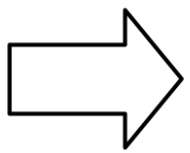
Lecture 15: Reshaping data

Last time: `pivot_longer`

id	bp1	bp2
A	100	120
B	140	115
C	120	125



id	measurement	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125

```
df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
```

Warm up

```
1 ex_df
```

```
  id x_1 x_2 y_1 y_2
1  1   3   5   0   2
2  2   1   8   1   7
3  3   4   9   2   9
```

```
1 ex_df |>
2   pivot_longer(cols = -id,
3                names_to = c("group", "obs"),
4                values_to = "value",
5                names_sep = "_")
```

Write down the data frame that will be returned by the code.

id	group	obs	value
1	x	1	3
1	x	2	5
1	y	1	0
1	y	2	2
⋮	⋮	⋮	⋮

Warmup

```r
ex_df |>
  pivot_longer(cols = -id,
               names_to = c("group", "obs"),
               values_to = "value",
               names_sep = "_")
```

```
# A tibble: 12 × 4
      id group obs   value
   <dbl> <chr> <chr> <dbl>
 1     1 x     1         3
 2     1 x     2         5
 3     1 y     1         0
 4     1 y     2         2
 5     2 x     1         1
 6     2 x     2         8
 7     2 y     1         1
 8     2 y     2         7
 9     3 x     1         4
10     3 x     2         9
11     3 y     1         2
```

`pivot_longer` in R

Consider the following example data:

```
  id bp_1 bp_2 hr_1 hr_2
1  1  100  120   60   77
2  2  120  115   75   81
3  3  125  130   80   93
```

What if we want the data to look like this:

```
# A tibble: 12 × 4
      id measurement stage value
   <dbl> <chr>       <chr> <dbl>
 1     1 bp          1       100
 2     1 bp          2       120
 3     1 hr          1        60
 4     1 hr          2        77
 5     2 bp          1       120
 6     2 bp          2       115
 7     2 hr          1        75
 8     2 hr          2        81
 9     3 bp          1       125
10     3 bp          2       130
```

```
pivot_longer(cols = -id,
             names_to=c("measurement",
                        "stage"),
             names_sep = "_",
             values_to = "value")
```

pivot_longer in R

```r
df2
```

```
  id bp_1 bp_2 hr_1 hr_2
1  1  100  120   60   77
2  2  120  115   75   81
3  3  125  130   80   93
```

```r
df2 |>
  pivot_longer(cols = -id,
               names_to = c("measurement", "stage"),
               names_sep = "_",
               values_to = "value")
```

```
# A tibble: 12 × 4
      id measurement stage value
   <dbl> <chr>       <chr> <dbl>
 1     1 bp          1       100
 2     1 bp          2       120
 3     1 hr          1        60
 4     1 hr          2        77
 5     2 bp          1       120
 6     2 bp          2       115
 7     2 hr          1        75
 8     2 hr          2        81
 9     3 bp          1       125
10     3 bp          2       130
```

pivot_longer in R

```r
df2 |>
  pivot_longer(cols = -id,
               names_to = c("measurement", "stage"),
               names_sep = "_",
               values_to = "value")
```

Step 1: Pivot

```
# A tibble: 6 × 3
     id measurement value
  <dbl> <chr>       <dbl>
1     1 bp_1          100
2     1 bp_2          120
3     1 hr_1           60
4     1 hr_2           77
5     2 bp_1          120
6     2 bp_2          115
```

Step 2: Separate columns

```
# A tibble: 6 × 4
     id measurement stage value
  <dbl> <chr>       <chr> <dbl>
1     1 bp          1       100
2     1 bp          2       120
3     1 hr          1        60
4     1 hr          2        77
5     2 bp          1       120
6     2 bp          2       115
```

Python

```
df2.melt(id_vars = 'id',
         var_name = 'measurement',
         value_name = 'value')
```

In Python

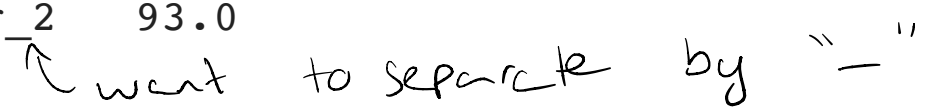
Step 1: Melt

```python
import pandas as pd

df2 = r.df2

df2_new = df2.melt(id_vars = 'id',
                   var_name = 'measurement',
                   value_name = 'value')
df2_new
```

	id	measurement	value
0	1.0	bp_1	100.0
1	2.0	bp_1	120.0
2	3.0	bp_1	125.0
3	1.0	bp_2	120.0
4	2.0	bp_2	115.0
5	3.0	bp_2	130.0
6	1.0	hr_1	60.0
7	2.0	hr_1	75.0
8	3.0	hr_1	80.0
9	1.0	hr_2	77.0
10	2.0	hr_2	81.0
11	3.0	hr_2	93.0



In Python

Step 2: Separate columns

```
df2 = r.df2

df2_new = df2.melt(id_vars = 'id',
                   var_name = 'measurement',
                   value_name = 'value')
df2_new['measurement'].str.split('_', expand=True)
```

create new columns for the split

take the 'measurement' column and split split by '_'

	0	1
0	bp	1
1	bp	1
2	bp	1
3	bp	2
4	bp	2
5	bp	2
6	hr	1
7	hr	1
8	hr	1
9	hr	2
10	hr	2
11	hr	2

In Python

Step 2: Separate columns

```
1  df2 = r.df2
2
3  df2_new = df2.melt(id_vars = 'id',
4                     var_name = 'measurement',
5                     value_name = 'value')
6  df2_new[['measurement', 'stage']] = (df2_new['measurement']
7                                       .str.split('_', expand=True))
8  df2_new
```

Stae right handSide in two columns: "measurement" and "stage"

Splitting "measurement" into two columns

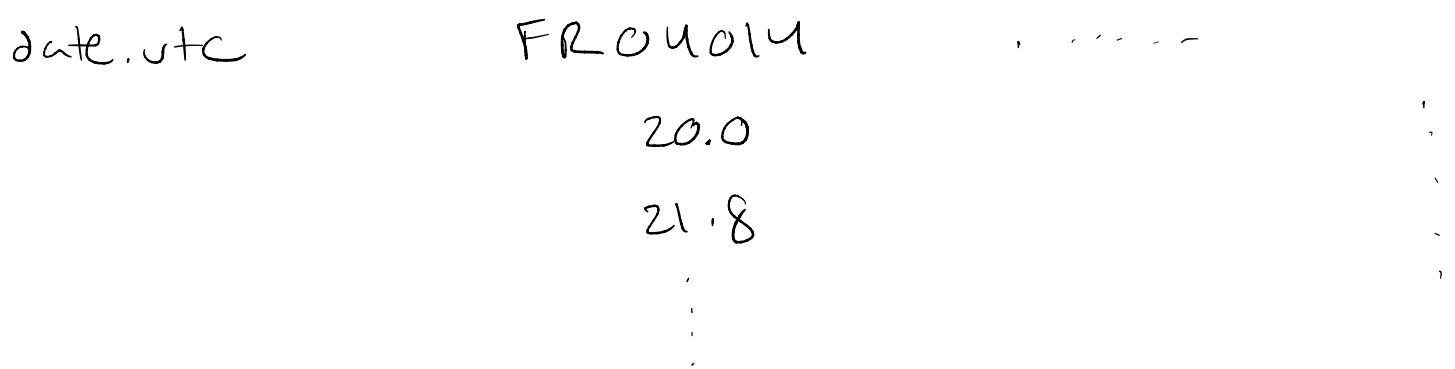
	id	measurement	value	stage
0	1.0	bp	100.0	1
1	2.0	bp	120.0	1
2	3.0	bp	125.0	1
3	1.0	bp	120.0	2
4	2.0	bp	115.0	2
5	3.0	bp	130.0	2
6	1.0	hr	60.0	1
7	2.0	hr	75.0	1
8	3.0	hr	80.0	1
9	1.0	hr	77.0	2
10	2.0	hr	81.0	2
11	3.0	hr	93.0	2

Going the other way

```
1 air_quality
```

```
                      date.utc location  value
1825  2019-06-21 00:00:00+00:00   FR04014   20.0
1826  2019-06-20 23:00:00+00:00   FR04014   21.8
1827  2019-06-20 22:00:00+00:00   FR04014   26.5
1828  2019-06-20 21:00:00+00:00   FR04014   24.9
1829  2019-06-20 20:00:00+00:00   FR04014   21.4
```

What if I want a separate column for each location?



Going the other way

take "long" data & make it "wider"

```
air_quality.pivot(index = 'date.utc',
                  columns = 'location',
                  values = 'value')
```

columns to keep as index

columns to turn into new columns

where to get the entries in the new columns

location	BETR801	FR04014	London Westminster
date.utc			
2019-04-09 01:00:00+00:00	22.5	24.4	NaN
2019-04-09 02:00:00+00:00	53.5	27.4	67.0
2019-04-09 03:00:00+00:00	54.5	34.2	67.0
2019-04-09 04:00:00+00:00	34.5	48.5	41.0
2019-04-09 05:00:00+00:00	46.5	59.5	41.0
...	...	...	...
2019-06-20 20:00:00+00:00	NaN	21.4	NaN
2019-06-20 21:00:00+00:00	NaN	24.9	NaN
2019-06-20 22:00:00+00:00	NaN	26.5	NaN
2019-06-20 23:00:00+00:00	NaN	21.8	NaN
2019-06-21 00:00:00+00:00	NaN	20.0	NaN

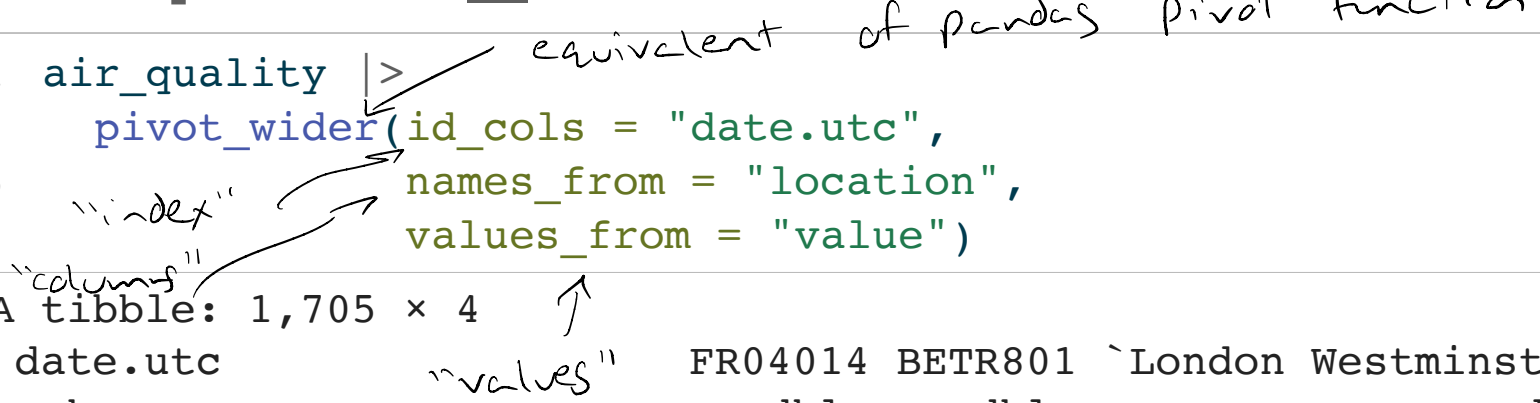
In R

```r
air_quality <- py$air_quality
head(air_quality)
```

```
                    date.utc location value
1825 2019-06-21 00:00:00+00:00  FR04014  20.0
1826 2019-06-20 23:00:00+00:00  FR04014  21.8
1827 2019-06-20 22:00:00+00:00  FR04014  26.5
1828 2019-06-20 21:00:00+00:00  FR04014  24.9
1829 2019-06-20 20:00:00+00:00  FR04014  21.4
1830 2019-06-20 19:00:00+00:00  FR04014  25.3
```

In R: pivot_wider

```r
air_quality |>
  pivot_wider(id_cols = "date.utc",
              names_from = "location",
              values_from = "value")
```




```
# A tibble: 1,705 × 4
   date.utc                  FR04014 BETR801 `London Westminster`
   <chr>                       <dbl>   <dbl>                <dbl>
 1 2019-06-21 00:00:00+00:00    20        NA                   NA
 2 2019-06-20 23:00:00+00:00    21.8      NA                   NA
 3 2019-06-20 22:00:00+00:00    26.5      NA                   NA
 4 2019-06-20 21:00:00+00:00    24.9      NA                   NA
 5 2019-06-20 20:00:00+00:00    21.4      NA                   NA
 6 2019-06-20 19:00:00+00:00    25.3      NA                   NA
 7 2019-06-20 18:00:00+00:00    23.9      NA                   NA
 8 2019-06-20 17:00:00+00:00    23.2      NA                   NA
 9 2019-06-20 16:00:00+00:00    19        NA                   NA
10 2019-06-20 15:00:00+00:00    19.3      NA                   NA
# ℹ 1,695 more rows
```

Class activity

https://sta279-s24.github.io/class_activities/ca_lecture_15.html