Introduction to Map Functions - Part 1

Data

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
1	Adelie	Torgersen	39.1	18.7	181	3750	male	2007
2	Adelie	Torgersen	39.5	17.4	186	3800	female	2007
3	Adelie	Torgersen	40.3	18.0	195	3250	female	2007
4	Adelie	Torgersen	*NA*	*NA*	*NA*	*NA*	*NA*	2007
5	Adelie	Torgersen	36.7	19.3	193	3450	female	2007
6	Adelie	Torgersen	39.3	20.6	190	3650	male	2007
7	Adelie	Torgersen	38.9	17.8	181	3625	female	2007
8	Adelie	Torgersen	39.2	19.6	195	4675	male	2007
9	Adelie	Torgersen	34.1	18.1	193	3475	*NA*	2007
10	Adelie	Torgersen	42.0	20.2	190	4250	*NA*	2007

Nesting

`nest()`: is a function used to group data into a nested structure.

Example 1: Nest the penguins dataset by sex

```
example_1 <- penguins %>%
  group_by(sex) %>%
  nest()
```

Output:

	sex	data
1	male	*<data.frame[168 x 7]>*
2	female	*<data.frame[165 x 7]>*
3	*NA*	*<data.frame[11 x 7]>*

Example 2: Unnest the dataset you created in Example 1.

```
example_2 <- example_1 %>%
  unnest(cols = c(data))
```

Output:

	sex	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
1	male	Adelie	Torgersen	39.1	18.7	181	3750	2007
2	male	Adelie	Torgersen	39.3	20.6	190	3650	2007
3	male	Adelie	Torgersen	39.2	19.6	195	4675	2007
4	male	Adelie	Torgersen	38.6	21.2	191	3800	2007
5	male	Adelie	Torgersen	34.6	21.1	198	4400	2007
6	male	Adelie	Torgersen	42.5	20.7	197	4500	2007
7	male	Adelie	Torgersen	46.0	21.5	194	4200	2007
8	male	Adelie	Biscoe	37.7	18.7	180	3600	2007
9	male	Adelie	Biscoe	38.2	18.1	185	3950	2007
10	male	Adelie	Biscoe	38.8	17.2	180	3800	2007
11	male	Adelie	Biscoe	40.6	18.6	183	3550	2007
12	male	Adelie	Biscoe	40.5	18.9	180	3950	2007

Pull

`pull()`: is a function that works like $. It calls a specific column of your dataframe.

Example 3: Consider the frequency table for species:

```
table(penguins$species)
```

```
   Adelie Chinstrap    Gentoo
      152        68       124
```

Make the same table using pull:

Map

The `map()` function is a powerful tool in the `tidyverse` package that allows you to apply a function to each element of a list or vector, and then returns a list containing the results. It's particularly useful when you have repetitive tasks to perform on multiple elements. This function is part of the `purrr` package within the `tidyverse`.

Example 4: Use the function you created with the values (without map)

```
square <- function(x) {
  y<-x^2
  return(y)
}
values <- c(2, 4, 6, 8)

example_4 <- square(values)
example_4
```

Output:

Example 5: Use map to apply the function to each element

Note: The result is a list with squared values. To get results back to a vector `unlist()`

```
example_5 <- map(values, square)
final_results <- unlist(example_5)
final_results
```

Output:

Combining Nesting and Mapping

Example 6: Combine `nest()` with `map()`, and compute the mean of bill_length_mm, the mean of body_mass_g, and the mean of bill_depth_mm for each value of the sex type.

```
example_6 <- penguins %>%
  group_by(sex) %>%
  nest() %>%
  mutate(mean_bill_length = map(data, ~ mean(pull(.x, bill_length_mm), na.rm = TRUE)),
         mean_body_mass = map(data, ~ mean(pull(.x, body_mass_g), na.rm = TRUE)),
         mean_bill_depth = map(data, ~ mean(pull(.x, bill_depth_mm), na.rm = TRUE)))
```

Output:

	sex	data	mean_bill_length	mean_body_mass	mean_bill_depth
1	male	*<data.frame[168 x 7]>*	45.85476	4545.685	17.89107
2	female	*<data.frame[165 x 7]>*	42.09697	3862.273	16.42545
3	*NA*	*<data.frame[11 x 7]>*	41.3	4005.556	16.64444