

Intro to R

Manipulating Data in R

Recap of Subsetting

- `pull()` to get values out of a data frame
- `select()` creates a smaller data frame with only certain columns
- you can `select()` based on patterns in the column names using tidyselect functions
- you can combine multiple tidyselect functions together like `select(starts_with("C"), ends_with("state"))`
- `filter()` can be used to filter out rows based on logical conditions
- avoid using quotes when referring to column names with `filter()`

Recap Subsetting Continued

- `==` is the same as equivalent to
- `&` means both conditions must be met to remain after `filter()`
- `|` means either conditions needs to be met to remain after `filter()`

Recap of Data Cleaning

- `count()` can help determine if we have NA values
- `filter()` automatically removes NA values - can't confirm or deny if condition is met (need `| is.na()` to keep them)
- `drop_na()` can help you remove NA values from a variable or an entire data frame
- NA values can change your calculation results
- think about what NA values represent

Recap of Data Cleaning

- `recode()` can help with simple recoding values of a variable (needs to be inside the `mutate` function)

[Cheatsheet](#)

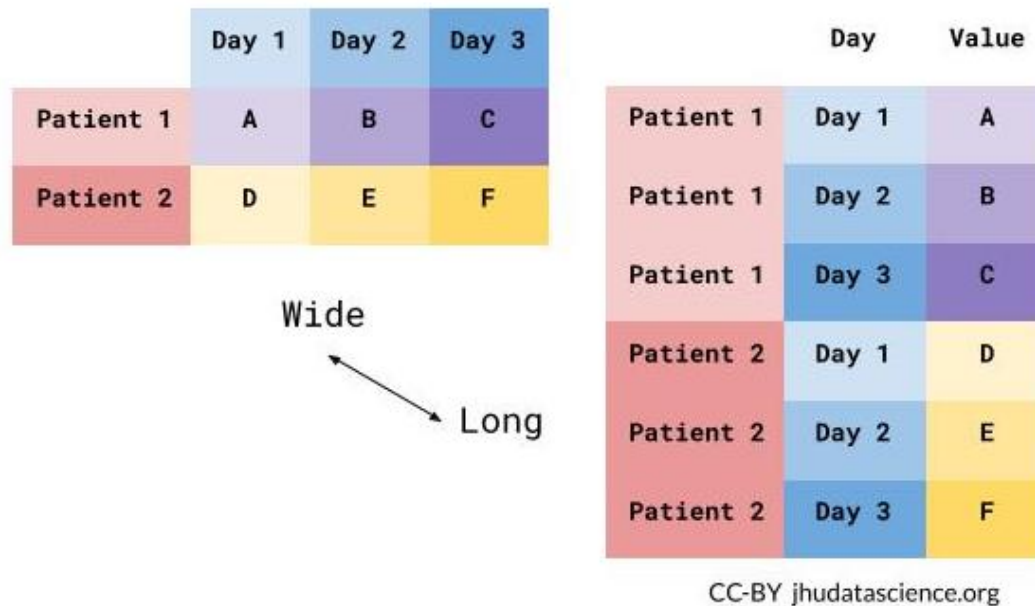
Manipulating Data

In this module, we will show you how to:

1. The two major forms of data (long and wide)
2. Reshape data from wide to long for data analysis and visualization

What is wide/long data?

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



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?

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

wide

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

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_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

```
# 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://hutchdatascience.org/SeattleStatSummer_R/data/Charm_City_Circulator_Rider

```
circ <- read_csv("https://hutchdatascience.org/SeattleStatSummer_R/data/Charm_City_Circulator_Rider.csv")
head(circ, 5)
```

```
# A tibble: 5 × 15
  day      date orangeBoardings orangeAlightings orangeAverage purpleBoardings
<chr>    <chr>           <dbl>             <dbl>             <dbl>           <dbl>
1 Monday  01/1...           877               1027               952             NA
2 Tuesday 01/1...           777               815                796             NA
3 Wednesday 01/1...        1203              1220              1212.           NA
4 Thursday 01/1...        1194              1233              1214.           NA
5 Friday   01/1...        1645              1643              1644             NA
# ... with 9 more variables: purpleAlightings <dbl>, purpleAverage <dbl>,
#   greenBoardings <dbl>, greenAlightings <dbl>, greenAverage <dbl>,
#   bannerBoardings <dbl>, bannerAlightings <dbl>, bannerAverage <dbl>,
#   daily <dbl>
```

Mission: Taking the averages by line

Let's imagine we want to create a table of average ridership by route/line. Results should look something like:

```
example <- tibble(line = c("orange", "purple", "green", "banner"),  
                  avg = c("600(?)", "700(?)", "500(?)", "400(?)")  
                )  
example
```

```
# A tibble: 4 × 2  
  line    avg  
  <chr> <chr>  
1 orange 600(?)  
2 purple 700(?)  
3 green  500(?)  
4 banner 400(?)
```

Reshaping data from wide to long

```
long <- circ %>%  
  pivot_longer(  
    cols = starts_with(c("orange", "purple", "green", "banner")),  
    names_to = "bus_type",  
    values_to = "number_of_individuals")
```

long

```
# A tibble: 13,752 × 5  
  day   date       daily bus_type          number_of_individuals  
  <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
```

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 × 5
  day   date       daily bus_type      number_of_individuals
  <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 Charm City Circulator

```
wide <- long %>% pivot_wider(names_from = "bus_type",
                             values_from = "number_of_individuals")
wide
```

```
# A tibble: 1,146 × 15
  day      date      daily orangeBoardings orangeAlightings orangeAverage
  <chr>    <chr>    <dbl>      <dbl>          <dbl>          <dbl>
1 Monday  01/11/2010  952         877            1027            952
2 Tuesday 01/12/2010  796         777            815             796
3 Wednesday 01/13/2010 1212.        1203           1220            1212.
4 Thursday 01/14/2010 1214.        1194           1233            1214.
5 Friday   01/15/2010 1644         1645           1643            1644
6 Saturday 01/16/2010 1490.        1457           1524            1490.
7 Sunday   01/17/2010  888.        839            938             888.
8 Monday   01/18/2010 1000.        999            1000            1000.
9 Tuesday  01/19/2010 1035         1023           1047            1035
10 Wednesday 01/20/2010 1396.        1375           1416            1396.
# ... with 1,136 more rows, and 9 more variables: purpleBoardings <dbl>,
#   purpleAlightings <dbl>, purpleAverage <dbl>, greenBoardings <dbl>,
#   greenAlightings <dbl>, greenAverage <dbl>, bannerBoardings <dbl>,
#   bannerAlightings <dbl>, bannerAverage <dbl>
```

Summary

- `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 =`

[Workshop Website](#)