

# 빅데이터 실습

5주차 1차시

데이터 입출력 [ 1 ] 파일에서 데이터 읽어오기



## 지난 시간에는

- 1/ Series와 Dataframe의 개념과 구조
- 2/ 색인과 통계



## ‘외부에서 제공해주는 데이터 사용하기’





## ‘외부에서 제공해주는 데이터 사용하기’

파일 다운로드를 통해  
제공된 데이터 사용

API<sup>1)</sup>를 통해 실시간으로  
제공되는 데이터 사용

웹 스크래핑을 통해 웹사이트의  
데이터를 직접 수집하여 사용

1) API(Application Programming Interface):  
응용 프로그램에서 사용할 수 있도록 프로그래밍 언어가 제공하는 기능을 제어할 수 있는 인터페이스

## 데이터 입출력 [ 1 ] 파일에서 데이터 읽어오기



### 학습개요

- 1/ csv 파일 데이터 입출력  
(read\_csv( ), to\_csv( ))
- 2/ Excel 파일 데이터 입출력  
(read\_excel( ), to\_excel( ))



01

# csv 파일 데이터 입출력





Run



Code



## 데이터 입출력

```
In [2]: import pandas as pd  
from pandas import Series, DataFrame
```

### 1. csv 파일 읽기 - read\_csv()

```
In [3]: # 1) 기본 csv 파일 읽기  
### data/ex1.csv 읽기 (컬럼명이 존재하는 csv 파일)
```

Out[3]:

# [read\_csv()] 기본 csv 파일 읽기

© data/ex1.csv 읽기

✓ data 디렉토리에 사용할 데이터들이 저장되어 있음

```
In [1]: import pandas  
from pandas
```

## 1. csv 파일

```
In [3]: # 1) 기본 csv  
### data/ex1.csv
```

Out [3]:

선수명	팀명	타율	안타	홈런	연봉
-----	----	----	----	----	----



# [read\_csv()] 기본 csv 파일 읽기

© data/ex1.csv 읽기

✓ ex1.csv를 워드패드로 열어보기

```
In [
a,b,c,d,message
1,2,3,4,hello
5,6,7,8,world
9,10,11,
```

CSV 형태 데이터

각각의 데이터들의 필드들이 즉 콤마로 구분되어 데이터가 저장된 걸 확인해 볼 수 있습니다.

# [read\_csv()] 기본 csv 파일 읽기

© data/ex1.csv 읽기

✓ read\_csv() : 첫 번째 인자(파일의 경로 지정)

✓ 상대경로로 지정

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

**절대경로**

첫 번째 위치 (최상위 경로)부터 작성된 경로

1. csv

**상대경로**

현재 파일 위치부터 작성된 상대적인 경로

```
In [3]: # 1) 기본 csv 파일 읽기  
### data/ex1.csv 읽기 (컬럼명이 존재하는 csv 파일)  
pd.read_csv('data/')
```

구분자의 기본값이 콤마(,)이므로 콤마를 기준으로 데이터들을 분리하여  
각각의 컬럼으로 저장된 것을 확인할 수 있습니다.

```
In [5]: # 2) sep 인자 활용하기
```



# [read\_csv()]

# sep 인자 활용하기

◎ 구분자가 다른 파일(data/ex2.txt) 읽기

✓ 워드패드로 열어보기

✓ 콤마가 아닌 다른 구분자인 경우 sep 인자 사용 가능

Out[2]:

	a	b	c	d	message
0	1	2	3	4	hello
1	5	6	7	8	world
2	9	10	11	12	foo

In [5]:

```
# 2) sep 인자 활용하기  
### 구분자  
|
```

**주의하기**

✓ 공백을 기준으로 데이터들의 필드가 구분되어 있는 것을 볼 수 있음

In [6]:

```
# 3) encoding 인자 활용하기
```

# [read\_csv()] sep 인자 활용하기

◎ 구분자가 다른 파일(data/ex2.txt) 읽기

✓ read\_csv('data/ex2.txt')

✓ 하나의 컬럼으로 인지됨

```
In [3]: # 2) sep 인자 활용하기
        ### 구분자가 다른 파일(data/ex2.txt) 읽기 (sep 인자)
        pd.read_csv('data/ex2.txt')
```

Out[3]:

	a	b	c	d	message
0		1	2	3	4 hello
1		5	6	7	8 world
2		9	10	11	12 foo



# [read\_csv()]

## sep 인자 활용하기

◎ 구분자가 다른 파일(data/ex2.txt) 읽기

✓ 각각을 columns으로 쪼개고 싶을 때는 인자값(sep)을 사용

Trusted

Python 3

0	1	2	3	4	hello
1	5	6	7	8	world
2	9	10	11	12	foo

```
In [4]: # 2) sep 인자 활용하기
#### 구분자가 다른 파일(data/ex2.txt) 읽기 (sep 인자)
pd.read_csv('data/ex2.txt', sep = ' ')
```

Out[4]:

	a	b	c	d	message
0	1	2	3	4	hello
1	5	6	7	8	world
2	9	10	11	12	foo

# [read\_csv()]

## sep 인자 활용하기

◎ 구분자가 다른 파일(data/ex2.txt) 읽기

✓ read\_csv() : 일반적인 텍스트 파일을 읽어 오는 것

### 주의하기

✓ 특정 구분자를 기준으로 필드들이 구분되어 있는 csv 형태의 파일은 모두 처리 가능하므로 파일의 확장자는 상관이 없음

```
In [4]: # 2) sep 인자 활용하기
      ### 구분자가 다른 파일(data/ex2.txt) 읽기 (sep 인자)
      pd.read_csv('data/ex2.txt', sep = ' ')
```

Out[4]:

	a	b	c	d	message
0	1	2	3	4	hello
1	5	6	7	8	world
2	9	10	11	12	foo



# [read\_csv()]

# encoding 인자 활용하기

© data/2020KB0야구.csv 파일 읽기

✓ 해당 파일을 바로 읽어보면 UnicodeDecodeError 발생

Trusted

Python 3

```
In [5]: # 3) encoding 인자 활용하기
        ### data/2020KB0야구.csv 파일 읽기
        pd.read_csv('data/2020KB0야구.csv', |
```

UnicodeDecodeError

Traceback (most recent call last)

<ipython-input-5-9aaef117aee8> in <module>

```
1 # 3) encoding 인자 활용하기
2 ### data/2020KB0야구.csv 파일 읽기
----> 3 pd.read_csv('data/2020KB0야구.csv', |
```

UnicodeDecodeError

문자열이 인코딩하는 방식과 맞지 않는 경우

C:\ProgramData\Anaconda3\lib\site-packages\pandas\io\parsers.py in read\_csv(filepath\_or\_buffer, sep, delimiter, header, names, index\_col, usecols, squeeze, prefix, mangle\_dupe\_cols, dtype, engine, converters, true\_values, false\_values, skipinitialspace, skiprows, skipfooter, nrows, na\_values, keep\_default\_na, na\_filter, verbose



# [read\_csv()]

# encoding 인자 활용하기

© data/2020KBO야구.csv 파일 읽기

- ✓ 윈도우 계열의 경우 cp949로 encoding
- ✓ 기본값 : utf8

```
In [6]: # 3) encoding 인자 활용하기
        ### data/2020KBO야구.csv 파일 읽기
        pd.read_csv('data/2020KBO야구.csv', encoding = 'cp949')
```

Out [6]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000.0
1	허경민	두산	0.332	145	7	48000.0
2	오재일	두산	0.312	147	16	47000.0
3	최주환	두산	0.306	156	16	27000.0
4	박건우	두산	0.304	148	14	45000.0



✓ columns을 기준으로 데이터가 나뉨

```
선수명,팀명,타율,안타,홈런,연봉  
페르난데스,두산,0.34,199,21,40000  
허경민,두산,0.332,145,7,48000  
오재일,두산,0.312,147,16,47000  
최주환,두산,0.306,156,16,27000  
박건우,두산,0.304,148,14,45000  
정수빈,두산,0.298,146,5,34000  
최용재,두산,0.295,13,0,2800  
김재호,두산,0.289,116,2,65000  
안권수,두산,0.27,10,0,2700  
박세혁,두산,0.269,97,4,23200  
김재환,두산,0.266,137,30,65000  
조수행,두산,0.263,10,0,4500  
권민석,두산,0.26,13,0,2700  
이유찬,두산,0.258,23,0,3200  
신성현,두산,0.25,1,0,4800  
장승현,두산,0.25,5,0,3500  
서예일,두산,0.24,6,0,3200  
국해성,두산,0.233,20,3,4500  
오재원,두산,0.232,36,5,30000  
임찬열,두산,0.227,5,0,2700  
김인태,두산,0.202,17,1,5000  
백동훈,두산,0.188,3,0,3600  
정상호,두산,0.163,14,0,7000  
강태을,롯데,0.455,5,2,2700  
신용수,롯데,0.429,3,0,2900  
손아섭,롯데,0.352,190,11,200000  
오윤석,롯데,0.298,50,4,4000
```

3      최주환   두산   0.306   156   16   27000.0

4      박건우   두산   0.304   148   14   45000.0

◎ 컬럼명이 포함되어 있지 않은 경우(데이터만 있는 경우)

✓ header나 names 인자를 사용할 수 있음



A screenshot of a text editor window showing a list of names and numerical data. The text is as follows:

```
베르난데스,두산,0.34,199,21,40000  
허경민,두산,0.332,145,7,48000  
오재일,두산,0.312,147,16,47000  
최주환,두산,0.306,156,16,27000  
박건우,두산,0.304,148,14,45000  
정수빈,두산,0.298,146,5,34000  
최용제,두산,0.295,13,0,2800  
김재호,두산,0.289,116,2,65000  
안권수,두산,0.27,10,0,2700  
박세혁,두산,0.269,97,4,23200  
김재환,두산,0.266,137,30,65000  
조수행,두산,0.263,10,0,4500  
권민석,두산,0.26,13,0,2700  
이유찬,두산,0.258,23,0,3200  
신성현,두산,0.25,1,0,4800  
장승현,두산,0.25,5,0,3500  
서예일,두산,0.24,6,0,3200  
국해성,두산,0.233,20,3,4500  
오재원,두산,0.232,36,5,30000  
양찬열,두산,0.227,5,0,2700  
김인태,두산,0.202,17,1,5000  
백동훈,두산,0.188,3,0,3600  
정상호,두산,0.163,14,0,7000  
강태용,롯데,0.455,5,2,2700  
신용수,롯데,0.429,3,0,2900  
손아섭,롯데,0.352,190,11,200000  
오윤석,롯데,0.298,50,4,4000  
정훈,롯데,0.295,121,11,6400
```

3 최주환 두산 0.306 156 16 27000.0

4 박건우 두산 0.304 148 14 45000.0



# [read\_csv()]

# header, names 인자 활용하기

◎ 컬럼명이 포함되어 있지 않은 경우

✓ 첫 번째 줄에 있는 데이터를 무조건 컬럼명으로 인지

259 rows × 6 columns

```
In [7]: # 4) header, names 인자 활용하기 (컬럼명이 파일에 포함되어 있지 않은 경우)
pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949')
```

Out[7]:

	페르난데스	두산	0.34	199	21	40000
0	허경민	두산	0.332	145	7	48000.0
1	오재일	두산	0.312	147	16	47000.0
2	최주환	두산	0.306	156	16	27000.0
3	박건우	두산	0.304	148	14	45000.0

# [read\_csv()]

## header, names 인자 활용하기

◎ header 인자 활용하기 : 컬럼명으로 사용할 라인 지정

- ✓ 기본값 : 0 (첫 번째 줄을 컬럼명으로 활용하라는 의미)
- ✓ None : 컬럼명이 없음을 의미. 즉, 첫 번째 줄부터 데이터로 읽기

259 rows × 6 columns

```
In [9]: # 4) header, names 인자 활용하기 (컬럼명이 파일에 포함되어 있지 않은 경우)
pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',
            header = None)
```

Out [9]:

	0	1	2	3	4	5
0	페르난데스	두산	0.340	199	21	40000.0
1	허경민	두산	0.332	145	7	48000.0
2	오재일	두산	0.312	147	16	47000.0
3	최주화	두산	0.306	156	16	27000.0



# [read\_csv()]

## header, names 인자 활용하기

◎ header 인자 활용하기 : 컬럼명으로 사용할 라인 지정

✓ 2개 줄이 컬럼명으로 사용하는 경우, 리스트 형태로 지정

Trusted

Python 3

259 rows × 6 columns

```
In [10]: # 4) header, names 인자 활용하기 (컬럼명이 파일에 포함되어 있지 않은 경우)
pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',
            header = [0,1])
```

Out [10]:

	페르난데스	두산	0.34	199	21	40000
	허경민	두산	0.332	145	7	48000
0	오재일	두산	0.312	147	16	47000.0
1	최주환	두산	0.306	156	16	27000.0
2	바거트	두산	0.304	148	14	45000.0

# [read\_csv()]

## header, names 인자 활용하기

◎ names 인자 활용하기 : 컬럼을 명시적으로 지정

✓ 기본적으로 컬럼명은 파일에 포함되어 있지 않기 때문에  
이 인자값을 컬럼명으로 지정하고, 파일의 첫 번째 줄부터 데이터로 읽음

```
In [11]: # 4) header, names 인자 활용하기 (컬럼명이 파일에 포함되어 있지 않은 경우)
pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',
            header = None)
```

...

```
In [12]: pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',
                    names = ['선수명', '티명', '타율', '안타', '홈런', '연봉'])
```

Out[12]:

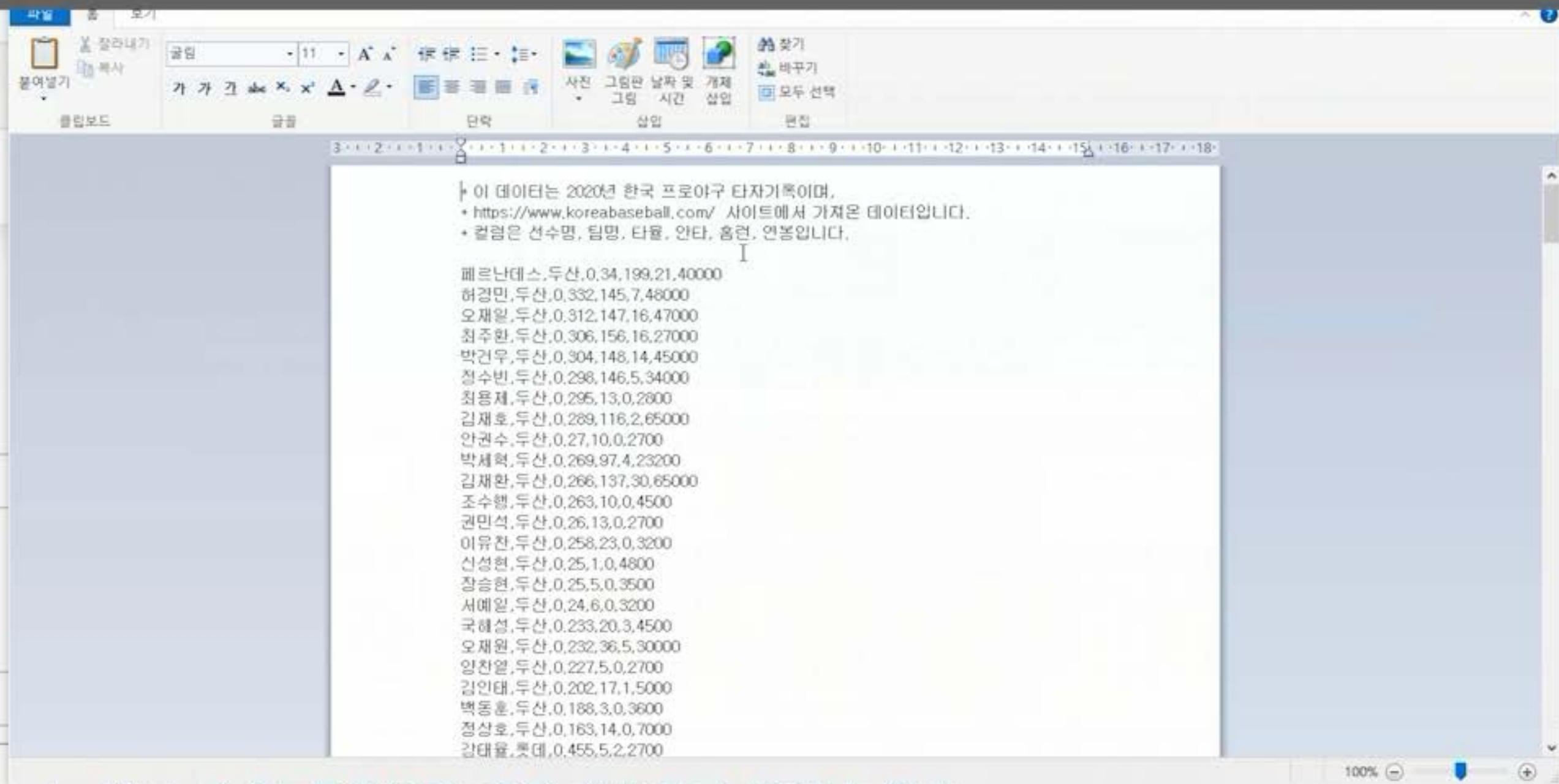
	선수명	티명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000.0
1	허경민	두산	0.332	145	7	48000.0



# [read\_csv()]

## 불필요한 줄은 제외하고 데이터 읽기

◎ 문서화 형태로 데이터의 설명을 적어둔 것



```
In [8]: # 5) 불필요한 줄은 제외하고 데이터 읽기  
# 5-1) skiprows를 이용하여, 특정 행을 읽지 않도록 함.
```

# [read\_csv()]

# 불필요한 줄은 제외하고 데이터 읽기

◎ 가장 쉬운 방법은 파일을 편집하는 것

✓ 네 줄을 지우고 데이터 읽어오기

이 데이터는 2020년 한국 프로야구 타자기록이며,  
\* <https://www.koreabaseball.com/> 사이트에서 가져온 데이터입니다.  
\* 컬럼은 선수명, 팀명, 타율, 안타, 홈런, 연봉입니다.

페르난데스,두산,0.34,199,21,40000  
허경민,두산,0.332,145,7,48000  
오재일,두산,0.312,147,16,47000  
최주환,두산,0.306,156,16,27000  
박건우,두산,0.304,148,14,45000  
정수빈,두산,0.298,146,5,34000  
최용재,두산,0.295,13,0,2800  
김재호,두산,0.289,116,2,65000  
안권수,두산,0.27,10,0,2700  
박세혁,두산,0.269,97,4,23200  
김재환,두산,0.266,137,30,65000  
조수행,두산,0.263,10,0,4500  
권민석,두산,0.26,13,0,2700  
이유찬,두산,0.258,23,0,3200  
산성현,두산,0.25,1,0,4800  
장승현,두산,0.25,5,0,3500  
서예일,두산,0.24,6,0,3200  
국해성,두산,0.233,20,3,4500  
오재원,두산,0.232,36,5,30000  
양찬열,두산,0.227,5,0,2700  
김인태,두산,0.202,17,1,5000  
백동훈,두산,0.188,3,0,3600  
정상호,두산,0.163,14,0,7000  
갈태을,롯데,0.455,5,2,2700

In [8]: # 5) 불필요한 줄은 제외하고 데이터 읽기  
# 5-1) skiprows를 이용하여, 특정 행을 읽지 않도록 함.



# [read\_csv()]

## 불필요한 줄은 제외하고 데이터 읽기

jupyter 데이터 입출력 (1) - csv, excel 파일 읽고 쓰기 (unsaved changes)



Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted



Python 3

Save New Cut Copy Paste Undo Redo Run Stop Restart Code

```
In [11]: # 4) header, names 인자 활용하기 (컬럼명이 파일에 포함되어 있지 않은 경우)
pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',
            header = None)
```

...

```
In [13]: pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',
                    names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'])
```

...

```
In [8]: # 5) 불필요한 줄은 제외하고 데이터 읽기
# 5-1) skiprows를 이용하여, 특정 행을 읽지 않도록 함.
pd.read_csv('data/2020KB0야구_주석포함.csv', encoding = 'cp949',
```



# [read\_csv()]

# 불필요한 줄은 제외하고 데이터 읽기

© skiprows를 이용하여, 특정 행을 읽지 않도록 함

File Edit View Insert Cell Kernel Widgets Help

Trusted

Python 3



Run



Code



```
In [13]: pd.read_csv('data/2020KB0야구_컬럼명미포함.csv', encoding = 'cp949',  
                    names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'])
```

...

```
In [16]: # 5) 불필요한 줄은 제외하고 데이터 읽기  
# 5-1) skiprows를 이용하여, 특정 행을 읽지 않도록 함.  
pd.read_csv('data/2020KB0야구_주석포함.csv', encoding = 'cp949',  
            skiprows = [0,1,2],  
            names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'])
```

Out [16]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000.0



# [read\_csv()]

## 불필요한 줄은 제외하고 데이터 읽기

© comment(주석) 인자를 이용하여 특정 문자로 시작되는 행은 데이터로 읽지 않고, 주석으로 처리

✓ 인자값(e.g. \*)로 시작되는 줄은 데이터에서 모두 제외

Trusted

Python 3

```
skiprows = [0, 1, 2],  
names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'])
```

```
In [17]: # 5-2) comment 인자를 이용하여, 주석은 데이터로 읽지 않음.  
pd.read_csv('data/2020KB0야구_주석포함.csv', encoding = 'cp949',  
            comment = '*',  
            names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'])
```

Out[17]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000.0
1	허경민	두산	0.332	145	7	48000.0
2	오재익	두산	0.312	147	16	47000.0

# [read\_csv()]

## 불필요한 줄은 제외하고 데이터 읽기

◎ 비어 있는 줄의 경우 기본적으로 데이터를 읽지 않도록 설정되어 있음

◎ 비어 있는 줄을 읽을 경우 : `skip_blank_lines = False`

```
pd.read_csv('data/2020KB0야구_주석포함.csv', encoding = 'cp949',  
            comment = '*',  
            names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'])
```

...

```
In [18]: pd.read_csv('data/2020KB0야구_주석포함.csv', encoding = 'cp949',  
                    comment = '*',  
                    names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'],  
                    skip_blank_lines = False)
```

Out[18]:

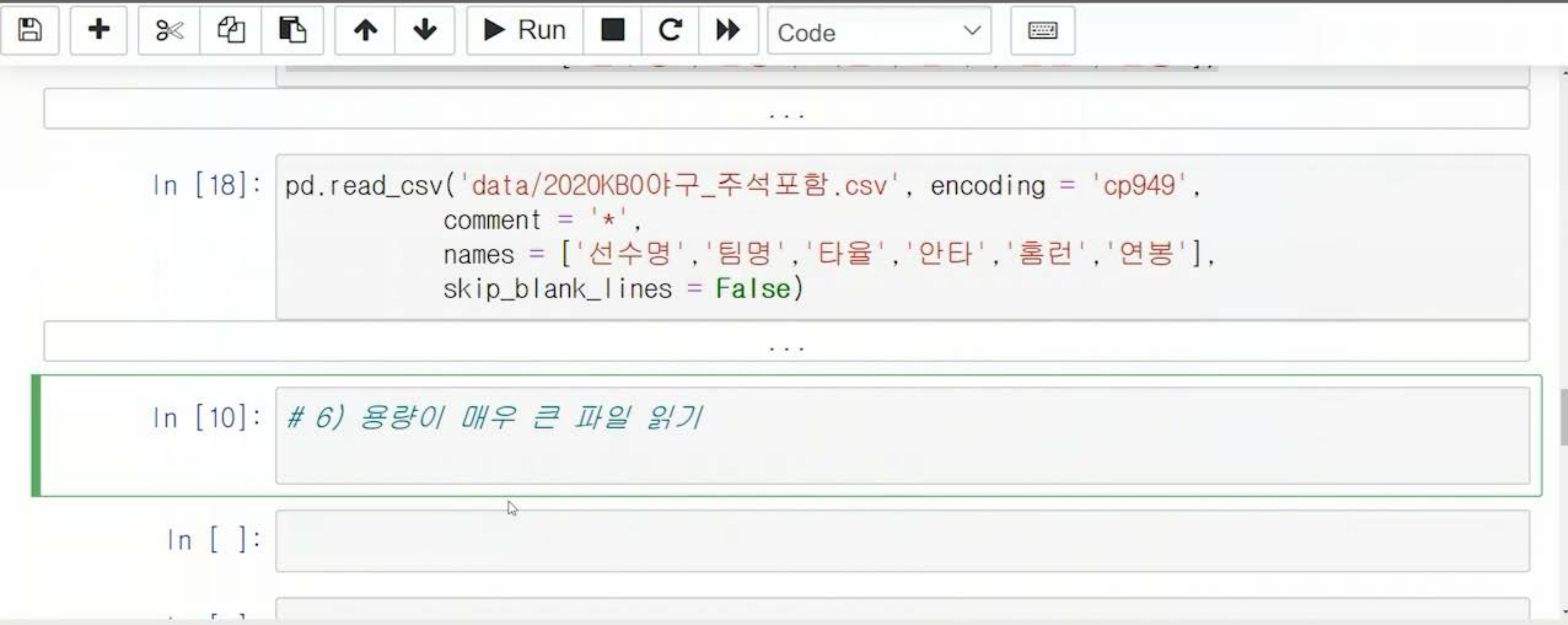
	선수명	팀명	타율	안타	홈런	연봉
0	NaN	NaN	NaN	NaN	NaN	NaN
1	페르난데스	두산	0.340	199.0	21.0	40000.0



# [read\_csv()]

## 용량이 매우 큰 파일 읽기

- ◎ 데이터를 한 번에 읽어서 데이터프레임으로 저장해 오는 것이 불가능할 수 있음
- ◎ 파일의 크기가 컴퓨터의 성능(가용 메모리 크기)보다 크면, Out Of Memory(OOM) Error 발생



The screenshot shows a Jupyter Notebook interface with a toolbar at the top containing icons for saving, adding, deleting, copying, pasting, undo, redo, running, and other standard editing functions. Below the toolbar, there are several code input cells. The first cell contains a code snippet for reading a CSV file. The second cell contains a comment in Korean. The third cell is empty.

```
In [18]: pd.read_csv('data/2020KB0야구_주석포함.csv', encoding = 'cp949',  
                    comment = '*',  
                    names = ['선수명', '팀명', '타율', '안타', '홈런', '연봉'],  
                    skip_blank_lines = False)
```

```
In [10]: # 6) 용량이 매우 큰 파일 읽기
```

```
In [ ]:
```



Run



Code



In [21]: # 6) 용량이 매우 큰 파일 읽기

```
pd.read_csv('data/2020KB00야구.csv', encoding = 'cp949', nrows = 5)
```

Out[21]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000
1	허경민	두산	0.332	145		
2	오재일	두산	0.312	147		
3	최주환	두산	0.306	156		
4	박건우	두산	0.304	148	14	45000

**주의하기**

✓ 기본적으로 앞에 있는 데이터만 읽어 올 수 있음

In [ ]:



# [read\_csv()]

# 용량이 매우 큰 파일 읽기

◎ chunksize

✓ chunksize=10으로 실행

Kernel

Widgets

Help

Trusted

Python 3

## 일반적인 read\_csv() 동작 방식

지금까지 실행해 본 read\_csv() 함수는 파일을 읽은 내용을 데이터프레임 형태로 반환해 주었음

## chunksize 인자

오재일	두산	0.312	147	16	47000
-----	----	-------	-----	----	-------

최준혁	두산	0.290	150	16	47000
-----	----	-------	-----	----	-------

파일을 읽을 준비가 되어 있는 텍스트 파일 리더(TextFileReader)라는 객체를 리턴

```
In [22]: pd.read_csv('data/2020KB00야구.csv', encoding = 'cp949', chunksize = 10)
```

```
Out[22]: <pandas.io.parsers.TextFileReader at 0x1f2151f1ac0>
```

```
In [ ]:
```

◎ chunksize

✓ chunksize의 인자값으로 리턴된 객체를 변수(커서)에 담아 놓음

✓ next(커서) : chunksize 개수만큼 데이터를 읽어 옴

1	허경민	두산	0.332	145	7	48000
2	오재일	두산	0.312	147	16	47000
3	최주환	두산	0.306	156	16	27000
4	박건우	두산	0.304	148	14	45000

```
In [23]: 커서 = pd.read_csv('data/2020KB00야구.csv', encoding = 'cp949', chunksize = 10)
```

```
In [ ]: next(커서)
```

```
In [ ]:
```



© chunksize

✓ 끊어서 읽어 온 데이터 저장하기

- ✓ 데이터목록 변수에 빈(empty) 리스트 할당
- ✓ 루프문에 커서를 활용할 수 있음
- ✓ next(커서)가 호출되면서 변수(data)에 저장
- ✓ 데이터목록 변수에 data를 append

```
In [27]: 데이터목록 = []  
for data in 커서:  
    데이터목록.append(data)
```

In [ ]:

# [to\_csv()]

## 결과를 csv 파일로 저장하기

### ◎ 데이터 가공하기

✓ 전체 데이터를 data에 저장

Kernel

Widgets

Help

Trusted

Python 3



Run



Code



In [ ]:

```
In [32]: # 7. 결과를 csv 파일로 저장하기 - to_csv()
data = pd.read_csv('data/2020KB00야구.csv', encoding = 'cp949')
data
```

Out[32]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000.0
1	허경민	두산	0.332	145	7	48000.0
2	오재일	두산	0.312	147	16	47000.0
3	최주환	두산	0.306	156	16	27000.0



# [to\_csv()]

## 결과를 csv 파일로 저장하기

### ◎ 팀 평균 타율 구하기

✓ pivot\_table()를 이용하여 result에 저장

```
In [32]: # 7. 결과를 csv 파일로 저장하기 - to_csv()
data = pd.read_csv('data/2020KB00야구.csv', encoding = 'cp949')
data.pivot_table(index = '팀명', values = '안타', aggfunc = 'sum')
```

Out[32]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000.0
1	허경민	두산	0.332	145	7	48000.0
2	오재일	두산	0.312	147	16	47000.0
3	최주환	두산	0.306	156	16	27000.0

# [to\_csv()] 결과를 csv 파일로 저장하기

◎ result를 csv 형태의 파일로 저장하기

✓ 팀별안타수.csv 파일 생성 확인

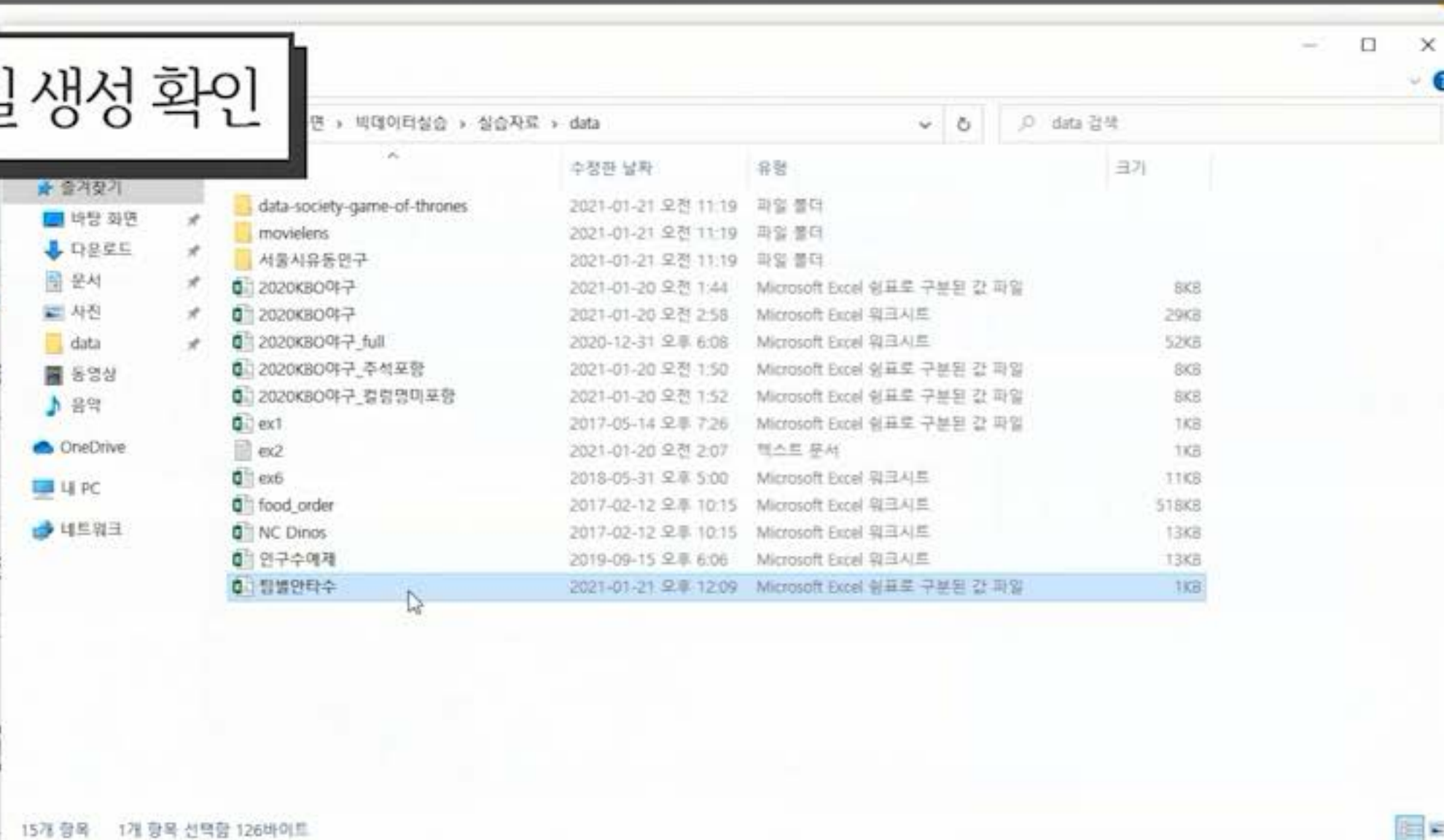
```
In [34]: # 7. 결과를  
data = pd.read_csv('data.csv')  
result = data.groupby('team').agg({'at_bat': 'sum', 'runs': 'sum', 'hits': 'sum', 'errors': 'sum', 'left_on_base': 'sum', 'strikeouts': 'sum', 'walks': 'sum', 'home_runs': 'sum', 'batting_average': 'sum', 'on_base_percentage': 'sum', 'slugging_percentage': 'sum', 'on_base_plus_slugging': 'sum', 'batting_average_plus_on_base_plus_slugging': 'sum'})
```

```
In [36]: result.to_csv('team_batting_stats.csv')
```

## 2. 엑셀 :

```
In [ ]:
```

```
In [ ]:
```





# [to\_csv()] 결과를 csv 파일로 저장하기

◎ result를 csv 형태의 파일로 저장하기

✓ 워드패드로 열어보면 한글이 깨짐

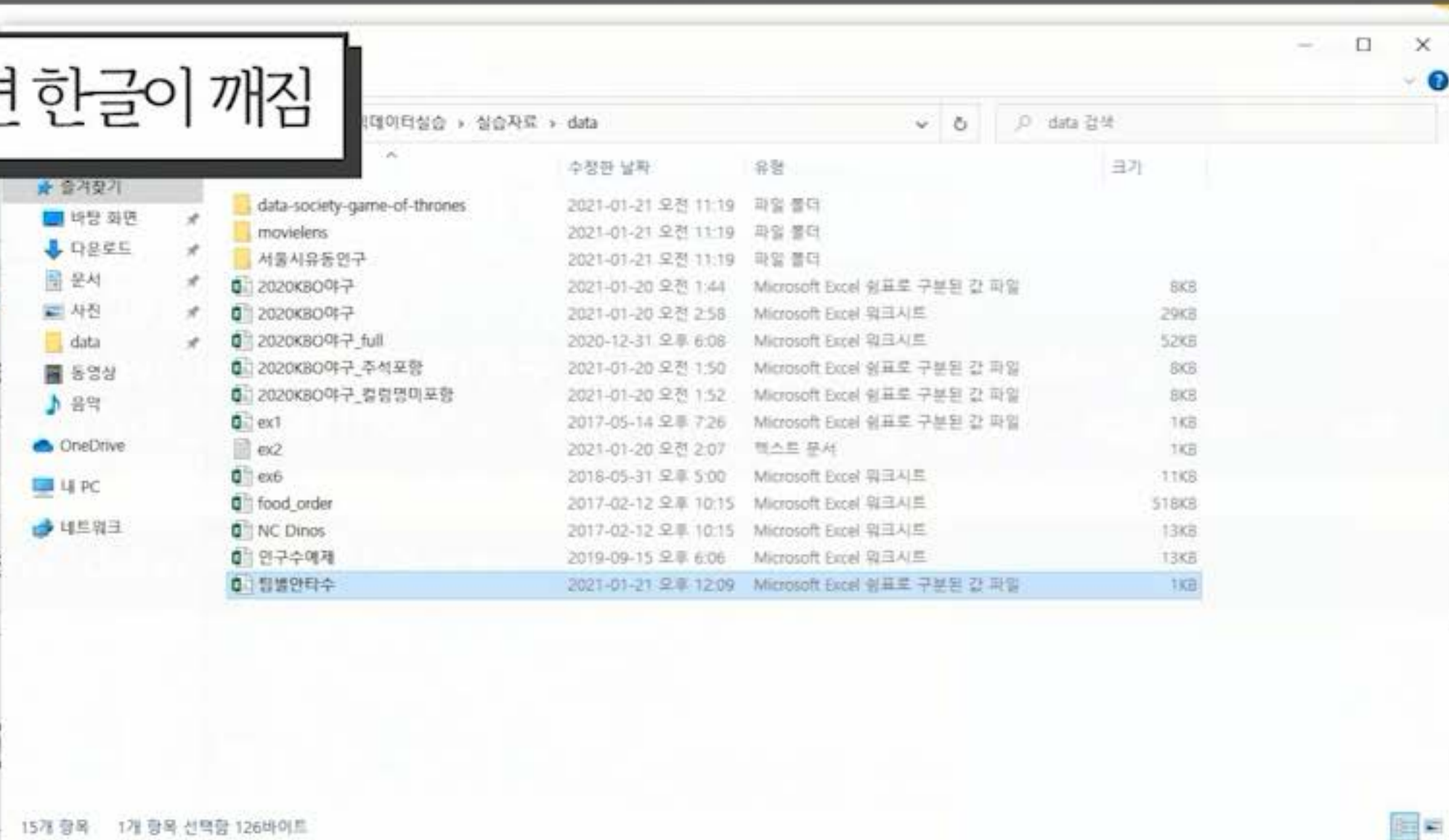
```
In [34]: # 7. 결과를  
data = pd.read_csv('data.csv')  
result = data
```

```
In [36]: result.to_csv('result.csv')
```

## 2. 엑셀 :

```
In [ ]:
```

```
In [ ]:
```



# [to\_csv()]

# 결과를 csv 파일로 저장하기

◎ result를 csv 형태의 파일로 저장하기

✓ 한글이 깨진 경우 encoding = 'cp949'

```
In [34]: # 7. 결과를 csv 파일로 저장하기 - to_csv()
data = pd.read_csv('data/2020KB00야구.csv', encoding = 'cp949')
result = data.pivot_table(index = '팀명', values = '안타', aggfunc = 'sum')
```

```
In [36]: result.to_csv('data/팀별안타수.csv', encoding = 'cp949')
```

## 2. 엑셀 파일 읽기 (read\_excel())

In [ ]:

In [ ]:



02

# Excel 파일 데이터 입출력



# read\_excel()

read\_csv( )에서  
제공하는 인자들과 흡사



## ◎ Excel로 열기

	A	B	C	D	E	F
1	선수명	팀명	타율	안타	홈런	연봉
2	김경호	SK	0.286	30	0	2900
3	김성민	SK	0.286	4	2	2700
4	고종욱	SK	0.283	77	3	17000
5	로맥	SK	0.282	137	32	90000
6	채태인	SK	0.281	45	7	10000
7	오타곤	SK	0.274	64	5	9500
8	김성현	SK	0.271	93	2	21000
9	오준혁	SK	0.27	40	3	3000
10	최정	SK	0.27	122	33	120000
11	최형	SK	0.265	35	2	7500
12	최지훈	SK	0.258	120	1	2700
13	김강민	SK	0.253	73	12	35000
14	유서준	SK	0.25	6	1	3100
15	한동민	SK	0.249	48	15	25000
16	박성현	SK	0.242	24	2	2700
17	정익윤	SK	0.241	45	1	30000
18	이흥련	SK	0.24	30	3	7000
19	정진기	SK	0.238	30	2	4600
20	최준우	SK	0.236	43	3	3100
21	남태혁	SK	0.197	13	1	3000
22	김창평	SK	0.192	15	0	3000
23	윤석민	SK	0.192	14	3	18000
24	이재원	SK	0.185	41	2	130000
25	이현석	SK	0.178	18	2	3000
26	정현	SK	0.152	17	2	3000
27	김재현	SK	0.143	2	0	5600
28	화이트	SK	0.136	3	1	
29	채현우	SK	0.13	3	0	3000
30	류효승	SK	0.125	1	1	2700
31	이거연	SK	0.125	1	0	2700
32						
33						
34						
35						

✓ 타자 기록 데이터가 팀별로 별도의 sheet에 저장되어 있음

## © read\_excel() 사용



```
In [39]: pd.read_excel('data/2020KB00야구.xlsx')
```

Out [39]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000
1	허경민	두산	0.332	145	7	48000
2	오재일	두산	0.312	147	16	47000
3	최주환	두산	0.306	156	16	27000
4	박건우	두산	0.304	148	14	45000
5	정수빈	두산	0.298	146	5	34000



◎ 첫 번째 sheet가 아닌 원하는 sheet를 읽어 오기

- ✓ sheet\_name = 0(기본값) : 첫 번째 sheet의 데이터를 읽어 오ム
- ✓ sheet\_name = 1 : 두 번째 sheet의 데이터를 읽어 오ム

```
In [39]: pd.read_excel('data/2020KB00야구.xlsx')
```

...

```
In [42]: pd.read_excel('data/2020KB00야구.xlsx', sheet_name = 2)
```

Out[42]:

	선수명	팀명	타율	안타	홈런	연봉
0	김기환	NC	0.500	1	0	2700
1	이재율	NC	0.444	4	0	3000
2	박민우	NC	0.345	161	8	52000
3	양의지	NC	0.328	151	33	200000

# [read\_excel()] Excel 파일 읽기

◎ 첫 번째 sheet가 아닌 원하는 sheet를 읽어 오기

✓ sheet\_name에 숫자 대신 sheet의 이름을 넣을 수도 있음

```
In [39]: pd.read_excel('data/2020KB00야구.xlsx')
```

...

```
In [43]: pd.read_excel('data/2020KB00야구.xlsx', sheet_name = '한화')
```

Out[43]:

	선수명	팀명	타율	안타	홈런	연봉
0	허관회	한화	0.500	1	0	2700
1	김현민	한화	0.400	2	0	2700
2	최승준	한화	0.333	1	0	4000
3	최재훈	한화	0.301	102	3	20000



◎ 첫 번째 sheet가 아닌 원하는 sheet를 읽어 오기

✓ sheet\_name에 리스트 형태로 여러 sheet를 읽어 올 수 있음

✓ keys = sheet의 이름, value = 해당 sheet의 데이터

```
In [44]: pd.read_excel('data/2020KB00야구.xlsx', sheet_name = ['LG', '두산'])
```

```
Out[44]: {'LG':   선수명  팀명   타율   안타   홈런   연봉
0   손호영  LG  0.367   11    0   2700
1   김현수  LG  0.331  181   22 130000
2   신민재  LG  0.308    8    0   5000
3   오지환  LG  0.300  158   10  60000
4   박용택  LG  0.300   65    2  80000
5   이형종  LG  0.296   85   17  20000
6   채은성  LG  0.293  122   15  32000
7   홍창기  LG  0.279  114    5   3800
8   라모스  LG  0.278  120   38  30000
9   김용의  LG  0.271   19    1  10500
```

## 주의하기

✓ 2개 이상의 시트 데이터를 읽어 올 때는 사전 타입(Dict)의 데이터로 리턴함

◎ 첫 번째 sheet가 아닌 원하는 sheet를 읽어 오기

✓ sheet\_name에 리스트 형태로 여러 sheet를 읽어 올 수 있음

```
In [45]: data = pd.read_excel('data/2020KB00야구.xlsx', sheet_name = ['LG', '두산'])
```

```
In [46]: type(data)
```

```
Out[46]: dict
```

```
In [47]: data.keys()
```

```
Out[47]: dict_keys(['LG', '두산'])
```

```
In [ ]:
```



# [read\_excel()]

## Excel 파일 읽기

◎ 첫 번째 sheet가 아닌 원하는 sheet를 읽어 오기

✓ sheet\_name에 리스트 형태로 여러 sheet를 읽어 올 수 있음

✓ sheet\_name에 sheet의 번호와 이름을 혼용하여 사용 가능

```
Out[47]: dict_keys(['LG', '두산'])
```

```
In [48]: data['LG']
```

...

```
In [49]: pd.read_excel('data/2020KB00야구.xlsx', sheet_name = [0, '한화'])
```

```
Out[49]: {0:      선수명  팀명      타율  안타  홈런      연봉
0  페르난데스  두산  0.340  199  21  40000
1   허경민  두산  0.332  145   7  48000
2   오재일  두산  0.312  147  16  47000
3   최주환  두산  0.306  156  16  27000
4   박건우  두산  0.304  148  14  45000
```

## ◎ 전체 sheet 읽어 오기

- ✓ sheet\_name에 리스트 형태로 모두 작성하거나 None으로 작성
- ✓ 2개 이상의 sheet이므로, 사전(Dict)형태

```
In [50]: pd.read_excel('data/2020KB00야구.xlsx', sheet_name = None)
```

```
Out[50]: {'두산':      선수명  팀명      타율  안타  홈런      연봉
0   페르난데스  두산  0.340  199  21  40000
1   허경민     두산  0.332  145   7  48000
2   오재일     두산  0.312  147  16  47000
3   최주환     두산  0.306  156  16  27000
4   박건우     두산  0.304  148  14  45000
5   정수빈     두산  0.298  146   5  34000
6   최용제     두산  0.295   13   0   2800
7   김재호     두산  0.289  116   2  65000
8   안권수     두산  0.270   10   0   2700
9   박세혁     두산  0.269   97   4  23200}
```



# [read\_excel()]

## Excel 파일 읽기

### ◎ 전체 sheet 읽어 오기

✓ 야구데이터에 모두 저장

Kernel

Widgets

Help

Trusted

Python 3



Run



Code



```
In [49]: pd.read_excel('data/2020KB00야구.xlsx', sheet_name = [0, '인원'])
```

...

```
In [51]: 야구데이터 = pd.read_excel('data/2020KB00야구.xlsx', sheet_name = None)
```

```
In [ ]:
```

## ◎ 전체 sheet 읽어 오기

✓ 야구데이터에는 10개의 구단 데이터가 쌓여 있음

```
In [51]: 야구데이터 = pd.read_excel('data/2020KB0야구.xlsx', sheet_name = None)
```

```
In [53]: 야구데이터['두산']
```

Out[53]:

	선수명	팀명	타율	안타	홈런	연봉
0	페르난데스	두산	0.340	199	21	40000
1	허경민	두산	0.332	145	7	48000
2	오재일	두산	0.312	147	16	47000
3	최주환	두산	0.306	156	16	27000
4	박건우	두산	0.304	148	14	45000



## ◎ 전체 sheet 읽어 오기

✓ 각 sheet의 데이터를 각각의 변수에 저장 가능

Help

Trusted

Python 3



Run



Code



```
In [51]: 야구데이터 = pd.read_excel('data/2020KB0야구.xlsx', sheet_name = None)
```

```
In [54]: LG = 야구데이터['LG']
```

Out[54]:

	선수명	팀명	타율	안타	홈런	연봉
0	손호영	LG	0.367	11	0	2700
1	김현수	LG	0.331	181	22	130000
2	신민재	LG	0.308	8	0	5000
3	오지환	LG	0.300	158	10	60000
4	박용택	LG	0.300	65	2	80000

## ◎ 전체 sheet 읽어 오기

✓ 순서대로 10개의 변수에 각각 데이터를 저장할 수 있음

Trusted

Python 3

```
In [51]: 야구데이터 = pd.read_excel('data/2020KB0야구.xlsx', sheet_name = None)
```

```
In [54]: 두산, SK, NC, LG, KT, KIA, 한화, 키움, 롯데, 삼성 = 야구데이터.values()
```

Out [54]:

	선수명	팀명	타율	안타	홈런	연봉
0	손호영	LG	0.367	11	0	2700
1	김현수	LG	0.331	181	22	130000
2	신민재	LG	0.308	8	0	5000
3	오지환	LG	0.300	158	10	60000
4	박용택	LG	0.300	65	2	80000



## ◎ 전체 sheet 읽어 오기

✓ 모든 sheet의 데이터를 읽어 올 수 있음

```
In [55]: 두산, SK, NC, LG, KT, KIA, 한화, 키움, 롯데, 삼성 = 야구데이터.values()
```

```
In [57]: 키움
```

Out[57]:

	선수명	팀명	타율	안타	홈런	연봉
0	임지열	키움	1.000	1	0	2700
1	김은성	키움	0.500	2	0	2900
2	이정후	키움	0.333	181	15	39000
3	이지영	키움	0.309	81	0	30000
4	김하성	키움	0.306	163	30	55000

# [to\_excel()]

## 결과를 Excel 파일로 저장하기

◎ 두산데이터.xlsx로 두산의 데이터프레임 저장

File

Edit

View

Insert

Cell

Kernel

Widgets

Help

Trusted

Python 3



Run



Code



```
In [55]: 두산, SK, NC, LG, KT, KIA, 한화, 키움, 롯데, 삼성 = 야구데이터.values()
```

```
In [58]: 두산.to_excel('data/두산데이터.xlsx')
```

AttributeError

Traceback (most recent call last)

<ipython-input-58-389665857c72> in <module>

----> 1 두산.to\_excel('data/두산데이터.xlsx')

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py in \_\_getattr\_\_(self, name)

5137

if self.\_info\_axis.\_can\_hold\_identifiers\_and\_holds\_name(name):

5138

return self[name]

-> 5139

return object.\_\_getattribute\_\_(self, name)

5140