MERGING DATAFRAMES WITH PANDAS

Merging DataFrames

Population DataFrame

```
In [1]: import pandas as pd

In [2]: population = pd.read_csv('pa_zipcode_population.csv')

In [3]: print(population)
   Zipcode  2010 Census Population
0    16855                     282
1    15681                    5241
2    18657                   11985
3    17307                    5899
4    15635                     220
```

Cities DataFrame

```
In [4]: cities = pd.read_csv('pa_zipcode_city.csv')

In [5]: print(cities)
    Zipcode              City State
0     17545           MANHEIM    PA
1     18455     PRESTON PARK     PA
2     17307      BIGLERVILLE     PA
3     15705           INDIANA    PA
4     16833     CURWENSVILLE     PA
5     16220            CROWN     PA
6     18618     HARVEYS LAKE     PA
7     16855  MINERAL SPRINGS     PA
8     16623         CASSVILLE    PA
9     15635       HANNASTOWN     PA
10    15681        SALTSBURG     PA
11    18657      TUNKHANNOCK     PA
12    15279       PITTSBURGH     PA
13    17231        LEMASTERS     PA
14    18821       GREAT BEND     PA
```

Merging

```
In [6]: pd.merge(population, cities)
Out[6]:
   Zipcode  2010 Census Population             City State
0    16855                     282  MINERAL SPRINGS    PA
1    15681                    5241       SALTSBURG    PA
2    18657                   11985     TUNKHANNOCK    PA
3    17307                    5899     BIGLERVILLE    PA
4    15635                     220      HANNASTOWN    PA
```

Medal DataFrames

```
In [7]: bronze = pd.read_csv('bronze_sorted.csv')

In [8]: gold = pd.read_csv('gold_sorted.csv')

In [9]: print(bronze)
   NOC          Country   Total
0  USA    United States  1052.0
1  URS     Soviet Union   584.0
2  GBR  United Kingdom   505.0
3  FRA           France   475.0
4  GER          Germany   454.0

In [10]: print(gold)
   NOC          Country   Total
0  USA    United States  2088.0
1  URS     Soviet Union   838.0
2  GBR  United Kingdom   498.0
3  ITA           Italy   460.0
4  GER          Germany   407.0
```

Merging all columns

```
In [11]: pd.merge(bronze, gold)
Out[11]:
Empty DataFrame
Columns: [NOC, Country, Total]
Index: []
```

Merging on

```
In [12]: pd.merge(bronze, gold, on='NOC')
Out[12]:
   NOC       Country_x  Total_x       Country_y  Total_y
0  USA   United States   1052.0   United States   2088.0
1  URS    Soviet Union    584.0    Soviet Union    838.0
2  GBR  United Kingdom    505.0  United Kingdom    498.0
3  GER         Germany    454.0         Germany    407.0
```

Merging on multiple columns

```
In [13]: pd.merge(bronze, gold, on=['NOC', 'Country'])
Out[13]:
   NOC         Country  Total_x  Total_y
0  USA   United States   1052.0   2088.0
1  URS    Soviet Union    584.0    838.0
2  GBR  United Kingdom    505.0    498.0
3  GER         Germany    454.0    407.0
```

Using suffixes

```
In [14]: pd.merge(bronze, gold, on=['NOC', 'Country'], suffixes=['_bronze', '_gold'])
Out[14]:
   NOC         Country  Total_bronze  Total_gold
0  USA   United States        1052.0      2088.0
1  URS    Soviet Union         584.0       838.0
2  GBR  United Kingdom         505.0       498.0
3  GER         Germany         454.0       407.0
```

Counties DataFrame

```
In [15]: counties = pd.read_csv('pa_counties.csv')

In [16]: print(counties)
          CITY NAME    COUNTY NAME
0         SALTSBURG        INDIANA
1  MINERAL SPRINGS     CLEARFIELD
2       BIGLERVILLE          ADAMS
3        HANNASTOWN   WESTMORELAND
4       TUNKHANNOCK        WYOMING

In [17]: print(cities.tail())
    Zipcode          City State
10    15681     SALTSBURG    PA
11    18657  TUNKHANNOCK     PA
12    15279   PITTSBURGH     PA
13    17231    LEMASTERS     PA
14    18821   GREAT BEND     PA
```

Specifying columns to merge

```
In [18]: pd.merge(counties, cities, left_on='CITY NAME', right_on='City')
Out[18]:
          CITY NAME    COUNTY NAME  Zipcode              City State
0         SALTSBURG        INDIANA    15681         SALTSBURG    PA
1  MINERAL SPRINGS     CLEARFIELD    16855  MINERAL SPRINGS     PA
2      BIGLERVILLE          ADAMS    17307      BIGLERVILLE     PA
3       HANNASTOWN  WESTMORELAND    15635       HANNASTOWN     PA
4      TUNKHANNOCK        WYOMING    18657      TUNKHANNOCK     PA
```

Switching left/right DataFrames

```
In [19]: pd.merge(cities, counties, left_on='City', right_on='CITY NAME')
Out[19]:
   Zipcode              City State         CITY NAME     COUNTY NAME
0    17307         BIGLERVILLE    PA       BIGLERVILLE           ADAMS
1    16855  MINERAL SPRINGS    PA  MINERAL SPRINGS      CLEARFIELD
2    15635         HANNASTOWN    PA       HANNASTOWN    WESTMORELAND
3    15681          SALTSBURG    PA        SALTSBURG         INDIANA
4    18657       TUNKHANNOCK    PA       TUNKHANNOCK         WYOMING
```

MERGING DATAFRAMES WITH PANDAS

Let's practice!

MERGING DATAFRAMES WITH PANDAS

Joining DataFrames

Medal DataFrames

```
In [1]: import pandas as pd

In [2]: bronze = pd.read_csv('bronze_sorted.csv')

In [3]: gold = pd.read_csv('gold_sorted.csv')

In [4]: print(bronze)
   NOC         Country   Total
0  USA   United States  1052.0
1  URS    Soviet Union   584.0
2  GBR  United Kingdom   505.0
3  FRA          France   475.0
4  GER         Germany   454.0

In [5]: print(gold)
   NOC         Country   Total
0  USA   United States  2088.0
1  URS    Soviet Union   838.0
2  GBR  United Kingdom   498.0
3  ITA           Italy   460.0
4  GER         Germany   407.0
```

Merging with inner join

```
In [6]: pd.merge(bronze, gold, on=['NOC', 'Country'],
   ...:          suffixes=['_bronze', '_gold'], how='inner')
Out[6]:
   NOC         Country  Total_bronze  Total_gold
0  USA   United States        1052.0      2088.0
1  URS    Soviet Union         584.0       838.0
2  GBR  United Kingdom         505.0       498.0
3  GER         Germany         454.0       407.0
```

Merging with left join

- Keeps all rows of the left DF in the merged DF
- For rows in the left DF with matches in the right DF:
  - Non-joining columns of right DF are appended to left DF
- For rows in the left DF with no matches in the right DF:
  - Non-joining columns are filled with nulls

Merging with left join

```
In [7]: pd.merge(bronze, gold, on=['NOC', 'Country'],
   ...:          suffixes=['_bronze', '_gold'], how='left')
Out[7]:
   NOC         Country  Total_bronze  Total_gold
0  USA   United States        1052.0      2088.0
1  URS    Soviet Union         584.0       838.0
2  GBR  United Kingdom         505.0       498.0
3  FRA          France         475.0         NaN
4  GER         Germany         454.0       407.0
```

Merging with right join

```
In [8]: pd.merge(bronze, gold, on=['NOC', 'Country'],
   ...:          suffixes=['_bronze', '_gold'], how='right')
Out[8]:
   NOC         Country  Total_bronze  Total_gold
0  USA   United States        1052.0      2088.0
1  URS    Soviet Union         584.0       838.0
2  GBR  United Kingdom         505.0       498.0
3  GER         Germany         454.0       407.0
4  ITA           Italy           NaN       460.0
```

Merging with outer join

```
In [9]: pd.merge(bronze, gold, on=['NOC', 'Country'],
   ...:          suffixes=['_bronze', '_gold'], how='outer')
Out[9]:
   NOC         Country  Total_bronze  Total_gold
0  USA   United States        1052.0      2088.0
1  URS    Soviet Union         584.0       838.0
2  GBR  United Kingdom         505.0       498.0
3  FRA          France         475.0         NaN
4  GER         Germany         454.0       407.0
5  ITA           Italy           NaN       460.0
```

Population & unemployment data

```
In [10]: population = pd.read_csv('population_00.csv', index_col=0)

In [11]: unemployment = pd.read_csv('unemployment_00.csv', index_col=0)

In [12]: print(population)
               2010 Census Population
Zip Code ZCTA
57538                             322
59916                             130
37660                           40038
2860                            45199

In [13]: print(unemployment)
       unemployment  participants
Zip
2860           0.11         34447
46167          0.02          4800
1097           0.33            42
80808          0.07          4310
```

Using .join(how='left')

```
In [16]: population.join(unemployment)
Out[16]:
                2010 Census Population  unemployment  participants
Zip Code ZCTA
57538                              322           NaN           NaN
59916                              130           NaN           NaN
37660                            40038           NaN           NaN
2860                             45199          0.11       34447.0
```

Using .join(how='right')

```
In [17]: population.join(unemployment, how= 'right')
Out[17]:
       2010 Census Population  unemployment  participants
Zip
2860                  45199.0          0.11         34447
46167                     NaN          0.02          4800
1097                      NaN          0.33            42
80808                     NaN          0.07          4310
```

Using .join(how='inner')

```
In [18]: population.join(unemployment, how='inner')
Out[18]:
      2010 Census Population  unemployment  participants
2860                   45199          0.11         34447
```

Using .join(how='outer')

```
In [19]: population.join(unemployment, how= 'outer')
Out[19]:
       2010 Census Population  unemployment  participants
1097                      NaN          0.33          42.0
2860                  45199.0          0.11       34447.0
37660                 40038.0           NaN           NaN
46167                     NaN          0.02        4800.0
57538                   322.0           NaN           NaN
59916                   130.0           NaN           NaN
80808                     NaN          0.07        4310.0
```

Which should you use?

- df1.append(df2): stacking vertically
- pd.concat([df1, df2]):
  - stacking many horizontally or vertically
  - simple inner/outer joins on Indexes
- df1.join(df2): inner/outer/left/right joins on Indexes
- pd.merge([df1, df2]): many joins on multiple columns

MERGING DATAFRAMES WITH PANDAS

Let's practice!

MERGING DATAFRAMES WITH PANDAS

Ordered merges

Software & hardware sales

```
In [1]: import pandas as pd

In [2]: software = pd.read_csv('feb-sales-Software.csv', parse_dates=['Date'])
   ...:                .sort_values('Date')

In [3]: hardware = pd.read_csv('feb-sales-Hardware.csv', parse_dates=['Date'])
   ...:                .sort_values('Date')
```

Software & hardware sales

```
In [4]: print(software)
                  Date           Company   Product  Units
2 2015-02-02 08:33:01             Hooli  Software      3
1 2015-02-03 14:14:18           Initech  Software     13
7 2015-02-04 15:36:29         Streeplex  Software     13
3 2015-02-05 01:53:06  Acme Coporation   Software     19
5 2015-02-09 13:09:55         Mediacore  Software      7
4 2015-02-11 20:03:08           Initech  Software      7
6 2015-02-11 22:50:44             Hooli  Software      4
0 2015-02-16 12:09:19             Hooli  Software     10
8 2015-02-21 05:01:26         Mediacore  Software      3

In [5]: print(hardware)
                  Date           Company   Product  Units
3 2015-02-02 20:54:49         Mediacore  Hardware      9
0 2015-02-04 21:52:45  Acme Coporation   Hardware     14
1 2015-02-07 22:58:10  Acme Coporation   Hardware      1
2 2015-02-19 10:59:33         Mediacore  Hardware     16
4 2015-02-21 20:41:47             Hooli  Hardware      3
```

Using merge()

```
In [6]: pd.merge(hardware, software)
Out[6]:
Empty DataFrame
Columns: [Date, Company, Product, Units]
Index: []
```

Using merge(how='outer')

```
In [7]: pd.merge(hardware, software, how='outer')
Out[7]:
                  Date          Company   Product  Units
0  2015-02-02 20:54:49        Mediacore  Hardware      9
1  2015-02-04 21:52:45  Acme Coporation  Hardware     14
2  2015-02-07 22:58:10  Acme Coporation  Hardware      1
3  2015-02-19 10:59:33        Mediacore  Hardware     16
4  2015-02-21 20:41:47            Hooli  Hardware      3
5  2015-02-02 08:33:01            Hooli  Software      3
6  2015-02-03 14:14:18          Initech  Software     13
7  2015-02-04 15:36:29        Streeplex  Software     13
8  2015-02-05 01:53:06  Acme Coporation  Software     19
9  2015-02-09 13:09:55        Mediacore  Software      7
10 2015-02-11 20:03:08          Initech  Software      7
11 2015-02-11 22:50:44            Hooli  Software      4
12 2015-02-16 12:09:19            Hooli  Software     10
13 2015-02-21 05:01:26        Mediacore  Software      3
```

Sorting merge(how='outer')

```
In [8]: pd.merge(hardware, software, how='outer').sorted_values('Date')
Out[8]:
                  Date          Company   Product  Units
0  2015-02-02 20:54:49        Mediacore  Hardware      9
1  2015-02-04 21:52:45  Acme Coporation  Hardware     14
2  2015-02-07 22:58:10  Acme Coporation  Hardware      1
3  2015-02-19 10:59:33        Mediacore  Hardware     16
4  2015-02-21 20:41:47            Hooli  Hardware      3
5  2015-02-02 08:33:01            Hooli  Software      3
6  2015-02-03 14:14:18          Initech  Software     13
7  2015-02-04 15:36:29        Streeplex  Software     13
8  2015-02-05 01:53:06  Acme Coporation  Software     19
9  2015-02-09 13:09:55        Mediacore  Software      7
10 2015-02-11 20:03:08          Initech  Software      7
11 2015-02-11 22:50:44            Hooli  Software      4
12 2015-02-16 12:09:19            Hooli  Software     10
13 2015-02-21 05:01:26        Mediacore  Software      3
```

Using merge_ordered()

```
In [9]: pd.merge_ordered(hardware, software)
Out[9]:
                  Date          Company   Product  Units
0  2015-02-02 08:33:01            Hooli  Software    3.0
1  2015-02-02 20:54:49        Mediacore  Hardware    9.0
2  2015-02-03 14:14:18          Initech  Software   13.0
3  2015-02-04 15:36:29        Streeplex  Software   13.0
4  2015-02-04 21:52:45  Acme Coporation  Hardware   14.0
5  2015-02-05 01:53:06  Acme Coporation  Software   19.0
6  2015-02-07 22:58:10  Acme Coporation  Hardware    1.0
7  2015-02-09 13:09:55        Mediacore  Software    7.0
8  2015-02-11 20:03:08          Initech  Software    7.0
9  2015-02-11 22:50:44            Hooli  Software    4.0
10 2015-02-16 12:09:19            Hooli  Software   10.0
11 2015-02-19 10:59:33        Mediacore  Hardware   16.0
12 2015-02-21 05:01:26        Mediacore  Software    3.0
13 2015-02-21 20:41:47            Hooli  Hardware    3.0
```

Using on & suffixes

```
In [10]: pd.merge_ordered(hardware, software, on=['Date', 'Company'],
    ...:                  suffixes=['_hardware', '_software']).head()
Out[10]:
                  Date          Company Product_hardware  \
0 2015-02-02 08:33:01            Hooli              NaN
1 2015-02-02 20:54:49        Mediacore         Hardware
2 2015-02-03 14:14:18          Initech              NaN
3 2015-02-04 15:36:29        Streeplex              NaN
4 2015-02-04 21:52:45  Acme Coporation         Hardware

   Units_hardware Product_software  Units_software
0             NaN         Software             3.0
1             9.0              NaN             NaN
2             NaN         Software            13.0
3             NaN         Software            13.0
4            14.0              NaN             NaN
```

Stocks data

```
In [11]: stocks = pd.read_csv('stocks-2013.csv')

In [12]: print(stocks)
          Date        AAPL         IBM       CSCO       MSFT
0   2013-01-31  497.822381  197.271905  20.699524  27.236667
1   2013-02-28  456.808953  200.735788  20.988947  27.704211
2   2013-03-31  441.840998  210.978001  21.335000  28.141000
3   2013-04-30  419.764998  204.733636  20.914545  29.870909
4   2013-05-31  446.452730  205.263639  22.386364  33.950909
5   2013-06-30  425.537999  200.850000  24.375500  34.632500
6   2013-07-31  429.157272  194.354546  25.378636  33.650454
7   2013-08-31  484.843635  187.125000  24.948636  32.485000
8   2013-09-30  480.184499  188.767000  24.080000  32.523500
9   2013-10-31  504.744783  180.710002  22.847391  34.382174
10  2013-11-30  524.616499  181.333502  22.204000  37.362500
11  2013-12-31  559.657613  179.114763  21.257619  37.455715
```

GDP data

```
In [13]: gdp = pd.read_csv('gdp-2013.csv')

In [14]: print(gdp)
         Date      GDP
0  2012-03-31  15973.9
1  2012-06-30  16121.9
2  2012-09-30  16227.9
3  2012-12-31  16297.3
4  2013-03-31  16475.4
5  2013-06-30  16541.4
6  2013-09-30  16749.3
7  2013-12-31  16999.9
```

Ordered merge

```
In [15]: pd.merge_ordered(stocks, gdp, on='Date')
Out[15]:
          Date        AAPL         IBM       CSCO       MSFT      GDP
0   2012-03-31         NaN         NaN        NaN        NaN  15973.9
1   2012-06-30         NaN         NaN        NaN        NaN  16121.9
2   2012-09-30         NaN         NaN        NaN        NaN  16227.9
3   2012-12-31         NaN         NaN        NaN        NaN  16297.3
4   2013-01-31  497.822381  197.271905  20.699524  27.236667      NaN
5   2013-02-28  456.808953  200.735788  20.988947  27.704211      NaN
6   2013-03-31  441.840998  210.978001  21.335000  28.141000  16475.4
7   2013-04-30  419.764998  204.733636  20.914545  29.870909      NaN
8   2013-05-31  446.452730  205.263639  22.386364  33.950909      NaN
9   2013-06-30  425.537999  200.850000  24.375500  34.632500  16541.4
10  2013-07-31  429.157272  194.354546  25.378636  33.650454      NaN
11  2013-08-31  484.843635  187.125000  24.948636  32.485000      NaN
12  2013-09-30  480.184499  188.767000  24.080000  32.523500  16749.3
13  2013-10-31  504.744783  180.710002  22.847391  34.382174      NaN
14  2013-11-30  524.616499  181.333502  22.204000  37.362500      NaN
15  2013-12-31  559.657613  179.114763  21.257619  37.455715  16999.9
```

Ordered merge with ffill

```
In [16]: pd.merge_ordered(stocks, gdp, on='Date', fill_method='ffill')
Out[16]:
          Date        AAPL         IBM       CSCO       MSFT      GDP
0   2012-03-31         NaN         NaN        NaN        NaN  15973.9
1   2012-06-30         NaN         NaN        NaN        NaN  16121.9
2   2012-09-30         NaN         NaN        NaN        NaN  16227.9
3   2012-12-31         NaN         NaN        NaN        NaN  16297.3
4   2013-01-31  497.822381  197.271905  20.699524  27.236667  16297.3
5   2013-02-28  456.808953  200.735788  20.988947  27.704211  16297.3
6   2013-03-31  441.840998  210.978001  21.335000  28.141000  16475.4
7   2013-04-30  419.764998  204.733636  20.914545  29.870909  16475.4
8   2013-05-31  446.452730  205.263639  22.386364  33.950909  16475.4
9   2013-06-30  425.537999  200.850000  24.375500  34.632500  16541.4
10  2013-07-31  429.157272  194.354546  25.378636  33.650454  16541.4
11  2013-08-31  484.843635  187.125000  24.948636  32.485000  16541.4
12  2013-09-30  480.184499  188.767000  24.080000  32.523500  16749.3
13  2013-10-31  504.744783  180.710002  22.847391  34.382174  16749.3
14  2013-11-30  524.616499  181.333502  22.204000  37.362500  16749.3
15  2013-12-31  559.657613  179.114763  21.257619  37.455715  16999.9
```

MERGING DATAFRAMES WITH PANDAS

Let's practice!