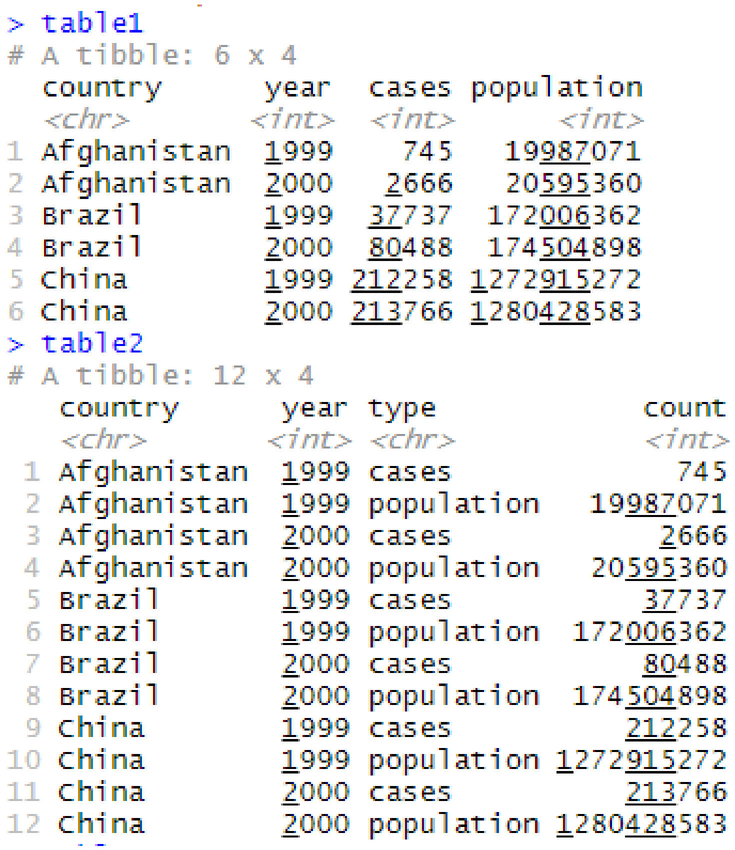
London
Business
School

Session 4:
Tidying data
Confidence Intervals

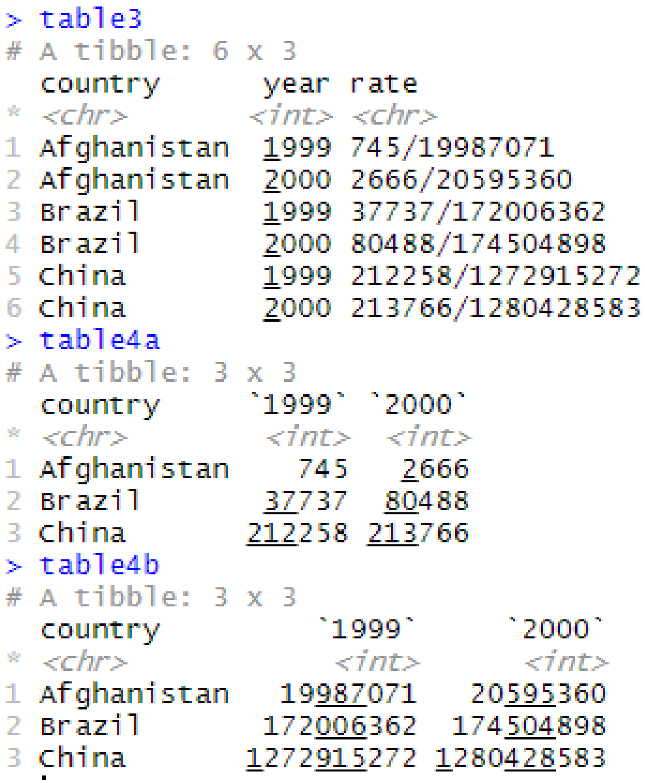
Kostis Christodoulou
London Business School

Tidy data (1/3)

The tables below show the same values of four variables **country, year, cases, population,** but each dataset organises the values in a different way.



```
> table1
# A tibble: 6 x 4
  country      year  cases population
  <chr>       <int>  <int>      <int>
1 Afghanistan  1999    745   19987071
2 Afghanistan  2000   2666   20595360
3 Brazil       1999  37737  172006362
4 Brazil       2000  80488  174504898
5 China        1999 212258 1272915272
6 China        2000 213766 1280428583
> table2
# A tibble: 12 x 4
   country      year type            count
   <chr>       <int> <chr>           <int>
 1 Afghanistan  1999 cases             745
 2 Afghanistan  1999 population   19987071
 3 Afghanistan  2000 cases            2666
 4 Afghanistan  2000 population   20595360
 5 Brazil       1999 cases           37737
 6 Brazil       1999 population  172006362
 7 Brazil       2000 cases           80488
 8 Brazil       2000 population  174504898
 9 China        1999 cases          212258
10 China        1999 population 1272915272
11 China        2000 cases          213766
12 China        2000 population 1280428583
```



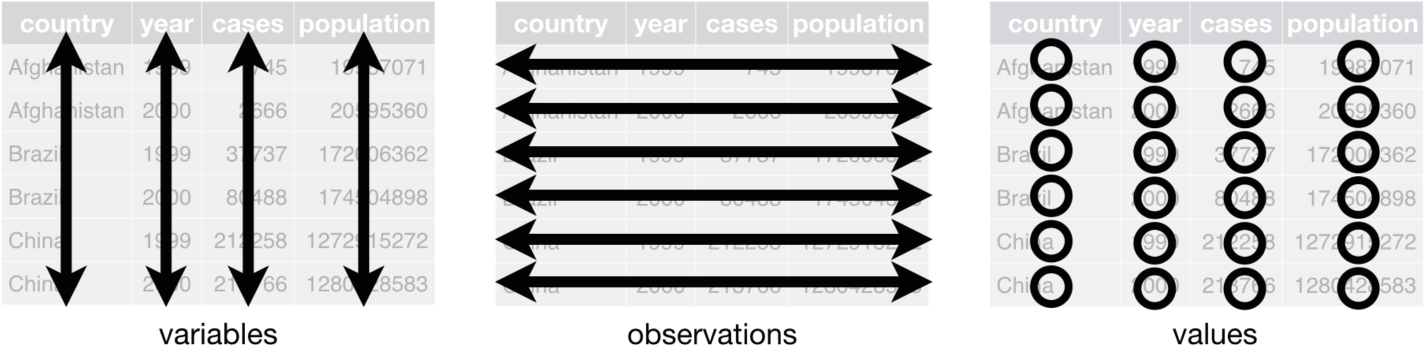
```
> table3
# A tibble: 6 x 3
  country      year rate
* <chr>       <int> <chr>
1 Afghanistan  1999 745/19987071
2 Afghanistan  2000 2666/20595360
3 Brazil       1999 37737/172006362
4 Brazil       2000 80488/174504898
5 China        1999 212258/1272915272
6 China        2000 213766/1280428583
> table4a
# A tibble: 3 x 3
  country     `1999` `2000`
* <chr>        <int>  <int>
1 Afghanistan    745   2666
2 Brazil       37737  80488
3 China       212258 213766
> table4b
# A tibble: 3 x 3
  country         `1999`     `2000`
* <chr>            <int>      <int>
1 Afghanistan   19987071   20595360
2 Brazil       172006362  174504898
3 China       1272915272 1280428583
```

Tidy data 2/3

Tidy data is a specific way of organising data to facilitate analysis with the tidyverse. The tidy data standard has been designed to facilitate exploratory data analysis; tidy datasets and tidy tools help make data analysis easier, allowing you to focus on the interesting domain problem, not on the logistics of cleaning data.

There are three rules which make a dataset tidy:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.



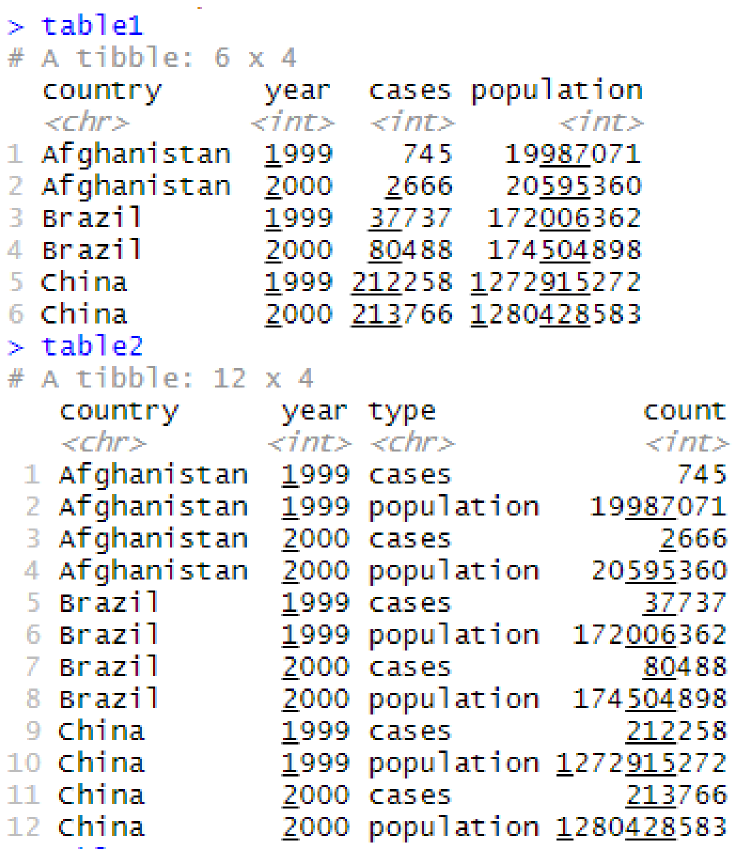
We often need to reshape our datasets and should have a way to go:

- from wide format to long (tidy) format using *gather()* or *pivot_longer()*
- from long (tidy) to wide format using *spread()* or *pivot_wider()*

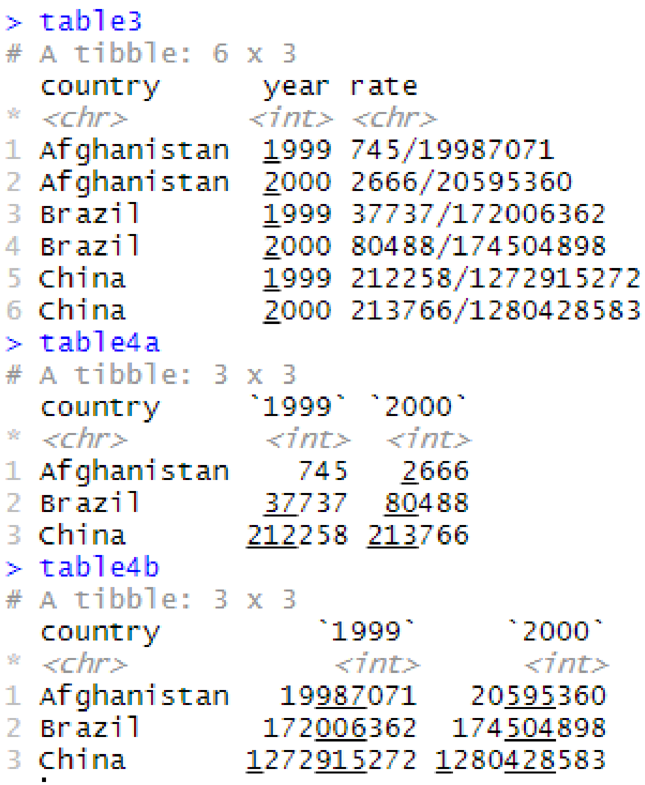
Tidy data 3/3

Five main ways data tend not to be tidy:

1. *Column headers are values, not variable names*.
2. *Multiple variables are stored in one column*.
3. *Variables are stored in both rows and columns*.
4. *Multiple types of observational units are stored in the same table*.
5. *A single observational unit is stored in multiple tables*.



```
> table1
# A tibble: 6 x 4
  country      year  cases population
  <chr>       <int>  <int>      <int>
1 Afghanistan  1999    745   19987071
2 Afghanistan  2000   2666   20595360
3 Brazil       1999  37737  172006362
4 Brazil       2000  80488  174504898
5 China        1999 212258 1272915272
6 China        2000 213766 1280428583
> table2
# A tibble: 12 x 4
   country      year type            count
   <chr>       <int> <chr>           <int>
 1 Afghanistan  1999 cases             745
 2 Afghanistan  1999 population   19987071
 3 Afghanistan  2000 cases            2666
 4 Afghanistan  2000 population   20595360
 5 Brazil       1999 cases           37737
 6 Brazil       1999 population  172006362
 7 Brazil       2000 cases           80488
 8 Brazil       2000 population  174504898
 9 China        1999 cases          212258
10 China        1999 population 1272915272
11 China        2000 cases          213766
12 China        2000 population 1280428583
```

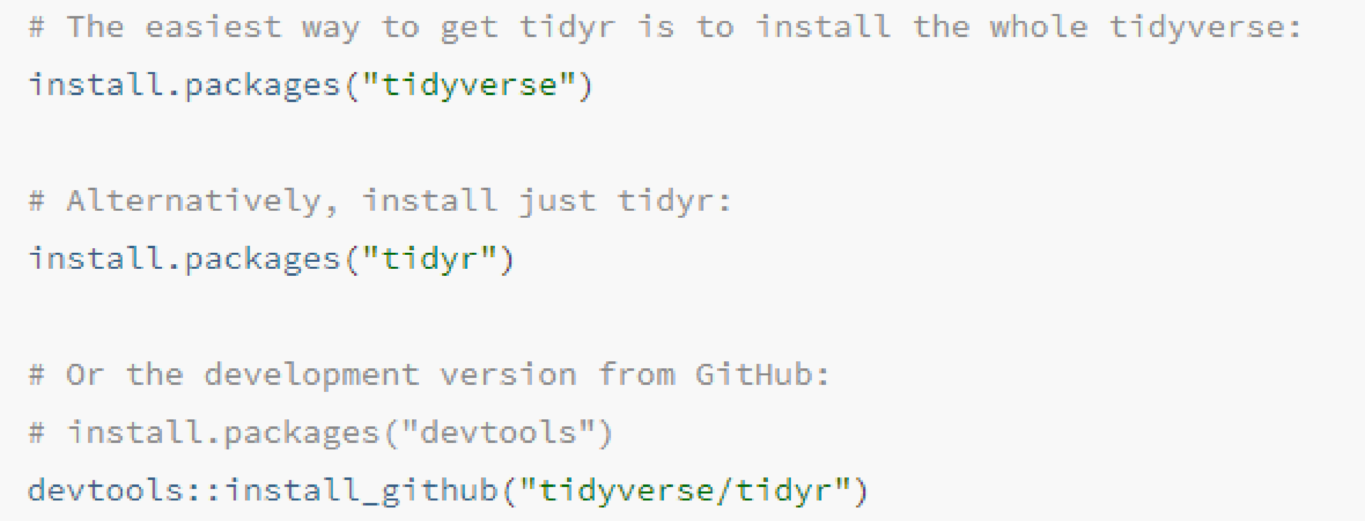


```
> table3
# A tibble: 6 x 3
  country      year rate
* <chr>       <int> <chr>
1 Afghanistan  1999 745/19987071
2 Afghanistan  2000 2666/20595360
3 Brazil       1999 37737/172006362
4 Brazil       2000 80488/174504898
5 China        1999 212258/1272915272
6 China        2000 213766/1280428583
> table4a
# A tibble: 3 x 3
  country     `1999` `2000`
* <chr>        <int>  <int>
1 Afghanistan    745   2666
2 Brazil       37737  80488
3 China       212258 213766
> table4b
# A tibble: 3 x 3
  country         `1999`     `2000`
* <chr>            <int>      <int>
1 Afghanistan   19987071   20595360
2 Brazil       172006362  174504898
3 China       1272915272 1280428583
```

Why tidy? (and not dirty/untidy/messy)

1. There’s a general advantage to picking one consistent way of storing data.

2. Having variables in columns allows to exploit R’s vectorised nature, as most built-in R functions work with vectors of values. That makes transforming tidy data feel particularly natural.

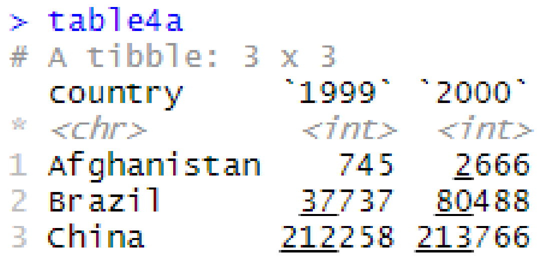


```
# The easiest way to get tidyr is to install the whole tidyverse:
install.packages("tidyverse")

# Alternatively, install just tidyr:
install.packages("tidyr")

# Or the development version from GitHub:
# install.packages("devtools")
devtools::install_github("tidyverse/tidyr")
```

Use the development version of ***tidyr*** to use
pivot_longer() and ***pivot_wider()***

pivot_longer() or ***gather()***

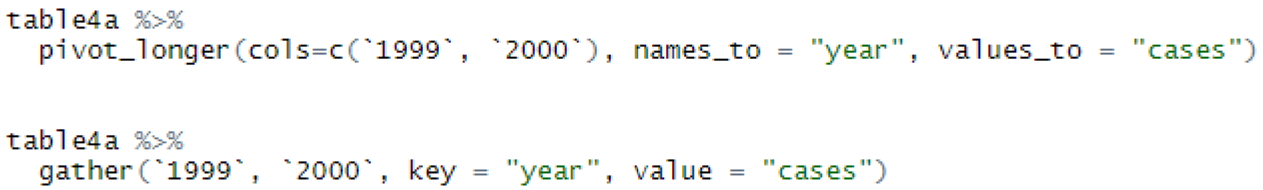
- A common problem is when column names are not **names** of variables, but **values** of a variable.
- Column names **1999** and **2000** represent values of the year variable, and each row represents two observations, not one.



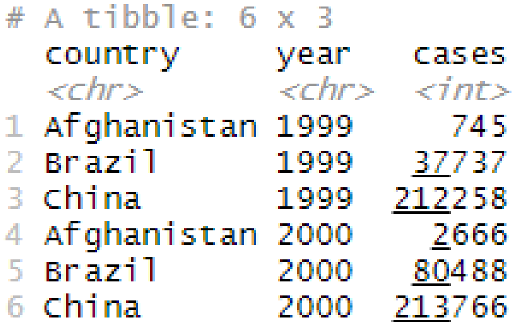
```
> table4a
# A tibble: 3 x 3
  country     `1999` `2000`
* <chr>        <int>  <int>
1 Afghanistan    745   2666
2 Brazil       37737  80488
3 China       212258 213766
```

We need to gather those columns into a new pair of variables. We need to map the names and the values.

1. The set of columns that represent values, not variables. In this example, those are the columns ***1999*** and ***2000***.
2. The name, or key, of the variable whose values will form the column names: **year.**
3. The name of the variable whose values are spread over the cells: ***cases***



```
table4a %>%
  pivot_longer(cols=c(`1999`, `2000`), names_to = "year", values_to = "cases")


table4a %>%
  gather(`1999`, `2000`, key = "year", value = "cases")
```



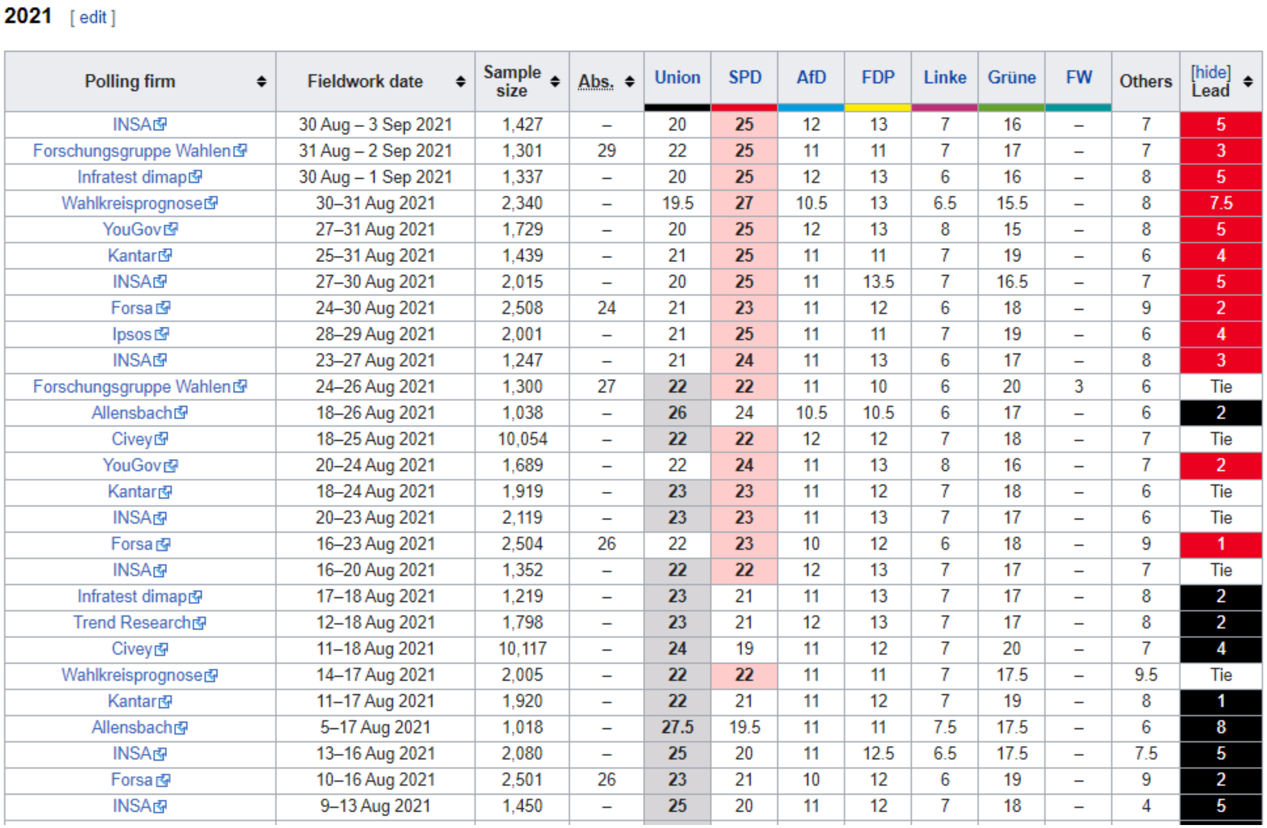
```
# A tibble: 6 x 3
  country     year   cases
  <chr>       <chr>  <int>
1 Afghanistan 1999     745
2 Brazil      1999   37737
3 China       1999  212258
4 Afghanistan 2000    2666
5 Brazil      2000   80488
6 China       2000  213766
```

Wide and long data

There are three rules which make a dataframe tidy:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

Is this data in tidy format? What are the variables and observations?



2021 [edit]

Polling firm	Fieldwork date	Sample size	Abs.	Union	SPD	AfD	FDP	Linke	Grüne	FW	Others	[hide] Lead
INSA	30 Aug – 3 Sep 2021	1,427	–	20	**25**	12	13	7	16	–	7	5
Forschungsgruppe Wahlen	31 Aug – 2 Sep 2021	1,301	29	22	**25**	11	11	7	17	–	7	3
Infratest dimap	30 Aug – 1 Sep 2021	1,337	–	20	**25**	12	13	6	16	–	8	5
Wahlkreisprognose	30–31 Aug 2021	2,340	–	19.5	**27**	10.5	13	6.5	15.5	–	8	7.5
YouGov	27–31 Aug 2021	1,729	–	20	**25**	12	13	8	15	–	8	5
Kantar	25–31 Aug 2021	1,439	–	21	**25**	11	11	7	19	–	6	4
INSA	27–30 Aug 2021	2,015	–	20	**25**	11	13.5	7	16.5	–	7	5
Forsa	24–30 Aug 2021	2,508	24	21	**23**	11	12	6	18	–	9	2
Ipsos	28–29 Aug 2021	2,001	–	21	**25**	11	11	7	19	–	6	4
INSA	23–27 Aug 2021	1,247	–	21	**24**	11	13	6	17	–	8	3
Forschungsgruppe Wahlen	24–26 Aug 2021	1,300	27	**22**	**22**	11	10	6	20	3	6	Tie
Allensbach	18–26 Aug 2021	1,038	–	**26**	24	10.5	10.5	6	17	–	6	2
Civey	18–25 Aug 2021	10,054	–	**22**	**22**	12	12	7	18	–	7	Tie
YouGov	20–24 Aug 2021	1,689	–	22	**24**	11	13	8	16	–	7	2
Kantar	18–24 Aug 2021	1,919	–	**23**	**23**	11	12	7	18	–	6	Tie
INSA	20–23 Aug 2021	2,119	–	**23**	**23**	11	13	7	17	–	6	Tie
Forsa	16–23 Aug 2021	2,504	26	22	**23**	10	12	6	18	–	9	1
INSA	16–20 Aug 2021	1,352	–	**22**	**22**	12	13	7	17	–	7	Tie
Infratest dimap	17–18 Aug 2021	1,219	–	**23**	21	11	13	7	17	–	8	2
Trend Research	12–18 Aug 2021	1,798	–	**23**	21	12	13	7	17	–	8	2
Civey	11–18 Aug 2021	10,117	–	**24**	19	11	12	7	20	–	7	4
Wahlkreisprognose	14–17 Aug 2021	2,005	–	**22**	**22**	11	11	7	17.5	–	9.5	Tie
Kantar	11–17 Aug 2021	1,920	–	**22**	21	11	12	7	19	–	8	1
Allensbach	5–17 Aug 2021	1,018	–	**27.5**	19.5	11	11	7.5	17.5	–	6	8
INSA	13–16 Aug 2021	2,080	–	**25**	20	11	12.5	6.5	17.5	–	7.5	5
Forsa	10–16 Aug 2021	2,501	26	**23**	21	10	12	6	19	–	9	2
INSA	9–13 Aug 2021	1,450	–	**25**	20	11	12	7	18	–	4	5

Wide and long data (1/3)

There are three rules which make a dataframe tidy:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

Is this data in tidy format? What are the variables and observations?

student_id	final_exam	midterm	group_project
2457625	79	68	71
1758293	92	73	67
1622247	71	87	74

Work with your neighbour and sketch out on a piece of paper what this data would look like in tidy format. Some hints:

1. What are the variables?
2. What variable pairings are of interest?
3. Is each observation in its own row ?

Wide and long data (2/3)

Dataframe is not tidy because the variable ***assignment*** is spread across multiple columns:

1. What are the variables? ***student_id, assignment, score***
2. What variable pairings are of interest? ***assignment-score***

student_id	final_exam	midterm	group_project
2457625	79	68	71
1758293	92	73	67
1622247	71	87	74

1. The names of the columns that represent values, not variables. Here, those are ***final_exam, midterm,*** and ***group_project*** which are values of the variable ***assignment***
2. The ***key***, or the names of the variable whose values will form the column names. Here that is ***assignment***.
3. The ***value***, or the name of the variable whose values are spread over the cells. Here that is ***score***.

student_id	assignment	score
2457625	final_exam	79
1758293	final_exam	92
1622247	final_exam	71
2457625	midterm	68
1758293	midterm	73
1622247	midterm	87
2457625	group_project	71
1758293	group_project	67
1622247	group_project	74

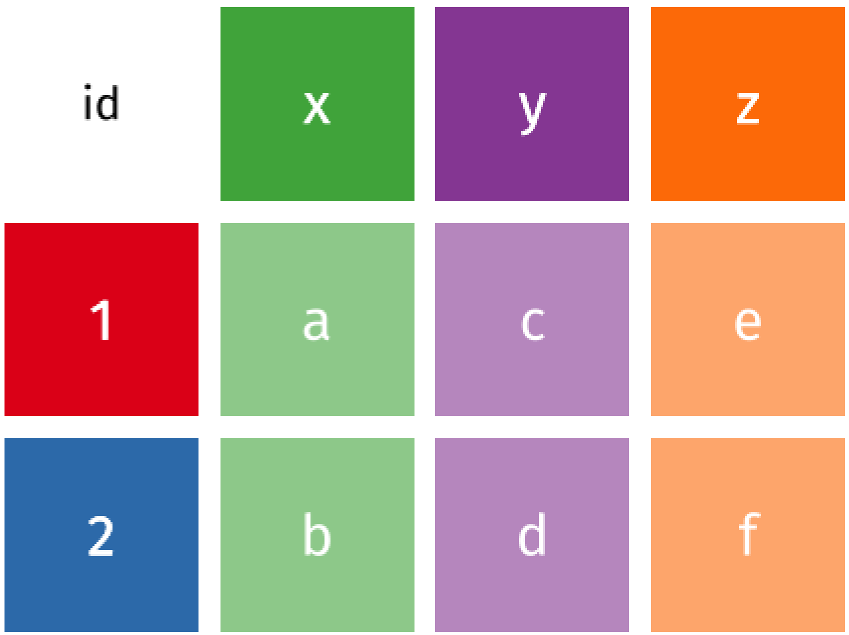
Wide and long data (3/3)

We often need to reshape our datasets and we have a way to go:

- from wide format to long (tidy) format using ***pivot_longer()*** or ***gather()***
- from long (tidy) to wide format using ***pivot_wider()*** or ***spread()***

```
pivot_longer(cols=c('final_exam', 'midterm', 'group_project'), names_to = "assignment", values_to = "score")

gather('final_exam', 'midterm', 'group_project', key = assignment, value = score)
```

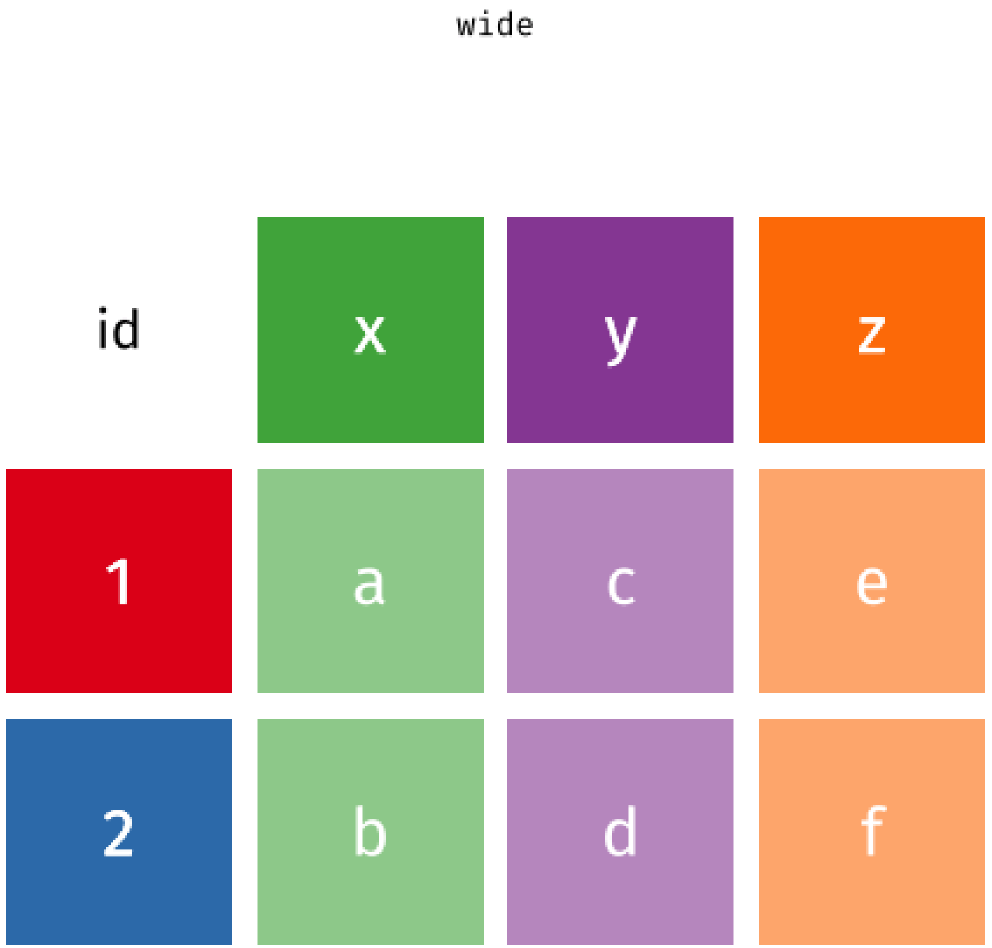
wide



id	x	y	z
1	a	c	e
2	b	d	f

https://github.com/gadenbuie/tidyexplain#tidy-d

Tidying data



wide

id	x	y	z
1	a	c	e
2	b	d	f

```
pivot_longer(cols=c('final_exam', 'midterm', 'group_project'), names_to = "assignment", values_to = "score")

gather('final_exam', 'midterm', 'group_project', key = assignment, value = score)

pivot_wider(names_from = assignment, values_from = score)

spread(key = assignment, value = score)
```

https://github.com/gadenbuie/tidyexplain#tidy-data

Worked Examples: Live Coding