```
#Bui Duc Manh
19521822
#linkgithub:https://github.com/DucManh75/MKTG5883.N22.CTTT.git
```

```
19521822
```

```
%matplotlib inline
import numpy as np
import pandas as pd

df = pd.read_csv("PastHires.csv")
df.head()
```

	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
0	10	Y	4	BS	N	N	Y
1	0	N	0	BS	Y	Y	Y
2	7	N	6	BS	N	N	N
3	2	Y	1	MS	Y	N	Y
4	20	N	2	PhD	Y	N	N

```
df.head(10)
```

	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
0	10	Y	4	BS	N	N	Y
1	0	N	0	BS	Y	Y	Y
2	7	N	6	BS	N	N	N
3	2	Y	1	MS	Y	N	Y
4	20	N	2	PhD	Y	N	N

```
df.tail(4)
```

	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
9	0	N	0	BS	N	N	N
10	1	N	1	PhD	Y	N	N
11	4	Y	1	BS	N	Y	Y
12	0	N	0	PhD	Y	N	Y

```
df.shape
```

```
(13, 7)
```

```
df.size
```

```
91
```

```
len(df)
```

```
13
```

```
df.columns

     Index(['Years Experience', 'Employed?', 'Previous employers',
            'Level of Education', 'Top-tier school', 'Interned', 'Hired'],
           dtype='object')


df['Hired']

     0     Y
     1     Y
     2     N
     3     Y
     4     N
     5     Y
     6     Y
     7     Y
     8     Y
     9     N
     10    N
     11    Y
     12    Y
     Name: Hired, dtype: object


df['Hired'][:5]

     0    Y
     1    Y
     2    N
     3    Y
     4    N
     Name: Hired, dtype: object


df['Hired'][5]

     'Y'


df[['Years Experience', 'Hired']]
```

	Years Experience	Hired
0	10	Y
1	0	Y
2	7	N
3	2	Y
4	20	N
5	0	Y
6	5	Y
7	3	Y
8	15	Y
9	0	N
10	1	N
11	4	Y
12	0	Y

```
df[['Years Experience', 'Hired']][:5]
```

	Years Experience	Hired
0	10	Y
1	0	Y
2	7	N
3	2	Y
4	20	N

```
df.sort_values(['Years Experience'])
```

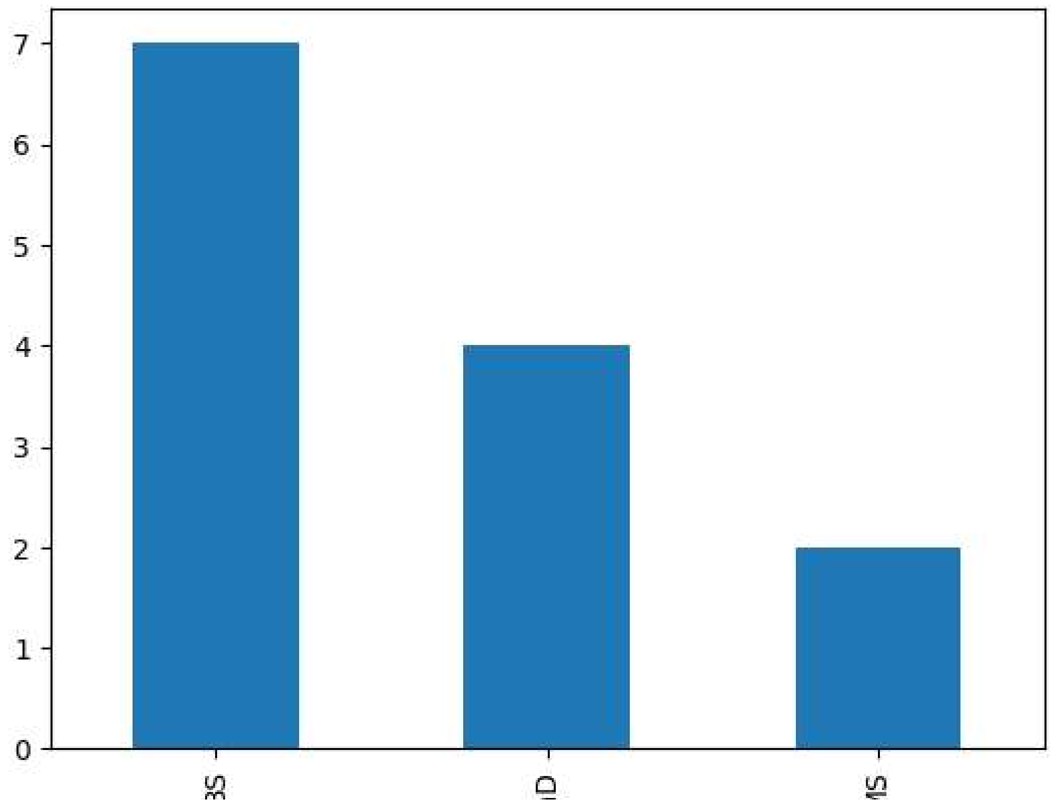
	Years Experience	Employed?	Previous employers	Level of Education	Top-tier school	Interned	Hired
1	0	N	0	BS	Y	Y	Y
5	0	N	0	PhD	Y	Y	Y
9	0	N	0	BS	N	N	N
12	0	N	0	PhD	Y	N	Y
10	1	N	1	PhD	Y	N	N
3	2	Y	1	MS	Y	N	Y
7	3	N	1	BS	N	Y	Y
11	4	Y	1	BS	N	Y	Y
6	5	Y	2	MS	N	Y	Y
2	7	N	6	BS	N	N	N
0	10	Y	4	BS	N	N	Y
8	15	Y	5	BS	N	N	Y
4	20	N	2	PhD	Y	N	N

```
degree_counts = df['Level of Education'].value_counts()
degree_counts
```

```
BS     7
PhD    4
MS     2
Name: Level of Education, dtype: int64
```

```
degree_counts.plot(kind='bar')
```

```
<Axes: >
```



```
import numpy as np
```

```
import pandas as pd
```

```
labels = ['a','b','c']
my_list = [10,20,30]
arr = np.array([10,20,30])
d = {'a':10,'b':20,'c':30}
```

```
pd.Series(data=my_list)
```

```
    0    10
    1    20
    2    30
    dtype: int64
```

```
pd.Series(data=my_list,index=labels)
```

```
    a    10
    b    20
    c    30
    dtype: int64
```

```
pd.Series(my_list,labels)
```

```
    a    10
    b    20
    c    30
    dtype: int64
```

```
pd.Series(arr)
```

```
    0    10
    1    20
    2    30
    dtype: int64
```

```
pd.Series(arr,labels)
```

```
    a    10
    b    20
    c    30
    dtype: int64
```

```
pd.Series(d)
```

```
    a    10
    b    20
```

```
     c    30
     dtype: int64
```

```
pd.Series(data=labels)
```

```
     0    a
     1    b
     2    c
     dtype: object
```

```
#Even functions (although unlikely that you will use this)
pd.Series([sum,print,len])
```

```
     0      <built-in function sum>
     1    <built-in function print>
     2      <built-in function len>
     dtype: object
```

```
serl = pd.Series([1,2,3,4],index = ['USA','Germany','USSR','Japan'])
```

```
serl
```

```
     USA        1
     Germany    2
     USSR       3
     Japan      4
     dtype: int64
```

```
ser2 = pd.Series([1,2,3,4],index = ['USA','Germany','USSR','Japan'])
```

```
ser2
```

```
     USA        1
     Germany    2
     USSR       3
     Japan      4
     dtype: int64
```

```
serl['USA']
```

```
     1
```

```
serl+ser2
```

```
     USA        2
     Germany    4
     USSR       6
     Japan      8
     dtype: int64
```

```
#DataFrame
```

```
import numpy as np
import pandas as pd
```

```
import random as randn
```

```
from numpy.random import randn
np.random.seed(101)
```

```
df = pd.DataFrame(randn(5,4),index='A B C D E'.split(),columns='W X Y Z'.split())
```

```
df
```

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965

```
df['W']
```

```
A    2.706850
B    0.651118
C   -2.018168
D    0.188695
E    0.190794
Name: W, dtype: float64
```

```
df[['W','Z']]
```

	W	Z
A	2.706850	0.503826
B	0.651118	0.605965
C	-2.018168	-0.589001
D	0.188695	0.955057
E	0.190794	0.683509

```
df.W
```

```
A    2.706850
B    0.651118
C   -2.018168
D    0.188695
E    0.190794
Name: W, dtype: float64
```

```
type(df['W'])
```

```
    pandas.core.series.Series
```

```
df['new'] = df['W'] + df['Y']
```

```
df
```

	W	X	Y	Z	new
A	2.706850	0.628133	0.907969	0.503826	3.614819
B	0.651118	-0.319318	-0.848077	0.605965	-0.196959
C	-2.018168	0.740122	0.528813	-0.589001	-1.489355
D	0.188695	-0.758872	-0.933237	0.955057	-0.744542
E	0.190794	1.978757	2.605967	0.683509	2.796762

```
df.drop('new',axis=1)
```

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057
E	0.190794	1.978757	2.605967	0.683509

```
df
```

	W	X	Y	Z	new
A	2.706850	0.628133	0.907969	0.503826	3.614819
B	0.651118	-0.319318	-0.848077	0.605965	-0.196959
C	-2.018168	0.740122	0.528813	-0.589001	-1.489355
D	0.188695	-0.758872	-0.933237	0.955057	-0.744542

```
df.drop('new',axis=1,inplace=True)
```

```
df
```

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057
E	0.190794	1.978757	2.605967	0.683509

```
df.drop('E',axis=0)
```

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057

```
df.loc['A']
```

```
W    2.706850
X    0.628133
Y    0.907969
Z    0.503826
Name: A, dtype: float64
```

```
df.iloc[2]
```

```
W   -2.018168
X    0.740122
Y    0.528813
Z   -0.589001
Name: C, dtype: float64
```

```
df.loc['B','Y']
```

```
-0.8480769834036315
```

```
df.loc[['A','B'],['W','Y']]
```

	W	Y
A	2.706850	0.907969
B	0.651118	-0.848077

```
df
```

	W	X	Y	Z

```
df>0
```

	W	X	Y	Z
A	True	True	True	True
B	True	False	False	True
C	False	True	True	False
D	True	False	False	True
E	True	True	True	True

```
df[df>0]
```

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	NaN	NaN	0.605965
C	NaN	0.740122	0.528813	NaN
D	0.188695	NaN	NaN	0.955057
E	0.190794	1.978757	2.605967	0.683509

```
df[df['W']>0]
```

	W	X	Y	Z

```
df[df['W']>0]['Y']
```

```
A    0.907969
B   -0.848077
D   -0.933237
E    2.605967
Name: Y, dtype: float64
```

```
df[df['W']>0][['Y','X']]
```

	Y	X
A	0.907969	0.628133
B	-0.848077	-0.319318
D	-0.933237	-0.758872
E	2.605967	1.978757

```
df[(df['W']>0) & (df['Y']>1)]
```

	W	X	Y	Z
E	0.190794	1.978757	2.605967	0.683509

```
df
```

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826

```
df.reset_index()
```

	index	W	X	Y	Z
0	A	2.706850	0.628133	0.907969	0.503826
1	B	0.651118	-0.319318	-0.848077	0.605965
2	C	-2.018168	0.740122	0.528813	-0.589001
3	D	0.188695	-0.758872	-0.933237	0.955057
4	E	0.190794	1.978757	2.605967	0.683509

```
newind = 'CA NY WY OR CO'.split()
df['States'] = newind
df
```

	W	X	Y	Z	Status	States
A	2.706850	0.628133	0.907969	0.503826	CA	CA
B	0.651118	-0.319318	-0.848077	0.605965	NY	NY
C	-2.018168	0.740122	0.528813	-0.589001	WY	WY
D	0.188695	-0.758872	-0.933237	0.955057	OR	OR
E	0.190794	1.978757	2.605967	0.683509	CO	CO

```
df.set_index('States')
```

	W	X	Y	Z	Status
States					
CA	2.706850	0.628133	0.907969	0.503826	CA
NY	0.651118	-0.319318	-0.848077	0.605965	NY
WY	-2.018168	0.740122	0.528813	-0.589001	WY
OR	0.188695	-0.758872	-0.933237	0.955057	OR

```
df
```

	W	X	Y	Z	Status	States
A	2.706850	0.628133	0.907969	0.503826	CA	CA
B	0.651118	-0.319318	-0.848077	0.605965	NY	NY
C	-2.018168	0.740122	0.528813	-0.589001	WY	WY
D	0.188695	-0.758872	-0.933237	0.955057	OR	OR
E	0.190794	1.978757	2.605967	0.683509	CO	CO

```
df.set_index('States',inplace=True)
df
```

	W	X	Y	Z	Status
States					
CA	2.706850	0.628133	0.907969	0.503826	CA
NY	0.651118	-0.319318	-0.848077	0.605965	NY
WY	-2.018168	0.740122	0.528813	-0.589001	WY
OR	0.188695	-0.758872	-0.933237	0.955057	OR
CO	0.190794	1.978757	2.605967	0.683509	CO

```
outside = ['G1','G1','G1','G2','G2','G2']
inside = [1,2,3,1,2,3]
hier_index = list(zip(outside,inside))
hier_index = pd.MultiIndex.from_tuples(hier_index)
```

```
hier_index
```

```
MultiIndex([('G1', 1),
            ('G1', 2),
            ('G1', 3),
            ('G2', 1),
            ('G2', 2),
            ('G2', 3)],
           )
```

```
df = pd.DataFrame(np.random.randn(6,2),index=hier_index,columns=['A','B'])
df
```

		A	B
G1	**1**	-0.497104	-0.754070
	2	-0.943406	0.484752
	3	-0.116773	1.901755
G2	**1**	0.238127	1.996652
	2	-0.993263	0.196800
	3	-1.136645	0.000366

```
df.loc['G1']
```

	A	B
1	-0.497104	-0.754070

```
df.loc['G1'].loc[1]
```

```
A   -0.497104
B   -0.754070
Name: 1, dtype: float64
```

```
df.index.names
df.index.names = ['Group','Num']
df
```

		A	B
Group	**Num**		
G1	**1**	-0.497104	-0.754070
	2	-0.943406	0.484752
	3	-0.116773	1.901755
G2	**1**	0.238127	1.996652
	2	-0.993263	0.196800
	3	-1.136645	0.000366

```
df.xs('G1')
```

	A	B

```
df.xs(['G1',1])
```

```
<ipython-input-91-c549ee06ce91>:1: FutureWarning: Passing lists as key for xs is deprecated and will be removed in a fu
  df.xs(['G1',1])
A   -0.497104
B   -0.754070
Name: (G1, 1), dtype: float64
```

```
df.xs(1,level='Num')
```

	A	B
Group		
G1	-0.497104	-0.754070
G2	0.238127	1.996652

```
#Missing Data
```

```
import numpy as np
import pandas as pd
```

```
df = pd.DataFrame({'A':[1,2,np.nan],'B':[5,np.nan,np.nan],'C':[1,2,3]})
df
```

	A	B	C
0	1.0	5.0	1
1	2.0	NaN	2
2	NaN	NaN	3

```
df.dropna()
```

	A	B	C
0	1.0	5.0	1

```
df.dropna(axis=1)
```

	C
0	1
1	2
2	3

```
df.dropna(thresh=2)
```

	A	B	C
0	1.0	5.0	1
1	2.0	NaN	2

```
df.fillna(value='FILL VALUE')
```

	A	B	C
0	1.0	5.0	1
1	2.0	FILL VALUE	2
2	FILL VALUE	FILL VALUE	3

```
df['A'].fillna(value=df['A'].mean())
```

```
0    1.0
1    2.0
2    1.5
Name: A, dtype: float64
```

```
#Create dataFrame
data = {'Company':['GOOG','GOOG','MSFT','MSFT','FB','FB'],
        'Person':['Sam','Charlie','Amy','Vanessa','Carl','Sarah'],
        'Sales':[200,120,340,124,243,350]}
df = pd.DataFrame(data)
df
```

	Company	Person	Sales
0	GOOG	Sam	200
1	GOOG	Charlie	120
2	MSFT	Amy	340
3	MSFT	Vanessa	124
4	FB	Carl	243
5	FB	Sarah	350

```
df.groupby('Company')
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7faa63e05f40>
```

```
by_comp = df.groupby('Company')
```

```
by_comp.mean()
```

	Sales
Company	
FB	296.5

```
df.groupby('Company').mean()
```

	Sales
Company	
FB	296.5
GOOG	160.0
MSFT	232.0

```
by_comp.std()
```

	Sales
Company	
FB	75.660426
GOOG	56.568542
MSFT	152.735065

```
by_comp.min(

)
```

	Person	Sales
Company		
FB	Carl	243

```
by_comp.max()
```

	Person	Sales
Company		
FB	Sarah	350
GOOG	Sam	200
MSFT	Vanessa	340

```
by_comp.count()
```

	Person	Sales
Company		
FB	2	2
GOOG	2	2
MSFT	2	2

```
by_comp.describe()
```

```
Sales
count  mean  std      min   25%   50%   75%   max
```

```
by_comp.describe().transpose()
```

	Company	FB	GOOG	MSFT
Sales	**count**	2.000000	2.000000	2.000000
	mean	296.500000	160.000000	232.000000
	std	75.660426	56.568542	152.735065
	min	243.000000	120.000000	124.000000
	25%	269.750000	140.000000	178.000000
	50%	296.500000	160.000000	232.000000
	75%	323.250000	180.000000	286.000000
	max	350.000000	200.000000	340.000000

```
by_comp.describe().transpose()['GOOG']
```

```
Sales  count      2.000000
       mean     160.000000
       std       56.568542
       min      120.000000
       25%      140.000000
       50%      160.000000
       75%      180.000000
       max      200.000000
Name: GOOG, dtype: float64
```

```
#Merging, Joining and Concatenating
```

```
df1 = pd.DataFrame({'A':['A0','A1','A2','A3'],
                    'B':['B0','B1','B2','B3'],
                    'C':['C0','C1','C2','C3'],
                    'D':['D0','D1','D2','D3']},
                   index=[0,1,2,3])


df2 = pd.DataFrame({'A':['A4','A5','A6','A7'],
                    'B':['B4','B5','B6','B7'],
                    'C':['C4','C5','C6','C7'],
                    'D':['D4','D5','D6','D7']},
                   index=[4,5,6,7])


df3 = pd.DataFrame({'A':['A8','A9','A10','A11'],
                    'B':['B8','B9','B10','B11'],
                    'C':['C8','C9','C10','C11'],
                    'D':['D8','D9','D10','D11']},
                   index=[8,9,10,11])
```

```
df1
```

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

```
df2
```

	A	B	C	D

```
df3
```

	A	B	C	D
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

```
pd.concat([df1,df2,df3])
```

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

```
pd.concat([df1,df2,df3],axis=1)
```

	A	B	C	D	A	B	C	D	A	B	C	D
0	A0	B0	C0	D0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	A1	B1	C1	D1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	A2	B2	C2	D2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	A3	B3	C3	D3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	A4	B4	C4	D4	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	A5	B5	C5	D5	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	A6	B6	C6	D6	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	A7	B7	C7	D7	NaN	NaN	NaN	NaN
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A8	B8	C8	D8
9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A9	B9	C9	D9
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A10	B10	C10	D10
11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A11	B11	C11	D11

```
left = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],
'A': ['A0', 'A1', 'A2', 'A3'],
'B': ['B0', 'B1', 'B2', 'B3']})
right = pd.DataFrame ({ 'key': ['K0', 'K1', 'K2', 'K3'],
'C': ['C0' ,'C1', 'C2', 'C3'],
'D': ['D0', 'D1', 'D2', 'D3']})
```

```
left
```

	key	A	B
0	K0	A0	B0
1	K1	A1	B1
2	K2	A2	B2

```
right
```

	key	C	D
0	K0	C0	D0
1	K1	C1	D1
2	K2	C2	D2
3	K3	C3	D3

```
pd.merge(left,right,how='inner',on='key')
```

	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K2	A2	B2	C2	D2
3	K3	A3	B3	C3	D3

```
left = pd.DataFrame({'key1': ['K0', 'K1', 'K2', 'K3'],
                     'key2': ['K0', 'K1', 'K0', 'K1'],
                        'A': ['A0', 'A1', 'A2', 'A3'],
                        'B': ['B0', 'B1', 'B2', 'B3']})
right = pd.DataFrame ({ 'key1': ['K0', 'K1', 'K2', 'K3'],
                       'key2': ['K0', 'K0', 'K0', 'K0'],
                        'C': ['C0' ,'C1', 'C2', 'C3'],
                        'D': ['D0', 'D1', 'D2', 'D3']})
```

```
pd.merge(left,right,on=['key1','key2'])
```

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K2	K0	A2	B2	C2	D2

```
pd.merge(left,right,how='outer',on=['key1','key2'])
```

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K1	K1	A1	B1	NaN	NaN
2	K2	K0	A2	B2	C2	D2
3	K3	K1	A3	B3	NaN	NaN
4	K1	K0	NaN	NaN	C1	D1
5	K3	K0	NaN	NaN	C3	D3

```
pd.merge(left,right,how='left',on=['key1','key2'])
```

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K1	K1	A1	B1	NaN	NaN

```
pd.merge(left,right,how='right',on=['key1','key2'])
```

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K1	K0	NaN	NaN	C1	D1
2	K2	K0	A2	B2	C2	D2
3	K3	K0	NaN	NaN	C3	D3

```
left = pd.DataFrame({ 'A': ['A0', 'A1', 'A2'],
                      'B': ['B0', 'B1', 'B2']},
                    index=['K0','K1','K2'])
right = pd.DataFrame ({ 'C': ['C0' ,'C1', 'C2'],
                        'D': ['D0', 'D1', 'D2']},
                      index=['K0','K2','K3'])
```

```
left.join(right)
```

	A	B	C	D
K0	A0	B0	C0	D0
K1	A1	B1	NaN	NaN
K2	A2	B2	C1	D1

```
left.join(right,how='outer')
```

	A	B	C	D
K0	A0	B0	C0	D0
K1	A1	B1	NaN	NaN
K2	A2	B2	C1	D1
K3	NaN	NaN	C2	D2

```
#Operations
```

```
df = pd.DataFrame({'coll':[1,2,3,4],'col2':[444,555,666,444],'col3':['abc','def','ghi','xyz']})
```

```
df.head()
```

	coll	col2	col3
0	1	444	abc
1	2	555	def
2	3	666	ghi
3	4	444	xyz

```
df['col2'].unique()
```

```
array([444, 555, 666])
```

```
df['col2'].nunique()
```

```
3
```

```
df['col2'].value_counts()
```

```
444    2
555    1
666    1
Name: col2, dtype: int64
```

```
newdf = df[(df['coll']>2) & (df['col2']==444)]
```

```
newdf
```

	coll	col2	col3
3	4	444	xyz

```
def time2(x):
  return x*2
```

```
df['coll'].apply(time2)
```

```
0    2
1    4
2    6
3    8
Name: coll, dtype: int64
```

```
df['col3'].apply(len)
```

```
0    3
1    3
2    3
3    3
Name: col3, dtype: int64
```

```
df['coll'].sum()
```

```
    10
```

```
del df['coll']
df
```

	col2	col3
0	444	abc
1	555	def
2	666	ghi
3	444	xyz

```
df.columns
```

```
    Index(['col2', 'col3'], dtype='object')
```

```
df.index
```

```
    RangeIndex(start=0, stop=4, step=1)
```

```
df
```

	col2	col3
0	444	abc
1	555	def
2	666	ghi
3	444	xyz

```
df.sort_values(by='col2')
```

	col2	col3
0	444	abc
3	444	xyz
1	555	def
2	666	ghi

```
df.isnull()
```

	col2	col3
0	False	False
1	False	False
2	False	False
3	False	False

```
df.dropna
```

```
<bound method DataFrame.dropna of    col2 col3
0   444  abc
1   555  def
2   666  ghi
3   444  xyz>
```

```
import numpy as np

df = pd.DataFrame({'coll':[1,2,3,np.nan],
                   'col2':[np.nan,555,666,444],
                   'col3':['abc','def','ghi','xyz']})
df.head()
```

	col1	col2	col3
0	1.0	NaN	abc
1	2.0	555.0	def
2	3.0	666.0	ghi
3	NaN	444.0	xyz

```
df.isnull()
```

	col1	col2	col3
0	False	True	False
1	False	False	False
2	False	False	False
3	True	False	False

```
df.dropna()
```

	col1	col2	col3
1	2.0	555.0	def
2	3.0	666.0	ghi

```
df.fillna('FILL')
```

	col1	**col2**	**col3**
0	1.0	FILL	abc

```
data = {'A':['foo','foo', 'foo', 'bar','bar','bar'],
'B':['one','one', 'two ', 'two','one', 'one'],
'C':['x','y','x','y','x','y'],
'D':[1,3,2,5,4,1]}
df = pd.DataFrame(data)
```

```
df
```

	A	**B**	**C**	**D**
0	foo	one	x	1
1	foo	one	y	3
2	foo	two	x	2
3	bar	two	y	5
4	bar	one	x	4
5	bar	one	y	1

```
df.pivot_table(values='D',index=['A','B'],columns=['C'])
```

	C	**x**	**y**
A	**B**		
bar	**one**	4.0	1.0
	two	NaN	5.0
foo	**one**	1.0	3.0
	two	2.0	NaN

```
#Data Input Ouput
import numpy as np
import pandas as pd
```

```
df = pd.read_csv('example.csv')
df
```

	a	b	c	d
0	0	1	2	3
1	4	5	6	7
2	8	9	10	11
3	12	13	14	15

```
df.to_csv('example.csv',index=False)
```