数据规整

分层索引	**stack , unstack**
重排列和层级排序	**swaplevel , sort_index**
按层级进行汇总统计	**group by**
使用DataFrame的列进行索引	**set_index , reset_index**
merge	
concat	
combine_first	

- **分层索引**

```
#Series
data = pd.Series(np.arange(1,10),index = [['a','a','a','b','b','c','c','d','d'],
                                          [1,2,3,1,3,1,2,2,3]])
data
Out[4]:
a  1    1
   2    2
   3    3
b  1    4
   3    5
c  1    6
   2    7
d  2    8
   3    9
dtype: int32

#部分索引选择子集
data.loc[['b','c']]
Out[6]:
b  1    4
   3    5
c  1    6
   2    7
dtype: int32

data.loc[:,2]
Out[8]:
a    2
```

```
c    7
d    8
dtype: int32

#使用unstack方法重新排列数据
data.unstack()
Out[11]:
     1    2    3
a  1.0  2.0  3.0
b  4.0  NaN  5.0
c  6.0  7.0  NaN
d  NaN  8.0  9.0

#stack : unstack的反操作
data.unstack().stack()
Out[12]:
a  1    1.0
   2    2.0
   3    3.0
b  1    4.0
   3    5.0
c  1    6.0
   2    7.0
d  2    8.0
   3    9.0
dtype: float64

#DataFrame
df = pd.DataFrame(np.arange(12).reshape((4,3)),
                  index = [['a','a','b','b'],[1,2,1,2]],
                  columns = [['Ohio','Ohio','Colorado'],['Green','Red','Green']])
df
Out[17]:
     Ohio     Colorado
    Green Red    Green
a 1     0   1        2
  2     3   4        5
b 1     6   7        8
  2     9  10       11

#给层级加上名称
df.index.names = ['key1','key2']
df.columns.names = ['State','Color']
df
Out[21]:
State      Ohio     Colorado
Color     Green Red    Green
key1 key2
a    1        0   1        2
     2        3   4        5
b    1        6   7        8
     2        9  10       11
```

- **重排列和层级排序**

```
#重新排列轴顺序 swaplevel()
df.swaplevel(i = 'key1',j = 'key2')
Out[23]:
State      Ohio     Colorado
Color     Green Red    Green
key2 key1
1    a        0   1        2
2    a        3   4        5
1    b        6   7        8
2    b        9  10       11

#按照层级进行字典排序  sort_index()
df.sort_index(level = 'key2')
Out[28]:
State      Ohio     Colorado
Color     Green Red    Green
key1 key2
a    1        0   1        2
b    1        6   7        8
a    2        3   4        5
b    2        9  10       11

#两者结合
df.swaplevel('key1','key2').sort_index(level = 'key2')
Out[30]:
State      Ohio     Colorado
Color     Green Red    Green
key2 key1
1    a        0   1        2
     b        6   7        8
2    a        3   4        5
     b        9  10       11
```

- **按层级进行汇总统计**

```
df
Out[39]:
State      Ohio     Colorado
Color     Green Red    Green
key1 key2
a    1        0   1        2
     2        3   4        5
b    1        6   7        8
     2        9  10       11
```

```
df.sum(level = 'key2')
Out[38]:
State  Ohio     Colorado
Color Green Red    Green
key2
1         6   8       10
2        12  14       16

df.sum(level = 'State',axis=1)
Out[41]:
State      Ohio  Colorado
key1 key2
a    1        1         2
     2        7         5
b    1       13         8
     2       19        11
```

- **使用DataFrame的列进行索引**

```
df = pd.DataFrame({'a':range(7),'b':range(7,0,-1),'c':
                   ['one','one','one','two','two','two','two'],'d':[0,1,2,0,1,2,3]})
df
Out[45]:
   a  b    c  d
0  0  7  one  0
1  1  6  one  1
2  2  5  one  2
3  3  4  two  0
4  4  3  two  1
5  5  2  two  2
6  6  1  two  3

#将df中的列作为索引,set_index
df.set_index(['c','d'])
Out[46]:
       a  b
c   d
one 0  0  7
    1  1  6
    2  2  5
two 0  3  4
    1  4  3
    2  5  2
    3  6  1
#若要保留列，则传入drop = False
df.set_index(['c','d'],drop = False)
Out[48]:
       a  b    c  d
c   d
one 0  0  7  one  0
    1  1  6  one  1
```

```
    2  2  5  one  2
two 0  3  4  two  0
    1  4  3  two  1
    2  5  2  two  2
    3  6  1  two  3
#reset_index : set_index的反操作
df.set_index(['c','d']).reset_index()
Out[49]:
     c  d  a  b
0  one  0  0  7
1  one  1  1  6
2  one  2  2  5
3  two  0  3  4
4  two  1  4  3
5  two  2  5  2
6  two  3  6  1
```

- **merge联合**

```
pd.merge(['right', "how='inner'", 'on=None', 'left_on=None', 'right_on=None',
       'left_index=False', 'right_index=False', 'sort=False', "suffixes=('_x', '_y')",
     'copy=True', 'indicator=False', 'validate=None'])

#on : 指定连接键，没有指定时会将同名键作为连接键。
df1 = pd.DataFrame({'key':['b','b','a','c','a','a','b'],'data1':range(7)})
df1
Out[53]:
  key  data1
0   b      0
1   b      1
2   a      2
3   c      3
4   a      4
5   a      5
6   b      6
df2 = pd.DataFrame({'key':['a','b','d'],'data2':range(3)})
df2
Out[56]:
  key  data2
0   a      0
1   b      1
2   d      2

pd.merge(df1,df2,on = 'key')
Out[57]:
  key  data1  data2
0   b      0      1
1   b      1      1
2   b      6      1
3   a      2      0
4   a      4      0
```

```
5   a      5      0

#left_on,right_on : 为每个连接对象指定连接键
df3 = pd.DataFrame({'lkey':['b','b','a','c','a','a','b'],'data1':range(7)})
df3
Out[59]:
  lkey  data1
0    b      0
1    b      1
2    a      2
3    c      3
4    a      4
5    a      5
6    b      6
df4 = pd.DataFrame({'rkey':['a','b','d'],'data2':range(3)})
df4
Out[61]:
  rkey  data2
0    a      0
1    b      1
2    d      2

pd.merge(df3,df4,left_on = 'lkey',right_on = 'rkey')
Out[67]:
  lkey  data1 rkey  data2
0    b      0    b      1
1    b      1    b      1
2    b      6    b      1
3    a      2    a      0
4    a      4    a      0
5    a      5    a      0

#how 连接方式 'inner', 'left', 'right', 'outer' ,默认为 'inner'
pd.merge(df3,df4,left_on = 'lkey',right_on = 'rkey',how = 'left')
Out[74]:
  lkey  data1 rkey  data2
0    b      0    b    1.0
1    b      1    b    1.0
2    a      2    a    0.0
3    c      3  NaN    NaN
4    a      4    a    0.0
5    a      5    a    0.0
6    b      6    b    1.0

pd.merge(df3,df4,left_on = 'lkey',right_on = 'rkey',how = 'right')
Out[75]:
  lkey  data1 rkey  data2
0    b    0.0    b      1
1    b    1.0    b      1
2    b    6.0    b      1
3    a    2.0    a      0
4    a    4.0    a      0
5    a    5.0    a      0
```

```
6  NaN    NaN    d      2

pd.merge(df3,df4,left_on = 'lkey',right_on = 'rkey',how = 'outer')
Out[76]:
  lkey  data1 rkey  data2
0    b    0.0    b    1.0
1    b    1.0    b    1.0
2    b    6.0    b    1.0
3    a    2.0    a    0.0
4    a    4.0    a    0.0
5    a    5.0    a    0.0
6    c    3.0  NaN    NaN
7  NaN    NaN    d    2.0

#left_index,right_index，无公共键时，可指定行索引作为连接键
left
Out[79]:
  key  value
0   a      0
1   b      1
2   a      2
3   a      3
4   b      4
5   c      5
right
Out[80]:
   group_val
a        3.5
b        7.0
pd.merge(left,right,left_on = 'key',right_index = True)
Out[83]:
  key  value  group_val
0   a      0        3.5
2   a      2        3.5
3   a      3        3.5
1   b      1        7.0
4   b      4        7.0
#两边的索引作为连接键
left1
Out[86]:
   Ohio  Nevada
a     1       2
c     3       4
e     5       6
right1
Out[87]:
   Missouri  Alabama
b         7        8
c         9       10
d        11       12
e        13       14
pd.merge(left1,right1,how = 'outer',left_index = True,right_index = True)
OR  left1.join(right1,how = 'outer')
```

```
Out[88]:
   Ohio  Nevada  Missouri  Alabama
a   1.0     2.0       NaN      NaN
b   NaN     NaN       7.0      8.0
c   3.0     4.0       9.0     10.0
d   NaN     NaN      11.0     12.0
e   5.0     6.0      13.0     14.0
```

- **concat 沿轴向连接**

```
pd.concat(['objs', 'axis=0', "join='outer'", 'join_axes=None', 'ignore_index=False',
         'keys=None', 'levels=None', 'names=None', 'verify_integrity=False',
       'sort=None', 'copy=True'])
#Series
s1 = pd.Series([0,1],index = ['a','b'])
s2 = pd.Series([2,3,4],index = ['c','d','e'])
s3 = pd.Series([5,6],index = ['f','g'])
pd.concat([s1,s2,s3])
Out[96]:
a    0
b    1
c    2
d    3
e    4
f    5
g    6
dtype: int64
#axis,控制轴
pd.concat([s1,s2,s3],axis = 1,sort = True)
Out[98]:
     0    1    2
a  0.0  NaN  NaN
b  1.0  NaN  NaN
c  NaN  2.0  NaN
d  NaN  3.0  NaN
e  NaN  4.0  NaN
f  NaN  NaN  5.0
g  NaN  NaN  6.0

#join_axes : 指定顺序
pd.concat([s1,s2,s3],axis = 1,join_axes = [['g','f','e','d','c','b','a']])
Out[104]:
     0    1    2
g  NaN  NaN  6.0
f  NaN  NaN  5.0
e  NaN  4.0  NaN
d  NaN  3.0  NaN
c  NaN  2.0  NaN
b  1.0  NaN  NaN
a  0.0  NaN  NaN
```

```
#keys:在连接轴上创建一个多层索引
pd.concat([s1,s2,s3],keys = ['one','two','three'])
Out[105]:
one    a    0
       b    1
two    c    2
       d    3
       e    4
three  f    5
       g    6
dtype: int64
#当axis = 1时，keys则成为了列头
pd.concat([s1,s2,s3],axis = 1,keys = ['one','two','three'],sort = True)
Out[107]:
   one  two  three
a  0.0  NaN    NaN
b  1.0  NaN    NaN
c  NaN  2.0    NaN
d  NaN  3.0    NaN
e  NaN  4.0    NaN
f  NaN  NaN    5.0
g  NaN  NaN    6.0

#DataFrame
df1
Out[113]:
   one  two
a    0    1
b    2    3
c    4    5
df2
Out[114]:
   three  four
a      5     6
c      7     8

pd.concat([df1,df2],axis = 1,keys = ['level1','level2'],
                    names = ['upper','lower'],sort = True)
Out[117]:
upper level1     level2
lower    one two  three four
a          0   1    5.0  6.0
b          2   3    NaN  NaN
c          4   5    7.0  8.0
```

- **联合重塑数据**

```
a
Out[122]:
f    NaN
```

```
e    2.5
d    0.0
c    3.5
b    4.5
a    NaN
dtype: float64
b
Out[123]:
a    0.0
b    NaN
c    2.0
d    NaN
e    NaN
f    5.0
dtype: float64

np.where(pd.isnull(a),b,a)
Out[125]: array([0. , 2.5, 0. , 3.5, 4.5, 5. ])

#使用combine_first
b.combine_first(a)
Out[126]:
a    0.0
b    4.5
c    2.0
d    0.0
e    2.5
f    5.0
dtype: float64
```