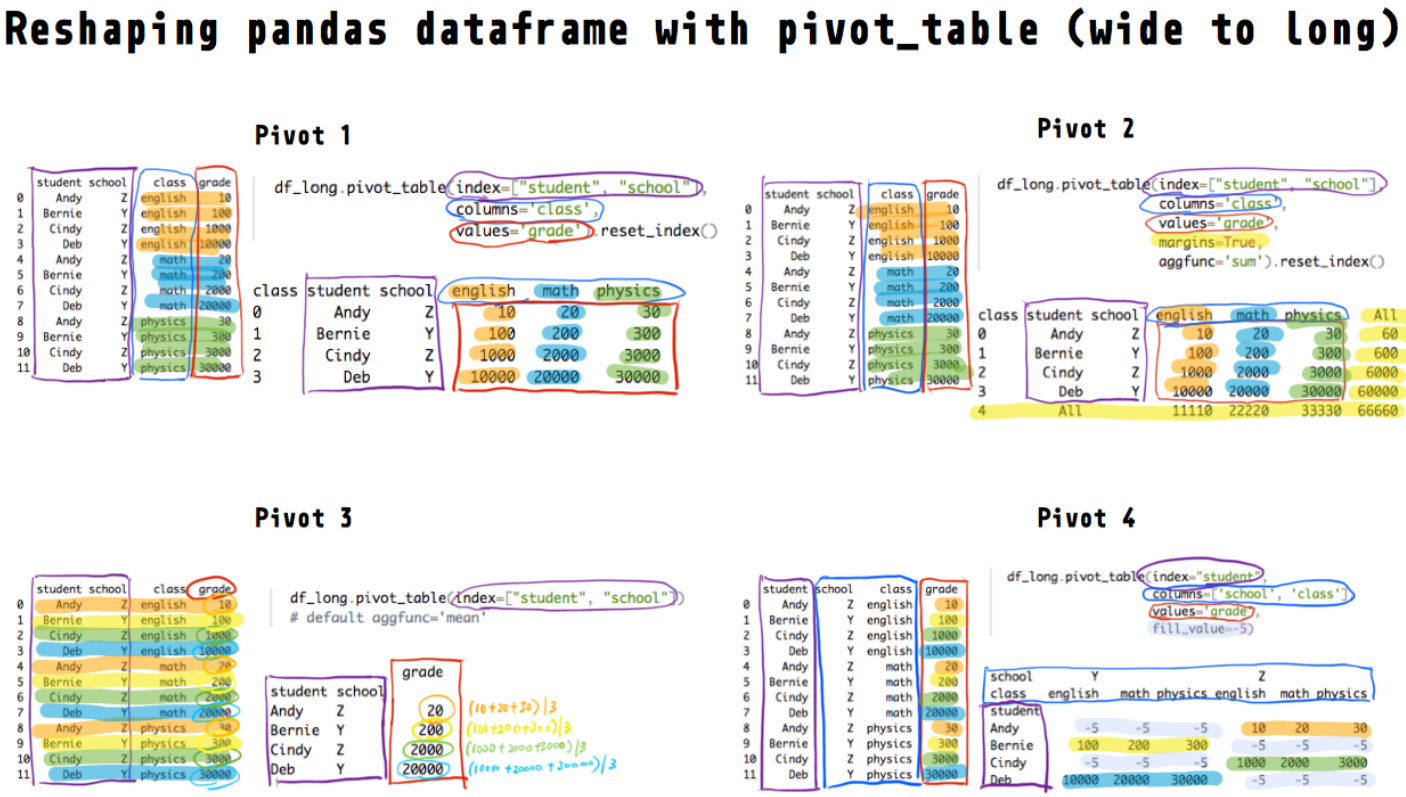
PYTHON PANDAS DATAFRAME RESHAPING

Reshape pandas dataframe with pivot_table in Python — tutorial and visualization

Convert long to wide with pd.pivot_table

Hause Lin
May 22 · 5 min read ★



How to use pd.pivot_table() to reshape pandas dataframes from long to wide in Python (run code here)

There are many different ways to reshape a pandas dataframe from **long to wide** form. But the `pivot_table()` method is the most flexible and probably the only one you need

to use once you learn it well, just like how you only need to learn one method `melt` to reshape from **wide to long** (see my other post below).

Reshape pandas dataframe with melt in Python — tutorial and visualization

Visualize how pd.melt reshapes pandas dataframes from wide to long form

towardsdatascience.com

This tutorial will walk you through reshaping dataframes using `pd.pivot_table` or the `pivot_table` method associated with pandas dataframes. In other languages like R, pivot is also known as spread or dcast.

I highly recommend you try the code in Python while you read this article. Try running this tutorial on my shared **DeepNote notebook** (you can only run but not edit this notebook).

Also, you might be interested a similar tutorial that describes the reverse (reshape from wide to long) with `pd.melt` and my numpy reshape tutorial.

Reshaping numpy arrays in Python — a step-by-step pictorial tutorial

Visualize how numpy reshapes arrays

towardsdatascience.com

. . .

Long versus wide dataframe

It's easiest to understand what a **long** dataframe is or looks like if we look at one and compare it with a wide dataframe.

	student	school	cLaSs	gRaDe
0	Andy	Z	english	10

1	Bernie	Y	english	100
2	Cindy	Z	english	1000
3	Deb	Y	english	10000
4	Andy	Z	math	20
5	Bernie	Y	math	200
6	Cindy	Z	math	2000
7	Deb	Y	math	20000
8	Andy	Z	physics	30
9	Bernie	Y	physics	300
10	Cindy	Z	physics	3000
11	Deb	Y	physics	30000

Long pandas dataframe can be pivoted or "unmelted" using pd.pivot_table() (run code here)

And below is the corresponding dataframe (with the same information) but in the **wide** form:

	student	school	english	math	physics
0	Andy	Z	10	20	30
1	Bernie	Y	100	200	300
2	Cindy	Z	1000	2000	3000
3	Deb	Y	10000	20000	30000

Wide pandas dataframe can be melted/stacked using pd.melt()

Before we begin our `pd.pivot_table` tutorial, let's recreate the wide dataframe above in Python with `pd.DataFrame`. Remember, you can also follow along with my shared notebook.

```
df_long = pd.DataFrame({
        "student":
            ["Andy", "Bernie", "Cindy", "Deb",
             "Andy", "Bernie", "Cindy", "Deb",
```

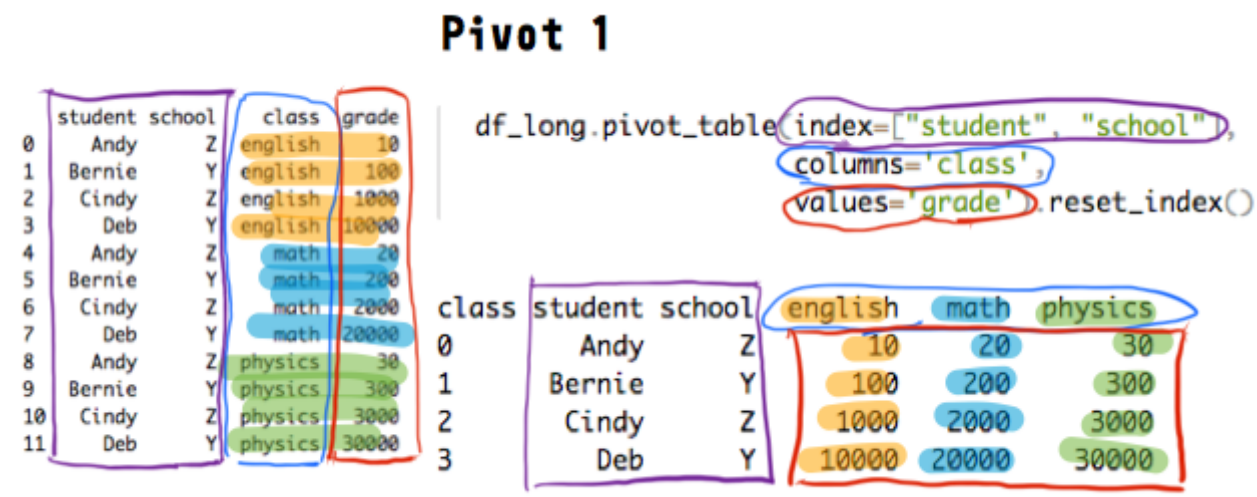
```
            "Andy", "Bernie", "Cindy", "Deb"],
        "school":
            ["Z", "Y", "Z", "Y",
             "Z", "Y", "Z", "Y",
             "Z", "Y", "Z", "Y"],
        "class":
            ["english", "english", "english", "english",
             "math", "math", "math", "math",
             "physics", "physics", "physics", "physics"],
        "grade":
            [10, 100, 1000, 10000,
             20, 200, 2000, 20000,
             30, 300, 3000, 30000]
})
```

. . .

Example 1

We often want to keep the identifier columns as they are (`index=["student", "school"]`), but pivot or "split" a column's values (`values="grade"`) based on another column (`columns="class"`). Compare the original and pivoted dataframes below and you'll understand what that means.

```
df_long.pivot_table(index=["student", "school"],
                    columns='class',
                    values='grade')
```



Long to wide: values in grade are split/pivoted into three separate columns (run code here)

Each unique value in the class column will be a new column (english, math, physics) in the pivoted/wide dataframe. We can also provide a **list** to the `columns` parameter.

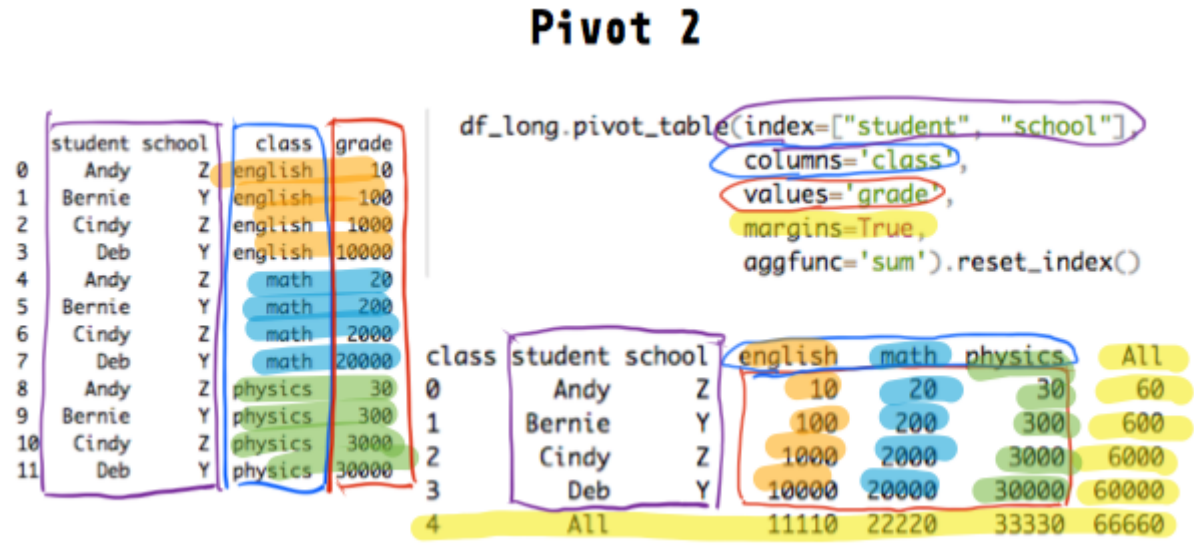
To get rid of the multi-index, use `reset_index()`.

Example 2

You can also aggregate each resulting row and column by specifying `margins=True` (default `False`).

```
df_long.pivot_table(index=["student", "school"],
                    columns='class',
                    values='grade',
                    margins=True,  # add margins
                    aggfunc='sum')  # sum margins (rows/columns)
```

Here we aggregate by computing the sum via `aggfunc='sum'` (default `'mean'`).



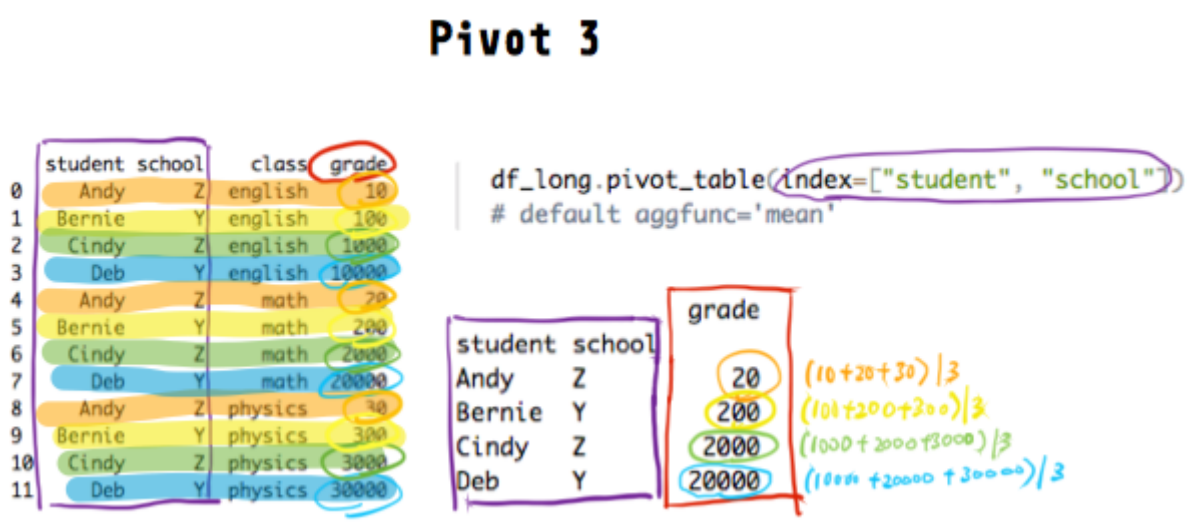
Long to wide: margins are included (run code here)

There are many other aggregation functions you can use (e.g., `'median' 'sum' 'max'`). You can also specify multiple functions as a list (e.g., `aggfunc=['mean', 'sum']`).

Example 3

If we don't specify any columns via `columns`, **all** remaining non-identifier **numeric** columns (only grade in this dataframe) will be pivoted (long to wide).

```
df_long.pivot_table(index=["student", "school"])
```



Long to wide: all non-identifier numeric columns are pivoted (run code here)

In the original long data, each student has **four** grades (english, math, physics), yet in the `pivot_table` example above, each student only has **one** grade after pivoting.

Why and how does it work? If you remember from the example above, the default is `aggfunc='mean'`. Thus, what the function did was it grouped the data by student and school (via `index=["student", "school"]`), and computed the mean value for each group.

If you use the `groupby` method associated with the pandas dataframe, you will get the same result as above.

```
df_long.groupby(['student', 'school']).mean().reset_index()
```

```
  student school  grade
0    Andy      Z     20
1  Bernie      Y    200
2   Cindy      Z   2000
3     Deb      Y  20000
```

If you change the default aggregation function (e.g., `aggfunc='max'`), you'll get different results. The examples below show you how to specify different aggregation functions and also show you how `groupby` can be used to perform the same pivot.

Note that you'll also see the class that is associated with each 'max' and 'first' value.

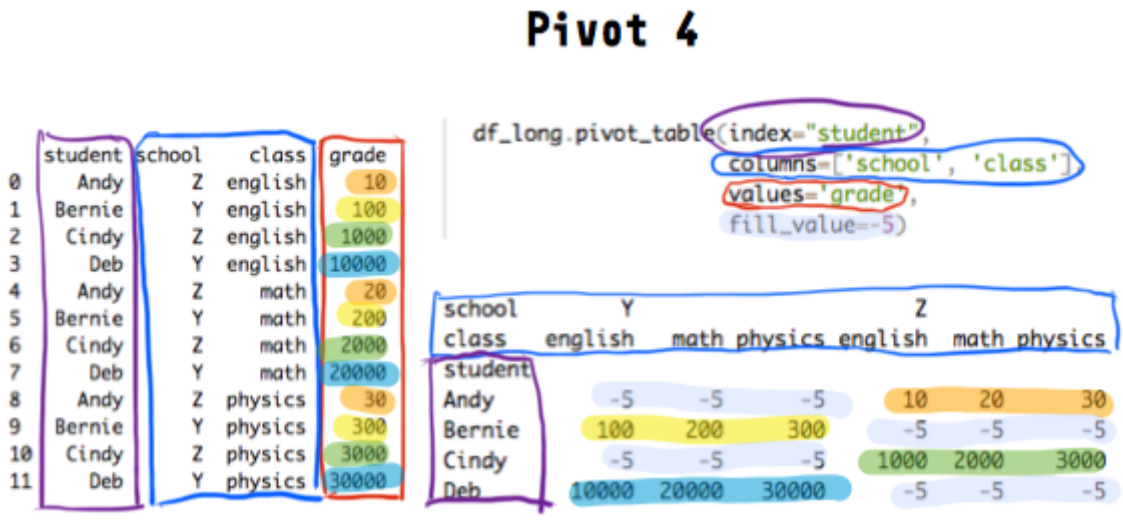
```
df_long.pivot_table(index=["student", "school"],
                    aggfunc=['max', 'first'])

# groupby equivalent
# df_long.groupby(["student", "school"]).agg(['max', 'first'])
```

		max		first	
		class	grade	class	grade
student	school				
Andy	Z	physics	30	english	10
Bernie	Y	physics	300	english	100
Cindy	Z	physics	3000	english	1000
Deb	Y	physics	30000	english	10000

Example 4

The final example shows you what happens when you pivot multiple columns (`columns=['school', 'class']`) and you can also deal with missing values after pivoting by replacing the `NaN` values with another value (-5 in the example below).

```
df_long.pivot_table(index="student",
                    columns=['school', 'class'],
                    values='grade',
                    fill_value=-5)  # replace NaN with -5
```



Long to wide: missing values replaced with -5 (default NaN) (run code here)

The `NaN` values are expected because each student belongs to only one school (Y or Z). For example, Andy is in school Z and therefore doesn't have grades in the Y columns.

Final remarks

I hope now you have a better understanding of how `pd.pivot_table` reshapes dataframes. I look forward to your thoughts and comments.

If you find this post useful, follow me and visit my site for more data science tutorials and also my other articles:

Reshape pandas dataframe with melt in Python — tutorial and visualization

Visualize how pd.melt reshapes pandas dataframes from wide to long form

towardsdatascience.com

Two Simple Ways to Loop More Effectively in Python

Use enumerate and zip to write better Python loops

towardsdatascience.com

4 Keyboard Shortcuts to Edit Text Efficiently and Improve Productivity

Navigate and move your cursor through text efficiently

medium.com

Real or Spurious Correlations: Attractive People You Date Are Nastier

Use Python to simulate data, test intuitions, and improve data science skills

towardsdatascience.com

Code and Develop More Productively With Terminal Multiplexer tmux

Simple tmux commands to improve your productivity

medium.com

Sign up for The Daily Pick

By Towards Data Science

Hands-on real-world examples, research, tutorials, and cutting-edge techniques delivered Monday to Thursday. Make learning your daily ritual. Take a look

Get this newsletter

Create a free Medium account to get The Daily Pick in your inbox.