Data and visualisation

Data Science in a Box

datasciencebox.org

Your turn: `AE 02 - Bechdel`

- The Bechdel test asks whether a work of fiction features at least two women who talk to each other about something other than a man, and there must be two women named characters.
- Go to our RStudio Server and start `AE 02 - Bechdel`. You will need to start an R project from the repository.
- Open and knit the R Markdown document `bechdel.Rmd`, review the document, and fill in the blanks.

What is in a dataset?

Dataset terminology

- Each row is an **observation**
- Each column is a **variable**

```
starwars
```

```
## # A tibble: 87 x 14
##   name    height  mass hair_color skin_color eye_color birth_year
##   <chr>    <int> <dbl> <chr>      <chr>      <chr>          <dbl>
## 1 Luke S~    172    77 blond      fair       blue            19
## 2 C-3PO      167    75 <NA>       gold       yellow         112
## 3 R2-D2       96    32 <NA>       white, bl~ red             33
## 4 Darth ~    202   136 none       white      yellow          41.9
## 5 Leia O~    150    49 brown      light      brown           19
## 6 Owen L~    178   120 brown, gr~ light      blue            52
## # ... with 81 more rows, and 7 more variables: sex <chr>,
## #   gender <chr>, homeworld <chr>, species <chr>, films <list>,
## #   vehicles <list>, starships <list>
```

Luke Skywalker

What's in the Star Wars data?

Take a `glimpse` at the data:

```
glimpse(starwars)
```

```
## Rows: 87
## Columns: 14
## $ name       <chr> "Luke Skywalker", "C-3PO", "R2-D2", "Darth V~
## $ height     <int> 172, 167, 96, 202, 150, 178, 165, 97, 183, 1~
## $ mass       <dbl> 77.0, 75.0, 32.0, 136.0, 49.0, 120.0, 75.0, ~
## $ hair_color <chr> "blond", NA, NA, "none", "brown", "brown, gr~
## $ skin_color <chr> "fair", "gold", "white, blue", "white", "lig~
## $ eye_color  <chr> "blue", "yellow", "red", "yellow", "brown", ~
## $ birth_year <dbl> 19.0, 112.0, 33.0, 41.9, 19.0, 52.0, 47.0, N~
## $ sex        <chr> "male", "none", "none", "male", "female", "m~
## $ gender     <chr> "masculine", "masculine", "masculine", "masc~
## $ homeworld  <chr> "Tatooine", "Tatooine", "Naboo", "Tatooine",~
## $ species    <chr> "Human", "Droid", "Droid", "Human", "Human",~
## $ films      <list> <"The Empire Strikes Back", "Revenge of the~
## $ vehicles   <list> <"Snowspeeder", "Imperial Speeder Bike">, <~
## $ starships  <list> <"X-wing", "Imperial shuttle">, <>, <>, "TI~
```

How many rows and columns does this dataset have? What does each row represent? What does each column represent?

```
?starwars
```

starwars {dplyr} R Documentation

Starwars characters

Description

This data comes from SWAPI, the Star Wars API, https://swapi.dev/

Usage

```
starwars
```

Format

A tibble with 87 rows and 14 variables:

name
: Name of the character

height
: Height (cm)

mass
: Weight (kg)

hair_color,skin_color,eye_color
: Hair, skin, and eye colors

birth_year
: Year born (BBY = Before Battle of Yavin)

How many rows and columns does this dataset have?

```
nrow(starwars) # number of rows
```

```
## [1] 87
```

```
ncol(starwars) # number of columns
```

```
## [1] 14
```

```
dim(starwars)  # dimensions (row column)
```

```
## [1] 87 14
```

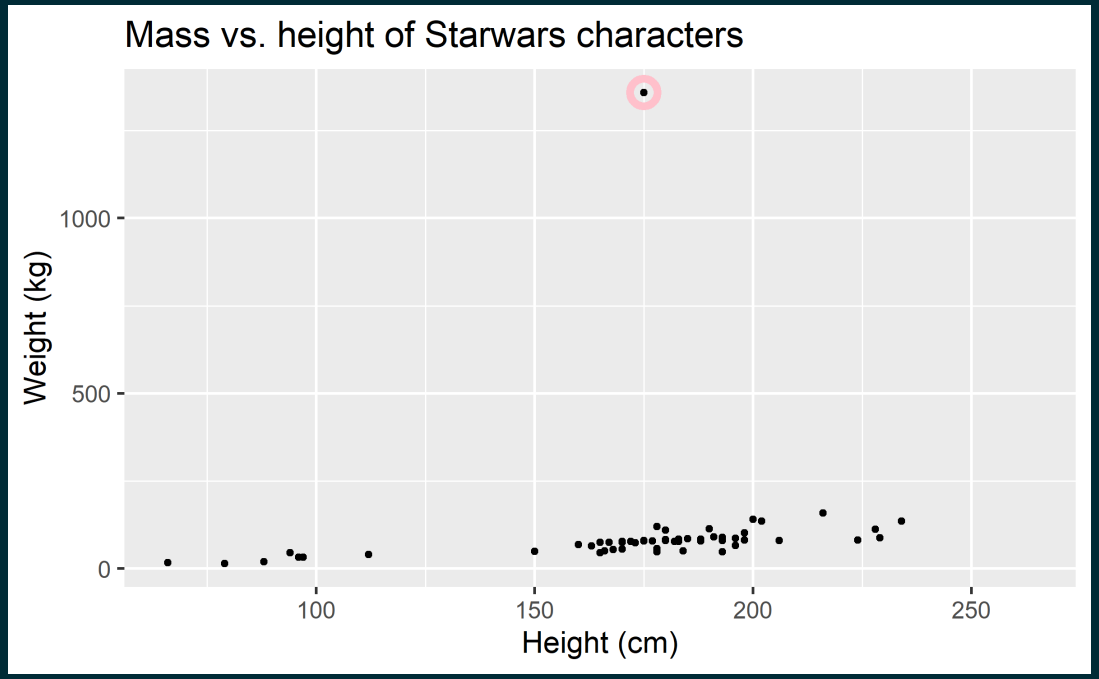
Exploratory data analysis

What is EDA?

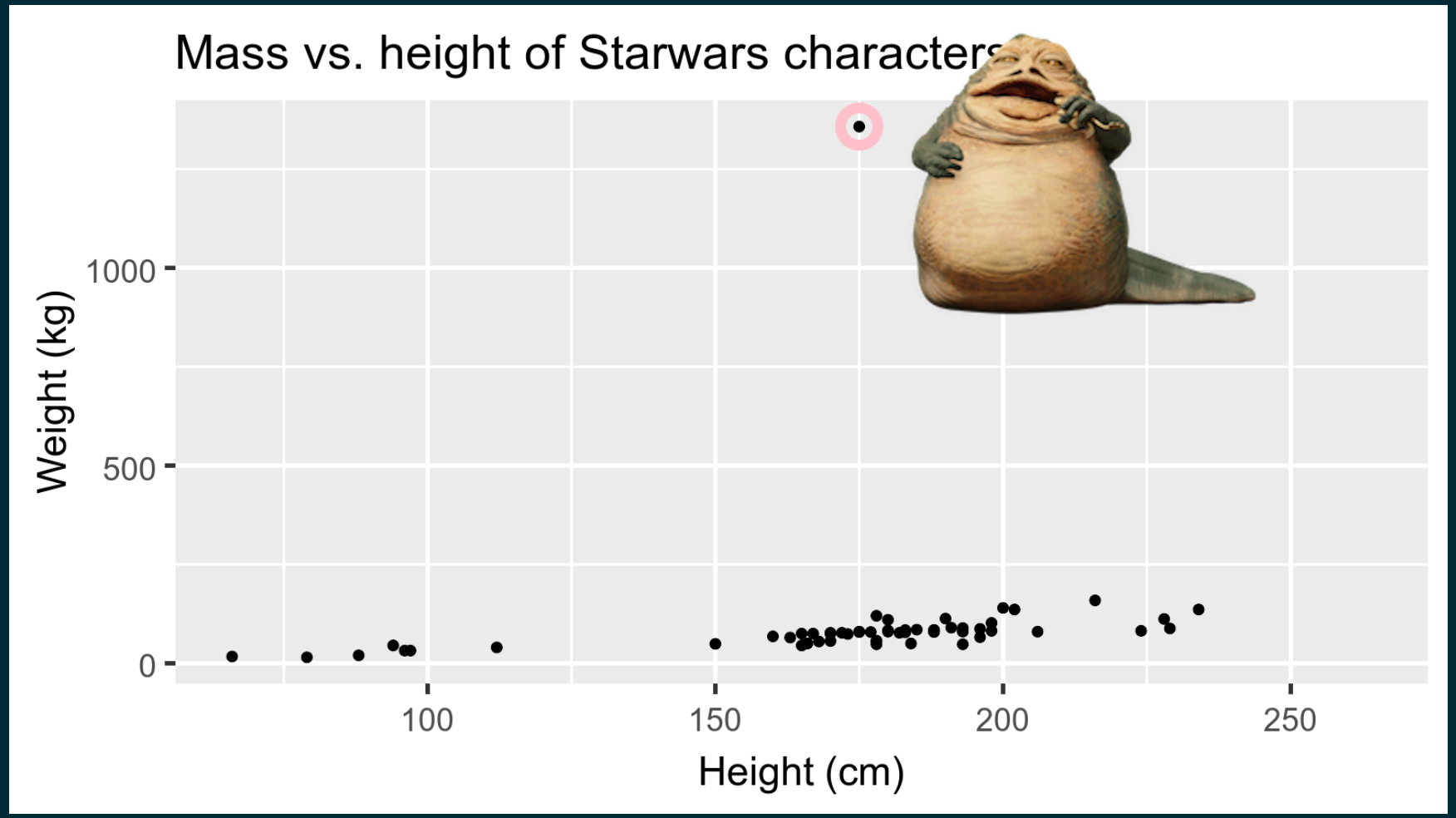
- Exploratory data analysis (EDA) is an approach to analysing data sets to summarize its main characteristics
- Often, this is visual -- this is what we'll focus on first
- But we might also calculate summary statistics and perform data wrangling/manipulation/transformation at (or before) this stage of the analysis -- this is what we'll focus on next

Mass vs. height

How would you describe the relationship between mass and height of Starwars characters? What other variables would help us understand data points that don't follow the overall trend? Who is the not so tall but really chubby character?



Jabba!


Mass vs. height of Starwars characters
Weight (kg)
1000
500
0
100
150
200
250
Height (cm)

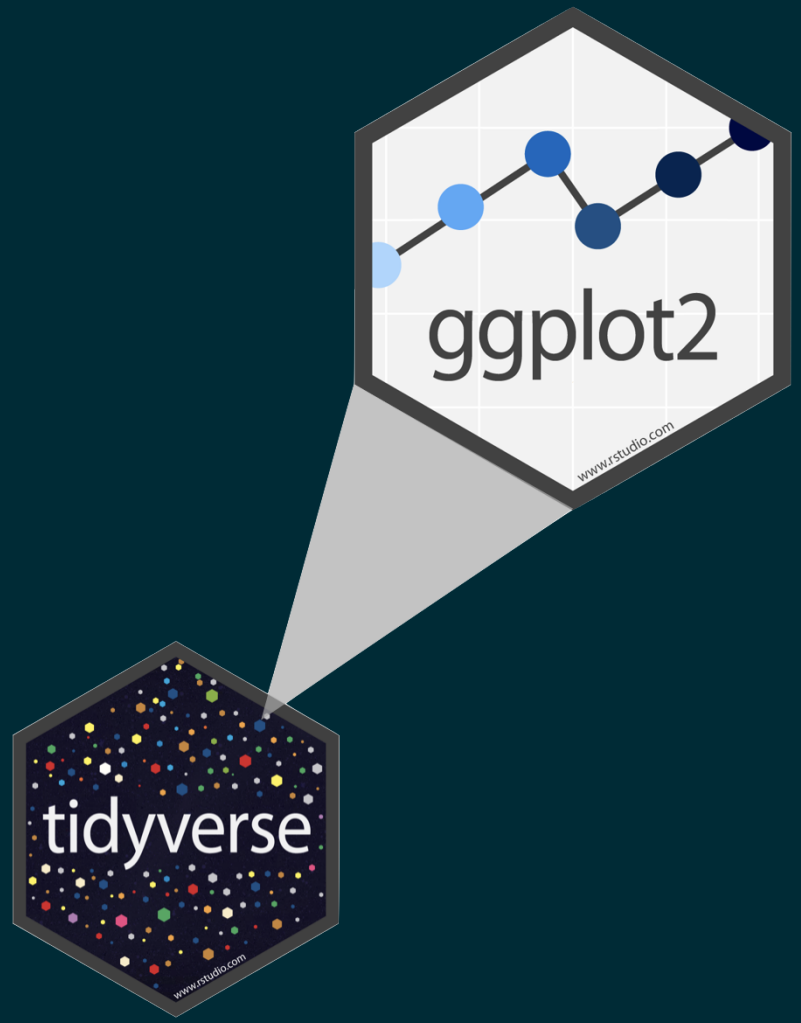
Data visualization

Data visualization

> *"The simple graph has brought more information to the data analyst's mind than any other device." --- John Tukey*

- Data visualization is the creation and study of the visual representation of data
- Many tools for visualizing data -- R is one of them
- Many approaches/systems within R for making data visualizations -- **ggplot2** is one of them, and that's what we're going to use

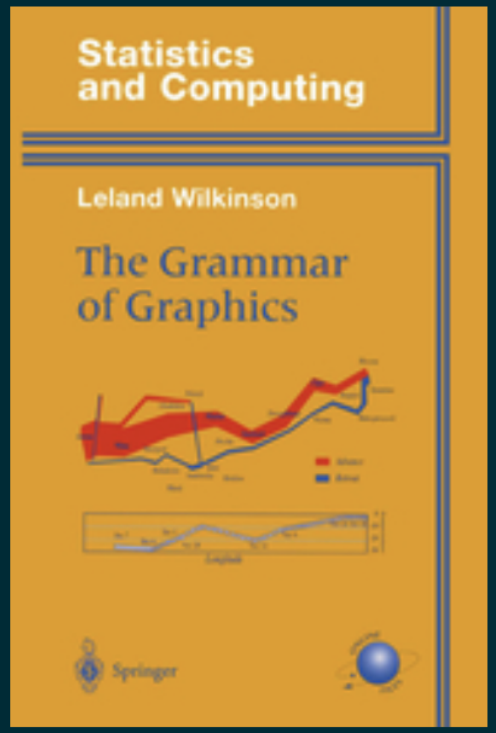
ggplot2 ∈ tidyverse

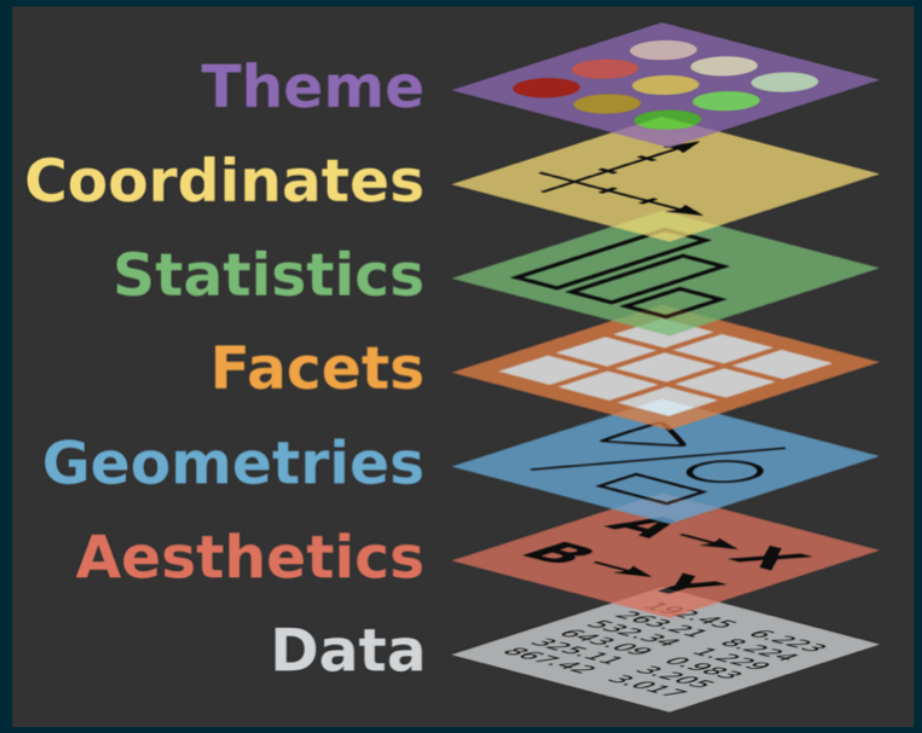


- **ggplot2** is tidyverse's data visualization package
- gg in "ggplot2" stands for Grammar of Graphics
- Inspired by the book **Grammar of Graphics** by Leland Wilkinson

Grammar of Graphics

A grammar of graphics is a tool that enables us to concisely describe the components of a graphic



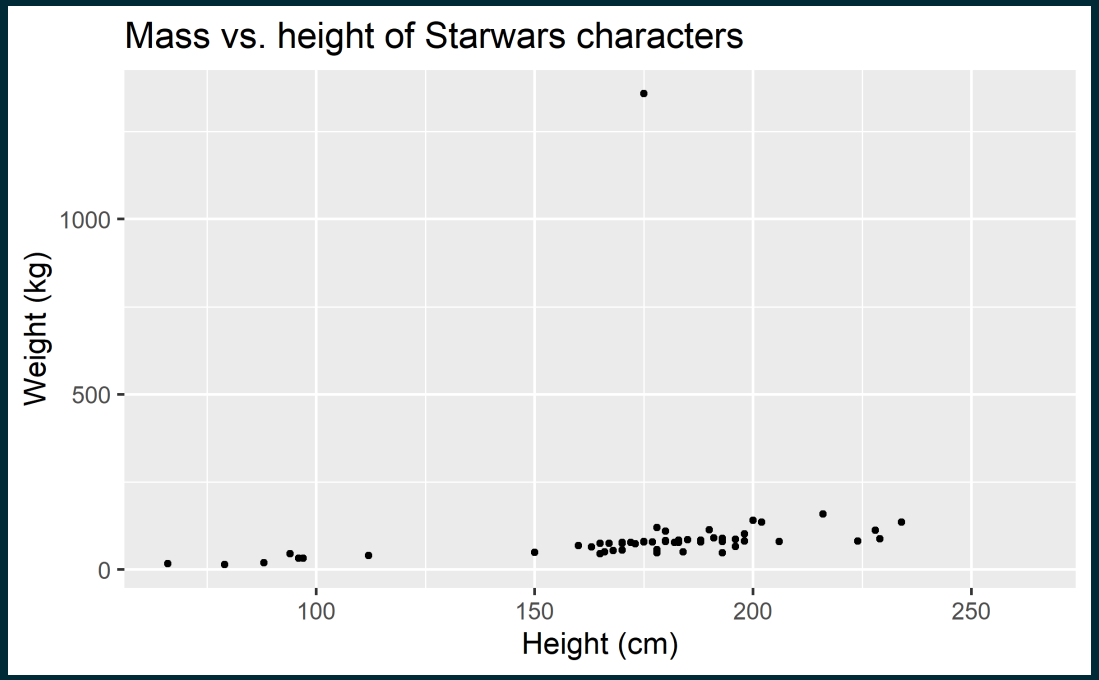


Source: BloggoType

Mass vs. height

```
ggplot(data = starwars, mapping = aes(x = height, y = mass)) +
  geom_point() +
  labs(title = "Mass vs. height of Starwars characters",
       x = "Height (cm)", y = "Weight (kg)")
```

```
## Warning: Removed 28 rows containing missing values (geom_point).
```



- What are the functions doing the plotting?
- What is the dataset being plotted?
- Which variables map to which features (aesthetics) of the plot?
- What does the warning mean?[+]

```
ggplot(data = starwars, mapping = aes(x = height, y = mass)) +
  geom_point() +
  labs(title = "Mass vs. height of Starwars characters",
       x = "Height (cm)", y = "Weight (kg)")
```

```
## Warning: Removed 28 rows containing missing values (geom_point).
```

[+]Suppressing warning to subsequent slides to save space

Hello ggplot2!

- `ggplot()` is the main function in ggplot2
- Plots are constructed in layers
- Structure of the code for plots can be summarized as

```
ggplot(data = [dataset],
       mapping = aes(x = [x-variable], y = [y-variable])) +
   geom_xxx() +
   other options
```

- The ggplot2 package comes with the tidyverse

```
library(tidyverse)
```

- For help with ggplot2, see ggplot2.tidyverse.org

Why do we visualize?

Anscombe's quartet

	set	x	y
1	I	10	8.04
2	I	8	6.95
3	I	13	7.58
4	I	9	8.81
5	I	11	8.33
6	I	14	9.96
7	I	6	7.24
8	I	4	4.26
9	I	12	10.84
10	I	7	4.82
11	I	5	5.68
12	II	10	9.14
13	II	8	8.14
14	II	13	8.74
15	II	9	8.77
16	II	11	9.26
17	II	14	8.10
18	II	6	6.13
19	II	4	3.10
20	II	12	9.13
21	II	7	7.26
22	II	5	4.74

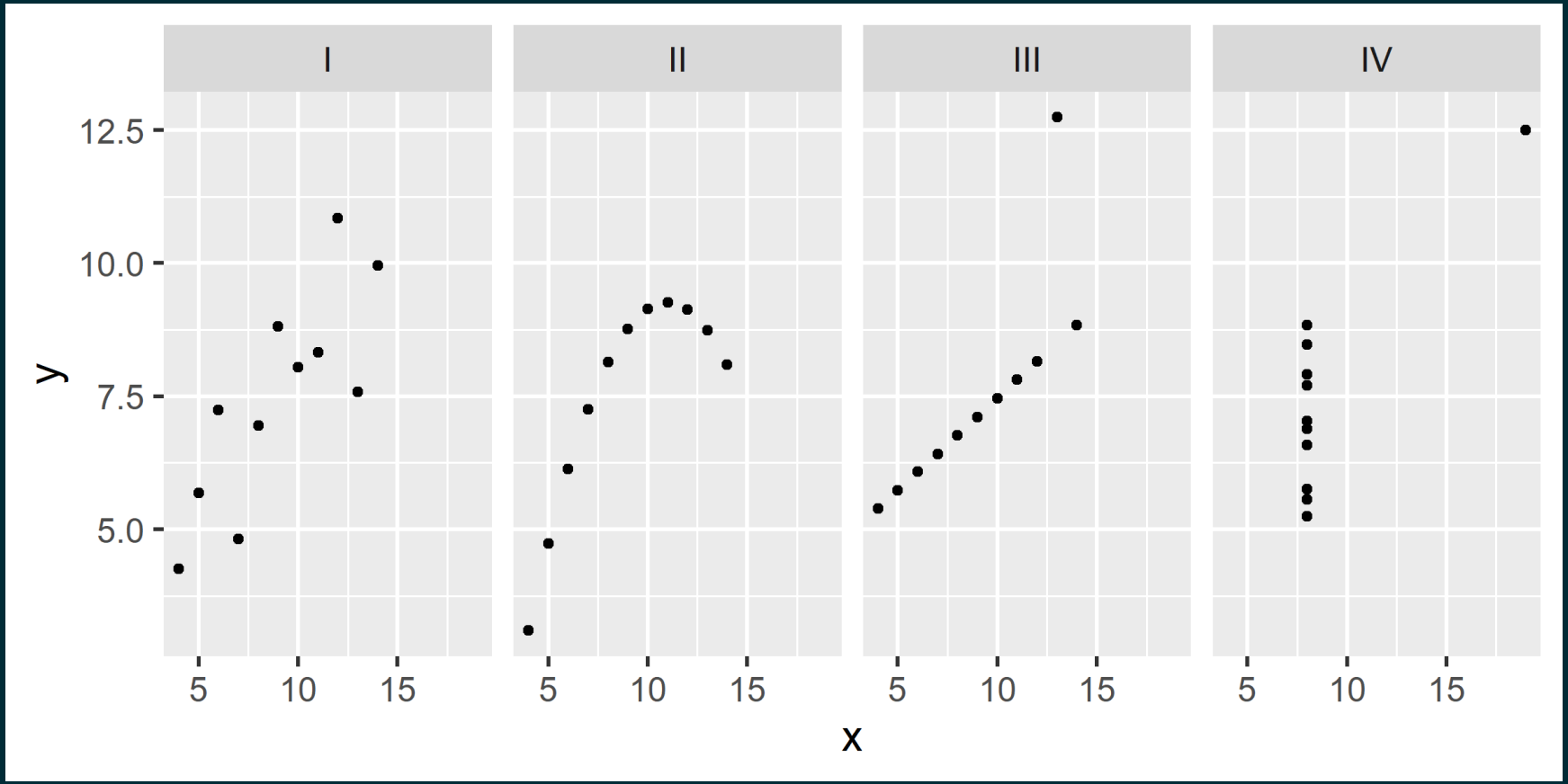
	set	x	y
23	III	10	7.46
24	III	8	6.77
25	III	13	12.74
26	III	9	7.11
27	III	11	7.81
28	III	14	8.84
29	III	6	6.08
30	III	4	5.39
31	III	12	8.15
32	III	7	6.42
33	III	5	5.73
34	IV	8	6.58
35	IV	8	5.76
36	IV	8	7.71
37	IV	8	8.84
38	IV	8	8.47
39	IV	8	7.04
40	IV	8	5.25
41	IV	19	12.50
42	IV	8	5.56
43	IV	8	7.91
44	IV	8	6.89

Summarising Anscombe's quartet

```
quartet %>%
  group_by(set) %>%
  summarise(
    mean_x = mean(x),
    mean_y = mean(y),
    sd_x = sd(x),
    sd_y = sd(y),
    r = cor(x, y)
  )
```

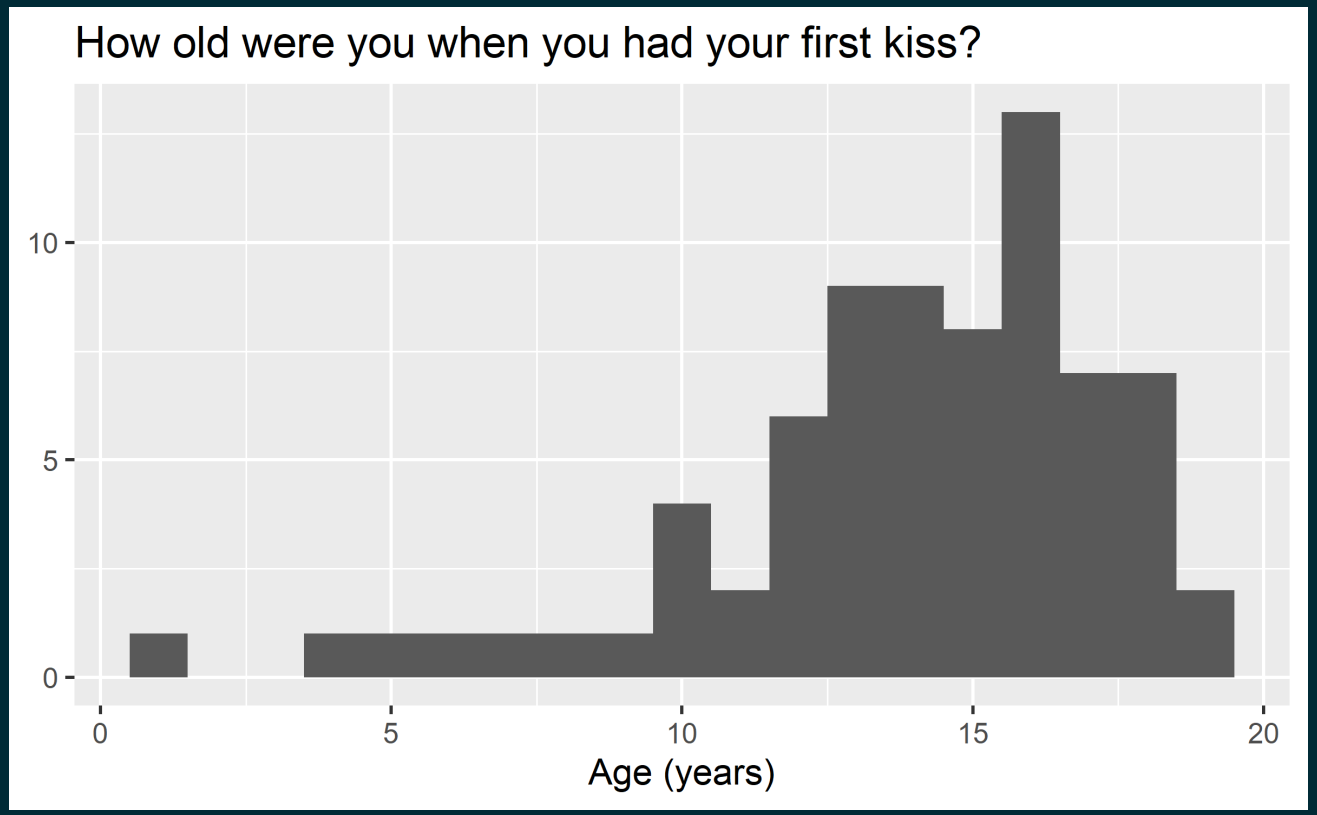
```
## # A tibble: 4 x 6
##   set   mean_x mean_y  sd_x  sd_y     r
##   <fct>  <dbl>  <dbl> <dbl> <dbl> <dbl>
## 1 I          9   7.50  3.32  2.03 0.816
## 2 II         9   7.50  3.32  2.03 0.816
## 3 III        9   7.5   3.32  2.03 0.816
## 4 IV         9   7.50  3.32  2.03 0.817
```

Visualizing Anscombe's quartet



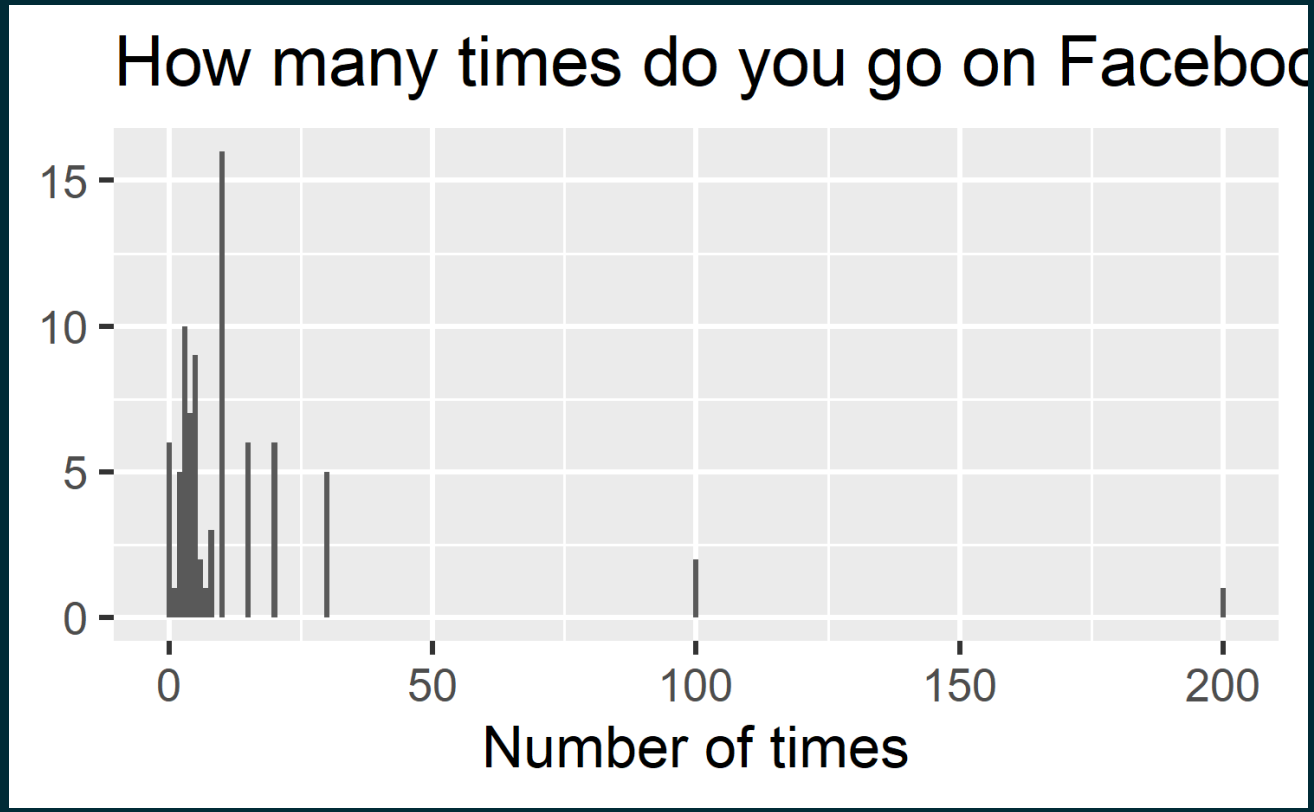
Age at first kiss

Do you see anything out of the ordinary?

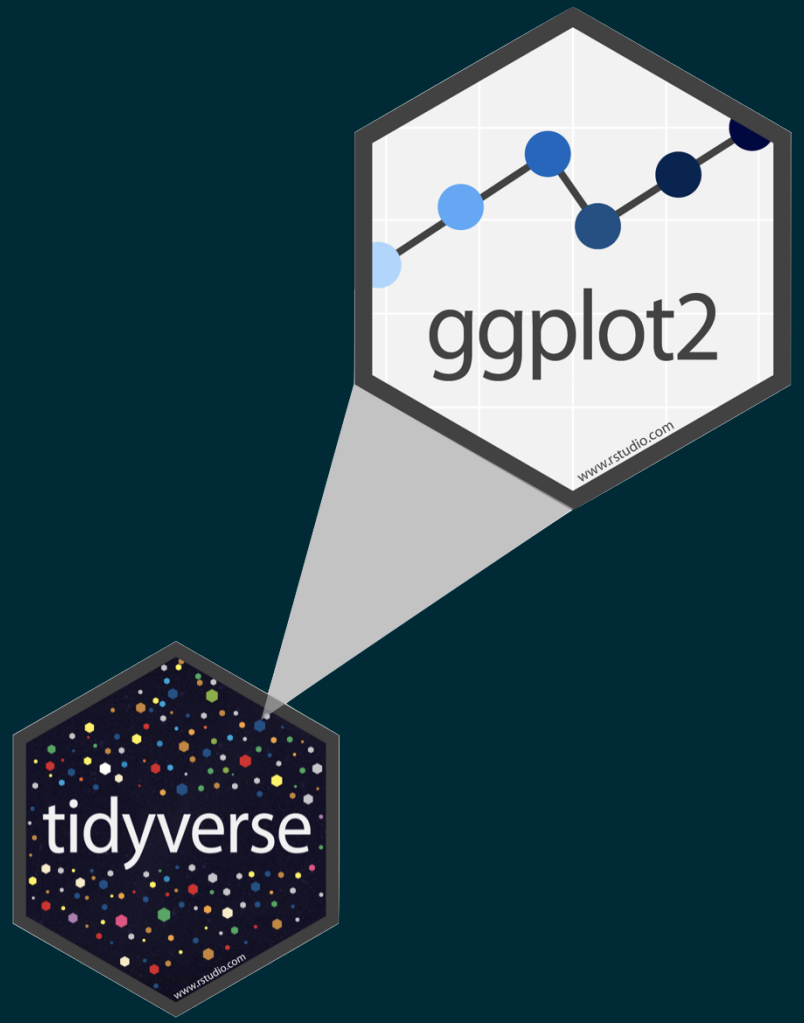


Facebook visits

How are people reporting lower vs. higher values of FB visits?



ggplot2 ∈ tidyverse

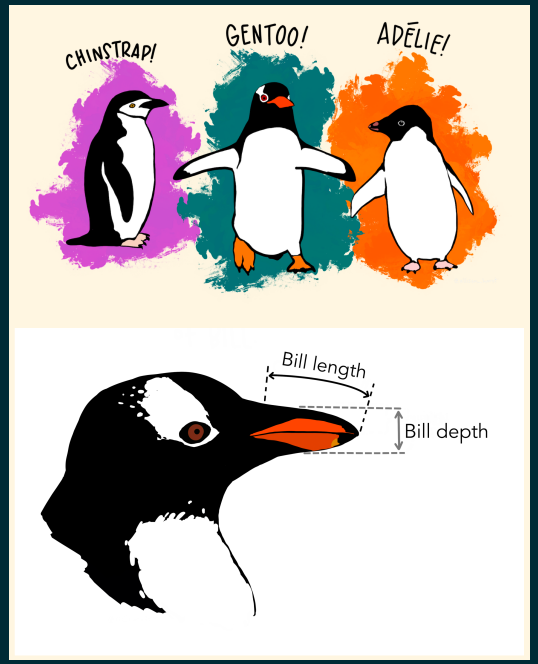


- **ggplot2** is tidyverse's data visualization package
- Structure of the code for plots can be summarized as

```
ggplot(data = [dataset],
       mapping = aes(x = [x-variable],
                     y = [y-variable])) +
   geom_xxx() +
   other options
```

Data: Palmer Penguins

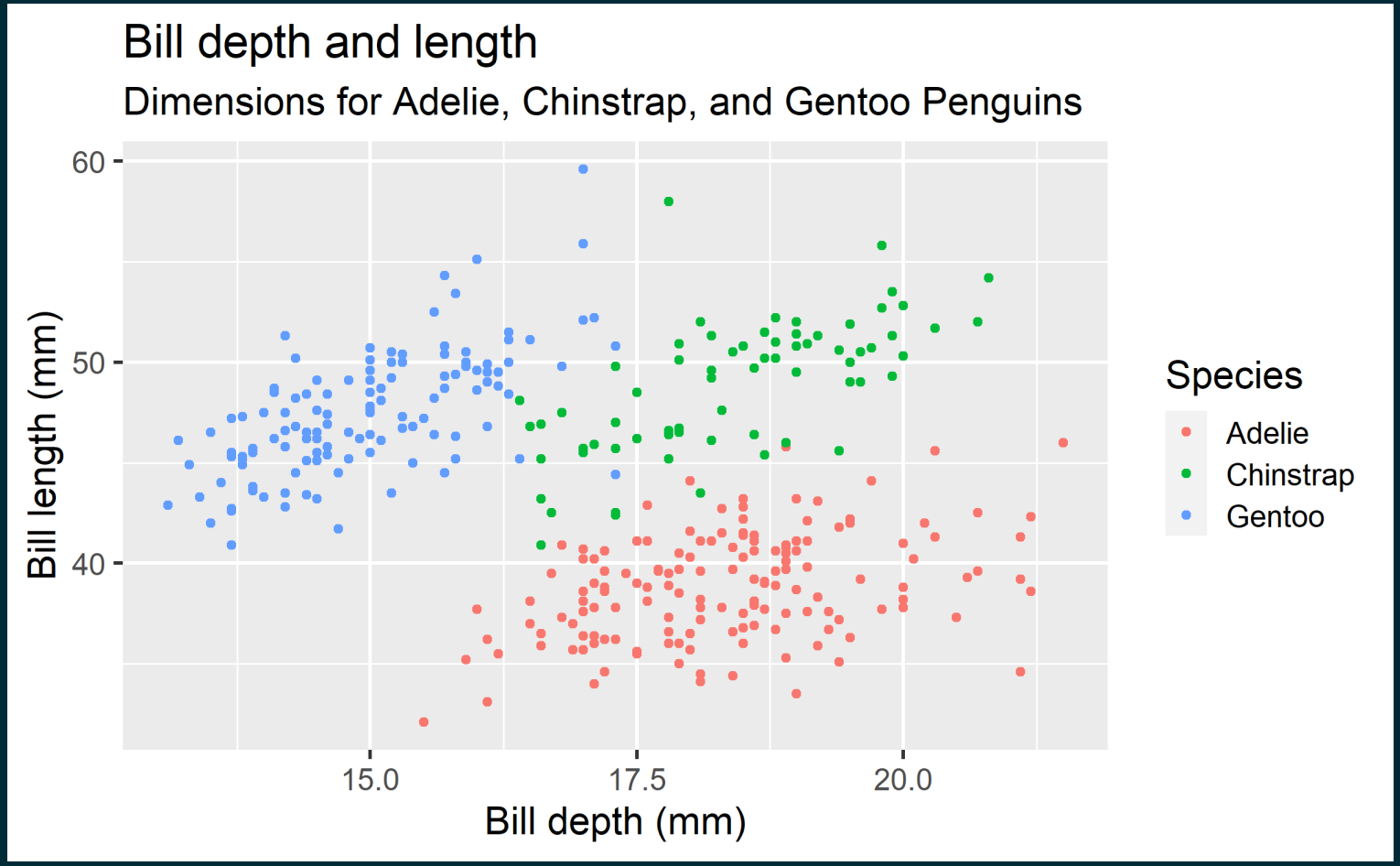
Measurements for penguin species, island in Palmer Archipelago, size (flipper length, body mass, bill dimensions), and sex.




```
library(palmerpenguins)
glimpse(penguins)
```

```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adeli~
## $ island            <fct> Torgersen, Torgersen, Torgersen, Torg~
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.~
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.~
## $ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195~
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 362~
## $ sex               <fct> male, female, female, NA, female, mal~
## $ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2~
```

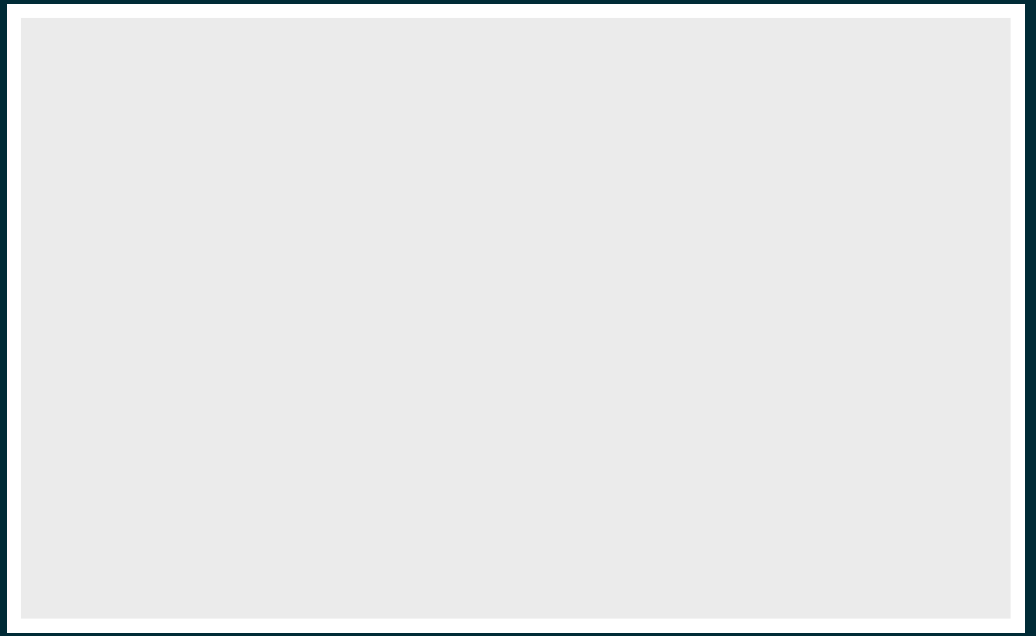
Plot Code


Bill depth and length
Dimensions for Adelie, Chinstrap, and Gentoo Penguins
Bill length (mm)
60
50
40
15.0
17.5
20.0
Bill depth (mm)
Species
Adelie
Chinstrap
Gentoo

Coding out loud

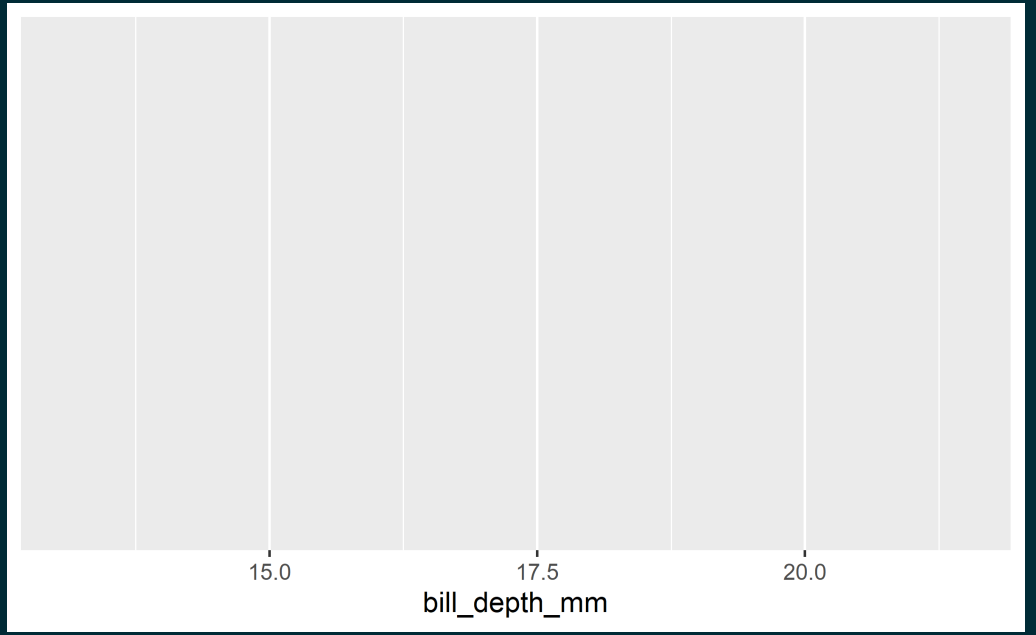
Start with the `penguins` data frame

```
ggplot(data = penguins)
```



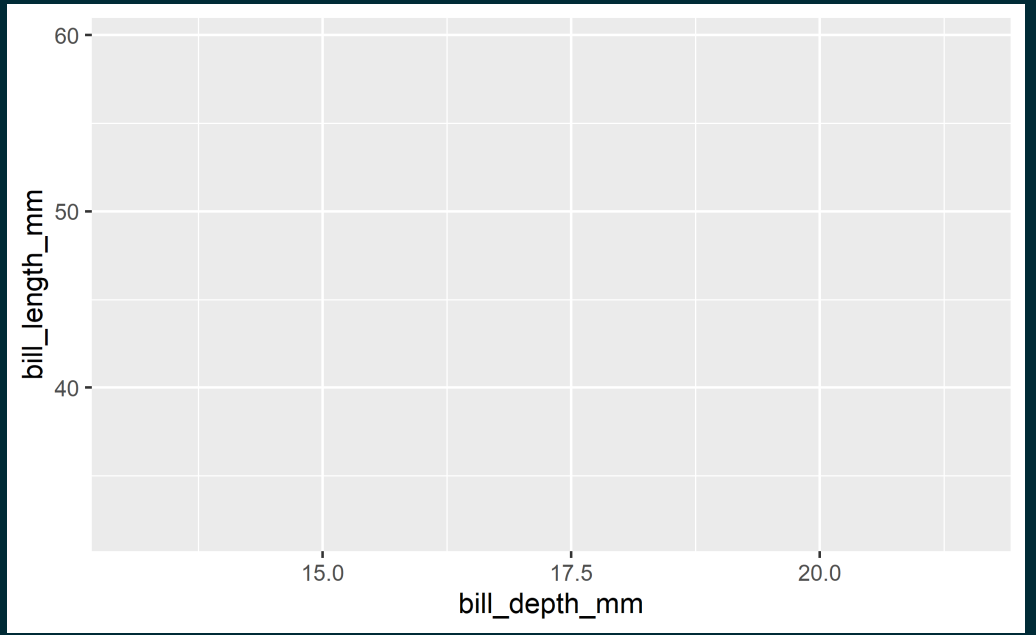
Start with the `penguins` data frame, **map bill depth to the x-axis**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm))
```



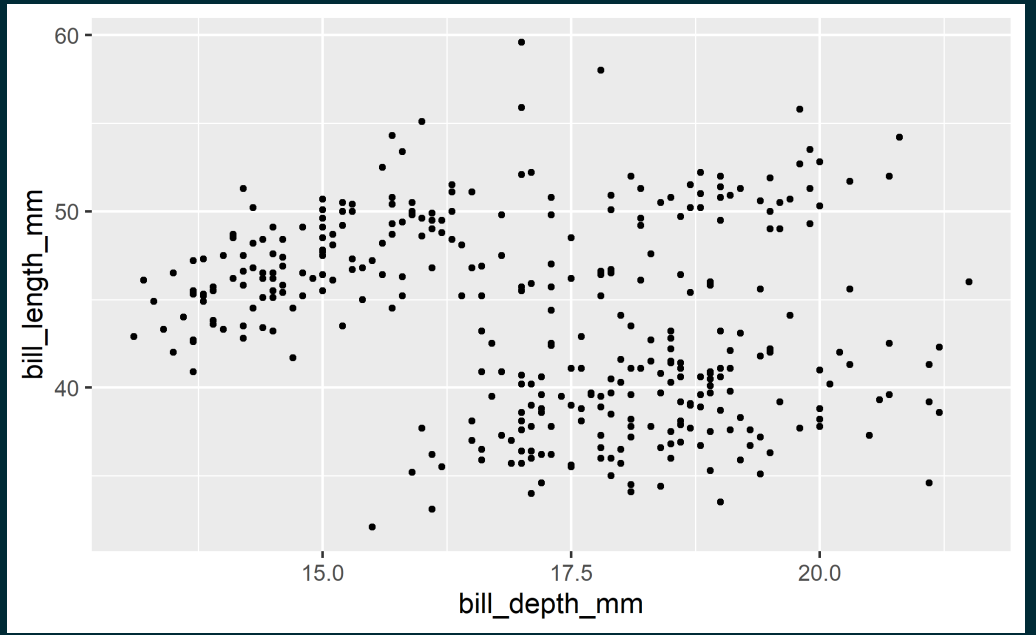
> Start with the `penguins` data frame, map bill depth to the x-axis **and map bill length to the y-axis.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm))
```



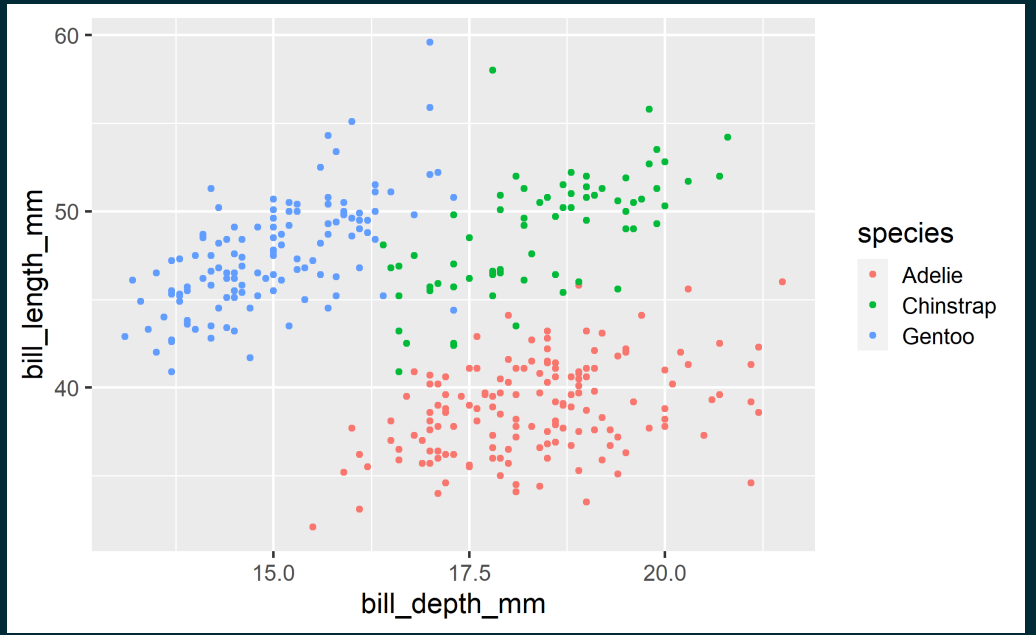
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. **Represent each observation with a point**

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



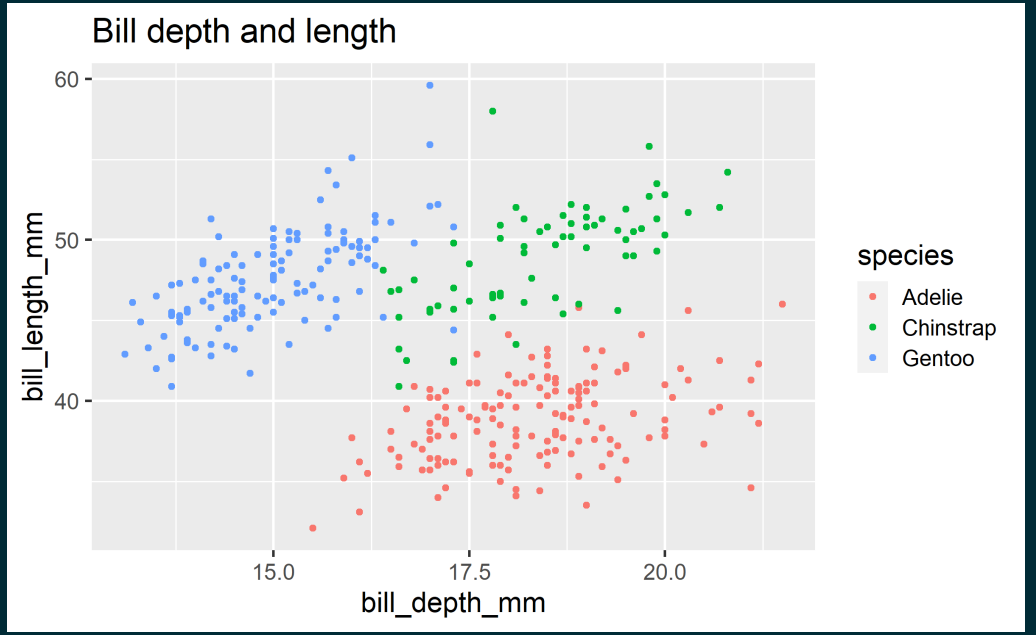
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point **and map species to the colour of each point.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point()
```



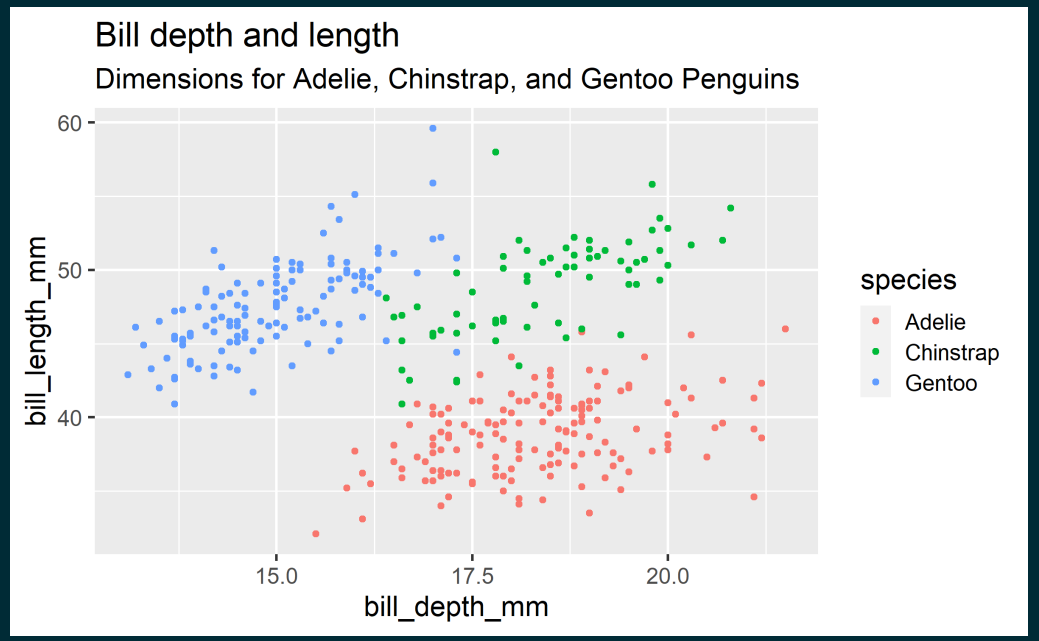
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. **Title the plot "Bill depth and length"**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length")
```



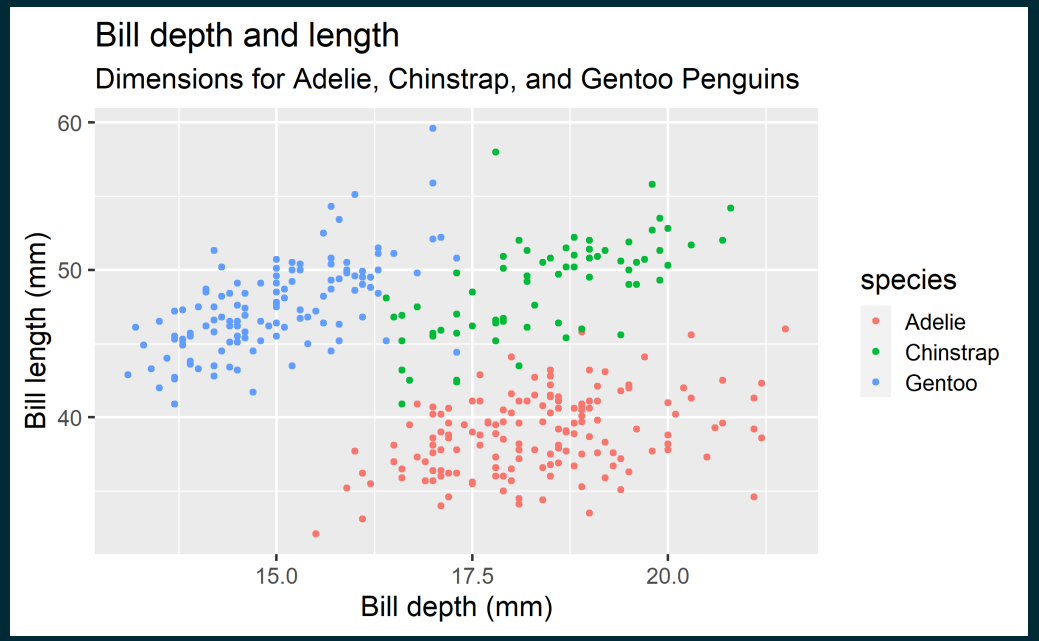
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", **add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins"**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
```



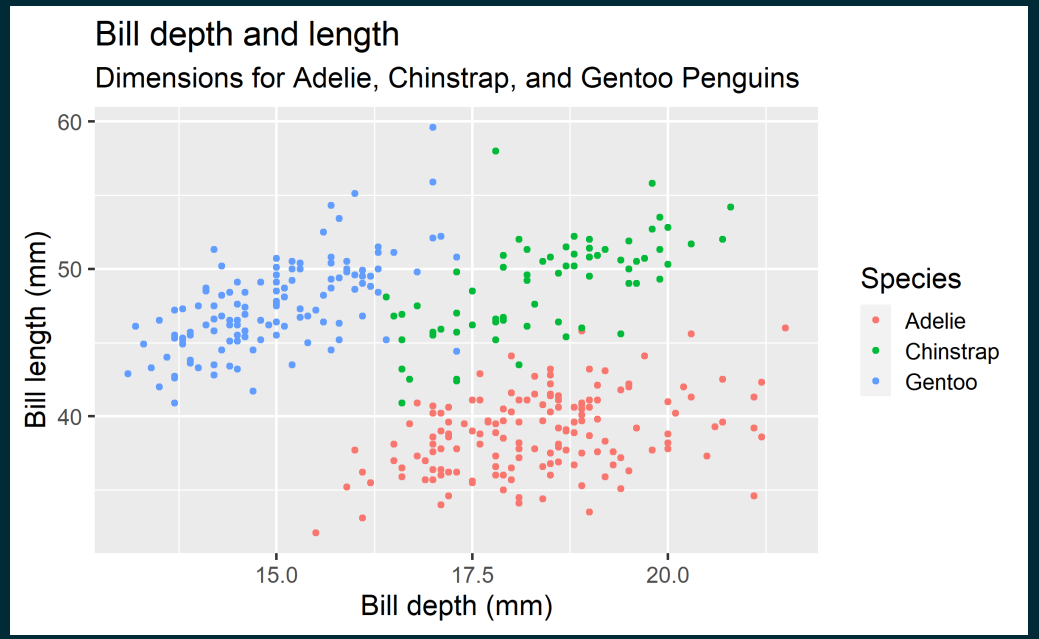
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", **label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
       x = "Bill depth (mm)", y = "Bill lengt
```



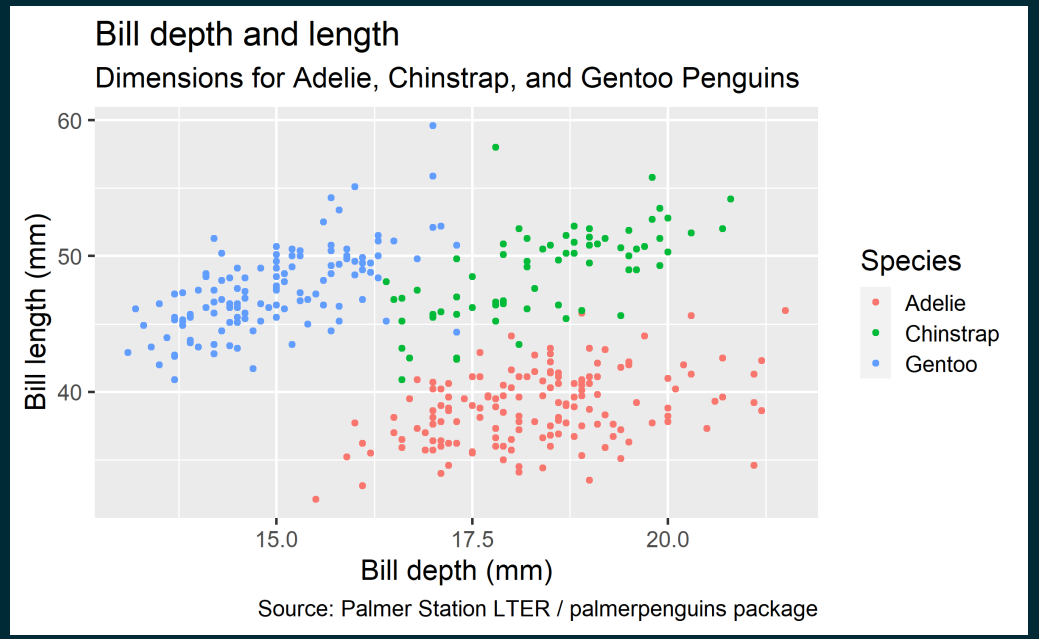
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively, **label the legend "Species"**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
       x = "Bill depth (mm)", y = "Bill lengt
       colour = "Species")
```



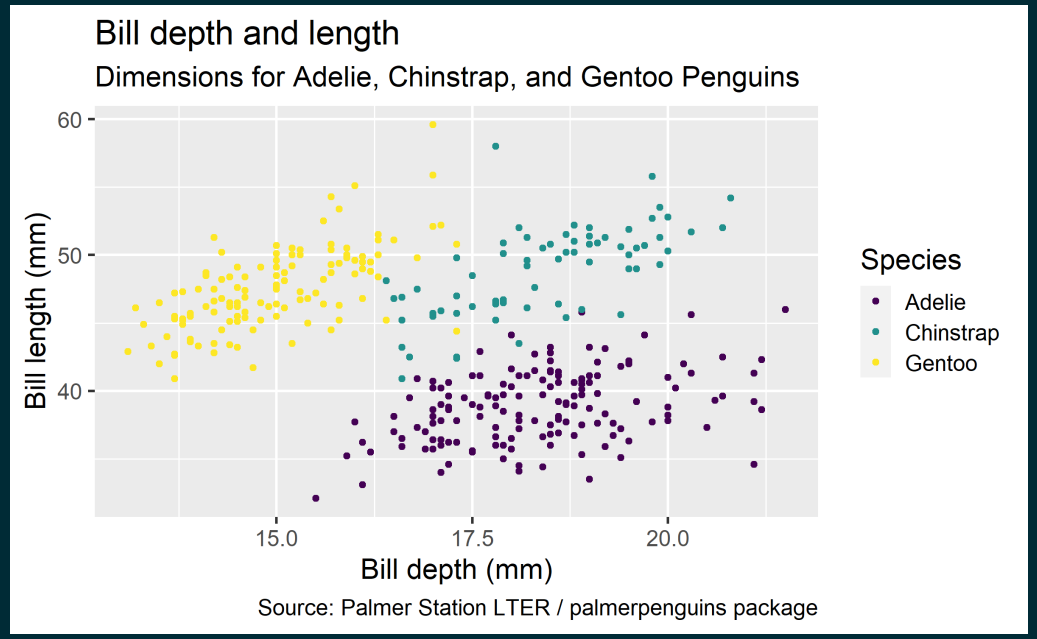
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively, label the legend "Species", **and add a caption for the data source.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
       x = "Bill depth (mm)", y = "Bill lengt
       colour = "Species",
       caption = "Source: Palmer Station LTER
```

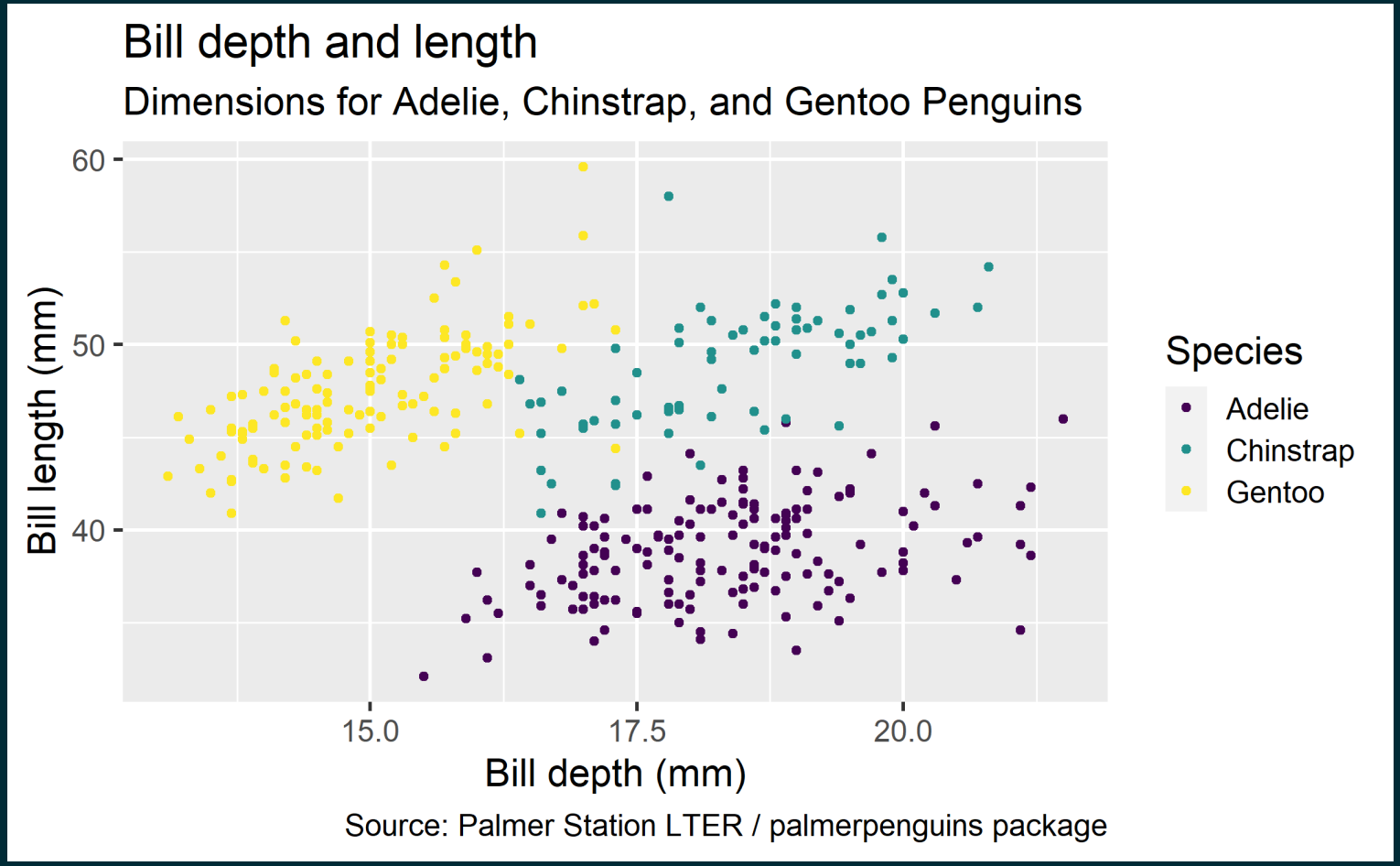


> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively, label the legend "Species", and add a caption for the data source. **Finally, use a discrete colour scale that is designed to be perceived by viewers with common forms of colour blindness.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
       x = "Bill depth (mm)", y = "Bill lengt
       colour = "Species",
       caption = "Source: Palmer Station LTER
  scale_colour_viridis_d()
```



Plot Code Narrative



Argument names

You can omit the names of first two arguments when building plots with `ggplot()`.

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```

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

Aesthetics

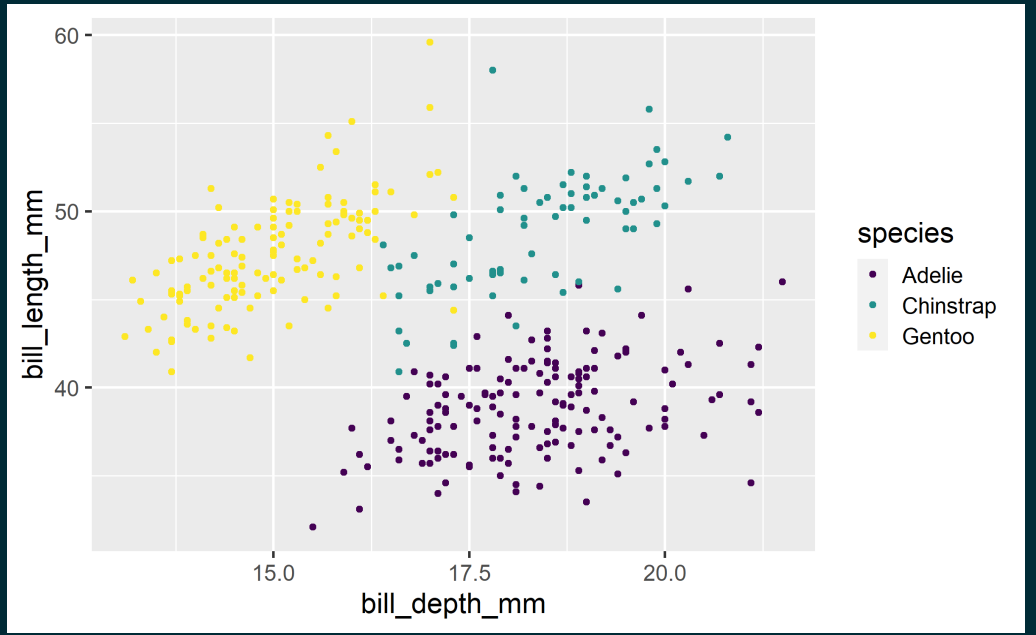
Aesthetics options

Commonly used characteristics of plotting characters that can be **mapped to a specific variable** in the data are

- `colour`
- `shape`
- `size`
- `alpha` (transparency)

Colour

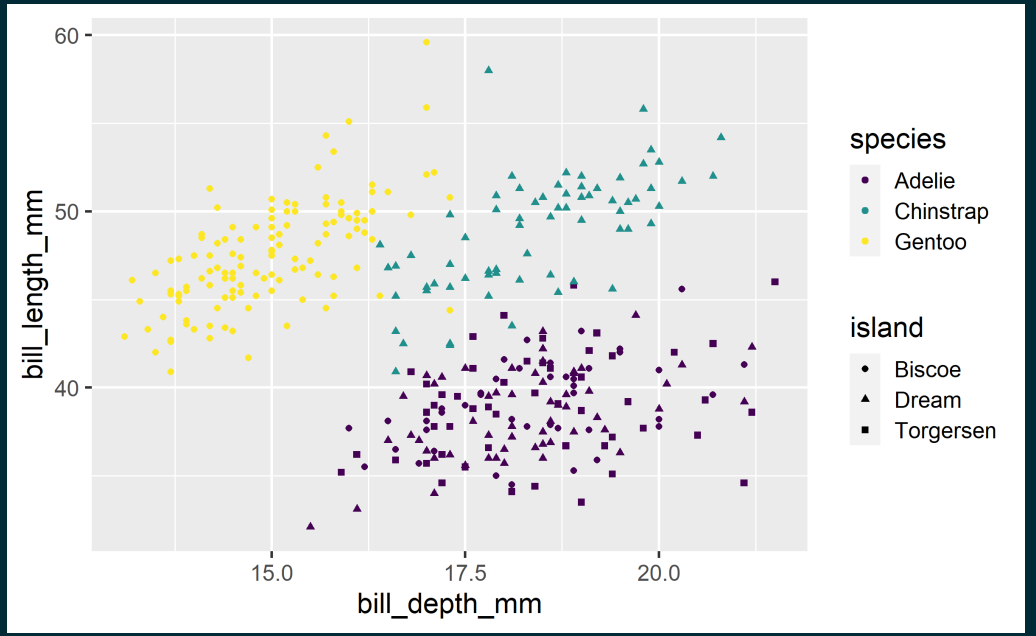
```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```



Shape

Mapped to a different variable than `colour`

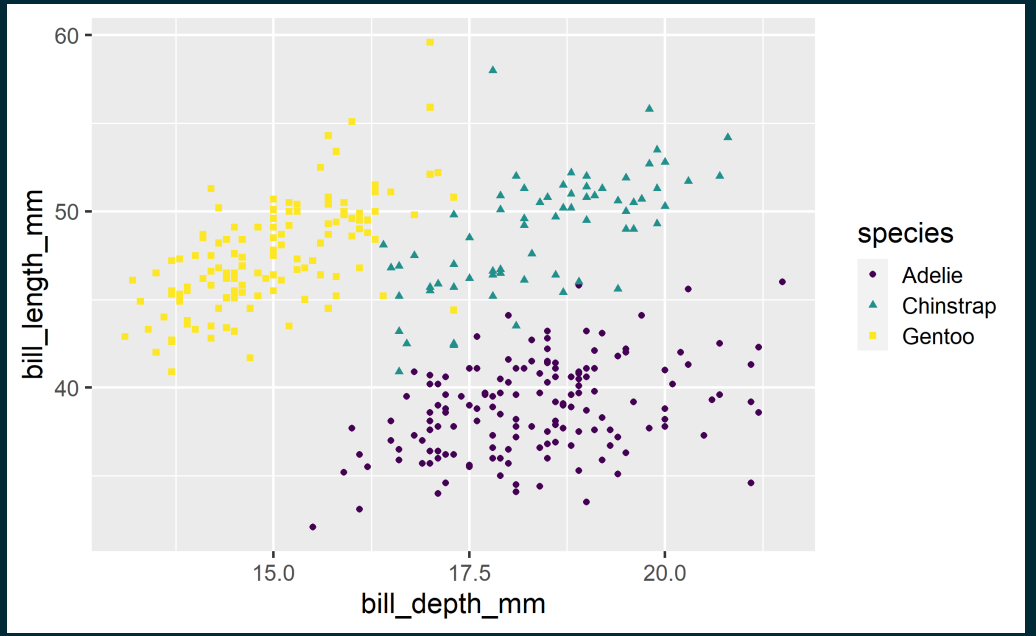
```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = island)) +
  geom_point() +
  scale_colour_viridis_d()
```



Shape

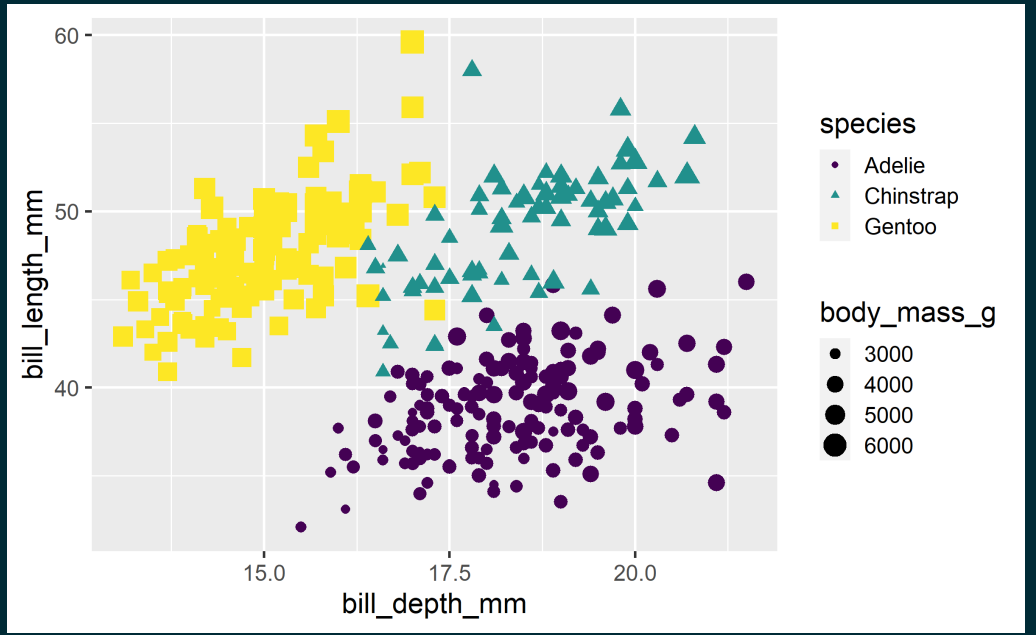
Mapped to same variable as `colour`

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = species)) +
  geom_point() +
  scale_colour_viridis_d()
```



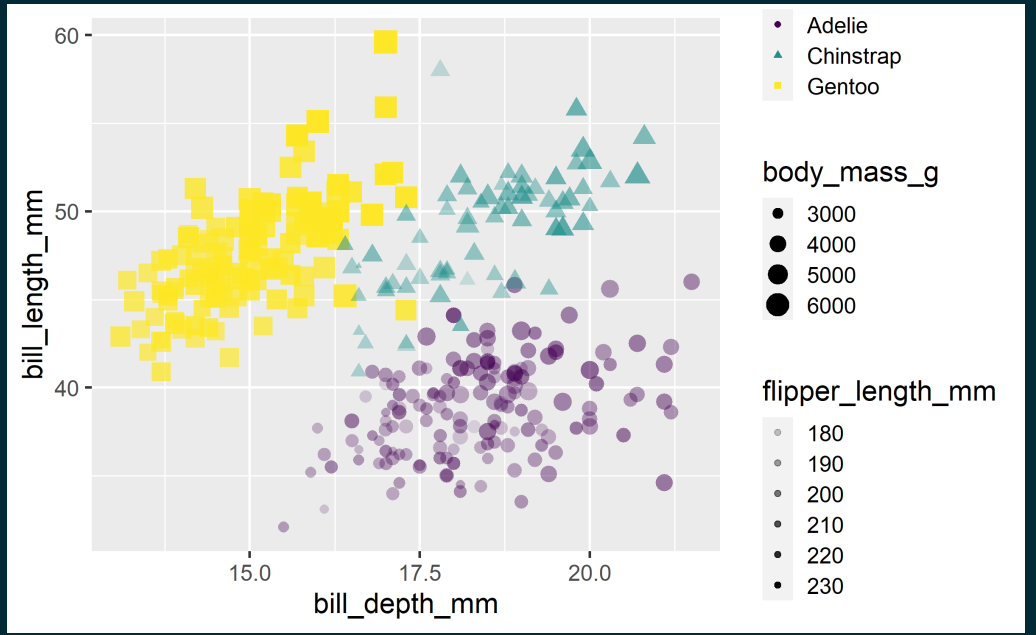
Size

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = species,
           size = body_mass_g)) +
  geom_point() +
  scale_colour_viridis_d()
```



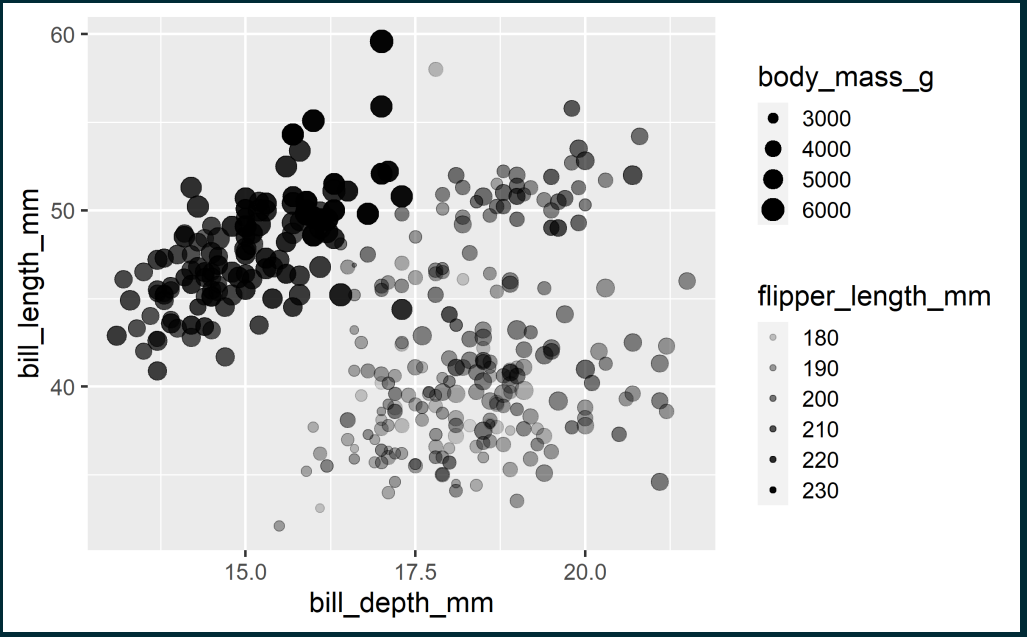
Alpha

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = species,
           size = body_mass_g,
           alpha = flipper_length_mm)) +
  geom_point() +
  scale_colour_viridis_d()
```



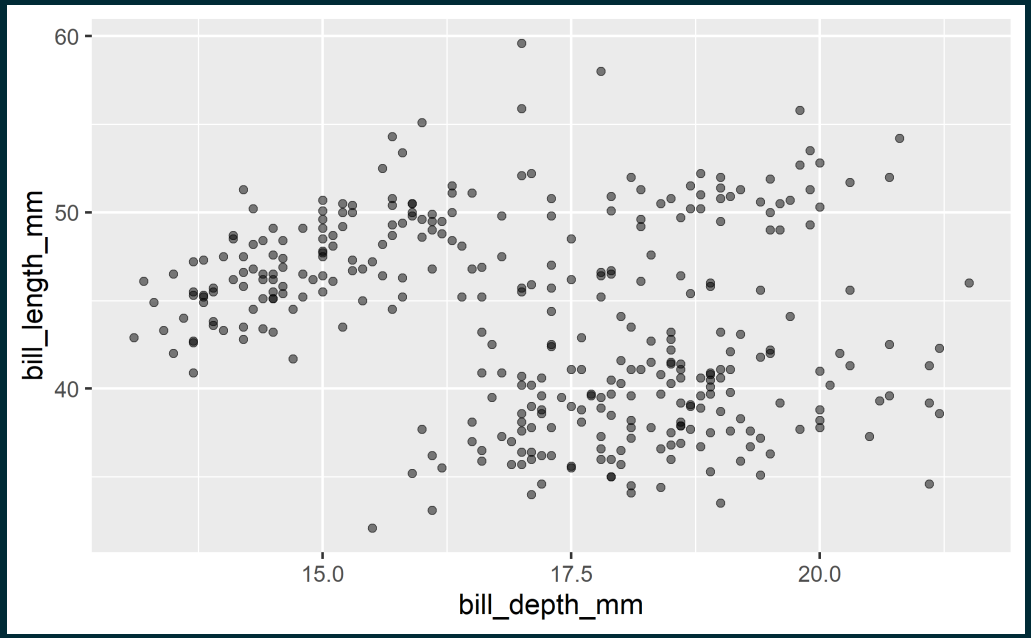
Mapping

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           size = body_mass_g,
           alpha = flipper_length_mm)) +
  geom_point()
```



Setting

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm)) +
  geom_point(size = 2, alpha = 0.5)
```



Mapping vs. setting

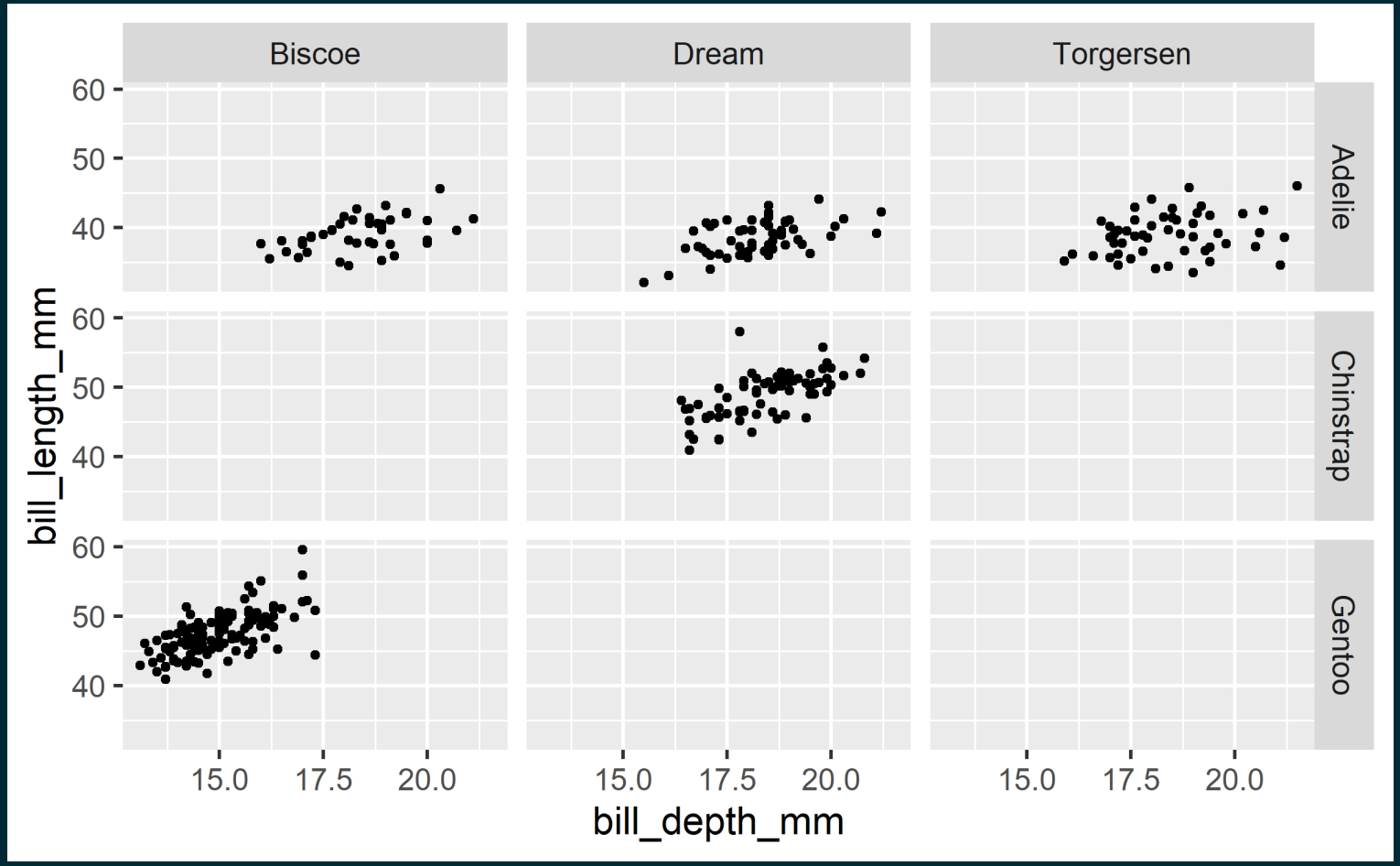
- **Mapping:** Determine the size, alpha, etc. of points based on the values of a variable in the data
  - goes into `aes()`
- **Setting:** Determine the size, alpha, etc. of points **not** based on the values of a variable in the data
  - goes into `geom_*()` (this was `geom_point()` in the previous example, but we'll learn about other geoms soon!)

Faceting

Faceting

- Smaller plots that display different subsets of the data
- Useful for exploring conditional relationships and large data

Plot Code

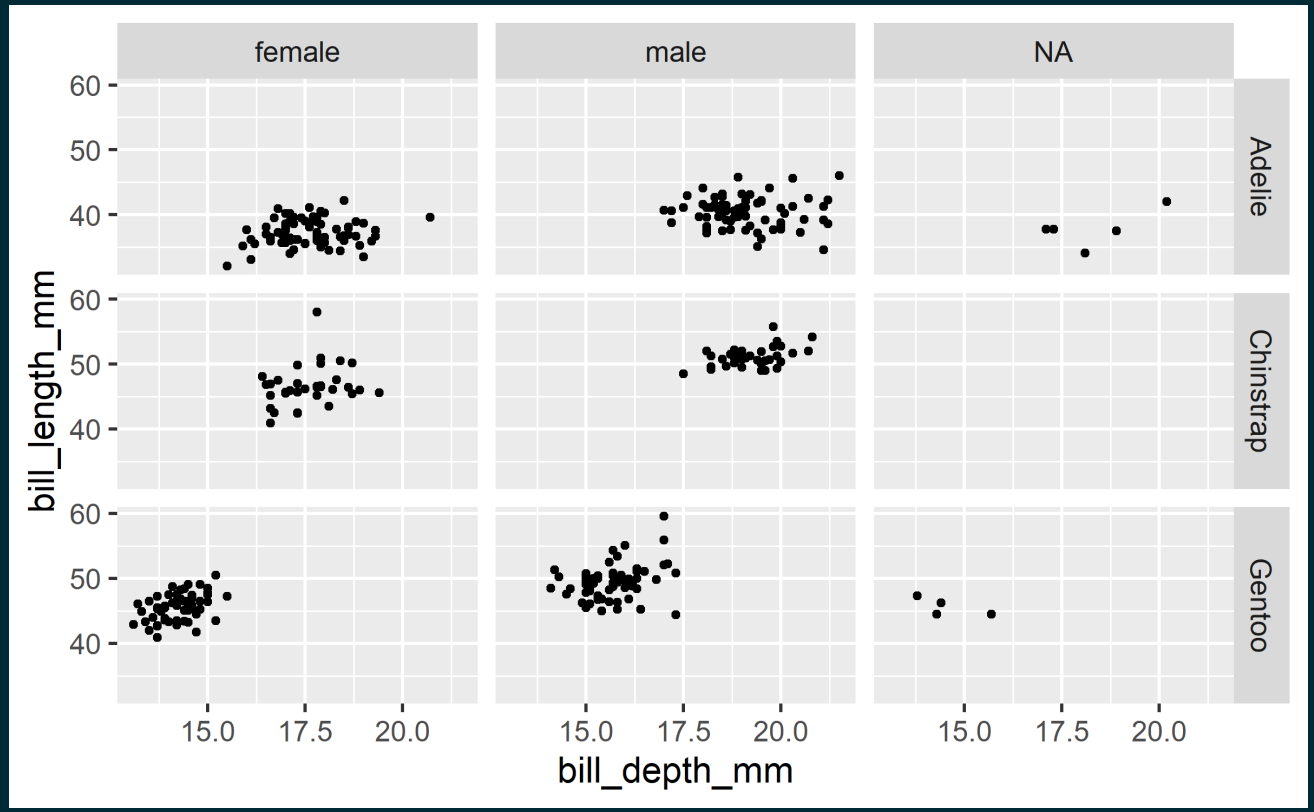

Biscoe
Dream
Torgersen
Adelie
Chinstrap
Gentoo
bill_length_mm
bill_depth_mm
60
50
40
15.0
17.5
20.0

Various ways to facet

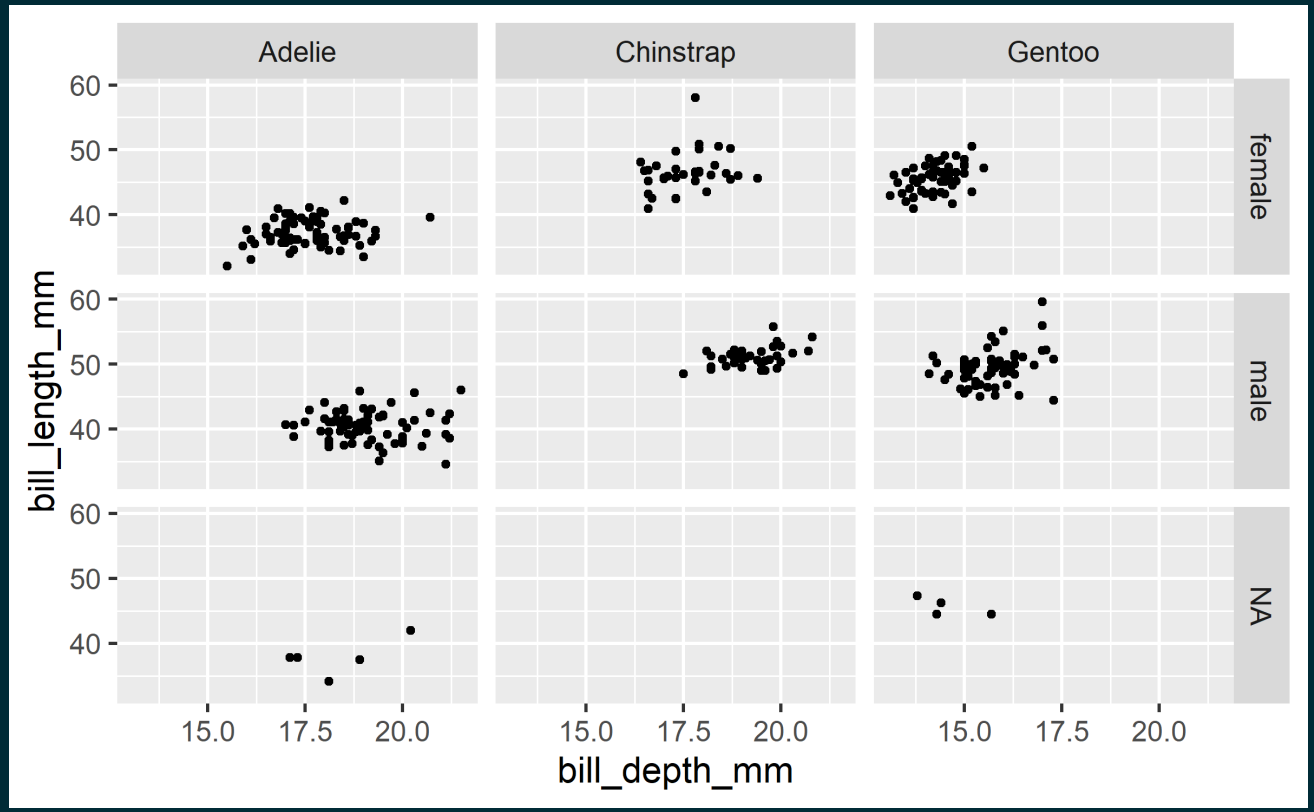
In the next few slides describe what each plot displays. Think about how the code relates to the output.

Note: The plots in the next few slides do not have proper titles, axis labels, etc. because we want you to figure out what's happening in the plots. But you should always label your plots!

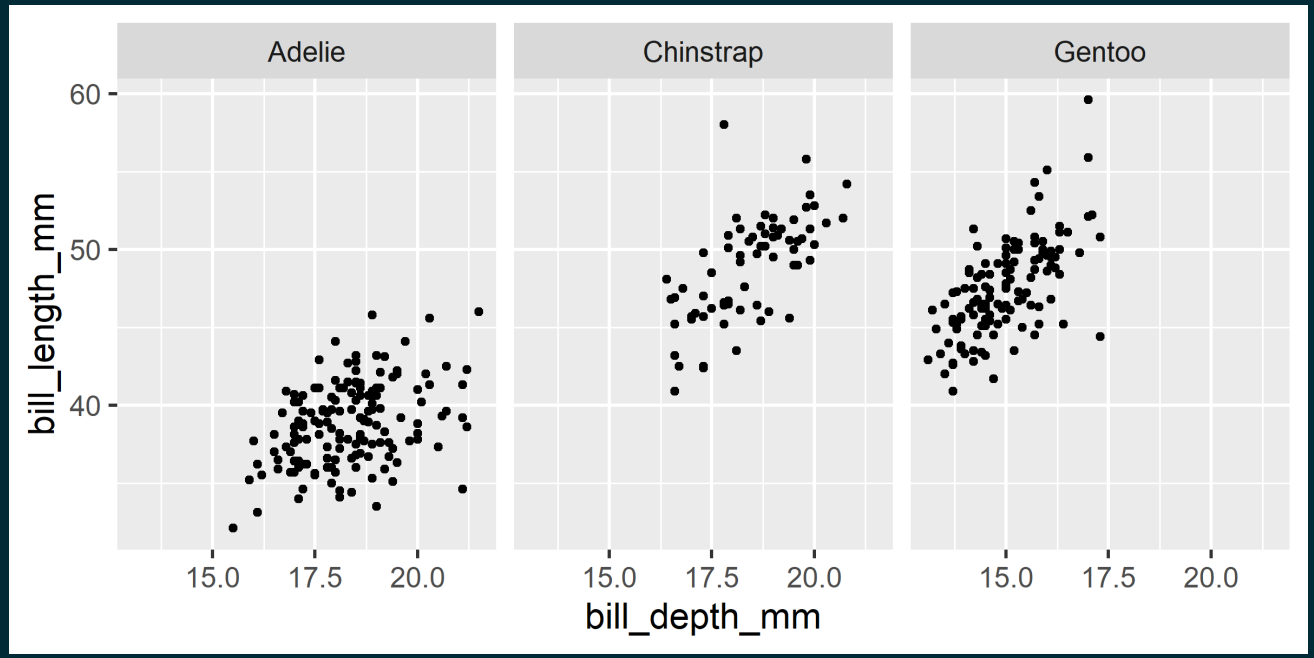
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(species ~ sex)
```



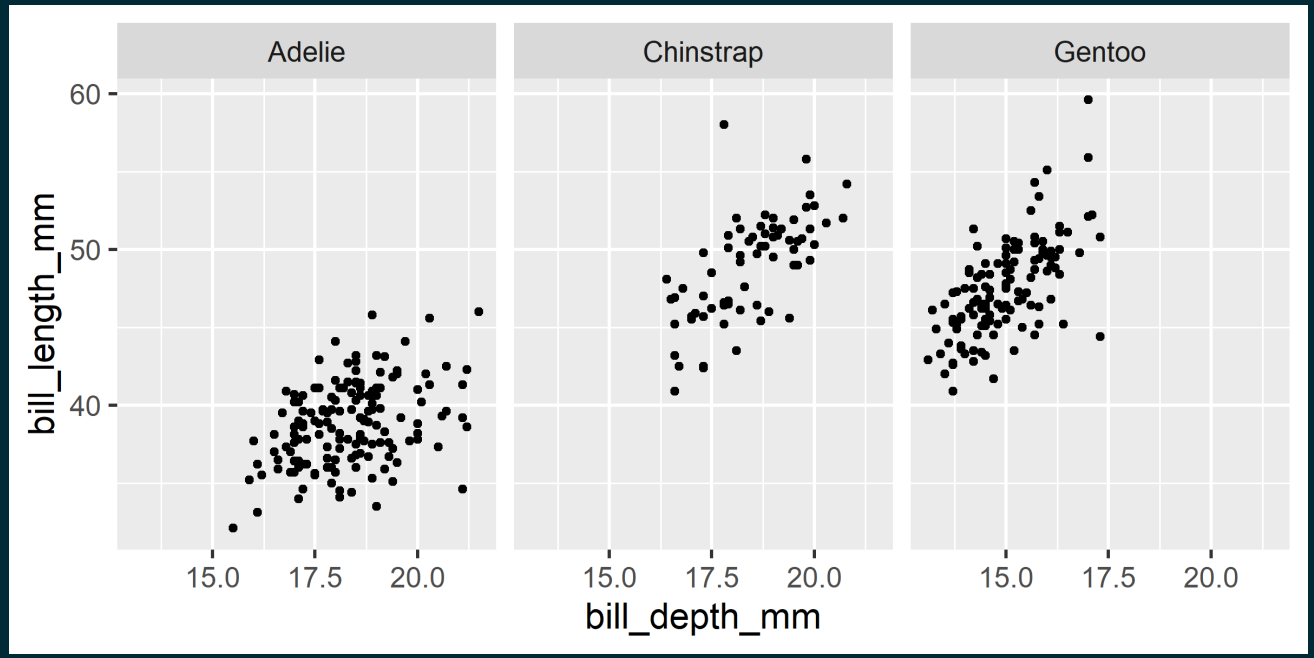
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(sex ~ species)
```



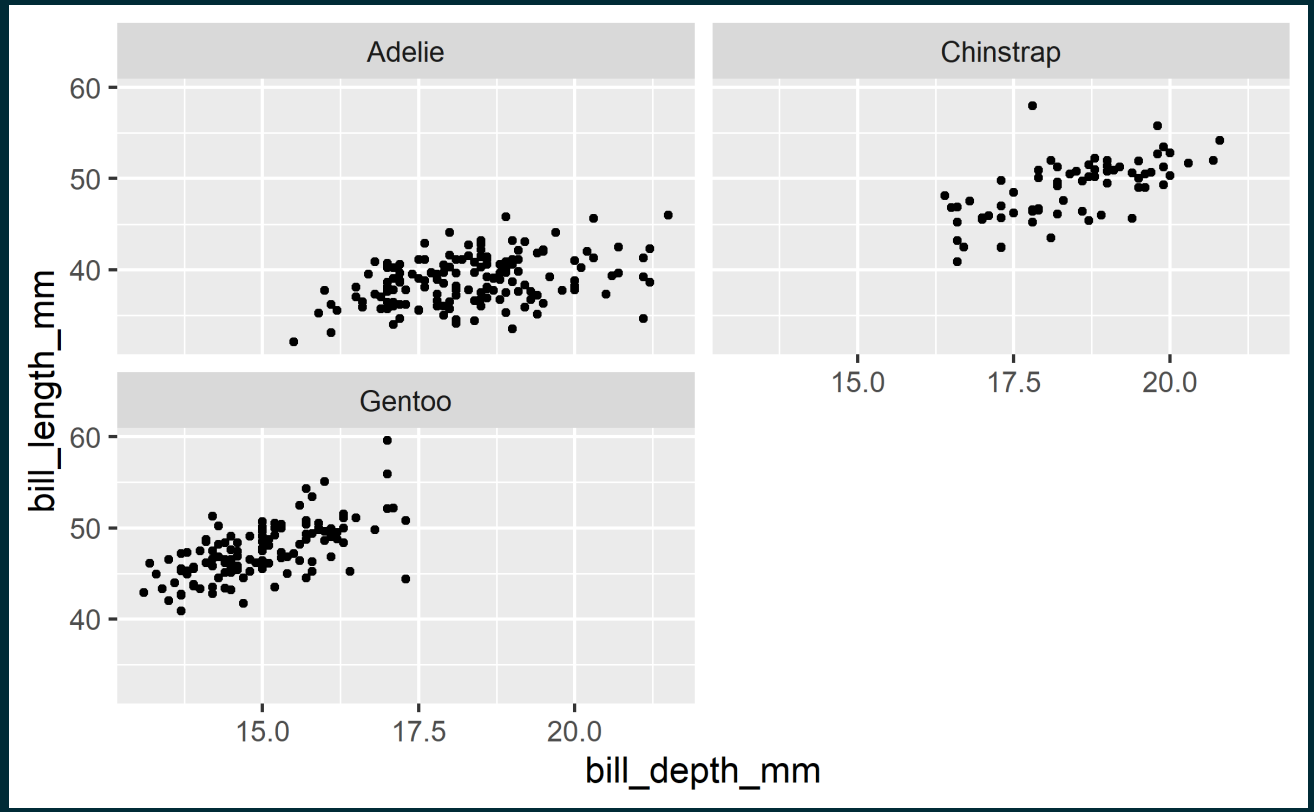
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_wrap(~ species)
```



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



```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_wrap(~ species, ncol = 2)
```

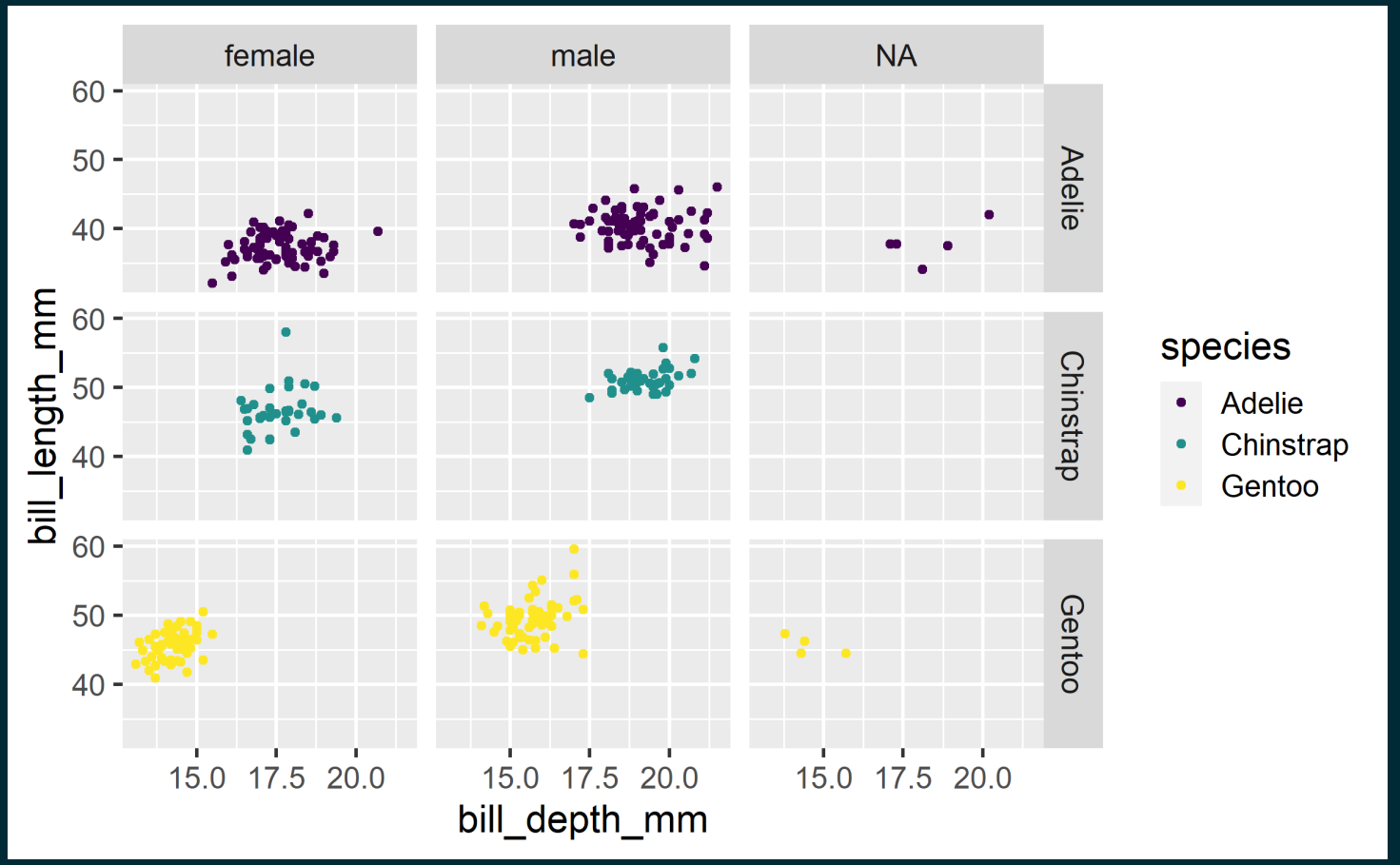


Faceting summary

- `facet_grid()`:
  - 2d grid
  - `rows ~ cols`
  - use `.` for no split
- `facet_wrap()`: 1d ribbon wrapped according to number of rows and columns specified or available plotting area

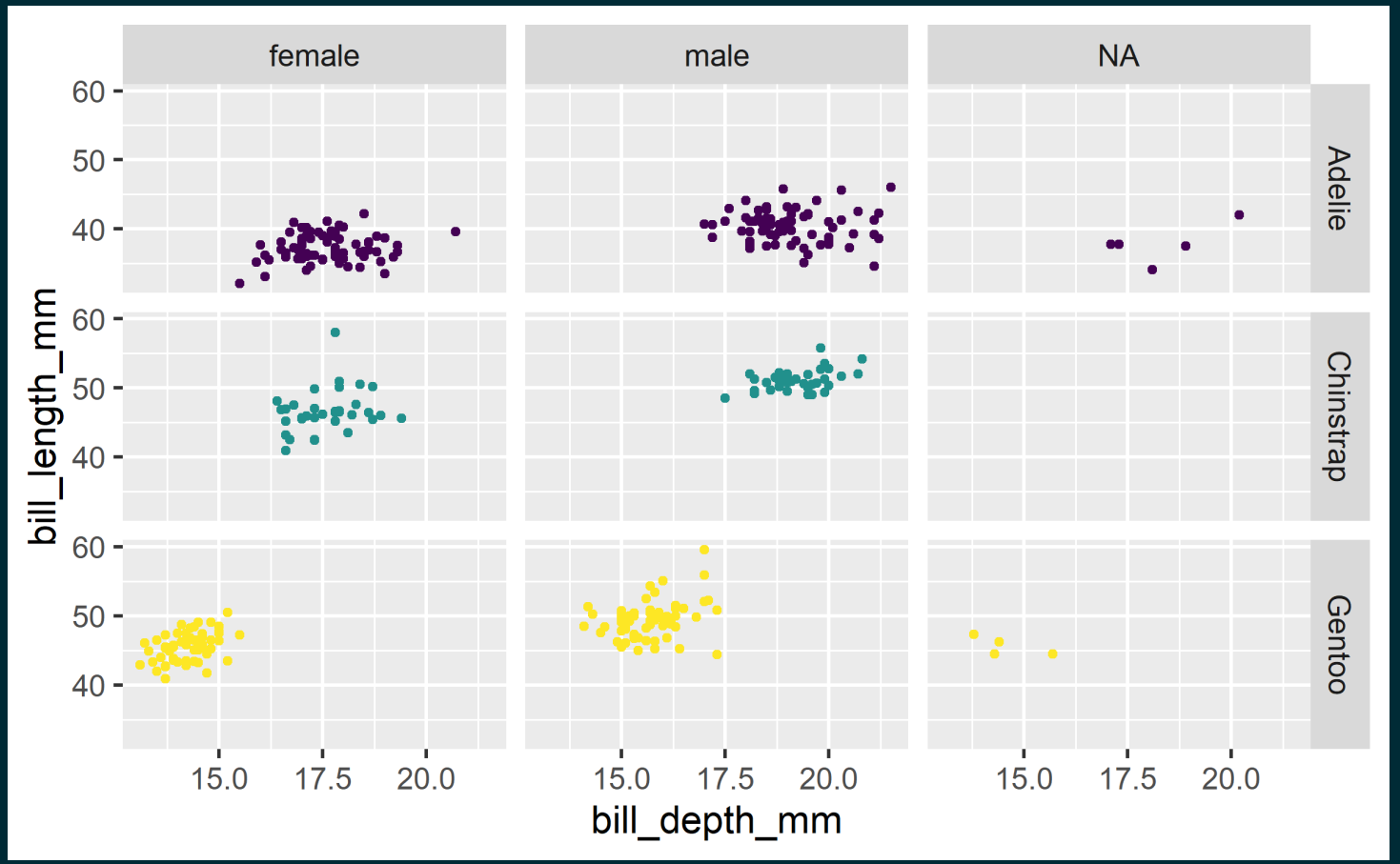
Facet and color

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



Face and color, no legend

```
ggplot(
  penguins,
  aes(x = bill_depth_mm,
      y = bill_length_mm,
      color = species)) +
  geom_point() +
  facet_grid(species ~ sex) +
  scale_color_viridis_d() +
  guides(color = FALSE)
```



Number of variables involved

- Univariate data analysis - distribution of single variable
- Bivariate data analysis - relationship between two variables
- Multivariate data analysis - relationship between many variables at once, usually focusing on the relationship between two while conditioning for others

Types of variables

- **Numerical variables** can be classified as **continuous** or **discrete** based on whether or not the variable can take on an infinite number of values or only non-negative whole numbers, respectively.
- If the variable is **categorical**, we can determine if it is **ordinal** based on whether or not the levels have a natural ordering.

Data

Data: Lending Club

- Thousands of loans made through the Lending Club, which is a platform that allows individuals to lend to other individuals
- Not all loans are created equal -- ease of getting a loan depends on (apparent) ability to pay back the loan
- Data includes loans *made*, these are not loan applications

Take a peek at data

```
library(openintro)
glimpse(loans_full_schema)
```

```
## Rows: 10,000
## Columns: 55
## $ emp_title                    <chr> "global config enginee~
## $ emp_length                   <dbl> 3, 10, 3, 1, 10, NA, 1~
## $ state                        <fct> NJ, HI, WI, PA, CA, KY~
## $ homeownership                <fct> MORTGAGE, RENT, RENT, ~
## $ annual_income                <dbl> 90000, 40000, 40000, 3~
## $ verified_income              <fct> Verified, Not Verified~
## $ debt_to_income               <dbl> 18.01, 5.04, 21.15, 10~
## $ annual_income_joint          <dbl> NA, NA, NA, NA, 57000,~
## $ verification_income_joint    <fct> , , , , Verified, , No~
## $ debt_to_income_joint         <dbl> NA, NA, NA, NA, 37.66,~
## $ delinq_2y                    <int> 0, 0, 0, 0, 0, 1, 0, 1~
## $ months_since_last_delinq     <int> 38, NA, 28, NA, NA, 3,~
## $ earliest_credit_line         <dbl> 2001, 1996, 2006, 2007~
## $ inquiries_last_12m           <int> 6, 1, 4, 0, 7, 6, 1, 1~
## $ total_credit_lines           <int> 28, 30, 31, 4, 22, 32,~
## $ open_credit_lines            <int> 10, 14, 10, 4, 16, 12,~
```

Selected variables

```
loans <- loans_full_schema %>%
  select(loan_amount, interest_rate, term, grade,
         state, annual_income, homeownership, debt_to_income)
glimpse(loans)
```

```
## Rows: 10,000
## Columns: 8
## $ loan_amount    <int> 28000, 5000, 2000, 21600, 23000, 5000, 2~
## $ interest_rate  <dbl> 14.07, 12.61, 17.09, 6.72, 14.07, 6.72, ~
## $ term           <dbl> 60, 36, 36, 36, 36, 36, 60, 60, 36, 36, ~
## $ grade          <ord> C, C, D, A, C, A, C, B, C, A, C, B, C, B~
## $ state          <fct> NJ, HI, WI, PA, CA, KY, MI, AZ, NV, IL, ~
## $ annual_income  <dbl> 90000, 40000, 40000, 30000, 35000, 34000~
## $ homeownership  <fct> MORTGAGE, RENT, RENT, RENT, RENT, OWN, M~
## $ debt_to_income <dbl> 18.01, 5.04, 21.15, 10.16, 57.96, 6.46, ~
```

Selected variables

variable	description
`loan_amount`	Amount of the loan received, in US dollars
`interest_rate`	Interest rate on the loan, in an annual percentage
`term`	The length of the loan, which is always set as a whole number of months
`grade`	Loan grade, which takes a values A through G and represents the quality of the loan and its likelihood of being repaid
`state`	US state where the borrower resides
`annual_income`	Borrower's annual income, including any second income, in US dollars
`homeownership`	Indicates whether the person owns, owns but has a mortgage, or rents
`debt_to_income`	Debt-to-income ratio

Variable types

variable	type
loan_amount	numerical, continuous
interest_rate	numerical, continuous
term	numerical, discrete
grade	categorical, ordinal
state	categorical, not ordinal
annual_income	numerical, continuous
homeownership	categorical, not ordinal
debt_to_income	numerical, continuous

Visualizing numerical data

Describing shapes of numerical distributions

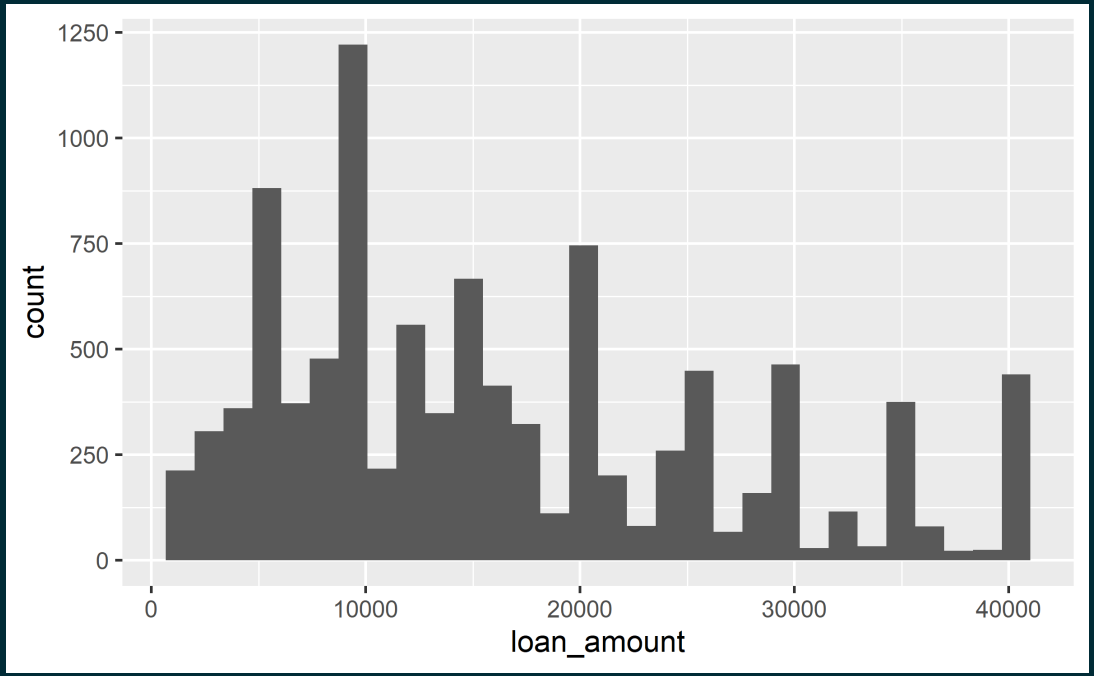
- shape:
  - skewness: right-skewed, left-skewed, symmetric (skew is to the side of the longer tail)
  - modality: unimodal, bimodal, multimodal, uniform
- center: mean (`mean`), median (`median`), mode (not always useful)
- spread: range (`range`), standard deviation (`sd`), inter-quartile range (`IQR`)
- unusual observations

Histogram

Histogram

```
ggplot(loans, aes(x = loan_amount)) +
  geom_histogram()
```

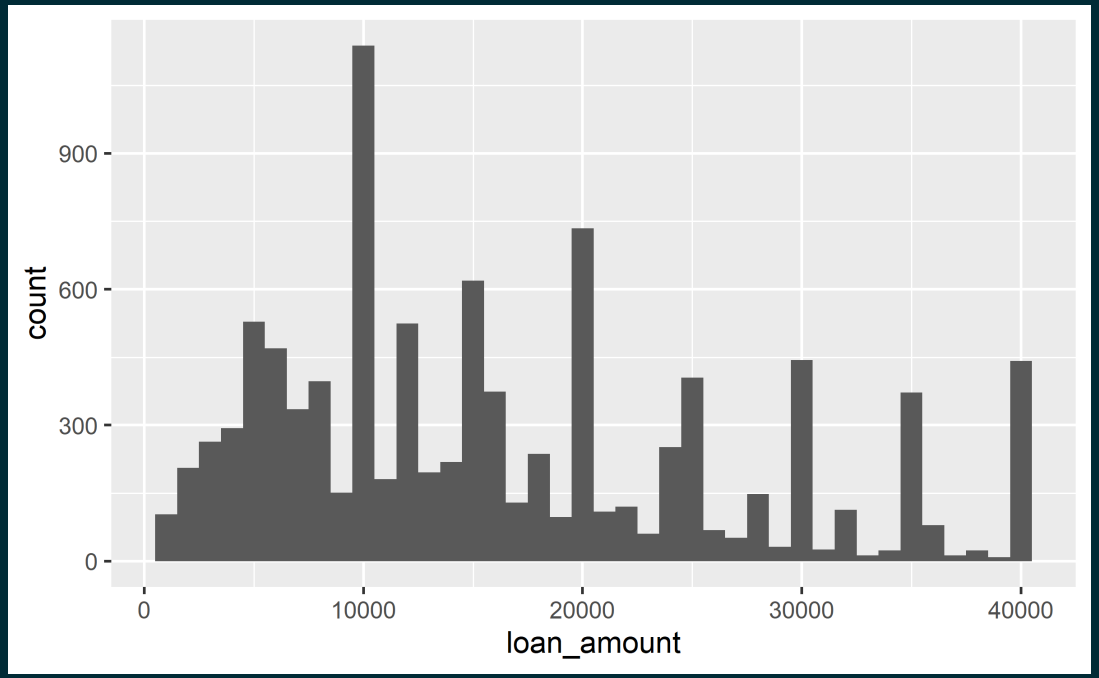
```
## `stat_bin()` using `bins = 30`. Pick better value with
## `binwidth`.
```



Histograms and binwidth

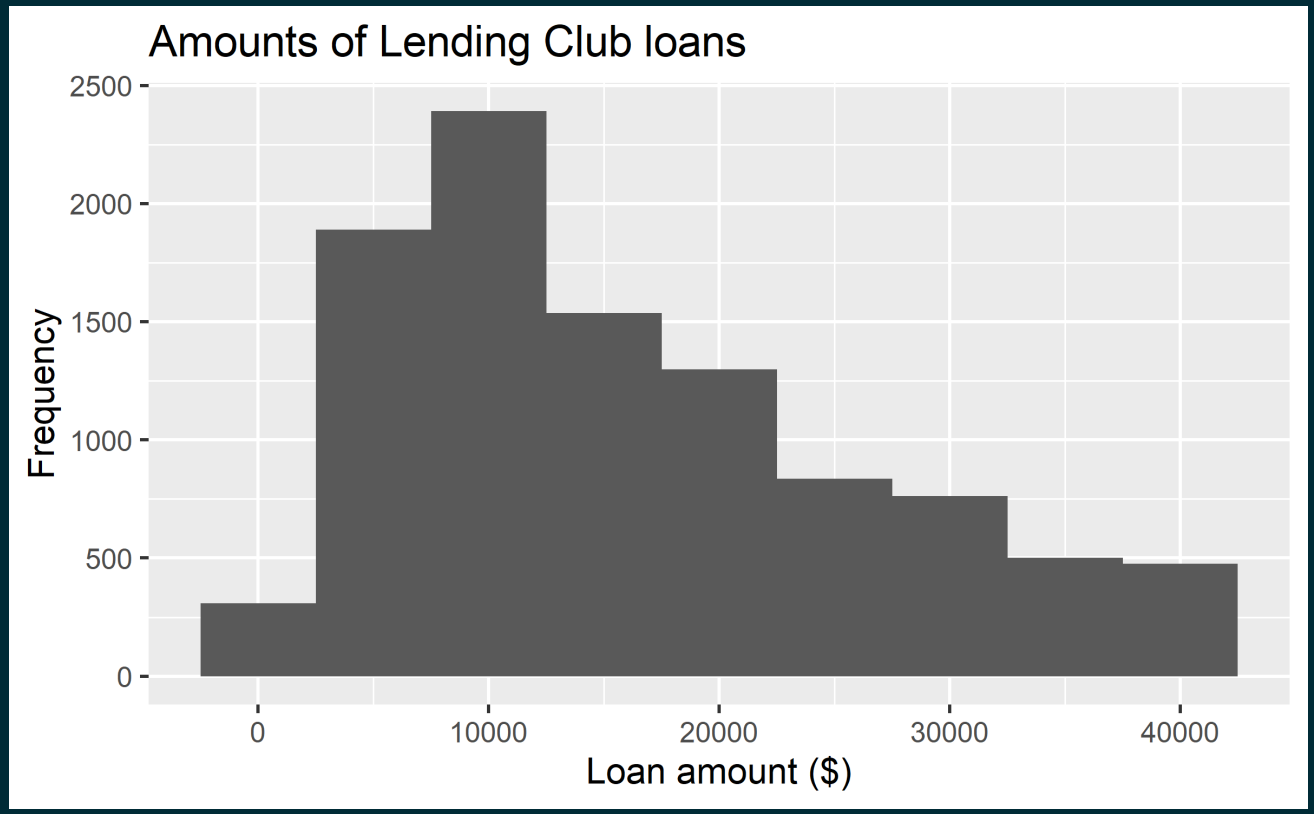
binwidth = 1000 | binwidth = 5000 | binwidth = 20000

```
ggplot(loans, aes(x = loan_amount)) +
  geom_histogram(binwidth = 1000)
```



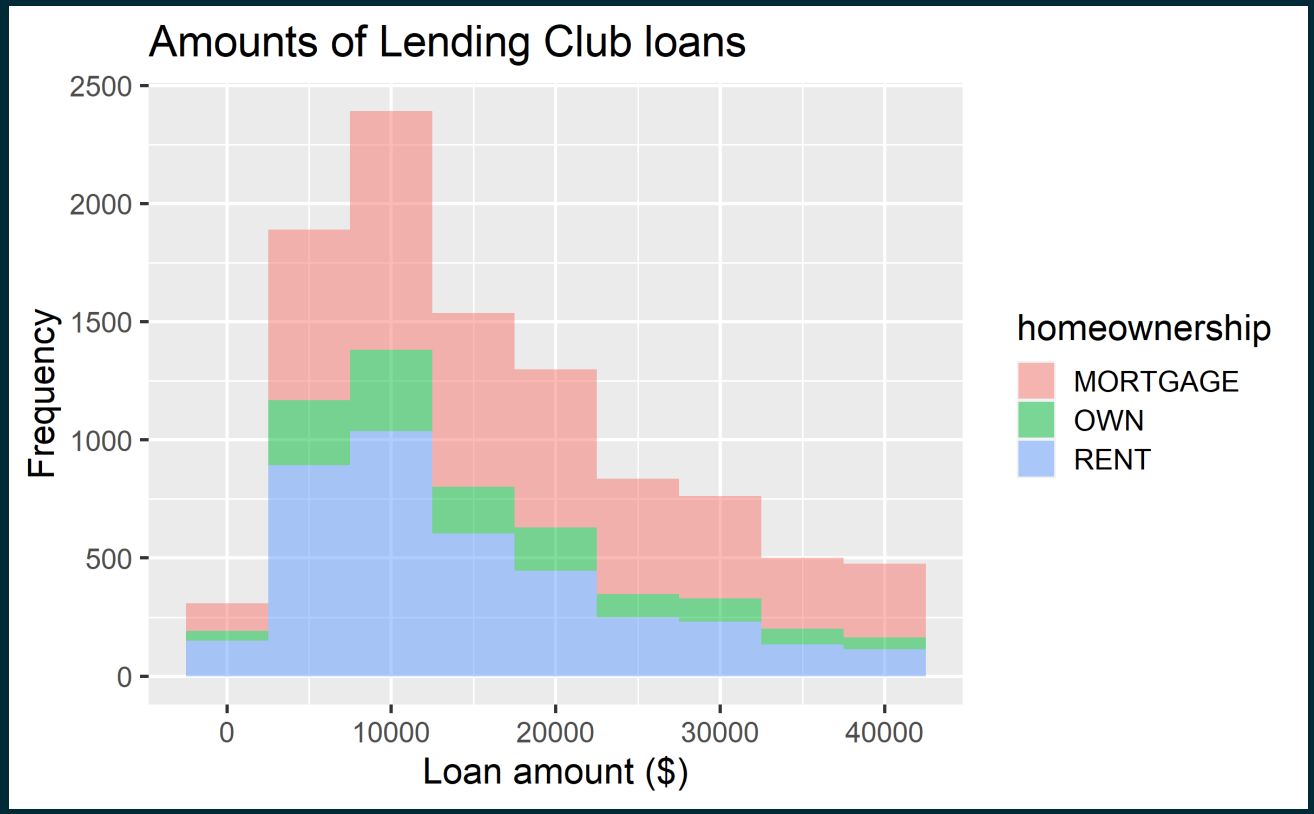
Customizing histograms

Plot Code



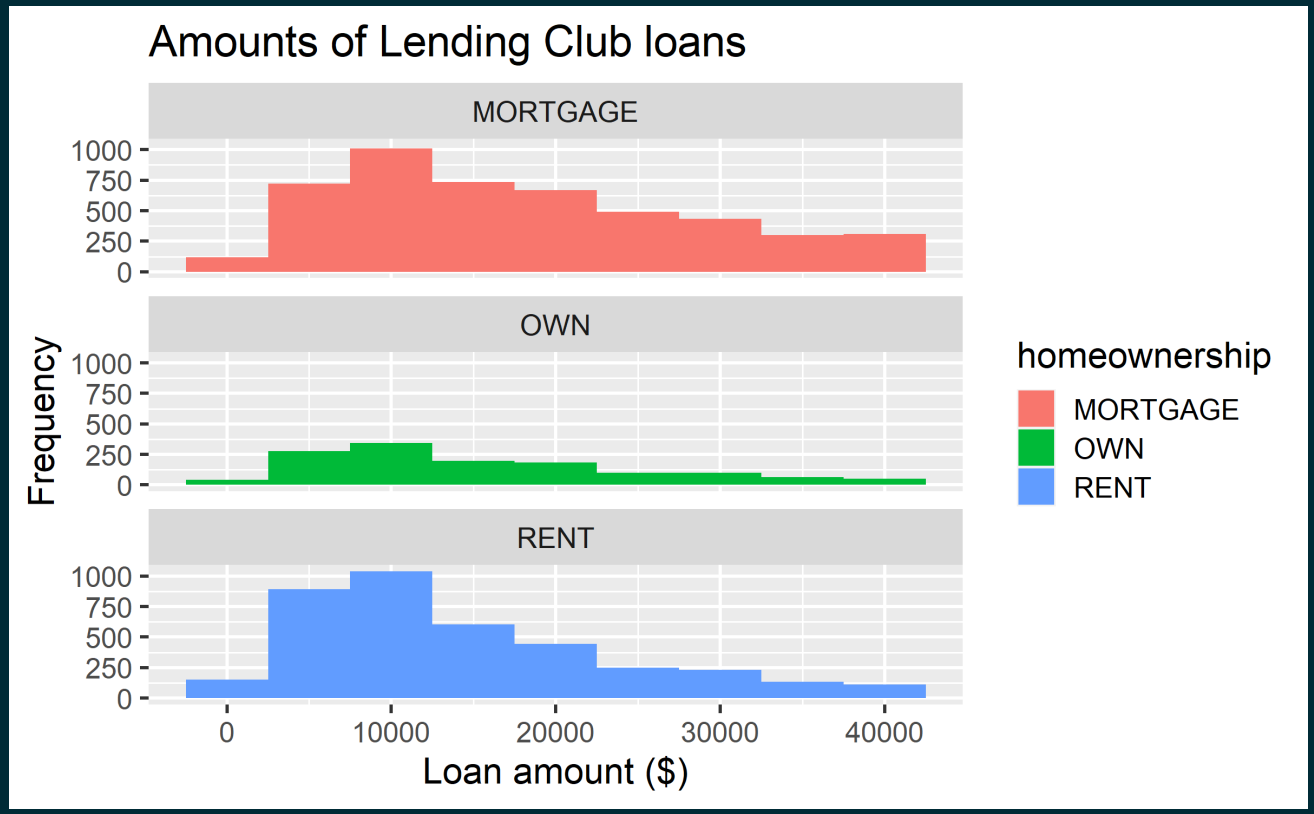
Fill with a categorical variable

Plot Code



Facet with a categorical variable

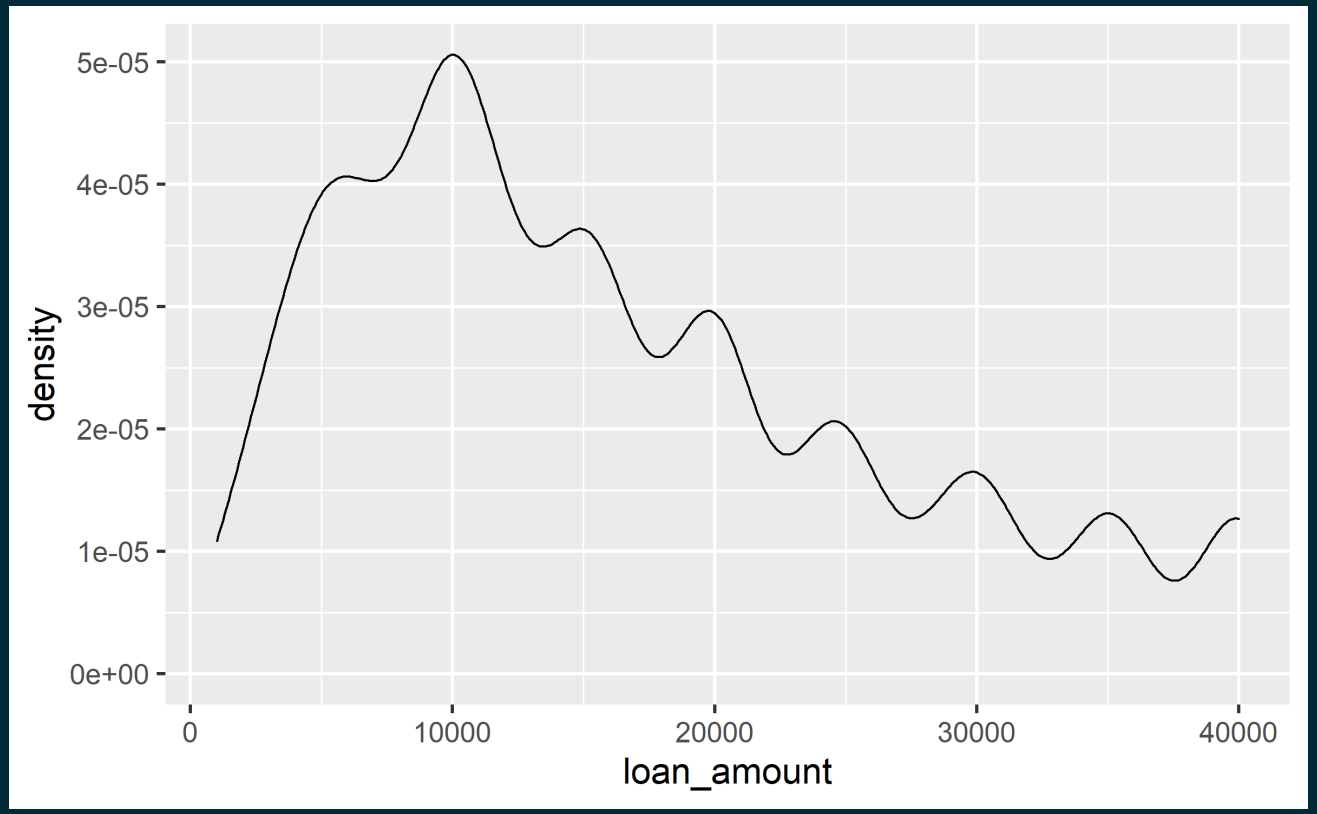
Plot Code



Density plot

Density plot

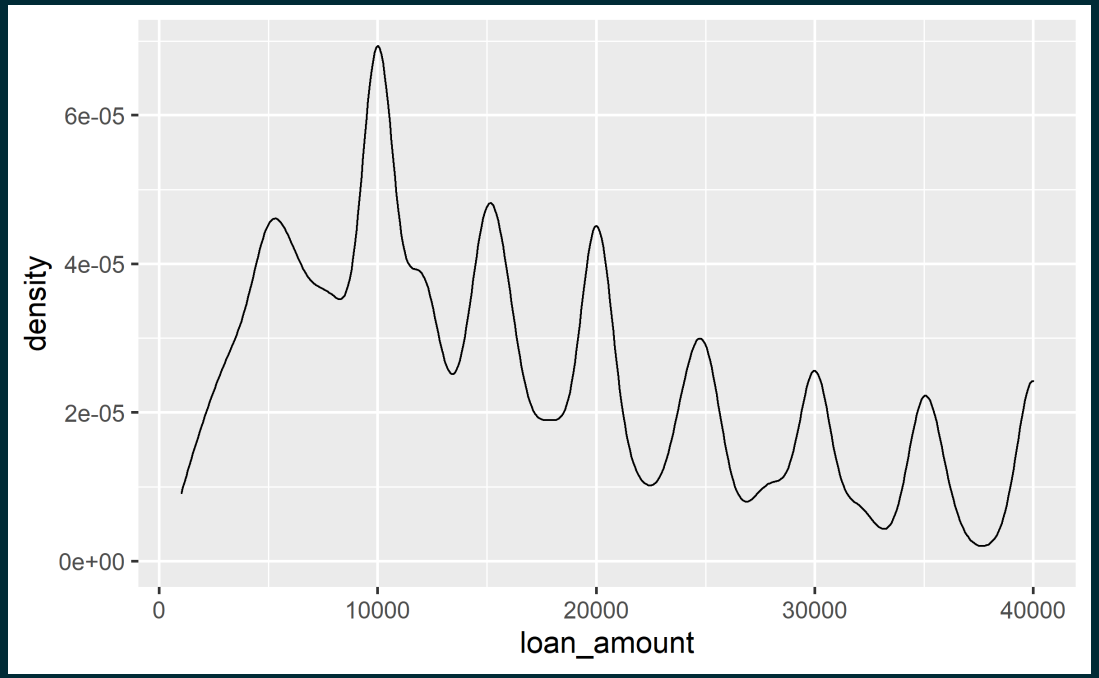
```
ggplot(loans, aes(x = loan_amount)) +
  geom_density()
```



Density plots and adjusting bandwidth

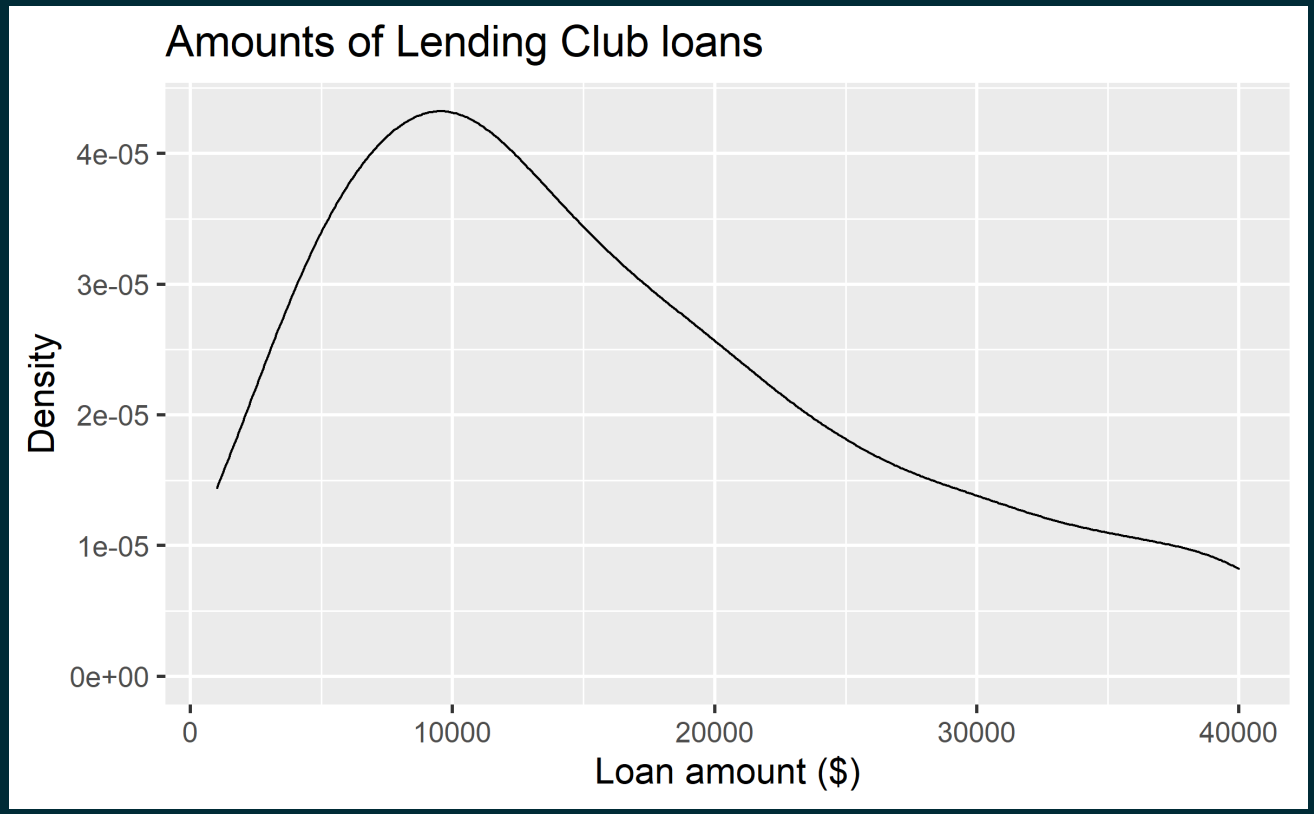
adjust = 0.5 adjust = 1 adjust = 2

```
ggplot(loans, aes(x = loan_amount)) +
  geom_density(adjust = 0.5)
```



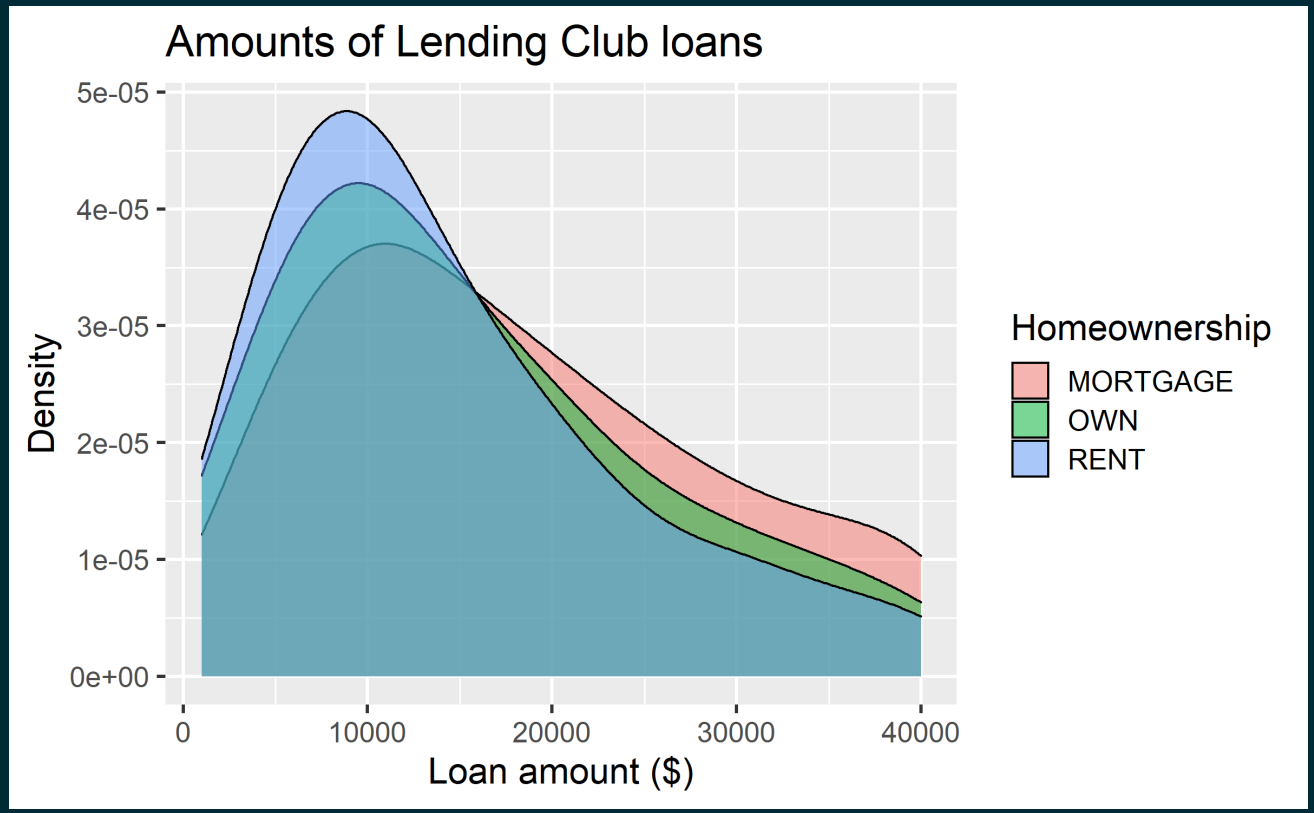
Customizing density plots

Plot Code



Adding a categorical variable

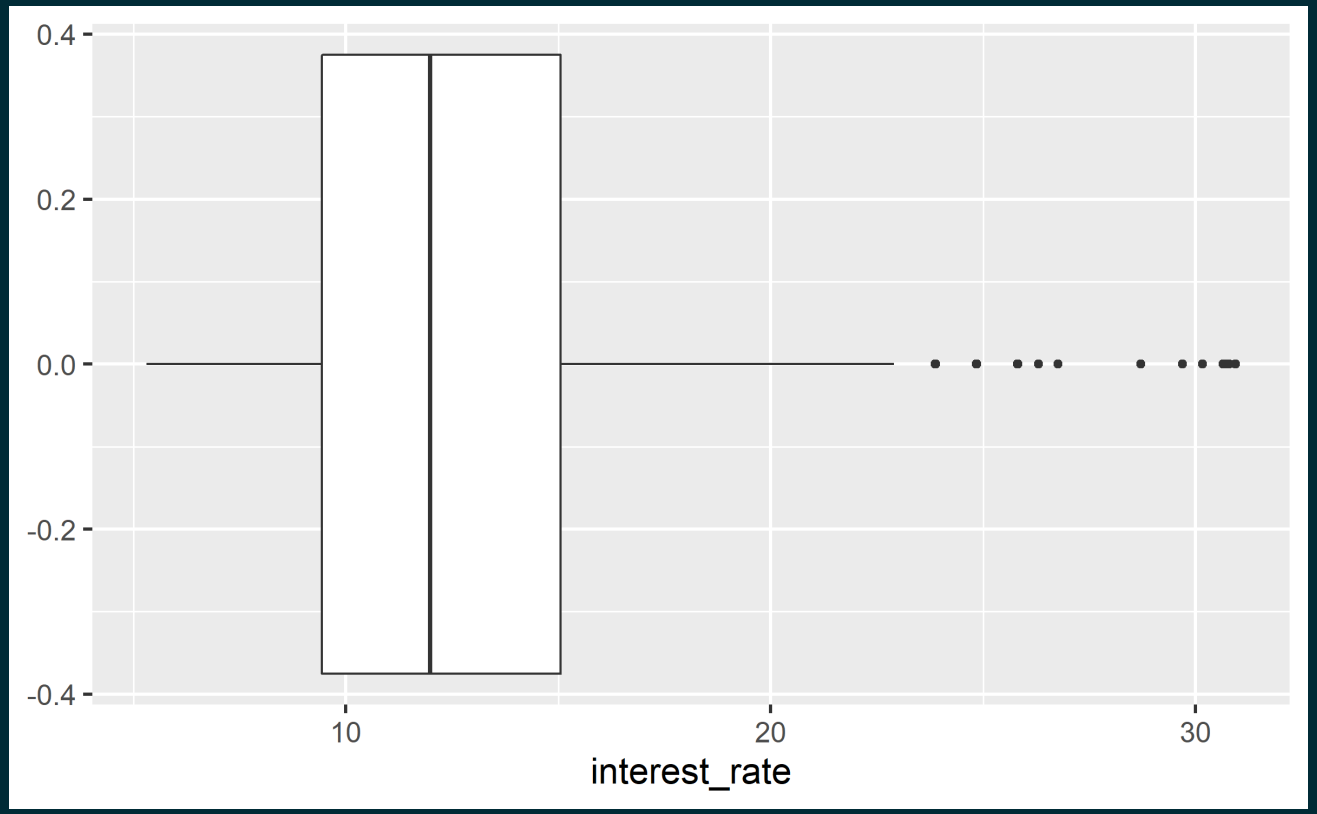
Plot Code



Box plot

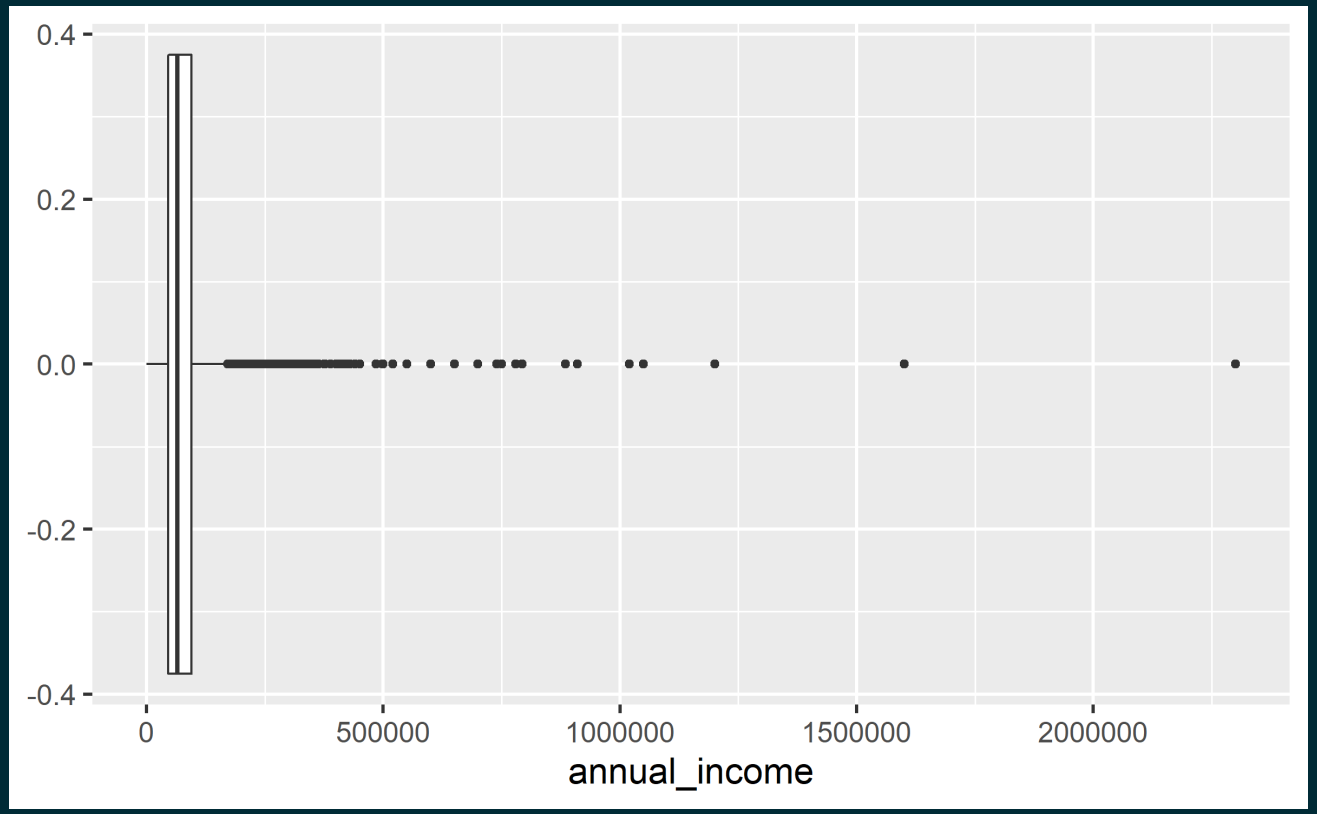
Box plot

```
ggplot(loans, aes(x = interest_rate)) +
  geom_boxplot()
```



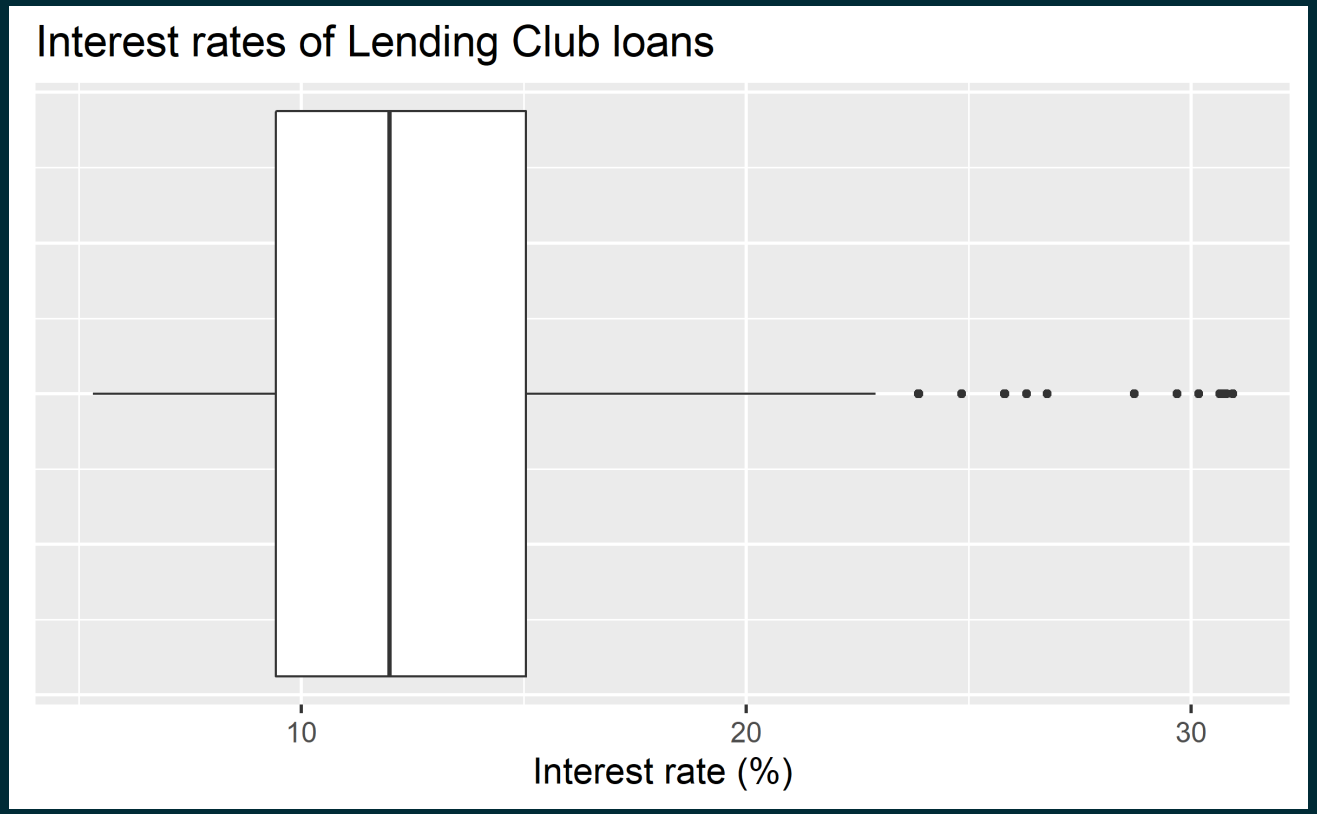
Box plot and outliers

```
ggplot(loans, aes(x = annual_income)) +
  geom_boxplot()
```



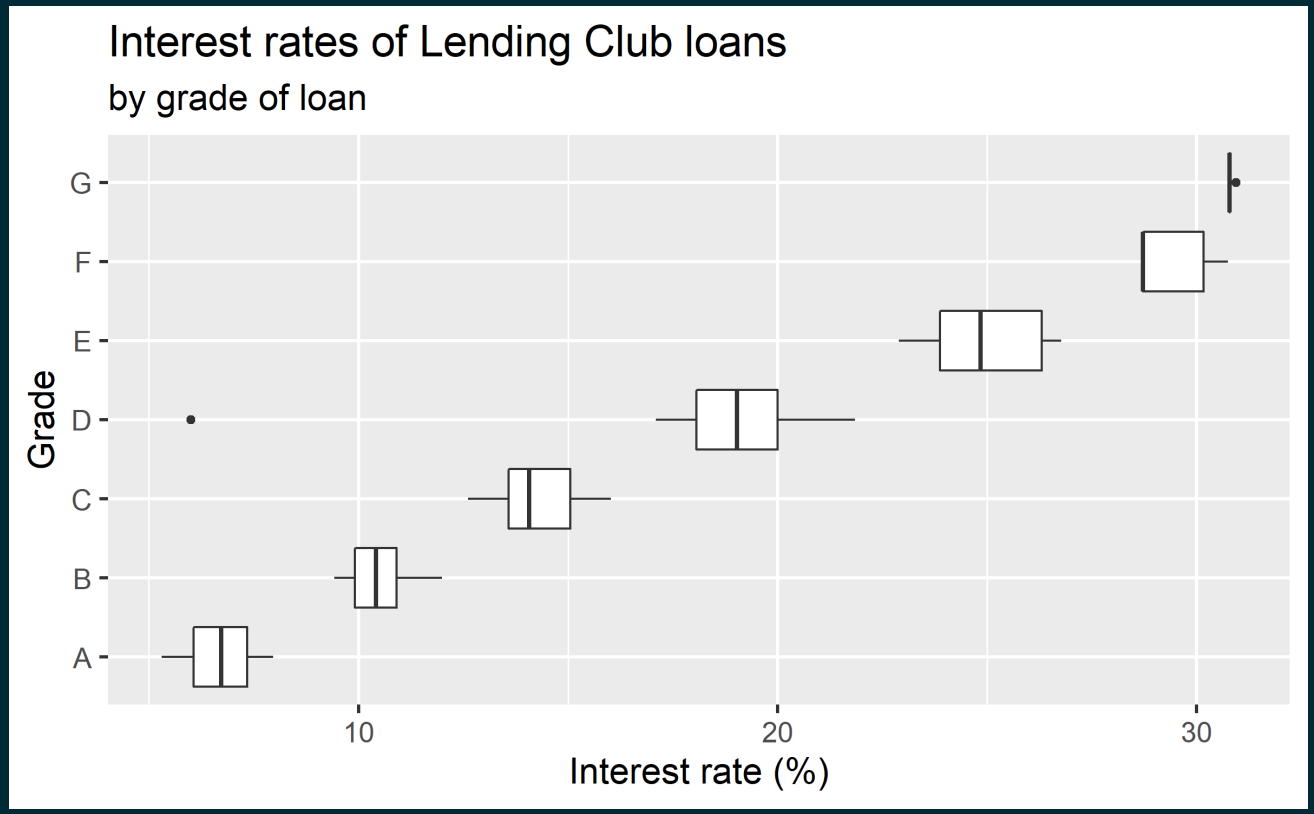
Customizing box plots

Plot Code



Adding a categorical variable

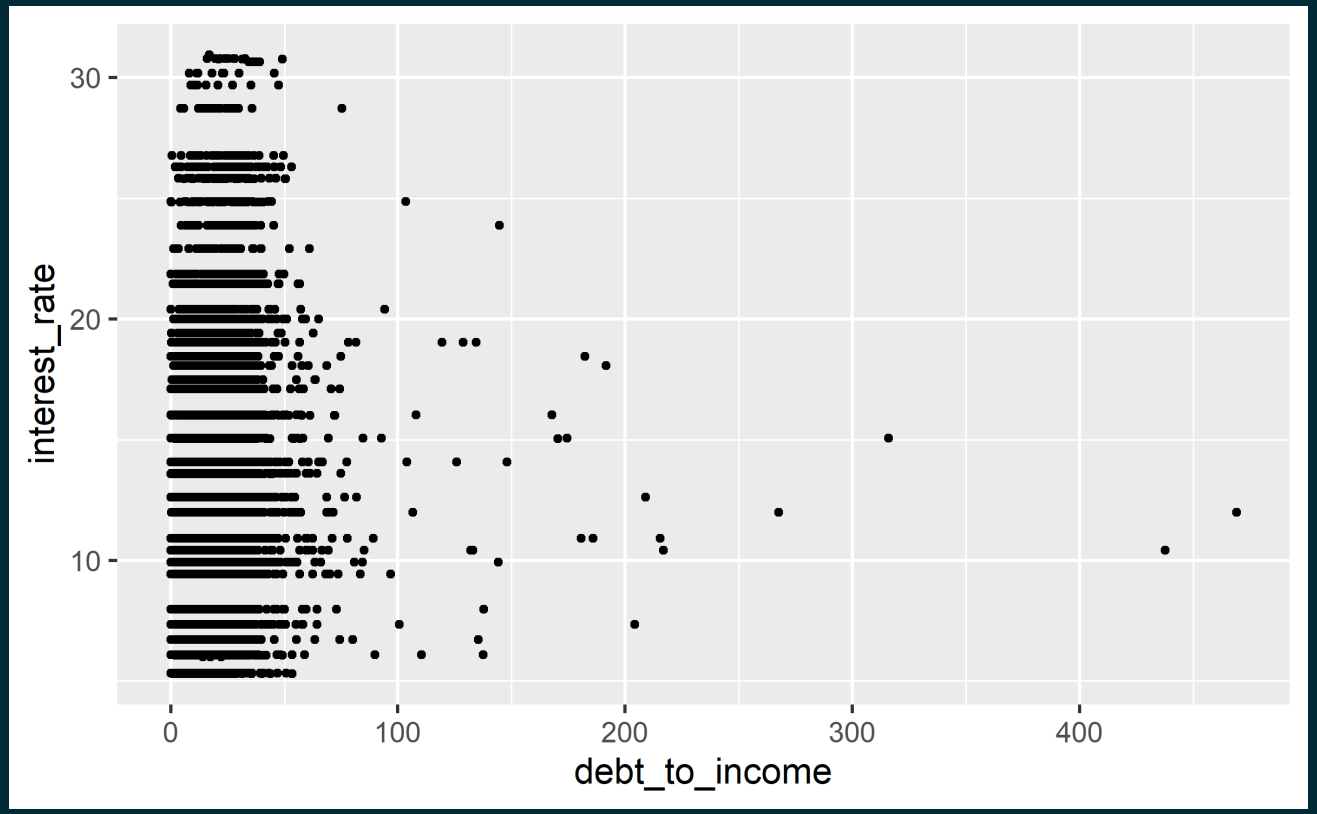
Plot Code



Relationships numerical variables

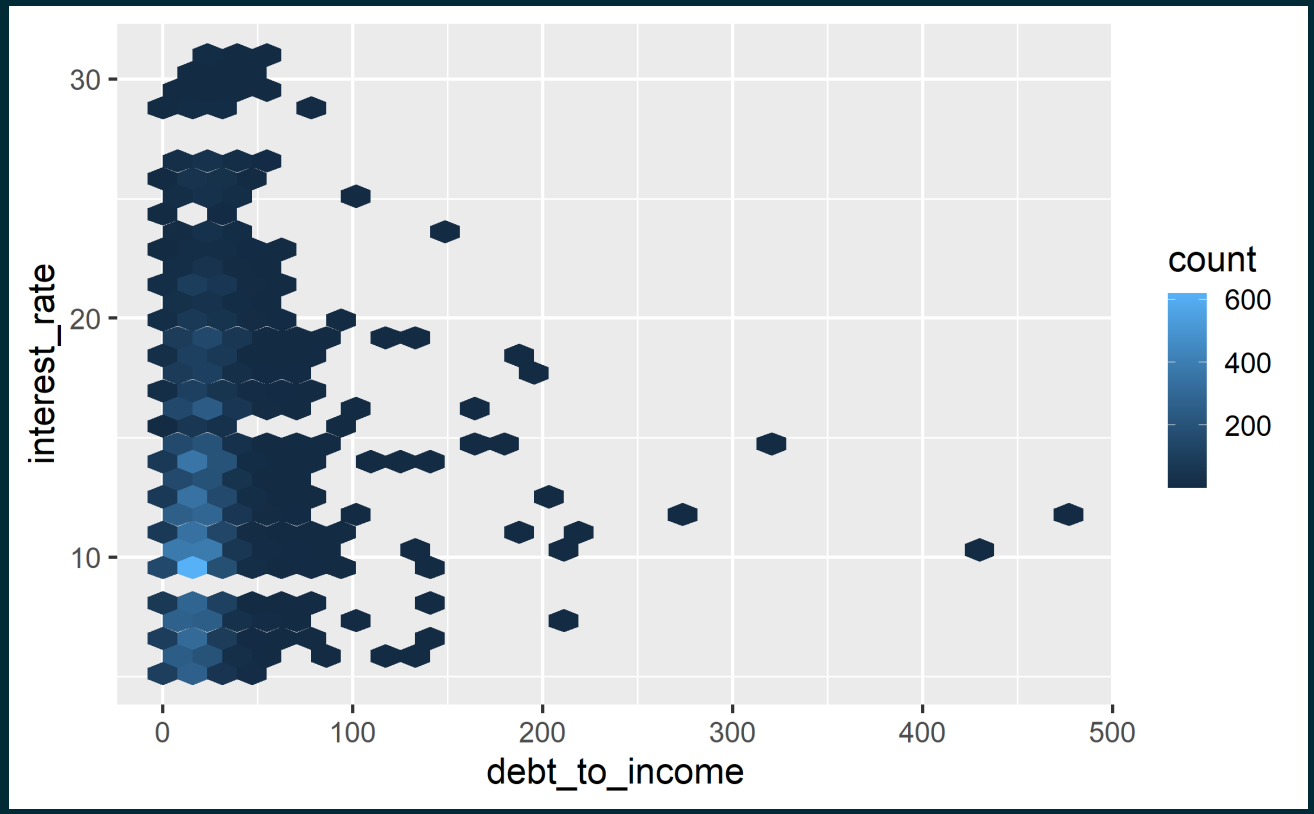
Scatterplot

```
ggplot(loans, aes(x = debt_to_income, y = interest_rate)) +
  geom_point()
```



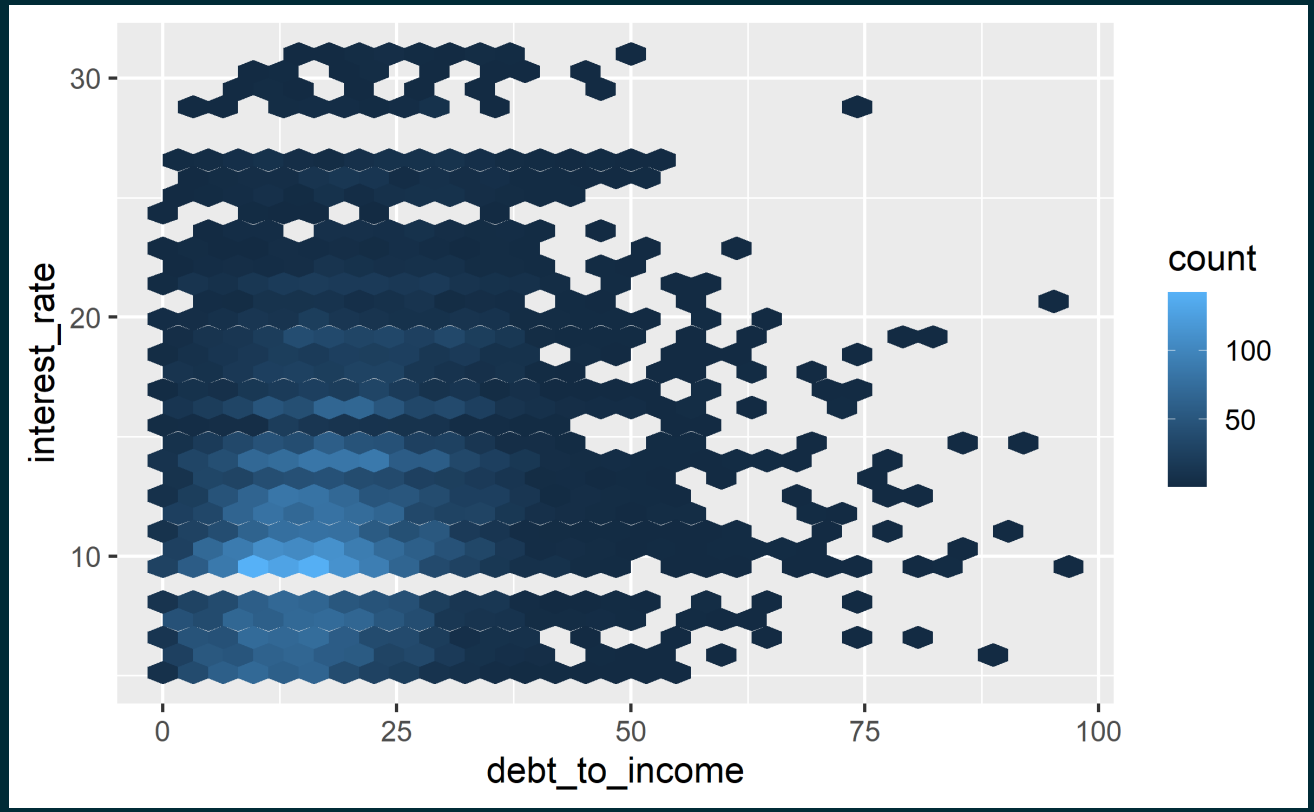
Hex plot

```
ggplot(loans, aes(x = debt_to_income, y = interest_rate)) +
  geom_hex()
```



Hex plot

```
ggplot(loans %>% filter(debt_to_income < 100),
       aes(x = debt_to_income, y = interest_rate)) +
  geom_hex()
```



Variables

- **Numerical** variables can be classified as **continuous** or **discrete** based on whether or not the variable can take on an infinite number of values or only non-negative whole numbers, respectively.
- If the variable is **categorical**, we can determine if it is **ordinal** based on whether or not the levels have a natural ordering.

Data

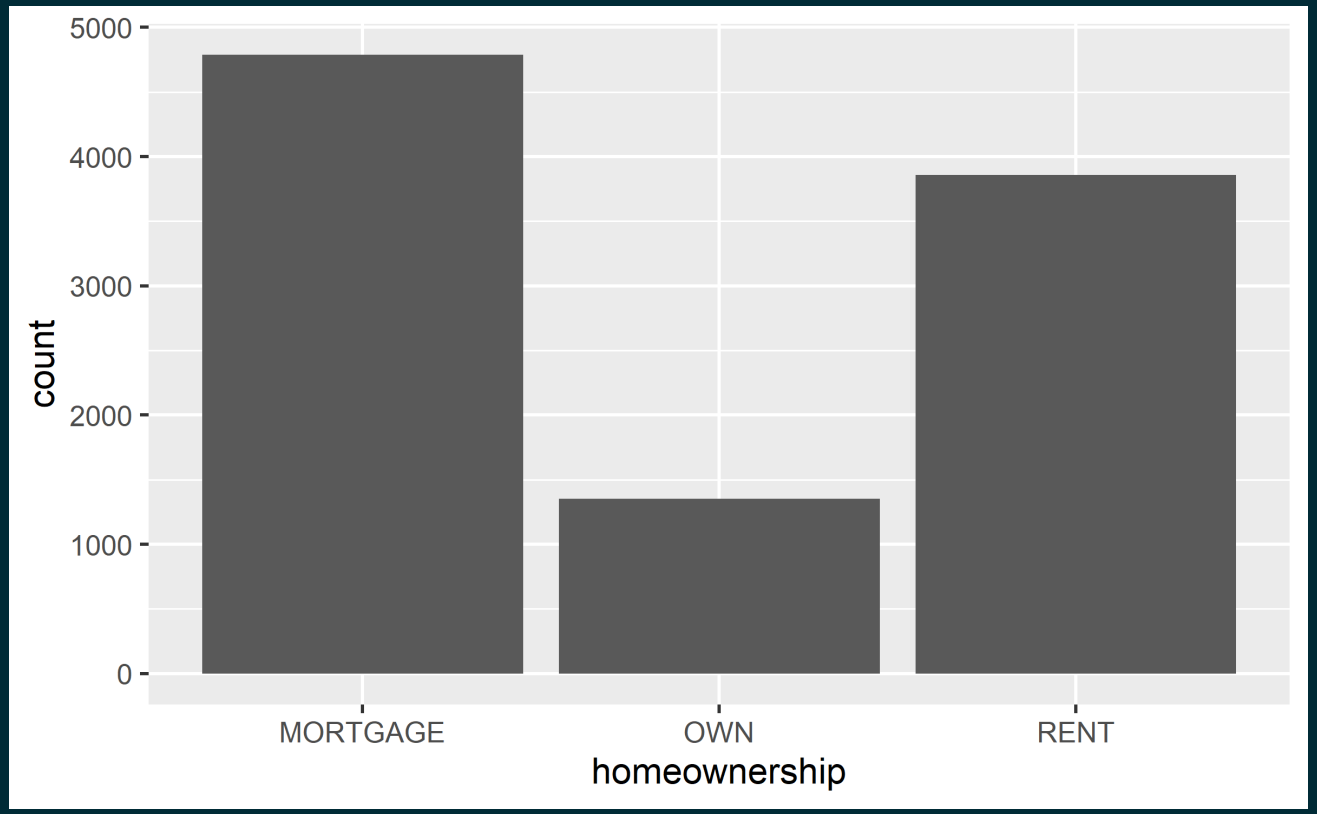
```
library(openintro)
loans <- loans_full_schema %>%
  select(loan_amount, interest_rate, term, grade,
         state, annual_income, homeownership, debt_to_income)
glimpse(loans)
```

```
## Rows: 10,000
## Columns: 8
## $ loan_amount    <int> 28000, 5000, 2000, 21600, 23000, 5000, 2~
## $ interest_rate  <dbl> 14.07, 12.61, 17.09, 6.72, 14.07, 6.72, ~
## $ term           <dbl> 60, 36, 36, 36, 36, 36, 60, 60, 36, 36, ~
## $ grade          <ord> C, C, D, A, C, A, C, B, C, A, C, B, C, B~
## $ state          <fct> NJ, HI, WI, PA, CA, KY, MI, AZ, NV, IL, ~
## $ annual_income  <dbl> 90000, 40000, 40000, 30000, 35000, 34000~
## $ homeownership  <fct> MORTGAGE, RENT, RENT, RENT, RENT, OWN, M~
## $ debt_to_income <dbl> 18.01, 5.04, 21.15, 10.16, 57.96, 6.46, ~
```

Bar plot

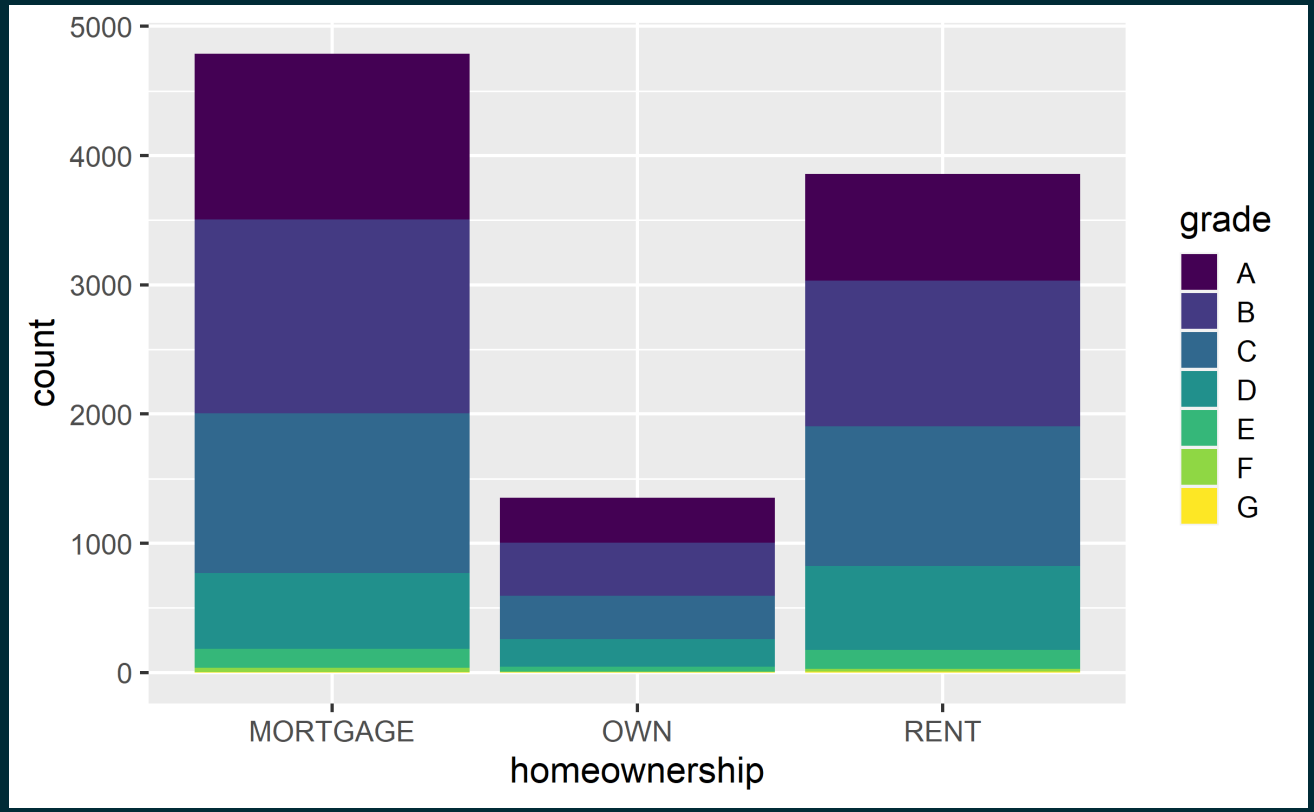
Bar plot

```
ggplot(loans, aes(x = homeownership)) +
  geom_bar()
```



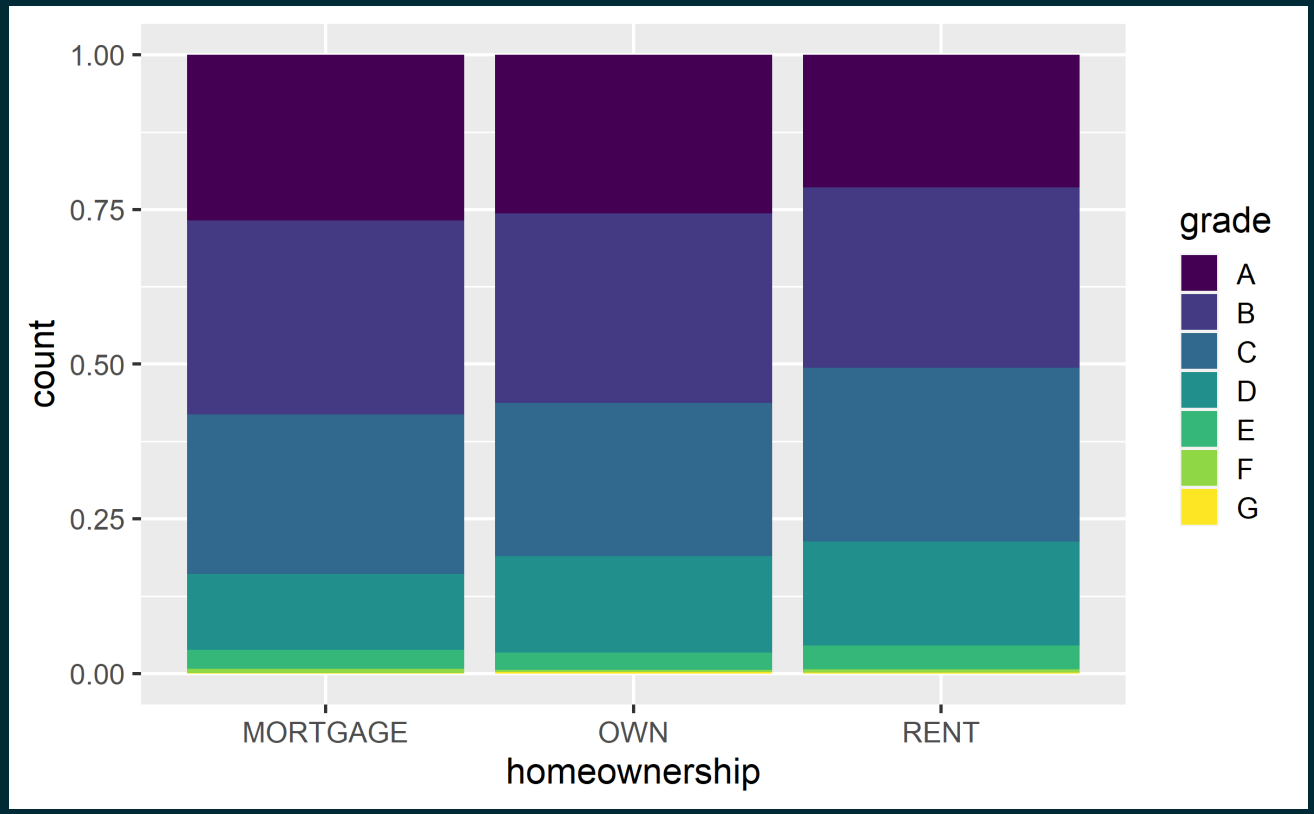
Segmented bar plot

```
ggplot(loans, aes(x = homeownership,
                  fill = grade)) +
  geom_bar()
```

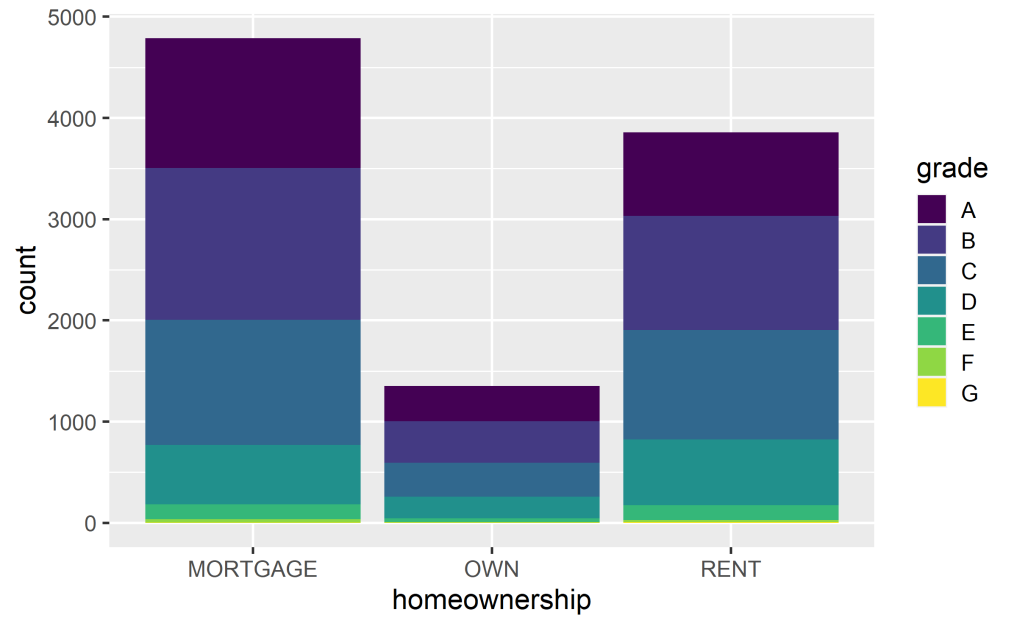


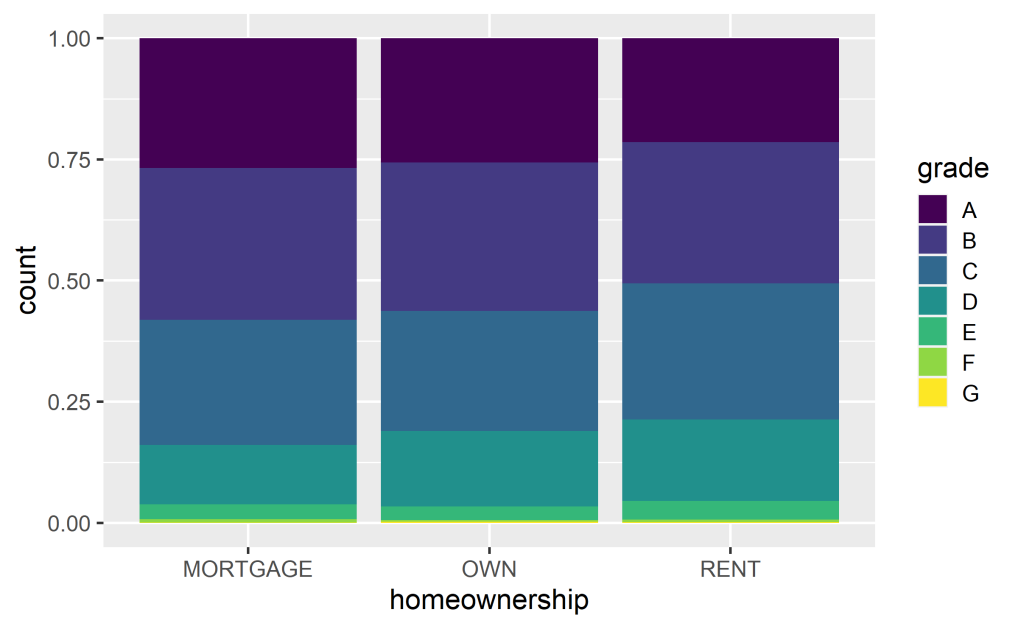
Segmented bar plot

```
ggplot(loans, aes(x = homeownership, fill = grade)) +
  geom_bar(position = "fill")
```



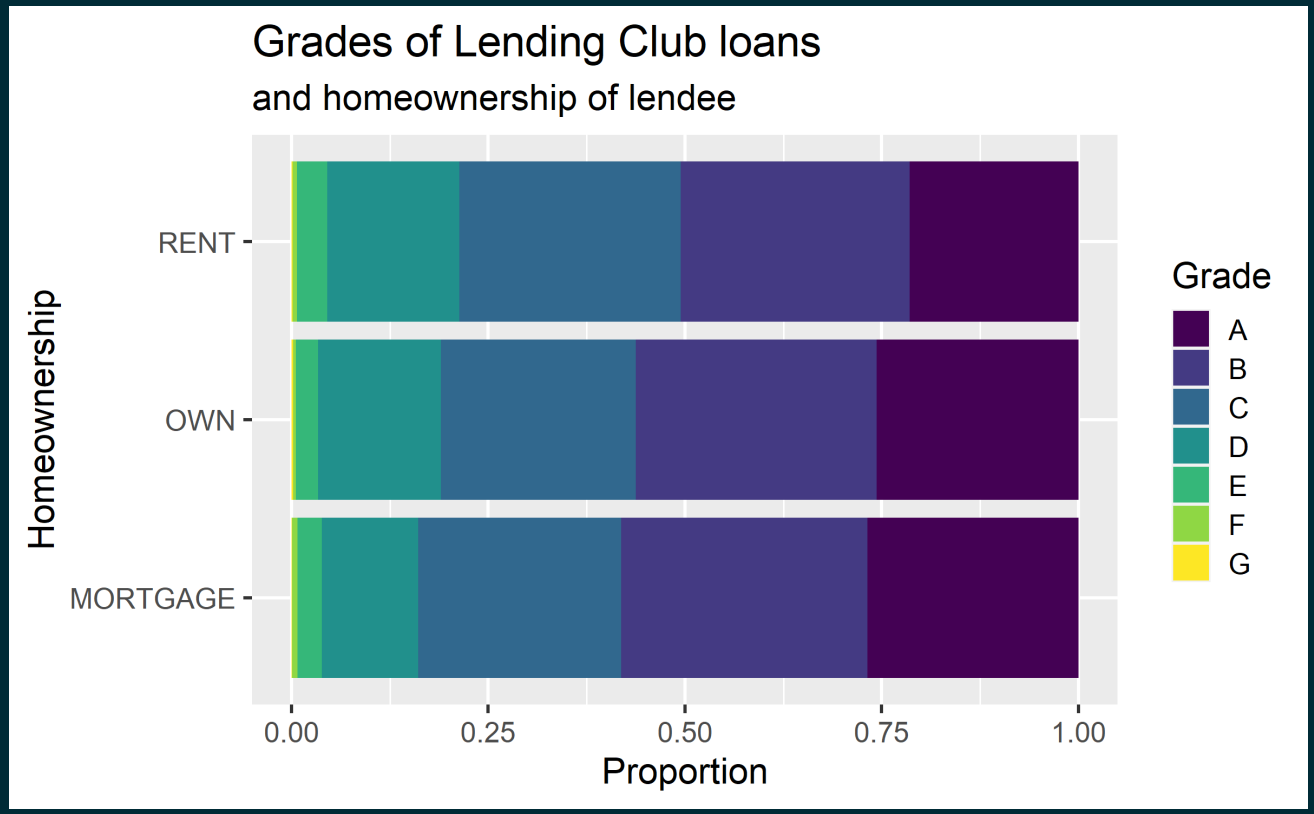
Which bar plot is a more useful representation for visualizing the relationship between homeownership and grade?





Customizing bar plots

Plot Code



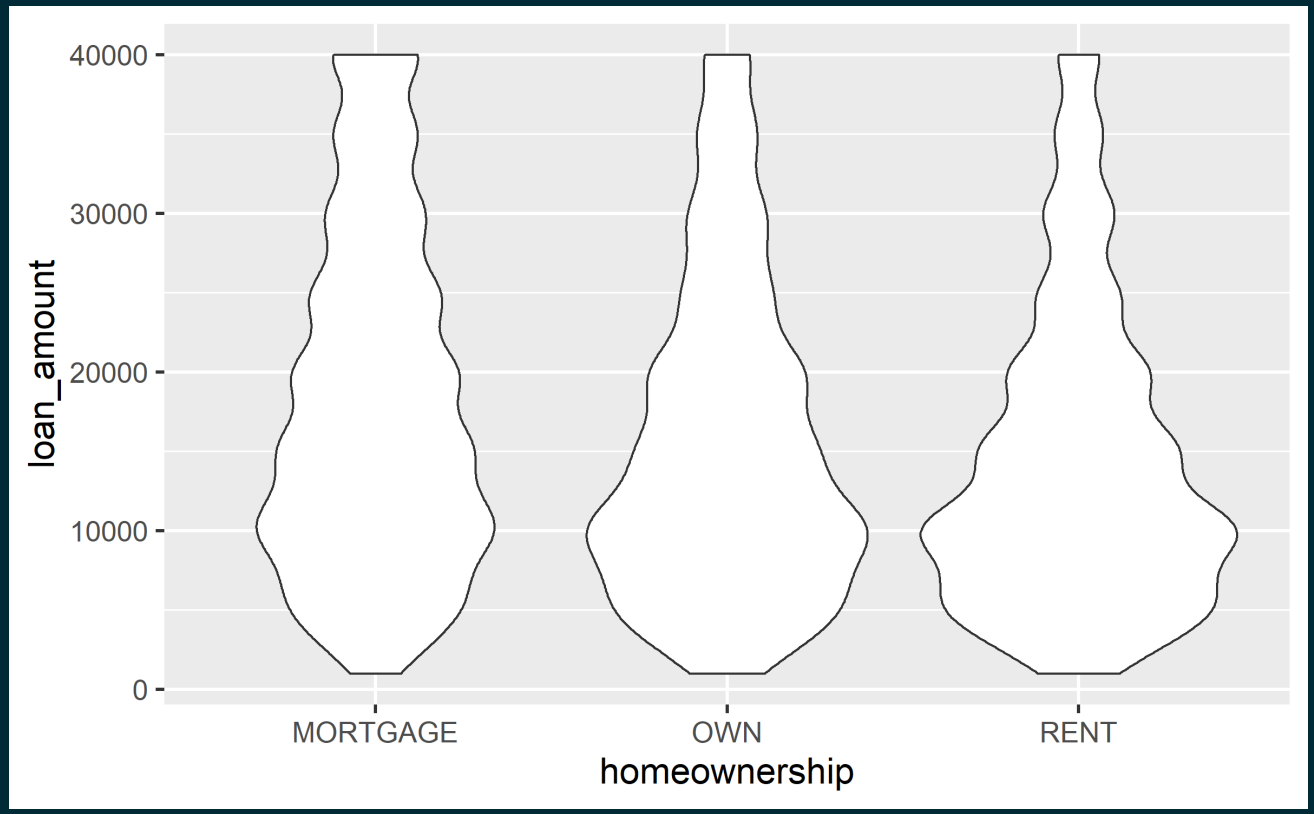
Relationships between numerical and categorical variables

Already talked about...

- Colouring and faceting histograms and density plots
- Side-by-side box plots

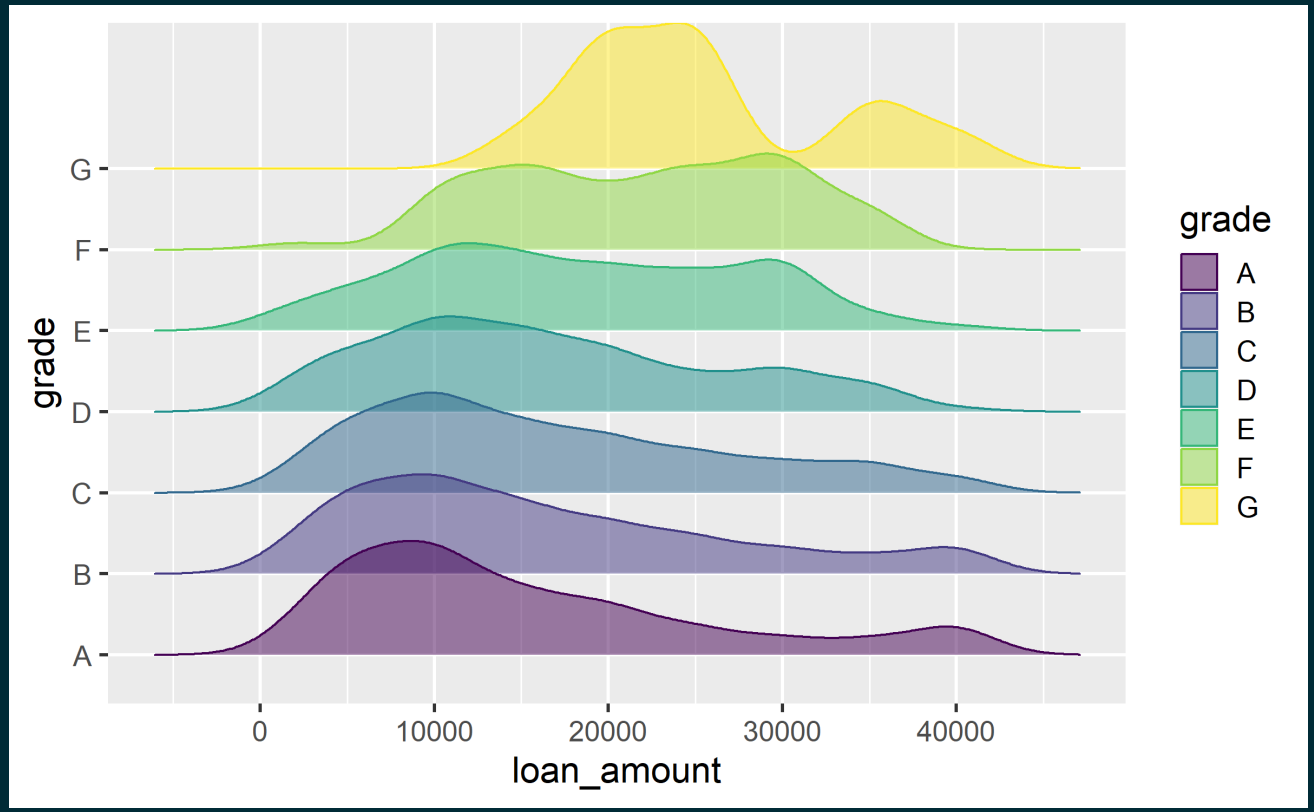
Violin plots

```
ggplot(loans, aes(x = homeownership, y = loan_amount)) +
  geom_violin()
```



Ridge plots

```
library(ggridges)
ggplot(loans, aes(x = loan_amount, y = grade, fill = grade, color = grade)) +
  geom_density_ridges(alpha = 0.5)
```



Your turn: `AE 03 - Starwars - Dataviz`

- Go to our RStudio Server and start `AE 03 - Starwars - Dataviz`. You will need to start an R project from the repository.
- Open and knit the R Markdown document `starwars.Rmd`, review the document, and fill in the blanks.

Followed by: `Lab 02 - Plastic Waste`

- Go to our RStudio Server and start `Lab 02 - Plastic Waste`. You will need to start an R project from the repository.
- Open and knit the R Markdown document `lab-02-plastic-waste.Rmd`.