总结

关键：[row , col]
对于Dataframe，如果相应**操控列col**，直接df ['col' 名] ，如果要**操作行row**，则df.loc['row'名]

iloc，则是对matrix的index做处理。

loc 与条件

还是关键 [row , col]
df.loc[df.column_name 条件]的意思是，选择column_name符合某些条件的row数据。

还可以基于多个列条件来筛选row数据。

Numpy的局限

不能处理混合数据类型的数据，这也是为什么我们要使用Pandas。
NumPy arrays are designed for numerical computation and **cannot easily handle collections of data that contain a mix of types** (e.g., strings, integers, and floats) in a single array.
A typical example of this limitation is a student attendance sheet. Each record might include a student’s name (a string), their marks (an integer), and their student ID (which serves as a label). NumPy arrays aren't well-suited for such cases because they can't store multiple types of data in the same array or use labels for indexing.
This is where **pandas** shines. pandas provides flexible and powerful tools to handle such **mixed-type datasets** and allows you to work with **labelled data**. It can store data in **DataFrames**, which are similar to tables in a database or Excel spreadsheet, making it ideal for data manipulation, analysis, and visualization tasks.

Series and Dataframe

Series

- **构造Series**

字典传入

pandas与Numpy的配合

slicing

与Numpy一样

series filtering

DataFrame

构造

传入DataFrame的是一个array，或字典

传入字典，相当于定义col name。

字典对应的value必须得是array-like object

index

df取得指定值

df 数据类型转换

传入字典，指定数据类型

常用的操作

- ### 更换index与赋值

- ### 数据类型转换

Dataframe Operation

NanN Not a number

缺失数据，统一用np.nan

原来可以直接放Series数据进series，并且缺失的数据，会自动以nan来填充。

isnull()

若为nan，则为true.

unique()

value_counts()

isin()

倒过来读：**the feature_name is in df**，若有则true，若无，则False

```
df.isin(['feature_name'])
```

配合使用any，**只要有一个匹配为true，则返回true**

用法

取特定数值的rows.

sum()

DataFrame Select/filtering data

.values： 取值

.loc: 通过row与col选取数据

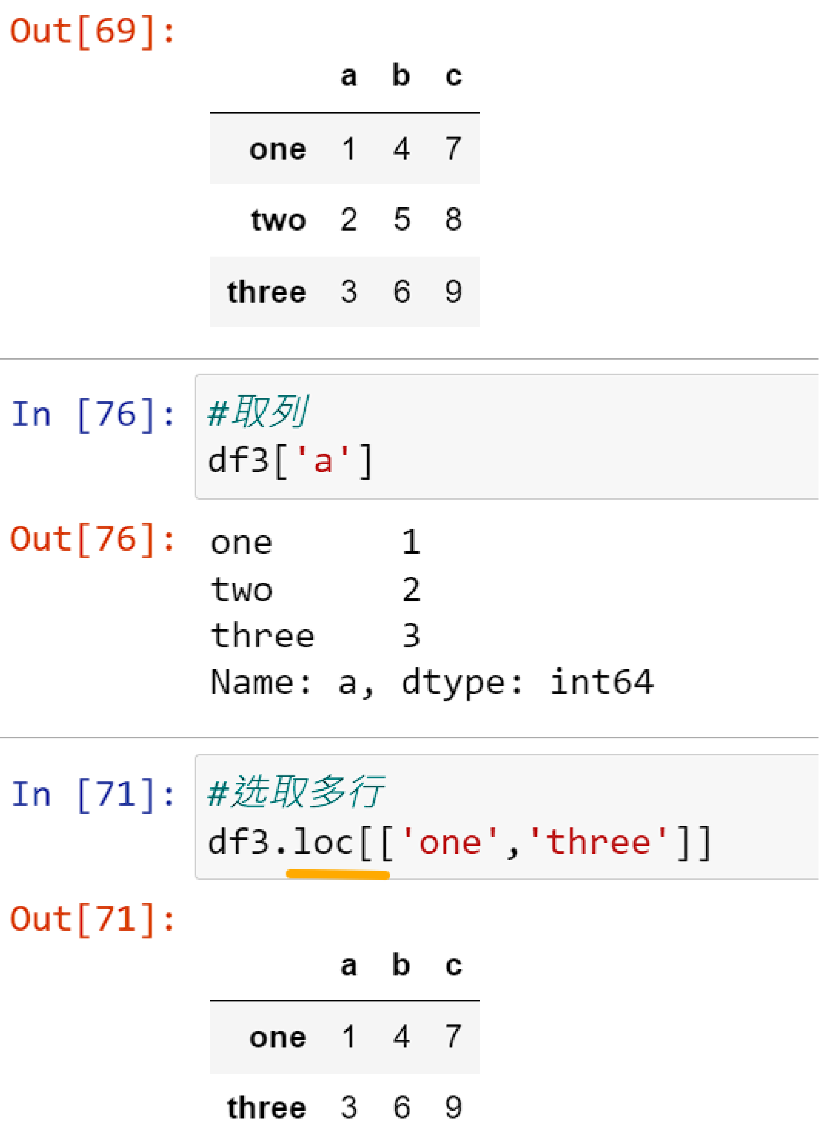
.iloc：通过index 来选取数据

loc

主要以选取row为主

记住一点，**loc [row , col]**，**loc[df['col'] < 某条件]**

若单纯取列，则直接[ ]取即可。



```
Out[69]:
```

	a	b	c
one	1	4	7
two	2	5	8
three	3	6	9

```
In [76]: #取列
         df3['a']

Out[76]: one      1
         two      2
         three    3
         Name: a, dtype: int64
```

```
In [71]: #选取多行
         df3.loc[['one','three']]

Out[71]:
```

	a	b	c
one	1	4	7
three	3	6	9

loc选取并赋值

In [81]:
```
#选取元素，并赋值
df3.loc['one','c']=100
df3
```

Out[81]:

	a	b	c
one	1	4	100
two	2	5	8
three	3	6	9

iloc

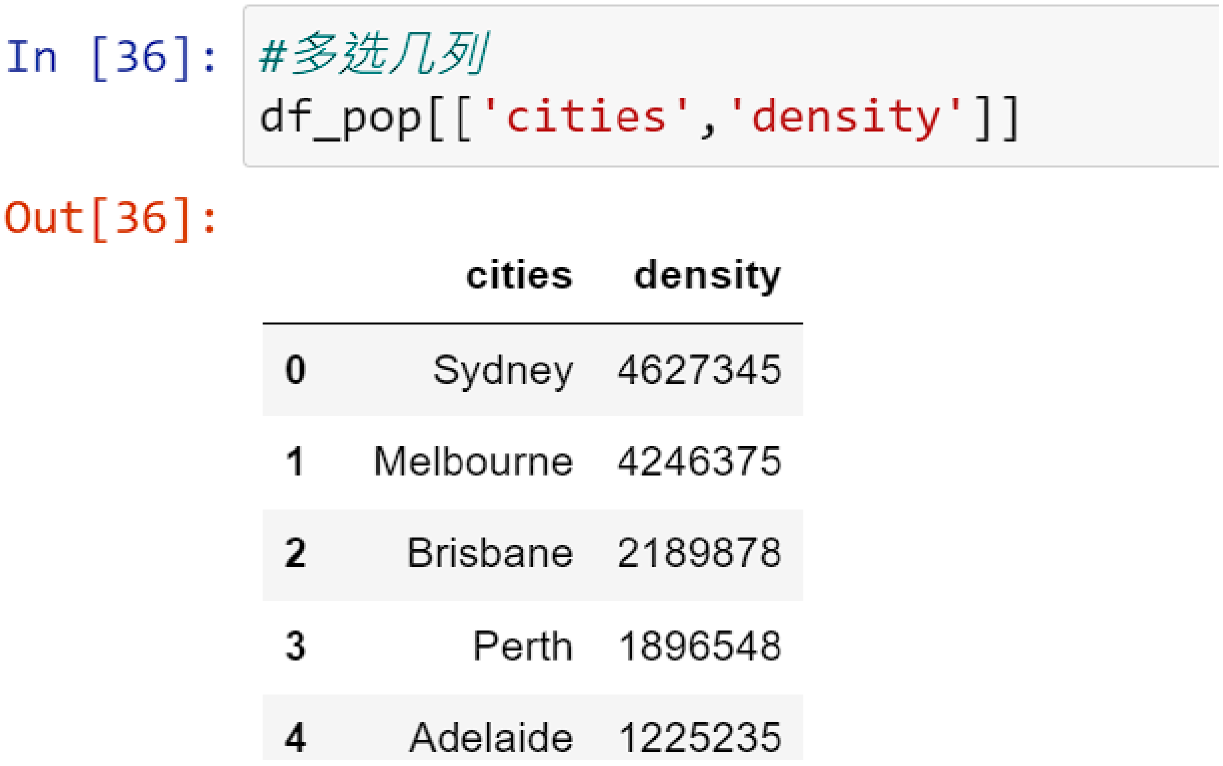
通过行列的index来获取对应的值。

In [45]:
```
dataframe_11.iloc[0,1]='rainbow'
dataframe_11
```

Out[45]:

	object	colour
one	ball	rainbow
two	pen	green
three	pencil	yellow
four	paper	red
five	mug	black

选取特定的列 [[ ]]



In [36]:
```python
#多选几列
df_pop[['cities','density']]
```

Out[36]:

	cities	density
0	Sydney	4627345
1	Melbourne	4246375
2	Brisbane	2189878
3	Perth	1896548
4	Adelaide	1225235

slicing

loc, slicing with condition

筛选出，某列符合某条件的row

筛选符合条件的特定的列

例子2

筛选的例子

获得每一列的数据类型 dtype()

计算某列中各项数据的比例

替换数据

需要配合使用numpy

题目例子用法