DataFrame 學習筆記

🏗 DataFrame 的建立方式

```
import pandas as pd
import numpy as np
```

透過 Series 建立

```
s1 = pd.Series([1, 2, 3, 4, 5])
s2 = pd.Series([6, 7, 8, 9, 10])
df = pd.DataFrame({"第1列": s1, "第2列": s2})
df
```

```
  第1列  第2列
0   1   6
1   2   7
2   3   8
3   4   9
4   5  10
```

透過字典建立

```
df = pd.DataFrame(
  {
    "name": ["tom", 'jack', 'alice', 'bob', 'allen'],
    "age": [15, 17, 20, 26, 30],
    "score": [60.5, 80, 30.6, 70, 83.5]
  }, index=[1, 2, 3, 4, 5], columns=["name", "score", "age"]
)
df
```

```
   name  score  age
1   tom   60.5   15
```

```
2  jack  80.0  17
3 alice  30.6  20
4   bob  70.0  26
5 allen  83.5  30
```

📋 DataFrame 的屬性

```
print('行索引：')
print(df.index)
print('列標籤：')
print(df.columns)
print('值')
print(df.values)
```

```
行索引：
Index([1, 2, 3, 4, 5], dtype='int64')
列標籤：
Index(['name', 'score', 'age'], dtype='object')
值
[['tom' 60.5 15]
 ['jack' 80.0 17]
 ['alice' 30.6 20]
 ['bob' 70.0 26]
 ['allen' 83.5 30]]
```

```
print('維度：', df.ndim)
print('形狀:', df.shape)
print('元素個數：', df.size)
print('資料型別：')
print(df.dtypes)
```

```
維度： 2
形狀: (5, 3)
元素個數： 15
```

```
資料型別：
name     object
score   float64
age       int64
dtype: object
```

🔄 行列轉置

```
print(df.T)
```

```
      1    2     3    4     5
name   tom jack alice  bob allen
score 60.5 80.0  30.6 70.0  83.5
age    15   17    20   26    30
```

🎯 獲取元素方式（loc / iloc / at / iat）

獲取某行

```
print(df.loc[4])
print(df.iloc[3])
```

```
name     bob
score   70.0
age       26
Name: 4, dtype: object
```

獲取某列

```
print(df.loc[:, 'name'])
print(df.iloc[:, 0])
```

```
1    tom
2   jack
3  alice
4    bob
5  allen
Name: name, dtype: object
```

獲取單個元素

```
print(df.at[3, 'score'])
print(df.iat[2, 1])
print(df.loc[3, 'score'])
print(df.iloc[2, 1])
```

```
30.6
30.6
30.6
30.6
```

獲取單列或多列資料

```
print(df['name'])
print(type(df['name']))

print(df.name)
print(type(df.name))

print(df['name']('name'))
print(type(df['name']('name')))

print(df['name', 'score']('name',%20'score'))
```

```
1    tom
2   jack
```

```
3   alice
4     bob
5   allen
Name: name, dtype: object
<class 'pandas.core.series.Series'>

1     tom
2    jack
3   alice
4     bob
5   allen
Name: name, dtype: object
<class 'pandas.core.series.Series'>

   name
1   tom
2  jack
3 alice
4   bob
5 allen
<class 'pandas.core.frame.DataFrame'>

   name  score
1   tom   60.5
2  jack   80.0
3 alice   30.6
4   bob   70.0
5 allen   83.5
```

🔍 檢視部分資料

```
print(df.head(2))
print(df.tail(3))
```

```
  name  score  age
1  tom   60.5   15
2 jack   80.0   17

   name  score  age
3 alice   30.6   20
4   bob   70.0   26
5 allen   83.5   30
```

✅ 資料篩選

使用布林索引

```
df[df.score > 70]
df[(df['score'] > 70) & (df.age < 20)]
```

```
   name  score  age
2  jack   80.0   17
```

隨機抽樣

```
df.sample(3)
```

```
   name  score  age
2  jack   80.0   17
3 alice   30.6   20
1   tom   60.5   15
```

📊 資料檢視與統計方法

```
print(df.head())  # 前 n 行
```

```
print(df.tail(1)) # 後 n 行
```

```
   name  score  age
1   tom   60.5   15
2  jack   80.0   17
3 alice   30.6   20
4   bob   70.0   26
5 allen   83.5   30

   name  score  age
5 allen   83.5   30
```

```
print(df.isin(['jack', 20]))
print(df.isna())
```

```
   name  score   age
1 False  False  False
2  True  False  False
3 False  False   True
4 False  False  False
5 False  False  False

   name  score   age
1 False  False  False
2 False  False  False
3 False  False  False
4 False  False  False
5 False  False  False
```

統計運算

```
print(df['score'].sum())
print(df.score.max())
print(df.age.min())
print(df.score.mean())
```

```
print(df.score.median())
print(df.age.mode())
```

```
324.6
83.5
15
64.92
70.0
0   15
1   17
2   20
3   26
4   30
Name: age, dtype: int64
```

描述統計

```
print(df.score.std())
print(df.score.var())
print(df.score.quantile(0.25))
print(df.describe())
print(df.count())
print(df.value_counts())
```

```
19.037375519400424
362.4216666666666
60.5

            score       age
count   6.000000   6.000000
mean   64.183333  20.166667
std    19.037376   6.493587
min    30.600000  15.000000
25%    60.500000  15.000000
50%    65.250000  17.500000
75%    77.500000  24.500000
max    83.500000  30.000000
```

```
name    6
score   6
age     6
dtype: int64

name   score  age
 tom    60.5   15    2
 alice  30.6   20    1
 allen  83.5   30    1
 bob    70.0   26    1
 jack   80.0   15    1
Name: count, dtype: int64
```

重複值處理

```
print(df.drop_duplicates())
print(df.duplicated(subset=['age']))
```

```
   name  score  age
1   tom   60.5   15
3  jack   80.0   15
4 alice   30.6   20
5   bob   70.0   26
6 allen   83.5   30

1   False
2    True
3    True
4   False
5   False
6   False
dtype: bool
```

替換與累積運算

```
print(df.replace(15, 30))
df.cumsum()
df.cummin(axis=0)
```

```
   name  score  age
1   tom   60.5   30
2   tom   60.5   30
3  jack   80.0   30
4 alice   30.6   20
5   bob   70.0   26
6 allen   83.5   30
```

DataFrame 練習題整理

```
import pandas as pd
import numpy as np
```

📝 案例1：學生成績分析

場景：某班級的學生成績資料如下，請完成以下任務：

1. 計算每位學生的總分和平均分。
2. 找出數學成績高於90分或英語成績高於85分的學生。
3. 按總分從高到低排序，並輸出前3名學生。

建立資料

```
data = {
    '姓名': ['張三', '李四', '王五', '趙六', '錢七'],
    '數學': [85, 92, 78, 88, 95],
    '英語': [90, 88, 85, 92, 80],
    '物理': [75, 80, 88, 85, 90]
}
scores = pd.DataFrame(data)
scores
```

```
  姓名 數學 英語 物理
0 張三  85  90  75
1 李四  92  88  80
2 王五  78  85  88
3 趙六  88  92  85
4 錢七  95  80  90
```

計算總分與平均分

```
scores["Total"]("數學",%20"英語",%20"物理").sum(axis=1)
scores["Average"]("數學",%20"英語",%20"物理").mean(axis=1)
scores
```

```
  姓名 數學 英語 物理 Total   Average
0 張三  85  90  75   250 83.333333
1 李四  92  88  80   260 86.666667
2 王五  78  85  88   251 83.666667
3 趙六  88  92  85   265 88.333333
4 錢七  95  80  90   265 88.333333
```

篩選條件

```
scores[(scores["數學"] > 90) | (scores["英語"] > 85)]
```

```
  姓名 數學 英語 物理 Total   Average
0 張三  85  90  75   250 83.333333
1 李四  92  88  80   260 86.666667
3 趙六  88  92  85   265 88.333333
4 錢七  95  80  90   265 88.333333
```

排序並輸出前3名

```
scores.sort_values("Total", ascending=False).head(3)
```

```
  姓名 數學 英語 物理 Total   Average
4 錢七  95  80  90   265 88.333333
3 趙六  88  92  85   265 88.333333
1 李四  92  88  80   260 86.666667
```

```
scores.nlargest(3, columns="Total")
```

```
  姓名 數學 英語 物理 Total   Average
3 趙六  88  92  85   265 88.333333
4 錢七  95  80  90   265 88.333333
1 李四  92  88  80   260 86.666667
```

📊 案例2：銷售資料分析

場景：某公司銷售資料如下，請完成以下任務：

1. 計算每種產品的總銷售額（銷售額 = 單價 × 銷量）。
2. 找出銷售額最高的產品。
3. 按銷售額從高到低排序，並輸出所有產品資訊。

建立資料

```
data = {
    '產品名稱': ['A', 'B', 'C', 'D'],
    '單價': [100, 150, 20, 120],
    '銷量': [50, 30, 20, 40],
}
df = pd.DataFrame(data)
df
```

```
  產品名稱  單價 銷量
0    A  100  50
1    B  150  30
```

```
2   C  20  20
3   D  120  40
```

計算總銷售額

```
df["Total sales price"] = df["單價"] * df["銷量"]
df
```

```
  產品名稱  單價  銷量  Total sales price
0   A  100  50           5000
1   B  150  30           4500
2   C  20  20            400
3   D  120  40           4800
```

找出銷售額最高的產品

```
df.loc[df["Total sales price"] == df["Total sales price"].max(), "產品名稱"]
```

```
0   A
Name: 產品名稱, dtype: object
```

排序所有產品資訊

```
df.sort_values("Total sales price", ascending=False)
```

```
  產品名稱  單價  銷量  Total sales price
0   A  100  50           5000
3   D  120  40           4800
1   B  150  30           4500
2   C  20  20            400
```

🛒 案例3：電商使用者行為分析

場景：某電商平臺的使用者行為資料如下，請完成以下任務：

1. 計算每位使用者的總消費金額（消費金額 = 商品單價 × 購買數量）。
2. 找出消費金額最高的使用者，並輸出其所有資訊。
3. 計算所有使用者的平均消費金額（保留2位小數）。
4. 統計電子產品的總購買數量。

建立資料

```
data = {
    '使用者ID': [101, 102, 103, 104, 105],
    '使用者名稱': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    '商品類別': ['電子產品', '服飾', '電子產品', '家居', '服飾'],
    '商品單價': [1200, 300, 800, 150, 200],
    '購買數量': [1, 3, 2, 5, 4]
}
df = pd.DataFrame(data)
df
```

```
  使用者ID 使用者名稱 商品類別 商品單價 購買數量
0  101 Alice 電子產品 1200   1
1  102   Bob 服飾 300   3
2  103 Charlie 電子產品 800   2
3  104 David 家居 150   5
4  105   Eve 服飾 200   4
```

計算總消費金額

```
df['總消費金額'] = df['商品單價'] * df['購買數量']
df
```

```
  使用者ID 使用者名稱 商品類別 商品單價 購買數量 總消費金額
0  101 Alice 電子產品 1200   1  1200
1  102   Bob 服飾 300   3  900
2  103 Charlie 電子產品 800   2  1600
3  104 David 家居 150   5  750
4  105   Eve 服飾 200   4  800
```

找出消費金額最高的使用者

```
df.loc[df['總消費金額'] == df['總消費金額'].max()]
```

```
  使用者ID 使用者名稱 商品類別  商品單價 購買數量 總消費金額
2   103 Charlie 電子產品   800    2   1600
```

計算平均消費金額

```
df['總消費金額'].mean().round(2)
```

```
1050.0
```

統計電子產品的總購買數量

```
df[df['商品類別'] == '電子產品']['購買數量'].sum()
```

```
3
```