Darya Filipchyk Class Practice CIS3120

In [1]:
```
from pandas import DataFrame
import numpy as np
```

In [2]:
```
df1 = DataFrame({'student': ['Jerry', 'Jerry', 'Jerry', 'Mary', 'Mary'
,
                             'Mary'],
                 'grade': [100, 90, 92, 88, 80, 86]},
                columns=['student', 'grade'])
df1
```

Out[2]:

	student	grade
0	Jerry	100
1	Jerry	90
2	Jerry	92
3	Mary	88
4	Mary	80
5	Mary	86

Find a mean grade for each student

The data is aggregated according to the group by key, producing a new Series that is now indexed by the unique values in the *student* column.

In [3]:
```
grouped = df1['grade'].groupby(df1['student'])
grouped.describe()
```

Out[3]:

	count	mean	std	min	25%	50%	75%	max
student								
Jerry	3.0	94.000000	5.291503	90.0	91.0	92.0	96.0	100.0
Mary	3.0	84.666667	4.163332	80.0	83.0	86.0	87.0	88.0

In [4]:
```
grouped.mean()
```

Out[4]:
```
student
Jerry    94.000000
Mary     84.666667
Name: grade, dtype: float64
```

Grouping by multiple keys.

Pass a list of columns to *groupby()*

In [5]:
```
df = DataFrame({'student': ['Jerry', 'Jerry', 'Jerry', 'Mary', 'Mary',
'Mary'],
                'course': ['CS30', 'CS30', 'CS32', 'CS30', 'CS32', 'CS
32'],
                'grade': [100, 90, 92, 88, 80, 86]},
                columns=['student', 'course', 'grade'])
df
```

Out[5]:

	student	course	grade
0	Jerry	CS30	100
1	Jerry	CS30	90
2	Jerry	CS32	92
3	Mary	CS30	88
4	Mary	CS32	80
5	Mary	CS32	86

In [6]:
```
grouped = df['grade'].groupby([df['student'], df['course']])
means = grouped.mean()

means
```

Out[6]:
```
student  course
Jerry    CS30      95
         CS32      92
Mary     CS30      88
         CS32      83
Name: grade, dtype: int64
```

The resulting series has a hierarchical index consisting of the unique pairs of keys.

In [7]:
```
means.unstack()
```

Out[7]:

course	CS30	CS32
student		
Jerry	95	92
Mary	88	83

Similar example

In [8]:
```
grouped = df['grade'].groupby([df['course'], df['student']])
means = grouped.mean()
means
```

Out[8]:
```
course  student
CS30    Jerry      95
        Mary       88
CS32    Jerry      92
        Mary       83
Name: grade, dtype: int64
```

In [9]:
```
means.unstack()
```

Out[9]:

student	Jerry	Mary
course		
CS30	95	88
CS32	92	83

Group key can be any array of the right length

This array does not have to be from the same DataFrame.

In [10]:
```
df
```

Out[10]:

	student	course	grade
0	Jerry	CS30	100
1	Jerry	CS30	90
2	Jerry	CS32	92
3	Mary	CS30	88
4	Mary	CS32	80
5	Mary	CS32	86

In [11]:
```
gr = df['grade'].groupby(['M', 'M', 'M', 'F', 'F', 'F'])
gr.mean()
```

Out[11]:
```
F    84.666667
M    94.000000
Name: grade, dtype: float64
```

Group keys can be column names.

Provided groups keys are in the same DataFrame as the data you are working with.

In [12]:
```
df
```

Out[12]:

	student	course	grade
0	Jerry	CS30	100
1	Jerry	CS30	90
2	Jerry	CS32	92
3	Mary	CS30	88
4	Mary	CS32	80
5	Mary	CS32	86

In [13]:
```
df.groupby('student').mean()
```

Out[13]:

	grade
student	
Jerry	94.000000
Mary	84.666667

A **nuisance** column is excluded from the result. For this example, *course* is a nuisance column because it's not numeric.

Group key column names

In [14]:
```
df.groupby(['course', 'student']).mean()
```

Out[14]:

		grade
course	**student**	
CS30	**Jerry**	95
	Mary	88
CS32	**Jerry**	92
	Mary	83

Group by method *size()* returns a Series containing group sizes:

In [15]:
```
df
```

Out[15]:

	student	course	grade
0	Jerry	CS30	100
1	Jerry	CS30	90
2	Jerry	CS32	92
3	Mary	CS30	88
4	Mary	CS32	80
5	Mary	CS32	86

In [16]:
```
df.groupby('student').size()
```

Out[16]:
```
student
Jerry    3
Mary     3
dtype: int64
```

Titanic

Seaborn library contains the database of passengers on the Titanic.

In [17]:
```
import numpy as np
import pandas as pd
import seaborn as sns
titanic = sns.load_dataset?
```

In [18]:
```
titanic = sns.load_dataset('titanic')
```

In [19]:
```
titanic.head()
```

Out[19]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True

Questions:

1. How many men survived vs how many women survived?
2. How many passengers survived per sex and class

How many men survived vs how many women survived.

In [20]:
```
grouped = titanic['survived'].groupby(titanic['sex'])
grouped.mean()
```

Out[20]:
```
sex
female    0.742038
male      0.188908
Name: survived, dtype: float64
```

Conclusion

Three of every four females on board survived.

One in five males survived.

How many passengers survived per sex and class.

In [21]:
```
grouped = titanic['survived'].groupby([titanic['sex'], titanic['class']])
m = grouped.mean()
m
```

Out[21]:
```
sex     class
female  First     0.968085
        Second    0.921053
        Third     0.500000
male    First     0.368852
        Second    0.157407
        Third     0.135447
Name: survived, dtype: float64
```

In [22]:
```
m.unstack()
```

Out[22]:

class	First	Second	Third
sex			
female	0.968085	0.921053	0.500000
male	0.368852	0.157407	0.135447

How many passengers survived per class and sex.

In [23]:
```
gr = titanic['survived'].groupby([titanic['class'], titanic['sex']])
me = gr.mean()
me
```

Out[23]:
```
class   sex
First   female    0.968085
        male      0.368852
Second  female    0.921053
        male      0.157407
Third   female    0.500000
        male      0.135447
Name: survived, dtype: float64
```

In [24]:
```
me.unstack()
```

Out[24]:

sex	female	male
class		
First	0.968085	0.368852
Second	0.921053	0.157407
Third	0.500000	0.135447

Pivot Table

In [25]:
```
titanic.pivot_table('survived', index='sex', columns='class')
```

Out[25]:

class	First	Second	Third
sex			
female	0.968085	0.921053	0.500000
male	0.368852	0.157407	0.135447

Conclusion:

first-class women survived with near certainty.

Pivot Table II

More than one aggregate

In [26]:
```
titanic.pivot_table(index='sex', columns='class',
                    aggfunc={'survived': 'sum', 'fare': 'mean'})
```

Out[26]:

	fare			survived		
class	First	Second	Third	First	Second	Third
sex						
female	106.125798	21.970121	16.118810	91	70	72
male	67.226127	19.741782	12.661633	45	17	47