In [23]:
```
import numpy as np
import pandas as pd
from pandas import Series, DataFrame
```

In [24]:
```
# Let's see how we would find outliers in a dataset

# First we'll seed the numpy generator
np.random.seed(12345)

#Next we'll create the dataframe
dframe = DataFrame(np.random.randn(1000,4))
```

In [25]:
```
#Show preview
dframe.head()
```

Out[25]:

	0	1	2	3
0	-0.204708	0.478943	-0.519439	-0.555730
1	1.965781	1.393406	0.092908	0.281746
2	0.769023	1.246435	1.007189	-1.296221
3	0.274992	0.228913	1.352917	0.886429
4	-2.001637	-0.371843	1.669025	-0.438570

In [26]:
```
# Lets describe the data
dframe.describe()
```

Out[26]:

	0	1	2	3
count	1000.000000	1000.000000	1000.000000	1000.000000
mean	-0.067684	0.067924	0.025598	-0.002298
std	0.998035	0.992106	1.006835	0.996794
min	-3.428254	-3.548824	-3.184377	-3.745356
25%	-0.774890	-0.591841	-0.641675	-0.644144
50%	-0.116401	0.101143	0.002073	-0.013611
75%	0.616366	0.780282	0.680391	0.654328
max	3.366626	2.653656	3.260383	3.927528

In [27]:
```
# Lets select the first column
col = dframe[0]
```

```
In [28]: # NOw we can check which values in the column are greater than 3, for instance.
         col[np.abs(col)>3]

Out[28]: 523   -3.428254
         900    3.366626
         Name: 0, dtype: float64
```

```
In [29]: # So we now know in column[0], rows 523 and 900 have values with abs > 3

         #How about all the columns?

         # We can use the "any" method
         dframe[(np.abs(dframe)>3).any(1)]
```

Out[29]:

	0	1	2	3
5	-0.539741	0.476985	3.248944	-1.021228
97	-0.774363	0.552936	0.106061	3.927528
102	-0.655054	-0.565230	3.176873	0.959533
305	-2.315555	0.457246	-0.025907	-3.399312
324	0.050188	1.951312	3.260383	0.963301
400	0.146326	0.508391	-0.196713	-3.745356
499	-0.293333	-0.242459	-3.056990	1.918403
523	-3.428254	-0.296336	-0.439938	-0.867165
586	0.275144	1.179227	-3.184377	1.369891
808	-0.362528	-3.548824	1.553205	-2.186301
900	3.366626	-2.372214	0.851010	1.332846

```
In [33]: # WE could also possibly cap the data at 3

         dframe[np.abs(dframe)>3] = np.sign(dframe) *3
```

```
In [34]: dframe.describe()
```

Out[34]:

	0	1	2	3
count	1000.000000	1000.000000	1000.000000	1000.000000
mean	-0.061623	0.074473	0.037153	0.009919
std	0.995875	0.989820	1.003604	0.989688
min	-2.969411	-2.989741	-2.925113	-2.881858
25%	-0.774132	-0.588138	-0.622310	-0.636641
50%	-0.115171	0.102787	0.012889	-0.010997
75%	0.619779	0.787953	0.682401	0.659019
max	3.000000	3.000000	3.000000	3.000000

```
In [ ]: # Next we'll learn about Permutation!
```