Series 자료형

#01. Pandas 패키지

쉽고 직관적인 관계형 또는 분류된 데이터로 작업 할 수 있도록 설계된 빠르고 유연하며 표현이 풍부한 데이터 구조를 제공하는 Python 패키지

실용적인 실제 데이터 분석을 수행하기 위한 고수준의 객체를 사용할 수 있다.

1) Pandas가 사용되는 경우

- SQL 테이블 또는 Excel 스프레드 시트에서와 같이 이질적으로 유형이 지정된 열이있는 데이블 형식 데이터
- 정렬되고 정렬되지 않은 시계열 데이터
- 행 및 열 레이블이 포함 된 임의의 행렬 데이터
- 일반적인 통계 데이터 세트

Pandas의 두 가지 주요 데이터 구조인 Series (1차원) 및 DataFrame (2차원)은 재무, 통계, 사회 과학 및 다양한 엔지니어링 분야의 일반적인 사용 사례의 대부분을 처리함

2) Pandas 데이터 구조

차원	이름	설명
1차원	Series	균일한 유형의 배열로 표시된 1차원 데이터
2차원	DataFrame	행과 열이있는 크기가 가변적인 테이블 형식의 2차원 데이터

3) 필요한 패키지 가져오기

`pandas` 패키지가 설치되어 있어야 한다.

```
from pandas import Series
```

#02. Series 자료형

1) Series 자료형의 구조

pandas 패키지에 포함되어 있는 자료형으로 `list > numpy array > series` 순으로 기능이 확장된다.

인덱스를 명시적으로 포함하고 있는 형태이며, 엑셀의 열 하나를 표현하고 있다고 이해할 수 있다.

기본 시리즈 만들기

리스트나 numpy 배열을 통해 생성할 수 있다. 즉, 시리즈는 리스트 자료형을 가공하여 생성된 데이터 구조.

```
items = [10,30,50,70,90]
column = Series(items)
column
```

인덱스를 활용한 개별 값 확인

```
print(column[0])
print(column[2])
print(column[4])
```

시리즈의 값만 추출

```
column.values
```

값에 대한 데이터 타입 확인

Numpy 배열임을 알 수 있다.

```
type(column.values)
```

List 자료형으로 변환

```
mylist = list(column.values)
mylist
```

시리즈의 색인(index)만 추출

```
column.index
```

색인의 데이터 타입 확인

```
type(column.index)
```

색인을 리스트로 변환

```
mylist = list(column.index)
mylist
```

조건검색

`이름[이름에 대한 조건식]`

```
in1 = column[column > 30]
in1
```

`AND` 검색

```
in2 = column[column <= 70][column > 10]
in2
```

`OR` 검색

```
in3 = column[(column <= 10) | (column >= 70)]
in3
```

3) Series 연산

```
# 인덱스를 직접 지정하면서 시리즈 만들기
# -> 지난주 주말에 대한 매출액
week1 = Series([290000, 310000], index=['sat', 'sun'])
week1
```

```
# -> 이번주 주말에 대한 매출액
week2 = Series([120000, 220000], index=['sun', 'sat'])
week2
```

```
# 시리즈 객체의 사칙연산
# -> index가 동일한 항목끼리 연산이 수행된다.
merge = week1 + week2
merge
```