# STAT 209
# Reshaping and "Tidy" Data

March 9, 2018

Colin Reimer Dawson

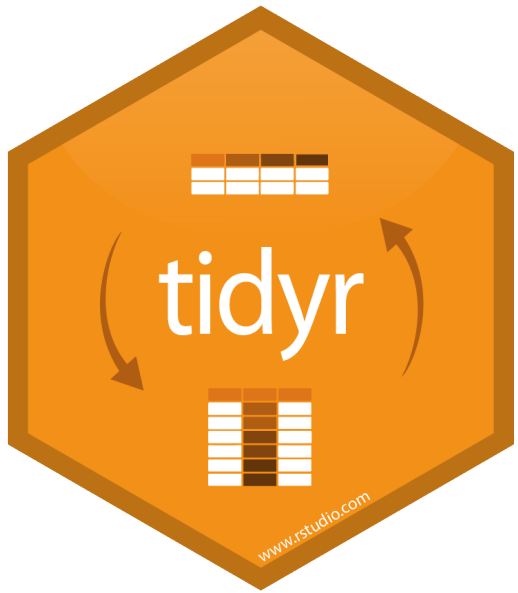
# Today

- Converting data between "wide" format and "long/narrow" (tidy) format
- New verbs:
  - `gather()`
  - `spread()`
- Lab8

# Coming Up...

- Next week: Some CS nuts and bolts (functions, loops) as applied to R

# The tidyr package




Another piece of the "tidyverse". Here is a reference sheet that includes both dplyr and tidyr verbs.

# Outline

# Redundant Data

**Tournament Winners**

| Tournament | Year | Winner | Winner Date of Birth |
|---|---|---|---|
| Indiana Invitational | 1998 | Al Fredrickson | 21 July 1975 |
| Cleveland Open | 1999 | Bob Albertson | 28 September 1968 |
| Des Moines Masters | 1999 | Al Fredrickson | 21 July 1975 |
| Indiana Invitational | 1999 | Chip Masterson | 14 March 1977 |

- Cases here are **tournaments**, but `Winner Date of Birth` tells us about a **person** (the entry in `Winner`)
- It is *perfectly predictable* from the `Winner` column; i.e., it contains redundant information
- (if we know the `Winner` → `DOB` mapping)
- This is inefficient for storage and retrieval

# Removing Redundancy

**Tournament Winners**

| Tournament | Year | Winner |
|---|---|---|
| Indiana Invitational | 1998 | Al Fredrickson |
| Cleveland Open | 1999 | Bob Albertson |
| Des Moines Masters | 1999 | Al Fredrickson |
| Indiana Invitational | 1999 | Chip Masterson |

**Winner Dates of Birth**

| Winner | Date of Birth |
|---|---|
| Chip Masterson | 14 March 1977 |
| Al Fredrickson | 21 July 1975 |
| Bob Albertson | 28 September 1968 |

- Here we separate data into multiple tables
- Tables map case ID (`Tournament`, `Winner`) to pieces of information (variables) about that case
- Results in tables that are long (many rows) but narrow (few columns)
- More efficient database practice
- However, need to be able to join

# Generic Congressional Ballot

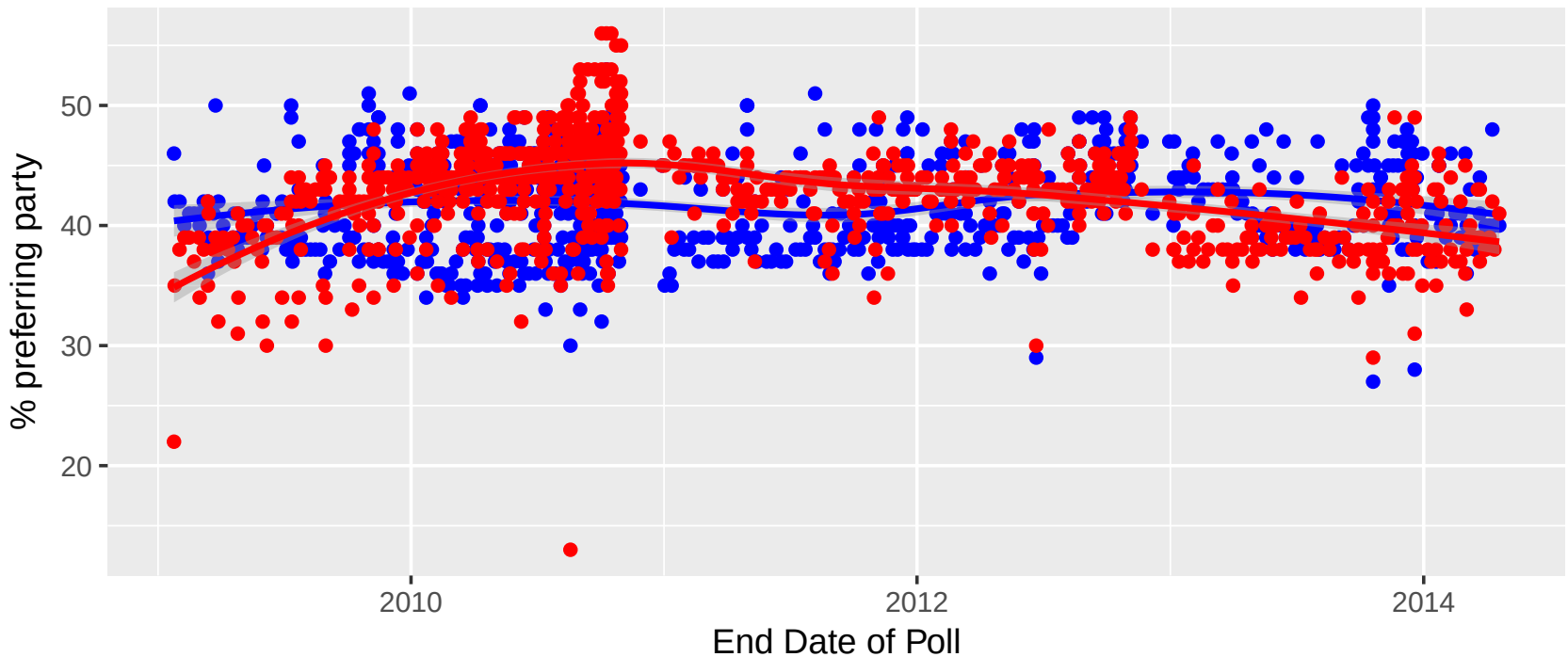
```
library(tidyverse)
url1 <- "https://raw.githubusercontent.com/TheUpshot"
url2 <- "leo-senate-model/master/fundamentals/data"
url3 <- "generic-approval/genericBallot.csv"
url <- paste(url1, url2, url3, sep = "/")
genericBallot <- read_csv(url)
head(genericBallot)
```

```
# A tibble: 6 x 7
   startdate    enddate Democrats Republicans  Sample              Poll
      <date>     <date>     <int>       <int>   <chr>             <chr>
1 2014-04-14 2014-04-20        40          41 3500 LV Rasmussen Reports
2 2014-04-07 2014-04-13        40          38 3500 LV Rasmussen Reports
3 2014-04-07 2014-04-10        48          42 1036 RV  McClatchy/Marist
4 2014-03-31 2014-04-06        40          39 3500 LV Rasmussen Reports
5 2014-03-26 2014-03-31        40          38 1578 RV        Quinnipiac
6 2014-03-24 2014-03-30        39          38 3500 LV Rasmussen Reports
# ... with 1 more variables: knownDate <date>
```

```
genericBallot <- genericBallot %>%  # Obama admin only
    filter(enddate > "2009-01-20" & enddate < "2017-01-20")
```

# Plotting Party Preference Over Time

```
genericBallot %>%
    ggplot(aes(x = enddate)) +
    geom_point(aes(y = Democrats), color = "blue") +
    geom_smooth(aes(y = Democrats), color = "blue") +
    geom_point(aes(y = Republicans), color = "red") +
    geom_smooth(aes(y = Republicans), color = "red") +
    ylab("% preferring party") + xlab("End Date of Poll")
```



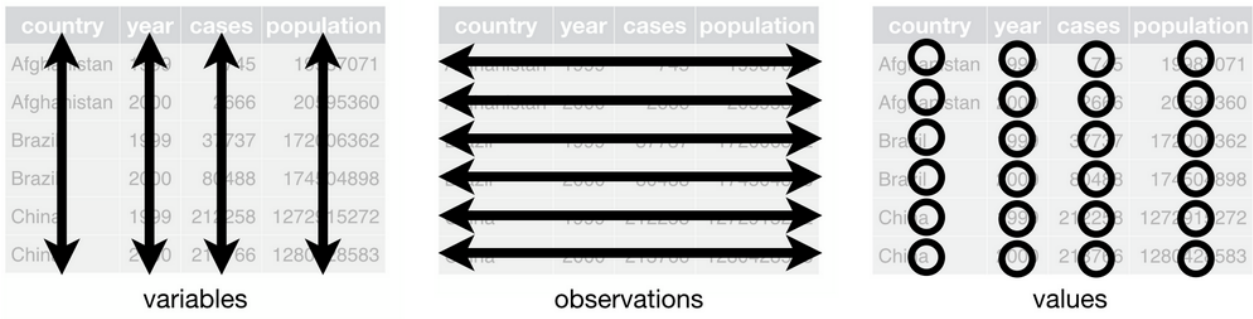
# Tidy Data?

```
table4a
```

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

- What are the variables here?
- Two variables measure the same thing, in different years.
- We can make this data "tidier"

# Tidy Data



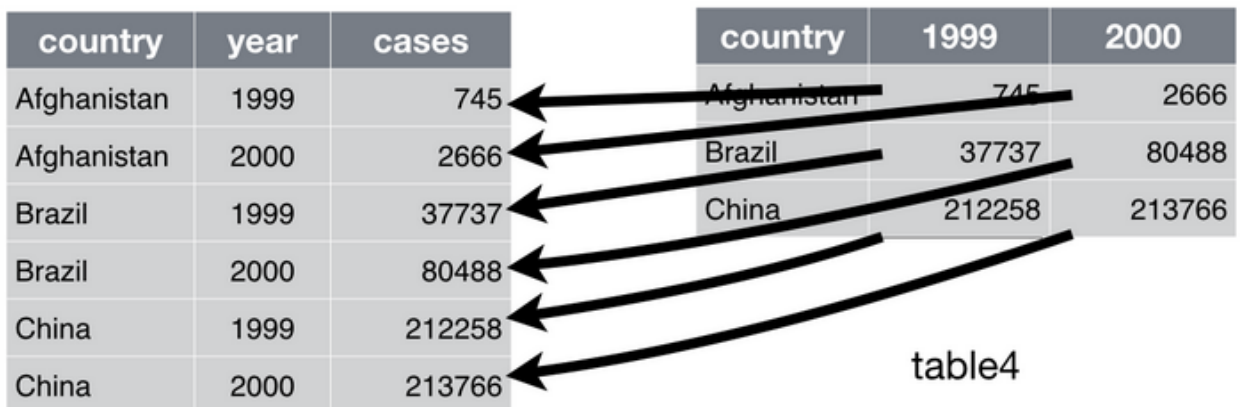

Data is **tidy** when each row represents a distinct "entity", and each variable records a property of that entity.[1]

[1] Your textbook talks about data being "tidy" or "not tidy". While there are certainly formats that *cannot* be considered "tidy", there may not be one unique format which is "tidy": it depends what you consider an entity and a property.

# Outline

# gather(): Convert wide to narrow

```
# we convert '1999' and '2000' to a single "cases" variable
# 'cases' is the new single variable
# 'year' is created to distinguish between the old variables
table4a %>%
    gather(
        key   = "year",  # new variable to distinguish values
        value = "cases", # new name for the values variable
        -country         # all variables contain "cases" except this one
    )
```



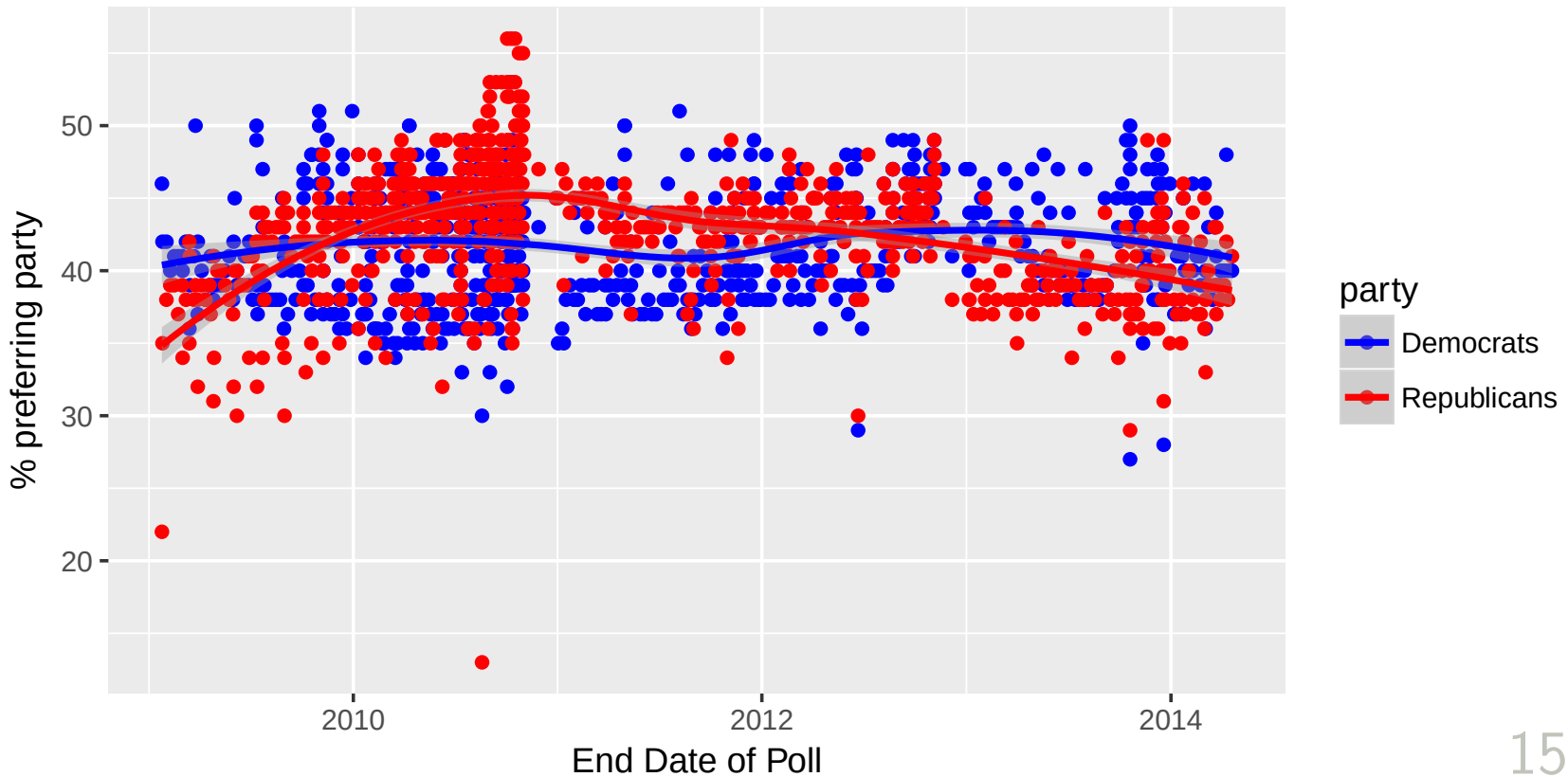
# Reshaping the Generic Ballot Data

```
genericBallot_long <- genericBallot %>%
    gather(key = "party", value = "share", Democrats, Republicans) %>%
    select(startdate, enddate, party, share)
genericBallot_long %>% mosaic::sample(size = 8)
```

```
# A tibble: 8 x 5
   startdate    enddate       party share orig.id
      <date>     <date>       <chr> <int>   <chr>
1 2011-08-17 2011-08-21 Republicans    37    1103
2 2011-06-18 2011-06-19   Democrats    44     327
3 2009-07-10 2009-07-12 Republicans    42    1546
4 2009-09-14 2009-09-20   Democrats    38     732
5 2013-03-25 2013-03-31 Republicans    37     912
6 2011-10-01 2011-10-02   Democrats    44     300
7 2010-05-18 2010-05-18 Republicans    46    1381
8 2013-12-16 2013-12-19 Republicans    31     831
```

# Easier to Plot

```
genericBallot_long %>%
    ggplot(aes(x = enddate, y = share, color = party)) +
    geom_point() +
    geom_smooth() +
    scale_color_manual(values = c("Democrats" = "blue", "Republicans" = "red")) +
    ylab("% preferring party") + xlab("End Date of Poll")
```



# Outline

# The Opposite Problem

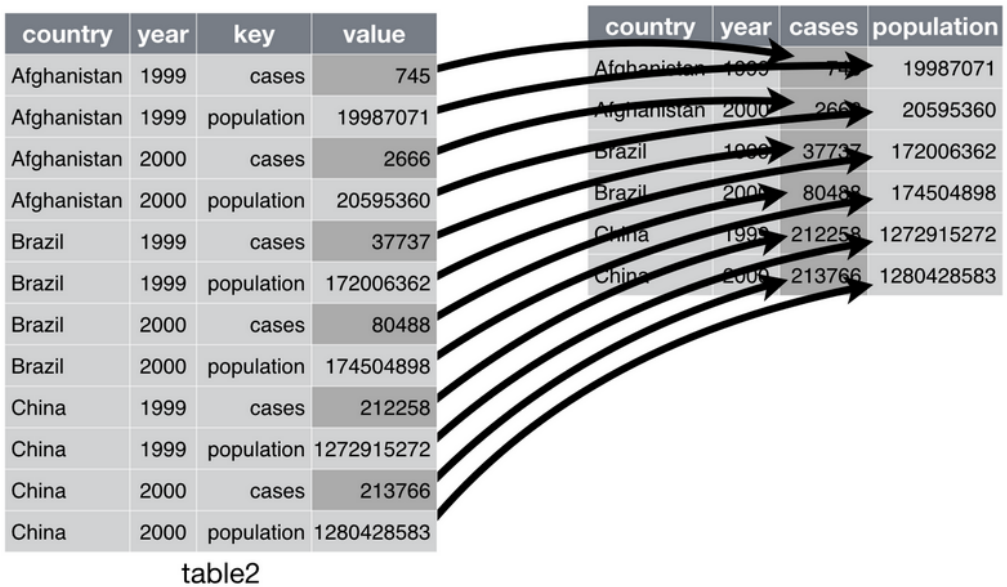
```
head(table2)
```

```
## # A tibble: 6 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
```

- What is a case here?
- What does count represent?
- Not tidy: doesn't map cases to properties

# spread(): Convert narrow to wide

```
table2 %>%
    spread(
        key = key,    # the key column tells us which variable
                      # 'value' is about
        value = value # contains the values that need to be split
                      # into two variables
        )
```



| country | year | key | value |
|---|---|---|---|
| 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 |

table2

| 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 |