

# Import Data with



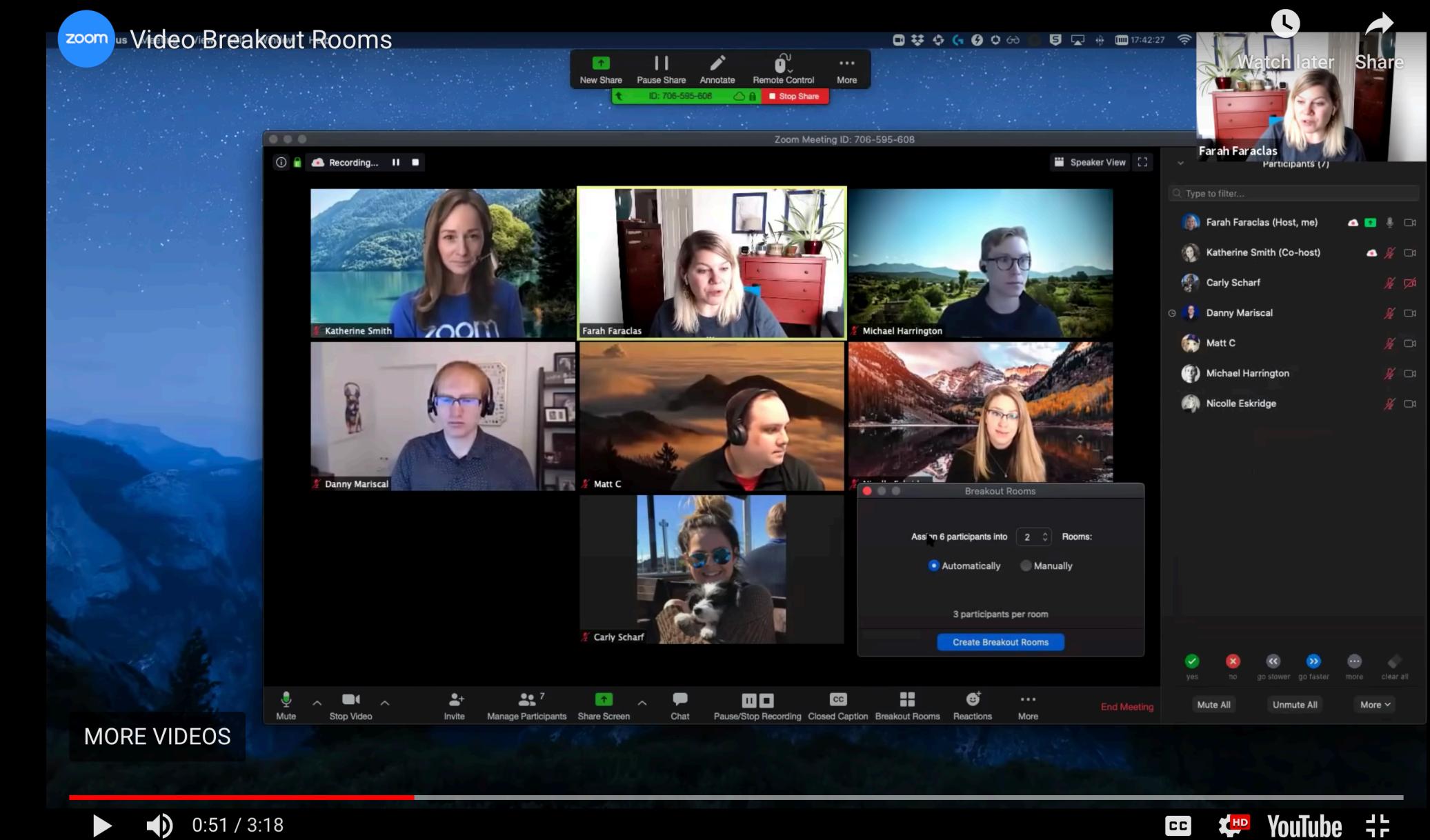
# Exercise: What do you already know?

- I will use Zoom to split you into groups of two.

- Ask your partner:

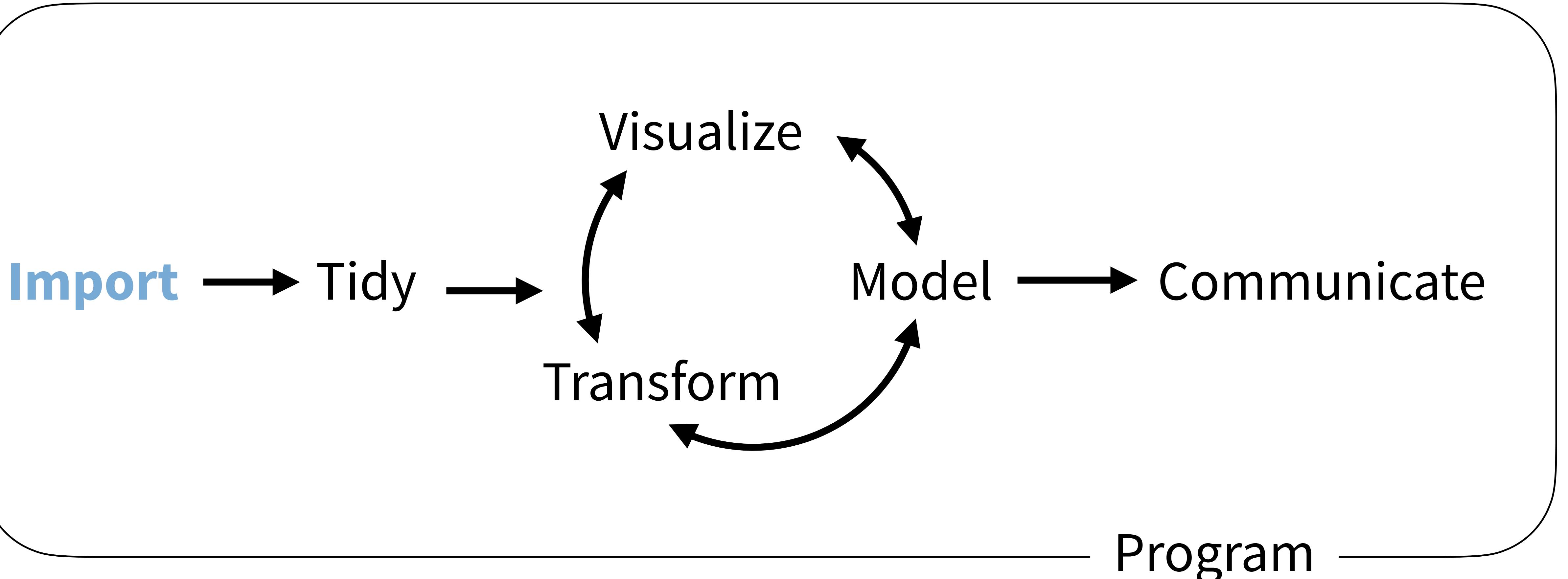
- *What sort of data do you need to get into R?*
- *Where does it currently reside?*
- *What problems have you run into importing data into R before?*

- 5 minutes





# (Applied) Data Science



# Data Import in the Tidyverse

## A package for each storage type!

package	accesses
readr	csv, tsv, etc.
haven	SPSS, Stata, and SAS files
readxl	excel files (.xls, .xlsx)
jsonlite	json
xml2	xml
httr	web API's
rvest	web pages (web scraping)
DBI	databases
sparklyr	data loaded into spark



**Open exercises/03-Import-Data-  
Exercises.Rmd**

# Importing Data

R

# readr



Simple, consistent functions for working  
with strings / csv data.

```
# install.packages("tidyverse")
library(readr)
```



Compared to **read.table** and its derivatives,  
readr functions are:

1. ~ 10 times faster
2. Return tibbles
3. Have more intuitive defaults. No row  
names, no strings as factors.



# readr functions

function	reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files
read_fwf()	Fixed width files
read_log()	Apache log files
read_table()	Space separated
read_tsv()	Tab delimited values



# readr functions

function	reads
<b>read_csv()</b>	<b>Comma separated values</b>
read_csv2()	Semi-colon separated values
read_delim()	General delimited files
read_fwf()	Fixed width files
read_log()	Apache log files
read_table()	Space separated
read_tsv()	Tab delimited values



# nimbus.csv

```
date,longitude,latitude,ozone
1985-10-01T00:00:00Z,-179.375,-87.5,.
1985-10-01T00:00:00Z,-178.125,-87.5,.
1985-10-01T00:00:00Z,-176.875,-87.5,.
1985-10-01T00:00:00Z,-175.625,-87.5,.
1985-10-01T00:00:00Z,-174.375,-87.5,.
1985-10-01T00:00:00Z,-173.125,-87.5,.
1985-10-01T00:00:00Z,-171.875,-87.5,.
1985-10-01T00:00:00Z,-170.625,-87.5,.
1985-10-01T00:00:00Z,-169.375,-87.5,.
```



# nimbus.csv

```
date,longitude,latitude,ozone  
1985-10-01T00:00:00Z,-179.375,-87.5,.  
1985-10-01T00:00:00Z,-178.125,-87.5,.  
1985-10-01T00:00:00Z,-176.875,-87.5,.  
1985-10-01T00:00:00Z,-175.625,-87.5,.  
1985-10-01T00:00:00Z,-174.375,-87.5,.  
1985-10-01T00:00:00Z,-173.125,-87.5,.  
1985-10-01T00:00:00Z,-171.875,-87.5,.  
1985-10-01T00:00:00Z,-170.625,-87.5,.  
1985-10-01T00:00:00Z,-169.375,-87.5,.
```





# read\_csv()

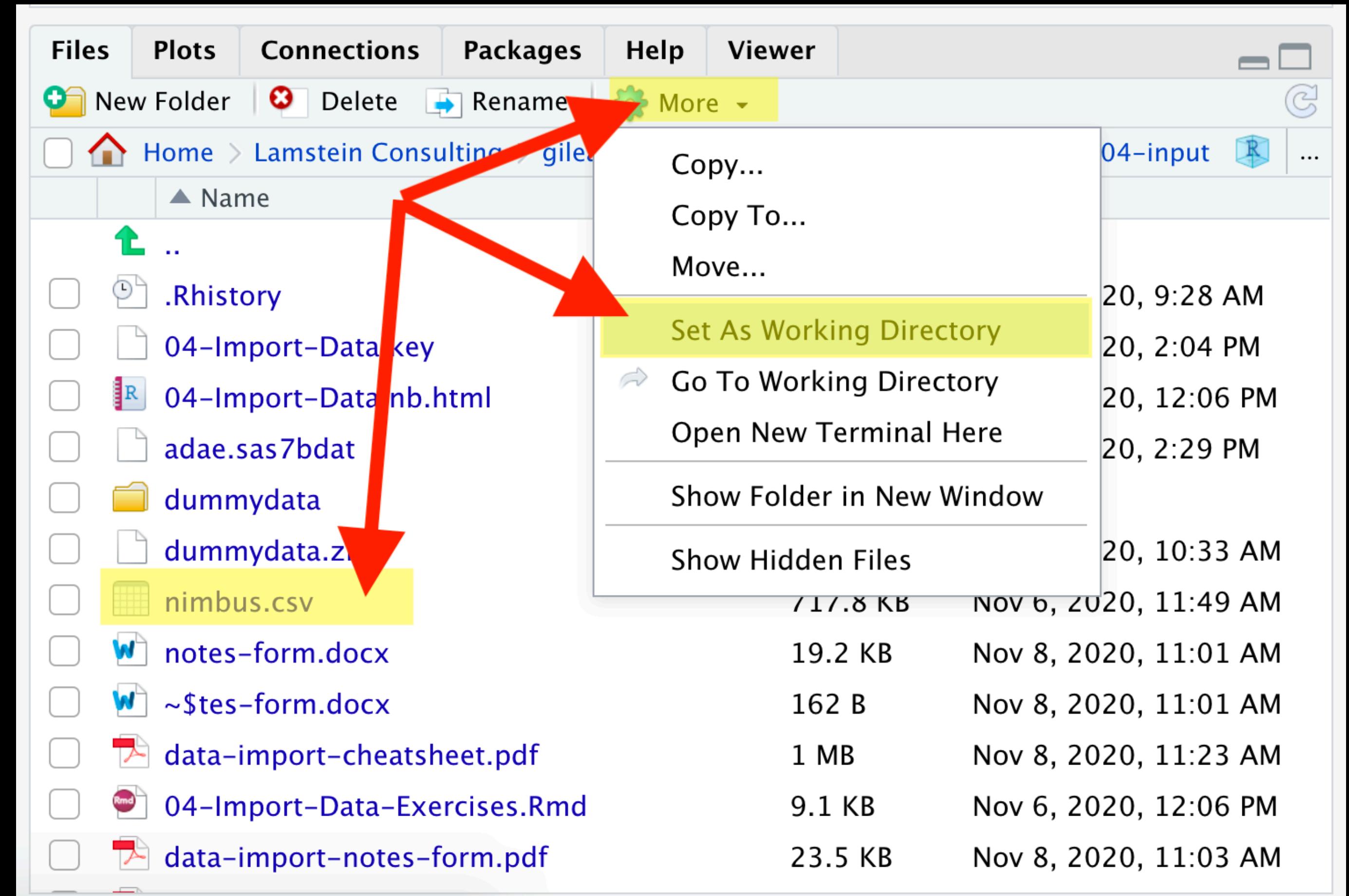
readr functions share a common syntax

```
df <- read_csv("path/to/file.csv", ...)
```

object to save  
output into

path from working  
directory to file





# Set Your Working Directory

# Your Turn 1

Find **nimbus.csv** (in your working directory). Then read it into an object. Then view the results.



# Your Turn 1

Find **nimbus.csv** (in your working directory). Then read it into an object. Then view the results.

```
nimbus <- read_csv("nimbus.csv")
```

```
nimbus
```

# tibbles

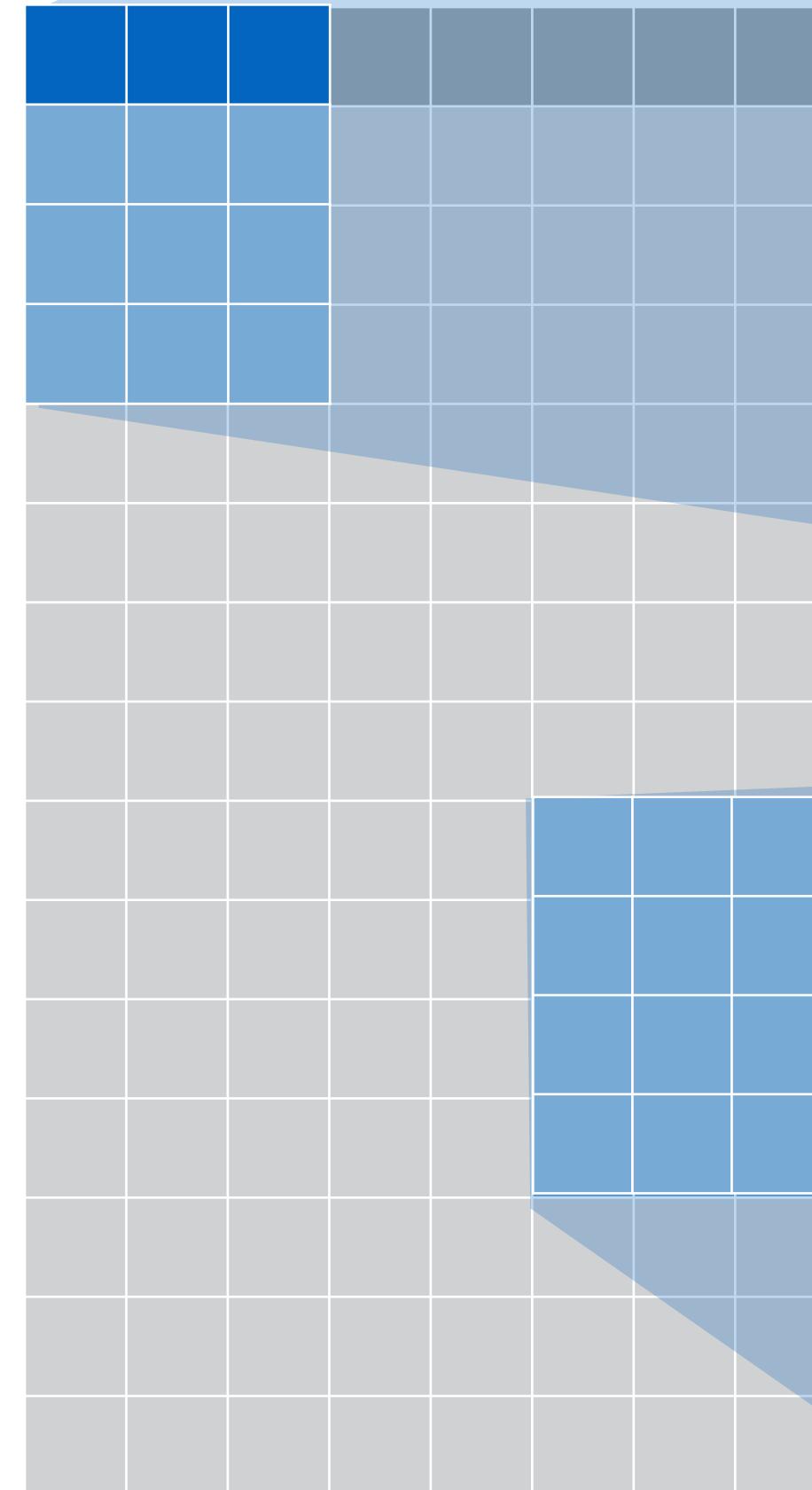
R

# read.csv() vs. read\_csv()

```
Console ~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse/0-course-development> 
217 1985-10-01 -144.375 -86.5 .
218 1985-10-01 -143.125 -86.5 .
219 1985-10-01 -141.875 -86.5 .
220 1985-10-01 -140.625 -86.5 .
221 1985-10-01 -139.375 -86.5 .
222 1985-10-01 -138.125 -86.5 .
223 1985-10-01 -136.875 -86.5 .
224 1985-10-01 -135.625 -86.5 .
225 1985-10-01 -134.375 -86.5 .
226 1985-10-01 -133.125 -86.5 .
227 1985-10-01 -131.875 -86.5 .
228 1985-10-01 -130.625 -86.5 .
229 1985-10-01 -129.375 -86.5 .
230 1985-10-01 -128.125 -86.5 .
231 1985-10-01 -126.875 -86.5 .
232 1985-10-01 -125.625 -86.5 .
233 1985-10-01 -124.375 -86.5 .
234 1985-10-01 -123.125 -86.5 .
235 1985-10-01 -121.875 -86.5 .
236 1985-10-01 -120.625 -86.5 .
237 1985-10-01 -119.375 -86.5 .
238 1985-10-01 -118.125 -86.5 .
239 1985-10-01 -116.875 -86.5 .
240 1985-10-01 -115.625 -86.5 .
241 1985-10-01 -114.375 -86.5 .
242 1985-10-01 -113.125 -86.5 .
243 1985-10-01 -111.875 -86.5 .
244 1985-10-01 -110.625 -86.5 .
245 1985-10-01 -109.375 -86.5 .
246 1985-10-01 -108.125 -86.5 .
247 1985-10-01 -106.875 -86.5 .
248 1985-10-01 -105.625 -86.5 .
249 1985-10-01 -104.375 -86.5 .
250 1985-10-01 -103.125 -86.5 .
[ reached getOption("max.print") -- omitted 24974 rows ]
>
```

```
Console ~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse/0-course-development> 
> nimbus
# A tibble: 25,224 x 4
  date      longitude   latitude ozone
  <dttm>        <dbl>     <dbl> <chr>
1 1985-10-01 -179.375  -87.5  .
2 1985-10-01 -178.125  -87.5  .
3 1985-10-01 -176.875  -87.5  .
4 1985-10-01 -175.625  -87.5  .
5 1985-10-01 -174.375  -87.5  .
6 1985-10-01 -173.125  -87.5  .
7 1985-10-01 -171.875  -87.5  .
8 1985-10-01 -170.625  -87.5  .
9 1985-10-01 -169.375  -87.5  .
10 1985-10-01 -168.125  -87.5  .
# ... with 25,214 more rows
> |
```





A large table to  
display

```
# A tibble: 234 × 6
  manufacturer     model   displ
  <chr>       <chr>   <dbl>
1 audi         a4      1.8
2 audi         a4      1.8
3 audi         a4      2.0
4 audi         a4      2.0
5 audi         a4      2.8
6 audi         a4      2.8
7 audi         a4      3.1
8 audi a4 quattro 1.8
9 audi a4 quattro 1.8
10 audi a4 quattro 2.0
# ... with 224 more rows, and 3
# more variables: year <int>,
# cyl <int>, trans <chr>
```

## tibble display

```
156 1999 6 auto(l4)
157 1999 6 auto(l4)
158 2008 6 auto(l4)
159 2008 8 auto(s4)
160 1999 4 manual(m5)
161 1999 4 auto(l4)
162 2008 4 manual(m5)
163 2008 4 manual(m5)
164 2008 4 auto(l4)
165 2008 4 auto(l4)
166 1999 4 auto(l4)
[ reached getOption("max.print") --
  omitted 68 rows ]
```

## data frame display



# tibbles

A type of data frame common throughout tidyverse packages.  
Tibbles enhance data frames in three ways:

- 1. Subsetting** - [ always returns a new tibble, [[ and \$ always return a new vector
- 2. No partial matching** - You must use full column names when subsetting
- 3. Display** - When you print a tibble, R provides a concise view of the data that fits on one screen





# tibble

A package with several helper functions for tibbles:

- **as\_tibble()** - convert a data frame to a tibble
- **as.data.frame()** - convert a tibble to a data frame
- **tribble()** - make a tibble (transversed)

```
tribble(  
  ~x, ~y,  
  1, "a",  
  2, "b",  
  3, "c")
```

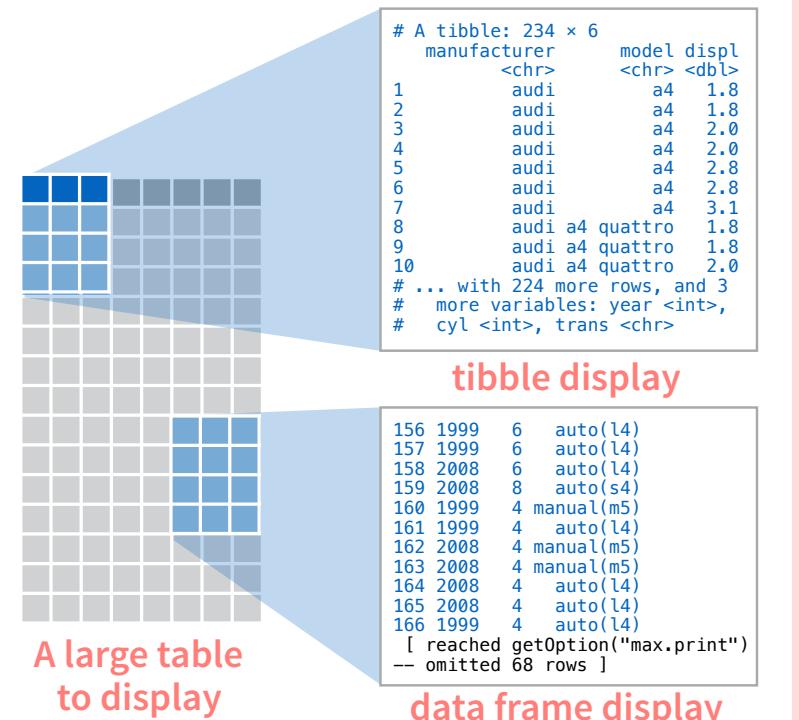
x	y
1	a
2	b
3	c



## Tibbles - an enhanced data frame

The **tibble** package provides a new S3 class for storing tabular data, the tibble. Tibbles inherit the data frame class, but improve two behaviors:

- **Display** - When you print a tibble, R provides a concise view of the data that fits on one screen.
- **Subsetting** - [ always returns a new tibble, [[ and \$ always return a vector.
- **No partial matching** - You must use full column names when subsetting



- Control the default appearance with options:  
`options(tibble.print_max = n, tibble.print_min = m, tibble.width = Inf)`
- View entire data set with `View(x, title)` or `glimpse(x, width = NULL, ...)`
- Revert to data frame with `as.data.frame()` (required for some older packages)

### Construct a tibble in two ways

`tibble(...)`

Construct by columns.

```
tibble(x = 1:3,
      y = c("a", "b", "c"))
```

Both make this tibble

`tibble(...)`

Construct by rows.

`tibble(`

```
~x, ~y,
1, "a",
2, "b",
3, "c")
```

A tibble: 3 x 2

x y

1	a
2	b
3	c

`as_tibble(x, ...)` Convert data frame to tibble.

`enframe(x, name = "name", value = "value")`

Converts named vector to a tibble with a names column and a values column.

`is_tibble(x)` Test whether x is a tibble.

# tibbles

## Tibbles - an enhanced data frame

The **tibble** package provides a new S3 class for storing tabular data, the tibble. Tibbles inherit the data frame class, but improve two behaviors:

- **Display** - When you print a tibble, R provides a concise view of the data that fits on one screen.
- **Subsetting** - [ always returns a new tibble, [[ and \$ always return a vector.
- **No partial matching** - You must use full column names when subsetting

A large table to display

tibble display

```
# A tibble: 234 x 6
  manufacturer model displ  year   cyl  trans
  <chr>        <chr> <dbl> <dbl> <dbl>
  1 audi         a4      1.8   1999  4    auto
  2 audi         a4      1.8   1999  4    auto
  3 audi         a4      2.0   1999  4    auto
  4 audi         a4      2.0   1999  4    auto
  5 audi         a4      2.0   1999  4    auto
  6 audi         a4      2.0   1999  4    auto
  7 audi         a4      2.0   1999  4    auto
  8 audi         a4      2.0   1999  4    auto
  9 audi         a4      2.0   1999  4    auto
  10 audi        a4      2.0   1999  4    auto
  # ... with 224 more rows, and 3
  #   more variables: year <int>,
  #   cyl <int>, trans <chr>
```

A large table to display

data frame display

```
156 1999 6 auto(14)
157 1999 6 auto(14)
158 2000 6 auto(14)
159 2008 8 auto(s4)
160 1999 4 manual(m5)
161 1999 4 auto(l4)
162 2008 4 manual(m5)
163 2008 4 manual(m5)
164 2008 4 auto(l4)
165 2008 4 auto(l4)
166 1999 4 auto(l4)
167 1999 4 auto(l4)
168 2008 4 manual(m5)
169 2008 4 manual(m5)
170 2008 4 manual(m5)
171 2008 4 manual(m5)
172 2008 4 manual(m5)
173 2008 4 manual(m5)
174 2008 4 manual(m5)
175 2008 4 manual(m5)
176 2008 4 manual(m5)
177 2008 4 manual(m5)
178 2008 4 manual(m5)
179 2008 4 manual(m5)
180 2008 4 manual(m5)
181 2008 4 manual(m5)
182 2008 4 manual(m5)
183 2008 4 manual(m5)
184 2008 4 manual(m5)
185 2008 4 manual(m5)
186 2008 4 manual(m5)
187 2008 4 manual(m5)
188 2008 4 manual(m5)
189 2008 4 manual(m5)
190 2008 4 manual(m5)
191 2008 4 manual(m5)
192 2008 4 manual(m5)
193 2008 4 manual(m5)
194 2008 4 manual(m5)
195 2008 4 manual(m5)
196 2008 4 manual(m5)
197 2008 4 manual(m5)
198 2008 4 manual(m5)
199 2008 4 manual(m5)
200 2008 4 manual(m5)
201 2008 4 manual(m5)
202 2008 4 manual(m5)
203 2008 4 manual(m5)
204 2008 4 manual(m5)
205 2008 4 manual(m5)
206 2008 4 manual(m5)
207 2008 4 manual(m5)
208 2008 4 manual(m5)
209 2008 4 manual(m5)
210 2008 4 manual(m5)
211 2008 4 manual(m5)
212 2008 4 manual(m5)
213 2008 4 manual(m5)
214 2008 4 manual(m5)
215 2008 4 manual(m5)
216 2008 4 manual(m5)
217 2008 4 manual(m5)
218 2008 4 manual(m5)
219 2008 4 manual(m5)
220 2008 4 manual(m5)
221 2008 4 manual(m5)
222 2008 4 manual(m5)
223 2008 4 manual(m5)
224 2008 4 manual(m5)
225 2008 4 manual(m5)
226 2008 4 manual(m5)
227 2008 4 manual(m5)
228 2008 4 manual(m5)
229 2008 4 manual(m5)
230 2008 4 manual(m5)
231 2008 4 manual(m5)
232 2008 4 manual(m5)
233 2008 4 manual(m5)
234 2008 4 manual(m5)
```

I reached getOption("max.print")  
-- omitted 68 rows ]

## Tidy Data with tidyverse

Tidy data is a way to organize tabular data. It provides a consistent data structure across packages. A table is tidy if:



### Reshape Data - change the layout of values in a table

Use `gather()` and `spread()` to reorganize the values of a table into a new layout. Each uses the idea of a key column: value column pair.

`gather(data, key, value, ..., na.rm = FALSE, convert = FALSE, factor_key = FALSE)`

`gather moves column names into a key column, gathering the column values into a single value column.`

`table4a`

`country year cases`

`A 1999 0.7K`

`B 37K`

`C 212K`

`D 213K`

`key value`

`gather(table4a, `1999`, `2000`, key = "year", value = "cases")`

`spread(table2, type, count)`

### Handle Missing Values

`drop_na(data, ...)` Drop rows containing NA's in ... columns.

`fill(data, ..., direction = c("down", "up"))` Fill in NA's in ... columns with most recent non-NA values.

`replace_na(data, replace = list(), ...)` Replace NA's by column.

`drop_na(x, x2)`

`fill(x, x2)`

`replace_na(x, list(x2 = 2), x2)`

### Expand Tables - quickly create tables with combinations of values

`complete(data, ..., fill = list())` Adds to the data missing combinations of the values of the variables listed in ...

`complete(mtcars, cyl, gear, carb)`

Create new tibble with all possible combinations of the values of the variables listed in ...

`expand(mtcars, cyl, gear, carb)`

Learn more at [browseVignettes\(package = c\("readr", "tibble", "tidyverse"\)\)](#) • readr 1.1.0 • tibble 1.2.12 • tidyverse 0.6.0 • Updated: 2016-12



# Parsing



# Quiz

What class is ozone?

```
nimbus %>% pluck("ozone") %>% class()
```

```
nimbus %>% pluck("ozone") %>% class()
```

```
[1] "character"
```



```
nimbus %>% pluck("ozone") %>% unique()
```

```
[1] "302" "304" "287" "274" "264" "242" "211" "195" "197" "196" "198" "193" "187"  
[14] "190" "199" "194" "213" "218" "221" "229" "209" "186" "188" "191" "189" "184"  
[27] "180" "190" "215" "312" "319" "320" "311" "300" "290" "267" "226" "210" "200"  
[40] "203" "201" "192" "204" "206" "208" "205" "223" "232" "238" "243" "220" "202"  
[53] "185" "219" "222" "216" "324" "336" "333" "323" "308" "295" "244" "212" "237"  
[66] "248" "239" "241" "250" "249" "252" "234" "318" "313" "326" "335" "337" "316"  
[79] "266" "207" "227" "251" "253" "257" "261" "214" "228" "273" "285" "288" "291"  
[92] "270" "254" "317" "325" "332" "340" "344" "338" "297" "247" "217" "225" "231"  
[105] "235" "236" "262" "260" "265" "272" "278" "280" "279" "255" "245" "224" "181"  
[118] "240" "269" "296" "307" "315" "321" "306" "299" "298" "283" "327" "322" "328"  
[131] "331" "310" "275" "233" "258" "276" "281" "289" "330" "346" "305" "334" "359"  
[144] "347" "314" "301" "256" "263" "277" "284" "282" "271" "246" "183" "182" "230"  
[157] "349" "351" "350" "342" "329" "355" "371" "309" "303" "292" "259" "268" "341"  
[170] "343" "348" "345" "354" "361" "372" "382" "376" "356" "293" "286" "353" "351"  
[183] "358" "360" "363" "370" "384" "380" "294" "339" "362" "352" "368" "373" "377
```



. = NA

nimbus

<b>date</b> <code>&lt;S3: POSIXct&gt;</code>	<b>longitude</b> <code>&lt;dbl&gt;</code>	<b>latitude</b> <code>&lt;dbl&gt;</code>	<b>ozone</b> <code>&lt;chr&gt;</code>
1985-10-01	-179.375	-87.5	.
1985-10-01	-178.125	-87.5	.
1985-10-01	-176.875	-87.5	.
1985-10-01	-175.625	-87.5	.
1985-10-01	-174.375	-87.5	.
1985-10-01	-173.125	-87.5	.
1985-10-01	-171.875	-87.5	.
1985-10-01	-170.625	-87.5	.
1985-10-01	169.375	87.5	.



# read\_csv()

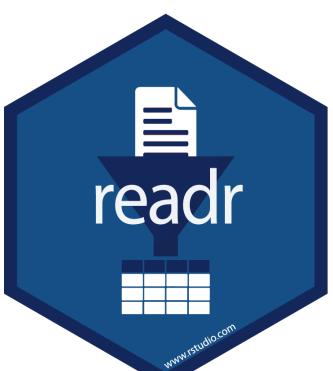
readr functions share a common syntax

```
nimbus <- read_csv("nimbus.csv", na = ".")
```

object to save  
output into

path from working  
directory to file

Value(s) to  
convert to NA



```
nimbus <- read_csv("nimbus.csv", na = ".")
```

date	longitude	latitude	ozone
<dtm>	<dbl>	<dbl>	<dbl>
1985-10-01 00:00:00	-179.	-73.5	302
1985-10-01 00:00:00	-178.	-73.5	302
1985-10-01 00:00:00	-177.	-73.5	302
1985-10-01 00:00:00	-176.	-73.5	302
1985-10-01 00:00:00	-174.	-73.5	302
1985-10-01 00:00:00	-173.	-73.5	302
1985-10-01 00:00:00	-172.	-73.5	304
1985-10-01 00:00:00	-171.	-73.5	304
1985-10-01 00:00:00	-164.	-73.5	287
1985-10-01 00:00:00	-163.	-73.5	287
... with 18,953 more rows			

<dbl> stands  
for "double"



Suppose

```
nimbus <- read_csv("nimbus.csv", na = ".")
```

<b>date</b> <small>&lt;S3: POSIXct&gt;</small>	<b>longitude</b> <small>&lt;dbl&gt;</small>	<b>latitude</b> <small>&lt;dbl&gt;</small>	<b>ozone</b> <small>&lt;chr&gt;</small>
1985-10-01	-179.375	-87.5	NA
1985-10-01	-178.125	-87.5	NA
1985-10-01	-176.875	-87.5	NA
1985-10-01	-175.625	-87.5	NA
1985-10-01	-174.375	-87.5	NA
1985-10-01	-173.125	-87.5	NA
1985-10-01	-171.875	-87.5	NA
1985-10-01	-170.625	-87.5	NA
1985-10-01	-169.375	-87.5	NA
1985-10-01	-168.125	-87.5	NA

**<chr>** stands for  
character string  
(not a number)



# read\_csv()

readr functions share a common syntax

```
nimbus <- read_csv("nimbus.csv", na = ".",
  col_types = list(ozone = col_double()))
```

Manually  
specify column  
types.

list

column  
name

Column type  
function



<b>type function</b>	<b>data type</b>
col_character()	character
col_date()	Date
col_datetime()	POSIXct (date-time)
col_double()	double (numeric)
col_factor()	factor
col_guess()	let readr guess (default)
col_integer()	integer
col_logical()	logical
col_number()	numbers mixed with non-number characters
col_numeric()	double or integer
col_skip()	do not read
col_time()	time



## **type function**

## **data type**

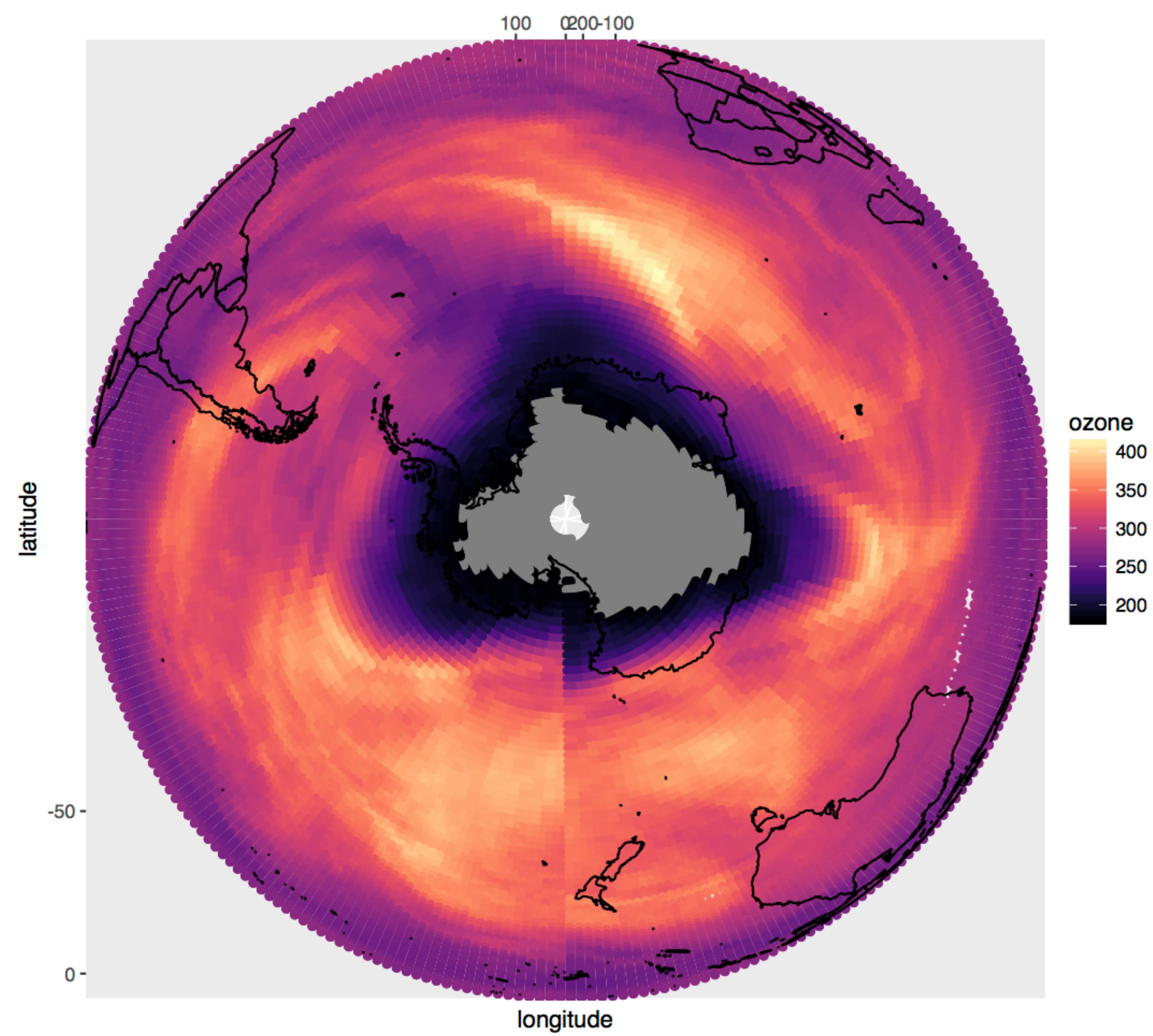
<b>type function</b>	<b>data type</b>
col_character()	character
col_date()	Date
col_datetime()	POSIXct (date-time)
<b>col_double()</b>	<b>double (numeric)</b>
col_factor()	factor
col_guess()	let readr guess (default)
col_integer()	integer
col_logical()	logical
col_number()	numbers mixed with non-number characters
col_numeric()	double or integer
col_skip()	do not read
col_time()	time



```
nimbus <- read_csv("nimbus.csv", na = ".",
col_types = list(ozone = col_double()))
```

```
library(viridis)
world <- map_data(map = "world")
nimbus %>%
  ggplot() +
  geom_point(aes(longitude, latitude, color = ozone)) +
  geom_path(aes(long, lat, group = group), data = world) +
  coord_map("ortho", orientation=c(-90, 0, 0)) +
  scale_color_viridis(option = "A")
```





# Writing



# readr functions

function	writes
write_csv()	Comma separated values
write_excel_csv()	CSV intended for opening in Excel
write_delim()	General delimited files
write_file()	Single string, written as is
write_lines()	Vector of strings, one element per line
write_tsv()	Tab delimited values



# write\_csv()

Saves data set as a csv on your computer.

```
write_csv(nimbus, file = "nimbus2.csv")
```

Table to save

file  
path to save at



# Reading SAS files

R

# haven



functions for reading in SAS, SPSS and  
Stata files

```
# install.packages("tidyverse")
library(haven)
```



# `read_sas()`

haven functions share a common syntax

```
df <- read_sas("path/to/file.csv", ...)
```

object to save  
output into

path from working  
directory to file



# Your Turn

The file **adae.sas7bdat** contains randomly generated data meant to mimic clinical trial data.

Load the data using **read\_sas** from the **haven** package.

View the resulting tibble both on the command line and by using the built-in Viewer (i.e. with **View(df)**). Are there any differences?

```
library(haven)
df = read_sas("adae.sas7bdat")
df
```

A tibble: 38 × 95

	STUDYID	SUBJID	SITEID	AGE	SEX	RACE	SAFFL	ARM	ARMCD	ACTARM
	<chr>	<chr>	<dbl>	<dbl>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
1	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
2	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
3	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
4	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
5	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
6	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
7	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
8	GS-US-...	1001	1000	20	F	WHITE	Y	[Pla...	P	[Plac...
9	GS-US-...	1002	1000	21	M	OTHER	Y	[20 ...	1	[20 m...
0	GS-US-...	1002	1000	21	M	OTHER	Y	[20 ...	1	[20 m...

... with 28 more rows, and 85 more variables: ACTARMCD <chr>, COHORT <chr>, COHORTN <dbl>, TRT01P <chr>, TRT01PN <dbl>,

04-Import-Data-Exercises.Rmd x df x

Filter Cols: << 1 - 50 >>

STUDYID Study Identifier SUBJID Subject Identifier SITEID Study Site Identifier AGE Age SEX Sex RACE Race SAFFL Safety Population Flag

	STUDYID Study Identifier	SUBJID Subject Identifier	SITEID Study Site Identifier	AGE Age	SEX Sex	RACE Race	SAFFL Safety Population Flag
1	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
2	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
3	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
4	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
5	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
6	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
7	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
8	GS-US-xxx-xxxx	1001	1000	20	F	WHITE	Y
9	GS-US-xxx-xxxx	1002	1000	21	M	OTHER	Y
10	GS-US-xxx-xxxx	1002	1000	21	M	OTHER	Y
11	GS-US-xxx-xxxx	1002	1000	21	M	OTHER	Y
12	GS-US-xxx-xxxx	1003	1000	20	M	WHITE	Y
13	GS-US-xxx-xxxx	1003	1000	20	M	WHITE	Y
14	GS-US-xxx-xxxx	1003	1000	20	M	WHITE	Y
15	GS-US-xxx-xxxx	1003	1000	20	M	WHITE	Y

# Final Exam!

## ggplot Exercises

1. Create a histogram of patient ages
2. Create a bar chart of patient races.

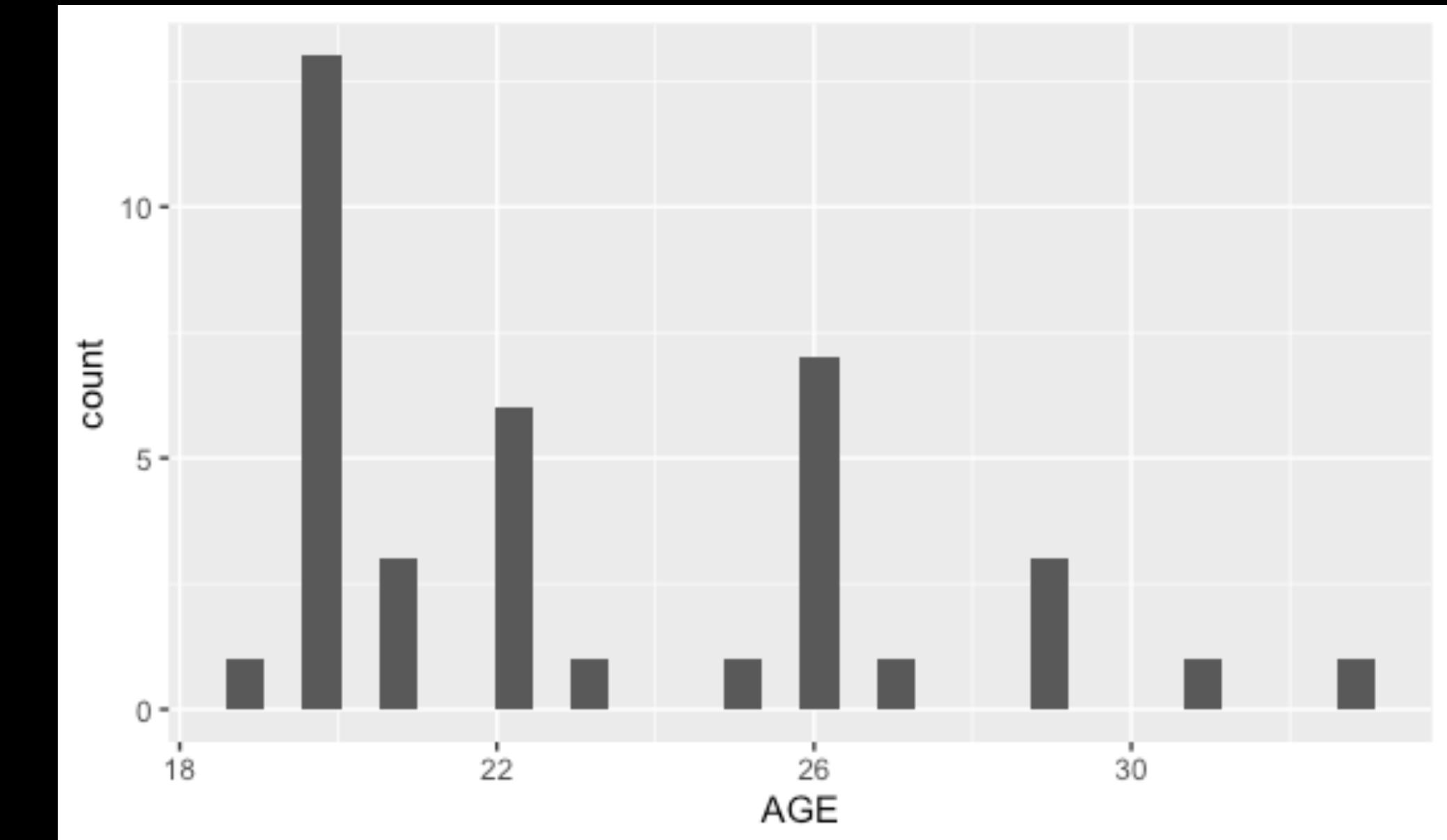
## dplyr exercises

3. How many males and females were in the study?
4. What is the average age of females in each cohort?

	SUBJID	SITEID	AGE	SEX	RACE
	Subject Identifier	Study Site Identifier	Age	Sex	Race
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1001	1000	20	F	WHITE
xxxx	1002	1000	21	M	OTHER
xxxx	1002	1000	21	M	OTHER
xxxx	1002	1000	21	M	OTHER
xxxx	1003	1000	20	M	WHITE
xxxx	1003	1000	20	M	WHITE
xxxx	1003	1000	20	M	WHITE
.....	1003	1000	20	M	WHITE

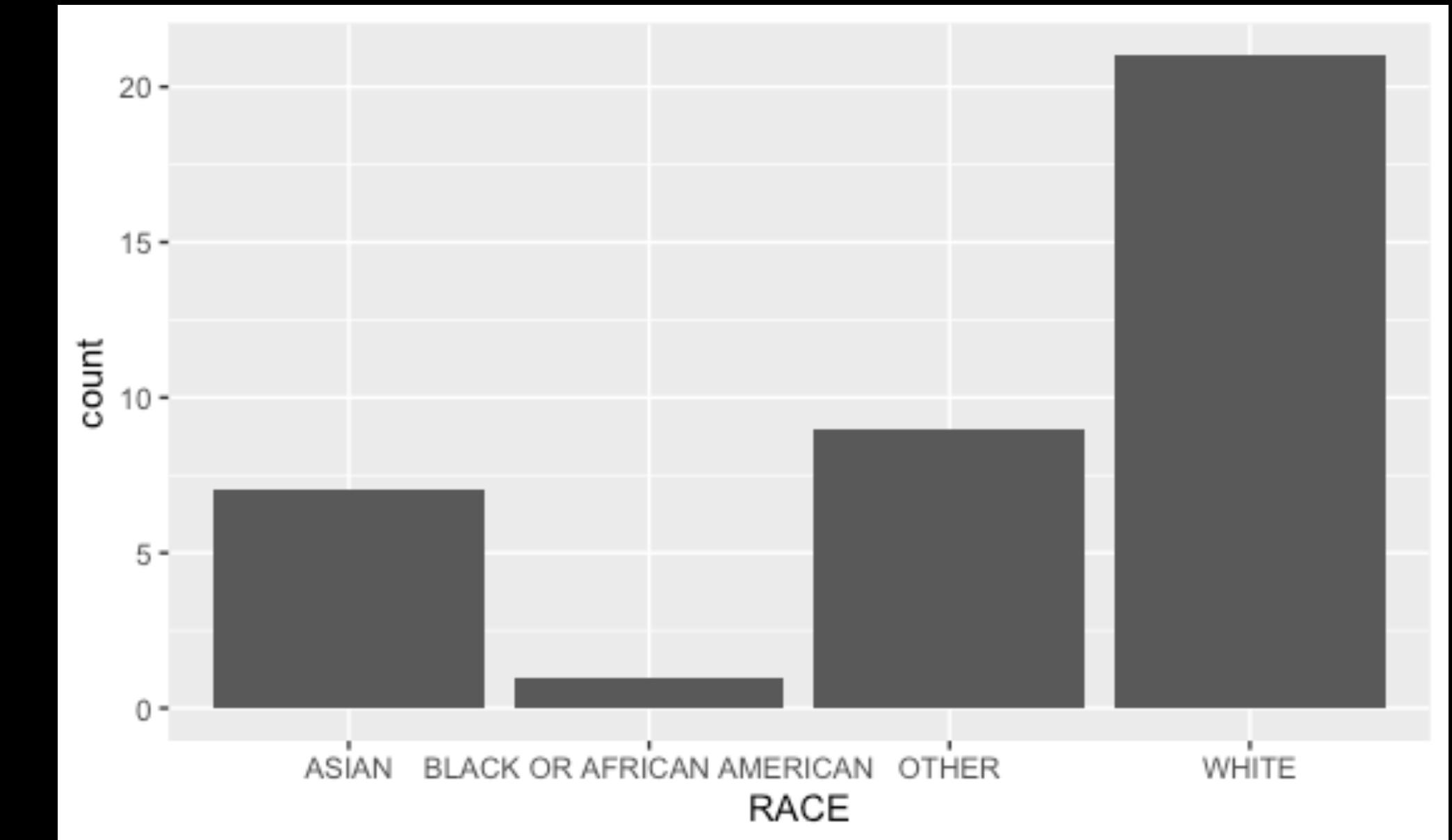
# Histogram of Ages

```
ggplot(df, aes(AGE)) +  
  geom_histogram()
```



# Bar Chart of Race

```
ggplot(df, aes(RACE)) +  
  geom_bar()
```



```
> df %>%  
+   group_by(SEX) %>%  
+   summarize(n=n())  
`summarise()` ungrouping ou  
# A tibble: 2 × 2  
  SEX       n  
  <chr> <int>  
1 F         18  
2 M         20
```

# Patients by Gender

```
> df %>%  
+   filter(SEX=="F") %>%  
+   group_by(COHORT) %>%  
+   summarize(av_age=mean(AGE))  
`summarise()` ungrouping output (i)  
# A tibble: 2 × 2  
  COHORT    av_age  
  <chr>      <dbl>  
1 Cohort 1    22.8  
2 Cohort 2    22
```

# Mean Female Age by Cohort

### Data Import

Main Ideas	Notes
	_____
	_____
	_____
	_____
	_____
	_____
	_____
	_____
	_____

Summary

How I can apply this to my work

# Notes form - please complete

# Homework

- Please practice reading in SAS files into R with **haven** ...
  - ... and visualizing the data with **ggplot2**
  - ... and transforming the data with **dplyr**
- Get stuck?
  - Refer to the **Notebooks** and **Notes** you completed
  - And the **Cheatsheets** you have
  - Ask **Google!**



Ari Lamstein

---

R Training and Consulting

---

AriLamstein.com

ari@lamsteinconsulting.com

San Francisco, California

# Post-Course Support

# Import Data with

