In [1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity='all'

import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:
```python
employee = pd.DataFrame({'Name':['Babubhai','Jagjivandas','Tribhuvandas','Zaveri'],
          'Joining_date':['04/04/93','06/08/98','01/05/90','15/07/92'],
          'Emp_id':['E120','E240','E360','E480']})
```

In [3]:
```python
employee
```

Out[3]:

	Name	Joining_date	Emp_id
0	Babubhai	04/04/93	E120
1	Jagjivandas	06/08/98	E240
2	Tribhuvandas	01/05/90	E360
3	Zaveri	15/07/92	E480

In [4]:
```python
employee.dtypes
```

Out[4]:
```
Name            object
Joining_date    object
Emp_id          object
dtype: object
```

In [5]:
```python
## changing datatype of datetime
employee['Joining_date']=pd.to_datetime(employee['Joining_date'])
employee.dtypes
employee
```

Out[5]:
```
Name                    object
Joining_date    datetime64[ns]
Emp_id                  object
dtype: object
```

Out[5]:

	Name	Joining_date	Emp_id
0	Babubhai	1993-04-04	E120
1	Jagjivandas	1998-06-08	E240
2	Tribhuvandas	1990-01-05	E360
3	Zaveri	1992-07-15	E480

In [6]:
```python
## extracing day,month,year,week etc all the details from datetime columnn
```

In [7]:
```python
employee['Month']=employee.Joining_date.dt.month
employee['Day']=employee.Joining_date.dt.day
employee['Year']=employee.Joining_date.dt.year
employee['Week_Number']=employee.Joining_date.dt.isocalendar().week ## which week in the year
employee['Day_of_the_week'] = employee.Joining_date.dt.dayofweek
employee['Day_Name'] = pd.to_datetime(employee['Joining_date']).dt.day_name() # which day in the week
employee['Month_Name']=pd.to_datetime(employee['Joining_date']).dt.month_name()
```

In [8]:
```python
employee
```

Out[8]:

	Name	Joining_date	Emp_id	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name
0	Babubhai	1993-04-04	E120	4	4	1993	13	6	Sunday	April
1	Jagjivandas	1998-06-08	E240	6	8	1998	24	0	Monday	June
2	Tribhuvandas	1990-01-05	E360	1	5	1990	1	4	Friday	January
3	Zaveri	1992-07-15	E480	7	15	1992	29	2	Wednesday	July

%d%/m/%y using strftime()

In [9]:
```python
## changing date time format to %d%m%y format, we use strftime()
```

In [10]:
```python
## strftime() will convert datetime format into STRING format
employee['Joining_date']=pd.to_datetime(employee['Joining_date']).dt.strftime('%d/%m/%y')
employee
print(employee['Joining_date'].dtypes)
```

Out[10]:

	Name	Joining_date	Emp_id	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name
0	Babubhai	04/04/93	E120	4	4	1993	13	6	Sunday	April
1	Jagjivandas	08/06/98	E240	6	8	1998	24	0	Monday	June
2	Tribhuvandas	05/01/90	E360	1	5	1990	1	4	Friday	January
3	Zaveri	15/07/92	E480	7	15	1992	29	2	Wednesday	July

```
object
```

In [11]:
```python
## you can see it is being converted into a string object
```

%d-%m-%y using strftime()

In [12]:
```python
employee['Joining_date']=pd.to_datetime(employee['Joining_date']).dt.strftime('%d-%m-%y')
employee
```

Out[12]:

	Name	Joining_date	Emp_id	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name
0	Babubhai	04-04-93	E120	4	4	1993	13	6	Sunday	April
1	Jagjivandas	06-08-98	E240	6	8	1998	24	0	Monday	June
2	Tribhuvandas	01-05-90	E360	1	5	1990	1	4	Friday	January
3	Zaveri	15-07-92	E480	7	15	1992	29	2	Wednesday	July

In [13]:
```python
## Let us find employees who joined after 15-07-1992
```

In [14]:
```python
from datetime import date
employee[pd.to_datetime(employee['Joining_date']) > pd.Timestamp(date(1992,7,15))]
```

Out[14]:

	Name	Joining_date	Emp_id	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name
0	Babubhai	04-04-93	E120	4	4	1993	13	6	Sunday	April
1	Jagjivandas	06-08-98	E240	6	8	1998	24	0	Monday	June

In [15]:
```python
## let find all the employees who joined between 1990 and 1998

from datetime import date

from datetime import date
employee[(pd.to_datetime(employee['Joining_date']) > pd.Timestamp(date(1990,1,4))) &
(pd.to_datetime(employee['Joining_date']) < pd.Timestamp(date(1998,7,10)))]
```

Out[15]:

	Name	Joining_date	Emp_id	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name
0	Babubhai	04-04-93	E120	4	4	1993	13	6	Sunday	April
1	Jagjivandas	06-08-98	E240	6	8	1998	24	0	Monday	June
2	Tribhuvandas	01-05-90	E360	1	5	1990	1	4	Friday	January
3	Zaveri	15-07-92	E480	7	15	1992	29	2	Wednesday	July

which is the minimum & maximum date in the dataframe

In [16]:
```python
pd.to_datetime(employee['Joining_date'].min())
```

Out[16]:
```
Timestamp('1990-01-05 00:00:00')
```

In [17]:
```python
pd.to_datetime(employee['Joining_date'].max())
```

Out[17]:
```
Timestamp('1992-07-15 00:00:00')
```

Current Timestamp

In [18]:
```python
## Current TimeStamp
timestamp=pd.to_datetime('now')
print('TimeStamp:{}'.format(timestamp))
```

```
TimeStamp:2021-04-10 10:50:57.837830
```

In [19]:
```python
## Current Day- the day this notebook was created

current_date=pd.to_datetime('now').date()
print('Current_Date:{}'.format(current_date))
```

```
Current_Date:2021-04-10
```

In [20]:
```python
## yesterday
yesterday=pd.to_datetime('now') - pd.Timedelta('1 day')
print('Yesterday:{}'.format(yesterday))
```

```
Yesterday:2021-04-09 10:50:57.885831
```

In [21]:
```python
## tomorrow
tomorrow=pd.to_datetime('now') + pd.Timedelta('1 day')
print('tomorrow:{}'.format(tomorrow))
```

```
tomorrow:2021-04-11 10:50:57.925830
```

In [22]:
```python
## alternate option for tomorrow
tomorrow=pd.to_datetime('now') + pd.DateOffset(days=1)
print('tomorrow:{}'.format(tomorrow))
```

```
tomorrow:2021-04-11 10:50:57.949836
```

In [23]:
```python
## Add Business day to current date --> Saturday and Sunday is excluded

add_business_day=pd.to_datetime('now').date() + pd.offsets.BDay(1)
print('add_business_day:{}'.format(add_business_day))
```

```
add_business_day:2021-04-12 00:00:00
```

In [24]:
```python
## Adding 1 month to the current date
add_one_month = pd.to_datetime('now').date() + pd.DateOffset(months=1)
print('Date after adding one month:{}'.format(add_one_month))
```

```
Date after adding one month:2021-05-10 00:00:00
```

In [25]:
```python
## Calculating Date difference in hours
diff_in_hrs = (pd.to_datetime('2021-04-10 21:05:11') - pd.to_datetime('2021-04-01')).total_seconds() // 3600
print('Date Difference in hours:{}'.format(diff_in_hrs))
```

```
Date Difference in hours:237.0
```

In [26]:
```python
## How many years is the employe with the company (Extract year from the current time and subtact from year column)

employee['no_of_years'] = pd.to_datetime('now').year - pd.to_datetime(employee['Joining_date']).dt.year
employee
```

Out[26]:

	Name	Joining_date	Emp_id	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name	no_of_years
0	Babubhai	04-04-93	E120	4	4	1993	13	6	Sunday	April	28
1	Jagjivandas	06-08-98	E240	6	8	1998	24	0	Monday	June	23
2	Tribhuvandas	01-05-90	E360	1	5	1990	1	4	Friday	January	31
3	Zaveri	15-07-92	E480	7	15	1992	29	2	Wednesday	July	29

Female birth dataset :

In [27]:
```python
female_birth = pd.read_csv('https://raw.githubusercontent.com/jbrownlee/Datasets/master/daily-total-female-births.csv')
female_birth.head(10)
```

Out[27]:

	Date	Births
0	1959-01-01	35
1	1959-01-02	32
2	1959-01-03	30
3	1959-01-04	31
4	1959-01-05	44
5	1959-01-06	29
6	1959-01-07	45
7	1959-01-08	43
8	1959-01-09	38
9	1959-01-10	27

In [28]:
```python
pd.to_datetime(female_birth['Date'].min()) #finding the min date
```

Out[28]:
```
Timestamp('1959-01-01 00:00:00')
```

In [29]:
```python
pd.to_datetime(female_birth['Date'].max()) #finding the max date
```

Out[29]:
```
Timestamp('1959-12-31 00:00:00')
```

In [30]:
```python
## difference between the min and max date
pd.to_datetime(female_birth['Date'].max()) - pd.to_datetime(female_birth['Date'].min())
```

Out[30]:
```
Timedelta('364 days 00:00:00')
```

In [31]:
```python
# checking the data type of Date column
female_birth['Date'].dtypes
```

Out[31]:
```
dtype('O')
```

In [32]:
```python
# changing it
female_birth['Date']=pd.to_datetime(female_birth['Date'])
```

In [33]:
```python
female_birth.dtypes
## it is changed
```

Out[33]:
```
Date      datetime64[ns]
Births             int64
dtype: object
```

In [34]:
```python
## let us create the helper columns here just how we created it earlier
```

In [35]:
```python
female_birth['Month']=female_birth.Date.dt.month
female_birth['Day']= female_birth.Date.dt.day
female_birth['Year']=female_birth.Date.dt.year
female_birth['Week_Number']=female_birth.Date.dt.isocalendar().week ## which week in the year
female_birth['Day_of_the_week'] = female_birth.Date.dt.dayofweek
female_birth['Day_Name'] = pd.to_datetime(female_birth['Date']).dt.day_name() # which day in the week
female_birth['Month_Name']=pd.to_datetime(female_birth['Date']).dt.month_name()
```

In [36]:
```python
female_birth.head()
```

Out[36]:

	Date	Births	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name
0	1959-01-01	35	1	1	1959	1	3	Thursday	January
1	1959-01-02	32	1	2	1959	1	4	Friday	January
2	1959-01-03	30	1	3	1959	1	5	Saturday	January
3	1959-01-04	31	1	4	1959	1	6	Sunday	January
4	1959-01-05	44	1	5	1959	2	0	Monday	January

In [37]:
```python
## Total female births in the month of February
female_birth[female_birth['Month_Name']=='February']['Births'].sum()
```

Out[37]:
```
1148
```

In [38]:
```python
## total number of births in January and February
female_birth[female_birth['Month_Name']=='February']['Births'].sum() + female_birth[female_birth['Month_Name']=='January']['Births'].sum()
```

Out[38]:
```
2361
```

In [39]:
```python
## Total number of female births using for loop

for i in female_birth['Month_Name'].unique():
    print('Female births in {0}:{1}'.format(i,female_birth[female_birth['Month_Name']==i]['Births'].sum()))
```

```
Female births in January:1213
Female births in February:1148
Female births in March:1218
Female births in April:1195
Female births in May:1208
Female births in June:1212
Female births in July:1300
Female births in August:1351
Female births in September:1446
Female births in October:1368
Female births in November:1350
Female births in December:1314
```

In [40]:
```python
## getting female births in each month using groupby
female_birth.groupby('Month_Name').sum()[['Births']]
```

Out[40]:

Month_Name	Births
April	1195
August	1351
December	1314
February	1148
January	1213
July	1300
June	1212
March	1218
May	1208
November	1350
October	1368
September	1446

In [41]:
```python
## it is evident September had maximum number of births and February had the least
```

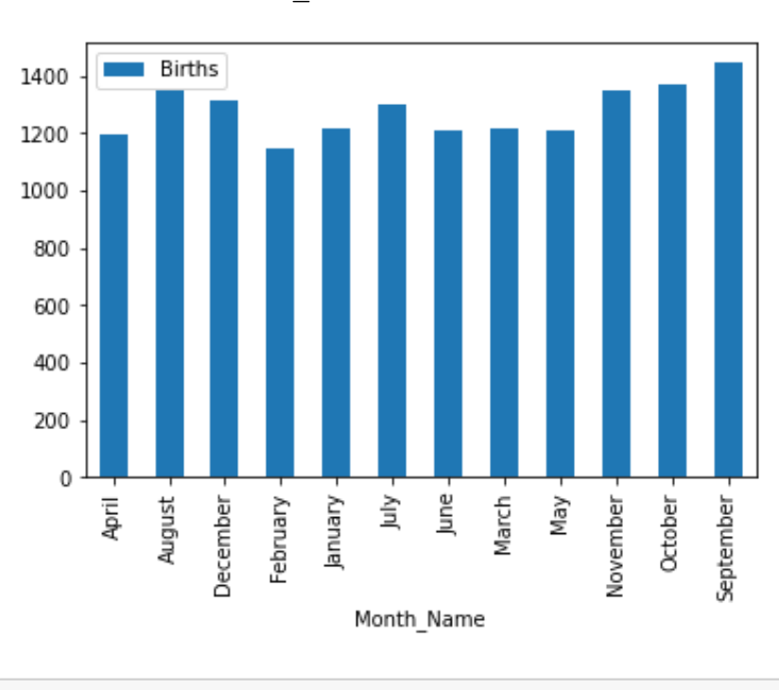
using pivot table for the same

In [42]:
```python
pd.pivot_table(female_birth,values=['Births'],index=['Month_Name'],aggfunc=np.sum)
pd.pivot_table(female_birth,values=['Births'],index=['Month_Name'],aggfunc=np.sum).plot.bar()
```

Out[42]:

Month_Name	Births
April	1195
August	1351
December	1314
February	1148
January	1213
July	1300
June	1212
March	1218
May	1208
November	1350
October	1368
September	1446

Out[42]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2617e066a30>
```



In [43]:
```python
female_birth.dtypes
```

Out[43]:
```
Date               datetime64[ns]
Births                      int64
Month                       int64
Day                         int64
Year                        int64
Week_Number                UInt32
Day_of_the_week             int64
Day_Name                   object
Month_Name                 object
dtype: object
```

In [44]:
```python
## we will convert Month_Name into Categorical variable and then specify the ordering
order = ['January','February','March','April','May','June','July','August','September','October','November','December']
```

In [45]:
```python
female_birth['Month_Name'] = pd.Categorical(female_birth['Month_Name'],order) #converting the month name using order
```

In [46]:
```python
female_birth.groupby('Month_Name').sum()[['Births']]
```

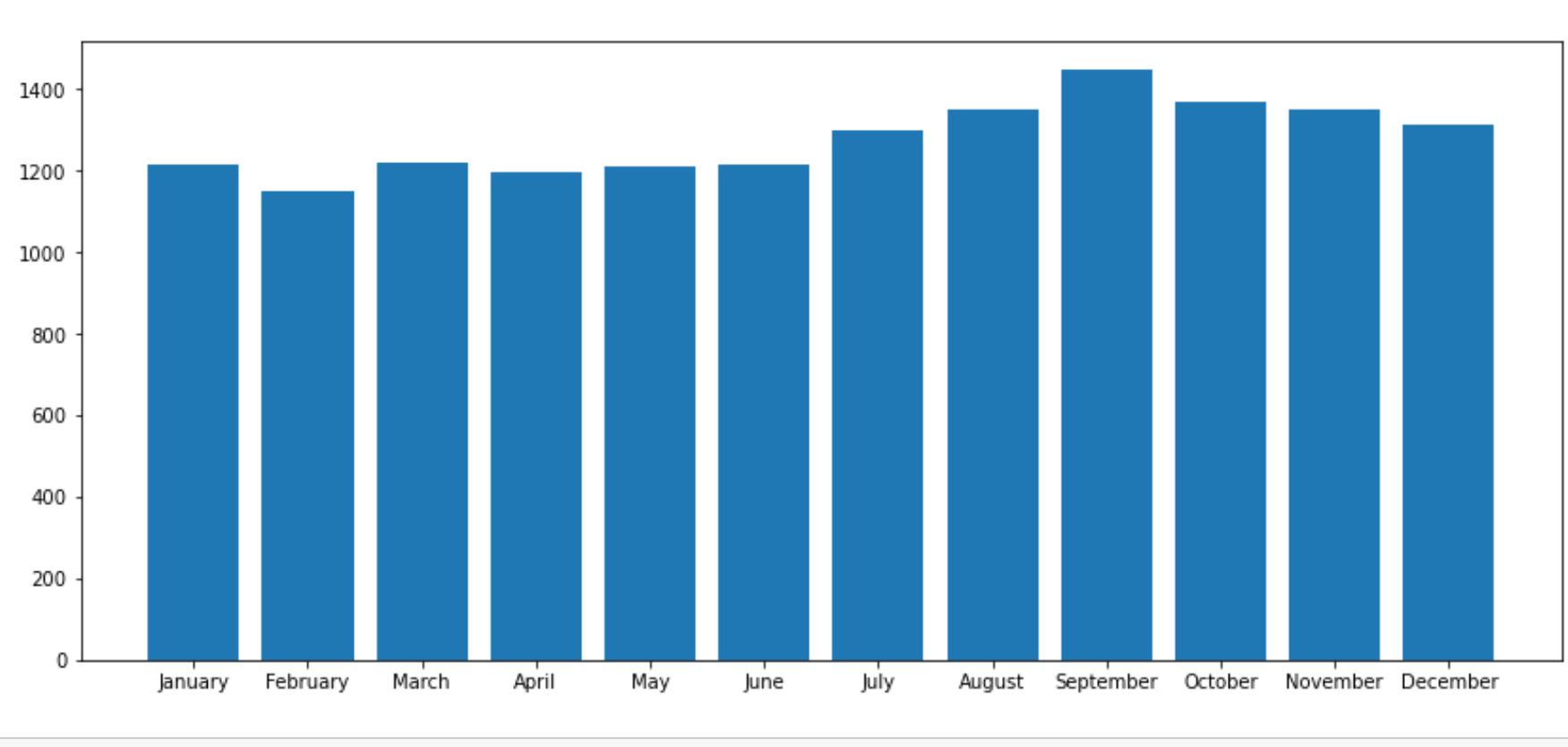
Out[46]:

Month_Name	Births
January	1213
February	1148
March	1218
April	1195
May	1208
June	1212
July	1300
August	1351
September	1446
October	1368
November	1350
December	1314

In [47]:
```python
plt.figure(figsize=(14,6))
plt.bar(female_birth.groupby('Month_Name').sum().index,female_birth.groupby('Month_Name').sum()['Births'])
plt.show()
```

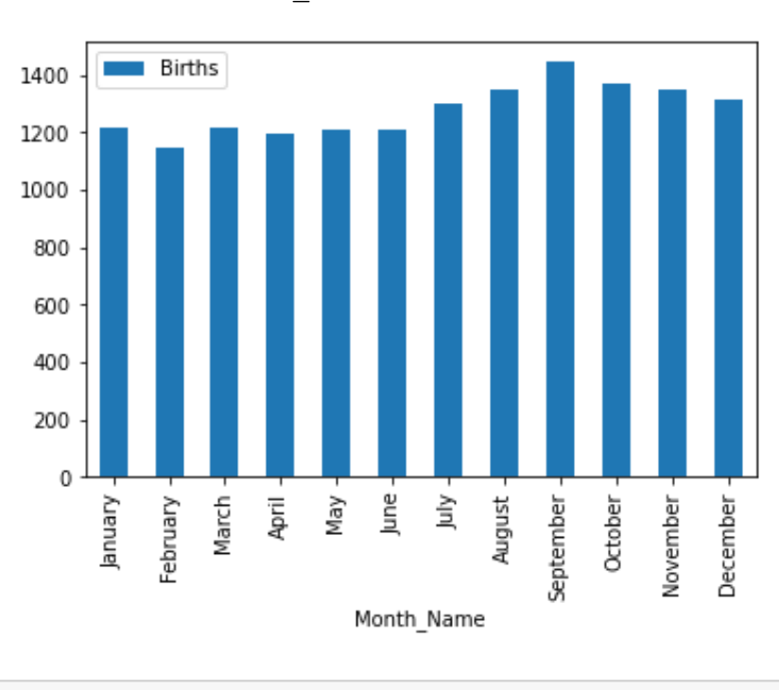
Out[47]:
```
<Figure size 1008x432 with 0 Axes>
```

Out[47]:
```
<BarContainer object of 12 artists>
```



In [48]:
```python
pd.pivot_table(female_birth,values=['Births'],index=['Month_Name'],aggfunc=np.sum).plot.bar()
```

Out[48]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2617e20ae20>
```



In [49]:
```python
## using Day_Name field
order=[ 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

female_birth['Day Name']=pd.Categorical(female_birth['Day_Name'],order)

female_birth.groupby('Day Name').sum()[['Births']]
```

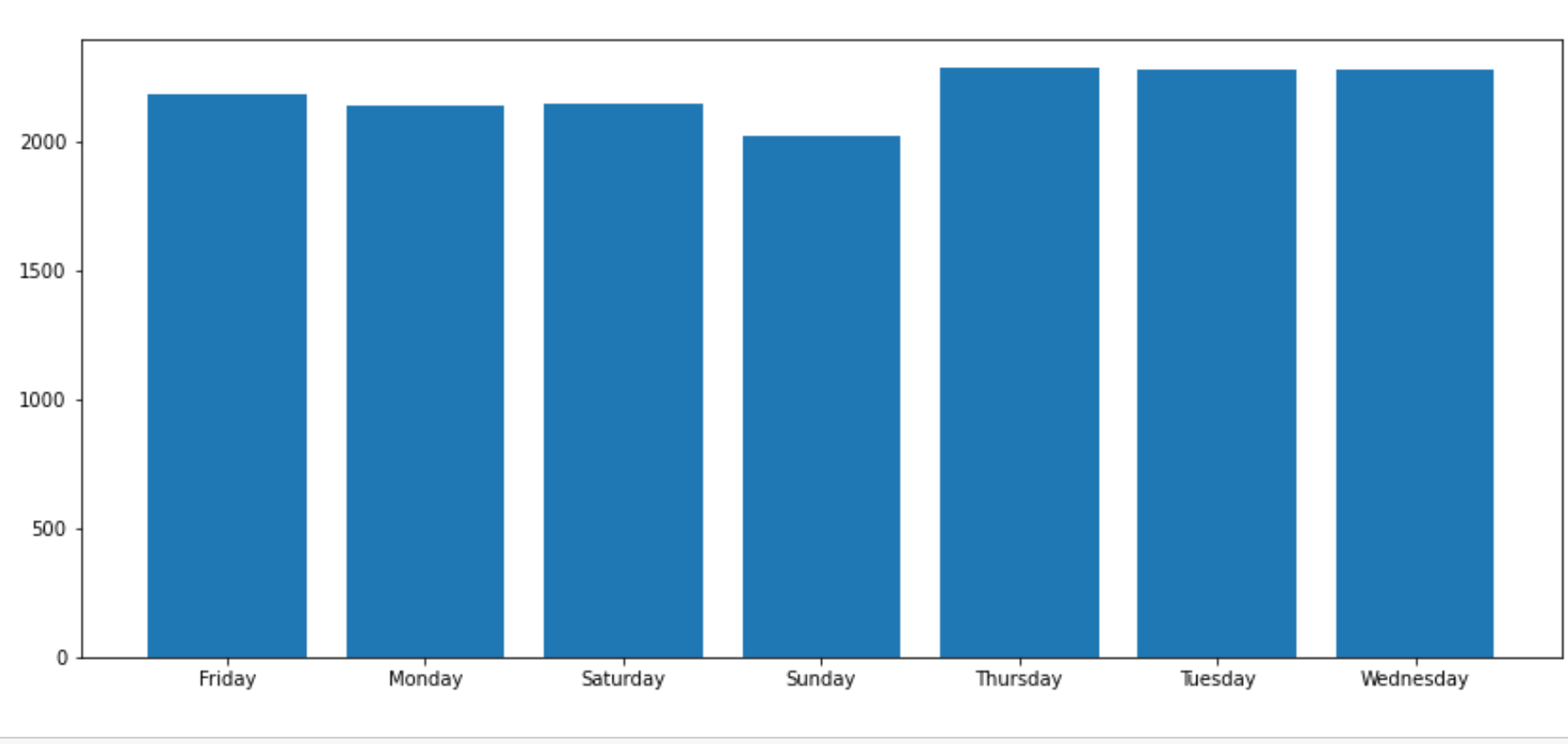
Out[49]:

Day Name	Births
Monday	2139
Tuesday	2275
Wednesday	2280
Thursday	2283
Friday	2182
Saturday	2142
Sunday	2022

In [50]:
```python
plt.figure(figsize=(14,6))
plt.bar(female_birth.groupby('Day_Name').sum().index,female_birth.groupby('Day_Name').sum()['Births'])
plt.show()
```

Out[50]:
```
<Figure size 1008x432 with 0 Axes>
```

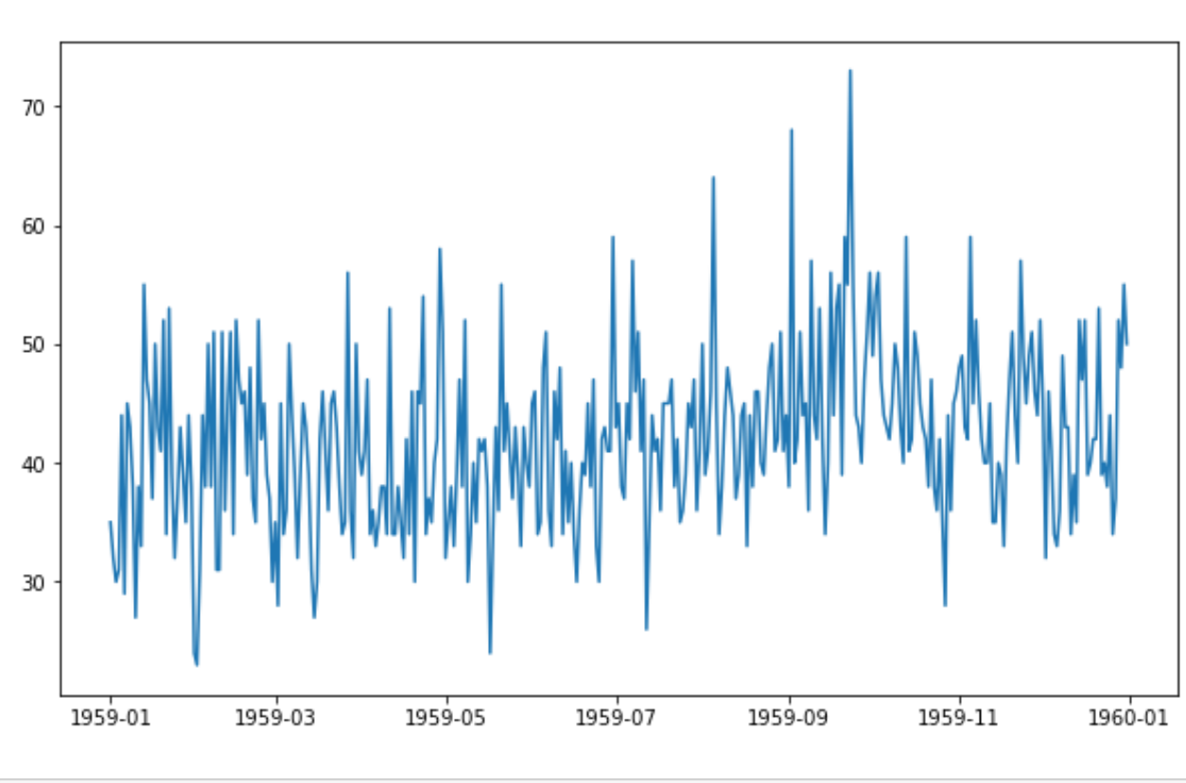
Out[50]:
```
<BarContainer object of 7 artists>
```



In [51]:
```python
## Plotting female births daily
plt.figure(figsize=(10,6))
plt.plot(female_birth['Date'],female_birth['Births'])
```

Out[51]:
```
<Figure size 720x432 with 0 Axes>
```

Out[51]:
```
[<matplotlib.lines.Line2D at 0x2617e2b2fd0>]
```



In [52]:
```python
# Get all records for the month of Janaury(1959-01-01 - 1959-01-31).
# Using boolean is not good method when we are dealing with large datasets.

female_birth[(pd.to_datetime(female_birth['Date']) > pd.Timestamp(date(1959,1,1))) &
 (pd.to_datetime(female_birth['Date']) < pd.Timestamp(date(1959,1,31)))]
```

Out[52]:

	Date	Births	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name	Day Name
1	1959-01-02	32	1	2	1959	1	4	Friday	January	Friday
2	1959-01-03	30	1	3	1959	1	5	Saturday	January	Saturday
3	1959-01-04	31	1	4	1959	1	6	Sunday	January	Sunday
4	1959-01-05	44	1	5	1959	2	0	Monday	January	Monday
5	1959-01-06	29	1	6	1959	2	1	Tuesday	January	Tuesday
6	1959-01-07	45	1	7	1959	2	2	Wednesday	January	Wednesday
7	1959-01-08	43	1	8	1959	2	3	Thursday	January	Thursday
8	1959-01-09	38	1	9	1959	2	4	Friday	January	Friday
9	1959-01-10	27	1	10	1959	2	5	Saturday	January	Saturday
10	1959-01-11	38	1	11	1959	2	6	Sunday	January	Sunday
11	1959-01-12	33	1	12	1959	3	0	Monday	January	Monday
12	1959-01-13	55	1	13	1959	3	1	Tuesday	January	Tuesday
13	1959-01-14	47	1	14	1959	3	2	Wednesday	January	Wednesday
14	1959-01-15	45	1	15	1959	3	3	Thursday	January	Thursday
15	1959-01-16	37	1	16	1959	3	4	Friday	January	Friday
16	1959-01-17	50	1	17	1959	3	5	Saturday	January	Saturday
17	1959-01-18	43	1	18	1959	3	6	Sunday	January	Sunday
18	1959-01-19	41	1	19	1959	4	0	Monday	January	Monday
19	1959-01-20	52	1	20	1959	4	1	Tuesday	January	Tuesday
20	1959-01-21	34	1	21	1959	4	2	Wednesday	January	Wednesday
21	1959-01-22	53	1	22	1959	4	3	Thursday	January	Thursday
22	1959-01-23	39	1	23	1959	4	4	Friday	January	Friday
23	1959-01-24	32	1	24	1959	4	5	Saturday	January	Saturday
24	1959-01-25	37	1	25	1959	4	6	Sunday	January	Sunday
25	1959-01-26	43	1	26	1959	5	0	Monday	January	Monday
26	1959-01-27	39	1	27	1959	5	1	Tuesday	January	Tuesday
27	1959-01-28	35	1	28	1959	5	2	Wednesday	January	Wednesday
28	1959-01-29	44	1	29	1959	5	3	Thursday	January	Thursday
29	1959-01-30	38	1	30	1959	5	4	Friday	January	Friday

In [53]:
```python
## converting date column into datetime index

female_birth=female_birth.set_index(['Date'])
female_birth
```

Out[53]:

Date	Births	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name	Day Name
1959-01-01	35	1	1	1959	1	3	Thursday	January	Thursday
1959-01-02	32	1	2	1959	1	4	Friday	January	Friday
1959-01-03	30	1	3	1959	1	5	Saturday	January	Saturday
1959-01-04	31	1	4	1959	1	6	Sunday	January	Sunday
1959-01-05	44	1	5	1959	2	0	Monday	January	Monday
...	...	...	...	...	...	...	...	...	...
1959-12-27	37	12	27	1959	52	6	Sunday	December	Sunday
1959-12-28	52	12	28	1959	53	0	Monday	December	Monday
1959-12-29	48	12	29	1959	53	1	Tuesday	December	Tuesday
1959-12-30	55	12	30	1959	53	2	Wednesday	December	Wednesday
1959-12-31	50	12	31	1959	53	3	Thursday	December	Thursday

365 rows × 9 columns

In [54]:
```python
## Getting all the data for year 1959
female_birth.loc['1959']
```

Out[54]:

Date	Births	Month	Day	Year	Week_Number	Day_of_the_week	Day_Name	Month_Name	Day Name
1959-01-01	35	1	1	1959	1	3	Thursday	January	Thursday
1959-01-02	32	1	2	1959	1	4	Friday	January	Friday
1959-01-03	30	1	3	1959	1	5	Saturday	January	Saturday
1959-01-04	31	1	4	1959	1	6	Sunday	January	Sunday
1959-01-05	44	1	5	1959	2	0	Monday	January	Monday
...	...	...	...	...	...	...	...	...	...
1959-12-27	37	12	27	1959	52	6	Sunday	December	Sunday
1959-12-28	52	12	28	1959	53	0	Monday	December	Monday
1959-12-29	48	12	29	1959	53	1	Tuesday	December	Tuesday
1959-12-30	55	12	30	1959	53	2	Wednesday	December	Wednesday
1959-12-31	50	12	31	1959	53	3	Thursday	December	Thursday

365 rows × 9 columns

In [55]:
```python
## Births for the month of February [1959-02-01] to [1959-02-28]
```

In [56]:
```python
female_birth.loc['1959-02-01':'1959-02-28'][['Births']]
```

Out[56]:

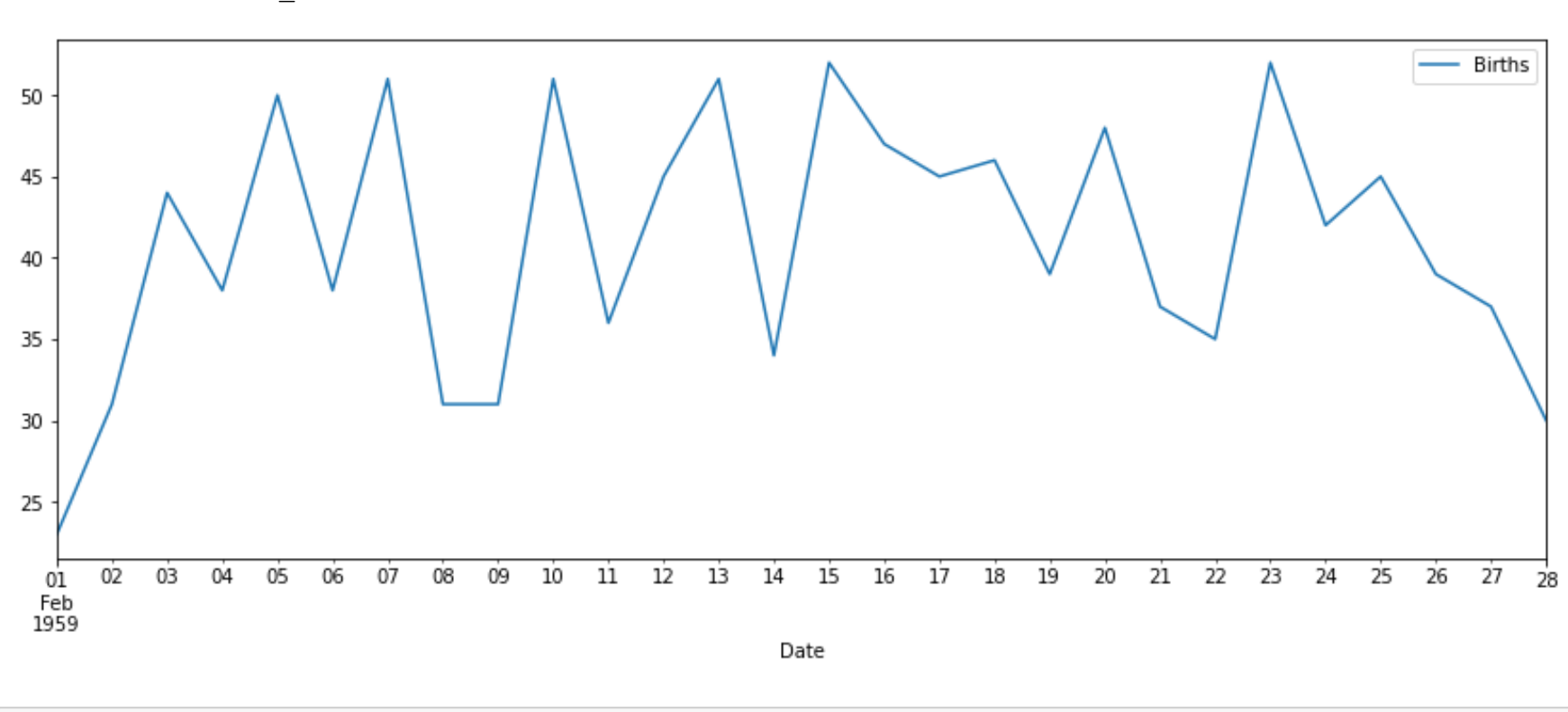
Date	Births
1959-02-01	23
1959-02-02	31
1959-02-03	44
1959-02-04	38
1959-02-05	50
1959-02-06	38
1959-02-07	51
1959-02-08	31
1959-02-09	31
1959-02-10	51
1959-02-11	36
1959-02-12	45
1959-02-13	51
1959-02-14	34
1959-02-15	52
1959-02-16	47
1959-02-17	45
1959-02-18	46
1959-02-19	39
1959-02-20	48
1959-02-21	37
1959-02-22	35
1959-02-23	52
1959-02-24	42
1959-02-25	45
1959-02-26	39
1959-02-27	37
1959-02-28	30

In [57]:
```python
## total number of births in Feb'1959
female_birth.loc['1959-02-01':'1959-02-28'][['Births']].sum()
```

Out[57]:
```
Births    1148
dtype: int64
```

In [58]:
```python
## Plotting line chart for month of february
female_birth.loc['1959-02-01':'1959-02-28'][['Births']].plot(figsize=(14,5))
```

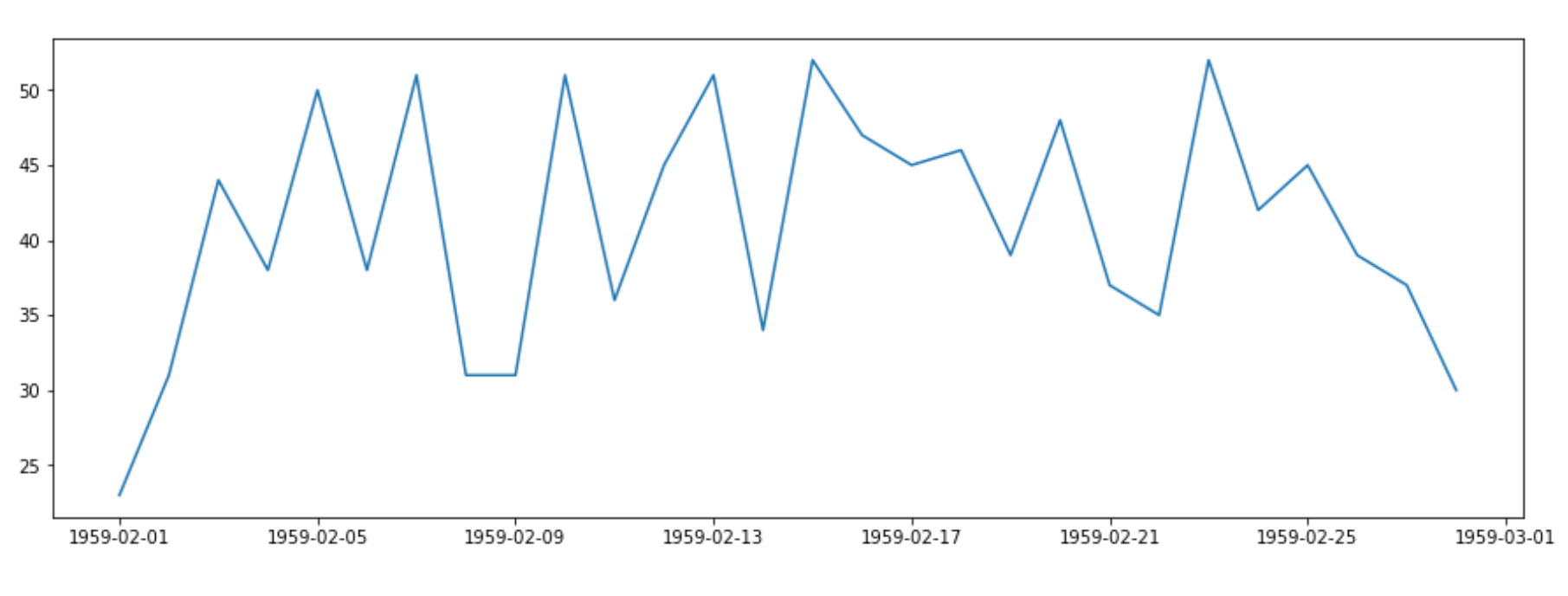
Out[58]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2617e2f4580>
```



In [59]:
```python
# plot line chart for January data using Matplotlib library
plt.figure(figsize=(15,5))
plt.plot(female_birth.loc['1959-02-01':'1959-02-28'][['Births']])
```

Out[59]:
```
<Figure size 1080x360 with 0 Axes>
```

Out[59]:
```
[<matplotlib.lines.Line2D at 0x2617e6672b0>]
```



The End