

# **Introduction to R Workshop**

**Session 2  
Reproducible  
Reporting**  
May 6, 2019

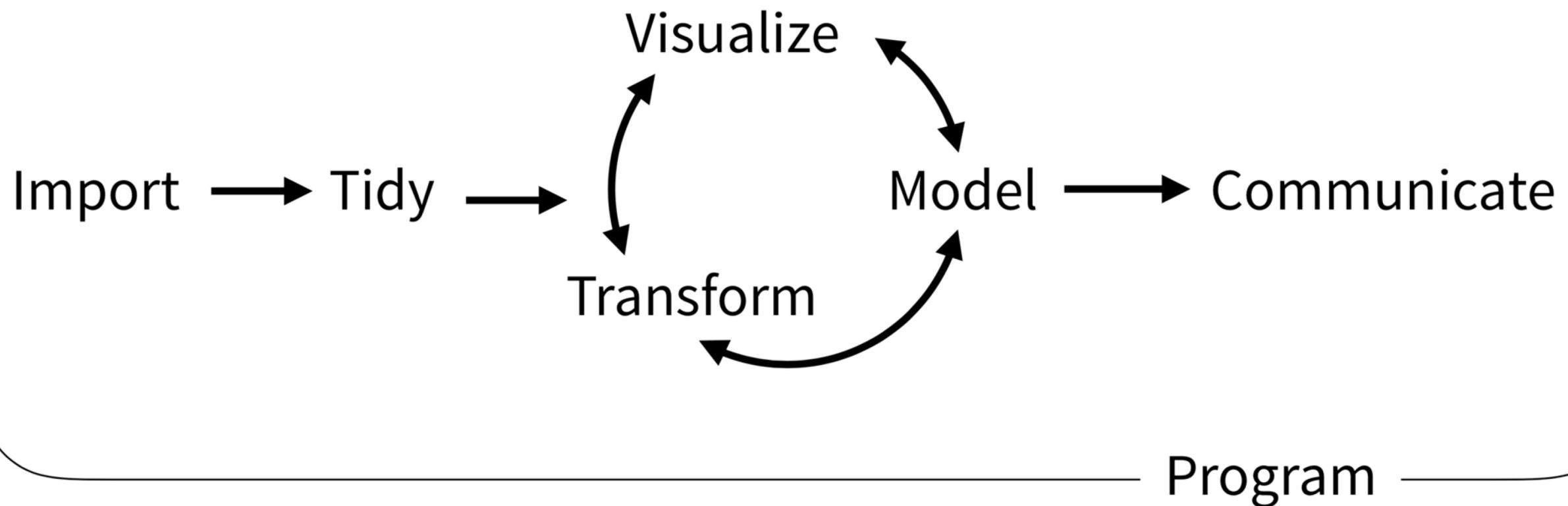
# Goals

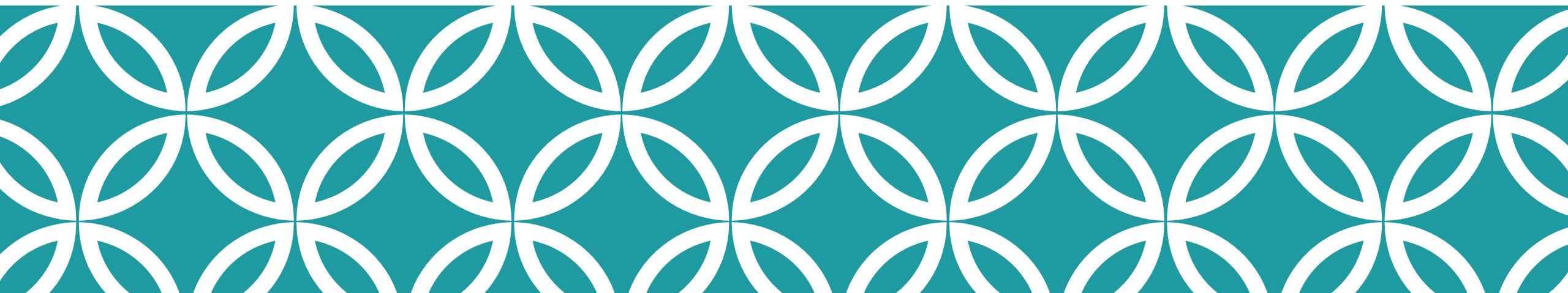
1. Learn to work within R Markdown for reproducible reports
2. Practice initial steps of the data import and analysis process

# Objectives

1. Understand why reproducible reporting is important
2. Create an R Markdown document and generate different types of output files
3. Import data from files into your R environment
4. Create basic summaries of data via tabulation and simple plots

# Typical Data Science Pipeline





**Why is reproducibility important?**

# Replication vs Reproduction

- ❖ Replication: other people collect new data
  - Scientific gold standard
  - Difficult and time-consuming
  
- ❖ Reproduction: other people analyze the same data
  - Does not by itself validate the analysis ...
  - Has been proposed as a minimal standard

# Case

37 y/o M informatician with PMH of email overload disorder

Request from informatics staff:

“Please provide detailed data from your 2 year old analysis of total departmental effort spent performing test cancellations for a SBAR calling out the need to invest effort in duplicate checking rules for the ongoing EHR implementation project”

Analysis is in an Excel file but original raw data is nowhere to be found

Consider the above scenario, but with someone else performing the original analysis

Would it be less work to start from scratch and rewrite the analysis?

# Point-and-Click Is Not Reproducible

- Interactive tools do not record user actions
- Manual documentation is error-prone
- Manual analyses cannot be repeated on new data sets or shared with collaborators



Computer code can precisely document each step of the analysis

# Why YOU Should Do Data Analysis Reproducibly

“Can we redo the analysis with this month’s data?”

“Why do the data in Table 1 not seem to agree with Figure 2?”

“Why did I decide to omit these six samples?”



**YOUR CLOSEST COLLABORATOR IS YOU FROM 6 MONTHS AGO  
(BUT YOU DON’T ANSWER E-MAILS)**

# Using R for Reproducibility

Programming in R (or another language) allows one to reproduce analysis steps exactly or perform same analysis on new data

Better practice is to create documentation about analysis to accompany and explain code

Best practice is include documentation and code in one place

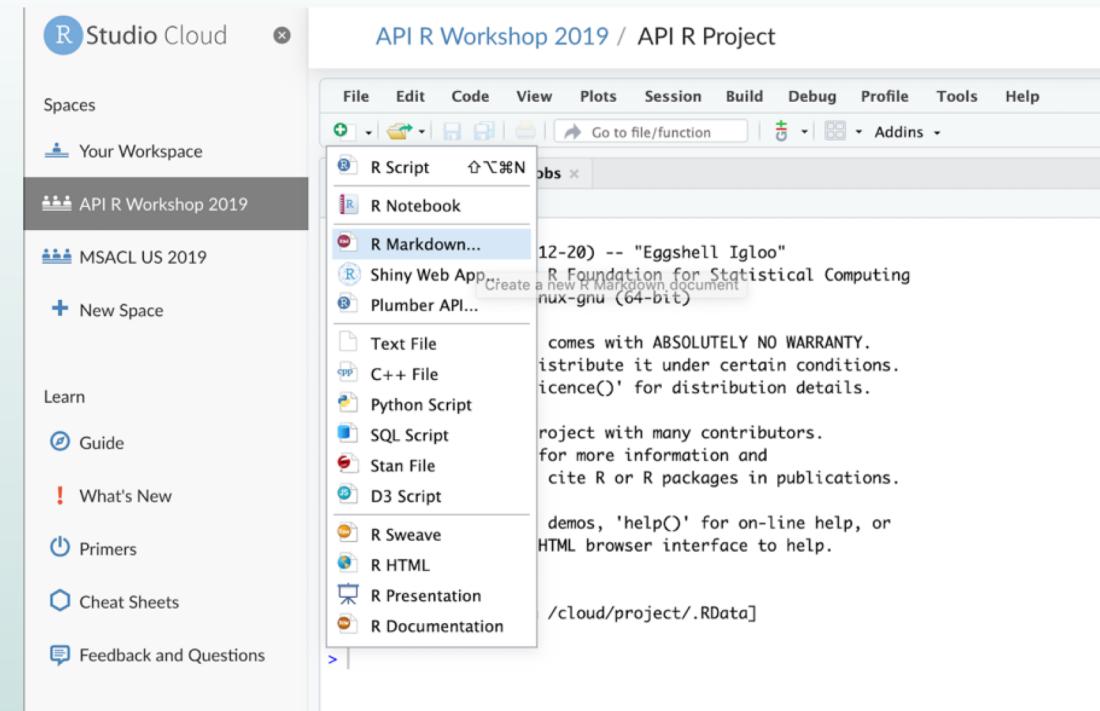


# Using R Markdown to Support Reproducibility



# Your Turn #1

1. Open a new R Markdown document within RStudio Cloud
2. Enter a Title and Author and leave the output format as HTML
3. Click each of the buttons in the top right corner of the grey areas and observe what happens
4. “Knit” the document by clicking the Knit button and save the file



When you click the **\*\*Knit\*\*** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
```{r cars}  
summary(cars)  
```
```

**## Including Plots**

You can also embed plots, for example:





# Header

Starts and ends  
with 3 dashes

A screenshot of the RStudio interface showing a document titled "sample\_markdown.Rmd". The code in the editor is:

```
1 ---  
2 title: "Sample Markdown"  
3 author: "Patrick Mathias"  
4 date: "4/18/2019"  
5 output: html_document  
6 ---  
7
```

Annotations with callouts point to specific parts of the code:

- A callout points to the first three lines (the YAML header) with the text "Starts and ends with 3 dashes".
- A callout points to the "output: html\_document" line with the text "Output format".
- A callout points to the "title: "Sample Markdown"" line with the text "Field name: Data".

YAML format  
Field name: Data

Output format

# Text

```
11  
12 - ## R Markdown  
13  
14 This is an R Markdown document. Markdown is a simple formatting syntax for  
authoring HTML, PDF, and MS Word documents. For more details on using R  
Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 When you click the **Knit** button a document will be generated that  
includes both content as well as the output of any embedded R code chunks  
.....
```

# for headers  
# for level 1  
## for level 2

Include hyperlinks  
with <>

- 1 asterisk for *italics* (\*italics\*)
- 2 asterisks for **bold** (\*\*bold\*\*)
- Hyphens (-bullet 1) for bullet points

# R Markdown cheatsheet

## syntax

```
Plain text  
End a line with two spaces to start a new paragraph.  
*italics* and _italics_  
**bold** and __bold__  
superscript^2^  
~~strikethrough~~  
[link](www.rstudio.com)  
  
# Header 1  
## Header 2  
### Header 3  
#### Header 4  
##### Header 5  
##### Header 6  
  
endash: --  
emdash: ---  
ellipsis: ...  
inline equation: $A = \pi \cdot r^2$  
image:   
  
horizontal rule (or slide break):  
***
```

## becomes

Plain text  
End a line with two spaces to start a new paragraph.  
italics and *italics*  
bold and **bold**  
superscript<sup>2</sup>  
strikethrough  
link

**Header 1**  
**Header 2**  
**Header 3**  
**Header 4**  
**Header 5**  
**Header 6**  
  
endash: –  
emdash: —  
ellipsis: ...  
inline equation:  $A = \pi \cdot r^2$   
  
image: 

horizontal rule (or slide break):

- > block quote
  - \* unordered list
    - \* item 2
      - + sub-item 1
      - + sub-item 2
  - 1. ordered list
    - 2. item 2
      - + sub-item 1
      - + sub-item 2

| Table Header | Second Header |
|--------------|---------------|
| Table Cell   | Cell 2        |
| Cell 3       | Cell 4        |

| Table Header | Second Header |
|--------------|---------------|
| Table Cell   | Cell 2        |
| Cell 3       | Cell 4        |

# Your Turn #2

1. Within your R Markdown document, add another level 2 heading at the end called “Document Info”
2. Under your new heading add a level 3 heading called “Author Info”
3. Add your name in bold under the new heading
4. Add your job title in italics
5. Knit the document



# Code chunks

Open/close with  
3 backticks

Language

Chunk  
name

Run chunk

Code in body of  
chunk

```
17
18  ``{r cars}
19  summary(cars)
20  ...
21
```



# Why name Code Chunks?

The screenshot shows an RStudio interface with a code editor and a terminal. In the code editor, several lines of R code are visible, including a code chunk starting with `## Including Plots` and another one for `cars` data. A tooltip is displayed over the code, listing various chunk types: Sample Markdown, R Markdown, Including Plots, and others. A blue speech bubble points from the left towards the tooltip, containing the text "Jump between chunks".

```
16 When you click the **Knit** button a document includes both content as well as the output of within the document. You can embed an R code c
17
18 - ``{r cars}
19 summary(cars)
20 ```
21
22 - ## Including Plots
23
24 Yo Sample Markdown s, for example:
25 Chunk 1: setup
26 - ``
27 pl R Markdown FALSE}
28 `` Chunk 2: cars
29
30 No Including Plots
31 No Chunk 3: pressure FALSE` parameter was ad
32 # Sample Markdown
```

Console Terminal × Jobs ×  
/cloud/project/ ↗

R version 3.5.2 (2018-12-20) -- "Faasheil Taloa"

Jump between  
chunks

# “Setup” Chunk

chunk name  
(optional)

“chunk option”  
don't show code in rendered document

```
6 + ```{r setup, include=FALSE}
7 library(tidyverse)
8 library(lubridate)
9 ...````
```

for dealing  
with dates

# Your Turn #3

The screenshot shows the RStudio interface with an R Markdown document open. The code window contains the following text:

```
16 When you click the **Knit** button a doc  
both content as well as the output of an  
document. You can embed an R code chunk  
17  
18 ```{r cars}  
summary(cars)  
```  
19  
20  
21  
22 ## Including Plots  
23  
24 You can also embed plots, for example:  
25  
26 ```{r pressure, echo=FALSE}  
plot(pressure)  
```  
27  
28  
29  
30 Note that the `echo = FALSE` parameter was added to the  
printing of the R code that generated the plot.  
31  
32  
33
```

A context menu is open over the code chunk starting at line 18, with the "Insert" option selected. A submenu is displayed, showing options for R, Bash, D3, Python, Rcpp, SQL, and Stan.

1. Insert a code chunk into white space within your open R Markdown document using Insert button on top right of code window
2. Add the following to your new code chunk:  
`mean(c(10, 20, 30))`
3. Execute code chunk by pressing Run button on top right of code chunk
4. Include the following within any text area (white space) and knit:  
``r mean(c(10, 20, 30))``



# Keyboard Shortcuts

Insert a code chunk into white space within your open R Markdown document using:

- Windows: CTRL+ALT+i
- Mac: COMMAND+OPTION+I

Execute using shortcuts:

- Windows: CTRL+SHIFT+ENTER
- Mac: COMMAND+SHIFT+ENTER

# Importing Files

# Working with R Markdown for this course

Each lesson has an R Markdown file

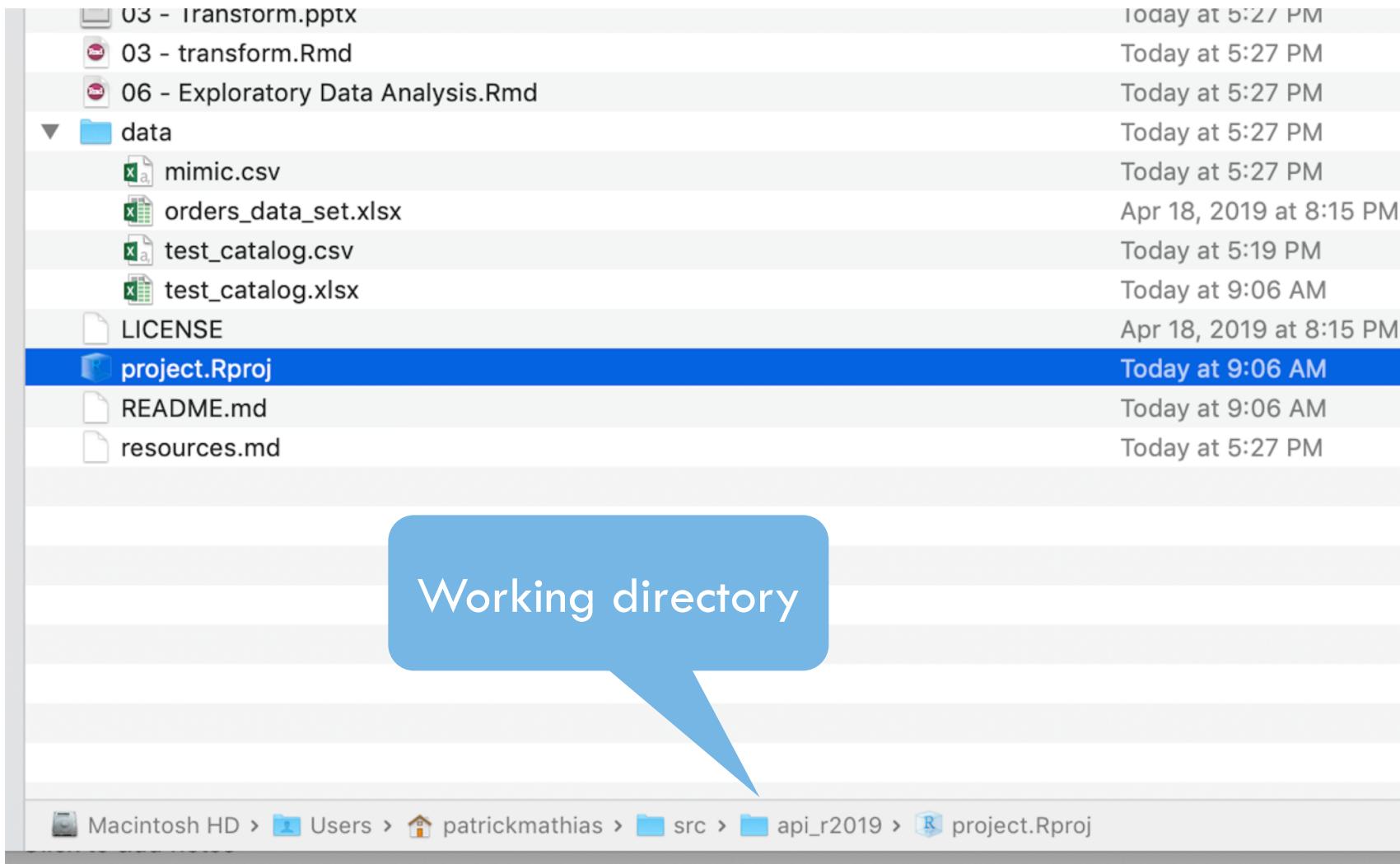
- Executable examples
- Exercises

Files used as “notebooks”: can document, execute, and iterate

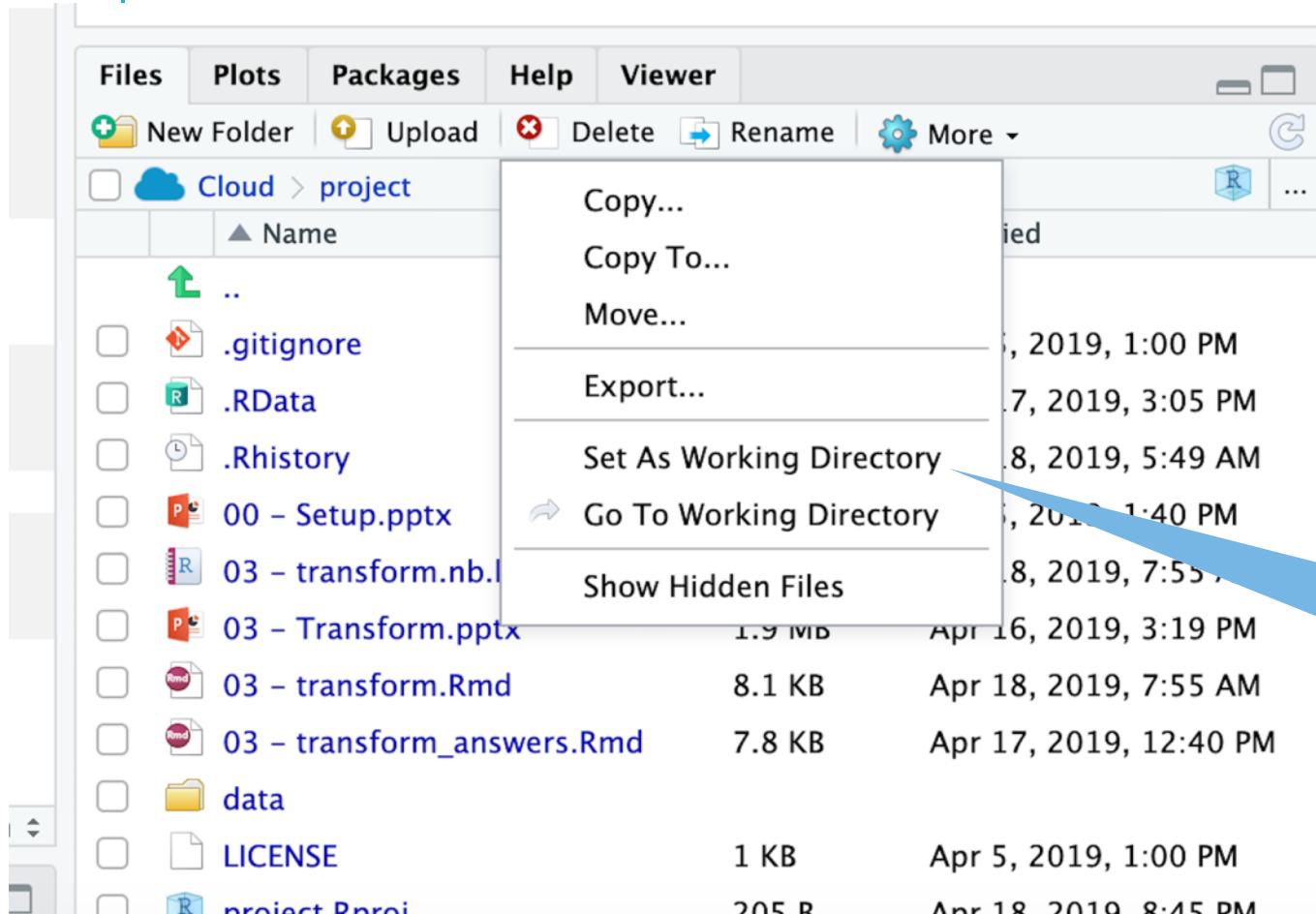
Best practice: work from notebook rather than console

Open “02 – Report.Rmd” and run the setup chunk.

# Working with Your file system



# Where am I? Your working directory



Navigate folder and file structure on your computer from Rstudio

getwd() function will tell you which folder you're in

...  
new working  
Navigate folders  
and set working  
directory



# Reading comma separated or tab delimited files

```
read_csv("data/test_catalog.csv")
```

File type  
(`csv`, `tsv`,  
`delim` for  
non-  
standard  
delimiters)

File path (if  
file in a  
folder  
within  
working  
directory)

File name



# Reading Excel files

```
read_excel("data/excel_file.xlsx",  
          sheet = 1)
```

File  
name

Specify  
sheet by  
number or  
"name"

Can also extract arbitrary  
rows and columns using  
"range = " argument

# Your Turn #4

| order_id | patient_id | description            | proc_code | order_class_c_d<br>escr | lab_status_<br>c | lab_status_c_<br>descr | order_status_<br>c | order_status_<br>c_descr |
|----------|------------|------------------------|-----------|-------------------------|------------------|------------------------|--------------------|--------------------------|
| 19766    | 511388     | PROTHROMBIN TIME PRO   |           | Normal                  |                  |                        | 4                  | Canceled                 |
|          |            | BASIC METABOLIC        |           |                         |                  |                        |                    |                          |
| 88444    | 511388     | PANEL                  | BMP       | Normal                  |                  |                        | 4                  | Canceled                 |
|          |            | THYROID<br>STIMULATING |           |                         |                  |                        |                    |                          |
| 40477    | 508061     | HORMONE                | TSH       | Normal                  |                  | 3 Final result         | 5                  | Completed                |
| 97641    | 508061     | T4, FREE               | T4FR      | Normal                  |                  | 3 Final result         | 5                  | Completed                |
|          |            | COMPREHENSIVE          |           |                         |                  |                        |                    |                          |
| 99868    | 505646     | METABOLIC PANEL        | COMP      | Normal                  |                  | 3 Final result         | 5                  | Completed                |

1. Find and import the orders data set and store in an object called "orders"

2. View the data

3. Run the summary function on the data



# View a quick summary

```
order_status_c_descr reason_for_canc_c reason_for_canc_c_descr
Length:45002          Min.   : 1.0   Length:45002
Class  :character    1st Qu.: 11.0   Class  :character
Mode   :character    Median  : 11.0   Mode   :character
                  Mean    : 437.2
                  3rd Qu.:1178.0
                  Max.   :1178.0
                  NA's   :37794
order_time           result_time
Min.   :2017-08-13 11:59:00  Min.   :2017-06-15 00:00:00
1st Qu.:2017-09-05 11:16:00  1st Qu.:2017-09-07 12:51:00
Median  :2017-09-27 08:48:00  Median  :2017-09-29 14:06:30
Mean    :2017-09-27 09:39:30  Mean    :2017-09-30 17:11:17
3rd Qu.:2017-10-19 13:45:00  3rd Qu.:2017-10-23 18:39:30
Max.   :2017-11-11 19:49:00  Max.   :2017-12-29 07:37:00
NA's   :7152
```

summary( ) function outputs quick statistical summaries for numerical and timestamp fields

Provides limited data (only a count) for character fields

Will provide counts of different categories for factor fields (categorical)

# What is NA?

“Not available”

- Equivalent to NULL in some other languages

Represents missing values

# Tabulating Data

# Orders Data Set

- Data set of outpatient laboratory orders
- 45,000 rows x 17 columns
- Queried from Epic orders data (Clarity)
- Deidentified and time shifted
- Includes timestamps for lab order, result, and provider review

| Variable                | Description   |
|-------------------------|---|
| order_id                | Key for order   |
| patient_id              | Key for patient   |
| description             | Text description of lab test  |
| proc_code               | Procedure code for lab test   |
| order_class_c_descr     | Setting test is intended to be performed in (eg. Normal = regular blood draw) |
| lab_status_c            | Code for status of laboratory result  |
| lab_status_c_descr      | Status of laboratory result   |
| order_status_c          | Code for status of order  |
| order_status_c_descr    | Status of order   |
| reason_for_canc_c       | Code for cancellation reason (if applicable)                                  |
| reason_for_canc_c_descr | Cancellation reason (if applicable)   |
| order_time              | Timestamp for time of original test order                                     |
| result_time             | Timestamp for most recent result in the record                                |
| review_time             | Timestamp for provider acknowledgment of review of result                     |
| department              | Clinic associated with test order   |
| ordering_route          | Structure/menu in health record from which order was placed                   |
| pref_list_type          | Category of preference list (if applicable)                                   |



# Create a one variable table

Data  
frame

Row  
variable

```
tabyl(orders, department)
```



# Two variable tables can help answer simple questions quickly

Which clinic cancelled the highest number of lab orders?

Data  
frame

Row  
variable

```
tabyl(orders, department,  
order_status_c_desc)
```

Column  
variable

# Your Turn #5

Which 3 departments ordered the highest number of labs using “Provider Preference Lists”?

What happens if you add another variable to the tabyl() function (3 variables)?





# Prettifying Table Output

```
kable(tabyl(orders, department,  
pref_list_type))
```

| department               | Clinic Preference List | None | Provider Preference List |
|--------------------------|------------------------|------|--------------------------|
| BEHAVIORAL HEALTH CLINIC | 40                     | 463  | 0                        |
| CARDIOLOGY CLINIC        | 888                    | 33   | 105                      |
| ENDOCRINOLOGY CLINIC     | 613                    | 94   | 779                      |
| FAMILY MEDICINE CLINIC   | 3135                   | 365  | 1                        |
| GASTROENTEROLOGY CLINIC  | 618                    | 89   | 74                       |

# Basic Plotting



# To make **any** kind of graph:

1. Choose a “tidy”  
**data frame**

```
ggplot(data = data_frame) +  
  geom_function(mapping = aes(mappings))
```

2. Pick a “geom”  
function

3. Write aesthetic  
**mappings**

# Creating a Histogram – Orders Over Time

initialize a plot  
with `ggplot()`

data frame

+ sign  
(at end of line)

```
ggplot(data = orders) +  
  geom_histogram(mapping = aes(x = order_time))
```

type of graph

mappings inside  
`aes()` function

“aesthetic”  
mapping

# Creating a Histogram – controlling bins

initialize a plot  
with `ggplot()`

data frame

+ sign  
(at end of line)

mappings inside  
`aes()` function

```
ggplot(data = orders) +  
  geom_histogram(mapping = aes(x = order_time),  
                 binwidth = 24*60*60)
```

type of graph

adjust size of  
bins  
(time unit = sec)

“aesthetic”  
mapping

# Creating a Histogram - Counts

initialize a plot  
with `ggplot()`

data frame

+ sign  
(at end of line)

mappings inside  
`aes()` function

```
ggplot(data = orders) +  
  geom_histogram(mapping = aes(x = order_status_c_descr),  
                 stat = "count")
```

type of graph

statistical  
transformation

"aesthetic"  
mapping

# Your Turn #6

1. Plot the distribution of review times by day.
2. Plot the distribution of order reviews by week instead of by day.



# **Knitting Other Document Types**

# Multiple Output Formats are Available

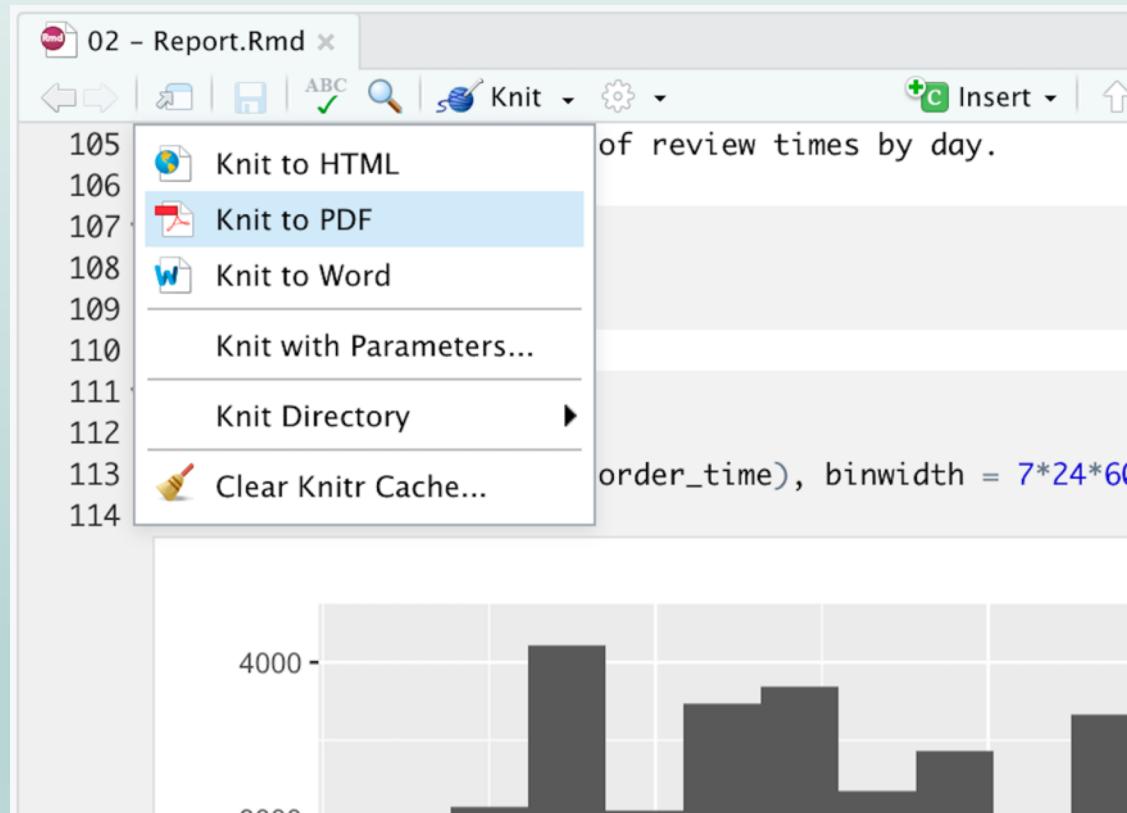
Pandoc universal document converter can create multiple document types:

- html
- pdf
- Docx

Can also create presentations and dashboards

- Including Powerpoint (most recent version of RStudio)

# Your Turn #7



Create a pdf of your lesson 2 markdown document using “Knit to pdf”



# Your Turn #7

Create a new R Markdown document but select “Presentation” instead of “Document” from menu

Select one of the HTML options and knit the document

