

# Reproducible Reports with Quarto

PSY 410: Data Science for Psychology

Dr. Sara Weston

2026-05-27

## Setup

## What is literate programming?

### Remember Session 1?

We opened this course with a striking finding: only 36% of psychology studies replicated.

...

Part of the problem was that analyses couldn't be reproduced — even by the original authors. Data was cleaned by hand. Numbers were copied into Word documents. Code only ran on one laptop.

...

**Quarto is part of the solution.** Today we learn to write documents where the analysis *is* the report.

## Copy-paste workflows break when anything changes

### Step 1: Analyze in R

```
# script.R
data <- read_csv("data.csv")
mean(data$depression)
# 18.5
```

**Step 2:** Copy to Word > “The mean depression score was 18.5 (SD = 5.2).”

## What goes wrong?

- Typos when copying numbers
- If data changes, you have to update manually
- No record of what code produced what result
- Can't reproduce your own analysis in 6 months

## Literate programming approach

**One document** contains:

- The code
- The results (automatically generated)
- The narrative (your interpretation)

**Benefits:**

- No copy-paste errors
- Data changes → results update automatically
- Complete record of what you did
- Easy to share with collaborators

## Introducing Quarto

**Quarto** is a publishing system that combines:

- **Markdown** for writing text
- **R code chunks** for analysis
- **Output formats** (HTML, PDF, Word)

. . .

It's the evolution of R Markdown (same company, newer technology).

## Anatomy of a Quarto document

### Three main components

1. **YAML header** — metadata (title, author, format)
2. **Markdown text** — your narrative
3. **Code chunks** — your R code

## YAML header

At the top of every `.qmd` file:

```
---
title: "My Analysis Report"
author: "Your Name"
date: "2026-05-27"
format: html
---
```

...

This controls:

- What appears at the top of the document
- What format to render (HTML, PDF, Word)
- Various options (table of contents, theme, etc.)

## Markdown basics

Markdown is a simple way to format text:

```
# Heading 1
## Heading 2
### Heading 3

**bold text**
*italic text*
```

```
- Bullet point
- Another bullet
```

```
1. Numbered list
2. Second item
```

```
[Link text](https://example.com)
```

## Markdown in action

You write:

```
**Depression** is a common  
mental health condition.  
  
Our research questions:  
  
1. Does CBT reduce depression?  
2. Do effects persist?
```

You get:

**Depression** is a common mental health condition.

Our research questions:

1. Does CBT reduce depression?
2. Do effects persist?

## Code chunks

Surround R code with three backticks:

```
```${r}  
# Load data  
data <- read_csv("depression_data.csv")  
  
# Compute mean  
mean(data$depression)  
```
```

...

When you render, both the code **and** output appear in the document.

## Running code chunks

In RStudio:

- **Run current chunk:** Click green arrow or Ctrl/Cmd + Shift + Enter
- **Run one line:** Ctrl/Cmd + Enter (like in scripts)
- **Run all chunks above:** Button in toolbar

## Rendering the document

**Render** = convert your .qmd to the final output (HTML/PDF/Word)

...

Click the **Render** button (or Ctrl/Cmd + Shift + K)

...

Quarto will:

1. Run all the R code
2. Capture the output
3. Combine with your text
4. Create the final document

## Code chunk options

### Controlling output

Add options with `#|` at the top of a chunk:

```
```{r}
#| echo: false
#| eval: true

# This code runs but doesn't show in the output
mean(depression_scores)
```
```

### Common chunk options

| Option                      | What it does                | When to use       |
|-----------------------------|-----------------------------|-------------------|
| <code>echo: false</code>    | Hide the code, show results | Final reports     |
| <code>eval: false</code>    | Show code, don't run it     | Example code      |
| <code>include: false</code> | Run code, hide everything   | Loading packages  |
| <code>message: false</code> | Hide messages               | Loading tidyverse |
| <code>warning: false</code> | Hide warnings               | Final reports     |

## Figure options

```
```{r}
#| fig-width: 8
#| fig-height: 6
#| fig-cap: "Depression scores by condition"

ggplot(data, aes(x = condition, y = depression)) +
  geom_boxplot()
```
```

- `fig-width` / `fig-height` — size in inches
- `fig-cap` — adds a caption below the figure

## Example: Setup chunk

Every document should start with a setup chunk:

```
```{r}
#| label: setup
#| include: false

library(tidyverse)
library(here)
```
```

- `include: false` means the chunk **runs** but nothing shows in the output
- Load all your packages here so they're available throughout

## Inline code

### Embedding results in text

Never hard-code numbers!

Bad:

```
The mean depression score was 18.5 (SD = 5.2).
```

...

**Good:**

```
The mean depression score was `r mean(data$depression)`  
(SD = `r sd(data$depression)`).
```

## Why inline code matters

```
# Create sample data  
depression_scores <- c(12, 18, 25, 22, 15, 20, 19, 21)
```

In your text, you write `r` expression and Quarto fills in the result.

## Inline code: before and after

**You write:**

```
The mean score was  
`r mean(depression_scores)`  
(SD = `r round(sd(depression_scores), 2)`).
```

**Quarto renders:**

The mean score was 19 (SD = 4.07).

## Formatting inline results

Use `round()` to control decimal places:

```
`r round(cor(x,y), 2)`
```

...

Use `scales::percent()` for percentages:

```
`r scales::percent(0.653)`
```

...

Use `scales::comma()` for large numbers:

```
`r scales::comma(15234)`
```

## Pair coding break

### Your turn: Create a mini-report

Create a new Quarto document (File → New File → Quarto Document):

1. Add a YAML header with your name and title
2. Write a short introduction (2-3 sentences)
3. Create a code chunk that loads `tidyverse` and a dataset (try `mpg`)
4. Create a visualization
5. Write a sentence with inline code reporting a summary statistic
6. Render to HTML

**Time: 10 minutes**

## Output formats

### HTML (default)

```
---  
title: "My Report"  
format: html  
---
```

#### Pros:

- Interactive (can include interactive plots)
- Easy to share (just send the file)
- Works on any device
- Can have code folding, tabs, etc.

#### Cons:

- Not suitable for print journals



## PDF

```
---  
title: "My Report"  
format: pdf  
---
```

### Pros:

- Print-ready
- Professional appearance
- Standard for journals

### Cons:

- Requires LaTeX installation
- Less flexible for complex layouts
- Not interactive

## Word

```
---  
title: "My Report"  
format: docx  
---
```

### Pros:

- Collaborators can edit with Track Changes
- Familiar to everyone
- Required by some journals

### Cons:

- Less control over formatting
- Can be finicky with complex tables/figures

## Multiple formats

You can render to multiple formats:

```

---
title: "My Report"
format:
  html:
    toc: true
  pdf:
    toc: true
  docx: default
---

```

...

Then choose which to render in RStudio dropdown.

## Tables

### Simple tables with kable()

```

therapy_summary <- tibble(
  Condition = c("Control", "CBT", "Mindfulness"),
  N = c(30, 30, 30),
  Mean = c(18.2, 12.4, 14.1),
  SD = c(5.1, 4.8, 5.3)
)

```

```
knitr::kable(therapy_summary)
```

| Condition   | N  | Mean | SD  |
|-------------|----|------|-----|
| Control     | 30 | 18.2 | 5.1 |
| CBT         | 30 | 12.4 | 4.8 |
| Mindfulness | 30 | 14.1 | 5.3 |

### Kable options

```
knitr::kable(
  therapy_summary,
  digits = 1,
  caption = "Depression scores by treatment condition",
  col.names = c("Condition", "N", "M", "SD")
)
```

Table 3: Depression scores by treatment condition

| Condition   | N  | M    | SD  |
|-------------|----|------|-----|
| Control     | 30 | 18.2 | 5.1 |
| CBT         | 30 | 12.4 | 4.8 |
| Mindfulness | 30 | 14.1 | 5.3 |

## Fancier tables: gt package

For publication-ready tables:

```
library(gt)

therapy_summary |>
  gt() |>
  tab_header(title = "Treatment Outcomes",
             subtitle = "Post-intervention BDI-II scores") |>
  fmt_number(columns = c(Mean, SD), decimals = 1)
```

gt is powerful but beyond our scope — use kable() for now.

## Workflow tips

### Best practices

1. **Name your chunks** — easier to find errors (`#| label: load-data`)
2. **One setup chunk at the top** — load all packages there
3. **Render often** — catch errors early

...

4. **Use relative paths** — `here::here("data/file.csv")`
5. **Cache expensive computations** — `#| cache: true`

## Structuring a psychology report

```
## Introduction
- Background
- Research question

## Method
- Participants
- Measures
- Procedure

## Results
- Descriptive statistics
- Main analyses
- Figures

## Discussion
- Interpretation
- Limitations
```

## Project organization

```
project/
  my-analysis.qmd
  data/
    survey_data.csv
  scripts/
    helper_functions.R
  output/
    my-analysis.html
    figures/

... .
```

Keep your .qmd file in the root, data in data/, outputs separate.

## End-of-deck exercise

### Create your final project draft

Convert your final project draft to a Quarto document (.qmd):

1. Structure it with sections: Introduction, Method, Results, Discussion
2. Include a setup chunk, data import code, and 2-3 captioned visualizations
3. Report at least one summary statistic with inline code
4. Add a table of descriptive statistics with `kable()`
5. Render to HTML and review

**This will become your final project report!**

## Wrapping up

### Why Quarto matters

**For you:**

- Never lose track of what code produced what result
- Update analysis when data changes
- Professional-looking reports with minimal effort

...

**For science:**

- Reproducibility — others can verify your work
- Transparency — full record of your analysis
- Sharing — collaborators can build on your work

### Quarto beyond this course

Quarto can create:

- **Books** — entire textbooks with chapters
- **Websites** — personal sites, lab sites
- **Presentations** — like these slides!
- **Dashboards** — interactive data apps
- **Journal articles** — many journals accept Quarto

...

You're learning a tool you'll use for years.

## Key takeaways

- **Quarto documents** = YAML + markdown + code chunks
- **Chunk options** (`echo`, `eval`, `include`) control what appears
- **Inline code** embeds results in text — never hard-code numbers!
- **Output formats** — HTML, PDF, Word from the same source
- **Tables** — `knitr::kable()` for simple, `gt` for fancy
- **Reproducibility** — future you will thank you

## Resources

- **R4DS Ch 28:** Quarto (comprehensive guide)
- **Quarto website:** [quarto.org](https://quarto.org)
- **RStudio Quarto tutorial:** [posit.co/blog/quarto-rstudio/](https://posit.co/blog/quarto-rstudio/)
- **Markdown guide:** [markdownguide.org](https://markdownguide.org)

## Before next class

### Read:

- R4DS Ch 8: Workflow: getting help

### Do:

- **Submit Assignment 8** — Create a complete reproducible report
- Continue working on your final project
- Practice rendering to different formats

## The one thing to remember

Quarto means your analysis and your report are the same document. Change one, the other updates. That's reproducibility.

See you Monday for practice & review!