Data Visualization

1. How many rows are in `penguins`? How many columns?

```
dim(penguins)        # rows, columns
```

```
## [1] 344   8
```

```
nrow(penguins)       # rows only
```

```
## [1] 344
```

```
ncol(penguins)       # columns only
```

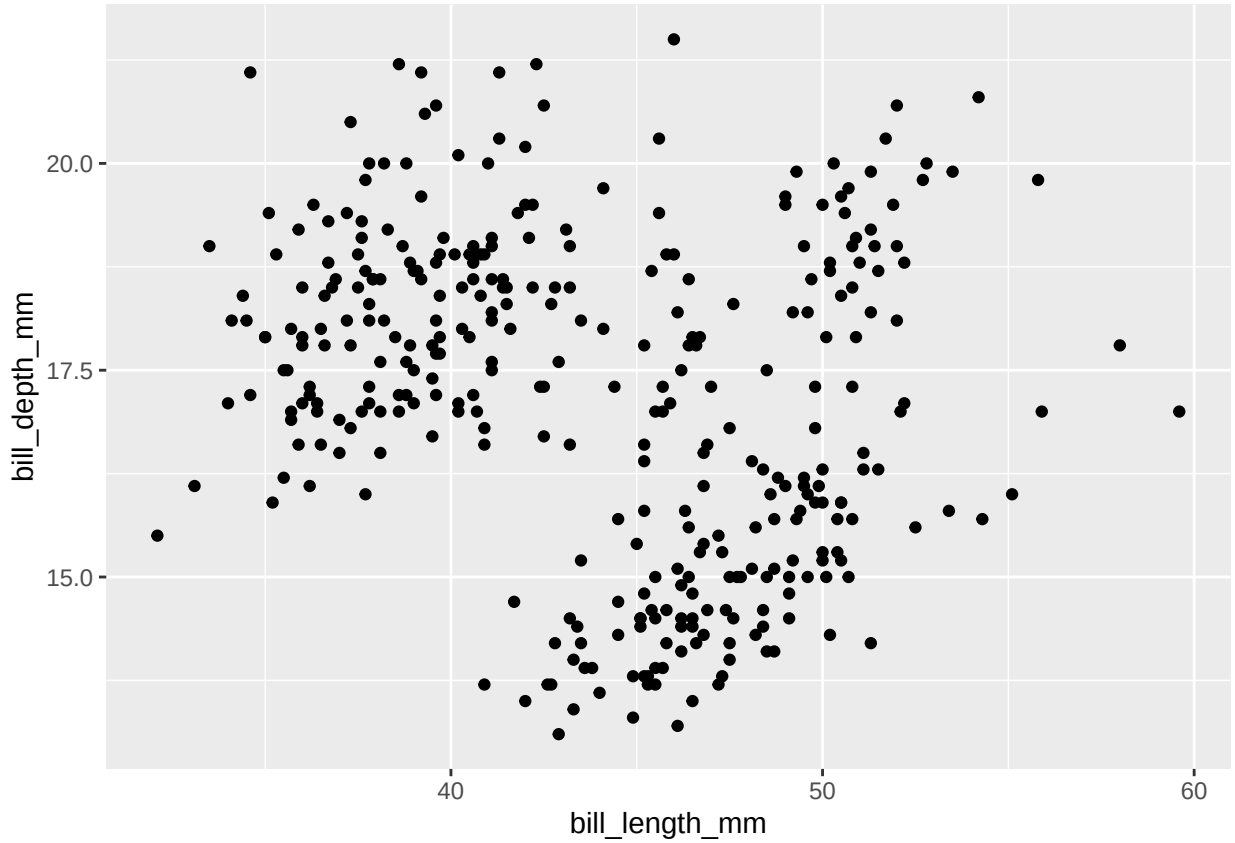
```
## [1] 8
```

2. What does the `bill_depth_mm` variable describe?

Answer: It is the penguin bill (beak) depth, measured in millimeters.

3. Scatterplot of `bill_depth_mm` (y) vs `bill_length_mm` (x). Describe relationship.

```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point(na.rm = TRUE)
```

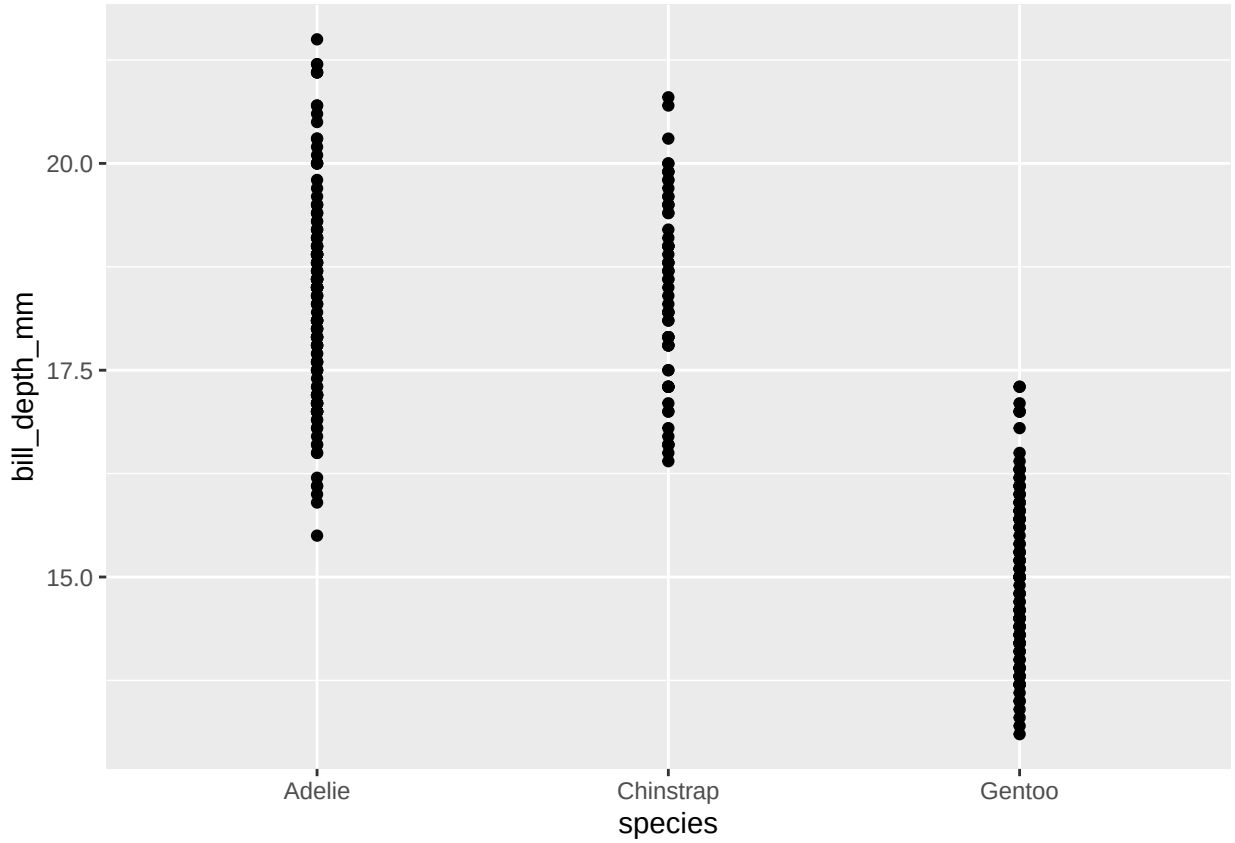


Answer: Overall it trends upward (longer bills often have deeper bills), but it also looks like there are different groups (species) mixed together.

4. What happens if you make a scatterplot of `species` vs. `bill_depth_mm`? Better geom?

```
ggplot(penguins, aes(x = species, y = bill_depth_mm)) +
  geom_point()
```

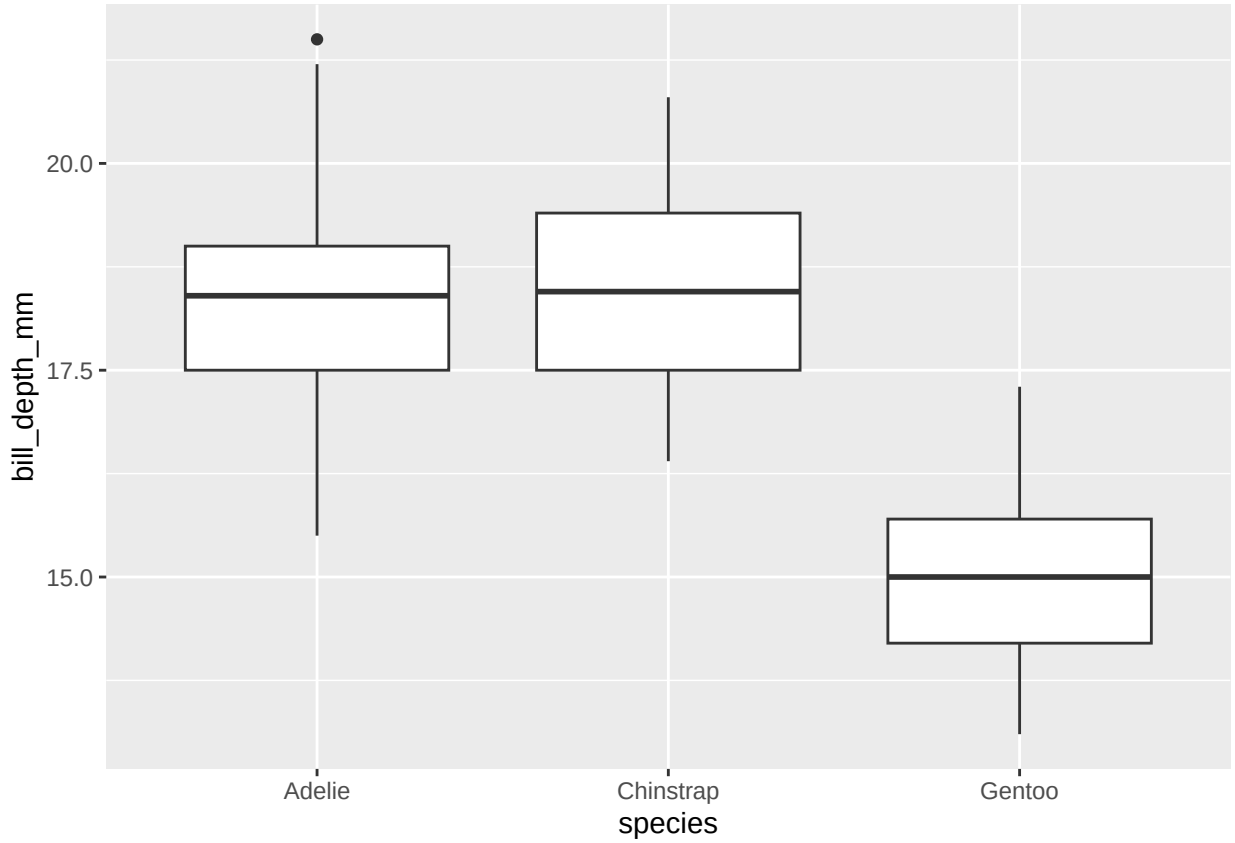
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Answer: Points stack on top of each other because `species` is categorical. A better choice is a boxplot (or violin) with optional jitter:

```
ggplot(penguins, aes(x = species, y = bill_depth_mm)) +
  geom_boxplot()
```

```
## Warning: Removed 2 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



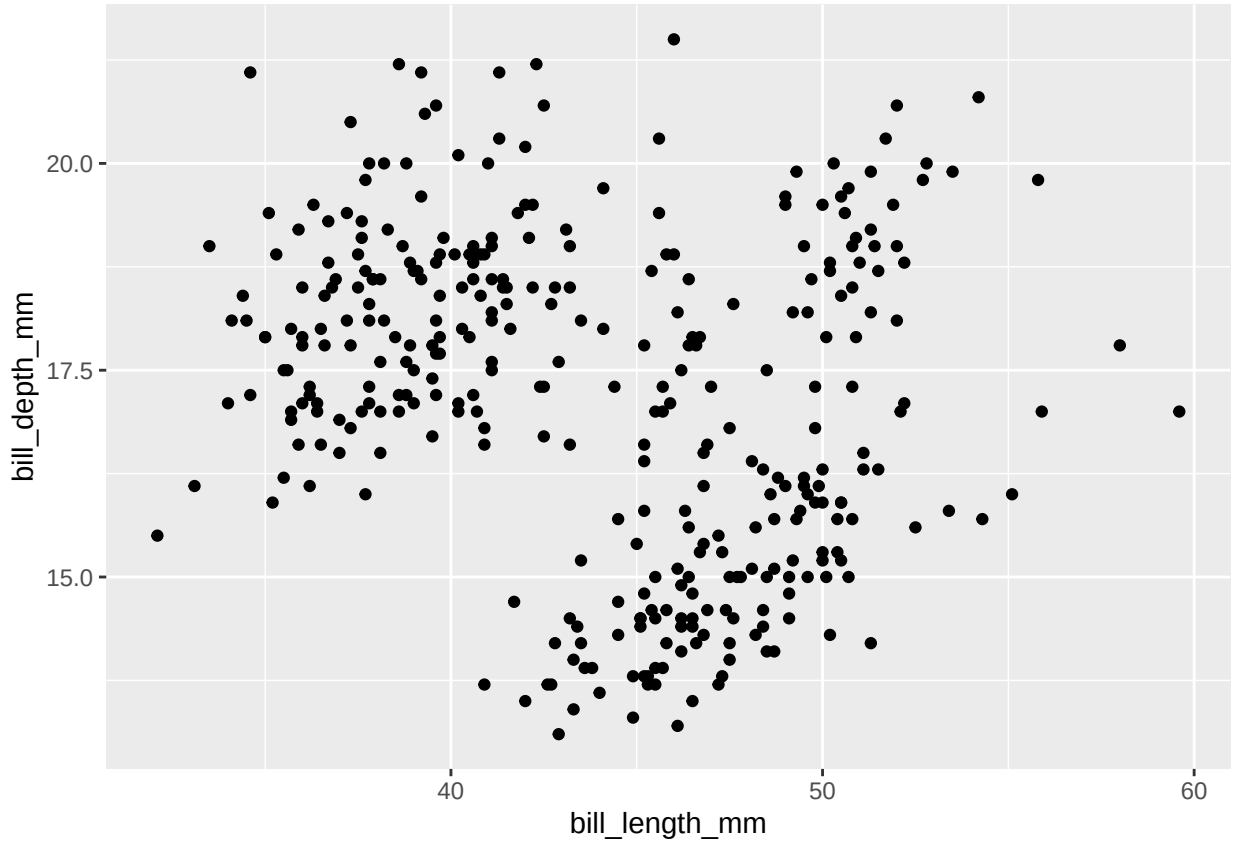
5. Why does this give an error and how to fix it?

```
ggplot(data = penguins) +
  geom_point()
```

Answer: `geom_point()` needs x and y mappings. Fix by adding `aes()`:

```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point()
```

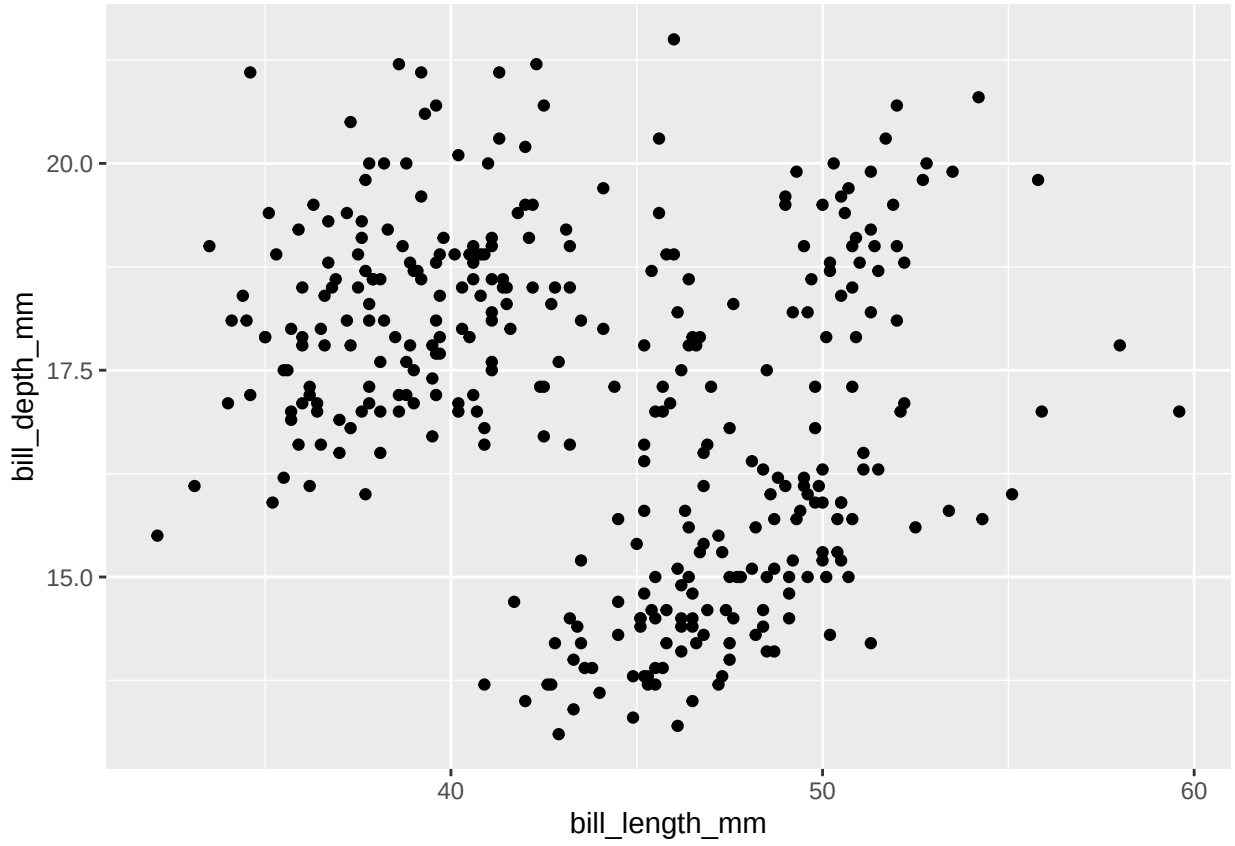
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



6. What does `na.rm` do in `geom_point()`? Default? Make a plot using `na.rm = TRUE`.

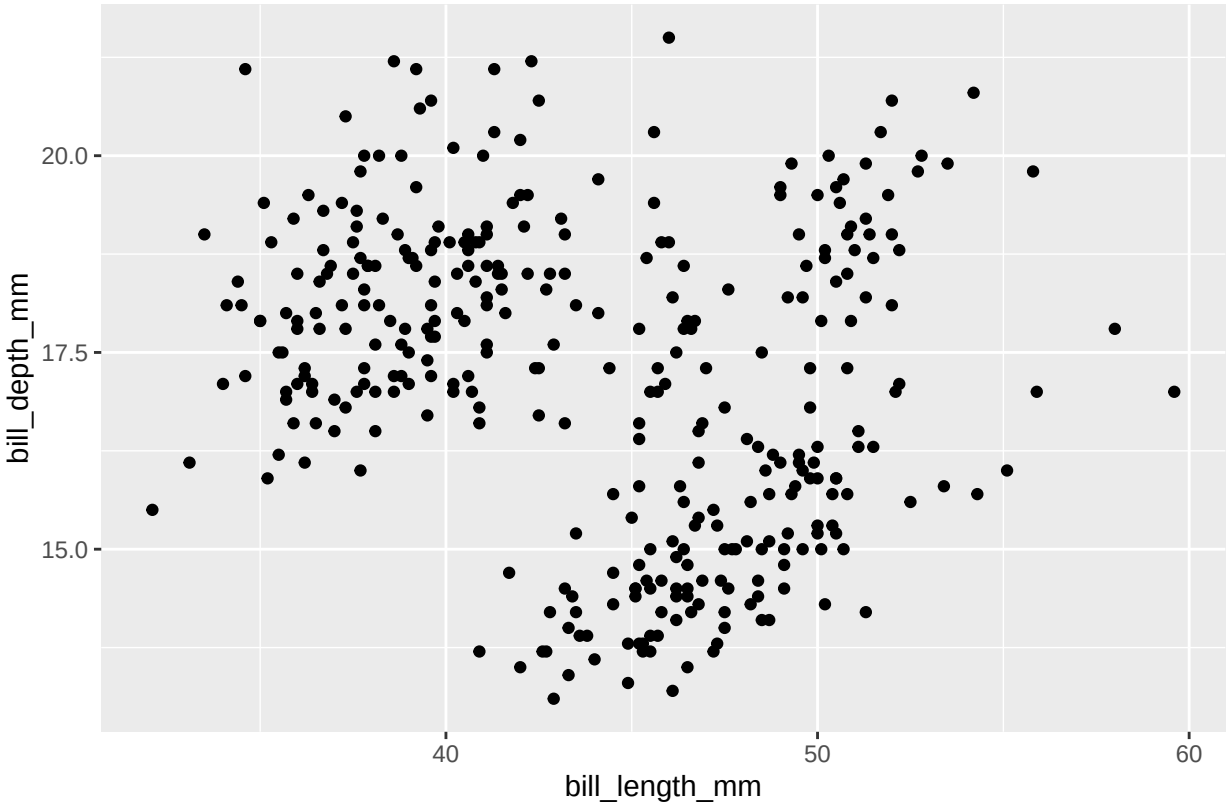
Answer: `na.rm` removes missing (NA) values instead of warning about them. Default is `FALSE`.

```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point(na.rm = TRUE)
```

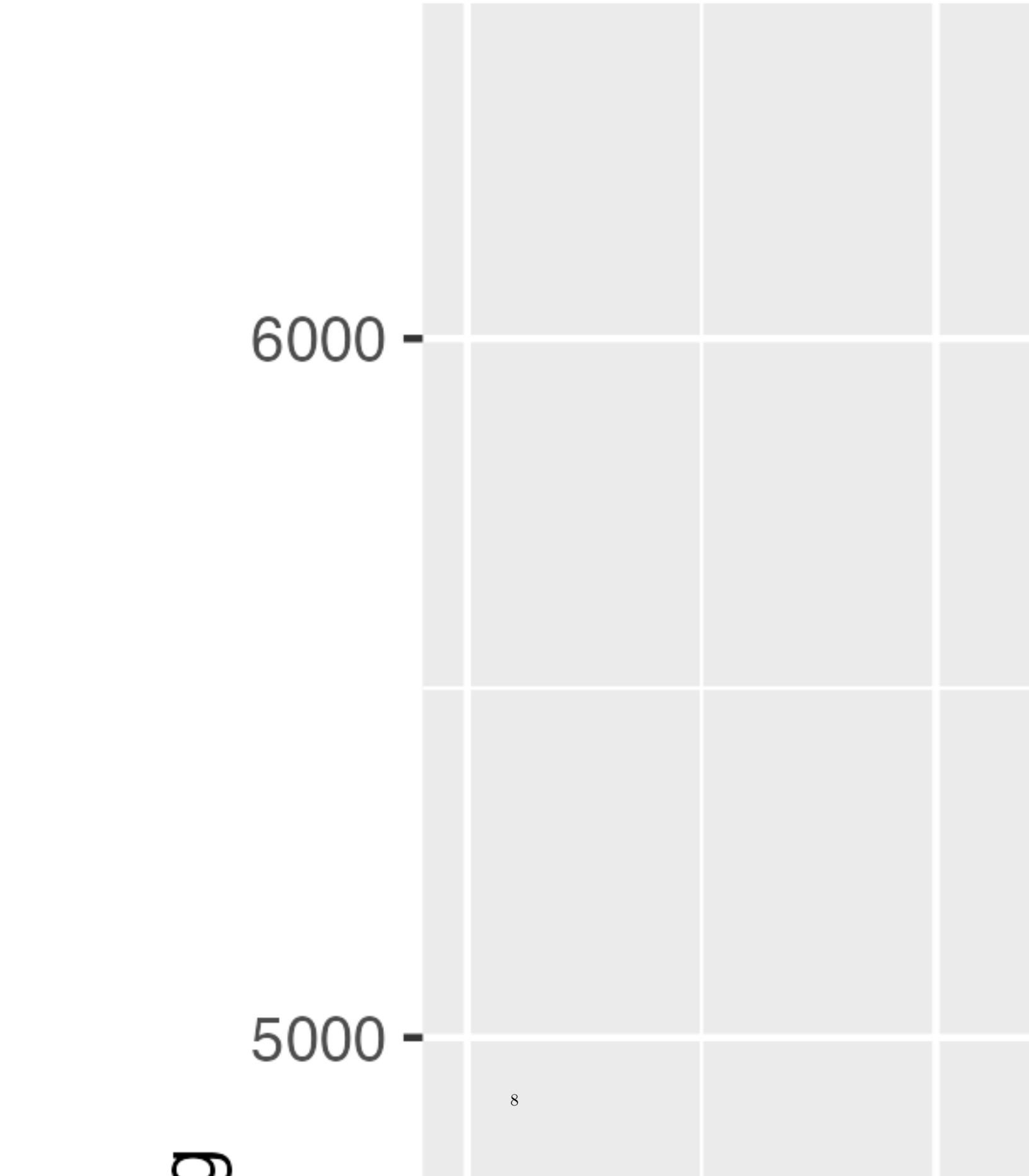


7. Add caption: "Data come from the palmerpenguins package."

```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point(na.rm = TRUE) +
  labs(caption = "Data come from the palmerpenguins package.")
```



8. Recreate the visualization. What aesthetic should `bill_depth_mm` map to? Global or geom?


6000
5000

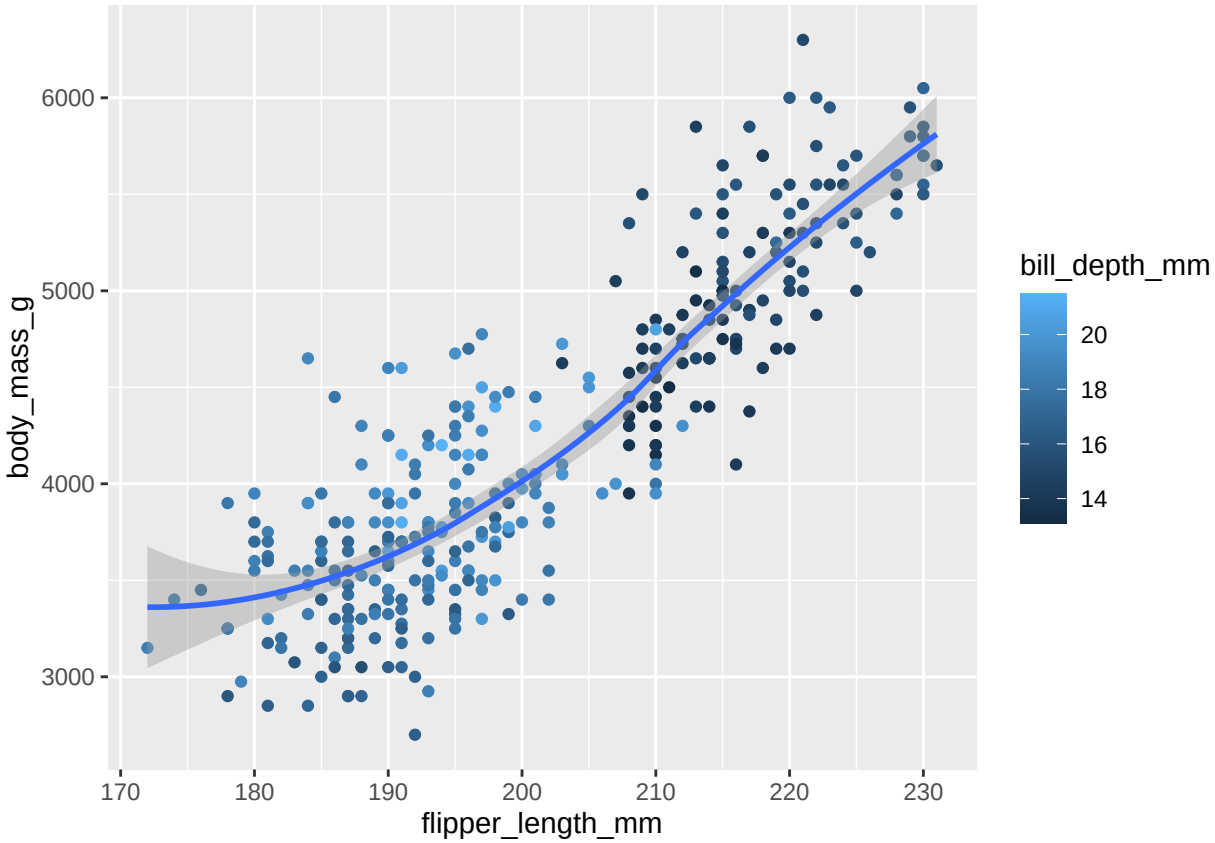
Answer: Map `bill_depth_mm` to **color**, and map it at the **geom level** (only points), so the smooth line stays one line.

```
ggplot(penguins, aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_point(aes(color = bill_depth_mm)) +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning: Removed 2 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



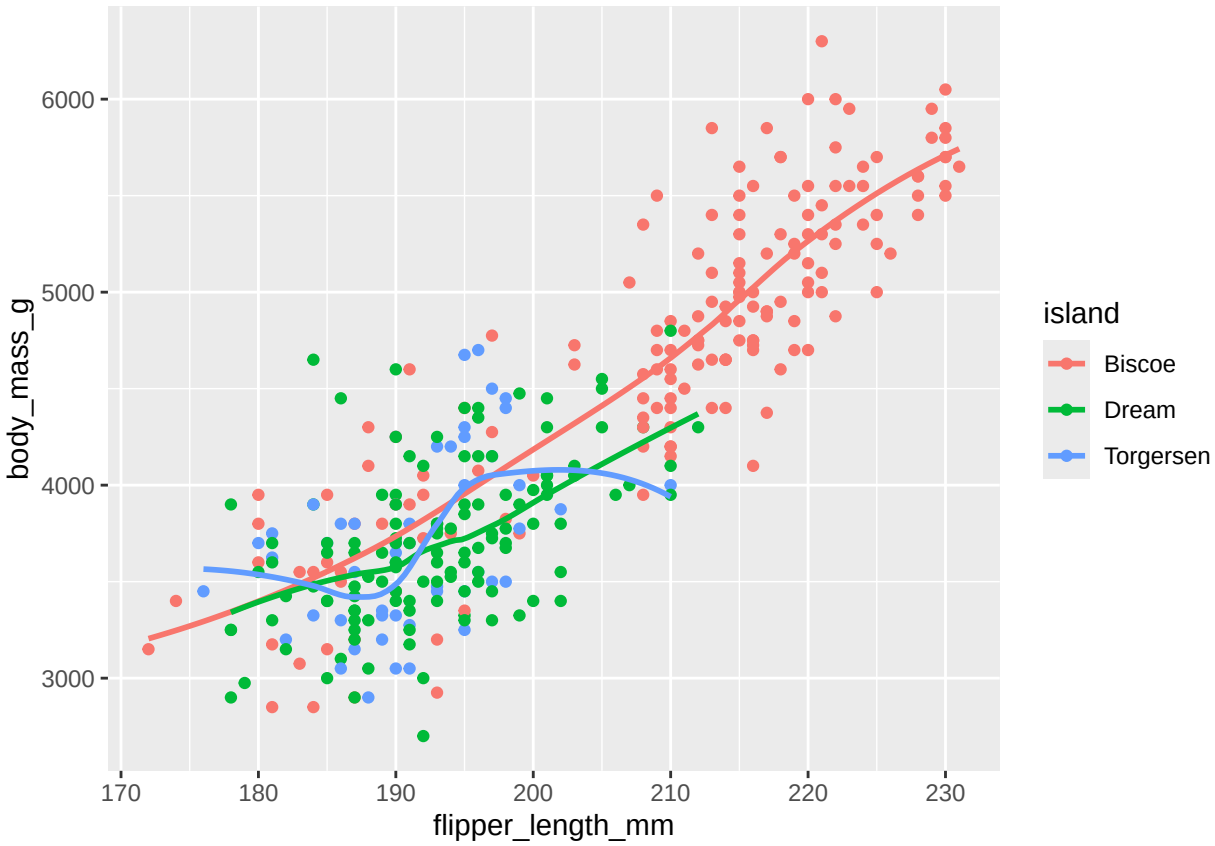
9. Predict the output, then run and check.

```
ggplot(
  data = penguins,
  mapping = aes(x = flipper_length_mm, y = body_mass_g, color = island)
) +
  geom_point() +
  geom_smooth(se = FALSE)
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning: Removed 2 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Answer: I expected points colored by island and separate smooth trend lines for each island, with **no** gray confidence band (because `se = FALSE`).

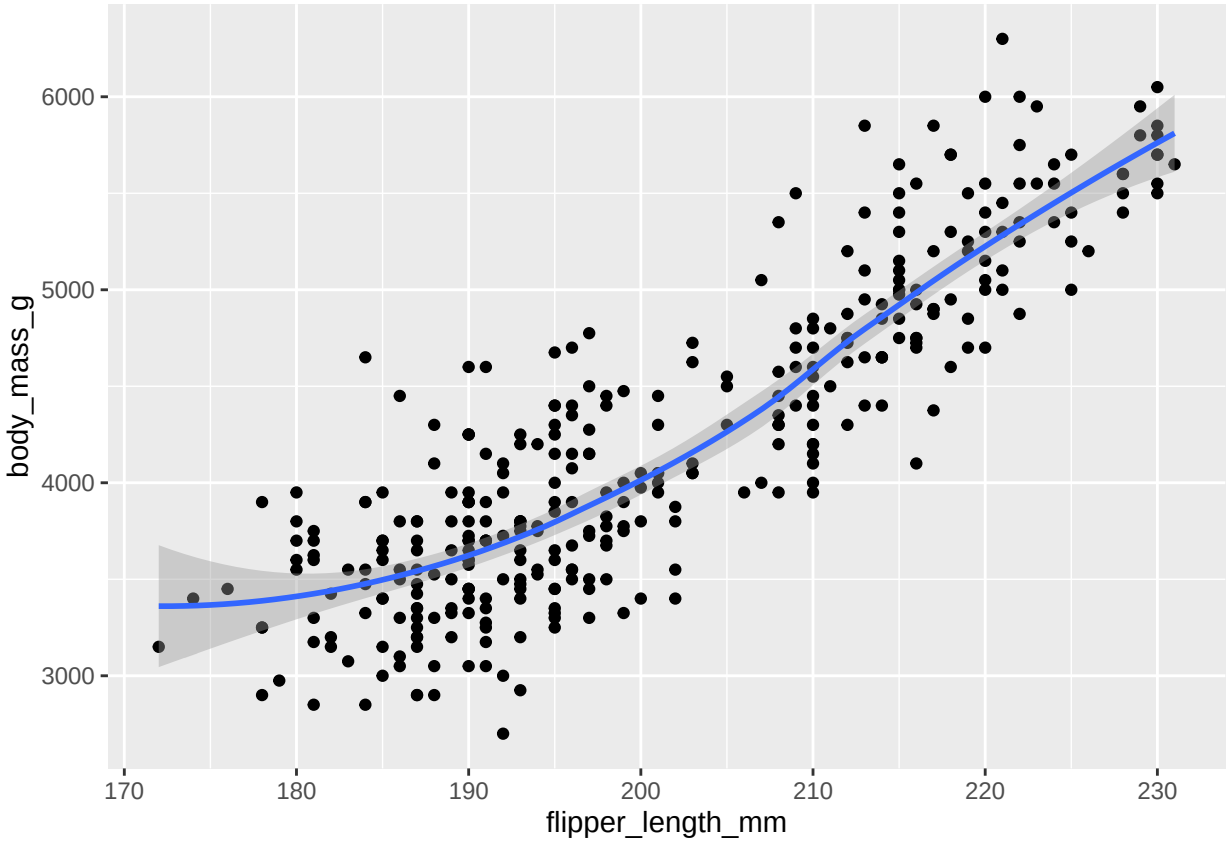
10. Will these two graphs look different? Why/why not?

```
ggplot(
  data = penguins,
  mapping = aes(x = flipper_length_mm, y = body_mass_g)
) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning: Removed 2 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

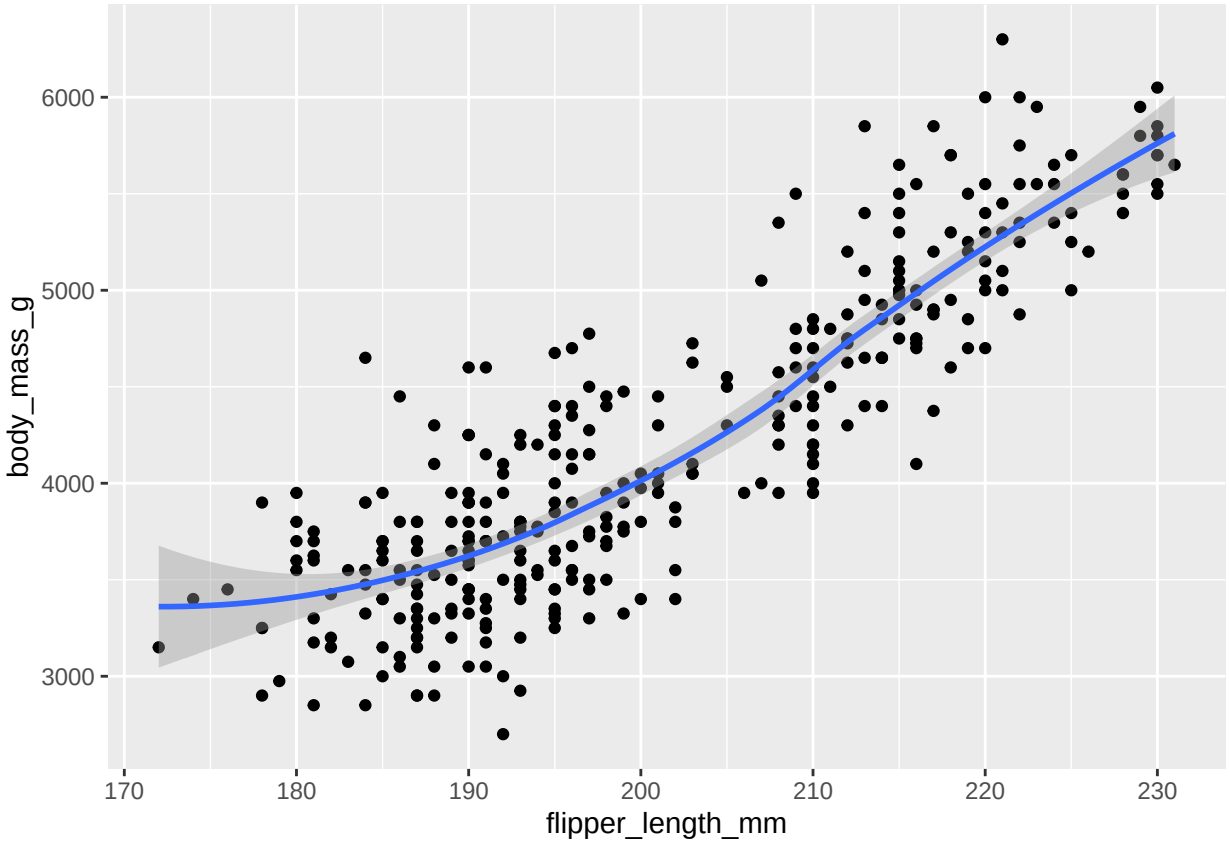
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
ggplot() +
  geom_point(
    data = penguins,
    mapping = aes(x = flipper_length_mm, y = body_mass_g)
  ) +
  geom_smooth(
    data = penguins,
    mapping = aes(x = flipper_length_mm, y = body_mass_g)
  )
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

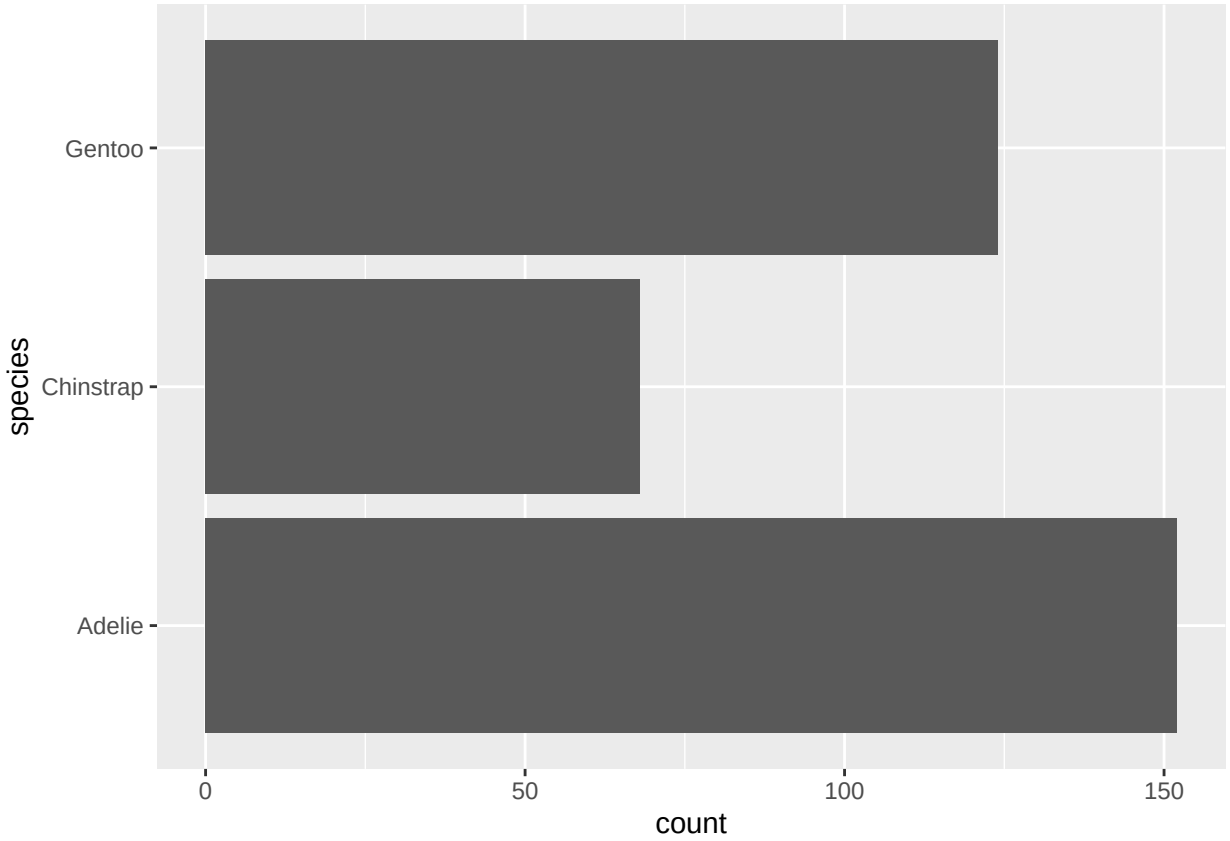
```
## Warning: Removed 2 rows containing non-finite outside the scale range (`stat_smooth()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Answer: They look the same because both plots use the same data and the same x/y mappings. One sets them globally, the other sets them inside each geom.

11. Bar plot of `species` where `species` is on the y aesthetic. How is it different?

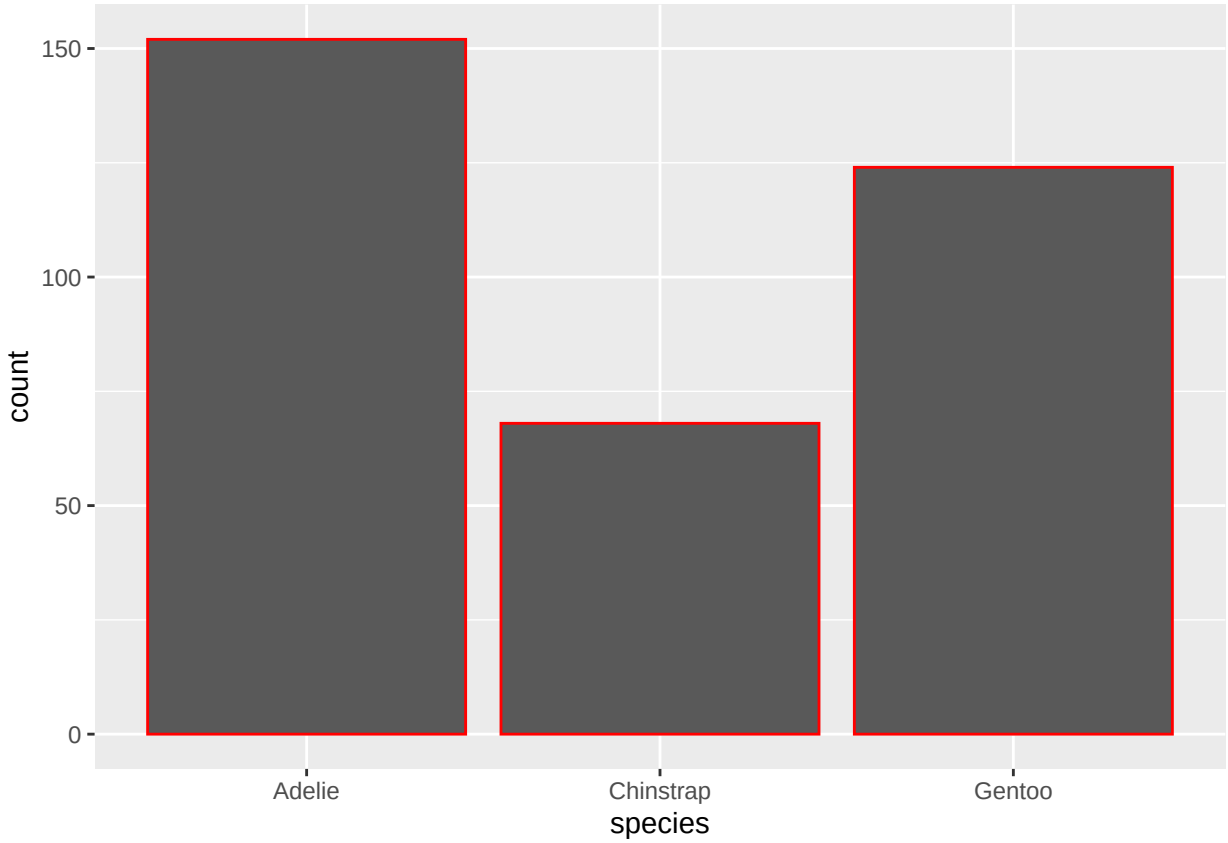
```
ggplot(penguins, aes(y = species)) +
  geom_bar()
```



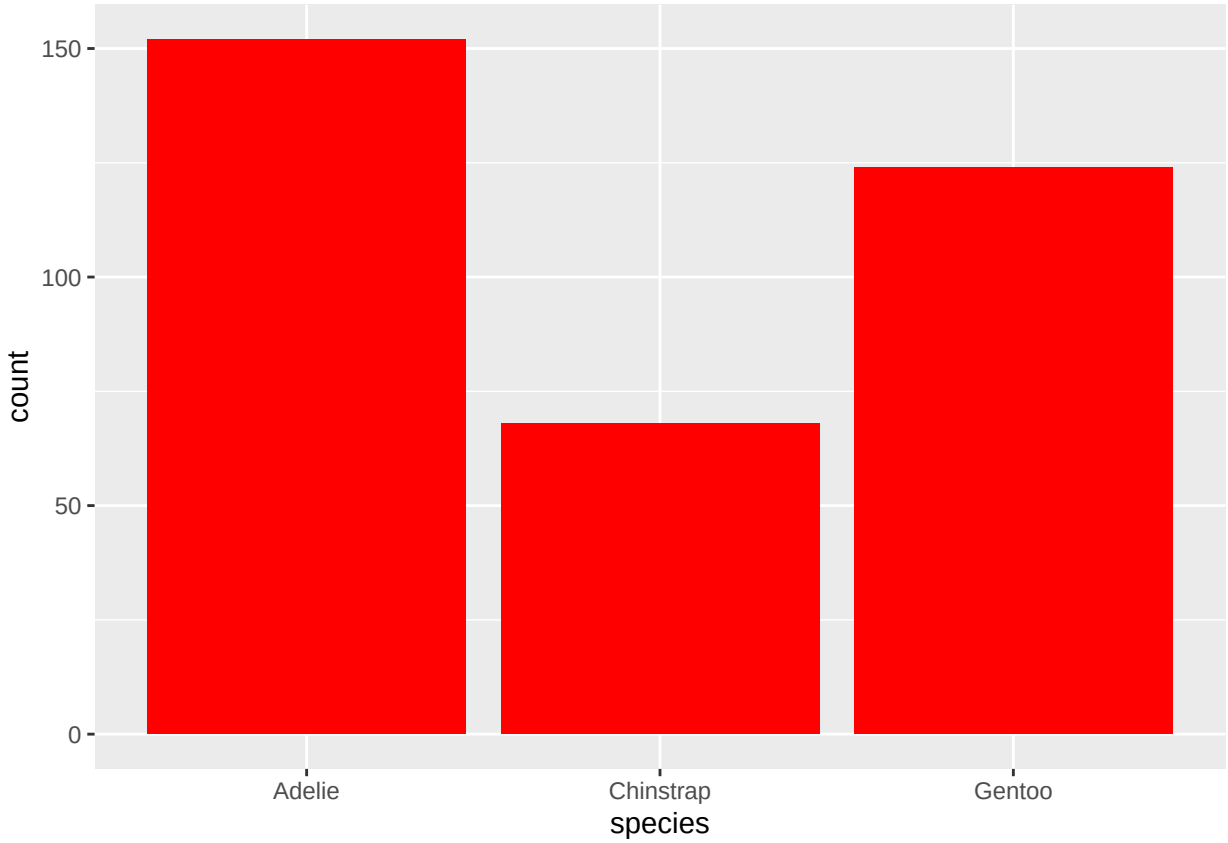
Answer: The bars are horizontal (same counts, just flipped direction).

12. How are the two plots different? Which is more useful for bars: `color` or `fill`?

```
ggplot(penguins, aes(x = species)) +
  geom_bar(color = "red")
```




```
ggplot(penguins, aes(x = species)) +
  geom_bar(fill = "red")
```



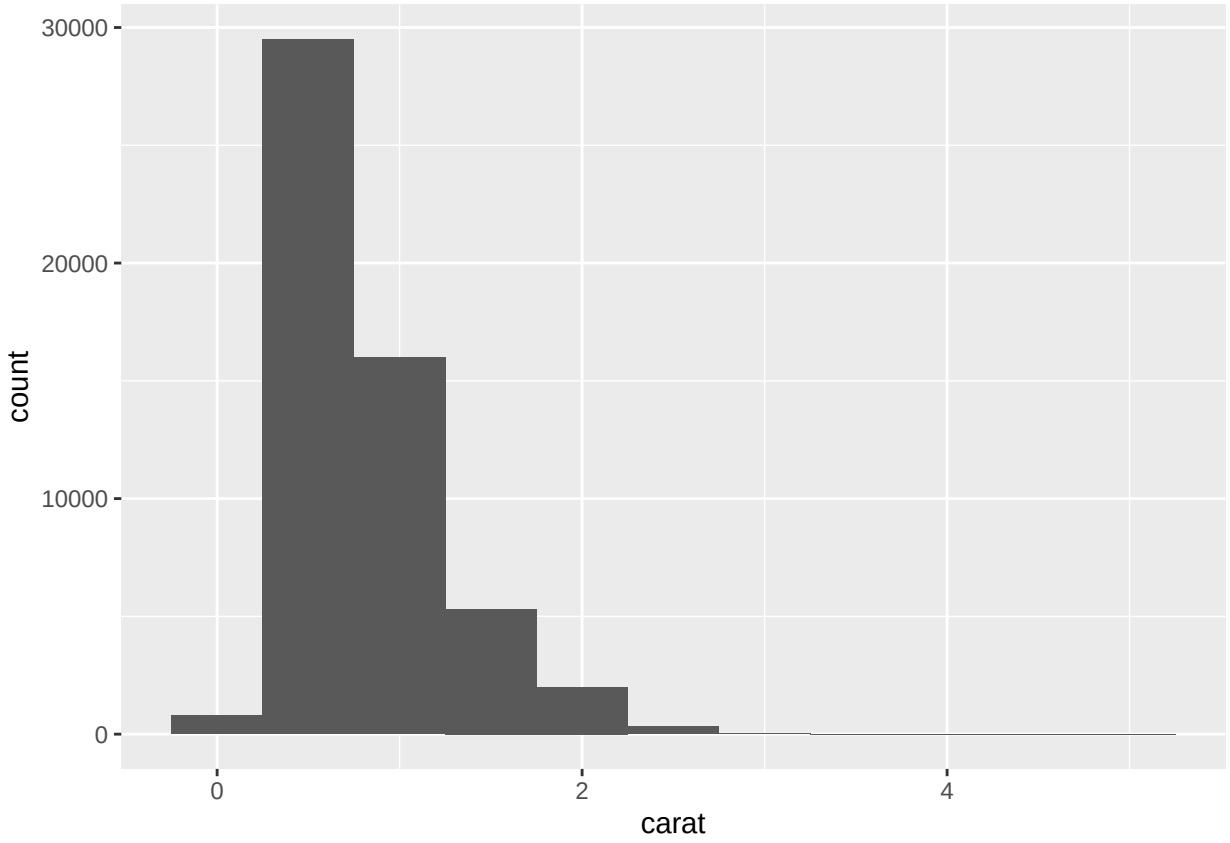
Answer: `color` changes the outline only. `fill` changes the inside of the bars. For bar color, **fill** is usually more useful.

13. What does the `bins` argument in `geom_histogram()` do?

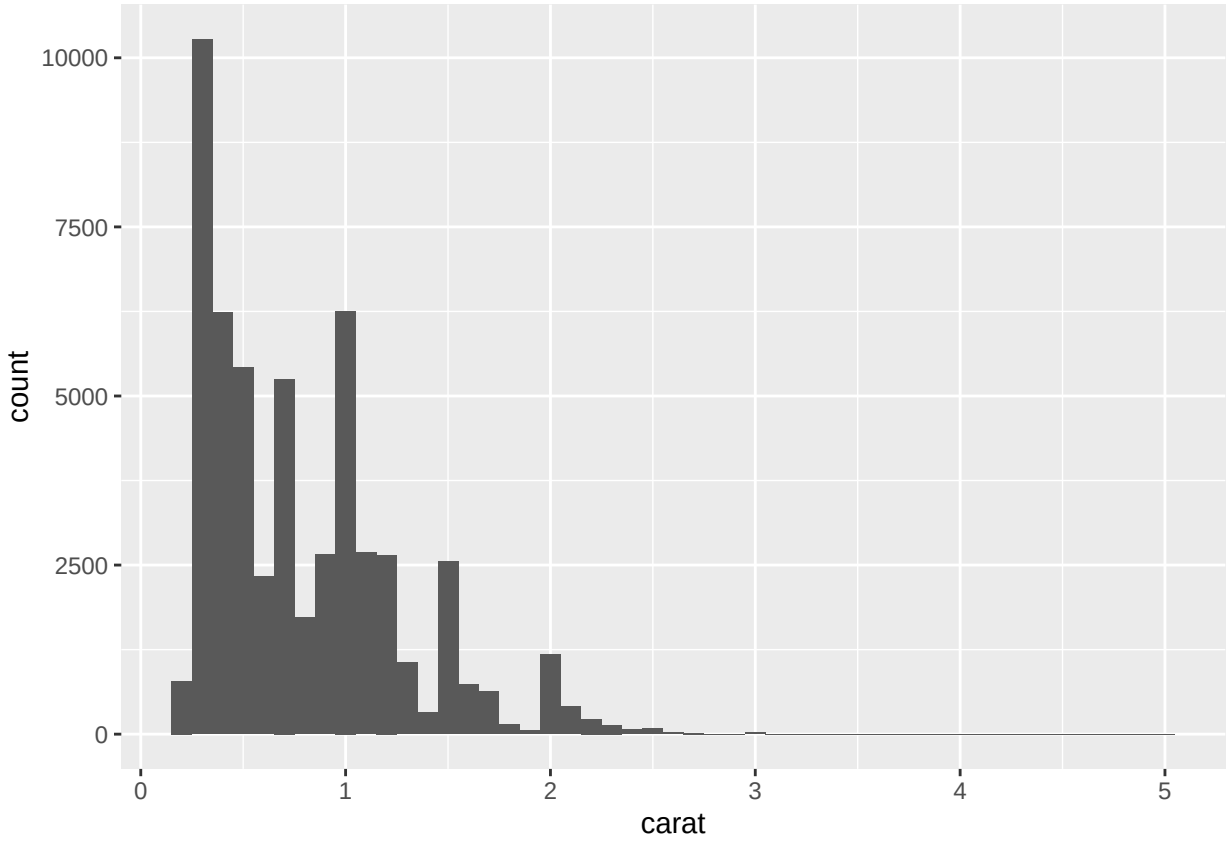
Answer: It controls how many bars (bins) the histogram uses (more bins = more detail, fewer bins = smoother look).

14. Histogram of `carat` in `diamonds`. Try binwidths. Which shows interesting patterns?

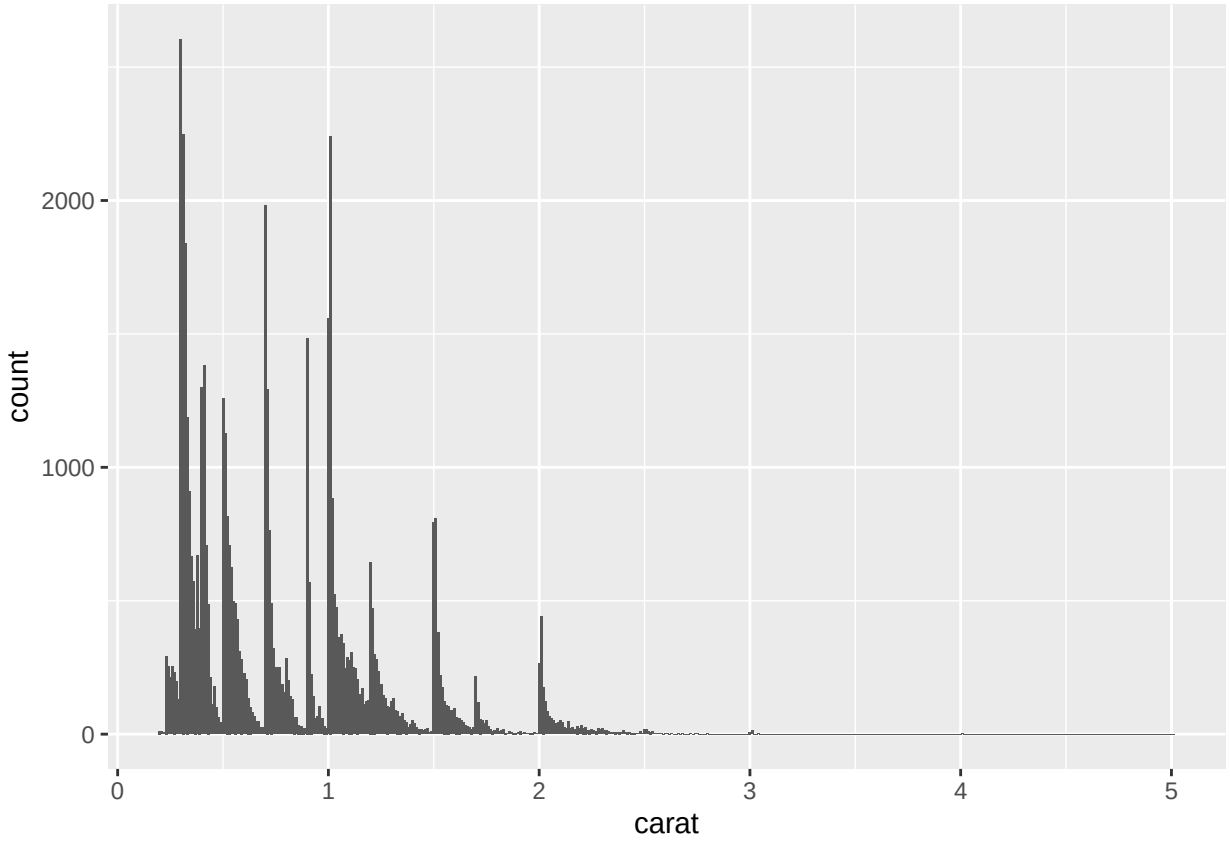
```
ggplot(diamonds, aes(x = carat)) +
  geom_histogram(binwidth = 0.5)
```




```
ggplot(diamonds, aes(x = carat)) +
  geom_histogram(binwidth = 0.1)
```




```
ggplot(diamonds, aes(x = carat)) +
  geom_histogram(binwidth = 0.01)
```



Answer: `binwidth = 0.01` shows the most interesting patterns because you can see spikes around common carat sizes (like 0.5, 1.0, etc.).

15. Which `mpg` variables are categorical vs numerical? How can you see this?

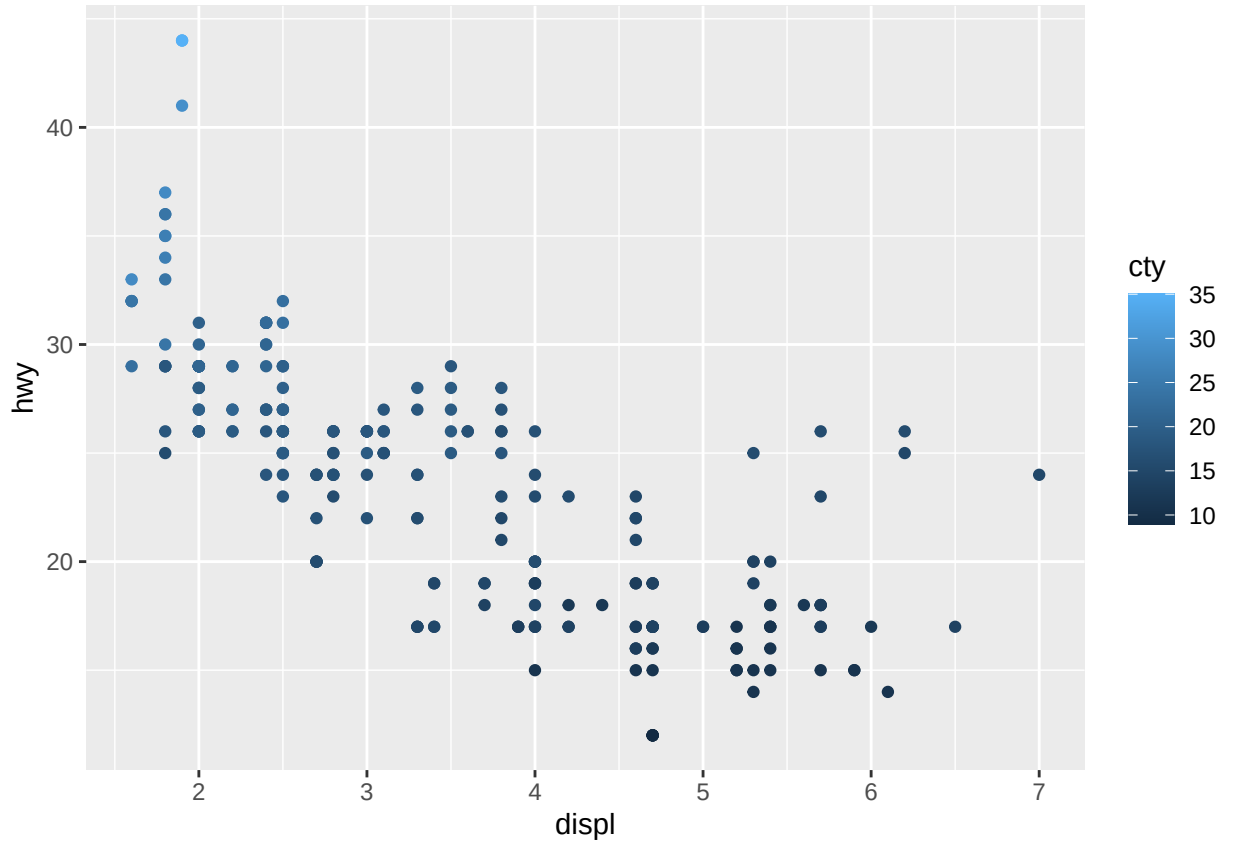
```
glimpse(mpg)
```

```
## Rows: 234
## Columns: 11
## $ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "audi", "~
## $ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 quattro", "~
## $ displ        <dbl> 1.8, 1.8, 2.0, 2.0, 2.8, 2.8, 3.1, 1.8, 1.8, 2.0, 2.0, 2.~
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, 200~
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, 8, ~
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "auto~
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4", "4~
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17, 1~
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25, 2~
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p~
## $ class        <chr> "compact", "compact", "compact", "compact", "compact", "c~
```

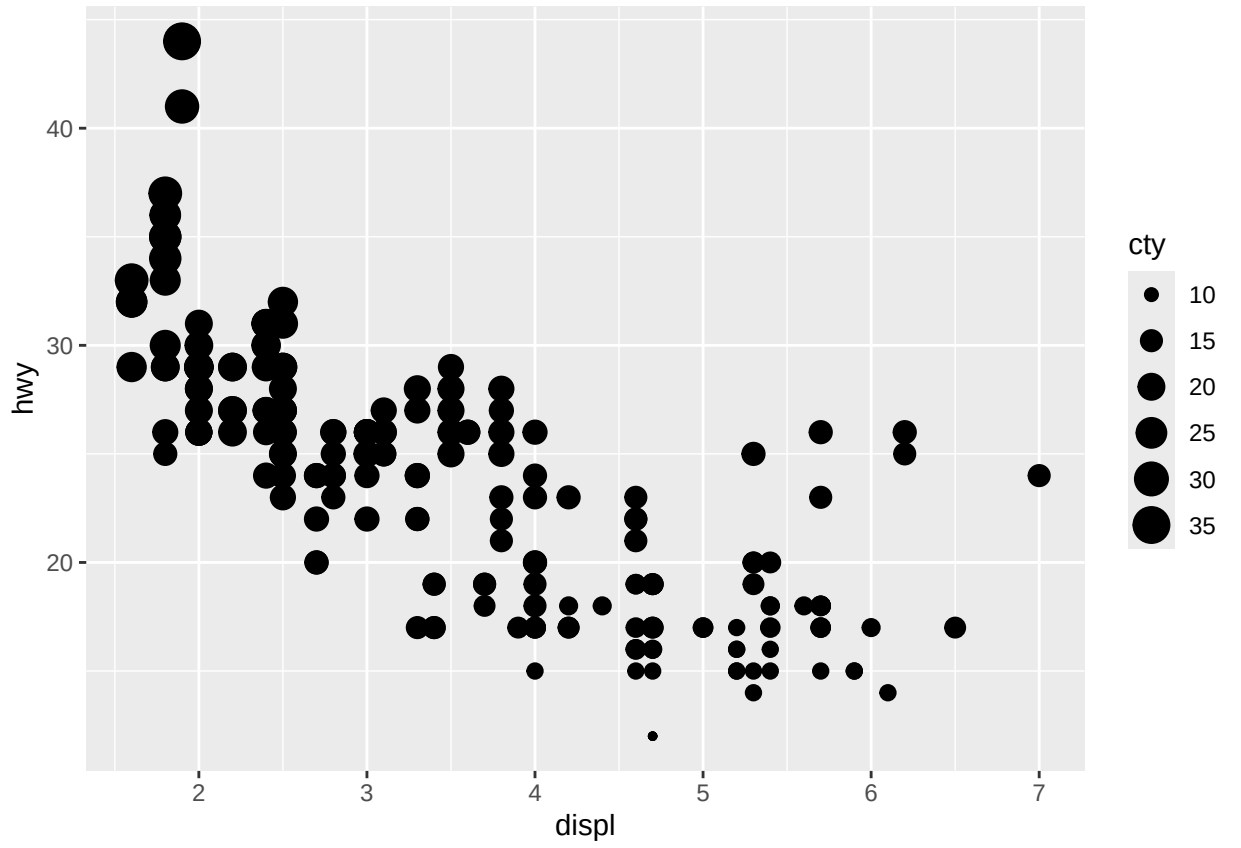
Answer (categorical): `manufacturer`, `model`, `trans`, `drv`, `fl`, `class`
Answer (numerical): `displ`, `year`, `cyl`, `cty`, `hwy`
You can see it in `glimpse(mpg)` (it shows the type: , , ,).

16. Scatterplot of `hwy` vs `displ`. Map a third **numerical** variable to color, size, both, then shape. What happens?

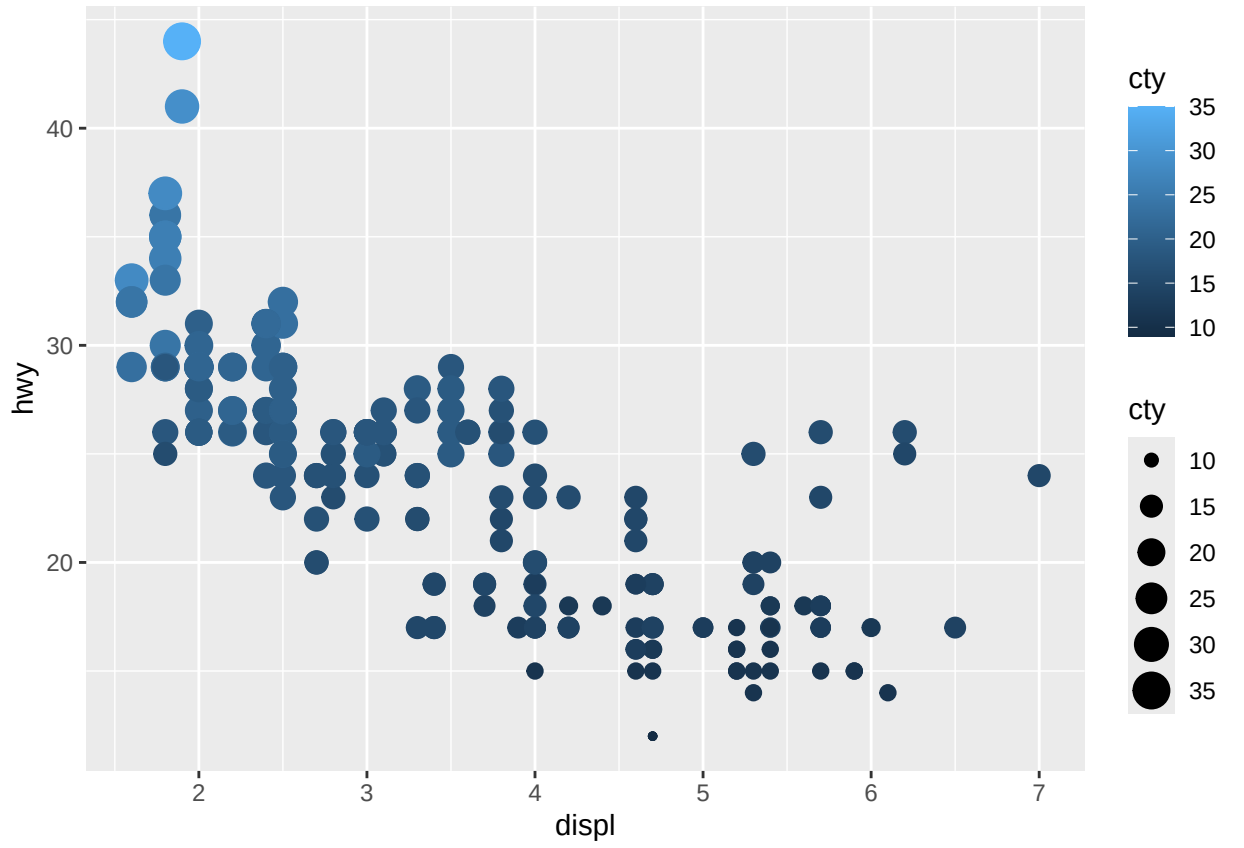
```
# color
ggplot(mpg, aes(x = displ, y = hwy, color = cty)) +
  geom_point()
```




```
# size
ggplot(mpg, aes(x = displ, y = hwy, size = cty)) +
  geom_point()
```




```
# color + size
ggplot(mpg, aes(x = displ, y = hwy, color = cty, size = cty)) +
  geom_point()
```

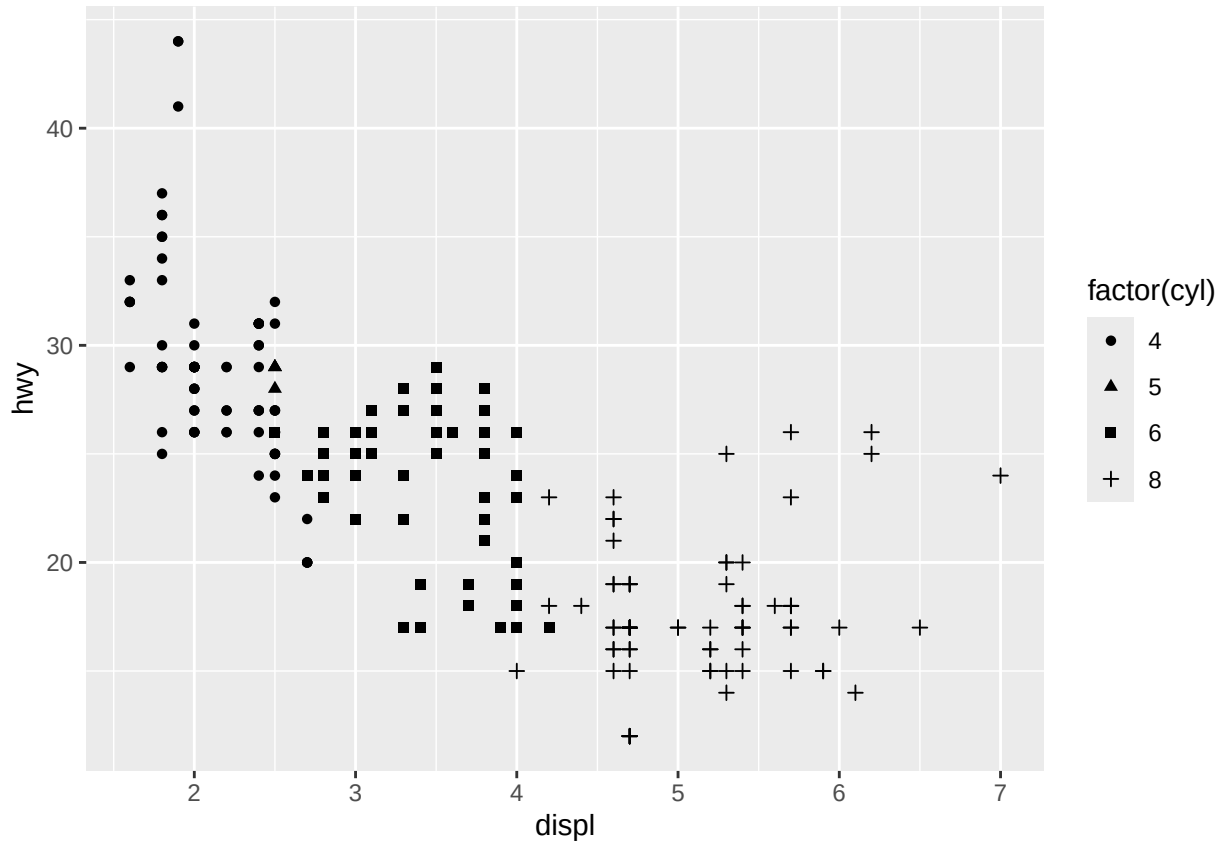



```
# shape (this will error because shape needs a discrete variable)
ggplot(mpg, aes(x = displ, y = hwy, shape = cty)) +
  geom_point()
```

Answer: `color` and `size` work fine with numerical variables (continuous scales). `shape` does **not** work with numerical variables unless you turn it into a categorical variable (like `factor(cyl)`).

Example shape using a categorical version:

```
ggplot(mpg, aes(x = displ, y = hwy, shape = factor(cyl))) +
  geom_point()
```

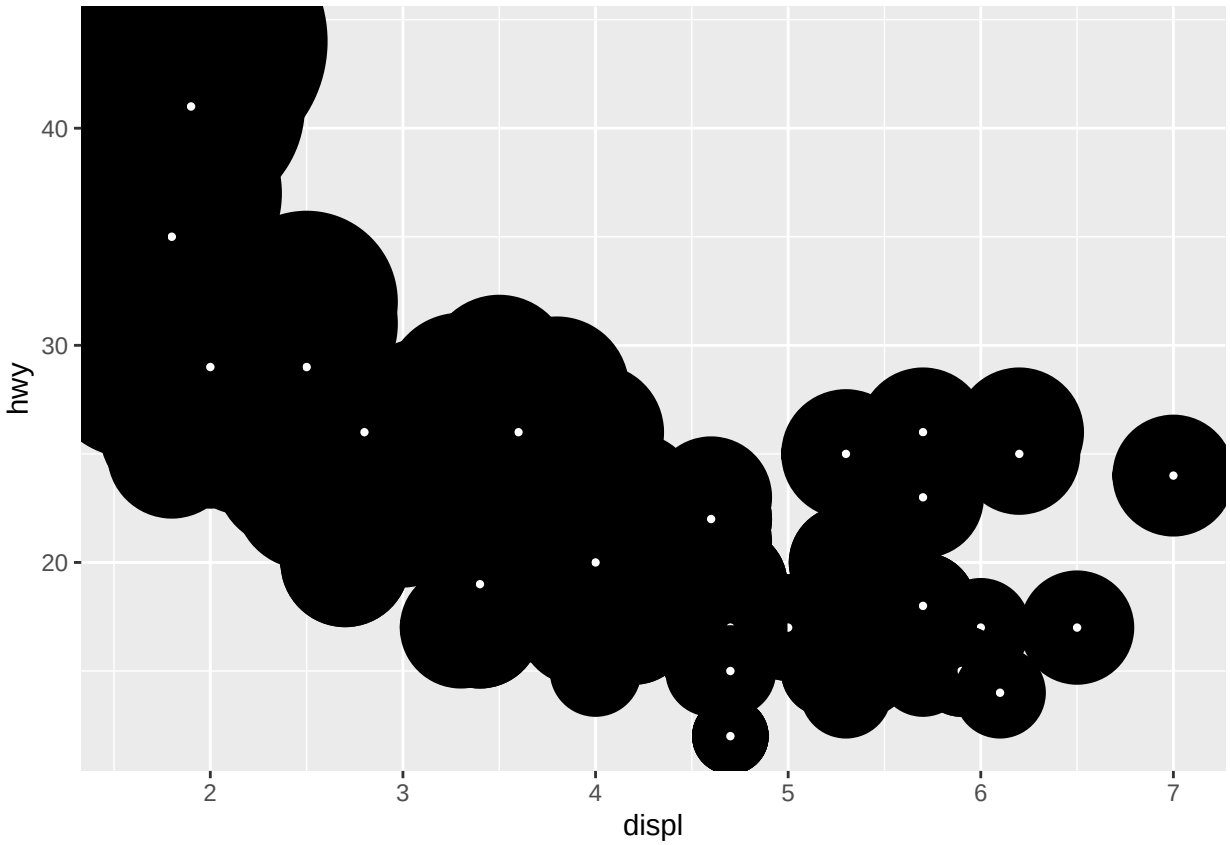


17. In the same plot, what happens if you map a third variable to `linewidth`?

```
ggplot(mpg, aes(x = displ, y = hwy, linewidth = cty)) +
  geom_point()
```

Answer: Usually you get a warning because `geom_point()` doesn't use `linewidth` the way line geoms do. For points, `stroke` controls border thickness (works best with shapes 21–25):

```
ggplot(mpg, aes(x = displ, y = hwy, stroke = cty)) +
  geom_point(shape = 21, fill = "white", color = "black")
```



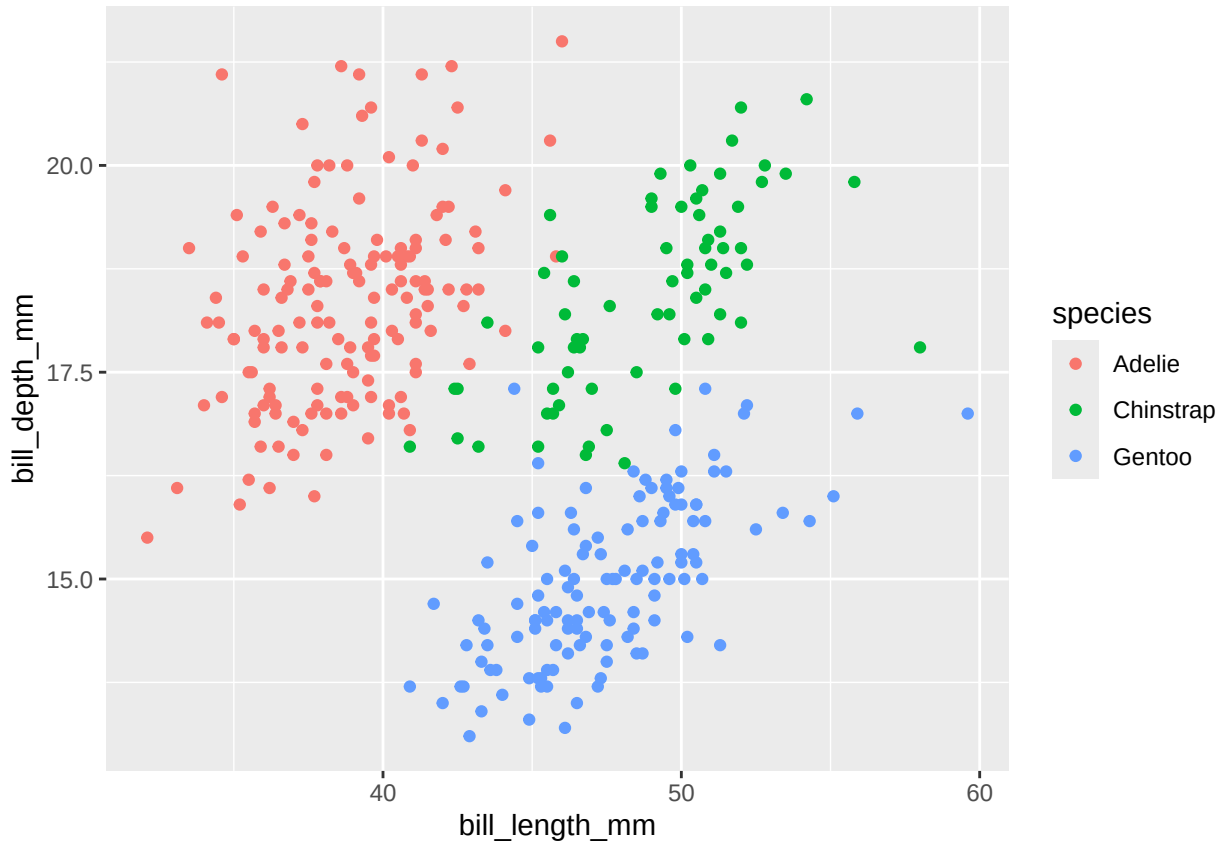
18. What happens if you map the same variable to multiple aesthetics?

Answer: It repeats the same info multiple ways (ex: color + shape). It can make groups easier to see, but it can also make the plot look too busy.

19. Scatterplot `bill_depth_mm` vs `bill_length_mm`, color by species. What does it reveal? What about faceting?

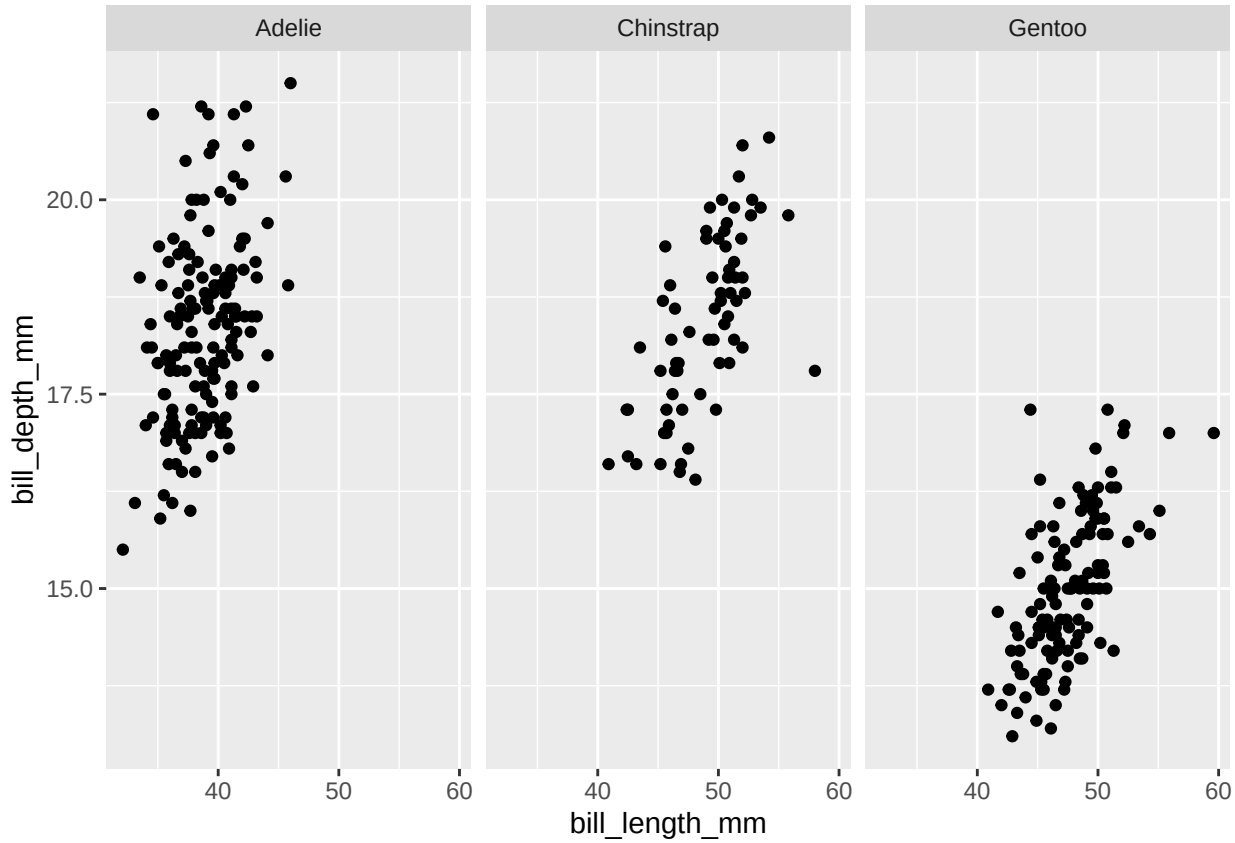
```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm, color = species)) +
  geom_point()
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point() +
  facet_wrap(~ species)
```

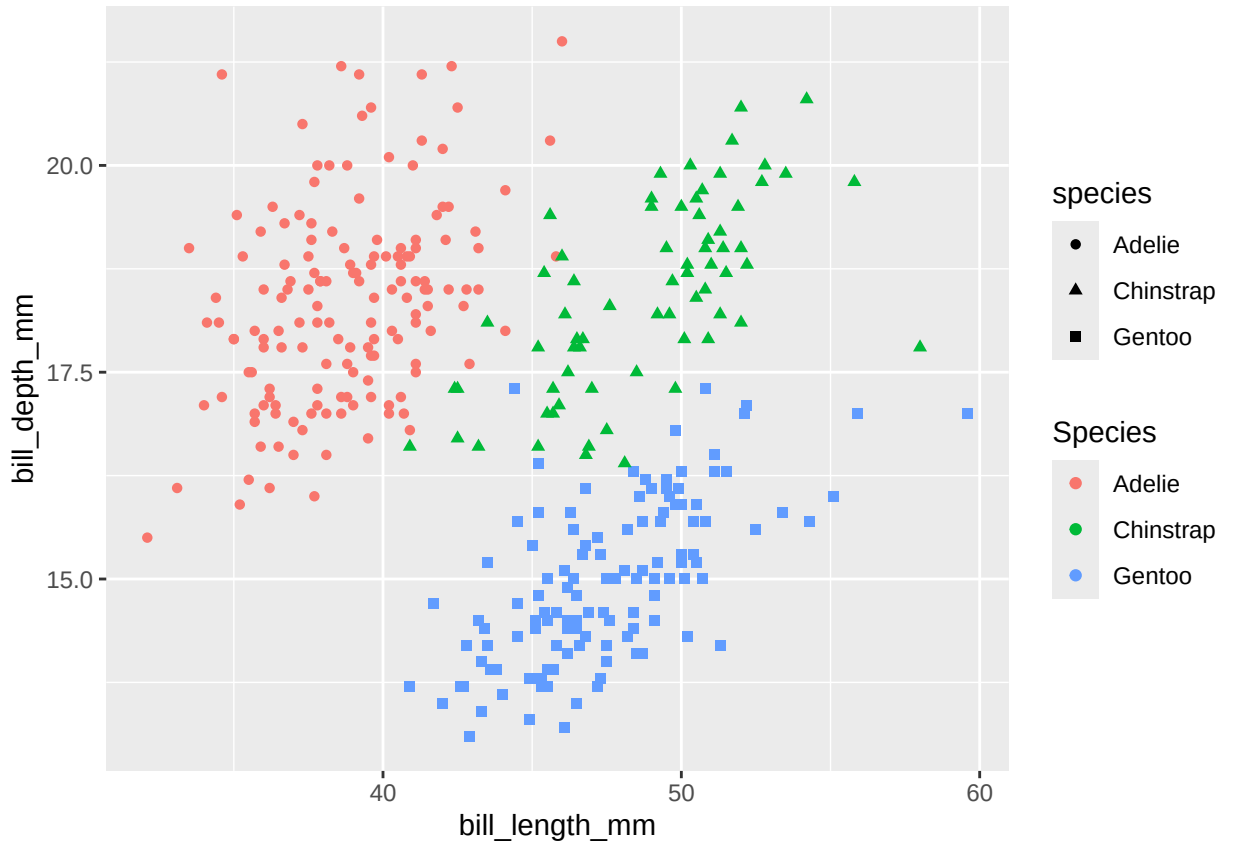
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Answer: Coloring shows the points form clear clusters by species. Faceting separates them so you can see the within-species patterns more clearly.

20. Why are there two legends? How to combine?

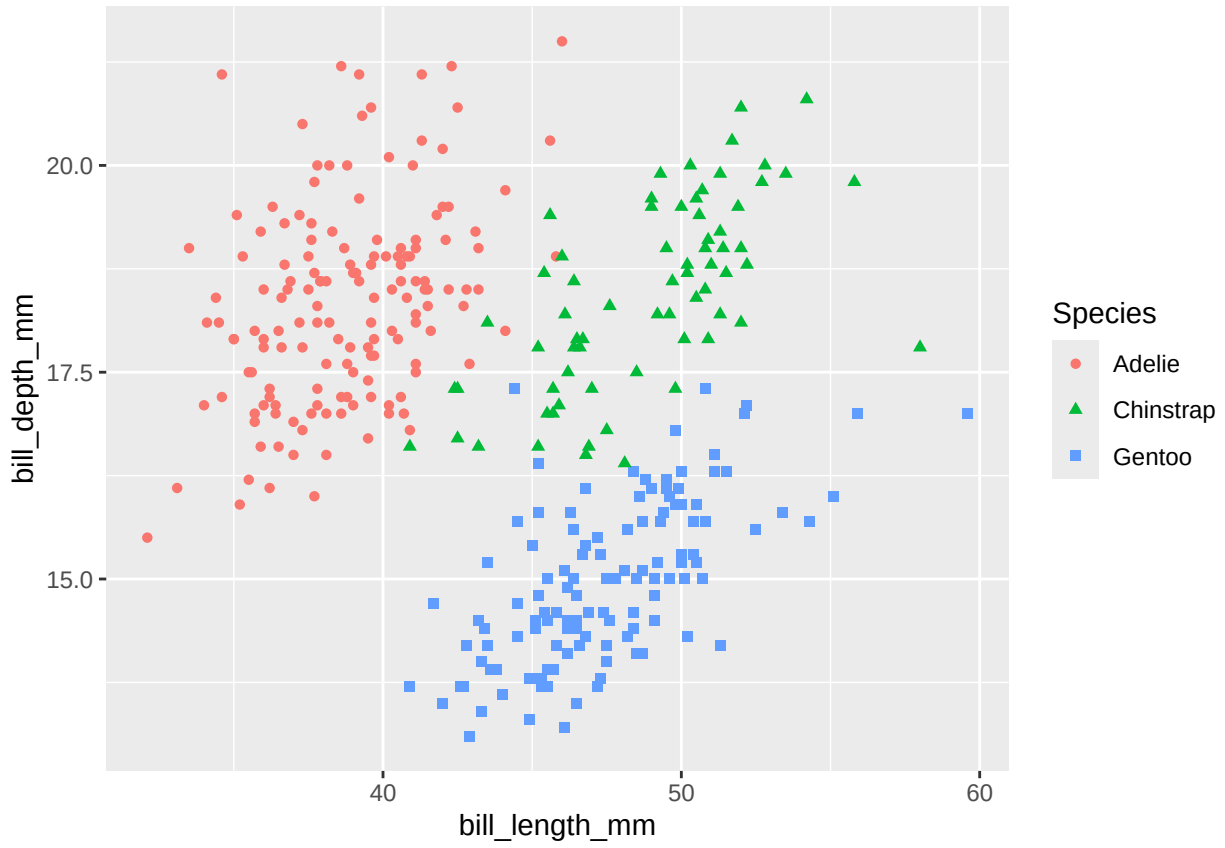
```
ggplot(
  data = penguins,
  mapping = aes(
    x = bill_length_mm, y = bill_depth_mm,
    color = species, shape = species
  )
) +
  geom_point() +
  labs(color = "Species")
```



Answer: There are two legends because color and shape have different legend titles. Combine by giving them the same title:

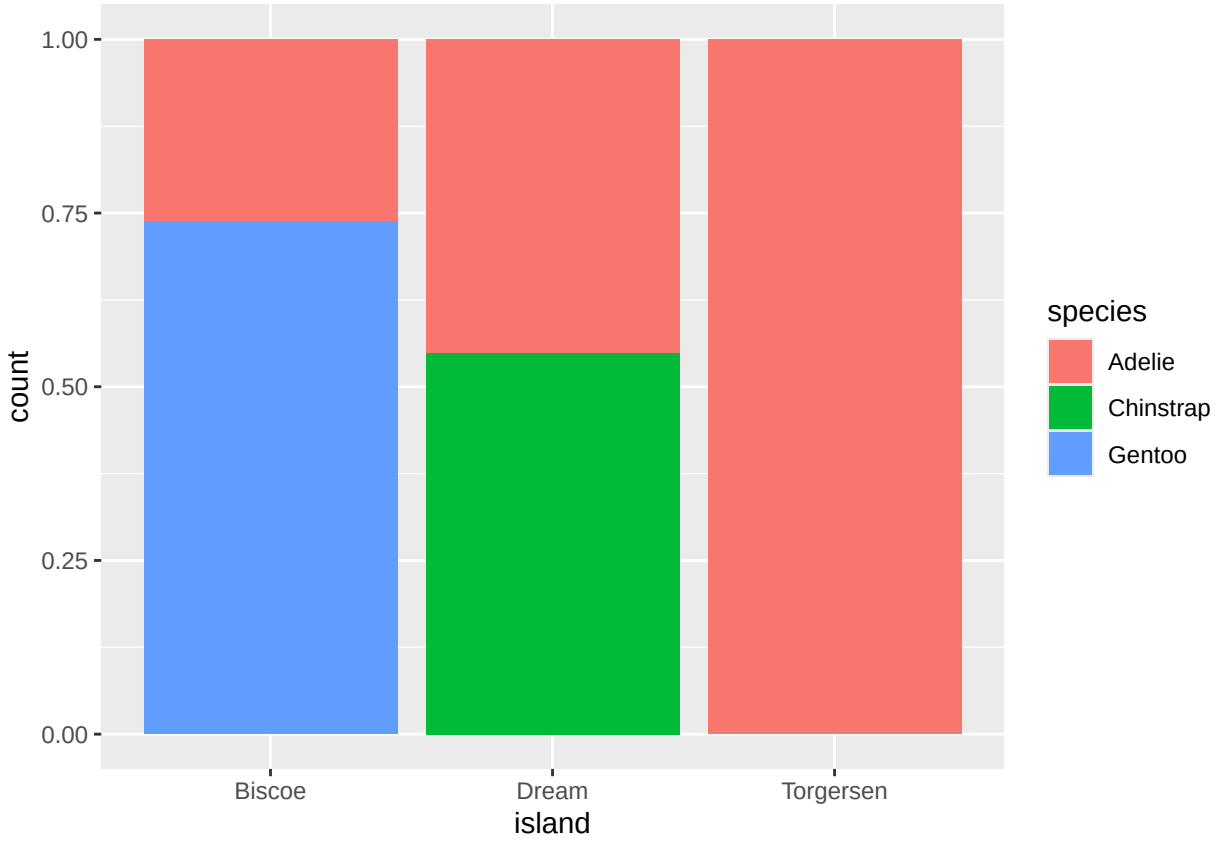
```
ggplot(
  data = penguins,
  mapping = aes(
    x = bill_length_mm, y = bill_depth_mm,
    color = species, shape = species
  )
) +
  geom_point() +
  labs(color = "Species", shape = "Species")
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

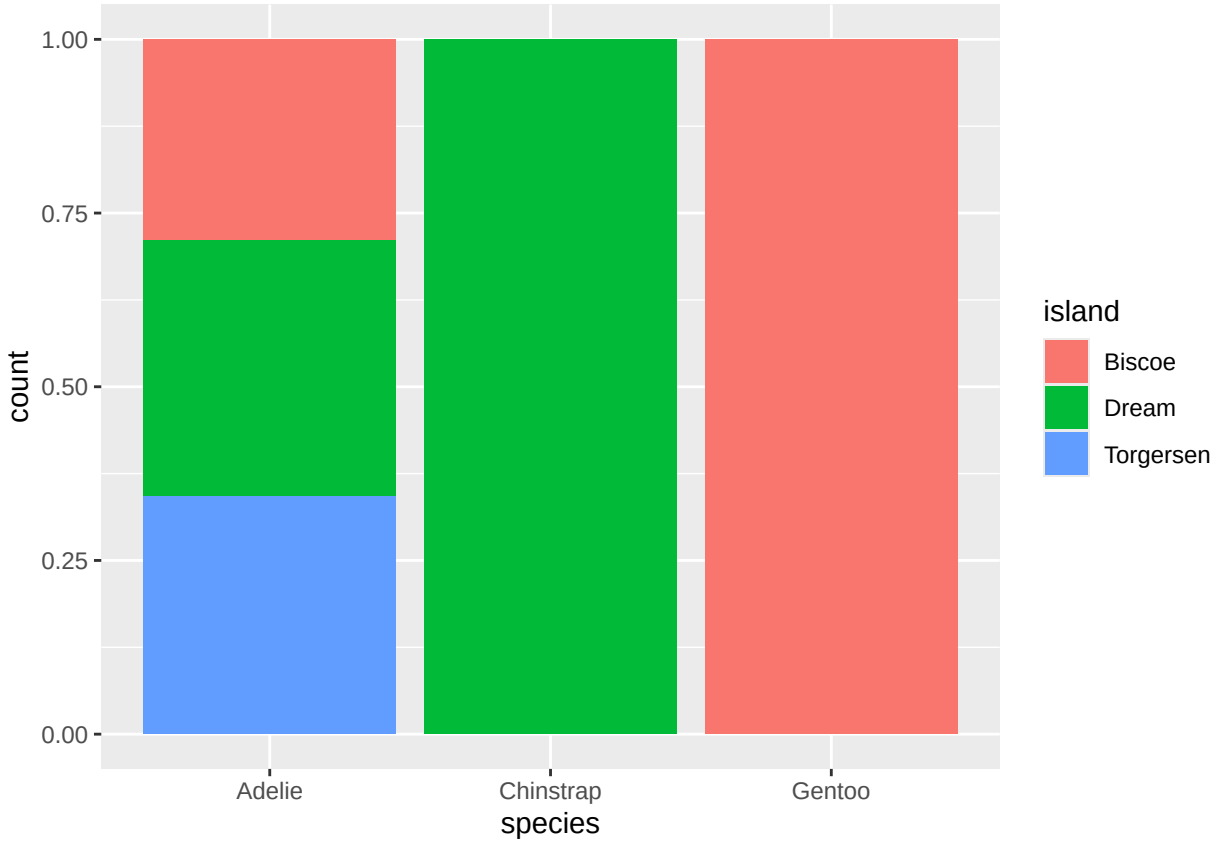


21. Stacked bar plots: what question can each answer?

```
ggplot(penguins, aes(x = island, fill = species)) +
  geom_bar(position = "fill")
```




```
ggplot(penguins, aes(x = species, fill = island)) +
  geom_bar(position = "fill")
```



Answer:
- First plot: "For each island, what *proportion* of penguins are each species?"
- Second plot: "For each species, what *proportion* come from each island?"

22. Which plot is saved as `mpg-plot.png`? Why?

```
ggplot(mpg, aes(x = class)) +
  geom_bar()
ggplot(mpg, aes(x = cty, y = hwy)) +
  geom_point()
ggsave("mpg-plot.png")
```

Answer: The second plot (cty vs hwy scatter) is saved because `ggsave()` saves the **last plot** that was created.

23. Save as PDF instead of PNG. How to find file types that work?

Answer: Change the filename to end in `.pdf`:

```
ggsave("mpg-plot.pdf")
```

To find what file types work, check the help page: `?ggsave` (it shows the `device` and supported formats).