16장 엑셀 파일 다루기

16.1 엑셀 파일을 읽고 쓰기

16.1.1 엑셀 파일의 데이터 읽기

■ pandas에서 엑셀 파일의 데이터를 읽어온다.
- 형식: df = pd.read_excel('excel_file.xlsx'[, sheet_name = number 혹은 '시트이름', index_col = number 혹은 '열이름'])

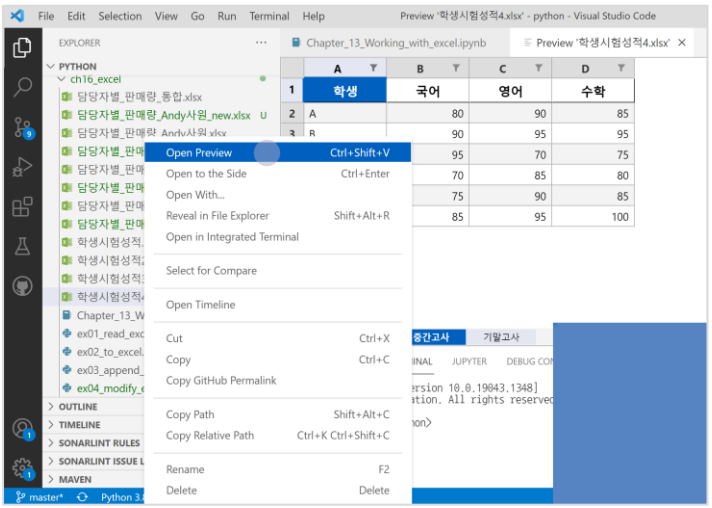
[ch16_excel/ex01_read_excel.py]

```
01  import pandas as pd
02  df = pd.read_excel('./ch16_excel/학생시험성적.xlsx')
03  print(df)
04  '''
05    학생  국어  영어  수학        평균
06  0  A  80  90  85  85.000000
07  1  B  90  95  95  93.333333
08  2  C  95  70  75  80.000000
09  3  D  70  85  80  78.333333
10  4  E  75  90  85  83.333333
11  '''
12
13  df = pd.read_excel('./ch16_excel/학생시험성적.xlsx', sheet_name=1)
14  print(df)
15  '''
16    학생  과학  사회   역사        평균
17  0  A  90  95   85  90.000000
18  1  B  85  90   80  85.000000
19  2  C  70  80   75  75.000000
20  3  D  75  90  100  88.333333
21  4  E  90  80   90  86.666667
22  '''
23
24  df = pd.read_excel('./ch16_excel/학생시험성적.xlsx', sheet_name='2차시험')
25  print(df)
26  '''
27    학생  과학  사회   역사        평균
28  0  A  90  95   85  90.000000
29  1  B  85  90   80  85.000000
30  2  C  70  80   75  75.000000
31  3  D  75  90  100  88.333333
32  4  E  90  80   90  86.666667
33  '''
34
35  df = pd.read_excel('./ch16_excel/학생시험성적.xlsx', sheet_name='2차시험', index_col=0)
36  print(df)
37  '''
38      과학  사회   역사        평균
39  학생
40  A   90  95   85  90.000000
41  B   85  90   80  85.000000
42  C   70  80   75  75.000000
43  D   75  90  100  88.333333
44  E   90  80   90  86.666667
45  '''
46
47  df = pd.read_excel('./ch16_excel/학생시험성적.xlsx',
48                     sheet_name='2차시험', index_col='학생')
49  print(df)
```

```
50  '''
51      과학  사회   역사        평균
52  학생
53  A   90  95   85  90.000000
54  B   85  90   80  85.000000
55  C   70  80   75  75.000000
56  D   75  90  100  88.333333
57  E   90  80   90  86.666667
58  '''
```

[꿀팁] Excel Viewer in vscode

- 참고: https://marketplace.visualstudio.com/items?itemName=GrapeCity.gc-excelviewer
- Excel Viewer를 설치한 후, excel 파일 > 오른쪽 마우스 클릭 > Open Previw 선택한다.



16.1.2 데이터를 엑셀 파일로 쓰기

- 엑셀 파일을 pandas의 DataFrame 데이터로 읽어오는 것은 간단하지만 DataFrame 데이터를 엑셀 파일로 쓰려면 다음과 같은 세 단계를 거쳐야 한다.

```
# 1. pandas의 ExcelWriter 객체 생성
excel_writer = pd.ExcelWriter('excel_output.xlsx', engine='xlswriter')

# 2. DataFrame 데이터를 지정된 엑셀 시트(sheet)에 쓰기
df1.to_excel(excel_writer[, index=True 혹은 False, sheet_name='시트이름1'])
df2.to_excel(excel_writer[, index=True 혹은 False, sheet_name='시트이름2'])

# 3. ExcelWriter 객체를 닫고, 지정된 엑셀 파일 생성
excel_writer.save()
```

- 다음으로 pandas의 DataFrame 데이터를 엑셀로 저장하는 예를 살펴본다.

[ch16_excel/ex02_to_excel.py]

```
01  import pandas as pd
```

```
02
03  excel_exam_data1 = {'학생': ['A', 'B', 'C', 'D', 'E', 'F'],
04                      '국어': [80, 90, 95, 70, 75, 85],
05                      '영어': [90, 95, 70, 85, 90, 95],
06                      '수학': [85, 95, 75, 80, 85, 100]}
07  df1 = pd.DataFrame(excel_exam_data1, columns=['학생', '국어', '영어', '수학'])
08  print(df1)
09  '''
10    학생  국어  영어   수학
11  0  A  80  90   85
12  1  B  90  95   95
13  2  C  95  70   75
14  3  D  70  85   80
15  4  E  75  90   85
16  5  F  85  95  100
17  '''
18
19  excel_writer = pd.ExcelWriter('./ch16_excel/학생시험성적2.xlsx', engine='xlsxwriter')
20  df1.to_excel(excel_writer, index=False)
21  excel_writer.save()
22
23  excel_writer2 = pd.ExcelWriter(
24      './ch16_excel/학생시험성적3.xlsx', engine='xlsxwriter')
25  df1.to_excel(excel_writer2, index=False, sheet_name='중간고사')
26  excel_writer2.save()
27
28  excel_exam_data2 = {'학생': ['A', 'B', 'C', 'D', 'E', 'F'],
29                      '국어': [85, 95, 75, 80, 85, 100],
30                      '영어': [80, 90, 95, 70, 75, 85],
31                      '수학': [90, 95, 70, 85, 90, 95]}
32  df2 = pd.DataFrame(excel_exam_data2, columns=['학생', '국어', '영어', '수학'])
33  print(df2)
34  '''
35    학생   국어  영어  수학
36  0  A   85  80  90
37  1  B   95  90  95
38  2  C   75  95  70
39  3  D   80  70  85
40  4  E   85  75  90
41  5  F  100  85  95
42  '''
43
44  excel_writer3 = pd.ExcelWriter(
45      './ch16_excel/학생시험성적4.xlsx', engine='xlsxwriter')
46  df1.to_excel(excel_writer3, index=False, sheet_name='중간고사')
47  df2.to_excel(excel_writer3, index=False, sheet_name='기말고사')
48  excel_writer3.save()
```

16.2 엑셀 파일 통합하기

16.2.1 효율적인 데이터 처리를 위한 엑셀 데이터 구조

■ 다음은 효율적인 데이터 처리를 위해 엑셀에서 데이터를 생성할 때 주의할 점이다.

- 열의 머리글(header)은 한 줄로만 만들고 데이터는 그 아래에 입력한다.
- 열 머리글이나 데이터 입력 부분에 셀 병합 기능은 이용하지 않는다.
- 데이터를 입력할 때 하나의 셀에 숫자와 단위를 같이 쓰지 않는다.
- 하나의 열에 입력한 값의 데이터 형식은 모두 일치해야 한다. 즉, 하나의 열에 문자열, 숫자, 날짜 등에 혼합해서 쓰지 않는다.

- 데이터를 연도, 분기, 월, 업체별, 제품별 등의 시트로 나누지 않는다. 즉, 가능하면 모든 데이터를 하나의 세트에 다 넣는다.

16.2.2 여러 개의 엑셀 파일 데이터를 통합하기

■ 여러 개의 엑셀 파일에서 데이터를 읽어와서 pandas의 DataFrame 데이터로 통합하는 방법을 알아본다.
- 76~83: 통합 결과를 엑셀 파일로 저장한다.

[ch16_excel/ex03_append_excel.py]

```
01  import glob
02  import pandas as pd
03  excel_data_files = ['./ch16_excel/담당자별_판매량_Andy사원.xlsx',
04                      './ch16_excel/담당자별_판매량_Becky사원.xlsx',
05                      './ch16_excel/담당자별_판매량_Chris사원.xlsx']
06
07  total_data = pd.DataFrame()
08
09  for f in excel_data_files:
10      df = pd.read_excel(f)
11      total_data = total_data.append(df)
12
13  print(total_data)
14  '''
15     제품명 담당자 지역  1분기  2분기  3분기  4분기
16  0   시계   A  가  198  123  120  137
17  1   구두   A  가  273  241  296  217
18  2  핸드백   A  가  385  316  355  331
19  0   시계   B  나  154  108  155  114
20  1   구두   B  나  200  223  213  202
21  2  핸드백   B  나  350  340  377  392
22  0   시계   C  다  168  102  149  174
23  1   구두   C  다  231  279  277  292
24  2  핸드백   C  다  365  383  308  323
25  '''
26
27  total_data = pd.DataFrame()
28
29  for f in excel_data_files:
30      df = pd.read_excel(f)
31      total_data = total_data.append(df, ignore_index=True)
32
33  print(total_data)
34  '''
35  제품명 담당자 지역  1분기  2분기  3분기  4분기
36  0   시계   A  가  198  123  120  137
37  1   구두   A  가  273  241  296  217
38  2  핸드백   A  가  385  316  355  331
39  3   시계   B  나  154  108  155  114
40  4   구두   B  나  200  223  213  202
41  5  핸드백   B  나  350  340  377  392
42  6   시계   C  다  168  102  149  174
43  7   구두   C  다  231  279  277  292
44  8  핸드백   C  다  365  383  308  323
45  '''
46
47
48  print(glob.glob("./ch16_excel/담당자별_판매량_*사원.xlsx"))
49  '''
50  ['./ch16_excel\\담당자별_판매량_Andy사원.xlsx', './ch16_excel\\담당자별_판매량_Becky사
```

```
원.xlsx', './ch16_excel\\담당자별_판매량_Chris사원.xlsx']
'''


excel_data_files1 = glob.glob("./ch16_excel/담당자별_판매량_*사원.xlsx")
total_data1 = pd.DataFrame()

for f in excel_data_files1:
    df = pd.read_excel(f)
    total_data1 = total_data1.append(df, ignore_index=True)

print(total_data1)
'''
제품명 담당자 지역  1분기  2분기  3분기  4분기
0   시계   A  가  198  123  120  137
1   구두   A  가  273  241  296  217
2  핸드백   A  가  385  316  355  331
3   시계   B  나  154  108  155  114
4   구두   B  나  200  223  213  202
5  핸드백   B  나  350  340  377  392
6   시계   C  다  168  102  149  174
7   구두   C  다  231  279  277  292
8  핸드백   C  다  365  383  308  323
'''

excel_file_name = './ch16_excel/담당자별_판매량_통합.xlsx'

excel_total_file_writer = pd.ExcelWriter(excel_file_name, engine='xlsxwriter')
total_data1.to_excel(excel_total_file_writer, index=False, sheet_name='담당자별_판매량_통합')
excel_total_file_writer.save()

print(glob.glob(excel_file_name))
# ['./ch16_excel/담당자별_판매량_통합.xlsx']
```

16.3 엑셀 파일로 읽어온 데이터 다루기

16.3.1 데이터를 추가하고 변경하기

- 엑셀 파일을 pandas로 읽은 후에 다음과 같은 방법으로 DataFrame 데이터에 값을 추가하거나 변경할 수 있다.

```
import pandas as pd

df = pd.read_excel('excel_file.xlsx')
df.loc[index_name, column_name] = value
```

- 앞에서 활용했던 '담당자별_판매량_Andy사원.xlsx' 엑셀 파일에서 특정 데이터의 값을 변경해 보자.

[ch16_excel/ex04_modify_excel.py]

```
import pandas as pd

df = pd.read_excel('./ch16_excel/담당자별_판매량_Andy사원.xlsx')
print(df)
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
```

```
0   시계   A  가  198  123  120  137
1   구두   A  가  273  241  296  217
2  핸드백   A  가  385  316  355  331
'''

df.loc[2, '4분기'] = 0
print(df)
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
0   시계   A  가  198  123  120  137
1   구두   A  가  273  241  296  217
2  핸드백   A  가  385  316  355    0
'''

df.loc[3, '제품명'] = '벨트'
df.loc[3, '담당자'] = 'A'
df.loc[3, '지역'] = '가'
df.loc[3, '1분기'] = 100
df.loc[3, '2분기'] = 150
df.loc[3, '3분기'] = 200
df.loc[3, '4분기'] = 250
print(df)
'''
   제품명 담당자 지역    1분기    2분기    3분기    4분기
0   시계   A  가  198.0  123.0  120.0  137.0
1   구두   A  가  273.0  241.0  296.0  217.0
2  핸드백   A  가  385.0  316.0  355.0    0.0
3   벨트   A  가  100.0  150.0  200.0  250.0
'''

df['담당자'] = 'Andy'
print(df)
'''
   제품명   담당자 지역    1분기    2분기    3분기    4분기
0   시계  Andy  가  198.0  123.0  120.0  137.0
1   구두  Andy  가  273.0  241.0  296.0  217.0
2  핸드백  Andy  가  385.0  316.0  355.0    0.0
3   벨트  Andy  가  100.0  150.0  200.0  250.0
'''

import glob
excel_file_name = './ch16_excel/담당자별_판매량_Andy사원_new.xlsx'

new_excel_file = pd.ExcelWriter(excel_file_name, engine='xlsxwriter')
df.to_excel(new_excel_file, index=False)
new_excel_file.save()

print(glob.glob(excel_file_name))
# ['./ch16_excel/담당자별_판매량_Andy사원_new.xlsx']
```

16.3.2 여러 개의 엑셀 파일에서 데이터 수정하기

■ 여러 개의 엑셀 파일에 대해 데이터를 수정한 후에 각각 다른 이름으로 저장한다.

[ch16_excel/ex05_glob_excel.py]

```
import glob
import re
import pandas as pd

# 원하는 문자열이 포함된 파일을 검색해 리스트를 할당한다.
```

```
excel_data_files1 = glob.glob("./ch16_excel/담당자별_판매량_*사원.xlsx")

# 리스트에 있는 엑셀 파일만큼 반복 수행한다.
for f in excel_data_files1:
    # 엑셀 파일에서 DataFrame 형식으로 데이터 가져온다.
    df = pd.read_excel(f)

    # 특정 열의 값을 변경한다.
    if(df.loc[1, '담당자'] == 'A'):
        df['담당자'] = 'Andy'
    elif(df.loc[1, '담당자'] == 'B'):
        df['담당자'] = 'Becky'
    elif(df.loc[1, '담당자'] == 'C'):
        df['담당자'] = 'Chris'

    # 엑셀 파일 이름에서 지정된 문자열 패턴을 찾아서 파일명을 변경한다.
    f_new = re.sub(".xlsx", "2.xlsx", f)
    print(f_new)

    # 수정된 데이터를 새로운 이름의 엑셀 파일로 저장한다.
    new_excel_file = pd.ExcelWriter(f_new, engine='xlsxwriter')
    df.to_excel(new_excel_file, index=False)
    new_excel_file.save()


print(glob.glob("./ch16_excel/담당자별_판매량_*사원?.xlsx"))
'''
['./ch16_excel\\담당자별_판매량_Andy사원2.xlsx',       './ch16_excel\\담당자별_판매량_Becky사원2.xlsx',
'./ch16_excel\\담당자별_판매량_Chris사원2.xlsx']
'''
```

16.3.3 엑셀의 필터 기능 수행하기

■ 파이썬을 이용해 엑셀의 필터 기능을 수행한다.

[ch16_excel/ex06_filter_excel.py]

```
import pandas as pd

df = pd.read_excel('./ch16_excel/담당자별_판매량_통합.xlsx')
print(df)
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
0   시계   A  가  198  123  120  137
1   구두   A  가  273  241  296  217
2  핸드백   A  가  385  316  355  331
3   시계   B  나  154  108  155  114
4   구두   B  나  200  223  213  202
5  핸드백   B  나  350  340  377  392
6   시계   C  다  168  102  149  174
7   구두   C  다  231  279  277  292
8  핸드백   C  다  365  383  308  323
'''

print(df['제품명'])
'''
0     시계
1     구두
2    핸드백
3     시계
4     구두
```

```
5    핸드백
6     시계
7     구두
8    핸드백
Name: 제품명, dtype: object
'''

handbag = df[df['제품명'] == '핸드백']
print(handbag)
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
2  핸드백   A  가  385  316  355  331
5  핸드백   B  나  350  340  377  392
8  핸드백   C  다  365  383  308  323
'''

handbag1 = df[df['제품명'].isin(['핸드백'])]
print(handbag1)
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
2  핸드백   A  가  385  316  355  331
5  핸드백   B  나  350  340  377  392
8  핸드백   C  다  365  383  308  323
'''

print(df[(df['제품명'] == '구두') | (df['제품명'] == '핸드백')])
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
1   구두   A  가  273  241  296  217
2  핸드백   A  가  385  316  355  331
4   구두   B  나  200  223  213  202
5  핸드백   B  나  350  340  377  392
7   구두   C  다  231  279  277  292
8  핸드백   C  다  365  383  308  323
'''

print(df[df['제품명'].isin(['구두', '핸드백'])])

print(df[(df['3분기'] >= 250)])
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
1   구두   A  가  273  241  296  217
2  핸드백   A  가  385  316  355  331
5  핸드백   B  나  350  340  377  392
7   구두   C  다  231  279  277  292
8  핸드백   C  다  365  383  308  323
'''

print(df[['제품명','1분기', '2분기']])
'''
   제품명  1분기  2분기
0   시계  198  123
1   구두  273  241
2  핸드백  385  316
3   시계  154  108
4   구두  200  223
5  핸드백  350  340
6   시계  168  102
7   구두  231  279
8  핸드백  365  383
'''

print(df.iloc[[0,2],:])
'''
   제품명 담당자 지역  1분기  2분기  3분기  4분기
```

```
90  0   시계   A  가  198  123  120  137
91  2  핸드백   A  가  385  316  355  331
92  '''
```

16.3.4 엑셀 데이터 계산하기

[ch16_excel/ex07_sum_excel.py]

```
01  # 행 데이터의 합계 구하기
02  import pandas as pd
03
04  df = pd.read_excel('./ch16_excel/담당자별_판매량_통합.xlsx')
05
06  handbag = df[(df['제품명'] == '핸드백')]
07  print(handbag)
08  '''
09     제품명 담당자 지역  1분기  2분기  3분기  4분기
10  2  핸드백   A  가  385  316  355  331
11  5  핸드백   B  나  350  340  377  392
12  8  핸드백   C  다  365  383  308  323
13  '''
14
15  handbag_sum = pd.DataFrame(handbag.sum(axis=1), columns=['연간판매량'])
16  print(handbag_sum)
17  '''
18     연간판매량
19  2   1387
20  5   1459
21  8   1379
22  '''
23
24  handbag_total = handbag.join(handbag_sum)
25  print(handbag_total)
26  '''
27     제품명 담당자 지역  1분기  2분기  3분기  4분기  연간판매량
28  2  핸드백   A  가  385  316  355  331   1387
29  5  핸드백   B  나  350  340  377  392   1459
30  8  핸드백   C  다  365  383  308  323   1379
31  '''
32
33  print(handbag_total.sort_values(by='연간판매량', ascending=True))
34  '''
35     제품명 담당자 지역  1분기  2분기  3분기  4분기  연간판매량
36  8  핸드백   C  다  365  383  308  323   1379
37  2  핸드백   A  가  385  316  355  331   1387
38  5  핸드백   B  나  350  340  377  392   1459
39  '''
40
41  print(handbag_total.sort_values(by='연간판매량', ascending=False))
42  '''
43     제품명 담당자 지역  1분기  2분기  3분기  4분기  연간판매량
44  5  핸드백   B  나  350  340  377  392   1459
45  2  핸드백   A  가  385  316  355  331   1387
46  8  핸드백   C  다  365  383  308  323   1379
47  '''
48
49
50  # 열 데이터의 합계 구하기
51  print(handbag_total.sum())
52  '''
53  제품명      핸드백핸드백핸드백
```

```
담당자            ABC
지역             가나다
1분기           1100
2분기           1039
3분기           1040
4분기           1046
연간판매량         4225
dtype: object
'''

handbag_sum2 = pd.DataFrame(handbag_total.sum(), columns=['합계'])
print(handbag_sum2)
'''
               합계
제품명    핸드백핸드백핸드백
담당자          ABC
지역           가나다
1분기         1100
2분기         1039
3분기         1040
4분기         1046
연간판매량       4225
'''

handbag_total2  = handbag_total.append(handbag_sum2.T)
print(handbag_total2)
'''
          제품명  담당자   지역   1분기   2분기   3분기   4분기 연간판
매량
2         핸드백    A    가   385   316   355   331  1387
5         핸드백    B    나   350   340   377   392  1459
8         핸드백    C    다   365   383   308   323  1379
합계  핸드백핸드백핸드백  ABC  가나다  1100  1039  1040  1046  4225
'''

handbag_total2.loc['합계', '제품명'] = '핸드백'
handbag_total2.loc['합계', '담당자'] = '전체'
handbag_total2.loc['합계', '지역'] = '전체'

print(handbag_total2)
'''
    제품명 담당자  지역   1분기   2분기   3분기   4분기 연간판매량
2   핸드백   A   가   385   316   355   331  1387
5   핸드백   B   나   350   340   377   392  1459
8   핸드백   C   다   365   383   308   323  1379
합계  핸드백  전체  전체  1100  1039  1040  1046  4225
'''


import pandas as pd

# 엑셀 파일을 pandas의 DataFrame 형식으로 읽어온다.
df = pd.read_excel('./ch16_excel/담당자별_판매량_통합.xlsx')

# 제품명 열에서 핸드백이 있는 행만 선택한다.
product_name = '핸드백'
handbag = df[(df['제품명']== product_name)]

# 행별로 합계를 구하고 마지막 열 다음에 추가한다.
handbag_sum = pd.DataFrame(handbag.sum(axis=1), columns = ['연간판매량'])
handbag_total = handbag.join(handbag_sum)

# 열별로 합해 분기별 합계와 연간판매량 합계를 구하고 마지막 행 다음에 추가한다.
handbag_sum2 = pd.DataFrame(handbag_total.sum(), columns=['합계'])
handbag_total2  = handbag_total.append(handbag_sum2.T)
```

```
119
120  # 지정된 항목의 문자열을 변경한다.
121  handbag_total2.loc['합계', '제품명'] = product_name
122  handbag_total2.loc['합계', '담당자'] = '전체'
123  handbag_total2.loc['합계', '지역'] = '전체'
124
125  # 결과를 확인한다.
126  print(handbag_total2)
127  '''
128      제품명 담당자  지역   1분기   2분기   3분기   4분기 연간판매량
129  2   핸드백   A   가   385   316   355   331  1387
130  5   핸드백   B   나   350   340   377   392  1459
131  8   핸드백   C   다   365   383   308   323  1379
132  합계  핸드백  전체  전체  1100  1039  1040  1046  4225
133  '''
```

16.4 엑셀 데이터의 시각화

16.4.1 그래프를 엑셀 파일에 넣기

■ 파이썬을 이용해 데이터와 그래프(이미지)를 엑셀 파일에 추가할 수 있다. 다음은 데이터와 이미지를 엑셀 파일에 넣는 방법이다.

```
# 1. pandas의 ExcelWriter 객체 생성
excel_writer = pd.ExcelWriter('excel_output.xlsx', engine='xlsxwriter')

# 2. DataFrame 데이터를 지정된 엑셀 시트에 쓰기
df.to_excel(excel_writer, index=False 혹은 True, sheet_name='시트이름')

# 3. ExcelWriter 객체에서 워크시트 객체 생성
worksheet = excel_writer.sheets['시트이름']

# 4. 워크시트에 차트가 들어갈 위치를 지정해 이미지 넣기
worksheet.insert_image('셀위치', image_file [,{ ' x_scale': x_scale_num, 'y_scale': y_scale_num}])
혹은
worksheet.insert_image(row_num, col_num, image_file [,{ ' x_scale': x_scale_num, 'y_scale': y_scale_num}])

# 5. ExcelWriter 객체를 닫고 엑셀 파일 출력
excel_writer.save()
```

■ 위의 방법으로 엑셀 파일에 그래프를 추가한다.

[ch16_excel/ex08_insert_image.py]

```
01  import matplotlib
02  import matplotlib.pyplot as plt
03  import pandas as pd
04
05  sales = {'시간': [9, 10, 11, 12, 13, 14, 15],
06           '제품1': [10, 15, 12, 11, 12, 14, 13],
07           '제품2': [9, 11, 14, 12, 13, 10, 12]}
08
09  df = pd.DataFrame(sales, index=sales['시간'], columns=['제품1', '제품2'])
10  df.index.name = '시간'  # index 라벨 추가
11
12  print(df)
13  '''
```

```
    제품1  제품2
시간
9    10    9
10   15   11
11   12   14
12   11   12
13   12   13
14   14   10
15   13   12
'''


matplotlib.rcParams['font.family'] = 'Malgun Gothic'  # '맑은 고딕'으로 설정
matplotlib.rcParams['axes.unicode_minus'] = False

product_plot = df.plot(grid=True, style=['-*', '-o'], title='시간대별 생산량')
product_plot.set_ylabel("생산량")

image_file = './ch16_excel/fig_for_excel1.png'  # 이미지 파일 경로 및 이름
plt.savefig(image_file, dpi=400)  # 그래프를 이미지 파일로 저장

plt.show()


import pandas as pd

# (1) pandas의 ExcelWriter 객체 생성
excel_file = './ch16_excel/data_image_to_excel.xlsx'
excel_writer = pd.ExcelWriter(excel_file, engine='xlsxwriter')

# (2) DataFrame 데이터를 지정된 엑셀 시트(Sheet)에 쓰기
df.to_excel(excel_writer, index=True, sheet_name='Sheet1')

# (3) ExcelWriter 객체에서 워크시트(worksheet) 객체 생성
worksheet = excel_writer.sheets['Sheet1']

# (4) 워크시트에 차트가 들어갈 위치를 지정해 이미지 넣기
worksheet.insert_image('D2', image_file, {'x_scale': 0.7, 'y_scale': 0.7})
# worksheet.insert_image(1, 3, image_file, {'x_scale': 0.7, 'y_scale': 0.7})

# (5) ExcelWriter 객체를 닫고 엑셀 파일 출력
excel_writer.save()
```

16.4.2 엑셀 차트 만들기

■ 파이썬으로 엑셀의 차트 기능을 수행할 수 있다. 이를 위한 단계는 다음과 같다.

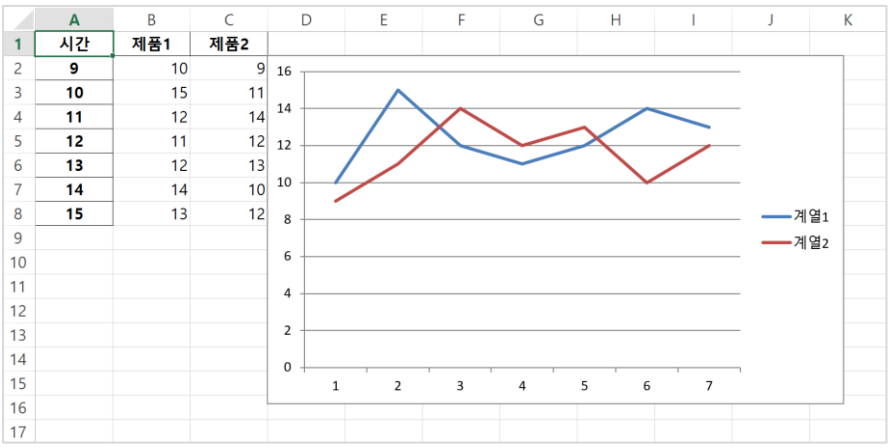
```
# 1. pandas의 ExcelWriter 객체 생성
excel_writer = pd.ExcelWriter('excel_output.xlsx', engine='xlsxwriter')

# 2. DataFrame 데이터를 지정된 엑셀 시트에 쓰기
df.to_excel(excel_writer, index=False 혹은 True, sheet_name='시트이름')

# 3. ExcelWriter 객체에서 워크북(workbook)과 워크시트(worksheet) 객체 생성
worksheet = excel_writer.book
worksheet = excel_writer.sheets['시트이름']

# 4. 차트 객체 생성(원하는 차트의 종류 지정)
chart = workbook.add_chart({'type': '차트유형'})
```

```
# 5. 차트를 생성하기 위한 데이터값의 범위 지정
chart.add_series({'values': values_range})

# 6. 워크시트에 차트가 들어갈 위치 지정해 차트 넣기
worksheet.insert_char('셀위치', chart)
혹은
worksheet.insert_chart(row_num,col_num, chart)

# 7. ExcelWriter 객체를 닫고 엑셀 파일 출력
excel_writer.save()
```

■ 파이썬에서 엑셀의 차트를 생성한다.

[ch16_excel/ex09_insert_chart1.py]

```
01  import matplotlib
02  import matplotlib.pyplot as plt
03  import pandas as pd
04
05  matplotlib.rcParams['font.family'] = 'Malgun Gothic'  # '맑은 고딕'으로 설정
06  matplotlib.rcParams['axes.unicode_minus'] = False
07
08  sales = {'시간': [9, 10, 11, 12, 13, 14, 15],
09           '제품1': [10, 15, 12, 11, 12, 14, 13],
10           '제품2': [9, 11, 14, 12, 13, 10, 12]}
11
12  df = pd.DataFrame(sales, index=sales['시간'], columns=['제품1', '제품2'])
13  df.index.name = '시간'  # index 라벨 추가
14
15  # (1) pandas의 ExcelWriter 객체 생성
16  excel_chart = pd.ExcelWriter(
17      './ch16_excel/data_chart_in_excel1.xlsx', engine='xlsxwriter')
18
19  # (2) DataFrame 데이터를 지정된 엑셀 시트(Sheet)에 쓰기
20  df.to_excel(excel_chart, index=True, sheet_name='Sheet1')
21
22  # (3) ExcelWriter 객체에서 워크북(workbook)과 워크시트(worksheet) 객체 생성
23  workbook = excel_chart.book
24  worksheet = excel_chart.sheets['Sheet1']
25
26  # (4) 차트 객체 생성(원하는 차트의 종류 지정)
27  chart = workbook.add_chart({'type': 'line'})
28
29  # (5) 차트 생성을 위한 데이터값의 범위 지정
30  chart.add_series({'values': '=Sheet1!$B$2:$B$8'})
31  chart.add_series({'values': '=Sheet1!$C$2:$C$8'})
32
33  # (6) 워크시트에 차트가 들어갈 위치를 지정해 차트 넣기
34  worksheet.insert_chart('D2', chart)
35
36  # (7) ExcelWriter 객체를 닫고 엑셀 파일 출력
37  excel_chart.save()
```



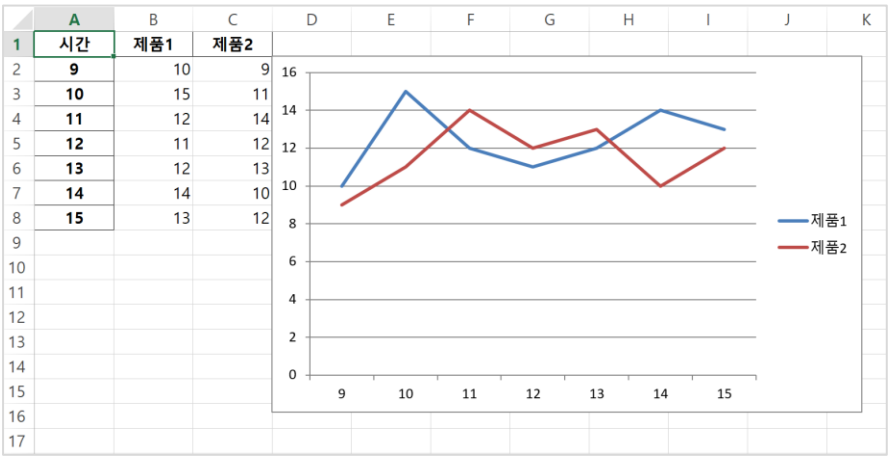
■ 출력된 엑셀 차트를 보면 '제품1'과 '제품2'의 생산량은 잘 표시돼나 x축의 값은 '시간'열이 아니라, 1부터 증가하는 숫자로 돼 있고 범례도 제대로 표시되지 않는다.

- 29~36: 차트에서 x축의 값을 원하는 열로 지정하고 범례를 제대로 표시하려면 add_series()에 인자 categories와 name을 추가하고 데이터값의 범위를 지정한다.

[ch16_excel/ex09_insert_chart2.py]

```
01  import matplotlib
02  import matplotlib.pyplot as plt
03  import pandas as pd
04
05  matplotlib.rcParams['font.family'] = 'Malgun Gothic'  # '맑은 고딕'으로 설정
06  matplotlib.rcParams['axes.unicode_minus'] = False
07
08  sales = {'시간': [9, 10, 11, 12, 13, 14, 15],
09           '제품1': [10, 15, 12, 11, 12, 14, 13],
10           '제품2': [9, 11, 14, 12, 13, 10, 12]}
11
12  df = pd.DataFrame(sales, index=sales['시간'], columns=['제품1', '제품2'])
13  df.index.name = '시간'  # index 라벨 추가
14
15  # (1) pandas의 ExcelWriter 객체 생성
16  excel_chart = pd.ExcelWriter(
17      './ch16_excel/data_chart_in_excel2.xlsx', engine='xlsxwriter')
18
19  # (2) DataFrame 데이터를 지정된 엑셀 시트(Sheet)에 쓰기
20  df.to_excel(excel_chart, index=True, sheet_name='Sheet1')
21
22  # (3) ExcelWriter 객체에서 워크북(workbook)과 워크시트(worksheet) 객체 생성
23  workbook = excel_chart.book
24  worksheet = excel_chart.sheets['Sheet1']
25
26  # (4) 차트 객체 생성(원하는 차트의 종류 지정)
27  chart = workbook.add_chart({'type': 'line'})
28
29  # (5) 차트 생성을 위한 데이터값의 범위 지정
30  chart.add_series({'values': '=Sheet1!$B$2:$B$8',
31                    'categories': '=Sheet1!$A$2:$A$8',
32                    'name': '=Sheet1!$B$1', })
33
34  chart.add_series({'values': '=Sheet1!$C$2:$C$8',
35                    'categories': '=Sheet1!$A$2:$A$8',
36                    'name': '=Sheet1!$C$1', })
37
38  # (6) 워크시트에 차트가 들어갈 위치를 지정해 차트 넣기
39  worksheet.insert_chart('D2', chart)
```

```
40
41  # (7) ExcelWriter 객체를 닫고 엑셀 파일 출력
42  excel_chart.save()
```



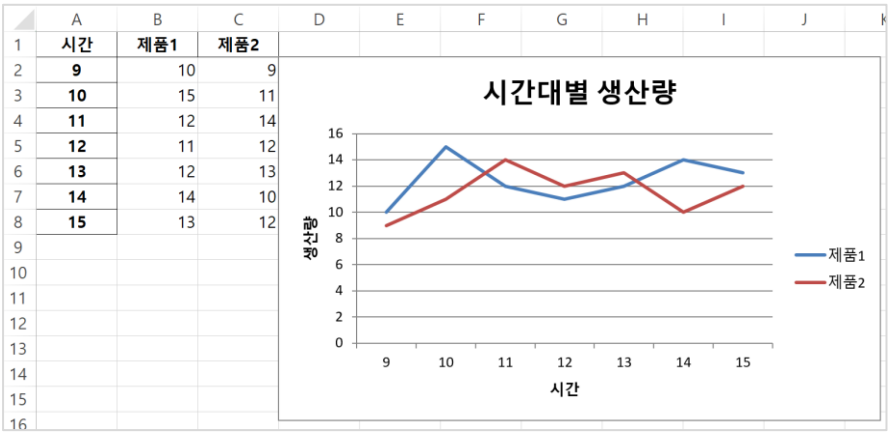
■ 현재까지 엑셀 차트를 보면 x축, y축 라벨이 없고 제목도 없다.

- 37~40: 엑셀 차트에 라벨과 제목을 추가한다.

[ch16_excel/ex09_insert_chart3.py]

```
01  import matplotlib
02  import matplotlib.pyplot as plt
03  import pandas as pd
04
05  matplotlib.rcParams['font.family'] = 'Malgun Gothic'  # '맑은 고딕'으로 설정
06  matplotlib.rcParams['axes.unicode_minus'] = False
07
08  sales = {'시간': [9, 10, 11, 12, 13, 14, 15],
09           '제품1': [10, 15, 12, 11, 12, 14, 13],
10           '제품2': [9, 11, 14, 12, 13, 10, 12]}
11
12  df = pd.DataFrame(sales, index=sales['시간'], columns=['제품1', '제품2'])
13  df.index.name = '시간'  # index 라벨 추가
14
15  # (1) pandas의 ExcelWriter 객체 생성
16  excel_chart = pd.ExcelWriter(
17      './ch16_excel/data_chart_in_excel3.xlsx', engine='xlsxwriter')
18
19  # (2) DataFrame 데이터를 지정된 엑셀 시트(Sheet)에 쓰기
20  df.to_excel(excel_chart, index=True, sheet_name='Sheet1')
21
22  # (3) ExcelWriter 객체에서 워크북(workbook)과 워크시트(worksheet) 객체 생성
23  workbook = excel_chart.book
24  worksheet = excel_chart.sheets['Sheet1']
25
26  # (4) 차트 객체 생성 (원하는 차트의 종류 지정)
27  chart = workbook.add_chart({'type': 'line'})
28
29  # (5) 차트 생성을 위한 데이터값의 범위 지정
30  chart.add_series({'values': '=Sheet1!$B$2:$B$8',
31                    'categories': '=Sheet1!$A$2:$A$8',
32                    'name': '=Sheet1!$B$1'})
33  chart.add_series({'values': '=Sheet1!$C$2:$C$8',
34                    'categories': '=Sheet1!$A$2:$A$8',
35                    'name': '=Sheet1!$C$1'})
36
```

```
# (5-1) 엑셀 차트에 x, y축 라벨과 제목 추가
chart.set_title({'name': '시간대별 생산량'})
chart.set_x_axis({'name': '시간'})
chart.set_y_axis({'name': '생산량'})

# (6) 워크시트에 차트가 들어갈 위치를 지정해 차트 넣기
worksheet.insert_chart('D2', chart)

# (7)  ExcelWriter 객체를 닫고 엑셀 파일 출력
excel_chart.save()
```



시간	제품1	제품2
9	10	9
10	15	11
11	12	14
12	11	12
13	12	13
14	14	10
15	13	12