

# Reading in Data

## *Data Wrangling, Session 6*

Kieran Healy

Code Horizons

July 28, 2024

**Reading in data with readr  
and haven**

# Load the packages, as always

```
library(here)      # manage file paths  
library(socviz)    # data and some useful functions  
library(tidyverse) # your friend and mine  
library(haven)     # for Stata, SAS, and SPSS files
```

We've put a lot of  
pieces in place at  
this point

Including several things we haven't fully exploited yet

# Data we want to get into R

Nice, clean CSV files.

More troublesome CSVs.

Other plain-text formats.

Foreign formats, like Stata.

Quite messy things like tables on web pages.

... and more besides.

# Reading in CSV files

CSV is not really a proper format at all!

Base R has `read.csv()`

Corresponding tidyverse “underscored” version: `read_csv()`.

It is pickier and more talkative than the Base R version.

# Where's the data? Using **here()**

If we're loading a file, it's coming from *somewhere*.

If it's on our local disk somewhere, we will need to interact with the file system. We should try to do this in a way that avoids *absolute* file paths.

```
# This is not portable  
df ← read_csv("/Users/kjhealy/Documents/data/misc/project/data/mydata.csv")
```

# Where's the data? Using **here()**

If we're loading a file, it's coming from *somewhere*.

If it's on our local disk somewhere, we will need to interact with the file system. We should try to do this in a way that avoids *absolute* file paths.

```
# This is not portable  
df ← read_csv("/Users/kjhealy/Documents/data/misc/project/data/mydata.csv")
```

We should also do it in a way that is *platform independent*.

This makes it easier to share your work, move it around, etc. Projects should be self-contained.



# Where's the data? Using `here()`

The `here` package, and `here()` function builds paths relative to the top level of your R project.

```
here() # this path will be different for you
```

```
[1] "/Users/kjhealy/Documents/courses/data_wrangling"
```

# Where's the data? Using `here()`

This seminar's files all live in an RStudio project. It looks like this:

```
/Users/kjhealy/Documents/courses/data_wrangling
├── 00_dummy_files
├── LICENSE
├── Makefile
├── README.md
├── README.qmd
├── _extensions
├── _freeze
├── _quarto.yml
├── _site
├── _targets
├── _targets.R
├── _variables.yml
├── build
├── code
├── course_notes.qmd
├── data
├── data_wrangling.Rproj
└── deploy.sh
```

I want to load files from the `data` folder, but I also want *you* to be able to load them. I'm writing this from somewhere deep in the `slides`

# Where's the data? Using `here()`

So:

```
## Load the file relative to the path from the top of the project, without separators, etc
organs ← read_csv(file = here("data", "organdonation.csv"))
```

organs

```
# A tibble: 238 × 21
  country year donors pop pop.dens gdp gdp.lag health health.lag pubhealth
  <chr>   <dbl> <dbl> <dbl>    <dbl> <dbl>   <dbl>   <dbl>    <dbl>    <dbl>
1 Austra... NA    NA    17065  0.220 16774  16591  1300    1224      4.8
2 Austra... 1991  12.1  17284  0.223 17171  16774  1379    1300      5.4
3 Austra... 1992  12.4  17495  0.226 17914  17171  1455    1379      5.4
4 Austra... 1993  12.5  17667  0.228 18883  17914  1540    1455      5.4
5 Austra... 1994  10.2  17855  0.231 19849  18883  1626    1540      5.4
6 Austra... 1995  10.2  18072  0.233 21079  19849  1737    1626      5.5
7 Austra... 1996  10.6  18311  0.237 21923  21079  1846    1737      5.6
8 Austra... 1997  10.3  18518  0.239 22961  21923  1948    1846      5.7
9 Austra... 1998  10.5  18711  0.242 24148  22961  2077    1948      5.9
10 Austra... 1999   8.67 18926  0.244 25445  24148  2231    2077      6.1
# i 228 more rows
# i 11 more variables: roads <dbl>, cerebvas <dbl>, assault <dbl>,
#   external <dbl>, txp.pop <dbl>, world <chr>, opt <chr>, consent.law <chr>,
#   consent.practice <chr>, consistent <chr>, ccode <chr>
```

# `read_csv()` comes in different varieties

`read_csv()` Field separator is a comma: ,

```
organs ← read_csv(file = here("data", "organdonation.csv"))
```

`read_csv2()` Field separator is a semicolon: ;

```
# Example only  
my_data ← read_csv2(file = here("data", "my_euro_file.csv"))
```

Both are special cases of `read_delim()`

# Other species are also catered to

`read_tsv()` Tab separated.

`read_fwf()` Fixed-width files.

`read_log()` Log files (i.e. computer log files).

`read_lines()` Just read in lines, without trying to parse them.

`read_table()` Data that's separated by one (or more) columns of space.

# You can read files remotely, too

You can give all of these functions local files, or they can point to URLs.

Compressed files will be automatically uncompressed.

(Be careful what you download from remote locations!)

```
organ_remote ← read_csv("http://kjhealy.co/organdonation.csv")
```

```
organ_remote
```

```
# A tibble: 238 × 21
```

	country	year	donors	pop	pop.dens	gdp	gdp.lag	health	health.lag	pubhealth
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	Austra...	NA	NA	17065	0.220	16774	16591	1300	1224	4.8
2	Austra...	1991	12.1	17284	0.223	17171	16774	1379	1300	5.4
3	Austra...	1992	12.4	17495	0.226	17914	17171	1455	1379	5.4
4	Austra...	1993	12.5	17667	0.228	18883	17914	1540	1455	5.4
5	Austra...	1994	10.2	17855	0.231	19849	18883	1626	1540	5.4
6	Austra...	1995	10.2	18072	0.233	21079	19849	1737	1626	5.5
7	Austra...	1996	10.6	18311	0.237	21923	21079	1846	1737	5.6
8	Austra...	1997	10.3	18518	0.239	22961	21923	1948	1846	5.7
9	Austra...	1998	10.5	18711	0.242	24148	22961	2077	1948	5.9
10	Austra...	1999	8.67	18926	0.244	25445	24148	2231	2077	6.1

```
# i 228 more rows
```

```
# i 11 more variables: roads <dbl>, cerebvas <dbl>, assault <dbl>,
```

```
# external <dbl>, top <dbl>, world <dbl>, cont <dbl>, concept <dbl>, robot
```

# An example: `read_table()`

```
England and Wales, Total Population, Death rates (period 1x1), Last modified: 02 Apr
2018; Methods Protocol: v6 (2017)

Year      Age      Female      Male      Total
1841      0      0.136067    0.169189    0.152777
1841      1      0.059577    0.063208    0.061386
1841      2      0.036406    0.036976    0.036689
1841      3      0.024913    0.026055    0.025480
1841      4      0.018457    0.019089    0.018772
1841      5      0.013967    0.014279    0.014123
1841      6      0.010870    0.011210    0.011040
1841      7      0.008591    0.008985    0.008788
1841      8      0.006860    0.007246    0.007053
1841      9      0.005772    0.006050    0.005911
1841     10      0.005303    0.005382    0.005343
1841     11      0.005114    0.005002    0.005057
1841     12      0.005145    0.004856    0.004999
1841     13      0.005455    0.004955    0.005202
```

```
1841     105      0.576987    1.727848    0.700373
1841     106      0.677711    6.000000    0.795287
1841     107      0.900000    .          0.900000
1841     108      1.388430    .          1.388430
1841     109      .          .          .
1841     110+      .          .          .
1842      0      0.148491    0.184007    0.166481
1842      1      0.063038    0.066596    0.064818
1842      2      0.035203    0.035854    0.035527
```

# An example: `read_table()`

England and Wales, Total Population, Death rates (period 1x1), Last modified: 02 Apr 2018; Methods Protocol: v6 (2017)

Year	Age	Female	Male	Total
1841	0	0.136067	0.169189	0.152777
1841	1	0.059577	0.063208	0.061386
1841	2	0.036406	0.036976	0.036689
1841	3	0.024913	0.026055	0.025480
1841	4	0.018457	0.019089	0.018772
1841	5	0.013967	0.014279	0.014123
1841	6	0.010870	0.011210	0.011040
1841	7	0.008591	0.008985	0.008788
1841	8	0.006860	0.007246	0.007053
1841	9	0.005772	0.006050	0.005911
1841	10	0.005303	0.005382	0.005343
1841	11	0.005114	0.005002	0.005057
1841	12	0.005145	0.004856	0.004999
1841	13	0.005455	0.004955	0.005202

1841	105	0.376987	1.727848	0.708373
1841	106	0.677711	6.000000	0.795287
1841	107	0.900000	.	0.900000
1841	108	1.388430	.	1.388430
1841	109	.	.	.
1841	110+	.	.	.
1842	0	0.148491	0.184007	0.166481
1842	1	0.063038	0.066596	0.064818
1842	2	0.035203	0.035854	0.035527

```
engmort ← read_table(here("data", "mortality.txt"),  
                      skip = 2, na = ".")
```

engmort

# A tibble: 222 × 5

	Year	Age	Female	Male	Total
	<dbl>	<chr>	<dbl>	<dbl>	<dbl>
1	1841	0	0.136	0.169	0.153
2	1841	1	0.0596	0.0632	0.0614
3	1841	2	0.0364	0.0370	0.0367
4	1841	3	0.0249	0.0261	0.0255
5	1841	4	0.0185	0.0191	0.0188
6	1841	5	0.0140	0.0143	0.0141
7	1841	6	0.0109	0.0112	0.0110
8	1841	7	0.00859	0.00898	0.00879
9	1841	8	0.00686	0.00725	0.00705
10	1841	9	0.00577	0.00605	0.00591

# i 212 more rows



# Attend to the **column specification**

```
engmort ← read_table(here("data", "mortality.txt"),  
                      skip = 2, na = ".")
```

— Column specification —

```
cols(  
  Year = col_double(),  
  Age = col_character(),  
  Female = col_double(),  
  Male = col_double(),  
  Total = col_double()  
)
```

The column specification tells you what the read function did. That is, how it interpreted each of the columns. It will also report if things don't go as expected.

Why is **age** imported in **character** format?

# Attend to the **column specification**

Absent you giving them a column specification, the **read\_** functions try to *guess* what the type of each column is. They do this by looking at the first thousand rows of each column. They may guess incorrectly!

# Normalizing names and recoding

```
read_table(here("data", "mortality.txt"),  
           skip = 2, na = ".")
```

```
# A tibble: 222 × 5  
  Year Age   Female   Male  
Total  
  <dbl> <chr>   <dbl>   <dbl>  
<dbl>  
1  1841 0      0.136   0.169   0.153  
2  1841 1      0.0596  0.0632  0.0614  
3  1841 2      0.0364  0.0370  0.0367  
4  1841 3      0.0249  0.0261  0.0255  
5  1841 4      0.0185  0.0191  0.0188  
6  1841 5      0.0140  0.0143  0.0141  
7  1841 6      0.0109  0.0112  0.0110  
8  1841 7      0.00859 0.00898  
0.00879  
9  1841 8      0.00686 0.00725  
0.00705  
10 1841 9      0.00577 0.00605  
0.00591  
# i 212 more rows
```

# Normalizing names and recoding

```
read_table(here("data", "mortality.txt"),  
           skip = 2, na = ".") ►  
janitor::clean_names()
```

```
# A tibble: 222 × 5  
  year age   female   male  
total  
  <dbl> <chr>   <dbl>   <dbl>  
<dbl>  
1  1841 0      0.136   0.169   0.153  
2  1841 1      0.0596  0.0632  0.0614  
3  1841 2      0.0364  0.0370  0.0367  
4  1841 3      0.0249  0.0261  0.0255  
5  1841 4      0.0185  0.0191  0.0188  
6  1841 5      0.0140  0.0143  0.0141  
7  1841 6      0.0109  0.0112  0.0110  
8  1841 7      0.00859 0.00898  
0.00879  
9  1841 8      0.00686 0.00725  
0.00705  
10 1841 9      0.00577 0.00605  
0.00591  
# i 212 more rows
```

# Normalizing names and recoding

```
read_table(here("data", "mortality.txt"),  
           skip = 2, na = ".") ▷  
janitor::clean_names() ▷  
mutate(age = as.integer(recode(age, "110+" = "110")))
```

```
# A tibble: 222 × 5  
  year age female   male  
total  
  <dbl> <int>   <dbl>   <dbl>  
<dbl>  
1  1841     0 0.136   0.169   0.153  
2  1841     1 0.0596  0.0632  0.0614  
3  1841     2 0.0364  0.0370  0.0367  
4  1841     3 0.0249  0.0261  0.0255  
5  1841     4 0.0185  0.0191  0.0188  
6  1841     5 0.0140  0.0143  0.0141  
7  1841     6 0.0109  0.0112  0.0110  
8  1841     7 0.00859 0.00898  
0.00879  
9  1841     8 0.00686 0.00725  
0.00705  
10 1841     9 0.00577 0.00605  
0.00591  
# i 212 more rows
```

# Janitor


The `janitor` package is very handy!

The main cost of normalizing names comes with, e.g., data where there is a codebook you need to consult. But in general it's worth it.

# Example: Colspecs

# More on column specifications

## CDC/NCHS data: Provisional COVID-19 Death Counts by Sex, Age, and State

 Centers for Disease Control and Prevention  
CDC 24/7: Saving Lives. Protecting People.™

Data.CDC.gov

[Home](#) [Data Catalog](#) [Developers](#) [Video Guides](#)

[Facebook](#) [Twitter](#) [YouTube](#) [Instagram](#)

[Sign In](#)

Provisional COVID-19 Death Counts by Sex,  
Age, and State NCHS

[View Data](#) [Visualize](#) [Export](#) [API](#) [...](#)

Deaths involving coronavirus disease 2019 (COVID-19), pneumonia, and influenza reported to NCHS by sex and age group and state.

NOTICE TO USERS: As of September 2, 2020, this data file includes the following age groups  
[More](#)

**Updated**  
April 14, 2021

**Data Provided by**  
National Center for Health Statistics



# More on column specifications

What's in this Dataset?

Rows	Columns
<b>52.3K</b>	<b>16</b>

Columns in this Dataset

Column Name	Description	Type		
<b>Data As Of</b>	Date of analysis	Date & Time	📅	▼
<b>Start Date</b>	First date of data period	Date & Time	📅	▼
<b>End Date</b>	Last date of data period	Date & Time	📅	▼
<b>Group</b>	Indicator of whether data measured by Month, by Year, or ...	Plain Text	T	▼
<b>Year</b>	Year in which death occurred	Number	#	▼
<b>Month</b>	Month in which death occurred	Number	#	▼
<b>State</b>	Jurisdiction of occurrence	Plain Text	T	▼
<b>Sex</b>	Sex	Plain Text	T	▼
<b>Age Group</b>	Age group	Plain Text	T	▼
<b>COVID-19 Deaths</b>	Deaths involving COVID-19 (ICD-code U07.1)	Number	#	▼
<b>Total Deaths</b>	Deaths from all causes of death	Number	#	▼
<b>Pneumonia Deaths</b>	Pneumonia Deaths (ICD-10 codes J12.0-J18.9)	Number	#	▼
<b>Pneumonia and COVID-19 Deaths</b>	Deaths with Pneumonia and COVID-19 (ICD-10 codes J12.0-...	Number	#	▼
<b>Influenza Deaths</b>	Influenza Deaths (ICD-10 codes J09-J11)	Number	#	▼
<b>Pneumonia, Influenza, or COVID-19 Deaths</b>	Deaths with Pneumonia, Influenza, or COVID-19 (ICD-10 co...	Number	#	▼
<b>Footnote</b>	Suppressed counts (1-9)	Plain Text	T	▼
<a href="#">Show Less</a>				

# Let's try to load it

```
nchs ← with_edition(1, read_csv(here("data", "SAS_on_2021-04-13.csv")))
```

— Column specification —

```
cols(  
  `Data As Of` = col_character(),  
  `Start Date` = col_character(),  
  `End Date` = col_character(),  
  Group = col_character(),  
  Year = col_logical(),  
  Month = col_logical(),  
  State = col_character(),  
  Sex = col_character(),  
  `Age Group` = col_character(),  
  `COVID-19 Deaths` = col_double(),  
  `Total Deaths` = col_double(),  
  `Pneumonia Deaths` = col_double(),  
  `Pneumonia and COVID-19 Deaths` = col_double(),  
  `Influenza Deaths` = col_double(),  
  `Pneumonia, Influenza, or COVID-19 Deaths` = col_double(),  
  Footnote = col_character()  
)
```

Warning: 88128 parsing failures.

row	col	expected	actual
-----	-----	----------	--------

2755	Year	1/0/T/F/TRUE/FALSE	2020
------	------	--------------------	------

'/Users/kjhealy/Documents/courses/data_wrangling/data/SAS_on_2021-04-13.csv'			
--	--	--	--

2756	Year	1/0/T/F/TRUE/FALSE	2020
------	------	--------------------	------

'/Users/kihealy/Documents/courses/data_wrangling/data/SAS_on_2021-04-13.csv'			
--	--	--	--

# Let's try to load it

```
problems(nchs)
```

```
# A tibble: 88,128 × 5
  row col expected actual file
  <int> <chr> <chr>      <chr> <chr>
1  2755 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
2  2756 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
3  2757 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
4  2758 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
5  2759 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
6  2760 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
7  2761 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
8  2762 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
9  2763 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
10 2764 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...'
# i 88,118 more rows
```

# Let's try to load it

```
problems(nchs)
```

```
# A tibble: 88,128 × 5
  row col expected actual file
  <int> <chr> <chr>      <chr> <chr>
1  2755 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
2  2756 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
3  2757 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
4  2758 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
5  2759 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
6  2760 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
7  2761 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
8  2762 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
9  2763 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
10 2764 Year 1/0/T/F/TRUE/FALSE 2020 '/Users/kjhealy/Documents/courses/data...
# i 88,118 more rows
```

Problems are stored as an attribute of the **nchs** object, so we can revisit them.

Parsing failures tend to cascade. Our data only has 56k rows but we got 88k failures.

# Take a look with `head()`

```
head(nchs)
```

```
# A tibble: 6 × 16
  `Data As Of` `Start Date` `End Date` Group Year Month State Sex `Age Group`
  <chr>         <chr>         <chr>    <chr> <lg1> <lg1> <chr> <chr> <chr>
1 04/07/2021   01/01/2020   04/03/2021 By T... NA     NA     Unit... All ... All Ages
2 04/07/2021   01/01/2020   04/03/2021 By T... NA     NA     Unit... All ... Under 1 ye...
3 04/07/2021   01/01/2020   04/03/2021 By T... NA     NA     Unit... All ... 0-17 years
4 04/07/2021   01/01/2020   04/03/2021 By T... NA     NA     Unit... All ... 1-4 years
5 04/07/2021   01/01/2020   04/03/2021 By T... NA     NA     Unit... All ... 5-14 years
6 04/07/2021   01/01/2020   04/03/2021 By T... NA     NA     Unit... All ... 15-24 years
# i 7 more variables: `COVID-19 Deaths` <dbl>, `Total Deaths` <dbl>,
#   `Pneumonia Deaths` <dbl>, `Pneumonia and COVID-19 Deaths` <dbl>,
#   `Influenza Deaths` <dbl>, `Pneumonia, Influenza, or COVID-19 Deaths` <dbl>,
#   Footnote <chr>
```

# Take a look with `tail()`

```
tail(nchs)
```

```
# A tibble: 6 × 16
  `Data As Of` `Start Date` `End Date` Group Year Month State Sex `Age Group`
  <chr>        <chr>        <chr>    <chr> <lg1> <lg1> <chr> <chr> <chr>
1 04/07/2021   04/01/2021   04/03/2021 By M... NA    NA    Puer... Fema... 45-54 years
2 04/07/2021   04/01/2021   04/03/2021 By M... NA    NA    Puer... Fema... 50-64 years
3 04/07/2021   04/01/2021   04/03/2021 By M... NA    NA    Puer... Fema... 55-64 years
4 04/07/2021   04/01/2021   04/03/2021 By M... NA    NA    Puer... Fema... 65-74 years
5 04/07/2021   04/01/2021   04/03/2021 By M... NA    NA    Puer... Fema... 75-84 years
6 04/07/2021   04/01/2021   04/03/2021 By M... NA    NA    Puer... Fema... 85 years a...
# i 7 more variables: `COVID-19 Deaths` <dbl>, `Total Deaths` <dbl>,
#   `Pneumonia Deaths` <dbl>, `Pneumonia and COVID-19 Deaths` <dbl>,
#   `Influenza Deaths` <dbl>, `Pneumonia, Influenza, or COVID-19 Deaths` <dbl>,
#   Footnote <chr>
```

# Take a look with `slice_sample()`

```
nchs ►  
  slice_sample(n = 10)
```

```
# A tibble: 10 × 16  
  `Data As Of` `Start Date` `End Date` Group   Year  Month State      Sex  
  <chr>        <chr>        <chr>   <chr>   <lgl> <lgl> <chr>    <chr>  
1 04/07/2021   07/01/2020   07/31/2020 By Month NA    NA    Kentucky Fema...  
2 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    New Hampshire All ...  
3 04/07/2021   01/01/2020   04/03/2021 By Total NA    NA    Virginia  All ...  
4 04/07/2021   01/01/2021   04/03/2021 By Year  NA    NA    New Jersey Fema...  
5 04/07/2021   04/01/2020   04/30/2020 By Month NA    NA    Louisiana  Male  
6 04/07/2021   02/01/2020   02/29/2020 By Month NA    NA    New Mexico Fema...  
7 04/07/2021   09/01/2020   09/30/2020 By Month NA    NA    Kentucky   Fema...  
8 04/07/2021   01/01/2021   04/03/2021 By Year  NA    NA    New York City Male  
9 04/07/2021   11/01/2020   11/30/2020 By Month NA    NA    Oregon     All ...  
10 04/07/2021   04/01/2021   04/03/2021 By Month NA    NA    Iowa       All ...  
# i 8 more variables: `Age Group` <chr>, `COVID-19 Deaths` <dbl>,  
# `Total Deaths` <dbl>, `Pneumonia Deaths` <dbl>,  
# `Pneumonia and COVID-19 Deaths` <dbl>, `Influenza Deaths` <dbl>,  
# `Pneumonia, Influenza, or COVID-19 Deaths` <dbl>, Footnote <chr>
```

# Aside: one that happened earlier ...

```
nchs %>%  
  slice_sample(n = 10)
```

```
## # A tibble: 10 x 16  
##   `Data As Of` `Start Date` `End Date` Group Year Month State Sex  
##   <chr>        <chr>        <chr>    <chr> <lgl> <lgl> <chr> <chr>  
## 1 04/07/2021  01/01/2020  04/03/2021 By Tot... NA    NA    Minnesota Male  
## 2 04/07/2021  02/01/2020  02/29/2020 By Mon... NA    NA    Georgia Male  
## 3 04/07/2021  02/01/2021  02/28/2021 By Mon... NA    NA    Maine Male  
## 4 04/07/2021  11/01/2020  11/30/2020 By Mon... NA    NA    New Jersey Female  
## 5 04/07/2021  01/01/2020  12/31/2020 By Year NA    NA    Rhode Island All Se...  
## 6 04/07/2021  01/01/2020  01/31/2020 By Mon... NA    TRUE   New York All Se...  
## 7 04/07/2021  05/01/2020  05/31/2020 By Mon... NA    NA    District of... Male  
## 8 04/07/2021  04/01/2021  04/03/2021 By Mon... NA    NA    North Carol... Female  
## 9 04/07/2021  03/01/2021  03/31/2021 By Mon... NA    NA    Kentucky Male  
## 10 04/07/2021  04/01/2021  04/03/2021 By Mon... NA    NA    New Mexico Female  
## # ... with 8 more variables: Age Group <chr>, COVID-19 Deaths <dbl>,  
## # Total Deaths <dbl>, Pneumonia Deaths <dbl>,  
## # Pneumonia and COVID-19 Deaths <dbl>, Influenza Deaths <dbl>,  
## # Pneumonia, Influenza, or COVID-19 Deaths <dbl>, Footnote <chr>
```



# Take a look with `slice()`

Let's look at the rows `read_csv()` complained about.

```
nchs ►  
slice(2750:2760)
```

```
# A tibble: 11 × 16  
  `Data As Of` `Start Date` `End Date` Group   Year Month State      Sex  
  <chr>        <chr>        <chr>   <chr>   <lg1> <lg1> <chr>      <chr>  
1 04/07/2021   01/01/2020   04/03/2021 By Total NA    NA    Puerto Rico Fema...  
2 04/07/2021   01/01/2020   04/03/2021 By Total NA    NA    Puerto Rico Fema...  
3 04/07/2021   01/01/2020   04/03/2021 By Total NA    NA    Puerto Rico Fema...  
4 04/07/2021   01/01/2020   04/03/2021 By Total NA    NA    Puerto Rico Fema...  
5 04/07/2021   01/01/2020   04/03/2021 By Total NA    NA    Puerto Rico Fema...  
6 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    United States All ...  
7 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    United States All ...  
8 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    United States All ...  
9 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    United States All ...  
10 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    United States All ...  
11 04/07/2021   01/01/2020   12/31/2020 By Year  NA    NA    United States All ...  
# i 8 more variables: `Age Group` <chr>, `COVID-19 Deaths` <dbl>,  
#   `Total Deaths` <dbl>, `Pneumonia Deaths` <dbl>,  
#   `Pneumonia and COVID-19 Deaths` <dbl>, `Influenza Deaths` <dbl>,  
#   `Pneumonia, Influenza, or COVID-19 Deaths` <dbl>, Footnote <chr>
```

# Take a look with `slice()`

```
nchs >  
  slice(2750:2760) >  
  select(Year, Month, State)
```

```
# A tibble: 11 × 3  
  Year Month State  
  <lgl> <lgl> <chr>  
1 NA    NA    Puerto Rico  
2 NA    NA    Puerto Rico  
3 NA    NA    Puerto Rico  
4 NA    NA    Puerto Rico  
5 NA    NA    Puerto Rico  
6 NA    NA    United States  
7 NA    NA    United States  
8 NA    NA    United States  
9 NA    NA    United States  
10 NA   NA    United States  
11 NA   NA    United States
```

Hm, something to do with the transition to national numbers maybe?

# Take a look with `select()` & `filter()`

```
nchs ►  
  select(Year, Month, State) ►  
  filter(State == "New York")
```

```
# A tibble: 969 × 3  
  Year Month State  
  <lgl> <lgl> <chr>  
1 NA    NA    New York  
2 NA    NA    New York  
3 NA    NA    New York  
4 NA    NA    New York  
5 NA    NA    New York  
6 NA    NA    New York  
7 NA    NA    New York  
8 NA    NA    New York  
9 NA    NA    New York  
10 NA   NA    New York  
# i 959 more rows
```

# Take a look with `is.na()`

```
nchs >  
  select(Year, Month, State) >  
  filter(!is.na(Year))
```

```
# A tibble: 0 × 3  
#   i 3 variables: Year <lgl>, Month <lgl>, State <chr>
```

It really has been read in as a completely empty column.  
That doesn't seem like it can be right.

# Take a look with `distinct()`

```
nchs >  
  select(Year) >  
  distinct(Year)
```

```
# A tibble: 1 × 1  
  Year  
  <lgl>  
1 NA
```

Again, it's been read in as a completely empty column.

# Take a look with `read_lines()`

Time to reach for a different kitchen knife.

```
read_lines(here("data", "SAS_on_2021-04-13.csv"), n_max = 10)
```

```
[1] "Data As Of,Start Date,End Date,Group,Year,Month,State,Sex,Age Group,COVID-19 Deaths,Total  
Deaths,Pneumonia Deaths,Pneumonia and COVID-19 Deaths,Influenza Deaths,\"Pneumonia, Influenza, or  
COVID-19 Deaths\",Footnote"  
[2] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,All  
Ages,539723,4161167,466437,263147,9037,750804,"  
[3] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,Under 1  
year,59,22626,246,10,21,316,"  
[4] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,0-17  
years,251,39620,667,46,179,1051,"  
[5] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,1-4  
years,31,4069,137,5,61,224,"  
[6] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,5-14  
years,89,6578,195,19,76,341,"  
[7] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,15-24  
years,804,42596,930,317,81,1493,"  
[8] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,18-29  
years,1996,75339,2184,884,150,3434,"  
[9] "04/07/2021,01/01/2020,04/03/2021,By Total,,,United States,All Sexes,25-34  
years,3543,88196,3493,1617,237,5638,"
```

# We can get the whole thing this way

```
raw_file ← read_lines(here("data", "SAS_on_2021-04-13.csv"))
```

This imports the data as a long, long character vector, with each element being a line.

```
# reminder: indexing 1D vectors  
letters[5:6]
```

```
[1] "e" "f"
```

# Now we're just looking at lines in a file

```
# This is not a tibble; we have to index it the basic way  
raw_file[2753:2758]
```

```
[1] "04/07/2021,01/01/2020,04/03/2021,By Total,,,Puerto Rico,Female,65-74 years,203,2650,410,151,,466,One or more data cells have counts  
between 1-9 and have been suppressed in accordance with NCHS confidentiality standards."  
[2] "04/07/2021,01/01/2020,04/03/2021,By Total,,,Puerto Rico,Female,75-84 years,234,4274,656,154,16,751,"  
[3] "04/07/2021,01/01/2020,04/03/2021,By Total,,,Puerto Rico,Female,85 years and over,222,6164,795,136,29,909,"  
[4] "04/07/2021,01/01/2020,12/31/2020,By Year,2020,,United States,All Sexes,All Ages,380949,3372967,349667,178222,8779,560025,"  
[5] "04/07/2021,01/01/2020,12/31/2020,By Year,2020,,United States,All Sexes,Under 1 year,48,19356,224,9,21,284,"  
[6] "04/07/2021,01/01/2020,12/31/2020,By Year,2020,,United States,All Sexes,0-17 years,189,33808,598,35,178,930,"
```

*There* you are, you bastard.

In this case, this is due to the kind of data this is, mixing multiple reporting levels and totals. That is, it's not a mistake in the *data*, but rather in the *parsing*.



# OK, let's go back to the colspec!

```
nchs ← with_edition(1, read_csv(here("data", "SAS_on_2021-04-13.csv")))
```

— Column specification —

```
cols(  
  `Data As Of` = col_character(),  
  `Start Date` = col_character(),  
  `End Date` = col_character(),  
  Group = col_character(),  
  Year = col_logical(),  
  Month = col_logical(),  
  State = col_character(),  
  Sex = col_character(),  
  `Age Group` = col_character(),  
  `COVID-19 Deaths` = col_double(),  
  `Total Deaths` = col_double(),  
  `Pneumonia Deaths` = col_double(),  
  `Pneumonia and COVID-19 Deaths` = col_double(),  
  `Influenza Deaths` = col_double(),  
  `Pneumonia, Influenza, or COVID-19 Deaths` = col_double(),  
  Footnote = col_character()  
)
```

We can just copy it from the console output! It's valid code.

# We use it with `col_types`

```
nchs <- with_edition(1, read_csv(here("data", "SAS_on_2021-04-13.csv"),
  col_types = cols(
    `Data As Of` = col_character(),
    `Start Date` = col_character(),
    `End Date` = col_character(),
    Group = col_character(),
    Year = col_logical(),
    Month = col_logical(),
    State = col_character(),
    Sex = col_character(),
    `Age Group` = col_character(),
    `COVID-19 Deaths` = col_double(),
    `Total Deaths` = col_double(),
    `Pneumonia Deaths` = col_double(),
    `Pneumonia and COVID-19 Deaths` = col_double(),
    `Influenza Deaths` = col_double(),
    `Pneumonia, Influenza, or COVID-19 Deaths` = col_double(),
    Footnote = col_character()
  )))
```

But we know we need to make some adjustments.

# Fixes

```
# Date format
us_style ← "%m/%d/%Y"

nchs ← with_edition(1, read_csv(
  here("data", "SAS_on_2021-04-13.csv"),
  col_types = cols(
    `Data As Of` = col_date(format = us_style),
    `Start Date` = col_date(format = us_style),
    `End Date` = col_date(format = us_style),
    Group = col_character(),
    Year = col_character(),
    Month = col_character(),
    State = col_character(),
    Sex = col_character(),
    `Age Group` = col_character(),
    `COVID-19 Deaths` = col_integer(),
    `Total Deaths` = col_integer(),
    `Pneumonia Deaths` = col_integer(),
    `Pneumonia and COVID-19 Deaths` = col_integer(),
    `Influenza Deaths` = col_integer(),
    `Pneumonia, Influenza, or COVID-19 Deaths` = col_integer(),
    Footnote = col_character()
  )
)
janitor::clean_names()
select(-footnote)
mutate(age_group = stringr::str_to_sentence(age_group))
filter(!stringr::str_detect(state, "Total"))
)
```

# Now let's look again

```
dim(nchs)
```

```
[1] 52326    15
```

```
nchs ►  
  select(year, month, state) ►  
  filter(!is.na(year))
```

```
# A tibble: 49,572 × 3  
  year month state  
  <chr> <chr> <chr>  
1 2020 <NA> United States  
2 2020 <NA> United States  
3 2020 <NA> United States  
4 2020 <NA> United States  
5 2020 <NA> United States  
6 2020 <NA> United States  
7 2020 <NA> United States  
8 2020 <NA> United States  
9 2020 <NA> United States  
10 2020 <NA> United States  
# i 49,562 more rows
```

# Now let's look again

```
nchs ▶  
distinct(year)
```

```
# A tibble: 3 × 1  
  year  
  <chr>  
1 <NA>  
2 2020  
3 2021
```

# Lessons learned

I said at the start that it was no fun, but also weirdly satisfying.

When `read_csv()` warns you of a parsing failure, **don't ignore it**.

`read_lines()` lets you get the file in a nearly unprocessed form.

The `colspec` output is your friend.

# If we wanted to ...

```
library(stringr) # it's back!
```

# If we wanted to ...

```
library(stringr) # it's back!  
nchs
```

```
# A tibble: 52,326 × 15  
  data_as_of start_date end_date   group   year  month state  
sex  age_group  
  <date>      <date>      <date>   <chr>   <chr> <chr> <chr>  
<chr> <chr>  
1 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... All ages  
2 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... Under 1 ...  
3 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 0-17 yea..  
4 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 1-4 years  
5 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 5-14 yea..  
6 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 15-24 ye..  
7 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 18-29 ye..  
8 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 25-34 ye..  
9 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 30-39 ye..  
10 2021-04-07 2020-01-01 2021-04-03 By Total <NA> <NA> United..  
All ... 35-44 ye..  
# i 52,316 more rows  
# i 6 more variables: covid_19_deaths <int>, total_deaths <int>,
```



# If we wanted to ...

```
library(stringr) # it's back!  
  
nchs ►  
  select(!(c(data_as_of:end_date, year, month)))
```

```
# A tibble: 52,326 × 10  
  group state sex age_group covid_19_deaths total_deaths  
pneumonia_deaths  
  <chr>    <chr> <chr> <chr>          <int>         <int>  
<int>  
1 By Total Unite... All ... All ages          539723        4161167  
466437  
2 By Total Unite... All ... Under 1 ...          59          22626  
246  
3 By Total Unite... All ... 0-17 yea...        251          39620  
667  
4 By Total Unite... All ... 1-4 years          31           4069  
137  
5 By Total Unite... All ... 5-14 yea...          89           6578  
195  
6 By Total Unite... All ... 15-24 ye...        804          42596  
930  
7 By Total Unite... All ... 18-29 ye...       1996          75339  
2184  
8 By Total Unite... All ... 25-34 ye...       3543          88196  
3493  
9 By Total Unite... All ... 30-39 ye...       5792         107348  
5276  
10 By Total Unite... All ... 35-44 ye...       9259         126848  
8203  
# i 52,316 more rows  
# i 3 more variables: pneumonia_and_covid_19_deaths <int>,
```

# If we wanted to ...

```
library(stringr) # it's back!

nchs >
  select(!(c(data_as_of:end_date, year, month))) >
  pivot_longer(covid_19_deaths:pneumonia_influenza_or_covid_19_deaths,
    names_to = "outcome",
    values_to = "n")
```

```
# A tibble: 313,956 × 6
  group      state      sex      age_group      outcome
  n
  <chr>      <chr>      <chr>      <chr>      <chr>
<int>
1 By Total United States All Sexes All ages covid_19_deaths
5.40e5
2 By Total United States All Sexes All ages total_deaths
4.16e6
3 By Total United States All Sexes All ages pneumonia_deaths
4.66e5
4 By Total United States All Sexes All ages pneumonia_and_covid_19_... 2.63e5
5 By Total United States All Sexes All ages influenza_deaths
9.04e3
6 By Total United States All Sexes All ages pneumonia_influenza_or_... 7.51e5
7 By Total United States All Sexes Under 1 year covid_19_deaths
5.9 e1
8 By Total United States All Sexes Under 1 year total_deaths
2.26e4
9 By Total United States All Sexes Under 1 year pneumonia_deaths
2.46e2
10 By Total United States All Sexes Under 1 year pneumonia_and_covid_19_... 1 e1
# i 313,946 more rows
```

# If we wanted to ...

```
library(stringr) # it's back!

nchs >
  select(!c(data_as_of:end_date, year, month))) >
  pivot_longer(covid_19_deaths:pneumonia_influenza_or_covid_19_deaths,
    names_to = "outcome",
    values_to = "n") >
  mutate(outcome = str_to_sentence(outcome),
    outcome = str_replace_all(outcome, "_", " "),
    outcome = str_replace(outcome, "(C|c)ovid 19", "COVID-19"))
```

```
# A tibble: 313,956 × 6
  group      state      sex      age_group      outcome
  <int>      <chr>      <chr>      <chr>      <chr>
1 By Total United States All Sexes All ages COVID-19 deaths
5.40e5
2 By Total United States All Sexes All ages Total deaths
4.16e6
3 By Total United States All Sexes All ages Pneumonia deaths
4.66e5
4 By Total United States All Sexes All ages Pneumonia and
COVID-19 ... 2.63e5
5 By Total United States All Sexes All ages Influenza deaths
9.04e3
6 By Total United States All Sexes All ages Pneumonia
influenza or ... 7.51e5
7 By Total United States All Sexes Under 1 year COVID-19 deaths
5.9 e1
8 By Total United States All Sexes Under 1 year Total deaths
2.26e4
9 By Total United States All Sexes Under 1 year Pneumonia deaths
2.46e2
10 By Total United States All Sexes Under 1 year Pneumonia and
COVID-19 ... 1 e1
# i 313,946 more rows
```

# If we wanted to ...

Put this in an object called `nchs_fmt`

# ... we could make a table or graph

```
nchs_fmt ►  
select(state, age_group, outcome, n)
```

```
# A tibble: 313,956 × 4  
  state      age_group outcome      n  
  <chr>    <chr>    <chr>    <int>  
1 United States All ages COVID-19 deaths 539723  
2 United States All ages Total deaths 4161167  
3 United States All ages Pneumonia deaths 466437  
4 United States All ages Pneumonia and COVID-19 deaths 263147  
5 United States All ages Influenza deaths 9037  
6 United States All ages Pneumonia influenza or COVID-19 deaths 750804  
7 United States Under 1 year COVID-19 deaths 59  
8 United States Under 1 year Total deaths 22626  
9 United States Under 1 year Pneumonia deaths 246  
10 United States Under 1 year Pneumonia and COVID-19 deaths 10  
# i 313,946 more rows
```

# Cleaned up (but not tidy)

```
nchs_fmt ▶  
  distinct(group)
```

```
# A tibble: 3 × 1  
  group  
  <chr>  
1 By Total  
2 By Year  
3 By Month
```

# Cleaned up (but not tidy)

```
nchs_fmt ▶  
  distinct(group)
```

```
# A tibble: 3 × 1  
  group  
  <chr>  
1 By Total  
2 By Year  
3 By Month
```

```
nchs_fmt ▶  
  distinct(age_group)
```

```
# A tibble: 17 × 1  
  age_group  
  <chr>  
1 All ages  
2 Under 1 year  
3 0-17 years  
4 1-4 years  
5 5-14 years  
6 15-24 years  
7 18-29 years  
8 25-34 years  
9 30-39 years  
10 35-44 years  
11 40-49 years  
12 45-54 years  
13 50-64 years  
14 55-64 years  
15 65-74 years  
16 75-84 years
```

# Make our plot

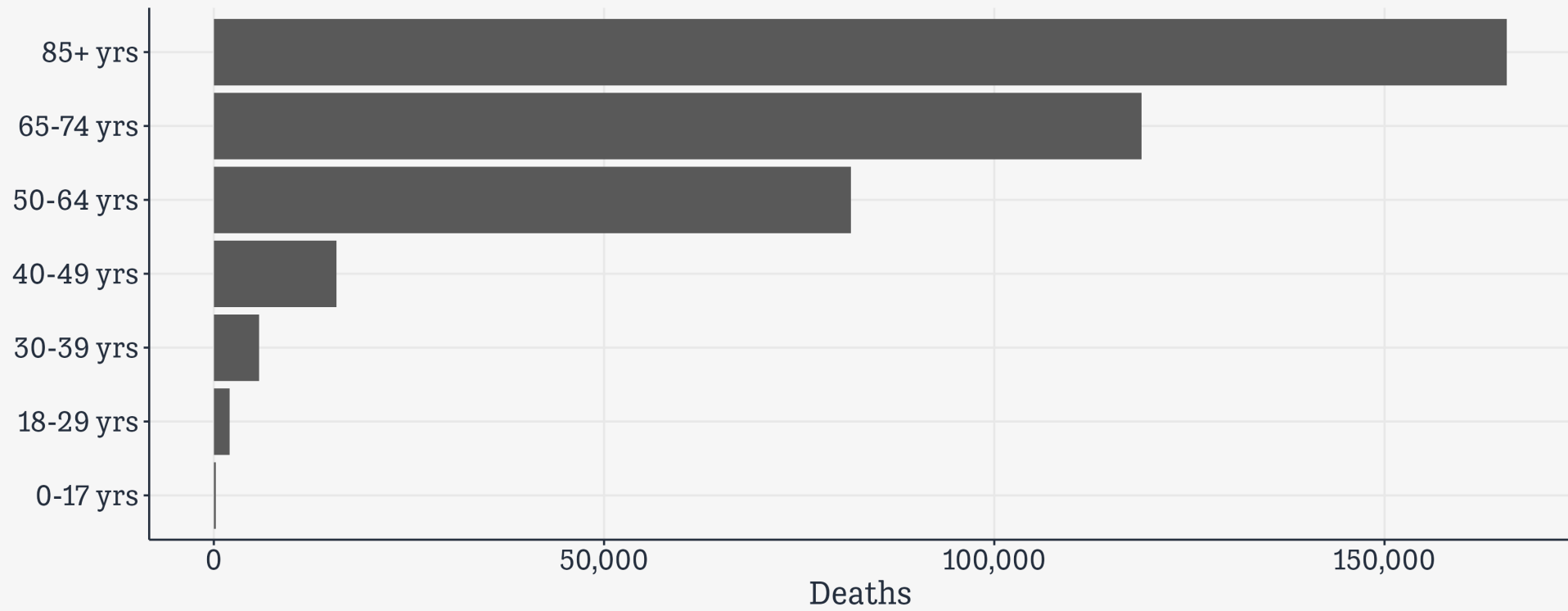
```
p_out ← nchs_fmt ▷
  filter(group %in% "By Total",
         sex %in% "All Sexes",
         state %in% "United States",
         age_group %in% c("0-17 years",
                          "18-29 years",
                          "30-39 years",
                          "40-49 years",
                          "50-64 years",
                          "65-74 years",
                          "85 years and over"),
         outcome %in% "COVID-19 deaths") ▷
  mutate(age_group = str_replace(age_group, "years", "yrs"),
         age_group = str_replace(age_group, " and over", ""),
         age_group = str_replace(age_group, "85", "85+")) ▷
  ggplot(mapping = aes(x = n, y = age_group)) +
  geom_col() + scale_x_continuous(labels = scales::comma) +
  labs(x = "Deaths", y = NULL, title = "U.S. COVID-19 mortality totals by age group")
```



# Result

p\_out

U.S. COVID-19 mortality totals by age group



Every dataset is  
different

**Dropping missings**

# Dropping missing values

```
df ← tribble(  
  ~a, ~b, ~c,  
  1, NA, 2,  
  NA, NA, NA,  
  2, 2, 2  
)
```

df

```
# A tibble: 3 × 3  
      a      b      c  
  <dbl> <dbl> <dbl>  
1     1    NA     2  
2    NA    NA    NA  
3     2     2     2
```

# Dropping missing values

```
# 2 Convenience function
```

```
df ▷
```

```
  drop_na()
```

```
# A tibble: 1 × 3
```

	a	b	c
	<dbl>	<dbl>	<dbl>
1	2	2	2

Drops all rows with *any* missing cases.

# Dropping missing values

What if we only want to drop all rows with *all* missing cases?

```
# 3  
df ►  
  # Anonymous function \(\x)  
  filter(!if_all(everything(), \(\x) is.na(x)))
```

```
# A tibble: 2 × 3  
      a      b      c  
  <dbl> <dbl> <dbl>  
1     1    NA     2  
2     2     2     2
```

```
# 4 Convenience function from janitor  
df ►  
  janitor::remove_empty("rows")
```

```
# A tibble: 2 × 3  
      a      b      c  
  <dbl> <dbl> <dbl>  
1     1    NA     2  
2     2     2     2
```

**Example: cleaning a table**

# Cleaning a table

With that in mind ... Some marketing data

SEGMENT	DESCRIPTION	R	F	M
Champions	Bought recently, buy often and spend the most	4- 5	4- 5	4- 5
Loyal Customers	Spend good money. Responsive to promotions	2- 5	3- 5	3- 5
Potential Loyalist	Recent customers, spent good amount, bought more than once	3- 5	1- 3	1- 3
New Customers	Bought more recently, but not often	4- 5	<= 1	<= 1
Promising	Recent shoppers, but haven't spent much	3- 4	<= 1	<= 1
Need Attention	Above average recency, frequency & monetary values	2- 3	2- 3	2- 3
About To Sleep	Below average recency, frequency & monetary values	2- 3	<= 2	<= 2
At Risk	Spent big money, purchased often but long time ago	<= 2	2- 5	2- 5
Can't Lose Them	Made big purchases and often, but long time ago	<= 1	4- 5	4- 5
Hibernating	Low spenders, low frequency, purchased long time ago	1- 2	1- 2	1- 2
Lost	Lowest recency, frequency & monetary scores	<= 2	<= 2	<= 2



# Cleaning a table

```
read_csv(here("data", "rfm_table.csv"))
```

```
# A tibble: 23 × 5
  SEGMENT DESCRIPTION R F M
  <chr> <chr> <chr> <chr> <chr>
1 <NA> <NA> <NA> <NA> <NA>
2 Champions Bought recently, buy often and spend th... 4- 5 4- 5 4- 5
3 <NA> <NA> <NA> <NA> <NA>
4 Loyal Customers Spend good money. Responsive to promoti... 2- 5 3- 5 3- 5
5 <NA> <NA> <NA> <NA> <NA>
6 Potential Loyalist Recent customers, spent good amount, bo... 3- 5 1- 3 1- 3
7 <NA> <NA> <NA> <NA> <NA>
8 New Customers Bought more recently, but not often 4- 5 ≤ 1 ≤ 1
9 <NA> <NA> <NA> <NA> <NA>
10 Promising Recent shoppers, but haven't spent much 3- 4 ≤ 1 ≤ 1
# i 13 more rows
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names()
```

```
# A tibble: 23 × 5
  segment      description      r      f      m
  <chr>      <chr>      <chr> <chr> <chr>
1 <NA>      <NA>      <NA> <NA> <NA>
2 Champions Bought recently, buy often and spend th... 4- 5 4- 5 4- 5
3 <NA>      <NA>      <NA> <NA> <NA>
4 Loyal Customers Spend good money. Responsive to promoti... 2- 5 3- 5 3- 5
5 <NA>      <NA>      <NA> <NA> <NA>
6 Potential Loyalist Recent customers, spent good amount, bo... 3- 5 1- 3 1- 3
7 <NA>      <NA>      <NA> <NA> <NA>
8 New Customers Bought more recently, but not often      4- 5 ≤ 1 ≤ 1
9 <NA>      <NA>      <NA> <NA> <NA>
10 Promising Recent shoppers, but haven't spent much 3- 4 ≤ 1 ≤ 1
# i 13 more rows
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names() >
janitor::remove_empty("rows")
```

```
# A tibble: 11 × 5
  segment      description      r      f      m
  <chr>      <chr>      <chr> <chr> <chr>
1 Champions Bought recently, buy often and spend th... 4- 5 4- 5 4- 5
2 Loyal Customers Spend good money. Responsive to promoti... 2- 5 3- 5 3- 5
3 Potential Loyalist Recent customers, spent good amount, bo... 3- 5 1- 3 1- 3
4 New Customers Bought more recently, but not often 4- 5 ≤ 1 ≤ 1
5 Promising Recent shoppers, but haven't spent much 3- 4 ≤ 1 ≤ 1
6 Need Attention Above average recency, frequency & mone... 2- 3 2- 3 2- 3
7 About To Sleep Below average recency, frequency & mone... 2- 3 ≤ 2 ≤ 2
8 At Risk Spent big money, purchased often but lo... ≤ 2 2- 5 2- 5
9 Can't Lose Them Made big purchases and often, but long ... ≤ 1 4- 5 4- 5
10 Hibernating Low spenders, low frequency, purchased ... 1- 2 1- 2 1- 2
11 Lost Lowest recency, frequency & monetary sc... ≤ 2 ≤ 2 ≤ 2
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names() >
janitor::remove_empty("rows") >
pivot_longer(cols = r:m)
```

```
# A tibble: 33 × 4
  segment      description name value
  <chr>      <chr>      <chr> <chr>
1 Champions Bought recently, buy often and spend the most r 4- 5
2 Champions Bought recently, buy often and spend the most f 4- 5
3 Champions Bought recently, buy often and spend the most m 4- 5
4 Loyal Customers Spend good money. Responsive to promotions r 2- 5
5 Loyal Customers Spend good money. Responsive to promotions f 3- 5
6 Loyal Customers Spend good money. Responsive to promotions m 3- 5
7 Potential Loyalist Recent customers, spent good amount, bought m... r 3- 5
8 Potential Loyalist Recent customers, spent good amount, bought m... f 1- 3
9 Potential Loyalist Recent customers, spent good amount, bought m... m 1- 3
10 New Customers Bought more recently, but not often r 4- 5
# i 23 more rows
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names() >
janitor::remove_empty("rows") >
pivot_longer(cols = r:m) >
separate(col = value, into = c("lo", "hi"),
         remove = FALSE, convert = TRUE,
         fill = "left")
```

```
# A tibble: 33 × 6
  segment      description name value  lo  hi
  <chr>      <chr>      <chr> <chr> <int> <int>
1 Champions Bought recently, buy often and sp... r    4- 5    4    5
2 Champions Bought recently, buy often and sp... f    4- 5    4    5
3 Champions Bought recently, buy often and sp... m    4- 5    4    5
4 Loyal Customers Spend good money. Responsive to p... r    2- 5    2    5
5 Loyal Customers Spend good money. Responsive to p... f    3- 5    3    5
6 Loyal Customers Spend good money. Responsive to p... m    3- 5    3    5
7 Potential Loyalist Recent customers, spent good amou... r    3- 5    3    5
8 Potential Loyalist Recent customers, spent good amou... f    1- 3    1    3
9 Potential Loyalist Recent customers, spent good amou... m    1- 3    1    3
10 New Customers Bought more recently, but not oft... r    4- 5    4    5
# i 23 more rows
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names() >
janitor::remove_empty("rows") >
pivot_longer(cols = r:m) >
separate(col = value, into = c("lo", "hi"),
         remove = FALSE, convert = TRUE,
         fill = "left") >
select(-value)
```

```
# A tibble: 33 × 5
  segment      description name      lo      hi
  <chr>      <chr>      <chr> <int> <int>
1 Champions Bought recently, buy often and spend th... r         4      5
2 Champions Bought recently, buy often and spend th... f         4      5
3 Champions Bought recently, buy often and spend th... m         4      5
4 Loyal Customers Spend good money. Responsive to promoti... r         2      5
5 Loyal Customers Spend good money. Responsive to promoti... f         3      5
6 Loyal Customers Spend good money. Responsive to promoti... m         3      5
7 Potential Loyalist Recent customers, spent good amount, bo... r         3      5
8 Potential Loyalist Recent customers, spent good amount, bo... f         1      3
9 Potential Loyalist Recent customers, spent good amount, bo... m         1      3
10 New Customers Bought more recently, but not often      r         4      5
# i 23 more rows
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names() >
janitor::remove_empty("rows") >
pivot_longer(cols = r:m) >
separate(col = value, into = c("lo", "hi"),
         remove = FALSE, convert = TRUE,
         fill = "left") >
select(-value) >
pivot_wider(names_from = name,
            values_from = lo:hi)
```

```
# A tibble: 11 × 8
  segment      description lo_r lo_f lo_m hi_r hi_f hi_m
  <chr>      <chr>      <int> <int> <int> <int> <int> <int>
1 Champions Bought recently, buy ...     4     4     4     5     5     5
2 Loyal Customers Spend good money. Res...     2     3     3     5     5     5
3 Potential Loyalist Recent customers, spe...     3     1     1     5     3     3
4 New Customers Bought more recently,...     4    NA    NA     5     1     1
5 Promising Recent shoppers, but ...     3    NA    NA     4     1     1
6 Need Attention Above average recency...     2     2     2     3     3     3
7 About To Sleep Below average recency...     2    NA    NA     3     2     2
8 At Risk Spent big money, purc...    NA     2     2     2     5     5
9 Can't Lose Them Made big purchases an...    NA     4     4     1     5     5
10 Hibernating Low spenders, low fre...     1     1     1     2     2     2
11 Lost Lowest recency, frequ...    NA    NA    NA     2     2     2
```

# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) >
janitor::clean_names() >
janitor::remove_empty("rows") >
pivot_longer(cols = r:m) >
separate(col = value, into = c("lo", "hi"),
         remove = FALSE, convert = TRUE,
         fill = "left") >
select(-value) >
pivot_wider(names_from = name,
            values_from = lo:hi) >
mutate(across(where(is.integer), replace_na, 0))
```

```
# A tibble: 11 × 8
  segment description lo_r lo_f lo_m hi_r hi_f hi_m
  <chr>    <chr>    <int> <int> <int> <int> <int> <int>
1 Champions Bought recently, buy ... 4 4 4 5 5 5
2 Loyal Customers Spend good money. Res... 2 3 3 5 5 5
3 Potential Loyalist Recent customers, spe... 3 1 1 5 3 3
4 New Customers Bought more recently,... 4 0 0 5 1 1
5 Promising Recent shoppers, but ... 3 0 0 4 1 1
6 Need Attention Above average recency... 2 2 2 3 3 3
7 About To Sleep Below average recency... 2 0 0 3 2 2
8 At Risk Spent big money, purc... 0 2 2 2 5 5
9 Can't Lose Them Made big purchases an... 0 4 4 1 5 5
10 Hibernating Low spenders, low fre... 1 1 1 2 2 2
11 Lost Lowest recency, frequ... 0 0 0 2 2 2
```



# Cleaning a table

```
read_csv(here("data", "rfm_table.csv")) ▷  
janitor::clean_names() ▷  
janitor::remove_empty("rows") ▷  
pivot_longer(cols = r:m) ▷  
separate(col = value, into = c("lo", "hi"),  
         remove = FALSE, convert = TRUE,  
         fill = "left") ▷  
select(-value) ▷  
pivot_wider(names_from = name,  
            values_from = lo:hi) ▷  
mutate(across(where(is.integer), replace_na, 0)) ▷  
select(segment,  
       lo_r, hi_r,  
       lo_f, hi_f,  
       lo_m, hi_m,  
       description)
```

```
# A tibble: 11 × 8  
  segment lo_r hi_r lo_f hi_f lo_m hi_m description  
  <chr>   <int> <int> <int> <int> <int> <int> <chr>  
1 Champions      4     5     4     5     4     5 Bought recently, buy ...  
2 Loyal Customers  2     5     3     5     3     5 Spend good money. Res...  
3 Potential Loyalist 3     5     1     3     1     3 Recent customers, spe...  
4 New Customers   4     5     0     1     0     1 Bought more recently,...  
5 Promising       3     4     0     1     0     1 Recent shoppers, but ...  
6 Need Attention  2     3     2     3     2     3 Above average recency...  
7 About To Sleep  2     3     0     2     0     2 Below average recency...  
8 At Risk         0     2     2     5     2     5 Spent big money, purc...  
9 Can't Lose Them 0     1     4     5     4     5 Made big purchases an...  
10 Hibernating    1     2     1     2     1     2 Low spenders, low fre...  
11 Lost           0     2     0     2     0     2 Lowest recency, frequ...
```

# A candidate for `rowwise()`?

```
rfm_table
```

```
# A tibble: 11 × 8
```

	segment	lo_r	hi_r	lo_f	hi_f	lo_m	hi_m	description
	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	Champions	4	5	4	5	4	5	Bought recently, buy ...
2	Loyal Customers	2	5	3	5	3	5	Spend good money. Res...
3	Potential Loyalist	3	5	1	3	1	3	Recent customers, spe...
4	New Customers	4	5	0	1	0	1	Bought more recently,...
5	Promising	3	4	0	1	0	1	Recent shoppers, but ...
6	Need Attention	2	3	2	3	2	3	Above average recency...
7	About To Sleep	2	3	0	2	0	2	Below average recency...
8	At Risk	0	2	2	5	2	5	Spent big money, purc...
9	Can't Lose Them	0	1	4	5	4	5	Made big purchases an...
10	Hibernating	1	2	1	2	1	2	Low spenders, low fre...
11	Lost	0	2	0	2	0	2	Lowest recency, frequ...

# A candidate for `rowwise()`?

This does what we expect:

```
rfm_table >
  mutate(sum_lo = lo_r + lo_f + lo_m,
         sum_hi = hi_r + hi_f + hi_m) >
  select(segment, sum_lo, sum_hi, everything())
```

# A tibble: 11 × 10

	segment <chr>	sum_lo <int>	sum_hi <int>	lo_r <int>	hi_r <int>	lo_f <int>	hi_f <int>	lo_m <int>	hi_m <int>	description <chr>
1	Champions	12	15	4	5	4	5	4	5	Bought rec...
2	Loyal Customers	8	15	2	5	3	5	3	5	Spend good...
3	Potential Loya...	5	11	3	5	1	3	1	3	Recent cus...
4	New Customers	4	7	4	5	0	1	0	1	Bought mor...
5	Promising	3	6	3	4	0	1	0	1	Recent sho...
6	Need Attention	6	9	2	3	2	3	2	3	Above aver...
7	About To Sleep	2	7	2	3	0	2	0	2	Below aver...
8	At Risk	4	12	0	2	2	5	2	5	Spent big ...
9	Can't Lose Them	8	11	0	1	4	5	4	5	Made big p...
10	Hibernating	3	6	1	2	1	2	1	2	Low spende...
11	Lost	0	6	0	2	0	2	0	2	Lowest rec...

This adds each column, elementwise.

# A candidate for `rowwise()`?

But this does not:

```
rfm_table >
  mutate(sum_lo = sum(lo_r, lo_f, lo_m),
         sum_hi = sum(hi_r, hi_f, hi_m)) >
  select(segment, sum_lo, sum_hi, everything())
```

# A tibble: 11 × 10

	segment	sum_lo	sum_hi	lo_r	hi_r	lo_f	hi_f	lo_m	hi_m	description
	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	Champions	55	105	4	5	4	5	4	5	Bought rec...
2	Loyal Customers	55	105	2	5	3	5	3	5	Spend good...
3	Potential Loya...	55	105	3	5	1	3	1	3	Recent cus...
4	New Customers	55	105	4	5	0	1	0	1	Bought mor...
5	Promising	55	105	3	4	0	1	0	1	Recent sho...
6	Need Attention	55	105	2	3	2	3	2	3	Above aver...
7	About To Sleep	55	105	2	3	0	2	0	2	Below aver...
8	At Risk	55	105	0	2	2	5	2	5	Spent big ...
9	Can't Lose Them	55	105	0	1	4	5	4	5	Made big p...
10	Hibernating	55	105	1	2	1	2	1	2	Low spende...
11	Lost	55	105	0	2	0	2	0	2	Lowest rec...

Sum is taking all the columns, adding them up (into a single number), and putting that result in each row

# A candidate for `rowwise()`?

Similarly, this will not give the answer we probably expect:

```
rfm_table >
  mutate(mean_lo = mean(c(lo_r, lo_f, lo_m)),
         mean_hi = mean(c(hi_r, hi_f, hi_m))) >
  select(segment, mean_lo, mean_hi, everything())
```

# A tibble: 11 × 10

	segment	mean_lo	mean_hi	lo_r	hi_r	lo_f	hi_f	lo_m	hi_m	description
	<chr>	<dbl>	<dbl>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	Champions	1.67	3.18	4	5	4	5	4	5	Bought rec...
2	Loyal Custom...	1.67	3.18	2	5	3	5	3	5	Spend good...
3	Potential Lo...	1.67	3.18	3	5	1	3	1	3	Recent cus...
4	New Customers	1.67	3.18	4	5	0	1	0	1	Bought mor...
5	Promising	1.67	3.18	3	4	0	1	0	1	Recent sho...
6	Need Attenti...	1.67	3.18	2	3	2	3	2	3	Above aver...
7	About To Sle...	1.67	3.18	2	3	0	2	0	2	Below aver...
8	At Risk	1.67	3.18	0	2	2	5	2	5	Spent big ...
9	Can't Lose T...	1.67	3.18	0	1	4	5	4	5	Made big p...
10	Hibernating	1.67	3.18	1	2	1	2	1	2	Low spende...
11	Lost	1.67	3.18	0	2	0	2	0	2	Lowest rec...

# A candidate for `rowwise()`?

But this will:

```
rfm_table >
  rowwise() >
  mutate(mean_lo = mean(c(lo_r, lo_f, lo_m)),
         mean_hi = mean(c(hi_r, hi_f, hi_m))) >
  select(segment, mean_lo, mean_hi, everything())
```

# A tibble: 11 × 10

# Rowwise:

	segment	mean_lo	mean_hi	lo_r	hi_r	lo_f	hi_f	lo_m	hi_m	description
	<chr>	<dbl>	<dbl>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	Champions	4	5	4	5	4	5	4	5	Bought rec...
2	Loyal Custom...	2.67	5	2	5	3	5	3	5	Spend good...
3	Potential Lo...	1.67	3.67	3	5	1	3	1	3	Recent cus...
4	New Customers	1.33	2.33	4	5	0	1	0	1	Bought mor...
5	Promising	1	2	3	4	0	1	0	1	Recent sho...
6	Need Attenti...	2	3	2	3	2	3	2	3	Above aver...
7	About To Sle...	0.667	2.33	2	3	0	2	0	2	Below aver...
8	At Risk	1.33	4	0	2	2	5	2	5	Spent big ...
9	Can't Lose T...	2.67	3.67	0	1	4	5	4	5	Made big p...
10	Hibernating	1	2	1	2	1	2	1	2	Low spende...
11	Lost	0	2	0	2	0	2	0	2	Lowest rec...

# Rowwise isn't very efficient

In general, you'll want to see if some vectorized (“operating on columns, but elementwise”) function exists, as it'll be faster.

And most of the time, R and the tidyverse “wants” you to work in vectorized, columnar terms ... hence your first move will often be to pivot the data into long format.

So, **rowwise()** is not likely to see a whole lot of further development.

# You may want `group_by()` instead

```
rfm_table >
  group_by(segment) >
  mutate(mean_lo = mean(c(lo_r, lo_f, lo_m)),
         mean_hi = mean(c(hi_r, hi_f, hi_m))) >
  select(segment, mean_lo, mean_hi, everything())
```

# A tibble: 11 × 10

# Groups: segment [11]

	segment	mean_lo	mean_hi	lo_r	hi_r	lo_f	hi_f	lo_m	hi_m	description
	<chr>	<dbl>	<dbl>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	Champions	4	5	4	5	4	5	4	5	Bought rec...
2	Loyal Custom...	2.67	5	2	5	3	5	3	5	Spend good...
3	Potential Lo...	1.67	3.67	3	5	1	3	1	3	Recent cus...
4	New Customers	1.33	2.33	4	5	0	1	0	1	Bought mor...
5	Promising	1	2	3	4	0	1	0	1	Recent sho...
6	Need Attenti...	2	3	2	3	2	3	2	3	Above aver...
7	About To Sle...	0.667	2.33	2	3	0	2	0	2	Below aver...
8	At Risk	1.33	4	0	2	2	5	2	5	Spent big ...
9	Can't Lose T...	2.67	3.67	0	1	4	5	4	5	Made big p...
10	Hibernating	1	2	1	2	1	2	1	2	Low spende...
11	Lost	0	2	0	2	0	2	0	2	Lowest rec...



# You may want `group_by()` instead

```
rfm_table >
  group_by(segment) >
  mutate(sum_lo = sum(lo_r, lo_f, lo_m),
         sum_hi = sum(hi_r, hi_f, hi_m)) >
  select(segment, sum_lo, sum_hi, everything())
```

# A tibble: 11 × 10

# Groups: segment [11]

	segment	sum_lo	sum_hi	lo_r	hi_r	lo_f	hi_f	lo_m	hi_m	description
	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	Champions	12	15	4	5	4	5	4	5	Bought rec...
2	Loyal Customers	8	15	2	5	3	5	3	5	Spend good...
3	Potential Loya...	5	11	3	5	1	3	1	3	Recent cus...
4	New Customers	4	7	4	5	0	1	0	1	Bought mor...
5	Promising	3	6	3	4	0	1	0	1	Recent sho...
6	Need Attention	6	9	2	3	2	3	2	3	Above aver...
7	About To Sleep	2	7	2	3	0	2	0	2	Below aver...
8	At Risk	4	12	0	2	2	5	2	5	Spent big ...
9	Can't Lose Them	8	11	0	1	4	5	4	5	Made big p...
10	Hibernating	3	6	1	2	1	2	1	2	Low spende...
11	Lost	0	6	0	2	0	2	0	2	Lowest rec...

# Foreign formats

**What about  
Stata?**

# Using **haven**

Haven is the Tidyverse's package for reading and managing files from Stata, SPSS, and SAS. You should prefer it to the older Base R package **foreign**, which has similar functionality.

We're going to import a General Social Survey dataset that's in Stata's **.dta** format.

```
library(haven)

# This will take a moment
gss_panel ← read_stata(here("data", "gss_panel_long.dta"))
```

We'll do some of the common recoding and reorganizing tasks that accompany this.

# The GSS panel

The data:

```
gss_panel
```

```
# A tibble: 14,610 × 2,757
  firstyear firstid  year    id vpsu  vstrat adults ballot dateintv famgen
    <dbl> <dbl+lbl> <dbl> <dbl> <dbl+> <dbl+> <dbl+> <dbl+1> <dbl+1b> <dbl+1>
1     2006 9      2006 9      2  1957      1 3 [BAL... 709      1 [1 G...
2     2006 9      2008 3001  NA    NA      2 3 [BAL... 503      1 [1 G...
3     2006 9      2010 6001 NA(i)  NA      2 3 [BAL... 508      1 [1 G...
4     2006 10     2010 6002 NA(i)  NA      1 1 [BAL... 408      1 [1 G...
5     2006 10     2006 10      2  1957      2 1 [BAL... 630      2 [2 G...
6     2006 10     2008 3002  NA    NA      2 1 [BAL... 426      2 [2 G...
7     2006 11     2008 3003  NA    NA      2 3 [BAL... 718      4 [2 G...
8     2006 11     2010 6003 NA(i)  NA    NA(n)  3 [BAL... 518      2 [2 G...
9     2006 11     2006 11      2  1957      2 3 [BAL... 630      4 [2 G...
10    2006 12     2010 6004 NA(i)  NA      4 1 [BAL... 324      2 [2 G...
# i 14,600 more rows
# i 2,747 more variables: form <dbl+lbl>, formwt <dbl>, gender1 <dbl+lbl>,
# hompop <dbl+lbl>, intage <dbl+lbl>, intid <dbl+lbl>, intyrs <dbl+lbl>,
# mode <dbl+lbl>, oversamp <dbl>, phase <dbl+lbl>, race <dbl+lbl>,
# reg16 <dbl+lbl>, region <dbl+lbl>, relate1 <dbl+lbl>, relhh1 <dbl+lbl>,
# relhhd1 <dbl+lbl>, respnum <dbl+lbl>, rvisitor <dbl+lbl>,
```

# The GSS panel

Many variables.

Stata's missing value types are preserved

Data types are things like `dbl+lbl` indicating that Stata's numeric values and variable labels have been preserved.

# The GSS panel

You can see the labeling system at work:

```
gss_panel ▶  
  select(degree) ▶  
  group_by(degree) ▶  
  tally()
```

```
# A tibble: 6 × 2  
  degree          n  
  <dbl+lbl>    <int>  
1      0 [LT HIGH SCHOOL] 1850  
2      1 [HIGH SCHOOL]    7274  
3      2 [JUNIOR COLLEGE] 1161  
4      3 [bachelor]        2767  
5      4 [graduate]       1556  
6 NA(d)              2
```

# The GSS panel

Values get pivoted, not labels, though.

```
gss_panel >
  select(sex, degree) >
  group_by(sex, degree) >
  tally() >
  pivot_wider(names_from = sex, values_from = n)
```

```
# A tibble: 6 × 3
  degree                `1`    `2`
  <dbl+lbl>          <int> <int>
1      0 [LT HIGH SCHOOL]    814  1036
2      1 [HIGH SCHOOL]      3131  4143
3      2 [JUNIOR COLLEGE]    440   721
4      3 [bachelor]         1293  1474
5      4 [graduate]          696   860
6 NA(d)                  NA      2
```



# The GSS panel

Option 1: Just drop all the labels.

```
gss_panel >
  zap_missing() >
  zap_labels()
```

```
# A tibble: 14,610 × 2,757
  firstyear firstid year   id vpsu vstrat adults ballot dateintv famgen
    <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>    <dbl> <dbl>
1     2006      9  2006     9     2  1957     1     3     709     1
2     2006      9  2008  3001    NA    NA     2     3     503     1
3     2006      9  2010  6001    NA    NA     2     3     508     1
4     2006     10  2010  6002    NA    NA     1     1     408     1
5     2006     10  2006     10     2  1957     2     1     630     2
6     2006     10  2008  3002    NA    NA     2     1     426     2
7     2006     11  2008  3003    NA    NA     2     3     718     4
8     2006     11  2010  6003    NA    NA    NA     3     518     2
9     2006     11  2006     11     2  1957     2     3     630     4
10    2006     12  2010  6004    NA    NA     4     1     324     2
# i 14,600 more rows
# i 2,747 more variables: form <dbl>, formwt <dbl>, gender1 <dbl>,
# hompop <dbl>, intage <dbl>, intid <dbl>, intyrs <dbl>, mode <dbl>,
# oversamp <dbl>, phase <dbl>, race <dbl>, reg16 <dbl>, region <dbl>,
# relate1 <dbl>, relhh1 <dbl>, relhhd1 <dbl>, respnum <dbl>, rvisitor <dbl>,
# samPCODE <dbl>, sample <dbl>, sex <dbl>, size <dbl>, spaneng <dbl>,
```

# The GSS panel

## Option 2: Convert the labels

Let's focus on a few measures of interest, and do some recoding.

```
## Categorical vars
cat_vars ← c("race", "sex", "degree", "relig", "income", "polviews", "fefam")

## Integer vars
int_vars ← c("year", "id", "ballot", "age", "tvhours")

## Survey design
wt_vars ← c("vpsu",
            "vstrat",
            "oversamp",
            "formwt",      # weight to deal with experimental randomization
            "wtssall",     # weight variable
            "sampcode",    # sampling error code
            "sample")      # sampling frame and method

my_gss_vars ← c(int_vars, cat_vars, wt_vars)
```

# Cut down the dataset

```
gss_sub ← gss_panel ▷  
  select(all_of(my_gss_vars))
```

```
gss_sub
```

```
# A tibble: 14,610 × 19
```

	year	id	ballot	age	tvhours	race	sex	degree	relig
	<dbl>	<dbl>	<dbl+lbl>	<dbl+lb>	<dbl+lbl>	<dbl+l>	<dbl+l>	<dbl+l>	<dbl+l>
1	2006	9	3 [BALLOT C]	23	NA(a) [iap]	2 [bla...	2 [fem...	3 [bac...	4 [non...
2	2008	3001	3 [BALLOT C]	25	NA(i)	3 [oth...	2 [fem...	3 [bac...	4 [non...
3	2010	6001	3 [BALLOT C]	27	NA(i)	2 [bla...	2 [fem...	3 [bac...	4 [non...
4	2010	6002	1 [BALLOT A]	36	3	1 [whi...	2 [fem...	4 [gra...	4 [non...
5	2006	10	1 [BALLOT A]	32	3	3 [oth...	2 [fem...	4 [gra...	4 [non...
6	2008	3002	1 [BALLOT A]	34	3	3 [oth...	2 [fem...	4 [gra...	4 [non...
7	2008	3003	3 [BALLOT C]	83	NA(i)	2 [bla...	2 [fem...	0 [LT ...	1 [pro...
8	2010	6003	3 [BALLOT C]	85	NA(i)	2 [bla...	2 [fem...	0 [LT ...	1 [pro...
9	2006	11	3 [BALLOT C]	81	NA(a) [iap]	2 [bla...	2 [fem...	0 [LT ...	1 [pro...
10	2010	6004	1 [BALLOT A]	51	10	3 [oth...	1 [mal...	1 [HIG...	2 [cat...

```
# i 14,600 more rows
```

```
# i 10 more variables: income <dbl+lbl>, polviews <dbl+lbl>, fefam <dbl+lbl>,  
# vpsu <dbl+lbl>, vstrat <dbl+lbl>, oversamp <dbl>, formwt <dbl>,  
# wtssall <dbl+lbl>, sampcode <dbl+lbl>, sample <dbl+lbl>
```

# The GSS Panel: Recoding

```
gss_sub ►  
  mutate(across(everything(), zap_missing)) ►  
  mutate(across(all_of(wt_vars), as.numeric)) ►  
  mutate(across(all_of(int_vars), as.integer)) ►  
  mutate(across(all_of(cat_vars), as_factor)) ►  
  mutate(across(all_of(cat_vars), fct_relabel, tolower)) ►  
  mutate(across(all_of(cat_vars), fct_relabel, tools::toTitleCase)) ►  
  mutate(income = stringr::str_replace(income, " - ", "-"))
```

```
# A tibble: 14,610 × 19  
   year   id ballot  age tvhours race  sex  degree  relig income polviews  
   <int> <int> <int> <int>   <int> <fct> <fct>   <fct>   <fct> <chr>   <fct>  
1  2006     9     3    23     NA Black Female Bachelor None $2500... Conserv...  
2  2008   3001     3    25     NA Other Female Bachelor None $2500... Extreme...  
3  2010   6001     3    27     NA Black Female Bachelor None $2500... Extreme...  
4  2010   6002     1    36      3 White Female Graduate None $2500... Liberal  
5  2006    10     1    32      3 Other Female Graduate None <NA> Slightl...  
6  2008   3002     1    34      3 Other Female Graduate None $2500... Moderate  
7  2008   3003     3    83     NA Black Female Lt High ... Prot... $2000... Liberal  
8  2010   6003     3    85     NA Black Female Lt High ... Prot... <NA> Moderate  
9  2006    11     3    81     NA Black Female Lt High ... Prot... <NA> Moderate  
10 2010   6004     1    51    10 Other Male  High Sch... Cath... Lt $1... Liberal  
# i 14,600 more rows  
# i 8 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,  
#   formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>
```

# How we'd actually write this

```
gss_sub ← gss_sub ►  
  mutate(across(everything(), zap_missing),  
    across(all_of(wt_vars), as.numeric),  
    across(all_of(int_vars), as.integer),  
    across(all_of(cat_vars), as_factor),  
    across(all_of(cat_vars), fct_relabel, tolower),  
    across(all_of(cat_vars), fct_relabel, tools::toTitleCase),  
    income = stringr::str_replace(income, " - ", "-"))
```

# The GSS panel: more recoding

Age quintiles: find the cutpoints

```
# seq can make all kinds of sequences  
seq(from = 0, to = 1, by = 0.2)
```

```
[1] 0.0 0.2 0.4 0.6 0.8 1.0
```

```
age_quintiles ← quantile(as.numeric(gss_panel$age),  
                          probs = seq(0, 1, 0.2),  
                          na.rm = TRUE)
```

```
## These are the quintile cutpoints  
age_quintiles
```

0%	20%	40%	60%	80%	100%
18	33	43	53	65	89

# The GSS panel: more recoding

Age quintiles: create the quintile variable

```
## Apply the cut
gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintiles),
                        include.lowest = TRUE)) >
  pull(agequint) > # grab a column and make it an ordinary vector
table()
```

[18,33]	(33,43]	(43,53]	(53,65]	(65,89]
3157	2680	2851	3057	2720

We'll need to clean up those labels.

# The GSS panel: more recoding

I told you that regexp stuff would pay off.

```
convert_agegrp ← function(x){  
  x ← stringr::str_remove(x, "\\(") # Remove open paren  
  x ← stringr::str_remove(x, "\\[") # Remove open bracket  
  x ← stringr::str_remove(x, "\\]") # Remove close bracket  
  x ← stringr::str_replace(x, ",", "-") # Replace comma with dash  
  x ← stringr::str_replace(x, "-89", "+") # Replace -89 with +  
  regex ← "^(*$)" # Matches everything in string to end of line  
  x ← stringr::str_replace(x, regex, "Age \\1") # Preface string with "Age"  
  x  
}
```



# The GSS panel: more recoding

```
gss_sub
```

```
# A tibble: 14,610 × 19
  year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int>   <int> <fct> <fct> <fct>   <fct> <chr> <fct>
1  2006     9     3   23     NA Black Female Bachelor None $2500... Conserv...
2  2008   3001     3   25     NA Other Female Bachelor None $2500... Extreme...
3  2010   6001     3   27     NA Black Female Bachelor None $2500... Extreme...
4  2010   6002     1   36      3 White Female Graduate None $2500... Liberal
5  2006    10     1   32      3 Other Female Graduate None <NA> Slightl...
6  2008   3002     1   34      3 Other Female Graduate None $2500... Moderate
7  2008   3003     3   83     NA Black Female Lt High ... Prot... $2000... Liberal
8  2010   6003     3   85     NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3   81     NA Black Female Lt High ... Prot... <NA> Moderate
10 2010   6004     1   51     10 Other Male High Sch... Cath... Lt $1... Liberal

# i 14,600 more rows
# i 8 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>
```

# The GSS panel: more recoding

```
gss_sub >
  mutate(agequint = cut(x = age,
    breaks = unique(age_quintil),
    include.lowest = TRUE))
```

```
# A tibble: 14,610 × 20
  year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int> <int> <fct> <fct> <fct> <fct> <chr> <fct>
1  2006     9     3   23    NA Black Female Bachelor None $2500... Conserv...
2  2008   3001     3   25    NA Other Female Bachelor None $2500... Extreme...
3  2010   6001     3   27    NA Black Female Bachelor None $2500... Extreme...
4  2010   6002     1   36     3 White Female Graduate None $2500... Liberal
5  2006    10     1   32     3 Other Female Graduate None <NA> Slightl...
6  2008   3002     1   34     3 Other Female Graduate None $2500... Moderate
7  2008   3003     3   83    NA Black Female Lt High ... Prot... $2000... Liberal
8  2010   6003     3   85    NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3   81    NA Black Female Lt High ... Prot... <NA> Moderate
10 2010   6004     1   51    10 Other Male High Sch... Cath... Lt $1... Liberal

# i 14,600 more rows
# i 9 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>, agequint <fct>
```

# The GSS panel: more recoding

```
gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintil),
                        include.lowest = TRUE)) >
  mutate(agequint = fct_relabel(agequint, convert_a
```

```
# A tibble: 14,610 × 20
  year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int> <int> <fct> <fct> <fct> <fct> <chr> <fct>
1  2006     9     3   23    NA Black Female Bachelor None $2500... Conserv...
2  2008   3001     3   25    NA Other Female Bachelor None $2500... Extreme...
3  2010   6001     3   27    NA Black Female Bachelor None $2500... Extreme...
4  2010   6002     1   36     3 White Female Graduate None $2500... Liberal
5  2006    10     1   32     3 Other Female Graduate None <NA> Slightl...
6  2008   3002     1   34     3 Other Female Graduate None $2500... Moderate
7  2008   3003     3   83    NA Black Female Lt High ... Prot... $2000... Liberal
8  2010   6003     3   85    NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3   81    NA Black Female Lt High ... Prot... <NA> Moderate
10 2010   6004     1   51    10 Other Male High Sch... Cath... Lt $1... Liberal

# i 14,600 more rows
# i 9 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>, agequint <fct>
```

# The GSS panel: more recoding

```
gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintil),
                        include.lowest = TRUE)) >
  mutate(agequint = fct_relabel(agequint, convert_agequint))
  mutate(year_f = droplevels(factor(year)))
```

```
# A tibble: 14,610 × 21
  year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int> <int> <fct> <fct> <fct> <fct> <chr> <fct>
1  2006     9     3   23    NA Black Female Bachelor None $2500... Conserv...
2  2008   3001     3   25    NA Other Female Bachelor None $2500... Extreme...
3  2010   6001     3   27    NA Black Female Bachelor None $2500... Extreme...
4  2010   6002     1   36     3 White Female Graduate None $2500... Liberal
5  2006    10     1   32     3 Other Female Graduate None <NA> Slightl...
6  2008   3002     1   34     3 Other Female Graduate None $2500... Moderate
7  2008   3003     3   83    NA Black Female Lt High ... Prot... $2000... Liberal
8  2010   6003     3   85    NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3   81    NA Black Female Lt High ... Prot... <NA> Moderate
10 2010   6004     1   51    10 Other Male High Sch... Cath... Lt $1... Liberal

# i 14,600 more rows
# i 10 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>, agequint <fct>,
# year_f <fct>
```

# The GSS panel: more recoding

```
gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintil),
                        include.lowest = TRUE)) >
  mutate(agequint = fct_relabel(agequint, convert_agequint)) >
  mutate(year_f = droplevels(factor(year))) >
  mutate(young = ifelse(age < 26, "Yes", "No"))
```

```
# A tibble: 14,610 × 22
   year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int> <int> <fct> <fct> <fct> <fct> <chr> <fct>
1  2006     9     3    23     NA Black Female Bachelor None $2500... Conserv...
2  2008    3001     3    25     NA Other Female Bachelor None $2500... Extreme...
3  2010    6001     3    27     NA Black Female Bachelor None $2500... Extreme...
4  2010    6002     1    36      3 White Female Graduate None $2500... Liberal
5  2006    10     1    32      3 Other Female Graduate None <NA> Slightl...
6  2008    3002     1    34      3 Other Female Graduate None $2500... Moderate
7  2008    3003     3    83     NA Black Female Lt High ... Prot... $2000... Liberal
8  2010    6003     3    85     NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3    81     NA Black Female Lt High ... Prot... <NA> Moderate
10 2010    6004     1    51    10 Other Male High Sch... Cath... Lt $1... Liberal

# i 14,600 more rows
# i 11 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>, agequint <fct>,
# year_f <fct>, young <chr>
```

# The GSS panel: more recoding

```
gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintile),
                        include.lowest = TRUE)) >
  mutate(agequint = fct_relabel(agequint, convert_agequint)) >
  mutate(year_f = droplevels(factor(year))) >
  mutate(young = ifelse(age < 26, "Yes", "No")) >
  mutate(fefam_d = fct_recode(fefam,
                              Agree = "Strongly Agree",
                              Disagree = "Strongly Disagree"))
```

```
# A tibble: 14,610 × 23
  year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int> <int> <fct> <fct> <fct> <fct> <chr> <fct>
1  2006     9     3    23      NA Black Female Bachelor None $2500... Conserv...
2  2008    3001     3    25      NA Other Female Bachelor None $2500... Extreme...
3  2010    6001     3    27      NA Black Female Bachelor None $2500... Extreme...
4  2010    6002     1    36       3 White Female Graduate None $2500... Liberal
5  2006    10     1    32       3 Other Female Graduate None <NA> Slightl...
6  2008    3002     1    34       3 Other Female Graduate None $2500... Moderate
7  2008    3003     3    83      NA Black Female Lt High ... Prot... $2000... Liberal
8  2010    6003     3    85      NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3    81      NA Black Female Lt High ... Prot... <NA> Moderate
10 2010    6004     1    51     10 Other Male High Sch... Cath... Lt $1... Liberal

# i 14,600 more rows
# i 12 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>, agequint <fct>,
# year_f <fct>, young <chr>, fefam_d <fct>
```

# The GSS panel: more recoding

```
gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintile),
                        include.lowest = TRUE)) >
  mutate(agequint = fct_relabel(agequint, convert_agequint)) >
  mutate(year_f = droplevels(factor(year))) >
  mutate(young = ifelse(age < 26, "Yes", "No")) >
  mutate(fefam_d = fct_recode(fefam,
                              Agree = "Strongly Agree",
                              Disagree = "Strongly Disagree")) >
  mutate(degree = factor(degree,
                         levels = levels(gss_sub$degree),
                         ordered = TRUE))
```

```
# A tibble: 14,610 × 23
  year   id ballot age tvhours race sex degree relig income polviews
  <int> <int> <int> <int> <int> <fct> <fct> <ord> <fct> <chr> <fct>
1  2006     9     3   23     NA Black Female Bachelor None $2500... Conserv...
2  2008   3001     3   25     NA Other Female Bachelor None $2500... Extreme...
3  2010   6001     3   27     NA Black Female Bachelor None $2500... Extreme...
4  2010   6002     1   36      3 White Female Graduate None $2500... Liberal
5  2006    10     1   32      3 Other Female Graduate None <NA> Slightl...
6  2008   3002     1   34      3 Other Female Graduate None $2500... Moderate
7  2008   3003     3   83     NA Black Female Lt High ... Prot... $2000... Liberal
8  2010   6003     3   85     NA Black Female Lt High ... Prot... <NA> Moderate
9  2006    11     3   81     NA Black Female Lt High ... Prot... <NA> Moderate
10 2010   6004     1   51     10 Other Male High Sch... Cath... Lt $1... Liberal
# i 14,600 more rows
# i 12 more variables: fefam <fct>, vpsu <dbl>, vstrat <dbl>, oversamp <dbl>,
# formwt <dbl>, wtssall <dbl>, sampcode <dbl>, sample <dbl>, agequint <fct>,
# year_f <fct>, young <chr>, fefam_d <fct>
```

# How we'd actually write this

```
gss_sub <- gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintiles),
                        include.lowest = TRUE),
         agequint = fct_relabel(agequint, convert_agegrp),
         year_f = droplevels(factor(year)), #<<
         young = ifelse(age < 26, "Yes", "No"),
         fefam_d = fct_recode(fefam,
                              Agree = "Strongly Agree",
                              Disagree = "Strongly Disagree"),
         degree = factor(degree,
                          levels = levels(gss_sub$degree),
                          ordered = TRUE))
```



# How we'd actually write this

```
gss_sub <- gss_sub >
  mutate(agequint = cut(x = age,
                        breaks = unique(age_quintiles),
                        include.lowest = TRUE),
         agequint = fct_relabel(agequint, convert_agegrp),
         year_f = factor(year),
         young = ifelse(age < 26, "Yes", "No"),
         fefam_d = fct_recode(fefam,
                              Agree = "Strongly Agree",
                              Disagree = "Strongly Disagree"),
         degree = factor(degree,
                          levels = levels(gss_sub$degree),
                          ordered = TRUE))
```

# How we'd actually write this

```
gss_sub <- gss_sub >
mutate(agequint = cut(x = age,
                      breaks = unique(age_quintiles),
                      include.lowest = TRUE),
       agequint = fct_relabel(agequint, convert_agegrp),
       year_f = droplevels(factor(year)),
       young = ifelse(age < 26, "Yes", "No"),
       fefam_d = fct_recode(fefam,
                           Agree = "Strongly Agree",
                           Disagree = "Strongly Disagree"),
       degree = factor(degree,
                       levels = levels(gss_sub$degree),
                       ordered = TRUE))
```

# GSS Panel

```
gss_sub ►  
  select(sex, year, year_f, age, young, fefam, fefam_d) ►  
  sample_n(15)
```

```
# A tibble: 15 × 7
```

	sex	year	year_f	age	young	fefam	fefam_d
	<fct>	<int>	<fct>	<int>	<chr>	<fct>	<fct>
1	Female	2008	2008	63	No	<NA>	<NA>
2	Male	2014	2014	35	No	Disagree	Disagree
3	Female	2010	2010	49	No	Agree	Agree
4	Female	2006	2006	66	No	<NA>	<NA>
5	Female	2014	2014	41	No	Strongly Disagree	Disagree
6	Female	2012	2012	41	No	Agree	Agree
7	Male	2006	2006	84	No	<NA>	<NA>
8	Female	2014	2014	50	No	<NA>	<NA>
9	Male	2006	2006	64	No	Disagree	Disagree
10	Male	2012	2012	61	No	Disagree	Disagree
11	Female	2008	2008	43	No	Disagree	Disagree
12	Female	2010	2010	40	No	Strongly Agree	Agree
13	Female	2010	2010	69	No	<NA>	<NA>
14	Male	2008	2008	25	Yes	Strongly Disagree	Disagree
15	Male	2008	2008	47	No	Disagree	Disagree

# GSS Panel

```
gss_sub ▶  
  select(sex, degree) ▶  
  group_by(sex, degree) ▶  
  tally() ▶  
  pivot_wider(names_from = sex, values_from = n)
```

```
# A tibble: 6 × 3  
  degree      Male Female  
  <ord>    <int> <int>  
1 Lt High School    814   1036  
2 High School    3131   4143  
3 Junior College   440    721  
4 Bachelor    1293   1474  
5 Graduate       696    860  
6 <NA>          NA      2
```



**More about factors**

# More on factors

We've already seen `fct_relabel()` and `fct_recode()` from `forcats`.

There are numerous other convenience functions for factors.

# More on factors

```
gss_sub ▶  
  count(degree)
```

```
# A tibble: 6 × 2  
  degree      n  
  <ord>    <int>  
1 Lt High School 1850  
2 High School    7274  
3 Junior College 1161  
4 Bachelor       2767  
5 Graduate       1556  
6 <NA>           2
```

```
levels(gss_sub$degree)
```

```
[1] "Lt High School" "High School"    "Junior College" "Bachelor"  
[5] "Graduate"
```



# More on factors

Make the **NA** values an explicit level

```
gss_sub >
  mutate(degree_na = fct_explicit_na(degree)) >
  count(degree_na)
```

```
# A tibble: 6 × 2
  degree_na      n
  <ord>        <int>
1 Lt High School 1850
2 High School    7274
3 Junior College 1161
4 Bachelor       2767
5 Graduate       1556
6 (Missing)       2
```

# More on factors

## Relevel by frequency

```
gss_sub >
  mutate(degree_freq = fct_infreq(degree)) >
  count(degree_freq)
```

```
# A tibble: 6 × 2
  degree_freq      n
  <ord>         <int>
1 High School    7274
2 Bachelor       2767
3 Lt High School 1850
4 Graduate       1556
5 Junior College 1161
6 <NA>           2
```

# More on factors

Relevel manually

```
is.ordered(gss_sub$sex)
```

```
[1] FALSE
```

```
levels(gss_sub$sex)
```

```
[1] "Male" "Female"
```

# More on factors

## Relevel manually

```
summary(lm(age ~ sex, data = gss_sub))
```

Call:

```
lm(formula = age ~ sex, data = gss_sub)
```

Residuals:

Min	1Q	Median	3Q	Max
-31.431	-13.972	-0.431	12.569	40.028

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	48.9720	0.2149	227.846	<2e-16 **
sexFemale	0.4594	0.2864	1.604	0.109

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 17.08 on 14463 degrees of freedom  
(145 observations deleted due to missingness)

Multiple R-squared: 0.0001779, Adjusted R-squared: 0.0001088

F-statistic: 2.573 on 1 and 14463 DF, p-value: 0.1087

# More on factors

Relevel manually

```
gss_sub ← gss_sub ►  
  mutate(sex = fct_relevel(sex, "Female"))  
  
levels(gss_sub$sex)
```

```
[1] "Female" "Male"
```

# More on factors

## Relevel manually

```
summary(lm(age ~ sex, data = gss_sub))
```

Call:

```
lm(formula = age ~ sex, data = gss_sub)
```

Residuals:

Min	1Q	Median	3Q	Max
-31.431	-13.972	-0.431	12.569	40.028

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	49.4313	0.1892	261.233	<2e-16 **
sexMale	-0.4594	0.2864	-1.604	0.109

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 17.08 on 14463 degrees of freedom

(145 observations deleted due to missingness)

Multiple R-squared: 0.0001779, Adjusted R-squared: 0.0001088

F-statistic: 2.573 on 1 and 14463 DF, p-value: 0.1087

# More on factors

## Interact or cross factors

```
gss_sub ← gss_sub ►  
  mutate(degree_by_race = fct_cross(race, degree))  
  
gss_sub ►  
  count(degree_by_race)
```

```
# A tibble: 16 × 2  
  degree_by_race      n  
  <fct>          <int>  
1 White:Lt High School 1188  
2 Black:Lt High School  379  
3 Other:Lt High School  283  
4 White:High School    5548  
5 Black:High School    1180  
6 Other:High School     546  
7 White:Junior College  885  
8 Black:Junior College  206  
9 Other:Junior College   70  
10 White:Bachelor      2334  
11 Black:Bachelor       233  
12 Other:Bachelor       200  
13 White:Graduate      1293  
14 Black:Graduate       116  
15 Other:Graduate       147  
16 NA                  0
```

# More on factors

Relevel manually by lumping ... the least frequent n

```
gss_sub ►  
  mutate(degree_n = fct_lump_n(degree, n = 3)) ►  
  count(degree_n)
```

```
# A tibble: 5 × 2  
  degree_n      n  
  <ord>      <int>  
1 Lt High School 1850  
2 High School   7274  
3 Bachelor      2767  
4 Other         2717  
5 <NA>           2
```



# More on factors

Relevel manually by lumping ...to other, manually

```
gss_sub ►  
  mutate(degree_o = fct_other(degree,  
                              keep = c("Lt High School",  
                                       "High School"))) ►  
  
  count(degree_o)
```

```
# A tibble: 4 × 2  
  degree_o      n  
  <ord>      <int>  
1 Lt High School 1850  
2 High School   7274  
3 Other         5484  
4 <NA>           2
```