Setup

```
1 import pandas as pd
2
3 weather = pd.read_csv('/content/nyc_weather_2018(1).csv')
4 weather.head()
```

Next steps: View recommended plots

Querying DataFrames

```
1 snow_data = weather.query('datatype == "SNOW" and value > 0')
2 snow_data.head()
```

```
1 import sqlite3
2
3 with sqlite3.connect('/content/weather.db') as connection:
4   snow_data_from_db = pd.read_sql('SELECT * FROM weather WHERE datatype == "SNOW" AND value >
5
6 snow_data.reset_index().drop(columns='index').equals(snow_data_from_db)
```

```
True
```

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

```
True
```

Merging DataFrames

```
1 station_info = pd.read_csv('/content/weather_stations.csv')
2 station_info.head()
```

Next steps: View recommended plots

```
1 weather.head()
```

Next steps: View recommended plots

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

```
count                   262
unique                  262
top       GHCND:US1CTFR0022
freq                      1
Name: id, dtype: object
```

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

```
count                 80256
unique                  109
```

```
top       GHCND:USW00094789
freq                   4270
Name: station, dtype: object
```

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

```
(262, 80256)
```

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

```
[262, 80256]
```

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

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

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

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

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

```
1 left_join.sort_index(axis=1).sort_values(['date', 'station']).reset_index().drop(columns='inc
2   right_join.sort_index(axis=1).sort_values(['date', 'station']).reset_index().drop(columns='
3 )
```

```
    True
```

```
1 get_info('shape', inner_join, left_join, right_join)
```

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

```
[(86238, 10), (86409, 10), (86409, 10)]
```

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

```
import sqlite3
with sqlite3.connect('/content/weather.db') as connection:
  inner_join_from_db = pd.read_sql(
    'SELECT * FROM weather JOIN stations ON weather.station == stations.id',
    connection
)
inner_join_from_db.shape == inner_join.shape
```

```
True
```

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

Next steps: View recommended plots

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

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

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

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

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

```
1 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)
```

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

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

```
1 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)
```

```
1 ny_in_name = station_info[station_info.name.str.contains('NY')]
2 ny_in_name.index.difference(weather.index).shape[0]\
3 + weather.index.difference(ny_in_name.index).shape[0]\
4 == weather.index.symmetric_difference(ny_in_name.index).shape[0]
```

```
True
```

```
1 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)
```

```
1 ny_in_name = station_info[station_info.name.str.contains('NY')]
2 ny_in_name.index.difference(weather.index).union(weather.index.difference(ny_in_name.index)).e
3   weather.index.symmetric_difference(ny_in_name.index)
4 )
```

```
True
```