05-Pandas常见操作

1.DataFrame修改index、columns

Python

```python
import pandas as pd

data = {'name': ['Tom', 'Nick', 'John'], 'Age': [20, 21, 19]}
df = pd.DataFrame(data)
df.index = ['name', 'Age', 'height']
df.columns = ['Name', 'Age']
df
```

2.DataFrame常见操作

```python
import pandas as pd

# 创建 DataFrame
data = {'name': ['Tom', 'Nick', 'John'], 'Age': [20, 21, 19]}
df = pd.DataFrame(data)
df.index = ['name', 'Age', 'height']
df.columns = ['Name', 'Age']
df
data = {'name': ['Tom', 'Nick', 'John'], 'Age': [20, 21, 19]}
df = pd.DataFrame(data)
print(df)

print(df)
# 根据条件筛选数据
df['Age'] = df['Age'].astype(int)
df[df['Age'] > 20]
# 修改数据
df.loc[1, 'Age'] = 25  # 修改第二行 'Age' 列的值
print(df)
# 排序数据
df.sort_values('Age', inplace=True)  # 根据 'Age' 列的值进行排序，并修改原数据
print(df)
```

3.数据合并

1、concat: concat 函数用于沿着一条轴（通常是行轴）连接两个或更多的对象。它不会改变轴标签。默认情况下，concat 沿着行方向（axis=0）连接对象。

```python
import pandas as pd

# 创建两个 DataFrame
df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2'],
                    'B': ['B0', 'B1', 'B2'],
                    'C': ['C0', 'C1', 'C2'],
                    'D': ['D0', 'D1', 'D2']},
                   index=[0, 1, 2])
df2 = pd.DataFrame({'A': ['A3', 'A4', 'A5'],
                    'B': ['B3', 'B4', 'B5'],
                    'C': ['C3', 'C4', 'C5'],
                    'D': ['D3', 'D4', 'D5']},
                   index=[3, 4, 5])
result = pd.concat([df1, df2])
print(result)
```

2、**join**: DataFrame 中的 join 方法用于将两个 DataFrame 连接在一起，类似于 SQL 中的 INNER JOIN。连接是基于索引（行标签）进行的。

```python
# 创建两个 DataFrame
df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2'],
                    'B': ['B0', 'B1', 'B2']},
                   index=[0, 1, 2])
df2 = pd.DataFrame({'C': ['C0', 'C1', 'C2'],
                    'D': ['D0', 'D1', 'D2']},
                   index=[0, 1, 2])
result = df1.join(df2, how='inner') # how='inner' 表示内连接，即只保留两个 DataFrame 中都有数据的行
print(result)
```

3、merge: merge 函数用于将两个 DataFrame 沿着一个或多个键（可以是行或列标签）进行连接，类似于 SQL 中的 JOIN。与 join 不同，merge 更灵活，允许你指定连接的键，还可以进行左连接、右连接或外连接。

```python
# 创建两个 DataFrame
df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2'],
                    'B': ['B0', 'B1', 'B2']},
                   index=[0, 1, 2])
df2 = pd.DataFrame({'C': ['C0', 'C1', 'C2'],
                    'D': ['D0', 'D1', 'D2']},
                   index=[0, 1, 3])

result = pd.merge(df1, df2, left_index=True, right_index=True) # left_index=True 表示进行左连接，即保留 df1 中的所有行，右连接同理。默认情况下，两个 DataFrame 的索引都会被保留下来。如果不想保留某个 DataFrame 的索引，可以使用 how='inner' 或 how='outer' 进行内连接或外连接。
print(result)
```

4.数据删除

```python
pandas.drop(labels=None, axis=0, index=None, columns=None, level=None, inplace=False, errors='raise')
```

参数说明：

- labels：要删除的行或列的标签。
- axis：删除行还是列，0 或 'index' 代表行，1 或 'columns' 代表列。
- index：要删除的行的索引。
- columns：要删除的列的标签。

- level：用于多层次索引（MultiIndex）。
- inplace：是否在原地修改 DataFrame。如果是 True，则直接在原 DataFrame 上进行修改，否则返回一个新的 DataFrame。
- errors：如果尝试删除不存在的标签会引发什么错误，'raise' 或 'ignore'。

```python
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6], 'C': [7, 8, 9]})
df.drop(columns='A', inplace=True)
df
```

5.创建多层索引

创建索引对象：使用 pd.MultiIndex.from_arrays() 方法创建多级索引对象。该方法接受两个列表作为参数，分别表示索引的级别。例如，下面的代码创建了一个两级索引对象，第一级为 ['A', 'A', 'B', 'B']，第二级为 [1, 2, 1, 2]：

```python
index = pd.MultiIndex.from_arrays([['A', 'A', 'B', 'B'], [1, 2, 1, 2]],
                                  names=['letters', 'numbers'])
df = pd.DataFrame({'col1': [10, 20, 30, 40]}, index=index)
df
result = df.loc['A']  # 选择第一级索引为 'A' 的数据
print(result)
result = df.loc[('A', 2)]  # 选择第一级索引为 'A'，第二级索引为 2 的数据
print(result)
```

6.数据对齐

```python
import pandas as pd

# 创建两个 Series 对象
s1 = pd.Series([1, 2, 3])
s2 = pd.Series([4, 5, 6])
# 算术运算
add_result = s1 + s2
sub_result = s1 - s2
mul_result = s1 * s2
div_result = s1 / s2
print("加法结果: ", add_result)
print("减法结果: ", sub_result)
print("乘法结果: ", mul_result)
print("除法结果: ", div_result)
```

7.DataFrame和Series之间的运算

在Pandas库中，DataFrame和Series都是进行数据操作的基本单位。它们之间可以方便地进行算术运算，包括加法、减法、乘法、除法等。

这些运算的规则是，对于两个相同大小的DataFrame或Series对象，它们会根据每个位置的索引值进行对应运算，并将结果保存在一个新的DataFrame或Series对象中。

```python
import pandas as pd

# 创建两个 DataFrame 对象
df1 = pd.DataFrame({'A': [1, 2], 'B': [3, 4]}, index=['x', 'y'])
df2 = pd.DataFrame({'A': [5, 6], 'B': [7, 8]}, index=['x', 'y'])
# 加法运算
add_result = df1 + df2
print(add_result)
```

类似地，我们也可以进行减法、乘法、除法等运算。