
Excel
python

Extracting Data from Excel using Python

Speaker : Samuel Oranyeli
Blog: samukweku.github.io

ADVANTAGES OF CODING

- Separate data from analysis.
- Reproducible and easily shared.
- Version control.
- Automation.
- Flexibility in extracting data.

Resources Used:

Datasets:

Worked-examples.xlsx is from the unpivotr package in R.

EMT1626-Start.xlsx and 004-MSPTDA-ExcelFiles folder is from ExcelisFun.

Third-Party Packages:

- Pandas ... slice, dice and subdue tabular data
- Numpy ... efficient numerical computing
- OpenPyXl ... excels at Excel
- More-Itertools ... extends the Itertools library
- Pyjanitor ... extends Pandas' method chaining

AMAZING SPREADSHEETS IN THE WILD

Beck Frydenborg @beckfrydenborg · Mar 16

After spending way too long cleaning messy, hierarchically formatted excel data like this so that I could create a database from it, I encourage anyone who collects data to, for the love of the Flying Spaghetti Monster, read up on @hadleywickham's Tidy Data paper. #rstats

A	B	C	D	E	F	G	H	I	J
	Sample Type:	ground water		ground water		ground water		ground water	
	Site:	A		B		C		D	
Parameter	Units								
Nitrogen, Kjeldahl	mg/L	3.20		1.20		0.50		0.40	
Nitrate Nitrite as N	mg/L	0.025	U	0.025	U	0.025	U	0.025	U
Phosphorus as P	mg/L	0.040		0.170		0.062		.04	J3

6 10 29

Australian Bureau of Statistics

1800.0 Australian Marriage Law Postal Survey, 2017

Released on 15 November 2017

Table 1 Response by State and Territory

	Response clear						Eligible Participants							
	Yes		No		Total		Response clear		Response not clear(a)		Non-responding		Total	
	no.	%	no.	%	no.	%	no.	%	no.	%	no.	%	no.	%
New South Wales	2,374,362	57.8	1,736,838	42.2	4,111,200	100	4,111,200	79.2	11,036	0.2	1,065,445	20.5	5,187,681	100
Victoria	2,145,629	64.9	1,161,098	35.1	3,306,727	100	3,306,727	81.4	11,028	0.3	743,634	18.3	4,061,389	100
Queensland	1,487,060	60.7	961,015	39.3	2,448,075	100	2,448,075	77.7	7,088	0.2	695,710	22.1	3,150,873	100
South Australia	592,528	62.5	356,247	37.5	948,775	100	948,775	79.5	2,778	0.2	242,027	20.3	1,193,580	100
Western Australia	801,575	63.7	455,924	36.3	1,257,499	100	1,257,499	78.3	3,188	0.2	346,333	21.6	1,607,020	100
Tasmania	191,948	63.6	109,655	36.4	301,603	100	301,603	79.5	805	0.2	77,020	20.3	379,428	100
Northern Territory(b)	48,686	60.6	31,690	39.4	80,376	100	80,376	58.2	229	0.2	57,496	41.6	138,101	100
Australian Capital Territory(c)	175,459	74.0	61,520	26.0	236,979	100	236,979	82.3	534	0.2	50,595	17.6	288,108	100
Australia	7,817,247	61.6	4,873,987	38.4	12,691,234	100	12,691,234	79.3	36,686	0.2	3,278,260	20.5	16,006,180	100

(a) Includes blank responses

(b) Includes Christmas Island and the Cocos (Keeling) Islands (within the Division of Lingiari)

(c) Includes Jervis Bay (within the Division of Fenner) and Norfolk Island (within the Division of Canberra)

Table 3
Crime in the United States
by State, 2016

State	Area		Population	Violent crime[1]	Murder and nonnegligent manslaughter	Rape (revised definition[2])	Rape (legacy definition[3])	Robbery	Aggravated assault	Property crime	Burglary	Larceny-theft	Motor vehicle theft
ALABAMA	Metropolitan Statistical Area		3,716,889										
		Area actually reporting	99.2%	20,793	331	1,489	1,084	4,174	14,799	114,605	26,966	78,072	9,567
		Estimated total	100.0%	20,949	332	1,498	1,091	4,200	14,919	115,564	27,156	78,771	9,637
	Cities outside metropolitan areas		520,422										
		Area actually reporting	96.6%	3,551	41	263	187	410	2,837	19,841	4,173	14,332	1,336
		Estimated total	100.0%	3,646	42	269	193	421	2,914	20,291	4,275	14,641	1,375
	Nonmetropolitan counties		625,989										
		Area actually reporting	94.5%	1,220	31	141	95	61	987	7,096	2,490	3,934	672
		Estimated total	100.0%	1,291	33	149	101	65	1,044	7,507	2,634	4,162	711
	State Total		**4,863,300**	**25,886**	**407**	**1,916**	**1,385**	**4,686**	**18,877**	**143,362**	**34,065**	**97,574**	**11,723**
		Rate per 100,000 inhabitants		532.3	8.4	39.4	28.5	96.4	388.2	2,947.8	700.5	2,006.3	241.1
ALASKA	Metropolitan Statistical Area		350,298										
		Area actually reporting	100.0%	3,751	35	583	439	741	2,392	17,143	2,243	12,620	2,280
	Cities outside metropolitan areas		128,053										
		Area actually reporting	97.4%	964	2	165	119	71	726	4,069	655	3,101	313
		Estimated total	100.0%	989	2	169	123	73	745	4,177	672	3,184	321
	Nonmetropolitan counties		263,543										
		Area actually reporting	100.0%	1,226	1		301	195	36	874	3,556	1,138	1,962	456
	State Total		**741,894**	**5,966**	**5**		**1,053**	**757**	**850**	**4,011**	**24,876**	**4,053**	**17,766**	**3,057**
		Rate per 100,000 inhabitants		804.2	7.0	141.9	102.0	114.6	540.6	3,353.0	546.3	2,394.7	412.1
ARIZONA	Metropolitan Statistical Area		6,583,506										
		Area actually reporting	98.3%	27,522	334	2,997	2,238	6,931	17,260	196,398	35,424	143,602	17,372
		Estimated total	100.0%	27,789	338	3,025	2,258	6,971	17,455	198,543	35,844	145,152	17,547
	Cities outside metropolitan areas		124,210										
		Area actually reporting	100.0%	4,342	32	250	184	65	3,995	5,937	1,199	4,069	669
	Nonmetropolitan counties		223,355										
		Area actually reporting	100.0%	452	10	15	10	19	408	1,952	693	1,054	205
	State Total		**6,931,071**	**32,583**	**380**	**3,290**	**2,452**	**7,055**	**21,858**	**206,432**	**37,736**	**150,275**	**18,421**
		Rate per 100,000 inhabitants		470.1	5.5	47.5	35.4	101.8	315.4	2,978.4	544.4	2,168.1	265.8
ARKANSAS	Metropolitan Statistical Area		1,851,896										
		Area actually reporting	99.9%	11,573	144	1,315	967	1,711	8,403	68,739	15,650	47,839	5,250
		Estimated total	100.0%	11,576	144	1,316	968	1,711	8,405	68,762	15,655	47,856	5,251
	Cities outside metropolitan areas		515,688										
		Area actually reporting	94.1%	2,925	44	471	343	347	2,063	18,438	4,926	12,537	975

Key Guides:

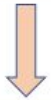
1. Location, relative to others
2. Manage null values
3. Reshape
4. Adapt to the whims of your data

Import Libraries

```
import pandas as pd
import numpy as np
import janitor
from more_itertools import windowed
from openpyxl import load_workbook
```

Read in Excel File

```
filename = 'worked-examples.xlsx'

df = pd.read_excel(filename)

df.head()
```



	Name	Age
0	Matilda	1
1	Nicholas	3
2	Olivia	5

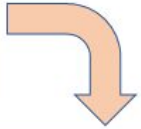
Read in a Specific Sheet

```
df = pd.read_excel('worked-examples.xlsx',
                   sheet_name = 'highlights')
df.head()
```

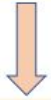
	Age	Height
0	1	2
1	3	4
2	5	6

Read in all the sheets

```
filename = 'EMT1626-Start.xlsx'

all_sheets = pd.read_excel(filename,sheet_name=None)

all_sheets
```



```
{'Data(1)':         Date   Sales   Product
 0 2020-12-01   14.35      Quad
 1 2020-12-02  144.42  Sunshine
 2 2020-12-04  207.00      Quad
 3 2020-12-01  247.33  Sunshine,
 'Data(2)':         Date   Sales   Product
 0 2020-12-01  179.09   Carlota
 1 2020-12-01  161.99   Carlota
 2 2020-12-04  172.46  Sunshine
 3 2020-12-04   64.03      Quad,
 'Data(3)':         Date   Sales  Product
 0 2020-12-02  209.38     Quad
 1 2020-12-04  203.27     Quad
 2 2020-12-04   12.00  Carlota
 3 2020-12-03  112.90  Carlota,
 'Report': Empty DataFrame
 Columns: [Unnamed: 0, Unnamed: 1, Unnamed: 2, Unnamed: 3, Unnamed: 4, Unnamed: 5, Unnamed: 6, Unnamed: 7,
Unnamed: 8, Unnamed: 9, Unnamed: 10, Unnamed: 11, Unnamed: 12, FilePath]
 Index: []}
```



```
combo = [data.assign(sheet=sheetname)
         for sheetname, data
         in all_sheets.items()
        ]

output = (pd.concat(combo,ignore_index=True)
          .dropna(how='all',axis=1))
```

	Date	Sales	Product	sheet
0	2020-12-01	14.35	Quad	Data(1)
1	2020-12-02	144.42	Sunshine	Data(1)
2	2020-12-04	207.00	Quad	Data(1)
3	2020-12-01	247.33	Sunshine	Data(1)
4	2020-12-01	179.09	Carlota	Data(2)
5	2020-12-01	161.99	Carlota	Data(2)
6	2020-12-04	172.46	Sunshine	Data(2)
7	2020-12-04	64.03	Quad	Data(2)
8	2020-12-02	209.38	Quad	Data(3)
9	2020-12-04	203.27	Quad	Data(3)
10	2020-12-04	12.00	Carlota	Data(3)
11	2020-12-03	112.90	Carlota	Data(3)

	A	B	C	D	E	F
1	Date	Product	Units	Sales		
2	8/11/2017	Aussie Round	216	7108.56		
3	14/11/2017	Tri Fly	84	551.88		
4	18/12/2017	Tri Fly	276	1443.48		
5	18/11/2017	Bellen	48	1362.72		
6	28/12/2016	Bellen	72	2027.52		
7	11/11/2017	Sunshine	96	1888.32		
8	12/09/2017	Aussie Round	60	1942.8		
9	23/11/2016	Aspen	144	2655.36		
10	28/11/2017	Aussie Round	204	6587.16		
11	18/12/2016	Tri Fly	72	484.56		
12	7/11/2017	Tri Fly	48	318.24		
13	4/11/2017	Aspen	96	1728.96		
14	24/10/2017	Carlota	72	2020.32		
15	12/02/2016	Aspen	96	1593.6		
16	7/12/2017	Tri Fly	264	1172.16		
17	27/11/2017	Quad	72	3107.52		
18	20/11/2017	Carlota	12	347.76		
19	2/12/2017	Aspen	72	1326.24		
20	1/11/2016	Quad	72	3053.52		
21	18/10/2017	Aspen	132	2195.16		
22	18/12/2016	Quad	108	4453.92		
23	19/12/2017	Aussie Round	72	2298.96		
24	7/12/2016	Sunshine	240	5068.8		
25	15/12/2017	Aussie Round	84	2651.04		
26	25/12/2016	Majestic Beaut	240	7994.4		
27	19/12/2016	Carlota	84	2561.16		
28	31/05/2017	Tri Fly	24	103.44		
29	28/11/2017	Carlota	108	3206.52		
30	12/12/2016	Sunshine	120	2565.6		
31	24/11/2016	Bellen	24	723.6		
32	24/11/2016	Quad	204	8253.84		
33	18/12/2016	Tri Fly	72	446.4		
34	16/12/2017	Bellen	72	2108.16		
35	16/04/2016	Sunshine	96	2187.84		
36	7/09/2016	Sunshine	84	1759.8		
37	30/11/2017	Bellen	72	2126.88		
38	4/11/2017	Majestic Beaut	60	2214.6		

Fran Gab Popi Sheet1

IMPORT MULTIPLE SHEETS FROM MULTIPLE FILES

READ IN MULTIPLE SHEETS FROM MULTIPLE FILES

```
from pathlib import Path
from fnmatch import fnmatch
import pandas as pd

folder = Path('004-MSPTDA-ExcelFiles')

all_files = list(folder.iterdir())

excel_only_files = [x for x in folder.iterdir()
                    if fnmatch(x.name.lower(),'*.xls*')
                   ]
```

ALL FILES

```
Out[3]:
[PosixPath('004-MSPTDA-ExcelFiles/Tacoma.XLSX'),
 PosixPath('004-MSPTDA-ExcelFiles/Seattle.xlsx'),
 PosixPath('004-MSPTDA-ExcelFiles/Oakland.csv'),
 PosixPath('004-MSPTDA-ExcelFiles/SanFrancisco.XLSM'),
 PosixPath('004-MSPTDA-ExcelFiles/Prices.accdb'),
 PosixPath('004-MSPTDA-ExcelFiles/Oakland.txt'),
 PosixPath('004-MSPTDA-ExcelFiles/Portland.xlsm'),
 PosixPath('004-MSPTDA-ExcelFiles/Oakland.xlsx')]
```

EXCEL ONLY FILES

```
Out[5]:
[PosixPath('004-MSPTDA-ExcelFiles/Tacoma.XLSX'),
 PosixPath('004-MSPTDA-ExcelFiles/Seattle.xlsx'),
 PosixPath('004-MSPTDA-ExcelFiles/SanFrancisco.XLSM'),
 PosixPath('004-MSPTDA-ExcelFiles/Portland.xlsm'),
 PosixPath('004-MSPTDA-ExcelFiles/Oakland.xlsx')]
```

HELPER FUNCTION

```
def process_files(xls_file):
    '''
    Picks an excel file,
    extracts the city name using the stem method from Path,
    filters out any sheet name that starts with Sheet,
    reads in the excel file using pandas, with the specified sheet names,
    concatenates the resulting dictionary,
    assigns city back to the dataframe
    and returns a dataframe
    '''

    xls = pd.ExcelFile(xls_file)

    city = xls_file.stem

    sheet_list = [sheet for sheet in xls.sheet_names
                  if not sheet.startswith('Sheet')]

    df = pd.read_excel(xls,sheet_list)

    outcome = pd.concat([data.assign(SalesPerson = SalesPerson)
                         for SalesPerson, data in df.items()],
                        ignore_index=True
                       ).assign(City = city)

    return outcome
```

```
combo = [process_files(xls_file) for xls_file in excel_only_files]
```

```
outcome = pd.concat(combo, ignore_index=True)
```

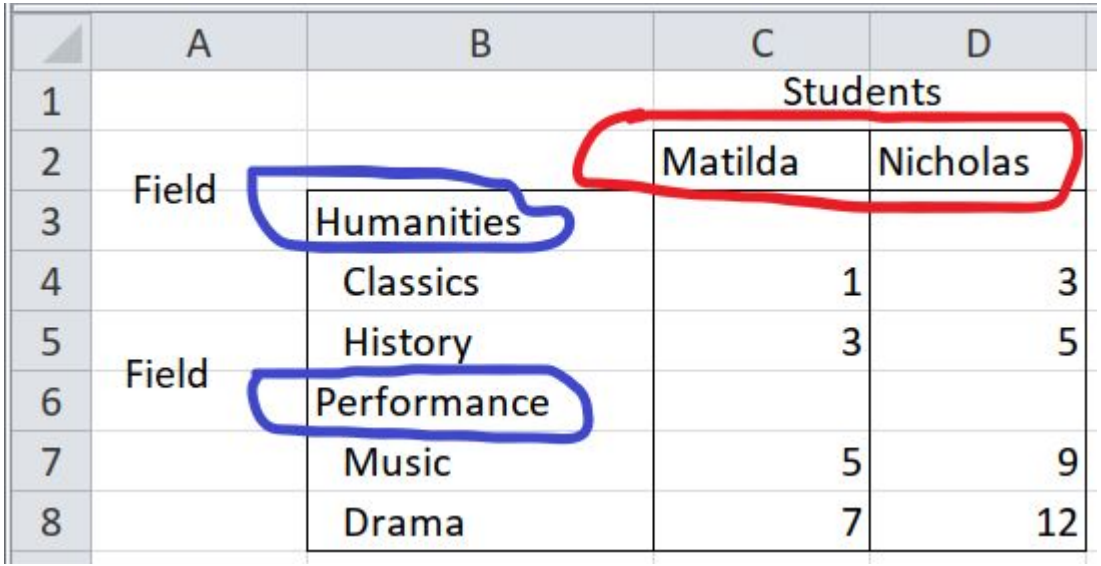
```
           Date         Product  Units    Sales SalesPerson      City
0    2016-12-30         Tri Fly     36   194.40       Sioux  Portland
1    2016-11-24         Carlota    108  3447.36       Sioux  Portland
2    2017-12-16         Tri Fly     84   345.24       Sioux  Portland
3    2016-11-24         Tri Fly     36   153.00       Sioux  Portland
4    2017-03-18            Quad     60  2455.80       Sioux  Portland
...         ...             ...    ...     ...         ...       ...
4018 2016-12-18  Majestic Beaut    264  8841.36      Tyrone  Portland
4019 2017-11-19        Sunshine    264  5739.36      Tyrone  Portland
4020 2016-11-15         Tri Fly    120   513.60      Tyrone  Portland
4021 2016-02-28        Sunshine     60  1355.40      Tyrone  Portland
4022 2016-11-15  Majestic Beaut     96  3486.72      Tyrone  Portland

[11326 rows x 6 columns],
           Date       Product  Units    Sales SalesPerson     City
0    2017-11-08  Aussie Round    216  7108.56        Fran  Oakland
1    2017-11-14       Tri Fly     84   551.88        Fran  Oakland
2    2017-12-18       Tri Fly    276  1443.48        Fran  Oakland
3    2017-11-18        Bellen     48  1362.72        Fran  Oakland
4    2016-12-28        Bellen     72  2027.52        Fran  Oakland
...         ...           ...    ...      ...         ...      ...
3683 2017-12-26          Quad     48  2074.08        Popi  Oakland
3684 2017-08-01      Sunshine     96  1855.68        Popi  Oakland
3685 2016-12-12        Bellen     60  1715.40        Popi  Oakland
3686 2016-12-24        Bellen     48  1377.60        Popi  Oakland
3687 2017-09-15      Sunshine    180  3465.00        Popi  Oakland

[11033 rows x 6 columns]]
```

```
In [12]: outcome.sample(20)
Out[12]:
           Date         Product  Units    Sales SalesPerson          City
32487 2016-12-31         Carlota     72  2229.12        Chin      Portland
14578 2017-11-11       Sunshine      60  1169.40          Mo       Seattle
8516  2016-12-12          Aspen     48   809.28      Shelia        Tacoma
47566 2017-12-09  Majestic Beaut    228  8219.40        Popi       Oakland
20907 2017-12-06   Aussie Round     240  7893.60        Miki  SanFrancisco
26440 2017-12-25         Carlota     24   669.36       Alden  SanFrancisco
148   2016-11-24  Majestic Beaut    240  8385.60         Sue        Tacoma
30023 2016-11-25         Tri Fly    228  1411.32       Sioux      Portland
42773 2017-12-30   Aussie Round     276  8564.28        Fran       Oakland
24066 2016-08-09   Aussie Round     264  8413.68         Han  SanFrancisco
47051 2017-11-12         Carlota     96  2982.72         Gab       Oakland
35681 2016-12-26          Bellen     12   365.76        Chin      Portland
11017 2016-12-06  Majestic Beaut    252  8998.92       Sindy       Seattle
6286  2016-08-11          Bellen     48  1480.32      Shelia        Tacoma
33569 2016-03-20         Tri Fly    264  1356.96        Chin      Portland
5633  2017-11-05   Aussie Round     60   1882.20      Shelia        Tacoma
9424  2016-04-17   Aussie Round     48   1507.20       Sindy       Seattle
39007 2017-06-21       Sunshine     60  1198.20       Tyrone      Portland
19254 2017-11-12         Tri Fly    156   703.56         Gigi       Seattle
41425 2017-04-11            Quad     48  2162.40        Fran       Oakland
```

PIVOT TABLE - SINGLE HEADER

	A	B	C	D
1				
2			Matilda	Nicholas
3		Humanities		
4		Classics	1	3
5		History	3	5
6		Performance		
7		Music	5	9
8		Drama	7	12

Courtesy: Nacnudus



	A	B	C	D
1			Students	
2			Matilda	Nicholas
3	Field	Humanities		
4		Classics	1	3
5		History	3	5
6	Field	Performance		
7		Music	5	9
8		Drama	7	12

Task Outline:

1. Remove null rows
2. Get Students, Fields, and Subjects into separate columns
3. Create appropriate headers

HELPER FUNCTIONS

```python
def extract_field_col(df,col,ref,new_col):
    '''
    Creates the field column and returns a dataframe.
    '''
    cond = df[col].isna()
    df[new_col] = np.where(cond,df[ref],np.nan)
    return df

def fill_col(df,col):
    '''
    Fills null values forward in a column
    and returns a dataframe.
    '''
    df[col] = df[col].ffill()
    return df

def remove_rows(df):
    '''
    Compares the first and last column,
    checks if any rows have the same text,
    removes the rows,
    and returns a dataframe.
    '''
    cond = df.iloc[:,0].eq(df.iloc[:,-1])
    df = df.loc[~cond]
    return df
```

CODE

```python
df = (pd.read_excel('worked-examples.xlsx',
                    sheet_name='pivot-hierarchy',
                    header=None)
      .remove_empty()
      .pipe(extract_field_col,2,1,'Field')
      .pipe(fill_col,'Field')
      .pipe(remove_rows)
      .fillna({1:'Subject',
               'Field':'Field'})
      .row_to_names(row_number=0, remove_row=True)
      .melt(id_vars=['Subject','Field'],
            value_name='Score',
            var_name='Student')
      )
```

RAW

	0	1	2	3
0	NaN	NaN	NaN	NaN
1	NaN	NaN	Matilda	Nicholas
2	NaN	Humanities	NaN	NaN
3	NaN	Classics	1	3
4	NaN	History	3	5
5	NaN	Performance	NaN	NaN
6	NaN	Music	5	9
7	NaN	Drama	7	12

TIDY

	Subject	Field	Student	Score
0	Classics	Humanities	Matilda	1
1	History	Humanities	Matilda	3
2	Music	Performance	Matilda	5
3	Drama	Performance	Matilda	7
4	Classics	Humanities	Nicholas	3
5	History	Humanities	Nicholas	5
6	Music	Performance	Nicholas	9
7	Drama	Performance	Nicholas	12

	0	1	2	3
0	NaN	NaN	NaN	NaN
1	NaN	NaN	Matilda	Nicholas
2	NaN	Humanities	NaN	NaN
3	NaN	Classics	1	3
4	NaN	History	3	5
5	NaN	Performance	NaN	NaN
6	NaN	Music	5	9
7	NaN	Drama	7	12

	1	2	3
0	NaN	Matilda	Nicholas
1	Humanities	NaN	NaN
2	Classics	1	3
3	History	3	5
4	Performance	NaN	NaN
5	Music	5	9
6	Drama	7	12

	1	2	3	Field
0	NaN	Matilda	Nicholas	NaN
1	Humanities	NaN	NaN	Humanities
2	Classics	1	3	NaN
3	History	3	5	NaN
4	Performance	NaN	NaN	Performance
5	Music	5	9	NaN
6	Drama	7	12	NaN

	1	2	3	Field
0	NaN	Matilda	Nicholas	NaN
1	Humanities	NaN	NaN	Humanities
2	Classics	1	3	Humanities
3	History	3	5	Humanities
4	Performance	NaN	NaN	Performance
5	Music	5	9	Performance
6	Drama	7	12	Performance

	1	2	3	Field
0	NaN	Matilda	Nicholas	NaN
2	Classics	1	3	Humanities
3	History	3	5	Humanities
5	Music	5	9	Performance
6	Drama	7	12	Performance

	1	2	3	Field
0	Subject	Matilda	Nicholas	Field
2	Classics	1	3	Humanities
3	History	3	5	Humanities
5	Music	5	9	Performance
6	Drama	7	12	Performance

	Subject	Matilda	Nicholas	Field
2	Classics	1	3	Humanities
3	History	3	5	Humanities
5	Music	5	9	Performance
6	Drama	7	12	Performance

	Subject	Field	Student	Score
0	Classics	Humanities	Matilda	1
1	History	Humanities	Matilda	3
2	Music	Performance	Matilda	5
3	Drama	Performance	Matilda	7
4	Classics	Humanities	Nicholas	3
5	History	Humanities	Nicholas	5
6	Music	Performance	Nicholas	9
7	Drama	Performance	Nicholas	12

CODE

```
df = (pd.read_excel('worked-examples.xlsx',
                    sheet_name='pivot-hierarchy',
                    header=None)
      .remove_empty()
      .pipe(extract_field_col,2,1,'Field')
      .pipe(fill_col,'Field')
      .pipe(remove_rows)
       .fillna({1:'Subject',
        'Field':'Field'})
         .row_to_names(row_number=0, remove_row=True)
          .melt(id_vars=['Subject','Field'],
      value_name='Score',
      var_name='Student')
       )
```

IMPLIED MULTIPLES

	B	C	D	E	F	G	H	I
1	Humanities				Performance			
2	**Classics**	**Grade**	**History**	**Grade**	**Music**	**Grade**	**Drama**	**Grade**
3	1	F	3	D	5	B	7	A
4	2	D	4	C	6	B	8	A

Courtesy: Nacnudus

HELPER FUNCTION

```python
def extract_col_grade(df,col,new_col,ref,text):

    cond = df[col]==text

    df[new_col] = np.where(cond,df[ref],np.nan)

    return df
```

CODE

```python
df = (pd.read_excel('worked-examples.xlsx',
                    sheet_name='implied-multiples',
                    header=None)
      .ffill(axis=1)
      .T
      .fillna('Field')
      .row_to_names(0,True)
      .melt(id_vars=['Field','Name'],
                    var_name='Student',
                    value_name = 'Score')
      .pipe(extract_col_grade,
            'Name','Grade',
            'Score','Grade')
      .bfill()
      .query('Name != "Grade"')
      .reset_index(drop = True)
      .rename(columns={"Name":'Subject'})
      )
```

RAW

	0	1	2	3	4	5	6	7	8
0	NaN	Humanities	NaN	NaN	NaN	Performance	NaN	NaN	NaN
1	Name	Classics	Grade	History	Grade	Music	Grade	Drama	Grade
2	Matilda	1	F	3	D	5	B	7	A
3	Olivia	2	D	4	C	6	B	8	A

TIDY

	Field	Subject	Student	Score	Grade
0	Humanities	Classics	Matilda	1	F
1	Humanities	History	Matilda	3	D
2	Performance	Music	Matilda	5	B
3	Performance	Drama	Matilda	7	A
4	Humanities	Classics	Olivia	2	D
5	Humanities	History	Olivia	4	C
6	Performance	Music	Olivia	6	B
7	Performance	Drama	Olivia	8	A

	0	1	2	3	4	5	6	7	8
0	NaN	Humanities	NaN	NaN	NaN	Performance	NaN	NaN	NaN
1	Name	Classics	Grade	History	Grade	Music	Grade	Drama	Grade
2	Matilda	1	F	3	D	5	B	7	A
3	Olivia	2	D	4	C	6	B	8	A

	0	1	2	...	6	7	8
0	NaN	Humanities	Humanities	...	Performance	Performance	Performance
1	Name	Classics	Grade	...	Grade	Drama	Grade
2	Matilda	1	F	...	B	7	A
3	Olivia	2	D	...	B	8	A

	0	1	2	3
0	NaN	Name	Matilda	Olivia
1	Humanities	Classics	1	2
2	Humanities	Grade	F	D
3	Humanities	History	3	4
4	Humanities	Grade	D	C
5	Performance	Music	5	6
6	Performance	Grade	B	B
7	Performance	Drama	7	8
8	Performance	Grade	A	A

	0	1	2	3
0	Field	Name	Matilda	Olivia
1	Humanities	Classics	1	2
2	Humanities	Grade	F	D
3	Humanities	History	3	4
4	Humanities	Grade	D	C
5	Performance	Music	5	6
6	Performance	Grade	B	B
7	Performance	Drama	7	8
8	Performance	Grade	A	A

	Field	Name	Matilda	Olivia
1	Humanities	Classics	1	2
2	Humanities	Grade	F	D
3	Humanities	History	3	4
4	Humanities	Grade	D	C
5	Performance	Music	5	6
6	Performance	Grade	B	B
7	Performance	Drama	7	8
8	Performance	Grade	A	A

	Field	Name	Student	Score
0	Humanities	Classics	Matilda	1
1	Humanities	Grade	Matilda	F
2	Humanities	History	Matilda	3
3	Humanities	Grade	Matilda	D
4	Performance	Music	Matilda	5
5	Performance	Grade	Matilda	B
6	Performance	Drama	Matilda	7
7	Performance	Grade	Matilda	A
8	Humanities	Classics	Olivia	2
9	Humanities	Grade	Olivia	D
10	Humanities	History	Olivia	4
11	Humanities	Grade	Olivia	C
12	Performance	Music	Olivia	6
13	Performance	Grade	Olivia	B
14	Performance	Drama	Olivia	8
15	Performance	Grade	Olivia	A

	Field	Name	Student	Score	Grade
0	Humanities	Classics	Matilda	1	F
1	Humanities	Grade	Matilda	F	F
2	Humanities	History	Matilda	3	D
3	Humanities	Grade	Matilda	D	D
4	Performance	Music	Matilda	5	B
5	Performance	Grade	Matilda	B	B
6	Performance	Drama	Matilda	7	A
7	Performance	Grade	Matilda	A	A
8	Humanities	Classics	Olivia	2	D
9	Humanities	Grade	Olivia	D	D
10	Humanities	History	Olivia	4	C
11	Humanities	Grade	Olivia	C	C
12	Performance	Music	Olivia	6	B
13	Performance	Grade	Olivia	B	B
14	Performance	Drama	Olivia	8	A
15	Performance	Grade	Olivia	A	A

	Field	Name	Student	Score	Grade
0	Humanities	Classics	Matilda	1	F
2	Humanities	History	Matilda	3	D
4	Performance	Music	Matilda	5	B
6	Performance	Drama	Matilda	7	A
8	Humanities	Classics	Olivia	2	D
10	Humanities	History	Olivia	4	C
12	Performance	Music	Olivia	6	B
14	Performance	Drama	Olivia	8	A

	Field	Subject	Student	Score	Grade
0	Humanities	Classics	Matilda	1	F
1	Humanities	History	Matilda	3	D
2	Performance	Music	Matilda	5	B
3	Performance	Drama	Matilda	7	A
4	Humanities	Classics	Olivia	2	D
5	Humanities	History	Olivia	4	C
6	Performance	Music	Olivia	6	B
7	Performance	Drama	Olivia	8	A

CODE

```
df = (pd.read_excel('worked-examples.xlsx',
                    sheet_name='implied-multiples',
                    header=None)
      .ffill(axis=1)
      .T
      .fillna('Field')
      .row_to_names(0,True)
      .melt(id_vars=['Field','Name'],
                     var_name='Student',
                     value_name = 'Score')
      .pipe(extract_col_grade,
            'Name','Grade',
            'Score','Grade')
      .bfill()
      .query('Name != "Grade"')
      .reset_index(drop = True)
      .rename(columns={"Name":'Subject'})
      )
```

PIVOT TABLE - CENTRE ALIGNED HEADERS

				Female				Male	
			Leah	Matilda	Olivia	Lenny	Max	Nicholas	Paul
		Classics	3	1	2	4	3	3	0
	Humanities	History	8	3	4	7	5	5	1
		Literature	1	1	9	3	12	7	5
		Philosophy	5	10	10	8	2	5	12
		Languages	5	4	5	9	8	3	8
		Music	4	10	10	2	4	5	6
	Performanc	Dance	4	5	6	4	12	9	2
		Drama	2	7	8	6	1	12	3

Courtesy: Nacnudus

Task Outline:

1. Find the border locations
2. Pull out section within the borders for Field and Subject and clean
3. Pull out section within the borders for gender and Student and clean
4. Merge dataframes into one
5. Perform final cleanup

	A	B	C	D	E	F	G	H	I	J
1										
2			Border →		Female	→			Male	→
3				Leah	Matilda	Olivia	Lenny	Max	Nicholas	Paul
4			Classics	3	1	2	4	3	3	0
5		Humanities	History	8	3	4	7	5	5	1
6			Literature	1	1	9	3	12	7	5
7			Philosophy	5	10	10	8	2	5	12
8	Field	↓	Languages	5	4	5	9	8	3	8
9			Music	4	10	10	2	4	5	6
10		Performanc	Dance	4	5	6	4	12	9	2
11			Drama	2	7	8	6	1	12	3

↑

HELPER FUNCTION

```
def rename_col(df):
    '''
    Merges first two rows of dataframe,
    assigns aggregation as the new column names,
    drops the first two rows from the dataframe
    and returns a dataframe
    '''

    df.columns = df.iloc[:2].add(',').sum().str.strip(',')

    df = df.iloc[2:]

    return df

def create_col(df,col,new_col):
    '''
    Extracts new_col from col;
    splits col into two,
    assigns first part of the split to new_col,
    second part to col
    and returns a dataframe
    '''

    df[new_col] = df[col].str.split(',').str[0]

    df[col] = df[col].str.split(',').str[-1]

    return df
```

Get border location

```
wb = load_workbook(filename='worked-examples.xlsx')

ws = wb['pivot-centre-aligned']

cols = set()
rows = set()

for line in ws:
    for cell in line:
        if cell.border.right.style:
            cols.add(cell.column)
        elif cell.border.bottom.style:
            rows.add(cell.row)

cols = sorted(cols)
rows = sorted(rows)

col_count = len(cols) - 1 # 'box' count
row_count = len(rows) - 1

rows = list(windowed(rows, row_count))
cols = list(windowed(cols, col_count))
```

```
In [18]: print(rows,cols)
[(3, 8), (8, 11)] [(3, 6), (6, 10)]
```

CODE

```
gender_and_names = [df.copy()
                    .iloc[:,start:end]
                    .dropna(how='all')
                    .ffill(axis=1)
                    .bfill(axis=1)
                    for start,end
                    in cols
                   ]

gender = pd.concat(gender_and_names,axis=1)
```

```
Fields = [df.copy()
          .iloc[start:end]
          .dropna(how='all',axis=1)
          .ffill()
          .bfill()
          .iloc[:,:2]
          for start,end in rows
         ]

fields = pd.concat(Fields)
```

```
sh = (pd.concat([fields,gender],axis=1)
      .fillna(value={1:'Field',
                     2:'Subject'},
              limit=1)
      .fillna('')
      .pipe(rename_col)
      .melt(id_vars=['Field','Subject'],
            var_name='Student',
            value_name='Score')
      .pipe(create_col,'Student','Gender')
)
```

	0	1	2	3	...	6	7	8	9
0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	...	NaN	NaN	Male	NaN
2	NaN	NaN	NaN	Leah	...	Lenny	Max	Nicholas	Paul
3	NaN	NaN	Classics	3	...	4	3	3	0
4	NaN	Humanities	History	8	...	7	5	5	1
5	NaN	NaN	Literature	1	...	3	12	7	5
6	NaN	NaN	Philosophy	5	...	8	2	5	12
7	NaN	NaN	Languages	5	...	9	8	3	8
8	NaN	NaN	Music	4	...	2	4	5	6
9	NaN	Performance	Dance	4	...	4	12	9	2
10	NaN	NaN	Drama	2	...	6	1	12	3

	1	2
3	Humanities	Classics
4	Humanities	History
5	Humanities	Literature
6	Humanities	Philosophy
7	Humanities	Languages,
	1	2
8	Performance	Music
9	Performance	Dance
10	Performance	Drama]

	1	2
3	Humanities	Classics
4	Humanities	History
5	Humanities	Literature
6	Humanities	Philosophy
7	Humanities	Languages
8	Performance	Music
9	Performance	Dance
10	Performance	Drama

CODE

```
Fields = [df.copy()
          .iloc[start:end]
          .dropna(how='all',axis=1)
          .ffill()
          .bfill()
          .iloc[:,:2]
          for start,end in rows
         ]

fields = pd.concat(Fields)
```

	3	4	5
1	Female	Female	Female
2	Leah	Matilda	Olivia
3	3	1	2
4	8	3	4
5	1	1	9
6	5	10	10
7	5	4	5
8	4	10	10
9	4	5	6
10	2	7	8,
	6	7	8	9
1	Male	Male	Male	Male
2	Lenny	Max	Nicholas	Paul
3	4	3	3	0
4	7	5	5	1
5	3	12	7	5
6	8	2	5	12
7	9	8	3	8
8	2	4	5	6
9	4	12	9	2
10	6	1	12	3]

	3	4	5	6	7	8	9
1	Female	Female	Female	Male	Male	Male	Male
2	Leah	Matilda	Olivia	Lenny	Max	Nicholas	Paul
3	3	1	2	4	3	3	0
4	8	3	4	7	5	5	1
5	1	1	9	3	12	7	5
6	5	10	10	8	2	5	12
7	5	4	5	9	8	3	8
8	4	10	10	2	4	5	6
9	4	5	6	4	12	9	2
10	2	7	8	6	1	12	3

CODE

```
gender_and_names = [df.copy()
                    .iloc[:,start:end]
                    .dropna(how='all')
                    .ffill(axis=1)
                    .bfill(axis=1)
                    for start,end
                    in cols
                   ]

gender = pd.concat(gender_and_names,axis=1)
```

	1	2	3	4	...	6	7	8	9
1	NaN	NaN	Female	Female	...	Male	Male	Male	Male
2	NaN	NaN	Leah	Matilda	...	Lenny	Max	Nicholas	Paul
3	Humanities	Classics	3	1	...	4	3	3	0
4	Humanities	History	8	3	...	7	5	5	1
5	Humanities	Literature	1	1	...	3	12	7	5
6	Humanities	Philosophy	5	10	...	8	2	5	12
7	Humanities	Languages	5	4	...	9	8	3	8
8	Performance	Music	4	10	...	2	4	5	6
9	Performance	Dance	4	5	...	4	12	9	2
10	Performance	Drama	2	7	...	6	1	12	3

	1	2	3	4	...	6	7	8	9
1	Field	Subject	Female	Female	...	Male	Male	Male	Male
2	NaN	NaN	Leah	Matilda	...	Lenny	Max	Nicholas	Paul
3	Humanities	Classics	3	1	...	4	3	3	0
4	Humanities	History	8	3	...	7	5	5	1
5	Humanities	Literature	1	1	...	3	12	7	5
6	Humanities	Philosophy	5	10	...	8	2	5	12
7	Humanities	Languages	5	4	...	9	8	3	8
8	Performance	Music	4	10	...	2	4	5	6
9	Performance	Dance	4	5	...	4	12	9	2
10	Performance	Drama	2	7	...	6	1	12	3

	1	2	3	4	...	6	7	8	9
1	Field	Subject	Female	Female	...	Male	Male	Male	Male
2			Leah	Matilda	...	Lenny	Max	Nicholas	Paul
3	Humanities	Classics	3	1	...	4	3	3	0
4	Humanities	History	8	3	...	7	5	5	1
5	Humanities	Literature	1	1	...	3	12	7	5
6	Humanities	Philosophy	5	10	...	8	2	5	12
7	Humanities	Languages	5	4	...	9	8	3	8
8	Performance	Music	4	10	...	2	4	5	6
9	Performance	Dance	4	5	...	4	12	9	2
10	Performance	Drama	2	7	...	6	1	12	3

	Field	Subject	Female,Leah	...	Male,Max	Male,Nicholas	Male,Paul
3	Humanities	Classics	3	...	3	3	0
4	Humanities	History	8	...	5	5	1
5	Humanities	Literature	1	...	12	7	5
6	Humanities	Philosophy	5	...	2	5	12
7	Humanities	Languages	5	...	8	3	8
8	Performance	Music	4	...	4	5	6
9	Performance	Dance	4	...	12	9	2
10	Performance	Drama	2	...	1	12	3

	Field	Subject	Student	Score
0	Humanities	Classics	Female,Leah	3
1	Humanities	History	Female,Leah	8
2	Humanities	Literature	Female,Leah	1
3	Humanities	Philosophy	Female,Leah	5
4	Humanities	Languages	Female,Leah	5
5	Performance	Music	Female,Leah	4
6	Performance	Dance	Female,Leah	4
7	Performance	Drama	Female,Leah	2
8	Humanities	Classics	Female,Matilda	1
9	Humanities	History	Female,Matilda	3

	Field	Subject	Student	Score	Gender
0	Humanities	Classics	Leah	3	Female
1	Humanities	History	Leah	8	Female
2	Humanities	Literature	Leah	1	Female
3	Humanities	Philosophy	Leah	5	Female
4	Humanities	Languages	Leah	5	Female
5	Performance	Music	Leah	4	Female
6	Performance	Dance	Leah	4	Female
7	Performance	Drama	Leah	2	Female
8	Humanities	Classics	Matilda	1	Female
9	Humanities	History	Matilda	3	Female

CODE

```
sh = (pd.concat([fields,gender],axis=1)
      .fillna(value={1:'Field',
                     2:'Subject'},
              limit=1)
      .fillna('')
      .pipe(rename_col)
      .melt(id_vars=['Field','Subject'],
            var_name='Student',
            value_name='Score')
      .pipe(create_col,'Student','Gender')
)
```

Q & A

Samukweku.github.io

github.com/samukweku