2. DataFrame 선언하기

```
import pandas as pd
import numpy as np

dataset = np.array([['kor', 70], ['math', 80]])
df = pd.DataFrame(dataset, columns = ['class', 'score'])
df
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	class	score
0	kor	70
1	math	80

4. DataFrame 출력

```
# !pip install scikit-learn
from sklearn.datasets import load_iris
iris = load_iris()
iris = pd.DataFrame(iris.data, columns = iris.feature_names)
```

```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
```

```
dtypes: float64(4)
memory usage: 4.8 KB
```

```
iris.describe()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

6. DataFrame 인덱스

```
# df.index
print(df)
print(df.index)
print(list(df.index))
df.index = ['A', 'B']
print(df)
```

```
  class score
0   kor    70
1  math    80
RangeIndex(start=0, stop=2, step=1)
[0, 1]
  class score
```

```
A   kor    70
B  math    80
```

```
# df.set_index
df.set_index('class', drop=True, append=False, inplace=True)
df
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	score
class	
kor	70
math	80

7. DataFrame 컬럼명 확인 및 변경

```
print(iris.columns)
iris.columns = ['sepal length', 'sepal width', 'petal length', 'petal width']
iris.head()
```

```
Index(['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)',
       'petal width (cm)'],
      dtype='object')
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

```
iris.columns = iris.columns.str.replace(' ', '_')
iris.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

8. DataFrame 컬럼의 데이터 타입 확인 및 변경

- int, float, bool, datetime, category, object

```
print(iris.dtypes)
```

```
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
dtype: object
```

```
iris.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

```
# astype을 이용하여 데이터 타입 변경
iris['sepal_length'] = iris['sepal_length'].astype('int')
iris[['sepal_width', 'petal_length']] = \
iris[['sepal_width', 'petal_length']].astype('int')

iris.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal_length	sepal_width	petal_length	petal_width
0	5	3	1	0.2
1	4	3	1	0.2
2	4	3	1	0.2

	sepal_length	sepal_width	petal_length	petal_width
3	4	3	1	0.2
4	5	3	1	0.2

제3절. row/column 선택, 추가, 삭제

1. row/column 선택

```
import pandas as pd
from sklearn.datasets import load_iris
iris = load_iris()
iris = pd.DataFrame(iris.data, columns = iris.feature_names)
```

```
# 행 선택
iris[1:4]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2

```
# 열 선택
iris['sepal length (cm)'].head(4)
iris[['sepal width (cm)', 'sepal length (cm)']].head(4)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
```

```
    text-align: right;
}
```

	sepal width (cm)	sepal length (cm)
0	3.5	5.1
1	3.0	4.9
2	3.2	4.7
3	3.1	4.6

```
# iloc: integer location
iris.iloc[1:4]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2

```
iris.iloc[[1, 3, 5], 2:4]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	petal length (cm)	petal width (cm)

	petal length (cm)	petal width (cm)
1	1.4	0.2
3	1.5	0.2
5	1.7	0.4

```
iris.iloc[:, [True, True, False, True]]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal width (cm)
0	5.1	3.5	0.2
1	4.9	3.0	0.2
2	4.7	3.2	0.2
3	4.6	3.1	0.2
4	5.0	3.6	0.2
...	...	...	...
145	6.7	3.0	2.3
146	6.3	2.5	1.9
147	6.5	3.0	2.0
148	6.2	3.4	2.3
149	5.9	3.0	1.8

150 rows × 3 columns

```
# loc: location
iris.loc[1:3]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2

```
iris.loc[[1, 2], 'sepal length (cm)':'petal length (cm)']
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)
1	4.9	3.0	1.4
2	4.7	3.2	1.3

```
# 선택한 값 변경하기
score = pd.DataFrame({'국어': [100, 80],
                      '수학': [75, 90],
                      '영어': [90, 95],
                      },
                      index = ['장화', '홍련'])
```

```
score
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	국어	수학	영어
장화	100	75	90
홍련	80	90	95

```
# iloc으로도 되고
score.iloc[0, 1] = 100

# loc으로도 됨
score.loc['장화', '수학'] = 65
```

```
score
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	국어	수학	영어
장화	100	65	90
홍련	80	90	95

```
new_students
```

```
.dataframe tbody tr th {
    vertical-align: top;
```

```
}

.dataframe thead th {
    text-align: right;
}
```

	국어	수학	영어
콩쥐	70	65	96
팥쥐	85	100	65

```
score
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	국어	수학	영어
장화	100	65	90
홍련	80	90	95

```
new_students = pd.DataFrame({'국어': [70, 85],
                             '수학': [65, 100],
                             '영어': [96, 65]},
                            index = ['콩쥐', '팥쥐'])
# append 사라짐
score = pd.concat([score, new_students])
score
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	국어	수학	영어
장화	100	65	90
홍련	80	90	95
콩쥐	70	65	96
팥쥐	85	100	65

```
science = [80, 70, 60, 50]
score['과학'] = science
score['학년'] = 1
```

```
score
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	국어	수학	영어	과학	학년
장화	100	65	90	80	1
홍련	80	90	95	70	1
콩쥐	70	65	96	60	1
팥쥐	85	100	65	50	1

```
score['과학'] = score['과학'] + 5
score['총점'] = score['국어'] + score['수학'] + score['영어'] + score['과학']
score
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
```

```
    text-align: right;
}
```

	국어	수학	영어	과학	학년	총점
장화	100	65	90	90	1	345
홍련	80	90	95	80	1	345
콩쥐	70	65	96	70	1	301
팥쥐	85	100	65	60	1	310

3. row/column 삭제

```
print(score)
score.drop('장화', inplace=True)
score.drop(columns=['과학', '학년', '총점'], inplace=True)
print(score)
```

```
     국어   수학  영어  과학  학년   총점
장화  100   65  90  90   1  345
홍련   80   90  95  80   1  345
콩쥐   70   65  96  70   1  301
팥쥐   85  100  65  60   1  310
```

제4절.조건에 맞는 데이터 탐색 및 수정

```
import pandas as pd
names = ['장화', '홍련', '콩쥐', '팥쥐', '해님', '달님']
korean_scores = [70, 85, None, 100, None, 85]
math_scores = [65, 100, 80, 95, None, 70]
students = pd.DataFrame({'이름': names,
                         '국어': korean_scores,
                         '수학': math_scores})

students[students['이름']=='장화']
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
```

```
    text-align: right;
}
```

	이름	국어	수학
0	장화	70.0	65.0

```
students[(students['국어']>=80) & (students['수학']>=80)]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학
1	홍련	85.0	100.0
3	팥쥐	100.0	95.0

```
students.loc[6, '이름':'수학'] = ['별님', 50, 60]
students
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학
0	장화	70.0	65.0
1	홍련	85.0	100.0
2	콩쥐	NaN	80.0
3	팥쥐	100.0	95.0

	이름	국어	수학
4	해님	NaN	NaN
5	달님	85.0	70.0
6	별님	50.0	60.0

```
students.loc[(students['국어']>=80)&(students['수학']>=70), '합격'] = 'Pass'
students
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격
0	장화	70.0	65.0	nan
1	홍련	85.0	100.0	Pass
2	콩쥐	NaN	80.0	nan
3	팥쥐	100.0	95.0	Pass
4	해님	NaN	NaN	nan
5	달님	85.0	70.0	Pass
6	별님	50.0	60.0	nan

```
students.loc[students['합격']!='Pass', '합격'] = 'Fail'
students
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격
0	장화	70.0	65.0	Fail
1	홍련	85.0	100.0	Pass
2	콩쥐	NaN	80.0	Fail
3	팥쥐	100.0	95.0	Pass
4	해님	NaN	NaN	Fail
5	달님	85.0	70.0	Pass
6	별님	50.0	60.0	Fail

```
students
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격
0	장화	70.0	65.0	Fail
1	홍련	85.0	100.0	Pass
2	콩쥐	NaN	80.0	Fail
3	팥쥐	100.0	95.0	Pass
4	해님	NaN	NaN	Fail
5	달님	85.0	70.0	Pass
6	별님	50.0	60.0	Fail

```
# 하나의 컬럼에 여러 개의 조건을 두어야 할 때
import numpy as np
condition_list = [(students['국어'] >= 90), # --> A
                  (students['국어'] >= 80) & (students['국어'] < 90),  # --> B
                  (students['국어'] >= 70) & (students['국어'] < 80)]  # --> C
choice_list = ['A', 'B', 'C']
students['점수'] = np.select(condition_list, choice_list, default='F')
students
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격	점수
0	장화	70.0	65.0	Fail	C
1	홍련	85.0	100.0	Pass	B
2	콩쥐	NaN	80.0	Fail	F
3	팥쥐	100.0	95.0	Pass	A
4	해님	NaN	NaN	Fail	F
5	달님	85.0	70.0	Pass	B
6	별님	50.0	60.0	Fail	F

```
# 결측값 탐색 및 수정
print(students.isna())
print(students.isna().sum())
print(students.isna().sum(1))
```

```
      이름     국어     수학     합격     점수
0  False  False  False  False  False
1  False  False  False  False  False
2  False   True  False  False  False
3  False  False  False  False  False
4  False   True   True  False  False
5  False  False  False  False  False
6  False  False  False  False  False
이름    0
국어    2
수학    1
합격    0
점수    0
dtype: int64
0    0
1    0
2    1
3    0
```

```
4    2
5    0
6    0
dtype: int64
```

```
students.notna()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격	점수
0	True	True	True	True	True
1	True	True	True	True	True
2	True	False	True	True	True
3	True	True	True	True	True
4	True	False	False	True	True
5	True	True	True	True	True
6	True	True	True	True	True

```
# dropna
students.dropna()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격	점수

	이름	국어	수학	합격	점수
0	장화	70.0	65.0	Fail	C
1	홍련	85.0	100.0	Pass	B
3	팥쥐	100.0	95.0	Pass	A
5	달님	85.0	70.0	Pass	B
6	별님	50.0	60.0	Fail	F

```
students.dropna(thresh=4) # 결측값이 아닌 값이 4개보다 많은 행만 남기기
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	국어	수학	합격	점수
0	장화	70.0	65.0	Fail	C
1	홍련	85.0	100.0	Pass	B
2	콩쥐	NaN	80.0	Fail	F
3	팥쥐	100.0	95.0	Pass	A
5	달님	85.0	70.0	Pass	B
6	별님	50.0	60.0	Fail	F

```
# 결측값 대체
health = pd.DataFrame({'연도': [2017, 2018, 2019, 2020, 2021, 2022]
                       ,'키': [160, 162, 165, None, None, 166]
                       ,'몸무게': [53, 52, None, 50, 51, 54]
                       ,'시력': [1.2, None, 1.2, 1.2, 1.1, 0.8]
                       ,'병결': [None, None, None, 2, None, 1]})
health
```

```
.dataframe tbody tr th {
    vertical-align: top;
}
```

```
.dataframe thead th {
    text-align: right;
}
```

	연도	키	몸무게	시력	병결
0	2017	160.0	53.0	1.2	NaN
1	2018	162.0	52.0	NaN	NaN
2	2019	165.0	NaN	1.2	NaN
3	2020	NaN	50.0	1.2	2.0
4	2021	NaN	51.0	1.1	NaN
5	2022	166.0	54.0	0.8	1.0

```
health.fillna(0) # 결측값 모두 0으로 채움
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	연도	키	몸무게	시력	병결
0	2017	160.0	53.0	1.2	0.0
1	2018	162.0	52.0	0.0	0.0
2	2019	165.0	0.0	1.2	0.0
3	2020	0.0	50.0	1.2	2.0
4	2021	0.0	51.0	1.1	0.0
5	2022	166.0	54.0	0.8	1.0

```
health.fillna(health.mean())  # None값을 제외하고, 수치형 자료의 평균으로 모든 결측치를 대체
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	연도	키	몸무게	시력	병결
0	2017	160.00	53.0	1.2	1.5
1	2018	162.00	52.0	1.1	1.5
2	2019	165.00	52.0	1.2	1.5
3	2020	163.25	50.0	1.2	2.0
4	2021	163.25	51.0	1.1	1.5
5	2022	166.00	54.0	0.8	1.0

```
health['병결'] = health['병결'].fillna(0)
health
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	연도	키	몸무게	시력	병결
0	2017	160.0	53.0	1.2	0.0
1	2018	162.0	52.0	NaN	0.0
2	2019	165.0	NaN	1.2	0.0
3	2020	NaN	50.0	1.2	2.0
4	2021	NaN	51.0	1.1	0.0
5	2022	166.0	54.0	0.8	1.0

```
health['몸무게'] = health['몸무게'].fillna(health['몸무게'].mean())
health
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	연도	키	몸무게	시력	병결
0	2017	160.0	53.0	1.2	0.0
1	2018	162.0	52.0	NaN	0.0
2	2019	165.0	52.0	1.2	0.0
3	2020	NaN	50.0	1.2	2.0
4	2021	NaN	51.0	1.1	0.0
5	2022	166.0	54.0	0.8	1.0

```
health.fillna(method='pad', inplace=True) # 결측값이 나오기 전 바로 앞에 나온 값으로 대체
health

# health['키'].ffill()
# health['시력'].bfill()
```

```
C:\Users\thdus\AppData\Local\Temp\ipykernel_27324\2472544055.py:1: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a future version. Use obj.ffill() or obj.bfill() instead.
  health.fillna(method='pad', inplace=True) # 결측값이 나오기 전 바로 앞에 나온 값으로 대체
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
```

```
    text-align: right;
}
```

	연도	키	몸무게	시력	병결
0	2017	160.0	53.0	1.2	0.0
1	2018	162.0	52.0	1.2	0.0
2	2019	165.0	52.0	1.2	0.0
3	2020	165.0	50.0	1.2	2.0
4	2021	165.0	51.0	1.1	0.0
5	2022	166.0	54.0	0.8	1.0

```
# 중복행 삭제
health['키'].drop_duplicates()
```

```
0    160.0
1    162.0
2    165.0
5    166.0
Name: 키, dtype: float64
```

```
health[['시력', '병결']].drop_duplicates()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	시력	병결
0	1.2	0.0
3	1.2	2.0
4	1.1	0.0

	시력	병결
5	0.8	1.0

제5절. 데이터 정렬

```
import pandas as pd
from sklearn.datasets import load_iris
iris = load_iris()
iris = pd.DataFrame(iris.data, columns=iris.feature_names)
iris
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

```
# index 정렬
iris.sort_index(ascending=False, inplace=True)
iris.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
149	5.9	3.0	5.1	1.8
148	6.2	3.4	5.4	2.3
147	6.5	3.0	5.2	2.0
146	6.3	2.5	5.0	1.9
145	6.7	3.0	5.2	2.3

```
iris.sort_index(axis=1, ascending=True, inplace=True) # 컬럼명을 기준으로 sorting
iris.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	petal length (cm)	petal width (cm)	sepal length (cm)	sepal width (cm)
149	5.1	1.8	5.9	3.0
148	5.4	2.3	6.2	3.4
147	5.2	2.0	6.5	3.0
146	5.0	1.9	6.3	2.5
145	5.2	2.3	6.7	3.0

```
# 값 정렬
iris.sort_values('petal length (cm)', inplace=True)
```

```
iris.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	petal length (cm)	petal width (cm)	sepal length (cm)	sepal width (cm)
22	1.0	0.2	4.6	3.6
13	1.1	0.1	4.3	3.0
14	1.2	0.2	5.8	4.0
35	1.2	0.2	5.0	3.2
16	1.3	0.4	5.4	3.9

```
iris.sort_values(['petal length (cm)', 'sepal length (cm)'], inplace=True)
iris
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	petal length (cm)	petal width (cm)	sepal length (cm)	sepal width (cm)
22	1.0	0.2	4.6	3.6
13	1.1	0.1	4.3	3.0
35	1.2	0.2	5.0	3.2
14	1.2	0.2	5.8	4.0
38	1.3	0.2	4.4	3.0
...	...	...	...	...

	petal length (cm)	petal width (cm)	sepal length (cm)	sepal width (cm)
131	6.4	2.0	7.9	3.8
105	6.6	2.1	7.6	3.0
117	6.7	2.2	7.7	3.8
122	6.7	2.0	7.7	2.8
118	6.9	2.3	7.7	2.6

150 rows × 4 columns

제6절. 데이터 결합

1. 단순 연결

```
import pandas as pd
HR1 = pd.DataFrame({'이름': ['장화', '홍련'],
                    '부서': ['영업', '회계'],
                    '직급': ['팀장', '사원']})
HR2 = pd.DataFrame({'이름': ['콩쥐', '팥쥐'],
                    '부서': ['사원', '팀장'],
                    '직급': ['영업', '인사']})
HR3 = pd.DataFrame({'이름': ['콩쥐', '팥쥐'],
                    '부서': ['영업', '인사'],
                    '급여': [3500, 2800]})

pd.concat([HR1, HR2], axis=0)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	부서	직급
0	장화	영업	팀장
1	홍련	회계	사원
0	콩쥐	사원	영업
1	팥쥐	팀장	인사

```
pd.concat([HR1, HR2], axis=0, ignore_index=True)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	부서	직급
0	장화	영업	팀장
1	홍련	회계	사원
2	콩쥐	사원	영업
3	팥쥐	팀장	인사

```
pd.concat([HR1, HR3], axis=0, ignore_index=True)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	부서	직급	급여
0	장화	영업	팀장	NaN
1	홍련	회계	사원	NaN
2	콩쥐	영업	NaN	3500.0
3	팥쥐	인사	NaN	2800.0

```
HR4 = pd.Series({1: 2500}, name = '급여')
pd.concat([HR1, HR4], axis=1)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	부서	직급	급여
0	장화	영업	팀장	NaN
1	홍련	회계	사원	2500.0

```
HR5 = pd.DataFrame({'급여': [4500, 3000, 3500]})
pd.concat([HR1, HR5], axis=1)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	이름	부서	직급	급여
0	장화	영업	팀장	4500
1	홍련	회계	사원	3000
2	NaN	NaN	NaN	3500

2. 조인

```
product = pd.DataFrame({'상품코드': ['G1', 'G2', 'G3', 'G4'],
                        '상품명': ['우유', '감자', '빵', '치킨']})
sale = pd.DataFrame({'주문번호': [1001, 1002, 1002, 1003, 1004],
                     '상품코드': ['G4', 'G3', 'G1', 'G3', 'G5'],
                     '주문수량': [1, 44, 2, 2, 3]})

# inner join
sale.merge(product, on='상품코드', how='inner')
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	주문번호	상품코드	주문수량	상품명
0	1001	G4	1	치킨
1	1002	G3	44	빵
2	1003	G3	2	빵
3	1002	G1	2	우유

```
# outer join
sale.merge(product, on='상품코드', how='outer', sort=True)   # sort 는 기준컬럼을 기준
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	주문번호	상품코드	주문수량	상품명
0	1002.0	G1	2.0	우유
1	NaN	G2	NaN	감자
2	1002.0	G3	44.0	빵
3	1003.0	G3	2.0	빵
4	1001.0	G4	1.0	치킨
5	1004.0	G5	3.0	NaN

```
sale.merge(product, left_on='상품코드', right_on='상품코드', how='left')
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	주문번호	상품코드	주문수량	상품명
0	1001	G4	1	치킨
1	1002	G3	44	빵
2	1002	G1	2	우유
3	1003	G3	2	빵
4	1004	G5	3	NaN

제7절. 데이터 요약

1. 그룹화와 집계

```
import pandas as pd
from sklearn.datasets import load_iris
IRIS = load_iris()
iris = pd.DataFrame(IRIS.data, columns=IRIS.feature_names)
target = IRIS.target
iris['class'] = target
```

```
iris['class'] = iris['class'].map({0:'setosa', 1:'versicolor', 2:'virginia'})
```

```
iris
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	class
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginia
146	6.3	2.5	5.0	1.9	virginia
147	6.5	3.0	5.2	2.0	virginia
148	6.2	3.4	5.4	2.3	virginia
149	5.9	3.0	5.1	1.8	virginia

150 rows × 5 columns