데이터 전처리 (2) - 행,열 병합

- 다른 데이터 프레임과 데이터 합치기 (열)
- 다른 데이터 프레임과 데이터 합치기 (행)

#01. 분석 준비

1. 패키지 참조

```
from pandas import DataFrame, read_excel
from pandas import merge, concat
```

2. 데이터 가져오기

```
고객 = read_excel("https://data.hossam.kr/C02/customer.xlsx")
고객
```

	고객번호	이름
0	1001	둘리
1	1002	도우너
2	1003	또치
3	1004	길동
4	1005	희동
5	1006	마이콜
6	1007	영희

```
매출 = read_excel('https://data.hossam.kr/C02/money.xlsx')
매출
```

	고객번호	금액
0	1001	10000

	고객번호	금액
1	1001	20000
2	1005	15000
3	1006	5000
4	1008	100000
5	1001	30000

#02. 다른 데이터 프레임과 데이터 합치기 (열)

1. 데이터 프레임 기본 병합

일치하는 데이터끼의 병합

두개의 데이터 프레임에서 이름이 동일한 컬럼을 기준으로 같은 데이터끼리 병합하고 일치하지 않는 데이터는 버려진다.

만약 양쪽 데이터프레임의 공통컬럼에 중복 데이터가 여러개 있는 경우는 모든 경우의 수를 따져서 조합을 만들어 낸다.(예: 1001번 둘리의 데이터)

> SQL의 equi 혹은 inner join과 동일

```
merge(고객, 매출)
```

	고객번호	이름	금액
0	1001	둘리	10000
1	1001	둘리	20000
2	1001	둘리	30000
3	1005	희동	15000
4	1006	마이콜	5000

왼쪽 데이터 프레임을 기준으로 병합

고객(왼쪽) 데이터를 기준으로 일치하는 매출 데이터를 병합한다.

고객 데이터의 모든 항목에 대한 출력이 보장된다.

> SQL의 LEFT OUTER JOIN과 동일

```
merge(고객, 매출, how="left")
```

	고객번호	이름	금액
0	1001	둘리	10000.0
1	1001	둘리	20000.0
2	1001	둘리	30000.0
3	1002	도우너	NaN
4	1003	또치	NaN
5	1004	길동	NaN
6	1005	희동	15000.0
7	1006	마이콜	5000.0
8	1007	영희	NaN

오른쪽 데이터 프레임을 기준으로 병합

> SQL의 RIGHT OUTER JOIN과 동일

```
merge(고객, 매출, how="right")
```

	고객번호	이름	금액
0	1001	둘리	10000
1	1001	둘리	20000
2	1005	희동	15000
3	1006	마이콜	5000
4	1008	NaN	100000
5	1001	둘리	30000

모든 데이터의 교차 병합

> SQL의 FULL OUTER JOIN과 동일

```
merge(고객, 매출, how="outer")
```

	고객번호	이름	금액
0	1001	둘리	10000.0
1	1001	둘리	20000.0
2	1001	둘리	30000.0
3	1002	도우너	NaN
4	1003	또치	NaN
5	1004	길동	NaN
6	1005	희동	15000.0
7	1006	마이콜	5000.0
8	1007	영희	NaN
9	1008	NaN	100000.0

2. 병합 대상 열 지정하기

샘플 데이터 가져오기

```
cd1 = read_excel("https://data.hossam.kr/C02/customer_data1.xlsx")
cd1
```

	고객명	날짜	데이터
0	민수	2018-01-01	20000
1	수영	2018-01-01	100000

```
cd2 = read_excel("https://data.hossam.kr/C02/customer_data2.xlsx")
cd2
```

	고객명	데이터
0	민수	21세
1	수영	20세

기본 병합

두 데이터 프레임에서 이름이 같은 열은 모두 키가 된다.

샘플 데이터에서는 `데이터`라는 이름의 변수가 df_left는 `int`, df_right는 `str`타입이므로 병합 기준을 충족하지 않는다.

그러므로 아래 코드는 에러

```
merge(cd1, cd2)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call
Cell In[11], line 1
----> 1 merge(cd1, cd2)

File c:\Users\leekh\AppData\Local\Programs\Python\Python311\Lib\site-
    131 @Substitution("\nleft : DataFrame or named Series")
    132 @Appender(_merge_doc, indents=0)
    133 def merge(
   ( ... )
    146     validate: str | None = None,
    147 ) -> DataFrame:
--> 148     op = _MergeOperation(
    149         left,
    150         right,
    151         how=how,
    152         on=on,
    153         left_on=left_on,
    154         right_on=right_on,
    155         left_index=left_index,
    156         right_index=right_index,
    157         sort=sort,
    158         suffixes=suffixes,
    159         indicator=indicator,
    160         validate=validate,
    161     )
    162     return op.get_result(copy=copy)

File c:\Users\leekh\AppData\Local\Programs\Python\Python311\Lib\site-
    733 (
    734     self.left_join_keys,
    735     self.right_join_keys,
    736     self.join_names,
    737 ) = self._get_merge_keys()
    739 # validate the merge keys dtypes. We may need to coerce
    740 # to avoid incompatible dtypes
--> 741 self._maybe_coerce_merge_keys()
```

```
    743 # If argument passed to validate,
    744 # check if columns specified as unique
    745 # are in fact unique.
    746 if validate is not None:

File c:\Users\leekh\AppData\Local\Programs\Python\Python311\Lib\site-
   1395     # unless we are merging non-string-like with string-like
   1396     elif (
   1397         inferred_left in string_types and inferred_right not
   1398     ) or (
   1399         inferred_right in string_types and inferred_left not
   1400     ):
→  1401         raise ValueError(msg)
   1403 # datetimelikes must match exactly
   1404 elif needs_i8_conversion(lk.dtype) and not needs_i8_conversic

ValueError: You are trying to merge on int64 and object columns. If y
```

병합 기준 설정

병합 기준 열이 아니면서 이름이 같은 열에는 `_x` 또는 `_y` 와 같은 접미사가 붙는다.

```
tmp = merge(cd1, cd2, on='고객명')
tmp
```

	고객명	날짜	데이터_x	데이터_y
0	민수	2018-01-01	20000	21세
1	수영	2018-01-01	100000	20세

```
tmp.rename(columns={'데이터_x': '금액', '데이터_y': '나이'})
```

	고객명	날짜	금액	나이
0	민수	2018-01-01	20000	21세
1	수영	2018-01-01	100000	20세

두 데이터 프레임의 모든 컬럼 이름이 다른 경우

왼쪽의 기준열 이름과 오른쪽의 기준열 이름을 각각 설정해야 한다.

```
df_left = DataFrame({'이름': ['영희', '철수'], '국어': [87, 91]})
df_left
```

	이름	국어
0	영희	87
1	철수	91

```
df_right = DataFrame({'성명': ['영희', '철수'], '영어': [90, 82]})
df_right
```

	성명	영어
0	영희	90
1	철수	82

```
r3 = merge(df_left, df_right, left_on=['이름'], right_on=['성명'])
r3
```

	이름	국어	성명	영어
0	영희	87	영희	90
1	철수	91	철수	82

```
r4 = r3.drop('성명', axis=1)
r4
```

	이름	국어	영어
0	영희	87	90
1	철수	91	82

인덱스를 활용한 병합

인덱스를 기준으로 한 병합

```
# 학생의 이름을 인덱스로 갖는 두 데이터 프레임
df_left = DataFrame({'수학': [90, 82]}, index=['민철', '봉구'])
df_left
```

	수학
민철	90
봉구	82

```
df_right = DataFrame({'국어': [90, 82]}, index=['민철', '철수'])
df_right
```

	국어
민철	90
철수	82

```
merge(df_left, df_right, left_index=True, right_index=True)
```

	수학	국어
민철	90	90

```
merge(df_left, df_right, left_index=True, right_index=True, how="left
```

	수학	국어
민철	90	90.0
봉구	82	NaN

```
merge(df_left, df_right, left_index=True, right_index=True, how="righ
```

	수학	국어
민철	90.0	90

	수학	국어
철수	NaN	82

```
merge(df_left, df_right, left_index=True, right_index=True, how="oute
```

	수학	국어
민철	90.0	90.0
봉구	82.0	NaN
철수	NaN	82.0

인덱스와 컬럼을 각각 기준으로 하기

```
df_left = DataFrame({'수학': [90, 82]}, index=['민철', '봉구'])
df_left
```

	수학
민철	90
봉구	82

```
df_right = DataFrame({'성명': ['민철', '철수'], '영어': [90, 82]})
df_right
```

	성명	영어
0	민철	90
1	철수	82

```
merge(df_left, df_right, left_index=True, right_on=['성명'])
```

	수학	성명	영어
0	90	민철	90

```
merge(df_left, df_right, left_index=True, right_on=['성명'], how='lef
```

	수학	성명	영어
0.0	90	민철	90.0
NaN	82	봉구	NaN

```
merge(df_left, df_right, left_index=True, right_on=['성명'], how='rig
```

	수학	성명	영어
0	90.0	민철	90
1	NaN	철수	82

```
tmp = merge(df_left, df_right, left_index=True, right_on=['성명'], ho
tmp
```

	수학	성명	영어
0.0	90.0	민철	90.0
NaN	82.0	봉구	NaN
1.0	NaN	철수	82.0

```
tmp2 = tmp.set_index('성명')
tmp2
```

	수학	영어
성명		
민철	90.0	90.0
봉구	82.0	NaN
철수	NaN	82.0

#03. 다른 데이터 프레임과 데이터 합치기 (행)

1. 행 단위 병합 기본 사용 방법

병합할 데이터 프레임을 리스트로 묶는다. (2개 이상 가능)

각 데이터프레임이 갖는 인덱스는 그대로 유지된다.(인덱스 중복 발생)

```
df1 = DataFrame({'이름': ['영희', '철수'], '국어': [87, 91]})
df1
```

	이름	국어
0	영희	87
1	철수	91

```
df2 = DataFrame({'이름': ['민철', '수현'], '국어': [78, 92]})
df2
```

	이름	국어
0	민철	78
1	수현	92

```
concat([df1, df2])
```

	이름	국어
0	영희	87
1	철수	91
0	민철	78
1	수현	92

2. 인덱스 재구성

`ignore_index=True` 파라미터를 설정하면 병합 후 인덱스를 재구성 한다.

```
concat([df1, df2], ignore_index=True)
```

	이름	국어
0	영희	87
1	철수	91
2	민철	78
3	수현	92

3. 서로 다른 변수를 갖는 데이터 프레임의 병합

```
df3 = DataFrame({'이름': ['영희', '철수'], '국어': [87, 91]})
df3
```

	이름	국어
0	영희	87
1	철수	91

```
df4 = DataFrame({'이름': ['민철', '수현'], '수학': [78, 92]})
df4
```

	이름	수학
0	민철	78
1	수현	92

```
concat([df3, df4], ignore_index=True)
```

	이름	국어	수학
0	영희	87.0	NaN
1	철수	91.0	NaN
2	민철	NaN	78.0
3	수현	NaN	92.0

3. 인덱스를 설정한 데이터 프레임의 병합

```
df5 = DataFrame({'국어': [87, 91]}, index=['영희', '철수'])
df5
```

	국어
영희	87
철수	91

```
df6 = DataFrame({'국어': [78, 92]}, index=['민철', '수현'])
df6
```

	국어
민철	78
수현	92

```
concat([df5, df6])
```

	국어
영희	87
철수	91
민철	78
수현	92

```
concat([df5, df6], ignore_index=True)
```

	국어
0	87
1	91
2	78

	국어
3	92

데이터 프레임 병합 처리 시 참고

`merge()` 함수

- 열단위 병합
- 변수, 인덱스 모두 기준으로 설정 가능
- left_on, right_on, left_index, right_index 파라미터가 있다.
- `join()`보다 사용 범위가 넓다.

`join()` 함수

- 열단위 병합
- 인덱스만을 기준으로 설정 가능
- left_on, right_on, left_index, right_index 파라미터가 없다.
- `merge()`보다 사용 범위가 좁다

`concat()` 함수

- axis 파라미터를 사용하면 열단위 병합도 가능