In [1]:

```
#question 1
#importing important files
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
```

In [2]:

```
df = pd.read_csv('complaint.csv')
df
```

Out[2]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State
0	250635	Cable Internet Speeds	22-04-2015	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland
1	223441	Payment disappear - service got disconnected	04-08-2015	04-Aug-15	10:22:56 AM	Internet	Acworth	Georgia
2	242732	Speed and Service	18-04-2015	18-Apr-15	9:55:47 AM	Internet	Acworth	Georgia
3	277946	Imposed a New Usage Cap of 300GB that punishe...	05-07-2015	05-Jul-15	11:59:35 AM	Internet	Acworth	Georgia
4	307175	not working and no service to boot	26-05-2015	26-May-15	1:25:26 PM	Internet	Acworth	Georgia
...	...	...	...	...	...	...	...	...
2072	338192	Speed throttling, speeds not at promised output	06-12-2015	06-Dec-15	6:35:59 PM	Customer Care Call	Yorkville	Illinois
2073	213550	Service Availability	04-02-2015	04-Feb-15	9:13:18 AM	Customer Care Call	Youngstown	Florida
2074	318775	Monthly Billing for Returned Modem	06-02-2015	06-Feb-15	1:24:39 PM	Customer Care Call	Ypsilanti	Michigan
2075	331188	complaint about comcast	06-09-2015	06-Sep-15	5:28:41 PM	Internet	Ypsilanti	Michigan
2076	360489	Extremely unsatisfied customer	23-06-2015	23-Jun-15	11:13:30 PM	Customer Care Call	Ypsilanti	Michigan

2077 rows × 11 columns

In [4]:

```
#adding an column of unit frequencies
#when we group the data frame according to some condition then
#we sum the frequencies and analyse accordingly
aux_df = df
aux_df['#complaints']=[1]*2077
aux_df
```

Out[4]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State
0	250635	Cable Internet Speeds	22-04-2015	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland
1	223441	Payment disappear - service got disconnected	04-08-2015	04-Aug-15	10:22:56 AM	Internet	Acworth	Georgia
2	242732	Speed and Service	18-04-2015	18-Apr-15	9:55:47 AM	Internet	Acworth	Georgia
3	277946	Imposed a New Usage Cap of 300GB that punishe...	05-07-2015	05-Jul-15	11:59:35 AM	Internet	Acworth	Georgia
4	307175	not working and no service to boot	26-05-2015	26-May-15	1:25:26 PM	Internet	Acworth	Georgia
...	...	...	...	...	...	...	...	...
2072	338192	Speed throttling, speeds not at promised output	06-12-2015	06-Dec-15	6:35:59 PM	Customer Care Call	Yorkville	Illinois
2073	213550	Service Availability	04-02-2015	04-Feb-15	9:13:18 AM	Customer Care Call	Youngstown	Florida
2074	318775	Monthly Billing for Returned Modem	06-02-2015	06-Feb-15	1:24:39 PM	Customer Care Call	Ypsilanti	Michigan
2075	331188	complaint about comcast	06-09-2015	06-Sep-15	5:28:41 PM	Internet	Ypsilanti	Michigan
2076	360489	Extremely unsatisfied customer	23-06-2015	23-Jun-15	11:13:30 PM	Customer Care Call	Ypsilanti	Michigan

2077 rows × 12 columns

In [5]:

```
#part 5 of question1 creating a data_frame to hold data about states
state_df = pd.DataFrame(columns = ['State','complaints'])
#freq_df = pd.DateFrame(columns = ['State','complaints'])
state_df['State'] = aux_df['State']
state_df['complaints'] = aux_df['#complaints']
f = state_df.groupby('State').complaints.sum()
#storing the names of the states
l = ['Alabama','Arizona','Arkansas','California','Colorado','Connecticut'
,'Delaware'
,'District Of Columbia'
,'District of Columbia'
,'Florida'
,'Georgia'
,'Illinois'
,'Indiana'
,'Iowa'
,'Kansas'
,'Kentucky'
,'Louisiana'
,'Maine'
,'Maryland'
,'Massachusetts'
,'Michigan'
,'Minnesota'
,'Mississippi'
,'Missouri'
,'Montana'
,'Nevada'
,'New Hampshire'
,'New Jersey'
,'New Mexico'
,'New York'
,'North Carolina'
,'Ohio'
,'Oregon'
,'Pennsylvania'
,'South Carolina'
,'Tennessee'
,'Texas'
,'Utah'
,'Vermont'
,'Virginia'
,'Washington'
,'West Virginia']
freq_df = pd.DataFrame()
freq_df['State'] = l
al = []
for val in f:
    al.append(val)
freq_df['complaints'] = al
#freq_df contains number of complaints per state
Max_complaints = freq_df['complaints'].max()
Min_complaints = freq_df['complaints'].min()
MX_STATE = freq_df[(freq_df['complaints'] == Max_complaints)]['State']
MN_STATE = freq_df[(freq_df['complaints'] == Min_complaints)]['State']
MX_STATE
```

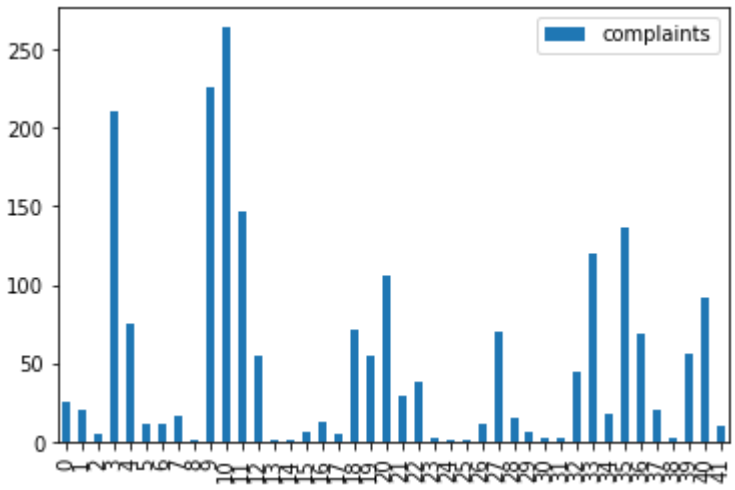
Out[5]:
```
10    Georgia
Name: State, dtype: object
```

In [6]:
```
MN_STATE #can be more than one
```

Out[6]:
```
8     District of Columbia
13                    Iowa
14                  Kansas
24                 Montana
25                  Nevada
Name: State, dtype: object
```

In [7]:
```
#part 4 of question1
freq_df.plot.bar()
```

Out[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd626f80190>
```



In [8]:

```
#part 6 of the question
df1 = aux_df[aux_df['Status'] == 'Open']
df1
```

Out[8]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State
3	277946	Imposed a New Usage Cap of 300GB that punishe...	05-07-2015	05-Jul-15	11:59:35 AM	Internet	Acworth	Georgia
9	371214	Raising Prices and Not Being Available To Ask...	28-06-2015	28-Jun-15	6:46:31 PM	Customer Care Call	Alameda	California
12	339282	Violating Open Internet Rules by Blocking HBO...	13-06-2015	13-Jun-15	4:03:18 PM	Internet	Albuquerque	New Mexico
23	370538	monopoly bundling practices	27-06-2015	27-Jun-15	9:04:34 PM	Internet	Alexandria	Virginia
24	270163	bait and switch	05-02-2015	05-Feb-15	3:55:24 PM	Internet	Algonquin	Illinois
...	...	...	...	...	...	...	...	...
2017	367577	charge my own router as unreturned equipment	26-06-2015	26-Jun-15	10:07:45 AM	Customer Care Call	West Lafayette	Indiana
2034	376295	Slow internet service	30-06-2015	30-Jun-15	10:57:27 PM	Internet	White House	Tennessee
2051	339481	Terrible internet service from	13-06-2015	13-Jun-15	7:14:02 PM	Customer Care Call	Woodbridge	Virginia
2057	305166	comcst data cap	24-05-2015	24-May-15	12:34:08 AM	Customer Care Call	Woodstock	Georgia
2072	338192	Speed throttling, speeds not at promised output	06-12-2015	06-Dec-15	6:35:59 PM	Customer Care Call	Yorkville	Illinois

338 rows × 12 columns

In [9]:

```
state_df = pd.DataFrame(columns = ['State','complaints'])
state_df['State'] = df1['State']
state_df['complaints'] = df1['#complaints']
f = state_df.groupby('State').complaints.sum()
freq_df1 = pd.DataFrame()
l1 = ['Alabama','Arizona','California','Colorado','Connecticut','Delaware','Distric
'Georgia',
'Illinois',
'Indiana',
'Maine',
'Maryland',
'Massachusetts',
'Michigan',
'Minnesota',
'Mississippi',
'New Hampshire',
'New Jersey',
'New Mexico',
'Oregon',
'Pennsylvania',
'South Carolina',
'Tennessee',
'Texas',
'Utah',
'Vermont',
'Virginia',
'Washington',
'West Virginia']
freq_df1['State'] = l1
frac_list = []
al1 = []
for val in f:
    al1.append(val)
for i in range(len(al1)):
    val = al1[i]//al[i]
    frac_list.append(val)
freq_df1['State'] = l1
freq_df1['frac_list'] = frac_list
print(freq_df1['frac_list'].max())
print(freq_df1['frac_list'].min())
#frac_list
```

```
34
0
```

In [10]:

```
#part 1 and 2 aux_df gets month and we use this dataframe to group cumulative
def getMonth(s):
    return s.split("-")[1]

aux_df['month'] = aux_df['Date_month_year'].apply(lambda x: getMonth(x))
aux_df
```

Out[10]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State
0	250635	Cable Internet Speeds	22-04-2015	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland
1	223441	Payment disappear - service got disconnected	04-08-2015	04-Aug-15	10:22:56 AM	Internet	Acworth	Georgia
2	242732	Speed and Service	18-04-2015	18-Apr-15	9:55:47 AM	Internet	Acworth	Georgia
3	277946	Imposed a New Usage Cap of 300GB that punishe...	05-07-2015	05-Jul-15	11:59:35 AM	Internet	Acworth	Georgia
4	307175	not working and no service to boot	26-05-2015	26-May-15	1:25:26 PM	Internet	Acworth	Georgia
...	...	...	...	...	...	...	...	...
2072	338192	Speed throttling, speeds not at promised output	06-12-2015	06-Dec-15	6:35:59 PM	Customer Care Call	Yorkville	Illinois
2073	213550	Service Availability	04-02-2015	04-Feb-15	9:13:18 AM	Customer Care Call	Youngstown	Florida
2074	318775	Monthly Billing for Returned Modem	06-02-2015	06-Feb-15	1:24:39 PM	Customer Care Call	Ypsilanti	Michigan
2075	331188	complaint about comcast	06-09-2015	06-Sep-15	5:28:41 PM	Internet	Ypsilanti	Michigan
2076	360489	Extremely unsatisfied customer	23-06-2015	23-Jun-15	11:13:30 PM	Customer Care Call	Ypsilanti	Michigan

2077 rows × 13 columns

In [11]:

```
monthly_df = pd.DataFrame()
```

In [16]:

```
monthly_df['month'] = aux_df['month']
monthly_df['complaints'] = aux_df['#complaints']
f = monthly_df.groupby('month').complaints.sum()
l = ['Apr','Aug','Dec','Feb','Jan','Jul','Jun','Mar','May','Nov','Oct','Sep']
f_df = pd.DataFrame()
```

In [20]:

```
#individuals columns in pandas dataframe only take list as input
f_df['month'] = l
v = []
for val in f:
    v.append(val);
f_df['frequencies'] = v
f_df
```

Out[20]:

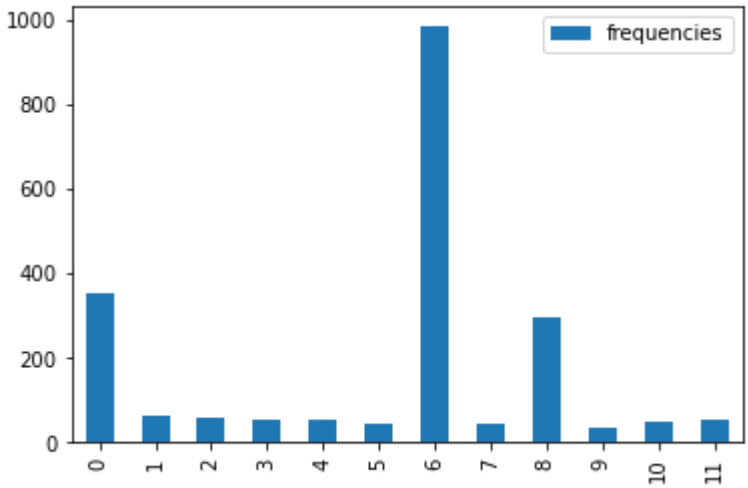
	month	frequencies
0	Apr	351
1	Aug	61
2	Dec	59
3	Feb	54
4	Jan	52
5	Jul	44
6	Jun	984
7	Mar	42
8	May	297
9	Nov	35
10	Oct	47
11	Sep	51

In [21]:

```
f_df.plot.bar()
```

Out[21]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd62669e6a0>
```



In [22]:

```
#=================================================================
#question 2 begins
import pandas as pd
import numpy as np
from sklearn import linear_model
```

In [23]:

```
df = pd.read_csv('Mart.csv')
df
```

Out[23]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemploy
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	
...	...	...	...	...	...	...	...	
6430	45	28-09-2012	713173.95	0	64.88	3.997	192.013558	
6431	45	05-10-2012	733455.07	0	64.89	3.985	192.170412	
6432	45	12-10-2012	734464.36	0	54.47	4.000	192.327265	
6433	45	19-10-2012	718125.53	0	56.47	3.969	192.330854	
6434	45	26-10-2012	760281.43	0	58.85	3.882	192.308899	

6435 rows × 8 columns

In [24]:

```
g = df.groupby('Store')
g
```

Out[24]:

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fd62650dd30
>
```

In [25]:

```
stores = g['Weekly_Sales'].max()
stores
```

Out[25]:

```
Store
1     2387950.20
2     3436007.68
3      605990.41
4     3676388.98
5      507900.07
6     2727575.18
7     1059715.27
8     1511641.09
9      905324.68
10    3749057.69
11    2306265.36
12    1768249.89
13    3595903.20
14    3818686.45
15    1368318.17
16    1004730.69
17    1309226.79
18    2027507.15
19    2678206.42
20    3766687.43
21    1587257.78
22    1962445.04
23    2734277.10
24    2386015.75
25    1295391.19
26    1573982.47
27    3078162.08
28    2026026.39
29    1130926.79
30     519354.88
31    2068942.97
32    1959526.96
33     331173.51
34    1620748.25
35    1781866.98
36     489372.02
37     605791.46
38     499267.66
39    2554482.84
40    1648829.18
41    2263722.68
42     674919.45
43     725043.04
44     376233.89
45    1682862.03
Name: Weekly_Sales, dtype: float64
```

In [26]:

```
#Part 1 output both the store with highest sale as well as the sale itself
c = 0
max_store_number = c
val = 0
for value in stores:
    c += 1
    if val < value:
        val = value
        max_store_number = c
print(val,max_store_number)
```

```
3818686.45 14
```

In [27]:

```
#getting the standard deviations and then finding the maximum
standard_deviations = g['Weekly_Sales'].std()
```

In [28]:

```
standard_deviations
```

Out[28]:

```
Store
1     155980.767761
2     237683.694682
3      46319.631557
4     266201.442297
5      37737.965745
6     212525.855862
7     112585.469220
8     106280.829881
9      69028.666585
10    302262.062504
11    165833.887863
12    139166.871880
13    265506.995776
14    317569.949476
15    120538.652043
16     85769.680133
17    112162.936087
18    176641.510839
19    191722.638730
20    275900.562742
21    128752.812853
22    161251.350631
23    249788.038068
24    167745.677567
25    112976.788600
26    110431.288141
27    239930.135688
28    181758.967539
29     99120.136596
30     22809.665590
31    125855.942933
32    138017.252087
33     24132.927322
34    104630.164676
35    211243.457791
36     60725.173579
37     21837.461190
38     42768.169450
39    217466.454833
40    119002.112858
41    187907.162766
42     50262.925530
43     40598.413260
44     24762.832015
45    130168.526635
Name: Weekly_Sales, dtype: float64
```

In [29]:

```
#finding the maximum standard deviation and the store number corresponding to it pa
c = 0
max_store_number = c
val = 0
for value in standard_deviations:
    c += 1
    if val < value:
        val = value
        max_store_number = c
print(val,max_store_number)
```

```
317569.9494755081 14
```

In [30]:

```
g = df['Weekly_Sales']
```

In [31]:

```
#part 3 of question2
mean_sales = g.mean()
mean_sales
```

Out[31]:

```
1046964.8775617732
```

In [32]:

```
g = df[(df.Holiday_Flag == 1)]
g
```

Out[32]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemploy
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	
31	1	10-09-2010	1507460.69	1	78.69	2.565	211.495190	
42	1	26-11-2010	1955624.11	1	64.52	2.735	211.748433	
47	1	31-12-2010	1367320.01	1	48.43	2.943	211.404932	
53	1	11-02-2011	1649614.93	1	36.39	3.022	212.936705	
...	...	...	...	...	...	...	...	
6375	45	09-09-2011	746129.56	1	71.48	3.738	186.673738	
6386	45	25-11-2011	1170672.94	1	48.71	3.492	188.350400	
6391	45	30-12-2011	869403.63	1	37.79	3.389	189.062016	
6397	45	10-02-2012	803657.12	1	37.00	3.640	189.707605	
6427	45	07-09-2012	766512.66	1	75.70	3.911	191.577676	

450 rows × 8 columns

In [33]:

```
h = g['Weekly_Sales']
```

In [34]:

```
#storing all dates of holydays in the list and outputing them according to the cond
holidays = []
for val in h:
    if val > mean_sales:
        dt = g[g.Weekly_Sales == val].Date
        holidays.append(dt)
holidays #holds the required answer
```

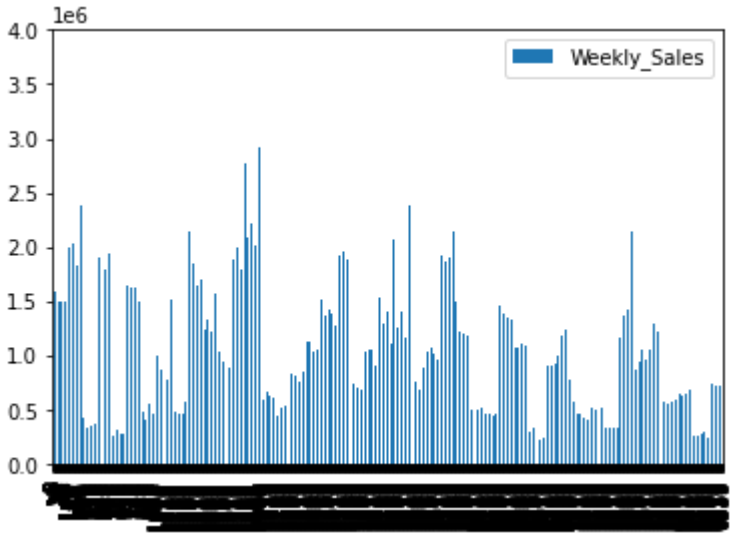
```
42    26-11-2010
Name: Date, dtype: object,
47    31-12-2010
Name: Date, dtype: object,
53    11-02-2011
Name: Date, dtype: object,
83    09-09-2011
Name: Date, dtype: object,
94    25-11-2011
Name: Date, dtype: object,
99    30-12-2011
Name: Date, dtype: object,
105    10-02-2012
Name: Date, dtype: object,
135    07-09-2012
Name: Date, dtype: object,
144    12-02-2010
Name: Date, dtype: object,
174    10-09-2010
Name: Date, dtype: object,
```

In [35]:

```
#part 4 of question2 extracting six months data manually and storing them in differ
#give a bar plot for yearly statics month wise
#this part is doudbt to be discussed with instructor
def getMonth(s):
    return s.split('-')[1]
def getYear(s):
    return s.split('-')[2]
df['Month_Number'] = df['Date'].apply(lambda x : getMonth(x))
df['Year_Number'] = df['Date'].apply(lambda x : getYear(x))
y1_df = pd.DataFrame()
y2_df = pd.DataFrame()
y3_df = pd.DataFrame()
y1_df['Weekly_Sales'] = df[df['Year_Number'] == '2010']['Weekly_Sales']
y1_df['Month_Number'] = df[df['Year_Number'] == '2010']['Month_Number']
y2_df['Month_Number'] = df[df['Year_Number'] == '2011']['Month_Number']
y2_df['Weekly_Sales'] = df[df['Year_Number'] == '2011']['Weekly_Sales']
y3_df['Weekly_Sales'] = df[df['Year_Number'] == '2012']['Weekly_Sales']
y3_df['Month_Number'] = df[df['Year_Number'] == '2012']['Month_Number']
y1_df.plot.bar()
```

Out[35]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd6264a7be0>
```

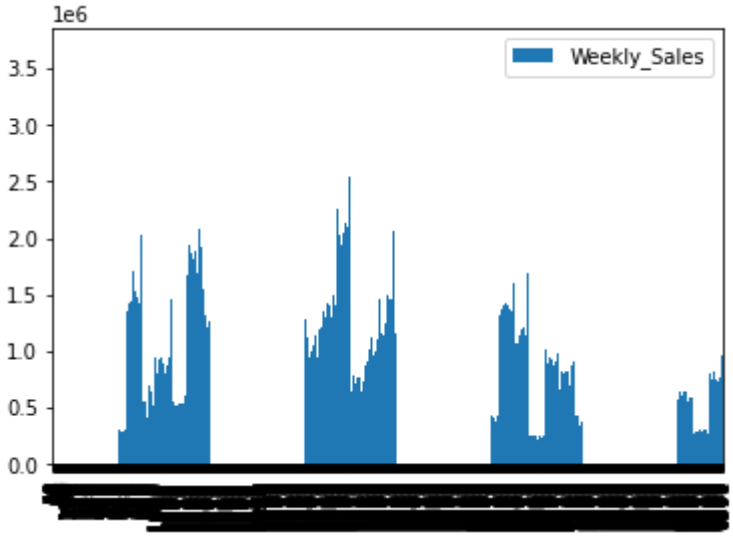


In [36]:

```
y2_df.plot.bar()
```

Out[36]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd624edb580>
```

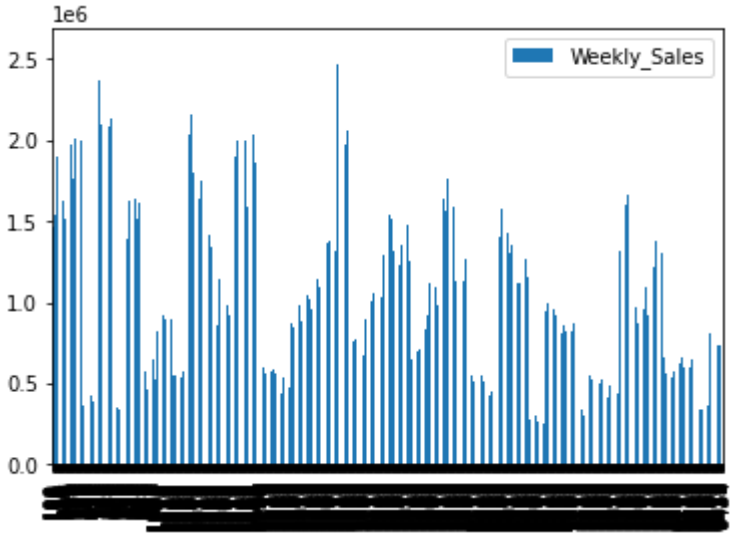


In [37]:

```
y3_df.plot.bar()
```

Out[37]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd62320faf0>
```



In [38]:

```
#statistical model part
import datetime as dt
g = df[df['Store']==2]
```

In [39]:

```
reg = linear_model.LinearRegression()
new_df = g.drop(['Weekly_Sales','Holiday_Flag','Temperature','Store','Fuel_Price'],
new_df['Date'] = pd.to_datetime(new_df['Date'])
new_df['Date'] = pd.to_numeric(new_df['Date'])
new_df
```

Out[39]:

	Date	CPI	Unemployment	Month_Number	Year_Number
143	1272758400000000000	210.752605	8.324	02	2010
144	1291248000000000000	210.897994	8.324	02	2010
145	1266537600000000000	210.945160	8.324	02	2010
146	1267142400000000000	210.975957	8.324	02	2010
147	1272844800000000000	211.006754	8.324	03	2010
...	...	...	...	...	...
281	1348790400000000000	222.616433	6.565	09	2012
282	1336608000000000000	222.815930	6.170	10	2012
283	1355097600000000000	223.015426	6.170	10	2012
284	1350604800000000000	223.059808	6.170	10	2012
285	1351209600000000000	223.078337	6.170	10	2012

143 rows × 5 columns

In [40]:

```
weekly_sales = g['Weekly_Sales']
```

In [41]:

```
reg.fit(new_df,weekly_sales)
```

Out[41]:

```
LinearRegression()
```

In [42]:

```
#CPI and Unemployment rates do not have any impact on the weekly sales
reg.coef_
```

Out[42]:

```
array([-5.50235846e-13,  0.00000000e+00,  0.00000000e+00,  0.00000000e
+00,
        0.00000000e+00])
```

In [ ]:

```
```