Lesson 2: Dataframe Solutions

Create a data frame solutions

This document contains the solutions for the create a data frame activity. You can use these solutions to check your work and ensure that your code is correct or troubleshoot your code if it is returning errors. If you haven't completed the activity yet, we suggest you go back and finish it before reading the solutions.

If you experience errors, remember that you can search the internet and the RStudio community for help: https://community.rstudio.com/#

Step 1: Load packages

Start by installing the required package; in this case, you will want to install `tidyverse`. If you have already installed and loaded `tidyverse` in this session, feel free to skip the code chunks in this step.

```
install.packages("tidyverse")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   4.0.0     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Step 2: Create data frame

Sometimes you will need to generate a data frame directly in `R`. There are a number of ways to do this; one of the most common is to create individual vectors of data and then combine them into a data frame using the `data.frame()` function.

Here's how this works. First, create a vector of names:

```
names <- c("Peter", "Jennifer", "Julie", "Alex")
```

Then create a vector of ages:

```
age <- c(15, 19, 21, 25)
```

With these two vectors, you can create a new data frame called `people`:

```
people <- data.frame(names, age)
```

Step 3: inspect the data frame

Now that you have this data frame, you can use some different functions to inspect it.

One common function you can use to preview the data is the `head()` function, which returns the columns and the first several rows of data. You can check out how the `head()` function works by running the chunk below:

```
head(people)
```

```
##      names age
## 1    Peter  15
## 2 Jennifer  19
## 3    Julie  21
## 4     Alex  25
```

In addition to `head()`, there are a number of other useful functions to summarize or preview your data. For example, the `str()` and `glimpse()` functions will both provide summaries of each column in your data arranged horizontally. You can check out these two functions in action by running the code chunks below:

```
str(people)
```

```
## 'data.frame':    4 obs. of  2 variables:
##  $ names: chr  "Peter" "Jennifer" "Julie" "Alex"
##  $ age  : num  15 19 21 25
```

```
glimpse(people)
```

```
## Rows: 4
## Columns: 2
## $ names <chr> "Peter", "Jennifer", "Julie", "Alex"
## $ age   <dbl> 15, 19, 21, 25
```

You can also use `colnames()` to get a list the column names in your data set. Run the code chunk below to check out this function:

```
colnames(people)
```

```
## [1] "names" "age"
```

Now that you have a data frame, you can work with it using all of the tools in `R`. For example, you could use `mutate()` if you wanted to create a new variable that would capture each person's age in twenty years. The code chunk below creates that new variable:

```
mutate(people, age_in_20 = age + 20)
```

```
##      names age age_in_20
## 1    Peter  15        35
## 2 Jennifer  19        39
## 3    Julie  21        41
## 4     Alex  25        45
```

Step 4: Try it yourself

To get more familiar with creating and using data frames, use the code chunks below to create your own custom data frame.

First, create a vector of any five different fruits. You can type directly into the code chunk below; just place your cursor in the box and click to type. Once you have input the fruits you want in your data frame, run the code chunk.

```
fruit <- c("Lemon", "Blueberry", "Grapefruit", "Mango", "Strawberry")
```

Now, create a new vector with a number representing your own personal rank for each fruit. Give a 1 to the fruit you like the most, and a 5 to the fruit you like the least. Remember, the scores need to be in the same order as the fruit above. So if your favorite fruit is last in the list above, the score `1` needs to be in the last position in the list below. Once you have input your rankings, run the code chunk.

```
rank <- c(4, 2, 5, 3, 1)
```

Finally, combine the two vectors into a data frame. You can call it `fruit_ranks`. Edit the code chunk below and run it to create your data frame.

```
fruit_ranks <- data.frame(fruit, rank)
```

After you run this code chunk, it will create a data frame with your fruits and rankings.