Tutorial 01

ENVX2001 – Applied Statistical Methods

Semester 1

Welcome

In this tutorial you will create a **Quarto document** from scratch. By the end you will have an HTML page with formatted text, a chart, and a table — all inside an RStudio Project.

You will learn how to:

1. Create an RStudio Project and organise your files.
2. Create a Quarto document and set up the YAML header.
3. Write formatted content with headings, lists, and emphasis.
4. Load data, plot a chart, and display a table.

You will work with **one** of these datasets throughout the tutorial. Pick whichever you like and download the CSV — you will need it shortly.

Dataset	File	Download
Studio Ghibli films	`ghibli_films.csv`	Download
Spotify top songs	`spotify_top50.csv`	Download
AFL teams	`afl_teams.csv`	Download
Fast food chains	`fast_food.csv`	Download

RStudio Projects

RStudio Projects keep all your files in one place and set your working directory automatically. This means you can use relative paths (like `data/myfile.csv`) instead of full file paths, making your code portable across computers.

Exercise 1: Set up your project

Follow along as the tutor demonstrates each step.

1. Open RStudio.
2. Go to **File > New Project… > New Directory > New Project**.
3. Name the project something like `tutorial01` and choose where to save it.
4. Inside the project, create a folder called `data`. You can do this in the **Files** pane (bottom-right) by clicking **New Folder**.
5. Move your chosen CSV file into the `data` folder.

If you are not sure how to move files, open your system's file manager (Finder on Mac, File Explorer on Windows) and drag the CSV into the `data` folder you just created.

> **ℹ Checkpoint**
>
> You should have an RStudio Project open with a `data/` folder containing your chosen CSV file. Check the **Files** pane to confirm.

Quarto Basics

Quarto documents combine text and code into a single file that renders to HTML, PDF, or Word. The document starts with a **YAML header** (the block between `---` markers) that controls the title, author, and output settings.

Exercise 2: Create your document

1. Go to **File > New File > Quarto Document…** and click **Create Empty Document**.
2. Copy the following YAML header into the top of your file:

```
---
title: "Exploring My Dataset"
author: "Your Name"
date: today
format:
  html:
    theme: cosmo
    toc: true
---
```

3. Save the file as `report.qmd` in your project folder.
4. Render the document with **Cmd+Shift+K** (Mac) or **Ctrl+Shift+K** (Windows/Linux).

Other themes worth trying later: `flatly`, `journal`, `darkly`. The full list is at quarto.org/docs/output-formats/html-themes.html.

> **ℹ Checkpoint**
>
> You should see a rendered HTML page with your title, your name, today's date, and an empty body. If you get an error, check that the YAML is indented exactly as shown above.

Writing in Quarto

The rest of the tutorial builds up the body of your document. Each exercise adds a new section. Render after each one to see your progress.

Exercise 3: Write about your data

Add some text below the YAML header. Write a short commentary on your dataset — pick out some items that stand out to you and explain why. Use the markdown formatting shown below as a guide.

```
## About the data

### My top picks

**[Item]** stands out because...

### Also worth a look

A few more that caught my eye:

- [Item 1] --- reason
- [Item 2] --- reason

### Surprises

*[Item]* was lower than I expected...
```

This section uses several markdown formatting features:

- ## and ### create section headings at different levels.
- **text** makes text **bold**.
- *text* makes text *italic*.
- Lines starting with - create bullet lists.

Render the document to check that your headings, bold text, and lists appear correctly.

> **i** Checkpoint
>
> Your rendered document should show formatted text with proper headings, bold and italic styling, and a bullet list. The table of contents on the right should list your new sections.

Exercise 4: Load and display your data

Now let's bring in the dataset. Add a **setup chunk** first, then a chunk to load and display the data. Replace the file name with whichever CSV you chose.

```
CODE
```

```
```{r}
#| label: setup
#| message: false
#| warning: false

library(tidyverse)
```
```

```
CODE
```{r}
#| label: data

my_data ← read_csv("data/ghibli_films.csv")
my_data
```
```

A few things to note:

- The #| lines are **chunk options**. They control how the chunk behaves when you render.
- `message: false` and `warning: false` stop package loading messages from cluttering your output.
- `read_csv()` reads your CSV file into R. The path is relative to your project folder.

If you see an error like "file does not exist", double-check that your CSV is inside the `data/` folder and that the file name matches exactly (including capitalisation).

> **ℹ Checkpoint**
>
> After rendering, you should see the tidyverse loading code followed by your data printed as a table. Every row from your CSV should be visible.

Exercise 5: Create a chart

Add a code chunk that creates a horizontal bar chart of your data. The example below uses the Ghibli dataset — adjust the column names to match your chosen dataset.

```
CODE
```{r}
#| label: fig-chart
#| fig-cap: "A quick look at the data"
#| fig-width: 8
#| fig-height: 5

ggplot(my_data, aes(x = reorder(title, rating), y = rating)) +
  geom_col() +
  coord_flip() +
  labs(x = NULL, y = "Rating")
```
```

What's happening here:

- `reorder(title, rating)` sorts the bars so the highest-rated item is at the top.
- `coord_flip()` turns the chart on its side, making the labels easier to read.

- `fig-cap` adds a caption below the figure.
- `fig-width` and `fig-height` control the plot dimensions in inches.

If you are not using the Ghibli dataset, you will also need to update the `labs()` line to match your axis label (e.g. `y = "Streams (millions)"` for Spotify, `y = "Premierships"` for AFL).

Column names for each dataset:

- Ghibli: `title`, `rating`
- Spotify: `title`, `streams_millions`
- AFL: `team`, `premierships`
- Fast food: `name`, `rating`

> ℹ **Checkpoint**
>
> You should see a horizontal bar chart with items sorted from lowest to highest, plus a caption underneath. If the bars are unsorted, check that you used `reorder()` correctly.

Exercise 6: Add a formatted table

Finally, add a clean summary table. This time we will hide the code so the reader only sees the output — just like you would in a real report.

```
CODE
```{r}
#| label: tbl-summary
#| tbl-cap: "Dataset overview"
#| echo: false

knitr::kable(my_data)
```
```

The new option here is `echo: false`, which hides the code chunk from the rendered output. The reader sees the table but not the R code that produced it. This is how you would typically present results in a report or assignment.

Try going back to your chart chunk (Exercise 5) and adding `echo: false` there too, then re-render. Notice how the document looks cleaner without the code blocks.

> ℹ **Checkpoint**
>
> Your document should now include a formatted table with a caption and no visible code. If you also hid the chart code, the document should read like a finished report.

Putting it all together

Take a moment to look at what you have. Starting from an empty project, you have created a Quarto document that includes:

- A YAML header with a title, author, date, and output settings.
- Written prose with headings, bold, italic, and lists.
- A code chunk that loads and displays a dataset.
- A chart with a caption.
- A formatted table with hidden code.

This is the same workflow you will use for labs and assignments throughout the semester: set up a project, write a document, bring in data, and present your results.

Other output formats

So far you have been rendering to HTML. Quarto can also produce Word documents and PDFs from the same `.qmd` file — your tutor will demonstrate this now.

To render a Word document, change the `format` section of your YAML header:

```
CODE
format:
  docx:
    toc: true
```

To render a PDF (using Typst), change it to:

```
CODE
format:
  typst:
    toc: true
```

Render after each change and compare the output. The same content works across all three formats — you only change the YAML.

If you want to go back to HTML afterwards, change `format` back to the original `html` block from Exercise 2.

> **i** Checkpoint
>
> You should have seen your document rendered as both a Word file and a PDF. The content is the same — only the output format changed.

This tutorial page itself is a Quarto document. You can view its source code by clicking the **Code** dropdown (top-right of the page) and selecting **View Source**. Everything you see here — the text, the layout, the formatting — is written in a `.qmd` file, just like the one you created today.

If you want to keep exploring, the Quarto documentation has more on output formats, cross-references, and interactive features.