

Data Reshaping

Data Wrangling in R

Reshaping: wide vs. long data

<https://github.com/gadenbuie/tidyexplain/blob/main/images/tidyr-pivoting.gif>

wide

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

What is wide/long data?

Data is stored *differently* in the tibble.

Wide: has many columns

```
# A tibble: 1 × 4
  State      June_vacc_rate May_vacc_rate April_vacc_rate
<chr>      <dbl>          <dbl>          <dbl>
1 Alabama    0.516           0.514           0.511
```

Long: column names become data

```
# A tibble: 3 × 3
  State      name          value
<chr>      <chr>          <dbl>
1 Alabama June_vacc_rate 0.516
2 Alabama May_vacc_rate  0.514
3 Alabama April_vacc_rate 0.511
```

What is wide/long data?

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

```
# A tibble: 2 × 4
  State      June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>      <dbl>         <dbl>         <dbl>
1 Alabama    0.516         0.514         0.511
2 Alaska     0.627         0.626         0.623
```

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

```
# A tibble: 6 × 3
  State      name          value
  <chr>      <chr>         <dbl>
1 Alabama June_vacc_rate 0.516
2 Alabama May_vacc_rate  0.514
3 Alabama April_vacc_rate 0.511
4 Alaska  June_vacc_rate 0.627
5 Alaska  May_vacc_rate  0.626
6 Alaska  April_vacc_rate 0.623
```

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

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

Wide: Easier for humans to read

```
# A tibble: 2 × 4
  State      June_vacc_rate May_vacc_rate April_vacc_rate
  <chr>      <dbl>          <dbl>          <dbl>
1 Alabama    0.516           0.514           0.511
2 Alaska     0.627           0.626           0.623
```

Long: Easier for R to make plots & do analysis

```
# A tibble: 6 × 3
  State      name          value
  <chr>      <chr>          <dbl>
1 Alabama June_vacc_rate 0.516
2 Alabama May_vacc_rate  0.514
3 Alabama April_vacc_rate 0.511
4 Alaska  June_vacc_rate 0.627
5 Alaska  May_vacc_rate  0.626
6 Alaska  April_vacc_rate 0.623
```

Pivoting using `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

The `reshape` command exists. Its arguments are considered more confusing, so we don't recommend it.

You might see old functions `gather` and `spread` when googling. These are older iterations of `pivot_longer` and `pivot_wider`, respectively.

`pivot_longer...`

Reshaping data from wide to long

`pivot_longer()` - puts column data into rows (`tidyr` package)

- First describe which columns we want to “pivot_longer”

```
{long_data} <- {wide_data} %>% pivot_longer(cols = {columns to pivot})
```

Reshaping data from wide to long

```
wide_data
```

```
# A tibble: 1 × 3
  June_vacc_rate May_vacc_rate April_vacc_rate
  <dbl>         <dbl>         <dbl>
1      0.516      0.514      0.511
```

```
long_data <- wide_data %>% pivot_longer(cols = everything())
long_data
```

```
# A tibble: 3 × 2
  name          value
  <chr>         <dbl>
1 June_vacc_rate 0.516
2 May_vacc_rate  0.514
3 April_vacc_rate 0.511
```

Reshaping data from wide to long

`pivot_longer()` - puts column data into rows (`tidyr` package)

- 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_data} <- {wide_data} %>% pivot_longer(cols = {columns to pivot},  
                                           names_to = {New column name: contains old column names},  
                                           values_to = {New column name: contains cell values})
```

Reshaping data from wide to long

```
wide_data
```

```
# A tibble: 1 × 3
  June_vacc_rate May_vacc_rate April_vacc_rate
      <dbl>         <dbl>         <dbl>
1     0.516         0.514         0.511
```

```
long_data <- wide_data %>% pivot_longer(cols = everything(),
                                         names_to = "Month",
                                         values_to = "Rate")
```

```
long_data
```

```
# A tibble: 3 × 2
  Month          Rate
  <chr>         <dbl>
1 June_vacc_rate 0.516
2 May_vacc_rate  0.514
3 April_vacc_rate 0.511
```

Newly created column names are enclosed in quotation marks.

Data used: Charm City Circulator

https://sisbid.github.io/Data-Wrangling/data/Charm_City_Circulator_Ridership.csv

```
circ <-  
  read_csv("https://sisbid.github.io/Data-Wrangling/data/Charm_City_Circulator_Ridership.csv")  
head(circ, 5)
```

```
# A tibble: 5 × 15  
  day      date orangeBoardings orangeAlightings orangeAverage purpleBoardings purpleAlightings purpleAverage  
  <chr>   <chr>           <dbl>             <dbl>             <dbl>           <dbl>             <dbl>           <dbl>  
1 Monday  01/1...           877               1027               952             NA                NA                NA  
2 Tuesday 01/1...           777               815                796             NA                NA                NA  
3 Wednesd.. 01/1...           1203              1220              1212.           NA                NA                NA  
4 Thursday 01/1...           1194              1233              1214.           NA                NA                NA  
5 Friday   01/1...           1645              1643              1644            NA                NA                NA  
# i 7 more variables: greenBoardings <dbl>, greenAlightings <dbl>, greenAverage <dbl>, bannerBoardings <dbl>,  
# bannerAlightings <dbl>, bannerAverage <dbl>, daily <dbl>
```

Reshaping data from wide to long

```
long <- circ %>%  
  pivot_longer(starts_with(c("orange", "purple", "green", "banner")))  
long
```

```
# A tibble: 13,752 × 5  
  day    date      daily name      value  
  <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  
# i 13,742 more rows
```

Reshaping data from wide to long

There are many ways to select the columns we want. Use `?tidyr_tidy_select` to look at more column selection options.

```
long <- circ %>%  
  pivot_longer( !c(day, date, daily))  
long
```

```
# A tibble: 13,752 × 5  
  day      date      daily name      value  
  <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  
# i 13,742 more rows
```

Cleaning up long data

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

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

```
# A tibble: 13,752 × 5  
  day    date      daily name          value  
  <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  
# i 13,742 more rows
```


Cleaning up long data

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(name, into = c("line", "type"), sep = "_")
long
```

```
# A tibble: 13,752 × 6
  day   date       daily line   type      value
<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
# i 13,742 more rows
```

`pivot_wider...`

Reshaping data from long to wide

`pivot_wider()` - spreads row data into columns (tidyr package)

- `names_from` = the old column whose contents will be spread into multiple new column names.
- `values_from` = the old column whose contents will fill in the values of those new columns.

```
{wide_data} <- {long_data} %>%  
  pivot_wider(names_from = {Old column name: contains new column names},  
             values_from = {Old column name: contains new cell values})
```

Reshaping data from long to wide

```
long_data
```

```
# A tibble: 3 × 2
  Month      Rate
  <chr>      <dbl>
1 June_vacc_rate 0.516
2 May_vacc_rate  0.514
3 April_vacc_rate 0.511
```

```
wide_data <- long_data %>% pivot_wider(names_from = "Month",
                                       values_from = "Rate")
```

```
wide_data
```

```
# A tibble: 1 × 3
  June_vacc_rate May_vacc_rate April_vacc_rate
  <dbl>          <dbl>          <dbl>
1      0.516      0.514      0.511
```

Reshaping Charm City Circulator

long

```
# A tibble: 13,752 × 6
  day      date      daily line  type      value
  <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
# i 13,742 more rows
```

Reshaping Charm City Circulator

```
wide <- long %>% pivot_wider(names_from = "type",  
                             values_from = "value")
```

```
wide
```

```
# A tibble: 4,584 × 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

```
# i 4,574 more rows
```

Adding prefixes

Prefixes when pivoting

the `datasets::airquality` data shows various air quality metrics measured in New York in 1973.

```
air <- datasets::airquality %>% select(Temp, Month, Day)
air
```

	Temp	Month	Day
1	67	5	1
2	72	5	2
3	74	5	3
4	62	5	4
5	56	5	5
6	66	5	6
7	65	5	7
8	59	5	8
9	61	5	9
10	69	5	10
11	74	5	11
12	69	5	12
13	66	5	13
14	68	5	14
15	58	5	15
16	64	5	16
17	66	5	17
18	57	5	18
19	68	5	19
20	62	5	20
21	59	5	21

Prefixes when pivoting

Let's pivot `Month` wider: but it might be helpful to add "Month" to the new column name so it isn't just numbers.

```
air %>% pivot_wider(names_from = "Month",  
                    values_from = "Temp")
```

```
# A tibble: 31 × 6  
  Day `5` `6` `7` `8` `9`  
  <int> <int> <int> <int> <int> <int>  
1     1     67     78     84     81     91  
2     2     72     74     85     81     92  
3     3     74     67     81     82     93  
4     4     62     84     84     86     93  
5     5     56     85     83     85     87  
6     6     66     79     83     87     84  
7     7     65     82     88     89     80  
8     8     59     87     92     90     78  
9     9     61     90     92     90     75  
10    10     69     87     89     92     73  
# i 21 more rows
```

Prefixes when pivoting

Much better!

```
air %>% pivot_wider(names_from = "Month",  
                    values_from = "Temp",  
                    names_prefix = "Month_")
```

```
# A tibble: 31 × 6  
  Day Month_5 Month_6 Month_7 Month_8 Month_9  
  <int> <int> <int> <int> <int> <int>  
1     1     67     78     84     81     91  
2     2     72     74     85     81     92  
3     3     74     67     81     82     93  
4     4     62     84     84     86     93  
5     5     56     85     83     85     87  
6     6     66     79     83     87     84  
7     7     65     82     88     89     80  
8     8     59     87     92     90     78  
9     9     61     90     92     90     75  
10    10     69     87     89     92     73  
# i 21 more rows
```

Summary

- `tidyr` package helps us convert between wide and long data
- `pivot_longer()` goes from wide -> long
 - Specify columns you want to pivot
 - Specify `names_to =` and `values_to =` for custom naming
- `pivot_wider()` goes from long -> wide
 - Specify `names_from =` and `values_from =`