Pandas를 이용한 데이터 준비

DataFrame: 행과 열로 구성된 일종의 스프레드시트

In [2]:
```
import pandas as pd

no = [20211021, 20205412, 20210578]
name = ["박형식", '공유', '아이유']
major = ['영어영문학과', '화학과', '수학과']

df = pd.DataFrame({'학번': no, '이름': name, '학과': major})
df
```

Out[2]:

	학번	이름	학과
0	20211021	박형식	영어영문학과
1	20205412	공유	화학과
2	20210578	아이유	수학과

In [4]:
```
Al_Class = [[20211021, '박형식', '영어영문학과'], [20205412, '공유', '화학과'], [202

df = pd.DataFrame(Al_Class, columns = ['학번', '이름', '학과'])
df
```

Out[4]:

	학번	이름	학과
0	20211021	박형식	영어영문학과
1	20205412	공유	화학과
2	20210578	아이유	수학과

In [6]:
```
df = pd.DataFrame([[20211021, '박형식', '영어영문학과'], [20205412, '공유', '화학과'
df
```

Out[6]:

	학번	이름	학과
0	20211021	박형식	영어영문학과
1	20205412	공유	화학과
2	20210578	아이유	수학과

txt, csv 파일 불러오기

CSV/ 콤마(,)로 구분된 파일

In [8]:
```
df = pd.read_csv('C:/Users/won/exam1.txt')
print(df)
```

```
   name   score   absent
0   kim      95        3
1  choi     100        0
2   lee      90        2
3  park      85        1
4   cho      77        5
```

CSV/ 탭(tab)으로 구분된 파일

In [9]:
```
df = pd.read_csv('C:/Users/won/exam2.txt', delimiter = "Wt")
print(df)
```

```
   name  score  absent
0   kim     95       3
1  choi    100       0
2   lee     90       2
3  park     85       1
4   cho     77       5
```

CSV/ 쉼표(,)로 구분되고 헤더가 없는 파일

In [11]:
```
df = pd.read_csv('C:/Users/won/exam3.txt', delimiter = ",", header = None)
print(df)
```

```
      0    1  2
0   kim   95  3
1  choi  100  0
2   lee   90  2
3  park   85  1
4   cho   77  5
```

Excel 파일 불러오기

Excel 데이터 가져와서 출력하기

In [12]:
```
birthData = pd.read_excel("C:/Users/won/연도별출생인구.xlsx")
birthData
```

Out[12]:

	연도	출생아수	천명당출생률
0	1951	728175	37.7
1	1952	775630	39.6
2	1953	830330	41.6
3	1954	892236	43.4
4	1955	961055	45.4
...	...	...	...
63	2014	435435	8.6
64	2015	438420	8.6
65	2016	406243	7.9
66	2017	357771	7.0
67	2018	326900	6.4

68 rows × 3 columns

info 함수로 데이터 파악하기

In [13]:
```
birthData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68 entries, 0 to 67
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   연도      68 non-null     int64  
 1   출생아수    68 non-null     int64  
 2   천명당출생률  68 non-null     float64
dtypes: float64(1), int64(2)
memory usage: 1.7 KB
```

describe 함수로 데이터 파악하기

In [14]:
```
birthData.describe()
```

Out[14]:

	연도	출생아수	천명당출생률
count	68.00000	6.800000e+01	68.000000
mean	1984.50000	7.451443e+05	22.483824
std	19.77372	2.271235e+05	12.619395
min	1951.00000	3.269000e+05	6.400000
25%	1967.75000	5.394685e+05	11.250000
50%	1984.50000	7.246800e+05	16.700000
75%	2001.25000	9.621715e+05	33.225000
max	2018.00000	1.099294e+06	45.400000

행과 열 일부 선택하기

특정 레코드 선택하기, 인덱스(index)를 활용한 슬라이싱(Slicing) 방식

In [15]:
```
birthData[0:3]
```

Out[15]:

	연도	출생아수	천명당출생률
0	1951	728175	37.7
1	1952	775630	39.6
2	1953	830330	41.6

In [16]:
```
birthData[50:55]
```

Out[16]:

	연도	출생아수	천명당출생률
50	2001	554895	11.6
51	2002	492111	10.2
52	2003	490543	10.2
53	2004	472761	9.8
54	2005	435031	8.9

In [17]:
```
birthData[:4]
```

Out[17]:

	연도	출생아수	천명당출생률
0	1951	728175	37.7
1	1952	775630	39.6
2	1953	830330	41.6
3	1954	892236	43.4

In [18]:
```
birthData[65:]
```

Out[18]:

	연도	출생아수	천명당출생률
65	2016	406243	7.9
66	2017	357771	7.0
67	2018	326900	6.4

특정 칼럼(열) 선택하기

[방식 1]

In [19]:
```
birthData.연도
```

Out[19]:
```
0     1951
1     1952
2     1953
3     1954
4     1955
      ...
63    2014
64    2015
65    2016
66    2017
67    2018
Name: 연도, Length: 68, dtype: int64
```

In [20]:
```
birthData.출생아수
```

Out[20]:
```
0     728175
1     775630
2     830330
3     892236
4     961055
      ...
63    435435
64    438420
65    406243
66    357771
67    326900
Name: 출생아수, Length: 68, dtype: int64
```

[방식 2: 따옴표, 칼럼 이름에 띄어쓰기가 있는 경우]

In [21]:
```
birthData["연도"]
```

Out[21]:
```
0     1951
1     1952
2     1953
3     1954
4     1955
      ...
63    2014
64    2015
65    2016
66    2017
67    2018
Name: 연도, Length: 68, dtype: int64
```

In [25]:
```
birthData["천명당출생률"]
```

Out[25]:
```
0     37.7
1     39.6
2     41.6
3     43.4
4     45.4
      ...
63     8.6
64     8.6
65     7.9
66     7.0
67     6.4
Name: 천명당출생률, Length: 68, dtype: float64
```

행과 열 선택하기

[방식 1]

In [26]:
```
birthData.연도[0:5]
```

Out[26]:
```
0    1951
1    1952
2    1953
3    1954
4    1955
Name: 연도, dtype: int64
```

[방식 2]

In [31]:
```
birthData["연도"][0:5]
```

Out[31]:
```
0    1951
1    1952
2    1953
3    1954
4    1955
Name: 연도, dtype: int64
```

[방식 1]

In [32]:
```
birthData.출생아수[10:15]
```

Out[32]:
```
10    1099164
11    1089951
12    1075203
13    1057241
14    1040544
Name: 출생아수, dtype: int64
```

[방식 2]

In [33]:
```
birthData["출생아수"][10:15]
```

Out[33]:
```
10    1099164
11    1089951
12    1075203
13    1057241
14    1040544
Name: 출생아수, dtype: int64
```

여러 개의 칼럼(열 선택하기)

대괄호가 2개 사용

In [34]:
```
birthData[ ["연도", "출생아수"]]
```

Out[34]:

	연도	출생아수
0	1951	728175
1	1952	775630
2	1953	830330
3	1954	892236
4	1955	961055
...	...	...
63	2014	435435
64	2015	438420
65	2016	406243
66	2017	357771
67	2018	326900

68 rows × 2 columns

응용: 행과 열의 선택 / 새로운 DF로 저장

In [36]:
```
df2 = birthData.출생아수[0:5]
df2
```

Out[36]:
```
0    728175
1    775630
2    830330
3    892236
4    961055
Name: 출생아수, dtype: int64
```

In [39]:
```
df3 = birthData[ ["연도", "출생아수"] ][10:15]
df3
```

Out[39]:

	연도	출생아수
10	1961	1099164
11	1962	1089951
12	1963	1075203
13	1964	1057241
14	1965	1040544

조건에 맞는 데이터 선택하기

query 질의 함수 활용하기

1990년부터 2000년까지의 데이터

In [40]:
```
birthData.query('1990<=연도<=2000')
```

Out[40]:

	연도	출생아수	천명당출생률
39	1990	649738	15.2
40	1991	709275	16.4
41	1992	730678	16.7
42	1993	715826	16.0
43	1994	721185	16.0
44	1995	715020	15.7
45	1996	691226	15.0
46	1997	668344	14.4
47	1998	634790	13.6
48	1999	614233	13.0
49	2000	634501	13.3

2000년 이후 50만명 이상 출생한 연도 데이터

In [42]:
```
birthData.query("연도>=2000 and 출생아수>=50000")
```

Out[42]:

	연도	출생아수	천명당출생률
49	2000	634501	13.3
50	2001	554895	11.6
51	2002	492111	10.2
52	2003	490543	10.2
53	2004	472761	9.8
54	2005	435031	8.9
55	2006	448153	9.2
56	2007	493189	10.0
57	2008	465892	9.4
58	2009	444849	9.0
59	2010	470171	9.4
60	2011	471265	9.4
61	2012	484550	9.6
62	2013	436455	8.6
63	2014	435435	8.6
64	2015	438420	8.6
65	2016	406243	7.9
66	2017	357771	7.0
67	2018	326900	6.4