

Package *tidyr* in R

Introduction

A same data can be represented or captured in multiple ways as shown here.
Also all are not equally easy to use

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

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

country	year	Type	Count
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
China	2000	cases	213766
China	2000	population	1280428583

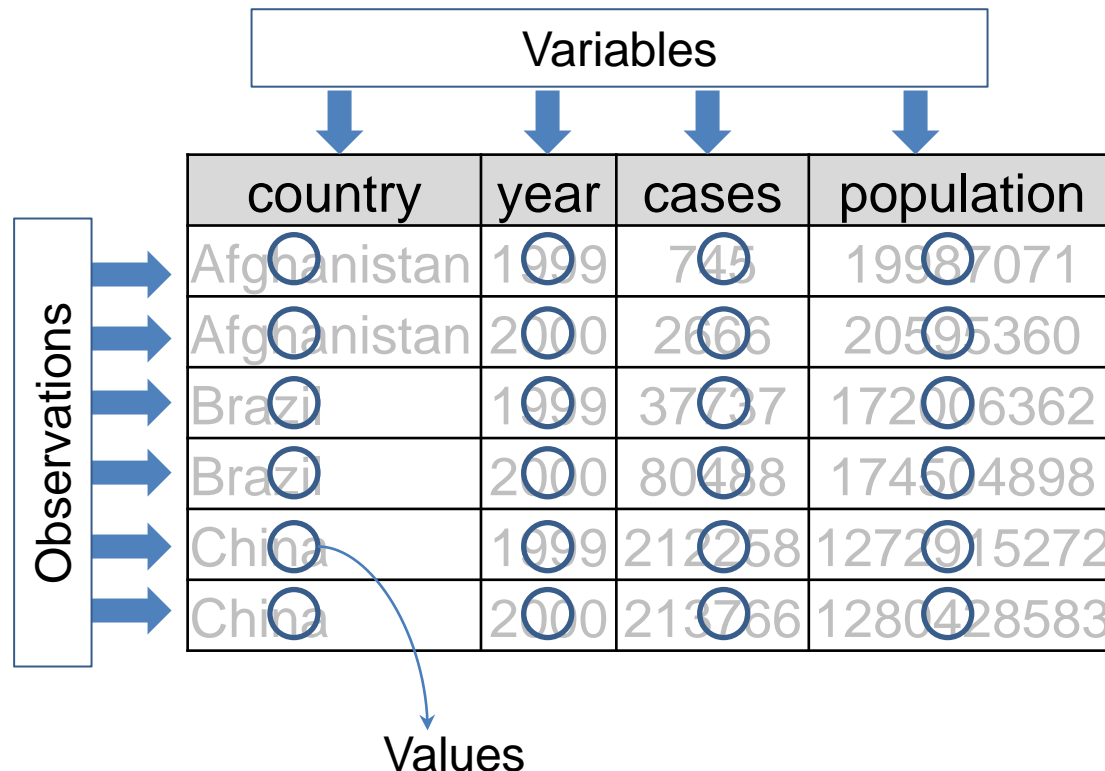
country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

country	1999	2000
Afghanistan	19987071	20595360
Brazil	172006362	174504898
China	1272915272	1280428583

Tidy data

Following are the three interrelated rules which makes a data set tidy

- Each variable must have its own column
- Each observation must have its own row
- Each value must have its own cell



That interrelationship leads to an even simpler set of practical instructions

1. Put each dataset in a tibble
2. Put each variable in a column

Prerequisites

tidyr is a member of the core tidyverse

```
install.packages("tidyverse")  
library(tidyverse)
```

Spreading and gathering

We need to resolve two common problems in the data

1. One variable might be spread across multiple columns
2. One observation might be scattered across multiple rows

1. the column names 1999 and 2000 represent values of the year variable

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

country	year	Type	Count
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
China	2000	cases	213766
China	2000	population	1280428583

2. Observation is scattered across multiple rows. an observation is a country in a year, but each observation is spread across two rows

Gathering

We need to gather where

1. some of the column names are not names of variables, but values of a variable i.e. one variable spread across multiple columns
2. Each row represents two observations, not one

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

column names 1999 and 2000
represent values of the year variable

To **gather** those columns into a new pair of variables. we need three parameters:

1. The set of columns that represent values, not variables.
2. The name of the variable whose values form the column names. It is called the **Key**
3. The name of the variable whose values are spread over the cells. It is called **Value**

Syntax: `gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE)`

Gathering

Syntax: `gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE)`

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



```
> table4a %>% gather(`1999`, `2000`, key= "year", value= "cases")
# A tibble: 6 × 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
```

```
> table4b
# A tibble: 3 × 3
  country `1999` `2000`
  <chr>   <int> <int>
1 Afghanistan 19987071 20595360
2      Brazil 172006362 174504898
3      China 1272915272 1280428583
```



```
> table4b %>% gather(`1999`, `2000`, key= "year", value= "population")
# A tibble: 6 × 3
  country year population
  <chr> <chr>   <int>
1 Afghanistan 1999 19987071
2      Brazil 1999 172006362
3      China 1999 1272915272
4 Afghanistan 2000 20595360
5      Brazil 2000 174504898
6      China 2000 1280428583
```

Note that “1999” and “2000” are non-syntactic names so need to surround them in backticks

Spreading

Spreading is the opposite of gathering. We use it when an observation is scattered across multiple rows. we need two parameters:

1. The column that contains variable names, the **key** column
2. The column that contains values forms multiple variables, the **value** column

Syntax: spread(data, key, value, fill = NA, convert = FALSE, drop = TRUE, sep = NULL)

an observation is a country in a year, but each observation is spread across two rows

```
> 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
```



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

gather() makes wide tables narrower and longer;
spread() makes long tables shorter and wider


Separate

`separate()` pulls apart one column into multiple columns, by splitting wherever a separator character appears. We can use this when one column contains two variables

Syntax: `separate(data, col, into, sep = "[^:alnum:]]+", remove = TRUE, convert = FALSE, ...)`

```
> table3
# A tibble: 6 × 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
```

By default, `separate()` will split values wherever it sees a non-alphanumeric character. We can convert to better types using `convert = TRUE`



```
> table3 %>% separate(rate, into = c("cases", "population"), sep = "/", convert = TRUE)
# A tibble: 6 × 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
```