Pandas - pivot_table, crosstab

In [1]:
```
import pandas as pd
import numpy  as np
import seaborn as sns

pd.set_option('precision', 4)
```

In [2]:
```
df = pd.DataFrame(
    {"A": ["Alice", "Alice", "Alice", "Alice", "Alice",
           "Bob", "Bob", "Bob", "Bob"],
     "B": ["one", "one", "one", "two", "two",
           "one", "one", "two", "two"],
     "C": ["small", "large", "large", "small",
           "small", "large", "small", "small", "large"],
     "D": [1, 2, 2, 3, 3, 4, 5, 6, 7],
     "E": [10,20,30,40,50,60,70,80,90]
     })

df
```

Out[2]:

	A	B	C	D	E
0	Alice	one	small	1	10
1	Alice	one	large	2	20
2	Alice	one	large	2	30
3	Alice	two	small	3	40
4	Alice	two	small	3	50
5	Bob	one	large	4	60
6	Bob	one	small	5	70
7	Bob	two	small	6	80
8	Bob	two	large	7	90

- index - Keys to group by on the pivot table index
- default aggregation function - group means

In [3]:
```
df.pivot_table(index=['A'])
```

Out[3]:

A	D	E
Alice	2.2	30
Bob	5.5	75

In [4]:
```
df.pivot_table(index=['A'], aggfunc=np.sum)
```

Out[4]:

A	D	E
Alice	11	150
Bob	22	300

In [5]:
```
df.pivot_table(index=['A'], aggfunc=len)
```

Out[5]:

A	B	C	D	E
Alice	5	5	5	5
Bob	4	4	4	4

- values- specify column(s) to aggregate using values

In [6]:
```
df.pivot_table(index=['A'], values='D',
               aggfunc=np.sum)
```

Out[6]:

A	D
Alice	11
Bob	22

In [7]:
```
df.pivot_table(index=['A'], values=['D','E'],
               aggfunc=np.sum)
```

Out[7]:

A	D	E
Alice	11	150
Bob	22	300

In [8]:
```
df.dtypes
```

Out[8]:
```
A     object
B     object
C     object
D      int64
E      int64
dtype: object
```

In [ ]:

In [9]:
```
df
```

Out[9]:

	A	B	C	D	E
0	Alice	one	small	1	10
1	Alice	one	large	2	20
2	Alice	one	large	2	30
3	Alice	two	small	3	40
4	Alice	two	small	3	50
5	Bob	one	large	4	60
6	Bob	one	small	5	70
7	Bob	two	small	6	80
8	Bob	two	large	7	90

- columns - Specify keys to group by on the pivot table column

In [10]:
```
df.pivot_table(index=['A'], columns=['C'],
               aggfunc=np.sum)
```

Out[10]:

	D		E	
C	**large**	**small**	**large**	**small**
A				
Alice	4	7	50	100
Bob	11	11	150	150

df.pivot_table(index=['A'], values='D', columns=['C'], aggfunc=np.sum)

- If dict is passed for aggfunc, the key is column to aggregate and value is function or list of functions

In [11]:
```
df.pivot_table(index=['A'], columns=['C'],
               aggfunc={'D': 'sum', 'E': 'mean'})
```

Out[11]:

	D		E	
C	large	small	large	small
A				
Alice	4	7	25.0	33.3333
Bob	11	11	75.0	75.0000

In [12]:
```
df.pivot_table(index=['A'], columns=['C'],
               aggfunc={'D': ['sum', np.mean],
                        'E': ['count', len]})
```

Out[12]:

	D				E			
	mean		sum		count		len	
C	large	small	large	small	large	small	large	small
A								
Alice	2.0	2.3333	4.0	7.0	2	3	2	3
Bob	5.5	5.5000	11.0	11.0	2	2	2	2

multi-index

In [13]:
```
df
```

Out[13]:

	A	B	C	D	E
0	Alice	one	small	1	10
1	Alice	one	large	2	20
2	Alice	one	large	2	30
3	Alice	two	small	3	40
4	Alice	two	small	3	50
5	Bob	one	large	4	60
6	Bob	one	small	5	70
7	Bob	two	small	6	80
8	Bob	two	large	7	90

- array of index values

In [14]:
```
df.pivot_table(index=['A', 'B'])
```

Out[14]:

		D	E
A	B		
Alice	one	1.6667	20
	two	3.0000	45
Bob	one	4.5000	65
	two	6.5000	85

In [15]:
```
df.pivot_table(index=['A', 'B'], aggfunc=np.sum)
```

Out[15]:

A	B	D	E
Alice	one	5	60
	two	6	90
Bob	one	9	130
	two	13	170

In [16]:
```
df.pivot_table(index=['A', 'B'], columns = ['C'],
               aggfunc=np.sum)
```

Out[16]:

		D		E	
	C	large	small	large	small
A	B				
Alice	one	4.0	1.0	50.0	10.0
	two	NaN	6.0	NaN	90.0
Bob	one	4.0	5.0	60.0	70.0
	two	7.0	6.0	90.0	80.0

In [17]:
```
df.pivot_table(index=['A', 'B'], columns = ['C'],
               aggfunc=np.sum).index
```

Out[17]:
```
MultiIndex([('Alice', 'one'),
            ('Alice', 'two'),
            (  'Bob', 'one'),
            (  'Bob', 'two')],
           names=['A', 'B'])
```

In [18]:
```
df.pivot_table(index=['A', 'B'], columns = ['C'],
               aggfunc=np.sum).columns
```

Out[18]:
```
MultiIndex([('D', 'large'),
            ('D', 'small'),
            ('E', 'large'),
            ('E', 'small')],
           names=[None, 'C'])
```

In [ ]:

In [19]:
```
df.pivot_table(index=['A', 'B'], values=['D'],
               columns=['C'], aggfunc=np.sum)
```

Out[19]:

		D	
	C	large	small
A	B		
Alice	one	4.0	1.0
	two	NaN	6.0
Bob	one	4.0	5.0
	two	7.0	6.0

Margins

In [20]:
```
df.pivot_table(index=['A', 'B'], values=['D'],
               columns=['C'],
               aggfunc=np.sum, margins=True)
```

Out[20]:

		D		
	C	large	small	All
A	B			
Alice	one	4.0	1.0	5
	two	NaN	6.0	6
Bob	one	4.0	5.0	9
	two	7.0	6.0	13
All		15.0	18.0	33

In [21]:
```
df.pivot_table(index=['A', 'B'], values=['D'],
               columns=['C'],
               aggfunc=len, margins=True,
               fill_value=0)
```

Out[21]:

		D		
	C	large	small	All
A	B			
Alice	one	2	1	3
	two	0	2	2
Bob	one	1	1	2
	two	1	1	2
All		4	5	9

In [ ]:
```
```

Tips dataset

In [22]:
```
tips = sns.load_dataset("tips")
tips.head()
```

Out[22]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [23]:
```
# group means - default pivot_table aggregation type

tips.pivot_table(index=['day', 'sex'])
```

Out[23]:

day	sex	size	tip	total_bill
Thur	Male	2.4333	2.9803	18.7147
	Female	2.4688	2.5756	16.7153
Fri	Male	2.1000	2.6930	19.8570
	Female	2.1111	2.7811	14.1456
Sat	Male	2.6441	3.0839	20.8025
	Female	2.2500	2.8018	19.6804
Sun	Male	2.8103	3.2203	21.8872
	Female	2.9444	3.3672	19.8722

In [24]:
```
tips.pivot_table(index=['day', 'sex'],
                 margins = True)
```

Out[24]:

day	sex	size	tip	total_bill
Thur	Male	2.4333	2.9803	18.7147
	Female	2.4688	2.5756	16.7153
Fri	Male	2.1000	2.6930	19.8570
	Female	2.1111	2.7811	14.1456
Sat	Male	2.6441	3.0839	20.8025
	Female	2.2500	2.8018	19.6804
Sun	Male	2.8103	3.2203	21.8872
	Female	2.9444	3.3672	19.8722
All		2.5697	2.9983	19.7859

In [25]:
```
tips.pivot_table(index=['day', 'sex'], values='total_bill',
                 margins = True)
```

Out[25]:

		total_bill
day	**sex**	
Thur	Male	18.7147
	Female	16.7153
Fri	Male	19.8570
	Female	14.1456
Sat	Male	20.8025
	Female	19.6804
Sun	Male	21.8872
	Female	19.8722
All		19.7859

In [26]:
```
tips['total_bill'].mean()
```

Out[26]:
```
19.78594262295082
```

In [ ]:
```

```

In [27]:
```
tips.pivot_table(index=['day', 'sex'], aggfunc = np.sum)
```

Out[27]:

day	sex	size	tip	total_bill
Thur	Male	73	89.41	561.44
	Female	79	82.42	534.89
Fri	Male	21	26.93	198.57
	Female	19	25.03	127.31
Sat	Male	156	181.95	1227.35
	Female	63	78.45	551.05
Sun	Male	163	186.78	1269.46
	Female	53	60.61	357.70

In [28]:
```
tips.pivot_table(index=['day', 'sex'], aggfunc = np.sum,
                 margins=True)
```

Out[28]:

day	sex	size	tip	total_bill
Thur	Male	73	89.41	561.44
	Female	79	82.42	534.89
Fri	Male	21	26.93	198.57
	Female	19	25.03	127.31
Sat	Male	156	181.95	1227.35
	Female	63	78.45	551.05
Sun	Male	163	186.78	1269.46
	Female	53	60.61	357.70
All		627	731.58	4827.77

In [ ]:

crosstab(...)

- Cross tabulation of two or more factors
- Default - frequency table

In [29]:
```
df
```

Out[29]:

	A	B	C	D	E
0	Alice	one	small	1	10
1	Alice	one	large	2	20
2	Alice	one	large	2	30
3	Alice	two	small	3	40
4	Alice	two	small	3	50
5	Bob	one	large	4	60
6	Bob	one	small	5	70
7	Bob	two	small	6	80
8	Bob	two	large	7	90

In [30]:
```
pd.crosstab(df.A, df.C )
```

Out[30]:

C	large	small
A		
Alice	2	3
Bob	2	2

- normalize - for percentages rather than counts
- if passed ‘all’ or True, will normalize over all values

In [31]:
```
pd.crosstab(df.A, df.C, normalize=True)
```

Out[31]:

A	large	small
Alice	0.2222	0.3333
Bob	0.2222	0.2222

In [32]:
```
# normalize over each row

pd.crosstab(df.A, df.C, normalize='index')
```

Out[32]:

A	large	small
Alice	0.4	0.6
Bob	0.5	0.5

In [33]:
```
# normalize over each column

pd.crosstab(df.A, df.C, normalize='columns')
```

Out[33]:

A	large	small
Alice	0.5	0.6
Bob	0.5	0.4

In [ ]:
```

```

In [34]:
```
pd.crosstab(df.A, df.B)
```

Out[34]:

A	one	two
Alice	3	2
Bob	2	2

In [35]:
```
# With and third series and an aggregation function

pd.crosstab(df.A, df.B, values=df.E, aggfunc=np.sum)
```

Out[35]:

A	B	one	two
Alice		60	90
Bob		130	170

Tips dataset

In [36]:
```
tips.head()
```

Out[36]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [37]:
```
pd.crosstab(tips.day, tips.time)
```

Out[37]:

day	time	Lunch	Dinner
Thur		61	1
Fri		7	12
Sat		0	87
Sun		0	76

In [38]:
```
# equivalent groupby

tips.groupby(['day', 'time'])['day'].count().unstack().fillna(0)
```

Out[38]:

time	Lunch	Dinner
day		
Thur	61.0	1.0
Fri	7.0	12.0
Sat	0.0	87.0
Sun	0.0	76.0

In [39]:
```
# Without unstack

tips.groupby(['day', 'time'])['day'].count().fillna(0)
```

Out[39]:
```
day   time
Thur  Lunch     61
      Dinner     1
Fri   Lunch      7
      Dinner    12
Sat   Dinner    87
Sun   Dinner    76
Name: day, dtype: int64
```

In [40]:
```
# Same as

tips.groupby(['day', 'time'])['time'].count().unstack().fillna(0)
```

Out[40]:

time	Lunch	Dinner
day		
Thur	61.0	1.0
Fri	7.0	12.0
Sat	0.0	87.0
Sun	0.0	76.0

In [41]:
```
# equivalent pivot_table

tips.pivot_table(index='day', columns='time',
                 aggfunc={'time':len}, fill_value=0)
```

Out[41]:

	time	
time	Lunch	Dinner
day		
Thur	61	1
Fri	7	12
Sat	0	87
Sun	0	76

In [42]:
```
# Margin totals

pd.crosstab(tips.day, tips.time, margins=True,
            margins_name="Total")
```

Out[42]:

time	Lunch	Dinner	Total
day			
Thur	61	1	62
Fri	7	12	19
Sat	0	87	87
Sun	0	76	76
Total	68	176	244

In [43]:
```
# Summarization with crosstab

pd.crosstab(tips.day, tips.time,
            values=tips.tip,
            aggfunc=np.sum)
```

Out[43]:

time	Lunch	Dinner
day		
Thur	168.83	3.00
Fri	16.68	35.28
Sat	NaN	260.40
Sun	NaN	247.39

In [44]:
```
# Margin totals with values and aggfunc

pd.crosstab(tips.day, tips.time,
            values = tips.total_bill,
            aggfunc = np.sum,
            margins=True, margins_name="Total")
```

Out[44]:

time	Lunch	Dinner	Total
day			
Thur	1077.55	18.78	1096.33
Fri	89.92	235.96	325.88
Sat	NaN	1778.40	1778.40
Sun	NaN	1627.16	1627.16
Total	1167.47	3660.30	4827.77

In [ ]: