데이터 분석과 파이썬

데이터 분석

어떤 기존 시스템 또는 계획 중인 시스템에서 **데이터**와 **데이터**의 흐름을 체계적으로 조사하는 것

출처) 네이버 지식백과

컴퓨터 언어

- 인간이 대화하기 위하여 사용하는 언어 ex)영어, 한국어, 중국어
- 인간이 컴퓨터에게 명령을 내리기 위하여 사용하는 언어 ex)파이썬, 자바, C

코딩

인간이 컴퓨터에게 명령을 내리기 위해서 코드를 작성하는 행동

파이썬의 특징

- 언어들 중 배우기가 쉬운 편에 속합니다.
- 데이터를 다루는데 용이합니다.
- 다양한 모듈/패키지가 존재하여 활용범위가 넓습니다.

변수

데이터를 저장/관리 할 공간이 필요

부동산 데이터를 활용하여 주택 가격 변화 추이, 아파트별 전세 가격 현황 등을 분석하고자 한다면, 부동산 데이터들을 저장할 공간이 필요합니다.

이렇듯 데이터를 저장하는 공간을 코딩에서는 **변수**라고 부릅니다.

변수 만드는 방법

a = 6

데이터를 저장하는 공간 이름 — 저장할 데이터

변수이름, **변수명** — **변수값**

변수명은 단순히 데이터를 저장하는 공간을 부르는 명칭으로 코딩하는 사람 마음대로 지을 수 있습니다.
다만, 아래 규칙을 지켜야 합니다.

<변수명 작성 규칙>

- `A = 6` : (o) 대/소문자를 구분합니다
- `a b = 6` : (x) 띄어쓰기는 할 수 없습니다.
- `a_b = 6` : (o) 특수기호는 밑줄 기호만 사용할 수 있습니다.
- `1a = 6` : (x) 숫자가 맨 앞으로 오거나 숫자 단독으로 사용할 수 없습니다.
- `a1 = 6` : (o) 맨 앞에 문자가 있으면 숫자를 사용할 수 있습니다.

자료형

데이터는 종류가 다양하기 때문에 저장할 데이터의 형태에 따라 변수의 형태도 달라집니다.
이를 **자료형**이라고 합니다.

파이썬에서는 아래와 같이 데이터의 형태에 따라 다양한 자료형이 존재합니다.

- 1) a=6

  데이터가 **정수**일 경우 -> **int**

- 2) b=3.14

  데이터가 **실수**일 경우 -> **float**

- 3) c="python"

  데이터가 **문자열**일 경우 -> **str**

- 4) d=[1, 2, 3, 4, 5]

  데이터가 **리스트**일 경우 -> **list**

- 5) e={"사과":"apple", "포도":"grape", "오렌지":"orange"}

  데이터가 **딕셔너리**일 경우 -> **dict**

- 6) f=True

  데이터가 **부울**일 경우 -> **bool**

정수/실수/문자열

- 1) a=6

  데이터가 정수일 경우 -> int
- 2) b=3.14

  데이터가 실수일 경우 -> float

a=5, b=2	사용법	결과
더하기	a + b	7
빼기	a – b	3
곱하기	a * b	10
나누기	a / b	2.5
몫 구하기	a // b	2
나머지 구하기	a % b	1
제곱 구하기	a ** b	25

- 3) c="python"

  데이터가 문자열일 경우 -> str

문자는 따옴표로 묶어주어야 합니다.
(작은따옴표, 큰따옴표 무관)

c="데이터분석" d="시작"	사용법	결과
문자 이어주기	c + d	데이터분석시작
글자 가져오기	c[0]	데
	c[1]	이
	c[1:4]	이터분
	c[-1]	석

리스트/딕셔너리

- 4) d=[1, 2, 3, 4, 5]
  데이터가 리스트일 경우 -> list

d=[1,2,3,4,5]	사용법	결과
값 가져오기	d[0]	1
	d[1]	2
	d[1:4]	[2,3,4]
	d[-1]	5
값 수정하기	d[0]=0	[0,2,3,4,5]
리스트 합치기	d + [6,7,8]	[0,2,3,4,5,6,7,8]
값 추가하기	d.append(9)	[0,2,3,4,5,9]

- 5) e={“가”:1, “나”:2, “다”:[3,4,5]}
  데이터가 딕셔너리일 경우 -> dict

	사용법	결과
값 가져오기	e["가"]	1
글자 가져오기	e["다"]	[3,4,5]
값 수정하기	e["나"]=6	{"가":1, "나":6, "다":[3,4,5]}
값 추가하기	e["라"]=7	{"가":1,"나":6, "다":[3,4,5], " 라":7}

판다스 (pandas)

판다스는 데이터를 분석할 수 있는 다양한 기능을 가진 모듈을 말합니다.
‘엑셀’과 유사하나
1) 엑셀보다 복잡한 기능을 쉽게 사용 가능하고, 2) 대용량의 데이터를 처리할 수 있습니다.

판다스에서는 데이터의 형태에 따라 다음과 같이 자료형을 표현합니다.

- 해당 컬럼의 데이터가 **정수**일 경우 -> **int64**
- 해당 컬럼의 데이터가 **실수**일 경우 -> **float64**
- 해당 컬럼의 데이터가 **문자열**일 경우 -> **object**
- 해당 컬럼의 데이터가 **날짜**일 경우 -> **datetime**
- 데이터가 **1차원 자료구조일** 경우 -> **Series**
- 데이터가 **2차원 자료구조일** 경우 -> **DataFrame**

데이터의 구조

인덱스

열 (column)

행 (row)

	지역명	규모구분	연도	월	분양가격(㎡)
0	서울	전체	2015	10	5841
1	서울	전용면적 60㎡이하	2015	10	5652
2	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882
3	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721
4	서울	전용면적 102㎡초과	2015	10	
5	인천	전체	2015	10	3163
6	인천	전용면적 60㎡이하	2015	10	3488
7	인천	전용면적 60㎡초과 85㎡이하	2015	10	3119

null

Series 만들기

Series : 1차원 자료구조

```
In [16]: import pandas as pd
```

pandas 모듈 불러오기

```
In [17]: se=pd.Series([10,20,30,40,50])
```

리스트로 시리즈를 만들어서 변수 se에 저장하기

```
In [18]: se
```

변수 se 출력하기

```
Out[18]: 0    10
         1    20
         2    30
         3    40
         4    50
         dtype: int64
```

```
In [19]: type(se)
```

변수 se의 형태(타입) 확인하기

```
Out[19]: pandas.core.series.Series
```

Dataframe 만들기

Dataframe : 2차원 자료구조

In [26]:
```
df=pd.DataFrame({"가":[11,12,13,14,15],
                 "나":[21,22,23,24,25],
                 "다":[31,32,33,34,35]})
```

딕셔너리로 데이터프레임을 만들어서
변수 df에 저장하기

In [27]:
```
df
```

변수 df 출력하기

Out[27]:

	가	나	다
0	11	21	31
1	12	22	32
2	13	23	33
3	14	24	34
4	15	25	35

In [29]:
```
type(df)
```

변수 df의 형태(타입) 확인하기

Out[29]:
```
pandas.core.frame.DataFrame
```

파일 불러오기

```
In [13]: import pandas as pd
```

```
In [14]: df = pd.read_csv('C:/Users/fastj/Downloads/house_price.csv', engine='python')
```

```
In [15]: df
```

Out[15]:

	지역명	규모구분	연도	월	분양가격(㎡)
0	서울	전체	2015	10	5841
1	서울	전용면적 60㎡이하	2015	10	5652
2	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882
3	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721
4	서울	전용면적 102㎡초과	2015	10	5879
5	인천	전체	2015	10	3163
6	인천	전용면적 60㎡이하	2015	10	3488
7	인천	전용면적 60㎡초과 85㎡이하	2015	10	3119

```
In [16]: type(df)
```

```
Out[16]: pandas.core.frame.DataFrame
```

csv파일 읽어와서
변수 df에 저장하기

데이터 살펴보기 – 상위/하위 별

```
In [78]: df.head()
```

Out[78]:

	지역명	규모구분	연도	월	분양가격(㎡)
0	서울	전체	2015	10	5841
1	서울	전용면적 60㎡이하	2015	10	5652
2	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882
3	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721
4	서울	전용면적 102㎡초과	2015	10	5879

상위 5개 데이터 보기

```
In [79]: df.head(10)
```

Out[79]:

상위 10개 데이터 보기

```
In [81]: df.tail()
```

Out[81]:

하위 5개 데이터보기

```
In [85]: df.shape
```

Out[85]: (3655, 5)

데이터의 행과 열 개수 확인하기

데이터 살펴보기 – 열(column) 별

```
In [97]: df["지역명"]
Out[97]: 0        서울
```

"지역명" 열의 데이터 보기

```
In [100]: col=["지역명","규모구분","연도"]
          df[col]
```

Out[100]:

	지역명	규모구분	연도
0	서울	전체	2015

"지역명", "규모구분",
"연도" 열들의 데이터 보기

데이터 살펴보기 – 행(row) 별

```
In [102]: df.loc[1]
```

```
Out[102]: 지역명                  서울
          규모구분        전용면적 60㎡이하
          연도                  2015
          월                      10
          분양가격(㎡)          5652
          Name: 1, dtype: object
```

1 행의 데이터 보기

```
In [104]: df.loc[1:5]
```

Out[104]:

	지역명	규모구분	연도	월	분양가격(㎡)
1	서울	전용면적 60㎡이하	2015	10	5652
2	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882
3	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721
4	서울	전용면적 102㎡초과	2015	10	5879
5	인천	전체	2015	10	3163

1행부터 5행까지의 데이터 보기

```
In [106]: df.reindex([1, 3, 5])
```

Out[106]:

	지역명	규모구분	연도	월	분양가격(㎡)
1	서울	전용면적 60㎡이하	2015	10	5652
3	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721
5	인천	전체	2015	10	3163

1행, 3행, 5행의 데이터 보기

데이터 살펴보기 – 행(row) & 열(column) 별

```
In [108]: df.loc[1, "지역명"]
Out[108]: '서울'
```

1행 & “지역명” 열의 데이터 보기

```
In [109]: df.loc[1, ["지역명","규모구분","연도"]]
Out[109]: 지역명              서울
          규모구분    전용면적 60㎡이하
          연도              2015
          Name: 1, dtype: object
```

1행 & "지역명", “규모구분”, “연도” 열들의 데이터 보기

```
In [110]: df.loc[[1,3,5], "지역명"]
Out[110]: 1    서울
          3    서울
          5    인천
          Name: 지역명, dtype: object
```

1행, 3행, 5행 & "지역명" 열의 데이터 보기

```
In [116]: df.loc[1:5,"지역명"]
Out[116]: 1    서울
          2    서울
          3    서울
          4    서울
          5    인천
          Name: 지역명, dtype: object
```

1행 ~ 5행 & "지역명" 열의 데이터 보기

```
In [117]: df.loc[[1, 3, 5], ["지역명","규모구분","연도"]]
```

Out[117]:

	지역명	규모구분	연도
1	서울	전용면적 60㎡이하	2015
3	서울	전용면적 85㎡초과 102㎡이하	2015
5	인천	전체	2015

1행, 3행, 5행 & "지역명", “규모구분”, “연도” 열들의 데이터 보기

컬럼 이름과 형태 바꾸기

In [166]:
```
df.dtypes
```

df의 데이터의 컬럼별 형태 보기

Out[166]:
```
지역명          object
규모구분         object
연도            int64
월             int64
분양가격(㎡)      object
dtype: object
```

In [167]:
```
df=df.rename(columns={'분양가격(㎡)':'분양가'})
```

컬럼 이름 바꾸기

In [157]:
```
df['분양가'] = df['분양가'].convert_objects(convert_numeric=True)
```

컬럼의 형태 바꾸기

In [163]:
```
df["평당분양가"]=df["분양가"]/(3.3)
```

새로운 컬럼 만들기

Out[163]:

	지역명	규모구분	연도	월	분양가	평당분양가
0	서울	전체	2015	10	5841.0	1770.000000
1	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273

컬럼 추가하기/ 위치 지정하여 추가하기

In [173]:
```
df["국가"] = "대한민국"
df[["국가"]]
```

Out[173]:

	국가
0	대한민국
1	대한민국

컬럼 추가하기

In [228]:
```
df.insert(0,"국가", "대한민국")
df
```

Out[228]:

	국가	지역명	규모구분	연도	월	분양가	평당분양가
0	대한민국	서울	전체	2015	10	5841.0	1770.000000
1	대한민국	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273

위치 지정하여
컬럼 추가하기

조건에 따라 컬럼 추가하기

In [229]:
```
df["평가"] = df["평당분양가"] > 1500
df
```

Out[229]:

	국가	지역명	규모구분	연도	월	분양가	평당분양가	평가
0	대한민국	서울	전체	2015	10	5841.0	1770.000000	True
1	대한민국	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	True
2	대한민국	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	True

In [231]:
```
df.loc[df["평당분양가"] > 1500, "평가"] = "비쌈"
df.loc[df["평당분양가"] <= 1500, "평가"] = "보통"
df
```

Out[231]:

	국가	지역명	규모구분	연도	월	분양가	평당분양가	평가
0	대한민국	서울	전체	2015	10	5841.0	1770.000000	비쌈
1	대한민국	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈
2	대한민국	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	비쌈
3	대한민국	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721.0	1733.636364	비쌈
4	대한민국	서울	전용면적 102㎡초과	2015	10	5879.0	1781.515152	비쌈
5	대한민국	인천	전체	2015	10	3163.0	958.484848	보통
6	대한민국	인천	전용면적 60㎡이하	2015	10	3488.0	1056.969697	보통

조건에 따라 컬럼 추가하기
(조건에 성립하면 True,
성립하지 않으면 False 값 넣기)

조건에 따라 컬럼 추가하기
(조건에 따라 특정 값 넣기)

조건이 2개 이상일 경우 & 활용해서 컬럼 추가하기

In [232]:
```
df.loc[df["평당분양가"] < 800, "평가"] = "저렴"
df.loc[(df["평당분양가"] >= 800) & (df["평당분양가"] <= 1500), "평가"] = "보통"
df.loc[df["평당분양가"] > 1500, "평가"] = "비쌈"
df
```

Out[232]:

	국가	지역명	규모구분	연도	월	분양가	평당분양가	평가
0	대한민국	서울	전체	2015	10	5841.0	1770.000000	비쌈
1	대한민국	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈

조건이 2개 이상일 경우, 2개의 조건 모두 참 일 때 값을 넣고 싶다면..

& 사용

조건이 2개 이상일 경우 | 활용해서 컬럼 추가하기

In [233]:
```
df.loc[(df["지역명"] == "서울") | (df["지역명"] == "인천") | (df["지역명"] == "경기") ,"광역권"]="수도권"
df
```

Out[233]:

	국가	지역명	규모구분	연도	월	분양가	평당분양가	평가	광역권
0	대한민국	서울	전체	2015	10	5841.0	1770.000000	비쌈	수도권
1	대한민국	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈	수도권
2	대한민국	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	비쌈	수도권
3	대한민국	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721.0	1733.636364	비쌈	수도권
4	대한민국	서울	전용면적 102㎡초과	2015	10	5879.0	1781.515152	비쌈	수도권
5	대한민국	인천	전체	2015	10	3163.0	958.484848	보통	수도권

조건이 2개 이상일 경우, 2개의 조건 중 하나라도 참 일 때 값을 넣고 싶다면..

사용

컬럼 확인하기 / 위치 조정하기

데이터프레임의 컬럼 확인하기

In [218]:
```
df.columns
```

Out[218]:
```
Index(['국가', '지역명', '규모구분', '연도', '월', '분양가', '평당분양가', '평가', '광역권'], dtype='object')
```

컬럼 위치 조정하기

In [234]:
```
df=pd.DataFrame(df, columns=['국가' , '광역권','지역명', '규모구분', '연도', '월', '분양가', '평당분양가', '평가'])
df
```

Out[234]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가
0	대한민국	수도권	서울	전체	2015	10	5841.0	1770.000000	비쌈
1	대한민국	수도권	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈
2	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	비쌈

컬럼 삭제하기

컬럼 삭제하기

```
In [59]: df=df.drop("국가", axis=1)
         df
```

Out[59]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜
0	수도권	서울	전체	2015	10	5841.0	1770.000000	비쌈	2015	10	2015-10-01
1	수도권	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈	2015	10	2015-10-01
2	수도권	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	비쌈	2015	10	2015-10-01
3	수도권	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721.0	1733.636364	비쌈	2015	10	2015-10-01

axis=1 → 열을 뜻함

axis=0 → 행을 뜻함

수학/통계 연산 활용하기

```
In [24]: df.describe()
```

Out[24]:

	연도	월	분양가	평당분양가
count	3655.000000	3655.000000	3424.000000	3424.000000
mean	2017.046512	6.441860	3140.904206	951.789153
std	1.098849	3.611217	1148.059496	347.896817
min	2015.000000	1.000000	1868.000000	566.060606
25%	2016.000000	3.000000	2390.000000	724.242424
50%	2017.000000	6.000000	2792.000000	846.060606
75%	2018.000000	10.000000	3408.000000	1032.727273
max	2019.000000	12.000000	8272.000000	2506.666667

해당 데이터프레임의 통계값 보기

※ 연산 가능한 컬럼들만 확인가능

```
In [25]: df["분양가"].mean()
Out[25]: 3140.904205607477
```

평균값 보기

```
In [26]: df["분양가"].max()
Out[26]: 8272.0
```

최댓값 보기

```
In [27]: df["분양가"].min()
Out[27]: 1868.0
```

최솟값 보기

index 정렬하기

```
In [28]: df.sort_index(axis=0, ascending=False)[:5]
```

Out[28]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가
3654	대한민국	NaN	제주	전용면적 102㎡초과	2019	4	3269.0	990.606061	보통
3653	대한민국	NaN	제주	전용면적 85㎡초과 102㎡이하	2019	4	3226.0	977.575758	보통
3652	대한민국	NaN	제주	전용면적 60㎡초과 85㎡이하	2019	4	3475.0	1053.030303	보통
3651	대한민국	NaN	제주	전용면적 60㎡이하	2019	4	3804.0	1152.727273	보통
3650	대한민국	NaN	제주	전체	2019	4	3441.0	1042.727273	보통

내림차순 정렬하기

```
In [29]: df.sort_index(axis=0, ascending=True)[:5]
```

Out[29]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가
0	대한민국	수도권	서울	전체	2015	10	5841.0	1770.000000	비쌈
1	대한민국	수도권	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈
2	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	비쌈
3	대한민국	수도권	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721.0	1733.636364	비쌈
4	대한민국	수도권	서울	전용면적 102㎡초과	2015	10	5879.0	1781.515152	비쌈

오름차순 정렬하기

컬럼 별로 정렬하기

```
In [30]: df.sort_values(by='지역명')[:5]
```

Out[30]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가
386	대한민국	NaN	강원	전용면적 60㎡이하	2016	2	2290.0	693.939394	저렴
385	대한민국	NaN	강원	전체	2016	2	2173.0	658.484848	저렴
1409	대한민국	NaN	강원	전용면적 102㎡초과	2017	2	2251.0	682.121212	저렴
2514	대한민국	NaN	강원	전용면적 102㎡초과	2018	3	2923.0	885.757576	보통
2513	대한민국	NaN	강원	전용면적 85㎡초과 102㎡이하	2018	3	2830.0	857.575758	보통

지역명 별로 정렬하기

```
In [32]: df.sort_values(by='분양가', ascending=False)[:5]
```

Out[32]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가
3572	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2019	4	8272.0	2506.666667	비쌈
3487	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2019	3	8191.0	2482.121212	비쌈
3402	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2019	2	8141.0	2466.969697	비쌈
3317	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2019	1	8105.0	2456.060606	비쌈
2638	대한민국	수도권	서울	전용면적 85㎡초과 102㎡이하	2018	5	8098.0	2453.939394	비쌈

분양가 별로 정렬하기

문자열로 변환하기

문자열 변환하기

In [33]:
```
df["연도(str)"] = df["연도"].astype('str')
df["월(str)"] = df["월"].astype('str')

df["날짜"] = df["연도(str)"] + "-" + df["월(str)"]

df
```

Out[33]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜
0	대한민국	수도권	서울	전체	2015	10	5841.0	1770.000000	비쌈	2015	10	2015-10
1	대한민국	수도권	서울	전용면적 60㎡이하	2015	10	5652.0	1712.727273	비쌈	2015	10	2015-10
2	대한민국	수도권	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.424242	비쌈	2015	10	2015-10

문자열을 이어서 새로운 컬럼의 값으로 넣기

Datetime으로 변환하기

```
In [34]: df.dtypes
```

```
Out[34]: 국가           object
         광역권          object
         지역명          object
         규모구분         object
         연도           int64
         월            int64
         분양가          float64
         평당분양가        float64
         평가           object
         연도(str)      object
         월(str)       object
         날짜           object
         dtype: object
```

```
In [35]: df['날짜'] = pd.to_datetime(df['날짜'])
```

```
In [37]: df.dtypes
```

```
Out[37]: 국가                    object
         광역권                   object
         지역명                   object
         규모구분                  object
         연도                    int64
         월                     int64
         분양가                   float64
         평당분양가                 float64
         평가                    object
         연도(str)               object
         월(str)                object
         날짜           datetime64[ns]
         dtype: object
```

datatime 형태로 변환하기

연 또는 월 데이터만 가져오기

```
In [40]: df['날짜'].dt.year

Out[40]: 0       2015
         1       2015
         2       2015
         3       2015
         4       2015
         5       2015
         6       2015
```

연 데이터만 가져오기

```
In [41]: df['날짜'].dt.month

Out[41]: 0       10
         1       10
         2       10
         3       10
         4       10
         5       10
```

월 데이터만 가져오기

비어 있는 값 확인하기

비어 있는 값 확인하기

In [48]:
```
df[df["지역명"].isnull()]
```

Out[48]:

국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜

In [49]:
```
df[df["광역권"].isnull()]
```

Out[49]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜
15	대한민국	NaN	부산	전체	2015	10	3112.0	943.030303	보통	2015	10	2015-10-01
16	대한민국	NaN	부산	전용면적 60㎡이하	2015	10	2950.0	893.939394	보통	2015	10	2015-10-01
17	대한민국	NaN	부산	전용면적 60㎡초과 85㎡이하	2015	10	2999.0	908.787879	보통	2015	10	2015-10-01
18	대한민국	NaN	부산	전용면적 85㎡초과 102㎡이하	2015	10	2957.0	896.060606	보통	2015	10	2015-10-01
19	대한민국	NaN	부산	전용면적 102㎡초과	2015	10	3500.0	1060.606061	보통	2015	10	2015-10-01

비어 있는 값에 값 채우기

비어 있는 값에 값 채우기

In [50]:
```
df.loc[df["광역권"].isnull(), "광역권"] = "비수도권"
```

In [51]:
```
df[df["광역권"].isnull()]
```

Out[51]:

국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜

In [52]:
```
df.tail()
```

Out[52]:

	국가	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜
3650	대한민국	비수도권	제주	전체	2019	4	3441.0	1042.727273	보통	2019	4	2019-04-01
3651	대한민국	비수도권	제주	전용면적 60㎡이하	2019	4	3804.0	1152.727273	보통	2019	4	2019-04-01
3652	대한민국	비수도권	제주	전용면적 60㎡초과 85㎡이하	2019	4	3475.0	1053.030303	보통	2019	4	2019-04-01
3653	대한민국	비수도권	제주	전용면적 85㎡초과 102㎡이하	2019	4	3226.0	977.575758	보통	2019	4	2019-04-01
3654	대한민국	비수도권	제주	전용면적 102㎡초과	2019	4	3269.0	990.606061	보통	2019	4	2019-04-01

새로운 컬럼 만들어서 비어 있는 값에 값 채우기 (1)

새로운 컬럼 만들어서 비어 있는 값에 값 채우기 (1)

In [60]:
```
df[df["분양가"].isnull()]
```

Out[60]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜
28	비수도권	광주	전용면적 85㎡초과 102㎡이하	2015	10	NaN	NaN	False	2015	10	2015-10-01
29	비수도권	광주	전용면적 102㎡초과	2015	10	NaN	NaN	False	2015	10	2015-10-01
34	비수도권	대전	전용면적 102㎡초과	2015	10	NaN	NaN	False	2015	10	2015-10-01

In [61]:
```
df.loc[df["분양가"].isnull(), "분양가(not null)"] = df["분양가"].mean()
```

In [62]:
```
df[df["분양가"].isnull()]
```

Out[62]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜	분양가(not null)
28	비수도권	광주	전용면적 85㎡초과 102㎡이하	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206
29	비수도권	광주	전용면적 102㎡초과	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206
34	비수도권	대전	전용면적 102㎡초과	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206

새로운 컬럼 만들어서 비어 있는 값에 값 채우기 (2)

새로운 컬럼 만들어서 비어 있는 값에 값 채우기 (2)

In [64]:
```
df['평당분양가(not null)'] = df['평당분양가'].fillna(0)
```

In [65]:
```
df[df["평당분양가"].isnull()]
```

Out[65]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	연도(str)	월(str)	날짜	분양가(not null)	평당분양가(not null)
28	비수도권	광주	전용면적 85㎡초과 102㎡이하	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206	0.0
29	비수도권	광주	전용면적 102㎡초과	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206	0.0
34	비수도권	대전	전용면적 102㎡초과	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206	0.0
81	비수도권	제주	전용면적 60㎡이하	2015	10	NaN	NaN	False	2015	10	2015-10-01	3140.904206	0.0

특정 조건에 해당하는 데이터 검색하기 (1)

등호 활용하여 특정 조건에 해당하는 데이터 검색하기

```
In [73]: df[df["지역명"] == "부산"]
```

Out[73]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	날짜
15	비수도권	부산	전체	2015	10	3112.0	943.0	보통	2015-10-01
16	비수도권	부산	전용면적 60㎡이하	2015	10	2950.0	893.0	보통	2015-10-01
17	비수도권	부산	전용면적 60㎡초과 85㎡이하	2015	10	2999.0	908.0	보통	2015-10-01

부등호 활용하여 특정 조건에 해당하는 데이터 검색하기

```
In [76]: df[df["평당분양가"] > 2000]
```

Out[76]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	날짜
88	수도권	서울	전용면적 85㎡초과 102㎡이하	2015	11	7092.0	2149.0	비쌈	2015-11-01
428	수도권	서울	전용면적 85㎡초과 102㎡이하	2016	3	7179.0	2175.0	비쌈	2016-03-01
511	수도권	서울	전용면적 60㎡이하	2016	4	6618.0	2005.0	비쌈	2016-04-01

특정 조건에 해당하는 데이터 검색하기 (2)

여러 특정 값에 해당하는 데이터 검색하기

```
In [77]: df[df["규모구분"].isin(["전용면적 60㎡이하", "전용면적 60㎡초과 85㎡이하"])]
```

Out[77]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	날짜
1	수도권	서울	전용면적 60㎡이하	2015	10	5652.0	1712.0	비쌈	2015-10-01
2	수도권	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.0	비쌈	2015-10-01
6	수도권	인천	전용면적 60㎡이하	2015	10	3488.0	1056.0	보통	2015-10-01

특정 문자가 들어가 있는 데이터 검색하기

```
In [98]: df[df["규모구분"].str.contains("102㎡")]
```

Out[98]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	날짜
3	수도권	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721.0	1733.0	비쌈	2015-10-01
4	수도권	서울	전용면적 102㎡초과	2015	10	5879.0	1781.0	비쌈	2015-10-01
8	수도권	인천	전용면적 85㎡초과 102㎡이하	2015	10	3545.0	1074.0	보통	2015-10-01

여러 조건을 모두 만족하는 데이터 검색하기

여러 조건을 모두 만족하는 데이터 검색하기

```
In [93]: df[df["평당분양가"] > 1300 & d`f["지역명"] == "부산"]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pandas₩core₩ops.py in na_op(x, y)
   1303         try:
-> 1304             result = op(x, y)
   1305         except TypeError:
```

※ 각각의 조건을 괄호( ) 로 묶어주어야 한다.

```
In [92]: df[(df["평당분양가"] > 1300) & (df["지역명"] == "부산")]
```

Out[92]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	날짜
2993	비수도권	부산	전용면적 85㎡초과 102㎡이하	2018	9	4456.0	1350.0	보통	2018-09-01
3078	비수도권	부산	전용면적 85㎡초과 102㎡이하	2018	10	4349.0	1317.0	보통	2018-10-01
3079	비수도권	부산	전용면적 102㎡초과	2018	10	4434.0	1343.0	보통	2018-10-01
3504	비수도권	부산	전용면적 102㎡초과	2019	3	4344.0	1316.0	보통	2019-03-01
3589	비수도권	부산	전용면적 102㎡초과	2019	4	4344.0	1316.0	보통	2019-04-01

여러 조건 중 하나만 성립해도 데이터 검색하기

여러 조건 중 하나만 성립해도 데이터 검색하기

```
In [99]: df[(df["평당분양가"] > 1300) | (df["지역명"] == "부산")]
```

Out[99]:

	광역권	지역명	규모구분	연도	월	분양가	평당분양가	평가	날짜
0	수도권	서울	전체	2015	10	5841.0	1770.0	비쌈	2015-10-01
1	수도권	서울	전용면적 60㎡이하	2015	10	5652.0	1712.0	비쌈	2015-10-01
2	수도권	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882.0	1782.0	비쌈	2015-10-01
3	수도권	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721.0	1733.0	비쌈	2015-10-01
4	수도권	서울	전용면적 102㎡초과	2015	10	5879.0	1781.0	비쌈	2015-10-01
15	비수도권	부산	전체	2015	10	3112.0	943.0	보통	2015-10-01
16	비수도권	부산	전용면적 60㎡이하	2015	10	2950.0	893.0	보통	2015-10-01

특정 조건으로 컬럼 검색하기

In [96]:
```
df.loc[df["지역명"] == "서울", ["평당분양가"]]
```

Out[96]:

	평당분양가
0	1770.0
1	1712.0
2	1782.0
3	1733.0
4	1781.0

특정 조건으로 컬럼 검색하기

In [97]:
```
df.loc[df["지역명"] == "서울", ["규모구분", "날짜","평당분양가"]]
```

Out[97]:

	규모구분	날짜	평당분양가
0	전체	2015-10-01	1770.0
1	전용면적 60㎡이하	2015-10-01	1712.0
2	전용면적 60㎡초과 85㎡이하	2015-10-01	1782.0
3	전용면적 85㎡초과 102㎡이하	2015-10-01	1733.0
4	전용면적 102㎡초과	2015-10-01	1781.0

특정 조건으로 여러 컬럼 검색하기

피벗테이블로 보기

```
In [101]: pd.pivot_table(df,index="지역명", values="평당분양가")
```

Out[101]:

지역명	평당분양가
강원	701.033816
경기	1179.009302
경남	825.325581

지역별 평당분양가 피벗테이블로 보기

```
In [104]: pd.pivot_table(df,index=["지역명","규모구분"], values="평당분양가")
```

Out[104]:

지역명	규모구분	평당분양가
강원	전용면적 102㎡초과	733.953488
	전용면적 60㎡이하	679.395349
	전용면적 60㎡초과 85㎡이하	675.418605
	전용면적 85㎡초과 102㎡이하	752.342857
	전체	673.604651
경기	전용면적 102㎡초과	1299.116279
	전용면적 60㎡이하	1175.162791

지역별/규모별 평당분양가
피벗테이블로 보기

피벗 테이블 : 여러 데이터 중에서 자신이 원하는 데이터만을 가지고
원하는 행과 열에 데이터를 배치하여 새로운 보고서를 만드는 기능

파일로 저장하기

csv 파일로 저장하기

```
In [163]: df.to_csv("C:/Users/fastj/Downloads/house_price_upgrade.csv")
```