**pandas用法大全**

2017年09月05日 21:01:40 [liufang0001](https://me.csdn.net/liufang0001) 阅读数：108324 标签： [pandas](http://so.csdn.net/so/search/s.do?q=pandas&t=blog) 更多

个人分类： [python](https://blog.csdn.net/liufang0001/article/category/6688808)

一、生成数据表   
1、首先导入pandas库，一般都会用到numpy库，所以我们先导入备用：

import numpy as np

import pandas as pd

* 1
* 2

2、导入CSV或者xlsx文件：

df = pd.DataFrame(pd.read\_csv('name.csv',header=1))

df = pd.DataFrame(pd.read\_excel('name.xlsx'))

* 1
* 2

3、用pandas创建数据表：

df = pd.DataFrame({"id":[1001,1002,1003,1004,1005,1006],

"date":pd.date\_range('20130102', periods=6),

"city":['Beijing ', 'SH', ' guangzhou ', 'Shenzhen', 'shanghai', 'BEIJING '],

"age":[23,44,54,32,34,32],

"category":['100-A','100-B','110-A','110-C','210-A','130-F'],

"price":[1200,np.nan,2133,5433,np.nan,4432]},

columns =['id','date','city','category','age','price'])

* 1
* 2
* 3
* 4
* 5
* 6
* 7

2、数据表信息查看   
1、维度查看：

df.shape

* 1

2、数据表基本信息（维度、列名称、数据格式、所占空间等）：

df.info()

* 1

3、每一列数据的格式：

df.dtypes

* 1

4、某一列格式：

df['B'].dtype

* 1

5、空值：

df.isnull()

* 1

6、查看某一列空值：

df.isnull()

* 1

7、查看某一列的唯一值：

df['B'].unique()

* 1

8、查看数据表的值：   
df.values   
9、查看列名称：

df.columns

* 1

10、查看前10行数据、后10行数据：

df.head() #默认前10行数据

df.tail() #默认后10 行数据

* 1
* 2

三、数据表清洗   
1、用数字0填充空值：

df.fillna(value=0)

* 1

2、使用列prince的均值对NA进行填充：

df['prince'].fillna(df['prince'].mean())

* 1

3、清楚city字段的字符空格：

df['city']=df['city'].map(str.strip)

* 1

4、大小写转换：

df['city']=df['city'].str.lower()

* 1

5、更改数据格式：

df['price'].astype('int')

* 1

6、更改列名称：

df.rename(columns={'category': 'category-size'})

* 1

7、删除后出现的重复值：

df['city'].drop\_duplicates()

* 1

8、删除先出现的重复值：

df['city'].drop\_duplicates(keep='last')

* 1

9、数据替换：

df['city'].replace('sh', 'shanghai')

* 1

四、数据预处理

df1=pd.DataFrame({"id":[1001,1002,1003,1004,1005,1006,1007,1008],

"gender":['male','female','male','female','male','female','male','female'],

"pay":['Y','N','Y','Y','N','Y','N','Y',],

"m-point":[10,12,20,40,40,40,30,20]})

* 1
* 2
* 3
* 4

1、数据表合并

df\_inner=pd.merge(df,df1,how='inner') # 匹配合并，交集

df\_left=pd.merge(df,df1,how='left') #

df\_right=pd.merge(df,df1,how='right')

df\_outer=pd.merge(df,df1,how='outer') #并集

* 1
* 2
* 3
* 4

2、设置索引列

df\_inner.set\_index('id')

* 1

3、按照特定列的值排序：

df\_inner.sort\_values(by=['age'])

* 1

4、按照索引列排序：

df\_inner.sort\_index()

* 1

5、如果prince列的值>3000，group列显示high，否则显示low：

df\_inner['group'] = np.where(df\_inner['price'] > 3000,'high','low')

* 1

6、对复合多个条件的数据进行分组标记

df\_inner.loc[(df\_inner['city'] == 'beijing') & (df\_inner['price'] >= 4000), 'sign']=1

* 1

7、对category字段的值依次进行分列，并创建数据表，索引值为df\_inner的索引列，列名称为category和size

pd.DataFrame((x.split('-') for x in df\_inner['category']),index=df\_inner.index,columns=['category','size']))

* 1

8、将完成分裂后的数据表和原df\_inner数据表进行匹配

df\_inner=pd.merge(df\_inner,split,right\_index=True, left\_index=True)

* 1

五、数据提取   
主要用到的三个函数：loc,iloc和ix，loc函数按标签值进行提取，iloc按位置进行提取，ix可以同时按标签和位置进行提取。   
1、按索引提取单行的数值

df\_inner.loc[3]

* 1

2、按索引提取区域行数值

df\_inner.iloc[0:5]

* 1

3、重设索引

df\_inner.reset\_index()

* 1

4、设置日期为索引

df\_inner=df\_inner.set\_index('date')

* 1

5、提取4日之前的所有数据

df\_inner[:'2013-01-04']

* 1

6、使用iloc按位置区域提取数据

df\_inner.iloc[:3,:2] #冒号前后的数字不再是索引的标签名称，而是数据所在的位置，从0开始，前三行，前两列。

* 1

7、适应iloc按位置单独提起数据

df\_inner.iloc[[0,2,5],[4,5]] #提取第0、2、5行，4、5列

* 1

8、使用ix按索引标签和位置混合提取数据

df\_inner.ix[:'2013-01-03',:4] #2013-01-03号之前，前四列数据

* 1

9、判断city列的值是否为北京

df\_inner['city'].isin(['beijing'])

* 1

10、判断city列里是否包含beijing和shanghai，然后将符合条件的数据提取出来

df\_inner.loc[df\_inner['city'].isin(['beijing','shanghai'])]

* 1

11、提取前三个字符，并生成数据表

pd.DataFrame(category.str[:3])

* 1

六、数据筛选   
使用与、或、非三个条件配合大于、小于、等于对数据进行筛选，并进行计数和求和。   
1、使用“与”进行筛选

df\_inner.loc[(df\_inner['age'] > 25) & (df\_inner['city'] == 'beijing'), ['id','city','age','category','gender']]

* 1

2、使用“或”进行筛选

df\_inner.loc[(df\_inner['age'] > 25) | (df\_inner['city'] == 'beijing'), ['id','city','age','category','gender']].sort(['age'])

* 1

3、使用“非”条件进行筛选

df\_inner.loc[(df\_inner['city'] != 'beijing'), ['id','city','age','category','gender']].sort(['id'])

* 1

4、对筛选后的数据按city列进行计数

df\_inner.loc[(df\_inner['city'] != 'beijing'), ['id','city','age','category','gender']].sort(['id']).city.count()

* 1

5、使用query函数进行筛选

df\_inner.query('city == ["beijing", "shanghai"]')

* 1

6、对筛选后的结果按prince进行求和

df\_inner.query('city == ["beijing", "shanghai"]').price.sum()

* 1

七、数据汇总   
主要函数是groupby和pivote\_table   
1、对所有的列进行计数汇总

df\_inner.groupby('city').count()

* 1

2、按城市对id字段进行计数

df\_inner.groupby('city')['id'].count()

* 1

3、对两个字段进行汇总计数

df\_inner.groupby(['city','size'])['id'].count()

* 1

4、对city字段进行汇总，并分别计算prince的合计和均值

df\_inner.groupby('city')['price'].agg([len,np.sum, np.mean])

* 1

八、数据统计   
数据采样，计算标准差，协方差和相关系数   
1、简单的数据采样

df\_inner.sample(n=3)

* 1

2、手动设置采样权重

weights = [0, 0, 0, 0, 0.5, 0.5]

df\_inner.sample(n=2, weights=weights)

* 1
* 2

3、采样后不放回

df\_inner.sample(n=6, replace=False)

* 1

4、采样后放回

df\_inner.sample(n=6, replace=True)

* 1

5、 数据表描述性统计

df\_inner.describe().round(2).T #round函数设置显示小数位，T表示转置

* 1

6、计算列的标准差

df\_inner['price'].std()

* 1

7、计算两个字段间的协方差

df\_inner['price'].cov(df\_inner['m-point'])

* 1

8、数据表中所有字段间的协方差

df\_inner.cov()

* 1

9、两个字段的相关性分析

df\_inner['price'].corr(df\_inner['m-point']) #相关系数在-1到1之间，接近1为正相关，接近-1为负相关，0为不相关

* 1

10、数据表的相关性分析

df\_inner.corr()

* 1

九、数据输出   
分析后的数据可以输出为xlsx格式和csv格式   
1、写入Excel

df\_inner.to\_excel('excel\_to\_python.xlsx', sheet\_name='bluewhale\_cc')

* 1

2、写入到CSV

df\_inner.to\_csv('excel\_to\_python.csv')