Database-style Operations on Dataframes About the data In this notebook, we will using daily weather data that was taken from the National Centers for Environmental Information (NCEI) API . The data collection notebook contains the process that was followed to collect the data.

Note: The NCEI is part of the National Oceanic and Atmospheric Administration (NOAA) and, as you can see from the URL for the API, this resource was created when the NCEI was called the NCDC. Should the URL for this resource change in the future, you can search for the NCEI weather API to find the updated one.

Background on the data Data meanings:

PRCP : precipitation in millimeters

SNOW : snowfall in millimeters

SNWD : snow depth in millimeters

TMAX : maximum daily temperature in Celsius

TMIN : minimum daily temperature in Celsius

TOBS : temperature at time of observation in Celsius

WESF : water equivalent of snow in millimeters

Setup

```
import pandas as pd
weather = pd.read_csv('nyc_weather_2018.csv')
weather.head()
```

	attributes	datatype	date	station	value
0	,,N,	PRCP	2018-01-01T00:00:00	GHCND:US1CTFR0039	0.0
1	,,N,	PRCP	2018-01-01T00:00:00	GHCND:US1NJBG0015	0.0
2	,,N,	SNOW	2018-01-01T00:00:00	GHCND:US1NJBG0015	0.0
3	,,N,	PRCP	2018-01-01T00:00:00	GHCND:US1NJBG0017	0.0
4	,,N,	SNOW	2018-01-01T00:00:00	GHCND:US1NJBG0017	0.0

Next steps: View recommended plots New interactive sheet

Querying DataFrames

The query() method is an easier way of filtering based on some criteria. For example, we can use it to find all entries where snow was recorded:

```
snowdata = weather.query('datatype == "SNOW" and value > 0')
snowdata.head(5)
```

	attributes	datatype	date	station	value
124	,,N,	SNOW	2018-01-01T00:00:00	GHCND:US1NYWC0019	25.0
723	,,N,	SNOW	2018-01-04T00:00:00	GHCND:US1NJBG0015	229.0
726	,,N,	SNOW	2018-01-04T00:00:00	GHCND:US1NJBG0017	10.0
730	,,N,	SNOW	2018-01-04T00:00:00	GHCND:US1NJBG0018	46.0
737	,,N,	SNOW	2018-01-04T00:00:00	GHCND:US1NJES0018	10.0

```
import sqlite3
```

```
with sqlite3.connect('weather.db') as connection:
    weather.to_sql('weather', connection, if_exists='replace', index=False)

    snow_data_from_db = pd.read_sql(
        'SELECT * FROM weather WHERE datatype == "SNOW" AND value > 0',
        connection
    )
```

```
snowdata.reset_index().drop(columns='index').equals(snow_data_from_db)
```

```
True
```

```
weather[(weather.datatype == 'SNOW') & (weather.value > 0)].equals(snowdata)
```

```
True
```

Merging DataFrames

We have data for many different stations each day; however, we don't know what the stations are just their IDs. We can join the data in the data/weather_stations.csv file which contains information from the stations endpoint of the NCEI API. Consult the weather_data_collection.ipynb notebook to see how this was collected. It looks like this:

```
station_info = pd.read_csv('weather_stations.csv')
station_info.head()
```

	id	name	latitude	longitude	elevation
0	GHCND:US1CTFR0022	STAMFORD 2.6 SSW, CT US	41.0641	-73.5770	36.6
1	GHCND:US1CTFR0039	STAMFORD 4.2 S, CT US	41.0378	-73.5682	6.4
2	GHCND:US1NJBG0001	BERGENFIELD 0.3 SW, NJ US	40.9213	-74.0020	20.1
3	GHCND:US1NJBG0002	SADDLE BROOK TWP 0.6 E, NJ US	40.9027	-74.0834	16.8
4	GHCND:US1NJBG0003	TENAFLY 1.3 W, NJ US	40.9147	-73.9775	21.6

Next steps: Generate code with station_info View recommended plots New interactive sheet

```
weather
```

	attributes	datatype	date	station	value
0	,,N,	PRCP	2018-01-01T00:00:00	GHCND:US1CTFR0039	0.0
1	,,N,	PRCP	2018-01-01T00:00:00	GHCND:US1NJBG0015	0.0
2	,,N,	SNOW	2018-01-01T00:00:00	GHCND:US1NJBG0015	0.0
3	,,N,	PRCP	2018-01-01T00:00:00	GHCND:US1NJBG0017	0.0
4	,,N,	SNOW	2018-01-01T00:00:00	GHCND:US1NJBG0017	0.0
...	...	...	...	...	...
80251	,,W,	WDF5	2018-12-31T00:00:00	GHCND:USW00094789	130.0
80252	,,W,	WSF2	2018-12-31T00:00:00	GHCND:USW00094789	9.8
80253	,,W,	WSF5	2018-12-31T00:00:00	GHCND:USW00094789	12.5
80254	,,W,	WT01	2018-12-31T00:00:00	GHCND:USW00094789	1.0
80255	,,W,	WT02	2018-12-31T00:00:00	GHCND:USW00094789	1.0

80256 rows × 5 columns

Next steps: Generate code with weather View recommended plots New interactive sheet

```
station_info.id.describe()
```

	id
count	262
unique	262
top	GHCND:USW00094789
freq	1

dtype: object

```
weather.station.describe()
```

	station
count	80256
unique	109
top	GHCND:USW00094789
freq	4270

dtype: object

```
station_info.shape[0], weather.shape[0]
```

```
(262, 80256)
```

```
def grc(*dfs):
  return [df.shape[0] for df in dfs]
grc(station_info, weather)
```

```
[262, 80256]
```

```
def getinf(attr, *dfs):
  return list(map(lambda x: getattr(x, attr), dfs))
getinf('shape', station_info,weather)
```

```
[(262, 5), (80256, 5)]
```

```
injoin = weather.merge(station_info,left_on='station', right_on='id')
injoin.sample(5, random_state=0)
```

	attributes	datatype	date	station	value	id	name	latitude	longitude	elevati
27422	,,W,	WDF5	2018-04-29T00:00:00	GHCND:USW00094741	310.0	GHCND:USW00094741	TETERBORO AIRPORT, NJ US	40.85000	-74.06139	2
19317	,,W,	WSF5	2018-03-24T00:00:00	GHCND:USW00094728	8.5	GHCND:USW00094728	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42
13778	,,W,	PGTM	2018-03-	GHCND:USW00054743	2351.0	GHCND:USW00054743	CALDWELL ESSEX CO	40.87639	-74.28306	52

```
weather.merge(station_info.rename(dict(id='station'),axis=1),on='station').sample(5, random_state=0)
```

	attributes	datatype	date	station	value	name	latitude	longitude	elevation
27422	,,W,	WDF5	2018-04-29T00:00:00	GHCND:USW00094741	310.0	TETERBORO AIRPORT, NJ US	40.85000	-74.06139	2.7
19317	,,W,	WSF5	2018-03-24T00:00:00	GHCND:USW00094728	8.5	NY CITY CENTRAL PARK, NY US	40.77898	-73.96925	42.7
13778	,,W,	PGTM	2018-03-01T00:00:00	GHCND:USW00054743	2351.0	CALDWELL ESSEX CO AIRPORT, NJ US	40.87639	-74.28306	52.7

```
left_join = station_info.merge(weather, left_on='id',right_on='station', how='left')
right_join = weather.merge(station_info, left_on='station',right_on='id', how='right')
```

```
right_join.tail()
```

	attributes	datatype	date	station	value	id	name	latitude	longitude	elevatio
80404	,,W,	WDF5	2018-12-31T00:00:00	GHCND:USW00094789	130.0	GHCND:USW00094789	JFK INTERNATIONAL AIRPORT, NY US	40.6386	-73.7622	3.
80405	,,W,	WSF2	2018-12-31T00:00:00	GHCND:USW00094789	9.8	GHCND:USW00094789	JFK INTERNATIONAL AIRPORT, NY US	40.6386	-73.7622	3.

```
left_join.sort_index(axis=1).sort_values(['date','station']).reset_index().drop(columns='index').equals(
    right_join.sort_index(axis=1).sort_values(['date','station']).reset_index().drop(columns='index')
)
```

```
True
```

```
getinf('shape', injoin, left_join, right_join)
```

```
[(80256, 10), (80409, 10), (80409, 10)]
```

```
outer_join = weather.merge(
    station_info[station_info.name.str.contains('NY')],
    left_on='station',right_on='id', how='outer', indicator=True
)
pd.concat([outer_join.sample(4, random_state=0),outer_join[outer_join.station.isna()].head(2)])
```

	attributes	datatype	date	station	value	id	name	latitude	longitude	elevati
17259	,,N,	SNOW	2018-07-20T00:00:00	GHCND:US1NJMS0075	0.0	NaN	NaN	NaN	NaN	Na
76178	,,W,	AWND	2018-01-12T00:00:00	GHCND:USW00094789	7.2	GHCND:USW00094789	JFK INTERNATIONAL AIRPORT, NY US	40.63860	-73.7622	3
73410	T,,W,2400	PRCP	2018-03-16T00:00:00	GHCND:USW00094745	0.0	GHCND:USW00094745	WESTCHESTER CO AIRPORT, NY US	41.06694	-73.7075	11

```
import sqlite3 as sq3

with sq3.connect('weather.db') as connection:
  station_info.to_sql('weather_stations', connection, if_exists='replace', index=False)
```

```
with sq3.connect('weather.db') as connection:
  inner_join_from_db = pd.read_sql('SELECT * FROM weather JOIN weather_stations ON weather.station == weather_stations.id', connection)
```

```
dirty_data = pd.read_csv('dirty_data2.csv', index_col='date').drop_duplicates().drop(columns='SNWD')
dirty_data.head()
```

	station	PRCP	SNOW	TMAX	TMIN	TOBS	WESF	inclement_weather
date								
2018-01-01T00:00:00	?	0.0	0.0	5505.0	-40.0	NaN	NaN	NaN
2018-01-02T00:00:00	GHCND:USC00280907	0.0	0.0	-8.3	-16.1	-12.2	NaN	False
2018-01-03T00:00:00	GHCND:USC00280907	0.0	0.0	-4.4	-13.9	-13.3	NaN	False
2018-01-04T00:00:00	?	20.6	229.0	5505.0	-40.0	NaN	19.3	True
2018-01-05T00:00:00	?	0.3	NaN	5505.0	-40.0	NaN	NaN	NaN

Next steps: Generate code with dirty_data | View recommended plots | New interactive sheet

```
valid_station = dirty_data.query('station != "?"').copy().drop(columns=['WESF','station'])
sta_with_wesf = dirty_data.query('station == "?"').copy().drop(columns=['station', 'TOBS', 'TMIN', 'TMAX'])
```

```
valid_station.merge(sta_with_wesf, left_index=True, right_index=True).query('WESF>0').head()
```

	PRCP_x	SNOW_x	TMAX	TMIN	TOBS	inclement_weather_x	PRCP_y	SNOW_y	WESF	inclement_weather_y
date										
2018-01-30T00:00:00	0.0	0.0	6.7	-1.7	-0.6	False	1.5	13.0	1.8	True
2018-03-08T00:00:00	48.8	NaN	1.1	-0.6	1.1	False	28.4	NaN	28.7	NaN
2018-03-13T00:00:00	4.1	51.0	5.6	-3.9	0.0	True	3.0	13.0	3.0	True
2018-03-21T00:00:00	0.0	0.0	2.8	-2.8	0.6	False	6.6	114.0	8.6	True
2018-04-02T00:00:00	9.1	127.0	12.8	-1.1	-1.1	True	14.0	152.0	15.2	True

```
valid_station.merge(sta_with_wesf, left_index=True, right_index=True, suffixes = ('','_?')).query('WESF>0').head()
```

	PRCP	SNOW	TMAX	TMIN	TOBS	inclement_weather	PRCP_?	SNOW_?	WESF	inclement_weather_?
date										
2018-01-30T00:00:00	0.0	0.0	6.7	-1.7	-0.6	False	1.5	13.0	1.8	True
2018-03-08T00:00:00	48.8	NaN	1.1	-0.6	1.1	False	28.4	NaN	28.7	NaN
2018-03-13T00:00:00	4.1	51.0	5.6	-3.9	0.0	True	3.0	13.0	3.0	True
2018-03-21T00:00:00	0.0	0.0	2.8	-2.8	0.6	False	6.6	114.0	8.6	True
2018-04-02T00:00:00	9.1	127.0	12.8	-1.1	-1.1	True	14.0	152.0	15.2	True

```
valid_station.join(sta_with_wesf, rsuffix='_?').query('WESF >0').head()
```

	PRCP	SNOW	TMAX	TMIN	TOBS	inclement_weather	PRCP_?	SNOW_?	WESF	inclement_weather_?
date										
2018-01-30T00:00:00	0.0	0.0	6.7	-1.7	-0.6	False	1.5	13.0	1.8	True
2018-03-08T00:00:00	48.8	NaN	1.1	-0.6	1.1	False	28.4	NaN	28.7	NaN
2018-03-13T00:00:00	4.1	51.0	5.6	-3.9	0.0	True	3.0	13.0	3.0	True
2018-03-21T00:00:00	0.0	0.0	2.8	-2.8	0.6	False	6.6	114.0	8.6	True
2018-04-02T00:00:00	9.1	127.0	12.8	-1.1	-1.1	True	14.0	152.0	15.2	True

```
weather.set_index('station', inplace=True)
station_info.set_index('id', inplace=True)
```

```
weather.index.intersection(station_info.index)
```

```
Index(['GHCND:US1CTFR0039', 'GHCND:US1NJBG0015', 'GHCND:US1NJBG0017',
       'GHCND:US1NJBG0018', 'GHCND:US1NJBG0023', 'GHCND:US1NJBG0030',
       'GHCND:US1NJBG0039', 'GHCND:US1NJBG0044', 'GHCND:US1NJES0018',
       'GHCND:US1NJES0024',
       ...
       'GHCND:US1NJMS0047', 'GHCND:US1NYSF0083', 'GHCND:US1NYNY0074',
       'GHCND:US1NJPS0018', 'GHCND:US1NJBG0037', 'GHCND:USC00284987',
       'GHCND:US1NJES0031', 'GHCND:US1NJMD0086', 'GHCND:US1NJMS0097',
       'GHCND:US1NJMN0081'],
      dtype='object', length=109)
```

```
weather.index.difference(station_info.index)
```

```
Index([], dtype='object')
```

```
station_info.index.difference(weather.index)
```

```
Index(['GHCND:US1CTFR0022', 'GHCND:US1NJBG0001', 'GHCND:US1NJBG0002',
       'GHCND:US1NJBG0005', 'GHCND:US1NJBG0006', 'GHCND:US1NJBG0008',
       'GHCND:US1NJBG0011', 'GHCND:US1NJBG0012', 'GHCND:US1NJBG0013',
       'GHCND:US1NJBG0020',
       ...
       'GHCND:USC00308322', 'GHCND:USC00308749', 'GHCND:USC00308946',
       'GHCND:USC00309117', 'GHCND:USC00309270', 'GHCND:USC00309400',
       'GHCND:USC00309466', 'GHCND:USC00309576', 'GHCND:USW00014708',
       'GHCND:USW00014786'],
      dtype='object', length=153)
```

```
ny_in_name = station_info[station_info.name.str.contains('NY')]

ny_in_name.index.difference(weather.index).shape[0]\
+ weather.index.difference(ny_in_name.index).shape[0]\
== weather.index.symmetric_difference(ny_in_name.index).shape[0]
```

```
True
```

```
weather.index.unique().union(station_info.index)
```

```
Index(['GHCND:US1CTFR0022', 'GHCND:US1CTFR0039', 'GHCND:US1NJBG0001',
       'GHCND:US1NJBG0002', 'GHCND:US1NJBG0003', 'GHCND:US1NJBG0005',
       'GHCND:US1NJBG0006', 'GHCND:US1NJBG0008', 'GHCND:US1NJBG0010',
       'GHCND:US1NJBG0011',
       ...
       'GHCND:USW00014708', 'GHCND:USW00014732', 'GHCND:USW00014734',
       'GHCND:USW00014786', 'GHCND:USW00054743', 'GHCND:USW00054787',
       'GHCND:USW00094728', 'GHCND:USW00094741', 'GHCND:USW00094745',
       'GHCND:USW00094789'],
      dtype='object', length=262)
```

```
ny_in_name = station_info[station_info.name.str.contains('NY')]
ny_in_name.index.difference(weather.index).union(weather.index.difference(ny_in_name.index)).equals(
weather.index.symmetric_difference(ny_in_name.index)
)
```

```
True
```

Start coding or generate with AI.