# Exercise: Understanding Grouping and Summarizing Data Frames

## Overview

For this exercise, you will work in your `dataviz-exercises` project linked to your personal remote GitHub repo of the same name. You will practice manipulating data and interacting with your GitHub remote. Much coding for your team project will be done using `.R` scripts, and to practice creating clean and clear documented scripts, you should use the `starter_script.R` in the `src/` directory as a template for your work. Don't overwrite `starter_script.R` or you will no longer have a clean file for starting scripts.

You will use the `diamonds` data set from `{ggplot2}`. You will practice grouping, mutating, and summarizing data frames.

---

## Mise en place: Loading Libraries and Functions

Prepare your environment by loading the `{tidyverse}` ecosystem and source any relevant functions needed.

## Data

You would normally have data to read from a file but for this exercise, you will use the `ggplot2::diamonds` data set. Examine the structure of your data frame so that you know variables that you might summarize and the variables you might group by. Always review the variable type as this provides information for data interpretation and for the appropriateness of certain statistical procedures. Use `str()` or `tibble::glimpse()`.

Make a note of whether the data frame is *grouped* (e.g., contains a "grouping structure").

---

# Part 1: Mutating Variables Without Grouping

Take `ggplot2::diamonds` and *add a new variable* that represents the **mean** of any **numeric** variable. Ensure that you exclude `NA` values when computing the mean. Do not overwrite the data frame, a step that is important when testing code. If you make and an error when overwriting, you may need to start for the top.

*Question 1:* Using words (not numbers), describe what this mean represents.

---

# Part 2: Grouping and Mutating Data Frames

## Grouping Data Frames

Take the `ggplot2::diamonds` data frame and pipe it to a *grouping* variable (e.g., `cut`) and then pipe that returned data frame to examine its *structure*. Make note of how the tibble's grouping structure has changed.

## Grouping and then Mutating

Now group the data by the same variable and repeat the same mutate as you did in Part 1.

*Question 2:* Using words (not numbers), describe what this mean represents. How does this differ from earlier?

*Compare the values of the mean variable added to the data frame with and without the grouping. Is the calculated variable in one of the data frames more similar to what you expected when you mutated the variable? If so, which one?*

---

# Part 3: Grouping and Summarizing

Using that same grouping variable, rather than *add a new variable* to the data frame that represents the mean of the numeric variable, *summarize* the data frame. Code your function to exclude `NA` values. Do not overwrite the data frame.

*Question 3:* What does this summary show?

---

# Part 4: Reusing Grouped Objects

Take your code from Part 3 and now assign the returned data frame to an object. Name your object something meaningfully, like `mean_price_by_varx_summary` where `x` stands for your grouping variable.

Now, take that object and use it to calculate the overall mean of the mean prices.

*Question 4:* Is this returned data frame what you expected to see? Is there a single value or multiple values based on the grouping?

---

# Part 5: Reusing Saved Data Frames

Your project will involved cleaning data, saving cleaned versions for yourself or your team, summarizing data, and perhaps saving summarized versions of data for yourself or your team.

1. Save your data frame to `data/` as an `.Rds` file.

2. Read that `.Rds` file and examine it for a grouping.

*Question 5:* What do you notice about the grouping? Is the data frame grouped? Ungrouped? Same grouping? Different grouping?

---

# Part 6: Saving and Checking Grouped Structures

In the example above, you **grouped then summarized**. Now take `ggplot2::diamonds` and **group then mutate** the mean price based on the grouping. Check the grouping structure.

Add a new variable that represents the **overall mean price** as you did earlier with `summarize()`.

*Question 6:* Is this returned data frame what you expected to see? What are the similarities and differences between `summarize() + mutate()` and `mutate() + mutate()`?

---

# Part 7: Practicing with Git

Remember that your Git commands will be in your RStudio terminal and not the RStudio R console.

## Save your File

Save `.R` script with your work. Remember not to use spaced in names. A script is source code, so save it in a sensible place in `src/`. Make note of it's location and name.

## Checkout your Feature Branch

Your `dataviz-exercises` project is linked to your own personal repo on GitHub. You can work on the `main` branch or you can create a feature branch.

## Stage your File Change

Now that you know the name and location or your file for the exercise, stage/add your `.R` script so that you can commit it and push it to your remote.

Use `git add <path to file>`

## Commit Changes with a Message

Commit messages help you identify the file changes you make (e.g., "completed grouping exercise" or "fixed xyz plot").

Use `git commit -m "short message describing your work"`

## Push Changes

Push your commit to the remote on GitHub. You can also check your repo on your GitHub account to see your work.

Use `git push`

# Extra Practice

- Try grouping by two or three variables, summarizing, and then summarizing again. What happens? What grouping is preserved?
- Compare results with and without grouping to see how persistent grouping changes your results.