7. pandas –데이터처리

이번시간에 학습할 내용은..

데이터 읽어오는 방법, 데이터 조작을 위한 작업에 대해 학습해요.

1. pandas에서 파일 읽어오기

➢ pandas 패키지에 일부형식의 파일을 간편하게 읽어올 수 있는 기능 제공

* 텍스트 데이터를 데이터프레임 구조로 읽어오는 주요 메소드

메소드	기능
read_csv	csv형식 포함한 텍스트 파일을 읽어옴.(구분자는 쉼표(,)기본값)
read_table	일반 텍스트파일을 읽어옴(기본 구분자는 tab(\t))
read_excel	엑셀파일을 읽어옴

➢ read_csv 이용한 파일 처리예

1. 데이터프레임 생성하는 작업

```
In [1]: import pandas as pd

In [5]: mysource={'시도':['서울','경기','인천','부산','대전'],
                  '구분':['특별시','도','광역시','광역시','광역시'],
                  '인구':['990만명','1300만명','350만명','300만명','150만명'],
                  '면적':[605.2,"10,171","1,029",767.4,539.8]}
```

```
In [6]: df=pd.DataFrame(mysource)
        df

Out[6]:
```

	구분	면적	시도	인구
0	특별시	605.2	서울	990만명
1	도	10,171	경기	1300만명
2	광역시	1,029	인천	350만명
3	광역시	767.4	부산	300만명
4	광역시	539.8	대전	150만명

2. 데이터프레임을 csv 형식으로 저장하는 작업을 아래와 같이 실행해요.

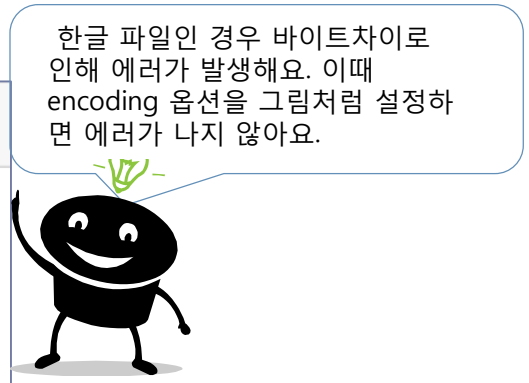
```
In [7]: df.to_csv('sample.csv')
```

3. 저장한 csv파일을 로딩해보아요.

```
In [11]: df1 = pd.read_csv('sample.csv',encoding='cp949')
         df1
```

Out[11]:

	Unnamed: 0	구분	면적	시도	인구
0	0	특별시	605.2	서울	990만명
1	1	도	10,171	경기	1300만명
2	2	광역시	1,029	인천	350만명
3	3	광역시	767.4	부산	300만명
4	4	광역시	539.8	대전	150만명



4. 컬럼을 사용자가 재정의하여 읽어올 수 있어요.

```
In [22]: df1=pd.read_csv('sample.csv',names=['id','구분','면적','시도','인구'],encoding='cp949')
         df1
```

Out[22]:

	id	구분	면적	시도	인구
0	NaN	구분	면적	시도	인구
1	0.0	특별시	605.2	서울	990만명
2	1.0	도	10,171	경기	1300만명
3	2.0	광역시	1,029	인천	350만명
4	3.0	광역시	767.4	부산	300만명
5	4.0	광역시	539.8	대전	150만명

5. 특정 컬럼을 색인으로 하여 데이터프레임을 생성할 수 있어요.

```
In [24]: df1=pd.read_csv('sample.csv',index_col='id',names=['id','구분','면적','시도','인구'],
                         encoding='cp949',keep_default_na=False)
         df1
```

Out[24]:

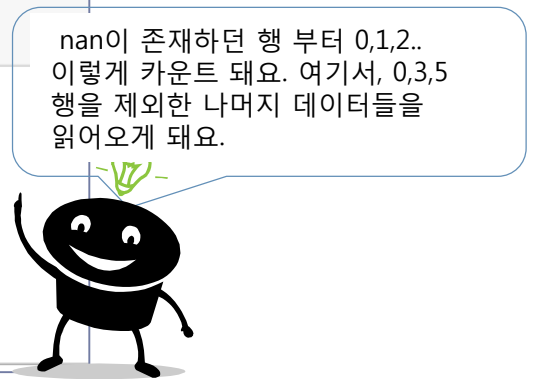
id	구분	면적	시도	인구
0	특별시	605.2	서울	990만명
1	도	10,171	경기	1300만명
2	광역시	1,029	인천	350만명
3	광역시	767.4	부산	300만명
4	광역시	539.8	대전	150만명

6. 특정 행을 제외하고 파일을 읽어올 수 있어요.

```
In [28]: df1=pd.read_csv('sample.csv',index_col='id',names=['id','구분','면적','시도','인구'],
                         encoding='cp949',skiprows=[0,3,5])
         df1
```

Out[28]:

id	구분	면적	시도	인구
0	특별시	605.2	서울	990만명
1	도	10,171	경기	1300만명
3	광역시	767.4	부산	300만명



참고) JSON 형식 파일 읽어오기

- JSON(JavaScript Object Notation) 은 자바스크립트로부터 파생된 데이터포맷으로, 웹브라우저와 서버간 데이터전송을 위한 표준포맷

JSON파일구조

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateNewDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Close", "onclick": "CloseDoc()"}
    ]
  }
}}
```

XML 형식으로 이렇게 표현할 수 있어요.

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

➢ json파일을 읽어오는 방법

```
In [1]: import json

In [2]: d1=json.load(open('restaurantinfo.json'))
        d1

Out[2]: {'food_name': {'best-of-best': 'bibimbab', 'one-of-best': 'bulgogi'},
         'kind1': 'koreanfood',
         'region': ['nosong', 'bibim', 'jungsung']}

In [4]: d1.keys()

Out[4]: dict_keys(['kind1', 'region', 'food_name'])
```

읽어올 json파일 경로를 지정하여 파일을 열어준다음 로딩

사전형 구조이므로, 키와 value값들로 구성되어 출력되는 것을 알 수 있어요.

2. 데이터처리를 위한 조작작업

➢ 분석에 필요한 데이터만!!- 데이터 필터링 작업

```
In [3]: import pandas as pd

In [5]: df=pd.read_excel('highwaybus.xlsx',encoding='cp949')
        df.head(5)
```

Out[5]:

	차종	선별	출발지	도착지	거리	총운행횟수	총이용인원	이용율
0	우등	88선	광주	울산	327.8	412	7283	63.1
1	고속	88선	광주	울산	327.8	145	3050	46.7
2	우등	88선	광주	울산신복	327.8	164	545	11.9
3	고속	88선	광주	울산신복	327.8	70	311	9.9
4	우등	88선	광주	동대구	219.3	1369	21873	57.1

```
In [7]: df_ex=df[df.차종=='우등']
        df_ex.head(5)
```

Out[7]:

	차종	선별	출발지	도착지	거리	총운행횟수	총이용인원	이용율
0	우등	88선	광주	울산	327.8	412	7283	63.1
2	우등	88선	광주	울산신복	327.8	164	545	11.9
4	우등	88선	광주	동대구	219.3	1369	21873	57.1
6	우등	88선	광주	구미	246.1	208	2634	45.2
7	우등	88선	광주	경주	286.9	139	2102	54.0

우리가 분석하고자 하는 대상이 "우등"고속버스에 대한 것이라면 불필요한 행은 제거해야겠죠?

그럼 우등고속을 대상으로 이용인원이 1000이상인 데이터를 분석하려면요?

이렇게 각 조건을 괄호로 묶은 다음 두 조건식을 연결하는 논리연산자 '&'를 쓰면 돼요.

```
In [12]: df_ge=df[(df.선별=='경부선') & (df.총이용인원>=1000)]
         df_ge.head()
```

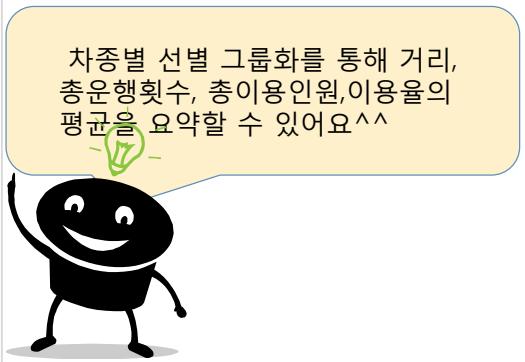
Out[12]:

	차종	선별	출발지	도착지	거리	총운행횟수	총이용인원	이용율
18	고속	경부선	서울경부	안성	77.8	3541	99277	62.3
19	고속	경부선	서울경부	안성중대	74.0	1744	29615	37.7
20	고속	경부선	서울경부	평택	69.8	2770	72167	57.9
21	우등	경부선	서울경부	대전복합	153.2	3555	77759	78.1
22	고속	경부선	서울경부	대전복합	153.2	1507	64208	94.7

➢ 그룹화 작업

	차종	선별	출발지	도착지	거리	총운행횟수	총이용인원	이용율
2	우등	88선	광주	울산	327.8	412	7283	63.1
3	고속	88선	광주	울산	327.8	145	3050	46.7
4	우등	88선	광주	울산신복	327.8	164	545	11.9
5	고속	88선	광주	울산신복	327.8	70	311	9.9
6	우등	88선	광주	동대구	219.3	1369	21873	57.1
7	고속	88선	광주	동대구	219.3	192	3732	43.2
8	우등	88선	광주	구미	246.1	208	2634	45.2
9	우등	88선	광주	경주	286.9	139	2102	54.0
10	우등	88선	광주	포항	303.2	234	3931	60.0
11	고속	88선	광주	포항	303.2	60	837	31.0
12	우등	88선	목포	동대구	295.7	131	1447	39.4
13	우등	88선	목포	구미	347.4	114	1732	54.3
14	우등	88선	전주	부산	265.1	752	9232	43.8
15	고속	88선	전주	부산	265.1	63	910	32.1
16	우등	88선	전주	울산	308.7	196	2932	53.4
17	고속	88선	전주	울산	308.7	123	2339	42.3
18	우등	88선	전주	동대구	198.5	464	7521	57.9
19	고속	88선	전주	동대구	198.5	5	132	58.7
20	고속	경부선	서울경부	안성	77.8	3541	99277	62.3
21	고속	경부선	서울경부	안성중대	74.0	1744	29615	37.7
22	고속	경부선	서울경부	평택	69.8	2770	72167	57.9
23	우등	경부선	서울경부	대전복합	153.2	3555	77759	78.1
24	고속	경부선	서울경부	대전복합	153.2	1507	64208	94.7
25	고속	경부선	서울경부	천안	84.1	5213	185241	79.0
26	우등	경부선	서울경부	공주	125.9	1633	36623	80.1
27	고속	경부선	서울경부	공주	125.9	490	13668	62.0
28	우등	경부선	서울경부	금산	188.3	241	3507	52.0
29	고속	경부선	서울경부	금산	188.3	271	6304	51.7
30	고속	경부선	서울경부	아산온양	100.0	1530	41208	59.9
31	고속	경부선	서울경부	조치원	119.1	953	20447	47.7
32	우등	경부선	서울경부	세종시	130.7	3656	63570	62.1
33	우등	경부선	서울경부	청주	123.0	3705	77586	74.8

차종	선별	거리	총운행횟수	총이용인원	이용율
고속	88선	278.628571	94.000000	1615.857143	37.700000
	경부선	235.382456	629.807018	15246.333333	48.919298
	경인선	314.527273	122.181818	2719.636364	61.190909
	구마선	316.521429	131.928571	2621.785714	46.385714
	남해선	208.138462	213.461538	3613.384615	35.130769
	동해선	277.950000	182.500000	3604.000000	52.800000
	영동선	197.671429	707.785714	18547.785714	110.192857
	호남선	287.998413	340.777778	7451.238095	47.763492
우등	88선	284.227273	380.272727	5566.545455	49.100000
	경부선	270.866667	844.333333	13786.229167	47.356250
	경인선	320.771429	618.928571	8937.071429	49.985714
	구마선	299.550000	753.687500	11092.625000	47.443750
	남해선	216.287500	668.250000	9492.125000	47.112500
	동해선	277.950000	918.500000	14627.500000	56.600000
	영동선	239.491667	1052.666667	19762.166667	61.533333
	호남선	281.391525	729.627119	13218.694915	56.003390



➢ 그룹화 연산

분석목적에 맞게 필요한 함수를 이용하여 연산

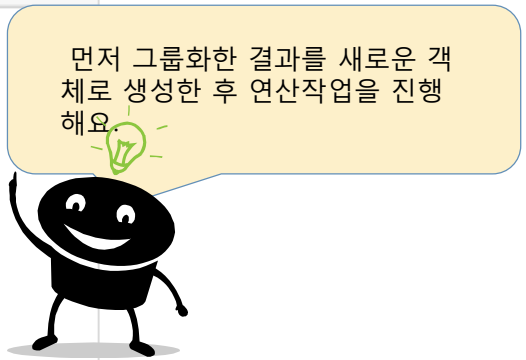
- count()-건수계산
- sum()-합계 계산
- mean()-평균계산
- max() –최댓값 계산
- first() –그룹화 작업에서 최초값 계산


```
df.groupby(df['선별']).count()
```

선별	차종	출발지	도착지	거리	총운행횟수	총이용인원	이용율
88선	18	18	18	18	18	18	18
경부선	105	105	105	105	105	105	105
경인선	25	25	25	25	25	25	25
구마선	30	30	30	30	30	30	30
남해선	21	21	21	21	21	21	21
동해선	4	4	4	4	4	4	4
영동선	26	26	26	26	26	26	26
호남선	122	122	122	122	122	122	122

```
In [18]: df1=df.groupby(df['선별'])
         df1['차종'].count()

Out[18]: 선별
         88선      18
         경부선     105
         경인선      25
         구마선      30
         남해선      21
         동해선       4
         영동선      26
         호남선     122
         Name: 차종, dtype: int64
```



➢ 빈도계산을 위한 피벗-교차표(cross_tab)

- 피벗테이블 형태로 각 그룹별 빈도계산을 할 때 유용

차종별 선별 운행노선 개수를 모두 요약하여 한 눈에 볼 수 있어요.

```
pd.crosstab(df['차종'],df['선별'])
```

선별 차종	88선	경부선	경인선	구마선	남해선	동해선	영동선	호남선
고속	7	57	11	14	13	2	14	63
우등	11	48	14	16	8	2	12	59