Chapter. 04

한 눈에 데이터 보기: 데이터 통합 및 집계

merge를 이용한 데이터 통합

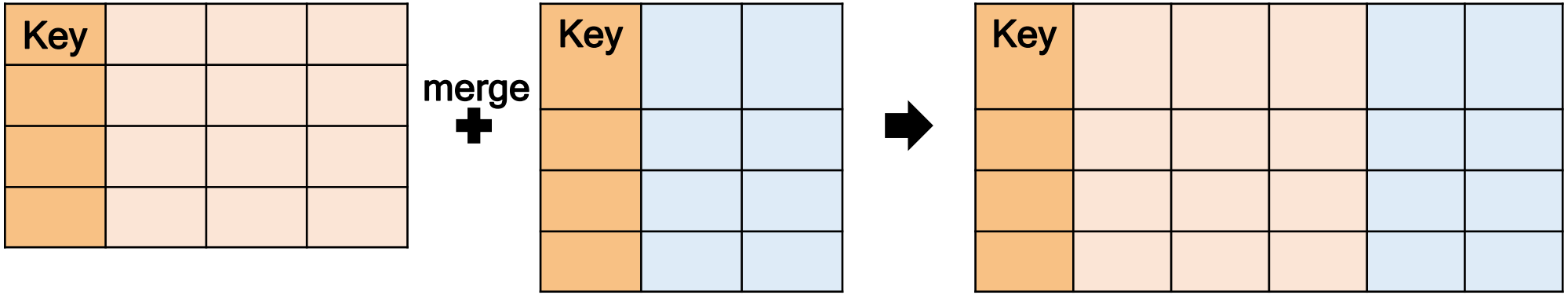
FAST CAMPUS
ONLINE
데이터 탐색과 전처리 I

강사. 안길승

I merge가 필요한 상황

- 효율적인 데이터 베이스 관리를 위해, 잘 정제된 데이터일지라도 데이터가 **키 변수**를 기준으로 나뉘어 저장되는 경우가 매우 흔함

- SQL에서는 JOIN을 이용하여 해결하며, python에서는 **merge**를 이용하여 해결함



pandas.merge

- 키 변수를 기준으로 두 개의 데이터 프레임을 병합(join)하는 함수

- 주요 입력
  - left: 통합 대상 데이터 프레임 1
  - right: 통합 대상 데이터 프레임 2
  - on: 통합 기준 key 변수 및 변수 리스트 (입력을 하지 않으면, 이름이 같은 변수를 key로 식별함)
  - left_on: 데이터 프레임 1의 key 변수 및 변수 리스트
  - right_on: 데이터 프레임 2의 key 변수 및 변수 리스트
  - left_index: 데이터 프레임 1의 인덱스를 key 변수로 사용할 지 여부
  - right_index: 데이터 프레임 2의 인덱스를 key 변수로 사용할 지 여부

Chapter. 04

한 눈에 데이터 보기: 데이터 통합 및 집계

concat을 이용한 데이터 통합

FAST CAMPUS
ONLINE
데이터 탐색과 전처리 I

강사. 안길승

I concat이 필요한 상황

- 센서, 로그, 거래 데이터 등과 같이 크기가 **매우 큰 데이터는** 시간과 ID 등에 따라 **분할되어 저장**됨

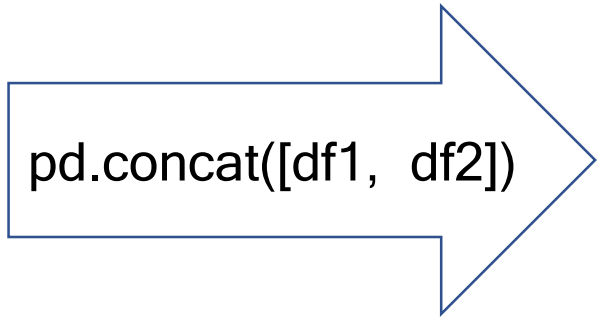
- pandas.concat 함수를 사용하면 손쉽게 해결할 수 있음

df1

X1	X2	X3	X4

df2

X1	X2	X3	X4



X1	X2	X3	X4

merged_df

- 통합해야 하는 데이터가 많은 경우에는 **빈 데이터프레임**을 생성한 뒤, 이 데이터프레임과 반복문을 사용하여 불러온 데이터를 concat 함수를 이용하면 효율적으로 통합할 수 있음

pandas.concat

- 둘 이상의 **데이터 프레임을 이어 붙이는데** 사용하는 함수

- 주요 입력
  - objs: DataFrame을 요소로 하는 리스트 (입력 예시: [df1, df2])로 입력 순서대로 병합이 됨
  - ignore_index: **True**면 기존 인덱스를 무시하고 **새로운 인덱스를 부여**하며, **False**면 **기존 인덱스를 사용**

df1

Index	X1	X2
a		
b		

df2

Index	X1	X2
a		
b		

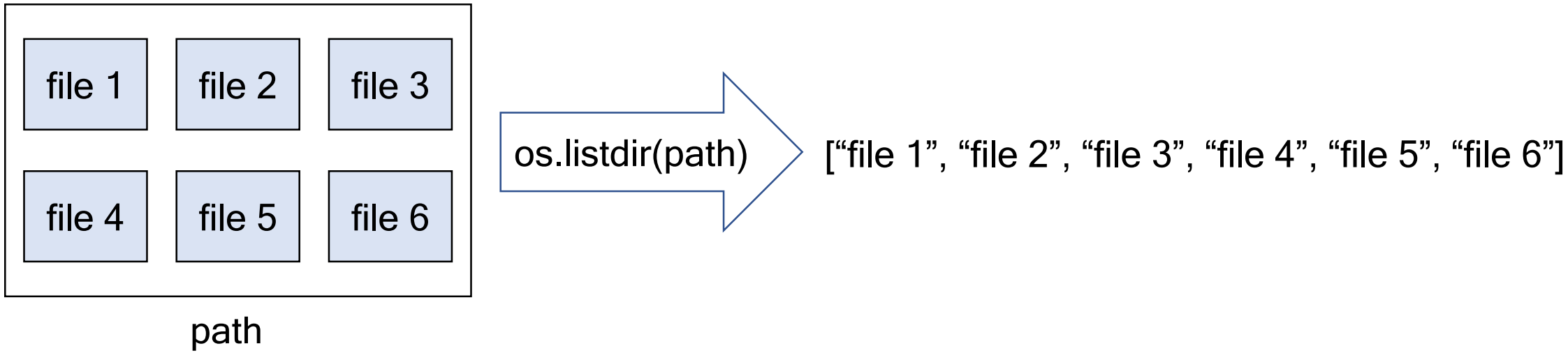
pd.concat([df1, df2], ignore_index = True)

Index	X1	X2
0		
1		
2		
3		

  - axis: **0**이면 **행 단위**로 병합을 수행하며, **1**이면 **열 단위**로 병합을 수행

os.listdir

- os.listdir(path): path 상에 있는 **모든 파일명을 리스트 형태**로 반환



I xlrd를 이용한 엑셀 시트 목록 가져오기

- xlrd는 엑셀 데이터를 다루기 위한 모듈로, 엑셀 내의 반복 작업을 하기 위해 주로 사용함

```
wb = xlrd.openworkbook(file, on_demand = True) # 엑셀 파일을 불러와 wb에 저장

wb.sheet_names() # wb에 있는 시트 목록을 리스트 형태로 반환
```

Chapter. 04

한 눈에 데이터 보기: 데이터 통합 및 집계

기초 통계 함수를 사용한 데이터 집계

FAST CAMPUS
ONLINE
데이터 탐색과 전처리 I

강사. 안길승

다양한 기초 통계 함수

- 기초 통계 함수는 DataFrame와 Series에 대해 모두 정의되어 있음

함수	내용
sum	합계 계산
mean	평균 계산
std	표준편차 계산
var	분산 계산
quantile	사분위수 계산
min	최소값 계산
max	최대값 계산

- axis를 설정해서 행별 혹은 열별 기초 통계를 구할 수 있음

I Tip. Axis 키워드

- axis 키워드는 numpy 및 pandas의 많은 함수에 사용되는 키워드로, 연산 등을 수행할 때 축의 방향을 결정하는 역할을 함

- axis가 0이면 행을, 1이면 열을 나타내지만 이렇게만 기억하면 논리적으로 이상한 점이 존재함
  - (예시 1) sum(axis = **0**): **열** 기준 합
  - (예시 2) concat([df1, df2], axis = **0**): **행** 단위 병합

Index	X1	X2
a	1	4
b	2	5
c	3	6

df.sum(axis = **1**) →

Index	
a	1+4
b	2+5
c	3+6

df

↓ df.sum(axis = **0**)

X1	X2
1+2+3	4+5+6

Tip. Axis 키워드

- axis 키워드는 그 함수의 **결과 구조**가 벡터 형태 (1차원)인지, 행렬 형태 (2차원)인지에 따라, 그 역할이 조금씩 다름

		결과	
		벡터	행렬
axis	0	결과가 행벡터	연산 과정이 행 기준
	1	결과가 열벡터	연산 과정이 열 기준

describe 함수

- 열별로 대표적인 기초 통계를 반환 (count, mean, std, min, 25%, 50%, 75%, max)

Chapter. 04

한 눈에 데이터 보기: 데이터 통합 및 집계

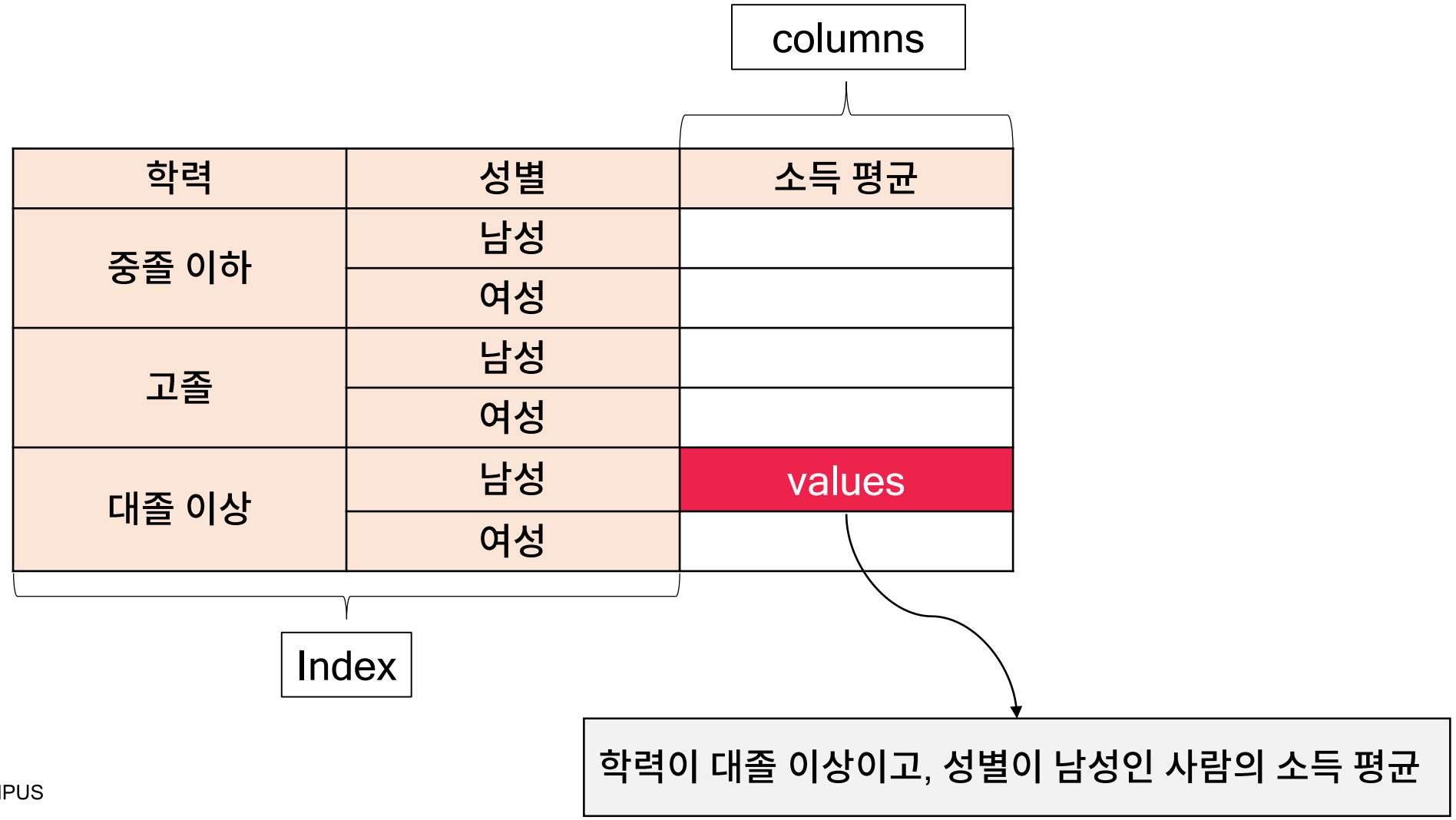
pivot을 이용한 데이터 집계

FAST CAMPUS
ONLINE
데이터 탐색과 전처리 I

강사. 안길승

I 피벗 테이블

- 피벗 테이블 (pivot table)은 데이터를 조건에 따른 변수들의 통계량을 요약한 테이블임



pandas.pivot_table

- 행 단위의 데이터 프레임을 피벗 테이블로 변환하는 함수

성별	소득
남성	200
남성	300
여성	150
여성	350
여성	250

df

pivot_table →

성별	소득 (평균)
남성	250
여성	250

pandas.pivot_table

- 주요 입력
  - data: 데이터 프레임
  - index: 행에 들어갈 조건
  - columns: 열에 들어갈 조건
  - values: 집계 대상 컬럼 목록
  - aggfunc: 집계 함수

Chapter. 04

한 눈에 데이터 보기: 데이터 통합 및 집계

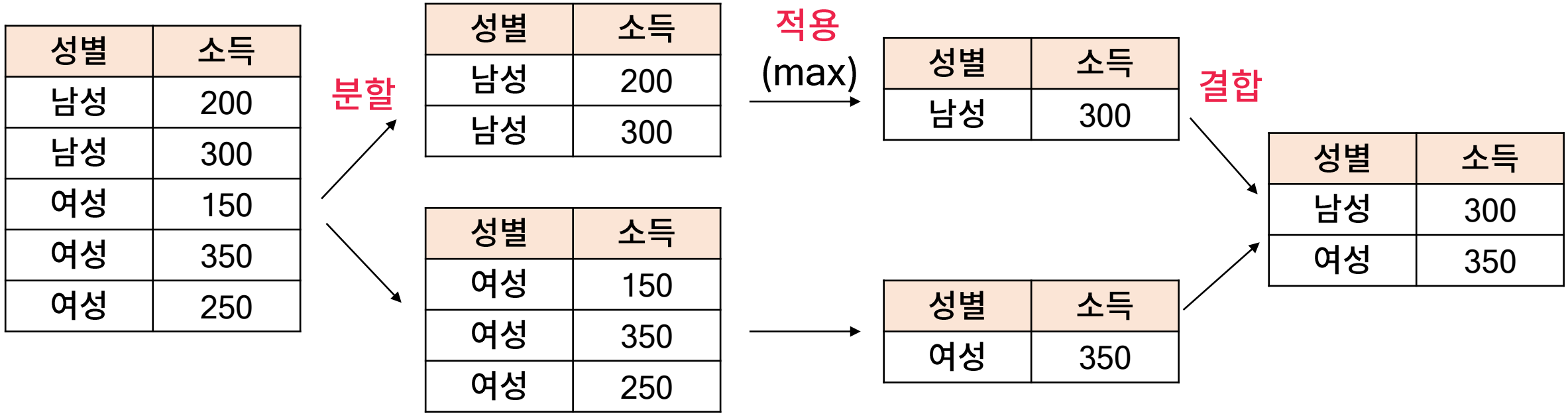
groupby를 이용한 데이터 집계

FAST CAMPUS
ONLINE
데이터 탐색과 전처리 I

강사. 안길승

groupby: 분할, 적용, 결합

- groupby는 **조건부 통계량**을 계산하기 위한 방법으로 크게 **분할, 적용, 결합**의 세 단계로 구성됨



성별	소득
남성	200
남성	300
여성	150
여성	350
여성	250

성별	소득
남성	200
남성	300

성별	소득
여성	150
여성	350
여성	250

성별	소득
남성	300

성별	소득
여성	350

성별	소득
남성	300
여성	350

DataFrame.groupby

- DataFrame을 분할 기준 컬럼을 기준으로 나누는 함수
- 사용 구조: df.groupby(분할기준 칼럼)[적용 기준 칼럼].집계함수
- 주요 입력
  - by: 분할 기준 컬럼 (목록)
  - as_index: 분할 기준 컬럼들을 인덱스로 사용할 것인지 여부 (default: True)
- 여러 개의 집계 함수나 사용자 정의 함수를 쓰고 싶다면 agg 함수를 사용해야 함

pivot_table과 groupby의 차이점

- pivot_table과 groupby 모두 조건부 통계량을 기준으로 데이터를 집계한다는 점에서 완전히 동일함
- 하지만 출력물 구조의 차이가 있으므로, 상황에 맞는 함수 선택이 필요함

	남성	여성
대졸	200	300
고졸	300	150

pivot_table ←

학력	성별	소득
대졸	남성	200
고졸	남성	300
고졸	여성	150
대졸	여성	350
대졸	여성	250

입력

→ groupby

학력	성별	소득
대졸	남성	200
	여성	300
고졸	남성	300
	여성	150

- 보통은 출력 결과 자체가 결과물인 경우에는 pivot_table을, 중간 산출물인 경우에는 groupby를 사용