# STAT 118: Notes D

**Pretty tables with `kableExtra`**


Emily Malcolm-White

```
#LOAD PACKAGES
library(tidyverse)
```

```
#LOAD DATA
library(palmerpenguins)
data(penguins)

#CLEAN UP DATA
penguins <- penguins %>%
  drop_na()
#sometimes this is appropriate. It's questionable here...
```

## mutate

The mutate function allows you create a new column which is a function of other columns. This can be useful to converting units.

For example, let's calculate create a new column which displays the body length weight in pounds (lbs) instead of grams. Recall: to convert from grams to pounds we need to multiply by 0.00220462

```
penguins <- penguins %>%
  mutate(body_mass_lbs = body_mass_g*0.00220462)
```


Figure 1: Artwork by @allisonhorst

This can also be useful for making new calculations based on existing data (for example, price and number of square feet could be used to calculate price per square foot).

## `case_when`

Case when can be used in combination with `mutate` to create a new column with a categorical variable conditional on the values in another column.


Figure 2: Artwork by @allisonhorst

For example:

```
penguins <- penguins %>%
  mutate(penguin_length_cat = case_when(bill_length_mm > 50 ~ 'whoa! huge bill!', TRUE ~ '
```

Tip

For those of you who have taken a computer science class before, you may notice that `case_when` is similar to using an `ifelse` statement. You can also use `ifelse` in R if you'd prefer!

```
penguins <- penguins %>%
  mutate(penguin_length_cat = ifelse(bill_length_mm > 50, 'whoa! huge bill!', '--' ))
```

## default printing style of a table

Let's consider our table from last class:

```
penguins %>%
  group_by(species) %>%
  summarise(average_bill_lenth = mean(bill_length_mm),
            average_bill_depth = mean(bill_depth_mm))
```

```
# A tibble: 3 x 3
  species   average_bill_lenth average_bill_depth
  <fct>                  <dbl>              <dbl>
1 Adelie                  38.8               18.3
2 Chinstrap               48.8               18.4
3 Gentoo                  47.6               15.0
```

When we knit this up it looks like of ugly...

## Using `kable` to get pretty tables

```
library(kableExtra)
```

```
penguins %>%
  group_by(species) %>%
  summarise(average_bill_length = mean(bill_length_mm),
            average_bill_depth = mean(bill_depth_mm)) %>%
  kbl()
```

Table 1: Average Bill Characteristics by Species

| Species | Average Bill Length | Average Bill Depth |
|---|---|---|
| Adelie | 38.82 | 18.35 |
| Chinstrap | 48.83 | 18.42 |
| Gentoo | 47.57 | 15.00 |

| species | average_bill_length | average_bill_depth |
|---|---|---|
| Adelie | 38.82397 | 18.34726 |
| Chinstrap | 48.83382 | 18.42059 |
| Gentoo | 47.56807 | 14.99664 |

```
#OR

table1 <- penguins %>%
  group_by(species) %>%
  summarise(average_bill_length = mean(bill_length_mm),
            average_bill_depth = mean(bill_depth_mm))

kbl(table1)
```

| species | average_bill_length | average_bill_depth |
|---|---|---|
| Adelie | 38.82397 | 18.34726 |
| Chinstrap | 48.83382 | 18.42059 |
| Gentoo | 47.56807 | 14.99664 |

## Options in kable

We customize the content so it's displaying the information more clearly.

```
table1 %>%
  kbl(col.names = c("Species", "Average Bill Length", "Average Bill Depth"),
    caption = "Average Bill Characteristics by Species",
    digits = 2)
```

Better...

Table 2: Average Bill Characteristics by Species

| Species | Average Bill Length | Average Bill Depth |
|---|---|---|
| Adelie | 38.82 | 18.35 |
| Chinstrap | 48.83 | 18.42 |
| Gentoo | 47.57 | 15.00 |

Table 3: Average Bill Characteristics by Species

| Species | Average Bill Length | Average Bill Depth |
|---|---|---|
| **Adelie** | 38.82 | 18.35 |
| **Chinstrap** | 48.83 | 18.42 |
| **Gentoo** | 47.57 | 15.00 |

## pretty styling

```
table1 %>%
  kbl(col.names = c("Species", "Average Bill Length", "Average Bill Depth"),
    caption = "Average Bill Characteristics by Species",
    digits = 2) %>%
  kable_styling()
```

Many options for customizing the look of the tables – more here: https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html

Let's try one...

Let's make each row's color correspond to

```
table1 %>%
  kbl(col.names = c("Species", "Average Bill Length", "Average Bill Depth"),
    caption = "Average Bill Characteristics by Species",
    digits = 2) %>%
  kable_paper() %>%
  column_spec(1, bold=T) %>%
  row_spec(2, color = "#c85bcc") %>%
  row_spec(3, color = "#067176") %>%
  row_spec(1, color = "#ff7501")
```

RStudio hosts a table contest every year!

- Here is a link to this year's contest https://www.rstudio.com/blog/rstudio-table-contest-2022/

- Here is a link to previous year's entries and winners to explore what is possible! https://community.rstudio.com/c/table-gallery/64

> **Tip**
>
> As a general rule, you should have the content of the table as you'd like it (the exact columns and rows you want) first and *then* you can make it pretty using the `kableExtra` package.