In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```python
# pd.set_option('display.max_rows', None)
```

In [3]:
```python
tips = pd.read_csv('tips.csv')
```

In [4]:
```python
tips['tip_pct'] = tips['tip'] / tips['total_bill']
```

In [5]:
```python
tips
```

Out[5]:

	total_bill	tip	sex	smoker	day	time	size	tip_pct
0	16.99	1.01	Female	No	Sun	Dinner	2	0.059447
1	10.34	1.66	Male	No	Sun	Dinner	3	0.160542
2	21.01	3.50	Male	No	Sun	Dinner	3	0.166587
3	23.68	3.31	Male	No	Sun	Dinner	2	0.139780
4	24.59	3.61	Female	No	Sun	Dinner	4	0.146808
...	...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3	0.203927
240	27.18	2.00	Female	Yes	Sat	Dinner	2	0.073584
241	22.67	2.00	Male	Yes	Sat	Dinner	2	0.088222
242	17.82	1.75	Male	No	Sat	Dinner	2	0.098204
243	18.78	3.00	Female	No	Thur	Dinner	2	0.159744

244 rows × 8 columns

In [6]:
```python
df = tips.groupby(['day','smoker'])[['size','tip','tip_pct','total_bill']].mean()
df
```

Out[6]:

		size	tip	tip_pct	total_bill
day	**smoker**				
Fri	**No**	2.250000	2.812500	0.151650	18.420000
	Yes	2.066667	2.714000	0.174783	16.813333
Sat	**No**	2.555556	3.102889	0.158048	19.661778
	Yes	2.476190	2.875476	0.147906	21.276667
Sun	**No**	2.929825	3.167895	0.160113	20.506667
	Yes	2.578947	3.516842	0.187250	24.120000
Thur	**No**	2.488889	2.673778	0.160298	17.113111
	Yes	2.352941	3.030000	0.163863	19.190588

suppose you wanted to compute a table of group means (the default pivot_table aggregation type) arranged by day and smoker on the rows:

In [7]:
```python
def top(df, n=5, column='tip_pct'):
    return df.sort_values(by=column)[-n:]
top(tips, n=10)
```

Out[7]:

	total_bill	tip	sex	smoker	day	time	size	tip_pct
51	10.29	2.60	Female	No	Sun	Dinner	2	0.252672
221	13.42	3.48	Female	Yes	Fri	Lunch	2	0.259314
93	16.32	4.30	Female	Yes	Fri	Dinner	2	0.263480
149	7.51	2.00	Male	No	Thur	Lunch	2	0.266312
109	14.31	4.00	Female	Yes	Sat	Dinner	2	0.279525
183	23.17	6.50	Male	Yes	Sun	Dinner	4	0.280535
232	11.61	3.39	Male	No	Sat	Dinner	2	0.291990
67	3.07	1.00	Female	Yes	Sat	Dinner	1	0.325733
178	9.60	4.00	Female	Yes	Sun	Dinner	2	0.416667
172	7.25	5.15	Male	Yes	Sun	Dinner	2	0.710345

In [8]:
```python
# tips.drop(['sex'], axis=1,inplace=True)
```

In [9]:
```python
tips.pivot_table(['total_bill', 'tip',  'size', 'tip_pct'],index=['day', 'smoker'])
```

Out[9]:

day	smoker	size	tip	tip_pct	total_bill
Fri	No	2.250000	2.812500	0.151650	18.420000
	Yes	2.066667	2.714000	0.174783	16.813333
Sat	No	2.555556	3.102889	0.158048	19.661778
	Yes	2.476190	2.875476	0.147906	21.276667
Sun	No	2.929825	3.167895	0.160113	20.506667
	Yes	2.578947	3.516842	0.187250	24.120000
Thur	No	2.488889	2.673778	0.160298	17.113111
	Yes	2.352941	3.030000	0.163863	19.190588

In [10]:
```python
tips.pivot_table(['total_bill', 'tip',  'size', 'tip_pct'],index=['day', 'smoker'],aggfunc='sum')
```

Out[10]:

day	smoker	size	tip	tip_pct	total_bill
Fri	No	9	11.25	0.606602	73.68
	Yes	31	40.71	2.621746	252.20
Sat	No	115	139.63	7.112145	884.78
	Yes	104	120.77	6.212055	893.62
Sun	No	167	180.57	9.126438	1168.88
	Yes	49	66.82	3.557756	458.28
Thur	No	112	120.32	7.213414	770.09
	Yes	40	51.51	2.785676	326.24

In [11]:
```python
tips.pivot_table(['tip_pct', 'size'], index=['time', 'day'],columns='smoker')
```

Out[11]:

		size		tip_pct	
	smoker	No	Yes	No	Yes
time	day				
Dinner	Fri	2.000000	2.222222	0.139622	0.165347
	Sat	2.555556	2.476190	0.158048	0.147906
	Sun	2.929825	2.578947	0.160113	0.187250
	Thur	2.000000	NaN	0.159744	NaN
Lunch	Fri	3.000000	1.833333	0.187735	0.188937
	Thur	2.500000	2.352941	0.160311	0.163863

We could augment this table to include partial totals by passing margins=True. This has the effect of adding All row and column labels, with corresponding values being the group statistics for all the data within a single tier:

In [12]:
```python
tips.pivot_table(['tip_pct', 'size'], index=['time', 'day'],columns='smoker', margins=True)
```

Out[12]:

		size			tip_pct		
	smoker	No	Yes	All	No	Yes	All
time	day						
Dinner	Fri	2.000000	2.222222	2.166667	0.139622	0.165347	0.158916
	Sat	2.555556	2.476190	2.517241	0.158048	0.147906	0.153152
	Sun	2.929825	2.578947	2.842105	0.160113	0.187250	0.166897
	Thur	2.000000	NaN	2.000000	0.159744	NaN	0.159744
Lunch	Fri	3.000000	1.833333	2.000000	0.187735	0.188937	0.188765
	Thur	2.500000	2.352941	2.459016	0.160311	0.163863	0.161301
All		2.668874	2.408602	2.569672	0.159328	0.163196	0.160803

To use a different aggregation function, pass it to aggfunc. For example, 'count' or len will give you a cross-tabulation (count or frequency) of group sizes:

In [13]:
```python
tips.pivot_table('tip_pct', index=['time', 'smoker'], columns='day', aggfunc=len, margins=True)
```

Out[13]:

	day	Fri	Sat	Sun	Thur	All
time	smoker					
Dinner	No	3.0	45.0	57.0	1.0	106
	Yes	9.0	42.0	19.0	NaN	70
Lunch	No	1.0	NaN	NaN	44.0	45
	Yes	6.0	NaN	NaN	17.0	23
All		19.0	87.0	76.0	62.0	244

If some combinations are empty (or otherwise NA), you may wish to pass a fill_value:

In [14]:
```
tips.pivot_table('tip_pct', index=['time', 'size', 'smoker'],columns='day', aggfunc='mean', fill_value=0)
```

Out[14]:

		day	Fri	Sat	Sun	Thur
time	**size**	**smoker**				
Dinner	**1**	**No**	0.000000	0.137931	0.000000	0.000000
		Yes	0.000000	0.325733	0.000000	0.000000
	2	**No**	0.139622	0.162705	0.168859	0.159744
		Yes	0.171297	0.148668	0.207893	0.000000
	3	**No**	0.000000	0.154661	0.152663	0.000000
		Yes	0.000000	0.144995	0.152660	0.000000
	4	**No**	0.000000	0.150096	0.148143	0.000000
		Yes	0.117750	0.124515	0.193370	0.000000
	5	**No**	0.000000	0.000000	0.206928	0.000000
		Yes	0.000000	0.106572	0.065660	0.000000
	6	**No**	0.000000	0.000000	0.103799	0.000000
Lunch	**1**	**No**	0.000000	0.000000	0.000000	0.181728
		Yes	0.223776	0.000000	0.000000	0.000000
	2	**No**	0.000000	0.000000	0.000000	0.166005
		Yes	0.181969	0.000000	0.000000	0.158843
	3	**No**	0.187735	0.000000	0.000000	0.084246
		Yes	0.000000	0.000000	0.000000	0.204952
	4	**No**	0.000000	0.000000	0.000000	0.138919
		Yes	0.000000	0.000000	0.000000	0.155410
	5	**No**	0.000000	0.000000	0.000000	0.121389
	6	**No**	0.000000	0.000000	0.000000	0.173706

Cross-Tabulations: Crosstab

A cross-tabulation (or crosstab for short) is a special case of a pivot table that computes group frequencies. Here is an example:

In [15]:
```
pd.crosstab([tips.time, tips.day], tips.smoker, margins=True)
```

Out[15]:

	smoker	No	Yes	All
time	**day**			
Dinner	**Fri**	3	9	12
	Sat	45	42	87
	Sun	57	19	76
	Thur	1	0	1
Lunch	**Fri**	1	6	7
	Thur	44	17	61
All		151	93	244

In [ ]:
```

```