Importing Data

In [8]:

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

In [2]:

```
colours = ["#D0DBEE", "#C2C4E2", "#EED4E5", "#D1E6DC", "#BDE2E2"]
```

parsing the date into a unit format

In [45]:

```
data = pd.read_csv(r'C:\Users\Dell G3\Downloads\\walmart.csv',parse_dates=['Date'])
```

```
C:\Users\Dell G3\anaconda3\lib\site-packages\pandas\core\tools\datetimes.p
y:1063: UserWarning: Parsing '19-02-2010' in DD/MM/YYYY format. Provide fo
rmat or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell G3\anaconda3\lib\site-packages\pandas\core\tools\datetimes.p
y:1063: UserWarning: Parsing '26-02-2010' in DD/MM/YYYY format. Provide fo
rmat or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell G3\anaconda3\lib\site-packages\pandas\core\tools\datetimes.p
y:1063: UserWarning: Parsing '19-03-2010' in DD/MM/YYYY format. Provide fo
rmat or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell G3\anaconda3\lib\site-packages\pandas\core\tools\datetimes.p
y:1063: UserWarning: Parsing '26-03-2010' in DD/MM/YYYY format. Provide fo
rmat or specify infer_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\Dell G3\anaconda3\lib\site-packages\pandas\core\tools\datetimes.p
y:1063: UserWarning: Parsing '16-04-2010' in DD/MM/YYYY format. Provide fo
rmat or specify infer_datetime_format=True for consistent parsing.
```

Exploratory Anaylsis

In [4]:

```
data.head()
```

Out[4]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployme
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.1(
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.1(
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.1(
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.1(
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.1(

In [5]:

```
data.describe()
```

Out[5]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unem
count	6435.000000	6.435000e+03	6435.000000	6435.000000	6435.000000	6435.000000	643
mean	23.000000	1.046965e+06	0.069930	60.663782	3.358607	171.578394	
std	12.988182	5.643666e+05	0.255049	18.444933	0.459020	39.356712	
min	1.000000	2.099862e+05	0.000000	-2.060000	2.472000	126.064000	
25%	12.000000	5.533501e+05	0.000000	47.460000	2.933000	131.735000	
50%	23.000000	9.607460e+05	0.000000	62.670000	3.445000	182.616521	
75%	34.000000	1.420159e+06	0.000000	74.940000	3.735000	212.743293	
max	45.000000	3.818686e+06	1.000000	100.140000	4.468000	227.232807	1

In [6]:

```
data.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 [7]:

```
data.isnull().sum()
```

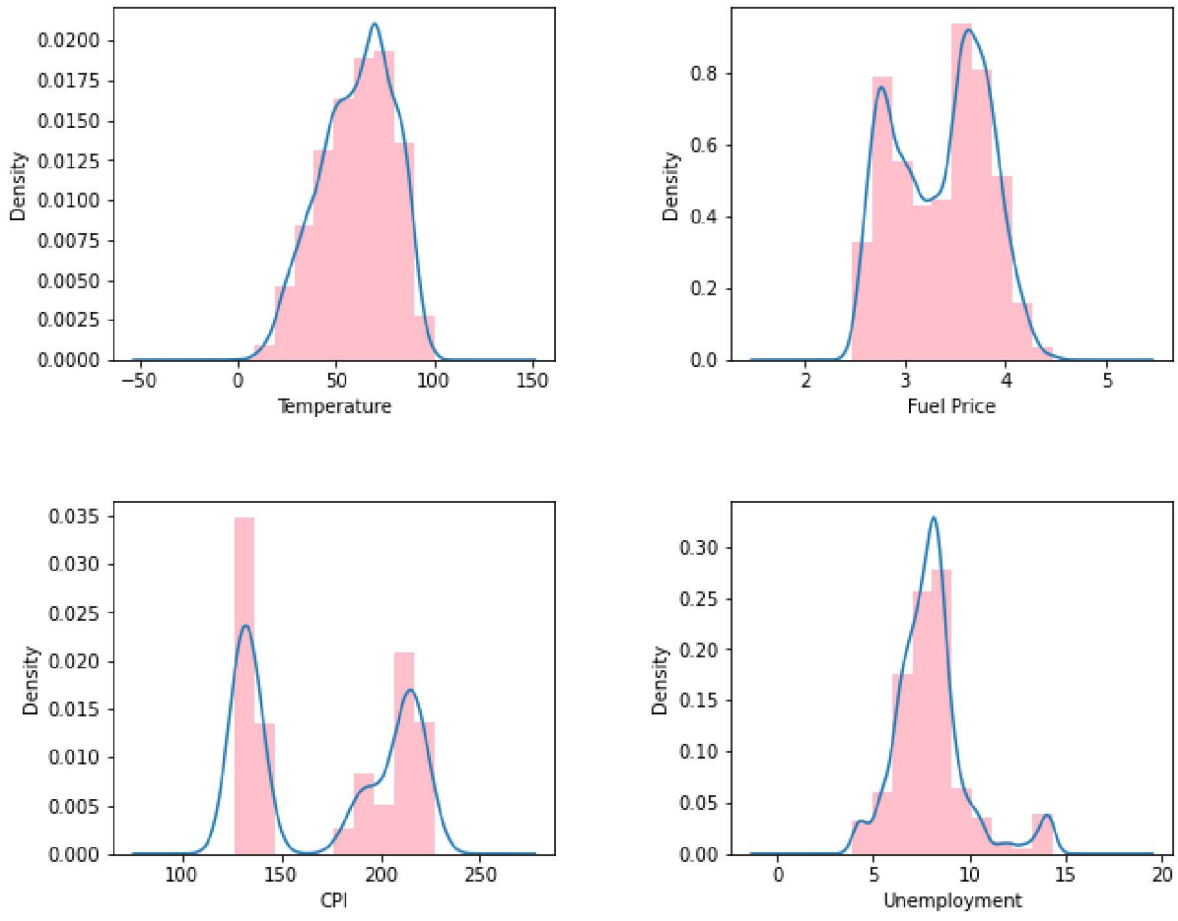
Out[7]:

```
Store           0
Date            0
Weekly_Sales    0
Holiday_Flag    0
Temperature     0
Fuel_Price      0
CPI             0
Unemployment    0
dtype: int64
```

Quantitive Variables Distributions

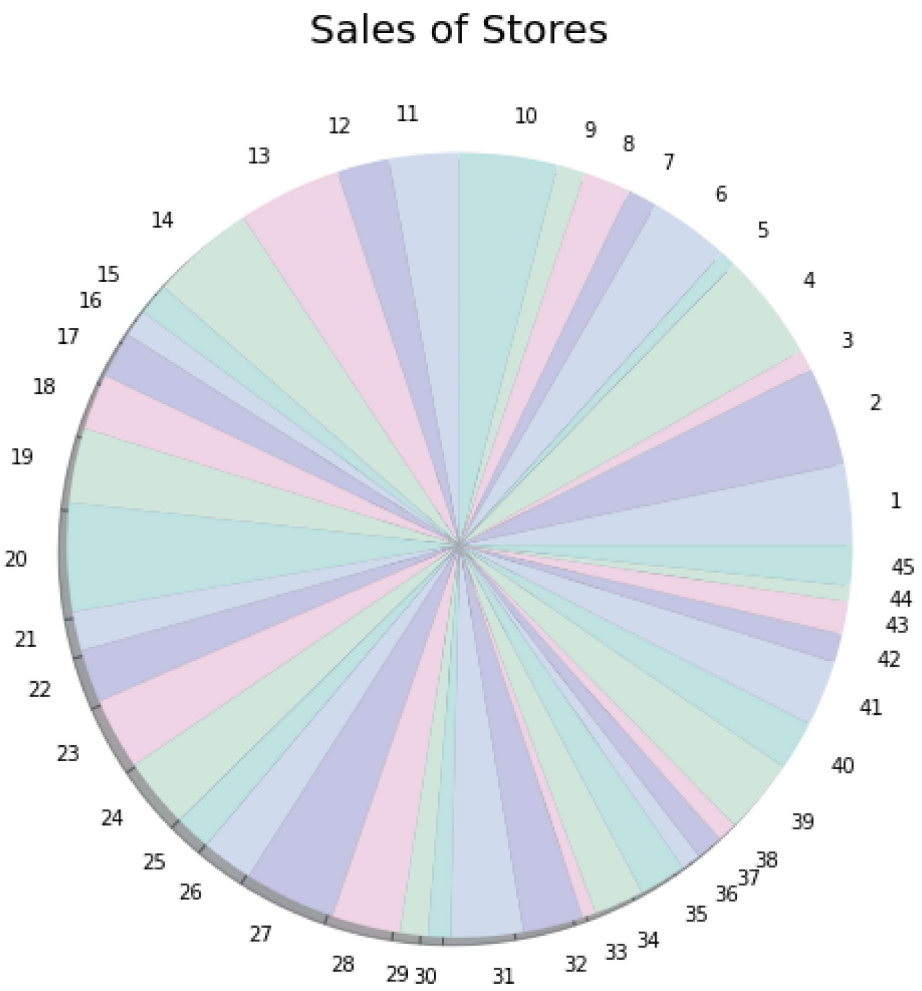
In [17]:

```
plt.subplots(figsize=(10, 8))
plt.subplots_adjust(left=0.1,bottom=0.1,right=0.9,top=0.9,wspace=0.4,hspace=0.4)

plt.subplot(2, 2, 1)
data["Temperature"].plot(kind="hist", density=True,color = "#FFC0CB")
data["Temperature"].plot(kind="kde")
plt.xlabel("Temperature")

plt.subplot(2, 2, 2)
data["Fuel_Price"].plot(kind="hist", density=True , color = "#FFC0CB")
data["Fuel_Price"].plot(kind="kde")
plt.xlabel("Fuel Price")

plt.subplot(2, 2, 3)
data["CPI"].plot(kind="hist", density=True , color = "#FFC0CB")
data["CPI"].plot(kind="kde")
plt.xlabel("CPI")

plt.subplot(2, 2, 4)
data["Unemployment"].plot(kind="hist", density=True , color = "#FFC0CB")
data["Unemployment"].plot(kind="kde")
plt.xlabel("Unemployment")

plt.show()
```



In [9]:

```
Total_sales = data.groupby('Store')['Weekly_Sales'].sum().to_frame().reset_index()
Total_sales['Weekly_Sales'].plot.pie(labels = Total_sales.Store,shadow=True, figsize=(9,9),
plt.title('Sales of Stores',size=20)
plt.show()
```



Discover Maximum Sales

In [10]:

```
Total_sales['Weekly_Sales'].max()
```

Out[10]:

```
301397792.46
```

In [11]:

```
Total_sales. loc[Total_sales['Weekly_Sales'] == Total_sales['Weekly_Sales'].max()]
```

Out[11]:

	Store	Weekly_Sales
19	20	3.013978e+08

> **Conclusion:** Store No. ***20*** has Maximum Weekly Sales

In [18]:

```
stores = data.groupby('Store')['Weekly_Sales'].sum().reset_index()
stores.head(10)
```

Out[18]:

	Store	Weekly_Sales
0	1	2.224028e+08
1	2	2.753824e+08
2	3	5.758674e+07
3	4	2.995440e+08
4	5	4.547569e+07
5	6	2.237561e+08
6	7	8.159828e+07
7	8	1.299512e+08
8	9	7.778922e+07
9	10	2.716177e+08

> Determine the Standard Deviation of each Store

In [19]:

```
sales_std = pd.DataFrame(data.groupby('Store')['Weekly_Sales'].std())
sales_std.head()
```

Out[19]:

Store	Weekly_Sales
1	155980.767761
2	237683.694682
3	46319.631557
4	266201.442297
5	37737.965745

In [20]:

```
sales_std['Weekly_Sales'].max()
```

Out[20]:

```
317569.9494755081
```

In [21]:

```
max_std = sales_std.loc[sales_std['Weekly_Sales'] == sales_std['Weekly_Sales'].max()]
max_std
```

Out[21]:

Store	Weekly_Sales
14	317569.949476

> **Conclusion:** Store No. ***14*** has Maximum standard deviation which mean it doesn't have stable sales performance

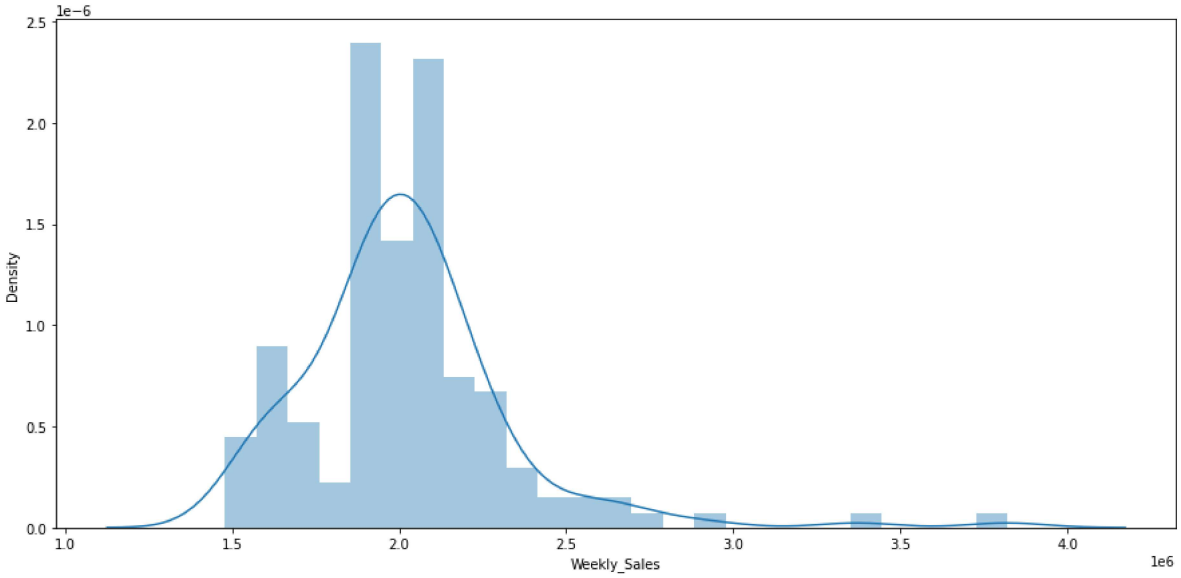
In [22]:

```
plt.figure(figsize=(15,7))
sns.distplot(data[data['Store'] == max_std.head(1).index[0]]['Weekly_Sales'])
```

```
C:\Users\Dell G3\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-lev
el function with similar flexibility) or `histplot` (an axes-level function
for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[22]:

```
<AxesSubplot:xlabel='Weekly_Sales', ylabel='Density'>
```



Discover the Impact of Holidays weeks

In [23]:

```
holidays = data[data.Holiday_Flag == 1]
holidays.head()
```

Out[23]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemploym
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.
31	1	2010-10-09	1507460.69	1	78.69	2.565	211.495190	7.
42	1	2010-11-26	1955624.11	1	64.52	2.735	211.748433	7.
47	1	2010-12-31	1367320.01	1	48.43	2.943	211.404932	7.
53	1	2011-11-02	1649614.93	1	36.39	3.022	212.936705	7.

In [24]:

```
holidays.drop('Holiday_Flag',axis='columns',inplace=True)
holidays.head()
```

```
C:\Users\Dell G3\AppData\Local\Temp\ipykernel_19836\1064631885.py:1: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  holidays.drop('Holiday_Flag',axis='columns',inplace=True)
```

Out[24]:

	Store	Date	Weekly_Sales	Temperature	Fuel_Price	CPI	Unemployment
1	1	2010-12-02	1641957.44	38.51	2.548	211.242170	8.106
31	1	2010-10-09	1507460.69	78.69	2.565	211.495190	7.787
42	1	2010-11-26	1955624.11	64.52	2.735	211.748433	7.838
47	1	2010-12-31	1367320.01	48.43	2.943	211.404932	7.838
53	1	2011-11-02	1649614.93	36.39	3.022	212.936705	7.742

In [25]:

```
holidays = holidays.groupby('Date')['Weekly_Sales'].sum().reset_index()
holidays.head(15)
```

Out[25]:

	Date	Weekly_Sales
0	2010-10-09	45634397.84
1	2010-11-26	65821003.24
2	2010-12-02	48336677.63
3	2010-12-31	40432519.00
4	2011-09-09	46763227.53
5	2011-11-02	47336192.79
6	2011-11-25	66593605.26
7	2011-12-30	46042461.04
8	2012-07-09	48330059.31
9	2012-10-02	50009407.92

In [26]:

```
Non_holidays = data[data.Holiday_Flag == 0]
Non_holidays.head()
```

Out[26]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployme
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.1(
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.1(
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.1(
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.1(
5	1	2010-12-03	1439541.59	0	57.79	2.667	211.380643	8.1(

In [27]:

```
Non_holidays.drop('Holiday_Flag',axis='columns',inplace=True)
Non_holidays.head()
```

```
C:\Users\Dell G3\AppData\Local\Temp\ipykernel_19836\718940665.py:1: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  Non_holidays.drop('Holiday_Flag',axis='columns',inplace=True)
```

Out[27]:

	Store	Date	Weekly_Sales	Temperature	Fuel_Price	CPI	Unemployment
0	1	2010-05-02	1643690.90	42.31	2.572	211.096358	8.106
2	1	2010-02-19	1611968.17	39.93	2.514	211.289143	8.106
3	1	2010-02-26	1409727.59	46.63	2.561	211.319643	8.106
4	1	2010-05-03	1554806.68	46.50	2.625	211.350143	8.106
5	1	2010-12-03	1439541.59	57.79	2.667	211.380643	8.106

In [28]:

```
Non_holidays = Non_holidays.groupby('Date')['Weekly_Sales'].sum().reset_index()
Non_holidays.head(100)
```

Out[28]:

	Date	Weekly_Sales
0	2010-01-10	42239875.87
1	2010-02-04	50423831.26
2	2010-02-07	48917484.50
3	2010-02-19	48276993.78
4	2010-02-26	43968571.13
...	...	...
95	2012-01-27	39834974.67
96	2012-02-03	46861034.97
97	2012-02-17	50197056.96
98	2012-02-24	45771506.57
99	2012-03-02	46085608.09

100 rows × 2 columns

calculate the sales of the Non-holiday season

In [29]:

```
np.mean(Non_holidays['Weekly_Sales'])
```

Out[29]:

```
46856537.10939851
```

Determine which Holidays have Positive Impact on the Weekly Sales

In [30]:

```
positive_impacted_holidays = holidays[ holidays.Weekly_Sales > np.mean(Non_holidays['Weekly
positive_impacted_holidays.head(100)
```

Out[30]:

	Date	Weekly_Sales
1	2010-11-26	65821003.24
2	2010-12-02	48336677.63
5	2011-11-02	47336192.79
6	2011-11-25	66593605.26
8	2012-07-09	48330059.31
9	2012-10-02	50009407.92

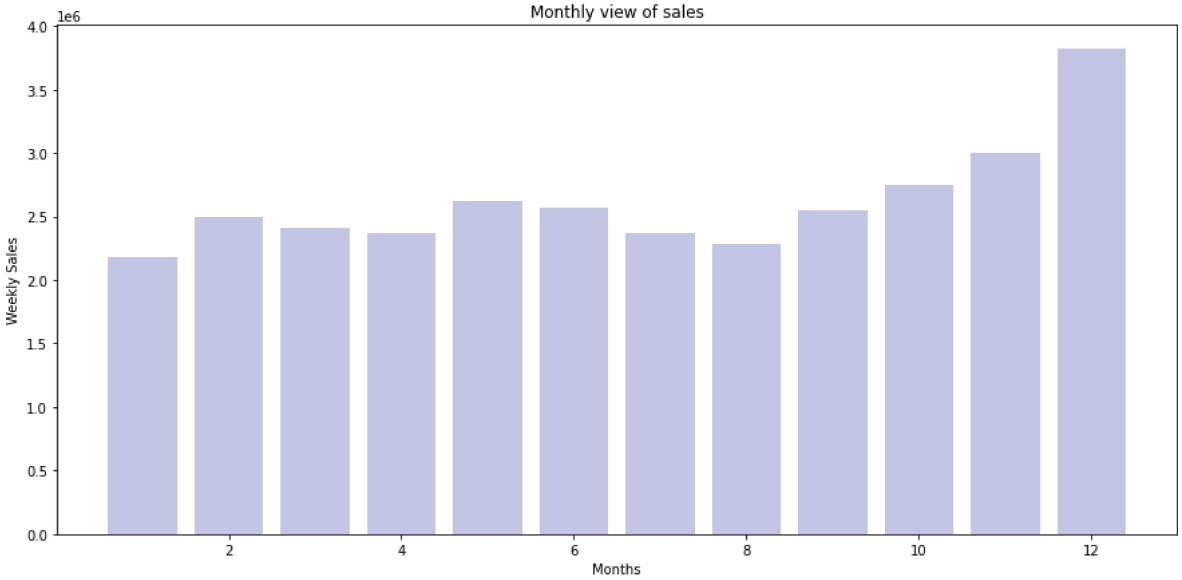
In [31]:

```
data['Month'] = pd.DatetimeIndex(data['Date']).month
```

Display the Seasons effects on the Weekly Sales
- monthly
- semesterly

In [32]:

```
plt.figure(figsize=(15,7))
plt.bar(data["Month"],data["Weekly_Sales"], color="#C2C4E2")
plt.xlabel("Months")
plt.ylabel("Weekly Sales")
plt.title("Monthly view of sales")
plt.show()
```



Conclusion: ***12th*** Month has The Maximum Weekly Sales

In [33]:

```
data.sort_values(by='Month',ascending=True)
```

Out[33]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemploy
5126	36	2012-01-06	306005.53	0	80.74	3.567	220.432210	
5485	39	2011-01-28	1158698.44	0	47.94	3.010	210.968241	
5484	39	2011-01-21	1243370.74	0	50.25	3.016	210.603107	
4679	33	2012-01-27	236920.49	0	56.33	3.675	130.314452	
407	3	2012-01-06	432268.53	0	81.55	3.501	225.251831	
...	...	...	...	...	...	...	...	
4816	34	2011-12-16	1151052.86	0	32.31	3.149	129.898065	1
4817	34	2011-12-23	1593655.96	0	32.45	3.103	129.984548	1
3576	26	2010-12-02	1015684.09	1	18.14	2.771	131.586613	
2478	18	2010-12-31	887907.01	1	26.10	3.177	132.815032	
3511	25	2011-12-08	667130.48	0	70.93	3.812	208.782006	

6435 rows × 9 columns

Define Semesters

In [34]:

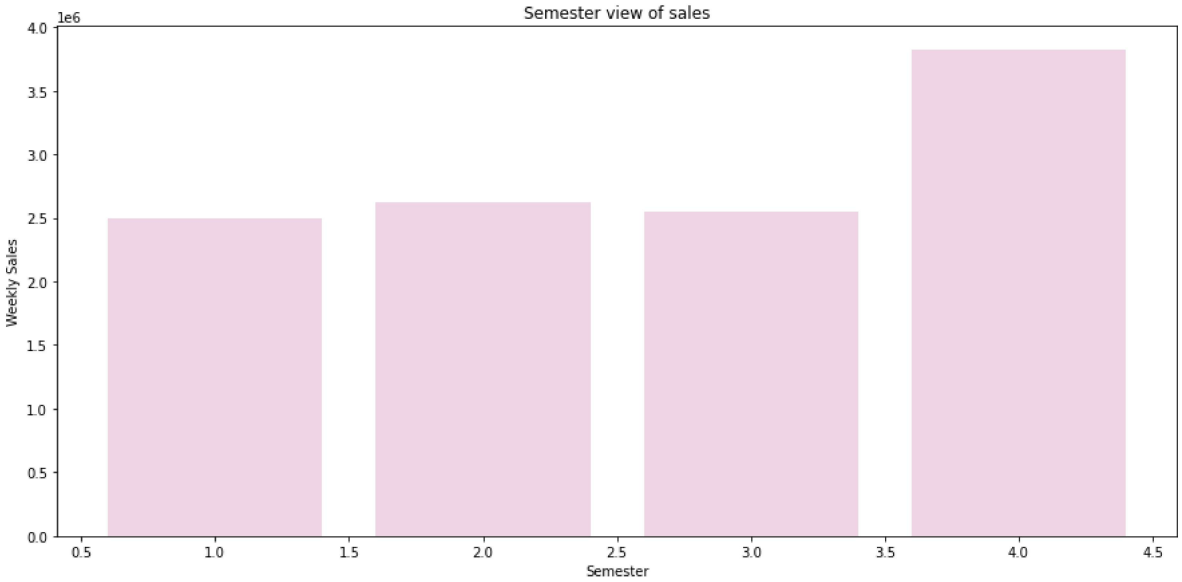
```
def semester(row):
    if row in [1,2,3]:
        return 1
    elif row in [4,5,6]:
        return 2
    elif row in [7,8,9]:
        return 3
    else:
        return 4

data['Semester'] = data['Month'].apply(lambda x: semester(x))
data.head(10)
```

Out[34]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployme
0	1	2010-05-02	1643690.90	0	42.31	2.572	211.096358	8.1
1	1	2010-12-02	1641957.44	1	38.51	2.548	211.242170	8.1
2	1	2010-02-19	1611968.17	0	39.93	2.514	211.289143	8.1
3	1	2010-02-26	1409727.59	0	46.63	2.561	211.319643	8.1
4	1	2010-05-03	1554806.68	0	46.50	2.625	211.350143	8.1
5	1	2010-12-03	1439541.59	0	57.79	2.667	211.380643	8.1
6	1	2010-03-19	1472515.79	0	54.58	2.720	211.215635	8.1
7	1	2010-03-26	1404429.92	0	51.45	2.732	211.018042	8.1
8	1	2010-02-04	1594968.28	0	62.27	2.719	210.820450	7.8
9	1	2010-09-04	1545418.53	0	65.86	2.770	210.622857	7.8

In [35]:

```
plt.figure(figsize=(15,7))
plt.bar(data["Semester"],data["Weekly_Sales"], color="#EED4E5")
plt.xlabel("Semester")
plt.ylabel("Weekly Sales")
plt.title("Semester view of sales")
plt.show()
```



Conclusion: End-of-Year Holidays have the most positive impact on the Sales as Thanksgiving and Christmas

Plot the Relations between Weekly Sales and the Other Numeric Features

In [36]:

```
FuelperSales= data.groupby('Fuel_Price')['Weekly_Sales'].sum().to_frame().reset_index().sor
FuelperSales.head(100)
```

Out[36]:

	Fuel_Price	Weekly_Sales
0	2.472	470281.03
1	2.513	434471.38
2	2.514	14211389.79
3	2.520	447519.44
4	2.533	431294.45
...	...	...
95	2.733	2876839.59
96	2.735	24851342.05
97	2.736	2863903.66
98	2.737	6652044.03
99	2.740	2958675.78

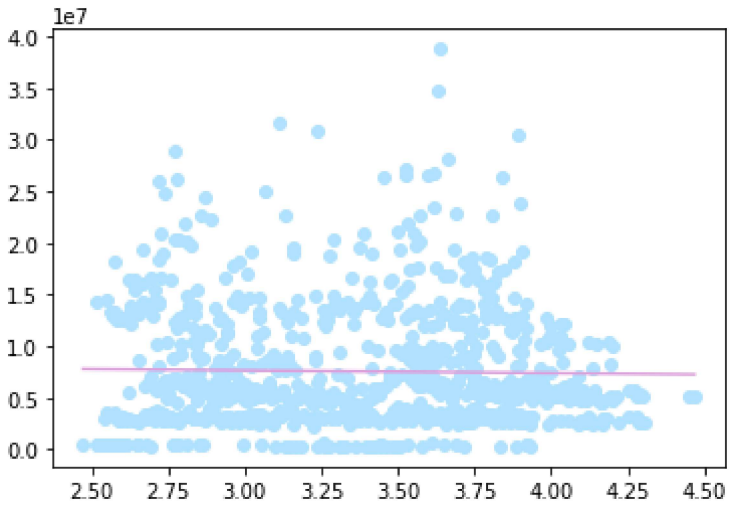
100 rows × 2 columns

In [37]:

```
plt.scatter(FuelperSales['Fuel_Price'],FuelperSales['Weekly_Sales'], color="#B0E2FF")
plt.plot(np.unique(FuelperSales['Fuel_Price']),np.poly1d(np.polyfit(FuelperSales['Fuel_Pric
                                                    ,FuelperSales['Weekly_Sales'], 1))(
```

Out[37]:

```
[<matplotlib.lines.Line2D at 0x1f8f9eb98b0>]
```



Conclusion: C1 : The correlation between Fuel Price and Weekly Sales tends to *Zero* , They

doesn't have visible effect on each other

In [38]:

```
tempperSales = data.groupby('Temperature')['Weekly_Sales'].sum().to_frame().reset_index().s
tempperSales.head(100)
```

Out[38]:

	Temperature	Weekly_Sales
0	-2.06	558027.77
1	5.54	817485.14
2	6.23	1083071.14
3	7.46	593875.46
4	9.51	775910.43
...	...	...
95	20.87	1019608.37
96	20.96	1507637.17
97	21.02	1370562.11
98	21.07	1067754.06
99	21.10	677231.63

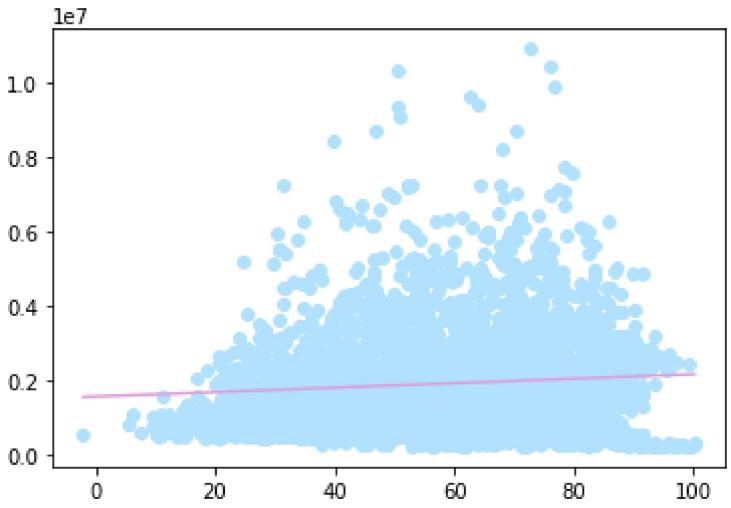
100 rows × 2 columns

In [39]:

```
plt.scatter(tempperSales['Temperature'],tempperSales['Weekly_Sales'] , color="#B0E2FF")
plt.plot(np.unique(tempperSales['Temperature']),np.poly1d(np.polyfit(tempperSales['Temperat
                          ,tempperSales['Weekly_Sales'], 1))(np.unique(tempperSales['Tempera
```

Out[39]:

```
[<matplotlib.lines.Line2D at 0x1f8f98e35b0>]
```



Conclusion: C2: The Temperature and Weekly Sales have Small positive correlation , They

have slightly impact on each other

In [40]:

```
CPIperSales = data.groupby('CPI')['Weekly_Sales'].sum().to_frame().reset_index().sort_value
CPIperSales.head(100)
```

Out[40]:

	CPI	Weekly_Sales
0	126.064000	11186672.61
1	126.076645	11236542.21
2	126.085452	11617124.06
3	126.089290	10644403.61
4	126.101935	11446981.89
...	...	...
95	129.845967	12101440.38
96	129.855533	13747691.87
97	129.898065	14602445.15
98	129.984548	18645632.92
99	130.071032	11307604.09

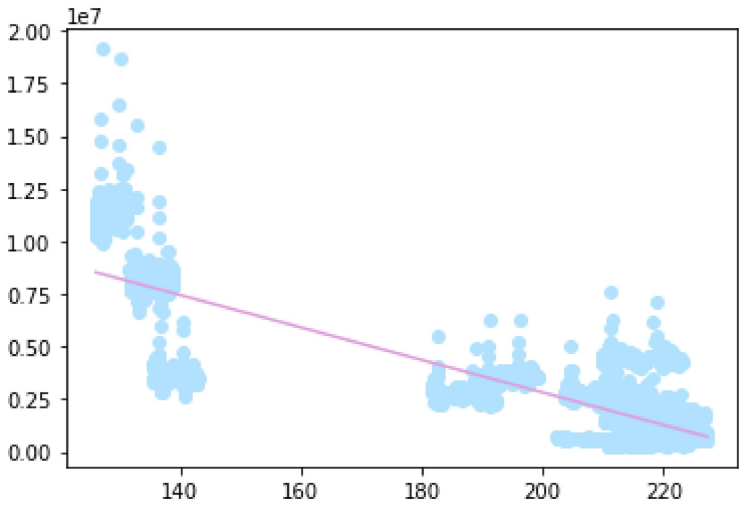
100 rows × 2 columns

In [41]:

```
plt.scatter(CPIperSales['CPI'],CPIperSales['Weekly_Sales'], color="#B0E2FF")
plt.plot(np.unique(CPIperSales['CPI']),np.poly1d(np.polyfit(CPIperSales['CPI']
                              , CPIperSales['Weekly_Sales'], 1))(np.unique(CPIperSales['CPI']
```

Out[41]:

```
[<matplotlib.lines.Line2D at 0x1f8f99bdee0>]
```



Conclusion: C3: Prevailing consumer price have Negative correlation with the Weekly Sales

In [42]:

```
UnemployerSales = data.groupby('Unemployment')['Weekly_Sales'].sum().to_frame().reset_inde
UnemployerSales.head(100)
```

Out[42]:

	Unemployment	Weekly_Sales
0	3.879	8589722.81
1	4.077	27796792.46
2	4.125	31216620.35
3	4.145	9479222.48
4	4.156	31514684.52
...	...	...
95	6.664	23407688.44
96	6.697	11300617.25
97	6.745	12645327.53
98	6.759	18820638.60
99	6.768	4659553.36

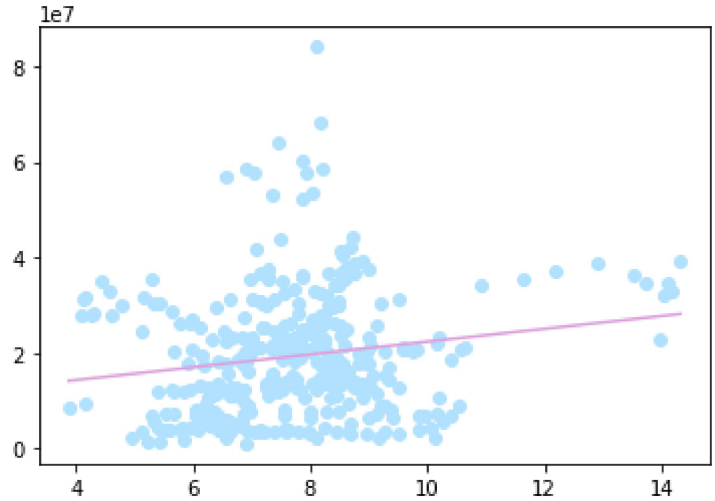
100 rows × 2 columns

In [43]:

```
plt.scatter(UnemployerSales['Unemployment'],UnemployerSales['Weekly_Sales'], color="#B0E2
plt.plot(np.unique(UnemployerSales['Unemployment']),np.poly1d(np.polyfit(UnemployerSales[
                    ,UnemployerSales['Weekly_Sales'], 1))(np.unique(UnemployerSales['Unem
```

Out[43]:

```
[<matplotlib.lines.Line2D at 0x1f8f9a346d0>]
```



Conclusion: C4: Unemployment have Positive Correlation with the Weekly Sales