



단국대학교  
SW 중심대학

# 전공별 시활용

## 데이터 입출력

정 복 문

[Bokmoon.jung@dankook.ac.kr](mailto:Bokmoon.jung@dankook.ac.kr)



# CONTENTS

- 1. 외부파일 읽기
  - 1-1. CSV 파일
  - 1-2. Excel 파일
  - 1-3. JSON 파일
- 2. 웹(web)에서 가져오기
  - 2-1. HTML 웹 페이지에서 표 속성 가져오기
  - 2-2. 웹 스크래핑
- 3. API 활용하여 데이터 수집하기
- 4. 데이터 저장하기
  - 4-1. CSV 파일로 저장
  - 4-2. JSON 파일로 저장
  - 4-3. Excel 파일로 저장
  - 4-4. 여러 개의 데이터프레임을 하나의 Excel 파일로 저장

## 1. 외부파일 읽기

- **판다스 데이터 입출력 도구**

- 판다스는 다양한 형태의 외부 파일을 읽어와서 데이터프레임으로 변환하는 함수를 제공.
- 어떤 파일이든 판다스 객체인 데이터프레임으로 변환되고 나면, 판다스의 모든 함수와 기능을 자유롭게 사용할 수 있음.
- 반대로, 데이터프레임을 다양한 유형의 파일로 저장할 수도 있음.

## 1. 외부파일 읽기

- 판다스 데이터 입출력 도구

File Format	Reader	Writer
CSV	read_csv	to_csv
JSON	read_json	to_json
HTML	read_html	to_html
Local clipboard	read_clipboard	to_clipboard
MS Excel	read_excel	to_excel
HDF5 Format	read_hdf	to_hdf
SQL	read_sql	to_sql

[표 2-1] 판다스 데이터 입출력 도구(출처: <http://pandas.pydata.org>)

## 1. 외부파일 읽기

### ▪ CSV 파일

- 데이터 값을 쉼표(,)로 구분하고 있다는 의미로 CSV(comma-separated values)라고 부르는 텍스트 파일
- 쉼표(,)로 열을 구분하고 줄바꿈으로 행을 구분함.
- `read_csv()` 함수에 확장자(.csv)를 포함하여 파일경로(파일명)을 입력하면, CSV 파일을 읽어와서 데이터프레임으로 변환함

CSV 파일 → 데이터프레임: `pandas.read_csv("파일 경로(이름)")`

- `read_csv()` 함수의 `header` 옵션은 데이터프레임의 열 이름으로 사용할 행을 지정함

## ■ CSV 파일

〈CSV 파일〉

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

\* header 옵션

- '열 이름'이 되는 행을 지정
- `read_csv(file, header=?)`

❶ **header=0** (기본 값: 0행을 열 지정): `df = read_csv(file)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

❷ **header=1** (1행을 열 지정): `df = read_csv(file, header=1)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	0	1	4	7
0	1	2	5	8
1	2	3	6	9

❸ **header=None** (행을 열 지정하지 않음): `df = read_csv(file, header=None)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

[그림 2-1] CSV 파일 읽기 - header 옵션 비교

## ■ CSV 파일

〈CSV 파일〉

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

\* index\_col 옵션

- '행 주소'가 되는 열을 지정
- `read_csv(file, index_col=?)`

❶ `index_col=False` (인덱스 지정하지 않음)

: `df = read_csv(file, index_col=False)`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

❷ `index_col='c0'` ('c0'열을 인덱스 지정)

: `df = read_csv(file, index_col='c0')`

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c1	c2	c3
0	1	4	7
1	2	5	8
2	3	6	9

[그림 2-2] CSV 파일 읽기 - index\_col 옵션 비교

## ■ CSV 파일

옵션	설명
path	파일의 위치(파일명 포함), URL
sep(또는 delimiter)	텍스트 데이터를 필드별로 구분하는 문자
header	열 이름으로 사용될 행의 번호(기본값은 0) header가 없고 첫 행부터 데이터가 있는 경우 None으로 지정 가능
index_col	행 인덱스로 사용할 열의 번호 또는 열 이름
names	열 이름으로 사용할 문자열의 리스트
skiprows	처음 몇 줄을 skip할 것인지 설정(숫자 입력) skip하려는 행의 번호를 담은 리스트로 설정 가능(예: [1, 3, 5])
parse_dates	날짜 텍스트를 datetime64로 변환할 것인지 설정(기본값은 False)
skip_footer	마지막 몇 줄을 skip할 것인지 설정(숫자 입력)
encoding	텍스트 인코딩 종류를 지정(예: 'utf-8')

[표 2-2] read\_csv() 함수의 옵션



## [ 예제 2-1 ] ① CSV 파일 미리보기

예제에서 불러올 CSV 파일의 내용을 확인하면, 데이터가 쉼표(,)와 행으로 구분된 것을 확인할 수 있다.

	A	B	C	D
1	c0	c1	c2	c3
2	0	1	4	7
3	1	2	5	8
4	2	3	6	9

### 〈CSV 파일〉 미리보기

```
1  c0,c1,c2,c3
2  0,1,4,7
3  1,2,5,8
4  2,3,6,9
```

## [ 예제 2-1 ] ② CSV 파일 읽어오기

header 옵션이 없으면 CSV 파일의 첫 행의 데이터(c0,c1,c2,c3)가 열 이름이 된다.  
 한편, index\_col 옵션을 지정하지 않으면, 행 인덱스는 정수 0, 1, 2가 자동으로 지정된다.  
 데이터프레임 df4의 경우, index\_col='c0' 옵션을 사용하여 'c0' 열이 행 인덱스가 되는 것을 볼 수 있다.

```

3 # 라이브러리 불러오기
4 import pandas as pd
5
6 # 파일 경로(파이썬 파일과 같은 폴더)를 찾고, 변수 file_path에 저장
7 file_path = './read_csv_sample.csv'
8
9 # read_csv() 함수로 데이터프레임 변환. 변수 df1에 저장
10 df1 = pd.read_csv(file_path)
11 print(df1)
12 print('\n')
13
14 # read_csv() 함수로 데이터프레임 변환. 변수 df2에 저장. header=None 옵션
15 df2 = pd.read_csv(file_path, header=None)
16 print(df2)
17 print('\n')
18
19 # read_csv() 함수로 데이터프레임 변환. 변수 df3에 저장. index_col=None 옵션
20 df3 = pd.read_csv(file_path, index_col=None)
21 print(df3)
22 print('\n')
23
24 # read_csv() 함수로 데이터프레임 변환. 변수 df4에 저장. index_col='c0' 옵션
25 df4 = pd.read_csv(file_path, index_col='c0')
26 print(df4)
    
```

〈실행 결과〉 코드 전부 실행

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

	0	1	2	3
0	c0	c1	c2	c3
1	0	1	4	7
2	1	2	5	8
3	2	3	6	9

	c0	c1	c2	c3
0	0	1	4	7
1	1	2	5	8
2	2	3	6	9

	c1	c2	c3
c0			
0	1	4	7
1	2	5	8
2	3	6	9

## 1. 외부파일 읽기

### ▪ Excel 파일

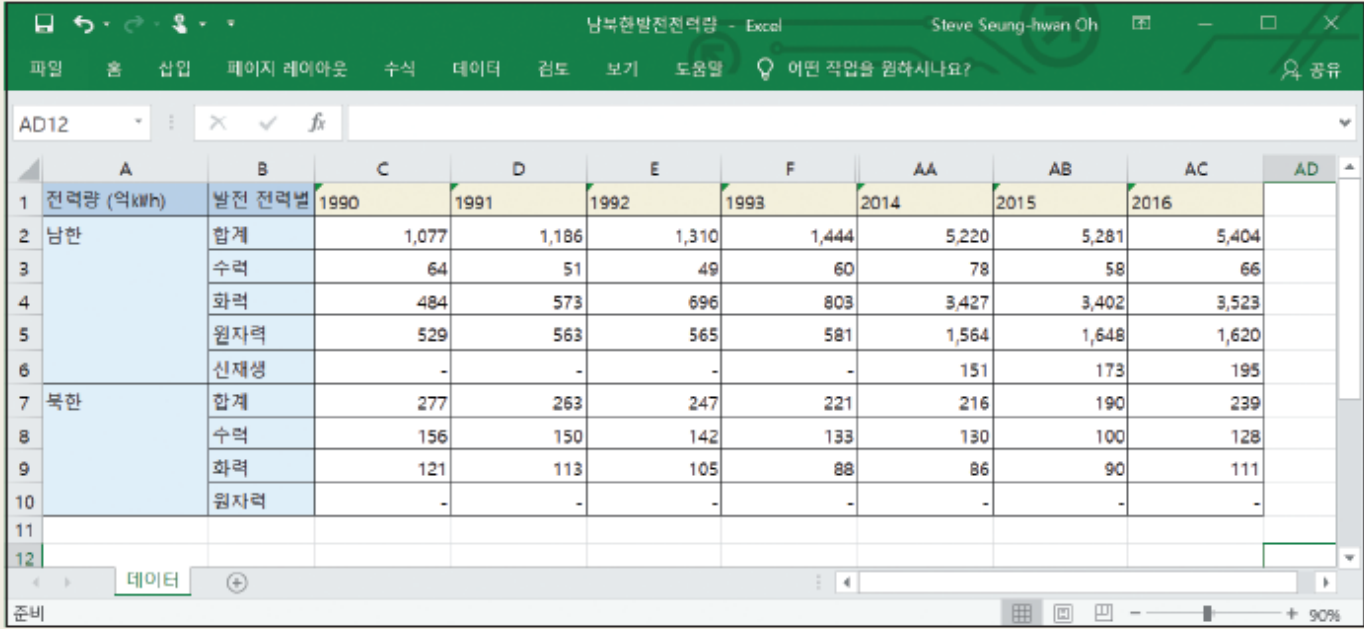
- Excel 파일(확장자: .xlsx)의 행과 열은 데이터프레임의 행, 열로 일대일 대응됨
- read\_excel() 함수의 사용법은 앞에서 살펴본 read\_csv() 함수와 거의 비슷함
- header, index\_col 등 대부분의 옵션을 그대로 사용할 수 있음

Excel 파일 → 데이터프레임: `pandas.read_excel("파일 경로(이름) ")`

# 1. 데이터 입출력

## [ 예제 2-2 ] ① Excel 파일 미리보기

Excel 파일은 남북한의 발전량을 정리한 통계자료이며, 파이썬 파일과 같은 폴더에 저장



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E	F	AA	AB	AC	AD
1	전력량 (억kWh)	발전 전력별	1990	1991	1992	1993	2014	2015	2016	
2	남한	합계	1,077	1,186	1,310	1,444	5,220	5,281	5,404	
3		수력	64	51	49	60	78	58	66	
4		화력	484	573	696	803	3,427	3,402	3,523	
5		원자력	529	563	565	581	1,564	1,648	1,620	
6		신재생	-	-	-	-	151	173	195	
7	북한	합계	277	263	247	221	216	190	239	
8		수력	156	150	142	133	130	100	128	
9		화력	121	113	105	88	86	90	111	
10		원자력	-	-	-	-	-	-	-	
11										
12										

코드 실행 전 Excel 파일 데이터 추출을 지원하는 openpyxl 라이브러리를 설치해야 함

```
pip install openpyxl
```

## [ 예제 2-2 ] ② Excel 파일 읽어오기

header 옵션을 추가하지 않은 경우에는 Excel 파일의 첫 행이 열 이름을 구성한다.  
한편, header=None 옵션을 사용하면, 정수형 인덱스(0, 1, 2, ...)를 열 이름으로 자동 할당한다.

```
3 import pandas as pd
4
5 # read_excel() 함수로 데이터프레임 변환
6 df1 = pd.read_excel('./남북한발전전력량.xlsx')          # header=0 (default 옵션)
7 df2 = pd.read_excel('./남북한발전전력량.xlsx', header=None) # header = None 옵션
8
9 # 데이터프레임 출력
10 print(df1)
11 print('\n')
12 print(df2)
```

<실행 결과> 코드 전부 실행

전력량 (kWh)	발전	전력별	1990	1991	1992	...	2012	2013	2014	2015	2016
0	남한	합계	1077	1186	1310	...	5096	5171	5220	5281	5404
1	NaN	수력	64	51	49	...	77	84	78	58	66
2	NaN	화력	484	573	696	...	3430	3581	3427	3402	3523
3	NaN	원자력	529	563	565	...	1503	1388	1564	1648	1620
4	NaN	신재생	-	-	-	...	86	118	151	173	195
5	북한	합계	277	263	247	...	215	221	216	190	239
6	NaN	수력	156	150	142	...	135	139	130	100	128
7	NaN	화력	121	113	105	...	80	82	86	90	111
8	NaN	원자력	-	-	-	...	-	-	-	-	-

[9 rows x 29 columns]

	0	1	2	3	4	...	24	25	26	27	28	
0	전력량 (kWh)	발전	전력별	1990	1991	1992	...	2012	2013	2014	2015	2016
1	남한	합계		1077	1186	1310	...	5096	5171	5220	5281	5404
2	NaN	수력		64	51	49	...	77	84	78	58	66
3	NaN	화력		484	573	696	...	3430	3581	3427	3402	3523
4	NaN	원자력		529	563	565	...	1503	1388	1564	1648	1620
5	NaN	신재생		-	-	-	...	86	118	151	173	195
6	북한	합계		277	263	247	...	215	221	216	190	239
7	NaN	수력		156	150	142	...	135	139	130	100	128
8	NaN	화력		121	113	105	...	80	82	86	90	111
9	NaN	원자력		-	-	-	...	-	-	-	-	-

[10 rows x 29 columns]

## 1. 외부파일 읽기

### ▪ JSON 파일

- JSON 파일(확장자: .json)은 JavaScript에서 유래한 데이터 공유를 목적으로 개발된 특수한 파일형식
- 파이썬 딕셔너리와 비슷하게 'key : value' 구조를 가짐
- read\_json() 함수를 사용하여, JSON 파일을 데이터프레임으로 변환함

JSON 파일 → 데이터프레임: `pandas.read_json("파일 경로(이름) ")`

## [ 예제 2-3 ] ① JSON 파일 미리보기

JSON 파일에는 주요 파이썬 패키지의 출시년도, 개발자, 오픈소스 정보가 들어 있음

```
1  {
2      "name": {"pandas": "",
3              "NumPy": "",
4              "matplotlib": ""},
5
6      "year": {"pandas": 2008,
7              "NumPy": 2006,
8              "matplotlib": 2003},
9
10     "developer": {"pandas": "Wes McKinney",
11                  "NumPy": "Travis Oliphant",
12                  "matplotlib": "John D. Hunter"},
13
14     "opensource": {"pandas": "True",
15                   "NumPy": "True",
16                   "matplotlib": "True"}
17 }
```

## [ 예제 2-2 ] ② JSON 파일 읽어오기

JSON 파일의 "name" 데이터("pandas", "NumPy", "matplotlib")가 인덱스로 지정된다.

```
3 import pandas as pd
4
5 # read_json() 함수로 데이터프레임 변환
6 df = pd.read_json('./read_json_sample.json')
7 print(df)
8 print('\n')
9 print(df.index)
```

〈실행 결과〉 코드 전부 실행

name	year	developer	opensource
NumPy	2006	Travis Oliphant	True
matplotlib	2003	John D. Hunter	True
pandas	2008	Wes Mckinney	True

Index(['NumPy', 'matplotlib', 'pandas'], dtype='object')



## 2. 웹(web)에서 가져오기

### ▪ HTML 웹 페이지에서 표 속성 가져오기

- read\_html() 함수는 HTML 웹 페이지에 있는 <table> 태그에서 표 형식의 데이터를 모두 찾아서 데이터프레임으로 변환함

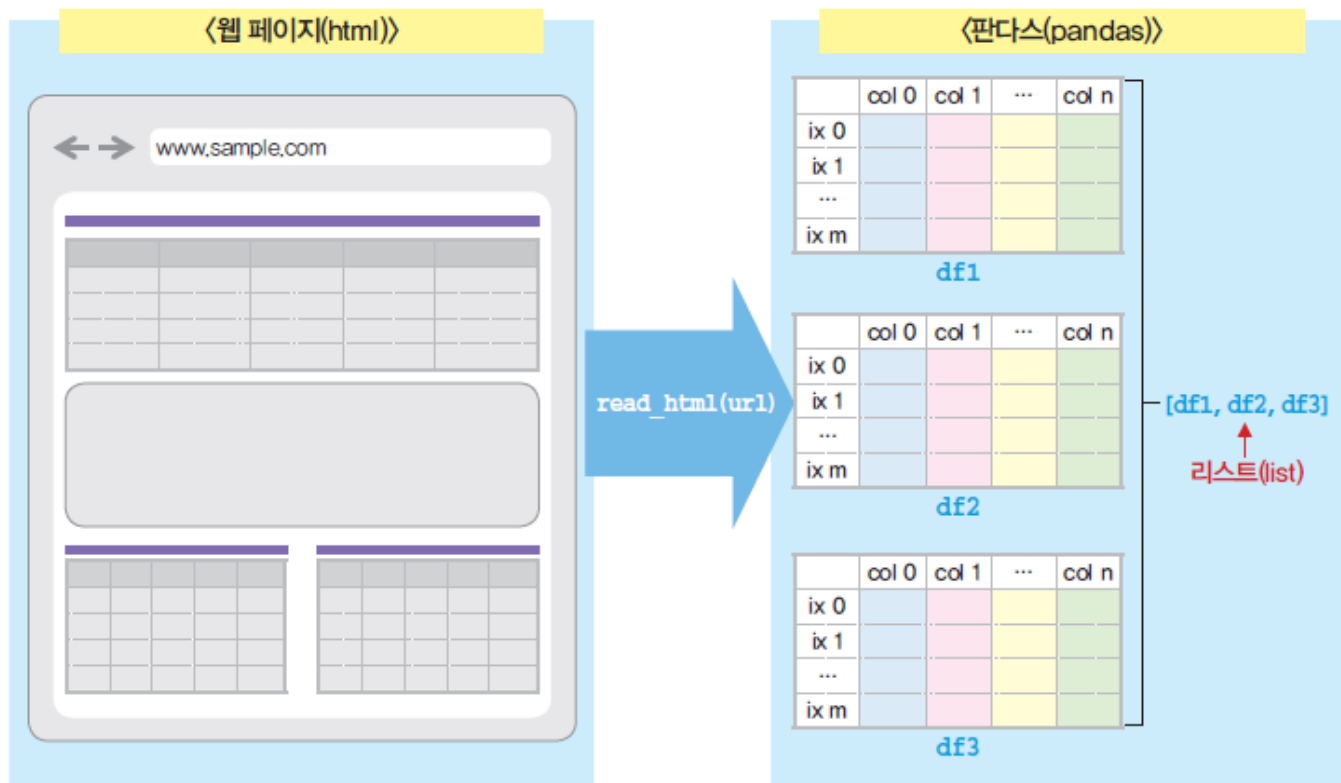
HTML 표 속성 읽기 : `pandas.read_html( "웹 주소(URL)" 또는 "HTML 파일 경로(이름)" )`

코드 실행 전 지원하는 lxml 라이브러리를 설치해야 함

```
pip install lxml
```

## 2. 웹(web)에서 가져오기

### ▪ HTML 웹 페이지에서 표 속성 가져오기



[그림 2-3] HTML 페이지의 표 가져오기

## [ 예제 2-4 ] 웹에서 표 정보 읽기

표 데이터들은 각각 별도의 데이터프레임으로 변환되기 때문에, 여러 개의 데이터프레임(표)을 원소로 갖는 리스트가 반환된다.

〈실행 결과〉 코드 전부 실행

```
3 import pandas as pd
4
5 # HTML 파일 경로 or 웹 페이지 주소를 url 변수에 저장
6 url = './sample.html'
7
8 # HTML 웹페이지의 표(table)를 가져와서 데이터프레임으로 변환
9 tables = pd.read_html(url)
10
11 # 표(table)의 개수 확인
12 print(len(tables))
13
14 # tables 리스트의 원소를 iteration하면서 각각 화면 출력
15 for i in range(len(tables)):
16     print("tables[%s]" % i)
17     print(tables[i])
18     print('\n')
19
20
21 # 파이썬 패키지 정보가 들어 있는 두 번째 데이터프레임을 선택하여 df 변수에 저장
22 df = tables[1]
23
24 # 'name' 열을 인덱스로 지정
25 df.set_index(['name'], inplace=True)
26 print(df)
```

2

tables[0]

Unnamed: 0	c0	c1	c2	c3	
0	0	0	1	4	7
1	1	1	2	5	8
2	2	2	3	6	9

tables[1]

	name	year	developer	opensource
0	NumPy	2006	Travis Oliphant	True
1	matplotlib	2003	John D. Hunter	True
2	pandas	2008	Wes McKinney	True

	year	developer	opensource
name			
NumPy	2006	Travis Oliphant	True
matplotlib	2003	John D. Hunter	True
pandas	2008	Wes McKinney	True

## 2. 웹(web)에서 가져오기

### ▪ 웹 스크래핑

- BeautifulSoup 등 웹 스크래핑(scraping) 도구로 수집한 내용을 파이썬 리스트, 딕셔너리 등으로 정리하고, DataFrame() 함수에 리스트/딕셔너리 형태로 전달하여 데이터프레임으로 변환함

코드 실행 전 지원하는 BeautifulSoup 라이브러리를 설치해야 함

```
pip install BeautifulSoup4
```

코드 실행 전 지원하는 requests 라이브러리를 설치해야 함

```
pip install requests
```

## [ 예제 2-5 ] 미국 ETF 리스트 가져오기

다음은 위키피디아에서 미국 ETF 리스트 데이터를 가져와서 데이터프레임으로 변환하는 예제이다.

The screenshot shows the Wikipedia page for 'List of American exchange-traded funds'. The page title is 'List of American exchange-traded funds'. Below the title, it says 'From Wikipedia, the free encyclopedia'. The main text describes the list as dynamic and incomplete, encouraging users to add missing items with reliable sources. It provides a table of notable American exchange-traded funds (ETFs) as of 2020, listing the number of funds, assets, and the largest ETF (SPDR S&P 500 ETF Trust). The table is partially visible, showing columns for 'Number of funds', 'Assets', and 'Largest ETF'. The 'Contents' section is expanded, showing a list of ETFs categorized by type: 1. Stock ETFs, 1.1 Broad market ETFs, 1.2 Index-tracking ETFs, 1.3 Style ETFs, 1.4 International ETFs, and 1.5 Sector ETFs. The 'Broad market ETFs' section is further expanded, listing several ETFs with their ticker symbols and descriptions.

### Stock ETFs [ edit ]

#### Broad market ETFs [ edit ]

- iShares Core S&P Total US Stock Mkt (NYSE Arca: ITOT<sup>🔗</sup>)
- iShares MSCI ACWI Index (Nasdaq: ACWI<sup>🔗</sup>)
- iShares Russell 3000 Index (NYSE Arca: IWV<sup>🔗</sup>)
- Schwab US Broad Market ETF (NYSE Arca: SCHB<sup>🔗</sup>)
- Schwab Fundamental U.S. Broad Market Index ETF (NYSE Arca: FNDB<sup>🔗</sup>)
- Vanguard Total World Stock (NYSE Arca: VT<sup>🔗</sup>), tracks the FTSE All-World Index
- Vanguard Total Stock Market (NYSE Arca: VTI<sup>🔗</sup>), tracks the MSCI US Broad Market Index
- Vanguard Total International Stock (NYSE Arca: VXUS<sup>🔗</sup>), tracks the MSCI All Country World ex-USA Investable Market Index
- Vanguard Russell 3000 (NYSE Arca: VTHR<sup>🔗</sup>), tracks 98% of the US market

#### Index-tracking ETFs [ edit ]

- DIAMONDS Trust, Series 1 (NYSE Arca:DIA), tracks the Dow Jones Industrial Average
- Guggenheim S&P 500 Equal Weight (NYSE Arca:RSP)
- iShares S&P Global 100 Index (NYSE Arca:IOO), tracks the S&P Global 100
- iShares S&P 500 Index (NYSE Arca:IVV), tracks the S&P 500
- SPDR S&P 500 (NYSE Arca:SPY), tracks the S&P 500
- SPDR Gender Diversity (NYSE Arca: |SHE), tracks a gender-diverse corporate leadership index
- Vanguard S&P 500 (NYSE Arca:VOO)
- iShares Russell 2000 Index (NYSE Arca:IWM), tracks the Russell 2000
- iShares S&P 100 Index (NYSE Arca:OEF), tracks the S&P 100
- PowerShares QQQ ("cubes") (NASDAQ:QQQ), tracks the NASDAQ-100

## [ 예제 2-5 ] 미국 ETF 리스트 가져오기

24번 라인의 `etfs[etf_ticker[0]] = [etf_market[0], etf_name[0]]` 와 같이 리스트를 원소로 갖는 딕셔너리를 정의하는 방법을 반드시 기억한다. 왼쪽의 딕셔너리 키는 열 이름이 되고, 오른쪽 리스트는 열 데이터가 된다. 예제에서는, ETF 거래코드(`etf_ticker`)가 데이터프레임의 열 이름이 된다.

```
3 # 라이브러리 불러오기
4 from bs4 import BeautifulSoup
5 import requests
6 import re
7 import pandas as pd
8
9 # 위키피디아 미국 ETF 웹 페이지에서 필요한 정보를 스크래핑하여 딕셔너리 형태로 변수 etfs에 저장
10 url = "https://en.wikipedia.org/wiki/List_of_American_exchange-traded_funds"
11 resp = requests.get(url)
12 soup = BeautifulSoup(resp.text, 'lxml')
13 rows = soup.select('div > ul > li')
14
15 etfs = {}
16 for row in rows:
17
18     try:
19         etf_name = re.findall('^(.*) \((NYSE', row.text)
20         etf_market = re.findall('\((.*)\)', row.text)
21         etf_ticker = re.findall('NYSE Arca\|(.*)\)', row.text)
22
23         if (len(etf_ticker) > 0) & (len(etf_market) > 0):
24             etfs[etf_ticker[0]] = [etf_market[0], etf_name[0]]
25
26     except AttributeError as err:
27         pass
28
29 # etfs 딕셔너리 출력
30 print(etfs)
31 print('\n')
32
33 # etfs 딕셔너리를 데이터프레임으로 변환
34 df = pd.DataFrame(etfs)
35 print(df)
```

<실행 결과> 코드 전부 실행

```
{'ITOT': ['NYSE Arca', 'iShares Core S&P Total US Stock Mkt'], 'IWV': ['NYSE Arca', 'iShares Russell 3000 Index'], 'SCHB': ['NYSE Arca', 'Schwab US Broad Market ETF'], 'FND': ['NYSE Arca', 'Schwab Fundamental U.S. Broad Market Index ETF'], 'VT': ['NYSE Arca', 'Vanguard Total World Stock'], 'VTI': ['NYSE Arca', 'Vanguard Total Stock Market'], 'VXUS': ['NYSE Arca', 'Vanguard Total International Stock'], 'VTHR': ['NYSE Arca', 'Vanguard Russell 3000'], 'DIA': ['NYSE Arca', 'DIAMONDS Trust, Series 1'], 'RSP': ['NYSE Arca', 'Guggenheim S&P 500 Equal Weight'], 'IOO': ['NYSE Arca', 'iShares S&P Global 100 Index'],
```

... 중략 ...

```
, 'EMCB': ['NYSE Arca', 'WisdomTree Emerging Markets Corporate Bond Fund'], 'EU': ['NYSE Arca', 'WisdomTree Euro Debt Fund'], 'ICB': ['NYSE Arca', 'WisdomTree Dreyfus Indian Rupee'], 'RRF': ['NYSE Arca', 'WisdomTree Global Real Return'], 'USDU': ['NYSE Arca', 'WisdomTree Bloomberg U.S. Dollar Bullish Fund'], 'WDTI': ['NYSE Arca', 'WisdomTree Managed Futures Strategy Fund']}
```

	ITOT	...	WDTI
0	NYSE Arca	...	NYSE Arca
1	iShares Core S&P Total US Stock Mkt	...	WisdomTree Managed Futures Strategy Fund

[2 rows x 378 columns]

378개 -> 381개로 증가

### 3. API 활용하여 데이터 수집하기

- 인터넷 업체가 제공하는 API를 통해서 수집한 데이터를, 판다스 자료구조로 변환하는 방법
- API를 통해 가져온 데이터를 판다스 데이터프레임으로 손쉽게 변환할 수 있음

#### ▪ Google 지오코딩 API

- 구글 지오코딩 : 장소 이름 또는 주소를 입력하면, 위도와 경도 좌표 정보를 변환해 주는 서비스
- 서비스를 이용하려면, 사용자 인증 후에 API 키를 발급받아야 함

## 3. API 활용하여 데이터 수집하기

### ▪ Google 지오코딩 API

#### 구글 지오코딩 API 발급 절차

1. 구글 지도 서비스(<https://cloud.google.com/maps-platform/places/?hl=ko>) 접속
2. 새 프로젝트 만들기(무료 평가판은 사용 가능하지만 신용카드 정보 등록 필요)
3. API 설정
4. 사용자 인증
5. API 키 발급



### 3. API 활용하여 데이터 수집하기

#### ▪ Google 지오코딩 API

- 프로그램과 구글을 연결할 때 필요한 API 키가 필요
  - API 키를 얻기 위해서는 구글 계정이 필요

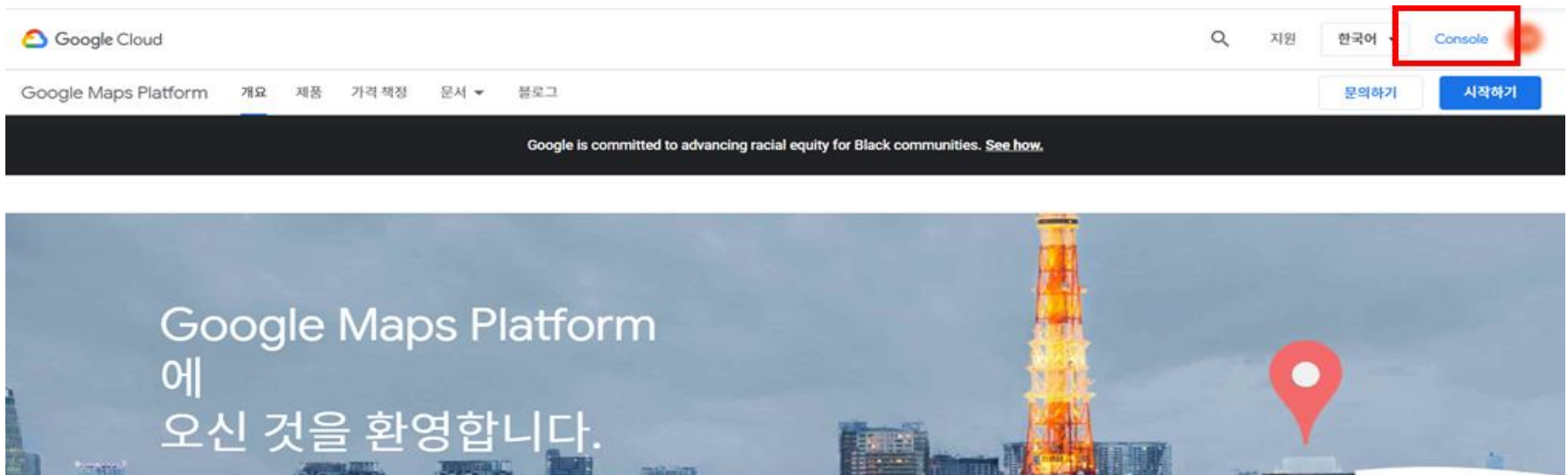
(구글 계정이 없는 학생들은 가입을 한 후 구글 계정으로 로그인 후 진행)

\*\* 구글맵 플랫폼을 사용하기 위해서 무료 체험판을 선택하여 신청하나 가입 작성 시 개인 정보를 입력 필요, API 키 사용에 대한 무료 부분 제한이 있으니 참고 \*\*

### 3. API 활용하여 데이터 수집하기

#### ▪ Google 지오코딩 API

- 구글 로그인 후 구글맵 플랫폼('https://cloud.google.com/maps-platform')에 접속

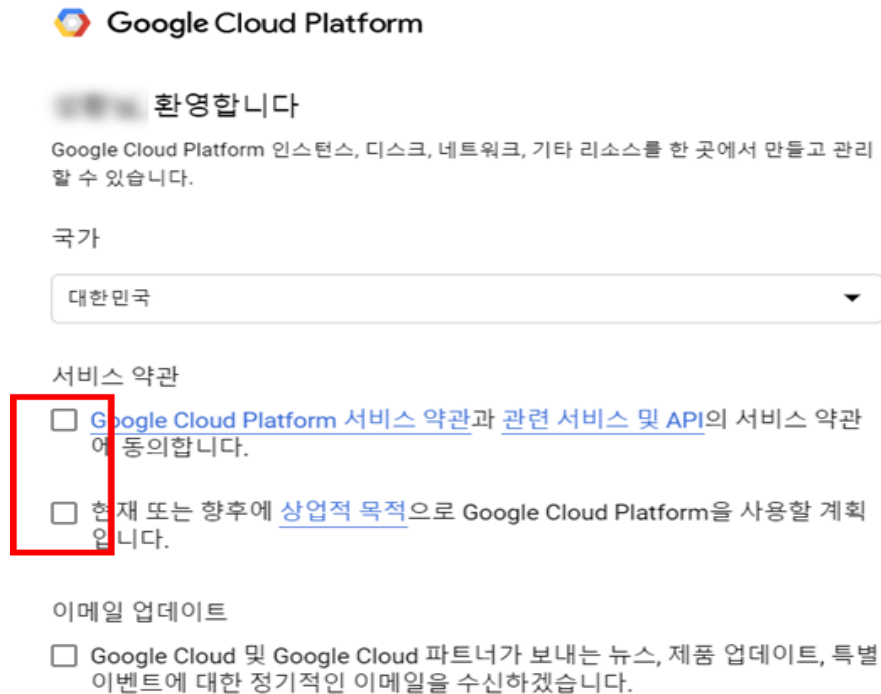


- '한국어'로 설정한 후 콘솔(Console) 버튼 -> 구글 클라우드 플랫폼으로 이동

## 3. API 활용하여 데이터 수집하기

### ▪ Google 지오큐딩 API

– 서비스 약관에 동의 후 계속



Google Cloud Platform

환영합니다

Google Cloud Platform 인스턴스, 디스크, 네트워크, 기타 리소스를 한 곳에서 만들고 관리할 수 있습니다.

국가

대한민국

서비스 약관

☒ Google Cloud Platform 서비스 약관과 [관련 서비스 및 API](#)의 서비스 약관에 동의합니다.

☐ 현재 또는 향후에 [상업적 목적](#)으로 Google Cloud Platform을 사용할 계획입니다.

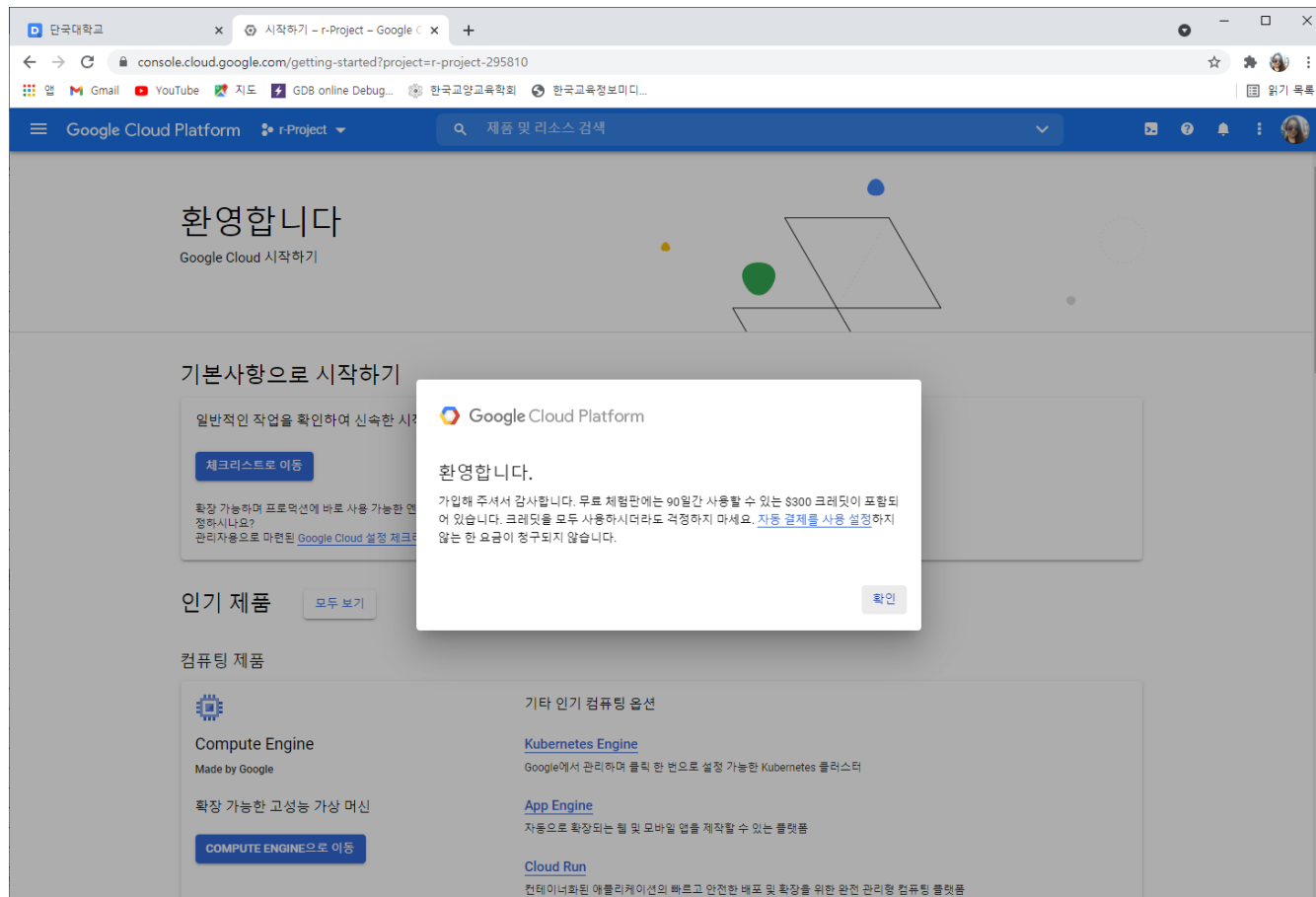
이메일 업데이트

☐ Google Cloud 및 Google Cloud 파트너가 보내는 뉴스, 제품 업데이트, 특별 이벤트에 대한 정기적인 이메일을 수신하겠습니다.

동의 및 계속하기

## 3. API 활용하여 데이터 수집하기

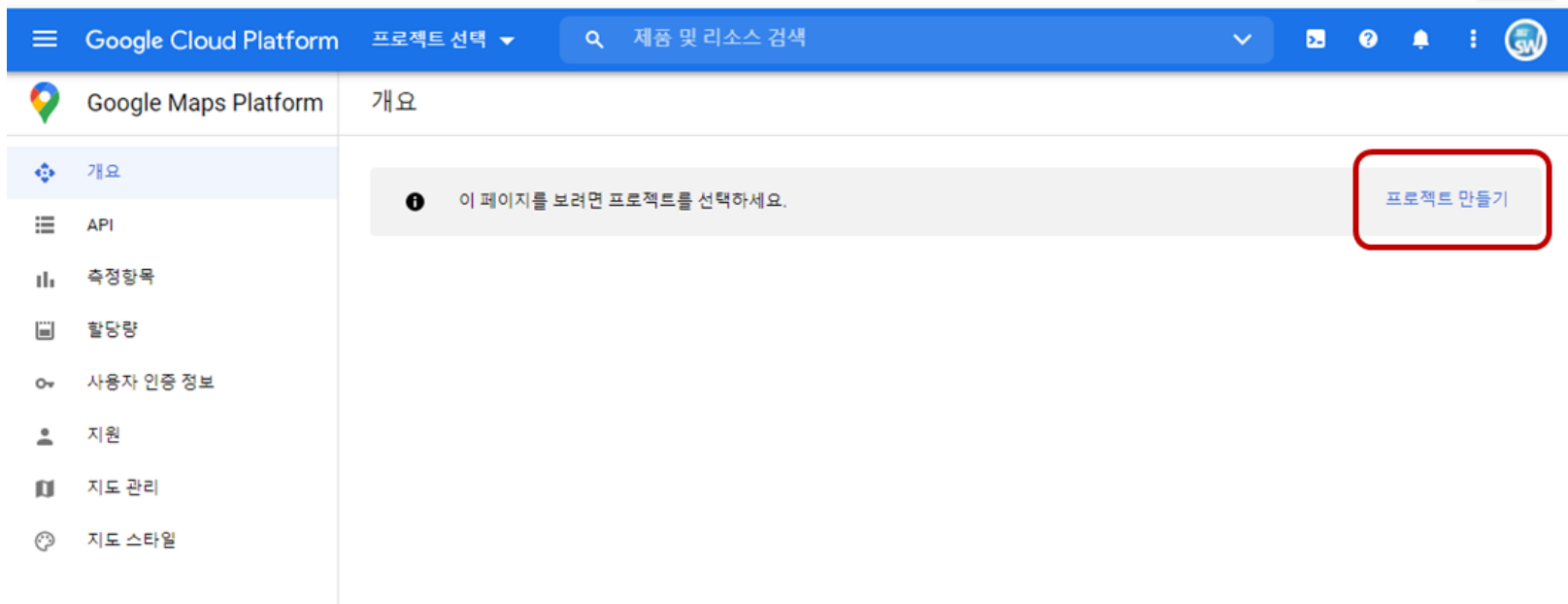
### ■ Google 지오큐딩 API



### 3. API 활용하여 데이터 수집하기

#### ▪ Google 지오코딩 API

- Google 지도 탭 우측 상단에 있는 프로젝트 만들기 버튼 클릭



## 3. API 활용하여 데이터 수집하기

### ■ Google 지오킨 API

- 프로젝트 이름( R-project) 작성 -> 만들기 버튼 클릭
- 프로젝트 할당량이 제한되어 있으니 참고

Google Cloud Platform

새 프로젝트

projects 할당량이 12개 남았습니다. 할당량 증가를 요청하거나 프로젝트를 삭제하세요. [자세히 알아보기](#)

[MANAGE QUOTAS](#)

프로젝트 이름 \*

R-project

프로젝트 ID: r-project-295809입니다. 나중에 변경할 수 없습니다. [수정](#)

위치 \*

조직 없음 [찾아보기](#)

상위 조직 또는 폴더










[만들기](#) 취소

Google Cloud Platform R-project

Google Maps Platform

Maps API 및 서비스

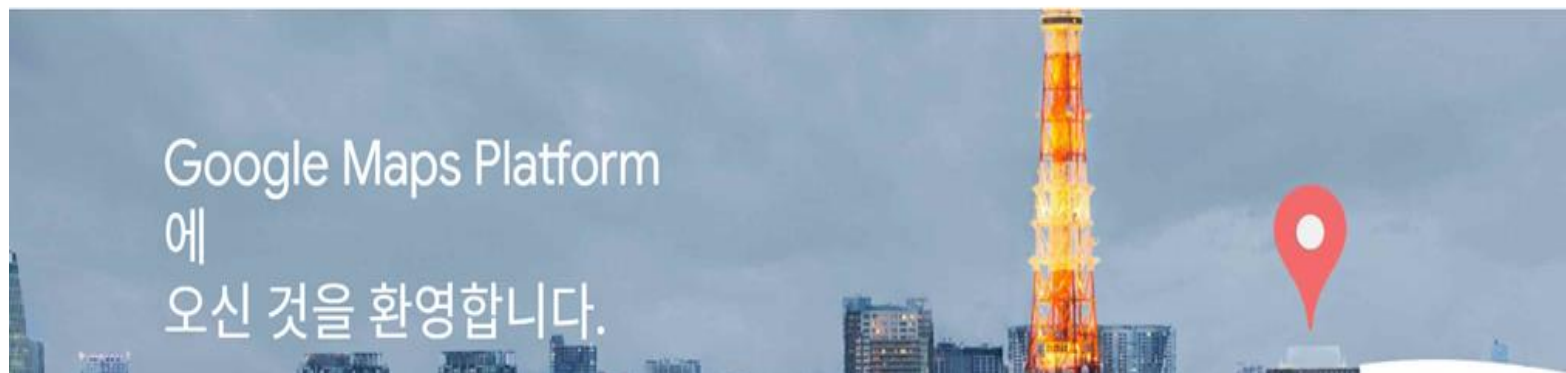
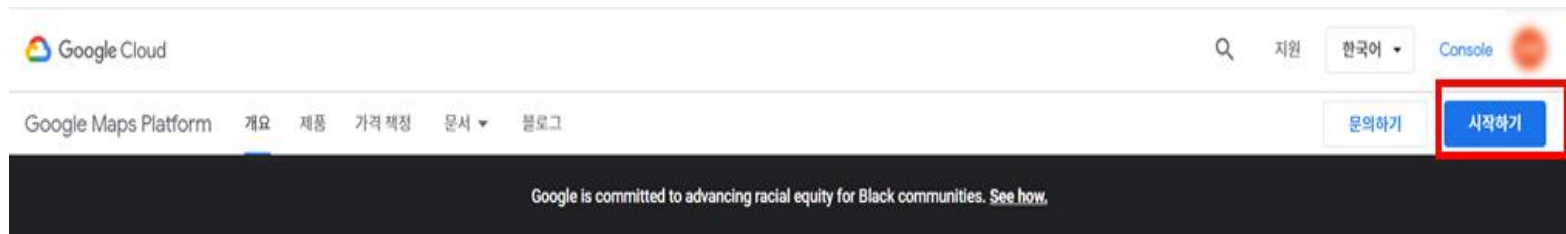
Maps API 및 서비스를 찾아보고 사용하려면 [Marketplace](#)를 둘러보세요.

 <b>Geolocation API</b> Google Location data from cell towers and WiFi nodes.	 <b>Maps SDK for Android</b> Google Maps for your native Android app.	 <b>Roads API</b> Google Snap-to-road functions accurately trace GPS b
 <b>Places API</b> Google Get detailed information about 100 million places	 <b>Maps JavaScript API</b> Google Maps for your website	 <b>Maps Embed API</b> Google Make places easily disp with interactive Google
 <b>Time Zone API</b> Google Time zone data for countries in the	 <b>Distance Matrix API</b> Google Travel time and distance for	 <b>Maps SDK for iOS</b> Google Maps for your native iOS app.

### 3. API 활용하여 데이터 수집하기

#### ▪ Google 지오코딩 API

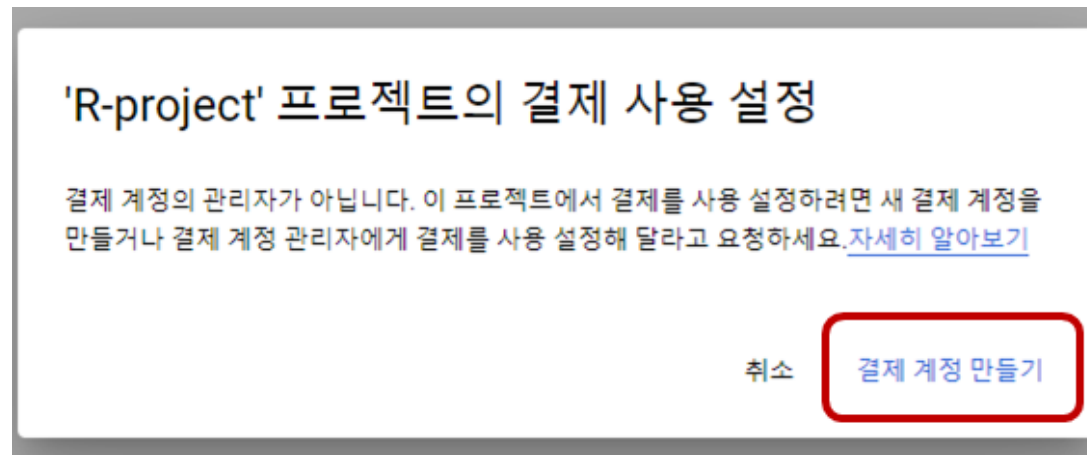
– 다시 구글맵 플랫폼에 접속 후 [시작하기] 버튼 클릭



### 3. API 활용하여 데이터 수집하기

#### ▪ Google 지오큐딩 API

– [결제 계정 만들기] 버튼을 클릭

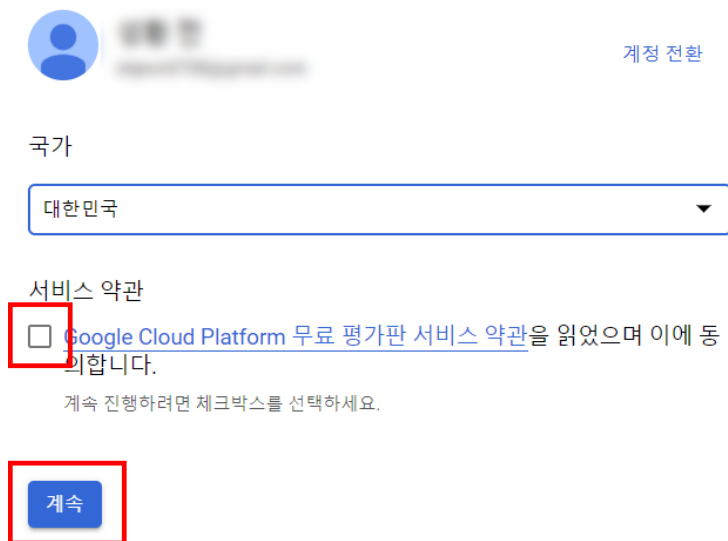




### 3. API 활용하여 데이터 수집하기

#### ▪ Google 지오키오 API

- 서비스 약관 체크박스를 클릭 후 [계속] 버튼을 클릭
- 개인정보 입력 후 진행함  
1/2단계



계정 전환

국가

대한민국

서비스 약관

☐ Google Cloud Platform 무료 평가판 서비스 약관을 읽었으며 이에 동의합니다.

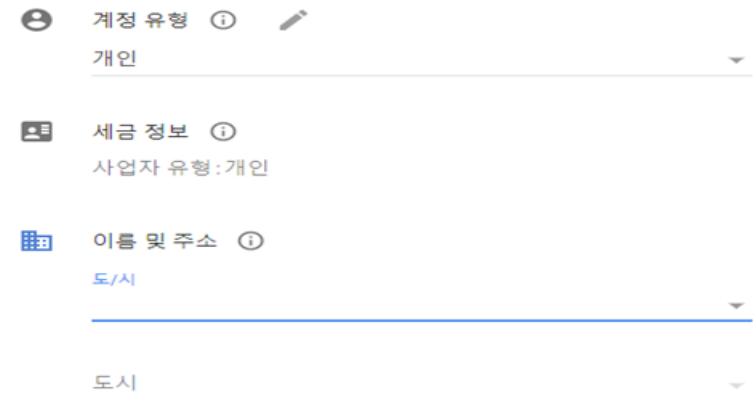
계속 진행하려면 체크박스를 선택하세요.

계속

Google Cloud Platform 무료로 사용해 보기

#### 2/2단계

##### 고객 정보



계정 유형 ⓘ

개인

세금 정보 ⓘ

사업자 유형: 개인

이름 및 주소 ⓘ

도/시


도시

- 이후 안내에 따라 결제 계정 정보 입력  
(구글맵은 일정 기간 동안 무료로 사용할 수 있음. 단, 결제 계정 정보는 등록해야 함)

### 3. API 활용하여 데이터 수집하기

#### ■ Google 지오코딩 API

- 모두 체크 후 사용 설정을 클릭

 환영합니다.

Google Maps Platform으로 할 수 있는 작업에는 제한이 없습니다.

#### 아래에서 제품 선택

API 사용 설정은 무료이며 사용량에 따라 요금이 청구됩니다.

☐ 지도

사용자에게 장소의 실제 모습을 보여주는 맞춤설정된 지도 환경을 구축하세요.

☐ 경로

사용자에게 출발지에서 목적지까지 가는 최적의 경로를 알려주세요.

☐ 장소

풍부한 상세정보로 사용자의 장소 탐색을 도와주세요.

사용 설정

취소



### 3. API 활용하여 데이터 수집하기

#### ▪ **googlemaps 라이브러리 설치(아나콘다 이용 시 참고)**

- 1) 아나콘다 프롬프트를 실행
- 2) 콘다 설치 명령 `conda install -c conda-forge googlemaps`
- 3) 설치 계속 여부 확인 : 'y' 입력하고 Enter를 누름
- 4) 설치가 완료됨

### 3. API 활용하여 데이터 수집하기

- **googlemaps 라이브러리 설치(아나콘다 이용 x 참고)**

```
pip install googlemaps
```

## [ 예제 2-6 ] ① 지오코딩 API 호출 결과 미리보기

구글 지오코딩 API를 사용하여 “해운대해수욕장” 위치 정보를 확인하면, 딕셔너리와 비슷한 형태를 갖는다.

예를 들면, 'location'을 키로 하는 데이터 중에서 'lat'와 매칭되는 숫자가 “해운대해수욕장”의 위도를 나타낸다. 경도는 'lng' 값과 매칭되는 숫자다.

### 〈참고〉 지오코딩 API 호출 결과 미리보기(해운대 해수욕장)

```
1 {'location': {'lat': 35.1586975, 'lng': 129.1603842},  
2  'location_type': 'APPROXIMATE',  
3  'viewport': {'northeast': {'lat': 35.1678193, 'lng': 129.1763916},  
4  'southwest': {'lat': 35.1495747, 'lng': 129.1443768}}}
```

## [ 예제 2-6 ] ② 지오코딩으로 위도, 경도 정보 가져오기

예제코드 9번 라인에 직접 발급받은 API 키를 입력한다. 3개의 장소("서울시청", "국립국악원", "해운대 해수욕장")에 대한 GPS(위도, 경도) 데이터를 2개의 리스트(lat, lng)에 저장한다. DataFrame() 메소드로 데이터프레임을 만들 때, '위도' 열에 lat 리스트를 매칭하고 '경도' 열에는 lng 리스트를 매칭한다. 장소명이 들어 있는 리스트(places)를 행 인덱스로 설정한다.

```
3  ## google 지오코딩 API를 통해 위도, 경도 데이터 가져오기
4
5  # 라이브러리 가져오기
6  import googlemaps
7  import pandas as pd
8
9  my_key = "----발급받은 API 키 입력-----"
10
11 # 구글맵스 객체 생성하기
12 maps = googlemaps.Client(key=my_key) # my key값 입력
13
14 lat = [] # 위도
15 lng = [] # 경도
16
17 # 장소(또는 주소) 리스트
18 places = ["서울시청", "국립국악원", "해운대해수욕장"]
19
20 i=0
21 for place in places:
22     i = i + 1
23     try:
24         print(i, place)
25         # 지오코딩 API 결과값 호출하여 geo_location 변수에 저장
26         geo_location = maps.geocode(place)[0].get('geometry')
27         lat.append(geo_location['location']['lat'])
28         lng.append(geo_location['location']['lng'])
29
30     except:
31         lat.append('')
32         lng.append('')
33         print(i)
```

발급받은 API 키 입력

```
35 # 데이터프레임으로 변환하기
36 df = pd.DataFrame({'위도':lat, '경도':lng}, index=places)
37 print('\n')
38 print(df)
```

〈실행 결과〉 코드 전부 실행

- 1 서울시청
- 2 국립국악원
- 3 해운대해수욕장

	위도	경도
서울시청	37.566295	126.977945
국립국악원	37.477759	127.008304
해운대해수욕장	35.158698	129.160384

### 3. API 활용하여 데이터 수집하기

#### ■ 머신러닝에 유용한 데이터셋 소스

1. 사이킷런(scikit-learn), 시본(seaborn) 등 파이썬 라이브러리 제공 데이터셋
2. 캐글(Kaggle) : <https://www.kaggle.com/>
3. UCI 머신러닝 저장소 : <https://archive.ics.uci.edu/ml/datasets.html>
4. 공공 데이터
  - (해외) WorldBank, WTO 등 국제기구
  - (국내) 공공데이터 포털, 국가통계포털 등



## [공공데이터 제공 사이트]

- 비즈니스 및 산업 전반에 데이터를 활용할 수 있다고 인식
  - > 정부에서는 국가가 보유하고 있는 데이터 중 민감한 정보를 포함하고 있지 않은 데이터를 공개
- 빅데이터를 수집할 때 가장 중요하게 고민해야 할 것은  
'어떤 데이터를 어디에서 어떻게 수집하느냐'를 결정하는 것임

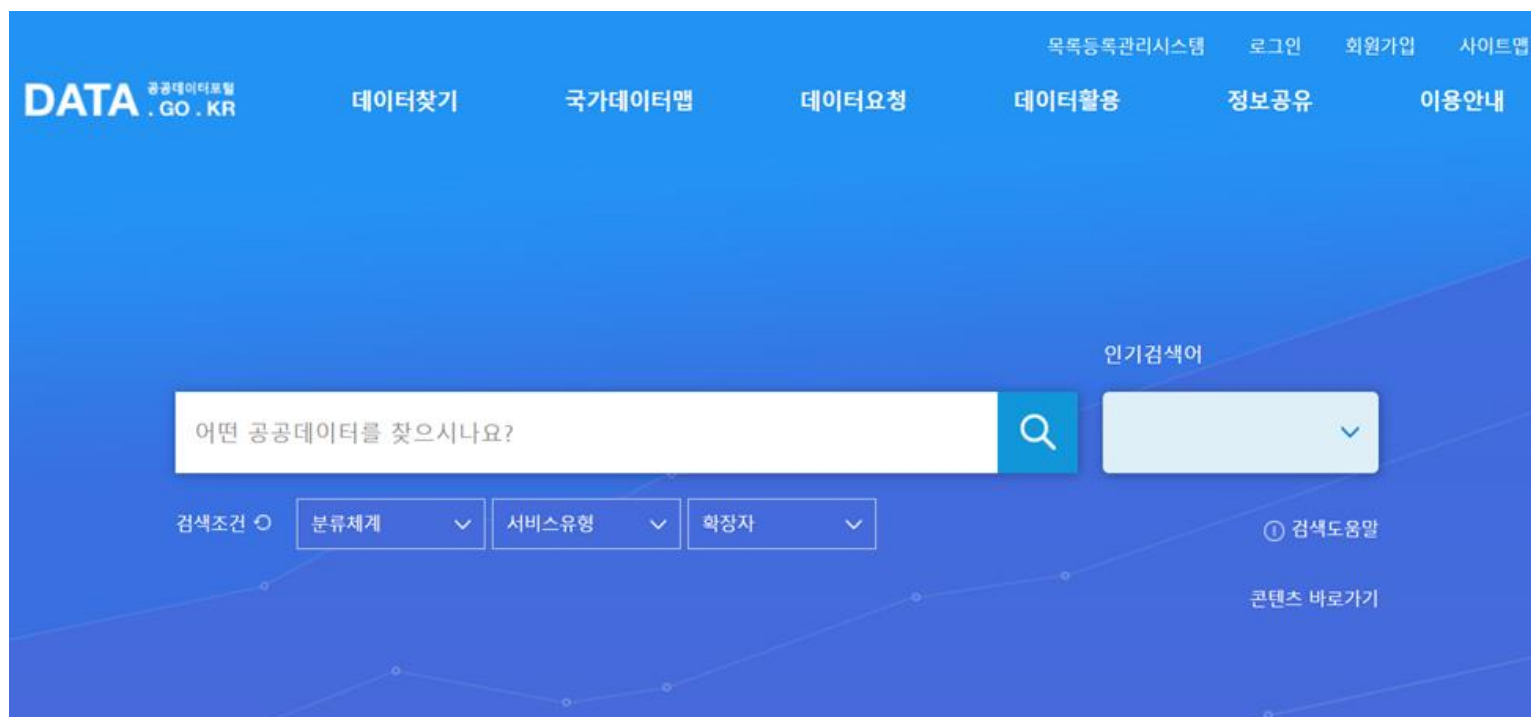
무료, 쉽고 빠르게 데이터 수집

다양한 종류의 공공데이터 정보를 제공하고 있는 웹사이트들  
(공공데이터 포털, 국가통계 포털, 기상청 날씨누리 등..)

### 1) 공공데이터 포털 (<https://www.data.go.kr>)

- 가장 풍부한 공공데이터를 제공하는 웹사이트로 공공기관이 관리하고 많은 양의 데이터를 무료로 활용
- '공공데이터의 제공 및 이용 활성화에 관한 법률'에 따라 각 기관별로 생성·보유하고 있는 공공데이터의 통합 관리 및 민간 개방을 위한 범정부 통합 단일 창구로 구축되어 운영 중인 시스템
- 관심 있는 키워드를 검색해서 다운로드 버튼을 클릭하면 다양한 파일 형식(csv, pdf, xls 등)으로 된 데이터를 쉽게 다운로드 가능
- 오픈 API 같은 경우는 데이터 활용 신청서를 제출하면 받을 수 있음
- 데이터를 비즈니스에 활용한 사례도 제공하고 있기 때문에 창업을 하고자 하는 사람들에게 좋은 참고자료가 되고 있음

## 1) 공공데이터 포털 (<https://www.data.go.kr>)



### 2) 국가통계 포털 (<https://kosis.kr/>)

- 국내외 주요 통계를 한곳에 모아 이용자가 원하는 통계를 한 번에 찾을 수 있도록 통계청이 제공하는 원스톱(One Stop) 통계 서비스 웹사이트
- 현재 300여개 기관이 작성하는 경제, 사회, 환경에 관한 1000여 종의 국가 승인통계를 수록하고 있으며, 국제 금융과 경제에 관한 국제통화기금(IMF), 월드뱅크(Worldbank), 경제협력개발기구(OECD) 등의 최신통계도 제공
- 편리한 검색 기능과 일반인들도 쉽게 이해할 수 있는 다양한 콘텐츠 및 통계 설명 자료 서비스를 제공하고 있으니 통계 자료가 필요할 때 좋은 사이트

# 1. 데이터 입출력

## 2) 국가통계 포털 (<https://kosis.kr/>)

**KOSIS** 국가통계포털 KOSIS KOREAN Statistical Information Service

통계표  🔍

로그인 | 회원가입 | English

국내통계	국제·북한통계	쉽게 보는 통계	온라인간행물	민원안내	서비스 소개
주제별 통계 기관별 통계 e-지방지표(통계표) e-지방지표(시각화) 과거·중지통계	국제통계 북한통계	대상별 접근 이슈별 접근 통계시각화콘텐츠	주제별 명칭별 기획간행물	FAQ Q&A KOSIS 길라잡이 홈페이지 개선의견 찾아가는 KOSIS	국가통계포털 소개 국가통계현황 국가통계 공표일정 새소식 Fact-Check 서비스 서비스정책 부가서비스

2016 2017 2018 2019 2020

임금/물가 국민계정 정부·재정 금융/무역·국제수지 환경 에너지 지역통계

통계목록

- 가구에너지패널조사
- 발전설비현황
- 상용자가발전업체조사
- 석유수급통계
- 신재생에너지보급실적조사
- 신재생에너지설비·연료산...

**신재생에너지보급...**  
한국에너지공단  
신재생에너지, 태양광, 에너지,  
에너지생산량, 성형탄  
설명자료 통계목록

**발전설비현황**  
한국전력거래소  
발전소, 원자력발전소, 발전  
량, 화력발전소, 발전  
설명자료 통계목록

**한국전력통계**  
한국전력공사  
전력, 전력사용량, 송전, 발전  
전력량, 전력판매량  
설명자료 통계목록

추천키워드 인기통계표

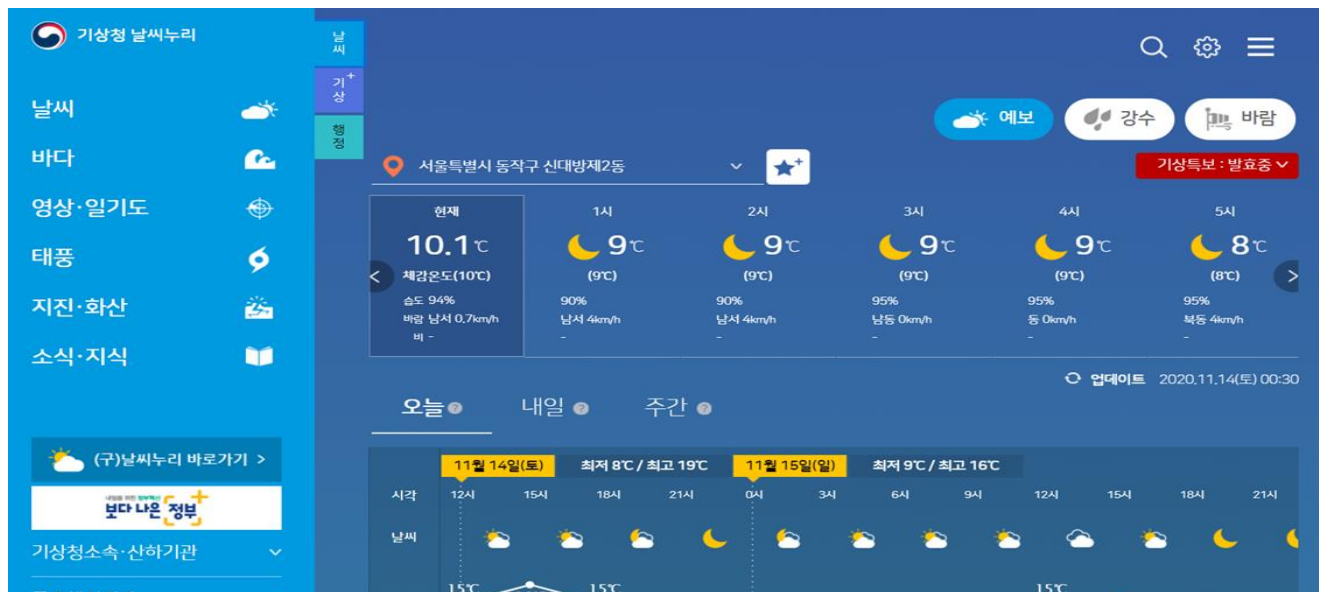
**코로나19 (COVID-19)**

통계시각화콘텐츠

- 통계로 시간여행
- 인구로 보는 대한민국
- 통계로 보는 자화상
- 지역경제 상황판

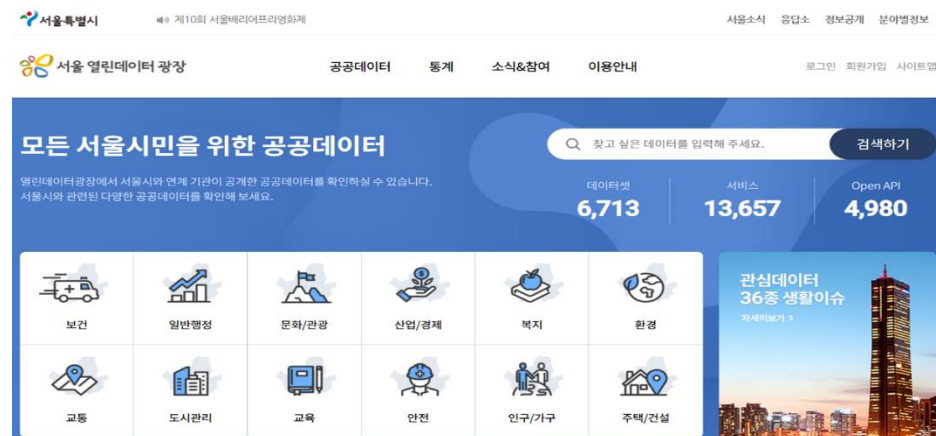
## 3) 기상청 날씨누리 (<https://www.weather.go.kr/>)

- 기상청에서 기상 관련 데이터를 공개하고 있는 사이트
- 기상예보, 태풍, 황사, 위성, 레이더 등 25종 자료를 쉽게 이용할 수 있으며, 현재 기상 자료를 실시간으로 얻을 수 있음
- 기상 관련 앱을 개발할 때 참고하면 좋은 사이트



## 4) 서울 열린 데이터 광장 (<http://data.seoul.go.kr/>)

- 서울시에서 운영하는 공공데이터 포털로 '서울시'에 관련된 데이터를 수집
- 서울시에 관련된 주제로 데이터 분석 시 참고하면 좋은 사이트
- 공공데이터 포털과 마찬가지로 다운로드 버튼을 클릭하면 무료로 쉽게 다운로드할 수 있고 오픈 API는 신청 후 사용 가능
- 서울시 외에도 경기도에서 운영하는 경기 데이터 드림(<https://data.gg.go.kr/>) 사이트가 있는데 서울시 공공데이터 사이트와 비교했을 시 데이터의 개수가 많지 않음



## 5) 네이버 데이터 랩 (<https://datalab.naver.com/>)

- 공공기관에서 운영하는 사이트는 아니지만 가장 쉽게 데이터를 수집할 수 있는 곳
- 장점 : 급상승 검색어나 지역별, 연령별 등 각종 통계를 확인할 수 있고, 블로그나 유튜브 등의 콘텐츠를 제작하는 과정에서 빠르게 해당 키워드의 검색량이나 추이 등을 확인할 수 있음
- 단점 : 심도 있게 데이터를 수집하기에는 한계가 있음
- 빠르게 키워드 검색량 정도를 확인할 때 사용하면 좋은 사이트



NAVER DataLab.

데이터랩 홈 급상승검색어 **검색어트렌드** 쇼팡인사이트 지역통계 댓글통계

**검색어트렌드** 네이버통합검색에서 특정 검색어가 얼마나 많이 검색되었는지 확인해보세요. 검색어를 기간별/연령별/성별로 조회할 수 있습니다.

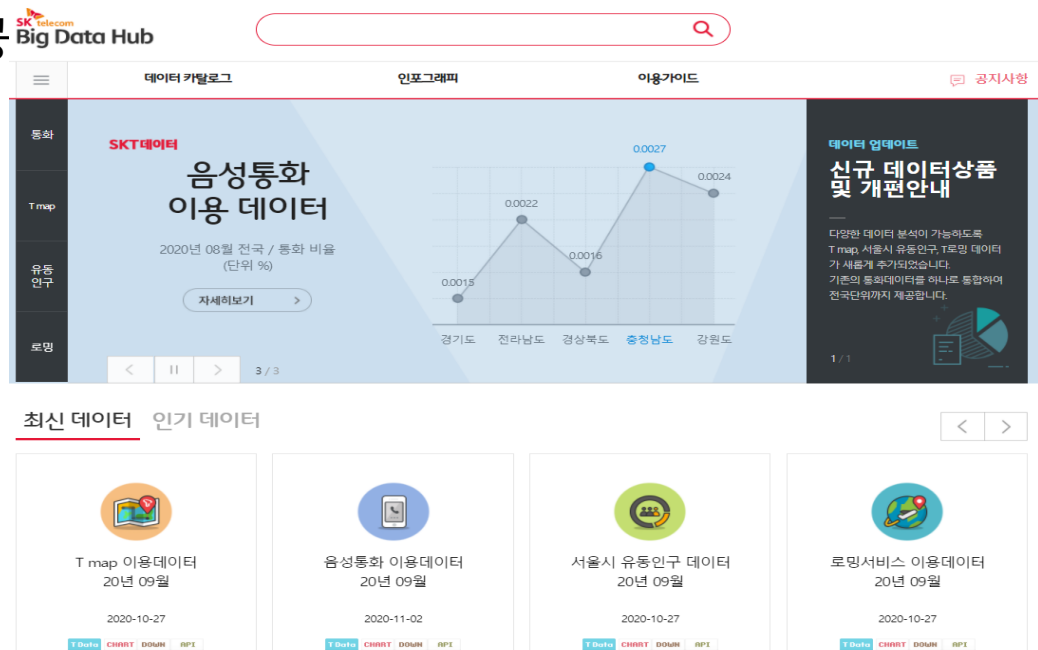
궁금한 주제를 설정하고, 하위 주제어에 해당하는 검색어를 콤마(,)로 구분입력해 주세요. 입력한 단어의 추이를 하나로 합산하여 해당 주제가 네이버에서 얼마나 검색되는지 조회할 수 있습니다. 예) 주제어 캠핑 : 캠핑, Camping, 캠핑용품, 겨울캠핑, 캠핑장, 글램핑, 오토캠핑, 캠핑카, 텐트, 캠핑요리

주제어1	주제어 1 입력	주제어 1에 해당하는 모든 검색어를 콤마(,)로 구분하여 최대 20개까지 입력
주제어2	주제어 2 입력	주제어 2에 해당하는 모든 검색어를 콤마(,)로 구분하여 최대 20개까지 입력
주제어3	주제어 3 입력	주제어 3에 해당하는 모든 검색어를 콤마(,)로 구분하여 최대 20개까지 입력
주제어4	주제어 4 입력	주제어 4에 해당하는 모든 검색어를 콤마(,)로 구분하여 최대 20개까지 입력
주제어5	주제어 5 입력	주제어 5에 해당하는 모든 검색어를 콤마(,)로 구분하여 최대 20개까지 입력



## 6) SK Data Hub(<https://www.bigdatahub.co.kr/>)

- 공공데이터 포털은 아니지만 빅데이터를 수집할 때 유용하게 사용할 수 있는 사이트
- 음성통화 이용 데이터, T-map 이용 데이터, 유동인구 데이터, 로밍 서비스 이용 데이터를 제공
- 다른 사이트들과 마찬가지로 로그인을 하면 해당 데이터를 무료로 다운로드할 수 있고, Open API를 제공



### 4. 데이터 저장하기

#### ▪ CSV 파일로 저장

- 판다스 데이터프레임은 2차원 배열로 구조화된 데이터이기 때문에 2차원 구조를 갖는 CSV 파일로 변환할 수 있음.
- 데이터프레임을 CSV 파일로 저장하려면 `to_csv()` 메소드를 적용함
- CSV 파일을 저장할 파일 경로와 파일명(확장자 포함)을 따옴표(" " 또는 ' ') 안에 입력함

CSV 파일로 저장: `DataFrame` 객체.`to_csv("파일 이름 (경로) ")`

## [ 예제 2-7 ] CSV 파일로 저장

14라인의 `print(df)` 명령에 의해 데이터프레임의 내용이 IPython 콘솔에 출력된다.

15라인은 `to_csv()` 메소드를 적용하여 파이썬 실행 파일이 위치하고 있는 현재 디렉토리에 파일명을 "df\_sample.csv" 저장하는 명령이다. csv 파일을 열어보면 쉼표와 줄바꿈으로 구분되는 2차원 구조가 확인된다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df에 저장
6 data = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7         'algol' : [ "A", "A+", "B"],
8         'basic' : [ "C", "B", "B+"],
9         'c++' : [ "B+", "C", "C+"],
10        }
11
12 df = pd.DataFrame(data)
13 df.set_index('name', inplace=True) # name 열을 인덱스로 지정
14 print(df)
15
16 # to_csv() 메소드를 사용하여 CSV 파일로 내보내기. 파일명은 df_sample.csv로 저장
17 df.to_csv("./df_sample.csv")
```

〈실행 결과〉 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

〈실행 결과〉 코드 전부 실행 ② - CSV 파일 내용 보기(File: example/part2/df\_sample.csv)

```
name,algol,basic,c++
Jerry,A,C,B+
Riah,A+,B,C
Paul,B,B+,C+
```

### 4. 데이터 저장하기

#### ▪ JSON 파일로 저장

- 데이터프레임을 JSON 파일로 저장하려면 `to_json()` 메소드를 이용
- JSON 파일의 이름(확장자 포함)을 저장하려는 파일 경로와 함께 따옴표 (" " 또는 ' ')안에 입력함

JSON 파일로 저장: `DataFrame 객체.to_json("파일 이름 (경로) ")`

## [ 예제 2-8 ] JSON 파일로 저장

print(df) 명령에 의해 데이터프레임의 내용이 출력된다.

to\_json() 메소드를 이용하여 데이터프레임을 현재 디렉터리에 JSON 파일로 변환하여 저장한다.

JSON 파일을 열어보면 데이터프레임의 행, 열이 JSON 파일의 형식에 맞춰 정리된다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df에 저장
6 data = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7         'algol' : [ "A", "A+", "B"],
8         'basic' : [ "C", "B", "B+"],
9         'c++' : [ "B+", "C", "C+"],
10        }
12 df = pd.DataFrame(data)
13 df.set_index('name', inplace=True) # name 열을 인덱스로 지정
14 print(df)
15
16 # to_json() 메소드를 사용하여 JSON 파일로 내보내기. 파일명은 df_sample.json로 저장
17 df.to_json("./df_sample.json")
```

〈실행 결과〉 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

〈실행 결과〉 코드 전부 실행 ② - JSON 파일 내용 보기(File: example/part2/df\_sample.json)

```
{
  "algol":{"Jerry":"A","Riah":"A+","Paul":"B"},
  "basic":{"Jerry":"C","Riah":"B","Paul":"B+"},
  "c++":{"Jerry":"B+","Riah":"C","Paul":"C+"}
}
```

### 4. 데이터 저장하기

#### ▪ Excel 파일로 저장

- 데이터프레임은 Excel 파일과 아주 유사한 구조를 가짐
- 데이터프레임의 행과 열은 Excel 파일의 행과 열로 일대일로 대응됨
- 데이터프레임을 Excel 파일로 저장할 때는 `to_excel()` 메소드를 적용함

Excel 파일로 저장: `DataFrame 객체.to_excel("파일 이름 (경로) ")`

- `to_excel()` 메소드를 사용하려면 `openpyxl` 라이브러리를 사전에 설치해야 함

## [ 예제 2-9 ] Excel 파일로 저장

IPython 콘솔에 데이터프레임의 내용이 출력된다.

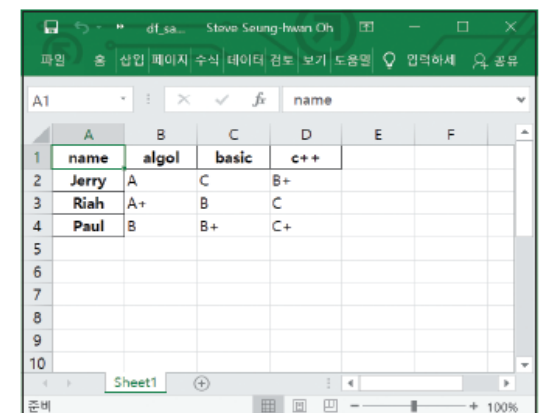
데이터프레임을 판다스 `to_excel()` 메소드를 이용하여 현재 디렉터리에 Excel 파일로 저장한다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df에 저장
6 data = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7         'algol' : [ "A", "A+", "B"],
8         'basic' : [ "C", "B", "B+"],
9         'c++' : [ "B+", "C", "C+"],
10        }
11
12 df = pd.DataFrame(data)
13 df.set_index('name', inplace=True) # name 열을 인덱스로 지정
14 print(df)
15
16 # to_excel() 메소드를 사용하여 Excel 파일로 내보내기. 파일명은 df_sample.xlsx로 저장
17 df.to_excel("./df_sample.xlsx")
```

〈실행 결과〉 코드 전부 실행 ① - IPython 콘솔에 출력되는 화면

	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

〈실행 결과〉 코드 전부 실행 ② - Excel 파일 내용 보기



### 4. 데이터 저장하기

#### ■ 여러 개의 데이터프레임을 하나의 Excel 파일로 저장

- 판다스 `ExcelWriter()` 함수는 Excel 워크북 객체를 생성함
- 데이터프레임에 `to_excel()` 메소드를 적용할 때, 삽입하려는 워크북 객체 (Excel 파일)를 인자로 전달함
- `sheet_name` 옵션에 Excel 파일의 시트 이름을 입력하여 삽입되는 시트 위치를 지정할 수 있음
- 데이터프레임을 삽입하는 시트 이름을 다르게 설정하면, 같은 Excel 파일의 서로 다른 시트에 여러 데이터프레임을 구분하여 저장함

데이터프레임 여러 개를 Excel 파일로 저장: `pandas.ExcelWriter( "파일 이름(경로)" )`



# 1. 데이터 입출력

## [ 예제 2-10 ] ExcelWriter() 활용

Print() 명령을 사용하여 두 데이터프레임 df1, df2 의 내용이 lpython 콘솔에 출력된다.

ExcelWriter() 함수로 생성한 워크북 객체를 writer 변수에 저장하고, "./df\_excelwriter.xlsx" 파일 경로에 Excel 파일로 저장된다.

폴더에 저장된 Excel 파일을 실행하여 열어보면 df1 (sheet1에 저장), df2 (sheet2에 저장)에 삽입된다.

```
3 import pandas as pd
4
5 # 판다스 DataFrame() 함수로 데이터프레임 변환. 변수 df1, df2에 저장
6 data1 = {'name' : [ 'Jerry', 'Riah', 'Paul'],
7          'algol' : [ "A", "A+", "B"],
8          'basic' : [ "C", "B", "B+"],
9          'c++' : [ "B+", "C", "C+" ]}
10
11 data2 = {'c0':[1,2,3],
12          'c1':[4,5,6],
13          'c2':[7,8,9],
14          'c3':[10,11,12],
15          'c4':[13,14,15]}
16
17 df1 = pd.DataFrame(data1)
18 df1.set_index('name', inplace=True) # name 열을 인덱스로 지정
19 print(df1)
20 print('\n')
21
22 df2 = pd.DataFrame(data2)
23 df2.set_index('c0', inplace=True) # c0 열을 인덱스로 지정
24 print(df2)
25
26 # df1을 'sheet1'으로, df2를 'sheet2'로 저장(Excel 파일명은 "df_excelwriter.xlsx")
27 writer = pd.ExcelWriter("./df_excelwriter.xlsx")
28 df1.to_excel(writer, sheet_name="sheet1")
29 df2.to_excel(writer, sheet_name="sheet2")
30 writer.save()
```

<실행 결과> 코드 전부 실행 ① - lPython 콘솔에 출력되는 화면

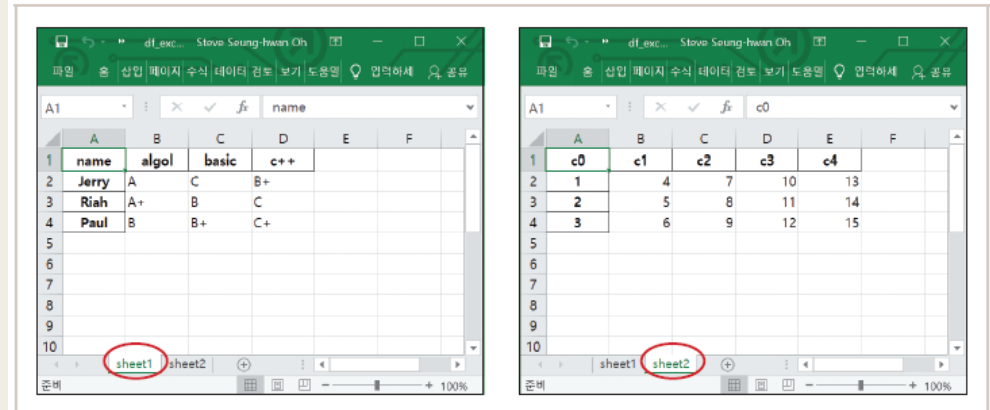
	algol	basic	c++
name			
Jerry	A	C	B+
Riah	A+	B	C
Paul	B	B+	C+

	c1	c2	c3	c4
c0				
1	4	7	10	13
2	5	8	11	14
3	6	9	12	15

<실행 결과> 코드 전부 실행 ② - Excel 파일 내용 보기

(File: example/part2/df\_excelwriter.xlsx)



감사합니다

「전공별 시활용」