Bonus

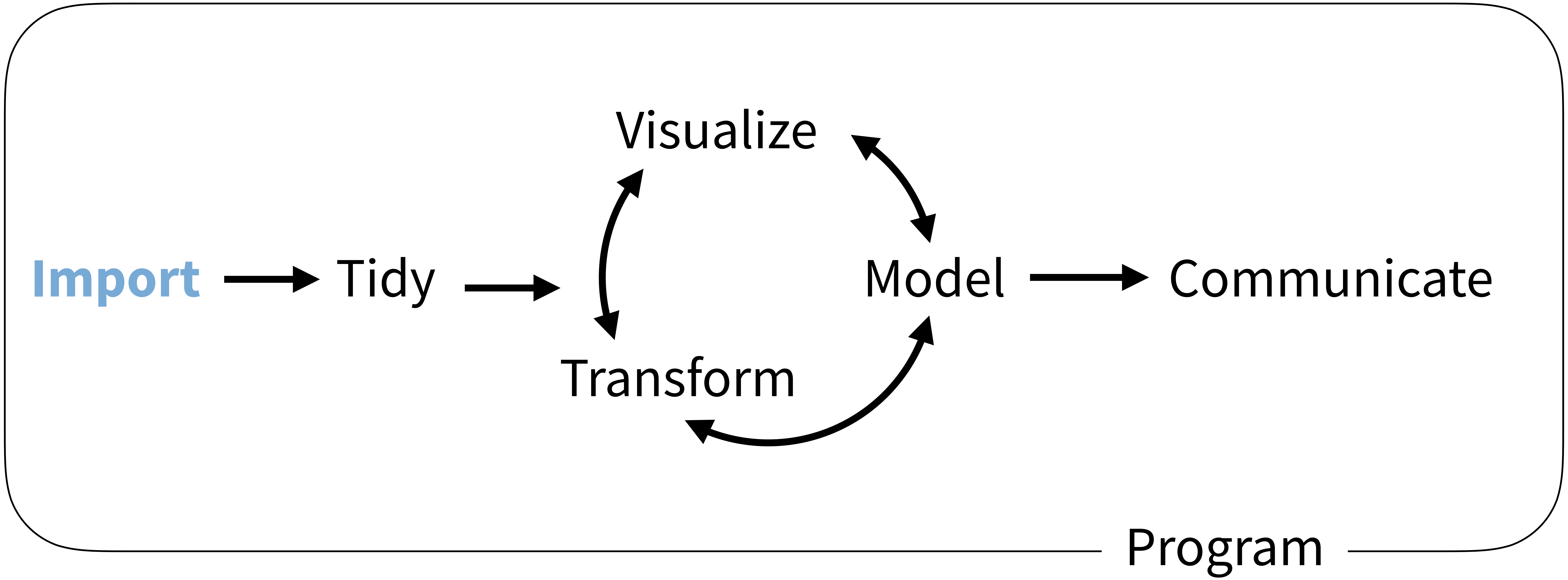
Import Data with

readr

www.rstudio.com

Open **04-Import-Data.Rmd**

(Applied) Data Science



Importing Data

readr

Simple, consistent functions for working with strings.

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

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
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

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

Your Turn 1

Find **nimbus.csv** on your server or computer. Then read it into an object. Then view the results.

02:00

Your Turn 1

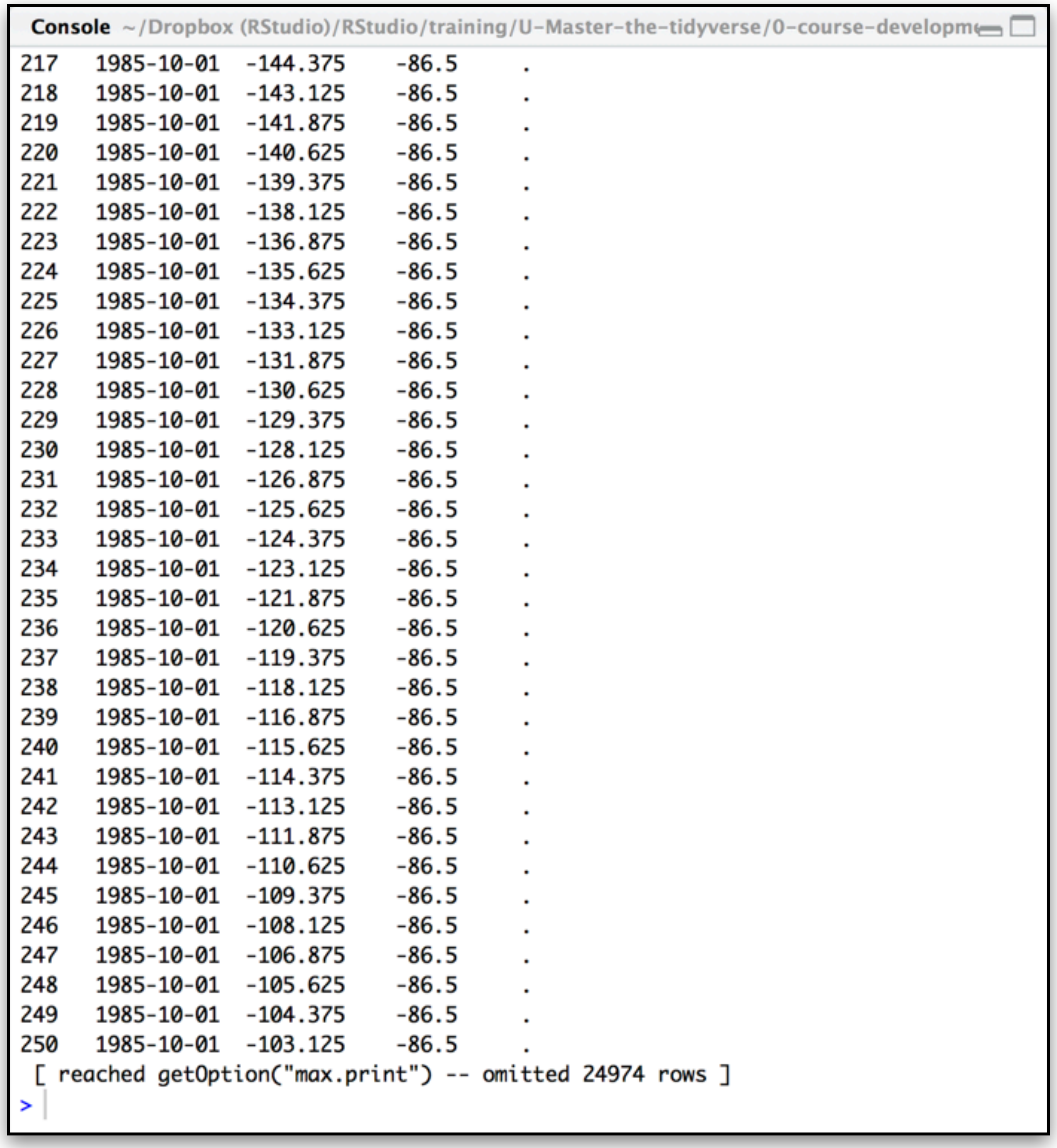
Find **nimbus.csv** on your server or computer. Then read it into an object. Then view the results.

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

nimbus
```

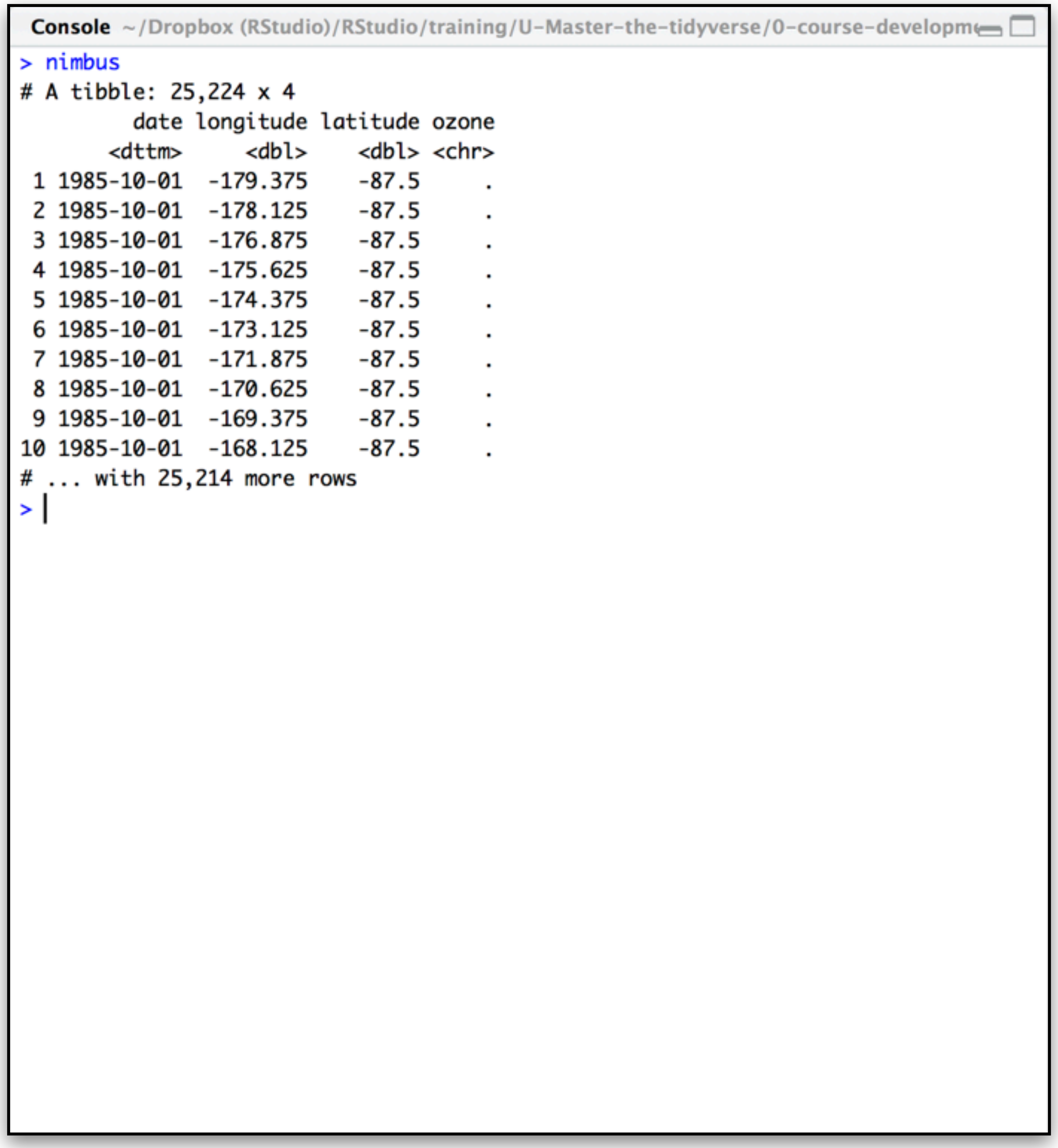
tibbles

read.csv() vs. read_csv()



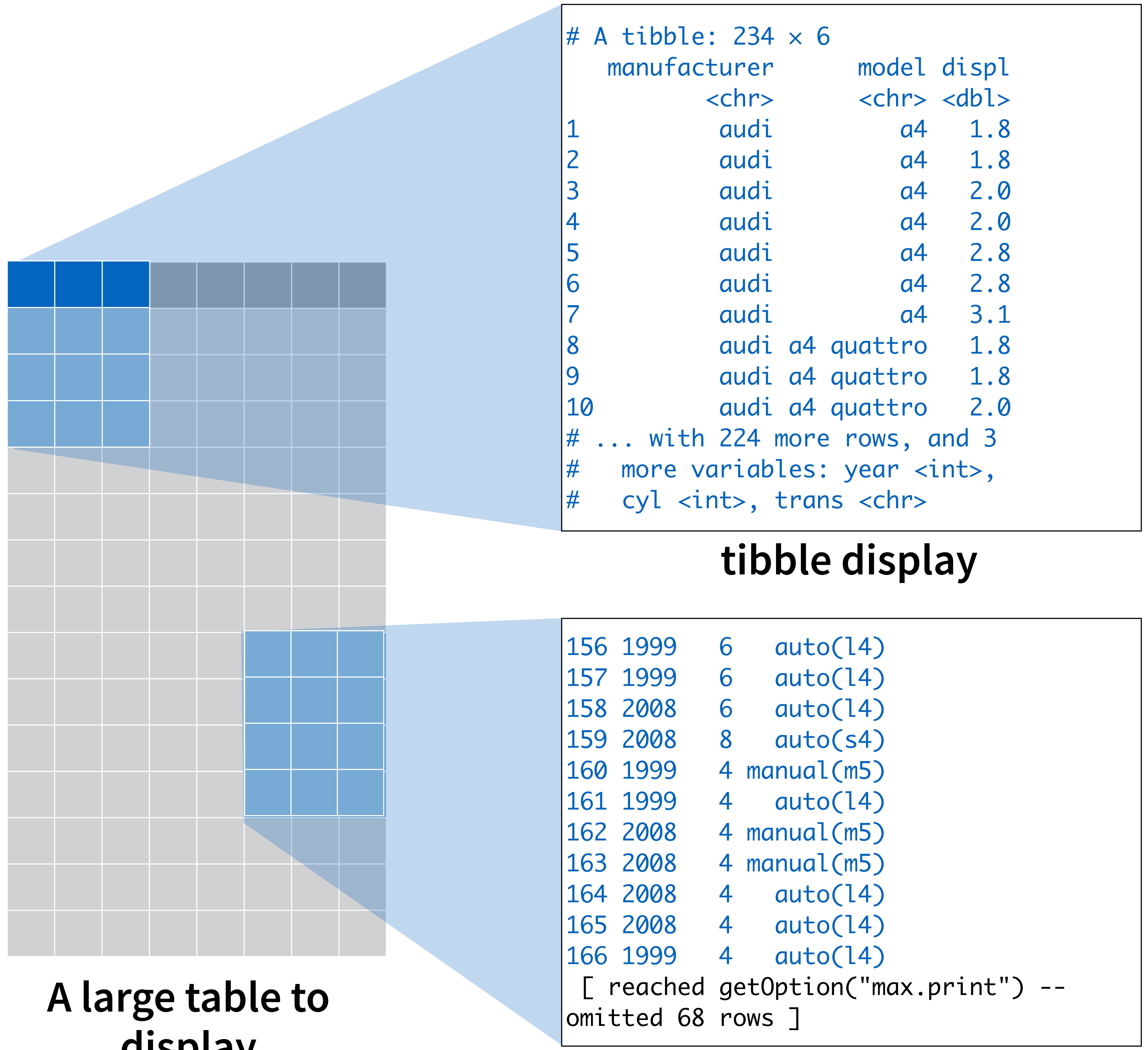
Console ~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse/0-course-developme

```
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-developme

```
> 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 tibble: 234 x 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

A large table to 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")
```

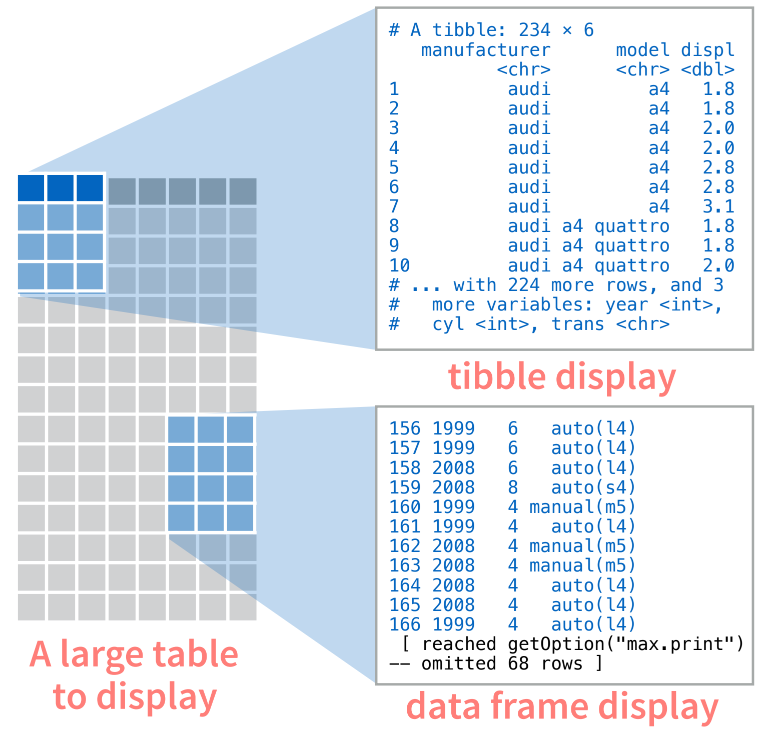
x	y
1	a
2	b
3	c

tibbles

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 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

- 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

tribble(…)
Construct by rows.
*tribble(
~x, ~y,
1, "a",
2, "b",
3, "c")*

```
A tibble: 3 × 2
      x     y
  <int> <dbl>
1     1     a
2     2     b
3     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.

Parsing

Quiz

What class is ozone?

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

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

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

```r
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" "." "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" "34
[170] "343" "348" "345" "354" "361" "372" "382" "376" "356" "293" "286" "353" "35
[183] "358" "360" "363" "370" "384" "380" "294" "339" "362" "352" "368" "373" "37
```

. = NA

```
nimbus
```

date <S3: POSIXct>	longitude <dbl>	latitude <dbl>	ozone <chr>
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

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

object to save output into

path from working directory to file

Value(s) to convert to NA

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

date <S3: POSIXct>	longitude <dbl>	latitude <dbl>	ozone <int>
1985-10-01	-179.375	-73.5	302
1985-10-01	-178.125	-73.5	302
1985-10-01	-176.875	-73.5	30
1985-10-01	-175.625	-73.5	30
1985-10-01	-174.375	-73.5	30
1985-10-01	-173.125	-73.5	304
1985-10-01	-171.875	-73.5	304
1985-10-01	-170.625	-73.5	304
1985-10-01	-164.375	-73.5	287

<int> stands for integer

suppose

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

date <S3: POSIXct>	longitude <dbl>	latitude <dbl>	ozone <chr>
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

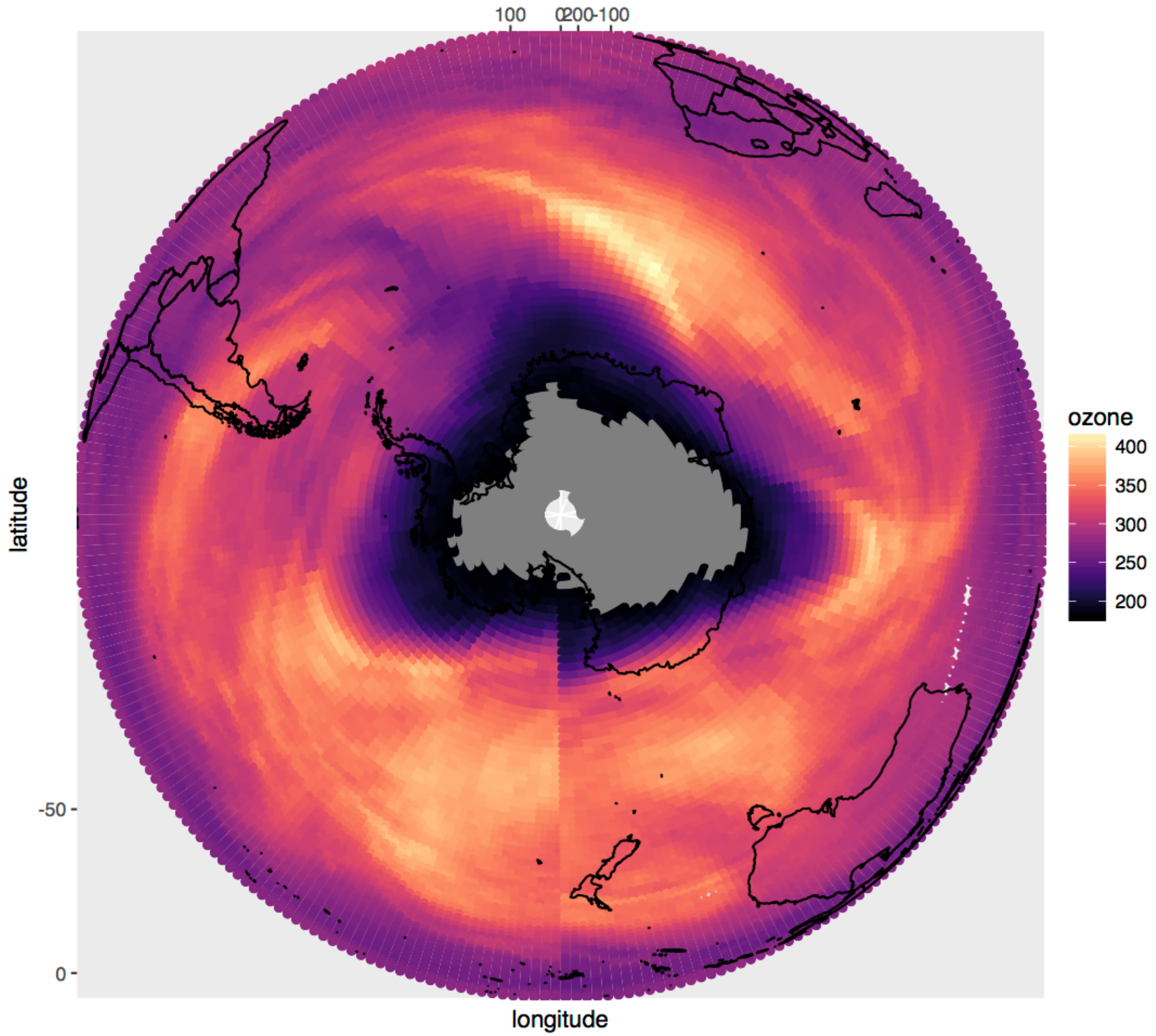
Column type function

type function	data type
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
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

```r
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")
```


100
0
200
-100
latitude
-50
0
longitude
ozone
400
350
300
250
200

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

Other types of data

package	accesses
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

Bonus

Import Data with

readr

www.rstudio.com