000000000', '2010-08-27T00:00:00.000000000',
       '2010-03-09T00:00:00.000000000', '2010-10-09T00:00:00.000000000',
       '2010-09-17T00:00:00.000000000', '2010-09-24T00:00:00.000000000',
       '2010-01-10T00:00:00.000000000', '2010-08-10T00:00:00.000000000',
       '2010-10-15T00:00:00.000000000', '2010-10-22T00:00:00.000000000',
       '2010-10-29T00:00:00.000000000', '2010-05-11T00:00:00.000000000',
       '2010-12-11T00:00:00.000000000', '2010-11-19T00:00:00.000000000',
       '2010-11-26T00:00:00.000000000', '2010-03-12T00:00:00.000000000',
       '2010-10-12T00:00:00.000000000', '2010-12-17T00:00:00.000000000',
       '2010-12-24T00:00:00.000000000', '2010-12-31T00:00:00.000000000',
       '2011-07-01T00:00:00.000000000', '2011-01-14T00:00:00.000000000',
       '2011-01-21T00:00:00.000000000', '2011-01-28T00:00:00.000000000',
       '2011-04-02T00:00:00.000000000', '2011-11-02T00:00:00.000000000',
       '2011-02-18T00:00:00.000000000', '2011-02-25T00:00:00.000000000',
       '2011-04-03T00:00:00.000000000', '2011-11-03T00:00:00.000000000',
       '2011-03-18T00:00:00.000000000', '2011-03-25T00:00:00.000000000',
       '2011-01-04T00:00:00.000000000', '2011-08-04T00:00:00.000000000',
       '2011-04-15T00:00:00.000000000', '2011-04-22T00:00:00.000000000',
       '2011-04-29T00:00:00.000000000', '2011-06-05T00:00:00.000000000',
       '2011-05-13T00:00:00.000000000', '2011-05-20T00:00:00.000000000',
       '2011-05-27T00:00:00.000000000', '2011-03-06T00:00:00.000000000',
       '2011-10-06T00:00:00.000000000', '2011-06-17T00:00:00.000000000',
       '2011-06-24T00:00:00.000000000', '2011-01-07T00:00:00.000000000',
       '2011-08-07T00:00:00.000000000', '2011-07-15T00:00:00.000000000',
       '2011-07-22T00:00:00.000000000', '2011-07-29T00:00:00.000000000',
       '2011-05-08T00:00:00.000000000', '2011-12-08T00:00:00.000000000',
       '2011-08-19T00:00:00.000000000', '2011-08-26T00:00:00.000000000',
       '2011-02-09T00:00:00.000000000', '2011-09-09T00:00:00.000000000',
       '2011-09-16T00:00:00.000000000', '2011-09-23T00:00:00.000000000',
       '2011-09-30T00:00:00.000000000', '2011-07-10T00:00:00.000000000',
       '2011-10-14T00:00:00.000000000', '2011-10-21T00:00:00.000000000',
       '2011-10-28T00:00:00.000000000', '2011-04-11T00:00:00.000000000',
       '2011-11-11T00:00:00.000000000', '2011-11-18T00:00:00.000000000',
       '2011-11-25T00:00:00.000000000', '2011-02-12T00:00:00.000000000',
       '2011-09-12T00:00:00.000000000', '2011-12-16T00:00:00.000000000',
       '2011-12-23T00:00:00.000000000', '2011-12-30T00:00:00.000000000',
       '2012-06-01T00:00:00.000000000', '2012-01-13T00:00:00.000000000',
       '2012-01-20T00:00:00.000000000', '2012-01-27T00:00:00.000000000',
       '2012-03-02T00:00:00.000000000', '2012-10-02T00:00:00.000000000',
       '2012-02-17T00:00:00.000000000', '2012-02-24T00:00:00.000000000',
       '2012-02-03T00:00:00.000000000', '2012-09-03T00:00:00.000000000',
       '2012-03-16T00:00:00.000000000', '2012-03-23T00:00:00.000000000',
       '2012-03-30T00:00:00.000000000', '2012-06-04T00:00:00.000000000',
       '2012-04-13T00:00:00.000000000', '2012-04-20T00:00:00.000000000',
       '2012-04-27T00:00:00.000000000', '2012-04-05T00:00:00.000000000',
       '2012-11-05T00:00:00.000000000', '2012-05-18T00:00:00.000000000',
       '2012-05-25T00:00:00.000000000', '2012-01-06T00:00:00.000000000',
       '2012-08-06T00:00:00.000000000', '2012-06-15T00:00:00.000000000',
       '2012-06-22T00:00:00.000000000', '2012-06-29T00:00:00.000000000',
       '2012-06-07T00:00:00.000000000', '2012-07-13T00:00:00.000000000',
       '2012-07-20T00:00:00.000000000', '2012-07-27T00:00:00.000000000',
       '2012-03-08T00:00:00.000000000', '2012-10-08T00:00:00.000000000',
       '2012-08-17T00:00:00.000000000', '2012-08-24T00:00:00.000000000',
       '2012-08-31T00:00:00.000000000', '2012-07-09T00:00:00.000000000',
       '2012-09-14T00:00:00.000000000', '2012-09-21T00:00:00.000000000',
       '2012-09-28T00:00:00.000000000', '2012-05-10T00:00:00.000000000',
       '2012-12-10T00:00:00.000000000', '2012-10-19T00:00:00.000000000',
       '2012-10-26T00:00:00.000000000'], dtype='datetime64[ns]')
```

In [42]:
```
df.Date.dt.strftime('%d-%m-%y').unique()
```

Out[42]:
```
array(['02-05-10', '02-12-10', '19-02-10', '26-02-10', '03-05-10',
       '03-12-10', '19-03-10', '26-03-10', '04-02-10', '04-09-10',
       '16-04-10', '23-04-10', '30-04-10', '05-07-10', '14-05-10',
       '21-05-10', '28-05-10', '06-04-10', '06-11-10', '18-06-10',
       '25-06-10', '07-02-10', '07-09-10', '16-07-10', '23-07-10',
       '30-07-10', '08-06-10', '13-08-10', '20-08-10', '27-08-10',
       '09-03-10', '09-10-10', '17-09-10', '24-09-10', '10-01-10',
       '10-08-10', '15-10-10', '22-10-10', '29-10-10', '11-05-10',
       '11-12-10', '19-11-10', '26-11-10', '12-03-10', '12-10-10',
       '17-12-10', '24-12-10', '31-12-10', '01-07-11', '14-01-11',
       '21-01-11', '28-01-11', '02-04-11', '02-11-11', '18-02-11',
       '25-02-11', '03-04-11', '03-11-11', '18-03-11', '25-03-11',
       '04-01-11', '04-08-11', '15-04-11', '22-04-11', '29-04-11',
       '05-06-11', '13-05-11', '20-05-11', '27-05-11', '06-03-11',
       '06-10-11', '17-06-11', '24-06-11', '07-01-11', '07-08-11',
       '15-07-11', '22-07-11', '29-07-11', '08-05-11', '08-12-11',
       '19-08-11', '26-08-11', '09-02-11', '09-09-11', '16-09-11',
       '23-09-11', '30-09-11', '10-07-11', '14-10-11', '21-10-11',
       '28-10-11', '11-04-11', '11-11-11', '18-11-11', '25-11-11',
       '12-02-11', '12-09-11', '16-12-11', '23-12-11', '30-12-11',
       '01-06-12', '13-01-12', '20-01-12', '27-01-12', '02-03-12',
       '02-10-12', '17-02-12', '24-02-12', '03-02-12', '03-09-12',
       '16-03-12', '23-03-12', '30-03-12', '04-06-12', '13-04-12',
       '20-04-12', '27-04-12', '05-04-12', '05-11-12', '18-05-12',
       '25-05-12', '06-01-12', '06-08-12', '15-06-12', '22-06-12',
       '29-06-12', '07-06-12', '13-07-12', '20-07-12', '27-07-12',
       '08-03-12', '08-10-12', '17-08-12', '24-08-12', '31-08-12',
       '09-07-12', '14-09-12', '21-09-12', '28-09-12', '10-05-12',
       '10-12-12', '19-10-12', '26-10-12'], dtype=object)
```

In [101…
```
df['year'] = pd.DatetimeIndex(df['Date']).year
df['month'] = pd.DatetimeIndex(df['Date']).month
```

In [102…
```
df.head()
```

Out[102]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.106	2010	5	1
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.106	2010	2	1
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.106	2010	2	1
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.106	2010	5	1

In [135…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6435 entries, 0 to 6434
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype         
---  ------        --------------  -----         
 0   Store         6435 non-null   int64         
 1   Date          6435 non-null   datetime64[ns]
 2   Weekly_Sales  6435 non-null   float64       
 3   Holiday_Flag  6435 non-null   int64         
 4   Temperature   6435 non-null   float64       
 5   Fuel_Price    6435 non-null   float64       
 6   CPI           6435 non-null   float64       
 7   Unemployment  6435 non-null   float64       
 8   year          6435 non-null   int64         
 9   month         6435 non-null   int64         
 10  semester      6435 non-null   int32         
dtypes: datetime64[ns](1), float64(5), int32(1), int64(4)
memory usage: 528.0 KB
```

In [149…
```
#change dates into days by creating new variables
day=df.Date.dt.strftime('%d').unique()
day
```

Out[149]:
```
array(['02', '19', '26', '03', '04', '16', '23', '30', '05', '14', '21',
       '28', '06', '18', '25', '07', '08', '13', '20', '27', '09', '17',
       '24', '10', '15', '22', '29', '11', '12', '31', '01'], dtype=object)
```

In [134…
```
month=df.Date.dt.strftime('%m').unique()
month
year=df.Date.dt.strftime('%y').unique()
year
```

Out[134]:
```
array(['10', '11', '12'], dtype=object)
```

In [107…
```
df['Date'].dt.month
```

Out[107]:
```
0        5
1       12
2        2
3        2
4        5
        ..
6430     9
6431     5
6432    12
6433    10
6434    10
Name: Date, Length: 6435, dtype: int64
```

In [108…
```
df_2012=df.loc[df['year'] == 2012]
df_2012.head()
```

Out[108]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
100	1	2012-06-01	1550369.92	0	49.01	3.157	219.714258	7.348	2012	6	1
101	1	2012-01-13	1459601.17	0	48.53	3.261	219.892526	7.348	2012	1	1
102	1	2012-01-20	1394393.84	0	54.11	3.268	219.985689	7.348	2012	1	1
103	1	2012-01-27	1319325.59	0	54.26	3.290	220.078852	7.348	2012	1	1
104	1	2012-03-02	1636339.65	0	56.55	3.360	220.172015	7.348	2012	3	1

In [140…
```
#Qtr1=month 1,2,3
#Qtr2=month 4,5,6
#Qtr3=month 7,8,9
#Qtr4=month 10,11,12
```

In [110…
```
df_2012_Q3 = df_2012.loc[(df_2012['month']>6) &(df_2012['month']<10)]
```

In [111…
```
df_2012_Q3
```

Out[111]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
109	1	2012-09-03	1675431.16	0	58.76	3.669	221.059189	7.348	2012	9	2
122	1	2012-08-06	1697230.96	0	78.30	3.452	221.749484	7.143	2012	8	2
127	1	2012-07-13	1527014.04	0	77.12	3.256	221.924158	6.908	2012	7	2
128	1	2012-07-20	1497954.76	0	80.42	3.311	221.932727	6.908	2012	7	2
129	1	2012-07-27	1439123.71	0	82.66	3.407	221.941295	6.908	2012	7	2
...	...	...	...	...	...	...	...	...	...	...	...
6426	45	2012-08-31	734297.87	0	75.09	3.867	191.461281	8.684	2012	8	2
6427	45	2012-07-09	766512.66	1	75.70	3.911	191.577676	8.684	2012	7	2
6428	45	2012-09-14	702238.27	0	67.87	3.948	191.699850	8.684	2012	9	2
6429	45	2012-09-21	723086.20	0	65.32	4.038	191.856704	8.684	2012	9	2
6430	45	2012-09-28	713173.95	0	64.88	3.997	192.013558	8.684	2012	9	2

540 rows × 11 columns

In [112…
```
max_Sales_2012Q3=df_2012_Q3.groupby('Store')['Weekly_Sales'].sum()
max_Sales_2012Q3
```

Out[112]:
```
Store
1     18633209.98
2     22396867.61
3      4966495.93
4     25652119.35
5      3880621.88
6     18341221.11
7      7322393.92
8     10873860.34
9      6528239.56
10    21169356.45
11    16094363.07
12    11777508.50
13    24319994.35
14    20140430.40
15     6909374.37
16     6441311.11
17    11533998.38
18    12507521.72
19    16644341.31
20    24665938.11
21     8403507.99
22    11818544.33
23    17103654.36
24    16125999.86
25     8309440.44
26    12417575.35
27    20191238.11
28    15055659.67
29     6127862.07
30     5181974.44
31    16454328.46
32    14142164.84
33     3177072.43
34    11476258.98
35    10252122.68
36     3578123.58
37     6250524.08
38     5129297.64
39    18899955.17
40    11647661.37
41    16373588.44
42     6830839.86
43     7376726.03
44     4020486.01
45     8851242.32
Name: Weekly_Sales, dtype: float64
```

In [113…
```
max_Sales_2012Q3=df_2012_Q3.groupby('Store')['Weekly_Sales'].sum()
max_Sales_2012Q3.idxmax()
```

Out[113]:
```
4
```

Some holidays have a negative impact on sales. Find out holidays which have higher sales than the mean sales in non-holiday season for all stores together

In [114…
```
df.head()
```

Out[114]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.106	2010	5	1
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.106	2010	2	1
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.106	2010	2	1
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.106	2010	5	1

In [115…
```
df_wallmart_holiday=df.groupby('Holiday_Flag')
df_holidays=df_wallmart_holiday.get_group(1)
df_nonholidays=df_wallmart_holiday.get_group(0)
mean_nonholiday_sales=df_nonholidays["Weekly_Sales"].mean()
print(mean_nonholiday_sales)
df_holidays[df_holidays['Weekly_Sales']>mean_nonholiday_sales]
```

```
1041256.3802088564
```

Out[115]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2
31	1	2010-10-09	1507460.69	1	78.69	2.565	211.495190	7.787	2010	10	2
42	1	2010-11-26	1955624.11	1	64.52	2.735	211.748433	7.838	2010	11	2
47	1	2010-12-31	1367320.01	1	48.43	2.943	211.404932	7.838	2010	12	2
53	1	2011-11-02	1649614.93	1	36.39	3.022	212.936705	7.742	2011	11	2
...	...	...	...	...	...	...	...	...	...	...	...
5819	41	2011-12-30	1264014.16	1	34.12	3.119	196.358610	6.759	2011	12	2
5825	41	2012-10-02	1238844.56	1	22.00	3.103	196.919506	6.589	2012	10	2
5855	41	2012-07-09	1392143.82	1	67.41	3.596	198.095048	6.432	2012	7	2
6334	45	2010-11-26	1182500.16	1	46.15	3.039	182.783277	8.724	2010	11	2
6386	45	2011-11-25	1170672.94	1	48.71	3.492	188.350400	8.523	2011	11	2

220 rows × 11 columns

In [116…
```
df.loc[(df['Holiday_Flag']==0)].Weekly_Sales.mean() #mean of sales when non holiday
```

Out[116]:
```
1041256.3802088564
```

In [117…
```
result=df[(df['Weekly_Sales']>1041256.38)&(df['Holiday_Flag']==1)]
#list of holidays where sales . (mean of sales when non holiday)
```

In [118…
```
result["Date"].unique
```

Out[118]:
```
<bound method Series.unique of 1      2010-12-02
31     2010-10-09
42     2010-11-26
47     2010-12-31
53     2011-11-02
          ...
5819   2011-12-30
5825   2012-10-02
5855   2012-07-09
6334   2010-11-26
6386   2011-11-25
Name: Date, Length: 220, dtype: datetime64[ns]>
```

In [119…
```
df.groupby(df.Holiday_Flag).mean()
```

Out[119]:

Holiday_Flag	Store	Weekly_Sales	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
0	23.0	1.041256e+06	61.448124	3.368467	171.601725	7.993514	2010.977444	6.172932	1.458647
1	23.0	1.122888e+06	50.232044	3.227464	171.268092	8.074127	2010.800000	10.500000	2.000000

In [120…
```
df.groupby(df.semester).mean()
df.groupby(df.semester).sum()
```

Out[120]:

semester	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month
1	74520	3.327977e+09	0	188817.75	10973.368	555623.872665	25974.570	6515775	11745
2	73485	3.409242e+09	450	201553.69	10639.267	548483.091752	25499.967	6424785	29925

Provide a monthly and semester view of sales in units and give insights

In [121…
```
df.head()
```

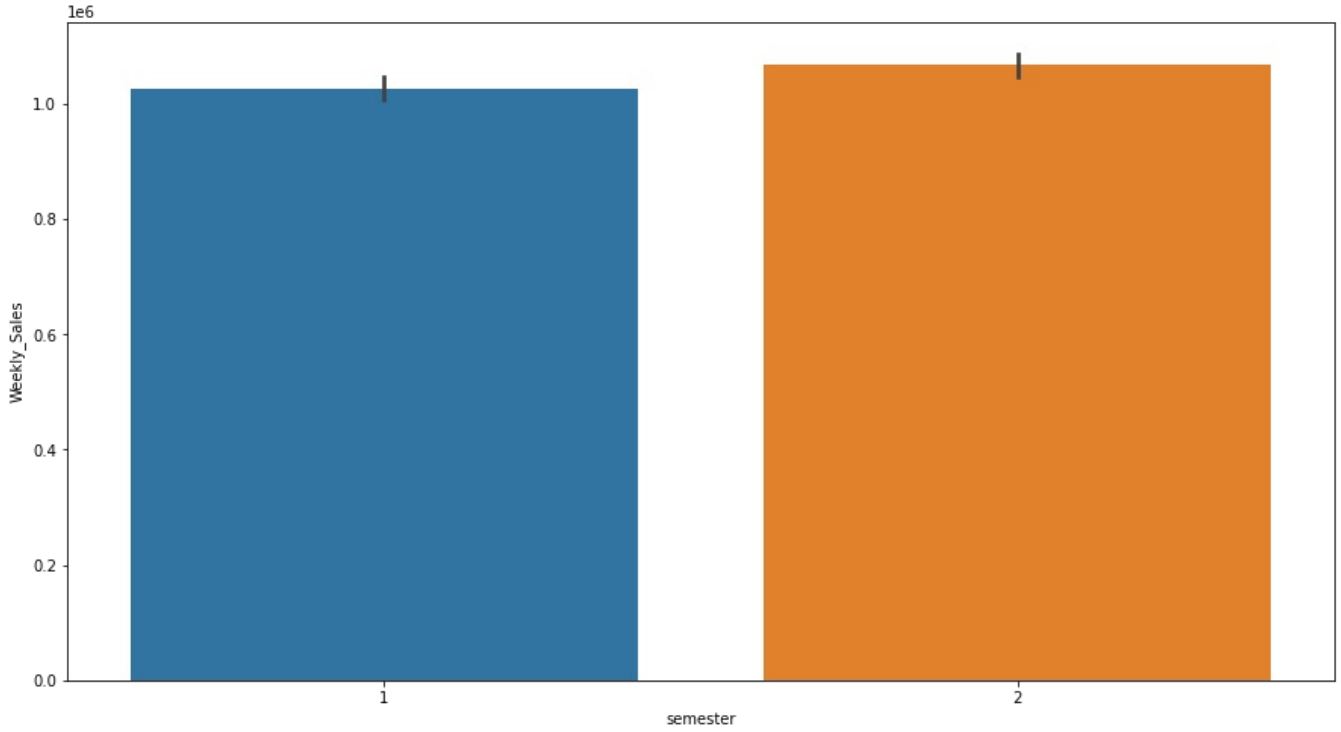
Out[121]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.106	2010	5	1
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.106	2010	2	1
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.106	2010	2	1
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.106	2010	5	1

In [86]:
```
df['semester'] = np.where(df['month']< 7, 1, 2)
# for month 1 to 6 , semester = 1
# for month 7 to 12 , semester = 2
```

In [122…
```
plt.figure(figsize = (15,8))
```

```
ax = sns.barplot(x="semester", y="Weekly_Sales", data=df)
```



for Store - 1 - build prediction models to forecast demand Demand Weekly_Sales - sales foir the given stores

In [137…
```
df.head()
```

Out[137]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.106	2010	5	1
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.106	2010	2	1
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.106	2010	2	1
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.106	2010	5	1

In [176…
```
df1=df.loc[df['Store'] == 1]
```

In [145…
```
df1
```

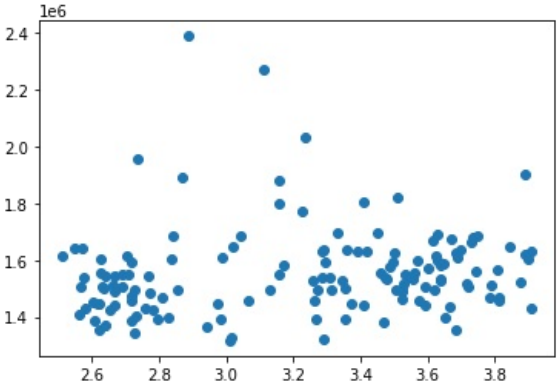
Out[145]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.106	2010	5	1
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.106	2010	2	1
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.106	2010	2	1
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.106	2010	5	1
...	...	...	...	...	...	...	...	...	...	...	...
138	1	2012-09-28	1437059.26	0	76.08	3.666	222.981658	6.908	2012	9	2
139	1	2012-05-10	1670785.97	0	68.55	3.617	223.181477	6.573	2012	5	1
140	1	2012-12-10	1573072.81	0	62.99	3.601	223.381296	6.573	2012	12	2
141	1	2012-10-19	1508068.77	0	67.97	3.594	223.425723	6.573	2012	10	2
142	1	2012-10-26	1493659.74	0	69.16	3.506	223.444251	6.573	2012	10	2

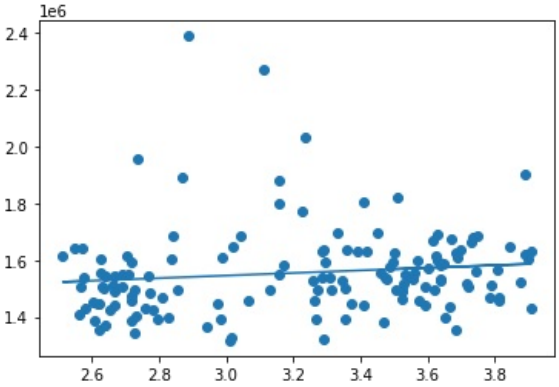
143 rows × 11 columns

In [155…
```
#Fuel_Price - Cost of fuel in the region

#CPI – Prevailing consumer price index

#Unemployment - Prevailing unemployment rate



x = df1['Fuel_Price']
y = df1['Weekly_Sales']

plt.scatter(x,y)
```

```
plt.show()

slope, intercept, r, p, std_err = stats.linregress(x,y)
print(r) # r should be between -1 to 1

def myfunc(x):
    return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x,y)
plt.plot(x, mymodel)
plt.show()
```



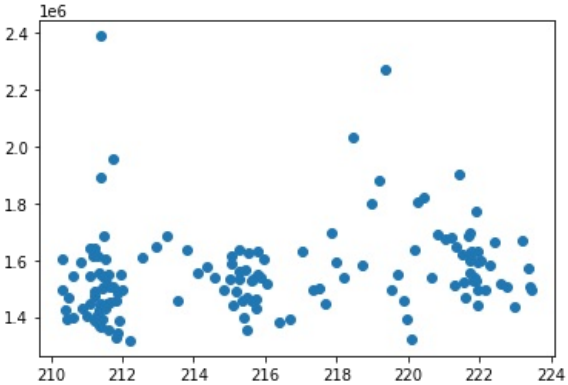

```
0.12459158039045629
```



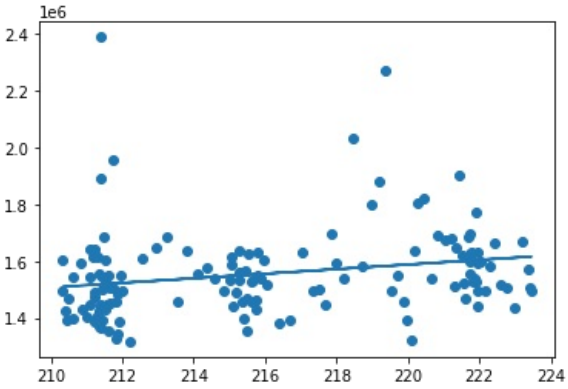
Hypothesize if CPI, Unemployment, and fuel price have any prices on sales.

In [156…

```
x = df1['CPI']
y = df1['Weekly_Sales']

plt.scatter(x,y)
plt.show()

slope, intercept, r, p, std_err = stats.linregress(x,y)
print(r) # r should be between -1 to 1

def myfunc(x):
    return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x,y)
plt.plot(x, mymodel)
plt.show()
```

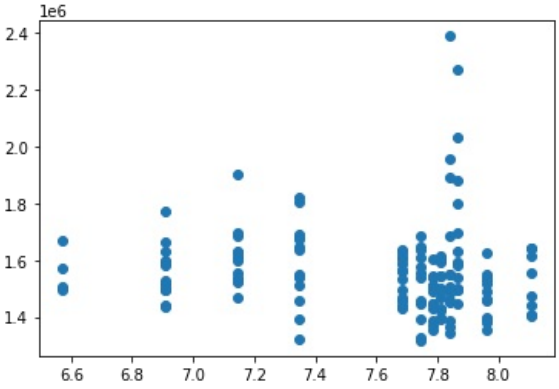



```
0.22540765942904442
```

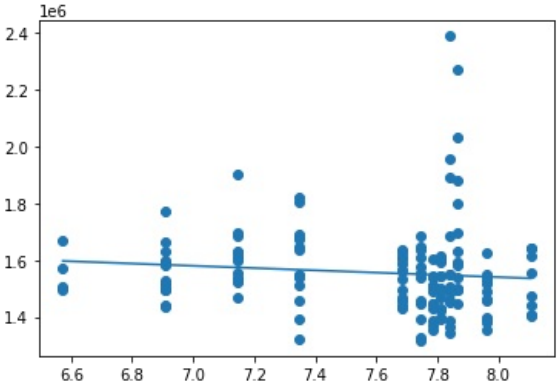


In [157…

```
x = df1['Unemployment']
y = df1['Weekly_Sales']

plt.scatter(x,y)
plt.show()

slope, intercept, r, p, std_err = stats.linregress(x,y)
print(r) # r should be between -1 to 1

def myfunc(x):
    return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x,y)
plt.plot(x, mymodel)
plt.show()
```



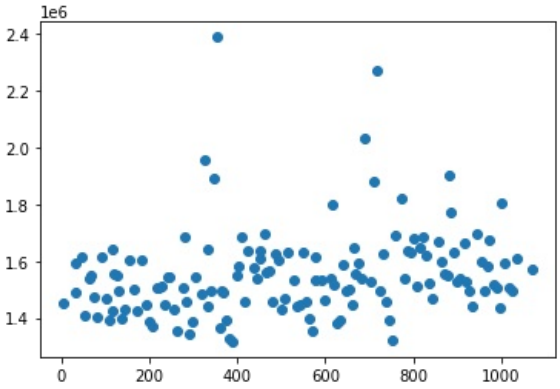

```
-0.09795539472957951
```



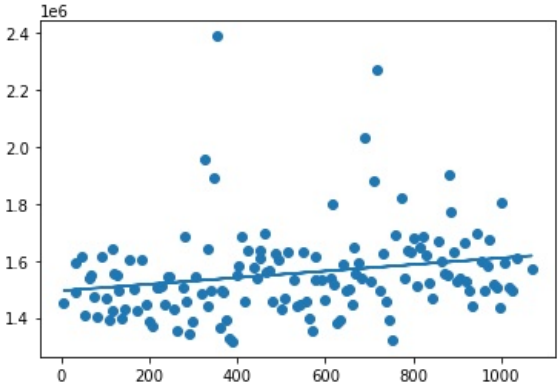
In [192…

```
x = df1['exp_day']
y = df1['Weekly_Sales']

plt.scatter(x,y)
plt.show()

slope, intercept, r, p, std_err = stats.linregress(x,y)
print(r) # r should be between -1 to 1

def myfunc(x):
    return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x,y)
plt.plot(x, mymodel)
```

```
plt.show()
```




```
0.21768075944264673
```



In [183…
```
x = df1[['CPI','Unemployment','exp_day']]
y = df1['Weekly_Sales']
```

In [184…
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

In [185…
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(x_train, y_train)
```

Out[185]:
```
LinearRegression()
```

In [186…
```
y_pred = regressor.predict(x_test)
```

In [187…
```
predicteddf = pd.DataFrame({"Actual": y_test, "predicted": y_pred})
predicteddf
```

Out[187]:

	Actual	predicted
45	1891034.93	1.505937e+06
118	1611096.05	1.583746e+06
16	1432069.95	1.490733e+06
56	1636263.41	1.528677e+06
22	1546074.18	1.494492e+06
7	1404429.92	1.517969e+06
108	1688420.76	1.583840e+06
134	1582083.40	1.566498e+06
130	1631135.79	1.556742e+06
101	1459601.17	1.571989e+06
94	2033320.66	1.602110e+06
127	1527014.04	1.560477e+06
8	1594968.28	1.486249e+06
96	1799682.38	1.605542e+06
140	1573072.81	1.551027e+06
33	1351791.03	1.500753e+06
84	1514259.78	1.580446e+06
120	1555444.55	1.578760e+06
119	1595901.87	1.578511e+06
24	1385065.20	1.494513e+06
63	1564819.81	1.542834e+06
86	1394561.83	1.588727e+06
60	1495064.75	1.532113e+06
26	1605491.78	1.496237e+06
62	1559889.00	1.540645e+06
18	1542561.09	1.502428e+06
113	1899676.88	1.575457e+06
106	1819870.00	1.579342e+06
44	1682614.26	1.504549e+06

In [188…
```
coeff_df= pd.DataFrame(regressor.coef_, x.columns, columns=['coefficent'])
coeff_df
```

Out[188]:

	coefficent
CPI	11704.068015
Unemployment	93350.098776
exp_day	31.781348

In [189…
```
from sklearn import metrics
print("Mean Absolute Error:", metrics.mean_absolute_error(y_test, y_pred))
print("Mean Squred Error:", metrics.mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error:", np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 116216.95169902028
Mean Squred Error: 25295466668.184544
Root Mean Squared Error: 159045.48616098647
```

In [190…
```
experiment_day_start=5
df1['Date'] = pd.to_datetime(df1['Date'], dayfirst=True)
df1['exp_day'] = (df1['Date']-df1['Date'].min()).dt.days + experiment_day_start
```

```
C:\Users\Dell\AppData\Local\Temp\ipykernel_10976\2528473243.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  df1['Date'] = pd.to_datetime(df1['Date'], dayfirst=True)
C:\Users\Dell\AppData\Local\Temp\ipykernel_10976\2528473243.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  df1['exp_day'] = (df1['Date']-df1['Date'].min()).dt.days + experiment_day_start
```

In [191…
```
df.head()
```

Out[191]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment	year	month	semester	exp_day
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.106	2010	5	1	117
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.106	2010	12	2	331
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.106	2010	2	1	45
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.106	2010	2	1	52
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.106	2010	5	1	118

In [ ]:
```

```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js