ANT 6973: DATA VISUALIZATION AND EXPLORATION

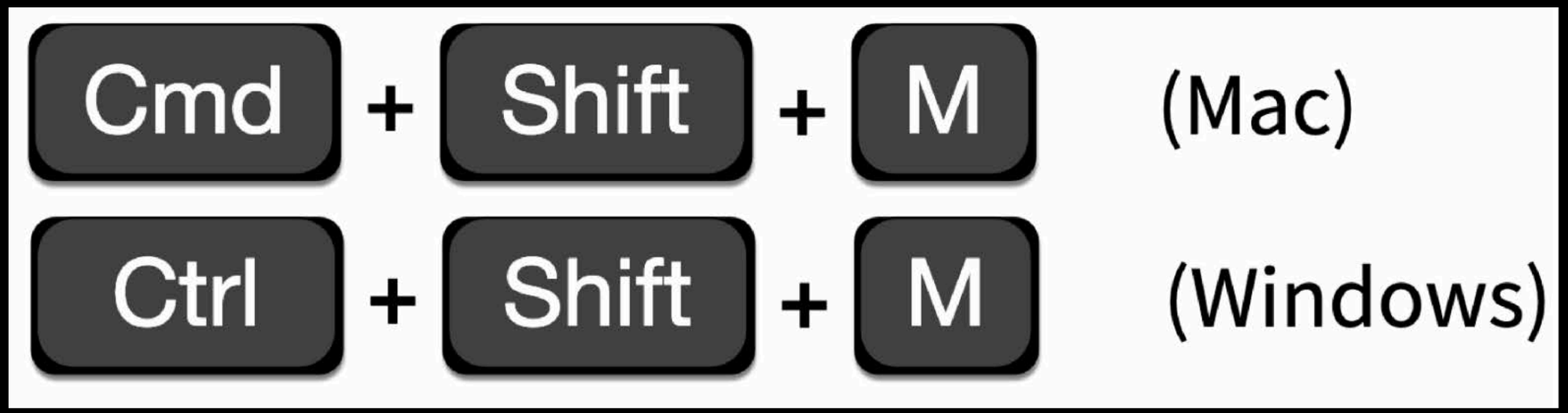
IMPORTING DATA FROM TEXT AND SPREADSHEETS

TODAY'S TOPICS

- Reading from plain text files (`readr`)
- Reading from spreadsheets (`readxl`)

REMINDER: PIPES

- Shortcut to type %>%



DATA IMPORT

PACKAGES FOR DATA IMPORT

`readr` Core part of


```
library("tidyverse")
```

`readxl` Not part of core

```
library("readxl")
```

PACKAGES FOR DATA IMPORT

haven

Import SAS, SPSS and STATA data files

`googlesheets4`

Import Google Sheet

PACKAGES FOR DATA IMPORT

OTHER TYPES OF DATA

Try one of the following packages to import other types of files

- **DBI** - databases
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)

And many more…

PREVIOUSLY...

- Import from a .csv file using `read_csv()`

```
my_data <- read_csv("path/to/data.csv")
```

New R object where data will be stored

`readr` function

Path of data file relative to project home

WRITE FILES

- Use the function `write_csv()` to write a data frame to disk.

```
write_csv(my_data, file = "path/my_data.csv")
```

`readr` function

Name of R object to export

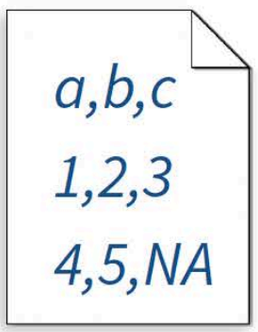
Name & path of file you want to create
note: folder must exist

FUNCTIONS

Function	Reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files (e.g., "|")
read_fwf()	Fixed width files
read_tsv()	Tab delimited values
read_table()	Space separated
read_table2()	Any number of whitespace characters

FUNCTIONS

Function	Reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files (e.g., "|")
read_fwf()	Fixed width files
read_tsv()	Tab delimited values
read_table()	Space separated



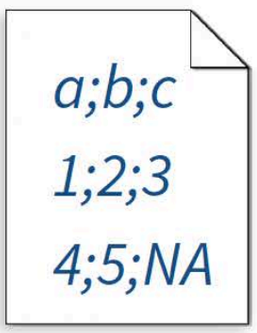
A	B	C
1	2	3
4	5	NA

Comma Delimited Files

read_csv("file.csv"**)**

FUNCTIONS

Function	Reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files (e.g., "|")
read_fwf()	Fixed width files
read_tsv()	Tab delimited values
read_table()	Space separated



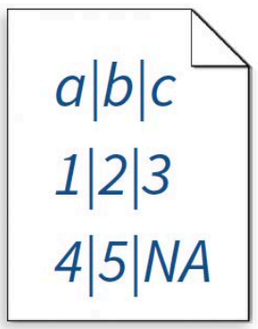
A	B	C
1	2	3
4	5	NA

Semi-colon Delimited Files

read_csv2("file2.csv"**)**

FUNCTIONS

Function	Reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files (e.g., "|")
read_fwf()	Fixed width files
read_tsv()	Tab delimited values
read_table()	Space separated



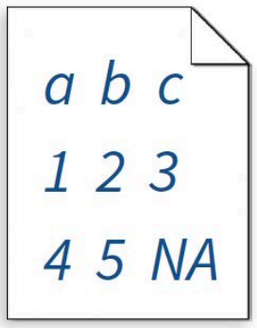
A	B	C
1	2	3
4	5	NA

Files with Any Delimiter

read_delim("file.txt", delim = "|"**)**

FUNCTIONS

Function	Reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files (e.g., "|")
read_fwf()	Fixed width files
read_tsv()	Tab delimited values
read_table()	Space separated



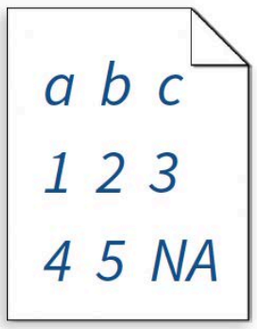
A	B	C
1	2	3
4	5	NA

Fixed Width Files

read_fwf("file.fwf", col_positions = c(1, 3, 5)**)**

FUNCTIONS

Function	Reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files (e.g., "|")
read_fwf()	Fixed width files
read_tsv()	Tab delimited values
read_table()	Space separated



A	B	C
1	2	3
4	5	NA

Tab Delimited Files

read_tsv("file.tsv"**)** Also **read_table().**

READ_X(): USEFUL OPTIONS

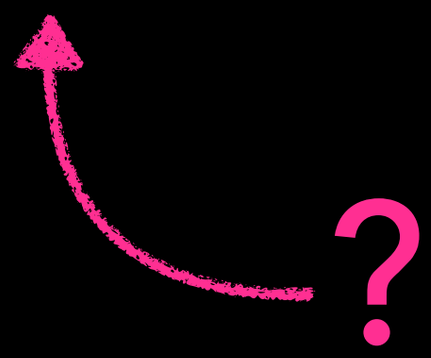
OPTION	DESCRIPTION
`skip`	How many lines of the start of the file should you skip?
`col_names`	What would you like to use as the column names?
`col_types`	What would you like to use as the column types?
`n_max`	How many rows do you want to read in?
`na`	How are missing values coded?
`guess_max`	How many rows should be used for guessing column types?

`col_names = FALSE` for files with no header

WHY USE NOT USE BASE R?

- ~10X faster than base R functions like `read.csv()`

- Returns a tibble rather than a data.frame



TIBBLE

- A type of data frame used throughout the tidyverse

`read.csv() -> data.frame`

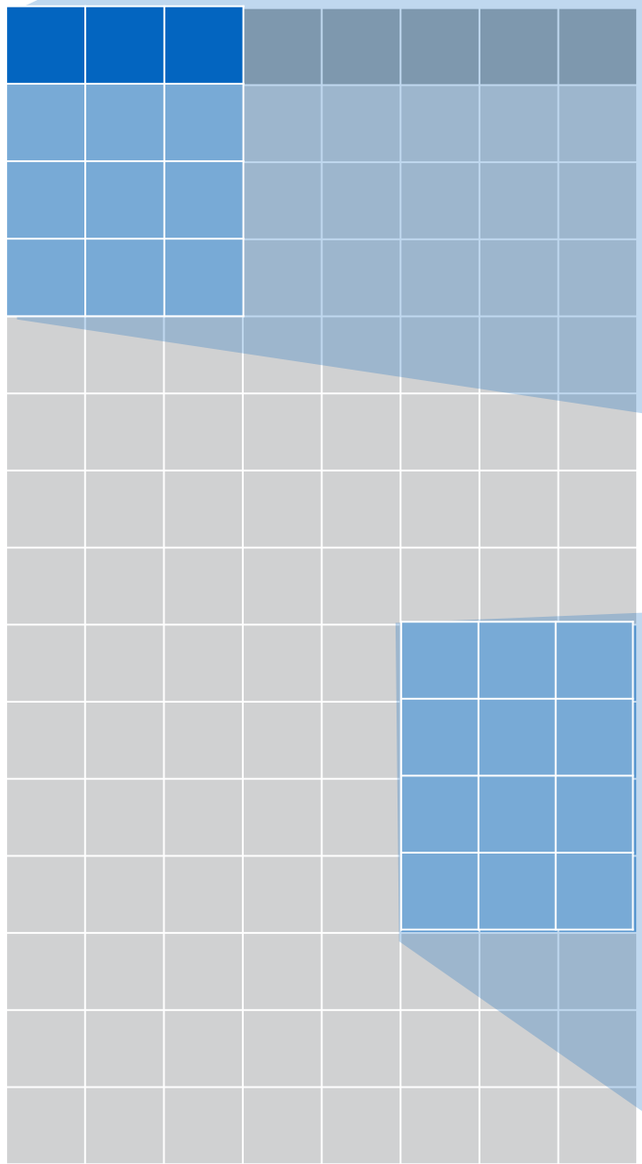
```
Console ~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse/0-course-developme
217   1985-10-01  -144.375    -86.5     .
218   1985-10-01  -143.125    -86.5     .
219   1985-10-01  -141.875    -86.5     .
220   1985-10-01  -140.625    -86.5     .
221   1985-10-01  -139.375    -86.5     .
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 ]
>
```

`read.csv()` -> `data.frame`

```
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 ]
>
```

`read_csv()` -> `tibble`

```
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 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>
```

Dimensions

Column types

Concise view of data that fits on screen.

All other columns listed

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 ]
```

Large number of rows printed before giving up.

Many rows and variables hidden.

data frame display

CREATING A TIBBLE

- From scratch

Function to construct a tibble

```
a <- 1:5
my_dat <- tibble(a  = a, b = a * 2, c = 1)
```

```
#> # A tibble: 5 x 3
#>       a     b     c
#>   <int> <dbl> <dbl>
#> 1     1     2     1
#> 2     2     4     1
#> 3     3     6     1
#> 4     4     8     1
#> 5     5    10     1
```

Columns are available in subsequent expressions

Only values of length 1 are recycled

CREATING A TIBBLE

- By coercion

```
head(mtcars)
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

CREATING A TIBBLE

- By coercion

Function to coerce a data frame or matrix to a tibble

```
as_tibble(mtcars)
```

Existing data frame or matrix

```
# A tibble: 32 x 11
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21        6  160    110  3.9   2.62  16.5     0     1     4     4
 2  21        6  160    110  3.9   2.88  17.0     0     1     4     4
 3  22.8      4  108     93  3.85  2.32  18.6     1     1     4     1
 4  21.4      6  258    110  3.08  3.22  19.4     1     0     3     1
 5  18.7      8  360    175  3.15  3.44  17.0     0     0     3     2
 6  18.1      6  225    105  2.76  3.46  20.2     1     0     3     1
 7  14.3      8  360    245  3.21  3.57  15.8     0     0     3     4
 8  24.4      4  147.    62  3.69  3.19  20       1     0     4     2
 9  22.8      4  141.    95  3.92  3.15  22.9     1     0     4     2
10  19.2      6  168.   123  3.92  3.44  18.3     1     0     4     4
# … with 22 more rows
```

But we're missing the row names...

TIBBLES AND ROW NAMES

- Tibbles do not use row names because they violate the principle of "tidy data."

- Names should be stored as a variable in a normal data column, not as a special attribute

TIBBLES AND ROW NAMES

Create new column (called "model") from existing row names

```
mtcars %>%
  rownames_to_column(var = "model") %>%
  as_tibble()
```

```
# A tibble: 32 x 12
   model              mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
   <chr>            <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 Mazda RX4         21       6  160    110  3.9   2.62  16.5     0     1     4     4
 2 Mazda RX4 Wag     21       6  160    110  3.9   2.88  17.0     0     1     4     4
 3 Datsun 710        22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
 4 Hornet 4 Drive    21.4     6  258    110  3.08  3.22  19.4     1     0     3     1
 5 Hornet Sportabout 18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
 6 Valiant           18.1     6  225    105  2.76  3.46  20.2     1     0     3     1
 7 Duster 360        14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
 8 Merc 240D         24.4     4  147.    62  3.69  3.19  20       1     0     4     2
 9 Merc 230          22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
10 Merc 280          19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
# … with 22 more rows
```

TIBBLES AND ROW NAMES

Create new column of sequential row IDs starting at 1

```
iris %>%
  rowid_to_column() %>%
  as_tibble()
```

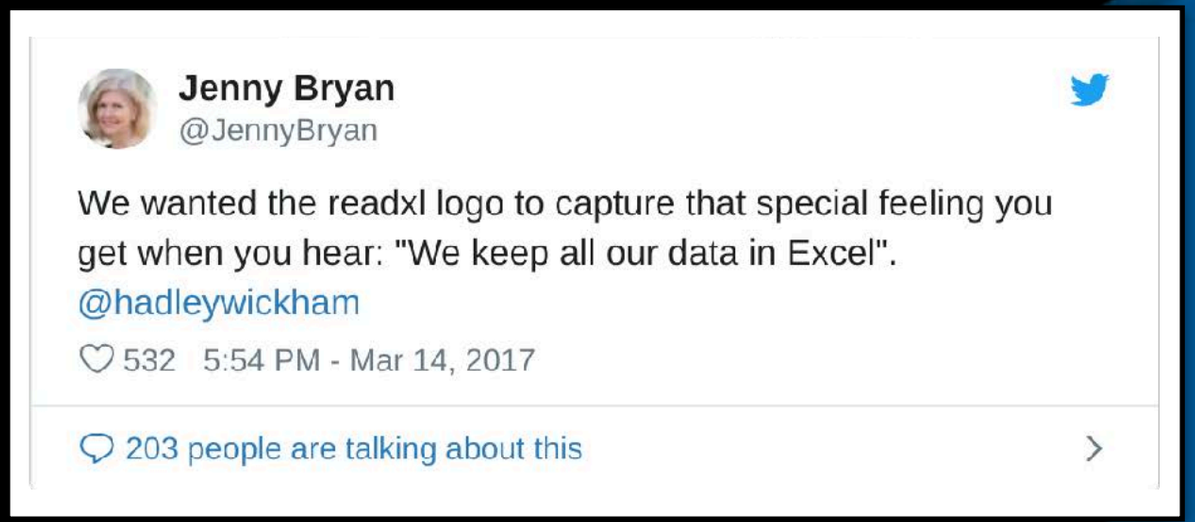
```
# A tibble: 150 x 6
   rowid Sepal.Length Sepal.Width Petal.Length Petal.Width Species
   <int>        <dbl>       <dbl>        <dbl>       <dbl> <fct>
 1     1          5.1         3.5          1.4         0.2 setosa
 2     2          4.9         3            1.4         0.2 setosa
 3     3          4.7         3.2          1.3         0.2 setosa
 4     4          4.6         3.1          1.5         0.2 setosa
 5     5          5           3.6          1.4         0.2 setosa
 6     6          5.4         3.9          1.7         0.4 setosa
 7     7          4.6         3.4          1.4         0.3 setosa
 8     8          5           3.4          1.5         0.2 setosa
 9     9          4.4         2.9          1.4         0.2 setosa
10    10          4.9         3.1          1.5         0.1 setosa
# … with 140 more rows
```

ACTIVITY: OZONE

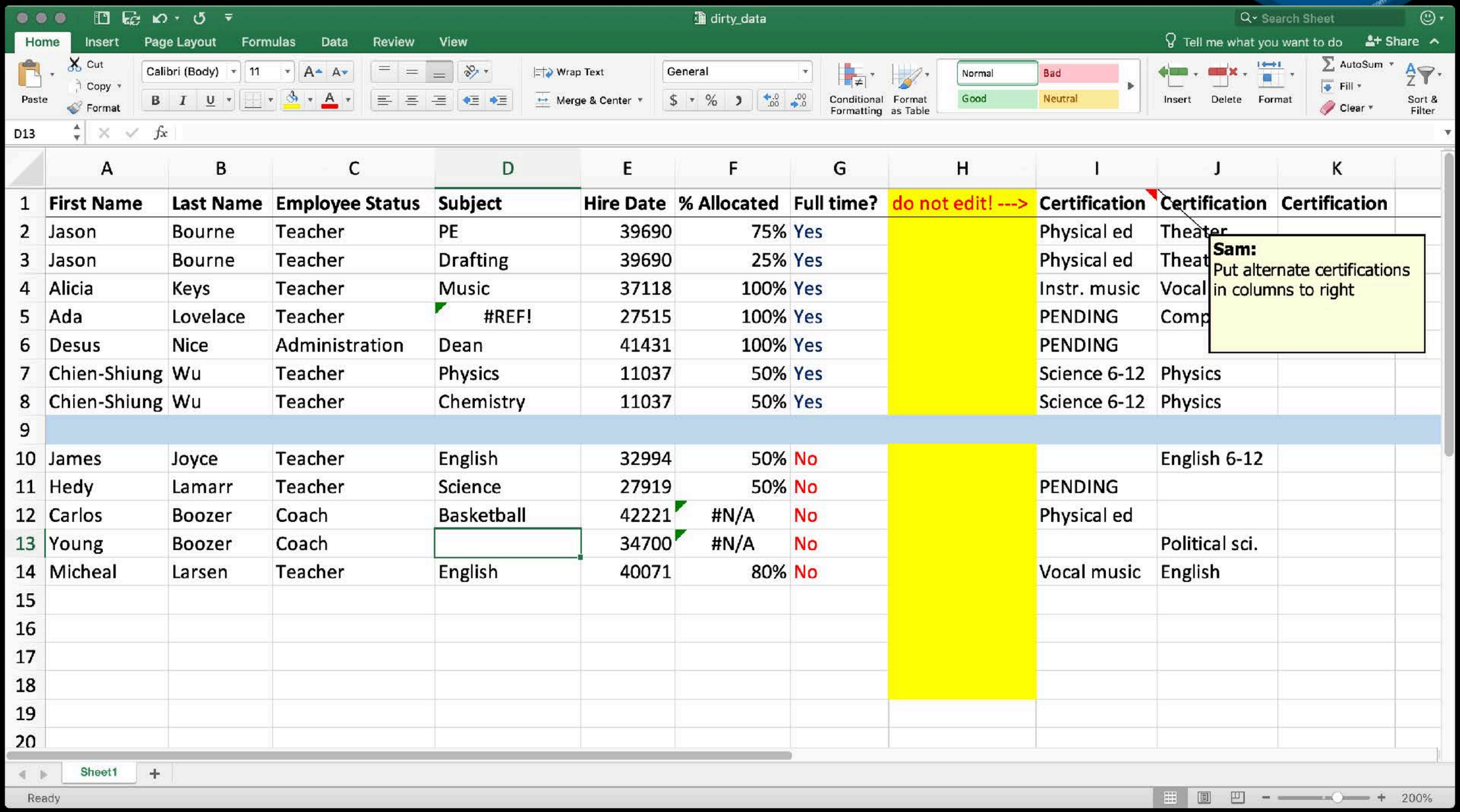
- Go to this week's assignments on the course website.
- Download ozone.Rmd and follow the instructions to complete the assignment.

IMPORTING SPREADSHEETS

- `readxl`



UGH...



	A	B	C	D	E	F	G	H	I	J	K
1	First Name	Last Name	Employee Status	Subject	Hire Date	% Allocated	Full time?	do not edit! --->	Certification	Certification	Certification
2	Jason	Bourne	Teacher	PE	39690	75%	Yes		Physical ed	Theater	
3	Jason	Bourne	Teacher	Drafting	39690	25%	Yes		Physical ed	Theat	
4	Alicia	Keys	Teacher	Music	37118	100%	Yes		Instr. music	Vocal	
5	Ada	Lovelace	Teacher	#REF!	27515	100%	Yes		PENDING	Comp	
6	Desus	Nice	Administration	Dean	41431	100%	Yes		PENDING		
7	Chien-Shiung	Wu	Teacher	Physics	11037	50%	Yes		Science 6-12	Physics	
8	Chien-Shiung	Wu	Teacher	Chemistry	11037	50%	Yes		Science 6-12	Physics	
9											
10	James	Joyce	Teacher	English	32994	50%	No			English 6-12	
11	Hedy	Lamarr	Teacher	Science	27919	50%	No		PENDING		
12	Carlos	Boozer	Coach	Basketball	42221	#N/A	No		Physical ed		
13	Young	Boozer	Coach		34700	#N/A	No			Political sci.	
14	Micheal	Larsen	Teacher	English	40071	80%	No		Vocal music	English	
15											
16											
17											
18											
19											
20											

data organization organizing data in spreadsheets

My collaborators sometimes ask me, "In what form would you like the data?" My response is always, "In its current form!" If the data need to be reformatted, it's much better for me to write a script than for them to do a bunch of cut-and-paste. I'm a strong proponent of data analysts being able to handle any data files they might receive.

But in many cases, I have to spend **a lot** of time writing scripts to rearrange the layout of the data. And how would you like your data analysts to spend their time? Reorganizing data, or really analyzing data?

Most of my collaborators enter and store their data in spreadsheets, and mostly Microsoft Excel. Before starting to enter data into a spreadsheet, it's good to spend some time thinking about the layout. The way that you organize the data in spreadsheets can have a big impact on your data analyst's quality of life.

This is a tutorial on that topic: *how to organize data in spreadsheets*. For complex, high-dimensional data, it may be better to use a formal database. But for many projects, spreadsheets are perfectly fine. But data in spreadsheets can be pretty and easy to work with, or they can be a sloppy mess requiring serious downstream reorganization efforts. We want to avoid the latter.

I don't think these ideas comes naturally to anyone. So if you're not happy with the structure of your current data files, don't despair! And also don't apply tedious and potentially error-prone hand-editing to revise the arrangement. Rather, apply these principles when designing the layout for your next dataset, to help make analyses easier.

READXL: A PRIMER

AutoSave OFF deaths

Home Insert Draw Page Layout Formulas Data Review View Acrobat Table Tell me Share Comments

Paste Font Alignment Number Conditional Formatting Format as Table Cell Styles Cells Editing Analyze Data Sensitivity Create and Share Adobe PDF

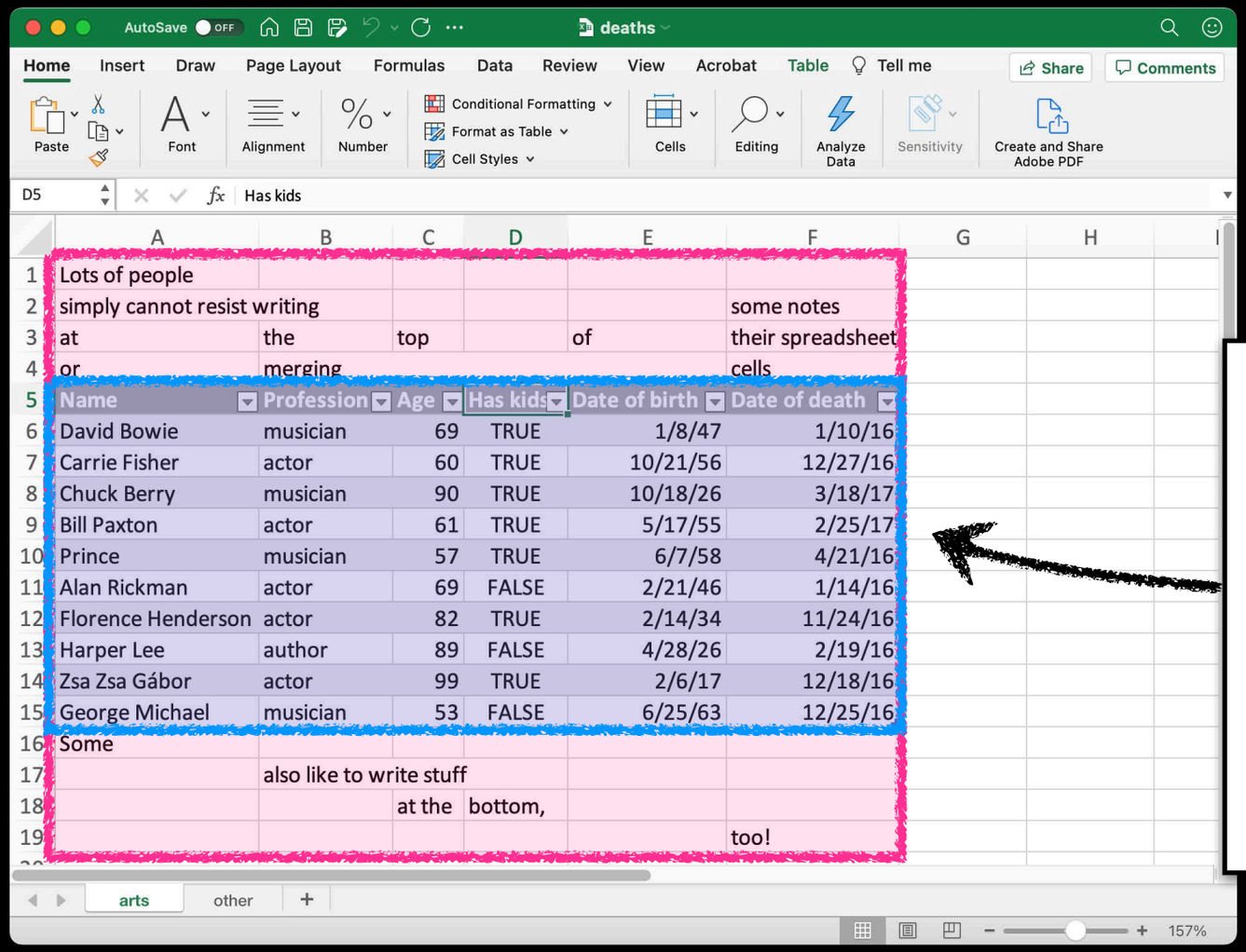
D5 fx Has kids

	A	B	C	D	E	F	G	H
1	Lots of people							
2	simply cannot resist writing					some notes		
3	at	the	top		of	their spreadsheets		
4	or	merging				cells		
5	Name	Profession	Age	Has kids	Date of birth	Date of death		
6	David Bowie	musician	69	TRUE	1/8/47	1/10/16		
7	Car							
8	Chu							
9	Bill							
10	Prince	musician	57	TRUE	6/7/58	4/21/16		
11	Alan Rickman	actor	69	FALSE	2/21/46	1/14/16		
12	Florence Henderson	actor	82	TRUE	2/14/34	11/24/16		
13	Harper Lee	author	89	FALSE	4/28/26	2/19/16		
14	Zsa Zsa Gábor	actor	99	TRUE	2/6/17	12/18/16		
15	George Michael	musician	53	FALSE	6/25/63	12/25/16		
16	Some							
17		also like to write stuff						
18			at the	bottom,				
19						too!		

arts other +

157%

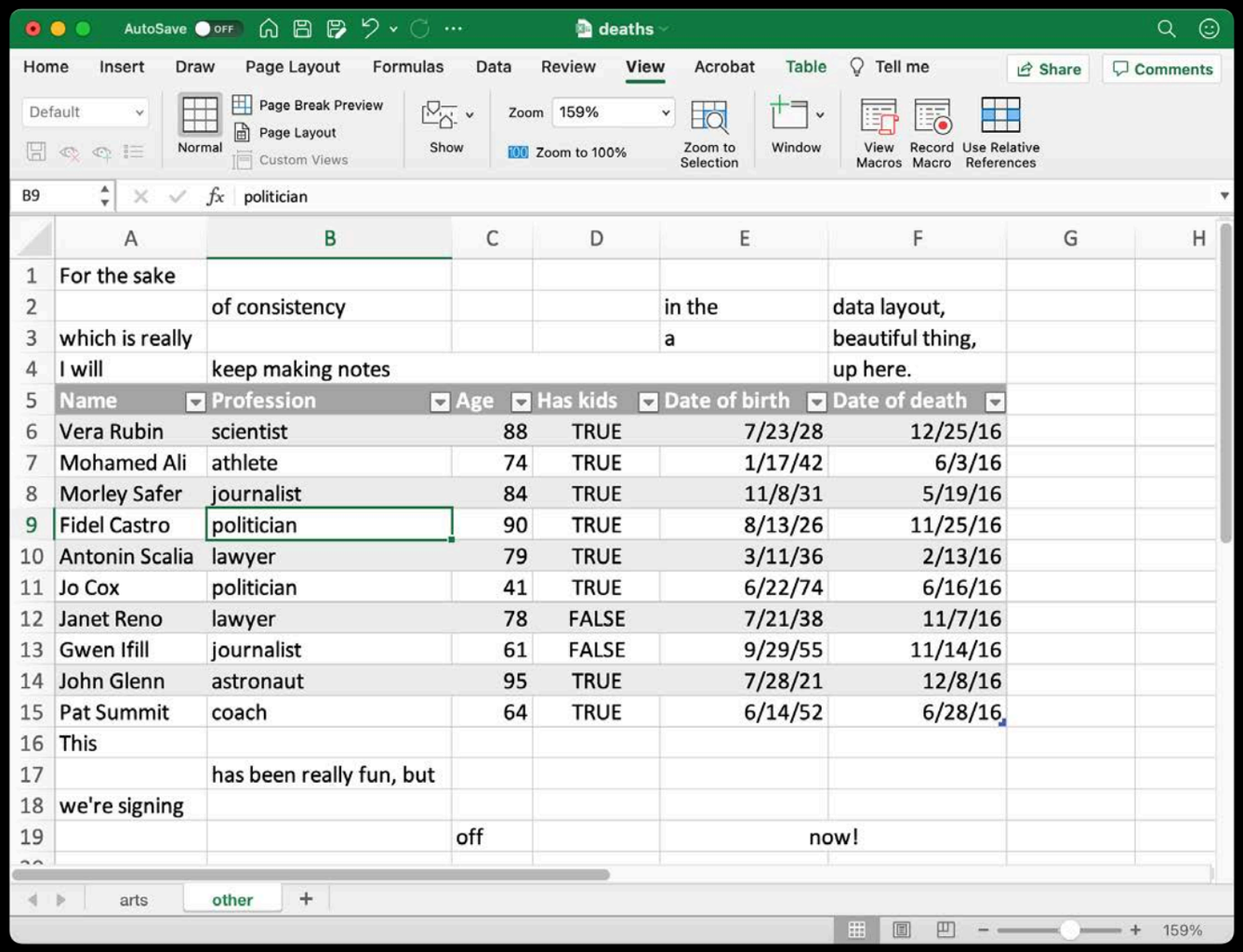
`readxl` is all about the data rectangle



Name	Profession	Age	Has kids	Date of birth	Date of death
David Bowie	musician	69	TRUE	1/8/47	1/10/16
Carrie Fisher	actor	60	TRUE	10/21/56	12/27/16
Chuck Berry	musician	90	TRUE	10/18/26	3/18/17
Bill Paxton	actor	61	TRUE	5/17/55	2/25/17
Prince	musician	57	TRUE	6/7/58	4/21/16
Alan Rickman	actor	69	FALSE	2/21/46	1/14/16
Florence Henderson	actor	82	TRUE	2/14/34	11/24/16
Harper Lee	author	89	FALSE	4/28/26	2/19/16
Zsa Zsa Gábor	actor	99	TRUE	2/6/17	12/18/16
George Michael	musician	53	FALSE	6/25/63	12/25/16

- 4 lines of non-data at top
- 1 row of column names
- 10 rows of data
- 4 rows of non-data at bottom
- Rectangle: 11 rows x 6 cols

```
✗ read_excel("deaths.xlsx", skip = 4)
✗ read_excel("deaths.xlsx", n_max = 14)
✓ read_excel("deaths.xlsx", skip = 4, n_max = 10)
✓ read_excel("deaths.xlsx", range = "A5:F15")
✗ read_excel("deaths.xlsx", range = cell_rows(5:15))
✗ read_excel("deaths.xlsx", range = cell_cols("A:F"))
✓ read_excel("deaths.xlsx", range = anchored("A5", dim = c(11, 6)))
```

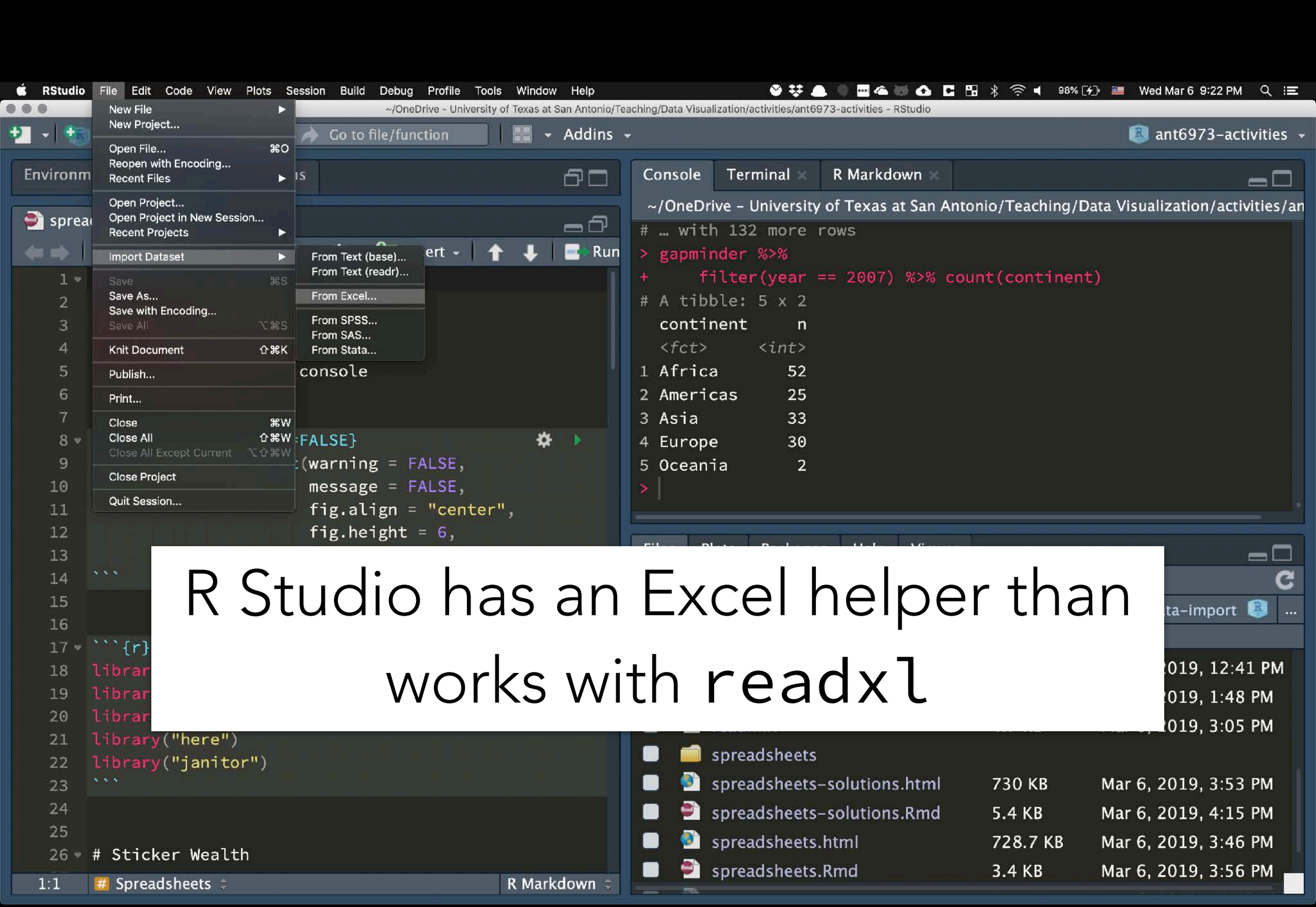


	A	B	C	D	E	F	G	H
1	For the sake							
2		of consistency			in the	data layout,		
3	which is really				a	beautiful thing,		
4	I will	keep making notes				up here.		
5	Name	Profession	Age	Has kids	Date of birth	Date of death		
6	Vera Rubin	scientist	88	TRUE	7/23/28	12/25/16		
7	Mohamed Ali	athlete	74	TRUE	1/17/42	6/3/16		
8	Morley Safer	journalist	84	TRUE	11/8/31	5/19/16		
9	Fidel Castro	politician	90	TRUE	8/13/26	11/25/16		
10	Antonin Scalia	lawyer	79	TRUE	3/11/36	2/13/16		
11	Jo Cox	politician	41	TRUE	6/22/74	6/16/16		
12	Janet Reno	lawyer	78	FALSE	7/21/38	11/7/16		
13	Gwen Ifill	journalist	61	FALSE	9/29/55	11/14/16		
14	John Glenn	astronaut	95	TRUE	7/28/21	12/8/16		
15	Pat Summit	coach	64	TRUE	6/14/52	6/28/16		
16	This							
17		has been really fun, but						
18	we're signing							
19			off		now!			

We can prepend the worksheet name to read from other sheets

```
read_excel("deaths.xlsx", range = "other!A5:F15")
read_excel("deaths.xlsx", sheet = "other", range = "A5:F15")
```

Or use the sheet argument



R Studio has an Excel helper than works with `readxl`

~/OneDrive - University of Texas at San Antonio/Teaching/Data Visualization/activities/ant6973-activities - RStudio

Import Excel Data

File/Url:

~/OneDrive - University of Texas at San Antonio/Teaching/Data Visualization/activities/ant6973-activities/08_data-import/spreadsheets/deaths.xlsx

Browse...

Data Preview:

Name *(character)*	Profession *(character)*	Age *(numeric)*	Has kids *(double)*	Date of birth *(date)*	Date of death *(date)*
David Bowie	musician	69	1	Character	2016-01-10
Carrie Fisher	actor	60	1	Numeric	2016-12-27
Chuck Berry	musician	90	1	Date	2017-03-18
Bill Paxton	actor	61	1	Include	2017-02-25
Prince	musician	57	1	Skip	2016-04-21
Alan Rickman	actor	69	0		2016-01-14
Florence Henderson	actor	82	1	1934-02-14	2016-11-24
Harper Lee	author	89	0	1926-04-28	2016-02-19
Zsa Zsa Gábor	actor	99	1	1917-02-06	2016-12-18
George Michael	musician	53	0	1963-06-25	2016-12-25

Previewing first 50 entries.

Import Options:

Name: deaths

Sheet: Default

Range: A1:D10

Max Rows: 10

Skip: 4

NA:

First Row as Names

Open Data Viewer

Code Preview:

```
library(readxl)
deaths <- read_excel("08_data-import/spreadsheets/deaths.xlsx",
    col_types = c("text", "text", "numeric",
        "numeric", "date", "date"), skip = 4,
    n_max = 10)
View(deaths)
```

Reading Excel files using readxl

Import

Cancel

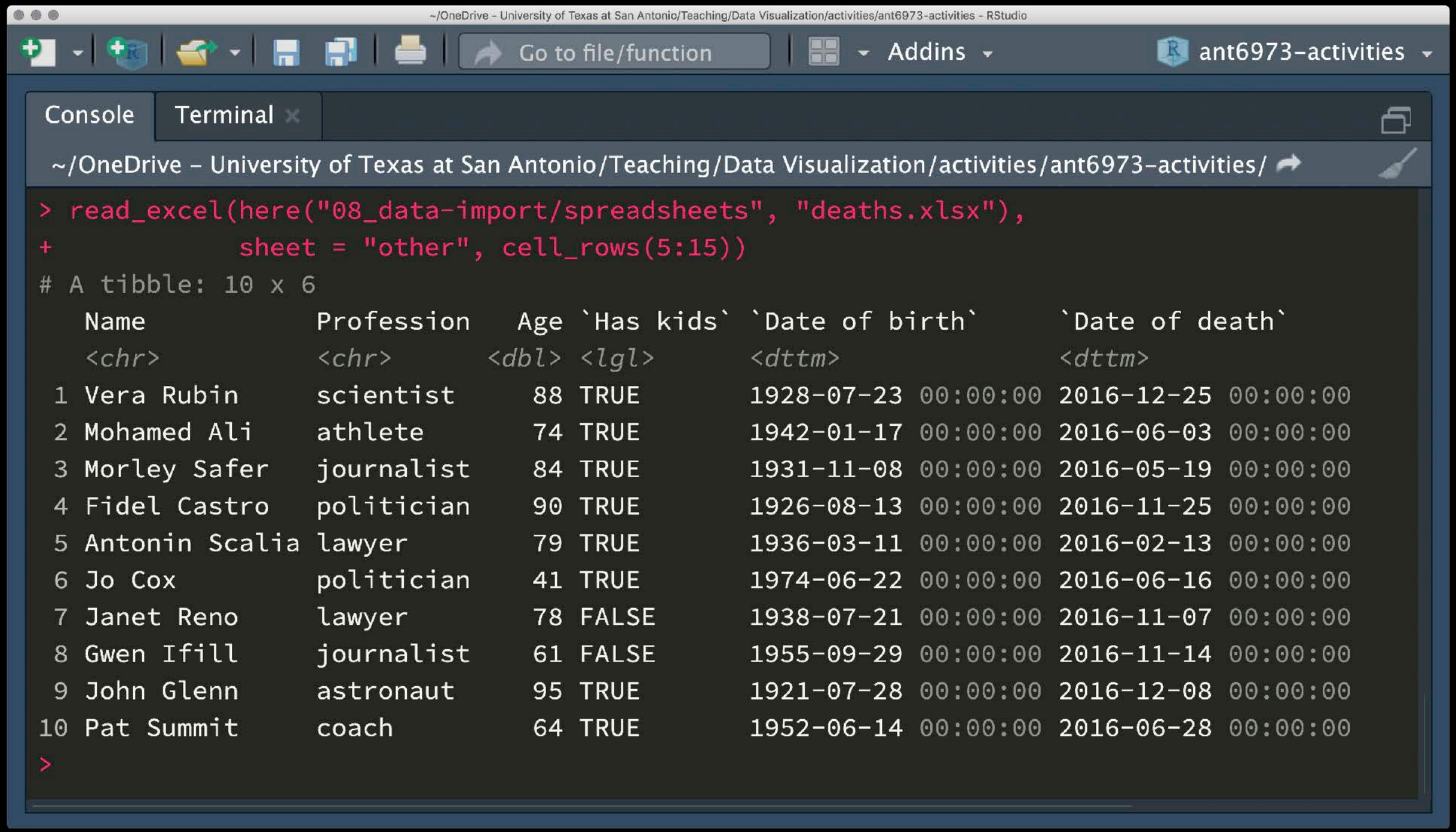
Environment | History | Connections | Console | Terminal | R Markdown

Global Env | Data | gm_2007 | Values | my_data

spreadsheets.Rmd 3.4 KB Mar 6, 2019, 3:56 PM

sst.R 1.4 KB Mar 6, 2019, 12:02 PM

1:1 | Spreadsheets | R Markdown



`readxl` is good at guessing column types

JANITOR

- A small package with a few simple and very convenient functions for cleaning messy data

JANITOR: CLEAN_NAMES()

- Makes column names that are unique and use a consistent style with `clean_names()`.

- By default, creates names that use only lowercase letters, numbers and _ character as a separator.

- Handles special characters and spaces.

- Appends numbers to duplicated names.

- Converts “%” to “percent” and "#" to "number".

JANITOR: CLEAN_NAMES()


```
read_excel("deaths.xlsx", range = c("A5:F15"))
```

Inconvenient names

```
# A tibble: 10 x 6
   Name              Profession   Age `Has kids` `Date of birth`     `Date of death`
   <chr>             <chr>      <dbl> <lgl>      <dttm>              <dttm>
 1 David Bowie       musician      69 TRUE       1947-01-08 00:00:00 2016-01-10 00:00:00
 2 Carrie Fisher     actor         60 TRUE       1956-10-21 00:00:00 2016-12-27 00:00:00
 3 Chuck Berry       musician      90 TRUE       1926-10-18 00:00:00 2017-03-18 00:00:00
 4 Bill Paxton       actor         61 TRUE       1955-05-17 00:00:00 2017-02-25 00:00:00
 5 Prince            musician      57 TRUE       1958-06-07 00:00:00 2016-04-21 00:00:00
 6 Alan Rickman      actor         69 FALSE      1946-02-21 00:00:00 2016-01-14 00:00:00
 7 Florence Hender… actor          82 TRUE       1934-02-14 00:00:00 2016-11-24 00:00:00
 8 Harper Lee        author        89 FALSE      1926-04-28 00:00:00 2016-02-19 00:00:00
 9 Zsa Zsa Gábor     actor         99 TRUE       1917-02-06 00:00:00 2016-12-18 00:00:00
10 George Michael    musician      53 FALSE      1963-06-25 00:00:00 2016-12-25 00:00:00
```

JANITOR: CLEAN_NAMES()


```
read_excel("deaths.xlsx", range = c("A5:F15")) %>%
  clean_names()
```

Clean, consistent names

```
# A tibble: 10 x 6
   name                profession   age has_kids date_of_birth       date_of_death
   <chr>               <chr>      <dbl> <lgl>    <dttm>              <dttm>
 1 David Bowie         musician      69 TRUE     1947-01-08 00:00:00 2016-01-10 00:00:00
 2 Carrie Fisher       actor         60 TRUE     1956-10-21 00:00:00 2016-12-27 00:00:00
 3 Chuck Berry         musician      90 TRUE     1926-10-18 00:00:00 2017-03-18 00:00:00
 4 Bill Paxton         actor         61 TRUE     1955-05-17 00:00:00 2017-02-25 00:00:00
 5 Prince              musician      57 TRUE     1958-06-07 00:00:00 2016-04-21 00:00:00
 6 Alan Rickman        actor         69 FALSE    1946-02-21 00:00:00 2016-01-14 00:00:00
 7 Florence Henderson  actor         82 TRUE     1934-02-14 00:00:00 2016-11-24 00:00:00
 8 Harper Lee          author        89 FALSE    1926-04-28 00:00:00 2016-02-19 00:00:00
 9 Zsa Zsa Gábor       actor         99 TRUE     1917-02-06 00:00:00 2016-12-18 00:00:00
10 George Michael      musician      53 FALSE    1963-06-25 00:00:00 2016-12-25 00:00:00
```

JANITOR: CLEAN_NAMES()

- Good idea to use *every time* you read in data!

JANITOR: OTHER STUFF

- Remove empty rows and columns with `remove_empty()`
- Convert excel dates (or things that look like dates in Excel) with `convert_to_date()`
- Use directionally consistent rounding with `round_half_up()`

ACTIVITY: SPREADSHEETS

- Go to this week's assignments on the course website.

- Download spreadsheets.Rmd and follow the instructions to complete the assignment.