12 Reshaping

Long- and wide-format data

Wide-format data

year	Adelie	Chinstrap	Gentoo
2007	50	26	34
2008	50	18	46
2009	52	24	44

Long-format data

Long- and wide-format data

- The package `tidyr` (included in `tidyverse`) has two very useful function for reshaping data.
  1. `pivot_longer()`
  2. `pivot_wider()`

Wide format → Long format data

- `pivot_longer()` functio converts an wide format data to a long format data
- It is required to mention which columns (variables) should be combined into a single variable and it will return two new variables based on the column names and values of the selected columns
  - The first variable will contain the names of the selected columns
  - The second variable will contain the values of the selected columns

Wide format → Long format data

- The syntax of the function `pivot_longer()`
  - `data`
  - `cols` → selected variables
  - `names_to` → selected variable (column) names
  - `values_to` → A character vector specifying the new column to create from the information stored in `names_to` argument

Wide format → Long format data

- **Wide-format data**

```
wdat
```

```
# A tibble: 3 × 4
   year Adelie Chinstrap Gentoo
  <int>  <int>     <int>  <int>
1  2007     50        26     34
2  2008     50        18     46
3  2009     52        24     44
```

- **Long-format data**

```
wdat %>%
  pivot_longer(
    cols = Adelie:Gentoo,
    names_to = "species",
    values_to = "body_mass"
    )
```

```
# A tibble: 9 × 3
   year species   body_mass
  <int> <chr>         <int>
1  2007 Adelie           50
2  2007 Chinstrap        26
3  2007 Gentoo           34
4  2008 Adelie           50
5  2008 Chinstrap        18
6  2008 Gentoo           46
7  2009 Adelie           52
8  2009 Chinstrap        24
9  2009 Gentoo           44
```

Long-format → Wide-format

- `pivot_wider()` function converts a long-format data to an wide-format data
- It is required to mention which columns (variables) should be combined and it will create two new variables based on the column names and values of the selected columns
  - The first variable will contain the names of the selected columns
  - The second variable will contain the values of the selected columns

Long-format → Wide-format

- The syntax of the function `pivot_wider()`
  - `data`
  - `id_cols` → unique idetifier of a column
  - `names_from` → selected variable names
  - `values_from` →

Long-format → Wide-format

- **Long format data**

```
penguins %>%
  count(year, species)
```

```
# A tibble: 9 × 3
   year species       n
  <int> <fct>     <int>
1  2007 Adelie       50
2  2007 Chinstrap    26
3  2007 Gentoo       34
4  2008 Adelie       50
5  2008 Chinstrap    18
6  2008 Gentoo       46
7  2009 Adelie       52
8  2009 Chinstrap    24
9  2009 Gentoo       44
```

- **Wide format data**

```
penguins %>%
  count(year, species) %>%
  pivot_wider(
    names_from = species,
    values_from = n)
```

```
# A tibble: 3 × 4
   year Adelie Chinstrap Gentoo
  <int>  <int>     <int>  <int>
1  2007     50        26     34
2  2008     50        18     46
3  2009     52        24     44
```

Practice 2.1

1. Starting with penguins, find counts of observation by species, island and year.
2. Starting with penguins, filter to only keep Adelie and Gentoo penguins, then find counts by species and sex.
3. Add a new column to penguins called year that contains:
    - “Year 1” if the year is 2007
    - “Year 2” if the year is 2008
    - “Year 3” if the year is 2009

Practice 2.1

4. Starting with penguins, only keep observations for chinstrap penguins, then only keep the `flipper_length_mm` and `body_mass_g` variables.

- Add a new column called fm_ratio that contains the ratio of flipper length to body mass for each penguin.
- Next, add another column named `ratio_bin` which contains the word "high" if `fm_ratio` is greater than or equal to 0.05, "low" if the ratio is less than 0.05, and "no record" if anything else (e.g. NA).