🎯 Project Purpose

Use the data to perform analysis and draw out useful insightes about walmart sales.

import the required

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
import numpy as np
```

Exploratory data analysis and preprocessing

In [2]:

```
df = pd.read_csv("walmart-sales-dataset-of-45stores.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	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 [4]:

```
df.tail()
```

Out[4]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
6430	45	28-09-2012	713173.95	0	64.88	3.997	192.013558	8.684
6431	45	05-10-2012	733455.07	0	64.89	3.985	192.170412	8.667
6432	45	12-10-2012	734464.36	0	54.47	4.000	192.327265	8.667
6433	45	19-10-2012	718125.53	0	56.47	3.969	192.330854	8.667
6434	45	26-10-2012	760281.43	0	58.85	3.882	192.308899	8.667

In [5]:

```
df.shape
```

Out[5]:

```
(6435, 8)
```

In [6]:

```
df.isna().sum().to_frame()
```

Out[6]:

	0
Store	0
Date	0
Weekly_Sales	0
Holiday_Flag	0
Temperature	0
Fuel_Price	0
CPI	0
Unemployment	0

In [7]:

```
df.duplicated().sum()
```

Out[7]:

```
0
```

In [8]:

```
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
```

Date dtype -> odject , We need now to change it into date dtype

```
In [9]:  df['Date']=pd.to_datetime(df['Date'],format='%d-%m-%Y')
```

```
In [10]: # we will check data types now :
         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   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       
dtypes: datetime64[ns](1), float64(5), int64(2)
memory usage: 402.3 KB
```

```
In [11]: df.describe().T
```

Out[11]:

	count	mean	min	25%	50%	75%	max	std
Store	6435.0	23.0	1.0	12.0	23.0	34.0	45.0	12.988182
Date	6435	2011-06-17 00:00:00	2010-02-05 00:00:00	2010-10-08 00:00:00	2011-06-17 00:00:00	2012-02-24 00:00:00	2012-10-26 00:00:00	NaN
Weekly_Sales	6435.0	1046964.877562	209986.25	553350.105	960746.04	1420158.66	3818686.45	564366.622054
Holiday_Flag	6435.0	0.06993	0.0	0.0	0.0	0.0	1.0	0.255049
Temperature	6435.0	60.663782	-2.06	47.46	62.67	74.94	100.14	18.444933
Fuel_Price	6435.0	3.358607	2.472	2.933	3.445	3.735	4.468	0.45902
CPI	6435.0	171.578394	126.064	131.735	182.616521	212.743293	227.232807	39.356712
Unemployment	6435.0	7.999151	3.879	6.891	7.874	8.622	14.313	1.875885

The store has maximum sales overall :

In [12]:

```
order = df.groupby('Store')['Weekly_Sales'].sum().reset_index().sort_values(by='Weekly_Sales', ascending=False)
order.head()
```

Out[12]:

	Store	Weekly_Sales
19	20	3.013978e+08
3	4	2.995440e+08
13	14	2.889999e+08
12	13	2.865177e+08
1	2	2.753824e+08

In [13]:

```
#now we want to check thee maximum sales which achived by a store
max_value = order['Weekly_Sales'].max()
max_value
```

Out[13]:

```
301397792.46
```

In [14]:

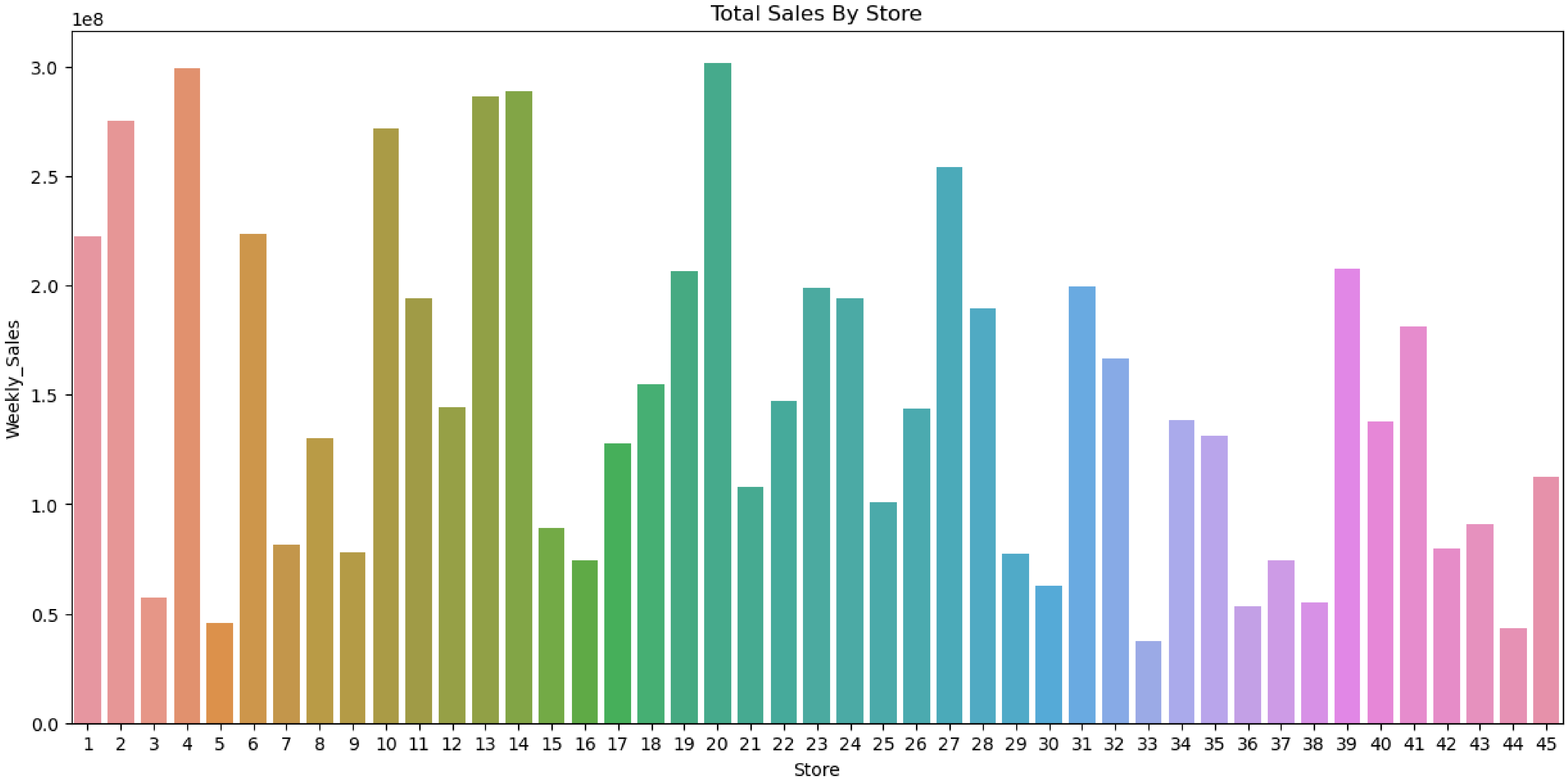
```
shop_with_maxSales = order.loc[order['Weekly_Sales'] == max_value, 'Store'].values[0]
shop_with_maxSales
```

Out[14]:

```
20
```

In [15]:

```
plt.figure(figsize=(15, 7))
barplot = sns.barplot(x='Store', y='Weekly_Sales', data=order)
barplot.set_title('Total Sales By Store')
plt.show()
```



Which store has maximum standard deviation i.e., the sales vary a lot

```
In [16]: std_sales = df.groupby('Store')['Weekly_Sales'].std().reset_index()
         std_sales.rename(columns = {'Weekly_Sales':'Sales Std'}, inplace = True)
         std_sales.head()
```

Out[16]:

	Store	Sales Std
0	1	155980.767761
1	2	237683.694682
2	3	46319.631557
3	4	266201.442297
4	5	37737.965745

In [17]:

```
#now we want to get store with max standered deviation
maxStd = std_sales['Sales Std'].max()
maxStd
```

Out[17]:

```
317569.9494755081
```

In [18]:

```
#which store achived this val
Store_with_maxStd = std_sales.loc[std_sales['Sales Std'].idxmax(), 'Store']
Store_with_maxStd
```

Out[18]:

```
14
```

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 [19]:

```
#Description to ur data : This is the historical data that covers sales from 2010-02-05 to 2012-11-01
super_bowl_dates = ['2010-02-12', '2011-02-11', '2012-02-10']
labour_day_dates = ['2010-09-10', '2011-09-09', '2012-09-07']
thanksgiving_dates = ['2010-11-26', '2011-11-25', '2012-11-23']
christmas_dates = ['2010-12-31', '2011-12-30', '2012-12-28']
```

In [20]:

```
super_bowl = df[df['Date'].isin(super_bowl_dates)]
labour_day = df[df['Date'].isin(labour_day_dates)]
thanksgiving = df[df['Date'].isin(thanksgiving_dates)]
christmas = df[df['Date'].isin(christmas_dates)]
NoHoliday = df[df['Holiday_Flag'] == 0]
```

In [21]:

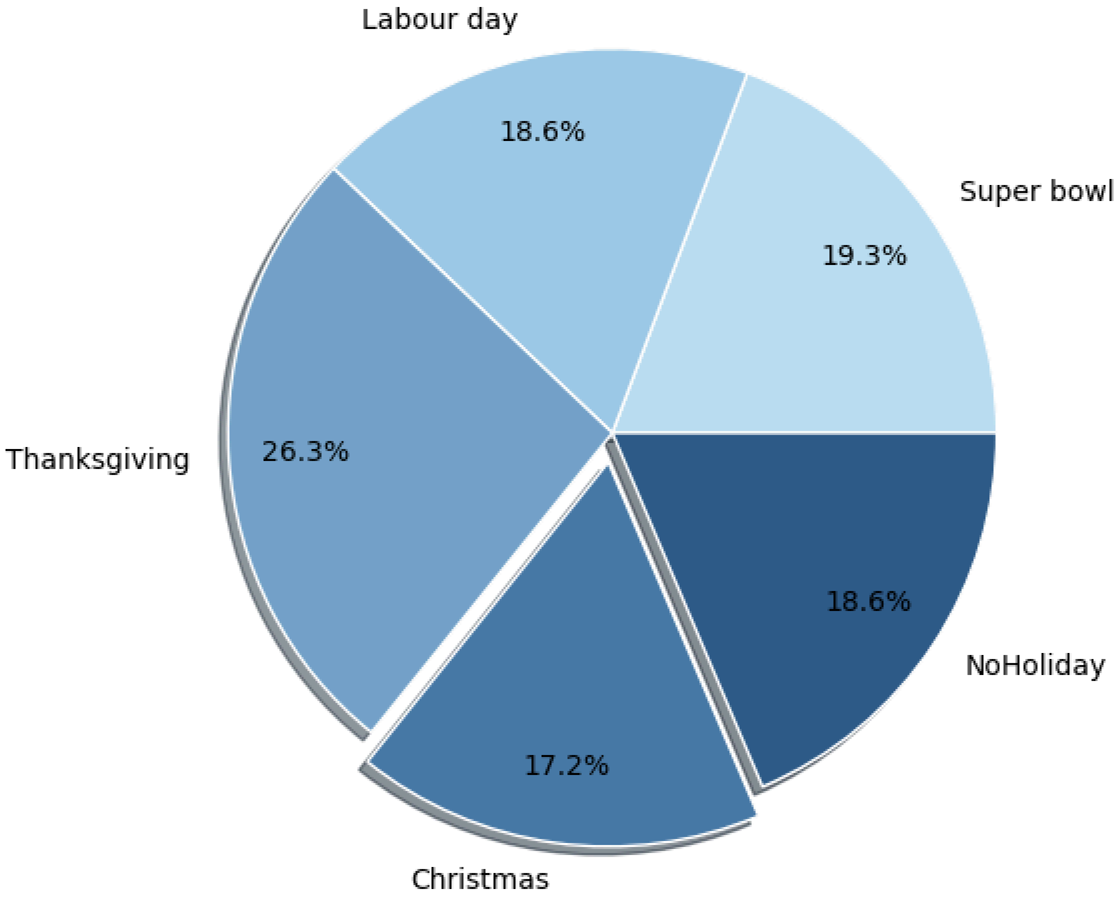
```
# Calculate the mean weekly sales for each holiday and non-holiday period
mean_sales = [super_bowl['Weekly_Sales'].mean(),
              labour_day['Weekly_Sales'].mean(),
              thanksgiving['Weekly_Sales'].mean(),
              christmas['Weekly_Sales'].mean(),
              NoHoliday['Weekly_Sales'].mean()]
```

In [22]:

```
holiday_Labels = ['Super bowl','Labour day','Thanksgiving','Christmas','NoHoliday']
Color = ["#B9DDF1", "#9FCAE6", "#73A4CA", "#497AA7", "#2E5B88"]
fig ,ax = plt.subplots()
ax.pie(mean_sales ,labels = holiday_Labels ,
       radius = 1.3 ,colors = Color , shadow = True ,
       autopct = '%1.1f%%' , pctdistance = 0.8 ,
       explode = [0,0,0,0.1,0] ,wedgeprops ={"linewidth": 1, "edgecolor": "white"})
```

Out[22]:

```
([<matplotlib.patches.Wedge at 0x1af05dc2010>,
  <matplotlib.patches.Wedge at 0x1af05ef8c10>,
  <matplotlib.patches.Wedge at 0x1af05efb5d0>,
  <matplotlib.patches.Wedge at 0x1af05f06250>,
  <matplotlib.patches.Wedge at 0x1af05f0cc50>],
 [Text(1.175413549507108, 0.8144341518103851, 'Super bowl'),
  Text(-0.32101248016988154, 1.3935031351149454, 'Labour day'),
  Text(-1.426783792859552, -0.09585409972093677, 'Thanksgiving'),
  Text(-0.21057247801162604, -1.515440276455672, 'Christmas'),
  Text(1.1924657499171112, -0.789256254504595, 'NoHoliday')],
 [Text(0.8548462178233514, 0.5923157467711891, '19.3%'),
  Text(-0.233463621941732, 1.013456825538142, '18.6%'),
  Text(-1.0376609402614922, -0.06971207252431764, '26.3%'),
  Text(-0.15689714047925074, -1.1291515785355988, '17.2%'),
  Text(0.8672478181215353, -0.5740045487306145, '18.6%')])
```



In [23]:

```
Holidays_meanSales_df = pd.DataFrame({'Holiday': holiday_Labels, 'MeanSales': mean_sales})
Holidays_meanSales_df['MeanSales']=Holidays_meanSales_df['MeanSales'].apply('${:,.2f}'.format)
Holidays_meanSales_df
```

Out[23]:

	Holiday	MeanSales
0	Super bowl	$1,079,127.99
1	Labour day	$1,042,427.29
2	Thanksgiving	$1,471,273.43
3	Christmas	$960,833.11
4	NoHoliday	$1,041,256.38

Our analysis for Weekly_Sales data reveals distinct patterns on sales and customerSpending during Holidays , It shows us that Christmas ,Labour day ,Super bowl contribute positively to mean weekly sales , But Customer spending is increased during holidays with related offers and events as in Thanksgiving day which have the higest_meanWeeklySales,reflecting increase in shopping activites ,for White Friday sales ,hoildays offers preparation.Surprisingly, weeks with NO holidays have High meanWeeklySales, emphasizing continous customer spending patterns out of the holidays.

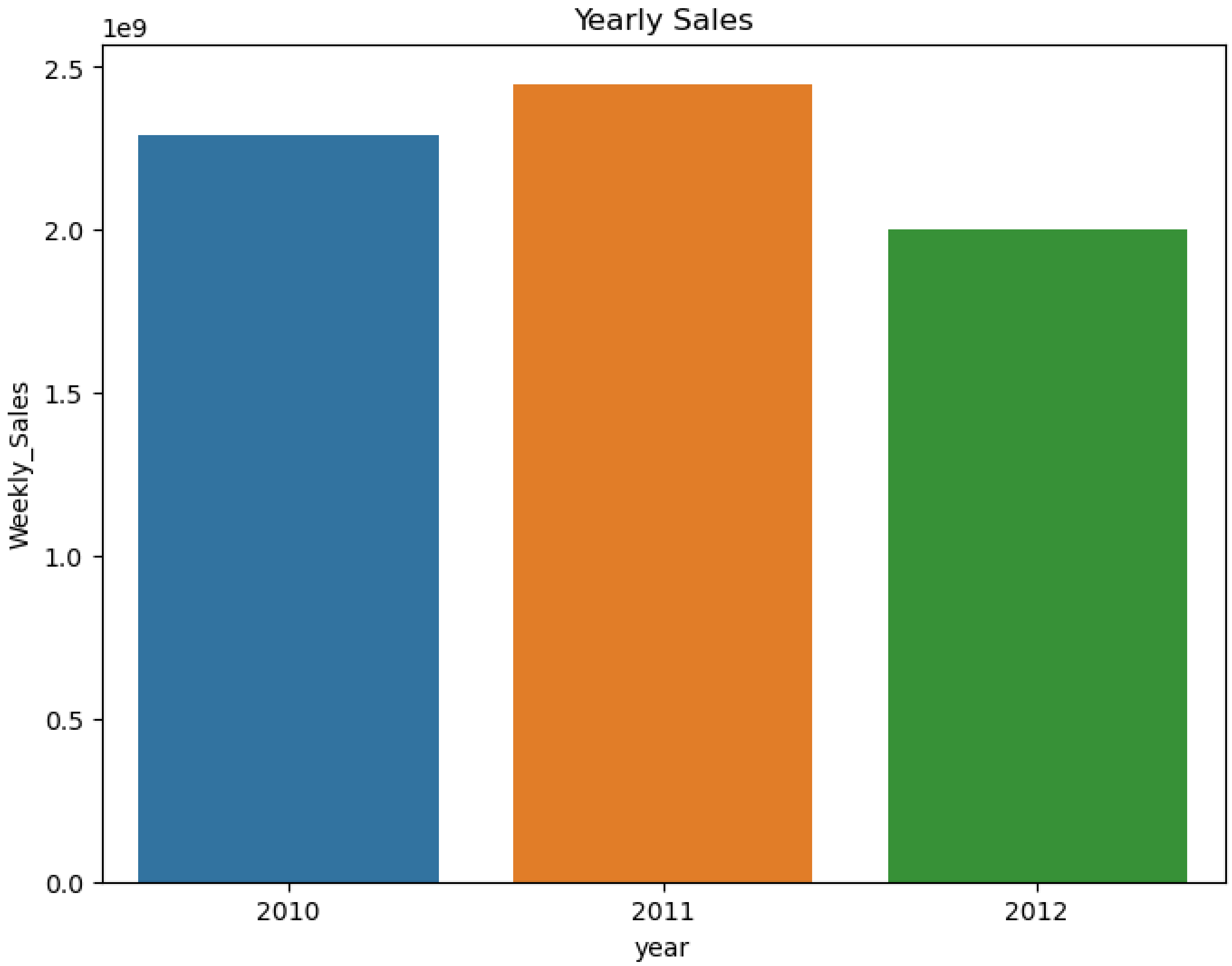
This insights highlight the importance of marketing strategies and inventory management based on observed trends during specific holidays. Understanding these patterns enables stackholders and retailers to optimize their approach to achieve the highest possible sales and enhance thier performance whaich help them in their business expand

a monthly and semester view of sales in units and given insights.

In [24]:

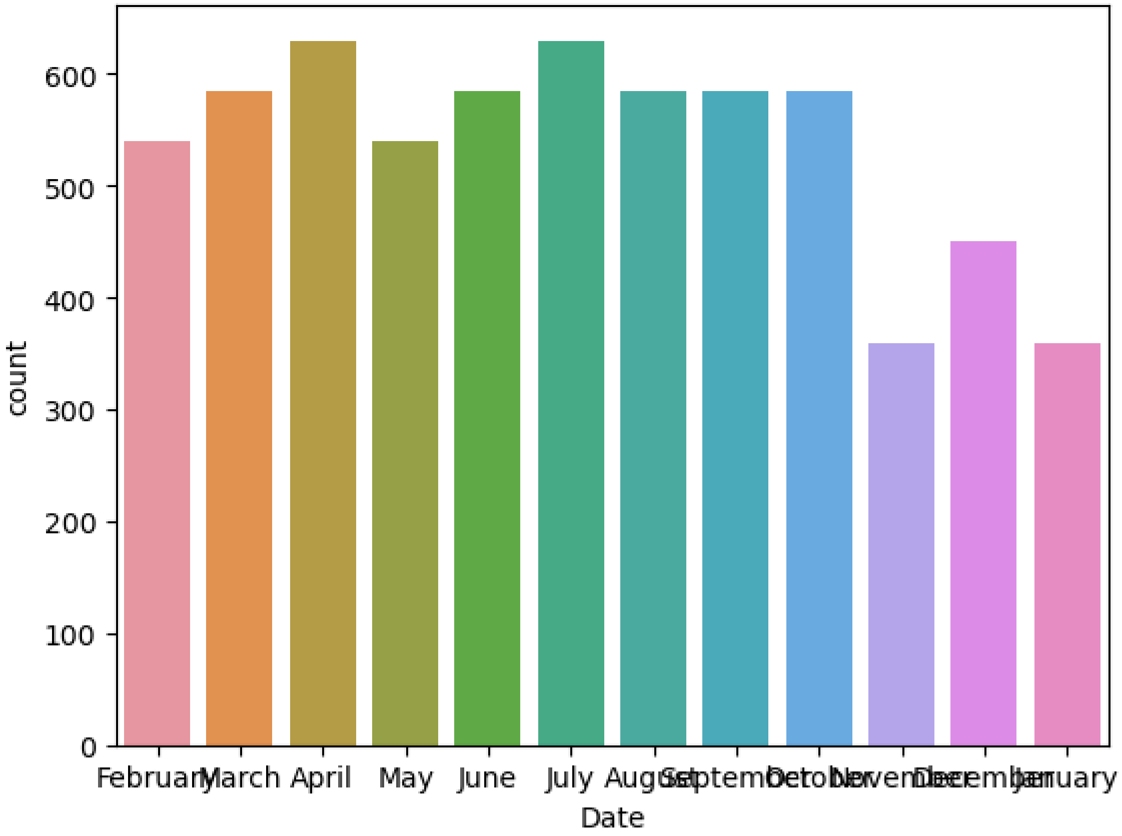
```
df['year'] = df['Date'].dt.year
yearly_sales = df.groupby('year')['Weekly_Sales'].sum().reset_index()
max_sales = yearly_sales['Weekly_Sales'].max()
max_years = yearly_sales[yearly_sales['Weekly_Sales'] == max_sales]['year']

fig, ax = plt.subplots(figsize=(8, 6))
sns.barplot(x='year', y='Weekly_Sales', data=yearly_sales, ax=ax)
plt.title('Yearly Sales')
print('Years with maximum sales:', max_years.values)
print('Maximum Sales:', max_sales)
print('Years Avg Sales:', yearly_sales['Weekly_Sales'].mean())
```

```
Years with maximum sales: [2011]
Maximum Sales: 2448200007.35
Years Avg Sales: 2245739662.3700004
```



In [25]:
```
sns.countplot(x=df['Date'].dt.month_name(),data = df )
```

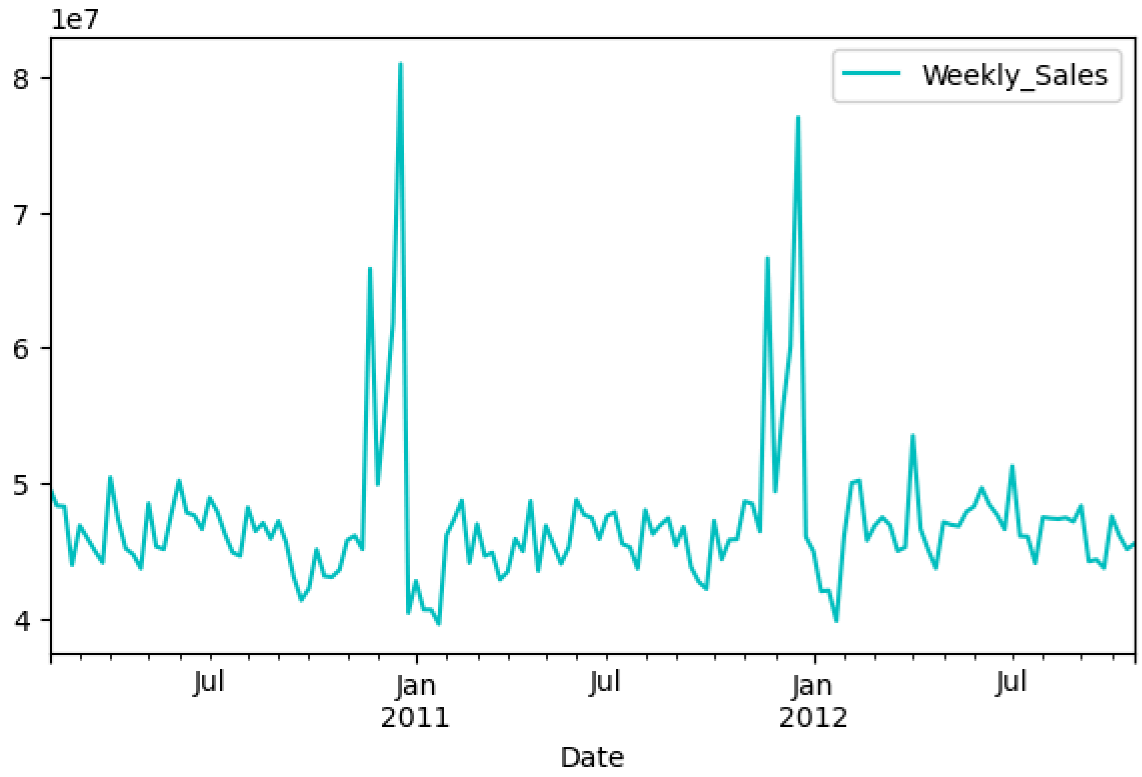
Out[25]:
```
<Axes: xlabel='Date', ylabel='count'>
```



It has been analyzed that july has the maximum times of Sales (purchases)

```
In [26]: df.groupby('Date')[['Weekly_Sales']].sum().plot(color= 'c', figsize= (7,4))
```

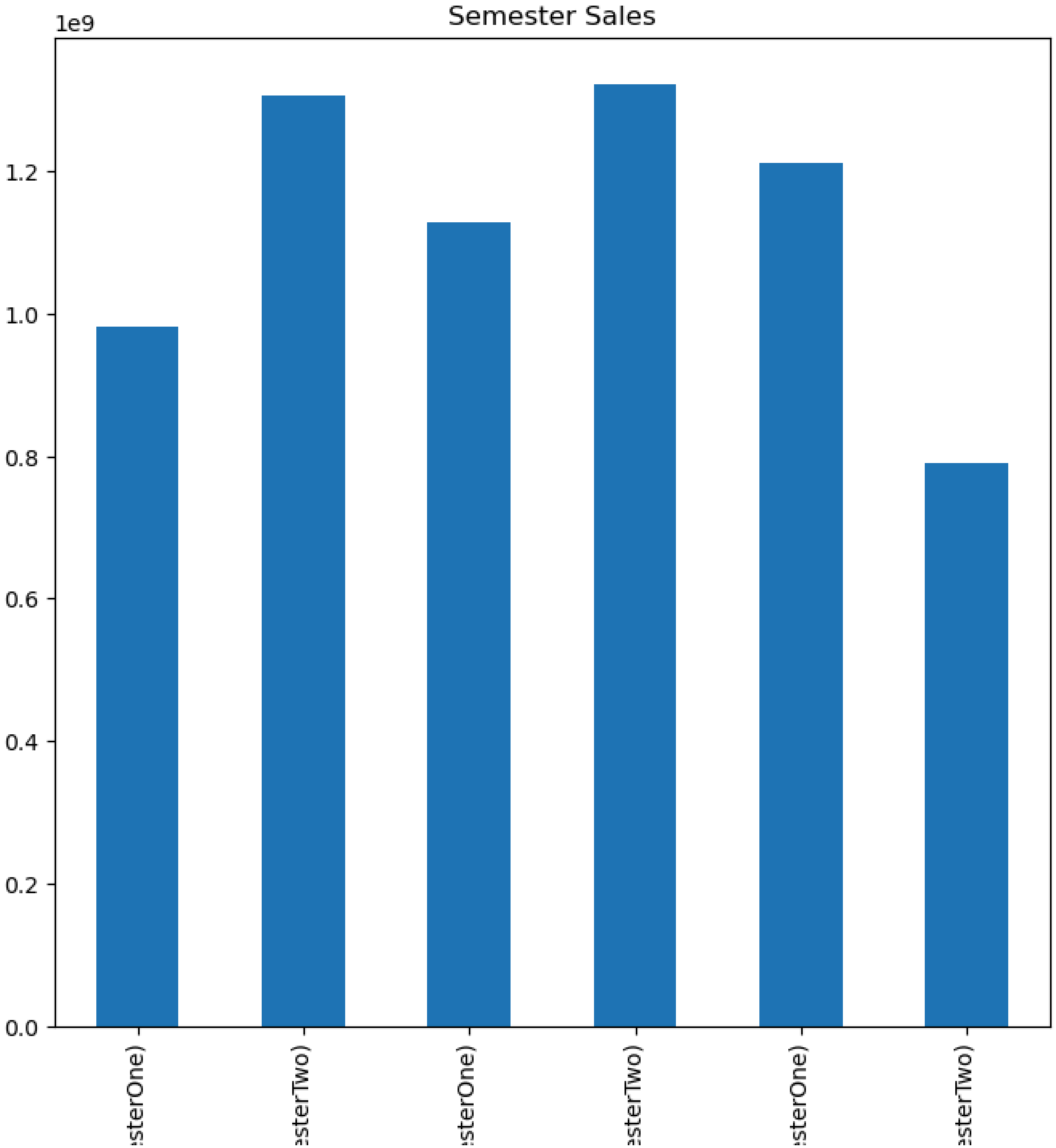
```
Out[26]: <Axes: xlabel='Date'>
```

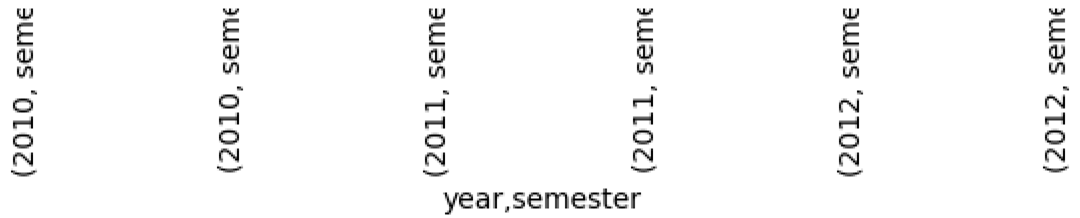


It has been analyzed that holidays have higher sales than other days

```
In [27]: df['semester'] = df['Date'].dt.month.apply(lambda x: 'semesterOne' if x <= 6 else 'semesterTwo')
         df.groupby(['year','semester'])['Weekly_Sales'].sum().plot(kind = 'bar' ,figsize= (8 , 8))
         plt.title('Semester Sales')
```

```
Out[27]: Text(0.5, 1.0, 'Semester Sales')
```


Semester Sales
1e9
1.2
1.0
0.8
0.6
0.4
0.2
0.0
esterOne)
esterTwo)
esterOne)
esterTwo)
esterOne)
esterTwo)



Semester two in 2011 and Semester two in 2010 ,which reveals the increase in sales in the second part of the year where holidays exists

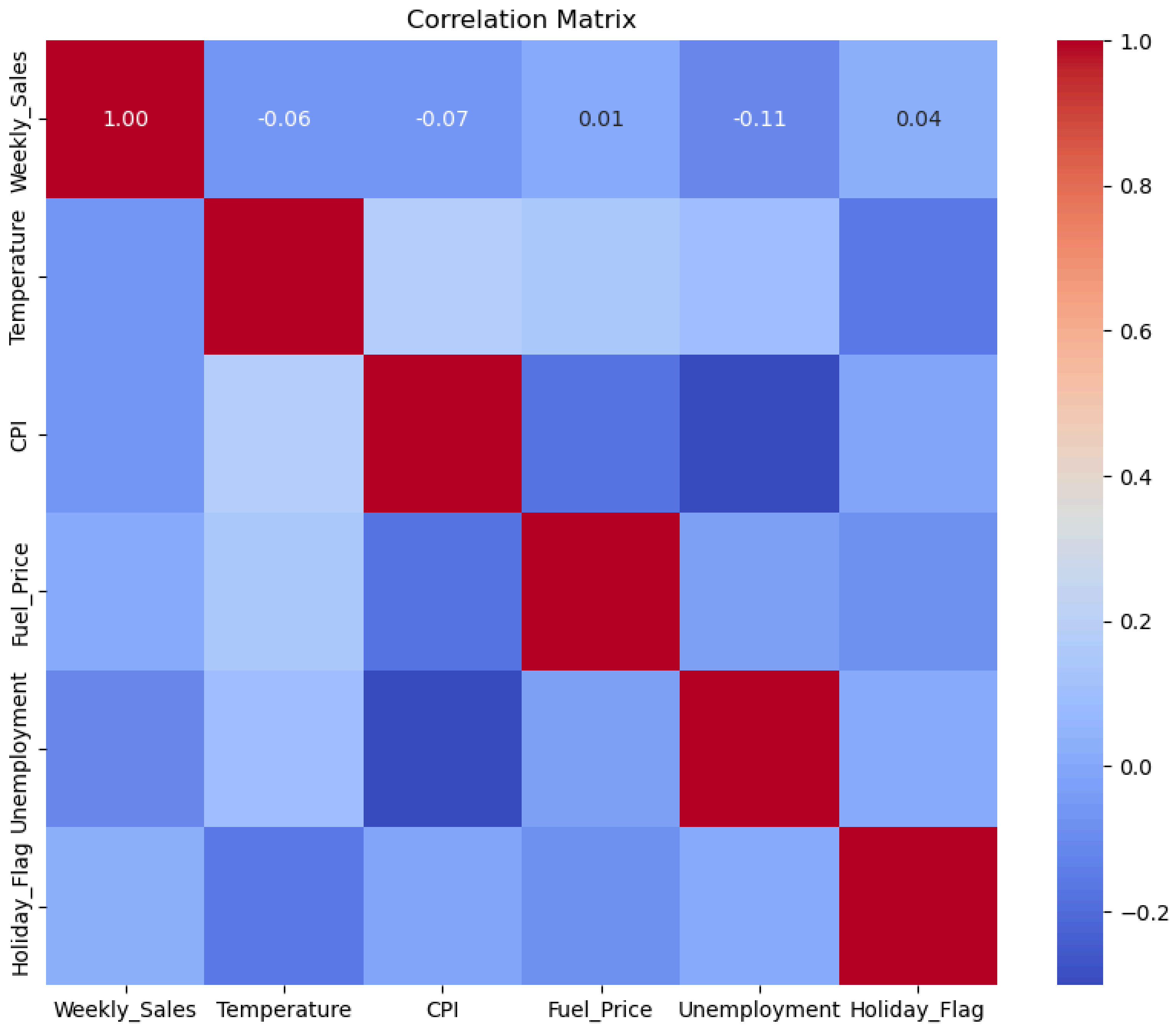
relations between weekly sales vs. other numeric features and give insights.

In [28]:

```
#correaltion with sales and all other numeric features
correlation_matrix = df[['Weekly_Sales', 'Temperature', 'CPI', 'Fuel_Price', 'Unemployment', 'Holiday_Flag']].corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", annot_kws={"size": 10})
plt.title('Correlation Matrix')
```

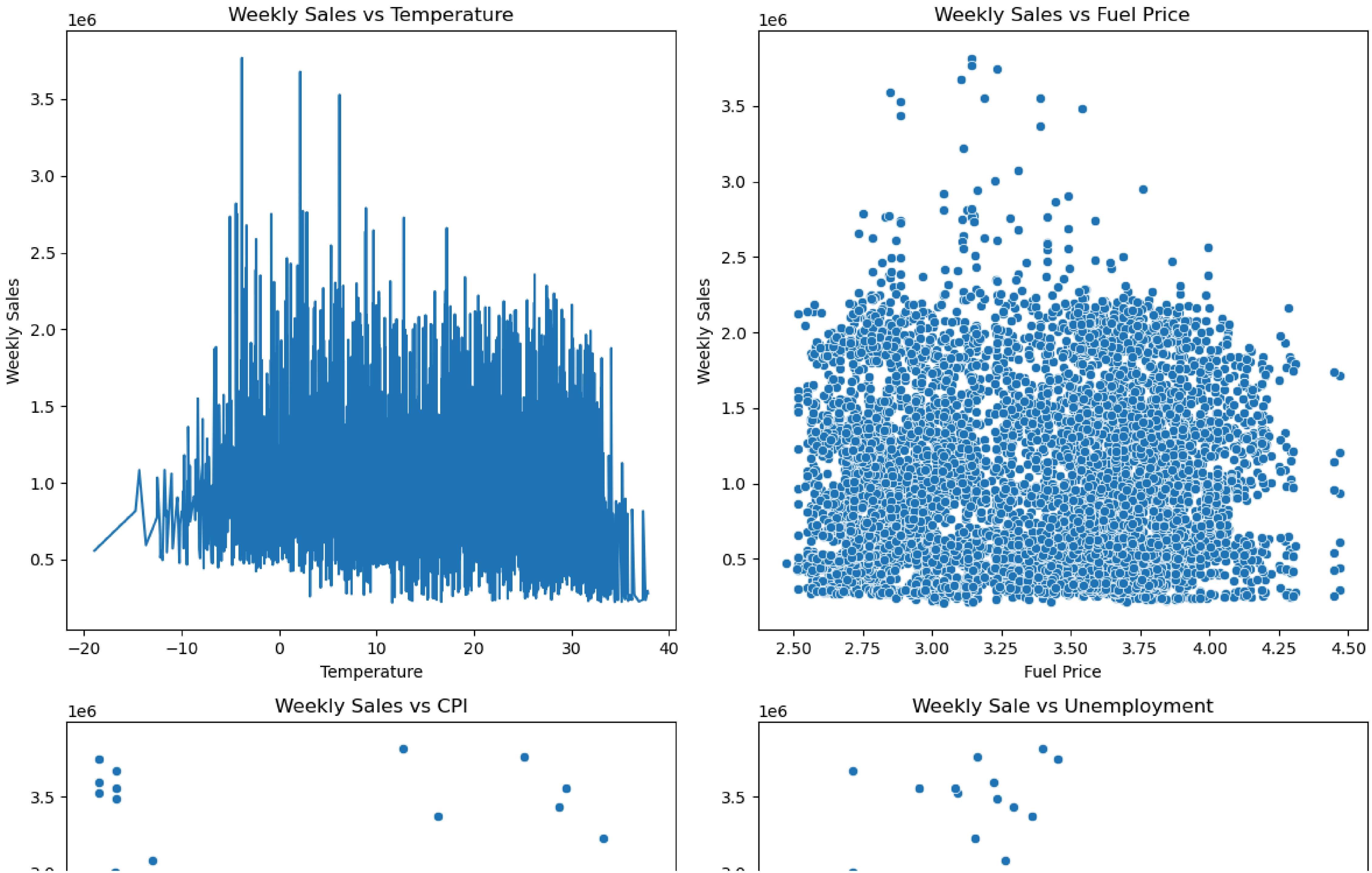
Out[28]:

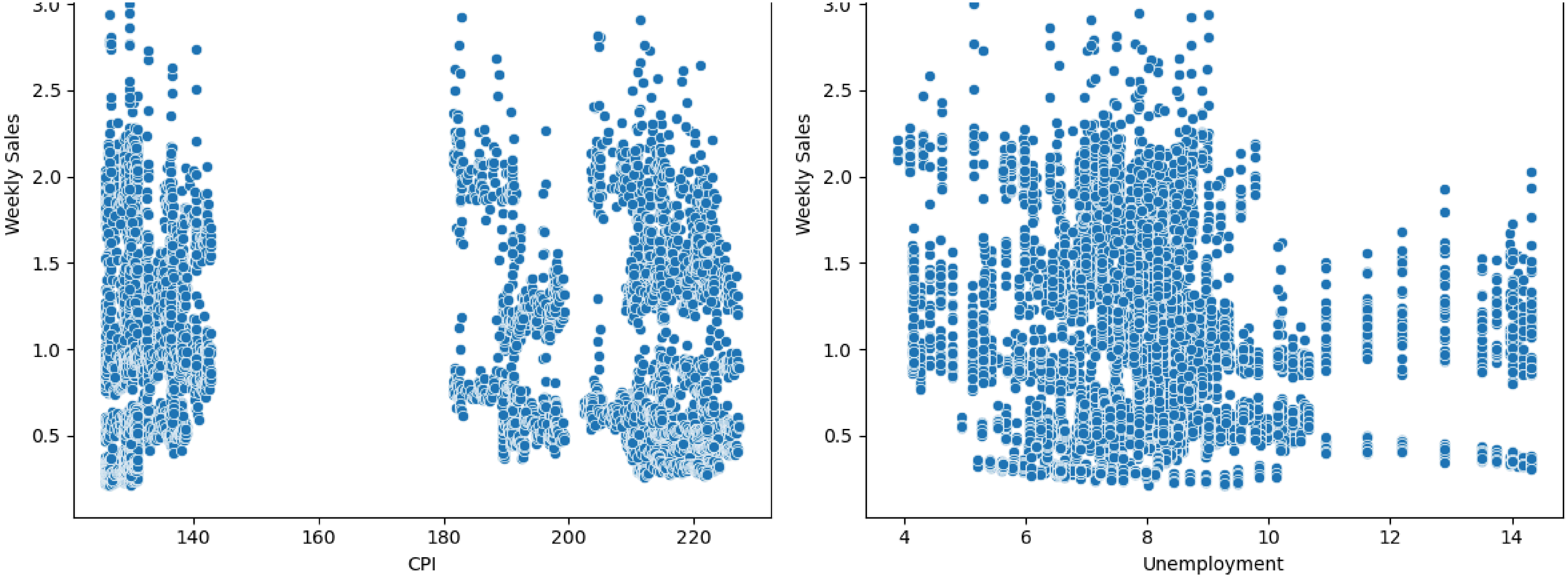
```
Text(0.5, 1.0, 'Correlation Matrix')
```


Correlation Matrix
Weekly_Sales
Temperature
CPI
Fuel_Price
Unemployment
Holiday_Flag
1.00
-0.06
-0.07
0.01
-0.11
0.04
Weekly_Sales
Temperature
CPI
Fuel_Price
Unemployment
Holiday_Flag
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

In [29]:

```
fig, axs = plt.subplots(2, 2, figsize=(12, 12))

df['Temperature_Celsius'] = (df['Temperature'] - 32) / 1.8
df.groupby('Temperature_Celsius')['Weekly_Sales'].mean().plot(ax=axs[0][0])
axs[0][0].set_title('Weekly Sales vs Temperature')
axs[0][0].set_xlabel('Temperature')
axs[0][0].set_ylabel('Weekly Sales')

sns.scatterplot(x='CPI', y='Weekly_Sales', data=df, ax=axs[1][0])
axs[1][0].set_title('Weekly Sales vs CPI')
axs[1][0].set_xlabel('CPI')
axs[1][0].set_ylabel('Weekly Sales')

sns.scatterplot(x='Fuel_Price', y='Weekly_Sales', data=df, ax=axs[0][1])
axs[0][1].set_title('Weekly Sales vs Fuel Price')
axs[0][1].set_xlabel('Fuel Price')
axs[0][1].set_ylabel('Weekly Sales')

sns.scatterplot(x= 'Unemployment', y='Weekly_Sales', data=df, ax=axs[1][1])
axs[1][1].set_title('Weekly Sale vs Unemployment')
axs[1][1].set_xlabel('Unemployment')
axs[1][1].set_ylabel('Weekly Sales')

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


Weekly Sales vs Temperature
1e6
3.5
3.0
2.5
2.0
1.5
1.0
0.5
Weekly Sales
−20
−10
0
10
20
30
40
Temperature
Weekly Sales vs Fuel Price
1e6
3.5
3.0
2.5
2.0
1.5
1.0
0.5
Weekly Sales
2.50
2.75
3.00
3.25
3.50
3.75
4.00
4.25
4.50
Fuel Price
Weekly Sales vs CPI
1e6
3.5
Weekly Sale vs Unemployment
1e6
3.5



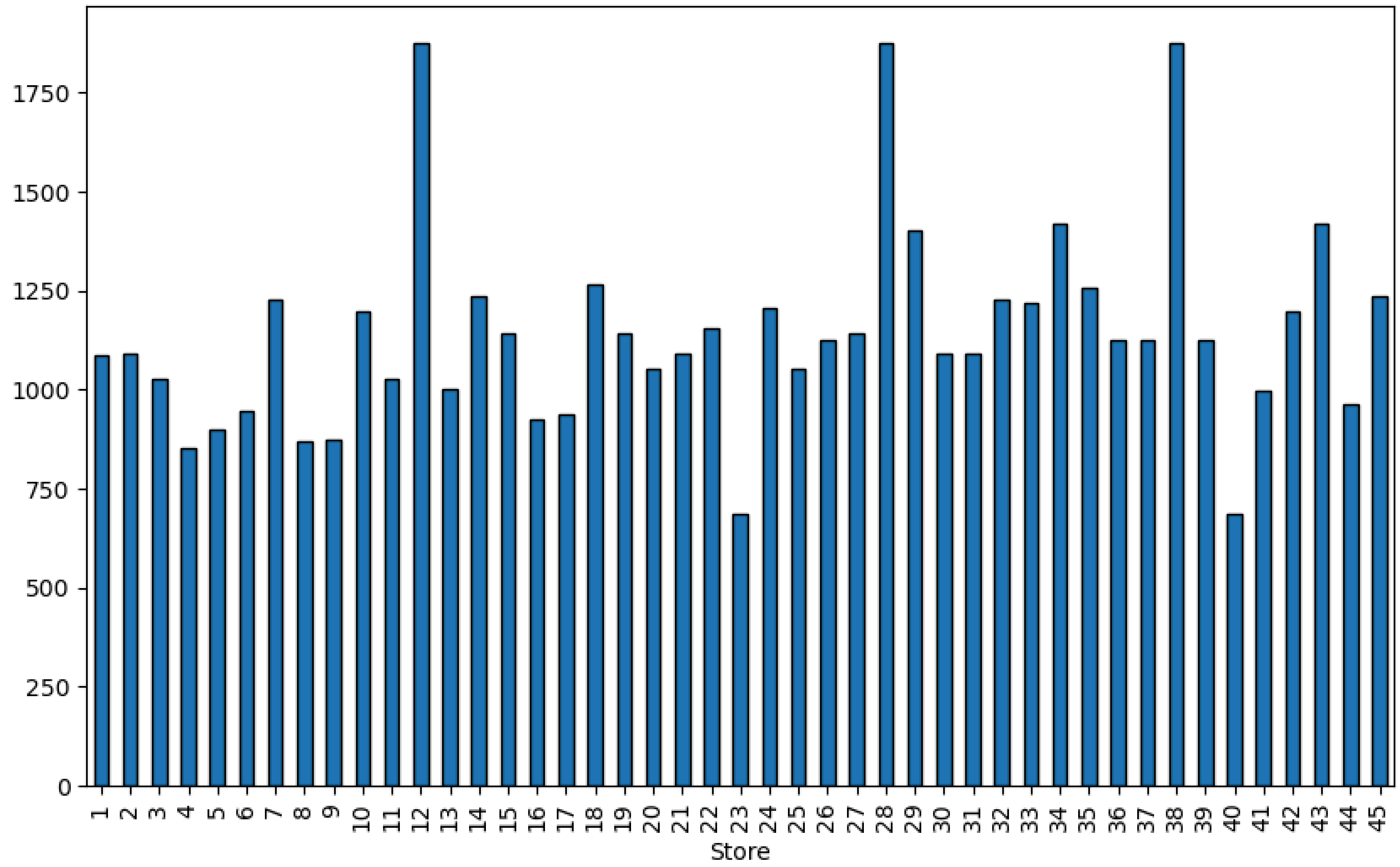
insights

-it has been analyzed that CPI(customer price index) is less than 140 and is higher around 220

-When unemployment rate increases it shows us that the weekly sales is deacresed ,Let's show which store has the maximum Unemployment :

In [30]:
```
df.groupby('Store')['Unemployment'].sum().plot(kind = 'bar' ,edgecolor = 'black' ,figsize = (10 , 6))
```

Out[30]:
```
<Axes: xlabel='Store'>
```



Stores [12 , 28 , 38] have the maximum unemployment rate

-When temperature decreases , meanWeekly sales increases as people increase their Purchases to feel warm

-Fuel price impacts sales ,as the Fuel Price increase , it has been analyzed that sales decreases

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```