# Data Reshaping

Data Wrangling in R

## Reshaping Data

In this module, we will show you how to:

1. Reshape data from wide (fat) to long (tall)
2. Reshape data from long (tall) to wide (fat)
3. Merge Data/Joins
4. Perform operations by a grouping variable

## What is wide/long data?

Data is stored *differently* in the tibble.

Wide: has many columns

```
# A tibble: 1 x 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>   <chr>          <chr>         <chr>
1 Alabama 37.2%          36.0%         32.4%
```

Long: column names become data

```
# A tibble: 3 x 3
  State   name            value
  <chr>   <chr>           <chr>
1 Alabama June_vacc_rate  37.2%
2 Alabama May_vacc_rate   36.0%
3 Alabama April_vacc_rate 32.4%
```

## What is wide/long data?

Wide: multiple columns per individual, values spread across multiple columns

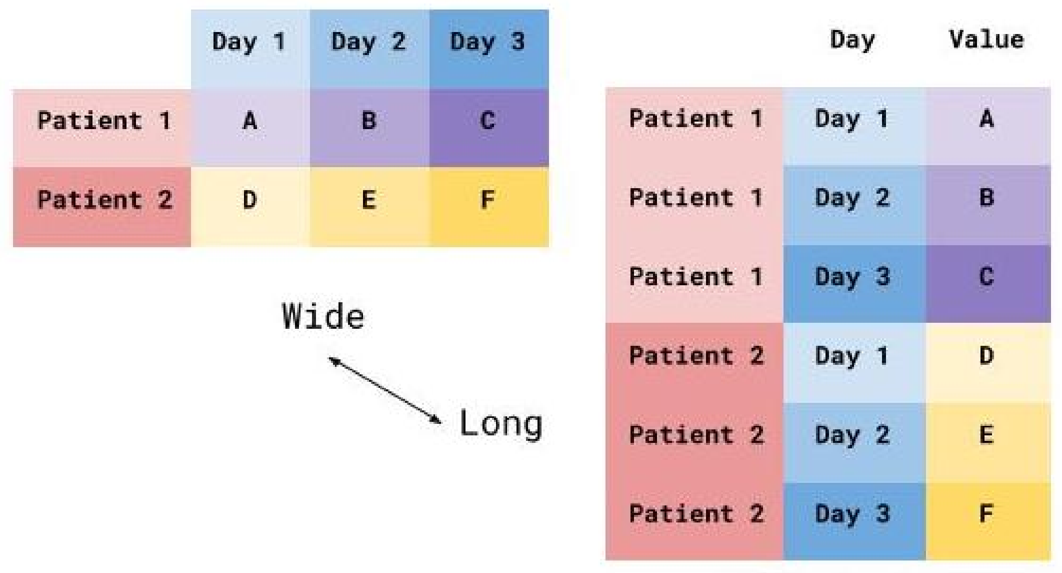
```
# A tibble: 2 x 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>   <chr>          <chr>         <chr>          
1 Alabama 37.2%          36.0%         32.4%          
2 Alaska  47.5%          46.2%         41.7%          
```

Long: multiple rows per observation, a single column contains the values

```
# A tibble: 6 x 3
  State   name            value
  <chr>   <chr>           <chr>
1 Alabama June_vacc_rate  37.2%
2 Alabama May_vacc_rate   36.0%
3 Alabama April_vacc_rate 32.4%
4 Alaska  June_vacc_rate  47.5%
5 Alaska  May_vacc_rate   46.2%
6 Alaska  April_vacc_rate 41.7%
```

## What is wide/long data?

Data is wide or long **with respect** to certain variables.

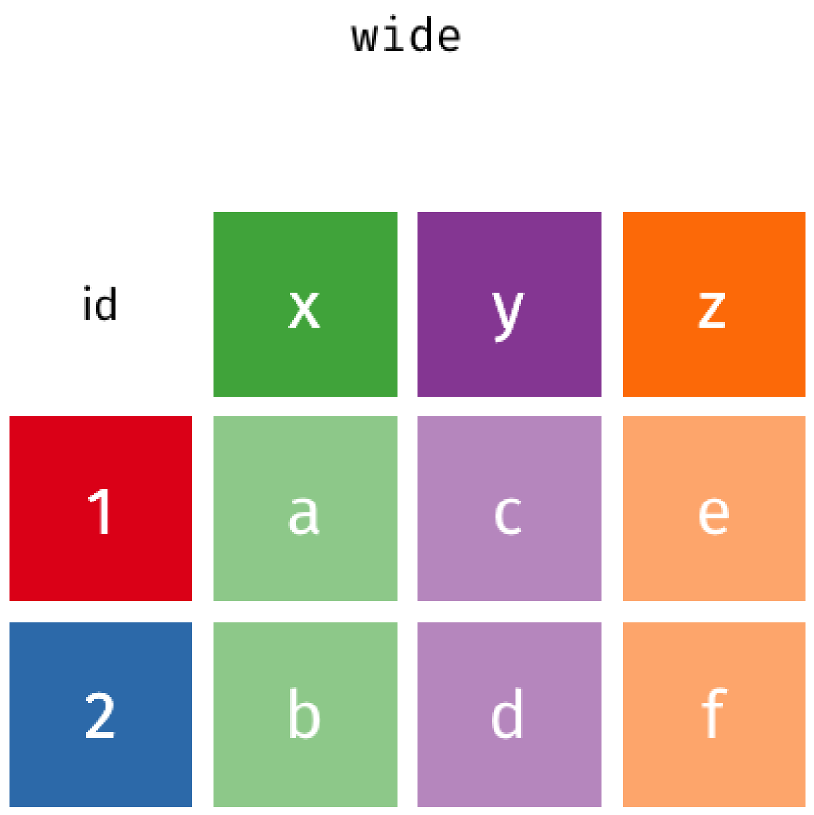


| | Day 1 | Day 2 | Day 3 |
|---|---|---|---|
| Patient 1 | A | B | C |
| Patient 2 | D | E | F |

| | Day | Value |
|---|---|---|
| Patient 1 | Day 1 | A |
| Patient 1 | Day 2 | B |
| Patient 1 | Day 3 | C |
| Patient 2 | Day 1 | D |
| Patient 2 | Day 2 | E |
| Patient 2 | Day 3 | F |

CC-BY jhudatascience.org

## What is wide/long data?


wide

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

https://github.com/gadenbuie/tidyexplain/blob/master/images/tidyr-spread-gather.gif

## Why do we need to switch between wide/long data?

Wide: **Easier for humans to read**

```
# A tibble: 2 x 4
  State   June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>   <chr>          <chr>         <chr>          
1 Alabama 37.2%          36.0%         32.4%          
2 Alaska  47.5%          46.2%         41.7%          
```

Long: **Easier for R to make plots & do analysis**

```
# A tibble: 6 x 3
  State   name            value
  <chr>   <chr>           <chr>
1 Alabama June_vacc_rate  37.2%
2 Alabama May_vacc_rate   36.0%
3 Alabama April_vacc_rate 32.4%
4 Alaska  June_vacc_rate  47.5%
5 Alaska  May_vacc_rate   46.2%
6 Alaska  April_vacc_rate 41.7%
```

## Data used: Charm City Circulator

http://jhudatascience.org/intro_to_r/data/Charm_City_Circulator_Ridership.csv

```
circ = read_csv(
  paste0("http://jhudatascience.org/intro_to_r/",
         "data/Charm_City_Circulator_Ridership.csv"))
head(circ, 5)
```

```
# A tibble: 5 x 15
  day     date    orangeBoardings orangeAlightings orangeAverage purpleBoardir
  <chr>   <chr>             <dbl>            <dbl>         <dbl>            <db
1 Monday  01/11/…             877             1027          952 
2 Tuesday 01/12/…             777              815          796 
3 Wednes… 01/13/…            1203             1220         1212.
4 Thursd… 01/14/…            1194             1233         1214.
5 Friday  01/15/…            1645             1643         1644 
# … with 9 more variables: purpleAlightings <dbl>, purpleAverage <dbl>,
#   greenBoardings <dbl>, greenAlightings <dbl>, greenAverage <dbl>,
#   bannerBoardings <dbl>, bannerAlightings <dbl>, bannerAverage <dbl>,
#   daily <dbl>
```

## tidyr package

`tidyr` allows you to “tidy” your data. We will be talking about:

- `pivot_longer` - make multiple columns into variables, (wide to long)
- `pivot_wider` - make a variable into multiple columns, (long to wide)
- `separate` - string into multiple columns
- `unite` - multiple columns into one string

The `reshape` command exists. It is a **confusing** function. Don’t use it.

## Reshaping data from wide (fat) to long (tall): tidyr

`tidyr::pivot_longer` - puts column data into rows.

- First describe which columns we want to "pivot_longer"
- `names_to =` gives a new name to the pivoted columns
- `values_to =` gives a new name to the values that used to be in those columns

```
long = circ %>%
  pivot_longer(starts_with(c("orange","purple","green","banner")),
               names_to = "var", values_to = "number")
long
```

```
# A tibble: 13,752 x 5
   day    date       daily var              number
   <chr>  <chr>      <dbl> <chr>             <dbl>
 1 Monday 01/11/2010   952 orangeBoardings     877
 2 Monday 01/11/2010   952 orangeAlightings   1027
 3 Monday 01/11/2010   952 orangeAverage       952
 4 Monday 01/11/2010   952 purpleBoardings      NA
 5 Monday 01/11/2010   952 purpleAlightings     NA
 6 Monday 01/11/2010   952 purpleAverage        NA
 7 Monday 01/11/2010   952 greenBoardings       NA
 8 Monday 01/11/2010   952 greenAlightings      NA
 9 Monday 01/11/2010   952 greenAverage         NA
```

## Reshaping data from wide (fat) to long (tall): tidyr

We have many columns here, so we could instead use the `!` to say which columns we *don't* want to pivot.

```
long = circ %>% pivot_longer(!c(day, date, daily),
                    names_to = "var", values_to = "number")
long
```

```
# A tibble: 13,752 x 5
   day    date       daily var               number
   <chr>  <chr>      <dbl> <chr>              <dbl>
 1 Monday 01/11/2010   952 orangeBoardings      877
 2 Monday 01/11/2010   952 orangeAlightings    1027
 3 Monday 01/11/2010   952 orangeAverage        952
 4 Monday 01/11/2010   952 purpleBoardings       NA
 5 Monday 01/11/2010   952 purpleAlightings      NA
 6 Monday 01/11/2010   952 purpleAverage         NA
 7 Monday 01/11/2010   952 greenBoardings        NA
 8 Monday 01/11/2010   952 greenAlightings       NA
 9 Monday 01/11/2010   952 greenAverage          NA
10 Monday 01/11/2010   952 bannerBoardings       NA
# … with 13,742 more rows
```

## Reshaping data from wide (fat) to long (tall): tidyr

```
long %>% count(var)
```

```
# A tibble: 12 x 2
   var                   n
   <chr>             <int>
 1 bannerAlightings   1146
 2 bannerAverage      1146
 3 bannerBoardings    1146
 4 greenAlightings    1146
 5 greenAverage       1146
 6 greenBoardings     1146
 7 orangeAlightings   1146
 8 orangeAverage      1146
 9 orangeBoardings    1146
10 purpleAlightings   1146
11 purpleAverage      1146
12 purpleBoardings    1146
```

## Making a separator

We will use `str_replace` from the `stringr` package to put _ in the names

```
long = long %>% mutate(
  var = str_replace(var, "Board", "_Board"),
  var = str_replace(var, "Alight", "_Alight"),
  var = str_replace(var, "Average", "_Average")
)
long
```

```
# A tibble: 13,752 x 5
   day    date       daily var                  number
   <chr>  <chr>      <dbl> <chr>                 <dbl>
 1 Monday 01/11/2010   952 orange_Boardings        877
 2 Monday 01/11/2010   952 orange_Alightings      1027
 3 Monday 01/11/2010   952 orange_Average          952
 4 Monday 01/11/2010   952 purple_Boardings         NA
 5 Monday 01/11/2010   952 purple_Alightings        NA
 6 Monday 01/11/2010   952 purple_Average           NA
 7 Monday 01/11/2010   952 green_Boardings          NA
 8 Monday 01/11/2010   952 green_Alightings         NA
 9 Monday 01/11/2010   952 green_Average            NA
10 Monday 01/11/2010   952 banner_Boardings         NA
# … with 13,742 more rows
```

## Reshaping data from wide (fat) to long (tall): tidyr

Now each `var` is Boardings, Averages, or Alightings. We use “`into =`” to name the new columns and “`sep =`” to show where the separation should happen.

```
long =
  long %>%
  separate(var, into = c("line", "type"), sep = "_")
long
```

```
# A tibble: 13,752 x 6
   day    date       daily line   type       number
   <chr>  <chr>      <dbl> <chr>  <chr>       <dbl>
 1 Monday 01/11/2010   952 orange Boardings     877
 2 Monday 01/11/2010   952 orange Alightings   1027
 3 Monday 01/11/2010   952 orange Average       952
 4 Monday 01/11/2010   952 purple Boardings      NA
 5 Monday 01/11/2010   952 purple Alightings     NA
 6 Monday 01/11/2010   952 purple Average        NA
 7 Monday 01/11/2010   952 green  Boardings      NA
 8 Monday 01/11/2010   952 green  Alightings     NA
 9 Monday 01/11/2010   952 green  Average        NA
10 Monday 01/11/2010   952 banner Boardings      NA
# … with 13,742 more rows
```

## Re-uniting all the lines

If we had the opposite problem, we could use the `unite` function:

```
reunited = long %>%
  unite(var, line, type, sep = "_")
reunited
```

```
# A tibble: 13,752 x 5
   day    date       daily var                number
   <chr>  <chr>      <dbl> <chr>               <dbl>
 1 Monday 01/11/2010   952 orange_Boardings      877
 2 Monday 01/11/2010   952 orange_Alightings    1027
 3 Monday 01/11/2010   952 orange_Average        952
 4 Monday 01/11/2010   952 purple_Boardings       NA
 5 Monday 01/11/2010   952 purple_Alightings      NA
 6 Monday 01/11/2010   952 purple_Average         NA
 7 Monday 01/11/2010   952 green_Boardings        NA
 8 Monday 01/11/2010   952 green_Alightings       NA
 9 Monday 01/11/2010   952 green_Average          NA
10 Monday 01/11/2010   952 banner_Boardings       NA
# … with 13,742 more rows
```

## Reshaping data from long (tall) to wide (fat): tidyr

In `tidyr`, the `pivot_wider` function spreads rows into columns. Now we have a long data set, but we want to separate the Average, Alightings and Boardings into different columns:

```
wide = long %>% pivot_wider(names_from = "type",
                            values_from = "number")
wide
```

```
# A tibble: 4,584 x 7
   day       date       daily line   Boardings Alightings Average
   <chr>     <chr>      <dbl> <chr>      <dbl>      <dbl>   <dbl>
 1 Monday    01/11/2010  952  orange       877       1027    952 
 2 Monday    01/11/2010  952  purple        NA         NA     NA 
 3 Monday    01/11/2010  952  green         NA         NA     NA 
 4 Monday    01/11/2010  952  banner        NA         NA     NA 
 5 Tuesday   01/12/2010  796  orange       777        815    796 
 6 Tuesday   01/12/2010  796  purple        NA         NA     NA 
 7 Tuesday   01/12/2010  796  green         NA         NA     NA 
 8 Tuesday   01/12/2010  796  banner        NA         NA     NA 
 9 Wednesday 01/13/2010 1212. orange      1203       1220   1212.
10 Wednesday 01/13/2010 1212. purple        NA         NA     NA 
# … with 4,574 more rows
```

## `pivot_longer`

```
sandwich %>%
  pivot_longer()
```