

You are currently looking at **version 1.0** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the [Jupyter Notebook FAQ](https://www.coursera.org/learn/python-data-analysis/resources/0dhYG) (<https://www.coursera.org/learn/python-data-analysis/resources/0dhYG>). course resource.

Merging Dataframes

In [1]: `import pandas as pd`

```
df = pd.DataFrame([{'Name': 'Chris', 'Item Purchased': 'Sponge', 'Cost': 22.50},
                   {'Name': 'Kevyn', 'Item Purchased': 'Kitty Litter', 'Cost': 2.50},
                   {'Name': 'Filip', 'Item Purchased': 'Spoon', 'Cost': 5.00}],
                  index=['Store 1', 'Store 1', 'Store 2'])
df
```

Out[1]:

	Cost	Item Purchased	Name
Store 1	22.5	Sponge	Chris
Store 1	2.5	Kitty Litter	Kevyn
Store 2	5.0	Spoon	Filip

In []: `df['Date'] = ['December 1', 'January 1', 'mid-May']`
`df`

In []: `df['Delivered'] = True`
`df`

In []: `df['Feedback'] = ['Positive', None, 'Negative']`
`df`

In []: `adf = df.reset_index()`
`adf['Date'] = pd.Series({0: 'December 1', 2: 'mid-May'})`
`adf`

```
In [ ]: staff_df = pd.DataFrame([{'Name': 'Kelly', 'Role': 'Director of HR'},
                                {'Name': 'Sally', 'Role': 'Course liasion'},
                                {'Name': 'James', 'Role': 'Grader'}])
staff_df = staff_df.set_index('Name')
student_df = pd.DataFrame([{'Name': 'James', 'School': 'Business'},
                            {'Name': 'Mike', 'School': 'Law'},
                            {'Name': 'Sally', 'School': 'Engineering'}])
student_df = student_df.set_index('Name')
print(staff_df.head())
print()
print(student_df.head())
```

```
In [ ]: pd.merge(staff_df, student_df, how='outer', left_index=True, right_index=True)
```

```
In [ ]: pd.merge(staff_df, student_df, how='inner', left_index=True, right_index=True)
```

```
In [ ]: pd.merge(staff_df, student_df, how='left', left_index=True, right_index=True)
```

```
In [ ]: pd.merge(staff_df, student_df, how='right', left_index=True, right_index=True)
```

```
In [ ]: staff_df = staff_df.reset_index()
student_df = student_df.reset_index()
pd.merge(staff_df, student_df, how='left', left_on='Name', right_on='Name')
```

```
In [ ]: staff_df = pd.DataFrame([{'Name': 'Kelly', 'Role': 'Director of HR', 'Location': 'State Street'},
                                {'Name': 'Sally', 'Role': 'Course liasion', 'Location': 'Washington Avenue'},
                                {'Name': 'James', 'Role': 'Grader', 'Location': 'Washington Avenue'}])
student_df = pd.DataFrame([{'Name': 'James', 'School': 'Business', 'Location': '1024 Billiard Avenue'},
                            {'Name': 'Mike', 'School': 'Law', 'Location': 'Fraternity House #22'},
                            {'Name': 'Sally', 'School': 'Engineering', 'Location': '512 Wilson Crescent'}])
pd.merge(staff_df, student_df, how='left', left_on='Name', right_on='Name')
```

```
In [ ]: staff_df = pd.DataFrame([{'First Name': 'Kelly', 'Last Name': 'Desjardins', 'Role': 'Director of HR'},
                                {'First Name': 'Sally', 'Last Name': 'Brooks', 'Role': 'Course liaison'},
                                {'First Name': 'James', 'Last Name': 'Wilde', 'Role': 'Grader'}])
student_df = pd.DataFrame([{'First Name': 'James', 'Last Name': 'Hammond', 'School': 'Business'},
                            {'First Name': 'Mike', 'Last Name': 'Smith', 'School': 'Law'},
                            {'First Name': 'Sally', 'Last Name': 'Brooks', 'School': 'Engineering'}])
staff_df
student_df
pd.merge(staff_df, student_df, how='inner', left_on=['First Name', 'Last Name'], right_on=['First Name', 'Last Name'])
```

Idiomatic Pandas: Making Code Pandorable

```
In [ ]: import pandas as pd
df = pd.read_csv('census.csv')
df
```

```
In [ ]: (df.where(df['SUMLEV']==50)
        .dropna()
        .set_index(['STNAME', 'CTYNAME'])
        .rename(columns={'ESTIMATESBASE2010': 'Estimates Base 2010'}))
```

```
In [ ]: df = df[df['SUMLEV']==50]
df.set_index(['STNAME', 'CTYNAME'], inplace=True)
df.rename(columns={'ESTIMATESBASE2010': 'Estimates Base 2010'})
```

```
In [ ]: import numpy as np
def min_max(row):
    data = row[['POPESTIMATE2010',
                'POPESTIMATE2011',
                'POPESTIMATE2012',
                'POPESTIMATE2013',
                'POPESTIMATE2014',
                'POPESTIMATE2015']]
    return pd.Series({'min': np.min(data), 'max': np.max(data)})
```

```
In [ ]: df.apply(min_max, axis=1)
```

```
In [ ]: import numpy as np
def min_max(row):
    data = row[['POPESTIMATE2010',
                'POPESTIMATE2011',
                'POPESTIMATE2012',
                'POPESTIMATE2013',
                'POPESTIMATE2014',
                'POPESTIMATE2015']]
    row['max'] = np.max(data)
    row['min'] = np.min(data)
    return row
df.apply(min_max, axis=1)
```

```
In [ ]: rows = ['POPESTIMATE2010',
                'POPESTIMATE2011',
                'POPESTIMATE2012',
                'POPESTIMATE2013',
                'POPESTIMATE2014',
                'POPESTIMATE2015']
df.apply(lambda x: np.max(x[rows]), axis=1)
```

Group by

```
In [ ]: import pandas as pd
import numpy as np
df = pd.read_csv('census.csv')
df = df[df['SUMLEV']==50]
df
```

```
In [ ]: %%timeit -n 10
for state in df['STNAME'].unique():
    avg = np.average(df.where(df['STNAME']==state).dropna()['CENSUS2010POP'])
    print('Counties in state ' + state + ' have an average population of ' + s
tr(avg))
```

```
In [ ]: %%timeit -n 10
for group, frame in df.groupby('STNAME'):
    avg = np.average(frame['CENSUS2010POP'])
    print('Counties in state ' + group + ' have an average population of ' + s
tr(avg))
```

```
In [ ]: df.head()
```

```
In [ ]: df = df.set_index('STNAME')

def fun(item):
    if item[0]<'M':
        return 0
    if item[0]<'Q':
        return 1
    return 2

for group, frame in df.groupby(fun):
    print('There are ' + str(len(frame)) + ' records in group ' + str(group) +
        ' for processing.')
```

```
In [ ]: df = pd.read_csv('census.csv')
df = df[df['SUMLEV']==50]
```

```
In [ ]: df.groupby('STNAME').agg({'CENSUS2010POP': np.average})
```

```
In [ ]: print(type(df.groupby(level=0)['POPESTIMATE2010','POPESTIMATE2011']))
print(type(df.groupby(level=0)['POPESTIMATE2010']))
```

```
In [ ]: (df.set_index('STNAME').groupby(level=0)['CENSUS2010POP']
        .agg({'avg': np.average, 'sum': np.sum}))
```

```
In [ ]: (df.set_index('STNAME').groupby(level=0)['POPESTIMATE2010','POPESTIMATE2011']
        .agg({'avg': np.average, 'sum': np.sum}))
```

```
In [ ]: (df.set_index('STNAME').groupby(level=0)['POPESTIMATE2010','POPESTIMATE2011']
        .agg({'POPESTIMATE2010': np.average, 'POPESTIMATE2011': np.sum}))
```

Scales

```
In [ ]: df = pd.DataFrame(['A+', 'A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-', 'D+',
    'D'],
    index=['excellent', 'excellent', 'excellent', 'good', 'good',
    'good', 'ok', 'ok', 'ok', 'poor', 'poor'])
df.rename(columns={0: 'Grades'}, inplace=True)
df
```

```
In [ ]: df['Grades'].astype('category').head()
```

```
In [ ]: grades = df['Grades'].astype('category',
    categories=['D', 'D+', 'C-', 'C', 'C+', 'B-', 'B',
    'B+', 'A-', 'A', 'A+'],
    ordered=True)
grades.head()
```

```
In [ ]: grades > 'C'
```

```
In [ ]: df = pd.read_csv('census.csv')
df = df[df['SUMLEV']==50]
df = df.set_index('STNAME').groupby(level=0)['CENSUS2010POP'].agg({'avg': np.average})
pd.cut(df['avg'],10)
```

Pivot Tables

```
In [ ]: #http://open.canada.ca/data/en/dataset/98f1a129-f628-4ce4-b24d-6f16bf24dd64
df = pd.read_csv('cars.csv')
```

```
In [ ]: df.head()
```

```
In [ ]: df.pivot_table(values='(kW)', index='YEAR', columns='Make', aggfunc=np.mean)
```

```
In [ ]: df.pivot_table(values='(kW)', index='YEAR', columns='Make', aggfunc=[np.mean, np.min], margins=True)
```

Date Functionality in Pandas

```
In [2]: import pandas as pd
import numpy as np
```

Timestamp

```
In [3]: pd.Timestamp('9/1/2016 10:05AM')
```

```
Out[3]: Timestamp('2016-09-01 10:05:00')
```

Period

```
In [4]: pd.Period('1/2016')
```

```
Out[4]: Period('2016-01', 'M')
```

```
In [5]: pd.Period('3/5/2016')
```

```
Out[5]: Period('2016-03-05', 'D')
```

DatetimeIndex

```
In [6]: t1 = pd.Series(list('abc'), [pd.Timestamp('2016-09-01'), pd.Timestamp('2016-09-02'), pd.Timestamp('2016-09-03')])
t1
```

```
Out[6]: 2016-09-01    a
        2016-09-02    b
        2016-09-03    c
        dtype: object
```

```
In [7]: type(t1.index)
```

```
Out[7]: pandas.tseries.index.DatetimeIndex
```

PeriodIndex

```
In [8]: t2 = pd.Series(list('def'), [pd.Period('2016-09'), pd.Period('2016-10'), pd.Period('2016-11')])
t2
```

```
Out[8]: 2016-09    d
        2016-10    e
        2016-11    f
        Freq: M, dtype: object
```

```
In [9]: type(t2.index)
```

```
Out[9]: pandas.tseries.period.PeriodIndex
```

Converting to Datetime

```
In [10]: d1 = ['2 June 2013', 'Aug 29, 2014', '2015-06-26', '7/12/16']
ts3 = pd.DataFrame(np.random.randint(10, 100, (4,2)), index=d1, columns=list('ab'))
ts3
```

```
Out[10]:
```

	a	b
2 June 2013	16	46
Aug 29, 2014	14	66
2015-06-26	59	99
7/12/16	27	17

```
In [11]: ts3.index = pd.to_datetime(ts3.index)
         ts3
```

```
Out[11]:
```

	a	b
2013-06-02	16	46
2014-08-29	14	66
2015-06-26	59	99
2016-07-12	27	17

```
In [12]: pd.to_datetime('4.7.12', dayfirst=True)
```

```
Out[12]: Timestamp('2012-07-04 00:00:00')
```

Timedeltas

```
In [13]: pd.Timestamp('9/3/2016')-pd.Timestamp('9/1/2016')
```

```
Out[13]: Timedelta('2 days 00:00:00')
```

```
In [14]: pd.Timestamp('9/2/2016 8:10AM') + pd.Timedelta('12D 3H')
```

```
Out[14]: Timestamp('2016-09-14 11:10:00')
```

Working with Dates in a Dataframe

```
In [15]: dates = pd.date_range('10-01-2016', periods=9, freq='2W-SUN')
         dates
```

```
Out[15]: DatetimeIndex(['2016-10-02', '2016-10-16', '2016-10-30', '2016-11-13',
                        '2016-11-27', '2016-12-11', '2016-12-25', '2017-01-08',
                        '2017-01-22'],
                        dtype='datetime64[ns]', freq='2W-SUN')
```



```
In [16]: df = pd.DataFrame({'Count 1': 100 + np.random.randint(-5, 10, 9).cumsum(),
                           'Count 2': 120 + np.random.randint(-5, 10, 9)}, index=dates)
df
```

```
Out[16]:
```

	Count 1	Count 2
2016-10-02	104	125
2016-10-16	109	122
2016-10-30	111	127
2016-11-13	117	126
2016-11-27	114	126
2016-12-11	109	121
2016-12-25	105	126
2017-01-08	105	125
2017-01-22	101	123

```
In [17]: df.index.weekday_name
```

```
Out[17]: array(['Sunday', 'Sunday', 'Sunday', 'Sunday', 'Sunday', 'Sunday',
                'Sunday', 'Sunday', 'Sunday'], dtype=object)
```

```
In [18]: df.diff()
```

```
Out[18]:
```

	Count 1	Count 2
2016-10-02	NaN	NaN
2016-10-16	5.0	-3.0
2016-10-30	2.0	5.0
2016-11-13	6.0	-1.0
2016-11-27	-3.0	0.0
2016-12-11	-5.0	-5.0
2016-12-25	-4.0	5.0
2017-01-08	0.0	-1.0
2017-01-22	-4.0	-2.0

```
In [19]: df.resample('M').mean()
```

Out[19]:

	Count 1	Count 2
2016-10-31	108.0	124.666667
2016-11-30	115.5	126.000000
2016-12-31	107.0	123.500000
2017-01-31	103.0	124.000000

```
In [20]: df['2017']
```

Out[20]:

	Count 1	Count 2
2017-01-08	105	125
2017-01-22	101	123

```
In [21]: df['2016-12']
```

Out[21]:

	Count 1	Count 2
2016-12-11	109	121
2016-12-25	105	126

```
In [22]: df['2016-12':]
```

Out[22]:

	Count 1	Count 2
2016-12-11	109	121
2016-12-25	105	126
2017-01-08	105	125
2017-01-22	101	123

```
In [ ]: df.asfreq('W', method='ffill')
```

```
In [ ]: import matplotlib.pyplot as plt
%matplotlib inline

df.plot()
```