Time Series Analysis

Introduction To Datetime Data Types

In [1]:
```
from datetime import datetime
```

In [2]:
```
datetime?
#datetime??
```

In [110]:
```
#help(datetime)
```

In [4]:
```
time_now = datetime.now()
```

In [5]:
```
time_now
```

Out[5]:
```
datetime.datetime(2020, 2, 14, 2, 25, 22, 358922)
```

In [6]:
```
time_now.year, time_now.month, time_now.day
```

Out[6]:
```
(2020, 2, 14)
```

In [7]:
```
delta = datetime(2020, 1, 17) - datetime(2019, 1, 17, 6, 40)
```

In [8]:
```
delta
```

Out[8]:
```
datetime.timedelta(days=364, seconds=62400)
```

In [9]:
```
delta.days
```

Out[9]:
```
364
```

In [10]:
```
delta.seconds
```
Out[10]:
```
62400
```

In [11]:
```
from datetime import timedelta
```

In [12]:
```
start_time = datetime(2020, 1, 17)
```

In [13]:
```
start_time + timedelta(20)
```
Out[13]:
```
datetime.datetime(2020, 2, 6, 0, 0)
```

In [14]:
```
start_time - timedelta(20)
```
Out[14]:
```
datetime.datetime(2019, 12, 28, 0, 0)
```

In [ ]:
```
time_stamp = datetime(2020, 1, 17)
```

In [ ]:
```
str(time_stamp)
```

In [ ]:
```
time_stamp.strftime('%Y-%m-%d')
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

How To Convert Between String and Datetime?

In [15]:

```
time_stamp = datetime(2020, 1, 17)
```

In [16]:

```
str(time_stamp)
```

Out[16]:

```
'2020-01-17 00:00:00'
```

In [17]:

```
time_stamp.strftime('%Y-%m-%d')
```

Out[17]:

```
'2020-01-17'
```

In [18]:

```
string_date = '2020-01-17'
```

In [19]:

```
datetime.strptime(string_date, '%Y-%m-%d')
```

Out[19]:

```
datetime.datetime(2020, 1, 17, 0, 0)
```

In [20]:

```
date_str = ['7/6/2020', '8/6/2020', '1/17/2020']
```

In [21]:

```
[datetime.strptime(x, '%m/%d/%Y') for x in date_str]
```

Out[21]:

```
[datetime.datetime(2020, 7, 6, 0, 0),
 datetime.datetime(2020, 8, 6, 0, 0),
 datetime.datetime(2020, 1, 17, 0, 0)]
```

dateutil Package

In [22]:
```python
from dateutil.parser import parse
```

In [23]:
```python
parse('2020-01-17')
```

Out[23]:
```
datetime.datetime(2020, 1, 17, 0, 0)
```

In [24]:
```python
parse('Jan 31, 2020 10:45 PM')
```

Out[24]:
```
datetime.datetime(2020, 1, 31, 22, 45)
```

In [25]:
```python
parse('17/1/2020', dayfirst=True)
```

Out[25]:
```
datetime.datetime(2020, 1, 17, 0, 0)
```

Pandas datetime Method

In [26]:
```python
import pandas as pd
```

In [27]:
```python
date_strs = ['2020-01-17 12:00:00', '2021-01-17 00:00:00']
```

In [29]:
```python
pd.to_datetime?
```

In [30]:
```python
pd.to_datetime(date_strs)
```

Out[30]:
```
DatetimeIndex(['2020-01-17 12:00:00', '2021-01-17 00:00:00'], dtype='datetime64[ns]', freq=None)
```

In [31]:
```python
idx = pd.to_datetime(date_strs + [None])
```

In [32]:
```python
idx
```

Out[32]:
```
DatetimeIndex(['2020-01-17 12:00:00', '2021-01-17 00:00:00', 'NaT'], dtype='datetime64[ns]', freq=None)
```

In [33]:

```
idx[2]
```

Out[33]:

```
NaT
```

In [34]:

```
pd.isnull(idx)
```

Out[34]:

```
array([False, False,  True])
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Time Series Basics

Introduction

In [35]:

```
import pandas as pandas
import numpy as np
```

In [36]:

```
from datetime import datetime

dates = [datetime(2020, 1, 17), datetime(2020, 1, 18),
         datetime(2020, 1, 19), datetime(2020, 1, 20),
         datetime(2020, 1, 21), datetime(2020, 1, 22)]
```

In [37]:

```
dates
```

Out[37]:

```
[datetime.datetime(2020, 1, 17, 0, 0),
 datetime.datetime(2020, 1, 18, 0, 0),
 datetime.datetime(2020, 1, 19, 0, 0),
 datetime.datetime(2020, 1, 20, 0, 0),
 datetime.datetime(2020, 1, 21, 0, 0),
 datetime.datetime(2020, 1, 22, 0, 0)]
```

In [38]:

```
time_series = pd.Series(np.random.randn(6), index=dates)
time_series
```

Out[38]:

```
2020-01-17   -0.061701
2020-01-18    2.090595
2020-01-19    1.448367
2020-01-20    1.671970
2020-01-21   -0.688016
2020-01-22   -0.886338
dtype: float64
```

In [39]:

```
time_series.index
```

Out[39]:

```
DatetimeIndex(['2020-01-17', '2020-01-18', '2020-01-19', '2020-01-20',
               '2020-01-21', '2020-01-22'],
              dtype='datetime64[ns]', freq=None)
```

In [40]:

```
time_series.index.dtype
```

Out[40]:

```
dtype('<M8[ns]')
```

In [41]:

```
print(time_series.index[0]); time_series.index[1]
```

```
2020-01-17 00:00:00
```

Out[41]:

```
Timestamp('2020-01-18 00:00:00')
```

Indexing, Selection, Subsetting

In [42]:

```
stamp = time_series.index[2]
```

In [43]:

```
print(time_series)
time_series[stamp]
```

```
2020-01-17   -0.061701
2020-01-18    2.090595
2020-01-19    1.448367
2020-01-20    1.671970
2020-01-21   -0.688016
2020-01-22   -0.886338
dtype: float64
```

Out[43]:

```
1.44836710468517
```

In [44]:

```
time_series['2020-01-19']
```

Out[44]:

```
1.44836710468517
```

In [45]:

```
longer_times = pd.Series(np.random.randn(500),
                         index=pd.date_range('17/1/2020', periods=500))

longer_times
```

Out[45]:

```
2020-01-17   -0.849014
2020-01-18    0.093746
2020-01-19   -1.609437
2020-01-20    1.552665
2020-01-21    0.875436
                ...
2021-05-26    1.552808
2021-05-27    0.514349
2021-05-28   -2.539020
2021-05-29   -0.231783
2021-05-30    1.249881
Freq: D, Length: 500, dtype: float64
```

In [46]:

```
longer_times['2020']
```

Out[46]:

```
2020-01-17   -0.849014
2020-01-18    0.093746
2020-01-19   -1.609437
2020-01-20    1.552665
2020-01-21    0.875436
                ...
2020-12-27   -1.035593
2020-12-28   -0.247575
2020-12-29    0.126091
2020-12-30   -0.748800
2020-12-31    0.867264
Freq: D, Length: 350, dtype: float64
```

In [47]:

```
longer_times['2020-01']
```

Out[47]:

```
2020-01-17   -0.849014
2020-01-18    0.093746
2020-01-19   -1.609437
2020-01-20    1.552665
2020-01-21    0.875436
2020-01-22   -0.003206
2020-01-23   -0.952588
2020-01-24   -1.221554
2020-01-25    1.664579
2020-01-26    0.019152
2020-01-27    0.399794
2020-01-28   -0.016040
2020-01-29    0.868748
2020-01-30    0.810674
2020-01-31    1.504935
Freq: D, dtype: float64
```

In [48]:

```
longer_times[datetime(2020, 1, 17):datetime(2020, 2, 17)]
```

Out[48]:

```
2020-01-17   -0.849014
2020-01-18    0.093746
2020-01-19   -1.609437
2020-01-20    1.552665
2020-01-21    0.875436
2020-01-22   -0.003206
2020-01-23   -0.952588
2020-01-24   -1.221554
2020-01-25    1.664579
2020-01-26    0.019152
2020-01-27    0.399794
2020-01-28   -0.016040
2020-01-29    0.868748
2020-01-30    0.810674
2020-01-31    1.504935
2020-02-01   -0.998766
2020-02-02    0.351994
2020-02-03    1.328684
2020-02-04   -0.682661
2020-02-05    0.603614
2020-02-06    0.382288
2020-02-07   -0.585124
2020-02-08   -1.398687
2020-02-09    1.227957
2020-02-10   -1.734838
2020-02-11    0.898652
2020-02-12    0.716881
2020-02-13    0.515387
2020-02-14   -0.455913
2020-02-15   -0.779052
2020-02-16    0.049465
2020-02-17    0.135756
Freq: D, dtype: float64
```

In [49]:

```
longer_times.truncate(after='17/2/2020')
```

Out[49]:

```
2020-01-17   -0.849014
2020-01-18    0.093746
2020-01-19   -1.609437
2020-01-20    1.552665
2020-01-21    0.875436
2020-01-22   -0.003206
2020-01-23   -0.952588
2020-01-24   -1.221554
2020-01-25    1.664579
2020-01-26    0.019152
2020-01-27    0.399794
2020-01-28   -0.016040
2020-01-29    0.868748
2020-01-30    0.810674
2020-01-31    1.504935
2020-02-01   -0.998766
2020-02-02    0.351994
2020-02-03    1.328684
2020-02-04   -0.682661
2020-02-05    0.603614
2020-02-06    0.382288
2020-02-07   -0.585124
2020-02-08   -1.398687
2020-02-09    1.227957
2020-02-10   -1.734838
2020-02-11    0.898652
2020-02-12    0.716881
2020-02-13    0.515387
2020-02-14   -0.455913
2020-02-15   -0.779052
2020-02-16    0.049465
2020-02-17    0.135756
Freq: D, dtype: float64
```

How to work with Time Series with Duplicate Indices?

In [50]:

```
dates = pd.DatetimeIndex(['1/1/2020', '1/2/2020', '1/2/2020',
                           '1/2/2020', '1/3/2020'])

dup_ts = pd.Series(np.arange(5), index=dates)
```

In [51]:

```
dup_ts
```

Out[51]:

```
2020-01-01    0
2020-01-02    1
2020-01-02    2
2020-01-02    3
2020-01-03    4
dtype: int32
```

In [52]:

```
dup_ts.index.is_unique
```

Out[52]:

```
False
```

In [53]:

```
dup_ts['1/3/2020']  # not duplicated
```

Out[53]:

```
4
```

In [54]:

```
dup_ts['1/2/2020']  # duplicated
```

Out[54]:

```
2020-01-02    1
2020-01-02    2
2020-01-02    3
dtype: int32
```

In [55]:

```
grouped = dup_ts.groupby(level=0)
grouped
```

Out[55]:

```
<pandas.core.groupby.generic.SeriesGroupBy object at 0x000000F534CA0188>
```

In [56]:

```python
print(dup_ts)
grouped.mean()
```

```
2020-01-01    0
2020-01-02    1
2020-01-02    2
2020-01-02    3
2020-01-03    4
dtype: int32
```

Out[56]:

```
2020-01-01    0
2020-01-02    2
2020-01-03    4
dtype: int32
```

In [57]:

```python
grouped.count()
```

Out[57]:

```
2020-01-01    1
2020-01-02    3
2020-01-03    1
dtype: int64
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```

How To work with Date Ranges, Frequencies, and Shifting?

In [59]:

```
import pandas as pandas
import numpy as np
from datetime import datetime
```

In [60]:

```
pd.date_range?
```

In [61]:

```
index = pd.date_range('2020-01-17', '2020-06-17')

index
```

Out[61]:

```
DatetimeIndex(['2020-01-17', '2020-01-18', '2020-01-19', '2020-01-20',
               '2020-01-21', '2020-01-22', '2020-01-23', '2020-01-24',
               '2020-01-25', '2020-01-26',
               ...
               '2020-06-08', '2020-06-09', '2020-06-10', '2020-06-11',
               '2020-06-12', '2020-06-13', '2020-06-14', '2020-06-15',
               '2020-06-16', '2020-06-17'],
              dtype='datetime64[ns]', length=153, freq='D')
```

```
In [62]:  pd.date_range(start='2020-01-17', periods=40)
```

```
Out[62]: DatetimeIndex(['2020-01-17', '2020-01-18', '2020-01-19', '2020-01-20',
                        '2020-01-21', '2020-01-22', '2020-01-23', '2020-01-24',
                        '2020-01-25', '2020-01-26', '2020-01-27', '2020-01-28',
                        '2020-01-29', '2020-01-30', '2020-01-31', '2020-02-01',
                        '2020-02-02', '2020-02-03', '2020-02-04', '2020-02-05',
                        '2020-02-06', '2020-02-07', '2020-02-08', '2020-02-09',
                        '2020-02-10', '2020-02-11', '2020-02-12', '2020-02-13',
                        '2020-02-14', '2020-02-15', '2020-02-16', '2020-02-17',
                        '2020-02-18', '2020-02-19', '2020-02-20', '2020-02-21',
                        '2020-02-22', '2020-02-23', '2020-02-24', '2020-02-25'],
                       dtype='datetime64[ns]', freq='D')
```

```
In [63]:  pd.date_range(end='2020-01-17', periods=40)
```

```
Out[63]: DatetimeIndex(['2019-12-09', '2019-12-10', '2019-12-11', '2019-12-12',
                        '2019-12-13', '2019-12-14', '2019-12-15', '2019-12-16',
                        '2019-12-17', '2019-12-18', '2019-12-19', '2019-12-20',
                        '2019-12-21', '2019-12-22', '2019-12-23', '2019-12-24',
                        '2019-12-25', '2019-12-26', '2019-12-27', '2019-12-28',
                        '2019-12-29', '2019-12-30', '2019-12-31', '2020-01-01',
                        '2020-01-02', '2020-01-03', '2020-01-04', '2020-01-05',
                        '2020-01-06', '2020-01-07', '2020-01-08', '2020-01-09',
                        '2020-01-10', '2020-01-11', '2020-01-12', '2020-01-13',
                        '2020-01-14', '2020-01-15', '2020-01-16', '2020-01-17'],
                       dtype='datetime64[ns]', freq='D')
```

```
In [64]:  pd.date_range('2020-01-01', '2021-01-01', freq='BM')
```

```
Out[64]: DatetimeIndex(['2020-01-31', '2020-02-28', '2020-03-31', '2020-04-30',
                        '2020-05-29', '2020-06-30', '2020-07-31', '2020-08-31',
                        '2020-09-30', '2020-10-30', '2020-11-30', '2020-12-31'],
                       dtype='datetime64[ns]', freq='BM')
```

In [65]:

```
pd.date_range('2020-05-02 12:56:31', periods=5)
```

Out[65]:

```
DatetimeIndex(['2020-05-02 12:56:31', '2020-05-03 12:56:31',
               '2020-05-04 12:56:31', '2020-05-05 12:56:31',
               '2020-05-06 12:56:31'],
              dtype='datetime64[ns]', freq='D')
```

In [66]:

```
pd.date_range('2020-05-02 12:56:31', periods=5, normalize=True)
```

Out[66]:

```
DatetimeIndex(['2020-05-02', '2020-05-03', '2020-05-04', '2020-05-05',
               '2020-05-06'],
              dtype='datetime64[ns]', freq='D')
```

Frequencies and Date Offsets

In [67]:

```
pd.date_range('2020-01-01', '2020-01-03 23:59', freq='5h')
```

Out[67]:

```
DatetimeIndex(['2020-01-01 00:00:00', '2020-01-01 05:00:00',
               '2020-01-01 10:00:00', '2020-01-01 15:00:00',
               '2020-01-01 20:00:00', '2020-01-02 01:00:00',
               '2020-01-02 06:00:00', '2020-01-02 11:00:00',
               '2020-01-02 16:00:00', '2020-01-02 21:00:00',
               '2020-01-03 02:00:00', '2020-01-03 07:00:00',
               '2020-01-03 12:00:00', '2020-01-03 17:00:00',
               '2020-01-03 22:00:00'],
              dtype='datetime64[ns]', freq='5H')
```

In [68]:

```
pd.date_range('2020-01-01', periods=10, freq='1h30min')
```

Out[68]:

```
DatetimeIndex(['2020-01-01 00:00:00', '2020-01-01 01:30:00',
               '2020-01-01 03:00:00', '2020-01-01 04:30:00',
               '2020-01-01 06:00:00', '2020-01-01 07:30:00',
               '2020-01-01 09:00:00', '2020-01-01 10:30:00',
               '2020-01-01 12:00:00', '2020-01-01 13:30:00'],
              dtype='datetime64[ns]', freq='90T')
```

How to Shift the Data through timestamp

In [71]:

```
pd.Series.shift?
```

In [72]:

```
ts = pd.Series(np.random.randn(5),
                 index=pd.date_range('1/1/2020', periods=5, freq='M'))

ts
```

Out[72]:

```
2020-01-31    1.057310
2020-02-29    0.180184
2020-03-31   -1.143177
2020-04-30   -0.557656
2020-05-31    0.959431
Freq: M, dtype: float64
```

In [73]:

```
ts.shift(3)
```

Out[73]:

```
2020-01-31         NaN
2020-02-29         NaN
2020-03-31         NaN
2020-04-30    1.057310
2020-05-31    0.180184
Freq: M, dtype: float64
```

In [74]:

```
ts.shift(-3)
```

Out[74]:

```
2020-01-31   -0.557656
2020-02-29    0.959431
2020-03-31         NaN
2020-04-30         NaN
2020-05-31         NaN
Freq: M, dtype: float64
```

In [75]:

```
print(ts)
ts.shift(2, freq='M')
```

```
2020-01-31    1.057310
2020-02-29    0.180184
2020-03-31   -1.143177
2020-04-30   -0.557656
2020-05-31    0.959431
Freq: M, dtype: float64
```

Out[75]:

```
2020-03-31    1.057310
2020-04-30    0.180184
2020-05-31   -1.143177
2020-06-30   -0.557656
2020-07-31    0.959431
Freq: M, dtype: float64
```

In [76]:

```
print(ts)
ts.shift(3, freq='D')
```

```
2020-01-31    1.057310
2020-02-29    0.180184
2020-03-31   -1.143177
2020-04-30   -0.557656
2020-05-31    0.959431
Freq: M, dtype: float64
```

Out[76]:

```
2020-02-03    1.057310
2020-03-03    0.180184
2020-04-03   -1.143177
2020-05-03   -0.557656
2020-06-03    0.959431
dtype: float64
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Time Zone Handling

In [77]:

```python
import pandas as pandas
import numpy as np
from datetime import datetime
import pytz
```

In [78]:

```python
pytz.common_timezones[-10:]
```

Out[78]:

```
['Pacific/Wake',
 'Pacific/Wallis',
 'US/Alaska',
 'US/Arizona',
 'US/Central',
 'US/Eastern',
 'US/Hawaii',
 'US/Mountain',
 'US/Pacific',
 'UTC']
```

In [79]:

```
tz = pytz.timezone('America/New_York')
tz
```

Out[79]:

```
<DstTzInfo 'America/New_York' LMT-1 day, 19:04:00 STD>
```

Time Zone Localization and Conversion

In [80]:

```
rng = pd.date_range('3/9/2020 9:30', periods=6, freq='D')

ts = pd.Series(np.random.randn(len(rng)), index=rng)

ts
```

Out[80]:

```
2020-03-09 09:30:00   -1.275343
2020-03-10 09:30:00   -2.120185
2020-03-11 09:30:00   -0.506625
2020-03-12 09:30:00   -0.943302
2020-03-13 09:30:00    0.605581
2020-03-14 09:30:00    0.164243
Freq: D, dtype: float64
```

In [81]:

```
print(ts.index.tz)
```

```
None
```

In [82]:

```
pd.date_range('3/9/2020 9:30', periods=10, freq='D', tz='UTC')
```

Out[82]:

```
DatetimeIndex(['2020-03-09 09:30:00+00:00', '2020-03-10 09:30:00+00:00',
               '2020-03-11 09:30:00+00:00', '2020-03-12 09:30:00+00:00',
               '2020-03-13 09:30:00+00:00', '2020-03-14 09:30:00+00:00',
               '2020-03-15 09:30:00+00:00', '2020-03-16 09:30:00+00:00',
               '2020-03-17 09:30:00+00:00', '2020-03-18 09:30:00+00:00'],
              dtype='datetime64[ns, UTC]', freq='D')
```

In [83]:

```
print(ts)
ts_utc = ts.tz_localize('UTC')
ts_utc
```

```
2020-03-09 09:30:00   -1.275343
2020-03-10 09:30:00   -2.120185
2020-03-11 09:30:00   -0.506625
2020-03-12 09:30:00   -0.943302
2020-03-13 09:30:00    0.605581
2020-03-14 09:30:00    0.164243
Freq: D, dtype: float64
```

Out[83]:

```
2020-03-09 09:30:00+00:00   -1.275343
2020-03-10 09:30:00+00:00   -2.120185
2020-03-11 09:30:00+00:00   -0.506625
2020-03-12 09:30:00+00:00   -0.943302
2020-03-13 09:30:00+00:00    0.605581
2020-03-14 09:30:00+00:00    0.164243
Freq: D, dtype: float64
```

In [84]:

```
ts_utc.index
```

Out[84]:

```
DatetimeIndex(['2020-03-09 09:30:00+00:00', '2020-03-10 09:30:00+00:00',
               '2020-03-11 09:30:00+00:00', '2020-03-12 09:30:00+00:00',
               '2020-03-13 09:30:00+00:00', '2020-03-14 09:30:00+00:00'],
              dtype='datetime64[ns, UTC]', freq='D')
```

In [85]:

```
ts_utc.tz_convert('America/New_York')
```

Out[85]:

```
2020-03-09 05:30:00-04:00   -1.275343
2020-03-10 05:30:00-04:00   -2.120185
2020-03-11 05:30:00-04:00   -0.506625
2020-03-12 05:30:00-04:00   -0.943302
2020-03-13 05:30:00-04:00    0.605581
2020-03-14 05:30:00-04:00    0.164243
Freq: D, dtype: float64
```

Operations Between Different Time Zones

In [86]:

```
rng = pd.date_range('3/7/2020 9:30', periods=10, freq='B')

ts = pd.Series(np.random.randn(len(rng)), index=rng)

ts
```

Out[86]:

```
2020-03-09 09:30:00   -1.931111
2020-03-10 09:30:00   -1.394172
2020-03-11 09:30:00   -0.063254
2020-03-12 09:30:00   -0.067925
2020-03-13 09:30:00    0.619939
2020-03-16 09:30:00    0.629845
2020-03-17 09:30:00   -0.185638
2020-03-18 09:30:00    0.882185
2020-03-19 09:30:00   -0.873995
2020-03-20 09:30:00   -1.269424
Freq: B, dtype: float64
```

In [87]:

```
ts1 = ts[:7].tz_localize('Europe/London')

ts2 = ts1[2:].tz_convert('Europe/Moscow')
```

In [88]:

```
result = ts1 + ts2
result.index
```

Out[88]:

```
DatetimeIndex(['2020-03-09 09:30:00+00:00', '2020-03-10 09:30:00+00:00',
               '2020-03-11 09:30:00+00:00', '2020-03-12 09:30:00+00:00',
               '2020-03-13 09:30:00+00:00', '2020-03-16 09:30:00+00:00',
               '2020-03-17 09:30:00+00:00'],
              dtype='datetime64[ns, UTC]', freq='B')
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
 
```

In [ ]:

```
 
```

In [ ]:

```
 
```

In [ ]:

```
 
```

In [ ]:

```
 
```

In [ ]:

```
 
```

Periods and Period Arithmetic

In [90]:

```
import pandas as pandas
import numpy as np
from datetime import datetime
```

In [91]:

```
pd.Period?
```

In [92]:

```
p = pd.Period(2020, freq='A-DEC')

p
```

Out[92]:

```
Period('2020', 'A-DEC')
```

In [93]:

```
p + 5
```

Out[93]:

```
Period('2025', 'A-DEC')
```

In [94]:

```
p - 2
```

Out[94]:

```
Period('2018', 'A-DEC')
```

In [95]:

```
pd.Period('2025', freq='A-DEC') - pd.Period('2020', freq='A-DEC')
```

Out[95]:

```
<5 * YearEnds: month=12>
```

In [96]:

```
rng = pd.period_range('2020-01-01', '2020-08-30', freq='M')

rng
```

Out[96]:

```
PeriodIndex(['2020-01', '2020-02', '2020-03', '2020-04', '2020-05', '2020-06',
             '2020-07', '2020-08'],
            dtype='period[M]', freq='M')
```

In [97]:

```
pd.Series(np.random.randn(8), index=rng)
```

Out[97]:

```
2020-01   -0.117408
2020-02   -0.946685
2020-03    0.999383
2020-04   -0.108584
2020-05    0.968578
2020-06   -0.484880
2020-07    0.935813
2020-08    0.657535
Freq: M, dtype: float64
```

Period Frequency Conversion

In [98]:

```
print(p)
p.asfreq('M', how='start')
```

```
2020
```

Out[98]:

```
Period('2020-01', 'M')
```

In [99]:

```
print(p)
p.asfreq('M', how='end')
```

```
2020
```

Out[99]:

```
Period('2020-12', 'M')
```

In [100]:

```
rng = pd.period_range('2020-01-01', '2020-08-30', freq='M')

rng
time_s = pd.Series(np.random.randn(8), index=rng)
```

In [101]:

```
time_s.asfreq('M', how='start')
```

Out[101]:

```
2020-01    0.089547
2020-02    1.551670
2020-03   -0.815674
2020-04    1.080670
2020-05   -1.227165
2020-06   -2.776933
2020-07    0.193944
2020-08   -0.580491
Freq: M, dtype: float64
```

In [102]:

```
time_s.asfreq('B', how='end')
```

Out[102]:

```
2020-01-31    0.089547
2020-02-28    1.551670
2020-03-31   -0.815674
2020-04-30    1.080670
2020-05-29   -1.227165
2020-06-30   -2.776933
2020-07-31    0.193944
2020-08-31   -0.580491
Freq: B, dtype: float64
```

Quarterly Period Frequencies

In [103]:

```
p = pd.Period('2020Q4', freq='Q-JAN')

p
```

Out[103]:

```
Period('2020Q4', 'Q-JAN')
```

In [104]:

```
p.asfreq('D', 'start')
```

Out[104]:

```
Period('2019-11-01', 'D')
```

In [105]:

```
p.asfreq('D', 'end')
```

Out[105]:

```
Period('2020-01-31', 'D')
```

In [106]:

```
rng = pd.period_range('2019Q3', '2020Q4', freq='Q-JAN')

ts = pd.Series(np.arange(len(rng)), index=rng)

ts
```

Out[106]:

```
2019Q3    0
2019Q4    1
2020Q1    2
2020Q2    3
2020Q3    4
2020Q4    5
Freq: Q-JAN, dtype: int32
```

Converting Timestamps to Periods (and Back)

In [107]:

```
rng = pd.date_range('2020-01-01', periods=3, freq='M')

ts = pd.Series(np.random.randn(3), index=rng)

ts
```

Out[107]:

```
2020-01-31   -0.216679
2020-02-29   -0.961288
2020-03-31   -1.856727
Freq: M, dtype: float64
```

In [108]:

```
pts = ts.to_period()
pts
```

Out[108]:

```
2020-01   -0.216679
2020-02   -0.961288
2020-03   -1.856727
Freq: M, dtype: float64
```

In [109]:

```
pts.to_timestamp(how='end')
```

Out[109]:

```
2020-01-31 23:59:59.999999999   -0.216679
2020-02-29 23:59:59.999999999   -0.961288
2020-03-31 23:59:59.999999999   -1.856727
Freq: M, dtype: float64
```

In [ ]:

```

```