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

from numpy.random import randn
```

In [11]:
```
#Now we'll learn about Index Hierarchy

#pandas allows you to have multiple index levels, which is very clear with this e

ser = Series(np.random.randn(6),index=[[1,1,1,2,2,2],['a','b','c','a','b','c']])
```

In [12]:
```
#Show Series with multiple index levels
ser
```

Out[12]:
```
1  a   -1.337299
   b   -0.690616
   c    1.792962
2  a    0.457808
   b    0.891199
   c   -1.366387
dtype: float64
```

In [14]:
```
# We can check the multiple levels
ser.index
```

...

In [15]:
```
#Now we can sleect specific subsets
ser[1]
```

Out[15]:
```
a   -1.337299
b   -0.690616
c    1.792962
dtype: float64
```

In [16]:
```
# We can also select from an internal index level
ser[:,'a']
```

Out[16]:
```
1   -1.337299
2    0.457808
dtype: float64
```

In [19]:
```
# We can also create Data Frames from Series with multiple levels
dframe = ser.unstack()

#Show
dframe
```

Out[19]:

	a	b	c
1	-1.337299	-0.690616	1.792962
2	0.457808	0.891199	-1.366387

In [20]:
```
#Can also reverse
dframe.unstack()
```

Out[20]:
```
a  1   -1.337299
   2    0.457808
b  1   -0.690616
   2    0.891199
c  1    1.792962
   2   -1.366387
dtype: float64
```

In [28]:
```
# We can also apply multiple level indexing to DataFrames
dframe2 = DataFrame(np.arange(16).reshape(4,4),
                    index=[['a','a','b','b'],[1,2,1,2]],
                    columns=[['NY','NY','LA','SF'],['cold','hot','hot','cold']])

dframe2
```

Out[28]:

		NY		LA	SF
		cold	**hot**	**hot**	**cold**
a	**1**	0	1	2	3
	2	4	5	6	7
b	**1**	8	9	10	11
	2	12	13	14	15

In [31]:
```
# We can also give these index levels names

#Name the index levels
dframe2.index.names = ['INDEX_1','INDEX_2']

#Name the column levels
dframe2.columns.names = ['Cities','Temp']

dframe2
```

Out[31]:

	Cities	NY		LA	SF
	Temp	cold	hot	hot	cold
INDEX_1	INDEX_2				
a	1	0	1	2	3
	2	4	5	6	7
b	1	8	9	10	11
	2	12	13	14	15

In [33]:
```
# We can also interchange level orders (note the axis=1 for columns)
dframe2.swaplevel('Cities','Temp',axis=1)
```

Out[33]:

	Temp	cold	hot		cold
	Cities	NY	NY	LA	SF
INDEX_1	INDEX_2				
a	1	0	1	2	3
	2	4	5	6	7
b	1	8	9	10	11
	2	12	13	14	15

In [34]:
```
#We can also sort levels
dframe2.sortlevel(1)
```

Out[34]:

	Cities	NY		LA	SF
	Temp	cold	hot	hot	cold
INDEX_1	INDEX_2				
a	1	0	1	2	3
b	1	8	9	10	11
a	2	4	5	6	7
b	2	12	13	14	15

In [35]:
```
#Note the change in sorting, now the Dframe index is sorted by the INDEX_2
```

In [37]:
```
#We can also perform operations on particular levels
dframe2.sum(level='Temp',axis=1)
```

Out[37]:

	Temp	cold	hot
INDEX_1	**INDEX_2**		
a	**1**	3	3
	2	11	11
b	**1**	19	19
	2	27	27

In [38]:
```
#Thats the end of this section! Next up, Section 5: Working with Data Part 1 !!!
```

In [ ]: