02_pandas_time_indexes

August 13, 2021

1 more on Pandas index

1.1 Credit:

this comes from Abernathys open book, which we will be looking at a lot! https://earth-env-data-science.github.io/lectures/core_python/python_fundamentals.html

1.2 Time Indexes

Indexes are very powerful. They label the data inside a pandas series or dataframe and let you intuitivly work with the data. They are a big part of why Pandas is so useful. There are different indices for different types of data. Time Indexes are especially great!

[1]:

```
# import pandas, etc
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
```

2 dates

Python has a special datatype specific to dates. Dates get treated differently than nummbers or strings, and this feature lets you do lots of powerful timeseries analysis.

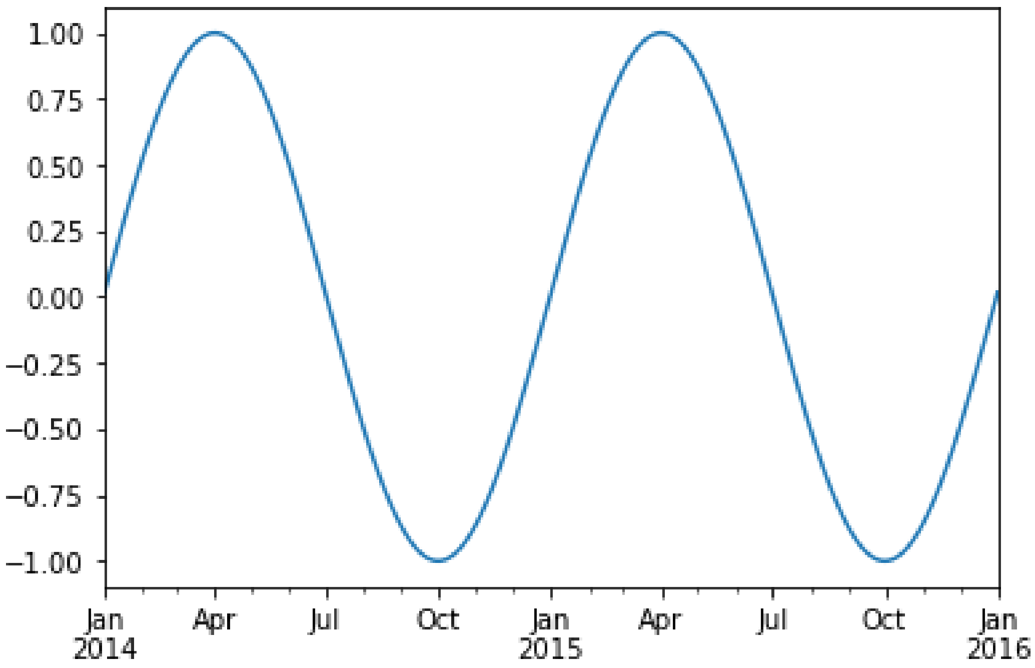
Below we make a special time series using `pd.date_range()` where we can speficy the start, end and frequency of points we want

[2]:

```
two_years = pd.date_range(start='2014-01-01', end='2016-01-01', freq='D')
timeseries = pd.Series(np.sin(2 *np.pi *two_years.dayofyear / 365),
                       index=two_years)
timeseries.plot()
```

[2]:

```
<AxesSubplot:>
```



What happened there? We made a range of times (data of a particular type, the pandas `datetime` type) using pandas `pd.date_range()`, then we used numpy (`np.sin()`) to make some fake data based on that time range.

I added the fake data, and the time range into a pandas `Series` called `timeseries`, which is like one column of a `DataFrame` using the function that creates series: `pd.Series()` , telling the function to used `two_years` as our index for our variable `timeseries`.

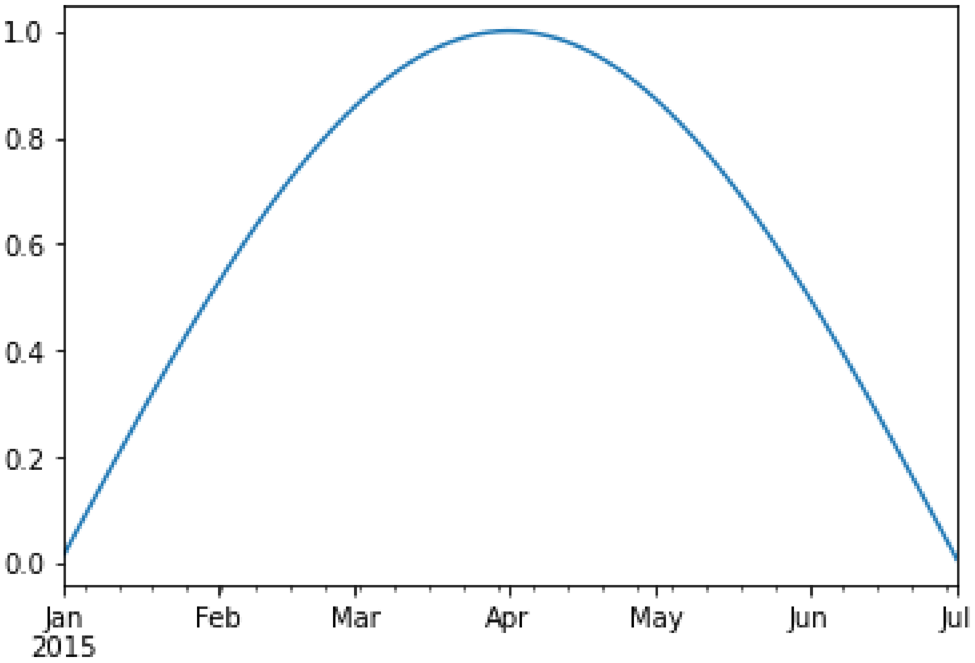
Because we used the special way pandas can create time indicies, when we plotted the `Series` using `timeseries.plot()`, matplotlib knows we are talking about time and labels everything nicely.

3 indexing and slicing

Let's say we want to just get some time subset of our data. Pandas has an easy way to let us do that using the `.loc` notation we've seen before:

```
[3]: timeseries.loc['2015-01-01':'2015-07-01'].plot()
```

```
[3]: <AxesSubplot:>
```



The TimeIndex object has lots of useful attributes

```
[4]: timeseries.index.month
```

```
[4]: Int64Index([ 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
                 …
                 12, 12, 12, 12, 12, 12, 12, 12, 12,  1],
                dtype='int64', length=731)
```

```
[5]: timeseries.index.day
```

```
[5]: Int64Index([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10,
                 …
                 23, 24, 25, 26, 27, 28, 29, 30, 31,  1],
                dtype='int64', length=731)
```

3.1 Reading real Data Files: NOAA Weather Station Data

In this example, we will use NOAA weather station data from https://www.ncdc.noaa.gov/data-access/land-based-station-data.

The details of files we are going to read are described in this README file.

We have a text file on our hard drive called `data.txt`. **Examine it by opening in jupyterlab.** What do you see?

To read it into pandas, we will use the read_csv function. This function is incredibly complex and powerful. You can use it to extract data from almost any text file. However, you need to understand how to use its various options. Its is awesome, it will let you read data out of almost any text file. However to get the data in a useful format we are going to have to do a few things to it. In the next few steps we will see how to clean and wrangle the data into a good format for use.

3.1.1 Cleaning and wrangling messy data into a format you can use is an incredibly important skill to master

If we just read the data in with no options, we get something that is a bit of a mess:

With no options, this is what we get.

[6]:
```
df = pd.read_csv('data.txt')

print(df.shape)
df.head()
```

```
(365, 1)
```

[6]:
```
  WBANNO LST_DATE CRX_VN LONGITUDE LATITUDE T_DAILY_MAX T_DAILY_MIN T_DAILY_MEAN
T_DAILY_AVG P_DAILY_CALC SOLARAD_DAILY SUR_TEMP_DAILY_TYPE SUR_TEMP_DAILY_MAX
SUR_TEMP_DAILY_MIN SUR_TEMP_DAILY_AVG RH_DAILY_MAX RH_DAILY_MIN RH_DAILY_AVG
SOIL_MOISTURE_5_DAILY SOIL_MOISTURE_10_DAILY SOIL_MOISTURE_20_DAILY
SOIL_MOISTURE_50_DAILY SOIL_MOISTURE_100_DAILY SOIL_TEMP_5_DAILY
SOIL_TEMP_10_DAILY SOIL_TEMP_20_DAILY SOIL_TEMP_50_DAILY SOIL_TEMP_100_DAILY
0  64756 20170101  2.422  -73.74   41.79     6.6 …
1  64756 20170102  2.422  -73.74   41.79     4.0 …
2  64756 20170103  2.422  -73.74   41.79     4.9 …
3  64756 20170104  2.422  -73.74   41.79     8.7 …
4  64756 20170105  2.422  -73.74   41.79    -0.5 …
```

Pandas wasn't able to automatically sort out all the columns because, as the name suggests, `pd.read_csv()` expects the data to be in comma separated value (csv) format. Our data is just separated by spaces.

Fortunatly we can put options into `pd.read_csv()` to tell it what the separation between data is by using the `sep=` keyword. This lets the function read the data correctly. The representation of space is `'\s+'`.

[7]:
```
df = pd.read_csv('data.txt', sep='\s+')
print(df.shape)
df.head()
```

```
(365, 28)
```

[7]:
```
   WBANNO  LST_DATE  CRX_VN  LONGITUDE  LATITUDE  T_DAILY_MAX  T_DAILY_MIN  \
0   64756  20170101   2.422     -73.74     41.79          6.6         -5.4
1   64756  20170102   2.422     -73.74     41.79          4.0         -6.8
2   64756  20170103   2.422     -73.74     41.79          4.9          0.7
3   64756  20170104   2.422     -73.74     41.79          8.7         -1.6
4   64756  20170105   2.422     -73.74     41.79         -0.5         -4.6
```

```
   T_DAILY_MEAN  T_DAILY_AVG  P_DAILY_CALC  …  SOIL_MOISTURE_5_DAILY  \
0           0.6          2.2           0.0  …                  -99.0
1          -1.4         -1.2           0.0  …                  -99.0
2           2.8          2.7          13.1  …                  -99.0
3           3.6          3.5           1.3  …                  -99.0
4          -2.5         -2.8           0.0  …                  -99.0

  SOIL_MOISTURE_10_DAILY  SOIL_MOISTURE_20_DAILY  SOIL_MOISTURE_50_DAILY  \
0                  -99.0                   0.207                   0.152
1                  -99.0                   0.205                   0.151
2                  -99.0                   0.205                   0.150
3                  -99.0                   0.215                   0.153
4                  -99.0                   0.215                   0.154

   SOIL_MOISTURE_100_DAILY  SOIL_TEMP_5_DAILY  SOIL_TEMP_10_DAILY  \
0                    0.175               -0.1                 0.0
1                    0.173               -0.2                 0.0
2                    0.173               -0.1                 0.0
3                    0.174               -0.1                 0.0
4                    0.177               -0.1                 0.0

   SOIL_TEMP_20_DAILY  SOIL_TEMP_50_DAILY  SOIL_TEMP_100_DAILY
0                 0.6                 1.5                  3.4
1                 0.6                 1.5                  3.3
2                 0.5                 1.5                  3.3
3                 0.5                 1.5                  3.2
4                 0.5                 1.4                  3.1

[5 rows x 28 columns]
```

excellent, much better. now the columns are all separated out well.

if we looked at all the data we will see there are lots of -99 and -9999 values in the file. If we look closely, we will see there are lots of -99 and -9999 values in the file. The README file tells us that these are values used to represent missing data. Let's tell this to pandas. We do this using an arguement specification `na_values =[listof bad data values]`. the `na` part stands for NaN (not a number) which will be filled in:

[8]:
```
df = pd.read_csv('data.txt', sep='\s+', na_values=[-9999.0, -99.0])
df.head()
```

[8]:
```
   WBANNO  LST_DATE  CRX_VN  LONGITUDE  LATITUDE  T_DAILY_MAX  T_DAILY_MIN  \
0   64756  20170101   2.422     -73.74     41.79          6.6         -5.4
1   64756  20170102   2.422     -73.74     41.79          4.0         -6.8
2   64756  20170103   2.422     -73.74     41.79          4.9          0.7
3   64756  20170104   2.422     -73.74     41.79          8.7         -1.6
4   64756  20170105   2.422     -73.74     41.79         -0.5         -4.6
```

```
   T_DAILY_MEAN  T_DAILY_AVG  P_DAILY_CALC  …  SOIL_MOISTURE_5_DAILY  \
0           0.6          2.2           0.0  …                    NaN
1          -1.4         -1.2           0.0  …                    NaN
2           2.8          2.7          13.1  …                    NaN
3           3.6          3.5           1.3  …                    NaN
4          -2.5         -2.8           0.0  …                    NaN

  SOIL_MOISTURE_10_DAILY  SOIL_MOISTURE_20_DAILY  SOIL_MOISTURE_50_DAILY  \
0                    NaN                   0.207                   0.152
1                    NaN                   0.205                   0.151
2                    NaN                   0.205                   0.150
3                    NaN                   0.215                   0.153
4                    NaN                   0.215                   0.154

   SOIL_MOISTURE_100_DAILY  SOIL_TEMP_5_DAILY  SOIL_TEMP_10_DAILY  \
0                    0.175               -0.1                 0.0
1                    0.173               -0.2                 0.0
2                    0.173               -0.1                 0.0
3                    0.174               -0.1                 0.0
4                    0.177               -0.1                 0.0

   SOIL_TEMP_20_DAILY  SOIL_TEMP_50_DAILY  SOIL_TEMP_100_DAILY
0                 0.6                 1.5                  3.4
1                 0.6                 1.5                  3.3
2                 0.5                 1.5                  3.3
3                 0.5                 1.5                  3.2
4                 0.5                 1.4                  3.1

[5 rows x 28 columns]
```

ok, another good step. Now all the bad data is represented by NaN, which is something that pandas and numpy are good at dealing with.

you can see we are slowing bulding up a good dataframe. We are cleaning out all the warts in the data. This is something you will do over and over!

Let's check out what is in our Dataframe:

[9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 365 entries, 0 to 364
Data columns (total 28 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   WBANNO                   365 non-null    int64
 1   LST_DATE                 365 non-null    int64
 2   CRX_VN                   365 non-null    float64
 3   LONGITUDE                365 non-null    float64
 4   LATITUDE                 365 non-null    float64
 5   T_DAILY_MAX              364 non-null    float64
 6   T_DAILY_MIN              364 non-null    float64
 7   T_DAILY_MEAN             364 non-null    float64
 8   T_DAILY_AVG              364 non-null    float64
 9   P_DAILY_CALC             364 non-null    float64
 10  SOLARAD_DAILY            364 non-null    float64
 11  SUR_TEMP_DAILY_TYPE      365 non-null    object
 12  SUR_TEMP_DAILY_MAX       364 non-null    float64
 13  SUR_TEMP_DAILY_MIN       364 non-null    float64
 14  SUR_TEMP_DAILY_AVG       364 non-null    float64
 15  RH_DAILY_MAX             364 non-null    float64
 16  RH_DAILY_MIN             364 non-null    float64
 17  RH_DAILY_AVG             364 non-null    float64
 18  SOIL_MOISTURE_5_DAILY    317 non-null    float64
 19  SOIL_MOISTURE_10_DAILY   317 non-null    float64
 20  SOIL_MOISTURE_20_DAILY   336 non-null    float64
 21  SOIL_MOISTURE_50_DAILY   364 non-null    float64
 22  SOIL_MOISTURE_100_DAILY  359 non-null    float64
 23  SOIL_TEMP_5_DAILY        364 non-null    float64
 24  SOIL_TEMP_10_DAILY       364 non-null    float64
 25  SOIL_TEMP_20_DAILY       364 non-null    float64
 26  SOIL_TEMP_50_DAILY       364 non-null    float64
 27  SOIL_TEMP_100_DAILY      364 non-null    float64
dtypes: float64(25), int64(2), object(1)
memory usage: 80.0+ KB
```

One problem here is that pandas did not recognize the LDT_DATE column as a date. Let's help it.

[10]:

```python
df = pd.read_csv('data.txt', sep='\s+',
                 na_values=[-9999.0, -99.0],
                 parse_dates=[1])
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 365 entries, 0 to 364
Data columns (total 28 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   WBANNO                   365 non-null    int64
 1   LST_DATE                 365 non-null    datetime64[ns]
 2   CRX_VN                   365 non-null    float64
 3   LONGITUDE                365 non-null    float64
 4   LATITUDE                 365 non-null    float64
 5   T_DAILY_MAX              364 non-null    float64
 6   T_DAILY_MIN              364 non-null    float64
 7   T_DAILY_MEAN             364 non-null    float64
 8   T_DAILY_AVG              364 non-null    float64
 9   P_DAILY_CALC            364 non-null    float64
 10  SOLARAD_DAILY           364 non-null    float64
 11  SUR_TEMP_DAILY_TYPE     365 non-null    object
 12  SUR_TEMP_DAILY_MAX      364 non-null    float64
 13  SUR_TEMP_DAILY_MIN      364 non-null    float64
 14  SUR_TEMP_DAILY_AVG      364 non-null    float64
 15  RH_DAILY_MAX            364 non-null    float64
 16  RH_DAILY_MIN            364 non-null    float64
 17  RH_DAILY_AVG            364 non-null    float64
 18  SOIL_MOISTURE_5_DAILY   317 non-null    float64
 19  SOIL_MOISTURE_10_DAILY  317 non-null    float64
 20  SOIL_MOISTURE_20_DAILY  336 non-null    float64
 21  SOIL_MOISTURE_50_DAILY  364 non-null    float64
 22  SOIL_MOISTURE_100_DAILY 359 non-null    float64
 23  SOIL_TEMP_5_DAILY       364 non-null    float64
 24  SOIL_TEMP_10_DAILY      364 non-null    float64
 25  SOIL_TEMP_20_DAILY      364 non-null    float64
 26  SOIL_TEMP_50_DAILY      364 non-null    float64
 27  SOIL_TEMP_100_DAILY     364 non-null    float64
dtypes: datetime64[ns](1), float64(25), int64(1), object(1)
memory usage: 80.0+ KB
```

now we see that the `LST_DATE` column is the special datetime64 type of data that lets pandas do all the cool stuff with time.

We are one step closer to a good data set!

the last step: we want to use the date as the index for our dataframe. It's the timestamp that ties all this data together. We can tell pandas to do this too by setting the index of our dataframe to that column

[11]:

```
df = df.set_index('LST_DATE')
df.head()
```

[11]:

```
            WBANNO  CRX_VN  LONGITUDE  LATITUDE  T_DAILY_MAX  T_DAILY_MIN  \
LST_DATE
2017-01-01   64756   2.422     -73.74     41.79          6.6         -5.4
2017-01-02   64756   2.422     -73.74     41.79          4.0         -6.8
2017-01-03   64756   2.422     -73.74     41.79          4.9          0.7
2017-01-04   64756   2.422     -73.74     41.79          8.7         -1.6
2017-01-05   64756   2.422     -73.74     41.79         -0.5         -4.6

            T_DAILY_MEAN  T_DAILY_AVG  P_DAILY_CALC  SOLARAD_DAILY  …  \
LST_DATE                                                             …
2017-01-01           0.6          2.2           0.0           8.68  …
2017-01-02          -1.4         -1.2           0.0           2.08  …
2017-01-03           2.8          2.7          13.1           0.68  …
2017-01-04           3.6          3.5           1.3           2.85  …
2017-01-05          -2.5         -2.8           0.0           4.90  …
```

```
           SOIL_MOISTURE_5_DAILY  SOIL_MOISTURE_10_DAILY  \
LST_DATE
2017-01-01                   NaN                     NaN
2017-01-02                   NaN                     NaN
2017-01-03                   NaN                     NaN
2017-01-04                   NaN                     NaN
2017-01-05                   NaN                     NaN

            SOIL_MOISTURE_20_DAILY  SOIL_MOISTURE_50_DAILY  \
LST_DATE
2017-01-01                   0.207                   0.152
2017-01-02                   0.205                   0.151
2017-01-03                   0.205                   0.150
2017-01-04                   0.215                   0.153
2017-01-05                   0.215                   0.154

            SOIL_MOISTURE_100_DAILY  SOIL_TEMP_5_DAILY  SOIL_TEMP_10_DAILY  \
LST_DATE
2017-01-01                    0.175               -0.1                 0.0
2017-01-02                    0.173               -0.2                 0.0
2017-01-03                    0.173               -0.1                 0.0
2017-01-04                    0.174               -0.1                 0.0
2017-01-05                    0.177               -0.1                 0.0

            SOIL_TEMP_20_DAILY  SOIL_TEMP_50_DAILY  SOIL_TEMP_100_DAILY
LST_DATE
2017-01-01                 0.6                 1.5                  3.4
2017-01-02                 0.6                 1.5                  3.3
2017-01-03                 0.5                 1.5                  3.3
2017-01-04                 0.5                 1.5                  3.2
2017-01-05                 0.5                 1.4                  3.1

[5 rows x 27 columns]
```

Now we can take advantage of all the cool time stuff that pandas can do.

Like let's use indexing to look at all the data from one day

[12]:
```
df.loc['2017-08-07']
```

[12]:
```
WBANNO                   64756
CRX_VN                   2.422
LONGITUDE               -73.74
LATITUDE                 41.79
T_DAILY_MAX               19.3
T_DAILY_MIN               12.3
T_DAILY_MEAN              15.8
```

```
T_DAILY_AVG                16.3
P_DAILY_CALC                4.9
SOLARAD_DAILY              3.93
SUR_TEMP_DAILY_TYPE           C
SUR_TEMP_DAILY_MAX         22.3
SUR_TEMP_DAILY_MIN         11.9
SUR_TEMP_DAILY_AVG         17.7
RH_DAILY_MAX               94.7
RH_DAILY_MIN               76.4
RH_DAILY_AVG               89.5
SOIL_MOISTURE_5_DAILY     0.148
SOIL_MOISTURE_10_DAILY    0.113
SOIL_MOISTURE_20_DAILY    0.094
SOIL_MOISTURE_50_DAILY    0.114
SOIL_MOISTURE_100_DAILY   0.151
SOIL_TEMP_5_DAILY          21.4
SOIL_TEMP_10_DAILY         21.7
SOIL_TEMP_20_DAILY         22.1
SOIL_TEMP_50_DAILY         22.2
SOIL_TEMP_100_DAILY        21.5
Name: 2017-08-07 00:00:00, dtype: object
```

Or use slicing to get a range:

[13]:

```
df.loc['2017-07-01':'2017-07-31']
```

[13]:

```
            WBANNO  CRX_VN  LONGITUDE  LATITUDE  T_DAILY_MAX  T_DAILY_MIN  \
LST_DATE
2017-07-01   64756   2.422     -73.74     41.79         28.0         19.7
2017-07-02   64756   2.422     -73.74     41.79         29.8         18.4
2017-07-03   64756   2.422     -73.74     41.79         28.3         15.0
2017-07-04   64756   2.422     -73.74     41.79         26.8         12.6
2017-07-05   64756   2.422     -73.74     41.79         28.0         11.9
2017-07-06   64756   2.422     -73.74     41.79         25.7         14.3
2017-07-07   64756   2.422     -73.74     41.79         25.8         16.8
2017-07-08   64756   2.422     -73.74     41.79         29.0         15.3
2017-07-09   64756   2.422     -73.74     41.79         26.3         10.9
2017-07-10   64756   2.422     -73.74     41.79         27.6         11.8
2017-07-11   64756   2.422     -73.74     41.79         27.4         19.2
2017-07-12   64756   2.422     -73.74     41.79         29.4         18.5
2017-07-13   64756   2.422     -73.74     41.79         29.5         18.3
2017-07-14   64756   2.422     -73.74     41.79         18.5         15.9
2017-07-15   64756   2.422     -73.74     41.79         26.6         16.5
2017-07-16   64756   2.422     -73.74     41.79         27.9         13.3
2017-07-17   64756   2.422     -73.74     41.79         29.2         16.1
2017-07-18   64756   2.422     -73.74     41.79         30.3         19.3
2017-07-19   64756   2.422     -73.74     41.79         31.2         19.1
2017-07-20   64756   2.422     -73.74     41.79         31.8         16.6
```

```
2017-07-21   64756   2.422     -73.74     41.79         30.6         16.6
2017-07-22   64756   2.422     -73.74     41.79         27.7         15.6
2017-07-23   64756   2.422     -73.74     41.79         26.4         18.5
2017-07-24   64756   2.422     -73.74     41.79         19.4         14.8
2017-07-25   64756   2.422     -73.74     41.79         18.6         13.7
2017-07-26   64756   2.422     -73.74     41.79         24.7         11.2
2017-07-27   64756   2.422     -73.74     41.79         24.2         15.2
2017-07-28   64756   2.422     -73.74     41.79         26.5         16.9
2017-07-29   64756   2.422     -73.74     41.79         24.2         10.4
2017-07-30   64756   2.422     -73.74     41.79         25.5          8.2
2017-07-31   64756   2.422     -73.74     41.79         29.4         10.1

            T_DAILY_MEAN  T_DAILY_AVG  P_DAILY_CALC  SOLARAD_DAILY  …  \
LST_DATE                                                            …
2017-07-01          23.9         23.8           0.2          19.28  …
2017-07-02          24.1         23.7           4.0          27.67  …
2017-07-03          21.7         21.4           0.0          27.08  …
2017-07-04          19.7         20.0           0.0          29.45  …
2017-07-05          20.0         20.7           0.0          26.90  …
2017-07-06          20.0         20.3           0.0          19.03  …
2017-07-07          21.3         20.0          11.5          13.88  …
2017-07-08          22.1         21.5           0.0          21.92  …
2017-07-09          18.6         19.4           0.0          29.72  …
2017-07-10          19.7         21.3           0.0          23.67  …
2017-07-11          23.3         22.6           8.5          17.79  …
2017-07-12          23.9         23.1           1.9          16.27  …
2017-07-13          23.9         23.4          23.3          13.61  …
2017-07-14          17.2         17.5           4.1           5.36  …
2017-07-15          21.5         21.0           0.8          21.13  …
2017-07-16          20.6         21.0           0.0          27.03  …
2017-07-17          22.6         22.9           0.0          20.47  …
2017-07-18          24.8         24.7           0.0          24.99  …
2017-07-19          25.1         25.0           0.0          27.69  …
2017-07-20          24.2         23.4           0.7          21.53  …
2017-07-21          23.6         23.6           0.0          25.55  …
2017-07-22          21.7         21.2           0.5          16.04  …
2017-07-23          22.5         22.2           0.0          19.03  …
2017-07-24          17.1         16.7          29.2           9.10  …
2017-07-25          16.2         16.2           0.0           7.35  …
2017-07-26          18.0         18.3           0.0          22.22  …
2017-07-27          19.7         19.5           0.0           8.28  …
2017-07-28          21.7         20.9           0.0          21.06  …
2017-07-29          17.3         18.1           0.0          21.28  …
2017-07-30          16.8         17.3           0.0          27.68  …
2017-07-31          19.7         20.1           0.0          25.49  …

           SOIL_MOISTURE_5_DAILY  SOIL_MOISTURE_10_DAILY  \
LST_DATE
2017-07-01                 0.157                   0.136
2017-07-02                 0.146                   0.135
2017-07-03                 0.141                   0.132
2017-07-04                 0.131                   0.126
2017-07-05                 0.116                   0.114
2017-07-06                 0.105                   0.104
2017-07-07                 0.114                   0.100
2017-07-08                 0.130                   0.106
2017-07-09                 0.119                   0.103
2017-07-10                 0.105                   0.096
2017-07-11                 0.106                   0.093
2017-07-12                 0.108                   0.094
2017-07-13                 0.134                   0.110
2017-07-14                 0.194                   0.151
2017-07-15                 0.190                   0.163
2017-07-16                 0.171                   0.154
2017-07-17                 0.155                   0.143
2017-07-18                 0.142                   0.132
2017-07-19                 0.126                   0.118
2017-07-20                 0.111                   0.103
2017-07-21                 0.100                   0.093
2017-07-22                 0.092                   0.086
2017-07-23                 0.087                   0.082
2017-07-24                 0.145                   0.118
2017-07-25                 0.167                   0.133
2017-07-26                 0.155                   0.128
2017-07-27                 0.144                   0.122
2017-07-28                 0.137                   0.117
2017-07-29                 0.126                   0.108
2017-07-30                 0.113                   0.099
2017-07-31                 0.101                   0.090

            SOIL_MOISTURE_20_DAILY  SOIL_MOISTURE_50_DAILY  \
LST_DATE
2017-07-01                   0.144                   0.129
2017-07-02                   0.143                   0.129
2017-07-03                   0.139                   0.128
2017-07-04                   0.136                   0.126
2017-07-05                   0.131                   0.125
2017-07-06                   0.126                   0.124
2017-07-07                   0.123                   0.123
2017-07-08                   0.122                   0.123
2017-07-09                   0.119                   0.121
2017-07-10                   0.113                   0.120
2017-07-11                   0.110                   0.120
2017-07-12                   0.108                   0.118
```

```
2017-07-13                   0.108                   0.118
2017-07-14                   0.114                   0.120
2017-07-15                   0.119                   0.122
2017-07-16                   0.123                   0.123
2017-07-17                   0.124                   0.122
2017-07-18                   0.122                   0.122
2017-07-19                   0.118                   0.122
2017-07-20                   0.114                   0.121
2017-07-21                   0.108                   0.120
2017-07-22                   0.104                   0.119
2017-07-23                   0.100                   0.118
2017-07-24                   0.102                   0.117
2017-07-25                   0.107                   0.116
2017-07-26                   0.108                   0.118
2017-07-27                   0.109                   0.118
2017-07-28                   0.110                   0.119
2017-07-29                   0.108                   0.118
2017-07-30                   0.104                   0.117
2017-07-31                   0.099                   0.116

            SOIL_MOISTURE_100_DAILY  SOIL_TEMP_5_DAILY  SOIL_TEMP_10_DAILY  \
LST_DATE
2017-07-01                    0.163               25.7                25.4
2017-07-02                    0.162               26.8                26.4
2017-07-03                    0.162               26.4                26.3
2017-07-04                    0.161               25.9                25.8
2017-07-05                    0.161               25.3                25.3
2017-07-06                    0.160               24.7                24.7
2017-07-07                    0.160               24.2                24.2
2017-07-08                    0.159               25.5                25.3
2017-07-09                    0.158               24.8                24.8
2017-07-10                    0.158               24.7                24.7
2017-07-11                    0.157               25.6                25.4
2017-07-12                    0.157               25.8                25.6
2017-07-13                    0.156               25.7                25.7
2017-07-14                    0.155               23.0                23.3
2017-07-15                    0.155               24.6                24.4
2017-07-16                    0.155               25.4                25.3
2017-07-17                    0.156               25.7                25.6
2017-07-18                    0.156               27.0                26.7
2017-07-19                    0.156               27.6                27.4
2017-07-20                    0.156               27.0                27.0
2017-07-21                    0.155               27.1                27.0
2017-07-22                    0.156               25.9                26.1
2017-07-23                    0.155               26.0                26.0
2017-07-24                    0.154               23.1                23.6
2017-07-25                    0.153               21.9                22.2
```

```
2017-07-26                    0.152               22.9                23.0
2017-07-27                    0.154               22.5                22.7
2017-07-28                    0.154               24.1                24.1
2017-07-29                    0.154               23.3                23.6
2017-07-30                    0.154               22.8                23.0
2017-07-31                    0.153               23.8                23.8

            SOIL_TEMP_20_DAILY  SOIL_TEMP_50_DAILY  SOIL_TEMP_100_DAILY
LST_DATE
2017-07-01                23.7                21.9                 19.9
2017-07-02                24.5                22.3                 20.1
2017-07-03                24.8                22.8                 20.3
2017-07-04                24.6                22.9                 20.6
2017-07-05                24.2                22.8                 20.7
2017-07-06                23.9                22.7                 20.9
2017-07-07                23.4                22.4                 20.8
2017-07-08                23.9                22.4                 20.8
2017-07-09                23.8                22.5                 20.8
2017-07-10                23.6                22.5                 20.9
2017-07-11                24.1                22.6                 20.9
2017-07-12                24.2                22.8                 21.0
2017-07-13                24.4                23.0                 21.0
2017-07-14                23.4                22.9                 21.2
2017-07-15                23.2                22.2                 21.2
2017-07-16                23.9                22.6                 21.1
2017-07-17                24.4                22.9                 21.2
2017-07-18                24.9                23.2                 21.3
2017-07-19                25.6                23.7                 21.5
2017-07-20                25.6                24.0                 21.7
2017-07-21                25.5                24.0                 21.9
2017-07-22                25.3                24.1                 22.0
2017-07-23                24.9                23.8                 22.1
2017-07-24                23.9                23.5                 22.1
2017-07-25                22.4                22.5                 21.9
2017-07-26                22.3                22.0                 21.7
2017-07-27                22.4                22.0                 21.4
2017-07-28                22.8                22.0                 21.3
2017-07-29                23.0                22.2                 21.3
2017-07-30                22.4                22.0                 21.3
2017-07-31                22.7                21.9                 21.2

[31 rows x 27 columns]
```

3.1.2 Quick Statistics

[14]:

```
df.describe()
```

[14]:

```
        WBANNO      CRX_VN     LONGITUDE      LATITUDE  T_DAILY_MAX  \
count    365.0  365.000000  3.650000e+02  3.650000e+02   364.000000
mean   64756.0    2.470767 -7.374000e+01  4.179000e+01    15.720055
std        0.0    0.085997  5.265234e-13  3.842198e-13    10.502087
min    64756.0    2.422000 -7.374000e+01  4.179000e+01   -12.300000
25%    64756.0    2.422000 -7.374000e+01  4.179000e+01     6.900000
50%    64756.0    2.422000 -7.374000e+01  4.179000e+01    17.450000
75%    64756.0    2.422000 -7.374000e+01  4.179000e+01    24.850000
max    64756.0    2.622000 -7.374000e+01  4.179000e+01    33.400000

       T_DAILY_MIN  T_DAILY_MEAN  T_DAILY_AVG  P_DAILY_CALC  SOLARAD_DAILY  \
count   364.000000    364.000000   364.000000    364.000000     364.000000
mean      4.037912      9.876374     9.990110      2.797802      13.068187
std       9.460676      9.727451     9.619168      7.238628       7.953074
min     -21.800000    -17.000000   -16.700000      0.000000       0.100000
25%      -2.775000      2.100000     2.275000      0.000000       6.225000
50%       4.350000     10.850000    11.050000      0.000000      12.865000
75%      11.900000     18.150000    18.450000      1.400000      19.740000
max      20.700000     25.700000    26.700000     65.700000      29.910000

       …  SOIL_MOISTURE_5_DAILY  SOIL_MOISTURE_10_DAILY  \
count  …             317.000000              317.000000
mean   …               0.183804                0.181000
std    …               0.047493                0.052697
min    …               0.075000                0.074000
25%    …               0.148000                0.137000
50%    …               0.192000                0.198000
75%    …               0.221000                0.219000
max    …               0.294000                0.321000

       SOIL_MOISTURE_20_DAILY  SOIL_MOISTURE_50_DAILY  \
count              336.000000              364.000000
mean                 0.156533                0.138286
std                  0.042775                0.019207
min                  0.069000                0.100000
25%                  0.118000                0.118000
50%                  0.169000                0.147000
75%                  0.188000                0.152250
max                  0.231000                0.170000

       SOIL_MOISTURE_100_DAILY  SOIL_TEMP_5_DAILY  SOIL_TEMP_10_DAILY  \
count               359.000000         364.000000          364.000000
mean                  0.162844          12.344231           12.308516
```

```
std                   0.013814           9.367742            9.350273
min                   0.128000          -0.700000           -0.400000
25%                   0.155000           2.275000            2.075000
50%                   0.166000          13.300000           13.350000
75%                   0.173000          21.025000           21.125000
max                   0.192000          27.600000           27.400000

       SOIL_TEMP_20_DAILY  SOIL_TEMP_50_DAILY  SOIL_TEMP_100_DAILY
count          364.000000          364.000000           364.000000
mean            12.060989           11.960989            11.971978
std              8.760899            8.082595             7.170197
min              0.200000            0.900000             1.900000
25%              2.575000            3.300000             4.100000
50%             13.100000           12.850000            11.650000
75%             20.400000           19.800000            19.325000
max             25.600000           24.100000            22.100000

[8 rows x 26 columns]
```

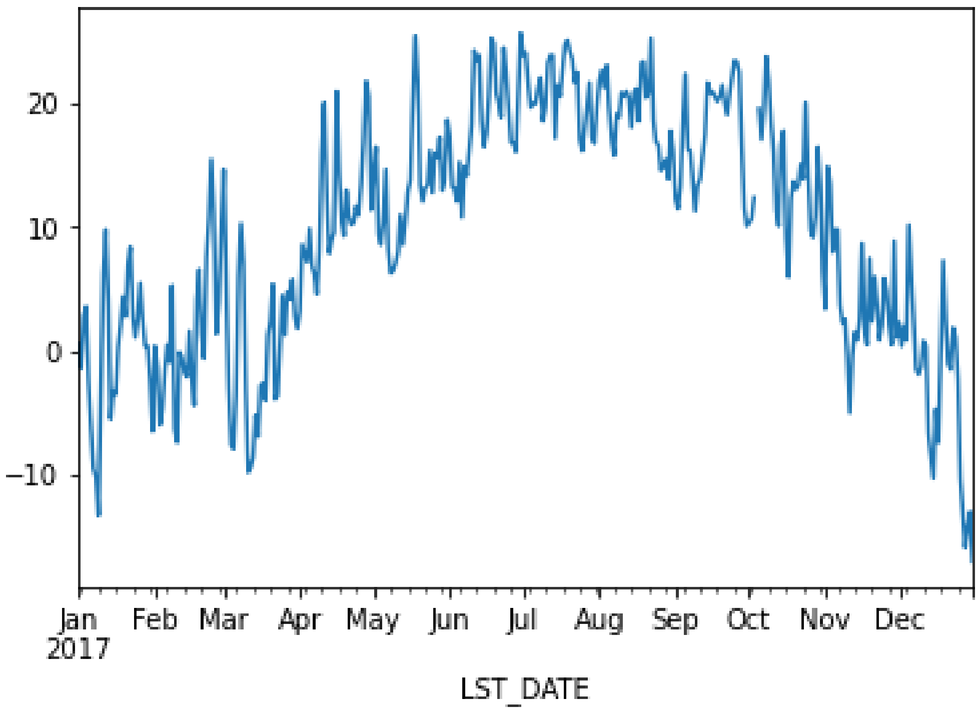
3.1.3 Plotting Values

We can now quickly make plots of the data

Pandas is very “time aware”:

```
[15]: df.T_DAILY_MEAN.plot()
```

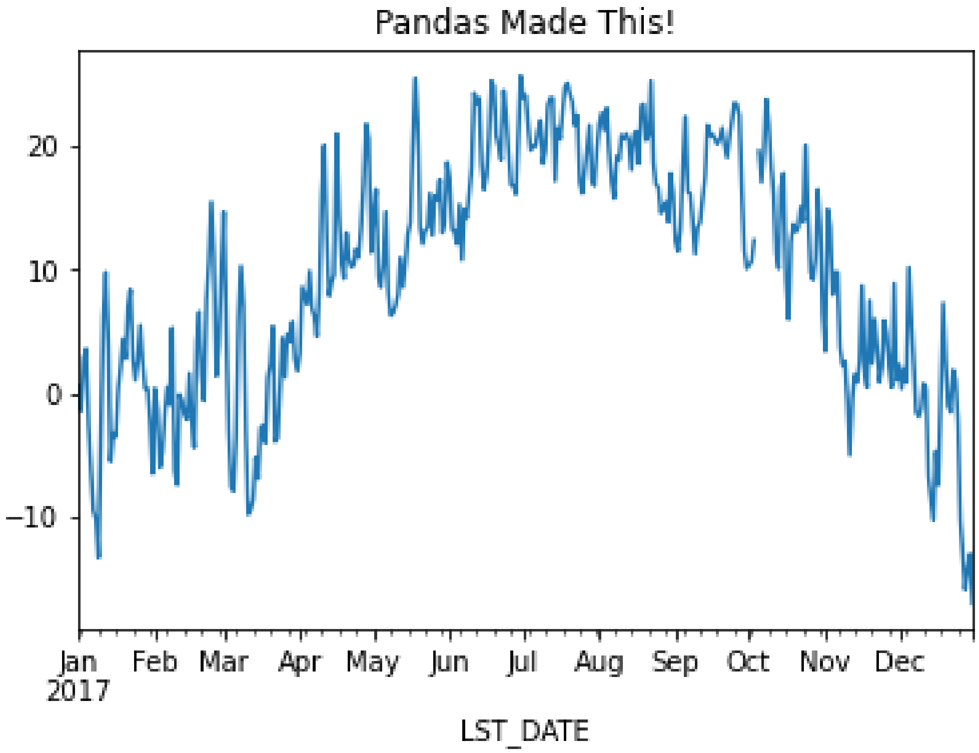
```
[15]: <AxesSubplot:xlabel='LST_DATE'>
```



Note: we could also manually create an axis and plot into it.

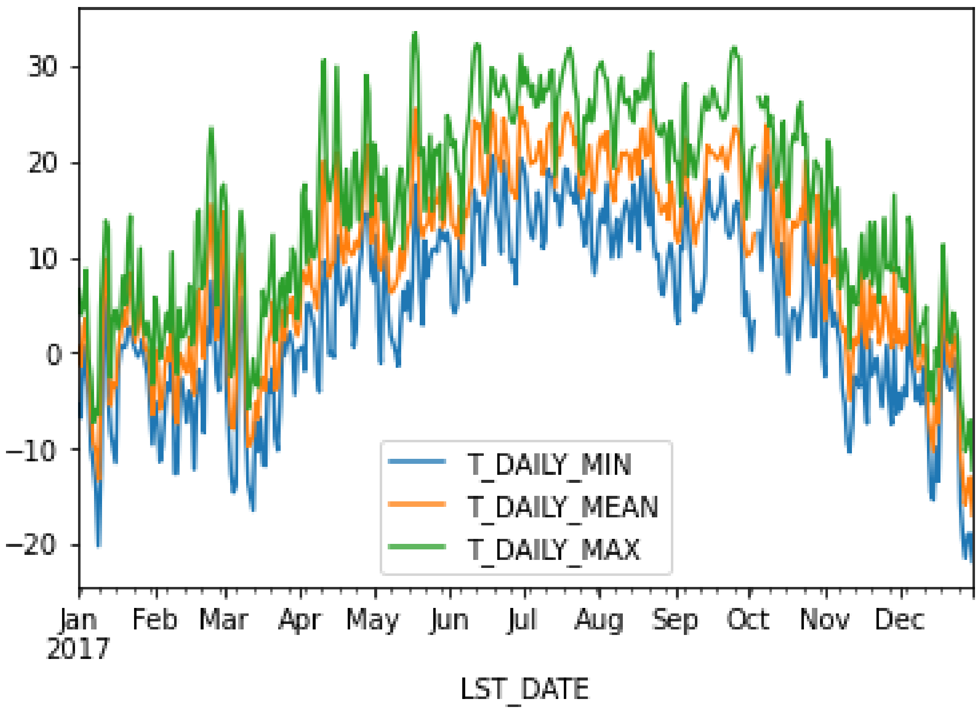
[16]:
```
fig, ax = plt.subplots()
df.T_DAILY_MEAN.plot(ax=ax)
ax.set_title('Pandas Made This!')
```

[16]:
```
Text(0.5, 1.0, 'Pandas Made This!')
```



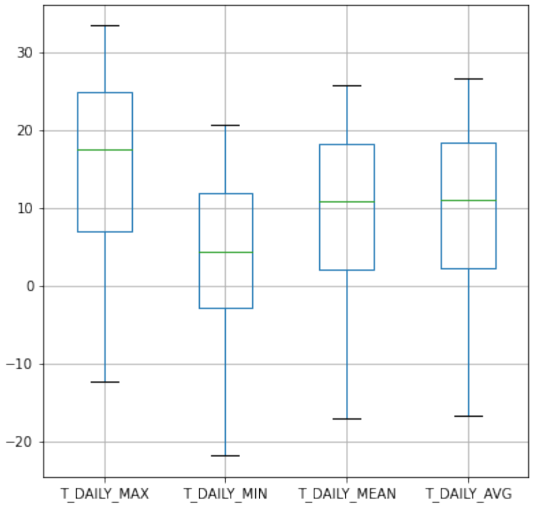
```
[17]: df[['T_DAILY_MIN', 'T_DAILY_MEAN', 'T_DAILY_MAX']].plot()
```

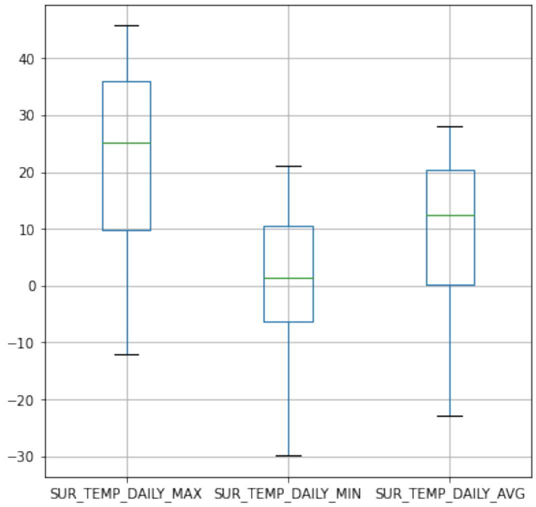
```
[17]: <AxesSubplot:xlabel='LST_DATE'>
```

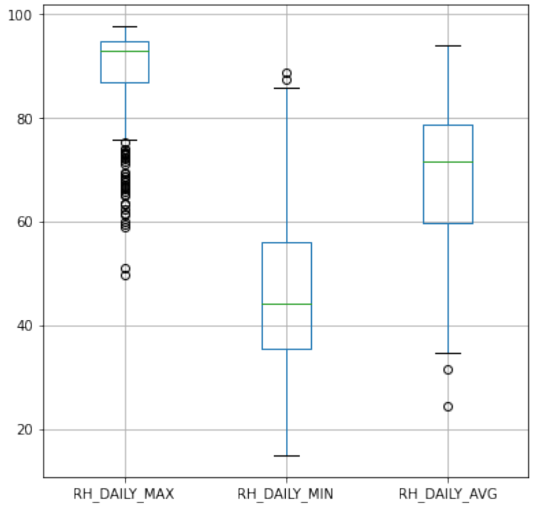


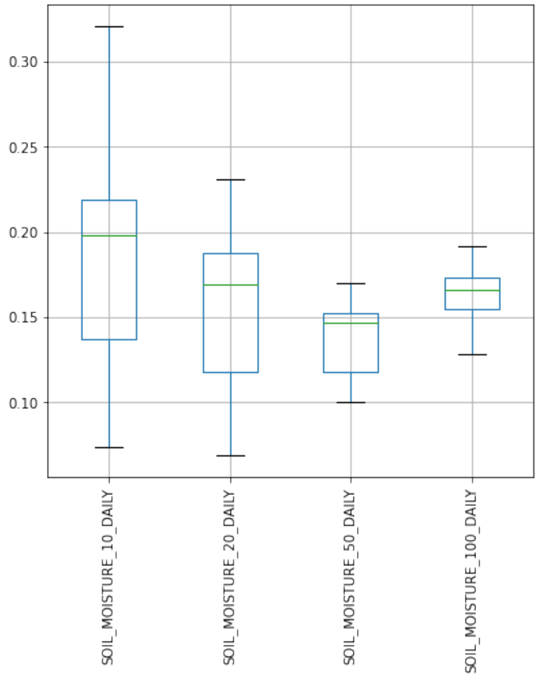
We can do some statistical plots too:

[18]:
```
fig, ax = plt.subplots(ncols=2, nrows=2, figsize=(14,14))

df.iloc[:, 4:8].boxplot(ax=ax[0,0])
df.iloc[:, 10:14].boxplot(ax=ax[0,1])
df.iloc[:, 14:17].boxplot(ax=ax[1,0])
df.iloc[:, 18:22].boxplot(ax=ax[1,1])



ax[1, 1].set_xticklabels(ax[1, 1].get_xticklabels(), rotation=90);
```









4 Resampling

Pandas understands how to do all sorts of time related things. It has many methods that let us manipulate timeseries in useful ways.

For example there is a method called resampling that oranizes the data based on a different time bin than the origional data.

Let's resample from daily to monthly data:

[19]:
```
# monthly reampler object
rs_obj = df.resample('MS')
rs_obj
```

[19]:
```
<pandas.core.resample.DatetimeIndexResampler object at 0x7f9170d67160>
```

The output of the `.resample()` function is an object where the data has been storted into bins where all the data in a bin come from the same month. We then need to do some calculation on that bin to get a vaule. THis is a `groupby` - type operation. we can use this to get monthly-means of all the data

[20]:
```
rs_obj.mean()
```

[20]:
```
             WBANNO    CRX_VN  LONGITUDE  LATITUDE  T_DAILY_MAX  T_DAILY_MIN  \
LST_DATE
2017-01-01  64756.0  2.422000     -73.74     41.79     3.945161    -3.993548
2017-02-01  64756.0  2.422000     -73.74     41.79     7.246429    -4.360714
2017-03-01  64756.0  2.422000     -73.74     41.79     5.164516    -5.335484
2017-04-01  64756.0  2.422000     -73.74     41.79    17.813333     5.170000
2017-05-01  64756.0  2.422000     -73.74     41.79    19.151613     7.338710
2017-06-01  64756.0  2.422000     -73.74     41.79    25.423333    12.176667
2017-07-01  64756.0  2.422000     -73.74     41.79    26.912903    15.183871
2017-08-01  64756.0  2.422000     -73.74     41.79    25.741935    12.954839
2017-09-01  64756.0  2.422000     -73.74     41.79    24.186667    11.300000
2017-10-01  64756.0  2.602645     -73.74     41.79    21.043333     7.150000
2017-11-01  64756.0  2.622000     -73.74     41.79    10.346667    -2.093333
2017-12-01  64756.0  2.622000     -73.74     41.79     1.496774    -7.412903

            T_DAILY_MEAN  T_DAILY_AVG  P_DAILY_CALC  SOLARAD_DAILY  …  \
LST_DATE                                                            …
2017-01-01     -0.025806     0.038710      3.090323       4.690000  …
2017-02-01      1.442857     1.839286      2.414286      10.364286  …
2017-03-01     -0.090323     0.167742      3.970968      13.113548  …
2017-04-01     11.493333    11.540000      2.300000      14.645000  …
2017-05-01     13.229032    13.638710      4.141935      16.519677  …
2017-06-01     18.796667    18.986667      3.743333      21.655000  …
2017-07-01     21.048387    20.993548      2.732258      20.566129  …
2017-08-01     19.351613    19.477419      2.758065      18.360000  …
2017-09-01     17.746667    17.463333      1.893333      15.154667  …
2017-10-01     14.100000    13.976667      3.500000      10.395000  …
2017-11-01      4.120000     4.336667      0.826667       6.723333  …
2017-12-01     -2.967742    -2.838710      2.109677       4.474194  …

            SOIL_MOISTURE_5_DAILY  SOIL_MOISTURE_10_DAILY  \
LST_DATE
2017-01-01               0.236900                0.248300
2017-02-01               0.226333                0.243000
2017-03-01               0.218033                0.229267
```

```
2017-04-01               0.199733                0.210300
2017-05-01               0.206613                0.210935
2017-06-01               0.185167                0.184300
2017-07-01               0.131226                0.115774
2017-08-01               0.143871                0.122258
2017-09-01               0.145167                0.139633
2017-10-01               0.140567                0.131467
2017-11-01               0.215433                0.211233
2017-12-01               0.231536                0.226143

            SOIL_MOISTURE_20_DAILY  SOIL_MOISTURE_50_DAILY  \
LST_DATE
2017-01-01                0.204550                0.152806
2017-02-01                0.207545                0.152857
2017-03-01                0.196258                0.153484
2017-04-01                0.190667                0.151000
2017-05-01                0.185613                0.147710
2017-06-01                0.173167                0.142533
2017-07-01                0.116613                0.121032
2017-08-01                0.105452                0.115290
2017-09-01                0.117267                0.112167
2017-10-01                0.084967                0.105667
2017-11-01                0.167067                0.149800
2017-12-01                0.177581                0.155516

            SOIL_MOISTURE_100_DAILY  SOIL_TEMP_5_DAILY  SOIL_TEMP_10_DAILY  \
LST_DATE
2017-01-01                 0.175194           0.209677            0.267742
2017-02-01                 0.175786           1.125000            1.100000
2017-03-01                 0.174548           2.122581            2.161290
2017-04-01                 0.172400          11.066667           10.666667
2017-05-01                 0.170000          16.454839           16.290323
2017-06-01                 0.167000          22.350000           22.166667
2017-07-01                 0.156677          24.993548           24.980645
2017-08-01                 0.151034          23.374194           23.519355
2017-09-01                 0.141926          20.256667           20.386667
2017-10-01                 0.133367          16.133333           16.186667
2017-11-01                 0.164367           7.230000            7.190000
2017-12-01                 0.169161           2.222581            2.187097

            SOIL_TEMP_20_DAILY  SOIL_TEMP_50_DAILY  SOIL_TEMP_100_DAILY
LST_DATE
2017-01-01            0.696774            1.438710             2.877419
2017-02-01            1.192857            1.492857             2.367857
2017-03-01            2.345161            2.700000             3.387097
2017-04-01            9.636667            8.426667             6.903333
2017-05-01           15.361290           14.270968            12.696774
```

```
2017-06-01           20.880000          19.370000           17.333333
2017-07-01           23.925806          22.745161           21.164516
2017-08-01           22.848387          22.193548           21.377419
2017-09-01           19.966667          19.766667           19.530000
2017-10-01           16.320000          16.836667           17.470000
2017-11-01            8.060000           9.543333           11.746667
2017-12-01            2.916129           4.190323            6.303226

[12 rows x 26 columns]
```

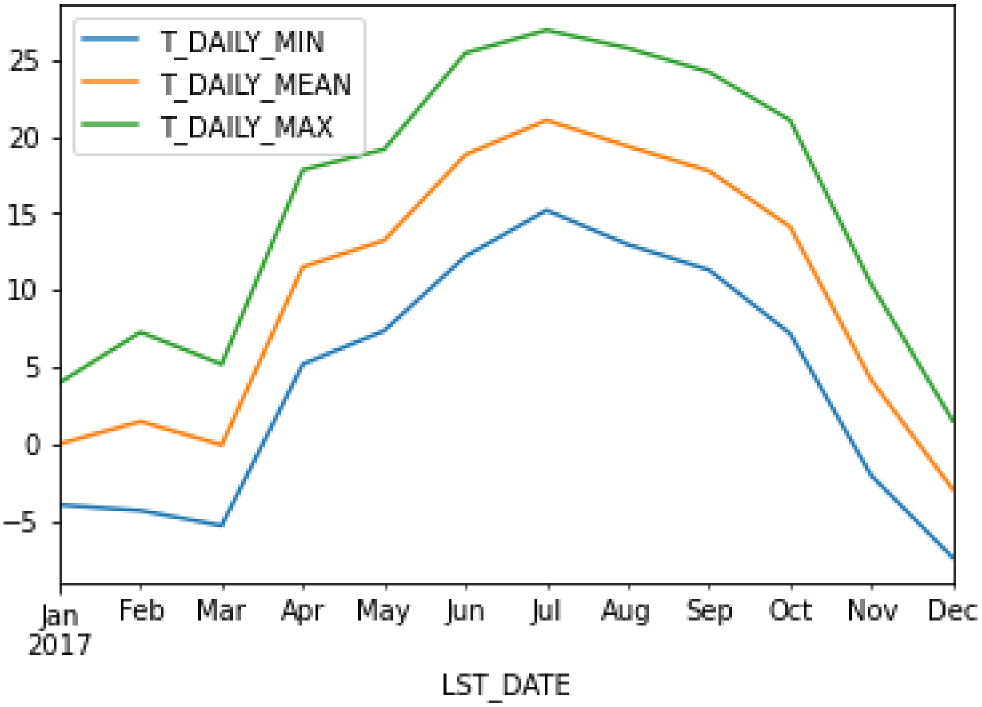
we can chain all these commands together to plot the monthly mean of average, high and low temperatures:

[21]:

```
df_mm = df.resample('MS').mean()
df_mm[['T_DAILY_MIN', 'T_DAILY_MEAN', 'T_DAILY_MAX']].plot()
```

[21]:

```
<AxesSubplot:xlabel='LST_DATE'>
```


T_DAILY_MIN
T_DAILY_MEAN
T_DAILY_MAX
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
LST_DATE