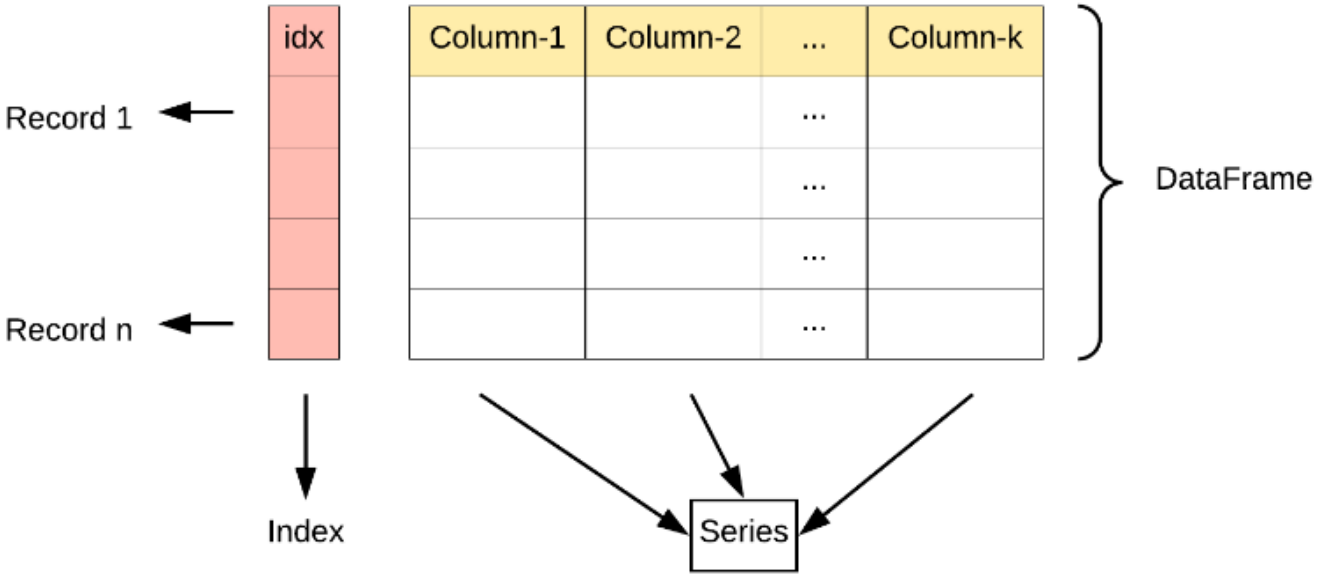
Series()

```
pandas.Series(data , index , dtype , name , copy)
```

- data : array
- index
- dtype
- name
- copy:(default:false)



DataFrame()

pandas.DataFrame(data, index, columns, dtype, copy)

Pandas 可以使用 `loc` 属性返回指定行的数据，如果没有设置索引，第一行索引为 **0**，第二行索引为 **1**，以此类推：

实例

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

# 数据载入到 DataFrame 对象
df = pd.DataFrame(data)

# 返回第一行
print(df.loc[0])
# 返回第二行
print(df.loc[1])
```

输出结果如下：

```
calories    420
duration     50
Name: 0, dtype: int64
calories    380
duration     40
Name: 1, dtype: int64
```

read_csv

head(n)-读取前面 n 行。default5 行。

tail(n)-读取尾部 n 行。

read_json()

nested_list.json 文件内容

```
{
    "school_name": "ABC primary school",
    "class": "Year 1",
    "students": [
    {
        "id": "A001",
        "name": "Tom",
        "math": 60,
        "physics": 66,
        "chemistry": 61
    },
    {
        "id": "A002",
        "name": "James",
        "math": 89,
        "physics": 76,
        "chemistry": 51
    },
    {
        "id": "A003",
        "name": "Jenny",
        "math": 79,
        "physics": 90,
        "chemistry": 78
    }]
}
```

嵌套数据：

```
import pandas as pd

df = pd.read_json('nested_list.json')

print(df)
```

以上实例输出结果为：

```
          school_name   class                                           students
0  ABC primary school  Year 1  {'id': 'A001', 'name': 'Tom', 'math': 60, 'phy...
1  ABC primary school  Year 1  {'id': 'A002', 'name': 'James', 'math': 89, 'p...
2  ABC primary school  Year 1  {'id': 'A003', 'name': 'Jenny', 'math': 79, 'p...
```

使用`json_normalize()`将内嵌数据解析

```
pandas.json_normalize(
    data , # 数据
    record_path = [ ], #用于展开的内嵌JSON数据
    meta = [ ], # 展示不用展开内嵌的数据
)
```

```python
import pandas as pd
import json

# 使用 Python JSON 模块载入数据
with open('nested_list.json','r') as f:
    data = json.loads(f.read())

# 展平数据
df_nested_list = pd.json_normalize(data, record_path =['students'])
print(df_nested_list)
```

以上实例输出结果为:

```
     id   name  math  physics  chemistry
0  A001    Tom    60       66         61
1  A002  James    89       76         51
2  A003  Jenny    79       90         78
```

```python
import pandas as pd
import json

print(pd.__version__)

normalize_list = open('./nest_list.json', 'r')
data = json.loads(normalize_list.read())
print(data)

# 展平数组
df_list = pd.json_normalize(
    data,
    record_path=['students'],
    meta=['school_name', 'class']
)

print(df_list)
```

数据清洗

对没有用的数据进行处理。

清洗空值-dropna()

```
DataFrame.dropna(
    axis, # 默认0表示逢空值去掉整行，1表示逢空值去掉整列
    how, # 默认any\
    thresh,
    subset,
    inplace
)
```

- how：默认为 **'any'** 如果一行（或一列）里任何一个数据有出现 NA 就去掉整行，如果设置 **how='all'** 一行（或列）都是 NA 才去掉这整行。
- thresh：设置需要多少非空值的数据才可以保留下来的。
- subset：设置想要检查的列。如果是多个列，可以使用列名的 list 作为参数。
- inplace：如果设置 True，将计算得到的值直接覆盖之前的值并返回 None，修改的是源数据。

通过`isnull()`判断各个单元格是否为空。

```
import pandas as pd

df = pd.read_csv('property-data.csv')

print (df['NUM_BEDROOMS'])
print (df['NUM_BEDROOMS'].isnull())
```

以上实例输出结果如下：

```
0      3
1      3
2    NaN
3      1
4      3
5    NaN
6      2
7      1
8     na
Name: NUM_BEDROOMS, dtype: object
0    False
1    False
2     True
3    False
4    False
5     True
6    False
7    False
8    False
```

```
DataFrame.fillna(value) # 将缺失值替换为指定的值

DataFrame.replace(old_value , new_value) # 将指定值替换为新值

DataFrame.duplicate() # 检查是否有重复数据

DataFrame.drop_duplicates() # 删除重复的数据
```

数据选择和切片

```
df['column_name'] # 选择指定的列

df.loc[row_index , column_name] # 通过位置选择数据

df.iloc[row_index , column_index] # 通过标签或位置选择数据

df.filter(items = ['column_name1' , 'column_name2']) # 选择指定的列

df.filter(regex = 'regex') # 选择列名匹配正则表达式的列

df.sample(n = 5)
```