In [1]:

```
import pandas as pd
```

In [2]:

```
from numpy.random import randn
```

In [3]:

```
pd.DataFrame(randn (6,5),index=['one','two',"three",'four','five','six'])
```

Out[3]:

	0	1	2	3	4
one	-1.955029	-0.672697	2.153437	-0.809270	-1.363634
two	-1.324220	3.117920	-1.148005	-1.466555	-1.859700
three	-0.706667	0.796683	1.923052	1.412627	0.050096
four	-1.341545	0.979559	-0.910649	0.778496	1.522634
five	-0.080940	-0.000502	-0.923466	0.015622	1.223975
six	-1.310568	-1.609145	0.444770	0.024410	0.270135

In [8]:

```
df=pd.DataFrame(randn (6,5),index=['one','two',"three",'four','five','six'],columns=['col1','col2',"col3",'col4','col5'])   # given other name to column instead of 1,2,3...   # we have define 'df' name to t
df
```

Out[8]:

	col1	col2	col3	col4	col5
one	-0.358708	-2.087290	-1.113366	-0.686718	-0.795131
two	-0.004546	1.057993	0.305177	0.931820	-0.036169
three	-0.254527	-3.607457	-0.165775	-1.425664	1.318990
four	-0.793640	-0.250715	1.792917	-1.115535	0.736387
five	0.098026	-1.210964	0.908060	-0.560035	-0.959230
six	-0.690569	-1.323990	-0.102974	-0.989804	0.761913

In [9]:

```
df
```

Out[9]:

	col1	col2	col3	col4	col5
one	-0.358708	-2.087290	-1.113366	-0.686718	-0.795131
two	-0.004546	1.057993	0.305177	0.931820	-0.036169
three	-0.254527	-3.607457	-0.165775	-1.425664	1.318990
four	-0.793640	-0.250715	1.792917	-1.115535	0.736387
five	0.098026	-1.210964	0.908060	-0.560035	-0.959230
six	-0.690569	-1.323990	-0.102974	-0.989804	0.761913

Conditional selections : with boolean arrays using data. With boolean indexing or logical selection, you pass an array or Series of True/False values

In [10]:

```
df>1          # in boolean will get answar in true and false,here,value greater than call as'True' and lower than 1 call as 'false'
```

Out[10]:

	col1	col2	col3	col4	col5
one	False	False	False	False	False
two	False	True	False	False	False
three	False	False	False	False	True
four	False	False	True	False	False
five	False	False	False	False	False
six	False	False	False	False	False

In [11]:

```
 df[df>1]  # here, wherever value is greater than'1' call as 'NaN'
```

Out[11]:

	col1	col2	col3	col4	col5
one	NaN	NaN	NaN	NaN	NaN
two	NaN	1.057993	NaN	NaN	NaN
three	NaN	NaN	NaN	NaN	1.31899
four	NaN	NaN	1.792917	NaN	NaN
five	NaN	NaN	NaN	NaN	NaN
six	NaN	NaN	NaN	NaN	NaN

In [16]:

```
# to check condition in specific column
df[df['col3'] > 1]
```

Out[16]:

	col1	col2	col3	col4	col5
four	-0.79364	-0.250715	1.792917	-1.115535	0.736387

In [18]:

```
df
```

Out[18]:

	col1	col2	col3	col4	col5
one	-0.358708	-2.087290	-1.113366	-0.686718	-0.795131
two	-0.004546	1.057993	0.305177	0.931820	-0.036169
three	-0.254527	-3.607457	-0.165775	-1.425664	1.318990
four	-0.793640	-0.250715	1.792917	-1.115535	0.736387
five	0.098026	-1.210964	0.908060	-0.560035	-0.959230
six	-0.690569	-1.323990	-0.102974	-0.989804	0.761913

In [23]:

```
df[df['col3']>0]
```

Out[23]:

	col1	col2	col3	col4	col5
two	-0.004546	1.057993	0.305177	0.931820	-0.036169
four	-0.793640	-0.250715	1.792917	-1.115535	0.736387
five	0.098026	-1.210964	0.908060	-0.560035	-0.959230

In [24]:

```
df[df['col3']>0]['col4']
```

Out[24]:

```
two     0.931820
four   -1.115535
five   -0.560035
Name: col4, dtype: float64
```

In [26]:

```
df[df['col3']>0][['col4','col5']]
```

Out[26]:

	col4	col5
two	0.931820	-0.036169
four	-1.115535	0.736387
five	-0.560035	-0.959230

In [29]:

```
df[(df['col2']>0) & (df['col3']>0)]
```

Out[29]:

	col1	col2	col3	col4	col5
two	-0.004546	1.057993	0.305177	0.93182	-0.036169

In [ ]:

In [ ]:

In [ ]: