데이터분석을위한 PANDAS

데이터 응용 - 합치기

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 연결

- 구성 형태와 속성 균일
- **행 또는 열 중에 한 방향으로 단순 이어 붙이는 것**
- 일관성 유지

```
pandas.concat( [ DF1,  ... , DFn] ,
                    axis=0,                 # 행 방향
                    ignore_index=False,     # 행 인덱스 유지
                    join='outer',           # 열 이름 합집합
                    keys=[]                 # 행/열 이름 재정의
                    )
```

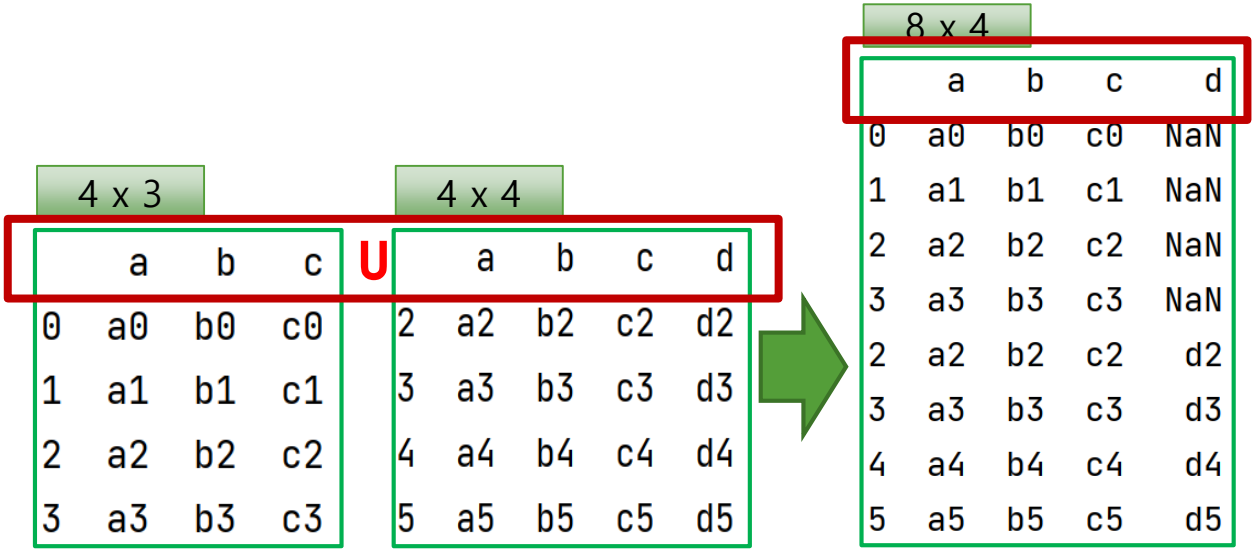
PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 연결

result1 = pd.concat([df1, df2])

행 방향 , 인덱스 유지, 열이름 합집합



	a	b	c
0	a0	b0	c0
1	a1	b1	c1
2	a2	b2	c2
3	a3	b3	c3

	a	b	c	d
2	a2	b2	c2	d2
3	a3	b3	c3	d3
4	a4	b4	c4	d4
5	a5	b5	c5	d5

	a	b	c	d
0	a0	b0	c0	NaN
1	a1	b1	c1	NaN
2	a2	b2	c2	NaN
3	a3	b3	c3	NaN
2	a2	b2	c2	d2
3	a3	b3	c3	d3
4	a4	b4	c4	d4
5	a5	b5	c5	d5

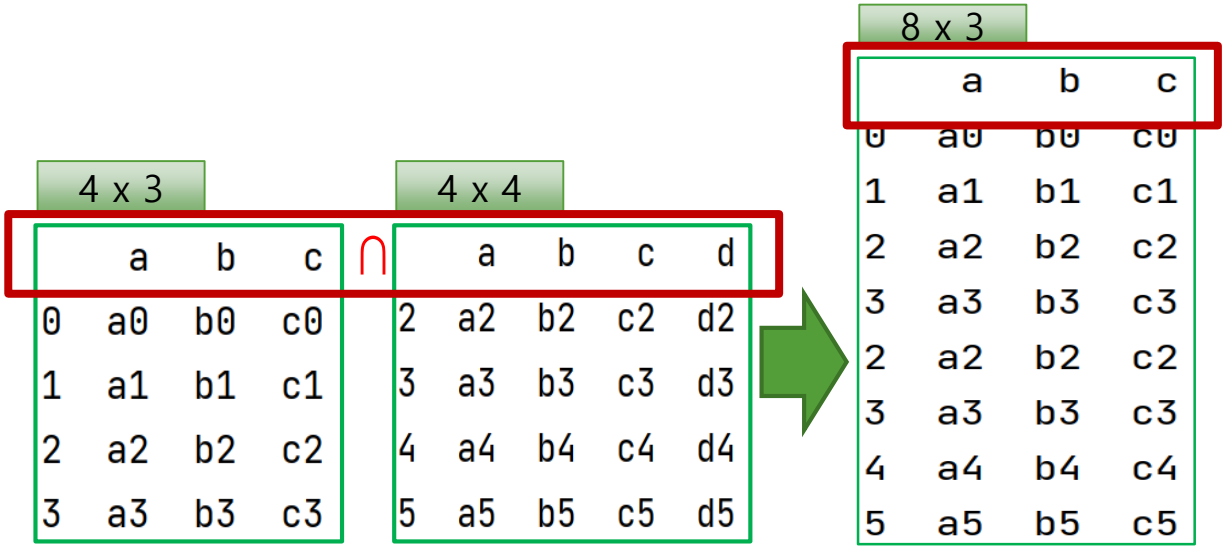
PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 연결

result1 = pd.concat([df1, df2], join= 'inner')

행 방향 , 인덱스 유지, 열이름 교집합



PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 연결

result1 = pd.concat([df1, df2], ignore_index=True)

행 방향 , 인덱스 변경, 합집합

4 x 3

	a	b	c
0	a0	b0	c0
1	a1	b1	c1
2	a2	b2	c2
3	a3	b3	c3

4 x 4

	a	b	c	d
2	a2	b2	c2	d2
3	a3	b3	c3	d3
4	a4	b4	c4	d4
5	a5	b5	c5	d5

8 x 4

	a	b	c	d
0	a0	b0	c0	NaN
1	a1	b1	c1	NaN
2	a2	b2	c2	NaN
3	a3	b3	c3	NaN
4	a2	b2	c2	d2
5	a3	b3	c3	d3
6	a4	b4	c4	d4
7	a5	b5	c5	d5

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 연결

result1 = pd.concat([df1, df2], key=['A', 'B'])

행 방향 , 멀티인덱스, 합집합

4 x 3

	a	b	c
0	a0	b0	c0
1	a1	b1	c1
2	a2	b2	c2
3	a3	b3	c3

4 x 4

	a	b	c	d
2	a2	b2	c2	d2
3	a3	b3	c3	d3
4	a4	b4	c4	d4
5	a5	b5	c5	d5

8 x 4

		a	b	c	d
A	0	a0	b0	c0	NaN
	1	a1	b1	c1	NaN
	2	a2	b2	c2	NaN
	3	a3	b3	c3	NaN
B	0	a2	b2	c2	d2
	1	a3	b3	c3	d3
	2	a4	b4	c4	d4
	3	a5	b5	c5	d5

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 연결

```
result1 = pd.concat( [df1, df2], axis='columns' )
```

열/컬럼 방향 , 기존 인덱스 유지, 행인덱스 합집합

4 x 3

	a	b	c
0	a0	b0	c0
1	a1	b1	c1
2	a2	b2	c2
3	a3	b3	c3

4 x 4

	a	b	c	d
2	a2	b2	c2	d2
3	a3	b3	c3	d3
4	a4	b4	c4	d4
5	a5	b5	c5	d5

6 x 7

	a	b	c	a	b	c	d
0	a0	b0	c0	NaN	NaN	NaN	NaN
1	a1	b1	c1	NaN	NaN	NaN	NaN
2	a2	b2	c2	a2	b2	c2	d2
3	a3	b3	c3	a3	b3	c3	d3
4	NaN	NaN	NaN	a4	b4	c4	d4
5	NaN	NaN	NaN	a5	b5	c5	d5

◆ 데이터프레임 합치기

▪ 병합

- SQL과 같은 관계형 데이터베이스와 유사한 **조인 연산을 수행**
- **어떤 기준에 의해 두 DataFrame을 병합(합치)하는 것**
- 기준이 되는 열이나 인덱스를 Key라함
- Series는 보통 DataFrame에 붙이는 용도

- 기준 키 찾는 순서
  - → 같은 이름의 컬럼 존재 ➔ 해당 컬럼을 on으로 설정
  - → 같은 이름의 컬럼 없으며 ➔ valueError 발생

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

```
pandas.merge( left,                 # 왼쪽 DataFrame/Series
              right,                # 오른쪽 DataFrame/Series
              how='inner',          # 병합 방식
              on=None,              # None : 공통 열 키로 설정
              left_on,              # 왼쪽에서 사용할 열 이름
              right_on,             # 오른쪽에서 사용할 열 이름
              left_index=False,     # 결합 기준으로 왼쪽 인덱스 사용 여부
              right_index=False,    # 결합 기준으로 오른쪽 인덱스 사용 여부
              sort=False,           # 결과를 키 기준으로 정렬 여부
              suffixes=('_x', '_y'), # 중복 열 이름 뒤에 붙일 접미사 )
```

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2, how='inner')

키 교차점 기준 병합

left	LEFT OUTER JOIN	왼쪽 프레임의 키만 사용.
right	RIGHT OUTER JOIN	오른쪽 프레임의 키만 사용.
outer	FULL OUTER JOIN	두 프레임의 키 합집합 사용.
inner	INNER JOIN	두 프레임의 키 교차점 사용.
cross	CROSS JOIN	두 프레임의 행의 데카르트 곱 생성.

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2, , how='inner')

열 방향 , 교집합, 동일 컬럼명 기준

	lkey	value
0	foo	1
1	bar	2
2	baz	3
3	foo	5

	rkey	value
2	foo	5
3	bar	6
4	baz	7
5	foo	8

동일 컬럼명

	lkey	value	rkey
0	foo	5	foo

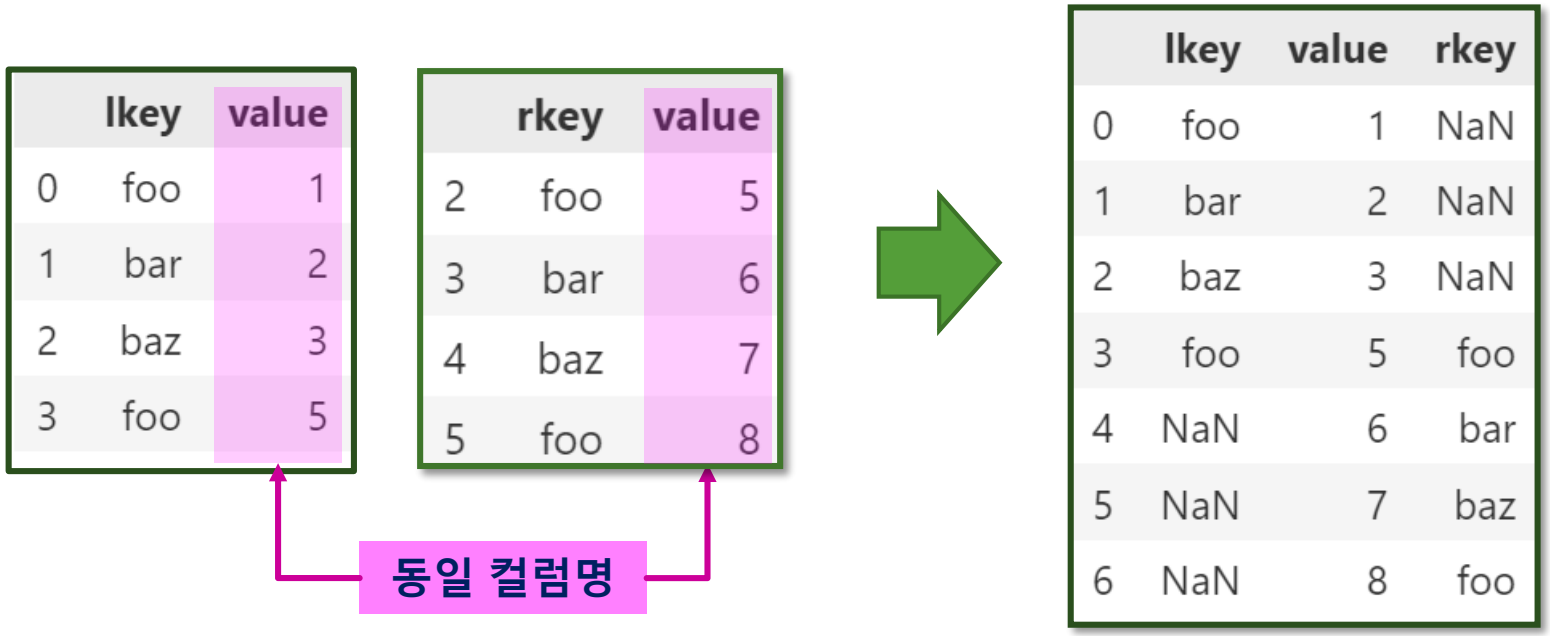
PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2 , how='outer')

열 방향 , 합집합, 동일 컬럼명 기준



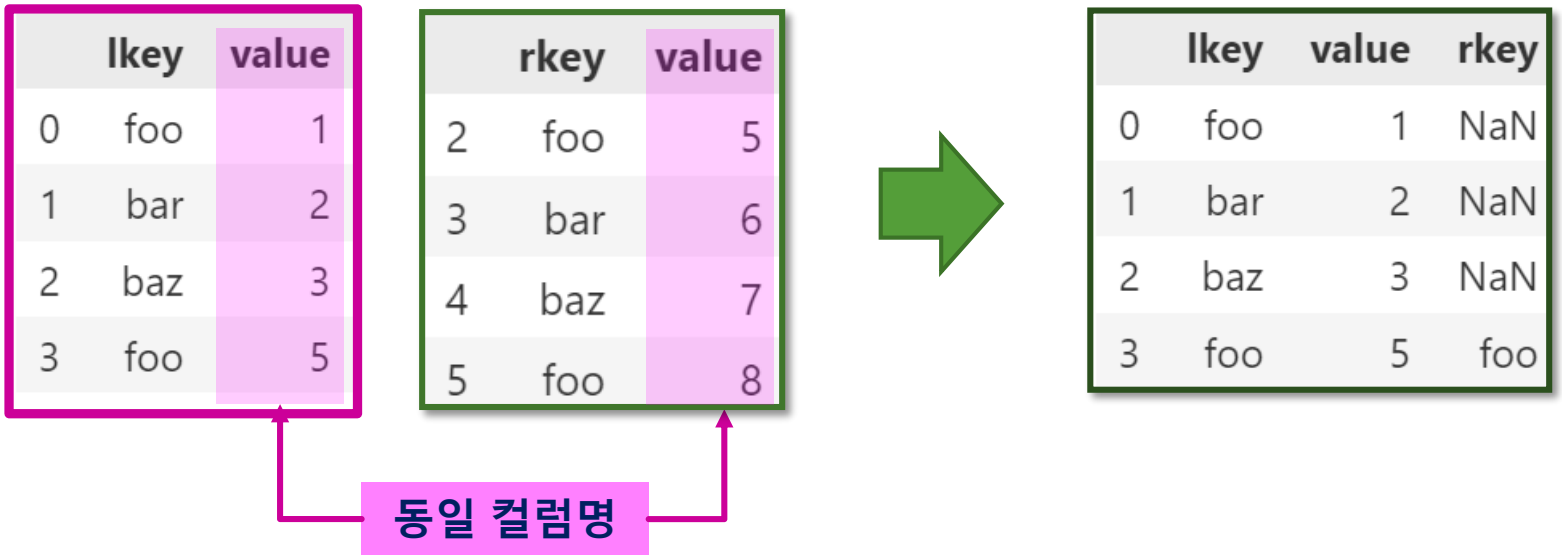
PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2 , how='left')

열 방향 , 왼쪽 기준 집합, 동일 컬럼명 기준



PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2 , how='right')

열 방향 , 오른쪽 기준 집합, 동일 컬럼명 기준

	lkey	value
0	foo	1
1	bar	2
2	baz	3
3	foo	5

	rkey	value
2	foo	5
3	bar	6
4	baz	7
5	foo	8

동일 컬럼명

	lkey	value	rkey
0	foo	5	foo
1	NaN	6	bar
2	NaN	7	baz
3	NaN	8	foo

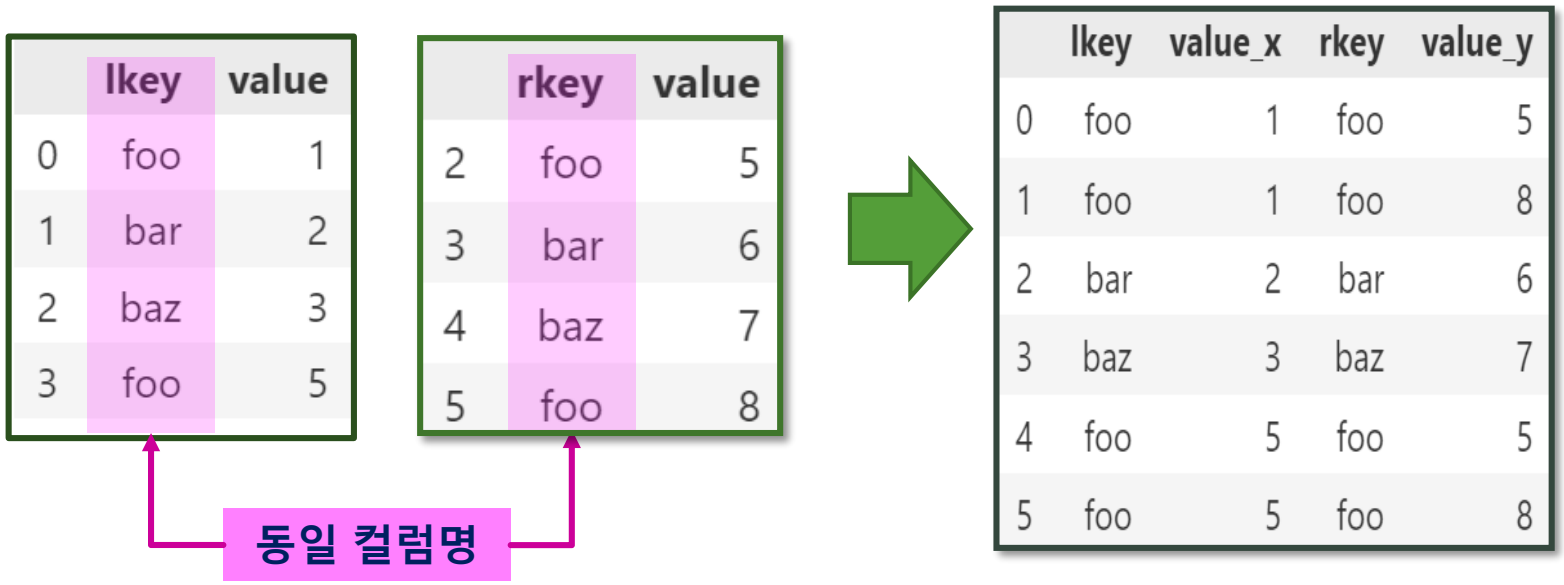
PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2 , left_on='lkey', right_on='rkey')

열 방향 , **서로 다른 열 기준 병합**



PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 병합

result1 = pd.merge(df1, df2 , left_index=True, right_index=True)

열 방향 , **행 인덱스 기준 병합**

	lkey	value
0	foo	1
1	bar	2
2	baz	3
3	foo	5

	rkey	value
2	foo	5
3	bar	6
4	baz	7
5	foo	8

동일 컬럼명

	lkey	value_x	rkey	value_y
2	baz	3	foo	5
3	foo	5	bar	6

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 결합 - join

- **merge()함수** 기반으로 만들어져서 **기본 작동 방식 비슷**
- **행 인덱스(index) 기준으로 옆으로(열을) 붙이는 간단한 결합**
- on=keys 옵션 존재
- Series는 보통 DataFrame에 붙이는 용도

- 동작 방식

  → 왼쪽 DF의 인덱스와 other의 인덱스를 기준으로 결합.

PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 결합 - join

```
DataFrame.join( other,          # DataFrame/Series/리스트/튜플
                on=None,        # 왼쪽 DF의 '열'을 조인 키로 사용
                how='left',     # 'left'|'right'|'outer'|'inner' (기본 left)
                lsuffix='',     # 중복 열 이름 충돌 시 왼쪽 접미사
                rsuffix='',     # 오른쪽 접미사
                sort=False,     # 키 기준 정렬 여부
                validate=None   # '1:1', '1:m', 'm:1', 'm:m' 유효성 체크 )
```

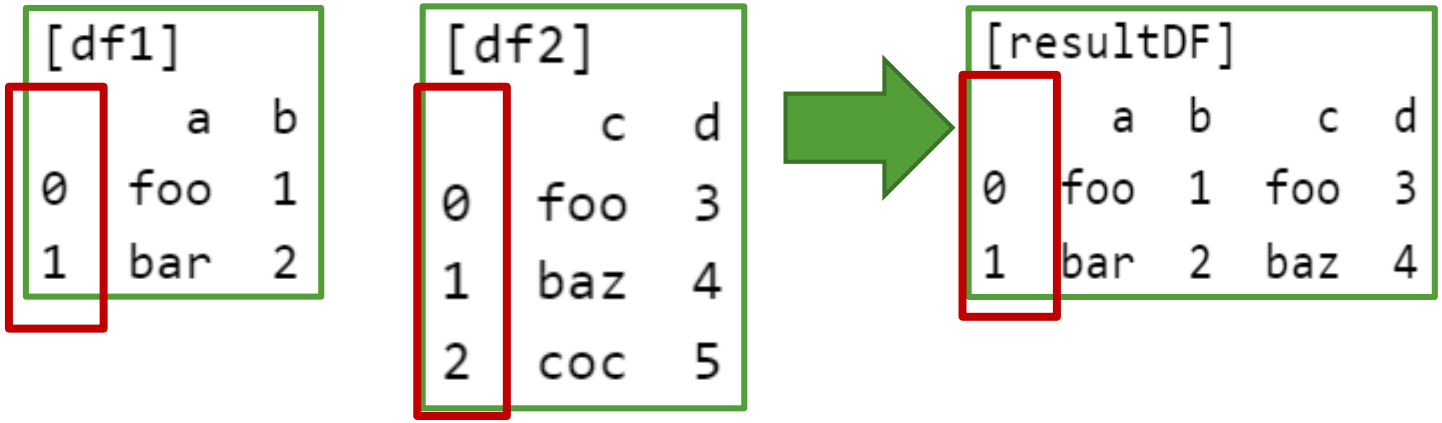
PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 결합 - join

result1 = df1.join(df2 , how='left')

열 방향 , 왼쪽 기준 집합



PANDAS 데이터 응용

◆ 데이터프레임 합치기

▪ 결합 - join

result1 = df1.join(df2 , how='right')

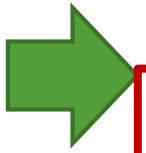
열 방향 , 오른쪽 기준 집합

[df1]

	a	b
0	foo	1
1	bar	2

[df2]

	c	d
0	foo	3
1	baz	4
2	coc	5



[resultDF]

	a	b	c	d
0	foo	1.0	foo	3
1	bar	2.0	baz	4
2	NaN	NaN	coc	5

데이터 응용 – 재배치/재구성

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ 형식 형태 변환

- **DataFrame의 행 인덱스, 열 인덱스, 값 속성 설정으로 형태 변경**
- 엑셀의 피벗 기능과 유사
- **긴(long) 형식 데이터 >>> 넓은(wide) 형식으로 변경**
- **주의!!! index+columns 조합은 고유해야 함!!**

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ 종류

항 목	pivot()	pivot_table()
목 적	• 모양만 바꾸기	• 집계까지 포함해 피벗
중복 키 (행×열 조합)	• 중복 불허 • 있으면 에러	• 자동 집계로 처리 • 중복 허용
기본 집계	• 없음	• 기본 mean(평균)
대표 옵션	• index, columns, values	• index, columns, values, fill_value • aggfunc,margins, margins_name
다중 집계	• 불가	• 여러 함수 가능 • (예: ['mean','sum'])

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot( )

DataFrame.pivot(index=None, columns=None, values=None)

- index　　새로운 행 인덱스로 사용할 열 이름
- columns　　새로운 열 인덱스로 사용할 열 이름
- values　　피벗된 테이블에 채워질 값 (숫자열 등)

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot_table( )

DataFrame.pivot_table(df, index=None, columns=None,
values=None, aggfunc='mean', fill_value=0)

- index 새로운 행 인덱스로 사용할 열 이름
- columns 새로운 열 인덱스로 사용할 열 이름
- values 피벗된 테이블에 채워질 값 (숫자열 등)
- aggfunc 중복 데이터에 대한 집계 함수 (기본은 'mean')
- margins 총합(또는 평균) 행과 열을 자동으로 추가
- margins_name='All' 합계 행/열 이름 지정 (기본값 'All')

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot( )

```
## DataFrame 인스턴스 준비
df = pd.DataFrame({
    '날짜'  : ['2025-01','2025-01','2025-02','2025-02','2025-02'],
    '지점'  : ['서울','부산','서울','부산','부산'],
    '품목'  : ['A','A','A','A','B'],
    '매출'  : [100, 120, 90, 130, 70],
    '수량'  : [10,  12,  9,  13,  7]
})

## 확인
display(df)
```

	날짜	지점	품목	매출	수량
0	2025-01	서울	A	100	10
1	2025-01	부산	A	120	12
2	2025-02	서울	A	90	9
3	2025-02	부산	A	130	13
4	2025-02	부산	B	70	7

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot( )

```
## ================================================
## [재구성] 날짜-지점 표로, 값은 매출
## ================================================

## (1) 집계 계산 처리 후 테이블화 <== 중복 제거
tmp = df.groupby(['날짜','지점'], as_index=False)['매출'].sum()

## (2) 재구성
wide_ok = tmp.pivot(index='날짜',
                    columns='지점',
                    values='매출')

display(wide_ok)
```

지점	부산	서울
날짜		
2025-01	120	100
2025-02	200	90

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot_table( )

```
## ================================================
## [재구성] 날짜-지점 표로, 값은 매출
## ================================================
## 합계로 집계
pt = pd.pivot_table( df,
                     index='날짜',          # 행 인덱스
                     columns='지점',        # 열 머리
                     values='매출',         # 값
                     aggfunc='sum',         # 집계 방식(기본 mean)
                     fill_value=0           # NaN 대신 0 채우기
)

## 확인
display(pt)
```

지점	부산	서울
날짜		
2025-01	120	100
2025-02	200	90

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot_table( )

```
## [재구성] 날짜-지점 표로, 값은 매출
## ----------------------------------------
## 합계로 집계, 합계 컬럼, 행 추가
pt2 = pd.pivot_table( df,
                      index='날짜',
                      columns='지점',
                      values='매출',
                      aggfunc='sum',
                      margins=True,          # 총합/평균 열.행 추가
                      margins_name='합계',   # 라벨 이름
                      fill_value=0 )

## 확인
display(pt2)
```

지점	부산	서울	합계
날짜			
2025-01	120	100	220
2025-02	200	90	290
합계	320	190	510

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot_table( )

```
## ----------------------------------------
## [재구성] 날짜-지점 표로, 값은 매출
## ----------------------------------------
pt_multi = pd.pivot_table(
    df,
    index=['날짜'],
    columns=['지점'],
    values=['매출','수량'],
    aggfunc={'매출':['sum','mean'], '수량':['sum','mean']},
    fill_value=0
)

display(pt_multi)
```

	매출				수량			
	mean		sum		mean		sum	
지점	부산	서울	부산	서울	부산	서울	부산	서울
날짜								
2025-01	120.0	100.0	120	100	12.0	10.0	12	10
2025-02	100.0	90.0	200	90	10.0	9.0	20	9

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot_table( )

```
## 멀티인덱스 확인
print(f'columns => {pt_multi.columns}')
print(f'nlevels => {pt_multi.columns.nlevels}개')

## 인덱스 구조 확인 후, 맞게 합계 행 추가
if isinstance(pt_multi.index, pd.MultiIndex):
    sum_index = tuple(['합계'] * pt_multi.index.nlevels)
else:
    sum_index = '합계'

## 행 주가
pt_multi.loc[sum_index] = pt_multi.sum(numeric_only=True)
```

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide 형식 형태 변환 – pivot_table( )

```
## 컬럼 수준 수 맞게 합계 열 추가
n_levels = pt_multi.columns.nlevels
sum_col_name = tuple(['합계'] * n_levels)
pt_multi[sum_col_name] = pt_multi.sum(axis=1, numeric_only=True)

## 확인
display(pt_multi)
```

	매출				수량				합계
	mean		sum		mean		sum		합계
지점	부산	서울	부산	서울	부산	서울	부산	서울	합계
날짜									
2025-01	120.0	100.0	120.0	100.0	12.0	10.0	12.0	10.0	968.0
2025-02	100.0	90.0	200.0	90.0	10.0	9.0	20.0	9.0	1056.0
합계	220.0	190.0	320.0	190.0	22.0	19.0	32.0	19.0	2024.0

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Wide >>> Long 형식 형태 변환

- **넓은(wide) 형태 데이터 ➔ 길게(long)** 펼쳐서 **정형화된 형태로 변환**
- 머신러닝/시각화 등에 적합한 형식으로 **역피벗(unpivot)**

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Long 형식 형태 변환 – melt( )

```
DataFrame.melt(  id_vars=None, value_vars=None,
                 var_name=None, value_name='value' )
```

- id_vars　　녹이지 않고 유지할 열
- value_vars　　펼칠 열들 (None이면 나머지 열 모두)
- var_name　　변수명 열 이름 지정
- value_name　　값 열 이름 지정

PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Long 형식 형태 변환 – melt( )

```
import pandas as pd

data = {
    '이름': ['철수', '영희'],
    '국어': [90, 85],
    '영어': [80, 95]
}

df = pd.DataFrame(data)
```

	이름	국어	영어
0	철수	90	80
1	영희	85	95

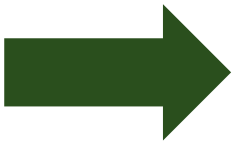
PANDAS 데이터 응용

◆ 데이터프레임 재배치/재구성

▪ Long 형식 형태 변환 – melt( )

```
# 국어/영어 점수를 긴 형식으로 변환
melted_df = df.melt( id_vars='이름',
                     value_vars=['국어', '영어'],
                     var_name='과목',
                     value_name='점수')
```

	이름	국어	영어
0	철수	90	80
1	영희	85	95



	이름	과목	점수
0	철수	국어	90
1	영희	국어	85
2	철수	영어	80
3	영희	영어	95