LAB - 8 Pandas Time Series Analysis

IMPORTING REQUIRED MODULES

In [1]:
```
import pandas as pd
```

In [2]:
```
# setting for pretty plots
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
plt.show()
```

In [3]:
```
# Reading in the data
data = pd.read_csv('amazon_stock.csv')
```

INSPECT TOP 10 ROWS

In [4]:
```
data.head()
```

Out[4]:

	None	ticker	Date	Open	High	Low	Close	Volume	Adj_Close
0	0	AMZN	3/27/2018	1572.40	1575.96	1482.32	1497.05	6793279	1497.05
1	1	AMZN	3/26/2018	1530.00	1556.99	1499.25	1555.86	5547618	1555.86
2	2	AMZN	3/23/2018	1539.01	1549.02	1495.36	1495.56	7843966	1495.56
3	3	AMZN	3/22/2018	1565.47	1573.85	1542.40	1544.10	6177737	1544.10
4	4	AMZN	3/21/2018	1586.45	1590.00	1563.17	1581.86	4667291	1581.86

REMOVE UNWANTED COLUMNS

In [5]:
```
# Remove first two columns (none and ticker) as they dont
#add any value to the dataset,
# Then, print head() to check if removed
data = data.drop(['None','ticker'],axis=1)
```

In [6]:
```
data.head()
```

Out[6]:

	Date	Open	High	Low	Close	Volume	Adj_Close
0	3/27/2018	1572.40	1575.96	1482.32	1497.05	6793279	1497.05
1	3/26/2018	1530.00	1556.99	1499.25	1555.86	5547618	1555.86
2	3/23/2018	1539.01	1549.02	1495.36	1495.56	7843966	1495.56
3	3/22/2018	1565.47	1573.85	1542.40	1544.10	6177737	1544.10
4	3/21/2018	1586.45	1590.00	1563.17	1581.86	4667291	1581.86

In [7]:
```
# Look at the datatypes of the various columns , call info()
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1316 entries, 0 to 1315
Data columns (total 7 columns):
Date         1316 non-null object
Open         1316 non-null float64
High         1316 non-null float64
Low          1316 non-null float64
Close        1316 non-null float64
Volume       1316 non-null int64
Adj_Close    1316 non-null float64
dtypes: float64(5), int64(1), object(1)
memory usage: 72.0+ KB
```

INSPECT THE DATATYPES OF COLUMNS

In [8]:
```
data.dtypes
```

Out[8]:
```
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Volume         int64
Adj_Close    float64
dtype: object
```

CONVERT 'DATE' STRING COLUMN INTO ACTUAL DATE OBJECT

In [9]:
```
data['Date'] = pd.to_datetime(data['Date'])
```

In [10]:
```
data.dtypes
```

Out[10]:
```
Date         datetime64[ns]
Open                float64
High                float64
Low                 float64
Close               float64
Volume                int64
Adj_Close           float64
dtype: object
```

LET US CHECK OUR DATA ONCE AGAIN , WITH HEAD()

In [11]:
```
data.head()
```

Out[11]:

	Date	Open	High	Low	Close	Volume	Adj_Close
0	2018-03-27	1572.40	1575.96	1482.32	1497.05	6793279	1497.05
1	2018-03-26	1530.00	1556.99	1499.25	1555.86	5547618	1555.86
2	2018-03-23	1539.01	1549.02	1495.36	1495.56	7843966	1495.56
3	2018-03-22	1565.47	1573.85	1542.40	1544.10	6177737	1544.10
4	2018-03-21	1586.45	1590.00	1563.17	1581.86	4667291	1581.86

SET DATE OBJECT TO BE INDEX

Here Date is one of the columns. But we want date to be the index. So, set date as index for the frame.Make inplace=True'

In [12]:
```
data.set_index(['Date'], inplace=True)
```

In [13]:
```
data.head()
```

Out[13]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-03-27	1572.40	1575.96	1482.32	1497.05	6793279	1497.05
2018-03-26	1530.00	1556.99	1499.25	1555.86	5547618	1555.86
2018-03-23	1539.01	1549.02	1495.36	1495.56	7843966	1495.56
2018-03-22	1565.47	1573.85	1542.40	1544.10	6177737	1544.10
2018-03-21	1586.45	1590.00	1563.17	1581.86	4667291	1581.86

UNDERSTAND STOCK DATA

In [14]:
```
data['Adj_Close'].plot(figsize=(12,6), title = 'Adjusted Closing Price')
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a43c953630>
```

UNDERSTAND DATE TIMEINDEX

In [15]:
```
from datetime import datetime
my_year = 2020
my_month = 5
my_day =1
my_hour = 13
my_minute = 36
my_second = 45
test_date = datetime(my_year, my_month, my_day)
test_date
```

Out[15]:
```
datetime.datetime(2020, 5, 1, 0, 0)
```

In [16]:
```
test_date = datetime(my_year, my_month, my_day,my_hour, my_minute, my_second)
print("The Day is : ", test_date.day)
print("The Hour is : ", test_date.hour)
print("The Month is : ", test_date.month)
```

```
The Day is :  1
The Hour is :  13
The Month is :  5
```

FIND MINIMUM AND MAXIMUM DATES FROM DATA FRAME, CALL INFO() METHOD

In [17]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1316 entries, 2018-03-27 to 2013-01-02
Data columns (total 6 columns):
Open         1316 non-null float64
High         1316 non-null float64
Low          1316 non-null float64
Close        1316 non-null float64
Volume       1316 non-null int64
Adj_Close    1316 non-null float64
dtypes: float64(5), int64(1)
memory usage: 72.0 KB
```

In [18]:
```
print("Minimum Date : ",data.index.min())
print("Maximum date : ",data.index.max())
```

```
Minimum Date :  2013-01-02 00:00:00
Maximum date :  2018-03-27 00:00:00
```

RETRIEVE INDEX OF EARLIEST AND LATEST DATES USING ARGMIN AND ARGMAX

In [19]:
```
print("Minimum Date Location : ",data.index.argmin())
print("Maximum date Location : ",data.index.argmax())
```

```
Minimum Date Location :  1315
Maximum date Location :  0
```

1.RESAMPLING OPERATION

RESAMPLE ENTIRE DATA FRAME

RESAMPLE DATA WITH YEAR END FREQUENCY ('Y') WITH AVERAGE STOCK PRICE

In [20]:
```
data.resample('Y').mean()
```

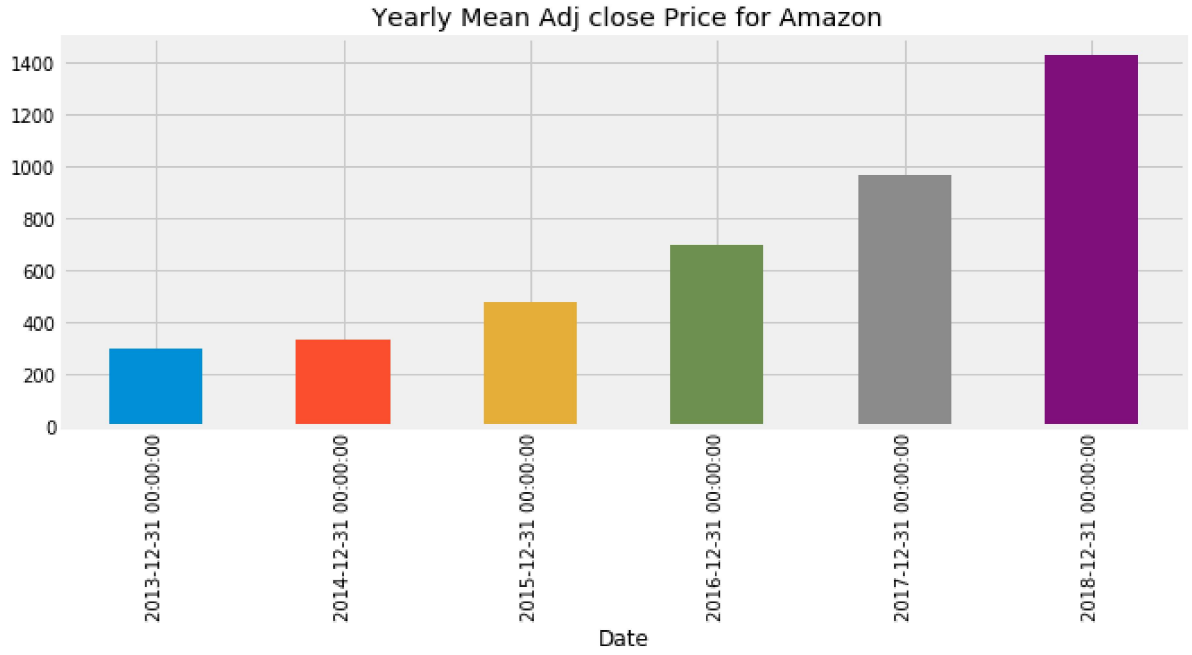
Out[20]:

Date	Open	High	Low	Close	Volume	Adj_Close
2013-12-31	297.877223	300.925966	294.656658	298.032235	2.967880e+06	298.032235
2014-12-31	332.798433	336.317462	328.545440	332.550976	4.083223e+06	332.550976
2015-12-31	478.126230	483.248272	472.875443	478.137321	3.797801e+06	478.137321
2016-12-31	699.669762	705.799103	692.646189	699.523135	4.122043e+06	699.523135
2017-12-31	967.565060	973.789752	959.991826	967.403996	3.466207e+06	967.403996
2018-12-31	1429.770000	1446.701017	1409.469661	1429.991186	5.586829e+06	1429.991186

RESAMPLE A SPECIFIC COLUMN

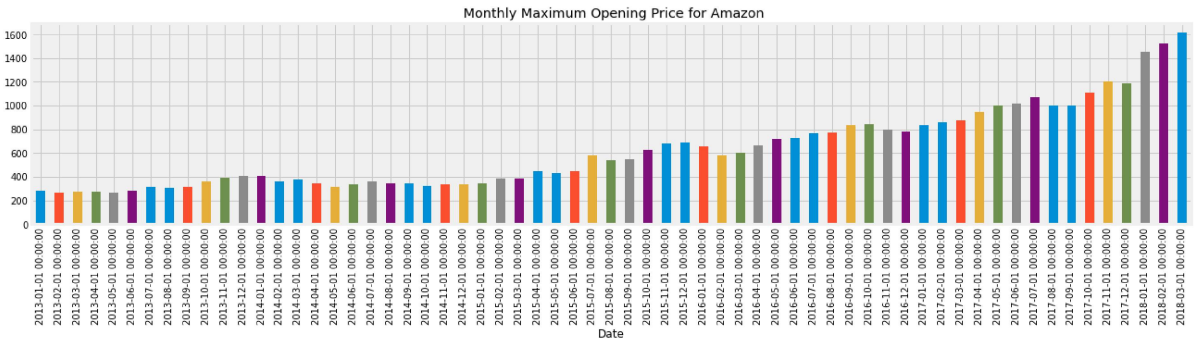
PLOT A BAR CHART TO SHOW THE YEARLY (USE 'A') MEAN ADJUSTED CLOSE PRICE

In [21]:
```
data['Adj_Close'].resample('A').mean().plot(kind = 'bar', figsize=(10,4))
plt.title(" Yearly Mean Adj close Price for Amazon")
plt.show()
```



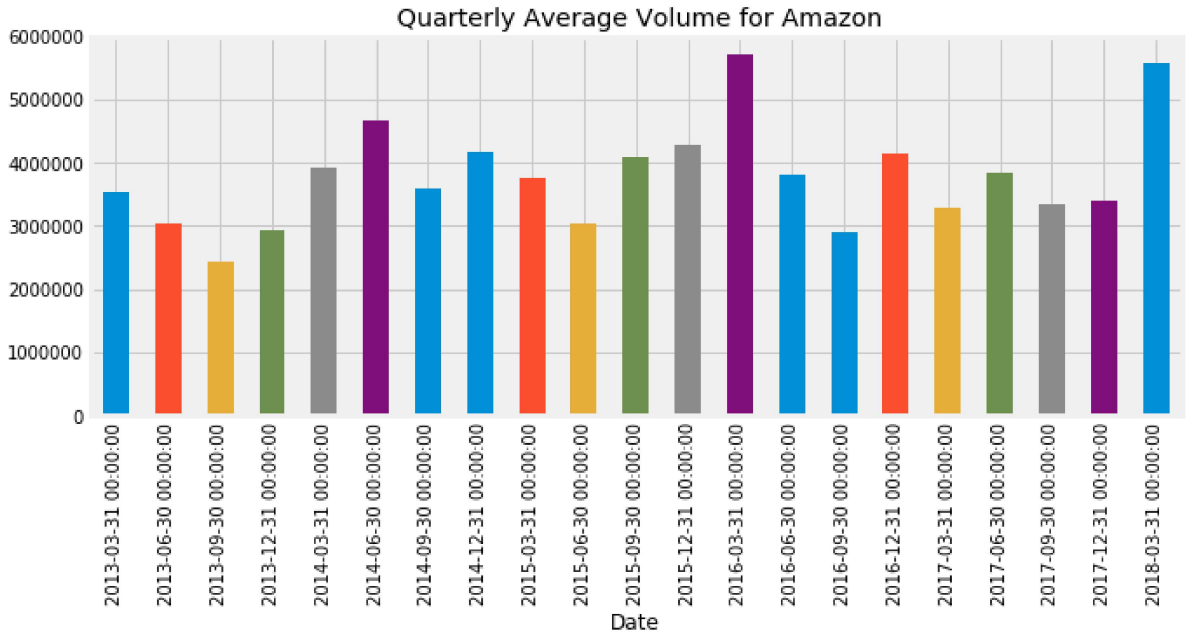
PLOT BAR CHART TO SHOW MONTHLY MAXIMUM (USE 'MS') OPENING PRICE FOR ALL YEARS

In [22]:
```
data['Open'].resample('MS').max().plot(kind = 'bar', figsize=(20,4))
plt.title(" Monthly Maximum Opening Price for Amazon")
plt.show()
```



PLOT BAR CHART OF QUATERLY (USE 'Q') AVERAGE VOLUME FOR ALL YEARS

In [34]:
```
data['Volume'].resample('Q').mean().plot(kind = 'bar', figsize=(10,4))
plt.title(" Quarterly Average Volume for Amazon")
plt.show()
```



2. TIME SHIFTING OPERATIONS

SHIFTING DATA FORWARD AND BACKWARD

SHOW HEAD OF DATA

In [24]:
```
data.head()
```

Out[24]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-03-27	1572.40	1575.96	1482.32	1497.05	6793279	1497.05
2018-03-26	1530.00	1556.99	1499.25	1555.86	5547618	1555.86
2018-03-23	1539.01	1549.02	1495.36	1495.56	7843966	1495.56
2018-03-22	1565.47	1573.85	1542.40	1544.10	6177737	1544.10
2018-03-21	1586.45	1590.00	1563.17	1581.86	4667291	1581.86

SHIFT DATA BY 1 DAY FOWARD

In [25]:
```
data.shift(periods = 1).head()
```

Out[25]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-03-27	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-26	1572.40	1575.96	1482.32	1497.05	6793279.0	1497.05
2018-03-23	1530.00	1556.99	1499.25	1555.86	5547618.0	1555.86
2018-03-22	1539.01	1549.02	1495.36	1495.56	7843966.0	1495.56
2018-03-21	1565.47	1573.85	1542.40	1544.10	6177737.0	1544.10

SHIFT DATA BY 1 DAY BACKWARD

In [26]:
```
data.shift(periods = -1).head()
```

Out[26]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-03-27	1530.00	1556.99	1499.25	1555.86	5547618.0	1555.86
2018-03-26	1539.01	1549.02	1495.36	1495.56	7843966.0	1495.56
2018-03-23	1565.47	1573.85	1542.40	1544.10	6177737.0	1544.10
2018-03-22	1586.45	1590.00	1563.17	1581.86	4667291.0	1581.86
2018-03-21	1550.34	1587.00	1545.41	1586.51	4507049.0	1586.51

SHIFTING TIME INDEX

In [27]:
```
data.head(10)
```

Out[27]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-03-27	1572.40	1575.96	1482.32	1497.05	6793279	1497.05
2018-03-26	1530.00	1556.99	1499.25	1555.86	5547618	1555.86
2018-03-23	1539.01	1549.02	1495.36	1495.56	7843966	1495.56
2018-03-22	1565.47	1573.85	1542.40	1544.10	6177737	1544.10
2018-03-21	1586.45	1590.00	1563.17	1581.86	4667291	1581.86
2018-03-20	1550.34	1587.00	1545.41	1586.51	4507049	1586.51
2018-03-19	1554.53	1561.66	1525.35	1544.93	6376619	1544.93
2018-03-16	1583.45	1589.44	1567.50	1571.68	5145054	1571.68
2018-03-15	1595.00	1596.91	1578.11	1582.32	4026744	1582.32
2018-03-14	1597.00	1606.44	1590.89	1591.00	4164395	1591.00

In [28]:
```
data.shift(periods = 3,freq='MS')
```

Out[28]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-06-01	1572.40	1575.9600	1482.3200	1497.0500	6793279	1497.0500
2018-06-01	1530.00	1556.9900	1499.2500	1555.8600	5547618	1555.8600
2018-06-01	1539.01	1549.0200	1495.3600	1495.5600	7843966	1495.5600
2018-06-01	1565.47	1573.8500	1542.4000	1544.1000	6177737	1544.1000
2018-06-01	1586.45	1590.0000	1563.1700	1581.8600	4667291	1581.8600
2018-06-01	1550.34	1587.0000	1545.4100	1586.5100	4507049	1586.5100
2018-06-01	1554.53	1561.6600	1525.3500	1544.9300	6376619	1544.9300
2018-06-01	1583.45	1589.4400	1567.5000	1571.6800	5145054	1571.6800
2018-06-01	1595.00	1596.9100	1578.1100	1582.3200	4026744	1582.3200
2018-06-01	1597.00	1606.4400	1590.8900	1591.0000	4164395	1591.0000
2018-06-01	1615.96	1617.5400	1578.0100	1588.1800	6427066	1588.1800
2018-06-01	1592.60	1605.3300	1586.7000	1598.3900	5115886	1598.3900
2018-06-01	1563.50	1578.9400	1559.0800	1578.8900	4417059	1578.8900
2018-06-01	1550.00	1554.8800	1545.2500	1551.8600	3512528	1551.8600
2018-06-01	1526.52	1545.9000	1522.5100	1545.0000	4174123	1545.0000
2018-06-01	1533.20	1542.1300	1528.0000	1537.6400	4561718	1537.6400
2018-06-01	1494.24	1525.3800	1481.0000	1523.6100	5233934	1523.6100
2018-06-01	1469.10	1501.0500	1455.0100	1500.2500	6587564	1500.2500
2018-06-01	1513.60	1518.4900	1465.0000	1493.4500	6835230	1493.4500
2018-05-01	1519.51	1528.7000	1512.0000	1512.4500	4426580	1512.4500
2018-05-01	1524.50	1526.7800	1507.2100	1511.9800	4708378	1511.9800
2018-05-01	1509.20	1522.8400	1507.0000	1521.9500	4909053	1521.9500
2018-05-01	1495.34	1500.0000	1486.5000	1500.0000	4327008	1500.0000
2018-05-01	1495.36	1502.5400	1475.7600	1484.7600	4732555	1484.7600
2018-05-01	1485.00	1503.4900	1478.9200	1482.9200	6216694	1482.9200
2018-05-01	1446.49	1488.7700	1446.4900	1468.3500	6388374	1468.3500
2018-05-01	1457.37	1465.8000	1446.5600	1448.6900	4410879	1448.6900
2018-05-01	1466.89	1468.9400	1436.8400	1461.7600	5598111	1461.7600
2018-05-01	1406.25	1452.0600	1403.3600	1451.0500	5881238	1451.0500
2018-05-01	1385.93	1419.7200	1383.5300	1414.5100	5858860	1414.5100
...	...	...	...	...	...	...
2013-05-01	261.53	269.9600	260.3000	269.4700	5293000	269.4700
2013-05-01	259.19	260.1600	257.0000	258.7000	2943700	258.7000
2013-05-01	263.20	263.2500	256.6000	257.2100	3403700	257.2100

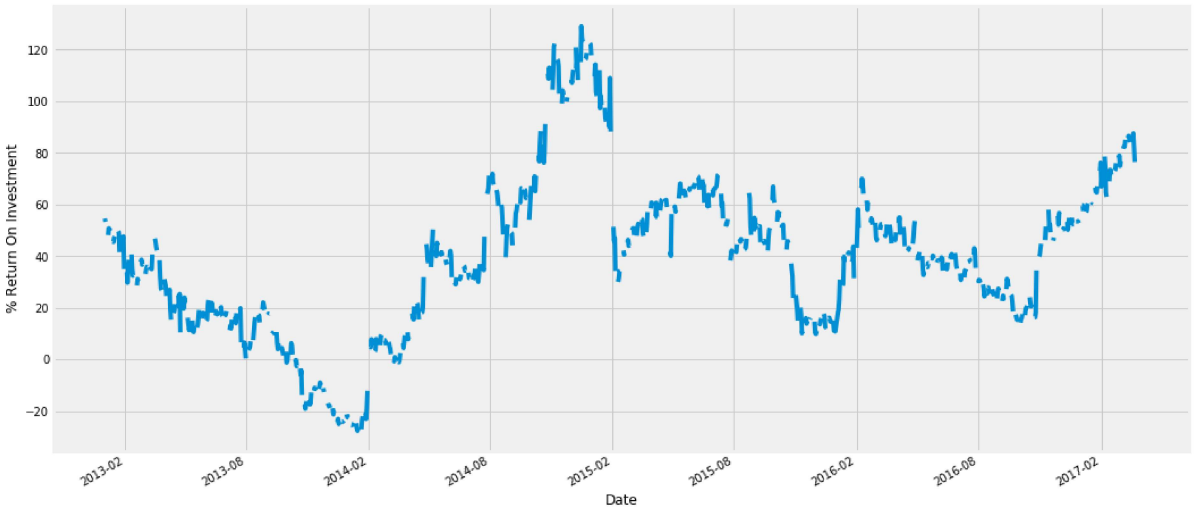
Date	Open	High	Low	Close	Volume	Adj_Close
2013-05-01	261.40	265.2500	260.5550	261.9500	3879200	261.9500
2013-05-01	264.10	264.1000	255.1100	260.2300	3975700	260.2300
2013-05-01	265.16	266.8900	261.1100	262.2200	2770400	262.2200
2013-05-01	262.00	268.0300	261.4600	266.8900	4012900	266.8900
2013-05-01	262.78	264.6840	259.0700	259.9800	3723600	259.9800
2013-05-01	268.93	268.9300	262.8000	265.0000	6115000	265.0000
2013-04-01	271.04	275.9400	263.6991	265.5000	6772100	265.5000
2013-04-01	283.00	284.2000	267.1100	272.7640	13075400	272.7640
2013-04-01	275.35	275.4600	258.3500	260.3500	10172600	260.3500
2013-04-01	283.78	284.4800	274.4000	276.0400	4321400	276.0400
2013-04-01	275.00	284.7200	274.4000	283.9900	4968100	283.9900
2013-04-01	269.37	276.6500	269.3700	273.6200	3417000	273.6200
2013-04-01	270.57	271.0900	266.6500	268.1100	2508900	268.1100
2013-04-01	271.62	272.1000	269.2300	270.1900	2137700	270.1900
2013-04-01	270.83	274.5000	269.6000	272.1200	2942000	272.1200
2013-04-01	271.43	271.9700	269.2100	270.4800	1884600	270.4800
2013-04-01	270.53	271.2400	267.8300	268.9300	2065600	268.9300
2013-04-01	270.68	272.7300	269.3000	271.9000	2326900	271.9000
2013-04-01	268.00	274.2600	267.5400	272.7300	4275000	272.7300
2013-04-01	265.10	268.4300	264.1100	267.9400	2413300	267.9400
2013-04-01	268.54	268.7400	262.3000	265.3400	2863400	265.3400
2013-04-01	268.17	269.5000	265.4010	266.3500	2265600	266.3500
2013-04-01	267.07	268.9800	263.5670	266.3800	3010700	266.3800
2013-04-01	262.97	269.7250	262.6700	268.4592	4910000	268.4592
2013-04-01	257.58	259.8000	256.6500	259.1500	1874200	259.1500
2013-04-01	257.27	260.8800	256.3700	258.4800	2750900	258.4800
2013-04-01	256.08	258.0999	253.2600	257.3100	3271000	257.3100

1316 rows × 6 columns

APPICATION - COMPUTING RETURN ON INVESTMENT

In [29]:
```
ROI = 100* (data['Adj_Close'].tshift(periods = - 365, freq ='D')/data['Adj_Clos
ROI.plot(figsize=(16,8))
plt.ylabel('% Return On Investment')
```

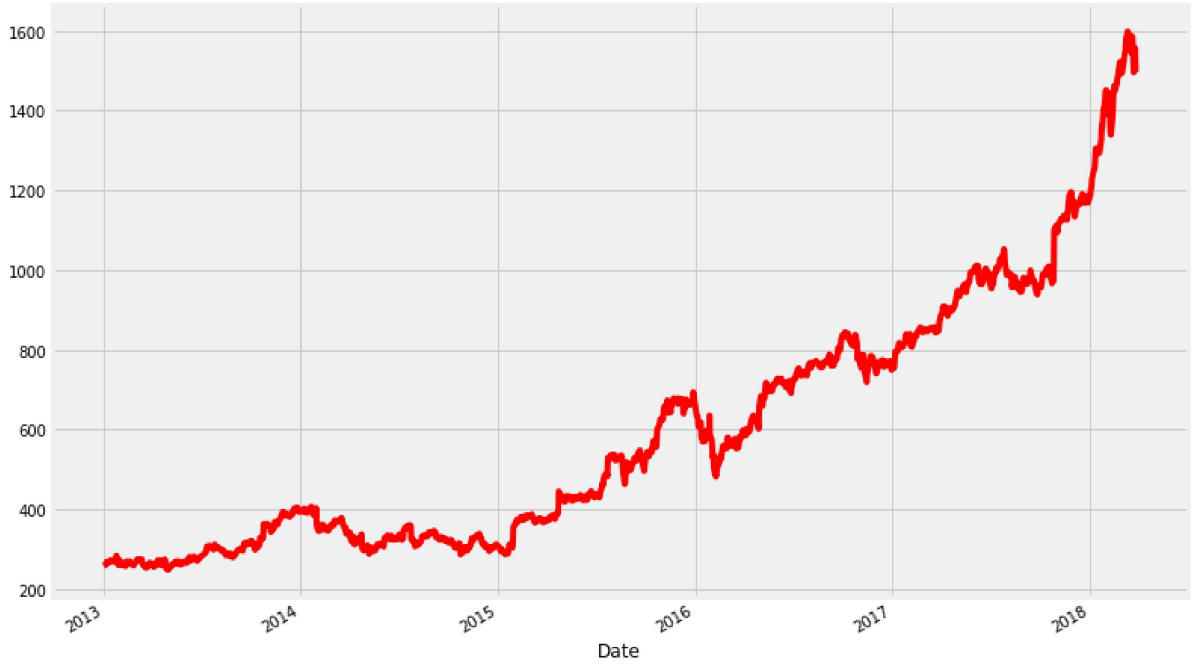
Out[29]: Text(0,0.5,'% Return On Investment')



ROLLING WINDOW OR MOVING WINDOW OPERATIONS

In [30]:
```
data['Adj_Close'].plot(figsize=(12,8), color='red')
```

Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x1a43d077d68>



FIND ROLLIMG MEAN FOR 7 DAYS AND SHOW TOP 10 ROWS

In [31]:
```
data.rolling(7).mean().head(10)
```

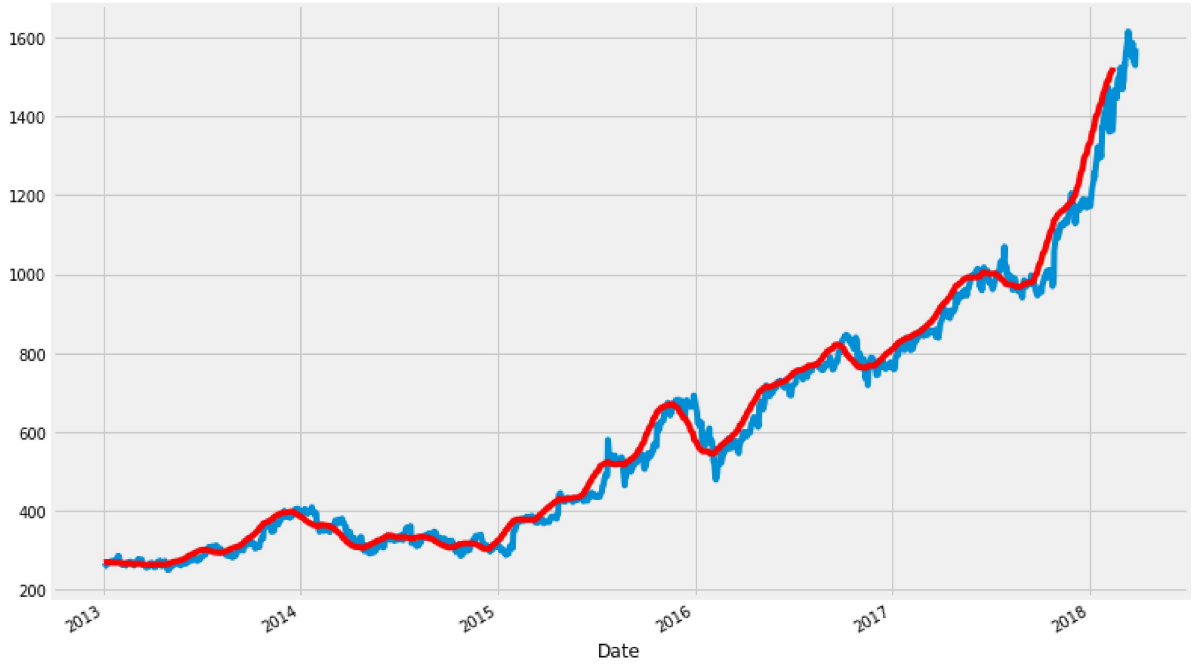
Out[31]:

Date	Open	High	Low	Close	Volume	Adj_Close
2018-03-27	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-26	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-23	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-22	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-21	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-20	NaN	NaN	NaN	NaN	NaN	NaN
2018-03-19	1556.885714	1570.640000	1521.894286	1543.695714	5.987651e+06	1543.695714
2018-03-16	1558.464286	1572.565714	1534.062857	1554.357143	5.752191e+06	1554.357143
2018-03-15	1567.750000	1578.268571	1545.328571	1558.137143	5.534923e+06	1558.137143
2018-03-14	1576.034286	1586.471429	1558.975714	1571.771429	5.009270e+06	1571.771429

PLOT A LINE CHART FOR OPEN COLUMN FOLLOWED BY AVERAGE ROLLING WINDOW OF 30 DAYS ON THE SAME OPEN COLUMN

In [32]:
```
data['Open'].plot(figsize=(12,8))
data['Open'].rolling(30).mean().plot(figsize=(12,8), color='red')
```

Out[32]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a43d116470>
```



In [ ]: