These materials adapted by Amelia McNamara from the RStudio CC BY-SA materials Introduction to R (2014) and Master the Tidyverse (2017).

Introduction to R & RStudio:

deck 09: Tidying data

Amelia McNamara

Visiting Assistant Professor of Statistical and Data Sciences

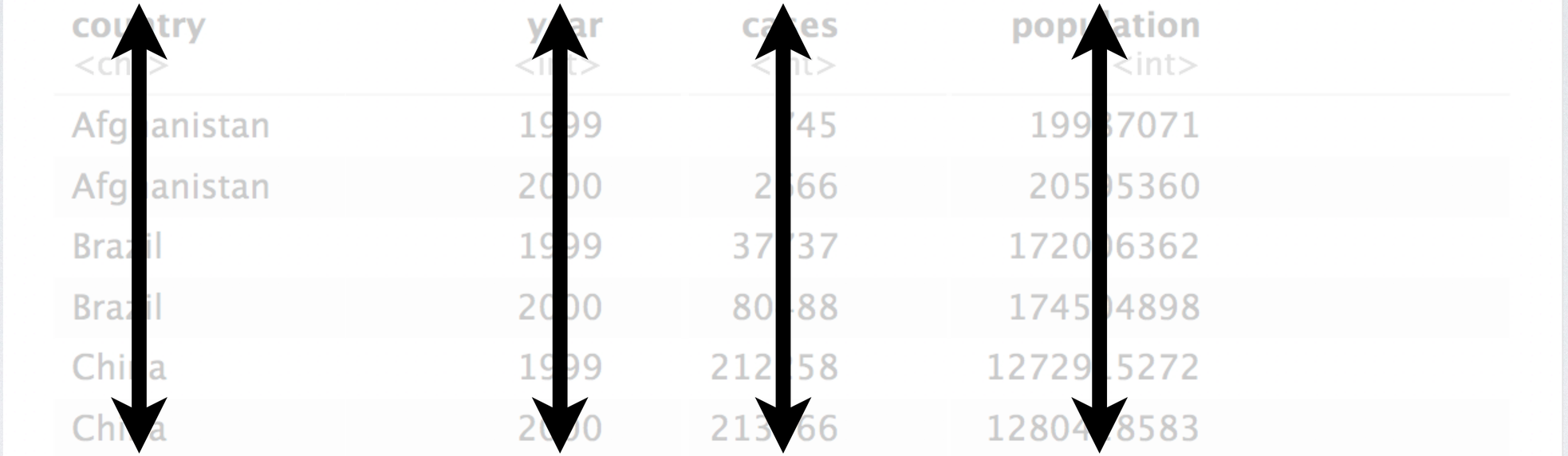
Smith College

January 2018

Quiz

What are the variables in this data set?

```
table1
```



country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

6 rows

Quiz

What are the variables in this data set?

```
table2
```

country	year	type	count
<chr>	<int>	<chr>	<int>
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272

1–10 of 12 rows

Previous 1 2 Next

```
table3
```

	country <chr>	year <int>	rate <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

6 rows

```
table4a
table4b
```

	country <chr>	1999 <int>	2000 <int>
1	Afghanistan	745	2666
2	Brazil	37737	80488
3	China	212258	213766

3 rows

	country <chr>	1999 <int>	2000 <int>
1	Afghanistan	19987071	20595360
2	Brazil	172006362	174504898
3	China	1272915272	1280428583

3 rows

```
table5
```

	country <chr>	century <chr>	year <chr>	rate <chr>
1	Afghanistan	19	99	745/19987071
2	Afghanistan	20	00	2666/20595360
3	Brazil	19	99	37737/172006362
4	Brazil	20	00	80488/174504898
5	China	19	99	212258/1272915272
6	China	20	00	213766/1280428583

6 rows

"Data comes in many formats, but R prefers just one: tidy data. "

- Garrett Grolemund

Tidy data

A data set is **tidy** iff:

1. Each **variable** is in its own **column**
2. Each **case** is in its own **row**
3. Each **value** is in its own **cell**

country <chr>	year <int>	cases <int>	population <int>
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

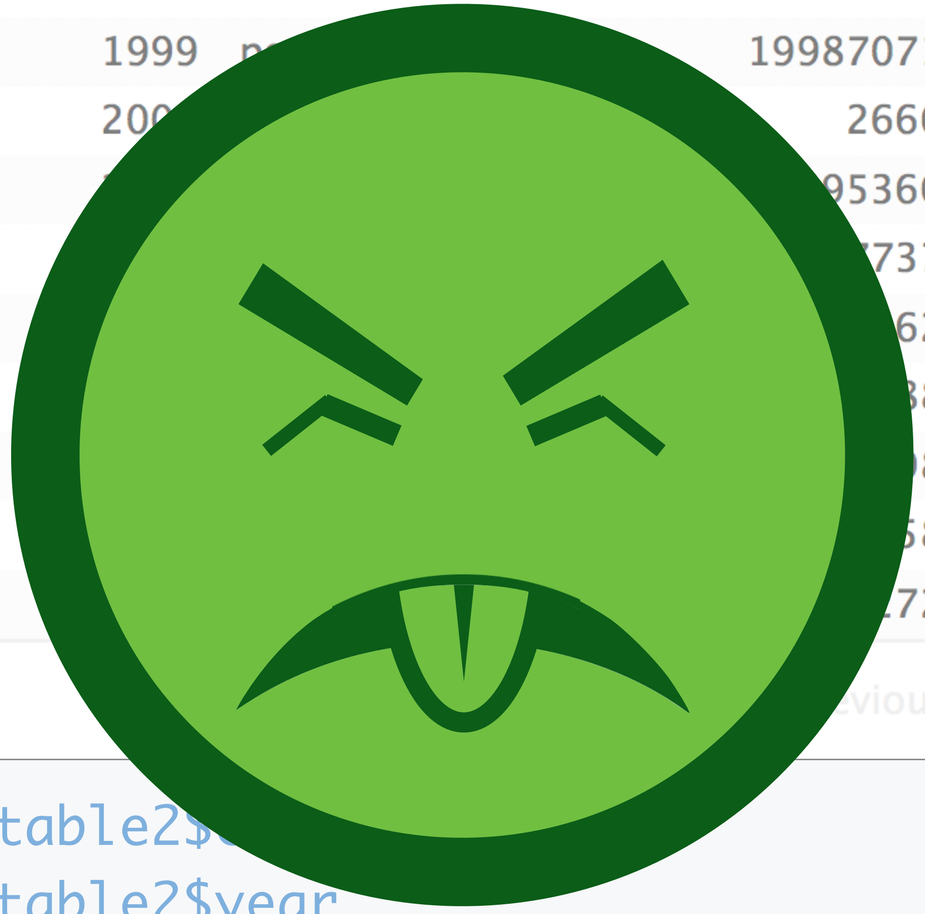
6 rows

```
table1$country
table1$year
table1$cases
table1$population
```

country <chr>	year <int>	type <chr>	count <int>
Afghanistan	1999	cases	745
Afghanistan	1999	p	19987071
Afghanistan	200		2666
Afghanistan			95360
Brazil			737
Brazil			62
Brazil			8
Brazil			8
China			58
China			72

1–10 of 12 rows

vious 1 2 Next



```
table2$
table2$year
table2$count[c(1,3,5,7,9,11)]
table2$count[c(2,4,6,8,10,12)]
```

country <chr>	year <int>	cases <int>	population <int>	rate <dbl>
Afghanistan	1999	745	19987071	0.0000372741
Afghanistan	2000	2666	20595360	0.0001294466
Brazil	1999	37737	172006362	0.0002193930
Brazil	2000	80488	174504898	0.0004612363
China	1999	212258	1272915272	0.0001667495
China	2000	213766	1280428583	0.0001669488

6 rows

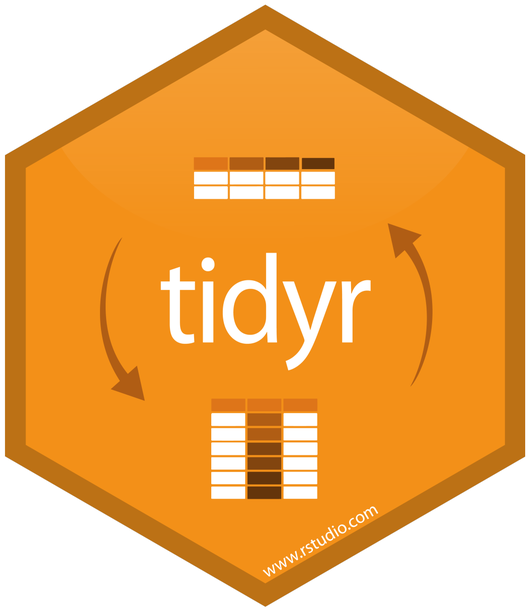
```
table1$cases / table1$population -> table1$rate
```

"Tidy data sets are all alike; but every messy data set is messy in its own way."

- Hadley Wickham

tidyr

tidyr

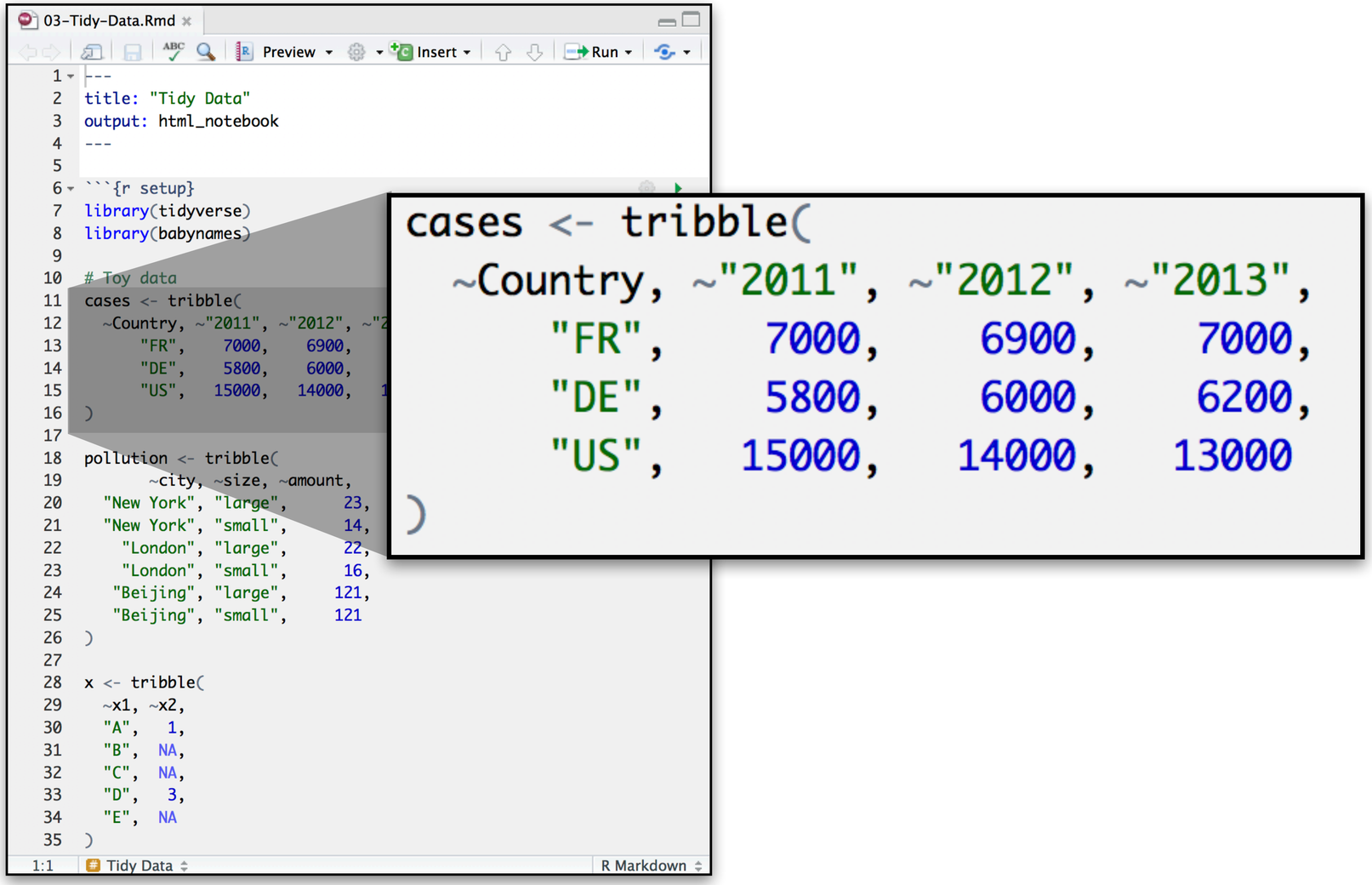


A package that reshapes the layout of tabular data.

Two functions to know: `gather()` and `spread()`

gather()

Toy data


03-Tidy-Data.Rmd
Preview
Insert
Run
---
title: "Tidy Data"
output: html_notebook

```{r setup}
library(tidyverse)
library(babynames)
# Toy data
cases <- tribble(
~Country, ~"2011", ~"2012", ~"2
"FR", 7000, 6900,
"DE", 5800, 6000,
"US", 15000, 14000, 1
)
pollution <- tribble(
~city, ~size, ~amount,
"New York", "large", 23,
"New York", "small", 14,
"London", "large", 22,
"London", "small", 16,
"Beijing", "large", 121,
"Beijing", "small", 121
)
x <- tribble(
~x1, ~x2,
"A", 1,
"B", NA,
"C", NA,
"D", 3,
"E", NA
)
1:1
Tidy Data
R Markdown
cases <- tribble(
~Country, ~"2011", ~"2012", ~"2013",
"FR", 7000, 6900, 7000,
"DE", 5800, 6000, 6200,
"US", 15000, 14000, 13000
)
```

Quiz

What are the variables in cases?

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Quiz

What are the variables in cases?

Country			
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

- Country
- Year
- Count

Your Turn 1

On a sheet of paper, draw how the cases data set would look if it had the same values grouped into three columns: *country*, *year*, *n*

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000

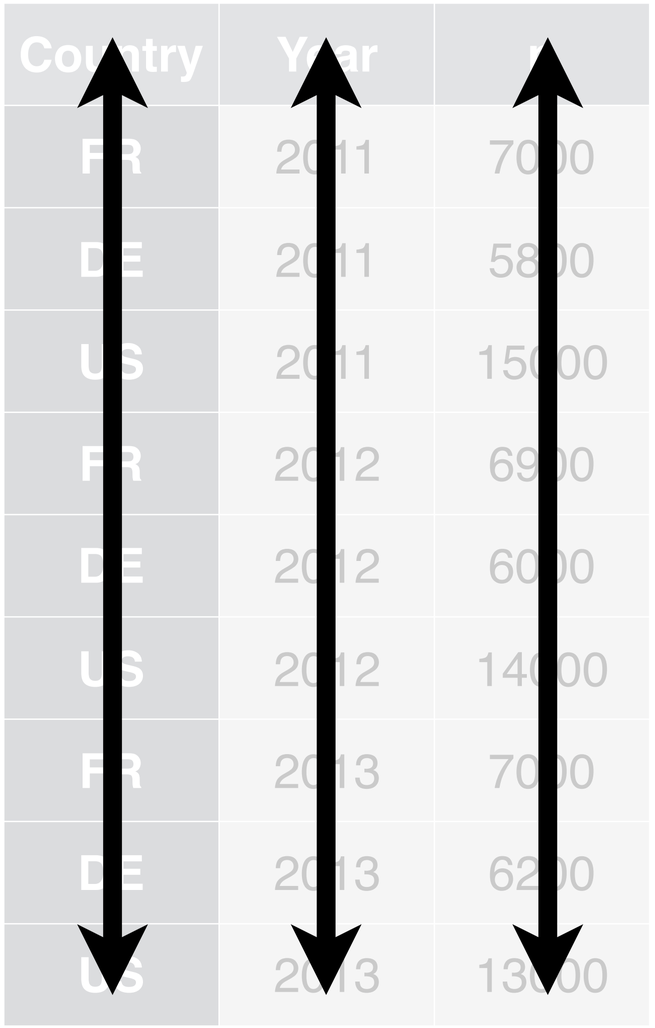
Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200

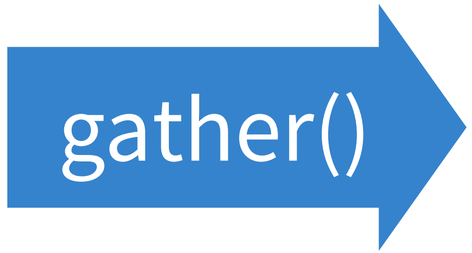
Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000



Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000



Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

1 2

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

key (former column names)

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

key value (former cells)

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

gather()

```
cases %>% gather(key = "year", value = "n", 2:4)
```

data frame to reshape

name of the new key column (a character string)

name of the new value column (a character string)

numeric indexes of columns to collapse (or names)

gather()

```
cases %>% gather("year","n", 2:4)
```

numeric indexes

	2	3	4
Country <chr>	**2011** <dbl>	**2012** <dbl>	**2013** <dbl>
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

gather()

```
cases %>% gather("year","n", "2011", "2012", "2013")
```

names

	2011	2012	2013
Country <chr>	**2011** <dbl>	**2012** <dbl>	**2013** <dbl>
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

gather()

```
cases %>% gather("year","n", -Country)
```

Everything except...

	Not Country	Not Country	Not Country
Country <chr>	**2011** <dbl>	**2012** <dbl>	**2013** <dbl>
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

Your Turn 2

Use **gather()** to reorganize **table4a** into three columns: *country*, *year*, and *cases*.

	country <chr>	1999 <int>	2000 <int>
1	Afghanistan	745	2666
2	Brazil	37737	80488
3	China	212258	213766

3 rows

```
table4a %>%
  gather(key = "year", value = "n", 2:3)
```

country <chr>	year <chr>	n <int>
Afghanistan	1999	745
Brazil	1999	37737
China	1999	212258
Afghanistan	2000	2666
Brazil	2000	80488
China	2000	213766

6 rows

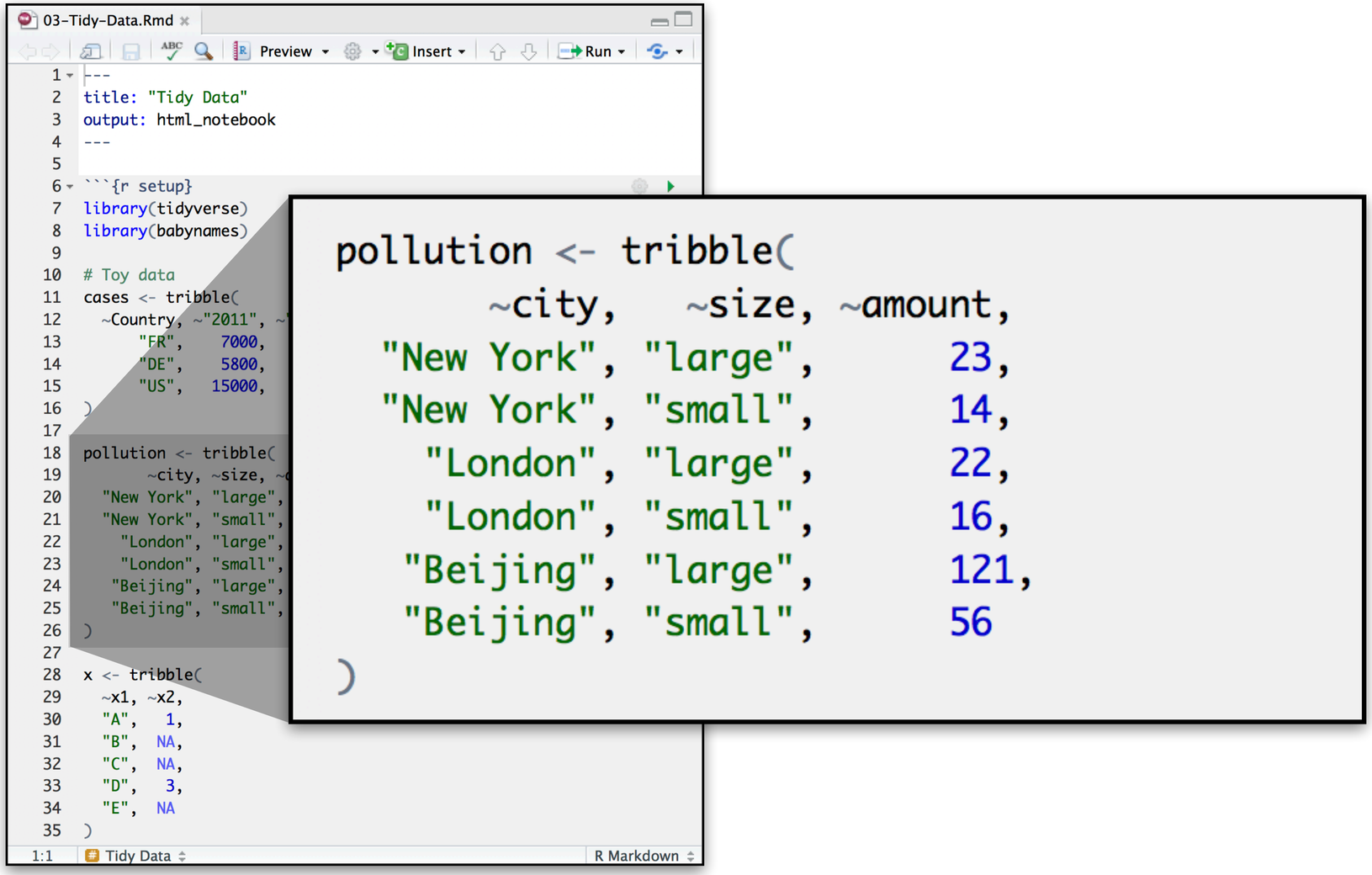
```
table4a %>%
  gather(key = "year", value = "n", 2:3, convert = TRUE)
```

country <chr>	year <int>	n <int>
Afghanistan	1999	745
Brazil	1999	37737
China	1999	212258
Afghanistan	2000	2666
Brazil	2000	80488
China	2000	213766

6 rows

spread()

Toy data


03-Tidy-Data.Rmd
Preview
Insert
Run
---
title: "Tidy Data"
output: html_notebook

```{r setup}
library(tidyverse)
library(babynames)
# Toy data
cases <- tribble(
~Country, ~"2011", ~
"FR", 7000,
"DE", 5800,
"US", 15000,
)
pollution <- tribble(
~city, ~size, ~
"New York", "large",
"New York", "small",
"London", "large",
"London", "small",
"Beijing", "large",
"Beijing", "small",
)
x <- tribble(
~x1, ~x2,
"A", 1,
"B", NA,
"C", NA,
"D", 3,
"E", NA
)
1:1
Tidy Data
R Markdown
pollution <- tribble(
~city, ~size, ~amount,
"New York", "large", 23,
"New York", "small", 14,
"London", "large", 22,
"London", "small", 16,
"Beijing", "large", 121,
"Beijing", "small", 56
)
```

Quiz

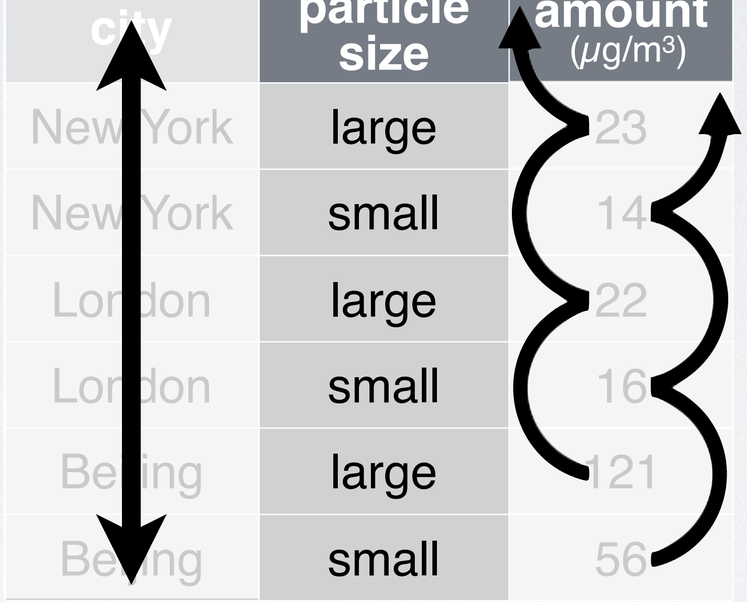
What are the variables in pollution?

city	particle size	amount ($\mu g/m^3$)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

Quiz

What are the variables in pollution?

city	particle size	amount ($\mu g/m^3$)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56



- City
- Amount of large particulate
- Amount of small particulate

Your Turn 3

On a sheet of paper, draw how this data set would look if it had the same values grouped into three columns: *city*, *large*, *small*

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14
London	22	

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14
London	22	16

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

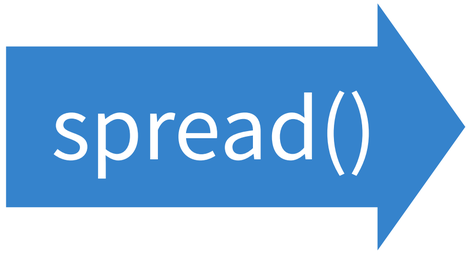
city	large	small
New York	23	14
London	22	16
Beijing	121	

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14
London	22	16
Beijing	121	56

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56



city	large	small
New York	23	14
London	22	16
Beijing	121	56

	1	2
city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14
London	22	16
Beijing	121	56

key (new column names)

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14
London	22	16
Beijing	121	56

key **value** (new cells)

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	large	small
New York	23	14
London	22	16
Beijing	121	56

spread()

```
pollution %>% spread(key = size, value = amount)
```

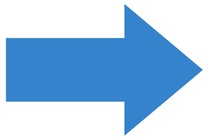
data frame to reshape

column to use for keys (becomes new column names)

column to use for values (becomes new column cells)

```
pollution %>% spread(size, amount)
```

	city	size	amount
1	New York	large	23
2	New York	small	14
3	London	large	22
4	London	small	16
5	Beijing	large	121
6	Beijing	small	56



	city	large	small
1	Beijing	121	56
2	London	22	16
3	New York	23	14

Your Turn 4

Use **spread()** to reorganize **table2** into four columns: *country*, *year*, *cases*, and *population*.

country <chr>	year <int>	type <chr>	count <int>
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362

```
table2 %>%
  spread(key = type, value = count)
```

	country <chr>	year <int>	cases <int>	population <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

6 rows

who

(Untidy Data)

who

Tuberculosis (TB) cases broken down by year, country, age, gender, and diagnosis method from the *2014 World Health Organization Global Tuberculosis Report*

```
View(who)
```

~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse - RStudio Source Editor

who

Filter

	country	iso2	iso3	year	new_sp_m014	new_sp_m1524	new_sp_m2534	new_sp_m3544	new_sp_m4554	new_sp_m5564	new_sp_m65	new_sp_f014	new_sp_f1524	new_sp_f2534	new_sp_f3544
1	Afghanistan	AF	AFG	1980	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
2	Afghanistan	AF	AFG	1981	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
3	Afghanistan	AF	AFG	1982	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
4	Afghanistan	AF	AFG	1983	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
5	Afghanistan	AF	AFG	1984	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
6	Afghanistan	AF	AFG	1985	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
7	Afghanistan	AF	AFG	1986	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
8	Afghanistan	AF	AFG	1987								NA	NA	NA	NA
9	Afghanistan	AF	AFG	1988								NA	NA	NA	NA
10	Afghanistan	AF	AFG	1989								NA	NA	NA	NA
11	Afghanistan	AF	AFG	1990								NA	NA	NA	NA
12	Afghanistan	AF	AFG	1991								NA	NA	NA	NA
13	Afghanistan	AF	AFG	1992								NA	NA	NA	NA
14	Afghanistan	AF	AFG	1993								NA	NA	NA	NA
15	Afghanistan	AF	AFG	1994	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
16	Afghanistan	AF	AFG	1995	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
17	Afghanistan	AF	AFG	1996	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
18	Afghanistan	AF	AFG	1997	0	10	6	3	5	2	0	5	38	36	14
19	Afghanistan	AF	AFG	1998	30	129	128	90	89	64	41	45	350	419	194
20	Afghanistan	AF	AFG	1999	8	55	55	47	34	21	8	25	139	160	110
21	Afghanistan	AF	AFG	2000	52	228	183	149	129	94	80	93	414	565	339
22	Afghanistan	AF	AFG	2001	129	379	349	274	204	139	103	146	799	888	586

Showing 1 to 22 of 7,240 entries

What variables does this data set contain?

who variables

country	iso2	iso3	year	new_sp_m014

country, **iso2**, **iso3** - country identifiers

year - year

other columns names - encode **type** of TB case, **sex**, and **age**

who codes

Type of TB case

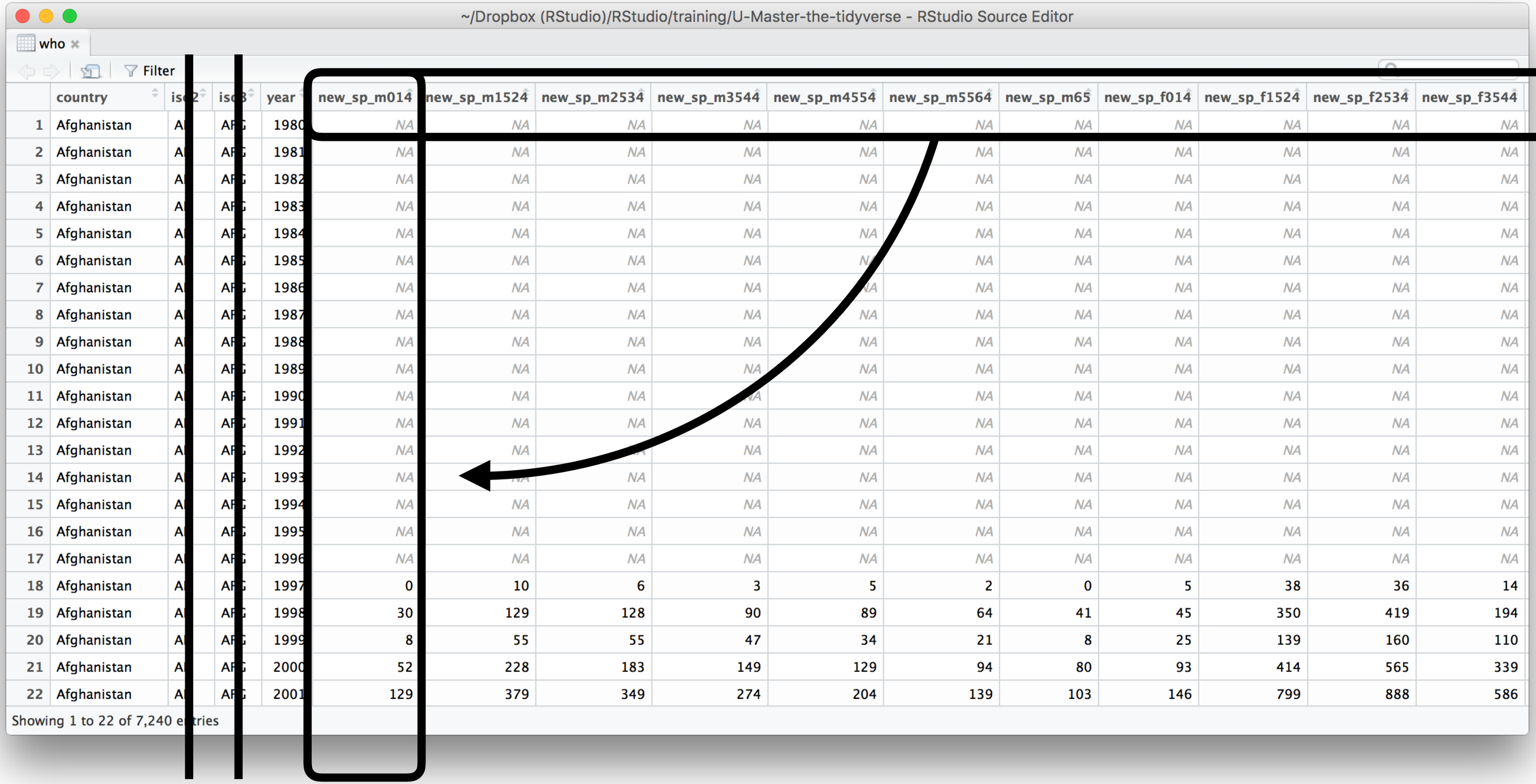
- **rel** - relapse
- **ep** - extra-pulmonary
- **sn**- pulmonary, smear negative
- **sp** -pulmonary, smear positive

Gender

- **m** - male
- **f** - female

Age group

- **014** - 0 to 14 years old
- **1524** - 15 to 24 years old
- **2534** - 25 to 34 years old
- **3544** - 35 to 44 years old
- **4554** - 45 to 54 years old
- **5564** - 55 to 64 years old
- **65** - 65 and older



	country	iso2	iso3	year	new_sp_m014	new_sp_m1524	new_sp_m2534	new_sp_m3544	new_sp_m4554	new_sp_m5564	new_sp_m65	new_sp_f014	new_sp_f1524	new_sp_f2534	new_sp_f3544
1	Afghanistan	AF	AFG	1980	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
2	Afghanistan	AF	AFG	1981	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
3	Afghanistan	AF	AFG	1982	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
4	Afghanistan	AF	AFG	1983	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
5	Afghanistan	AF	AFG	1984	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
6	Afghanistan	AF	AFG	1985	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
7	Afghanistan	AF	AFG	1986	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
8	Afghanistan	AF	AFG	1987	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
9	Afghanistan	AF	AFG	1988	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
10	Afghanistan	AF	AFG	1989	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
11	Afghanistan	AF	AFG	1990	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
12	Afghanistan	AF	AFG	1991	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
13	Afghanistan	AF	AFG	1992	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
14	Afghanistan	AF	AFG	1993	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
15	Afghanistan	AF	AFG	1994	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
16	Afghanistan	AF	AFG	1995	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
17	Afghanistan	AF	AFG	1996	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
18	Afghanistan	AF	AFG	1997	0	10	6	3	5	2	0	5	38	36	14
19	Afghanistan	AF	AFG	1998	30	129	128	90	89	64	41	45	350	419	194
20	Afghanistan	AF	AFG	1999	8	55	55	47	34	21	8	25	139	160	110
21	Afghanistan	AF	AFG	2000	52	228	183	149	129	94	80	93	414	565	339
22	Afghanistan	AF	AFG	2001	129	379	349	274	204	139	103	146	799	888	586

Showing 1 to 22 of 7,240 entries

Your Turn 5

Gather the **5th through 60th** columns of who into a pair of key:value columns named ***codes*** and ***n***.

Then select just the **county**, **year**, **codes** and **n** variables.

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3)
```

country <chr>	year <int>	codes <chr>	n <int>
Afghanistan	1980	new_sp_m014	*NA*
Afghanistan	1981	new_sp_m014	*NA*
Afghanistan	1982	new_sp_m014	*NA*
Afghanistan	1983	new_sp_m014	*NA*
Afghanistan	1984	new_sp_m014	*NA*
Afghanistan	1985	new_sp_m014	*NA*
Afghanistan	1986	new_sp_m014	*NA*
Afghanistan	1987	new_sp_m014	*NA*
Afghanistan	1988	new_sp_m014	*NA*
Afghanistan	1989	new_sp_m014	*NA*

separate()

separate()

Splits a column by dividing values at a specific character.

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes, into = c("new", "type", "sexage"), sep = "_")
```

a column to split

names of new columns to make

string to split on
(Defaults to any non_alpha-numeric character)

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(                                                    )
```

country <chr>	year <int>	codes <chr>	n <int>
Afghanistan	1980	new_sp_m014	*NA*
Afghanistan	1981	new_sp_m014	*NA*
Afghanistan	1982	new_sp_m014	*NA*
Afghanistan	1983	new_sp_m014	*NA*
Afghanistan	1984	new_sp_m014	*NA*
Afghanistan	1985	new_sp_m014	*NA*
Afghanistan	1986	new_sp_m014	*NA*
Afghanistan	1987	new_sp_m014	*NA*
Afghanistan	1988	new_sp_m014	*NA*
Afghanistan	1989	new_sp_m014	*NA*

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes                                    )
```

country <chr>	year <int>	codes <chr>	n <int>
Afghanistan	1980	new_sp_m014	*NA*
Afghanistan	1981	new_sp_m014	*NA*
Afghanistan	1982	new_sp_m014	*NA*
Afghanistan	1983	new_sp_m014	*NA*
Afghanistan	1984	new_sp_m014	*NA*
Afghanistan	1985	new_sp_m014	*NA*
Afghanistan	1986	new_sp_m014	*NA*
Afghanistan	1987	new_sp_m014	*NA*
Afghanistan	1988	new_sp_m014	*NA*
Afghanistan	1989	new_sp_m014	*NA*

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes, into = c("new", "type", "sexage")                    )
```

country <chr>	year <int>	codes <chr>	new <chr>	type <chr>	sexage <chr>	n <int>
Afghanistan	1980	new_sp_m014				*NA*
Afghanistan	1981	new_sp_m014				*NA*
Afghanistan	1982	new_sp_m014				*NA*
Afghanistan	1983	new_sp_m014				*NA*
Afghanistan	1984	new_sp_m014				*NA*
Afghanistan	1985	new_sp_m014				*NA*
Afghanistan	1986	new_sp_m014				*NA*
Afghanistan	1987	new_sp_m014				*NA*
Afghanistan	1988	new_sp_m014				*NA*
Afghanistan	1989	new_sp_m014				*NA*

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes, into = c("new", "type", "sexage"), sep = "_")
```

country <chr>	year <int>	codes <chr>	new <chr>	type <chr>	sexage <chr>	n <int>
Afghanistan	1980	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1981	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1982	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1983	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1984	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1985	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1986	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1987	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1988	new_sp_m014	new	sp	m014	*NA*
Afghanistan	1989	new_sp_m014	new	sp	m014	*NA*

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes, c("new", "type", "sexage"), sep = "_") %>%
  select(-new)
```

country <chr>	year <int>	type <chr>	sexage <chr>	n <int>
Afghanistan	1980	sp	m014	NA
Afghanistan	1981	sp	m014	NA
Afghanistan	1982	sp	m014	NA
Afghanistan	1983	sp	m014	NA
Afghanistan	1984	sp	m014	NA
Afghanistan	1985	sp	m014	NA
Afghanistan	1986	sp	m014	NA
Afghanistan	1987	sp	m014	NA

separate()

Splits a column by dividing values at a specific character.

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes, c("new", "type", "sexage"), sep = c(4, 7))
```

locations to split at
(Split after 4th and 7th characters)

Your Turn 6

Separate the sexage column into sex and age columns.

```
who %>%
  gather("codes", "n", 5:60) %>%
  select(-iso2, -iso3) %>%
  separate(codes, c("new", "type", "sexage"), sep = "_") %>%
  select(-new) %>%
  separate(sexage, into = c("sex", "age"), sep = 1)
```

	country <chr>	year <int>	type <chr>	sex <chr>	age <chr>	n <int>
1	Afghanistan	1980	sp	m0	14	*NA*
2	Afghanistan	1981	sp	m0	14	*NA*
3	Afghanistan	1982	sp	m0	14	*NA*
4	Afghanistan	1983	sp	m0	14	*NA*
5	Afghanistan	1984	sp	m0	14	*NA*
6	Afghanistan	1985	sp	m0	14	*NA*
7	Afghanistan	1986	sp	m0	14	*NA*

unite()

unite()

Unites columns into single column by combining cells.

```
unite(data, col, …, sep = "")
```

data frame to reshape

name of new column to make (in quotes)

two or more columns to combine

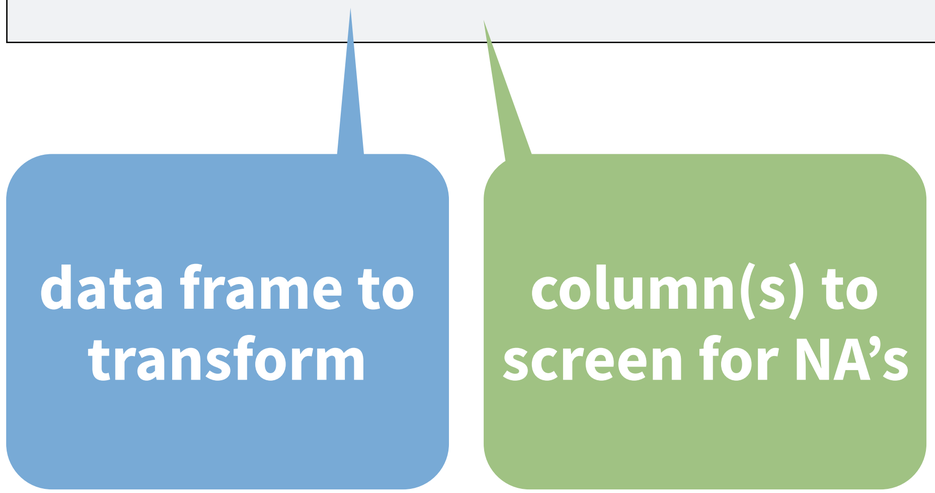
separator to place between elements in new column (Defaults to an underscore)

Missing Values

drop_na()

Drops rows that contain NA's in the specified columns.

```
drop_na(x, x2)
```



drop_na()

Drops rows that contain NA's in the specified columns.

```
drop_na(x, x2)
```

x

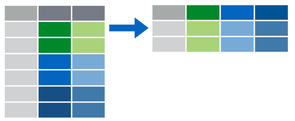
x1	x2
A	1
B	NA
C	NA
D	3
E	NA

x1	x2
A	1
D	3

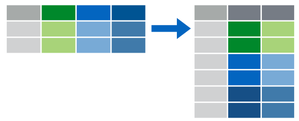
```
who %>%
  gather("codes", "n", 5:60) %>%
  separate(codes, c("new", "type", "sexage"), sep = "_") %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 2) %>%
  drop_na(n)
```

	country	year	type	sex	age	n
1	Afghanistan	1997	sp	m0	14	0
2	Afghanistan	1998	sp	m0	14	30
3	Afghanistan	1999	sp	m0	14	8
4	Afghanistan	2000	sp	m0	14	52
5	Afghanistan	2001	sp	m0	14	129
6	Afghanistan	2002	sp	m0	14	90
7	Afghanistan	2003	sp	m0	14	127
8	Afghanistan	2004	sp	m0	14	139
9	Afghanistan	2005	sp	m0	14	151

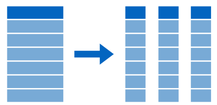
Recap



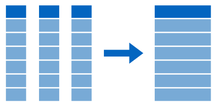
Move values into column names with **spread()**



Move column names into values with **gather()**



Split a column with **separate()** or **separate_rows()**



Unite columns with **unite()**