In [1]:

```
import pandas as pd
```

1. 시리즈 만들기

`pd.Series(리스트)`

- 시리즈 = 엑셀 시트의 열 1개를 의미(1차원 리스트형태)

In [6]:

```
# 시리즈 만들기
list = ['amy', 170, 240]
```

In [9]:

```
# 타입 확인하기
print(type(list))
s = pd.Series(list)
print(type(s))
print(s)
```

```
<class 'list'>
<class 'pandas.core.series.Series'>
0    amy
1    170
2    240
dtype: object
```

0, 1, 2 : Series의 인덱스
amy, 170, 240 : 각 인덱스의 데이터

2. 시리즈의 index와 value 가져오기

- 시리즈에는 **index**와 **value**가 있다.

  시리즈에서 `index` 가져오기 : `Series.index`
  시리즈에서 `value` 가져오기 : `Series.values`

- 시리즈의 인덱스는 리스트의 인덱스와 다른 개념이다.
- 시리즈의 인덱스는 데이터의 이름이고, 행 번호는 따로 있다.

In [10]:

```
s.index
```

Out[10]:

```
RangeIndex(start=0, stop=3, step=1)
```

In [11]:

```
s.values
```

Out[11]:

```
array(['amy', 170, 240], dtype=object)
```

3. 시리즈의 index 지정하기

`Series.index = 인덱스 리스트`

- 시리즈의 인덱스는 숫자, 문자열 모두 가능하다.

In [12]:

```
# 시리즈 인덱스 지정하기
s.index = ['name', 'height', 'footSize']
```

In [13]:

```
# 시리즈 출력하기
s
```

Out[13]:

```
name         amy
height       170
footSize     240
dtype: object
```

In [14]:

```
# 인덱스 확인하기
s.index
```

Out[14]:

```
Index(['name', 'height', 'footSize'], dtype='object')
```

4. 시리즈의 통계값 사용하기

- 평균 : `Series.mean()`
- 최소값 : `Series.min()`
- 최대값 : `Series.max()`
- 중간값 : `Series.median()`
- 표준편차 : `Series.std()`
- **Series**의 통계값은 **Series**의 **value**가 모두 숫자형일 때 사용할 수 있다.

In [15]:

```
s2 = pd.Series([10, 20, 30, 40, 50])
```

In [17]:

```
print("평균 : ", s2.mean())
print("최소값 : ", s2.min())
print("최대값 : ", s2.max())
print("중간값 : ", s2.median())
print("표준편차 : ", s2.std())
```

```
평균 :  30.0
최소값 :  10
최대값 :  50
중간값 :  30.0
표준편차 :  15.811388300841896
```

4.1. 시리즈 요약통계

`Series.describe()`

In [18]:

```
s2.describe()
```

Out[18]:

```
count      5.000000
```

```
mean      30.000000
std       15.811388
min       10.000000
25%       20.000000
50%       30.000000
75%       40.000000
max       50.000000
dtype: float64
```

5. 시리즈 주요 메서드

- 값 정렬 : `Series.sort_values()`
- 인덱스 정렬 : `Series.sort_index()`
- 인덱스 리셋 : `Series.reset_index()`
- 특정 *값*을 가진 시리즈 값을 교체 : `Series.replace(찾을 값, 교체할 값)`
- 시리즈를 데이터프레임으로 변환 : `Series.to_frame()`

In [28]:

```
s3 = pd.Series([1, 3, 2, 4, 10])
```

In [30]:

```
# s3의 value 중 10을 5로 교체
s3 = s3.replace(10, 5)
s3
```

Out[30]:

```
0    1
1    3
2    2
3    4
4    5
dtype: int64
```

In [31]:

```
# s3 정렬(디폴트는 오름차순, ascending = True)
s3.sort_values()
```

Out[31]:

```
0    1
2    2
1    3
3    4
4    5
dtype: int64
```

In [32]:

```
# s3 정렬(내림차순 : ascending = False)
s3.sort_values(ascending = False)
```

Out[32]:

```
4    5
3    4
1    3
2    2
0    1
dtype: int64
```

In [33]:

```
# s3을 데이터프레임으로 만들기
s3.to_frame()
```

Out[33]:

	0
0	1
1	3
2	2
3	4
4	5

In [34]:

```
# 시리즈 인덱스 새로 만들기
s3.index = ['zero', 'one', 'two', 'three', 'four']
print(s3.index)
print(s3)
```

```
Index(['zero', 'one', 'two', 'three', 'four'], dtype='object')
zero     1
one      3
two      2
three    4
four     5
dtype: int64
```

In [35]:

```
# 시리즈 인덱스 새로 만들기
s.reset_index()
```

Out[35]:

	index	0
0	name	amy
1	height	170
2	footSize	240

기존의 인덱스는 지워지는 것이 아니라 한 **column** 안에 백업이 된다.
그리고 나서 새로운 인덱스가 생성된다.