In [ ]:

```
df2.reindex()
```

In [ ]:

```
df2.reindex([1,2,99,101])
```

In [ ]:

```
df2.reindex([1,2,99,101], method = 'ffill' )
```

In [ ]:

```
df2.reindex([1,2,99,101], fill_value= "不知道" )
```

In [ ]:

```
df2.reindex([1,2,99,101], fill_value= "不知道" ).dtypes
```

4.4 实战：为PM2.5数据建立索引

要求:

尝试在读入文件时直接建立索引
尝试使用Date (LST)建立单一索引
    提示：为便于操作，最好先重命名
尝试使用Year,Month,Day,Hour建立复合索引
尝试修改索引名

5 案例行的基本操作

5.1 案例排序

5.1.1 用索引排序

df.sort_index(

level ： （多重索引时）指定用于排序的级别顺序号/名称
ascending = True ： 是否为升序排列，多列时以表形式提供
inplace = False ：
na_position = 'last': 缺失值的排列顺序，first/last

)

In [ ]:

```
df2 = pd.read_excel("高校信息.xlsx", sheet_name = 0)
df2.set_index(['类型','学校名称'], inplace = True)
df2
```

In [ ]:

```
df2.sort_index()
```

In [ ]:

```
df2.sort_index(ascending = [True, False])
```

In [ ]:

```
df2.sort_index(level = '学校名称')
```

5.1.2 用变量值排序

df.sort_values(

by : 指定用于排序的变量名，多列时以列表形式提供
ascending = True : 是否为升序排列
inplace = False :
na_position = 'last': 缺失值的排列顺序，first/last

)

In [ ]:

```
df2.sort_values(['所在省份','所在城市'],0, False)
```

5.2 案例筛选

筛选操作的实质：基于T/F值进行筛选

In [ ]:

```
sellist = [True, False, True]
df2.iloc[sellist]
```

5.2.1 按照绝对位置进行筛选

df.iloc

意为integer-location，即按照行列序号进行检索
可以同时指定行列，指定列时，需要先用","表明为列序号

df.iat

本质上和iloc是一回事，可以看作别名

In [ ]:

```
df2.iloc[0:3] # 不包括右侧界值
```

In [ ]:

```
df2.iloc[[0,3]]
```

In [ ]:

```
df2.iloc[:,0:3] # 只指定列范围，不包括右侧边界
```

In [ ]:

```
df2.iloc[1:4,[0,3]] # 同时指定行列范围
```

5.2.2 按照索引值进行筛选

df.loc

按照给出的索引值进行筛选
筛选范围包括上下界值
出现未知索引值时会报错

df.at

本质上和loc是一回事，可以看作别名

In [ ]:

```
df2 = pd.read_excel("高校信息.xlsx",
                    sheet_name = "full", index_col = '名次')
df2
```

In [ ]:

```
df2.loc[2:4] # 切片本身就是一个列表，因此不需要加[]
```

In [ ]:

```
df2 = pd.read_excel("高校信息.xlsx",
                    sheet_name = "full", index_col = '学校名称')
df2
```

In [ ]:

```
df2.loc['北京大学':'复旦大学']
```

In [ ]:

```
df2.loc[['北京大学', '复旦大学']]
```

In [ ]:

```
df2.loc[['北京大学','复旦大学'],['名次','所在省份','总分']]
```

针对多重索引的行筛选命令

df.xs(key, axis=0, level=None, drop_level=True)
    指定具体检索用的多重索引级别
df.IndexSlice
    针对多重索引的花式作死使用方式

In [ ]:

```
df2 = pd.read_excel("高校信息.xlsx",
                    sheet_name = "full", index_col = [3, 1])
df2
```

In [ ]:

```
df2.loc[['财经','师范']] # 只使用最高级别的索引检索
```

使用loc命令进行筛选，多重索引组和筛选时按顺序输入即可

In [ ]:

```
df2.loc[[('财经','中央财经大学'),('综合','北京大学')]] # 完整检索
```

In [ ]:

```
df2.xs('中央财经大学', level = 1, drop_level = False)
```

5.2.3 使用混合模式进行筛选

df.ix

首先使用索引值进行筛选，无索引值匹配时，改用行列序号进行筛选。
这种自动选择带来了很多用户使用上的困扰，因此Pandas官方已经决定取消该命令

In [ ]:

```
df2.reset_index().ix[0:4, ['名次','所在省份','总分']]
```

5.2.4 直接进行条件筛选

df[筛选条件]

5.2.4.1 按照数据范围进行筛选

In [ ]:

```
df2[df2.名次 > 10]
```

5.2.4.2 列表筛选

df.isin(values)

返回结果为相应的位置是否匹配给出的values
values为序列：对应每个具体值
values为字典：对应各个变量名称
values为数据框：同时对应数值和变量名称

In [ ]:

```
df2.名次.isin([1, 3, 5])
```

In [ ]:

```
df2[df2.名次.isin([1, 3, 5])]
```

In [ ]:

```
df2[df2.所在省份.isin(['北京','上海'])]
```

In [ ]:

```
df2 = pd.read_excel("高校信息.xlsx",
                    sheet_name = "full", index_col = '类型')
df2[df2.index.isin(['财经', '师范'])]
```

5.2.4.3 多重条件的联合筛选

In [ ]:

```
df2[df2.名次 > 10][df2[df2.名次 > 10].名次 < 90]
```

In [ ]:

```
df3 = df2[df2.名次 > 10]
df3[df3.名次 < 90]
```

5.2.5 用类SQL语句进行筛选

df.query(

expr : 类SQL语句表达式
可以使用前缀'@'引用环境变量
等号为==，而不是=
注意：目前还不支持like语句
inplace = False : 是否直接替换原数据框

)

用法举例：

query('(a < b) & (b < c)')
如果索引没有名称，df.query('index < b < c')
可以进行多重索引的指定，如df.query("code=='600801'")

In [ ]:

```
df2.query("名次 >10 and 名次 < 90 and 类型 != '综合'")
```

In [ ]:

```
limit = 5
df2.query("名次<=@limit & 类型 == '综合'")
```

In [ ]:

```
df2.query("名次<=@limit & 类型 != '综合'")
```

5.3 实战：筛选数据中所需的案例

高校信息数据：

分别使用索引、非索引和类SQL方式，筛选出 教育部主管的总分低于70分的大学
只将主管部门设定为索引，重新实现上述需求

北京PM2.5数据：

筛选出 PM2.5 > 200 的案例
筛选出 2016年10月，PM2.5 > 100 的案例
筛选出 2016年10月上班时间（9:00AM-5:00PM），PM2.5 > 100 的案例

6 变量变换

6.1 计算新变量

6.1.1 新变量为常数

df['varname'] = value

In [ ]:

```
# df2.newvar = 1 # 注意该命令不会报错!
df2['cons'] = 1
df2
```

6.1.2 新变量基于一个/多个原变量做简单四则运算

df['varname'] = df['oldvar'] * 100

df['varname'] = df.oldvar * 100

Q：课程学习遇到不懂怎么办？

- ✧ 本课程提供额外福利：QQ 群供学员交流心得，群号：630030855，可直接扫下方的二维码进入。
- ✧ 老师有空时也会参与讨论，但请不要把你的工作问题直接让老师解决。
- ✧ 老师鼓励学员多思考、多动手，通过自己努力解决问题，这样能发现乐趣、有成就感、成长快。