```
import pandas as pd
df = pd.read_csv('D:/Python/Pandas/wode/大学排名new2019.csv',engine='python',index_col = False)
df
```

	排名	学校名称	省市	总分	指标得分
0	1	清华大学	北京	综合	94.6
1	2	北京大学	北京	综合	76.5
2	3	浙江大学	浙江	综合	72.9
3	4	上海交通大学	上海	综合	72.1
4	5	复旦大学	上海	综合	65.6
5	6	中国科学技术大学	安徽	理工	60.9
6	7	华中科技大学	湖北	综合	58.9
7	7	南京大学	江苏	综合	58.9
8	9	中山大学	广东	综合	58.2
9	10	哈尔滨工业大学	黑龙江	理工	56.7
10	11	北京航空航天大学	北京	理工	56.3
11	12	武汉大学	湖北	综合	56.2
12	13	同济大学	上海	理工	55.7
13	14	西安交通大学	陕西	综合	55.0
14	15	四川大学	四川	综合	54.4
15	16	北京理工大学	北京	理工	54.0
16	17	东南大学	江苏	综合	53.6
17	18	南开大学	天津	综合	52.8
18	19	天津大学	天津	理工	52.3
19	20	华南理工大学	广东	理工	52.0
20	21	中南大学	湖南	综合	50.3
21	22	北京师范大学	北京	师范	49.7
22	23	山东大学	山东	综合	49.1
23	23	厦门大学	福建	综合	49.1
24	25	吉林大学	吉林	综合	48.9
25	26	大连理工大学	辽宁	理工	48.6
26	27	电子科技大学	四川	理工	48.4
27	28	湖南大学	湖南	综合	48.1
28	29	苏州大学	江苏	综合	47.3
29	30	西北工业大学	陕西	理工	46.7
...	...	...	...	...	...
70	71	中国矿业大学	江苏	理工	38.6
71	72	中国地质大学（北京）	北京	理工	38.5
72	73	东北财经大学	辽宁	财经	38.2
73	73	西南财经大学	四川	财经	38.2
74	73	西南大学	重庆	综合	38.2

	排名	学校名称	省市	总分	指标得分
75	76	东北师范大学	吉林	师范	38.1
76	76	南京邮电大学	江苏	综合	38.1
77	76	中国政法大学	北京	政法	38.1
78	79	河海大学	江苏	理工	38.0
79	80	南京信息工程大学	江苏	理工	37.9
80	81	西北农林科技大学	陕西	农业	37.8
81	82	中国石油大学（华东）	山东	理工	37.4
82	83	合肥工业大学	安徽	理工	37.3
83	84	陕西师范大学	陕西	师范	37.2
84	85	华南师范大学	广东	师范	37.1
85	85	江苏大学	江苏	综合	37.1
86	87	南京工业大学	江苏	理工	37.0
87	87	中国石油大学（北京）	北京	理工	37.0
88	89	西北大学	陕西	综合	36.9
89	89	浙江工业大学	浙江	理工	36.9
90	91	北京林业大学	北京	林业	36.8
91	91	湖南师范大学	湖南	师范	36.8
92	91	浙江师范大学	浙江	师范	36.8
93	94	首都师范大学	北京	师范	36.4
94	95	汕头大学	广东	综合	36.3
95	96	中国传媒大学	北京	语言	36.2
96	97	杭州电子科技大学	浙江	理工	36.1
97	98	扬州大学	江苏	综合	36.0
98	99	安徽大学	安徽	综合	35.9
99	100	华侨大学	福建	综合	35.7

100 rows × 5 columns

```
df.describe()
```

	排名	指标得分
count	100.000000	100.000000
mean	50.230000	44.905000
std	28.882505	9.949482
min	1.000000	35.700000
25%	25.750000	38.175000
50%	50.500000	41.600000
75%	73.750000	48.675000
max	100.000000	94.600000

```
df['xlm'] = range(0,100)
df
```

	排名	学校名称	省市	总分	指标得分	xlm
0	1	清华大学	北京	综合	94.6	0
1	2	北京大学	北京	综合	76.5	1
2	3	浙江大学	浙江	综合	72.9	2
3	4	上海交通大学	上海	综合	72.1	3
4	5	复旦大学	上海	综合	65.6	4
5	6	中国科学技术大学	安徽	理工	60.9	5
6	7	华中科技大学	湖北	综合	58.9	6
7	7	南京大学	江苏	综合	58.9	7
8	9	中山大学	广东	综合	58.2	8
9	10	哈尔滨工业大学	黑龙江	理工	56.7	9
10	11	北京航空航天大学	北京	理工	56.3	10
11	12	武汉大学	湖北	综合	56.2	11
12	13	同济大学	上海	理工	55.7	12
13	14	西安交通大学	陕西	综合	55.0	13
14	15	四川大学	四川	综合	54.4	14
15	16	北京理工大学	北京	理工	54.0	15
16	17	东南大学	江苏	综合	53.6	16
17	18	南开大学	天津	综合	52.8	17
18	19	天津大学	天津	理工	52.3	18
19	20	华南理工大学	广东	理工	52.0	19
20	21	中南大学	湖南	综合	50.3	20
21	22	北京师范大学	北京	师范	49.7	21
22	23	山东大学	山东	综合	49.1	22
23	23	厦门大学	福建	综合	49.1	23
24	25	吉林大学	吉林	综合	48.9	24
25	26	大连理工大学	辽宁	理工	48.6	25
26	27	电子科技大学	四川	理工	48.4	26
27	28	湖南大学	湖南	综合	48.1	27
28	29	苏州大学	江苏	综合	47.3	28
29	30	西北工业大学	陕西	理工	46.7	29
...	...	...	...	...	...	...
70	71	中国矿业大学	江苏	理工	38.6	70
71	72	中国地质大学（北京）	北京	理工	38.5	71
72	73	东北财经大学	辽宁	财经	38.2	72
73	73	西南财经大学	四川	财经	38.2	73
74	73	西南大学	重庆	综合	38.2	74
75	76	东北师范大学	吉林	师范	38.1	75

	排名	学校名称	省市	总分	指标得分	xlm
76	76	南京邮电大学	江苏	综合	38.1	76
77	76	中国政法大学	北京	政法	38.1	77
78	79	河海大学	江苏	理工	38.0	78
79	80	南京信息工程大学	江苏	理工	37.9	79
80	81	西北农林科技大学	陕西	农业	37.8	80
81	82	中国石油大学（华东）	山东	理工	37.4	81
82	83	合肥工业大学	安徽	理工	37.3	82
83	84	陕西师范大学	陕西	师范	37.2	83
84	85	华南师范大学	广东	师范	37.1	84
85	85	江苏大学	江苏	综合	37.1	85
86	87	南京工业大学	江苏	理工	37.0	86
87	87	中国石油大学（北京）	北京	理工	37.0	87
88	89	西北大学	陕西	综合	36.9	88
89	89	浙江工业大学	浙江	理工	36.9	89
90	91	北京林业大学	北京	林业	36.8	90
91	91	湖南师范大学	湖南	师范	36.8	91
92	91	浙江师范大学	浙江	师范	36.8	92
93	94	首都师范大学	北京	师范	36.4	93
94	95	汕头大学	广东	综合	36.3	94
95	96	中国传媒大学	北京	语言	36.2	95
96	97	杭州电子科技大学	浙江	理工	36.1	96
97	98	扬州大学	江苏	综合	36.0	97
98	99	安徽大学	安徽	综合	35.9	98
99	100	华侨大学	福建	综合	35.7	99

100 rows × 6 columns

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

	排名	学校名称	省市	总分	指标得分
0	1	清华大学	北京	综合	94.6
1	2	北京大学	北京	综合	76.5
2	3	浙江大学	浙江	综合	72.9
3	4	上海交通大学	上海	综合	72.1
4	5	复旦大学	上海	综合	65.6
5	6	中国科学技术大学	安徽	理工	60.9
6	7	华中科技大学	湖北	综合	58.9
7	7	南京大学	江苏	综合	58.9
8	9	中山大学	广东	综合	58.2
9	10	哈尔滨工业大学	黑龙江	理工	56.7

	排名	学校名称	省市	总分	指标得分
10	11	北京航空航天大学	北京	理工	56.3
11	12	武汉大学	湖北	综合	56.2
12	13	同济大学	上海	理工	55.7
13	14	西安交通大学	陕西	综合	55.0
14	15	四川大学	四川	综合	54.4
15	16	北京理工大学	北京	理工	54.0
16	17	东南大学	江苏	综合	53.6
17	18	南开大学	天津	综合	52.8
18	19	天津大学	天津	理工	52.3
19	20	华南理工大学	广东	理工	52.0
20	21	中南大学	湖南	综合	50.3
21	22	北京师范大学	北京	师范	49.7
22	23	山东大学	山东	综合	49.1
23	23	厦门大学	福建	综合	49.1
24	25	吉林大学	吉林	综合	48.9
25	26	大连理工大学	辽宁	理工	48.6
26	27	电子科技大学	四川	理工	48.4
27	28	湖南大学	湖南	综合	48.1
28	29	苏州大学	江苏	综合	47.3
29	30	西北工业大学	陕西	理工	46.7
...	...	...	...	...	...
70	71	中国矿业大学	江苏	理工	38.6
71	72	中国地质大学（北京）	北京	理工	38.5
72	73	东北财经大学	辽宁	财经	38.2
73	73	西南财经大学	四川	财经	38.2
74	73	西南大学	重庆	综合	38.2
75	76	东北师范大学	吉林	师范	38.1
76	76	南京邮电大学	江苏	综合	38.1
77	76	中国政法大学	北京	政法	38.1
78	79	河海大学	江苏	理工	38.0
79	80	南京信息工程大学	江苏	理工	37.9
80	81	西北农林科技大学	陕西	农业	37.8
81	82	中国石油大学（华东）	山东	理工	37.4
82	83	合肥工业大学	安徽	理工	37.3
83	84	陕西师范大学	陕西	师范	37.2
84	85	华南师范大学	广东	师范	37.1
85	85	江苏大学	江苏	综合	37.1
86	87	南京工业大学	江苏	理工	37.0
87	87	中国石油大学（北京）	北京	理工	37.0
88	89	西北大学	陕西	综合	36.9

	排名	学校名称	省市	总分	指标得分
89	89	浙江工业大学	浙江	理工	36.9
90	91	北京林业大学	北京	林业	36.8
91	91	湖南师范大学	湖南	师范	36.8
92	91	浙江师范大学	浙江	师范	36.8
93	94	首都师范大学	北京	师范	36.4
94	95	汕头大学	广东	综合	36.3
95	96	中国传媒大学	北京	语言	36.2
96	97	杭州电子科技大学	浙江	理工	36.1
97	98	扬州大学	江苏	综合	36.0
98	99	安徽大学	安徽	综合	35.9
99	100	华侨大学	福建	综合	35.7

100 rows × 5 columns

```
df[['排名','省市']]
```

	排名	省市
0	1	北京
1	2	北京
2	3	浙江
3	4	上海
4	5	上海
5	6	安徽
6	7	湖北
7	7	江苏
8	9	广东
9	10	黑龙江
10	11	北京
11	12	湖北
12	13	上海
13	14	陕西
14	15	四川
15	16	北京
16	17	江苏
17	18	天津
18	19	天津
19	20	广东
20	21	湖南
21	22	北京
22	23	山东
23	23	福建

	排名	省市
24	25	吉林
25	26	辽宁
26	27	四川
27	28	湖南
28	29	江苏
29	30	陕西
...	...	...
70	71	江苏
71	72	北京
72	73	辽宁
73	73	四川
74	73	重庆
75	76	吉林
76	76	江苏
77	76	北京
78	79	江苏
79	80	江苏
80	81	陕西
81	82	山东
82	83	安徽
83	84	陕西
84	85	广东
85	85	江苏
86	87	江苏
87	87	北京
88	89	陕西
89	89	浙江
90	91	北京
91	91	湖南
92	91	浙江
93	94	北京
94	95	广东
95	96	北京
96	97	浙江
97	98	江苏
98	99	安徽
99	100	福建

100 rows × 2 columns

```
df['省市'] = '中国'+df['省市']
```

```
df
```

	排名	学校名称	省市	总分	指标得分
0	1	清华大学	中国北京	综合	94.6
1	2	北京大学	中国北京	综合	76.5
2	3	浙江大学	中国浙江	综合	72.9
3	4	上海交通大学	中国上海	综合	72.1
4	5	复旦大学	中国上海	综合	65.6
5	6	中国科学技术大学	中国安徽	理工	60.9
6	7	华中科技大学	中国湖北	综合	58.9
7	7	南京大学	中国江苏	综合	58.9
8	9	中山大学	中国广东	综合	58.2
9	10	哈尔滨工业大学	中国黑龙江	理工	56.7
10	11	北京航空航天大学	中国北京	理工	56.3
11	12	武汉大学	中国湖北	综合	56.2
12	13	同济大学	中国上海	理工	55.7
13	14	西安交通大学	中国陕西	综合	55.0
14	15	四川大学	中国四川	综合	54.4
15	16	北京理工大学	中国北京	理工	54.0
16	17	东南大学	中国江苏	综合	53.6
17	18	南开大学	中国天津	综合	52.8
18	19	天津大学	中国天津	理工	52.3
19	20	华南理工大学	中国广东	理工	52.0
20	21	中南大学	中国湖南	综合	50.3
21	22	北京师范大学	中国北京	师范	49.7
22	23	山东大学	中国山东	综合	49.1
23	23	厦门大学	中国福建	综合	49.1
24	25	吉林大学	中国吉林	综合	48.9
25	26	大连理工大学	中国辽宁	理工	48.6
26	27	电子科技大学	中国四川	理工	48.4
27	28	湖南大学	中国湖南	综合	48.1
28	29	苏州大学	中国江苏	综合	47.3
29	30	西北工业大学	中国陕西	理工	46.7
...	...	...	...	...	...
70	71	中国矿业大学	中国江苏	理工	38.6
71	72	中国地质大学（北京）	中国北京	理工	38.5
72	73	东北财经大学	中国辽宁	财经	38.2
73	73	西南财经大学	中国四川	财经	38.2
74	73	西南大学	中国重庆	综合	38.2

	排名	学校名称	省市	总分	指标得分
75	76	东北师范大学	中国吉林	师范	38.1
76	76	南京邮电大学	中国江苏	综合	38.1
77	76	中国政法大学	中国北京	政法	38.1
78	79	河海大学	中国江苏	理工	38.0
79	80	南京信息工程大学	中国江苏	理工	37.9
80	81	西北农林科技大学	中国陕西	农业	37.8
81	82	中国石油大学（华东）	中国山东	理工	37.4
82	83	合肥工业大学	中国安徽	理工	37.3
83	84	陕西师范大学	中国陕西	师范	37.2
84	85	华南师范大学	中国广东	师范	37.1
85	85	江苏大学	中国江苏	综合	37.1
86	87	南京工业大学	中国江苏	理工	37.0
87	87	中国石油大学（北京）	中国北京	理工	37.0
88	89	西北大学	中国陕西	综合	36.9
89	89	浙江工业大学	中国浙江	理工	36.9
90	91	北京林业大学	中国北京	林业	36.8
91	91	湖南师范大学	中国湖南	师范	36.8
92	91	浙江师范大学	中国浙江	师范	36.8
93	94	首都师范大学	中国北京	师范	36.4
94	95	汕头大学	中国广东	综合	36.3
95	96	中国传媒大学	中国北京	语言	36.2
96	97	杭州电子科技大学	中国浙江	理工	36.1
97	98	扬州大学	中国江苏	综合	36.0
98	99	安徽大学	中国安徽	综合	35.9
99	100	华侨大学	中国福建	综合	35.7

100 rows × 5 columns

```
df['date'] = '2020-12-06'
```

```
df
```

	排名	学校名称	省市	总分	指标得分	date
0	1	清华大学	中国北京	综合	94.6	2020-12-06
1	2	北京大学	中国北京	综合	76.5	2020-12-06
2	3	浙江大学	中国浙江	综合	72.9	2020-12-06
3	4	上海交通大学	中国上海	综合	72.1	2020-12-06
4	5	复旦大学	中国上海	综合	65.6	2020-12-06
5	6	中国科学技术大学	中国安徽	理工	60.9	2020-12-06
6	7	华中科技大学	中国湖北	综合	58.9	2020-12-06

	排名	学校名称	省市	总分	指标得分	date
7	7	南京大学	中国江苏	综合	58.9	2020-12-06
8	9	中山大学	中国广东	综合	58.2	2020-12-06
9	10	哈尔滨工业大学	中国黑龙江	理工	56.7	2020-12-06
10	11	北京航空航天大学	中国北京	理工	56.3	2020-12-06
11	12	武汉大学	中国湖北	综合	56.2	2020-12-06
12	13	同济大学	中国上海	理工	55.7	2020-12-06
13	14	西安交通大学	中国陕西	综合	55.0	2020-12-06
14	15	四川大学	中国四川	综合	54.4	2020-12-06
15	16	北京理工大学	中国北京	理工	54.0	2020-12-06
16	17	东南大学	中国江苏	综合	53.6	2020-12-06
17	18	南开大学	中国天津	综合	52.8	2020-12-06
18	19	天津大学	中国天津	理工	52.3	2020-12-06
19	20	华南理工大学	中国广东	理工	52.0	2020-12-06
20	21	中南大学	中国湖南	综合	50.3	2020-12-06
21	22	北京师范大学	中国北京	师范	49.7	2020-12-06
22	23	山东大学	中国山东	综合	49.1	2020-12-06
23	23	厦门大学	中国福建	综合	49.1	2020-12-06
24	25	吉林大学	中国吉林	综合	48.9	2020-12-06
25	26	大连理工大学	中国辽宁	理工	48.6	2020-12-06
26	27	电子科技大学	中国四川	理工	48.4	2020-12-06
27	28	湖南大学	中国湖南	综合	48.1	2020-12-06
28	29	苏州大学	中国江苏	综合	47.3	2020-12-06
29	30	西北工业大学	中国陕西	理工	46.7	2020-12-06
...	...	...	...	...	...	...
70	71	中国矿业大学	中国江苏	理工	38.6	2020-12-06
71	72	中国地质大学（北京）	中国北京	理工	38.5	2020-12-06
72	73	东北财经大学	中国辽宁	财经	38.2	2020-12-06
73	73	西南财经大学	中国四川	财经	38.2	2020-12-06
74	73	西南大学	中国重庆	综合	38.2	2020-12-06
75	76	东北师范大学	中国吉林	师范	38.1	2020-12-06
76	76	南京邮电大学	中国江苏	综合	38.1	2020-12-06
77	76	中国政法大学	中国北京	政法	38.1	2020-12-06
78	79	河海大学	中国江苏	理工	38.0	2020-12-06
79	80	南京信息工程大学	中国江苏	理工	37.9	2020-12-06
80	81	西北农林科技大学	中国陕西	农业	37.8	2020-12-06
81	82	中国石油大学（华东）	中国山东	理工	37.4	2020-12-06
82	83	合肥工业大学	中国安徽	理工	37.3	2020-12-06
83	84	陕西师范大学	中国陕西	师范	37.2	2020-12-06
84	85	华南师范大学	中国广东	师范	37.1	2020-12-06
85	85	江苏大学	中国江苏	综合	37.1	2020-12-06

	排名	学校名称	省市	总分	指标得分	date
86	87	南京工业大学	中国江苏	理工	37.0	2020-12-06
87	87	中国石油大学（北京）	中国北京	理工	37.0	2020-12-06
88	89	西北大学	中国陕西	综合	36.9	2020-12-06
89	89	浙江工业大学	中国浙江	理工	36.9	2020-12-06
90	91	北京林业大学	中国北京	林业	36.8	2020-12-06
91	91	湖南师范大学	中国湖南	师范	36.8	2020-12-06
92	91	浙江师范大学	中国浙江	师范	36.8	2020-12-06
93	94	首都师范大学	中国北京	师范	36.4	2020-12-06
94	95	汕头大学	中国广东	综合	36.3	2020-12-06
95	96	中国传媒大学	中国北京	语言	36.2	2020-12-06
96	97	杭州电子科技大学	中国浙江	理工	36.1	2020-12-06
97	98	扬州大学	中国江苏	综合	36.0	2020-12-06
98	99	安徽大学	中国安徽	综合	35.9	2020-12-06
99	100	华侨大学	中国福建	综合	35.7	2020-12-06

100 rows × 6 columns

```
df['date'] = pd.to_datetime(df['date'])
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
排名       100 non-null int64
学校名称     100 non-null object
省市       100 non-null object
总分       100 non-null object
指标得分     100 non-null float64
date    100 non-null datetime64[ns]
dtypes: datetime64[ns](1), float64(1), int64(1), object(3)
memory usage: 4.8+ KB
```

```
df.iloc[:10,:]
```

	排名	学校名称	省市	总分	指标得分	date
0	1	清华大学	中国北京	综合	94.6	2020-12-06
1	2	北京大学	中国北京	综合	76.5	2020-12-06
2	3	浙江大学	中国浙江	综合	72.9	2020-12-06
3	4	上海交通大学	中国上海	综合	72.1	2020-12-06
4	5	复旦大学	中国上海	综合	65.6	2020-12-06
5	6	中国科学技术大学	中国安徽	理工	60.9	2020-12-06
6	7	华中科技大学	中国湖北	综合	58.9	2020-12-06
7	7	南京大学	中国江苏	综合	58.9	2020-12-06
8	9	中山大学	中国广东	综合	58.2	2020-12-06

	排名	学校名称	省市	总分	指标得分	date
9	10	哈尔滨工业大学	中国黑龙江	理工	56.7	2020-12-06

```
df.iloc[:,:4].head(5)
```

	排名	学校名称	省市	总分
0	1	清华大学	中国北京	综合
1	2	北京大学	中国北京	综合
2	3	浙江大学	中国浙江	综合
3	4	上海交通大学	中国上海	综合
4	5	复旦大学	中国上海	综合

```
df.iloc[:,[1,4]].head(5)
```

	学校名称	指标得分
0	清华大学	94.6
1	北京大学	76.5
2	浙江大学	72.9
3	上海交通大学	72.1
4	复旦大学	65.6

```
df['指标得分'] == 58.9
```

```
0     False
1     False
2     False
3     False
4     False
5     False
6      True
7      True
8     False
9     False
10    False
11    False
12    False
13    False
14    False
15    False
16    False
17    False
18    False
19    False
20    False
21    False
22    False
23    False
24    False
25    False
26    False
27    False
28    False
```

```
29    False
      ...
70    False
71    False
72    False
73    False
74    False
75    False
76    False
77    False
78    False
79    False
80    False
81    False
82    False
83    False
84    False
85    False
86    False
87    False
88    False
89    False
90    False
91    False
92    False
93    False
94    False
95    False
96    False
97    False
98    False
99    False
Name: 指标得分, Length: 100, dtype: bool
```

```
df.loc[df['指标得分'] == 58.9,:]
```

	排名	学校名称	省市	总分	指标得分	date
6	7	华中科技大学	中国湖北	综合	58.9	2020-12-06
7	7	南京大学	中国江苏	综合	58.9	2020-12-06

```
df.loc[df['学校名称'].isin(['北京大学','清华大学']),['排名','学校名称','指标得分']]#此处插播一条 isin 函数的广告，这个函数能够帮助我们快速判断源数据中某一列（Series）的值是否等于列表中的值。任意
```

	排名	学校名称	指标得分
0	1	清华大学	94.6
1	2	北京大学	76.5

```
df.loc[(df['学校名称']=='北京大学')|(df['省市']=='中国湖南')&(df['指标得分']>df['指标得分'].mean()),:]
```

	排名	学校名称	省市	总分	指标得分	date
1	2	北京大学	中国北京	综合	76.5	2020-12-06
20	21	中南大学	中国湖南	综合	50.3	2020-12-06
27	28	湖南大学	中国湖南	综合	48.1	2020-12-06

第二部分

```
df2 = pd.read_excel('D:/Python/Pandas/wode/18dssxb.xls',sheet_name='Sheet2')
df2
```

	姓名	身份证号码	民族	政治面貌	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	移动电话	电子邮箱	备注
0	杨思琳	430181200007049024	汉族	共青团员	NaN	2013-09-01	大学本科	学校	1.880741e+10	18807413436	2681034248@qq.com	NaN
1	彭菁	430621200009152000	汉族	共青团员	NaN	2013-05-01	大学本科	学校	1.387401e+10	13874012110	1932576451@qq.com	NaN
2	陈静	430422200009200000	汉族	共青团员	NaN	2013-04-01	大学本科	学校	1.897473e+10	18974728613	1489663212@qq.com	NaN
3	米长德	43052419990308599X	汉族	共青团员	NaN	2013-04-01	大学本科	学校	NaN	18022313071	1425495519@qq.com	NaN
4	祝荣华	430421200006081024	汉族	共青团员	NaN	2013-05-01	大学本科	学校	1.567494e+10	15674944599	1785157698@qq.com	NaN
5	夏星宇	430121200006072000	汉族	共青团员	NaN	2014-09-01	大学本科	学校	NaN	16670154776	1940106723@qq.com	NaN
6	高双宁	430124200104235968	汉族	共青团员	NaN	2012-09-01	大学本科	学校	1.917433e+10	19174333504	3150306982@qq.com	NaN
7	罗银银	500222199807204992	汉族	共青团员	NaN	2012-09-01	大学本科	学校	1.992215e+10	19922154216	2715342890@qq.com	NaN
8	刘胜男	432524200002153024	汉族	共青团员	NaN	2011-10-01	大学本科	学校	1.997433e+10	19974333642	1548768792@qq.com	NaN
9	胡佳颖	431126200011249984	汉族	共青团员	NaN	2015-10-01	大学本科	学校	NaN	15574619481	839083795@qq.com	NaN

	姓名	身份证号码	民族	政治面貌	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	移动电话	电子邮箱	备注
10	张瑞	430703200007257984	汉族	共青团员	NaN	2014-10-01	大学本科	学校	1.577365e+10	15773653506	3124675870@qq.com	NaN

```
df1 = pd.read_excel('D:/Python/Pandas/wode/18dssxb.xls',sheet_name='Sheet1')
df1
```

	姓名	身份证号码	民族	政治面貌	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	移动电话	电子邮箱
0	曹盛	430981200006097024	汉族	共青团员	NaT	2013-05-01	大学本科	学校	1.527448e+10	1.527448e+10	1808105723@qq.com
1	唐慧婷	430523200007294016	汉族	共青团员	NaT	2014-06-01	大学本科	学校	1.816924e+10	1.816924e+10	1814622565@qq.com
2	李志杰	430481199910059008	汉族	共青团员	NaT	2017-10-01	大学本科	学校	1.587445e+10	1.587445e+10	2061068217@qq.com
3	马金瑶	430221200010267008	汉族	共青团员	NaT	2015-10-01	大学本科	学校	NaN	1.910743e+10	2503302793@qq.com
4	张雯	430822200007268992	苗族	共青团员	NaT	2015-10-01	大学本科	学校	NaN	1.777448e+10	361245783@qq.com
5	刘美琴	430524199906288000	汉族	中共党员	2020-05-01	2014-10-01	大学本科	学校	1.587331e+10	1.587331e+10	862376942@qq.com
6	肖含	430722200001150976	汉族	共青团员	NaT	2012-09-01	大学本科	学校	1.734737e+10	1.734737e+10	1463535552@qq.com
7	文丝雨	430302200004201024	汉族	共青团员	NaT	2015-09-01	大学本科	学校	1.397322e+10	1.397322e+10	550350931@qq.com

	姓名	身份证号码	民族	政治面貌	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	移动电话	电子邮箱
8	廖迎平	430422199912289024	汉族	共青团员	NaT	2018-10-01	大学本科	学校	1.527443e+10	1.527443e+10	1607208070@.qq.com
9	孙海燕	432524200009043008	汉族	中共预备党员	2019-12-01	2013-03-01	大学本科	学校	1.820738e+10	1.820738e+10	1085740981@qq.com
10	康嘉鹏	432524199611276992	汉族	共青团员	NaT	2012-10-01	大学本科	学校	1.557538e+10	1.557538e+10	2095793572@qq.com
11	刘泽民	431103200101284992	汉族	共青团员	NaT	2012-12-01	大学本科	学校	NaN	NaN	NaN

```
df3 = pd.read_excel('D:/Python/Pandas/wode/18dssxb.xls',sheet_name='Sheet3')
df3
```

	姓名	身份证号码	民族	政治面貌	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	移动电话	电子邮箱	备注
0	杨丹妮	410825200005267008	汉族	共青团员	NaT	2013-10-01	大学本科	学校	NaN	1.853017e+10	1607771850@qq.com	NaN
1	李发明	430421200102203968	汉族	共青团员	NaT	2014-06-01	大学本科	学校	NaN	1.507346e+10	1842644298@qq.com	NaN
2	黎晴晴	430181200001272000	汉族	共青团员	NaT	2013-03-01	大学本科	学校	NaN	1.887338e+10	1004843167@qq.com	NaN
3	唐婉婷	43112220000824006x	汉族	共青团员	NaT	2014-06-01	大学本科	学校	7.464652e+09	1.817470e+10	2396525124@qq.com	NaN

	姓名	身份证号码	民族	政治面貌	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	移动电话	电子邮箱	备注
4	王骞	433125200108123008	土家族	共青团员	NaT	2014-06-01	大学本科	学校	NaN	1.320491e+10	792825587@qq.com	NaN
5	邓雅丽	431021200009054016	汉族	共青团员	NaT	2014-09-01	大学本科	学校	NaN	NaN	NaN	NaN
6	林诗敏	430923200008177984	汉族	共青团员	NaT	2014-09-01	大学本科	学校	1.589843e+10	1.589843e+10	2904585857@qq.com	NaN
7	刘志庆	330822200006035 71X	汉族	中共预备党员	2020-12-01	2012-10-01	大学本科	学校	NaN	1.561609e+10	3144338032@qq.com	NaN

```
df = pd.concat([df1,df2,df3],axis=0,ignore_index=True)#这种情况只能自定义keys去修改列的排序问题了
df
```

	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	备注	姓名	政治面貌	民族	电子邮箱	移动电话	身份证号码
0	NaT	2013-05-01	大学本科	学校	1.527448e+10	NaN	曹盛	共青团员	汉族	1808105723@qq.com	1.527448e+10	430981200006097024
1	NaT	2014-06-01	大学本科	学校	1.816924e+10	NaN	唐慧婷	共青团员	汉族	1814622565@qq.com	1.816924e+10	430523200007294016
2	NaT	2017-10-01	大学本科	学校	1.587445e+10	NaN	李志杰	共青团员	汉族	2061068217@qq.com	1.587445e+10	430481199910059008
3	NaT	2015-10-01	大学本科	学校	NaN	NaN	马金瑶	共青团员	汉族	2503302793@qq.com	1.910743e+10	430221200010267008

	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	备注	姓名	政治面貌	民族	电子邮箱	移动电话	身份证号码
4	NaT	2015-10-01	大学本科	学校	NaN	NaN	张雯	共青团员	苗族	361245783@qq.com	1.777448e+10	430822200007268992
5	2020-05-01 00:00:00	2014-10-01	大学本科	学校	1.587331e+10	NaN	刘美琴	中共党员	汉族	862376942@qq.com	1.587331e+10	430524199906288000
6	NaT	2012-09-01	大学本科	学校	1.734737e+10	NaN	肖含	共青团员	汉族	1463535552@qq.com	1.734737e+10	430722200001150976
7	NaT	2015-09-01	大学本科	学校	1.397322e+10	NaN	文丝雨	共青团员	汉族	550350931@qq.com	1.397322e+10	430302200004201024
8	NaT	2018-10-01	大学本科	学校	1.527443e+10	NaN	廖迎平	共青团员	汉族	1607208070@.qq.com	1.527443e+10	430422199912289024
9	2019-12-01 00:00:00	2013-03-01	大学本科	学校	1.820738e+10	NaN	孙海燕	中共预备党员	汉族	1085740981@qq.com	1.820738e+10	432524200009043008
10	NaT	2012-10-01	大学本科	学校	1.557538e+10	NaN	康嘉鹏	共青团员	汉族	2095793572@qq.com	1.557538e+10	432524199611276992
11	NaT	2012-12-01	大学本科	学校	NaN	NaN	刘泽民	共青团员	汉族	NaN	NaN	431103200101284992
12	NaN	2013-09-01	大学本科	学校	1.880741e+10	NaN	杨思琳	共青团员	汉族	2681034248@qq.com	1.880741e+10	430181200007049024
13	NaN	2013-05-01	大学本科	学校	1.387401e+10	NaN	彭菁	共青团员	汉族	1932576451@qq.com	1.387401e+10	430621200009152000
14	NaN	2013-04-01	大学本科	学校	1.897473e+10	NaN	陈静	共青团员	汉族	1489663212@qq.com	1.897473e+10	430422200009200000

	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	备注	姓名	政治面貌	民族	电子邮箱	移动电话	身份证号码
15	NaN	2013-04-01	大学本科	学校	NaN	NaN	米长德	共青团员	汉族	1425495519@qq.com	1.802231e+10	43052419990308599X
16	NaN	2013-05-01	大学本科	学校	1.567494e+10	NaN	祝荣华	共青团员	汉族	1785157698@qq.com	1.567494e+10	430421200006081024
17	NaN	2014-09-01	大学本科	学校	NaN	NaN	夏星宇	共青团员	汉族	1940106723@qq.com	1.667015e+10	430121200006072000
18	NaN	2012-09-01	大学本科	学校	1.917433e+10	NaN	高双宁	共青团员	汉族	3150306982@qq.com	1.917433e+10	430124200104235968
19	NaN	2012-09-01	大学本科	学校	1.992215e+10	NaN	罗银银	共青团员	汉族	2715342890@qq.com	1.992215e+10	500222199807204992
20	NaN	2011-10-01	大学本科	学校	1.997433e+10	NaN	刘胜男	共青团员	汉族	1548768792@qq.com	1.997433e+10	432524200002153024
21	NaN	2015-10-01	大学本科	学校	NaN	NaN	胡佳颖	共青团员	汉族	839083795@qq.com	1.557462e+10	431126200011249984
22	NaN	2014-10-01	大学本科	学校	1.577365e+10	NaN	张瑞	共青团员	汉族	3124675870@qq.com	1.577365e+10	430703200007257984
23	NaT	2013-10-01	大学本科	学校	NaN	NaN	杨丹妮	共青团员	汉族	1607771850@qq.com	1.853017e+10	410825200005267008
24	NaT	2014-06-01	大学本科	学校	NaN	NaN	李发明	共青团员	汉族	1842644298@qq.com	1.507346e+10	430421200102203968
25	NaT	2013-03-01	大学本科	学校	NaN	NaN	黎晴晴	共青团员	汉族	1004843167@qq.com	1.887338e+10	430181200001272000
26	NaT	2014-06-01	大学本科	学校	7.464652e+09	NaN	唐婉婷	共青团员	汉族	2396525124@qq.com	1.817470e+10	43112220000824006x

	入党年月	入团年月	全日制教育学历	团员所在领域	固定电话	备注	姓名	政治面貌	民族	电子邮箱	移动电话	身份证号码
27	NaT	2014-06-01	大学本科	学校	NaN	NaN	王骞	共青团员	土家族	792825587@qq.com	1.320491e+10	433125200108123008
28	NaT	2014-09-01	大学本科	学校	NaN	NaN	邓雅丽	共青团员	汉族	NaN	NaN	431021200009054016
29	NaT	2014-09-01	大学本科	学校	1.589843e+10	NaN	林诗敏	共青团员	汉族	2904585857@qq.com	1.589843e+10	430923200008177984
30	2020-12-01 00:00:00	2012-10-01	大学本科	学校	NaN	NaN	刘志庆	中共预备党员	汉族	3144338032@qq.com	1.561609e+10	33082220000603571X

```
df1 = pd.read_excel('D:/Python/Pandas/wode/grade.xlsx',sheet_name='Sheet2',index=['中国历史地理','公共关系','食品安全与健康','大学计算机基础','大学体育1','大学英语1'])#索引列不给列名即可
df1
```

	成绩1	学分	总学时	考核方式
（网络）中国历史地理概况	86	1.5	26	NaN
（网络）公共关系与人际交往能力	76	1.5	26	考试
食品安全与健康	92	1.5	26	考试
大学计算机基础	80	2.0	56	考试
大学体育1	82	1.0	34	考试
大学英语1	66	3.0	64	考试

```
df2 = pd.read_excel('D:/Python/Pandas/wode/grade.xlsx',sheet_name='Sheet3')#.set_index(['中国历史地理','公共关系','sb','大学计算机基础','大学体育1'])
df2
```

	成绩2	学分	已修学分
（网络）中国历史地理概况	86	1.5	26
（网络）公共关系与人际交往能力	96	4.0	64
食品安全与健康	89	3.0	54
大学计算机基础	80	2.0	56
大学英语	82	1.0	34

```
df = pd.merge(left=df1,right=df2,left_index=True,right_index=True,how='left')#inner,right,outer都可以试试
df
```

	成绩1	学分_x	总学时	考核方式	成绩2	学分_y	已修学分
（网络）中国历史地理概况	86	1.5	26	NaN	86.0	1.5	26.0
（网络）公共关系与人际交往能力	76	1.5	26	考试	96.0	4.0	64.0
食品安全与健康	92	1.5	26	考试	89.0	3.0	54.0
大学计算机基础	80	2.0	56	考试	80.0	2.0	56.0
大学体育1	82	1.0	34	考试	NaN	NaN	NaN
大学英语1	66	3.0	64	考试	NaN	NaN	NaN

```
df.dropna()#dropna 函数默认删除所有出现空值的行，即只要一行中任意一个字段为空，就会被删除。
```

	成绩1	学分_x	总学时	考核方式	成绩2	学分_y	已修学分
（网络）公共关系与人际交往能力	76	1.5	26	考试	96.0	4.0	64.0
食品安全与健康	92	1.5	26	考试	89.0	3.0	54.0
大学计算机基础	80	2.0	56	考试	80.0	2.0	56.0

```
df.dropna(subset=['考核方式'])#删除指定空值
```

	成绩1	学分_x	总学时	考核方式	成绩2	学分_y	已修学分
（网络）公共关系与人际交往能力	76	1.5	26	考试	96.0	4.0	64.0
食品安全与健康	92	1.5	26	考试	89.0	3.0	54.0
大学计算机基础	80	2.0	56	考试	80.0	2.0	56.0
大学体育1	82	1.0	34	考试	NaN	NaN	NaN
大学英语1	66	3.0	64	考试	NaN	NaN	NaN

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

	学分	学分_x	学分_y	已修学分	总学时	成绩1	成绩2	考核方式
（网络）中国历史地理概况	1.5	NaN	NaN	NaN	26.0	86.0	NaN	NaN
（网络）公共关系与人际交往能力	1.5	NaN	NaN	NaN	26.0	76.0	NaN	考试
食品安全与健康	1.5	NaN	NaN	NaN	26.0	92.0	NaN	考试
大学计算机基础	2.0	NaN	NaN	NaN	56.0	80.0	NaN	考试
大学体育1	1.0	NaN	NaN	NaN	34.0	82.0	NaN	考试
大学英语1	3.0	NaN	NaN	NaN	64.0	66.0	NaN	考试
（网络）中国历史地理概况	1.5	NaN	NaN	26.0	NaN	NaN	86.0	NaN
（网络）公共关系与人际交往能力	4.0	NaN	NaN	64.0	NaN	NaN	96.0	NaN
食品安全与健康	3.0	NaN	NaN	54.0	NaN	NaN	89.0	NaN
大学计算机基础	2.0	NaN	NaN	56.0	NaN	NaN	80.0	NaN
大学英语	1.0	NaN	NaN	34.0	NaN	NaN	82.0	NaN

	学分	学分_x	学分_y	已修学分	总学时	成绩1	成绩2	考核方式
（网络）中国历史地理概况	NaN	1.5	1.5	26.0	26.0	86.0	86.0	NaN
（网络）公共关系与人际交往能力	NaN	1.5	4.0	64.0	26.0	76.0	96.0	考试
食品安全与健康	NaN	1.5	3.0	54.0	26.0	92.0	89.0	考试
大学计算机基础	NaN	2.0	2.0	56.0	56.0	80.0	80.0	考试
大学体育1	NaN	1.0	NaN	NaN	34.0	82.0	NaN	考试
大学英语1	NaN	3.0	NaN	NaN	64.0	66.0	NaN	考试

```
df4 = df3.iloc[:,:2]
df4
```

	学分	学分_x
（网络）中国历史地理概况	1.5	NaN
（网络）公共关系与人际交往能力	1.5	NaN
食品安全与健康	1.5	NaN
大学计算机基础	2.0	NaN
大学体育1	1.0	NaN
大学英语1	3.0	NaN
（网络）中国历史地理概况	1.5	NaN
（网络）公共关系与人际交往能力	4.0	NaN
食品安全与健康	3.0	NaN
大学计算机基础	2.0	NaN
大学英语	1.0	NaN
（网络）中国历史地理概况	NaN	1.5
（网络）公共关系与人际交往能力	NaN	1.5
食品安全与健康	NaN	1.5
大学计算机基础	NaN	2.0
大学体育1	NaN	1.0
大学英语1	NaN	3.0

```
df4.drop_duplicates()#drop_duplicates 方法去重默认会删掉完全重复的行（每个值都一样的行），
```

	学分	学分_x
（网络）中国历史地理概况	1.5	NaN
大学计算机基础	2.0	NaN
大学体育1	1.0	NaN
大学英语1	3.0	NaN
（网络）公共关系与人际交往能力	4.0	NaN
（网络）中国历史地理概况	NaN	1.5
大学计算机基础	NaN	2.0
大学体育1	NaN	1.0

	学分	学分_x
大学英语1	NaN	3.0

```
df4.drop_duplicates(subset=['学分'])#指定列
```

	学分	学分_x
（网络）中国历史地理概况	1.5	NaN
大学计算机基础	2.0	NaN
大学体育1	1.0	NaN
大学英语1	3.0	NaN
（网络）公共关系与人际交往能力	4.0	NaN
（网络）中国历史地理概况	NaN	1.5

keep 值等于 last，保留最后一行数据，不输入 keep 值时，系统默认会给 keep 赋 值为 first，就会保留第一行数据而删掉其他的

```
df4.drop_duplicates(subset=['学分'],keep='last')
```

	学分	学分_x
（网络）中国历史地理概况	1.5	NaN
（网络）公共关系与人际交往能力	4.0	NaN
食品安全与健康	3.0	NaN
大学计算机基础	2.0	NaN
大学英语	1.0	NaN
大学英语1	NaN	3.0

```
# 按条件索引/筛选
df.loc[(df['学分_y']>2) & (df['学分_x']>0) ,:]
```

	成绩1	学分_x	总学时	考核方式	成绩2	学分_y	已修学分
（网络）公共关系与人际交往能力	76	1.5	26	考试	96.0	4.0	64.0
食品安全与健康	92	1.5	26	考试	89.0	3.0	54.0

```
df.sort_values('成绩1',ascending=False).head(3)
```

	成绩1	学分_x	总学时	考核方式	成绩2	学分_y	已修学分
食品安全与健康	92	1.5	26	考试	89.0	3.0	54.0
（网络）中国历史地理概况	86	1.5	26	NaN	86.0	1.5	26.0
大学体育1	82	1.0	34	考试	NaN	NaN	NaN

补充一个知识点，如果跟着文章操作，会发现无论是删空的 dropna，还是去重的 drop_duplicates，或者是排序的 sort_values，在对源数据进行操作后，源数据 并未改变，这是因为我们没有对这几个函数的 inplace 值进行设置，如果设置成 inplace = True，删空、去重和排序都会在源数据上生效。 但这里为了避免出现不必要的错误而无法更改，更建议大家把操作后的源数据赋值 给新的变量，如 new = df.dropna()，而不是将源数据的 inplace 参数设置为 True。

```
df.groupby('学分_x').max()#（常用的计算方法包括 sum、max、min、mean、std）
```

	成绩1	总学时	成绩2	学分_y	已修学分
学分_x					
1.0	82	34	NaN	NaN	NaN
1.5	92	26	96.0	4.0	64.0
2.0	80	56	80.0	2.0	56.0
3.0	66	64	NaN	NaN	NaN

```
df.groupby('学分_x')['总学时','成绩2'].sum()
```

	总学时	成绩2
学分_x		
1.0	34	0.0
1.5	78	271.0
2.0	56	80.0
3.0	64	0.0

作为汇总的依据列，默认转化为索引列，如果我们不希望它变成索引，向 groupby 内传入参数 as_index = False 即可

```
df.groupby('学分_x',as_index=False).max()
```

	学分_x	成绩1	总学时	成绩2	学分_y	已修学分
0	1.0	82	34	NaN	NaN	NaN
1	1.5	92	26	96.0	4.0	64.0
2	2.0	80	56	80.0	2.0	56.0
3	3.0	66	64	NaN	NaN	NaN

```
df = pd.read_excel('D:/Python/Pandas/wode/grade.xlsx').head(20)
df
```

	序号	开课学期	课程编号	课程名称	成绩	学分	总学时	绩点	考核方式	课程属性	课程性质
0	1	2018-2019-1	4130124	（网络）中国历史地理概况	86	1.5	26	3.6	考试	公选	通识选修课
1	2	2018-2019-1	4130125	（网络）公共关系与人际交往能力	76	1.5	26	2.6	考试	公选	通识选修课
2	3	2018-2019-1	4150035	食品安全与健康	92	1.5	26	4.2	考试	公选	通识选修课
3	4	2018-2019-1	4200002	大学计算机基础	80	2.0	56	3.0	考试	必修	公共必修课
4	5	2018-2019-1	4200004	大学体育1	82	1.0	34	3.2	考试	必修	公共必修课
5	6	2018-2019-1	4200015	大学英语1	66	3.0	64	1.6	考试	必修	公共必修课

	序号	开课学期	课程编号	课程名称	成绩	学分	总学时	绩点	考核方式	课程属性	课程性质
6	7	2018-2019-1	4200033	高等数学B1	96	4.0	64	4.6	考试	必修	学科基础课
7	8	2018-2019-1	4200044	思想道德修养与法律基础	89	3.0	54	3.9	考试	必修	公共必修课
8	9	2018-2019-1	4200046	就业指导-职业规划	87	0.5	8	3.7	考查	必修	公共必修课
9	10	2018-2019-1	4300196	管理学原理B	91	3.0	45	4.1	考试	必修	学科基础课
10	11	2018-2019-1	4300534	专业导论（电子商务）	90	1.0	16	4.0	考查	必修	学科基础课
11	12	2018-2019-2	4150105	茶成分与茶健康	90	1.5	26	4.0	考试	公选	通识选修课
12	13	2018-2019-2	4200003	大学生心理健康	89	1.0	16	3.9	考查	必修	公共必修课
13	14	2018-2019-2	4200005	大学体育2	60	1.0	34	1.0	考试	必修	公共必修课
14	15	2018-2019-2	4200016	大学英语2	74	3.0	64	2.4	考试	必修	公共必修课
15	16	2018-2019-2	4200021	动态网站设计	82	3.0	64	3.2	考试	必修	公共必修课
16	17	2018-2019-2	4200034	高等数学B2	63	4.0	64	1.3	考试	必修	学科基础课
17	18	2018-2019-2	4200039	军事理论	83	2.0	36	3.3	考查	必修	公共必修课
18	19	2018-2019-2	4200048	中国近现代史纲要	89	2.0	36	3.9	考试	必修	公共必修课
19	20	2018-2019-2	4300109	C程序设计	78	3.5	64	2.8	考试	必修	学科基础课

```
pd.cut(x = df['学分'],bins=[0,1.6,2.1,3.1,4.1])#,right=False 设置right 左闭右开
```

```
0     (0.0, 1.6]
1     (0.0, 1.6]
2     (0.0, 1.6]
3     (1.6, 2.1]
4     (0.0, 1.6]
5     (2.1, 3.1]
6     (3.1, 4.1]
7     (2.1, 3.1]
8     (0.0, 1.6]
9     (2.1, 3.1]
10    (0.0, 1.6]
11    (0.0, 1.6]
12    (0.0, 1.6]
13    (0.0, 1.6]
14    (2.1, 3.1]
15    (2.1, 3.1]
16    (3.1, 4.1]
17    (1.6, 2.1]
18    (1.6, 2.1]
19    (3.1, 4.1]
```

```
Name: 学分, dtype: category
Categories (4, interval[float64]): [(0.0, 1.6] < (1.6, 2.1] < (2.1, 3.1] < (3.1, 4.1]]
```

```
df['分类'] = pd.cut(x = df['学分'],bins=[0,1.6,2.1,3.1,4.1],labels=['D','C','B','A'],right=False )#非常高效，一行半代码就搞定了分组、判断和打标的过程
df
```

	序号	开课学期	课程编号	课程名称	成绩	学分	总学时	绩点	考核方式	课程属性	课程性质	分类
0	1	2018-2019-1	4130124	（网络）中国历史地理概况	86	1.5	26	3.6	考试	公选	通识选修课	D
1	2	2018-2019-1	4130125	（网络）公共关系与人际交往能力	76	1.5	26	2.6	考试	公选	通识选修课	D
2	3	2018-2019-1	4150035	食品安全与健康	92	1.5	26	4.2	考试	公选	通识选修课	D
3	4	2018-2019-1	4200002	大学计算机基础	80	2.0	56	3.0	考试	必修	公共必修课	C
4	5	2018-2019-1	4200004	大学体育1	82	1.0	34	3.2	考试	必修	公共必修课	D
5	6	2018-2019-1	4200015	大学英语1	66	3.0	64	1.6	考试	必修	公共必修课	B
6	7	2018-2019-1	4200033	高等数学B1	96	4.0	64	4.6	考试	必修	学科基础课	A
7	8	2018-2019-1	4200044	思想道德修养与法律基础	89	3.0	54	3.9	考试	必修	公共必修课	B
8	9	2018-2019-1	4200046	就业指导-职业规划	87	0.5	8	3.7	考查	必修	公共必修课	D
9	10	2018-2019-1	4300196	管理学原理B	91	3.0	45	4.1	考试	必修	学科基础课	B
10	11	2018-2019-1	4300534	专业导论（电子商务）	90	1.0	16	4.0	考查	必修	学科基础课	D
11	12	2018-2019-2	4150105	茶成分与茶健康	90	1.5	26	4.0	考试	公选	通识选修课	D
12	13	2018-2019-2	4200003	大学生心理健康	89	1.0	16	3.9	考查	必修	公共必修课	D
13	14	2018-2019-2	4200005	大学体育2	60	1.0	34	1.0	考试	必修	公共必修课	D
14	15	2018-2019-2	4200016	大学英语2	74	3.0	64	2.4	考试	必修	公共必修课	B
15	16	2018-2019-2	4200021	动态网站设计	82	3.0	64	3.2	考试	必修	公共必修课	B
16	17	2018-2019-2	4200034	高等数学B2	63	4.0	64	1.3	考试	必修	学科基础课	A
17	18	2018-2019-2	4200039	军事理论	83	2.0	36	3.3	考查	必修	公共必修课	C
18	19	2018-2019-2	4200048	中国近现代史纲要	89	2.0	36	3.9	考试	必修	公共必修课	C
19	20	2018-2019-2	4300109	C程序设计	78	3.5	64	2.8	考试	必修	学科基础课	A

```
df = pd.read_excel('D:/Python/Pandas/wode/grade.xlsx',sheet_name='Sheet4')
df.head(6)
```

	姓名	科目	综合成绩
0	李华	一模	651
1	李华	二模	579
2	李华	三模	580
3	王雷	一模	475
4	王雷	二模	455
5	王雷	三模	432

```
df.groupby('姓名')['综合成绩'].max().reset_index()
```

	姓名	综合成绩
0	发明	710
1	李华	651
2	泽民	678
3	王雷	475

```
df1 = df.groupby('姓名')['综合成绩'].apply(max).reset_index()
```

```
df2 = df.groupby('姓名')['综合成绩'].apply(min).reset_index() #指定一个列展示不会显示该列的列名，两个以上会
df2
```

	姓名	综合成绩
0	发明	568
1	李华	579
2	泽民	498
3	王雷	432

```
df3 = pd.merge(left=df1,right=df2,left_on='姓名',right_on='姓名',how='inner')#适用于每个表的相同列进行连接
df3
```

	姓名	综合成绩_x	综合成绩_y
0	发明	710	568
1	李华	651	579
2	泽民	678	498
3	王雷	475	432

```
df4 = pd.merge(left=df1,right=df2,left_index=True,right_index=True,how='inner')#直接连接，不给定以哪一列为标准
df4
```

	姓名_x	综合成绩_x	姓名_y	综合成绩_y
0	发明	710	发明	568
1	李华	651	李华	579
2	泽民	678	泽民	498
3	王雷	475	王雷	432

left_index 与 right_index 是当我们用索引（这两个表的名字在索引中）连接时指 定的参数，设置为 on 表示用该表的索引作为连接的条件（或者说桥梁）

```
df = pd.read_excel('D:/Python/Pandas/wode/grade.xlsx',sheet_name='Sheet5')
df
```

	省份	城市	资金总量
0	北京	北京	171062
1	上海	上海	132820
2	广东	深圳	83942
3	广东	广州	59131
4	浙江	杭州	45287
5	四川	成都	39828
6	重庆	重庆	39483
7	江苏	南京	35536
8	天津	天津	31788
9	江苏	苏州	31652

要求：返回每个省份资金总量排名第二的城市数据

```
def get(x):
    if len(x)<=1:
        return x.iloc[0,:]
    else:
        return x.iloc[1,:]#
```

```
df.groupby('省份')['城市','资金总量'].apply(get)
```

	城市	资金总量
省份		
上海	上海	132820
北京	北京	171062
四川	成都	39828
天津	天津	31788
广东	广州	59131
江苏	苏州	31652
浙江	杭州	45287
重庆	重庆	39483

由于数据较少，但是效果达到了

```
df.groupby('省份')['城市','资金总量']
```

```
<pandas.core.groupby.groupby.DataFrameGroupBy object at 0x000001FA6C868668>
```

这里面的x代表的是每个分组后的数据列，相当于每个面团，每个面团有n行数据，可以用len()统计总行数，需要多理解

```
import os
```

```
# df_all = pd.DataFrame()
# for name in os.listdir('D:/Python/data/'):
#     print(name)
#     df = pd.read_csv(f'D:/Python/data/{name}',engine='python',encoding='utf-8', index_col=0)
#     df_all = df_all.append(df)#这里必须给定对象，要不然df_all还是一个空dataframe对象，这一点跟列表有区别
# df_all.info()
```

```
a = []
for i in range(20):
    a.append(i)
a
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

```
#思路：1是将所有表的数据全部整合在数据清洗；2就是每个表先数据清洗然后再整合在清洗，思路2更符合逻辑，更具灵活性
```

```
df = pd.read_csv('D:/Python/Pandas/data/第一期下集.csv',engine='python',encoding='utf-8', index_col=0)
df.groupby('episodes')['oper_name','time_point'].apply(max).reset_index()
```

	episodes	time_point
0	第一期下	4175

```
df_all = pd.DataFrame()
for name in os.listdir('D:/Python/Pandas/data/'):
    print(name)
    df = pd.read_csv(f'D:/Python/Pandas/data/{name}',engine='python',encoding='utf-8', index_col=0)
    result = df.groupby('episodes')['oper_name','time_point'].apply(max).reset_index()
    df_all = pd.concat([df_all,result])#concat连接至少需要两个表
df_f = df_all.sort_values('time_point',ascending=False)
df_f
```

```
第一期上集.csv
第一期下集.csv
第三期上集.csv
第三期下集.csv
第二期上集.csv
第二期下集.csv
第五期上集.csv
第五期下集.csv
```

```
第四期上集.csv
第四期下集.csv
```

	episodes	time_point
0	第一期上	8239
0	第五期上	8158
0	第三期上	7838
0	第二期上	6767
0	第四期上	6616
0	第四期下	6572
0	第三期下	6450
0	第二期下	6037
0	第五期下	5652
0	第一期下	4175

这个案例还比较简单，明天想一个难一点的，至少要处理几步以上的

```
df = pd.read_excel('D:/Python/06 TGI指数分析实战/TGI/TGI指数案例数据.xlsx')
df.head()
```

	品牌名称	买家昵称	付款日期	订单状态	实付金额	邮费	省份	城市	购买数量
0	viva la vida	做快淘饭	2019-04-18 00:03:00	交易成功	22.32	0	北京	北京市	1
1	viva la vida	作自有世崇	2019-02-17 00:03:51	交易成功	87.00	0	上海	上海市	1
2	viva la vida	作雪白室	2019-04-18 00:01:43	交易成功	97.66	0	福建省	福州市	2
3	viva la vida	作美女购物主	2019-01-11 23:35:01	交易成功	37.23	0	河南省	安阳市	3
4	viva la vida	作美女购物主	2019-02-18 14:16:03	交易成功	29.50	0	河南省	安阳市	2

```
#分析哪些城市的人有高客单偏好，帮我筛选 5个
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28832 entries, 0 to 28831
Data columns (total 9 columns):
品牌名称    28832 non-null object
买家昵称    28832 non-null object
付款日期    28832 non-null datetime64[ns]
订单状态    28832 non-null object
实付金额    28832 non-null float64
邮费      28832 non-null int64
省份      28832 non-null object
城市      28832 non-null object
购买数量    28832 non-null int64
dtypes: datetime64[ns](1), float64(1), int64(2), object(5)
memory usage: 2.0+ MB
```

单次购买大于 50 元就算高客单的客户了 特征，高客单，即客户单次购买超过 50 元 • 目标群体，就是各个城市，这里我们可以分别计算出所有城市客户的高客单偏好 • 至于总体，就非常直白了，计算所涉及到的所有客户即为总体

```
df1 = df.groupby('买家昵称')['实付金额'].mean().reset_index()
df1.head()
```

	买家昵称	实付金额
0	.blue_ram	49.450
1	.christiny	22.000
2	.willn1	34.570
3	.托托m	37.475
4	0000妮	13.500

```
df1['客单类别'] = ['高客单' if i>50 else '低客单' for i in df1['实付金额']]
df1.head()
```

	买家昵称	实付金额	客单类别
0	.blue_ram	49.450	低客单
1	.christiny	22.000	低客单
2	.willn1	34.570	低客单
3	.托托m	37.475	低客单
4	0000妮	13.500	低客单

```
#df1['客单类别']
```

```
# def getif(x):
#     if x > 50:
#         return '高客单'
#     else:
#         return '低客单'
# df1['客单类别'] = df1['实付金额'].apply(getif)
# df1.head()
```

```
#df.duplicated('买家昵称')#去重，默认保存重复数据中的第一个，即first，如这里的3,4
```

```
df_dup = df.loc[df.duplicated('买家昵称') == False,:]
df_dup.head()
```

	品牌名称	买家昵称	付款日期	订单状态	实付金额	邮费	省份	城市	购买数量
0	viva la vida	做快淘饭	2019-04-18 00:03:00	交易成功	22.32	0	北京	北京市	1
1	viva la vida	作自有世崇	2019-02-17 00:03:51	交易成功	87.00	0	上海	上海市	1
2	viva la vida	作雪白室	2019-04-18 00:01:43	交易成功	97.66	0	福建省	福州市	2
3	viva la vida	作美女购物主	2019-01-11 23:35:01	交易成功	37.23	0	河南省	安阳市	3
5	viva la vida	作卢阳口才室	2019-06-16 04:15:56	交易成功	42.50	0	浙江省	衢州市	3

```
df_merge = pd.merge(left=df1,right=df_dup,left_on='买家昵称',right_on='买家昵称',how='left')#以左边的表为标准进行匹配
```

```
df_merge.head()
```

	买家昵称	实付金额_x	客单类别	品牌名称	付款日期	订单状态	实付金额_y	邮费	省份	城市	购买数量
0	.blue_ram	49.450	低客单	viva la vida	2019-02-04 17:49:34.000	交易成功	49.450	0	上海	上海市	1
1	.christiny	22.000	低客单	viva la vida	2019-01-29 14:17:15.000	交易成功	22.000	0	江苏省	南京市	1
2	.willn1	34.570	低客单	viva la vida	2019-01-11 03:46:18.000	交易成功	34.570	0	山东省	烟台市	2
3	.托托m	37.475	低客单	viva la vida	2019-01-11 02:26:33.000	交易成功	37.475	0	上海	上海市	3
4	0000妮	13.500	低客单	viva la vida	2019-06-28 16:53:26.458	交易成功	13.500	0	广东省	揭阳市	1

```
df_merge = df_merge[['买家昵称','客单类别','省份','城市']]
df_merge.head()
```

	买家昵称	客单类别	省份	城市
0	.blue_ram	低客单	上海	上海市
1	.christiny	低客单	江苏省	南京市
2	.willn1	低客单	山东省	烟台市
3	.托托m	低客单	上海	上海市
4	0000妮	低客单	广东省	揭阳市

```
#这里用到了透视表
result = pd.pivot_table(df_merge,index=['省份','城市'],columns='客单类别',aggfunc='count')#统计计数
result.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead tr th {
    text-align: left;
}

.dataframe thead tr:last-of-type th {
    text-align: right;
}
```

		买家昵称	
	客单类别	**低客单**	**高客单**
省份	**城市**		
上海	**上海市**	2818.0	2374.0
云南省	**临沧市**	3.0	2.0
	丽江市	1.0	3.0
	保山市	6.0	2.0

		买家昵称	
	客单类别	低客单	高客单
省份	城市		
	大理白族自治州	9.0	8.0

```
#层次化索引
result['买家昵称']['高客单'].reset_index().head()#这里是层次化索引带来的效果，省份和城市会自己带上耶
```

	省份	城市	高客单
0	上海	上海市	2374.0
1	云南省	临沧市	2.0
2	云南省	丽江市	3.0
3	云南省	保山市	2.0
4	云南省	大理白族自治州	8.0

```
#pd.merge横向合并,既可以是一列作为表，也可以是多列做表
tgi =  pd.merge(left=result['买家昵称']['高客单'].reset_index(),right=result['买家昵称']['低客
单'].reset_index(),left_on=['省份','城市'],right_on=['省份','城市'],how='inner')
tgi.head(10)
```

	省份	城市	高客单	低客单
0	上海	上海市	2374.0	2818.0
1	云南省	临沧市	2.0	3.0
2	云南省	丽江市	3.0	1.0
3	云南省	保山市	2.0	6.0
4	云南省	大理白族自治州	8.0	9.0
5	云南省	德宏傣族景颇族自治州	2.0	4.0
6	云南省	文山壮族苗族自治州	7.0	4.0
7	云南省	昆明市	71.0	100.0
8	云南省	昭通市	3.0	6.0
9	云南省	普洱市	6.0	7.0

```
tgi['客单总数'] = tgi['高客单']+tgi['低客单']
tgi['高客单占比'] = tgi['高客单']/tgi['客单总数']
tgi#.head()
```

	省份	城市	高客单	低客单	客单总数	高客单占比
0	上海	上海市	2374.0	2818.0	5192.0	0.457242
1	云南省	临沧市	2.0	3.0	5.0	0.400000
2	云南省	丽江市	3.0	1.0	4.0	0.750000
3	云南省	保山市	2.0	6.0	8.0	0.250000
4	云南省	大理白族自治州	8.0	9.0	17.0	0.470588
5	云南省	德宏傣族景颇族自治州	2.0	4.0	6.0	0.333333

	省份	城市	高客单	低客单	客单总数	高客单占比
6	云南省	文山壮族苗族自治州	7.0	4.0	11.0	0.636364
7	云南省	昆明市	71.0	100.0	171.0	0.415205
8	云南省	昭通市	3.0	6.0	9.0	0.333333
9	云南省	普洱市	6.0	7.0	13.0	0.461538
10	云南省	曲靖市	7.0	13.0	20.0	0.350000
11	云南省	楚雄彝族自治州	1.0	4.0	5.0	0.200000
12	云南省	玉溪市	6.0	7.0	13.0	0.461538
13	云南省	红河哈尼族彝族自治州	9.0	9.0	18.0	0.500000
14	云南省	西双版纳傣族自治州	3.0	7.0	10.0	0.300000
15	内蒙古自治区	乌兰察布市	6.0	5.0	11.0	0.545455
16	内蒙古自治区	乌海市	3.0	4.0	7.0	0.428571
17	内蒙古自治区	兴安盟	3.0	6.0	9.0	0.333333
18	内蒙古自治区	包头市	8.0	17.0	25.0	0.320000
19	内蒙古自治区	呼伦贝尔市	4.0	16.0	20.0	0.200000
20	内蒙古自治区	呼和浩特市	20.0	26.0	46.0	0.434783
21	内蒙古自治区	巴彦淖尔市	5.0	7.0	12.0	0.416667
22	内蒙古自治区	赤峰市	6.0	6.0	12.0	0.500000
23	内蒙古自治区	通辽市	3.0	5.0	8.0	0.375000
24	内蒙古自治区	鄂尔多斯市	3.0	12.0	15.0	0.200000
25	内蒙古自治区	锡林郭勒盟	2.0	1.0	3.0	0.666667
26	内蒙古自治区	阿拉善盟	1.0	2.0	3.0	0.333333
27	北京	北京市	1203.0	1298.0	2501.0	0.481008
28	吉林省	吉林市	8.0	17.0	25.0	0.320000
29	吉林省	四平市	3.0	8.0	11.0	0.272727
...	...	...	...	...	...	...
316	辽宁省	锦州市	9.0	11.0	20.0	0.450000
317	辽宁省	阜新市	1.0	6.0	7.0	0.142857
318	辽宁省	鞍山市	17.0	34.0	51.0	0.333333
319	重庆	重庆市	161.0	298.0	459.0	0.350763
320	陕西省	咸阳市	10.0	15.0	25.0	0.400000
321	陕西省	商洛市	1.0	4.0	5.0	0.200000
322	陕西省	安康市	3.0	3.0	6.0	0.500000
323	陕西省	宝鸡市	8.0	6.0	14.0	0.571429
324	陕西省	延安市	8.0	10.0	18.0	0.444444
325	陕西省	榆林市	11.0	18.0	29.0	0.379310
326	陕西省	汉中市	4.0	16.0	20.0	0.200000
327	陕西省	渭南市	5.0	9.0	14.0	0.357143
328	陕西省	西安市	111.0	200.0	311.0	0.356913
329	陕西省	铜川市	3.0	4.0	7.0	0.428571
330	青海省	海西蒙古族藏族自治州	1.0	3.0	4.0	0.250000

	省份	城市	高客单	低客单	客单总数	高客单占比
331	青海省	西宁市	6.0	12.0	18.0	0.333333
332	青海省	黄南藏族自治州	NaN	1.0	NaN	NaN
333	黑龙江省	七台河市	1.0	4.0	5.0	0.200000
334	黑龙江省	伊春市	3.0	5.0	8.0	0.375000
335	黑龙江省	佳木斯市	7.0	14.0	21.0	0.333333
336	黑龙江省	双鸭山市	2.0	6.0	8.0	0.250000
337	黑龙江省	哈尔滨市	85.0	132.0	217.0	0.391705
338	黑龙江省	大兴安岭地区	NaN	3.0	NaN	NaN
339	黑龙江省	大庆市	18.0	33.0	51.0	0.352941
340	黑龙江省	牡丹江市	4.0	12.0	16.0	0.250000
341	黑龙江省	绥化市	2.0	14.0	16.0	0.125000
342	黑龙江省	鸡西市	3.0	6.0	9.0	0.333333
343	黑龙江省	鹤岗市	2.0	1.0	3.0	0.666667
344	黑龙江省	黑河市	3.0	4.0	7.0	0.428571
345	黑龙江省	齐齐哈尔市	10.0	14.0	24.0	0.416667

346 rows × 6 columns

```
tgi.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 346 entries, 0 to 345
Data columns (total 6 columns):
省份        346 non-null object
城市        346 non-null object
高客单       332 non-null float64
低客单       329 non-null float64
客单总数      315 non-null float64
高客单占比     315 non-null float64
dtypes: float64(4), object(2)
memory usage: 18.9+ KB
```

```
tgi = tgi.dropna()#删除空值
tgi.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 315 entries, 0 to 345
Data columns (total 6 columns):
省份        315 non-null object
城市        315 non-null object
高客单       315 non-null float64
低客单       315 non-null float64
客单总数      315 non-null float64
高客单占比     315 non-null float64
dtypes: float64(4), object(2)
memory usage: 17.2+ KB
```

```
total_p = tgi['高客单'].sum()/tgi['客单总数'].sum()
total_p
```

```
0.41528303343887557
```

```
tgi['tgi指数'] = tgi['高客单占比']/total_p*100
tgi_fin = tgi.sort_values('tgi指数',ascending=False)
tgi_fin
```

	省份	城市	高客单	低客单	客单总数	高客单占比	tgi指数
149	新疆维吾尔自治区	哈密市	4.0	1.0	5.0	0.800000	192.639702
152	新疆维吾尔自治区	巴音郭楞蒙古自治州	10.0	3.0	13.0	0.769231	185.230483
2	云南省	丽江市	3.0	1.0	4.0	0.750000	180.599721
277	甘肃省	白银市	3.0	1.0	4.0	0.750000	180.599721
34	吉林省	辽源市	2.0	1.0	3.0	0.666667	160.533085
44	四川省	广安市	6.0	3.0	9.0	0.666667	160.533085
136	广西壮族自治区	河池市	4.0	2.0	6.0	0.666667	160.533085
25	内蒙古自治区	锡林郭勒盟	2.0	1.0	3.0	0.666667	160.533085
343	黑龙江省	鹤岗市	2.0	1.0	3.0	0.666667	160.533085
97	山西省	临汾市	9.0	5.0	14.0	0.642857	154.799761
6	云南省	文山壮族苗族自治州	7.0	4.0	11.0	0.636364	153.236127
240	湖北省	天门市	5.0	3.0	8.0	0.625000	150.499768
53	四川省	资阳市	6.0	4.0	10.0	0.600000	144.479777
146	新疆维吾尔自治区	克拉玛依市	6.0	4.0	10.0	0.600000	144.479777
323	陕西省	宝鸡市	8.0	6.0	14.0	0.571429	137.599787
140	广西壮族自治区	贺州市	4.0	3.0	7.0	0.571429	137.599787
210	河南省	鹤壁市	5.0	4.0	9.0	0.555556	133.777571
15	内蒙古自治区	乌兰察布市	6.0	5.0	11.0	0.545455	131.345252
125	广东省	肇庆市	13.0	11.0	24.0	0.541667	130.433132
79	安徽省	黄山市	7.0	6.0	13.0	0.538462	129.661338
191	河北省	邢台市	8.0	7.0	15.0	0.533333	128.426468
281	福建省	三明市	10.0	9.0	19.0	0.526316	126.736646
37	四川省	乐山市	11.0	10.0	21.0	0.523810	126.133139
287	福建省	福州市	145.0	135.0	280.0	0.517857	124.699807
225	海南省	临高县	2.0	2.0	4.0	0.500000	120.399814
251	湖北省	随州市	4.0	4.0	8.0	0.500000	120.399814
48	四川省	泸州市	16.0	16.0	32.0	0.500000	120.399814
143	新疆维吾尔自治区	乌鲁木齐市	30.0	30.0	60.0	0.500000	120.399814
13	云南省	红河哈尼族彝族自治州	9.0	9.0	18.0	0.500000	120.399814
129	广西壮族自治区	北海市	5.0	5.0	10.0	0.500000	120.399814
...	...	...	...	...	...	...	...

	省份	城市	高客单	低客单	客单总数	高客单占比	tgi指数
181	江西省	鹰潭市	1.0	4.0	5.0	0.200000	48.159926
187	河北省	沧州市	6.0	24.0	30.0	0.200000	48.159926
326	陕西省	汉中市	4.0	16.0	20.0	0.200000	48.159926
142	广西壮族自治区	防城港市	1.0	4.0	5.0	0.200000	48.159926
11	云南省	楚雄彝族自治州	1.0	4.0	5.0	0.200000	48.159926
260	湖南省	永州市	3.0	12.0	15.0	0.200000	48.159926
321	陕西省	商洛市	1.0	4.0	5.0	0.200000	48.159926
24	内蒙古自治区	鄂尔多斯市	3.0	12.0	15.0	0.200000	48.159926
333	黑龙江省	七台河市	1.0	4.0	5.0	0.200000	48.159926
19	内蒙古自治区	呼伦贝尔市	4.0	16.0	20.0	0.200000	48.159926
248	湖北省	荆门市	4.0	16.0	20.0	0.200000	48.159926
50	四川省	眉山市	2.0	8.0	10.0	0.200000	48.159926
110	广东省	云浮市	2.0	8.0	10.0	0.200000	48.159926
171	江西省	上饶市	4.0	17.0	21.0	0.190476	45.866596
64	安徽省	亳州市	2.0	9.0	11.0	0.181818	43.781751
289	福建省	龙岩市	4.0	19.0	23.0	0.173913	41.878196
102	山西省	晋中市	4.0	19.0	23.0	0.173913	41.878196
135	广西壮族自治区	梧州市	2.0	10.0	12.0	0.166667	40.133271
301	贵州省	铜仁市	1.0	5.0	6.0	0.166667	40.133271
296	贵州省	六盘水市	2.0	10.0	12.0	0.166667	40.133271
95	山东省	菏泽市	2.0	11.0	13.0	0.153846	37.046097
197	河南省	商丘市	2.0	12.0	14.0	0.142857	34.399947
153	新疆维吾尔自治区	昌吉回族自治州	1.0	6.0	7.0	0.142857	34.399947
317	辽宁省	阜新市	1.0	6.0	7.0	0.142857	34.399947
237	湖北省	仙桃市	1.0	7.0	8.0	0.125000	30.099954
341	黑龙江省	绥化市	2.0	14.0	16.0	0.125000	30.099954
117	广东省	汕尾市	2.0	15.0	17.0	0.117647	28.329368
30	吉林省	延边朝鲜族自治州	1.0	8.0	9.0	0.111111	26.755514
198	河南省	安阳市	2.0	18.0	20.0	0.100000	24.079963
298	贵州省	毕节市	1.0	10.0	11.0	0.090909	21.890875

315 rows × 7 columns

```
#TGI 指数能够显示偏好的强弱，但很容易让人忽略具体的样本量大小
```

```
tgi_fin = tgi_fin.loc[(tgi_fin['客单总数']>tgi_fin['客单总数'].mean()) & (tgi_fin['高客单']>tgi_fin['高客
单'].mean()) ,:].reset_index()
tgi_fin
```

	index	省份	城市	高客单	低客单	客单总数	高客单占比	tgi指数
0	287	福建省	福州市	145.0	135.0	280.0	0.517857	124.699807

	index	省份	城市	高客单	低客单	客单总数	高客单占比	tgi指数
1	124	广东省	珠海市	49.0	52.0	101.0	0.485149	116.823582
2	27	北京	北京市	1203.0	1298.0	2501.0	0.481008	115.826450
3	283	福建省	厦门市	105.0	118.0	223.0	0.470852	113.380991
4	111	广东省	佛山市	118.0	135.0	253.0	0.466403	112.309708
5	173	江西省	南昌市	63.0	73.0	136.0	0.463235	111.546887
6	46	四川省	成都市	287.0	334.0	621.0	0.462158	111.287429
7	0	上海	上海市	2374.0	2818.0	5192.0	0.457242	110.103682
8	164	江苏省	无锡市	135.0	162.0	297.0	0.454545	109.454376
9	120	广东省	深圳市	438.0	528.0	966.0	0.453416	109.182440
10	112	广东省	广州市	533.0	654.0	1187.0	0.449031	108.126539
11	216	浙江省	温州市	100.0	124.0	224.0	0.446429	107.499834
12	215	浙江省	杭州市	318.0	396.0	714.0	0.445378	107.246893
13	170	江苏省	镇江市	39.0	50.0	89.0	0.438202	105.518938
14	285	福建省	泉州市	57.0	77.0	134.0	0.425373	102.429693
15	244	湖北省	武汉市	275.0	373.0	648.0	0.424383	102.191200
16	267	湖南省	长沙市	108.0	149.0	257.0	0.420233	101.192062
17	87	山东省	济南市	80.0	111.0	191.0	0.418848	100.858483
18	159	江苏省	南通市	61.0	85.0	146.0	0.417808	100.608064
19	214	浙江省	宁波市	121.0	169.0	290.0	0.417241	100.471569
20	7	云南省	昆明市	71.0	100.0	171.0	0.415205	99.981132
21	306	辽宁省	大连市	94.0	133.0	227.0	0.414097	99.714383
22	212	浙江省	台州市	45.0	65.0	110.0	0.409091	98.508939
23	221	浙江省	金华市	51.0	74.0	125.0	0.408000	98.246248
24	130	广西壮族自治区	南宁市	54.0	79.0	133.0	0.406015	97.768270
25	158	江苏省	南京市	235.0	354.0	589.0	0.398981	96.074554
26	100	山西省	太原市	65.0	99.0	164.0	0.396341	95.438877
27	337	黑龙江省	哈尔滨市	85.0	132.0	217.0	0.391705	94.322435
28	113	广东省	惠州市	36.0	56.0	92.0	0.391304	94.225941
29	213	浙江省	嘉兴市	59.0	93.0	152.0	0.388158	93.468277
30	310	辽宁省	沈阳市	113.0	180.0	293.0	0.385666	92.868116
31	58	天津	天津市	203.0	335.0	538.0	0.377323	90.859339
32	161	江苏省	常州市	62.0	105.0	167.0	0.371257	89.398664
33	299	贵州省	贵阳市	47.0	80.0	127.0	0.370079	89.114823
34	168	江苏省	苏州市	183.0	312.0	495.0	0.369697	89.022893
35	66	安徽省	合肥市	76.0	131.0	207.0	0.367150	88.409525
36	328	陕西省	西安市	111.0	200.0	311.0	0.356913	85.944562
37	36	吉林省	长春市	49.0	89.0	138.0	0.355072	85.501317
38	319	重庆	重庆市	161.0	298.0	459.0	0.350763	84.463486
39	188	河北省	石家庄市	48.0	92.0	140.0	0.342857	82.559872
40	96	山东省	青岛市	90.0	174.0	264.0	0.340909	82.090782

	index	省份	城市	高客单	低客单	客单总数	高客单占比	tgi指数
41	218	浙江省	绍兴市	46.0	89.0	135.0	0.340741	82.050244
42	208	河南省	郑州市	78.0	159.0	237.0	0.329114	79.250511
43	108	广东省	东莞市	87.0	197.0	284.0	0.306338	73.766083

TGI指数就是某一群体中的特定人群占比 与 该群体在总人数中的占比 之间的占比

在这里就是：(高客单/该城市客单总数)/(所有城市高客单数/所有城市客单总数)*100 以100为基准 如果你想学好数据分析，那么这些东西你一定要尽快地吸收，TGI 在这里，需要确定高客单的判断标准，以及客单总数的最小样本要求，高客单的最小样本要求，因为会存在有些城市没有一个高客单，那这时就是一个空值

RFM 建模实战

R，Rencency，即每个客户有多少天没回购了，可以理解为最近一次购买到现在 隔了多少天。 • F，Frequency，是每个客户购买了多少次。 • M，Monetary，代表每个客户平均购买金额，这里也可以是累计购买金额。

在实际的业务场景中，一个用户在一天内的多次消费行为，应该从整体上看作一次

```
df = pd.read_excel('D:/Python/07 RFM建模实战/RFM/PYTHON-RFM实战数据.xlsx')
df.head()
```

	品牌名称	买家昵称	付款日期	订单状态	实付金额	邮费	省份	城市	购买数量
0	数据不吹牛	叫我李2	2019-01-01 00:17:59	交易成功	186	6	上海	上海市	1
1	数据不吹牛	0cyb1992	2019-01-01 00:59:54	交易成功	145	0	广东省	广州市	1
2	数据不吹牛	萝污萌莉	2019-01-01 07:48:48	交易成功	194	8	山东省	东营市	1
3	数据不吹牛	atblovemyy	2019-01-01 09:15:49	付款以后用户退款成功，交易自动关闭	84	0	江苏省	镇江市	1
4	数据不吹牛	小星期鱼	2019-01-01 09:59:33	付款以后用户退款成功，交易自动关闭	74	0	上海	上海市	1

```
df['订单状态'].unique()#发现只有两种订单状态
```

```
array(['交易成功', '付款以后用户退款成功，交易自动关闭'], dtype=object)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28833 entries, 0 to 28832
Data columns (total 9 columns):
品牌名称    28833 non-null object
买家昵称    28833 non-null object
付款日期    28833 non-null datetime64[ns]
订单状态    28833 non-null object
```

```
实付金额    28833 non-null int64
邮费      28833 non-null int64
省份      28833 non-null object
城市      28832 non-null object
购买数量    28833 non-null int64
dtypes: datetime64[ns](1), int64(3), object(5)
memory usage: 2.0+ MB
```

```
df = df.loc[df['订单状态']=='交易成功',:]
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 27793 entries, 0 to 28832
Data columns (total 9 columns):
品牌名称    27793 non-null object
买家昵称    27793 non-null object
付款日期    27793 non-null datetime64[ns]
订单状态    27793 non-null object
实付金额    27793 non-null int64
邮费      27793 non-null int64
省份      27793 non-null object
城市      27792 non-null object
购买数量    27793 non-null int64
dtypes: datetime64[ns](1), int64(3), object(5)
memory usage: 2.1+ MB
```

RFM模型只需要时间，客户，金额这个三个字段

```
df = df[['买家昵称','付款日期','实付金额']]
df.head(10)
```

	买家昵称	付款日期	实付金额
0	叫我李2	2019-01-01 00:17:59	186
1	0cyb1992	2019-01-01 00:59:54	145
2	萝污萌莉	2019-01-01 07:48:48	194
5	重碎叠	2019-01-01 10:00:07	197
6	iho_jann	2019-01-01 10:00:08	168
7	2jill27	2019-01-01 10:00:11	121
8	yjessieni	2019-01-01 10:00:14	211
9	4张洁85	2019-01-01 11:41:02	170
10	8tb4249_11	2019-01-01 12:08:49	124
11	一李一	2019-01-01 12:22:16	247

```
r = df.groupby('买家昵称')['付款日期'].max().reset_index()#要拿到所有用户最近一次付款时间，只需要按买家昵称分组，再选取付款日期的最大值,这里并没有统计每个用户一天的购物次数
# len(r)
r.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25420 entries, 0 to 25419
Data columns (total 2 columns):
```

```
买家昵称    25420 non-null object
付款日期    25420 non-null datetime64[ns]
dtypes: datetime64[ns](1), object(1)
memory usage: 397.3+ KB
```

```
pd.to_datetime('2019-9-9')>pd.to_datetime('2019-9-1')#要写成字符串哦，细心点
```

```
True
```

```
r['R'] = (pd.to_datetime('2019-7-1')-r['付款日期']).dt.days
r = r[['买家昵称','R']]
r.head()
```

	买家昵称	R
0	.blue_ram	146
1	.christiny	152
2	.willn1	170
3	.托托m	170
4	0000妮	2

```
df['日期标签'] = df['付款日期'].astype(str).str[:10]
# df
```

```
dup_f = df.groupby(['日期标签','买家昵称'])['付款日期'].count().reset_index()
# dup_f
f = dup_f.groupby('买家昵称')['付款日期'].count().reset_index()
f.columns = ['买家昵称','F']
f.head()
```

	买家昵称	F
0	.blue_ram	1
1	.christiny	1
2	.willn1	1
3	.托托m	1
4	0000妮	1

上一步计算出了每个用户购买频次，这里我们只需要得到每个用户总金额，再用总 金额除以购买频次，就能拿到用户平均支付金额

```
sum_m = df.groupby('买家昵称')['实付金额'].sum().reset_index()
sum_m.columns = ['买家昵称','总金额']
sum_m.head()
```

	买家昵称	总金额
0	.blue_ram	49
1	.christiny	183

	买家昵称	总金额
2	.willn1	34
3	.托托m	37
4	0000妮	164

```
com_m = pd.merge(sum_m,f,left_on='买家昵称',right_on='买家昵称',how='inner')
com_m['M'] = com_m['总金额']/com_m['F']
com_m.head()
```

	买家昵称	总金额	F	M
0	.blue_ram	49	1	49.0
1	.christiny	183	1	183.0
2	.willn1	34	1	34.0
3	.托托m	37	1	37.0
4	0000妮	164	1	164.0

```
rfm = pd.merge(com_m,r,left_on='买家昵称',right_on='买家昵称',how='inner')
rfm = rfm.iloc[:,[0,2,3,4]]
rfm.head()
```

	买家昵称	F	M	R
0	.blue_ram	1	49.0	146
1	.christiny	1	183.0	152
2	.willn1	1	34.0	170
3	.托托m	1	37.0	170
4	0000妮	1	164.0	2

以 R 值为例，R 代表了用户有多少天没来下单，这个值越大，用户流失的可能性越 大，我们当然不希望用户流失，所以 R 越大，分值越小。 • F 值代表了用户购买频次，M 值则是用户平均支付金额，这两个指标是越大越好， 即数值越大，得分越高。

```
#确定了一个打分框架
rfm['R-score'] = pd.cut(rfm['R'],bins=[0,30,60,90,120,100000],labels=[5,4,3,2,1],right=False).astype(float)
```

```
rfm['F-score'] = pd.cut(rfm['F'],bins=[1,2,3,4,5,100000],labels=[1,2,3,4,5],right=False).astype(float)
```

```
rfm['M-score'] = pd.cut(rfm['M'],bins=[0,50,100,150,200,100000],labels=[1,2,3,4,5],right=False).astype(float)
```

```
rfm.head()
```

	买家昵称	F	M	R	R-score	F-score	M-score
0	.blue_ram	1	49.0	146	1.0	1.0	1.0
1	.christiny	1	183.0	152	1.0	1.0	4.0
2	.willn1	1	34.0	170	1.0	1.0	1.0

	买家昵称	F	M	R	R-score	F-score	M-score
3	.托托m	1	37.0	170	1.0	1.0	1.0
4	0000妮	1	164.0	2	5.0	1.0	4.0

```
rfm['R是否大于均值'] = (rfm['R-score']>rfm['R-score'].mean())*1
rfm['F是否大于均值'] = (rfm['F-score']>rfm['F-score'].mean())*1
rfm['M是否大于均值'] = (rfm['M-score']>rfm['M-score'].mean())*1
```

```
rfm.head()
```

	买家昵称	F	M	R	R-score	F-score	M-score	R是否大于均值	F是否大于均值	M是否大于均值
0	.blue_ram	1	49.0	146	1.0	1.0	1.0	0	0	0
1	.christiny	1	183.0	152	1.0	1.0	4.0	0	0	1
2	.willn1	1	34.0	170	1.0	1.0	1.0	0	0	0
3	.托托m	1	37.0	170	1.0	1.0	1.0	0	0	0
4	0000妮	1	164.0	2	5.0	1.0	4.0	1	0	1

```
rfm['人均数值'] = (rfm['R是否大于均值']*100)+(rfm['F是否大于均值']*10)+(rfm['M是否大于均值']*1)
rfm.head()
```

	买家昵称	F	M	R	R-score	F-score	M-score	R是否大于均值	F是否大于均值	M是否大于均值	人均数值
0	.blue_ram	1	49.0	146	1.0	1.0	1.0	0	0	0	0
1	.christiny	1	183.0	152	1.0	1.0	4.0	0	0	1	1
2	.willn1	1	34.0	170	1.0	1.0	1.0	0	0	0	0
3	.托托m	1	37.0	170	1.0	1.0	1.0	0	0	0	0
4	0000妮	1	164.0	2	5.0	1.0	4.0	1	0	1	101

```
def get_kehu(x):
    if x == 111:
        return '重要价值客户'
    elif x == 110:
        return '消费潜力用户'
    elif x == 101:
        return '频次深耕用户'
    elif x == 100:
        return '新客户'
    elif x == 11:
        return '重要价值流失预警用户'
    elif x == 10:
        return '一般用户'
    elif x == 1:
        return '高消费唤回用户'
    elif x == 0:
        return '流失客户'
rfm['用户类别'] = rfm['人均数值'].apply(get_kehu)
rfm.head()
```

	买家昵称	F	M	R	R-score	F-score	M-score	R是否大于均值	F是否大于均值	M是否大于均值	人均数值	用户类别
0	.blue_ram	1	49.0	146	1.0	1.0	1.0	0	0	0	0	流失客户

	买家昵称	F	M	R	R-score	F-score	M-score	R是否大于均值	F是否大于均值	M是否大于均值	人均数值	用户类别
1	.christiny	1	183.0	152	1.0	1.0	4.0	0	0	1	1	高消费唤回用户
2	.willn1	1	34.0	170	1.0	1.0	1.0	0	0	0	0	流失客户
3	.托托m	1	37.0	170	1.0	1.0	1.0	0	0	0	0	流失客户
4	0000妮	1	164.0	2	5.0	1.0	4.0	1	0	1	101	频次深耕用户

```
count = rfm['用户类别'].value_counts().reset_index()
count.columns = ['客户类型','人数']
count['人数占比'] = count['人数']/count['人数'].sum()
count.head()
```

	客户类型	人数	人数占比
0	高消费唤回用户	7338	0.288670
1	流失客户	6680	0.262785
2	频次深耕用户	5427	0.213493
3	新客户	4224	0.166168
4	重要价值客户	756	0.029740

```
rfm['总金额'] = rfm['F']*rfm['M']
mon = rfm.groupby('用户类别')['总金额'].sum().reset_index()
mon.columns = ['客户类型','消费金额']
mon['金额占比'] = mon['消费金额']/mon['消费金额'].sum()
mon.head()
```

	客户类型	消费金额	金额占比
0	一般用户	25803.0	0.007349
1	新客户	270869.0	0.077142
2	流失客户	444617.0	0.126624
3	消费潜力用户	64075.0	0.018248
4	重要价值客户	269230.0	0.076675

```
#图表自己画哦,今天感觉很是有点没思路，但是多看几遍掌握这种能力以后工作就基本有指望了
```

同期群分析实战

```

```