Git and GitHub

For the following questions, consider the file system below. You begin working in a shell at the Home directory, `~`.

```
~
├── practice
│   ├── ex1.R
│   └── ex2.R
└── my-project
    ├── README.md
    └── scripts
        ├── data-cleaning.R
        └── modeling.R
```

1. If you run the `ls` command, what will the output be?

2. What commands would you run to move into `my-project` and designate that directory as a git repository?

3. When you designate an existing directory as a git repo and it already has files in it, they will be neither staged nor committed. What single command could you use to stage the existing three files? Note that you can add multiple files at once by separating them by a space (see the wedding example)[1].

4. Once all of the files are staged, write the command can be used to take a snapshot of the state of those files along with a helpful caption.

5. Say that the project has a Part 1 and a Part 2. These existing files accomplish Part 1. To accomplish Part 2, you add several dozen lines of new code to `data-cleaning.R` and write and save a new script called `visualization.R` right alongside it. What commands would you run to store those changes along with a helpful description?

[1] There are several other ways to do this. Feel free to check online for different methods.

Data Wrangling II

For this problem set, you'll be working with the `storms` tibble built into `dplyr`. This is a larger data frame than previous, so you may need to inspect it more carefully in R before proceeding with these questions. Please answer the question in common english and provide the code needed to produce the output used in your answer.

On the *challenging* exercises, a helpful approach is to start by sketching the data frame that would answer the question. What are the dimensions? What does each row refer to? Which columns are needed? Then piece together your pipeline to lead from `storms` to that data frame.

```
library(dplyr)
storms
```

```
# A tibble: 20,778 × 13
   name   year month   day  hour   lat  long status      category  wind pressure
   <chr> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <fct>          <dbl> <int>    <int>
 1 Amy    1975     6    27     0  27.5 -79   tropical d…       NA    25     1013
 2 Amy    1975     6    27     6  28.5 -79   tropical d…       NA    25     1013
 3 Amy    1975     6    27    12  29.5 -79   tropical d…       NA    25     1013
 4 Amy    1975     6    27    18  30.5 -79   tropical d…       NA    25     1013
 5 Amy    1975     6    28     0  31.5 -78.8 tropical d…       NA    25     1012
 6 Amy    1975     6    28     6  32.4 -78.7 tropical d…       NA    25     1012
 7 Amy    1975     6    28    12  33.3 -78   tropical d…       NA    25     1011
 8 Amy    1975     6    28    18  34   -77   tropical d…       NA    30     1006
 9 Amy    1975     6    29     0  34.4 -75.8 tropical s…       NA    35     1004
10 Amy    1975     6    29     6  34   -74.8 tropical s…       NA    40     1002
# ℹ 20,768 more rows
# ℹ 2 more variables: tropicalstorm_force_diameter <int>,
#   hurricane_force_diameter <int>
```

6. What are the dimensions of the data frame? What does each row correspond to (i.e. what is the unit of observation)?

7. Calculate the average wind speed and pressure for each category of storm.

8. Find the number of observations and the maximum wind speed for each combination of storm status and category. Sort the results by maximum wind speed in descending order[2].

9. Use the `.by` shortcut to calculate the average pressure for each year. Repeat the calculation using `group_by()`. Do you get the same answer?

10. For each storm name, calculate the z-score of wind speed (standardized within each storm) and store it in a new column called `wind_z`. Return the data frame that includes this new column.

11. *Challenging*. For each storm category, arrange the storms in descending order by their maximum wind speed. Retain only the top three storms in each category[3].

[2]Hint 1: `n()` is a function that returns the row count when used inside `summarize()`. Hint 2: You can group by more than one variable.

[3]Hint: If you group by multiple variables, a call to `summarize()` will peel off the outermost grouping (the last argument in `group_by()`) and keep the others

12. *Challenging*: Create a summary that shows, for each year: the number of unique storms, the average duration (in hours) of storms, and the proportion of observations that are category 4 or 5 storms.

13. You may have noticed that your results for the answers in this problem set contain a value of `NA`. By default, most statistical summaries in R (`mean()`, `max()`, etc.) with return `NA` if any of the values are `NA`. If you prefer, you can calculate the mean after dropping the `NA` observations. One way to do this is to use the `na.rm = TRUE` option for functions that allow it (see `?mean`). The other way is to use the `drop_na()` function in `dplyr` (see `?drop_na()`).

    Revisit your code above and, if you haven't already, drop the `NA` values in a column before calculating summary statistics.

Quarto

The following questions address the quarto document below. Consult the Quarto Guide and the Quarto Reference for the authoritative documentation.

````
---
title: Interstellar Analysis
author: Murphy Cooper
format: typst
---

## Introduction

Readings from the radio telescope indicate unusual activity in the gamma-Q sector.


```{r}
#| echo: false
ggplot(df, aes(x = hrs, y = freq)) +
  geom_line()
```
````

14. What is the effect `echo: false`?

15. Which line would you change if you wanted this document to render to an HTML file?

16. If you wanted to display the code but not actually run it, write option setting(s) that you would include at the top of the code cell.

17. A classmate copies this code cell into their R console, where it runs without complaint and makes the plot. But when they run `quarto render` on the document, they get the error `could not find function "ggplot"`. Why does the code run in the console but fail during rendering? What would you add to the document to fix it?

Practice with `hello-penguins.qmd`

These exercises are hands-on. Download `hello-penguins.qmd` from Ed (it's the same file we used in class), open its folder in Positron, and run `quarto preview hello-penguins.qmd` so the document re-renders every time you save. Look things up in the Quarto Guide and the Quarto Reference as you go. Many of these are features you'll want in your project reports (the website's search tool is also helpful).

Essentials

18. Add an `author` and a `date` to the metadata. Then change the HTML `theme` and add a table of contents.
19. The text in the Species section says the figure is a bar plot. Read the code and fix the description so it matches what the figure actually shows. While you're at it, give the section a more descriptive *header*.
20. The first code cell prints package startup *messages* into your report. Use *cell options* to hide both the messages and the code in that cell, while still running it.
21. Give the scatterplot a *caption* and change its width and height. Then give the cell a label so you can *cross-reference* the figure, and change the prose so it says something like "Figure 1 shows…" with the figure number generated automatically (e.g. `@fig-bills`).
22. The `gt` package is loaded but never used. Pipe the penguins table into `gt()` so it renders as a formatted table. Give it a caption and a label, and cross-reference it in the text the same way you did the figure.
23. Add a sentence to the Data section that reports how many penguins are in the dataset. Instead of typing the number, use *inline code* so the number is computed from the data when you render.

Showing off

24. Add `code-fold: true` to the HTML format options. What changes in the rendered document?
25. Wrap the figure and the table in a *tabset* so readers can click between two tabs, one labeled "Plot" and one labeled "Table".
26. Give the document a light theme and a dark theme so readers get a toggle switch in the upper right corner.
27. Install the `plotly` package, then wrap your scatterplot in `plotly::ggplotly()`. Hover over the points, then click and drag. What do you get in the HTML output that a static figure can't do?
28. Change the format to `revealjs`, then use `##` headers so the figure and the table each get their own slide. Change the transition style.

More questions coming soon . . .