使用Pandas探索能源数据集

读取数据集read_csv( )

In [228]:
```
import pandas as pd
opsd_daily = pd.read_csv('opsd_germany_daily.csv')
```

In [229]:
```
opsd_daily.shape
```

Out[229]:
```
(4383, 5)
```

In [230]:
```
opsd_daily.head(3)
```

Out[230]:

	Date	Consumption	Wind	Solar	Wind+Solar
0	2006-01-01	1069.184	NaN	NaN	NaN
1	2006-01-02	1380.521	NaN	NaN	NaN
2	2006-01-03	1442.533	NaN	NaN	NaN

In [231]:
```
opsd_daily.tail(3)
```

Out[231]:

	Date	Consumption	Wind	Solar	Wind+Solar
4380	2017-12-29	1295.08753	584.277	29.854	614.131
4381	2017-12-30	1215.44897	721.247	7.467	728.714
4382	2017-12-31	1107.11488	721.176	19.980	741.156

数据查询 loc[ ]

In [232]:
```
opsd_daily.loc[0]
```

Out[232]:
```
Date           2006-01-01
Consumption      1069.184
Wind                  NaN
Solar                 NaN
Wind+Solar            NaN
Name: 0, dtype: object
```

In [233]:
```
opsd_daily.loc[4380]
```

Out[233]:
```
Date           2017-12-29
Consumption    1295.08753
Wind               584.277
Solar               29.854
Wind+Solar         614.131
Name: 4380, dtype: object
```

设置数据索引set_index()

In [234]:
```
opsd_daily = opsd_daily.set_index('Consumption')
```

In [235]:
```
opsd_daily.tail(3)
```

Out[235]:

	Date	Wind	Solar	Wind+Solar
Consumption				
1295.08753	2017-12-29	584.277	29.854	614.131
1215.44897	2017-12-30	721.247	7.467	728.714
1107.11488	2017-12-31	721.176	19.980	741.156

In [236]:
```
opsd_daily = pd.read_csv('opsd_germany_daily.csv')
opsd_daily = opsd_daily.set_index('Date')
```

In [237]:
```
opsd_daily.tail(3)
```

Out[237]:

Date	Consumption	Wind	Solar	Wind+Solar
2017-12-29	1295.08753	584.277	29.854	614.131
2017-12-30	1215.44897	721.247	7.467	728.714
2017-12-31	1107.11488	721.176	19.980	741.156

In [238]:
```
opsd_daily.loc['2017-12-30']
```

Out[238]:
```
Consumption    1215.44897
Wind            721.24700
Solar             7.46700
Wind+Solar      728.71400
Name: 2017-12-30, dtype: float64
```

In [239]:
```
opsd_daily.loc['2016-12-25']
```

Out[239]:
```
Consumption    1117.673
Wind            719.778
Solar             6.608
Wind+Solar      726.386
Name: 2016-12-25, dtype: float64
```

1. data = pd.read_csv('data_path') # 读取数据

2. data.set_index('index_name') # 设置索引

3. data.loc[idx] # 根据设置的索引读取数据

In [240]:
```
opsd_daily.loc['2017-12']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~/opt/anaconda3/envs/pose_estimation/lib/python3.7/site-packages/pandas/core/indexes/base.py in get_loc
(self, key, method, tolerance)
   3079             try:
-> 3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: '2017-12'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
/var/folders/0r/3nv9krpx00sfg0jn2t3lt8700000gn/T/ipykernel_57926/4010321873.py in <module>
----> 1 opsd_daily.loc['2017-12']

~/opt/anaconda3/envs/pose_estimation/lib/python3.7/site-packages/pandas/core/indexing.py in __getitem__
(self, key)
    893
    894             maybe_callable = com.apply_if_callable(key, self.obj)
--> 895             return self._getitem_axis(maybe_callable, axis=axis)
    896
    897     def _is_scalar_access(self, key: Tuple):

~/opt/anaconda3/envs/pose_estimation/lib/python3.7/site-packages/pandas/core/indexing.py in _getitem_ax
is(self, key, axis)
   1122         # fall thru to straight lookup
   1123         self._validate_key(key, axis)
-> 1124         return self._get_label(key, axis=axis)
   1125
   1126     def _get_slice_axis(self, slice_obj: slice, axis: int):

~/opt/anaconda3/envs/pose_estimation/lib/python3.7/site-packages/pandas/core/indexing.py in _get_label(
self, label, axis)
   1071     def _get_label(self, label, axis: int):
   1072         # GH#5667 this will fail if the label is not present in the axis.
-> 1073         return self.obj.xs(label, axis=axis)
   1074
   1075     def _handle_lowerdim_multi_index_axis0(self, tup: Tuple):

~/opt/anaconda3/envs/pose_estimation/lib/python3.7/site-packages/pandas/core/generic.py in xs(self, key
, axis, level, drop_level)
   3737                     raise TypeError(f"Expected label or tuple of labels, got {key}") from e
   3738         else:
-> 3739             loc = index.get_loc(key)
   3740
   3741             if isinstance(loc, np.ndarray):

~/opt/anaconda3/envs/pose_estimation/lib/python3.7/site-packages/pandas/core/indexes/base.py in get_loc
(self, key, method, tolerance)
   3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:
-> 3082                 raise KeyError(key) from err
   3083
   3084         if tolerance is not None:

KeyError: '2017-12'
```

pd.read_csv( ) 实用参数：index_col ; parse_dates

In [241]:
```
opsd_daily = pd.read_csv('opsd_germany_daily.csv', index_col=0, parse_dates=True)
```

In [242]:
```
opsd_daily.loc['2017-12-25']
```

Out[242]:
```
Consumption    1111.28338
Wind            587.81000
Solar            15.76500
Wind+Solar      603.57500
Name: 2017-12-25 00:00:00, dtype: float64
```

In [243]:
```
opsd_daily.loc['2017-12']
```

Out[243]:

Date	Consumption	Wind	Solar	Wind+Solar
2017-12-01	1592.96187	52.323	19.266	71.589
2017-12-02	1391.85405	126.274	16.459	142.733
2017-12-03	1330.26226	387.490	12.411	399.901
2017-12-04	1620.97758	479.798	10.747	490.545
2017-12-05	1643.72307	611.488	10.953	622.441
2017-12-06	1639.08265	632.501	7.618	640.119
2017-12-07	1628.47979	743.725	42.994	786.719
2017-12-08	1618.05658	652.830	20.504	673.334
2017-12-09	1415.34531	712.317	12.344	724.661
2017-12-10	1318.10964	622.944	9.922	632.866
2017-12-11	1614.15862	415.109	5.669	420.778
2017-12-12	1647.36346	590.101	13.250	603.351
2017-12-13	1651.90418	721.540	36.880	758.420
2017-12-14	1636.54375	666.438	21.030	687.468
2017-12-15	1576.93197	176.418	21.653	198.071
2017-12-16	1382.87708	277.391	15.904	293.295
2017-12-17	1297.21916	250.726	12.620	263.346
2017-12-18	1578.69079	134.843	15.897	150.740
2017-12-19	1586.48230	99.098	8.793	107.891
2017-12-20	1559.68569	90.880	8.799	99.679
2017-12-21	1520.37206	259.039	7.313	266.352
2017-12-22	1423.23782	228.773	10.065	238.838
2017-12-23	1272.17085	748.074	8.450	756.524
2017-12-24	1141.75730	812.422	9.949	822.371
2017-12-25	1111.28338	587.810	15.765	603.575
2017-12-26	1130.11683	717.453	30.923	748.376
2017-12-27	1263.94091	394.507	16.530	411.037
2017-12-28	1299.86398	506.424	14.162	520.586
2017-12-29	1295.08753	584.277	29.854	614.131
2017-12-30	1215.44897	721.247	7.467	728.714
2017-12-31	1107.11488	721.176	19.980	741.156

In [244]:
```
opsd_daily.loc['2017']
```

Out[244]:

Date	Consumption	Wind	Solar	Wind+Solar
2017-01-01	1130.41300	307.125	35.291	342.416
2017-01-02	1441.05200	295.099	12.479	307.578
2017-01-03	1529.99000	666.173	9.351	675.524
2017-01-04	1553.08300	686.578	12.814	699.392
2017-01-05	1547.23800	261.758	20.797	282.555
...	...	...	...	...
2017-12-27	1263.94091	394.507	16.530	411.037
2017-12-28	1299.86398	506.424	14.162	520.586
2017-12-29	1295.08753	584.277	29.854	614.131
2017-12-30	1215.44897	721.247	7.467	728.714
2017-12-31	1107.11488	721.176	19.980	741.156

365 rows × 4 columns

1. 读取excel文件：转换成.csv格式，然后使用pandas包进行读取。

```
read_csv('file_name.csv')
```

2. set_index()设置索引

3. loc[]按照索引读取数据

4. read_csv('file_name.csv', index_col, parse_dates)

读取文件的同时，自动设置索引，解析日期。

可视化时间序列

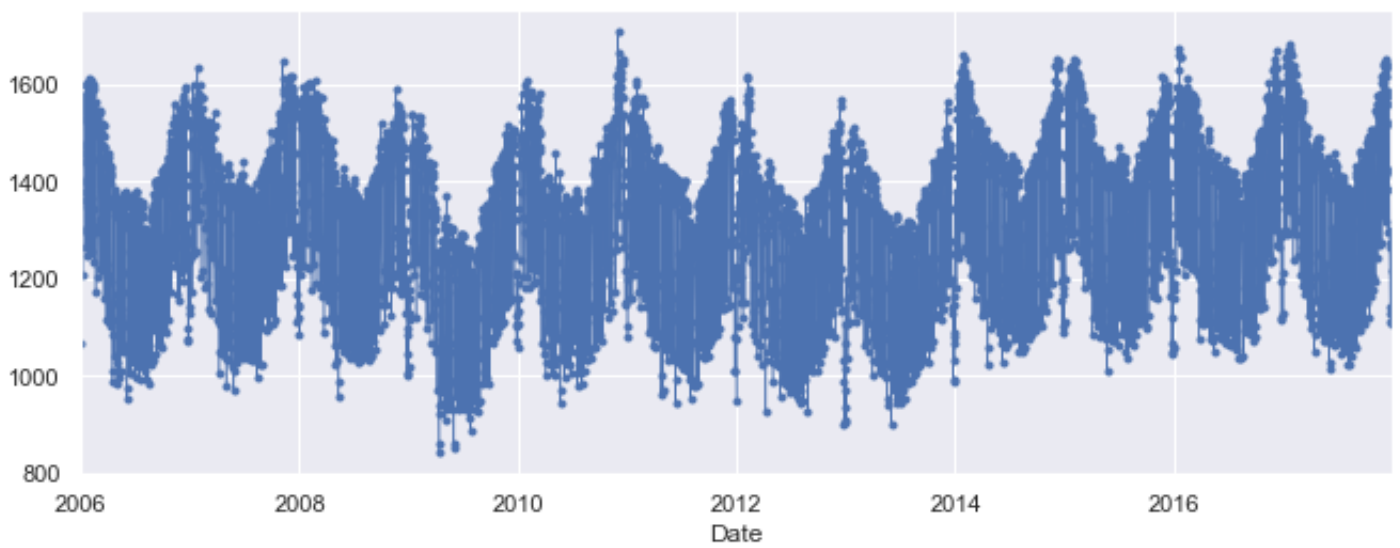
In [245]:
```
import matplotlib.pyplot as plt # 画图工具包
import seaborn as sns # 图样式设置
sns.set(rc={'figure.figsize':(11, 4)})
```

折线图：opsd_daily['Consumption'].plot(linewidth=0.5,marker='.');

默认横坐标：数据索引（dates）

可视化数据设置：'Consumption'

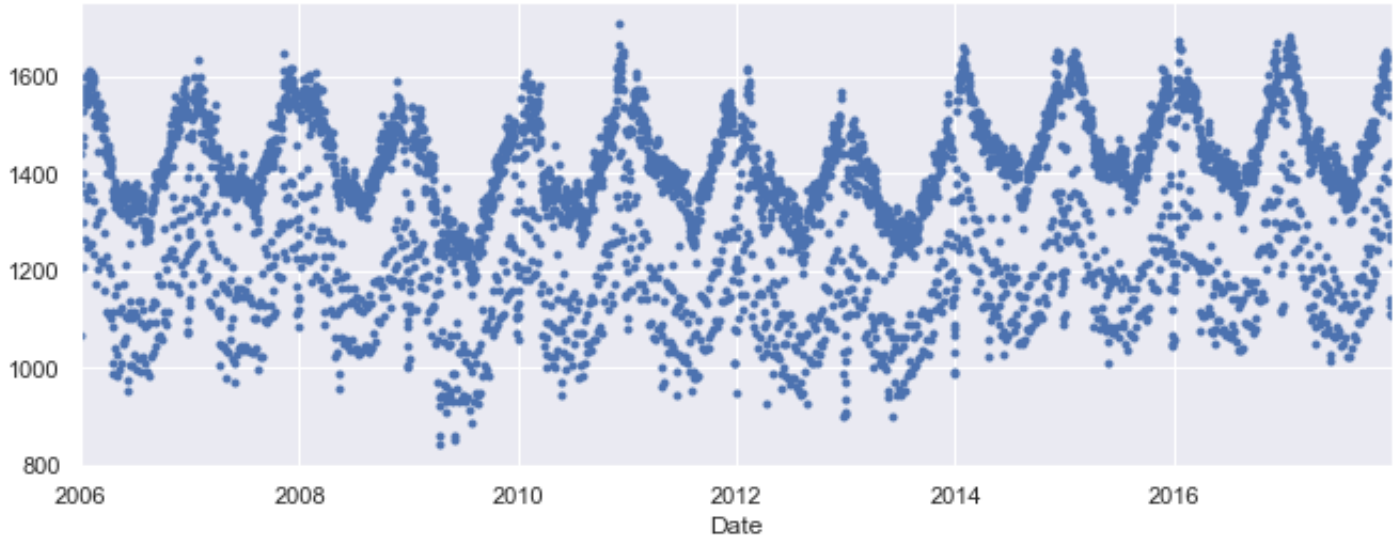
In [246]:
```
opsd_daily['Consumption'].plot(linewidth=0.5,marker='.');  #折线图
```



数据点过于密集：1.使用散点图 2.数据降采样

折线图基础上设置linestyle：opsd_daily['Consumption'].plot(marker='.',linestyle='None');

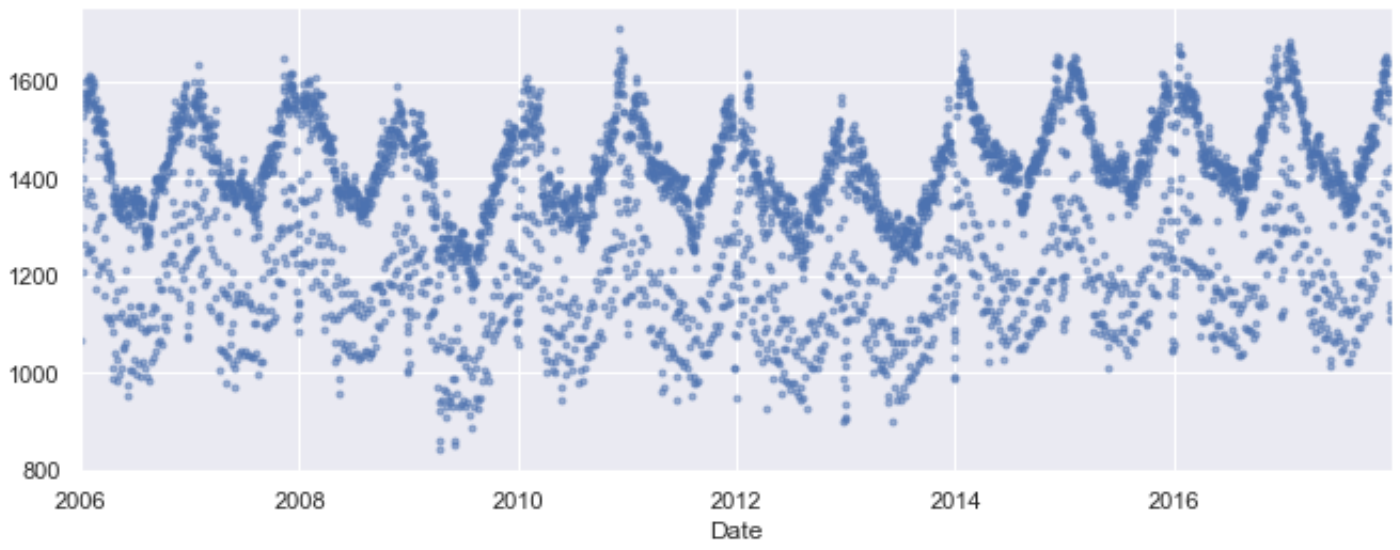
In [247]:
```
opsd_daily['Consumption'].plot(marker='.',linestyle='None'); #散点图
```



数据点重叠：设置透明度

设置alpha：opsd_daily['Consumption'].plot(marker='.',linestyle='None',alpha=0.5);

In [248]:
```
opsd_daily['Consumption'].plot(marker='.',linestyle='None',alpha=0.5);
```

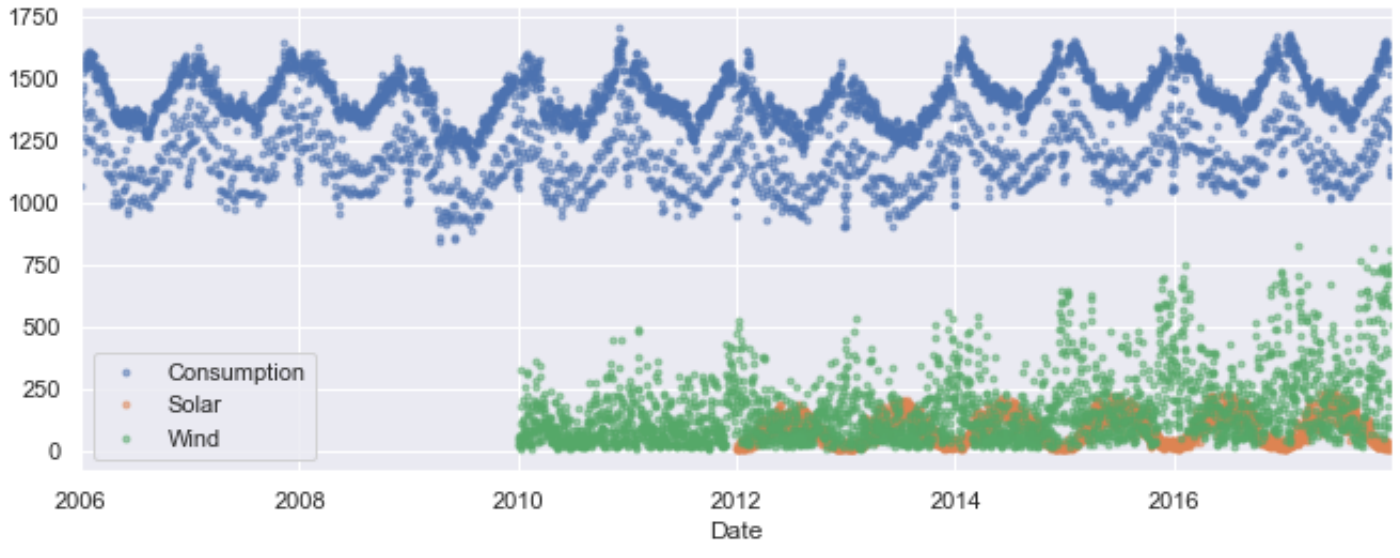


如何一张图，可视化多栏数据：列表输入

设置可视化数据的类型列表 ['Consumption', 'Solar', 'Wind']

axes = opsd_daily[['Consumption', 'Solar', 'Wind']].plot(marker='.', alpha=0.5, linestyle='None')

In [249]:
```
axes = opsd_daily[['Consumption', 'Solar', 'Wind']].plot(marker='.', alpha=0.5, linestyle='None')
```



1.折线图 2.散点图 3.透明度 4.可视化不同数据

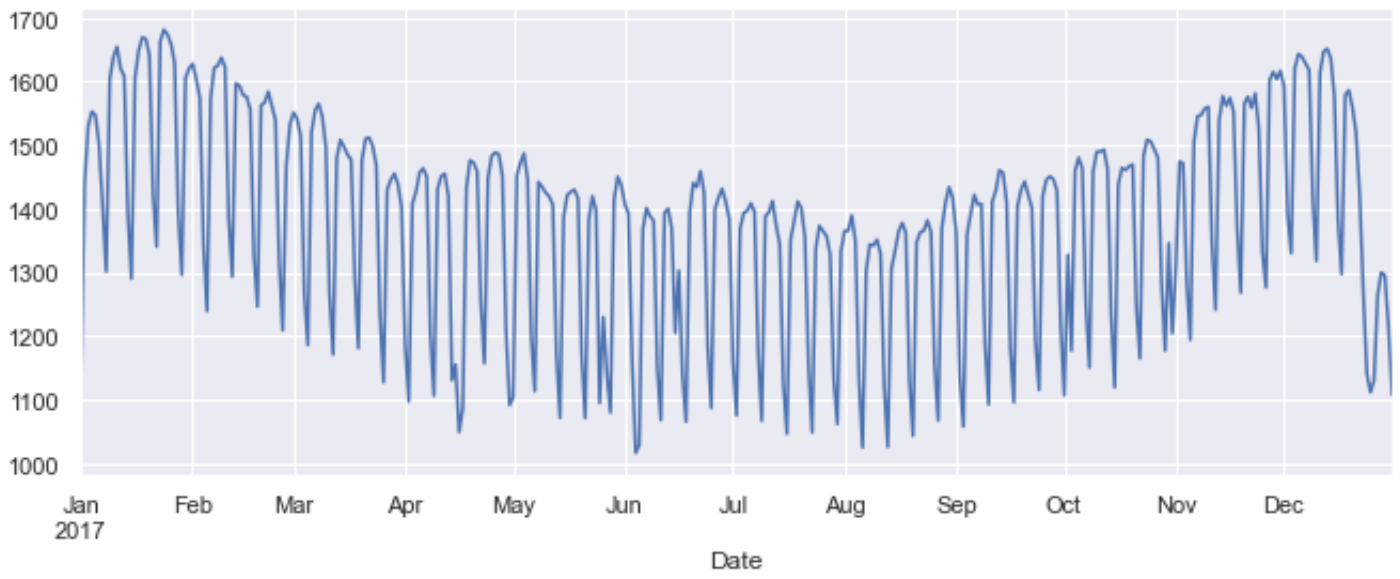
如何设置横坐标区间，实现某一时间段的数据可视化

结合.loc[ ]函数，对时间段定位： opsd_daily.loc['2017', 'Consumption'].plot()

'2017'：横坐标范围。

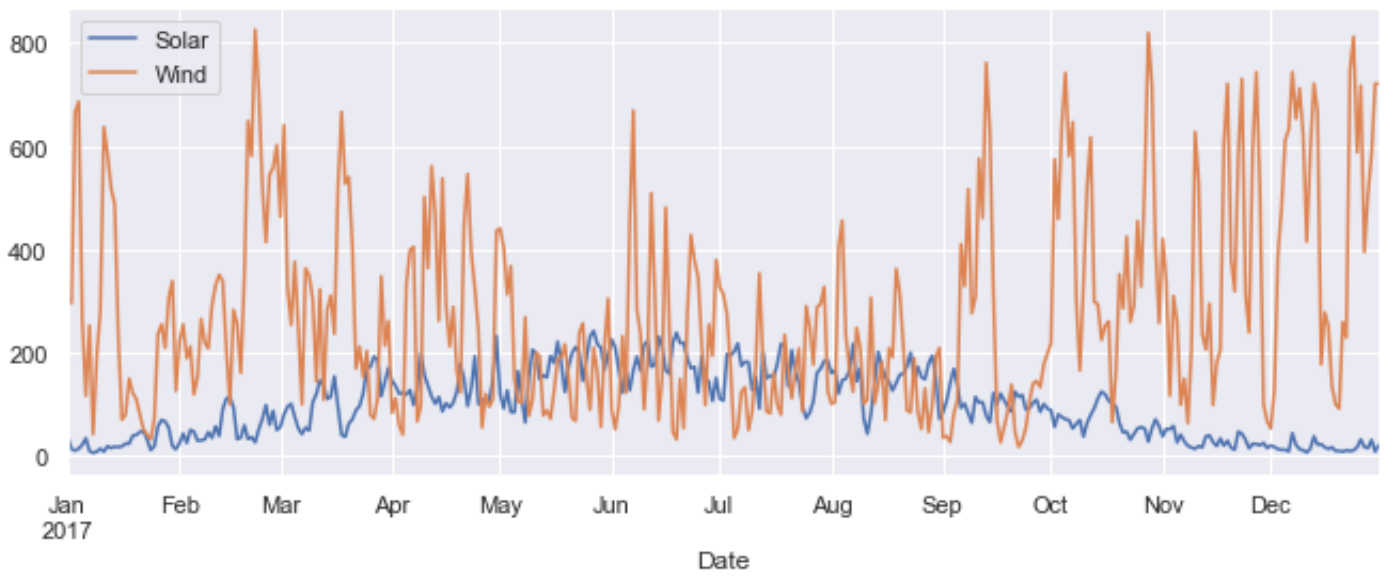
'Consumption'：可视化数据类型

In [250]:
```
ax = opsd_daily.loc['2017', 'Consumption'].plot()
```



某一段时间+多栏数据

In [251]:
```
ax = opsd_daily.loc['2017', ['Solar','Wind']].plot()
```



增加图的可读性：设置标签

ax = opsd_daily.loc['2017', ['Solar','Wind']].plot()

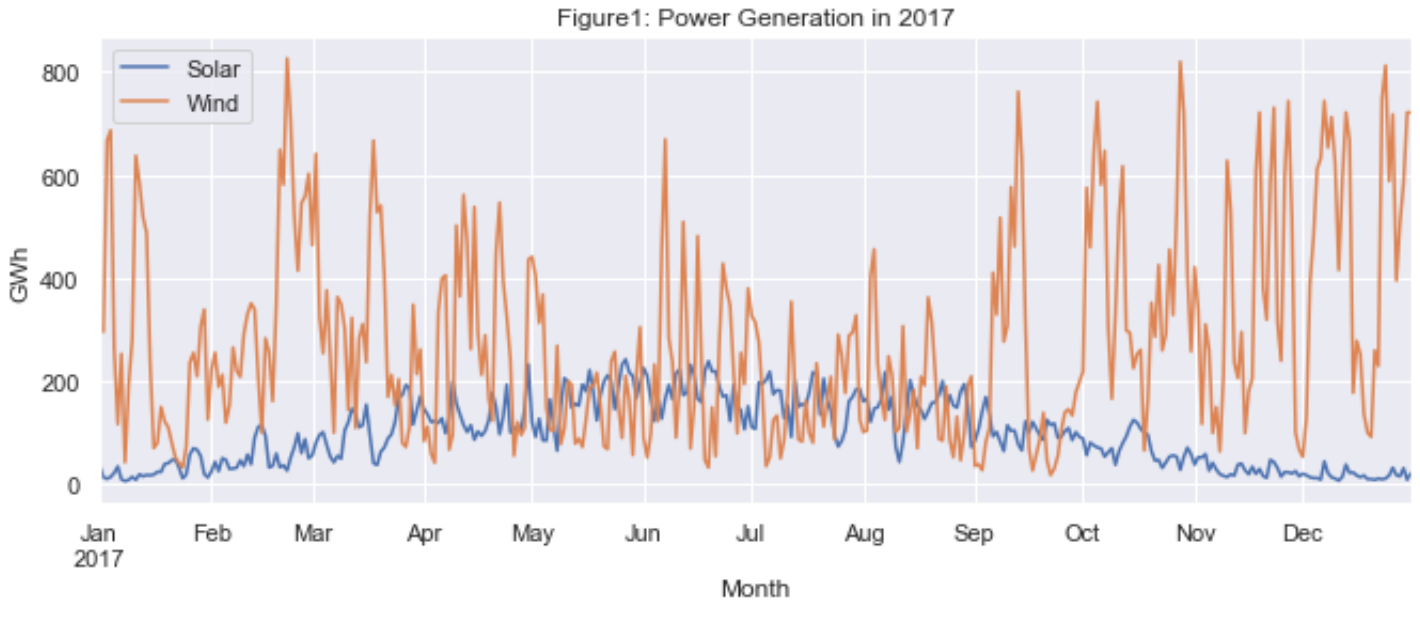
ax.set_xlabel()

ax.set_ylabel()

ax.set_title()

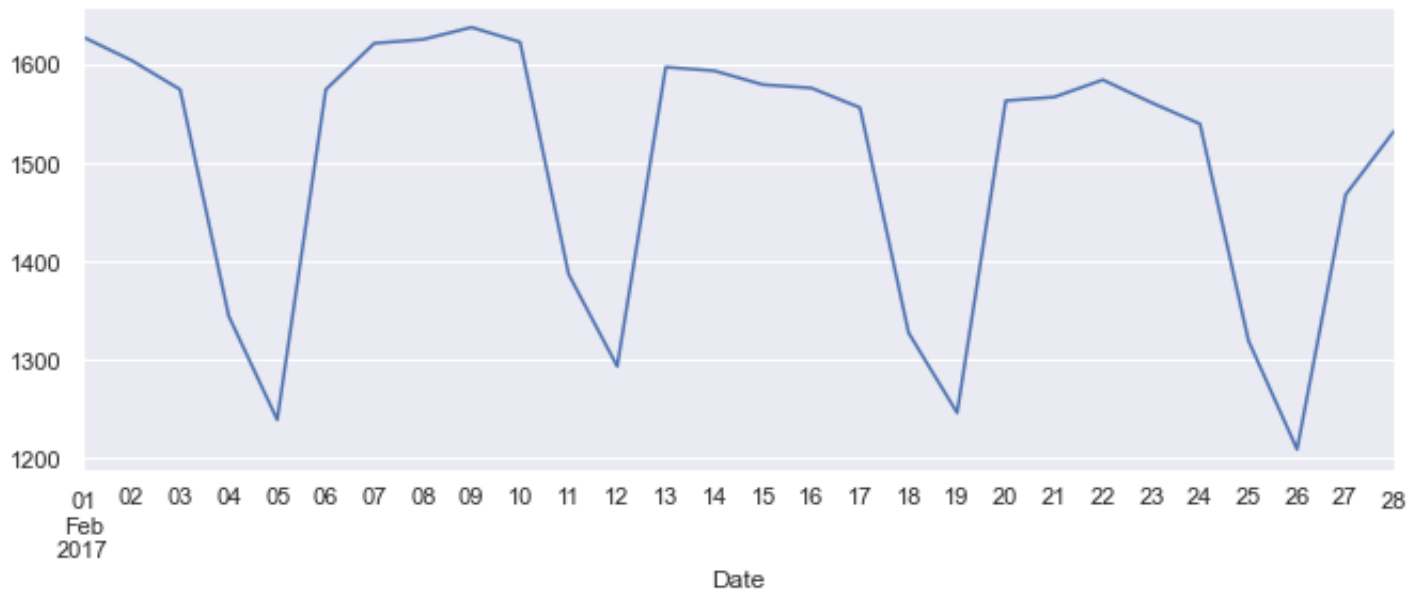
In [252]:
```
ax = opsd_daily.loc['2017', ['Solar','Wind']].plot()
ax.set_xlabel('Month')
ax.set_ylabel('GWh')
ax.set_title('Figure1: Power Generation in 2017')
```

Out[252]: Text(0.5, 1.0, 'Figure1: Power Generation in 2017')

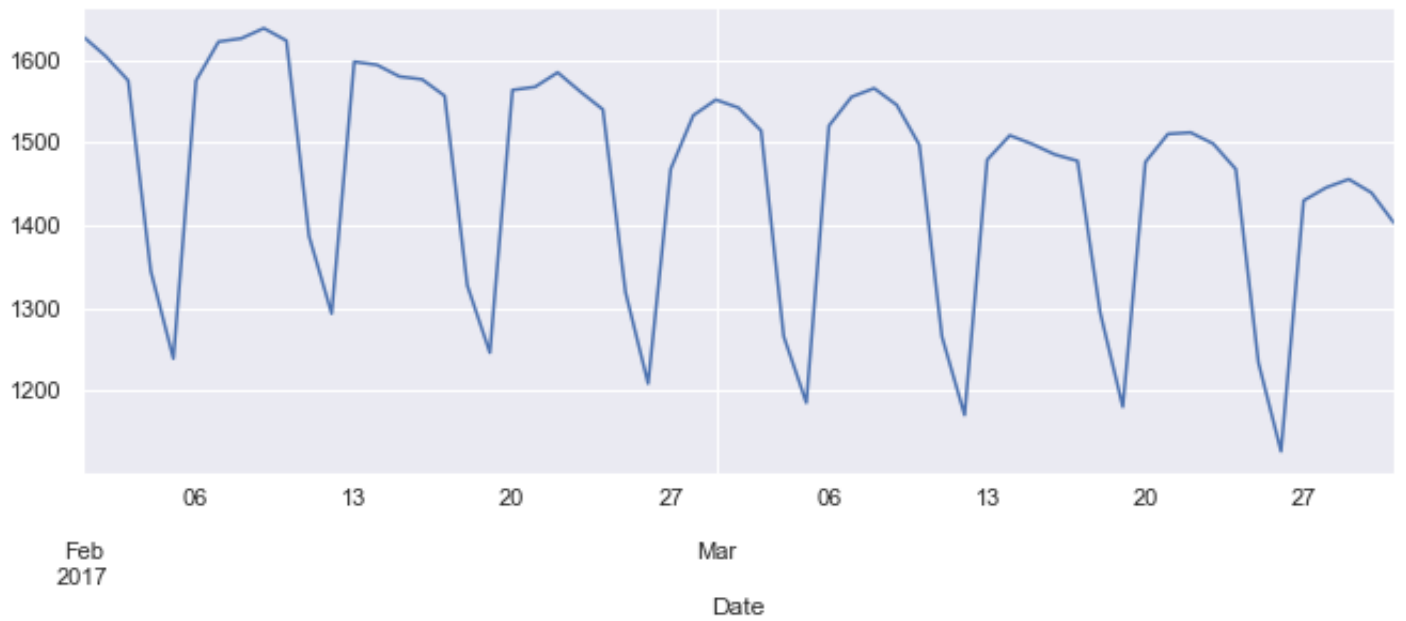


设置竖直线

In [253]:
```
ax = opsd_daily.loc['2017-02', 'Consumption'].plot()
```



In [254]:
```
ax = opsd_daily.loc['2017-02':'2017-03', 'Consumption'].plot()
```



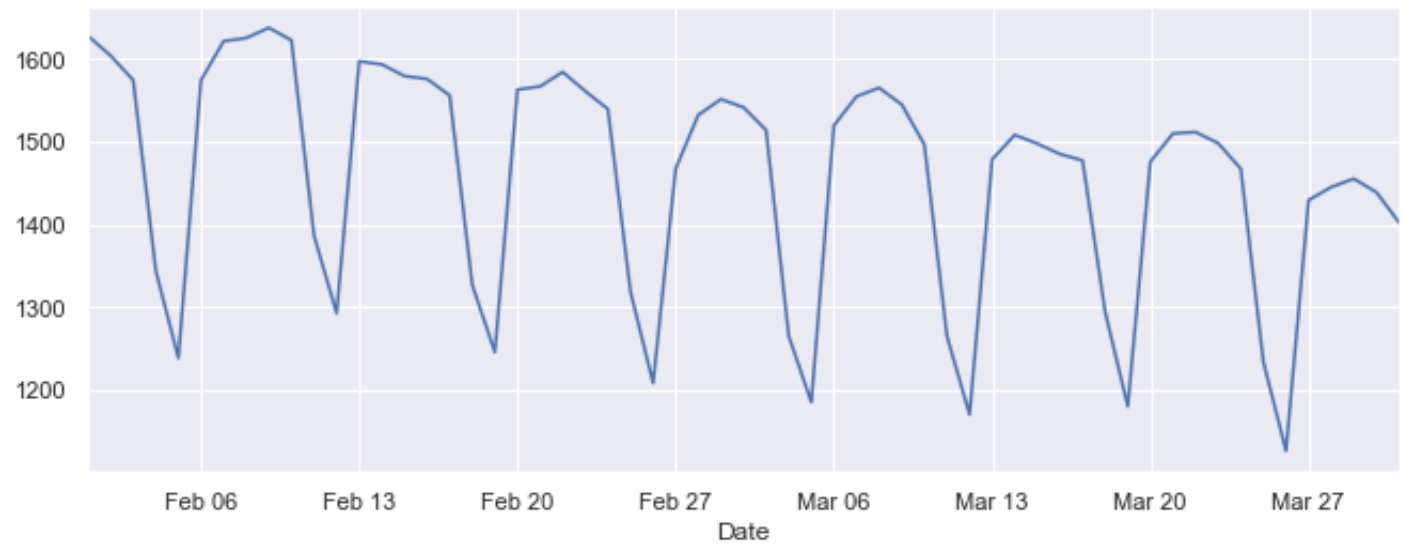
竖直线默认以月为间隔

自定义竖直线间隔： import matplotlib.dates as mdates

ax.xaxis.set_major_locator(mdates.WeekdayLocator(byweekday=mdates.MONDAY))

ax.xaxis.set_major_formatter(mdates.DateFormatter('%b %d'))

In [255]:
```
import matplotlib.dates as mdates
ax = opsd_daily.loc['2017-02':'2017-03', 'Consumption'].plot()
# Set x-axis major ticks to weekly interval, on Mondays
ax.xaxis.set_major_locator(mdates.WeekdayLocator(byweekday=mdates.MONDAY))
ax.xaxis.set_major_formatter(mdates.DateFormatter('%b %d'))
```



1. 控制横坐标范围。 2. 通过设置标签、网格增加图的可读性

数据处理：asfreq()、resample()、rolling()

数据上采样asfreq()： 数据插值、补全缺失数据

In [256]:
```
#times_sample = pd.to_datetime()
# Select the specified dates and just the Consumption column
consum_sample = opsd_daily.loc[['2013-02-03', '2013-02-06', '2013-02-08'], ['Consumption']].copy()
consum_sample
```

Out[256]:

Date	Consumption
2013-02-03	1109.639
2013-02-06	1451.449
2013-02-08	1433.098

以天为单位插值: 'D'

consum_freq = consum_sample.asfreq('D')

In [257]:
```
consum_freq = consum_sample.asfreq('D')
consum_freq
```

Out[257]:

Date	Consumption
2013-02-03	1109.639
2013-02-04	NaN
2013-02-05	NaN
2013-02-06	1451.449
2013-02-07	NaN
2013-02-08	1433.098

插值方式：method='ffill' 向上补全

consum_freq = consum_sample.asfreq(freq='D', method='ffill')

In [258]:
```
consum_freq = consum_sample.asfreq(freq='D', method='ffill')
# method : {‘bfill’, ‘ffill’}
consum_freq
```

Out[258]:

Date	Consumption
2013-02-03	1109.639
2013-02-04	1109.639
2013-02-05	1109.639
2013-02-06	1451.449
2013-02-07	1451.449
2013-02-08	1433.098

插值方式：method='bfill' 向下补全

consum_freq = consum_sample.asfreq(freq='D', method='bfill')

In [259]:
```
consum_freq = consum_sample.asfreq(freq='D', method='bfill')
consum_freq
```

Out[259]:

Date	Consumption
2013-02-03	1109.639
2013-02-04	1451.449
2013-02-05	1451.449
2013-02-06	1451.449
2013-02-07	1433.098
2013-02-08	1433.098

数据下采样 resample(): 求平均、求和、减少数据冗余

opsd_weekly_mean = opsd_daily[data_columns].resample('W').mean()

下采样周期： 'W' 每周

下采样方式： mean( ) 求平均

In [260]:
```
data_columns = ['Consumption', 'Wind', 'Solar', 'Wind+Solar']
# Resample to weekly frequency, aggregating with mean
opsd_weekly_mean = opsd_daily[data_columns].resample('W').mean()
```

In [261]:
```
print(opsd_daily.shape[0])
print(opsd_weekly_mean.shape[0])
```

```
4383
627
```

In [262]:
```
opsd_weekly_mean.head(3)
```

Out[262]:

Date	Consumption	Wind	Solar	Wind+Solar
2006-01-01	1069.184000	NaN	NaN	NaN
2006-01-08	1381.300143	NaN	NaN	NaN
2006-01-15	1486.730286	NaN	NaN	NaN

opsd_monthly = opsd_daily[data_columns].resample('M').sum()

下采样周期： 'M' 每月

下采样方式： sum( ) 求和

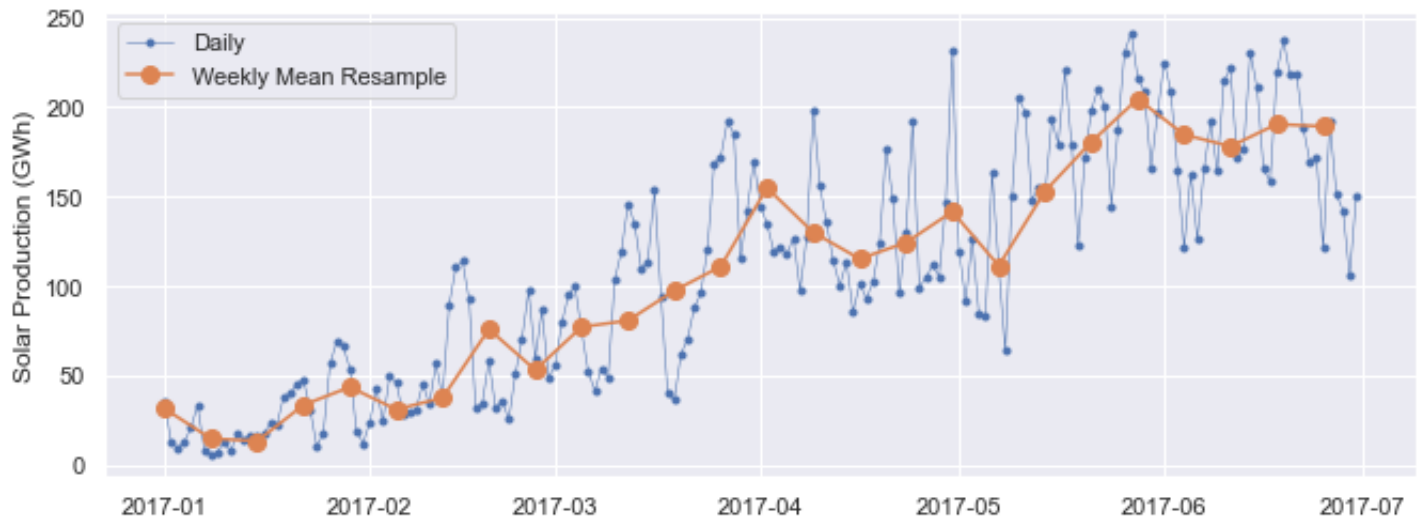
In [263]:
```
opsd_monthly = opsd_daily[data_columns].resample('M').sum()
opsd_monthly.head(3)
```

Out[263]:

Date	Consumption	Wind	Solar	Wind+Solar
2006-01-31	45304.704	0.0	0.0	0.0
2006-02-28	41078.993	0.0	0.0	0.0
2006-03-31	43978.124	0.0	0.0	0.0

利用下采样后的数据画图

In [264]:
```
start, end = '2017-01', '2017-06'
fig, ax = plt.subplots()
ax.plot(opsd_daily.loc[start:end, 'Solar'],marker='.', linestyle='-', linewidth=0.5, label='Daily')
ax.plot(opsd_weekly_mean.loc[start:end, 'Solar'],marker='o', markersize=8, linestyle='-', label='Weekly
ax.set_ylabel('Solar Production (GWh)')
ax.legend();
```



滑动窗口 rolling( ): 求数据变化趋势

opsd_7d = opsd_daily[data_columns].rolling(7, center=True).mean()

窗口大小： 7 ， 每周的变化趋势

计算方式： mean( ), max(), min(), sum()

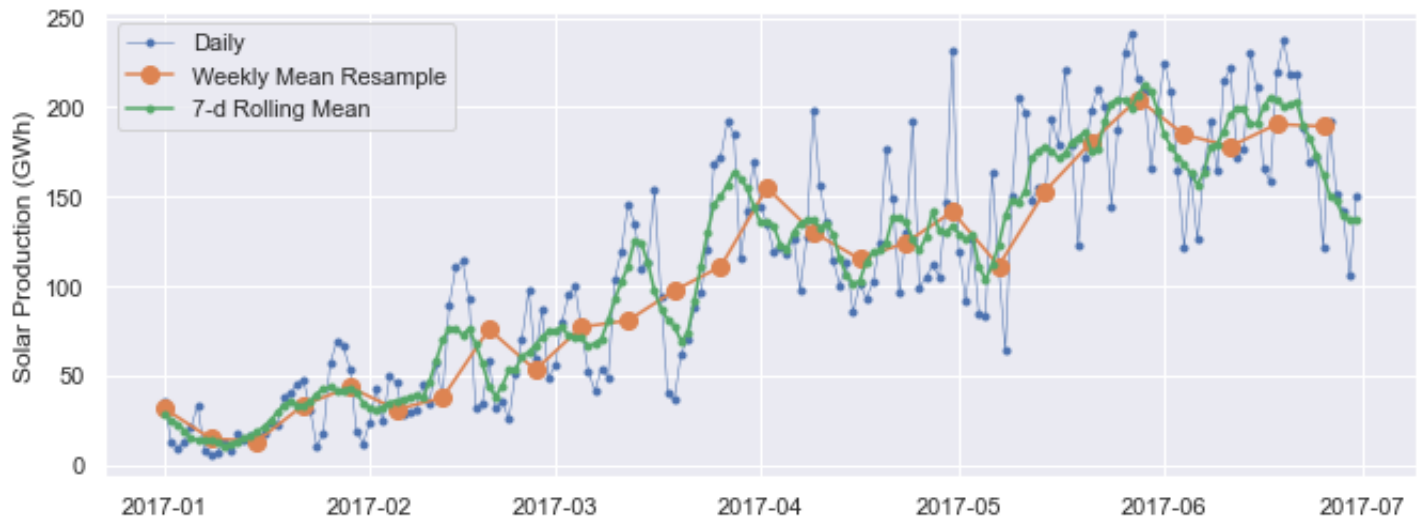
In [265]:
```
opsd_7d = opsd_daily[data_columns].rolling(7, center=True).mean()
opsd_7d.head(10)
```

Out[265]:

Date	Consumption	Wind	Solar	Wind+Solar
2006-01-01	NaN	NaN	NaN	NaN
2006-01-02	NaN	NaN	NaN	NaN
2006-01-03	NaN	NaN	NaN	NaN
2006-01-04	1361.471429	NaN	NaN	NaN
2006-01-05	1381.300143	NaN	NaN	NaN
2006-01-06	1402.557571	NaN	NaN	NaN
2006-01-07	1421.754429	NaN	NaN	NaN
2006-01-08	1438.891429	NaN	NaN	NaN
2006-01-09	1449.769857	NaN	NaN	NaN
2006-01-10	1469.994857	NaN	NaN	NaN

周变化趋势图

In [266]:
```
start, end = '2017-01', '2017-06'
# Plot daily, weekly resampled, and 7-day rolling mean time series together
fig, ax = plt.subplots()
ax.plot(opsd_daily.loc[start:end, 'Solar'],marker='.', linestyle='-', linewidth=0.5, label='Daily')
ax.plot(opsd_weekly_mean.loc[start:end, 'Solar'],marker='o', markersize=8, linestyle='-', label='Weekly
ax.plot(opsd_7d.loc[start:end, 'Solar'],marker='.', linestyle='-', label='7-d Rolling Mean')
ax.set_ylabel('Solar Production (GWh)')
ax.legend();
```

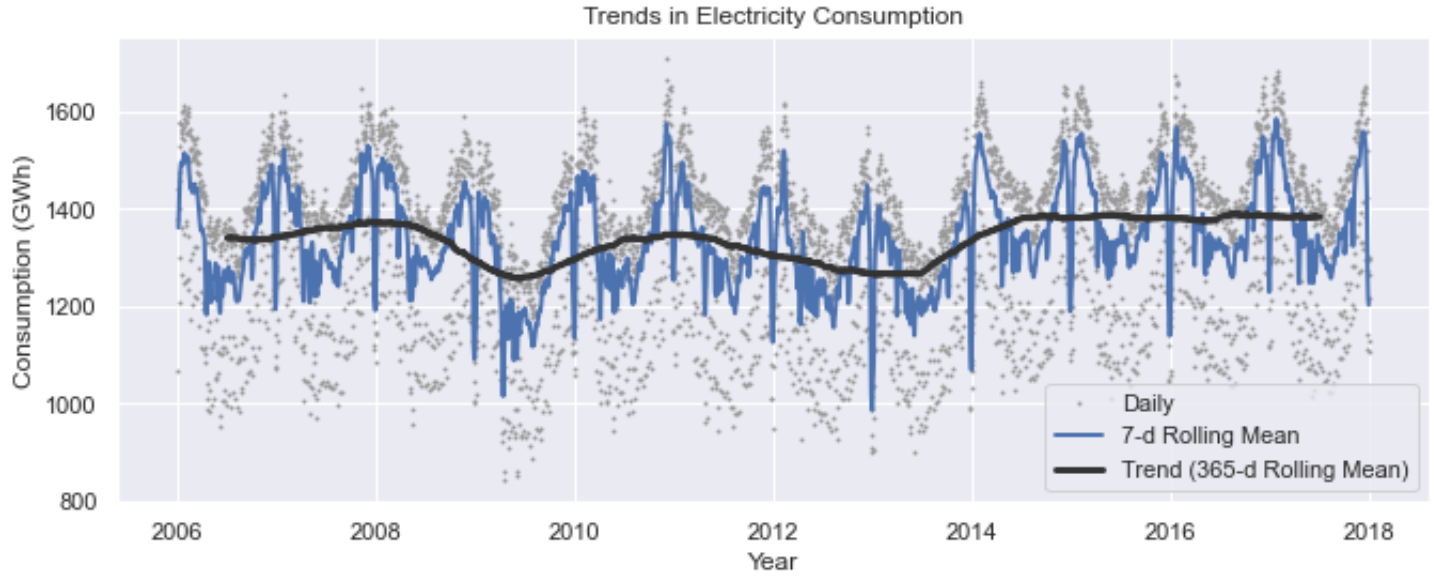


psd_365d = opsd_daily[data_columns].rolling(window=365, center=True).mean()

窗口大小：365, 对应年变化趋势

In [267]:
```
opsd_365d = opsd_daily[data_columns].rolling(window=365, center=True).mean()
```

In [268]:
```
import matplotlib.dates as mdates
fig, ax = plt.subplots()
ax.plot(opsd_daily['Consumption'], marker='.', markersize=2, color='0.6',linestyle='None', label='Daily'
ax.plot(opsd_7d['Consumption'], linewidth=2, label='7-d Rolling Mean')
ax.plot(opsd_365d['Consumption'], color='0.2', linewidth=3,label='Trend (365-d Rolling Mean)')
# Set x-ticks to yearly interval and add legend and labels
ax.legend()
ax.set_xlabel('Year')
ax.set_ylabel('Consumption (GWh)')
ax.set_title('Trends in Electricity Consumption');
```



In [269]:
```
ig, ax = plt.subplots()
for nm in ['Wind', 'Solar', 'Wind+Solar']:
    ax.plot(opsd_365d[nm], label=nm)
    # Set x-ticks to yearly interval, adjust y-axis limits, add legend and labels
    ax.xaxis.set_major_locator(mdates.YearLocator())
    ax.set_ylim(0, 400)
    ax.legend()
    ax.set_ylabel('Production (GWh)')
    ax.set_title('Trends in Electricity Production (365-d Rolling Means)');
```