

# 데이터분석입문

## Lecture 14. pandas로 데이터 분석 시작하기

동양미래대학교  
인공지능소프트웨어학과  
강 환수

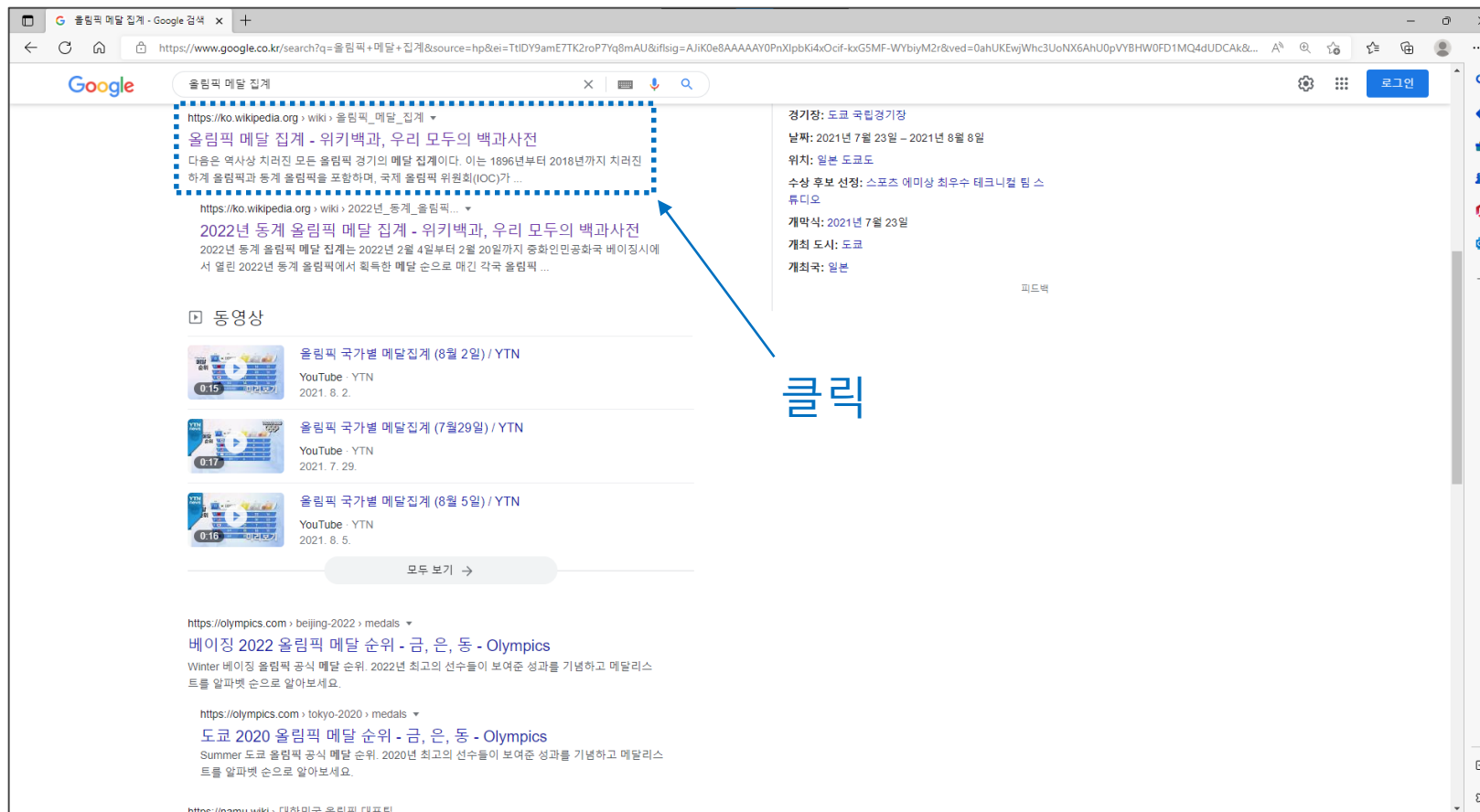
- ❖ 01. 위키피디아 데이터 엑셀로 저장하기
- ❖ 02. pandas로 인구 구조 분석하기

# 01. 위키피디아 데이터 엑셀로 저장하기

02. pandas로 인구 구조 분석하기

## ❖ 올림픽 메달 기록 데이터 (1/2)

- pandas 라이브러리 실습을 위해 올림픽 메달 기록 데이터를 활용하겠습니다.
- Google에서 "올림픽 메달 집계"라고 검색 → "올림픽 메달 집계 - 위키백과, 우리 모두의 백과사전" 검색 결과 클릭



## ❖ 올림픽 메달 기록 데이터 (2/2)

- 위키백과(Wikipedia)에서 제공하는 하계 및 동계 올림픽 메달 획득 결과 표

URL 주소를 이용하여  
아래 표 정보를 읽어오겠습니다.

국가 (IOC 코드)	하계 참가 횟수	1	2	3	계	동계 참가 횟수	1	2	3	계	전제 참가 횟수	1	2	3	총합
아프가니스탄 (AFG)	14	0	0	2	2	0	0	0	0	0	14	0	0	2	2
알제리 (ALG)	13	5	4	8	17	3	0	0	0	0	16	5	4	8	17
아르헨티나 (ARG)	24	21	25	28	74	19	0	0	0	0	43	21	25	28	74
아르메니아 (ARM)	6	2	6	6	14	7	0	0	0	0	13	2	6	6	14
오스트레일리아 (AUS) [ANZ] [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4	5	12
오스트리아 (AUT)	26	147	163	187	497	19	5	5	5	15	45	152	168	192	512
아제르바이잔 (AZE)	6	7	11	24	42	6	0	0	0	0	12	7	11	24	42

## ❖ pandas 라이브러리를 이용하여 URL로부터 데이터 읽어 오기

```
import pandas as pd
```

```
df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')  
print(df)
```

```
[
```

	국가 (IOC 코드)	하계	참가 횟수	Unnamed: 2	Unnamed: 3	Unnamed: 4	계	₩
0	아프가니스탄 (AFG)		14	0	0	2	2	
1	알제리 (ALG)	13		5	4	8	17	
2	아르헨티나 (ARG)	24		21	25	28	74	
3	아르메니아 (ARM)	6		2	6	6	14	
4	오스트랄라시아 (ANZ) [ANZ]		2	3	4	5	12	
...	...	...	...	...	...	...	...	...
148	독립 (IOA) [IOA]		3	1	0	1	2	
149	독립 참가 (IOP) [IOP]		1	0	1	2	3	
150	러시아 출신 올림픽 선수 (OAR)			0	0	0	0	0
151	혼성 (ZZX) [ZZX]		3	8	5	4	17	
152	총합	28		5116	5082	5490	15688	

대괄호 [ ]가 있습니다!  
그 뜻은 자료형이 리스트라는 거겠죠?  
df의 구성을 살펴보겠습니다.

	동계	참가	횟수	Unnamed: 7	Unnamed: 8	Unnamed: 9	계	전체	참가	횟수	₩
0		0		0	0	0	0	14			
1		3		0	0	0	0	16			
2		19		0	0	0	0	43			
3		7		0	0	0	0	13			
4		0		0	0	0	0	2			

## ❖ 데이터 프레임 df[0] 살펴보기

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df[0]
```

	국가 (IOC 코드)	하계 참가 횟수	Unnamed: 2	Unnamed: 3	Unnamed: 4	계	동계 참가 횟수	Unnamed: 7	Unnamed: 8	Unnamed: 9	계.1	전계 참가 횟수	Unnamed: 12	Unnamed: 13	Unnamed: 14	총합
0	아프가니스탄 (AFG)	14	0	0	2	2	0	0	0	0	0	14	0	0	2	2
1	알제리 (ALG)	13	5	4	8	17	3	0	0	0	0	16	5	4	8	17
2	아르헨티나 (ARG)	24	21	25	28	74	19	0	0	0	0	43	21	25	28	74
3	아르메니아 (ARM)	6	2	6	6	14	7	0	0	0	0	13	2	6	6	14
4	오스트랄라시아 (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4	5	12
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

df[0]에 우리가 원하는 데이터가 담겨있네요!

## ❖ 인덱스(Index)와 열 이름(Column Name) 확인

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df[0]
```

	국가 (IOC 코드)	하계 참가 횟수	Unnamed: 2	Unnamed: 3	Unnamed: 4	계	동계 참가 횟수	Unnamed: 7	Unnamed: 8	Unnamed: 9	계.1	전계 참가 횟수	Unnamed: 12	Unnamed: 13	Unnamed: 14	총합
0	아프가니스탄 (AFG)	14	0	0	2	2	0	0	0	0	0	14	0	0	2	2
1	알제리 (ALG)	13	5	4	8	17	3	0	0	0	0	16	5	4	8	17
2	아르헨티나 (ARG)	24	21	25	28	74	19	0	0	0	0	43	21	25	28	74
3	아르메니아 (ARM)	6	2	6	6	14	7	0	0	0	0	13	2	6	6	14
4	오스트랄라시아 (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4	5	12
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

열 이름

인덱스



## ❖ 인덱스를 '국가 (IOC 코드)'로 변경하기

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df2 = df[0].set_index('국가 (IOC 코드)')
df2
```

	하계 참가 횟수	Unnamed: 2	Unnamed: 3	Unnamed: 4	계	동계 참가 횟수	Unnamed: 7	Unnamed: 8	Unnamed: 9	계.1	전체 참가 횟수	Unnamed: 12	Unnamed: 13	Unnamed: 14	총합
국가 (IOC 코드)															
아프가니스탄 (AFG)	14	0	0	2	2	0	0	0	0	0	14	0	0	2	2
알제리 (ALG)	13	5	4	8	17	3	0	0	0	0	16	5	4	8	17
아르헨티나 (ARG)	24	21	25	28	74	19	0	0	0	0	43	21	25	28	74
아르메니아 (ARM)	6	2	6	6	14	7	0	0	0	0	13	2	6	6	14
오스트랄라시아 (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4	5	12
...															

인덱스가 "국가 (IOC 코드)"로  
변경되었습니다.

## ❖ 하계 정보만 추출하기

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df2 = df[0].set_index('국가 (IOC 코드)')

summer = df2.iloc[:, :5]
# iloc은 integer location의 약어로, 데이터 프레임의 행이나 열의 순서를 나타내는 정수로 특정 값을 추출합니다.
summer
```

국가 (IOC 코드)	하계 참가 횟수	Unnamed: 2	Unnamed: 3	Unnamed: 4	계
아프가니스탄 (AFG)	14	0	0	2	2
알제리 (ALG)	13	5	4	8	17
아르헨티나 (ARG)	24	21	25	28	74
아르메니아 (ARM)	6	2	6	6	14
오스트랄라시아 (ANZ) [ANZ]	2	3	4	5	12
...	...	...	...	...	...
독립 (IOA) [IOA]	3	1	0	1	2
독립 참가 (IOP) [IOP]	1	0	1	2	3
러시아 출신 올림픽 선수 (OAR)	0	0	0	0	0
혼성 (ZZX) [ZZX]	3	8	5	4	17
총합	28	5116	5082	5490	15688

153 rows × 5 columns

iloc( ) 함수를 활용하여  
하계(Summer) 정보만 추출하였습니다.

## ❖ 컬럼 이름 설정하기

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df2 = df[0].set_index('국가 (IOC 코드)')

summer = df2.iloc[:, :5]
# iloc은 integer location의 약어로, 데이터 프레임의 행이나 열의 순서를 나타내는 정수로 특정 값을 추출합니다.
summer.columns = ['경수기', '금', '은', '동', '합계']
summer
```

	경수기	금	은	동	합계
국가 (IOC 코드)					
아프가니스탄 (AFG)	14	0	0	2	2
알제리 (ALG)	13	5	4	8	17
아르헨티나 (ARG)	24	21	25	28	74
아르메니아 (ARM)	6	2	6	6	14
오스트랄라시아 (ANZ) [ANZ]	2	3	4	5	12
...	...	...	...	...	...
독립 (IOA) [IOA]	3	1	0	1	2
독립 참가 (IOP) [IOP]	1	0	1	2	3
러시아 출신 올림픽 선수 (OAR)	0	0	0	0	0
혼성 (ZZX) [ZZX]	3	8	5	4	17
총합	28	5116	5082	5490	15688

153 rows × 5 columns

데이터 프레임의 columns에  
컬럼 이름을 설정하였습니다.

## ❖ 내림차순으로 정렬하기

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df2 = df[0].set_index('국가 (IOC 코드)')

summer = df2.iloc[:, :5]
# iloc은 integer location의 약어로, 데이터 프레임의 행이나 열의 순서를 나타내는 정수로 특정 값을 추출합니다.
summer.columns = ['경수기', '금', '은', '동', '합계']
summer = summer.sort_values('금', ascending=False)
summer
```

	경수기	금	은	동	합계
국가 (IOC 코드)					
총합	28	5116	5082	5490	15688
미국 (USA) [P] [Q] [R] [Z] [F]	27	1022	795	706	2523
소련 (URS) [URS]	9	395	319	296	1010
영국 (GBR) [GBR] [Z]	28	263	295	293	851
중화인민공화국 (CHN) [CHN]	10	224	167	155	546
...	...	...	...	...	...
레바논 (LIB)	17	0	2	2	4
세르비아 몬테네그로 (SCG) [SCG]	1	0	2	0	2
지부티 (DJI) [B]	8	0	0	1	1
키프로스 (CYP)	10	0	1	0	1
아프가니스탄 (AFG)	14	0	0	2	2

153 rows × 5 columns

sort\_values( ) 함수를 이용하면  
원하는 열을 기준으로  
데이터 순서를 정렬할 수 있습니다.

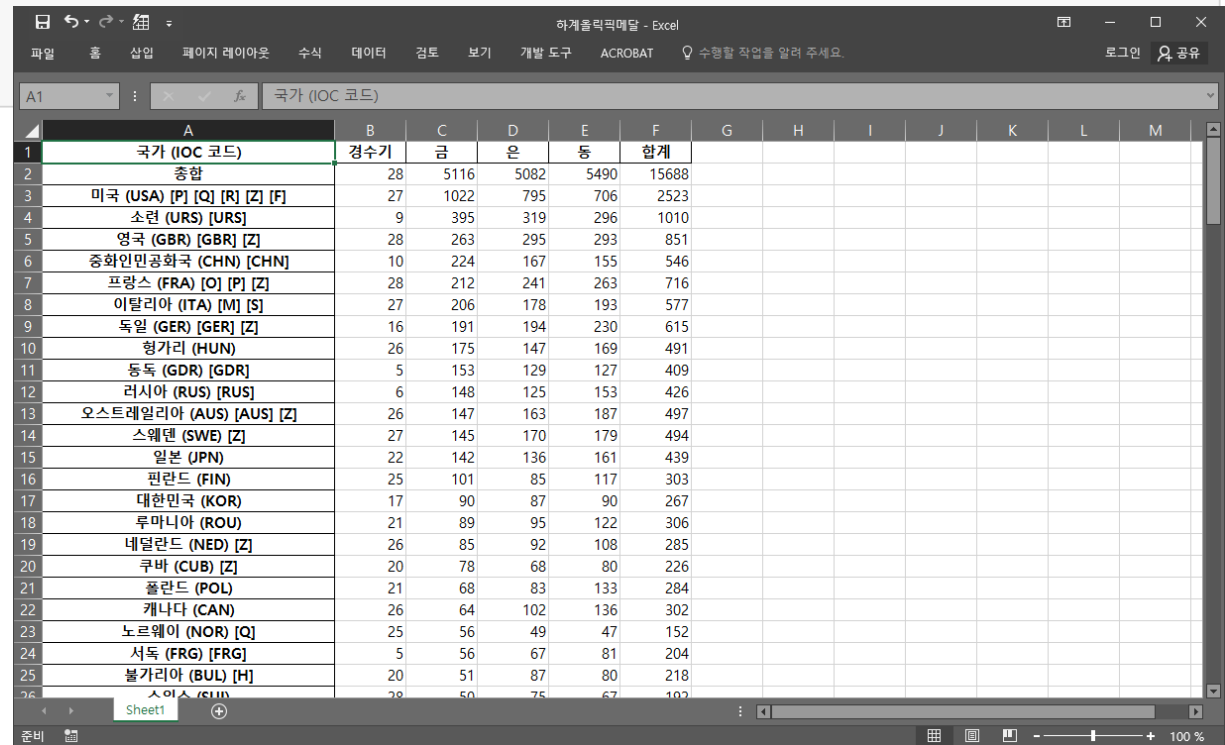
## ❖ 엑셀 파일로 저장하기

```
import pandas as pd

df = pd.read_html('https://ko.wikipedia.org/wiki/%EC%98%AC%EB%A6%BC%ED%94%BD_%EB%A9%94%EB%8B%AC_%EC%A7%91%EA%B3%84')
df2 = df[0].set_index('국가 (IOC 코드)')

summer = df2.iloc[:, :5]
# iloc은 integer location의 약어로, 데이터 프레임의 행이나 열의 순서를 나타내는 정수로 특정 값을 추출합니다.
summer.columns = ['경수기', '금', '은', '동', '합계']
summer = summer.sort_values('금', ascending=False)
summer.to_excel('하계올림픽메달.xlsx')
```

to\_excel( ) 함수를 이용하여 작업했던  
"summer" 데이터 프레임을  
엑셀 파일로 저장합니다.



A	B	C	D	E	F	G	H	I	J	K	L	M
국가 (IOC 코드)	경수기	금	은	동	합계							
총합	28	5116	5082	5490	15688							
미국 (USA) [P] [Q] [R] [Z] [F]	27	1022	795	706	2523							
소련 (URS) [URS]	9	395	319	296	1010							
영국 (GBR) [GBR] [Z]	28	263	295	293	851							
중화인민공화국 (CHN) [CHN]	10	224	167	155	546							
프랑스 (FRA) [O] [P] [Z]	28	212	241	263	716							
이탈리아 (ITA) [M] [S]	27	206	178	193	577							
독일 (GER) [GER] [Z]	16	191	194	230	615							
헝가리 (HUN)	26	175	147	169	491							
동독 (GDR) [GDR]	5	153	129	127	409							
러시아 (RUS) [RUS]	6	148	125	153	426							
오스트레일리아 (AUS) [AUS] [Z]	26	147	163	187	497							
스웨덴 (SWE) [Z]	27	145	170	179	494							
일본 (JPN)	22	142	136	161	439							
핀란드 (FIN)	25	101	85	117	303							
대한민국 (KOR)	17	90	87	90	267							
루마니아 (ROU)	21	89	95	122	306							
네덜란드 (NED) [Z]	26	85	92	108	285							
쿠바 (CUB) [Z]	20	78	68	80	226							
폴란드 (POL)	21	68	83	133	284							
캐나다 (CAN)	26	64	102	136	302							
노르웨이 (NOR) [Q]	25	56	49	47	152							
서독 (FRG) [FRG]	5	56	67	81	204							
불가리아 (BUL) [H]	20	51	87	80	218							
소련 (URS)	28	50	75	67	192							

## 02. pandas로 인구 구조 분석하기

01. 위키피디아 데이터 엑셀로 저장하기

### ❖ 알고리즘(Algorithm) 설계하기

- Step 1) 데이터를 읽어온다.
  - ◆ ① 전체 데이터를 총 인구수로 나누어 비율로 변환한다.
  - ◆ ② 총 인구수와 연령 구간 인구수를 삭제한다.
- Step 2) 궁금한 지역의 이름을 입력 받는다.
- Step 3) 궁금한 지역의 인구 구조를 저장한다.
- Step 4) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역을 찾는다.
  - ◆ ① 전국의 모든 지역 중 한 곳(B)를 선택한다.
  - ◆ ② 궁금한 지역의 이름을 입력 받는다.
  - ◆ ③ ②를 100세 이상 인구수에 해당하는 값까지 반복한 후 차이의 제곱을 모두 더한다.
  - ◆ ④ 전국의 모든 지역에 대해 반복하며 그 차이가 가장 작은 지역을 찾는다.
- Step 5) 가장 비슷한 곳의 인구 구조와 궁금한 지역의 인구 구조를 시각화한다.

## 02. pandas로 인구 구조 분석하기

### ❖ Step 1) 데이터를 읽어 오기 (1/6)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)
df.head()
```

	2021년11월_계_총인구수	2021년11월_계_연령구간인구수	2021년11월_계_0세	2021년11월_계_1세	2021년11월_계_2세	2021년11월_계_3세	2021년11월_계_4세	2021년11월_계_5세	2021년11월_계_6세	2021년11월_계_7세	...	2021년11월_계_91세	2021년11월_계_92세	2021년11월_계_93세	2021년11월_계_94세	2021년11월_계_95세	2021년11월_계_96세	2021년11월_계_97세	2021년11월_계_98세
행정구역																			
서울특별시 (1100000000)	9,520,880	9,520,880	43,492	45,054	49,318	51,995	56,210	63,757	67,763	66,950	...	8,147	6,779	5,328	4,048	2,587	1,862	1,368	1,279
서울특별시 종로구 (1111000000)	145,073	145,073	486	475	557	576	664	773	848	859	...	183	139	114	104	71	45	28	25
서울특별시 종로구 청운효자동 (1111051500)	12,006	12,006	43	48	58	54	80	74	96	93	...	10	10	10	7	7	3		
서울특별시 종로구 사직동 (1111053000)	9,367	9,367	38	27	40	46	55	67	74	71	...	15	19	6	7	8	9	2	3
서울특별시 종로구 삼청동 (1111054000)	2,467	2,467	7	5	3	10	14	6	17	8	...	7	2	4	3	1	0	2	0

5 rows × 103 columns

데이터를 확인해 보면  
숫자 사이에 쉼표(,)가 있습니다.  
자료형을 확인해 보겠습니다.



### ❖ Step 1) 데이터를 읽어 오기 (2/6)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)
df.dtypes
```

```
2021년11월_계_총인구수      object
2021년11월_계_연령구간인구수  object
2021년11월_계_0세          object
2021년11월_계_1세          object
2021년11월_계_2세          object
...
2021년11월_계_96세          object
2021년11월_계_97세          object
2021년11월_계_98세          object
2021년11월_계_99세          object
2021년11월_계_100세 이상    object
Length: 103, dtype: object
```

정수형 자료형이 아니라 객체(object) 입니다.  
정수형 자료형으로 변환하기 전에  
숫자 사이에 있는 쉼표(,)부터 제거하겠습니다.

## 02. pandas로 인구 구조 분석하기

### ❖ Step 1) 데이터를 읽어 오기 (3/6)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(' ', '', regex=True)
df.head()
```

replace( ) 함수를 이용하면 전체 데이터 프레임에서  
심표(,)를 한 번에 "으로 바꿀 수가 있습니다.

	2021년 11월_계 _총인구 수	2021년 11월_계 _연령구 간인구 수	2021 년11 월_계 _0세	2021 년11 월_계 _1세	2021 년11 월_계 _2세	2021 년11 월_계 _3세	2021 년11 월_계 _4세	2021 년11 월_계 _5세	2021 년11 월_계 _6세	2021 년11 월_계 _7세	...	2021 년11 월_계 _91 세	2021 년11 월_계 _92 세	2021 년11 월_계 _93 세	2021 년11 월_계 _94 세	2021 년11 월_계 _95 세	2021 년11 월_계 _96 세	2021 년11 월_계 _97 세	2021 년11 월_계 _98 세	2021 년11 월_계 _99 세	20 년 계 _1 이
행정구역																					
서울특별시 (1100000000)	9520880	9520880	43492	45054	49318	51995	56210	63757	67763	66950	...	8147	6779	5328	4048	2587	1862	1368	1274	816	20
서울특별시 종로구 (1111000000)	145073	145073	486	475	557	576	664	773	848	859	...	183	139								
서울특별시 종로구 청운 효자동 (1111051500)	12006	12006	43	48	58	54	80	74	96	93	...	10	10								
서울특별시 종로구 사직 동 (1111053000)	9367	9367	38	27	40	46	55	67	74	71	...	15	19	6	7	8	9	2	3	1	
서울특별시 종로구 삼청 동 (1111054000)	2467	2467	7	5	3	10	14	6	17	8	...	7	2	4	3	1	0	2	0	2	

regex 옵션은 regular expression의 약자로,  
정규 표현식으로 문자열이 완전히 일치하지 않더라도  
문자열의 일부분만 치환하고 싶을 경우 True로 설정해줍니다.

이제 정수형 자료형으로  
변환해 보겠습니다.

5 rows × 103 columns

### ❖ Step 1) 데이터를 읽어 오기 (4/6)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df.dtypes
```

```
2021년11월_계_총인구수      int64
2021년11월_계_연령구간인구수  int64
2021년11월_계_0세          int64
2021년11월_계_1세          int64
2021년11월_계_2세          int64
...
2021년11월_계_96세          int64
2021년11월_계_97세          int64
2021년11월_계_98세          int64
2021년11월_계_99세          int64
2021년11월_계_100세 이상    int64
Length: 103, dtype: object
```

객체(object)에서 정수형 자료형  
"int64"로 변경되었습니다.

## ❖ Step 1) 데이터를 읽어 오기 (5/6)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df = df.div(df['2021년11월_계_총인구수'], axis='index')
df.head(3)
```

㉑ 전체 데이터를 총 인구수로 나누어 비율로 변환한다.

	2021년11월_계_총인구수	2021년11월_계_연령구간인구수	2021년11월_계_0세	2021년11월_계_1세	2021년11월_계_2세	2021년11월_계_3세	2021년11월_계_4세	2021년11월_계_5세	2021년11월_계_6세	2021년11월_계_7세	...	2021년11월_계_91세	2021년11월_계_92세	2021년11월_계_93세	2021년11월_계_94세	2021년11월_계_95세
행정구역																
서울특별시 (1100000000)	1.0	1.0	0.004568	0.004732	0.005180	0.005461	0.005904	0.006697	0.007117	0.007032	...	0.000856	0.000712	0.000560	0.000425	0.000312
서울특별시 종로구 (1111000000)	1.0	1.0	0.003350	0.003274	0.003839	0.003970	0.004577	0.005328	0.005845	0.005921	...	0.001261	0.000958	0.000786	0.000717	0.000612
서울특별시 종로구 청운효자동 (1111051500)	1.0	1.0	0.003582	0.003998	0.004831	0.004498	0.006663	0.006164	0.007996	0.007746	...	0.000833	0.000833	0.000833	0.000583	0.000412

3 rows × 103 columns

## ❖ Step 1) 데이터를 읽어 오기 (6/6)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df = df.div(df['2021년11월_계_총인구수'], axis='index')
del df['2021년11월_계_총인구수'], df['2021년11월_계_연령구간인구수']

df.head(3)
```

② 총 인구수와 연령 구간 인구수를 삭제한다.

	2021년 11월_계 _0세	2021년 11월_계 _1세	2021년 11월_계 _2세	2021년 11월_계 _3세	2021년 11월_계 _4세	2021년 11월_계 _5세	2021년 11월_계 _6세	2021년 11월_계 _7세	2021년 11월_계 _8세	2021년 11월_계 _9세	...	2021년 11월_계 _91세	2021년 11월_계 _92세	2021년 11월_계 _93세	2021년 11월_계 _94세
행정구역															
서울특별시 (1100000000)	0.004568	0.004732	0.005180	0.005461	0.005904	0.006697	0.007117	0.007032	0.007115	0.007801	...	0.000856	0.000712	0.000560	0.0004...
서울특별시 종로구 (1111000000)	0.003350	0.003274	0.003839	0.003970	0.004577	0.005328	0.005845	0.005921	0.005735	0.006679	...	0.001261	0.000958	0.000786	0.0007...
서울특별시 종로구 청운 효자동 (1111051500)	0.003582	0.003998	0.004831	0.004498	0.006663	0.006164	0.007996	0.007746	0.006913	0.009995	...	0.000833	0.000833	0.000833	0.0005...

3 rows × 101 columns

## ❖ Steps 2~3) 궁금한 지역의 이름을 입력 받고 해당 지역의 인구 구조를 저장하기 (1/2)

```
import pandas as pd

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df = df.div(df['2021년11월_계_총인구수'], axis='index')
del df['2021년11월_계_총인구수'], df['2021년11월_계_연령구간인구수']

name = input('원하는 지역의 이름을 입력하세요: ')

a = df.index.str.contains(name)

df2 = df[a]
df2
```

df.index.str.contains( ) 함수는  
데이터 프레임의 인덱스 문자열로부터,  
원하는 문자열이 포함된 행을 찾아냅니다.

원하는 지역의 이름을 입력하세요: 신도림

	2021년 11월_계 _0세	2021년 11월_계 _1세	2021년 11월_계 _2세	2021년 11월_계 _3세	2021년 11월_계 _4세	2021년 11월_계 _5세	2021년 11월_계 _6세	2021년 11월_계 _7세	2021년 11월_계 _8세	2021년 11월_계 _9세	...	2021년 11월_계 _91세	2021년 11월_계 _92세	2021년 11월_계 _93세	2021년 11월_계 _94세
행정구역															
서울특별시 구로구 신도 림동 (1153051000)	0.007619	0.007508	0.00887	0.008453	0.008926	0.010483	0.011845	0.011206	0.010399	0.010872	...	0.000389	0.000445	0.000361	0.00030

1 rows × 101 columns

### ❖ Steps 2~3) 궁금한 지역의 이름을 입력 받고 해당 지역의 인구 구조를 저장하기 (2/2)

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df = df.div(df['2021년11월_계_총인구수'], axis='index')
del df['2021년11월_계_총인구수'], df['2021년11월_계_연령구간인구수']

name = input('원하는 지역의 이름을 입력하세요: ')

a = df.index.str.contains(name)

df2 = df[a]

plt.rc('font', family='Malgun Gothic')
df2.T.plot()
plt.show()
```

원하는 지역의 이름을 입력하세요: 신도림



pandas의 Series나 DataFrame은  
plot() 메서드를 내장하고 있습니다.

## 02. pandas로 인구 구조 분석하기

### ❖ Steps 4~5) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역의 인구 구조를 시각화하기 (1/5)

```
# A의 인구 비율에서 B의 인구 비율을 뺀다.  
x = df.sub(df2.iloc[0], axis='columns') # axis=1 이라고 적어도 됩니다.  
x.head(3)
```

	2021년11 월_계_0 세	2021년11 월_계_1 세	2021년11 월_계_2 세	2021년11 월_계_3 세	2021년11 월_계_4 세	2021년11 월_계_5 세	2021년11 월_계_6 세	2021년11 월_계_7 세	2021년11 월_계_8 세	2021년11 월_계_9 세	...	2021년 11월_계 _91세	2021년 11월_계 _92세	2021년 11월_계 _93세
행정구역														
서울특별시 (1100000000)	-0.003051	-0.002775	-0.003690	-0.002992	-0.003022	-0.003786	-0.004728	-0.004174	-0.003284	-0.003071	...	0.000466	0.000267	0.000198
서울특별시 종로구 (1111000000)	-0.004269	-0.004233	-0.005031	-0.004482	-0.004349	-0.005154	-0.006000	-0.005284	-0.004664	-0.004193	...	0.000872	0.000513	0.000424
서울특별시 종로구 청운 효자동 (1111051500)	-0.004037	-0.003510	-0.004039	-0.003955	-0.002262	-0.004319	-0.003849	-0.003460	-0.003486	-0.000877	...	0.000444	0.000388	0.000471

3 rows × 101 columns





### ❖ Steps 4~5) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역의 인구 구조를 시각화하기 (2/5)

```
## 차이의 제곱 값을 모두 더한다.
```

```
y = np.power(x, 2)  
z = y.sum(axis='columns')  
z
```

행정구역

서울특별시 (1100000000)	0.000614
서울특별시 종로구 (1111000000)	0.001200
서울특별시 종로구 청운효자동(1111051500)	0.000589
서울특별시 종로구 사직동(1111053000)	0.000831
서울특별시 종로구 삼청동(1111054000)	0.001854

...	
제주특별자치도 서귀포시 서홍동(5013058000)	0.000688
제주특별자치도 서귀포시 대륜동(5013059000)	0.000279
제주특별자치도 서귀포시 대천동(5013060000)	0.000263
제주특별자치도 서귀포시 중문동(5013061000)	0.000394
제주특별자치도 서귀포시 예래동(5013062000)	0.001971

Length: 3862, dtype: float64

## 02. pandas로 인구 구조 분석하기

❖ Steps 4~5) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역의 인구 구조를 시각화하기 (3/5)

```
z = z[z[:, ] != 0]
```

age.csv 파일 내 일부 행정구역의 연령별 인구수 정보가 전부 0인 경우가 있습니다.  
이 행정구역들의 경우 z 값이 0으로 계산되기 때문에 제외시켜줍니다.

age.csv

H853																	
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	
826	대구광역시 달성군 가창면(2771031000)	7,784	7,784	24	25	27	29	33	49	44	42	48	68	53	59	36	
827	대구광역시 달성군 하빈면(2771033000)	3,606	3,606	7	12	7	6	3	5	10	10	14	16	10	18	7	
828	대구광역시 달성군 구지면(2771038000)	18,766	18,766	257	288	268	305	302	278	273	270	221	238	199	201	138	
829	인천광역시 (2800000000)	2,946,319	2,946,319	14,611	16,087	18,070	19,316	21,083	24,083	25,953	25,278	25,591	28,122	28,000	27,245	26,069	
830	인천광역시 중구 (2811000000)	143,052	143,052	801	828	1,036	1,081	1,171	1,260	1,391	1,306	1,349	1,460	1,427	1,441	1,260	
831	인천광역시 중구 연안동(2811052000)	6,141	6,141	15	11	18	18	14	28	23	29	33	32	30	29	28	
832	인천광역시 중구 신포동(2811053000)	5,094	5,094	17	14	16	15	20	26	21	31	27	43	42	26	28	
833	인천광역시 중구 신흥동(2811054000)	13,202	13,202	53	66	86	63	79	88	77	88	97	105	93	119	95	
834	인천광역시 중구 도원동(2811056000)	3,919	3,919	6	6	4	11	14	17	12	13	13	21	18	24	23	
835	인천광역시 중구 울목동(2811057000)	3,193	3,193	1	3	8	5	10	5	15	8	11	14	15	23	14	
836	인천광역시 중구 동인천동(2811058500)	5,825	5,825	15	15	16	17	13	17	26	16	21	32	24	26	24	
837	인천광역시 중구 개항동(2811061500)	7,040	7,040	14	10	19	22	29	24	28	30	36	35	32	38	34	
838	인천광역시 중구 영종동(2811062000)	19,016	19,016	79	110	107	132	119	125	118	121	114	164	148	150	148	
839	인천광역시 중구 영종1동(2811062200)	43,784	43,784	464	469	614	653	687	760	858	762	771	786	743	725	564	
840	인천광역시 중구 운서동(2811062800)	31,886	31,886	131	118	144	140	180	158	207	200	216	213	272	271	293	
841	인천광역시 중구 용유동(2811063000)	3,952	3,952	6	6	4	5	6	12	6	8	10	15	10	10	9	
842	인천광역시 중구영종출장소 (2811400000)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
843	인천광역시 중구용유출장소 (2811800000)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
844	인천광역시 동구 (2814000000)	61,716	61,716	200	233	291	310	307	405	444	420	391	506	469	498	459	
845	인천광역시 동구 만석동(2814051000)	6,936	6,936	32	36	35	35	36	45	48	59	35	65	61	61	53	
846	인천광역시 동구 화수1.화평동(2814052500)	5,945	5,945	11	11	12	13	21	22	24	25	44	36	40	35	37	
847	인천광역시 동구 화수2동(2814053000)	7,613	7,613	17	18	12	22	22	34	49	40	35	60	66	62	54	

### ❖ Steps 4~5) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역의 인구 구조를 시각화하기 (4/5)

```
i = z.sort_values().index[:5]  
i
```

```
Index(['경기도 하남시 (4145000000)', '경기도 남양주시 별내동(4136057000)',  
      '서울특별시 영등포구 문래동(1156060500)', '경기도 성남시 중원구 도촌동(4113367000)',  
      '충청남도 아산시 (4420000000)'],  
      dtype='object', name='행정구역')
```

sort\_values( ) 함수를 활용하여 오름차순으로 정렬하고,  
index[:5]를 활용하여 차이가 가장 작은 지역 5곳을 찾습니다.

## 02. pandas로 인구 구조 분석하기



### ❖ Steps 4~5) 궁금한 지역의 인구 구조와 가장 비슷한 인구 구조를 가진 지역의 인구 구조를 시각화하기 (5/5)

```
plt.rc('font', family='Malgun Gothic')
plt.rcParams['axes.unicode_minus'] = False
df.loc[i].T.plot(figsize=(10, 6))
plt.xlabel('나이')
plt.ylabel('인구 비율')
plt.grid(True)
plt.show()
```

loc은 location의 약자로 인덱스를 기준으로  
행 데이터를 읽기 위해서 사용됩니다.

[참고] iloc은 행 번호를 기준으로  
행 데이터를 읽기 위해서 사용됩니다.



## ❖ 전체 코드

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_csv('age.csv', encoding='cp949', index_col=0)

df = df.replace(',', '', regex=True)

df = df.apply(pd.to_numeric)
df = df.div(df['2021년11월_계_총인구수'], axis='index')
del df['2021년11월_계_총인구수'], df['2021년11월_계_연령구간인구수']

name = input('원하는 지역의 이름을 입력하세요: ')

a = df.index.str.contains(name)

df2 = df[a]    # 궁금한 지역 (A)의 인구 구조 정보가 저장되어 있습니다.

# A의 인구 비율에서 B의 인구 비율을 뺀다.
x = df.sub(df2.iloc[0], axis='columns')    # axis=1 이라고 적어도 됩니다.

# 차이의 제곱 값을 모두 더한다.
y = np.power(x, 2)
z = y.sum(axis='columns')
z = z[z[:] != 0]

i = z.sort_values().index[:5]

plt.rc('font', family='Malgun Gothic')
plt.rcParams['axes.unicode_minus'] = False
df.loc[i].T.plot(figsize=(10, 6))
plt.xlabel('나이')
plt.ylabel('인구 비율')
plt.grid(True)
plt.show()
```

- ❖ 01. 위키피디아 데이터 엑셀로 저장하기
- ❖ 02. pandas로 인구 구조 분석하기

# THANK YOU!

## Q & A

- Name: 강환수
- Office: 동양미래대학교 2호관 706호 (02-2610-1941)
- E-mail: [hsknag@dongyang.ac.kr](mailto:hsknag@dongyang.ac.kr)
- Homepage: <https://github.com/ai7dnn/2023-DA>