A Minimal Book Example

John Doe

2023-02-07

Contents

Chapter 1

About

1.1 Setup

```
library(tidyverse)
#> -- Attaching packages ------------------- tidyverse 1.3.2 --
#> v ggplot2 3.4.0      v purrr   1.0.0
#> v tibble  3.1.8      v dplyr   1.0.10
#> v tidyr   1.2.1      v stringr 1.5.0
#> v readr   2.1.3      v forcats 0.5.2
#> -- Conflicts ---------------------- tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

Chapter 2

Intdoduction

Chapter 3

R with R Studio

Course Contents

1. 2022-12-07: Introduction: About the course [lead by TK] - An introduction to open and public data, and data science
2. **2022-12-14: Exploratory Data Analysis (EDA) 1 [lead by hs] - R Basics with RStudio and/or RStudio.cloud; Toy Data**
3. 2022-12-21: Exploratory Data Analysis (EDA) 2 [lead by hs]
   - R Markdown; Introduction to `tidyverse` I; Public Data, WDI
4. 2023-01-11: Exploratory Data Analysis (EDA) 3 [lead by hs]
   - Introduction to `tidyverse`II; WDI, WIR, etc
5. 2023-01-18: Exploratory Data Analysis (EDA) 4 [lead by hs]
   - Introduction to `tidyverse` III; WDI, WIR, etc
6. 2023-01-25: Exploratory Data Analysis (EDA) 5 [lead by hs]
   - Introduction to `tidyverse` III; WDI, WIR, etc
7. 2023-02-01: Introduction to PPDAC (Problem-Plan-Data-Analysis-Conclusion) Cycle: [lead by TK]
8. 2023-02-08: Model building I [lead by TK] -Collecting and visualizing data and Introduction to WDI
   (World Development Indicators by World Bank)
9. 2023-02-15: Model building II [lead by TK] -Analyzing data and communications
10. 2023-02-22: Project Presentation

3.1 Learning Resources

3.1.1 Textbooks and References

- "R for Data Science" by Hadley Wickham and Garrett Grolemund:
  - Free Online Book: https://r4ds.had.co.nz
- Visit `bookdown` site: https://bookdown.org
  - Many more on the archive page.

3.2 Interactive Exercises

- Posit Primers:https://posit.cloud/learn/primers:
  - The Basics, Work with Data, Visualize Data, Tidy Your Data, Report Reproducibly
- {swirl} Learn R, in R: https://swirlstats.com
  - Designed and developed by a team at Johns Hopkins University for `coursera` courses

---

3.3 Posit Primers created by `learnr`

- `learnr` Interactive Tutorials for R

3.3.1 Posit Primers https://posit.cloud/learn/primers

1. The Basics – r4ds: Explore, I

- Visualization Basics
- Programming Basics

2. Work with Data – r4ds: Wrangle, I

- Working with Tibbles
- Isolating Data with dplyr
- Deriving Information with dplyr

3. Visualize Data – r4ds: Explore, II
4. Tidy Your Data – r4ds: Wrangle, II
5. Iterate – r4ds: Program
6. Write Functions – r4ds: Program

3.4 Data Science and EDA

3.4.1 Wikipedia https://en.wikipedia.org/wiki/Data_science

> An inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data.

- Create Insights
- Impact Decision Making
- Maintain & Improve Overtime

3.5 What is R?

3.5.1 R (programming language), Wikipedia

- **R is a programming language** and **free software** environment for **statistical computing and graphics** supported by the R Foundation for Statistical Computing.
- The R language is widely used among statisticians and data miners for developing statistical software and data analysis.
- A **GNU package**, the official R software environment is written primarily in C, Fortran, and R itself (thus, it is partially self-hosting) and is freely available under the GNU General Public License.

3.5.2 History of R and more

"R Programming for Data Science" by Roger Peng

- Chapter 2. History and Overview of R
- Overview and History of R: Youtube video

3.6 Why R? – Responses by Hadley Wickham

3.6.1 r4ds: R is a great place to start your data science journey because

- R is an environment designed from the ground up to support data science.
- R is not just a programming language, but it is also an interactive environment for doing data science.

- To support interaction, R is a much more flexible language than many of its peers.

3.6.2 Why R today?

When you talk about choosing programming languages, I always say you shouldn't pick them based on technical merits, but rather pick them based on the community. And I think the R community is like really, really strong, vibrant, free, welcoming, and embraces a wide range of domains. So, if there are like people like you using R, then your life is going to be much easier. That's the first reason.

Interview: "Advice to Young (and Old) Programmers, H. Wickham"

3.7 What is RStudio? https://posit.com

> RStudio is an integrated development environment, or IDE, for R programming.

3.7.1 R Studio (Wikipedia)

RStudio is an integrated development environment (IDE) for R, a programming language for statistical computing and graphics. It is available in two formats: RStudio Desktop is a regular desktop application while RStudio Server runs on a remote server and allows accessing RStudio using a web browser.

3.8 Installation of R and R Studio

3.8.1 R Installation

To download R, go to CRAN, the comprehensive R archive network. CRAN is composed of a set of mirror servers distributed around the world and is used to distribute R and R packages. Don't try and pick a mirror that's close to you: instead use the cloud mirror, https://cloud.r-project.org, which automatically figures it out for you.

A new major version of R comes out once a year, and there are 2-3 minor releases each year. It's a good idea to update regularly.

3.8.2 R Studio Installation

Download and install it from http://www.rstudio.com/download.

RStudio is updated a couple of times a year. When a new version is available, RStudio will let you know.

3.9 R Studio

3.9.1 The First Step

1. Start R Studio Application
2. Top Menu: File > New Project > New Directory > New Project > *Directory name or Browse the directory and choose the parent directory you want to create the directory*

3.9.2 When You Start the Project

1. Go to the directory you created
2. Double click _'Directory Name'.Rproj

Or,

1. Start R Studio
2. File > Open Project (or choose from Recent Project)

In this way the working directory of the session is set to the project directory and R can search releted files without difficulty (`getwd()`, `setwd()`)

3.10 Posit Cloud

RStudio Cloud is a lightweight, cloud-based solution that allows anyone to do, share, teach and learn data science online.

3.10.1 Cloud Free

- Up to 15 projects total
- 1 shared space (5 members and 10 projects max)
- 15 project hours per month
- Up to 1 GB RAM per project
- Up to 1 CPU per project
- Up to 1 hour background execution time

3.10.2 How to Start Posit Cloud

1. Go to https://posit.cloud/

2. Sign Up: *top right*

- Email address or Google account

3. New Project: *Project Name*
4. R Console

3.11 Let's Get Started

Start RStudio and create a project, or login to Posit Cloud and create a project.

3.11.1 The First Examples

Input the following codes into Console in the left bottom pane.

- The first two:

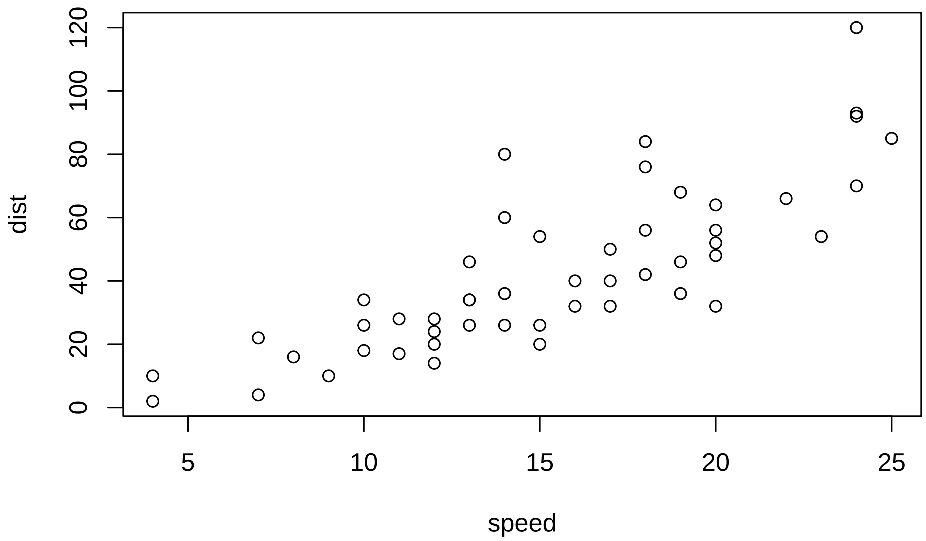
```
head(cars)
#>   speed dist
#> 1     4    2
#> 2     4   10
#> 3     7    4
#> 4     7   22
#> 5     8   16
#> 6     9   10
```

```
str(cars)
#> 'data.frame':	50 obs. of  2 variables:
#>  $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
#>  $ dist : num  2 10 4 22 16 10 18 26 34 17 ...
```

- Two more:

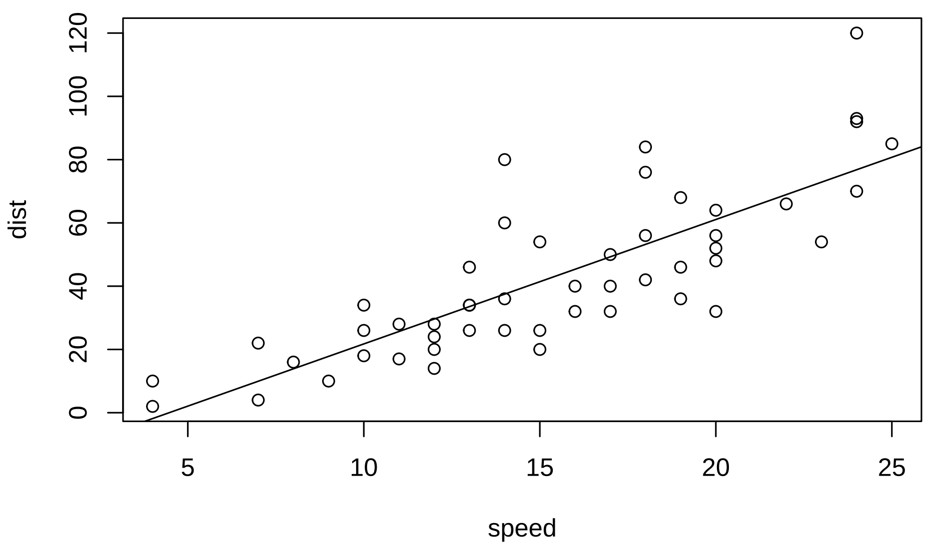
```
summary(cars)
#>      speed           dist       
#>  Min.   : 4.0   Min.   :  2.00  
#>  1st Qu.:12.0   1st Qu.: 26.00  
#>  Median :15.0   Median : 36.00  
#>  Mean   :15.4   Mean   : 42.98  
#>  3rd Qu.:19.0   3rd Qu.: 56.00  
#>  Max.   :25.0   Max.   :120.00
```

```
plot(cars)
```



- And three more:

```
plot(cars) # cars: Speed and Stopping Distances of Cars
abline(lm(cars$dist~cars$speed))
```



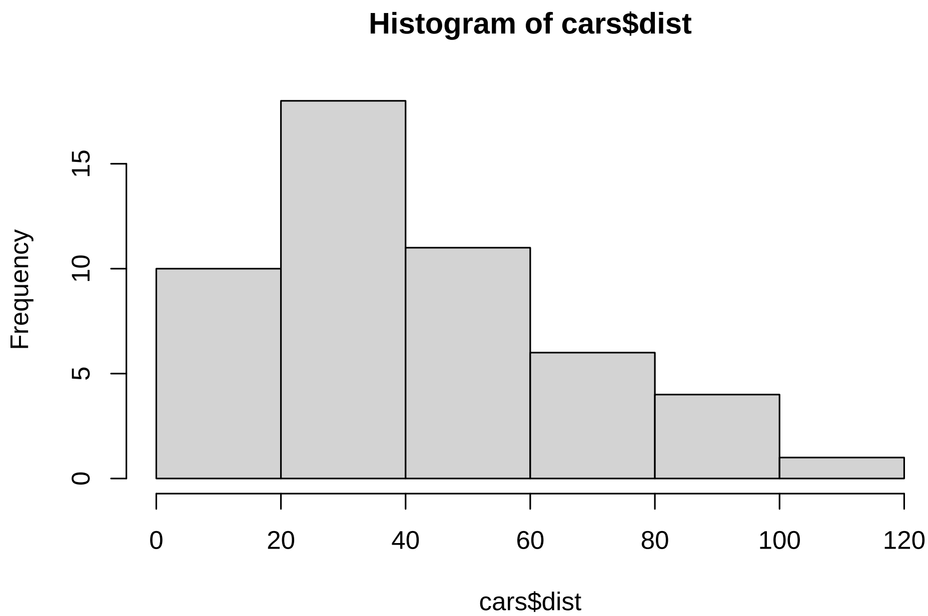
```
lm(cars$dist~cars$speed)
#> 
#> Call:
#> lm(formula = cars$dist ~ cars$speed)
#> 
#> Coefficients:
#> (Intercept)   cars$speed  
#>     -17.579        3.932
```

```
summary(lm(cars$dist~cars$speed))
#> 
#> Call:
#> lm(formula = cars$dist ~ cars$speed)
#> 
#> Residuals:
#>     Min      1Q  Median      3Q     Max 
#> -29.069  -9.525  -2.272   9.215  43.201 
#> 
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)    
#> (Intercept) -17.5791     6.7584  -2.601   0.0123 *  
#> cars$speed    3.9324     0.4155   9.464 1.49e-12 ***
#> ---
#> Signif. codes:  
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Residual standard error: 15.38 on 48 degrees of freedom
#> Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438 
#> F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

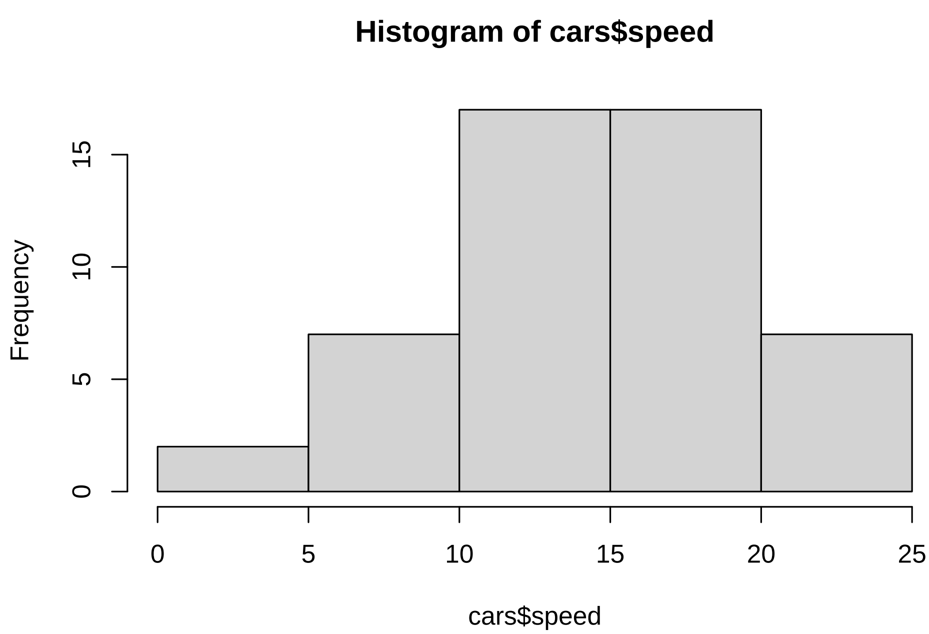
3.11.1.1 Brief Explanation

- `head(cars)`: The first 6 rows of the pre-installed data `cars`.
- `str(cars)`: The data structure of the pre-installed data `cars`.
- `summary(cars)`: The summary of the pre-installed data `cars`.
- `plot(cars)`: A scatter plot of the pre-installed data `cars`.
    - `plot(cars$dist~cars$speed)`
    - `cars$dist`, `cars$[[2]]`, `cars[,2]` are same
- `abline(lm(cars$dist~cars$speed))`: Add a regression line of a linear model
- `lm(cars$dist~cars$speed)`: The equation of the regression line
- `summary(lm(cars$dist~cars$speed)`: The summary of the linear regression model

```
hist(cars$dist)
```




```
hist(cars$speed)
```



3.11.1.2 View and help

- `View(cars)`
- `?cars`: same as `help(cars)`
- `??cars`: same as ‘help.search(“cars”)

3.11.1.3 `datasets`

- `?datasets`
- `library(help = "datasets")`
- `data()` shows all data already attached and available.

3.11.2 Practicum

Pick a data in the datasets package and try

- `head()`
- `str()`
- `summary()`

and some more.

3.11.3 `iris`

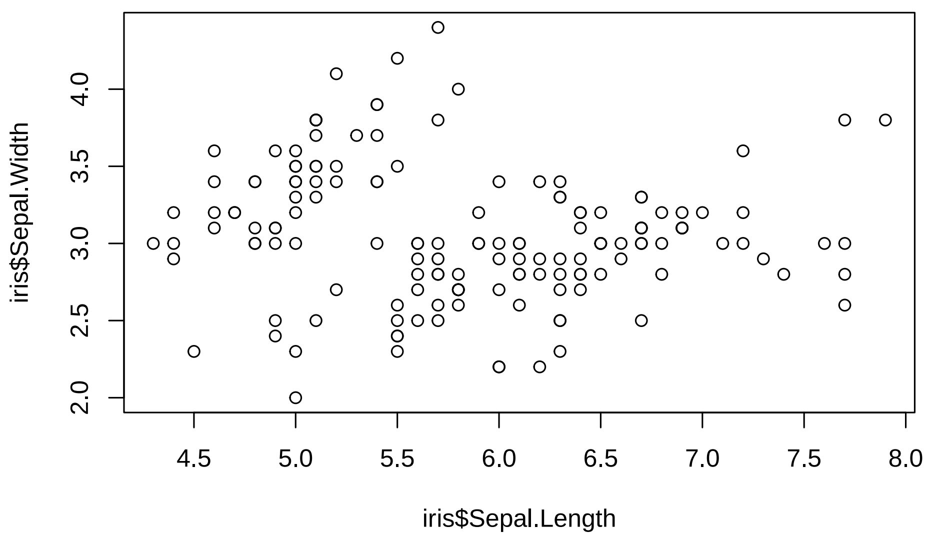
```
head(iris)
#>   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
#> 1          5.1         3.5          1.4         0.2  setosa
#> 2          4.9         3.0          1.4         0.2  setosa
#> 3          4.7         3.2          1.3         0.2  setosa
#> 4          4.6         3.1          1.5         0.2  setosa
#> 5          5.0         3.6          1.4         0.2  setosa
#> 6          5.4         3.9          1.7         0.4  setosa
```

```
str(iris)
#> 'data.frame':    150 obs. of  5 variables:
#>  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
#>  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
#>  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
#>  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
#>  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 .
```

```
summary(iris)
#>   Sepal.Length    Sepal.Width     Petal.Length
#>  Min.   :4.300   Min.   :2.000   Min.   :1.000
#>  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600
#>  Median :5.800   Median :3.000   Median :4.350
#>  Mean   :5.843   Mean   :3.057   Mean   :3.758
#>  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100
#>  Max.   :7.900   Max.   :4.400   Max.   :6.900
#>   Petal.Width          Species
#>  Min.   :0.100   setosa    :50
#>  1st Qu.:0.300   versicolor:50
#>  Median :1.300   virginica :50
#>  Mean   :1.199
#>  3rd Qu.:1.800
#>  Max.   :2.500
```

Can you plot?

```
plot(iris$Sepal.Length, iris$Sepal.Width)
```



Chapter 4

`tidyverse` Packages

4.0.1 Brief Introduction to R on RStudio

4.0.1.1 Four Panes and Tabs

1. Top Left: Source Editor
2. Top Right: Environment, History, etc.
3. Bottom Left: Console, Terminal, Render, Background Jobs
4. Bottom Right: Files, Plots, Packages, Help, Viewer, Presentation

---

4.0.1.2 Set up

- Highly recommend to set the language to be "English".
- Create "data" directory.

```
Sys.setenv(LANG = "en")
dir.create("data")
```

4.0.1.3 Three Ways to Run Codes

1. Console - Bottom Left Pane
2. R Script - pull down menu under File
3. R Notebook, R Markdown - pull down menu under File

4.0.2 Second Way: R Script

4.0.2.1 Examples: R Scripts in Moodle

- `basics.R`
- `coronavirus.R`

1. Copy a script in Moodle: *{file name}.R*
2. In RStudio (create Project in RStudio) choose File > New File > R Script and paste it.
3. Choose File > Save with a name; e.g. *{file names}* (.R will be added automatically)

To run a code: at the cursor press *Ctrl+Shift+Enter* (Win) or *Cmd+Shift+Enter* (Mac).

4.0.3 Packages

R packages are extensions to the R statistical programming language. R packages contain code, data, and documentation in a standardised collection format that can be installed by users of R, typically via a centralised software repository such as CRAN (the Comprehensive R Archive Network).

4.0.3.1 Installation and attachement

You can install packages by "Install Packages…" under "Tool" in the top menu.

- `install.packages("tidyverse")`
- `install.packages("rmarkdown")`

4.0.4 Third Way: R Notebook

Choose R Notebook from the pull down File menu in the top bar.

4.0.5 Edit YAML

Default* is as follows

```
---
title: "R Notebook"
output: html_notebook
---
```

Template

```
---
title: "Title of R Notebook"
author: "ID and Your Name"
date: "2023-02-07"
output:
  html_notebook:
#    number_sections: yes
#    toc: true
#    toc_float: true
---
```

- Don't change the format. Indention matters.
- The statement after # is ignored.
- Date is automatically inserted when you compile the file.
- You can replace "2023-02-07" by "2022-12-14" or in any date format surrounded by double quotation marks.
- Section numbers: - default is `number_sections: no`.
- Table of contents, `toc: true` - default is `toc: false`.
- Floating table of contents in HTML output, `toc_float: true` - default is `toc_float: false`

4.0.6 Create a Code Chunk to Attach Packages

Insert Chunk in Code pull down menu in the top bar, or use the C button on top. You can use shortcut keys listed under Tools in the top bar.

```
library(tidyverse)
#> -- Attaching packages ------------------- tidyverse 1.3.2 --
#> v ggplot2 3.4.0      v purrr   1.0.0
#> v tibble  3.1.8      v dplyr   1.0.10
#> v tidyr   1.2.1      v stringr 1.5.0
#> v readr   2.1.3      v forcats 0.5.2
#> -- Conflicts ---------------------- tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

4.1 First Example

4.1.1 Importing data

Let us assign the `iris` data in the pre-installed package `datasets` to `df_iris`. You can give any name starting from an alphabet, though there are some rules.

```
df_iris <- datasets::iris
class(df_iris)
```

```
#> [1] "data.frame"
```

The class of data `iris` is `data.frame`, the basic data class of R. You can assign the same data as a `tibble`, the data class of `tidyverse` as follows.

```
tbl_iris <- as_tibble(datasets::iris)
class(tbl_iris)
#> [1] "tbl_df"     "tbl"        "data.frame"
```

- `df_iris <- iris` can replace `df_iris <- datasets::iris` because the package `datasets` is installed and attached as default. Since you may have other data called `iris` included in a different package or you may have changed `iris` before, it is safer to specify the name of the package with the name of the data.
- Within R Notebook or in Console, you may get different output, and `tf_iris` and `tbl_iris` behave differently. It is because of the default settings of R Markdown.

4.1.2 Look at the data

4.1.2.1 Several ways to view the data.

The `View` command open up a window to show the contents of the data and you can use the filter as well.

```
View(df_iris)
```

The following simple command also shows the data.

```
df_iris
#>     Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1            5.1         3.5          1.4         0.2
#> 2            4.9         3.0          1.4         0.2
#> 3            4.7         3.2          1.3         0.2
#> 4            4.6         3.1          1.5         0.2
#> 5            5.0         3.6          1.4         0.2
#> 6            5.4         3.9          1.7         0.4
#> 7            4.6         3.4          1.4         0.3
#> 8            5.0         3.4          1.5         0.2
#> 9            4.4         2.9          1.4         0.2
#> 10           4.9         3.1          1.5         0.1
#> 11           5.4         3.7          1.5         0.2
#> 12           4.8         3.4          1.6         0.2
#> 13           4.8         3.0          1.4         0.1
#> 14           4.3         3.0          1.1         0.1
```

```
#> 15          5.8         4.0          1.2         0.2
#> 16          5.7         4.4          1.5         0.4
#> 17          5.4         3.9          1.3         0.4
#> 18          5.1         3.5          1.4         0.3
#> 19          5.7         3.8          1.7         0.3
#> 20          5.1         3.8          1.5         0.3
#> 21          5.4         3.4          1.7         0.2
#> 22          5.1         3.7          1.5         0.4
#> 23          4.6         3.6          1.0         0.2
#> 24          5.1         3.3          1.7         0.5
#> 25          4.8         3.4          1.9         0.2
#> 26          5.0         3.0          1.6         0.2
#> 27          5.0         3.4          1.6         0.4
#> 28          5.2         3.5          1.5         0.2
#> 29          5.2         3.4          1.4         0.2
#> 30          4.7         3.2          1.6         0.2
#> 31          4.8         3.1          1.6         0.2
#> 32          5.4         3.4          1.5         0.4
#> 33          5.2         4.1          1.5         0.1
#> 34          5.5         4.2          1.4         0.2
#> 35          4.9         3.1          1.5         0.2
#> 36          5.0         3.2          1.2         0.2
#> 37          5.5         3.5          1.3         0.2
#> 38          4.9         3.6          1.4         0.1
#> 39          4.4         3.0          1.3         0.2
#> 40          5.1         3.4          1.5         0.2
#> 41          5.0         3.5          1.3         0.3
#> 42          4.5         2.3          1.3         0.3
#> 43          4.4         3.2          1.3         0.2
#> 44          5.0         3.5          1.6         0.6
#> 45          5.1         3.8          1.9         0.4
#> 46          4.8         3.0          1.4         0.3
#> 47          5.1         3.8          1.6         0.2
#> 48          4.6         3.2          1.4         0.2
#> 49          5.3         3.7          1.5         0.2
#> 50          5.0         3.3          1.4         0.2
#> 51          7.0         3.2          4.7         1.4
#> 52          6.4         3.2          4.5         1.5
#> 53          6.9         3.1          4.9         1.5
#> 54          5.5         2.3          4.0         1.3
#> 55          6.5         2.8          4.6         1.5
#> 56          5.7         2.8          4.5         1.3
#> 57          6.3         3.3          4.7         1.6
#> 58          4.9         2.4          3.3         1.0
#> 59          6.6         2.9          4.6         1.3
```

```
#> 60          5.2         2.7         3.9         1.4
#> 61          5.0         2.0         3.5         1.0
#> 62          5.9         3.0         4.2         1.5
#> 63          6.0         2.2         4.0         1.0
#> 64          6.1         2.9         4.7         1.4
#> 65          5.6         2.9         3.6         1.3
#> 66          6.7         3.1         4.4         1.4
#> 67          5.6         3.0         4.5         1.5
#> 68          5.8         2.7         4.1         1.0
#> 69          6.2         2.2         4.5         1.5
#> 70          5.6         2.5         3.9         1.1
#> 71          5.9         3.2         4.8         1.8
#> 72          6.1         2.8         4.0         1.3
#> 73          6.3         2.5         4.9         1.5
#> 74          6.1         2.8         4.7         1.2
#> 75          6.4         2.9         4.3         1.3
#> 76          6.6         3.0         4.4         1.4
#> 77          6.8         2.8         4.8         1.4
#> 78          6.7         3.0         5.0         1.7
#> 79          6.0         2.9         4.5         1.5
#> 80          5.7         2.6         3.5         1.0
#> 81          5.5         2.4         3.8         1.1
#> 82          5.5         2.4         3.7         1.0
#> 83          5.8         2.7         3.9         1.2
#> 84          6.0         2.7         5.1         1.6
#> 85          5.4         3.0         4.5         1.5
#> 86          6.0         3.4         4.5         1.6
#> 87          6.7         3.1         4.7         1.5
#> 88          6.3         2.3         4.4         1.3
#> 89          5.6         3.0         4.1         1.3
#> 90          5.5         2.5         4.0         1.3
#> 91          5.5         2.6         4.4         1.2
#> 92          6.1         3.0         4.6         1.4
#> 93          5.8         2.6         4.0         1.2
#> 94          5.0         2.3         3.3         1.0
#> 95          5.6         2.7         4.2         1.3
#> 96          5.7         3.0         4.2         1.2
#> 97          5.7         2.9         4.2         1.3
#> 98          6.2         2.9         4.3         1.3
#> 99          5.1         2.5         3.0         1.1
#> 100         5.7         2.8         4.1         1.3
#> 101         6.3         3.3         6.0         2.5
#> 102         5.8         2.7         5.1         1.9
#> 103         7.1         3.0         5.9         2.1
#> 104         6.3         2.9         5.6         1.8
```

```
#> 105          6.5         3.0          5.8         2.2
#> 106          7.6         3.0          6.6         2.1
#> 107          4.9         2.5          4.5         1.7
#> 108          7.3         2.9          6.3         1.8
#> 109          6.7         2.5          5.8         1.8
#> 110          7.2         3.6          6.1         2.5
#> 111          6.5         3.2          5.1         2.0
#> 112          6.4         2.7          5.3         1.9
#> 113          6.8         3.0          5.5         2.1
#> 114          5.7         2.5          5.0         2.0
#> 115          5.8         2.8          5.1         2.4
#> 116          6.4         3.2          5.3         2.3
#> 117          6.5         3.0          5.5         1.8
#> 118          7.7         3.8          6.7         2.2
#> 119          7.7         2.6          6.9         2.3
#> 120          6.0         2.2          5.0         1.5
#> 121          6.9         3.2          5.7         2.3
#> 122          5.6         2.8          4.9         2.0
#> 123          7.7         2.8          6.7         2.0
#> 124          6.3         2.7          4.9         1.8
#> 125          6.7         3.3          5.7         2.1
#> 126          7.2         3.2          6.0         1.8
#> 127          6.2         2.8          4.8         1.8
#> 128          6.1         3.0          4.9         1.8
#> 129          6.4         2.8          5.6         2.1
#> 130          7.2         3.0          5.8         1.6
#> 131          7.4         2.8          6.1         1.9
#> 132          7.9         3.8          6.4         2.0
#> 133          6.4         2.8          5.6         2.2
#> 134          6.3         2.8          5.1         1.5
#> 135          6.1         2.6          5.6         1.4
#> 136          7.7         3.0          6.1         2.3
#> 137          6.3         3.4          5.6         2.4
#> 138          6.4         3.1          5.5         1.8
#> 139          6.0         3.0          4.8         1.8
#> 140          6.9         3.1          5.4         2.1
#> 141          6.7         3.1          5.6         2.4
#> 142          6.9         3.1          5.1         2.3
#> 143          5.8         2.7          5.1         1.9
#> 144          6.8         3.2          5.9         2.3
#> 145          6.7         3.3          5.7         2.5
#> 146          6.7         3.0          5.2         2.3
#> 147          6.3         2.5          5.0         1.9
#> 148          6.5         3.0          5.2         2.0
#> 149          6.2         3.4          5.4         2.3
```

```
#> 150          5.9         3.0          5.1         1.8
#>          Species
#> 1         setosa
#> 2         setosa
#> 3         setosa
#> 4         setosa
#> 5         setosa
#> 6         setosa
#> 7         setosa
#> 8         setosa
#> 9         setosa
#> 10        setosa
#> 11        setosa
#> 12        setosa
#> 13        setosa
#> 14        setosa
#> 15        setosa
#> 16        setosa
#> 17        setosa
#> 18        setosa
#> 19        setosa
#> 20        setosa
#> 21        setosa
#> 22        setosa
#> 23        setosa
#> 24        setosa
#> 25        setosa
#> 26        setosa
#> 27        setosa
#> 28        setosa
#> 29        setosa
#> 30        setosa
#> 31        setosa
#> 32        setosa
#> 33        setosa
#> 34        setosa
#> 35        setosa
#> 36        setosa
#> 37        setosa
#> 38        setosa
#> 39        setosa
#> 40        setosa
#> 41        setosa
#> 42        setosa
#> 43        setosa
```

```
#> 44      setosa
#> 45      setosa
#> 46      setosa
#> 47      setosa
#> 48      setosa
#> 49      setosa
#> 50      setosa
#> 51  versicolor
#> 52  versicolor
#> 53  versicolor
#> 54  versicolor
#> 55  versicolor
#> 56  versicolor
#> 57  versicolor
#> 58  versicolor
#> 59  versicolor
#> 60  versicolor
#> 61  versicolor
#> 62  versicolor
#> 63  versicolor
#> 64  versicolor
#> 65  versicolor
#> 66  versicolor
#> 67  versicolor
#> 68  versicolor
#> 69  versicolor
#> 70  versicolor
#> 71  versicolor
#> 72  versicolor
#> 73  versicolor
#> 74  versicolor
#> 75  versicolor
#> 76  versicolor
#> 77  versicolor
#> 78  versicolor
#> 79  versicolor
#> 80  versicolor
#> 81  versicolor
#> 82  versicolor
#> 83  versicolor
#> 84  versicolor
#> 85  versicolor
#> 86  versicolor
#> 87  versicolor
#> 88  versicolor
```

```
#> 89  versicolor
#> 90  versicolor
#> 91  versicolor
#> 92  versicolor
#> 93  versicolor
#> 94  versicolor
#> 95  versicolor
#> 96  versicolor
#> 97  versicolor
#> 98  versicolor
#> 99  versicolor
#> 100 versicolor
#> 101  virginica
#> 102  virginica
#> 103  virginica
#> 104  virginica
#> 105  virginica
#> 106  virginica
#> 107  virginica
#> 108  virginica
#> 109  virginica
#> 110  virginica
#> 111  virginica
#> 112  virginica
#> 113  virginica
#> 114  virginica
#> 115  virginica
#> 116  virginica
#> 117  virginica
#> 118  virginica
#> 119  virginica
#> 120  virginica
#> 121  virginica
#> 122  virginica
#> 123  virginica
#> 124  virginica
#> 125  virginica
#> 126  virginica
#> 127  virginica
#> 128  virginica
#> 129  virginica
#> 130  virginica
#> 131  virginica
#> 132  virginica
#> 133  virginica
```

```
#> 134  virginica
#> 135  virginica
#> 136  virginica
#> 137  virginica
#> 138  virginica
#> 139  virginica
#> 140  virginica
#> 141  virginica
#> 142  virginica
#> 143  virginica
#> 144  virginica
#> 145  virginica
#> 146  virginica
#> 147  virginica
#> 148  virginica
#> 149  virginica
#> 150  virginica
```

The output within R Notebook is a tibble style. Try the same command in Console.

```
slice(df_iris, 1:10)
#>    Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1           5.1         3.5          1.4         0.2
#> 2           4.9         3.0          1.4         0.2
#> 3           4.7         3.2          1.3         0.2
#> 4           4.6         3.1          1.5         0.2
#> 5           5.0         3.6          1.4         0.2
#> 6           5.4         3.9          1.7         0.4
#> 7           4.6         3.4          1.4         0.3
#> 8           5.0         3.4          1.5         0.2
#> 9           4.4         2.9          1.4         0.2
#> 10          4.9         3.1          1.5         0.1
#>    Species
#> 1   setosa
#> 2   setosa
#> 3   setosa
#> 4   setosa
#> 5   setosa
#> 6   setosa
#> 7   setosa
#> 8   setosa
#> 9   setosa
#> 10  setosa
```

```
1:10
#>  [1]  1  2  3  4  5  6  7  8  9 10
```

4.2 ‘

4.2.0.1 Data Structure

Let us look at the structure of the data. You can try `str(df_iris)` on Console or by adding a code chunk in R Notebook introducing later.

```
glimpse(df_iris)
#> Rows: 150
#> Columns: 5
#> $ Sepal.Length <dbl> 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.~
#> $ Sepal.Width  <dbl> 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.~
#> $ Petal.Length <dbl> 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.~
#> $ Petal.Width  <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.~
#> $ Species      <fct> setosa, setosa, setosa, setosa, setos~
```

There are six types of data in R; Double, Integer, Character, Logical, Raw, Complex.

The names after $ are column names. If you call `df_iris$Species`, you have the Species column. Species is in the 5th collumn, `typeof(df_iris[[5]])` does the same as the next.

`df_iris[2,4]` =0.2 is the fourth entry of Sepal.Width.

```
typeof(df_iris$Species)
#> [1] "integer"
```

```
class(df_iris$Species)
#> [1] "factor"
```

For `factors = fct` see the R Document or an explanation in Factor in R: Categorical Variable & Continuous Variables.

```
typeof(df_iris$Sepal.Length)
#> [1] "double"
class(df_iris$Sepal.Length)
#> [1] "numeric"
```

Q1. What are the differences of`df_iris`, `slice(df_iris, 1:10)` and `glimpse(df_iris)` above?

Q2. What are the differences of`df_iris`, `slice(df_iris, 1:10)` and `glimpse(df_iris)` in the console?

4.2.0.2 Summary of the Data

The following is very convenient to get the summary information of a data.

```
summary(df_iris)
#>   Sepal.Length    Sepal.Width     Petal.Length
#>  Min.   :4.300   Min.   :2.000   Min.   :1.000
#>  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600
#>  Median :5.800   Median :3.000   Median :4.350
#>  Mean   :5.843   Mean   :3.057   Mean   :3.758
#>  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100
#>  Max.   :7.900   Max.   :4.400   Max.   :6.900
#>   Petal.Width          Species
#>  Min.   :0.100   setosa    :50
#>  1st Qu.:0.300   versicolor:50
#>  Median :1.300   virginica :50
#>  Mean   :1.199
#>  3rd Qu.:1.800
#>  Max.   :2.500
```

Minimum, 1st Quadrant (25%), Median, Mean, 3rd Quadrant (75%), Maximum, and the count of each factor.

4.2.1 Visualizing Data

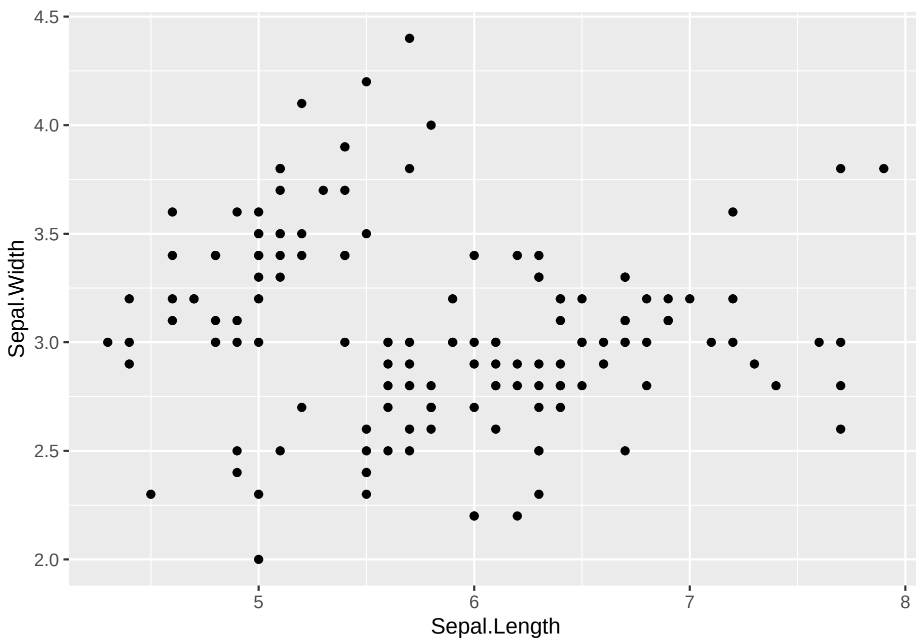
4.2.1.1 Scatter Plot

We use `ggplot` to draw graphs. The scatter plot is a projection of data with two variables x and y.

```
ggplot(data = <data>, aes(x = <column name for x>, y = <column name for y>)) +
  geom_point()
```

```
ggplot(data = df_iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_point()
```

```
ggplot(data = df_iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_point()
```

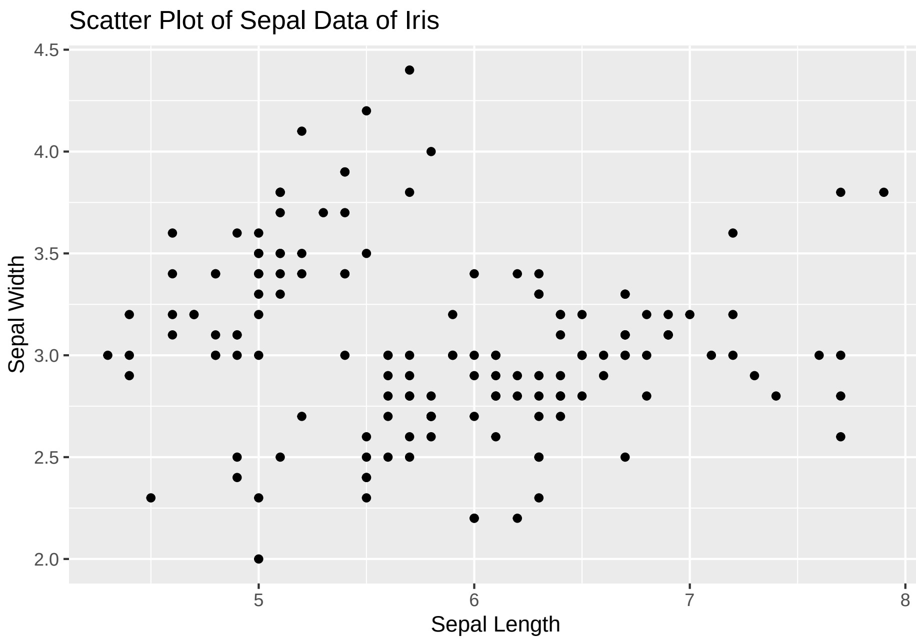


4.2.1.2 Scatter Plot with Labels

Add title and labels adding `labs()`.

```
ggplot(data = <data>, aes(x = <column name for x>, y = <column name for y>)) +
  geom_point() +
  labs(title = "Title", x = "Label for x", y = "Label for y")
```

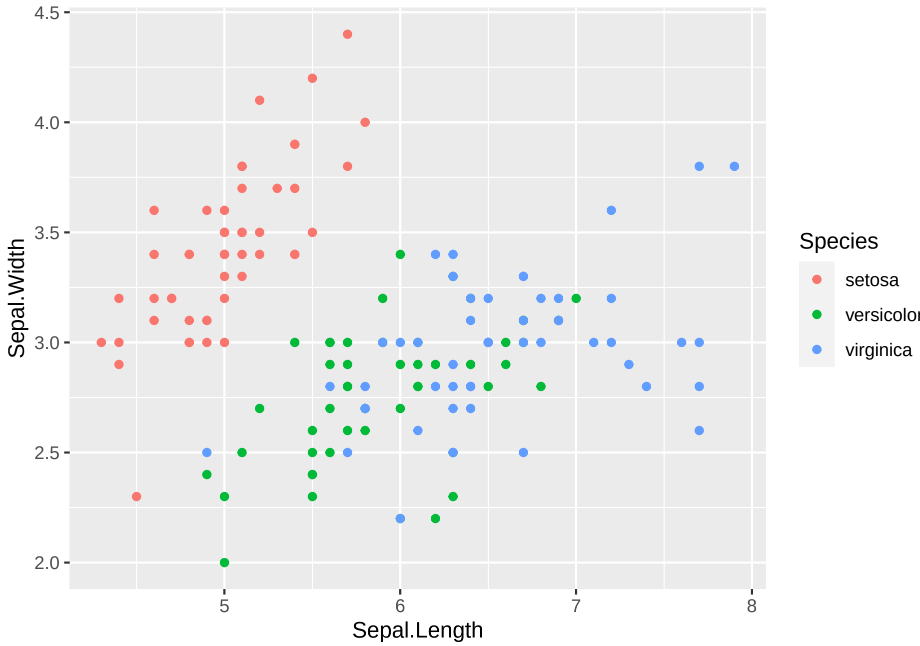
```
ggplot(data = df_iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_point() +
  labs(title = "Scatter Plot of Sepal Data of Iris", x = "Sepal Length", y = "Sepal Wid
```



4.2.1.3 Scatter Plot with Colors

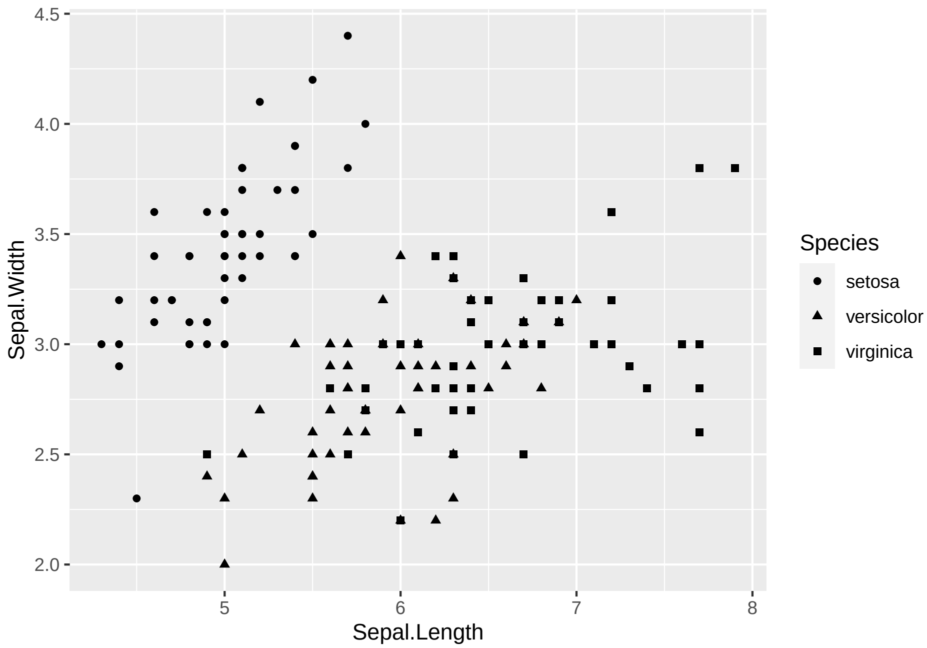
Add different colors automatically to each species. Can you see each group?

```
ggplot(data = df_iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point()
```




```
ggplot(data = df_iris, aes(x = Sepal.Length, y = Sepal.Width, shape = Species)) +
  geom_point()
```

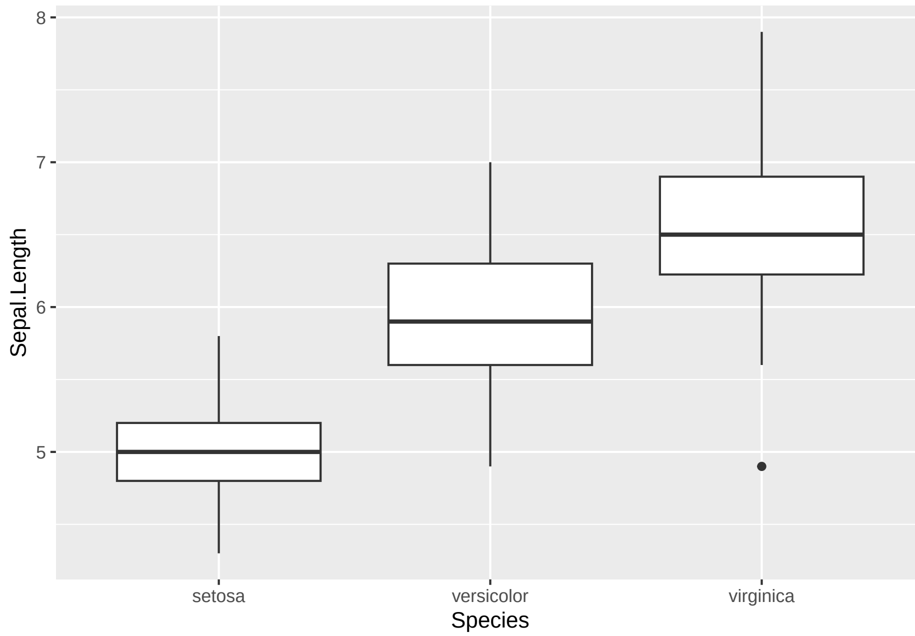
4.2.1.4 Scatter Plot with Shapes



4.2.1.5 Boxplot

The boxplot compactly displays the distribution of a continuous variable.

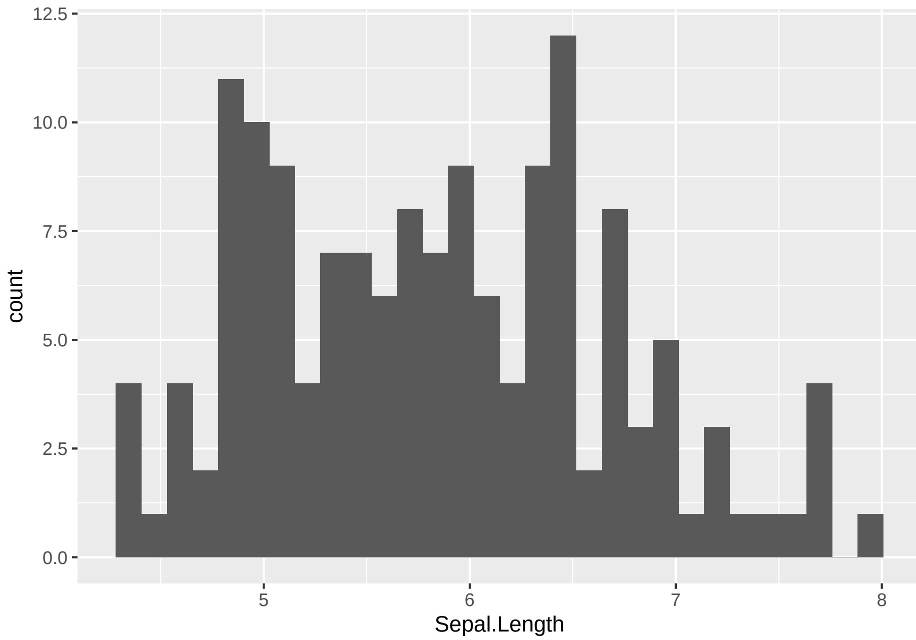
```
ggplot(data = df_iris, aes(x = Species, y = Sepal.Length)) +
  geom_boxplot()
```



4.2.1.6 Histogram

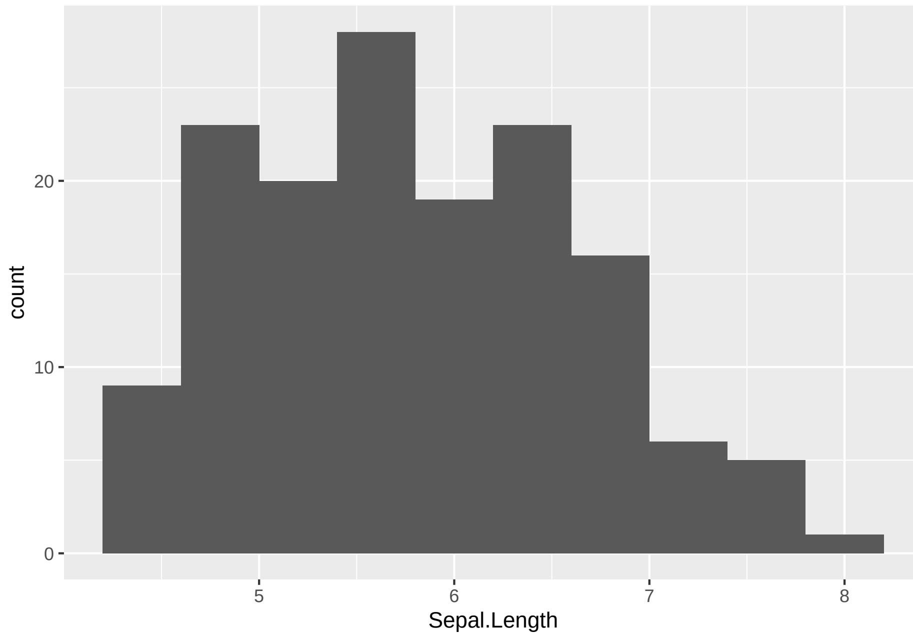
Visualize the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin. Histograms (geom_histogram()) display the counts with bars.

```
ggplot(data = df_iris, aes(x = Sepal.Length)) +
  geom_histogram()
```



Change the number of bins by `bins = <number>`.

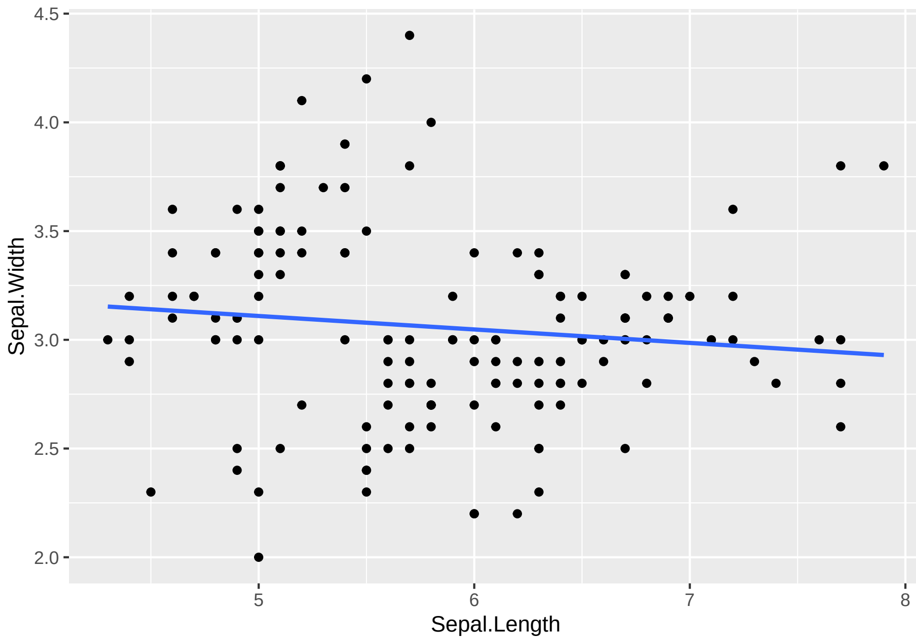
```
ggplot(data = df_iris, aes(x = Sepal.Length)) +
  geom_histogram(bins = 10)
```



4.2.2 Data Modeling

Professor Kaizoji will cover the mathematical models and hypothesis testings.

```
ggplot(data = df_iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



4.3 Comments on Week 2

4.3.0.1 Helpful Resources

- Cheat Sheet in RStudio: https://www.rstudio.com/resources/cheatsheets/
  - RStudio IED
  - Base R Cheat Sheet
- 'Quick R' by DataCamp: https://www.statmethods.net/management
- An Introduction to R

4.3.0.2 Practicum

- Posit Primers: The Basics: https://posit.cloud/learn/primers/1
  - Complete Visualization Basics and Programming Basics

4.3.0.3 Assignments - See Moodle

1. Assignment Week 2-1: Introduction Plus Forum

- Due: Tuesday, 20 December 2022, 11:59 PM

2. Assignment Week 2-2: Quiz 1 on R Basics

- Due: Tuesday, 20 December 2022, 11:59 PM

4.4 Swirl: An interactive learning environment for R and statistics

- {`swirl`} website: https://swirlstats.com
- JHU Data Science in coursera uses `swirl` for exercises.

4.4.1 Swirl Courses

1. R Programming: The basics of programming in R
2. Regression Models: The basics of regression modeling in R
3. Statistical Inference: The basics of statistical inference in R
4. Exploratory Data Analysis: The basics of exploring data in R

You can install other `swirl` courses as well

- Swirl Courses Organized by Title
- Swirl Courses Organized by Author's Name
- Github: swirl courses
  - `install_course("Course Name Here")`

4.4.2 Install and Start Swirl Courses

4.4.2.1 Three Steps to Start Swirl

```
install.packages("swirl") # Only the first time.
library(swirl) # Everytime you start swirl
swirl() # Everytime you start or resume swirl
```

4.4.2.2 R Programming: The basics of programming in R

```
 1: Basic Building Blocks      2: Workspace and Files      3: Sequences of Numbers
 4: Vectors                    5: Missing Values           6: Subsetting Vectors
 7: Matrices and Data Frames   8: Logic                    9: Functions
10: lapply and sapply         11: vapply and tapply       12: Looking at Data
13: Simulation                14: Dates and Times         15: Base Graphics
```

4.4.2.3 Recommended Sections in Order

1, 3, 4, 5, 6, 7, 12, 15, 14, 8, 9, 10, 11, 13, 2

- Section 2 discusses the directories and file systems of a computer
- Sections 9, 10, 11 are for programming

4.4.2.4 Controling a `swirl` Session

- … <– That's your cue to press Enter to continue
- You can exit swirl and return to the R prompt (>) at any time by pressing the Esc key.
- If you are already at the prompt, type bye() to exit and save your progress. When you exit properly, you'll see a short message letting you know you've done so.

When you are at the R prompt (>):

1. Typing skip() allows you to skip the current question.
2. Typing play() lets you experiment with R on your own; swirl will ignore what you do…
3. UNTIL you type nxt() which will regain swirl's attention.
4. Typing bye() causes swirl to exit. Your progress will be saved.
5. Typing main() returns you to swirl's main menu.
6. Typing info() displays these options again.

4.4.2.5 Final Remark

You will encounter the message like 'Would you like to receive credit for completing this course on Coursera.org?' at the end of each course. This is for **`coursera`** courses. Select 'NO'.

4.5 More on R Script: Examples

4.5.1 R Scripts in Moodle

- basics.R
- coronavirus.R

1. Copy a script in Moodle: *{file name}.R*
2. In RStudio (Workspace in RStudio.cloud, Project in RStudio) choose File > New File > R Script and paste it.
3. Choose File > Save with a name; e.g. *{file names}* (.R will be added automatically)

4.5.2 `basics.R`

The script with the outputs.

```
#################
#
# basics.R
#
################
# 'Quick R' by DataCamp may be a handy reference:
#     https://www.statmethods.net/management/index.html
# Cheat Sheet at RStudio: https://www.rstudio.com/resources/cheatsheets/
# Base R Cheat Sheet: https://github.com/rstudio/cheatsheets/raw/main/base-r.pdf
# To execute the line: Control + Enter (Window and Linux), Command + Enter (Mac)
## try your experiments on the console

## calculator

3 + 7

### +, -, *, /, ^ (or **), %%, %/%

3 + 10 / 2

3^2

2^3

2*2*2
```

```
### assignment: <-, (=, ->, assign())

x <- 5

x

#### object_name <- value, '<-' shortcut: Alt (option) + '-' (hyphen or minus)
#### Object names must start with a letter and can only contain letter, numbers, _ and .

this_is_a_long_name <- 5^3

this_is_a_long_name

char_name <- "What is your name?"

char_name

#### Use 'tab completion' and 'up arrow'

### ls(): list of all assignments

ls()
ls.str()

#### check Environment in the upper right pane

### (atomic) vectors

5:10

a <- seq(5,10)

a

b <- 5:10

identical(a,b)

seq(5,10,2) # same as seq(from = 5, to = 10, by = 2)

c1 <- seq(0,100, by = 10)

c2 <- seq(0,100, length.out = 10)

c1

c2

length(c1)

#### ? seq   ? length   ? identical

(die <- 1:6)

zero_one <- c(0,1) # same as 0:1

die + zero_one # c(1,2,3,4,5,6) + c(0,1). re-use

d1 <- rep(1:3,2) # repeat


d1

die == d1

d2 <- as.character(die == d1)
```

```
d2

d3 <- as.numeric(die == d1)

d3

### class() for class and typeof() for mode
### class of vectors: numeric, charcters, logical
### types of vectors: doubles, integers, characters, logicals (complex and raw)

typeof(d1); class(d1)

typeof(d2); class(d2)

typeof(d3); class(d3)

sqrt(2)

sqrt(2)^2

sqrt(2)^2 - 2

typeof(sqrt(2))

typeof(2)

typeof(2L)

5 == c(5)

length(5)

### Subsetting

(A_Z <- LETTERS)

A_F <- A_Z[1:6]

A_F

A_F[3]

A_F[c(3,5)]

large <- die > 3

large

even <- die %in% c(2,4,6)

even

A_F[large]

A_F[even]

A_F[die < 4]

### Compare df with df1 <- data.frame(number = die, alphabet = A_F)
df <- data.frame(number = die, alphabet = A_F, stringsAsFactors = FALSE)

df

df$number

df$alphabet
```

```
df[3,2]

df[4,1]

df[1]

class(df[1])

class(df[[1]])

identical(df[[1]], die)

identical(df[1],die)

####################
# The First Example
####################

plot(cars)

# Help

? cars

# cars is in the 'datasets' package

data()

# help(cars) does the same as ? cars
# You can use Help tab in the right bottom pane

help(plot)
? par

head(cars)

str(cars)

summary(cars)

x <- cars$speed
y <- cars$dist

min(x)
mean(x)
quantile(x)

plot(cars)

abline(lm(cars$dist ~ cars$speed))

summary(lm(cars$dist ~ cars$speed))

boxplot(cars)

hist(cars$speed)
hist(cars$dist)
hist(cars$dist, breaks = seq(0,120, 10))
```

4.5.3 coronavirus.R

The script and its outputs. **coronavirus.csv** is very large

```
# https://coronavirus.jhu.edu/map.html
# JHU Covid-19 global time series data
```

```r
# See R pakage coronavirus at: https://github.com/RamiKrispin/coronavirus
# Data taken from: https://github.com/RamiKrispin/coronavirus/tree/master/csv
# Last Updated
Sys.Date()

## Download and read csv (comma separated value) file
coronavirus <- read.csv("https://github.com/RamiKrispin/coronavirus/raw/master/csv/coronavirus.csv")
# write.csv(coronavirus, "data/coronavirus.csv")

## Summaries and structures of the data
head(coronavirus)
str(coronavirus)
coronavirus$date <- as.Date(coronavirus$date)
str(coronavirus)

range(coronavirus$date)
unique(coronavirus$country)
unique(coronavirus$type)

## Set Country
COUNTRY <- "Japan"
df0 <- coronavirus[coronavirus$country == COUNTRY,]
head(df0)
tail(df0)
(pop <- df0$population[1])
df <- df0[c(1,6,7,13)]
str(df)
head(df)
### alternatively,
head(df0[c("date", "type", "cases", "population")])
###

## Set types
df_confirmed <- df[df$type == "confirmed",]
df_death <- df[df$type == "death",]
df_recovery <- df[df$data_type == "recovery",]
head(df_confirmed)
head(df_death)
head(df_recovery)

## Histogram
plot(df_confirmed$date, df_confirmed$cases, type = "h")
plot(df_death$date, df_death$cases, type = "h")
# plot(df_recovered$date, df_recovered$cases, type = "h") # no data for recovery

## Scatter plot and correlation
plot(df_confirmed$cases, df_death$cases, type = "p")
cor(df_confirmed$cases, df_death$cases)


## In addition set a period
start_date <- as.Date("2021-07-01")
end_date <- Sys.Date()
df_date <- df[df$date >=start_date & df$date <= end_date,]
##

## Set types
df_date_confirmed <- df_date[df_date$type == "confirmed",]
df_date_death <- df_date[df_date$type == "death",]
df_date_recovery <- df_date[df_date$data_type == "recovery",]
head(df_date_confirmed)
head(df_date_death)
head(df_date_recovery)

## Histogram
plot(df_date_confirmed$date, df_date_confirmed$cases, type = "h")
plot(df_date_death$date, df_date_death$cases, type = "h")
```

```r
# plot(df_date_recovered$date, df_date_recovered$cases, type = "h") # no data for recovery

plot(df_date_confirmed$cases, df_date_death$cases, type = "p")
cor(df_date_confirmed$cases, df_date_death$cases)

### Q0. Change the values of the location and the period and see the outcomes.
### Q1. What is the correlation between df_confirmed$cases and df_death$cases?
### Q2. Do we have a larger correlation value if we shift the dates to implement the time-lag?
### Q3. Do you have any other questions to explore?

#### Extra
plot(df_confirmed$date, df_confirmed$cases, type = "h",
     main = paste("Comfirmed Cases in",COUNTRY),
     xlab = "Date", ylab = "Number of Cases")
```

:::

4.6 gapminder Package

4.6.1 Hans Rosling (1948 – 2017)

> Hans Rosling was a Swedish physician, academic, and public speaker. He was a professor of international health at Karolinska Institute[4] and was the co-founder and chairman of the Gapminder Foundation, which developed the Trendalyzer software system. (wikipedia)

- Books:
  - Factfulness: Ten Reasons We're Wrong About The World - And Why Things Are Better Than You Think, 2018
  - How I Learned to Understand the World: A Memoir, 2020
- Gapminder: https://www.gapminder.org
  - You are probably wrong about: Upgrade Your World View
  - Bubble Chart: Income vs Life Expectancy over time, 1800 - 2020
    - How many variables?
- Videos: The best stats you've ever seen, Hans Rosling

4.6.2 Factfulness is ... *From the book*

recognizing when a decision feels urgent and remembering that it rarely is.

To control the urgency instinct, take small steps.

- Take a breath. When your urgency instinct is triggered, your other instincts kick in and your analysis shuts down. Ask for more time and more information. It's rarely now or never and it's rarely either/or.
- Insist on the data. If something is urgent and important, it should be measured. Beware of data that is relevant but inaccurate, or accurate but irrelevant. Only relevant and accurate data is useful.
- Beware of fortune-tellers. Any prediction about the future is uncertain. Be wary of predictions that fail to acknowledge that. Insist on a full range of scenarios, never just the best or worst case. Ask how often such predictions have been right before.
- Be wary of drastic action. Ask what the side effects will be. Ask how the idea has been tested. Step-by-step practical improvements, and evaluation of their impact, are less dramatic but usually more effective.

```r
# install.packages("gapminder")
library(gapminder)
```

```r
df <- gapminder
df
#> # A tibble: 1,704 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
```

```
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
#> # ... with 1,694 more rows
```

```
glimpse(df)
#> Rows: 1,704
#> Columns: 6
#> $ country   <fct> "Afghanistan", "Afghanistan", "Afghanist~
#> $ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia~
#> $ year      <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982~
#> $ lifeExp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, ~
#> $ pop       <int> 8425333, 9240934, 10267083, 11537966, 13~
#> $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, ~
```

```
summary(df)
#>         country        continent        year
#>  Afghanistan:  12   Africa  :624   Min.   :1952
#>  Albania    :  12   Americas:300   1st Qu.:1966
#>  Algeria    :  12   Asia    :396   Median :1980
#>  Angola     :  12   Europe  :360   Mean   :1980
#>  Argentina  :  12   Oceania : 24   3rd Qu.:1993
#>  Australia  :  12                  Max.   :2007
#>  (Other)    :1632
#>     lifeExp           pop              gdpPercap
#>  Min.   :23.60   Min.   :6.001e+04   Min.   :   241.2
#>  1st Qu.:48.20   1st Qu.:2.794e+06   1st Qu.:  1202.1
#>  Median :60.71   Median :7.024e+06   Median :  3531.8
#>  Mean   :59.47   Mean   :2.960e+07   Mean   :  7215.3
#>  3rd Qu.:70.85   3rd Qu.:1.959e+07   3rd Qu.:  9325.5
#>  Max.   :82.60   Max.   :1.319e+09   Max.   :113523.1
#>
```

4.6.3 Questions

- List questions based on this data.
- What do you want to see?
- What kind of chart do you want to construct?

Review

- R on R Studio/Posit Cloud (RStudio Cloud)
- Three ways to run codes
    1. Console
    2. R Script
    3. Code Chunk in R Notebook
- Packages
    1. `tidyverse`
    2. `rmarkdown`
    3. `gapminder`

Chapter 5

R Markdown

What is R Markdown: https://vimeo.com/178485416

- Code Chunks
- Text
- YAML Metadata

5.1 What is R Markdown and R Notebook

R Markdown provides an authoring framework for data science. You can use a single R Markdown file to both

- save and execute code
- generate high quality reports that can be shared with an audience

R Notebooks are an implementation of Literate Programming that allows for direct interaction with R while producing a reproducible document with publication-quality output.

An **R Notebook** is an R Markdown document *with chunks that can be executed independently and interactively, with output visible immediately beneath the input.*

(Reference: R Markdown: The Definitive Guide, 3.2 Notebook)

5.1.1 Two Goodies

- **Important**: Implementation of Reproducible Research and Literate Programming

- **Useful** to Render into Various Formats: R Notebook (HTML), R Markdown (HTML), PDF, MS Word, MS Powerpoint, Ioslides Presentation (HTML), Slidy Presentation (HTML), Beamer Presentation (PDF), etc.

5.2 Reproducible Research and Literate Programming

5.2.1 Literate Programming by D. Knuth

Literate programming is an approach to programming introduced by Donald Knuth in which a program is given as an explanation of the program logic in a natural language, such as English, interspersed with snippets of macros and traditional source code, from which a compilable source code can be generated

5.2.2 D. Knuth

Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do.

5.2.3 Reproducible Research - Quote from a Coursera Course

Reproducible research is the idea that data analyses, and more generally, scientific claims, are published with their data and software code so that others may verify the findings and build upon them. The need for reproducibility is increasing dramatically as data analyses become more complex, involving larger datasets and more sophisticated computations. Reproducibility allows for people to focus on the actual content of a data analysis, rather than on superficial details reported in a written summary. In addition, reproducibility makes an analysis more useful to others because the data and code that actually conducted the analysis are available.

5.2.4 R Markdown workflow, R for Data Science

R Markdown is also important because it so tightly integrates prose and code. This makes it a great analysis notebook because it lets you develop code and record your thoughts. It:

- Records what you did and why you did it. Regardless of how great your memory is, if you don't record what you do, there will come a time when you have forgotten important details. Write them down so you don't forget!

- Supports rigorous thinking. You are more likely to come up with a strong analysis if you record your thoughts as you go, and continue to reflect on them. This also saves you time when you eventually write up your analysis to share with others.

- Helps others understand your work. It is rare to do data analysis by yourself, and you'll often be working as part of a team. A lab notebook helps you share why you did it with your colleagues or lab mates.

5.2.5 Records of EDA and Communication

1. Memo on a scratch paper: R Scripts
2. Record on a notebook: R Notebook (an R Markdown format)
3. Short paper or a digital communication: R Notebook
4. Paper or a report: R Markdown (html, pdf, MS Word, etc.)
5. Presentation (html, pdf, MS Powerpoint, etc.)
6. Publication of a Book

- BOOKDOWN: Write HTML, PDF, ePub, and Kindle books with R Markdown. Free online document is provided in pdf as well
- Arxive Page

5.3 Let's Get Started

1. Start R Studio - *Update R Studio if old*
2. Create a Project
3. Tool > Install Packages `rmarkdown`
    - Or on Console: `install.packages("rmarkdown")`
4. Tool > Install Packages `tinytex` (for pdf generation)
    - Alternatively, `install.packages('tinytex')`
    - If TeX is not installed: `tinytex::install_tinytex()` # install TinyTeX
        - If you are not sure, please check on `Terminal` in the left below pane:
            * which latex, which mktexlsr
5. Let's try!
    a. File > New File > R Notebook
    b. Save with a file name, say, test-notebook
    c. Preview by [Preview] button
    d. Run Code Chunk `plot(cars)` and then Preview again.
    e. Knit PDF, Word (and HTML)

5.4 Templates

5.4.1 RNotebook_Template

Template to submit your assignment of this course: `RNotebook_Template.nb.html`

```
title: "Title of R Notebook"
author: "ID and Your Name"
date: "2023-02-07"
output:
  html_notebook: null
```

5.4.1.1 YAML

- Change the title
- Write ID and your name
- Date is auto-generated and inserted. If you wish, you can replace "2023-02-07" by your favorite date style.

5.4.1.2 Code Chunk

- When you execute or run a code within the notebook, the results appear beneath the code.
- Try executing this chunk by clicking the Run button, a triangle pointing right, within the chunk or by placing your cursor inside it and pressing Ctrl+Shift+Enter (Win) or Cmd+Shift+Enter (Mac).
- Add a new chunk by clicking the Insert Chunk button on the toolbar or by pressing Ctrl+Option+I (Win) or Cmd+Option+I (Mac).
- When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the Preview button or press Ctrl+Shift+K (Win) or Cmd+Shift+K (Mac) to preview the HTML file).
- The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike Knit, *Preview does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.*

5.4.2 Testing R Markdown Formats

Various Output Formats: `test-rmarkdown.nb.html`

```
title: "Testing R Markdown Formats"
author: "DS-SL"
date: "2023-02-07"
output:
  html_notebook:
```

```
  number_sections: yes
pdf_document:
  number_sections: yes
html_document:
  df_print: paged
  number_sections: yes
word_document:
  number_sections: yes
powerpoint_presentation: default
ioslides_presentation:
  widescreen: yes
  smaller: yes
slidy_presentation: default
beamer_presentation: default
```

5.4.3 Comments on Presentation Formats and Options

- For slides, a new slide starts at ##, the second-level heading.
- `---` is page break for presentation formats.
- For Word and Powerpoint, you can add your template. See the documents in References
  - Use R Markdown to create a Word document [similar for PowerPoint]
  - Save the rendered Word file as: `ref-doc-style.docx`
  - Edit the styles of the file `ref-doc-style.docx`
  - Add `ref-doc-style.docx` as reference_doc in YAML with indention as below

```
word_document:
  number_sections: yes
  reference_doc: ref-doc-style.docx
powerpoint_presentation:
  reference_doc: ref-ppt-style.pptx
```

- You can use `Output Options` at the bottom of the gear icon next to Preview/knit button.

5.5 Markdown Language – or use WYSIWYG editor

- Headers: #, ##, ###, ####
- Lists: 1. 2. …, *
- Links: linked phrase
- Images: `![alt text](figures/filename.jpg)`
- Block quotes" > (block)
- LaTeX equations: e.g. `$\frac{a}{b}$` for $\frac{a}{b}$
- Horizontal rules: Three or more asterisks or dashes (*** or - - -)
- Tables
- Footnotes
- Bibliographies and Citations
- Slide breaks
- *Italicized text* by `_italic_`, **Bold text** by `**bold**`
- Superscripts, Subscripts, Strikethrough text

5.5.1 Visual R Markdown

R Studio introduced Visual Editor towards the end of 2021. It seems to be stable but it is not perfect to go back and forth from the original editor using tags. I always use the original editor and I am confident on all the functions of it but I do not have much experience on Visual Editor. [My Note in QALL401 2021]

- https://rstudio.github.io/visual-markdown-editing/

5.6 References

- Posit Primers: Report Reproducibly
- Markdown Quick Reference: Top Menu Bar > Help > Markdown Quick Reference
- Cheat Sheet (Top Menu Bar: Help > Cheat Sheets): RMarkdown Cheat Sheet, RMarkdown Reference Guide
- Books:
  - In Textbook: R for Data Science: Communicate
  - R Markdown: The Definitive Guide by Yihui Xie, J. J. Allaire, Garrett Grolemund
  - R Markdown Cookbook by Yihui Xie, Christophe Dervieux, Emily Riederer
- Markdown: R Markdown is based on the Markdown language of Pandoc
  - Pandoc's Markdown: Detailed Information
  - Markdown Tutorials: Interactive Practicum
  - DARING FIREBALL: Markdown (detailed explanation and editor as Dingus)
- Post error messages to a web search engine.

Chapter 6

Data Transforamtion with `dplyr`

```
library(tidyverse)
#> -- Attaching packages ------------------- tidyverse 1.3.2 --
#> v ggplot2 3.4.0      v purrr   1.0.0
#> v tibble  3.1.8      v dplyr   1.0.10
#> v tidyr   1.2.1      v stringr 1.5.0
#> v readr   2.1.3      v forcats 0.5.2
#> -- Conflicts ---------------------- tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

6.1 `dplyr` Overview

dplyr is a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges:

- `select()` picks variables based on their names.
- `filter()` picks cases based on their values.
- `mutate()` adds new variables that are functions of existing variables
- `summarise()` reduces multiple values down to a single summary.
- `arrange()` changes the ordering of the rows.
- `group_by()` takes an existing tbl and converts it into a grouped tbl.

You can learn more about them in vignette("dplyr"). As well as these single-table verbs, dplyr also provides a variety of two-table verbs, which you can learn about in vignette("two-table").

If you are new to dplyr, the best place to start is the data transformation chapter in R for data science.

6.2 `select`: Subset columns using their names and types

Helper Function	Use	Example
-	Columns except	select(babynames, -prop)
:	Columns between (inclusive)	select(babynames, year:n)
contains()	Columns that contains a string	select(babynames, contains("n"))
ends_with()	Columns that ends with a string	select(babynames, ends_with("n"))
matches()	Columns that matches a regex	select(babynames, matches("n"))
num_range()	Columns with a numerical suffix in the range	Not applicable with babynames
one_of()	Columns whose name appear in the given set	select(babynames, one_of(c("sex", "gender")))
starts_with()	Columns that starts with a string	select(babynames, starts_with("n"))

6.3 `filter`: Subset rows using column values

Logical operator	tests	Example
>	Is x greater than y?	x > y
>=	Is x greater than or equal to y?	x >= y
<	Is x less than y?	x < y
<=	Is x less than or equal to y?	x <= y
==	Is x equal to y?	x == y
!=	Is x not equal to y?	x != y
is.na()	Is x an NA?	is.na(x)
!is.na()	Is x not an NA?	!is.na(x)

6.4 `arrange` and `Pipe %>%`

- `arrange()` orders the rows of a data frame by the values of selected columns.

Unlike other `dplyr` verbs, `arrange()` largely ignores grouping; you need to explicitly mention grouping variables (‘or use .by_group = TRUE) in order to group by them, and functions of variables are evaluated once per data frame, not once per group.

- `pipes` in R for Data Science.

6.5 `mutate`

- Create, modify, and delete columns
- Useful mutate functions
  - +, -, log(), etc., for their usual mathematical meanings
  - lead(), lag()
  - dense_rank(), min_rank(), percent_rank(), row_number(), cume_dist(), ntile()
  - cumsum(), cummean(), cummin(), cummax(), cumany(), cumall()
  - na_if(), coalesce()### `group_by()` and `summarise()`

6.6 group_by

6.7 summarise or summarize

6.7.0.1 Summary functions

So far our summarise() examples have relied on sum(), max(), and mean(). But you can use any function in summarise() so long as it meets one criteria: the function must take a vector of values as input and return a single value as output. Functions that do this are known as summary functions and they are common in the field of descriptive statistics. Some of the most useful summary functions include:

1. Measures of location - mean(x), median(x), quantile(x, 0.25), min(x), and max(x)
2. Measures of spread - sd(x), var(x), IQR(x), and mad(x)
3. Measures of position - first(x), nth(x, 2), and last(x)
4. Counts - n_distinct(x) and n(), which takes no arguments, and returns the size of the current group or data frame.
5. Counts and proportions of logical values - sum(!is.na(x)), which counts the number of TRUEs returned by a logical test; mean(y == 0), which returns the proportion of TRUEs returned by a logical test.

- if_else(), recode(), case_when()

6.8 Learn dplyr by Examples

6.8.1 Data iris

```
iris
#>     Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1            5.1         3.5          1.4         0.2
#> 2            4.9         3.0          1.4         0.2
#> 3            4.7         3.2          1.3         0.2
#> 4            4.6         3.1          1.5         0.2
#> 5            5.0         3.6          1.4         0.2
#> 6            5.4         3.9          1.7         0.4
#> 7            4.6         3.4          1.4         0.3
#> 8            5.0         3.4          1.5         0.2
#> 9            4.4         2.9          1.4         0.2
#> 10           4.9         3.1          1.5         0.1
#> 11           5.4         3.7          1.5         0.2
#> 12           4.8         3.4          1.6         0.2
#> 13           4.8         3.0          1.4         0.1
#> 14           4.3         3.0          1.1         0.1
#> 15           5.8         4.0          1.2         0.2
#> 16           5.7         4.4          1.5         0.4
#> 17           5.4         3.9          1.3         0.4
#> 18           5.1         3.5          1.4         0.3
#> 19           5.7         3.8          1.7         0.3
#> 20           5.1         3.8          1.5         0.3
#> 21           5.4         3.4          1.7         0.2
#> 22           5.1         3.7          1.5         0.4
#> 23           4.6         3.6          1.0         0.2
#> 24           5.1         3.3          1.7         0.5
#> 25           4.8         3.4          1.9         0.2
#> 26           5.0         3.0          1.6         0.2
#> 27           5.0         3.4          1.6         0.4
#> 28           5.2         3.5          1.5         0.2
#> 29           5.2         3.4          1.4         0.2
#> 30           4.7         3.2          1.6         0.2
#> 31           4.8         3.1          1.6         0.2
#> 32           5.4         3.4          1.5         0.4
```

```
#> 33          5.2         4.1          1.5         0.1
#> 34          5.5         4.2          1.4         0.2
#> 35          4.9         3.1          1.5         0.2
#> 36          5.0         3.2          1.2         0.2
#> 37          5.5         3.5          1.3         0.2
#> 38          4.9         3.6          1.4         0.1
#> 39          4.4         3.0          1.3         0.2
#> 40          5.1         3.4          1.5         0.2
#> 41          5.0         3.5          1.3         0.3
#> 42          4.5         2.3          1.3         0.3
#> 43          4.4         3.2          1.3         0.2
#> 44          5.0         3.5          1.6         0.6
#> 45          5.1         3.8          1.9         0.4
#> 46          4.8         3.0          1.4         0.3
#> 47          5.1         3.8          1.6         0.2
#> 48          4.6         3.2          1.4         0.2
#> 49          5.3         3.7          1.5         0.2
#> 50          5.0         3.3          1.4         0.2
#> 51          7.0         3.2          4.7         1.4
#> 52          6.4         3.2          4.5         1.5
#> 53          6.9         3.1          4.9         1.5
#> 54          5.5         2.3          4.0         1.3
#> 55          6.5         2.8          4.6         1.5
#> 56          5.7         2.8          4.5         1.3
#> 57          6.3         3.3          4.7         1.6
#> 58          4.9         2.4          3.3         1.0
#> 59          6.6         2.9          4.6         1.3
#> 60          5.2         2.7          3.9         1.4
#> 61          5.0         2.0          3.5         1.0
#> 62          5.9         3.0          4.2         1.5
#> 63          6.0         2.2          4.0         1.0
#> 64          6.1         2.9          4.7         1.4
#> 65          5.6         2.9          3.6         1.3
#> 66          6.7         3.1          4.4         1.4
#> 67          5.6         3.0          4.5         1.5
#> 68          5.8         2.7          4.1         1.0
#> 69          6.2         2.2          4.5         1.5
#> 70          5.6         2.5          3.9         1.1
#> 71          5.9         3.2          4.8         1.8
#> 72          6.1         2.8          4.0         1.3
#> 73          6.3         2.5          4.9         1.5
#> 74          6.1         2.8          4.7         1.2
#> 75          6.4         2.9          4.3         1.3
#> 76          6.6         3.0          4.4         1.4
#> 77          6.8         2.8          4.8         1.4
#> 78          6.7         3.0          5.0         1.7
#> 79          6.0         2.9          4.5         1.5
#> 80          5.7         2.6          3.5         1.0
#> 81          5.5         2.4          3.8         1.1
#> 82          5.5         2.4          3.7         1.0
#> 83          5.8         2.7          3.9         1.2
#> 84          6.0         2.7          5.1         1.6
#> 85          5.4         3.0          4.5         1.5
#> 86          6.0         3.4          4.5         1.6
#> 87          6.7         3.1          4.7         1.5
#> 88          6.3         2.3          4.4         1.3
#> 89          5.6         3.0          4.1         1.3
#> 90          5.5         2.5          4.0         1.3
#> 91          5.5         2.6          4.4         1.2
#> 92          6.1         3.0          4.6         1.4
#> 93          5.8         2.6          4.0         1.2
#> 94          5.0         2.3          3.3         1.0
#> 95          5.6         2.7          4.2         1.3
#> 96          5.7         3.0          4.2         1.2
#> 97          5.7         2.9          4.2         1.3
#> 98          6.2         2.9          4.3         1.3
#> 99          5.1         2.5          3.0         1.1
```

```
#> 100          5.7         2.8          4.1         1.3
#> 101          6.3         3.3          6.0         2.5
#> 102          5.8         2.7          5.1         1.9
#> 103          7.1         3.0          5.9         2.1
#> 104          6.3         2.9          5.6         1.8
#> 105          6.5         3.0          5.8         2.2
#> 106          7.6         3.0          6.6         2.1
#> 107          4.9         2.5          4.5         1.7
#> 108          7.3         2.9          6.3         1.8
#> 109          6.7         2.5          5.8         1.8
#> 110          7.2         3.6          6.1         2.5
#> 111          6.5         3.2          5.1         2.0
#> 112          6.4         2.7          5.3         1.9
#> 113          6.8         3.0          5.5         2.1
#> 114          5.7         2.5          5.0         2.0
#> 115          5.8         2.8          5.1         2.4
#> 116          6.4         3.2          5.3         2.3
#> 117          6.5         3.0          5.5         1.8
#> 118          7.7         3.8          6.7         2.2
#> 119          7.7         2.6          6.9         2.3
#> 120          6.0         2.2          5.0         1.5
#> 121          6.9         3.2          5.7         2.3
#> 122          5.6         2.8          4.9         2.0
#> 123          7.7         2.8          6.7         2.0
#> 124          6.3         2.7          4.9         1.8
#> 125          6.7         3.3          5.7         2.1
#> 126          7.2         3.2          6.0         1.8
#> 127          6.2         2.8          4.8         1.8
#> 128          6.1         3.0          4.9         1.8
#> 129          6.4         2.8          5.6         2.1
#> 130          7.2         3.0          5.8         1.6
#> 131          7.4         2.8          6.1         1.9
#> 132          7.9         3.8          6.4         2.0
#> 133          6.4         2.8          5.6         2.2
#> 134          6.3         2.8          5.1         1.5
#> 135          6.1         2.6          5.6         1.4
#> 136          7.7         3.0          6.1         2.3
#> 137          6.3         3.4          5.6         2.4
#> 138          6.4         3.1          5.5         1.8
#> 139          6.0         3.0          4.8         1.8
#> 140          6.9         3.1          5.4         2.1
#> 141          6.7         3.1          5.6         2.4
#> 142          6.9         3.1          5.1         2.3
#> 143          5.8         2.7          5.1         1.9
#> 144          6.8         3.2          5.9         2.3
#> 145          6.7         3.3          5.7         2.5
#> 146          6.7         3.0          5.2         2.3
#> 147          6.3         2.5          5.0         1.9
#> 148          6.5         3.0          5.2         2.0
#> 149          6.2         3.4          5.4         2.3
#> 150          5.9         3.0          5.1         1.8
#>        Species
#> 1       setosa
#> 2       setosa
#> 3       setosa
#> 4       setosa
#> 5       setosa
#> 6       setosa
#> 7       setosa
#> 8       setosa
#> 9       setosa
#> 10      setosa
#> 11      setosa
#> 12      setosa
#> 13      setosa
#> 14      setosa
#> 15      setosa
```

```
#> 16      setosa
#> 17      setosa
#> 18      setosa
#> 19      setosa
#> 20      setosa
#> 21      setosa
#> 22      setosa
#> 23      setosa
#> 24      setosa
#> 25      setosa
#> 26      setosa
#> 27      setosa
#> 28      setosa
#> 29      setosa
#> 30      setosa
#> 31      setosa
#> 32      setosa
#> 33      setosa
#> 34      setosa
#> 35      setosa
#> 36      setosa
#> 37      setosa
#> 38      setosa
#> 39      setosa
#> 40      setosa
#> 41      setosa
#> 42      setosa
#> 43      setosa
#> 44      setosa
#> 45      setosa
#> 46      setosa
#> 47      setosa
#> 48      setosa
#> 49      setosa
#> 50      setosa
#> 51  versicolor
#> 52  versicolor
#> 53  versicolor
#> 54  versicolor
#> 55  versicolor
#> 56  versicolor
#> 57  versicolor
#> 58  versicolor
#> 59  versicolor
#> 60  versicolor
#> 61  versicolor
#> 62  versicolor
#> 63  versicolor
#> 64  versicolor
#> 65  versicolor
#> 66  versicolor
#> 67  versicolor
#> 68  versicolor
#> 69  versicolor
#> 70  versicolor
#> 71  versicolor
#> 72  versicolor
#> 73  versicolor
#> 74  versicolor
#> 75  versicolor
#> 76  versicolor
#> 77  versicolor
#> 78  versicolor
#> 79  versicolor
#> 80  versicolor
#> 81  versicolor
#> 82  versicolor
```

```
#> 83  versicolor
#> 84  versicolor
#> 85  versicolor
#> 86  versicolor
#> 87  versicolor
#> 88  versicolor
#> 89  versicolor
#> 90  versicolor
#> 91  versicolor
#> 92  versicolor
#> 93  versicolor
#> 94  versicolor
#> 95  versicolor
#> 96  versicolor
#> 97  versicolor
#> 98  versicolor
#> 99  versicolor
#> 100 versicolor
#> 101  virginica
#> 102  virginica
#> 103  virginica
#> 104  virginica
#> 105  virginica
#> 106  virginica
#> 107  virginica
#> 108  virginica
#> 109  virginica
#> 110  virginica
#> 111  virginica
#> 112  virginica
#> 113  virginica
#> 114  virginica
#> 115  virginica
#> 116  virginica
#> 117  virginica
#> 118  virginica
#> 119  virginica
#> 120  virginica
#> 121  virginica
#> 122  virginica
#> 123  virginica
#> 124  virginica
#> 125  virginica
#> 126  virginica
#> 127  virginica
#> 128  virginica
#> 129  virginica
#> 130  virginica
#> 131  virginica
#> 132  virginica
#> 133  virginica
#> 134  virginica
#> 135  virginica
#> 136  virginica
#> 137  virginica
#> 138  virginica
#> 139  virginica
#> 140  virginica
#> 141  virginica
#> 142  virginica
#> 143  virginica
#> 144  virginica
#> 145  virginica
#> 146  virginica
#> 147  virginica
#> 148  virginica
#> 149  virginica
```

```
#> 150  virginica
```

```
summary(iris)
#>   Sepal.Length    Sepal.Width     Petal.Length
#>  Min.   :4.300   Min.   :2.000   Min.   :1.000
#>  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600
#>  Median :5.800   Median :3.000   Median :4.350
#>  Mean   :5.843   Mean   :3.057   Mean   :3.758
#>  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100
#>  Max.   :7.900   Max.   :4.400   Max.   :6.900
#>   Petal.Width          Species
#>  Min.   :0.100   setosa    :50
#>  1st Qu.:0.300   versicolor:50
#>  Median :1.300   virginica :50
#>  Mean   :1.199
#>  3rd Qu.:1.800
#>  Max.   :2.500
```

6.8.2 select 1 - columns 1, 2, 5

```
select(iris, c(1,2,5))
#>     Sepal.Length Sepal.Width    Species
#> 1            5.1         3.5     setosa
#> 2            4.9         3.0     setosa
#> 3            4.7         3.2     setosa
#> 4            4.6         3.1     setosa
#> 5            5.0         3.6     setosa
#> 6            5.4         3.9     setosa
#> 7            4.6         3.4     setosa
#> 8            5.0         3.4     setosa
#> 9            4.4         2.9     setosa
#> 10           4.9         3.1     setosa
#> 11           5.4         3.7     setosa
#> 12           4.8         3.4     setosa
#> 13           4.8         3.0     setosa
#> 14           4.3         3.0     setosa
#> 15           5.8         4.0     setosa
#> 16           5.7         4.4     setosa
#> 17           5.4         3.9     setosa
#> 18           5.1         3.5     setosa
#> 19           5.7         3.8     setosa
#> 20           5.1         3.8     setosa
#> 21           5.4         3.4     setosa
#> 22           5.1         3.7     setosa
#> 23           4.6         3.6     setosa
#> 24           5.1         3.3     setosa
#> 25           4.8         3.4     setosa
#> 26           5.0         3.0     setosa
#> 27           5.0         3.4     setosa
#> 28           5.2         3.5     setosa
#> 29           5.2         3.4     setosa
#> 30           4.7         3.2     setosa
#> 31           4.8         3.1     setosa
#> 32           5.4         3.4     setosa
#> 33           5.2         4.1     setosa
#> 34           5.5         4.2     setosa
#> 35           4.9         3.1     setosa
#> 36           5.0         3.2     setosa
#> 37           5.5         3.5     setosa
#> 38           4.9         3.6     setosa
#> 39           4.4         3.0     setosa
#> 40           5.1         3.4     setosa
#> 41           5.0         3.5     setosa
```

```
#> 42          4.5         2.3     setosa
#> 43          4.4         3.2     setosa
#> 44          5.0         3.5     setosa
#> 45          5.1         3.8     setosa
#> 46          4.8         3.0     setosa
#> 47          5.1         3.8     setosa
#> 48          4.6         3.2     setosa
#> 49          5.3         3.7     setosa
#> 50          5.0         3.3     setosa
#> 51          7.0         3.2 versicolor
#> 52          6.4         3.2 versicolor
#> 53          6.9         3.1 versicolor
#> 54          5.5         2.3 versicolor
#> 55          6.5         2.8 versicolor
#> 56          5.7         2.8 versicolor
#> 57          6.3         3.3 versicolor
#> 58          4.9         2.4 versicolor
#> 59          6.6         2.9 versicolor
#> 60          5.2         2.7 versicolor
#> 61          5.0         2.0 versicolor
#> 62          5.9         3.0 versicolor
#> 63          6.0         2.2 versicolor
#> 64          6.1         2.9 versicolor
#> 65          5.6         2.9 versicolor
#> 66          6.7         3.1 versicolor
#> 67          5.6         3.0 versicolor
#> 68          5.8         2.7 versicolor
#> 69          6.2         2.2 versicolor
#> 70          5.6         2.5 versicolor
#> 71          5.9         3.2 versicolor
#> 72          6.1         2.8 versicolor
#> 73          6.3         2.5 versicolor
#> 74          6.1         2.8 versicolor
#> 75          6.4         2.9 versicolor
#> 76          6.6         3.0 versicolor
#> 77          6.8         2.8 versicolor
#> 78          6.7         3.0 versicolor
#> 79          6.0         2.9 versicolor
#> 80          5.7         2.6 versicolor
#> 81          5.5         2.4 versicolor
#> 82          5.5         2.4 versicolor
#> 83          5.8         2.7 versicolor
#> 84          6.0         2.7 versicolor
#> 85          5.4         3.0 versicolor
#> 86          6.0         3.4 versicolor
#> 87          6.7         3.1 versicolor
#> 88          6.3         2.3 versicolor
#> 89          5.6         3.0 versicolor
#> 90          5.5         2.5 versicolor
#> 91          5.5         2.6 versicolor
#> 92          6.1         3.0 versicolor
#> 93          5.8         2.6 versicolor
#> 94          5.0         2.3 versicolor
#> 95          5.6         2.7 versicolor
#> 96          5.7         3.0 versicolor
#> 97          5.7         2.9 versicolor
#> 98          6.2         2.9 versicolor
#> 99          5.1         2.5 versicolor
#> 100         5.7         2.8 versicolor
#> 101         6.3         3.3  virginica
#> 102         5.8         2.7  virginica
#> 103         7.1         3.0  virginica
#> 104         6.3         2.9  virginica
#> 105         6.5         3.0  virginica
#> 106         7.6         3.0  virginica
#> 107         4.9         2.5  virginica
#> 108         7.3         2.9  virginica
```

```
#> 109          6.7         2.5  virginica
#> 110          7.2         3.6  virginica
#> 111          6.5         3.2  virginica
#> 112          6.4         2.7  virginica
#> 113          6.8         3.0  virginica
#> 114          5.7         2.5  virginica
#> 115          5.8         2.8  virginica
#> 116          6.4         3.2  virginica
#> 117          6.5         3.0  virginica
#> 118          7.7         3.8  virginica
#> 119          7.7         2.6  virginica
#> 120          6.0         2.2  virginica
#> 121          6.9         3.2  virginica
#> 122          5.6         2.8  virginica
#> 123          7.7         2.8  virginica
#> 124          6.3         2.7  virginica
#> 125          6.7         3.3  virginica
#> 126          7.2         3.2  virginica
#> 127          6.2         2.8  virginica
#> 128          6.1         3.0  virginica
#> 129          6.4         2.8  virginica
#> 130          7.2         3.0  virginica
#> 131          7.4         2.8  virginica
#> 132          7.9         3.8  virginica
#> 133          6.4         2.8  virginica
#> 134          6.3         2.8  virginica
#> 135          6.1         2.6  virginica
#> 136          7.7         3.0  virginica
#> 137          6.3         3.4  virginica
#> 138          6.4         3.1  virginica
#> 139          6.0         3.0  virginica
#> 140          6.9         3.1  virginica
#> 141          6.7         3.1  virginica
#> 142          6.9         3.1  virginica
#> 143          5.8         2.7  virginica
#> 144          6.8         3.2  virginica
#> 145          6.7         3.3  virginica
#> 146          6.7         3.0  virginica
#> 147          6.3         2.5  virginica
#> 148          6.5         3.0  virginica
#> 149          6.2         3.4  virginica
#> 150          5.9         3.0  virginica
```

6.8.3 select 2 - except Species

```
select(iris, -Species)
#>     Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1            5.1         3.5          1.4         0.2
#> 2            4.9         3.0          1.4         0.2
#> 3            4.7         3.2          1.3         0.2
#> 4            4.6         3.1          1.5         0.2
#> 5            5.0         3.6          1.4         0.2
#> 6            5.4         3.9          1.7         0.4
#> 7            4.6         3.4          1.4         0.3
#> 8            5.0         3.4          1.5         0.2
#> 9            4.4         2.9          1.4         0.2
#> 10           4.9         3.1          1.5         0.1
#> 11           5.4         3.7          1.5         0.2
#> 12           4.8         3.4          1.6         0.2
#> 13           4.8         3.0          1.4         0.1
#> 14           4.3         3.0          1.1         0.1
#> 15           5.8         4.0          1.2         0.2
#> 16           5.7         4.4          1.5         0.4
#> 17           5.4         3.9          1.3         0.4
```

```
#> 18          5.1         3.5          1.4         0.3
#> 19          5.7         3.8          1.7         0.3
#> 20          5.1         3.8          1.5         0.3
#> 21          5.4         3.4          1.7         0.2
#> 22          5.1         3.7          1.5         0.4
#> 23          4.6         3.6          1.0         0.2
#> 24          5.1         3.3          1.7         0.5
#> 25          4.8         3.4          1.9         0.2
#> 26          5.0         3.0          1.6         0.2
#> 27          5.0         3.4          1.6         0.4
#> 28          5.2         3.5          1.5         0.2
#> 29          5.2         3.4          1.4         0.2
#> 30          4.7         3.2          1.6         0.2
#> 31          4.8         3.1          1.6         0.2
#> 32          5.4         3.4          1.5         0.4
#> 33          5.2         4.1          1.5         0.1
#> 34          5.5         4.2          1.4         0.2
#> 35          4.9         3.1          1.5         0.2
#> 36          5.0         3.2          1.2         0.2
#> 37          5.5         3.5          1.3         0.2
#> 38          4.9         3.6          1.4         0.1
#> 39          4.4         3.0          1.3         0.2
#> 40          5.1         3.4          1.5         0.2
#> 41          5.0         3.5          1.3         0.3
#> 42          4.5         2.3          1.3         0.3
#> 43          4.4         3.2          1.3         0.2
#> 44          5.0         3.5          1.6         0.6
#> 45          5.1         3.8          1.9         0.4
#> 46          4.8         3.0          1.4         0.3
#> 47          5.1         3.8          1.6         0.2
#> 48          4.6         3.2          1.4         0.2
#> 49          5.3         3.7          1.5         0.2
#> 50          5.0         3.3          1.4         0.2
#> 51          7.0         3.2          4.7         1.4
#> 52          6.4         3.2          4.5         1.5
#> 53          6.9         3.1          4.9         1.5
#> 54          5.5         2.3          4.0         1.3
#> 55          6.5         2.8          4.6         1.5
#> 56          5.7         2.8          4.5         1.3
#> 57          6.3         3.3          4.7         1.6
#> 58          4.9         2.4          3.3         1.0
#> 59          6.6         2.9          4.6         1.3
#> 60          5.2         2.7          3.9         1.4
#> 61          5.0         2.0          3.5         1.0
#> 62          5.9         3.0          4.2         1.5
#> 63          6.0         2.2          4.0         1.0
#> 64          6.1         2.9          4.7         1.4
#> 65          5.6         2.9          3.6         1.3
#> 66          6.7         3.1          4.4         1.4
#> 67          5.6         3.0          4.5         1.5
#> 68          5.8         2.7          4.1         1.0
#> 69          6.2         2.2          4.5         1.5
#> 70          5.6         2.5          3.9         1.1
#> 71          5.9         3.2          4.8         1.8
#> 72          6.1         2.8          4.0         1.3
#> 73          6.3         2.5          4.9         1.5
#> 74          6.1         2.8          4.7         1.2
#> 75          6.4         2.9          4.3         1.3
#> 76          6.6         3.0          4.4         1.4
#> 77          6.8         2.8          4.8         1.4
#> 78          6.7         3.0          5.0         1.7
#> 79          6.0         2.9          4.5         1.5
#> 80          5.7         2.6          3.5         1.0
#> 81          5.5         2.4          3.8         1.1
#> 82          5.5         2.4          3.7         1.0
#> 83          5.8         2.7          3.9         1.2
#> 84          6.0         2.7          5.1         1.6
```

```
#> 85          5.4         3.0          4.5         1.5
#> 86          6.0         3.4          4.5         1.6
#> 87          6.7         3.1          4.7         1.5
#> 88          6.3         2.3          4.4         1.3
#> 89          5.6         3.0          4.1         1.3
#> 90          5.5         2.5          4.0         1.3
#> 91          5.5         2.6          4.4         1.2
#> 92          6.1         3.0          4.6         1.4
#> 93          5.8         2.6          4.0         1.2
#> 94          5.0         2.3          3.3         1.0
#> 95          5.6         2.7          4.2         1.3
#> 96          5.7         3.0          4.2         1.2
#> 97          5.7         2.9          4.2         1.3
#> 98          6.2         2.9          4.3         1.3
#> 99          5.1         2.5          3.0         1.1
#> 100         5.7         2.8          4.1         1.3
#> 101         6.3         3.3          6.0         2.5
#> 102         5.8         2.7          5.1         1.9
#> 103         7.1         3.0          5.9         2.1
#> 104         6.3         2.9          5.6         1.8
#> 105         6.5         3.0          5.8         2.2
#> 106         7.6         3.0          6.6         2.1
#> 107         4.9         2.5          4.5         1.7
#> 108         7.3         2.9          6.3         1.8
#> 109         6.7         2.5          5.8         1.8
#> 110         7.2         3.6          6.1         2.5
#> 111         6.5         3.2          5.1         2.0
#> 112         6.4         2.7          5.3         1.9
#> 113         6.8         3.0          5.5         2.1
#> 114         5.7         2.5          5.0         2.0
#> 115         5.8         2.8          5.1         2.4
#> 116         6.4         3.2          5.3         2.3
#> 117         6.5         3.0          5.5         1.8
#> 118         7.7         3.8          6.7         2.2
#> 119         7.7         2.6          6.9         2.3
#> 120         6.0         2.2          5.0         1.5
#> 121         6.9         3.2          5.7         2.3
#> 122         5.6         2.8          4.9         2.0
#> 123         7.7         2.8          6.7         2.0
#> 124         6.3         2.7          4.9         1.8
#> 125         6.7         3.3          5.7         2.1
#> 126         7.2         3.2          6.0         1.8
#> 127         6.2         2.8          4.8         1.8
#> 128         6.1         3.0          4.9         1.8
#> 129         6.4         2.8          5.6         2.1
#> 130         7.2         3.0          5.8         1.6
#> 131         7.4         2.8          6.1         1.9
#> 132         7.9         3.8          6.4         2.0
#> 133         6.4         2.8          5.6         2.2
#> 134         6.3         2.8          5.1         1.5
#> 135         6.1         2.6          5.6         1.4
#> 136         7.7         3.0          6.1         2.3
#> 137         6.3         3.4          5.6         2.4
#> 138         6.4         3.1          5.5         1.8
#> 139         6.0         3.0          4.8         1.8
#> 140         6.9         3.1          5.4         2.1
#> 141         6.7         3.1          5.6         2.4
#> 142         6.9         3.1          5.1         2.3
#> 143         5.8         2.7          5.1         1.9
#> 144         6.8         3.2          5.9         2.3
#> 145         6.7         3.3          5.7         2.5
#> 146         6.7         3.0          5.2         2.3
#> 147         6.3         2.5          5.0         1.9
#> 148         6.5         3.0          5.2         2.0
#> 149         6.2         3.4          5.4         2.3
#> 150         5.9         3.0          5.1         1.8
```

6.8.4 select 3 - change column names

```r
select(iris, sl = Sepal.Length, sw = Sepal.Width, sp = Species)
#>      sl  sw         sp
#> 1   5.1 3.5     setosa
#> 2   4.9 3.0     setosa
#> 3   4.7 3.2     setosa
#> 4   4.6 3.1     setosa
#> 5   5.0 3.6     setosa
#> 6   5.4 3.9     setosa
#> 7   4.6 3.4     setosa
#> 8   5.0 3.4     setosa
#> 9   4.4 2.9     setosa
#> 10  4.9 3.1     setosa
#> 11  5.4 3.7     setosa
#> 12  4.8 3.4     setosa
#> 13  4.8 3.0     setosa
#> 14  4.3 3.0     setosa
#> 15  5.8 4.0     setosa
#> 16  5.7 4.4     setosa
#> 17  5.4 3.9     setosa
#> 18  5.1 3.5     setosa
#> 19  5.7 3.8     setosa
#> 20  5.1 3.8     setosa
#> 21  5.4 3.4     setosa
#> 22  5.1 3.7     setosa
#> 23  4.6 3.6     setosa
#> 24  5.1 3.3     setosa
#> 25  4.8 3.4     setosa
#> 26  5.0 3.0     setosa
#> 27  5.0 3.4     setosa
#> 28  5.2 3.5     setosa
#> 29  5.2 3.4     setosa
#> 30  4.7 3.2     setosa
#> 31  4.8 3.1     setosa
#> 32  5.4 3.4     setosa
#> 33  5.2 4.1     setosa
#> 34  5.5 4.2     setosa
#> 35  4.9 3.1     setosa
#> 36  5.0 3.2     setosa
#> 37  5.5 3.5     setosa
#> 38  4.9 3.6     setosa
#> 39  4.4 3.0     setosa
#> 40  5.1 3.4     setosa
#> 41  5.0 3.5     setosa
#> 42  4.5 2.3     setosa
#> 43  4.4 3.2     setosa
#> 44  5.0 3.5     setosa
#> 45  5.1 3.8     setosa
#> 46  4.8 3.0     setosa
#> 47  5.1 3.8     setosa
#> 48  4.6 3.2     setosa
#> 49  5.3 3.7     setosa
#> 50  5.0 3.3     setosa
#> 51  7.0 3.2 versicolor
#> 52  6.4 3.2 versicolor
#> 53  6.9 3.1 versicolor
#> 54  5.5 2.3 versicolor
#> 55  6.5 2.8 versicolor
#> 56  5.7 2.8 versicolor
#> 57  6.3 3.3 versicolor
#> 58  4.9 2.4 versicolor
#> 59  6.6 2.9 versicolor
#> 60  5.2 2.7 versicolor
#> 61  5.0 2.0 versicolor
#> 62  5.9 3.0 versicolor
#> 63  6.0 2.2 versicolor
#> 64  6.1 2.9 versicolor
```

```
#> 65  5.6 2.9 versicolor
#> 66  6.7 3.1 versicolor
#> 67  5.6 3.0 versicolor
#> 68  5.8 2.7 versicolor
#> 69  6.2 2.2 versicolor
#> 70  5.6 2.5 versicolor
#> 71  5.9 3.2 versicolor
#> 72  6.1 2.8 versicolor
#> 73  6.3 2.5 versicolor
#> 74  6.1 2.8 versicolor
#> 75  6.4 2.9 versicolor
#> 76  6.6 3.0 versicolor
#> 77  6.8 2.8 versicolor
#> 78  6.7 3.0 versicolor
#> 79  6.0 2.9 versicolor
#> 80  5.7 2.6 versicolor
#> 81  5.5 2.4 versicolor
#> 82  5.5 2.4 versicolor
#> 83  5.8 2.7 versicolor
#> 84  6.0 2.7 versicolor
#> 85  5.4 3.0 versicolor
#> 86  6.0 3.4 versicolor
#> 87  6.7 3.1 versicolor
#> 88  6.3 2.3 versicolor
#> 89  5.6 3.0 versicolor
#> 90  5.5 2.5 versicolor
#> 91  5.5 2.6 versicolor
#> 92  6.1 3.0 versicolor
#> 93  5.8 2.6 versicolor
#> 94  5.0 2.3 versicolor
#> 95  5.6 2.7 versicolor
#> 96  5.7 3.0 versicolor
#> 97  5.7 2.9 versicolor
#> 98  6.2 2.9 versicolor
#> 99  5.1 2.5 versicolor
#> 100 5.7 2.8 versicolor
#> 101 6.3 3.3  virginica
#> 102 5.8 2.7  virginica
#> 103 7.1 3.0  virginica
#> 104 6.3 2.9  virginica
#> 105 6.5 3.0  virginica
#> 106 7.6 3.0  virginica
#> 107 4.9 2.5  virginica
#> 108 7.3 2.9  virginica
#> 109 6.7 2.5  virginica
#> 110 7.2 3.6  virginica
#> 111 6.5 3.2  virginica
#> 112 6.4 2.7  virginica
#> 113 6.8 3.0  virginica
#> 114 5.7 2.5  virginica
#> 115 5.8 2.8  virginica
#> 116 6.4 3.2  virginica
#> 117 6.5 3.0  virginica
#> 118 7.7 3.8  virginica
#> 119 7.7 2.6  virginica
#> 120 6.0 2.2  virginica
#> 121 6.9 3.2  virginica
#> 122 5.6 2.8  virginica
#> 123 7.7 2.8  virginica
#> 124 6.3 2.7  virginica
#> 125 6.7 3.3  virginica
#> 126 7.2 3.2  virginica
#> 127 6.2 2.8  virginica
#> 128 6.1 3.0  virginica
#> 129 6.4 2.8  virginica
#> 130 7.2 3.0  virginica
#> 131 7.4 2.8  virginica
```

```
#> 132 7.9 3.8  virginica
#> 133 6.4 2.8  virginica
#> 134 6.3 2.8  virginica
#> 135 6.1 2.6  virginica
#> 136 7.7 3.0  virginica
#> 137 6.3 3.4  virginica
#> 138 6.4 3.1  virginica
#> 139 6.0 3.0  virginica
#> 140 6.9 3.1  virginica
#> 141 6.7 3.1  virginica
#> 142 6.9 3.1  virginica
#> 143 5.8 2.7  virginica
#> 144 6.8 3.2  virginica
#> 145 6.7 3.3  virginica
#> 146 6.7 3.0  virginica
#> 147 6.3 2.5  virginica
#> 148 6.5 3.0  virginica
#> 149 6.2 3.4  virginica
#> 150 5.9 3.0  virginica
```

6.8.5 `filter` - by names

```
filter(iris, Species == "virginica")
#>    Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1           6.3         3.3          6.0         2.5
#> 2           5.8         2.7          5.1         1.9
#> 3           7.1         3.0          5.9         2.1
#> 4           6.3         2.9          5.6         1.8
#> 5           6.5         3.0          5.8         2.2
#> 6           7.6         3.0          6.6         2.1
#> 7           4.9         2.5          4.5         1.7
#> 8           7.3         2.9          6.3         1.8
#> 9           6.7         2.5          5.8         1.8
#> 10          7.2         3.6          6.1         2.5
#> 11          6.5         3.2          5.1         2.0
#> 12          6.4         2.7          5.3         1.9
#> 13          6.8         3.0          5.5         2.1
#> 14          5.7         2.5          5.0         2.0
#> 15          5.8         2.8          5.1         2.4
#> 16          6.4         3.2          5.3         2.3
#> 17          6.5         3.0          5.5         1.8
#> 18          7.7         3.8          6.7         2.2
#> 19          7.7         2.6          6.9         2.3
#> 20          6.0         2.2          5.0         1.5
#> 21          6.9         3.2          5.7         2.3
#> 22          5.6         2.8          4.9         2.0
#> 23          7.7         2.8          6.7         2.0
#> 24          6.3         2.7          4.9         1.8
#> 25          6.7         3.3          5.7         2.1
#> 26          7.2         3.2          6.0         1.8
#> 27          6.2         2.8          4.8         1.8
#> 28          6.1         3.0          4.9         1.8
#> 29          6.4         2.8          5.6         2.1
#> 30          7.2         3.0          5.8         1.6
#> 31          7.4         2.8          6.1         1.9
#> 32          7.9         3.8          6.4         2.0
#> 33          6.4         2.8          5.6         2.2
#> 34          6.3         2.8          5.1         1.5
#> 35          6.1         2.6          5.6         1.4
#> 36          7.7         3.0          6.1         2.3
#> 37          6.3         3.4          5.6         2.4
#> 38          6.4         3.1          5.5         1.8
#> 39          6.0         3.0          4.8         1.8
#> 40          6.9         3.1          5.4         2.1
```

```
#> 41          6.7         3.1          5.6         2.4
#> 42          6.9         3.1          5.1         2.3
#> 43          5.8         2.7          5.1         1.9
#> 44          6.8         3.2          5.9         2.3
#> 45          6.7         3.3          5.7         2.5
#> 46          6.7         3.0          5.2         2.3
#> 47          6.3         2.5          5.0         1.9
#> 48          6.5         3.0          5.2         2.0
#> 49          6.2         3.4          5.4         2.3
#> 50          5.9         3.0          5.1         1.8
#>      Species
#> 1  virginica
#> 2  virginica
#> 3  virginica
#> 4  virginica
#> 5  virginica
#> 6  virginica
#> 7  virginica
#> 8  virginica
#> 9  virginica
#> 10 virginica
#> 11 virginica
#> 12 virginica
#> 13 virginica
#> 14 virginica
#> 15 virginica
#> 16 virginica
#> 17 virginica
#> 18 virginica
#> 19 virginica
#> 20 virginica
#> 21 virginica
#> 22 virginica
#> 23 virginica
#> 24 virginica
#> 25 virginica
#> 26 virginica
#> 27 virginica
#> 28 virginica
#> 29 virginica
#> 30 virginica
#> 31 virginica
#> 32 virginica
#> 33 virginica
#> 34 virginica
#> 35 virginica
#> 36 virginica
#> 37 virginica
#> 38 virginica
#> 39 virginica
#> 40 virginica
#> 41 virginica
#> 42 virginica
#> 43 virginica
#> 44 virginica
#> 45 virginica
#> 46 virginica
#> 47 virginica
#> 48 virginica
#> 49 virginica
#> 50 virginica
```

6.8.6 `arrange` - ascending and descending order

```
arrange(iris, Sepal.Length, desc(Sepal.Width))
#>     Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1            4.3         3.0          1.1         0.1
#> 2            4.4         3.2          1.3         0.2
#> 3            4.4         3.0          1.3         0.2
#> 4            4.4         2.9          1.4         0.2
#> 5            4.5         2.3          1.3         0.3
#> 6            4.6         3.6          1.0         0.2
#> 7            4.6         3.4          1.4         0.3
#> 8            4.6         3.2          1.4         0.2
#> 9            4.6         3.1          1.5         0.2
#> 10           4.7         3.2          1.3         0.2
#> 11           4.7         3.2          1.6         0.2
#> 12           4.8         3.4          1.6         0.2
#> 13           4.8         3.4          1.9         0.2
#> 14           4.8         3.1          1.6         0.2
#> 15           4.8         3.0          1.4         0.1
#> 16           4.8         3.0          1.4         0.3
#> 17           4.9         3.6          1.4         0.1
#> 18           4.9         3.1          1.5         0.1
#> 19           4.9         3.1          1.5         0.2
#> 20           4.9         3.0          1.4         0.2
#> 21           4.9         2.5          4.5         1.7
#> 22           4.9         2.4          3.3         1.0
#> 23           5.0         3.6          1.4         0.2
#> 24           5.0         3.5          1.3         0.3
#> 25           5.0         3.5          1.6         0.6
#> 26           5.0         3.4          1.5         0.2
#> 27           5.0         3.4          1.6         0.4
#> 28           5.0         3.3          1.4         0.2
#> 29           5.0         3.2          1.2         0.2
#> 30           5.0         3.0          1.6         0.2
#> 31           5.0         2.3          3.3         1.0
#> 32           5.0         2.0          3.5         1.0
#> 33           5.1         3.8          1.5         0.3
#> 34           5.1         3.8          1.9         0.4
#> 35           5.1         3.8          1.6         0.2
#> 36           5.1         3.7          1.5         0.4
#> 37           5.1         3.5          1.4         0.2
#> 38           5.1         3.5          1.4         0.3
#> 39           5.1         3.4          1.5         0.2
#> 40           5.1         3.3          1.7         0.5
#> 41           5.1         2.5          3.0         1.1
#> 42           5.2         4.1          1.5         0.1
#> 43           5.2         3.5          1.5         0.2
#> 44           5.2         3.4          1.4         0.2
#> 45           5.2         2.7          3.9         1.4
#> 46           5.3         3.7          1.5         0.2
#> 47           5.4         3.9          1.7         0.4
#> 48           5.4         3.9          1.3         0.4
#> 49           5.4         3.7          1.5         0.2
#> 50           5.4         3.4          1.7         0.2
#> 51           5.4         3.4          1.5         0.4
#> 52           5.4         3.0          4.5         1.5
#> 53           5.5         4.2          1.4         0.2
#> 54           5.5         3.5          1.3         0.2
#> 55           5.5         2.6          4.4         1.2
#> 56           5.5         2.5          4.0         1.3
#> 57           5.5         2.4          3.8         1.1
#> 58           5.5         2.4          3.7         1.0
#> 59           5.5         2.3          4.0         1.3
#> 60           5.6         3.0          4.5         1.5
#> 61           5.6         3.0          4.1         1.3
#> 62           5.6         2.9          3.6         1.3
#> 63           5.6         2.8          4.9         2.0
#> 64           5.6         2.7          4.2         1.3
```

```
#> 65        5.6      2.5      3.9      1.1
#> 66        5.7      4.4      1.5      0.4
#> 67        5.7      3.8      1.7      0.3
#> 68        5.7      3.0      4.2      1.2
#> 69        5.7      2.9      4.2      1.3
#> 70        5.7      2.8      4.5      1.3
#> 71        5.7      2.8      4.1      1.3
#> 72        5.7      2.6      3.5      1.0
#> 73        5.7      2.5      5.0      2.0
#> 74        5.8      4.0      1.2      0.2
#> 75        5.8      2.8      5.1      2.4
#> 76        5.8      2.7      4.1      1.0
#> 77        5.8      2.7      3.9      1.2
#> 78        5.8      2.7      5.1      1.9
#> 79        5.8      2.7      5.1      1.9
#> 80        5.8      2.6      4.0      1.2
#> 81        5.9      3.2      4.8      1.8
#> 82        5.9      3.0      4.2      1.5
#> 83        5.9      3.0      5.1      1.8
#> 84        6.0      3.4      4.5      1.6
#> 85        6.0      3.0      4.8      1.8
#> 86        6.0      2.9      4.5      1.5
#> 87        6.0      2.7      5.1      1.6
#> 88        6.0      2.2      4.0      1.0
#> 89        6.0      2.2      5.0      1.5
#> 90        6.1      3.0      4.6      1.4
#> 91        6.1      3.0      4.9      1.8
#> 92        6.1      2.9      4.7      1.4
#> 93        6.1      2.8      4.0      1.3
#> 94        6.1      2.8      4.7      1.2
#> 95        6.1      2.6      5.6      1.4
#> 96        6.2      3.4      5.4      2.3
#> 97        6.2      2.9      4.3      1.3
#> 98        6.2      2.8      4.8      1.8
#> 99        6.2      2.2      4.5      1.5
#> 100       6.3      3.4      5.6      2.4
#> 101       6.3      3.3      4.7      1.6
#> 102       6.3      3.3      6.0      2.5
#> 103       6.3      2.9      5.6      1.8
#> 104       6.3      2.8      5.1      1.5
#> 105       6.3      2.7      4.9      1.8
#> 106       6.3      2.5      4.9      1.5
#> 107       6.3      2.5      5.0      1.9
#> 108       6.3      2.3      4.4      1.3
#> 109       6.4      3.2      4.5      1.5
#> 110       6.4      3.2      5.3      2.3
#> 111       6.4      3.1      5.5      1.8
#> 112       6.4      2.9      4.3      1.3
#> 113       6.4      2.8      5.6      2.1
#> 114       6.4      2.8      5.6      2.2
#> 115       6.4      2.7      5.3      1.9
#> 116       6.5      3.2      5.1      2.0
#> 117       6.5      3.0      5.8      2.2
#> 118       6.5      3.0      5.5      1.8
#> 119       6.5      3.0      5.2      2.0
#> 120       6.5      2.8      4.6      1.5
#> 121       6.6      3.0      4.4      1.4
#> 122       6.6      2.9      4.6      1.3
#> 123       6.7      3.3      5.7      2.1
#> 124       6.7      3.3      5.7      2.5
#> 125       6.7      3.1      4.4      1.4
#> 126       6.7      3.1      4.7      1.5
#> 127       6.7      3.1      5.6      2.4
#> 128       6.7      3.0      5.0      1.7
#> 129       6.7      3.0      5.2      2.3
#> 130       6.7      2.5      5.8      1.8
#> 131       6.8      3.2      5.9      2.3
```

```
#> 132          6.8         3.0          5.5         2.1
#> 133          6.8         2.8          4.8         1.4
#> 134          6.9         3.2          5.7         2.3
#> 135          6.9         3.1          4.9         1.5
#> 136          6.9         3.1          5.4         2.1
#> 137          6.9         3.1          5.1         2.3
#> 138          7.0         3.2          4.7         1.4
#> 139          7.1         3.0          5.9         2.1
#> 140          7.2         3.6          6.1         2.5
#> 141          7.2         3.2          6.0         1.8
#> 142          7.2         3.0          5.8         1.6
#> 143          7.3         2.9          6.3         1.8
#> 144          7.4         2.8          6.1         1.9
#> 145          7.6         3.0          6.6         2.1
#> 146          7.7         3.8          6.7         2.2
#> 147          7.7         3.0          6.1         2.3
#> 148          7.7         2.8          6.7         2.0
#> 149          7.7         2.6          6.9         2.3
#> 150          7.9         3.8          6.4         2.0
#>         Species
#> 1        setosa
#> 2        setosa
#> 3        setosa
#> 4        setosa
#> 5        setosa
#> 6        setosa
#> 7        setosa
#> 8        setosa
#> 9        setosa
#> 10       setosa
#> 11       setosa
#> 12       setosa
#> 13       setosa
#> 14       setosa
#> 15       setosa
#> 16       setosa
#> 17       setosa
#> 18       setosa
#> 19       setosa
#> 20       setosa
#> 21    virginica
#> 22   versicolor
#> 23       setosa
#> 24       setosa
#> 25       setosa
#> 26       setosa
#> 27       setosa
#> 28       setosa
#> 29       setosa
#> 30       setosa
#> 31   versicolor
#> 32   versicolor
#> 33       setosa
#> 34       setosa
#> 35       setosa
#> 36       setosa
#> 37       setosa
#> 38       setosa
#> 39       setosa
#> 40       setosa
#> 41   versicolor
#> 42       setosa
#> 43       setosa
#> 44       setosa
#> 45   versicolor
#> 46       setosa
#> 47       setosa
```

```
#> 48      setosa
#> 49      setosa
#> 50      setosa
#> 51      setosa
#> 52  versicolor
#> 53      setosa
#> 54      setosa
#> 55  versicolor
#> 56  versicolor
#> 57  versicolor
#> 58  versicolor
#> 59  versicolor
#> 60  versicolor
#> 61  versicolor
#> 62  versicolor
#> 63   virginica
#> 64  versicolor
#> 65  versicolor
#> 66      setosa
#> 67      setosa
#> 68  versicolor
#> 69  versicolor
#> 70  versicolor
#> 71  versicolor
#> 72  versicolor
#> 73   virginica
#> 74      setosa
#> 75   virginica
#> 76  versicolor
#> 77  versicolor
#> 78   virginica
#> 79   virginica
#> 80  versicolor
#> 81  versicolor
#> 82  versicolor
#> 83   virginica
#> 84  versicolor
#> 85   virginica
#> 86  versicolor
#> 87  versicolor
#> 88  versicolor
#> 89   virginica
#> 90  versicolor
#> 91   virginica
#> 92  versicolor
#> 93  versicolor
#> 94  versicolor
#> 95   virginica
#> 96   virginica
#> 97  versicolor
#> 98   virginica
#> 99  versicolor
#> 100  virginica
#> 101 versicolor
#> 102  virginica
#> 103  virginica
#> 104  virginica
#> 105  virginica
#> 106 versicolor
#> 107  virginica
#> 108 versicolor
#> 109 versicolor
#> 110  virginica
#> 111  virginica
#> 112 versicolor
#> 113  virginica
#> 114  virginica
```

```
#> 115  virginica
#> 116  virginica
#> 117  virginica
#> 118  virginica
#> 119  virginica
#> 120 versicolor
#> 121 versicolor
#> 122 versicolor
#> 123  virginica
#> 124  virginica
#> 125 versicolor
#> 126 versicolor
#> 127  virginica
#> 128 versicolor
#> 129  virginica
#> 130  virginica
#> 131  virginica
#> 132  virginica
#> 133 versicolor
#> 134  virginica
#> 135 versicolor
#> 136  virginica
#> 137  virginica
#> 138 versicolor
#> 139  virginica
#> 140  virginica
#> 141  virginica
#> 142  virginica
#> 143  virginica
#> 144  virginica
#> 145  virginica
#> 146  virginica
#> 147  virginica
#> 148  virginica
#> 149  virginica
#> 150  virginica
```

6.8.7 mutate - rank

```
iris %>% mutate(sl_rank = min_rank(Sepal.Length)) %>% arrange(sl_rank)
#>     Sepal.Length Sepal.Width Petal.Length Petal.Width
#> 1            4.3         3.0          1.1         0.1
#> 2            4.4         2.9          1.4         0.2
#> 3            4.4         3.0          1.3         0.2
#> 4            4.4         3.2          1.3         0.2
#> 5            4.5         2.3          1.3         0.3
#> 6            4.6         3.1          1.5         0.2
#> 7            4.6         3.4          1.4         0.3
#> 8            4.6         3.6          1.0         0.2
#> 9            4.6         3.2          1.4         0.2
#> 10           4.7         3.2          1.3         0.2
#> 11           4.7         3.2          1.6         0.2
#> 12           4.8         3.4          1.6         0.2
#> 13           4.8         3.0          1.4         0.1
#> 14           4.8         3.4          1.9         0.2
#> 15           4.8         3.1          1.6         0.2
#> 16           4.8         3.0          1.4         0.3
#> 17           4.9         3.0          1.4         0.2
#> 18           4.9         3.1          1.5         0.1
#> 19           4.9         3.1          1.5         0.2
#> 20           4.9         3.6          1.4         0.1
#> 21           4.9         2.4          3.3         1.0
#> 22           4.9         2.5          4.5         1.7
#> 23           5.0         3.6          1.4         0.2
```

```
#> 24          5.0         3.4          1.5         0.2
#> 25          5.0         3.0          1.6         0.2
#> 26          5.0         3.4          1.6         0.4
#> 27          5.0         3.2          1.2         0.2
#> 28          5.0         3.5          1.3         0.3
#> 29          5.0         3.5          1.6         0.6
#> 30          5.0         3.3          1.4         0.2
#> 31          5.0         2.0          3.5         1.0
#> 32          5.0         2.3          3.3         1.0
#> 33          5.1         3.5          1.4         0.2
#> 34          5.1         3.5          1.4         0.3
#> 35          5.1         3.8          1.5         0.3
#> 36          5.1         3.7          1.5         0.4
#> 37          5.1         3.3          1.7         0.5
#> 38          5.1         3.4          1.5         0.2
#> 39          5.1         3.8          1.9         0.4
#> 40          5.1         3.8          1.6         0.2
#> 41          5.1         2.5          3.0         1.1
#> 42          5.2         3.5          1.5         0.2
#> 43          5.2         3.4          1.4         0.2
#> 44          5.2         4.1          1.5         0.1
#> 45          5.2         2.7          3.9         1.4
#> 46          5.3         3.7          1.5         0.2
#> 47          5.4         3.9          1.7         0.4
#> 48          5.4         3.7          1.5         0.2
#> 49          5.4         3.9          1.3         0.4
#> 50          5.4         3.4          1.7         0.2
#> 51          5.4         3.4          1.5         0.4
#> 52          5.4         3.0          4.5         1.5
#> 53          5.5         4.2          1.4         0.2
#> 54          5.5         3.5          1.3         0.2
#> 55          5.5         2.3          4.0         1.3
#> 56          5.5         2.4          3.8         1.1
#> 57          5.5         2.4          3.7         1.0
#> 58          5.5         2.5          4.0         1.3
#> 59          5.5         2.6          4.4         1.2
#> 60          5.6         2.9          3.6         1.3
#> 61          5.6         3.0          4.5         1.5
#> 62          5.6         2.5          3.9         1.1
#> 63          5.6         3.0          4.1         1.3
#> 64          5.6         2.7          4.2         1.3
#> 65          5.6         2.8          4.9         2.0
#> 66          5.7         4.4          1.5         0.4
#> 67          5.7         3.8          1.7         0.3
#> 68          5.7         2.8          4.5         1.3
#> 69          5.7         2.6          3.5         1.0
#> 70          5.7         3.0          4.2         1.2
#> 71          5.7         2.9          4.2         1.3
#> 72          5.7         2.8          4.1         1.3
#> 73          5.7         2.5          5.0         2.0
#> 74          5.8         4.0          1.2         0.2
#> 75          5.8         2.7          4.1         1.0
#> 76          5.8         2.7          3.9         1.2
#> 77          5.8         2.6          4.0         1.2
#> 78          5.8         2.7          5.1         1.9
#> 79          5.8         2.8          5.1         2.4
#> 80          5.8         2.7          5.1         1.9
#> 81          5.9         3.0          4.2         1.5
#> 82          5.9         3.2          4.8         1.8
#> 83          5.9         3.0          5.1         1.8
#> 84          6.0         2.2          4.0         1.0
#> 85          6.0         2.9          4.5         1.5
#> 86          6.0         2.7          5.1         1.6
#> 87          6.0         3.4          4.5         1.6
#> 88          6.0         2.2          5.0         1.5
#> 89          6.0         3.0          4.8         1.8
#> 90          6.1         2.9          4.7         1.4
```

```
#> 91          6.1         2.8          4.0         1.3
#> 92          6.1         2.8          4.7         1.2
#> 93          6.1         3.0          4.6         1.4
#> 94          6.1         3.0          4.9         1.8
#> 95          6.1         2.6          5.6         1.4
#> 96          6.2         2.2          4.5         1.5
#> 97          6.2         2.9          4.3         1.3
#> 98          6.2         2.8          4.8         1.8
#> 99          6.2         3.4          5.4         2.3
#> 100         6.3         3.3          4.7         1.6
#> 101         6.3         2.5          4.9         1.5
#> 102         6.3         2.3          4.4         1.3
#> 103         6.3         3.3          6.0         2.5
#> 104         6.3         2.9          5.6         1.8
#> 105         6.3         2.7          4.9         1.8
#> 106         6.3         2.8          5.1         1.5
#> 107         6.3         3.4          5.6         2.4
#> 108         6.3         2.5          5.0         1.9
#> 109         6.4         3.2          4.5         1.5
#> 110         6.4         2.9          4.3         1.3
#> 111         6.4         2.7          5.3         1.9
#> 112         6.4         3.2          5.3         2.3
#> 113         6.4         2.8          5.6         2.1
#> 114         6.4         2.8          5.6         2.2
#> 115         6.4         3.1          5.5         1.8
#> 116         6.5         2.8          4.6         1.5
#> 117         6.5         3.0          5.8         2.2
#> 118         6.5         3.2          5.1         2.0
#> 119         6.5         3.0          5.5         1.8
#> 120         6.5         3.0          5.2         2.0
#> 121         6.6         2.9          4.6         1.3
#> 122         6.6         3.0          4.4         1.4
#> 123         6.7         3.1          4.4         1.4
#> 124         6.7         3.0          5.0         1.7
#> 125         6.7         3.1          4.7         1.5
#> 126         6.7         2.5          5.8         1.8
#> 127         6.7         3.3          5.7         2.1
#> 128         6.7         3.1          5.6         2.4
#> 129         6.7         3.3          5.7         2.5
#> 130         6.7         3.0          5.2         2.3
#> 131         6.8         2.8          4.8         1.4
#> 132         6.8         3.0          5.5         2.1
#> 133         6.8         3.2          5.9         2.3
#> 134         6.9         3.1          4.9         1.5
#> 135         6.9         3.2          5.7         2.3
#> 136         6.9         3.1          5.4         2.1
#> 137         6.9         3.1          5.1         2.3
#> 138         7.0         3.2          4.7         1.4
#> 139         7.1         3.0          5.9         2.1
#> 140         7.2         3.6          6.1         2.5
#> 141         7.2         3.2          6.0         1.8
#> 142         7.2         3.0          5.8         1.6
#> 143         7.3         2.9          6.3         1.8
#> 144         7.4         2.8          6.1         1.9
#> 145         7.6         3.0          6.6         2.1
#> 146         7.7         3.8          6.7         2.2
#> 147         7.7         2.6          6.9         2.3
#> 148         7.7         2.8          6.7         2.0
#> 149         7.7         3.0          6.1         2.3
#> 150         7.9         3.8          6.4         2.0
#>        Species sl_rank
#> 1       setosa       1
#> 2       setosa       2
#> 3       setosa       2
#> 4       setosa       2
#> 5       setosa       5
#> 6       setosa       6
```

```
#> 7       setosa       6
#> 8       setosa       6
#> 9       setosa       6
#> 10      setosa      10
#> 11      setosa      10
#> 12      setosa      12
#> 13      setosa      12
#> 14      setosa      12
#> 15      setosa      12
#> 16      setosa      12
#> 17      setosa      17
#> 18      setosa      17
#> 19      setosa      17
#> 20      setosa      17
#> 21  versicolor      17
#> 22   virginica      17
#> 23      setosa      23
#> 24      setosa      23
#> 25      setosa      23
#> 26      setosa      23
#> 27      setosa      23
#> 28      setosa      23
#> 29      setosa      23
#> 30      setosa      23
#> 31  versicolor      23
#> 32  versicolor      23
#> 33      setosa      33
#> 34      setosa      33
#> 35      setosa      33
#> 36      setosa      33
#> 37      setosa      33
#> 38      setosa      33
#> 39      setosa      33
#> 40      setosa      33
#> 41  versicolor      33
#> 42      setosa      42
#> 43      setosa      42
#> 44      setosa      42
#> 45  versicolor      42
#> 46      setosa      46
#> 47      setosa      47
#> 48      setosa      47
#> 49      setosa      47
#> 50      setosa      47
#> 51      setosa      47
#> 52  versicolor      47
#> 53      setosa      53
#> 54      setosa      53
#> 55  versicolor      53
#> 56  versicolor      53
#> 57  versicolor      53
#> 58  versicolor      53
#> 59  versicolor      53
#> 60  versicolor      60
#> 61  versicolor      60
#> 62  versicolor      60
#> 63  versicolor      60
#> 64  versicolor      60
#> 65   virginica      60
#> 66      setosa      66
#> 67      setosa      66
#> 68  versicolor      66
#> 69  versicolor      66
#> 70  versicolor      66
#> 71  versicolor      66
#> 72  versicolor      66
#> 73   virginica      66
```

```
#> 74     setosa      74
#> 75 versicolor      74
#> 76 versicolor      74
#> 77 versicolor      74
#> 78  virginica      74
#> 79  virginica      74
#> 80  virginica      74
#> 81 versicolor      81
#> 82 versicolor      81
#> 83  virginica      81
#> 84 versicolor      84
#> 85 versicolor      84
#> 86 versicolor      84
#> 87 versicolor      84
#> 88  virginica      84
#> 89  virginica      84
#> 90 versicolor      90
#> 91 versicolor      90
#> 92 versicolor      90
#> 93 versicolor      90
#> 94  virginica      90
#> 95  virginica      90
#> 96 versicolor      96
#> 97 versicolor      96
#> 98  virginica      96
#> 99  virginica      96
#> 100 versicolor    100
#> 101 versicolor    100
#> 102 versicolor    100
#> 103  virginica    100
#> 104  virginica    100
#> 105  virginica    100
#> 106  virginica    100
#> 107  virginica    100
#> 108  virginica    100
#> 109 versicolor    109
#> 110 versicolor    109
#> 111  virginica    109
#> 112  virginica    109
#> 113  virginica    109
#> 114  virginica    109
#> 115  virginica    109
#> 116 versicolor    116
#> 117  virginica    116
#> 118  virginica    116
#> 119  virginica    116
#> 120  virginica    116
#> 121 versicolor    121
#> 122 versicolor    121
#> 123 versicolor    123
#> 124 versicolor    123
#> 125 versicolor    123
#> 126  virginica    123
#> 127  virginica    123
#> 128  virginica    123
#> 129  virginica    123
#> 130  virginica    123
#> 131 versicolor    131
#> 132  virginica    131
#> 133  virginica    131
#> 134 versicolor    134
#> 135  virginica    134
#> 136  virginica    134
#> 137  virginica    134
#> 138 versicolor    138
#> 139  virginica    139
#> 140  virginica    140
```

```
#> 141  virginica     140
#> 142  virginica     140
#> 143  virginica     143
#> 144  virginica     144
#> 145  virginica     145
#> 146  virginica     146
#> 147  virginica     146
#> 148  virginica     146
#> 149  virginica     146
#> 150  virginica     150
```

6.8.8 group_by and summarize

```
iris %>%
  group_by(Species) %>%
  summarize(sl = mean(Sepal.Length), sw = mean(Sepal.Width),
  pl = mean(Petal.Length), pw = mean(Petal.Width))
#> # A tibble: 3 x 5
#>   Species       sl    sw    pl    pw
#>   <fct>      <dbl> <dbl> <dbl> <dbl>
#> 1 setosa      5.01  3.43  1.46 0.246
#> 2 versicolor  5.94  2.77  4.26 1.33
#> 3 virginica   6.59  2.97  5.55 2.03
```

- mean: `mean()` or `mean(x, na.rm = TRUE)` - arithmetic mean (average)
- median: `median()` or `median(x, na.rm = TRUE)` - mid value

For more examples see

dplr_iris

6.9 References of `dplyr`

- Textbook: R for Data Science, Part II Explore

6.9.1 RStudio Primers: See References in Moodle at the bottom

1. The Basics – r4ds: Explore, I

- Visualization Basics
- Programming Basics

2. **Work with Data** – r4ds: Wrangle, I

- **Working with Tibbles**
- **Isolating Data with dplyr**
- **Deriving Information with dplyr**

3. Visualize Data – r4ds: Explore, II
4. Tidy Your Data – r4ds: Wrangle, II
5. Iterate – r4ds: Program
6. Write Functions – r4ds: Program

6.10 Learn dplyr by Examples II - gapminder

6.10.1 ggplot2 Overview

ggplot2 is a system for declaratively creating graphics, based on The Grammar of Graphics. You provide the data, tell ggplot2 how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details.

Examples

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy))

ggplot(data = mpg) +
  geom_boxplot(mapping = aes(x = class, y = hwy))
```

Template

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

6.10.1.1 Gapminder and R Package gapminder

Gapminder was founded by Ola Rosling, Anna Rosling Rönnlund, and Hans Rosling

- Gapminder: https://www.gapminder.org
  - Test on Top: You are probably wrong about - upgrade your worldview
  - Bubble Chart: https://www.gapminder.org/tools/#$chart-type=bubbles&url=v1
  - Dallar Street: https://www.gapminder.org/tools/#$chart-type=bubbles&url=v1
  - Data: https://www.gapminder.org/data/
- R Package gapminder by Jennifer Bryan
  - Package site: https://CRAN.R-project.org/package=gapminder
  - Site: https://github.com/jennybc/gapminder
  - Documents: https://www.rdocumentation.org/packages/gapminder/versions/0.3.0
- Package Help `?gapminder` or `gapminder` in the search window of Help
  - The main data frame gapminder has 1704 rows and 6 variables:
    * country: factor with 142 levels
    * continent: factor with 5 levels
    * year: ranges from 1952 to 2007 in increments of 5 years
    * lifeExp: life expectancy at birth, in years
    * pop: population
    * gdpPercap: GDP per capita (US$, inflation-adjusted)

```
library(tidyverse)
library(gapminder)
library(WDI)
```

6.10.1.2 R Package gapminder data

```
df <- gapminder
df %>% slice(1:10)
#> # A tibble: 10 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
```

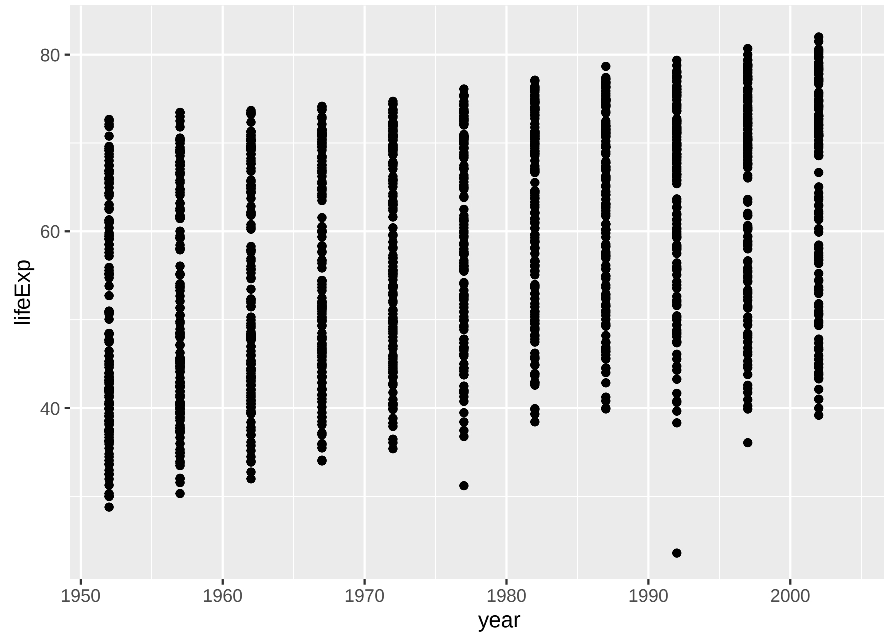
```
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
```

```
glimpse(df)
#> Rows: 1,704
#> Columns: 6
#> $ country   <fct> "Afghanistan", "Afghanistan", "Afghanist~
#> $ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia~
#> $ year      <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982~
#> $ lifeExp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, ~
#> $ pop       <int> 8425333, 9240934, 10267083, 11537966, 13~
#> $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, ~
```

```
summary(df)
#>         country        continent        year     
#>  Afghanistan:  12   Africa  :624   Min.   :1952  
#>  Albania    :  12   Americas:300   1st Qu.:1966  
#>  Algeria    :  12   Asia    :396   Median :1980  
#>  Angola     :  12   Europe  :360   Mean   :1980  
#>  Argentina  :  12   Oceania : 24   3rd Qu.:1993  
#>  Australia  :  12                  Max.   :2007  
#>  (Other)    :1632                                
#>     lifeExp           pop              gdpPercap       
#>  Min.   :23.60   Min.   :6.001e+04   Min.   :   241.2  
#>  1st Qu.:48.20   1st Qu.:2.794e+06   1st Qu.:  1202.1  
#>  Median :60.71   Median :7.024e+06   Median :  3531.8  
#>  Mean   :59.47   Mean   :2.960e+07   Mean   :  7215.3  
#>  3rd Qu.:70.85   3rd Qu.:1.959e+07   3rd Qu.:  9325.5  
#>  Max.   :82.60   Max.   :1.319e+09   Max.   :113523.1  
#> 
```

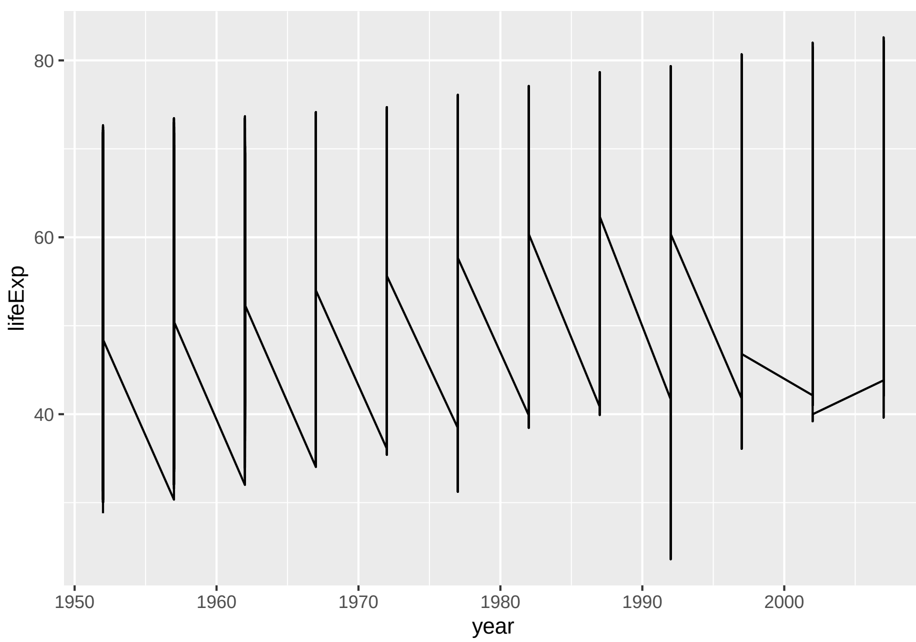
```
ggplot(df, aes(x = year, y = lifeExp)) + geom_point()
```

6.10.1.3 Tidyverse::ggplot



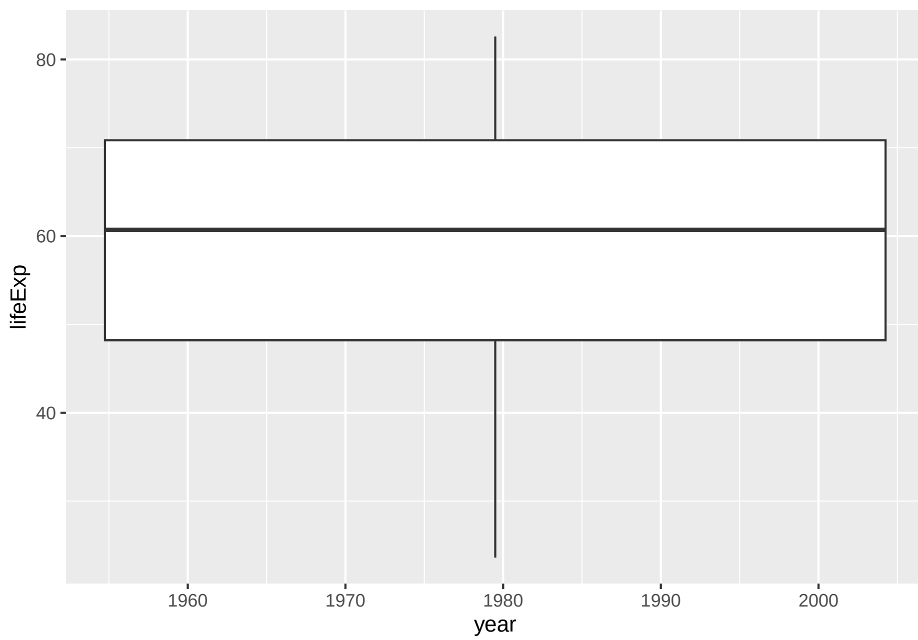
6.10.1.3.1 First Try - with failures

```
ggplot(df, aes(x = year, y = lifeExp)) + geom_line()
```



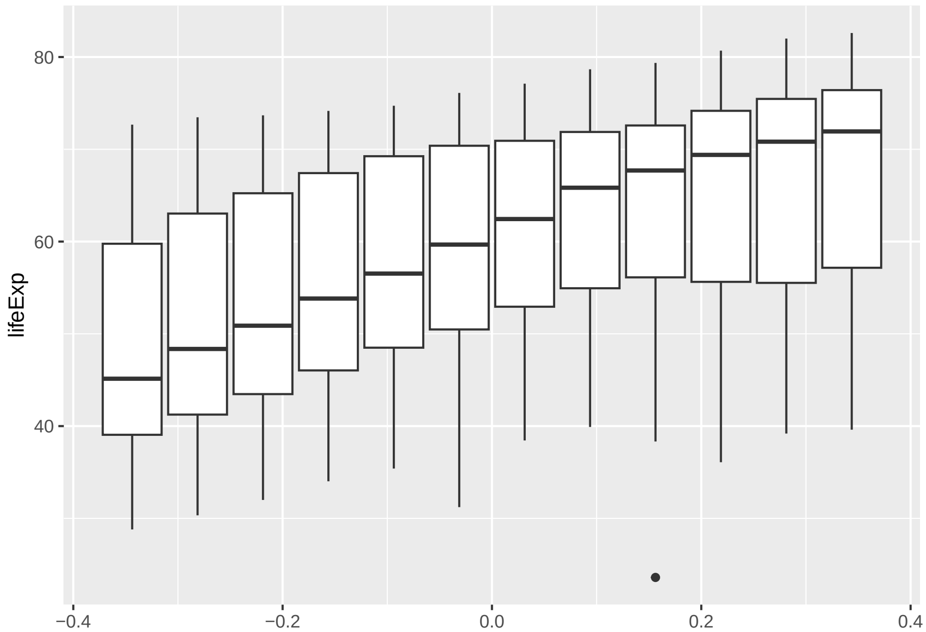

```
ggplot(df, aes(x = year, y = lifeExp)) + geom_boxplot()
#> Warning: Continuous x aesthetic
```

```
#> i did you forget `aes(group = ...)`?
```

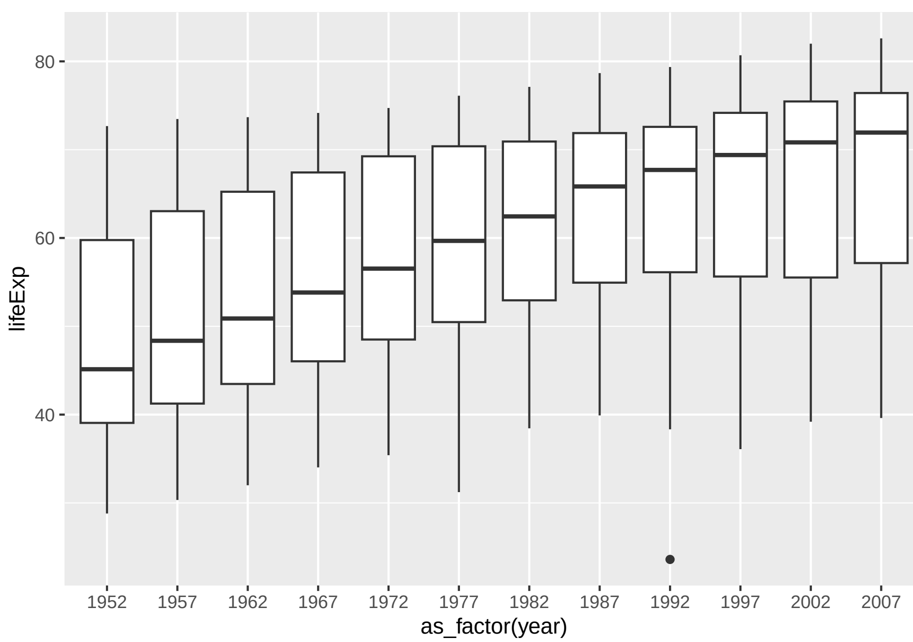


```
typeof(pull(df, year)) # same as typeof(df$year)
#> [1] "integer"
```

```
ggplot(df, aes(y = lifeExp, group = year)) + geom_boxplot()
```



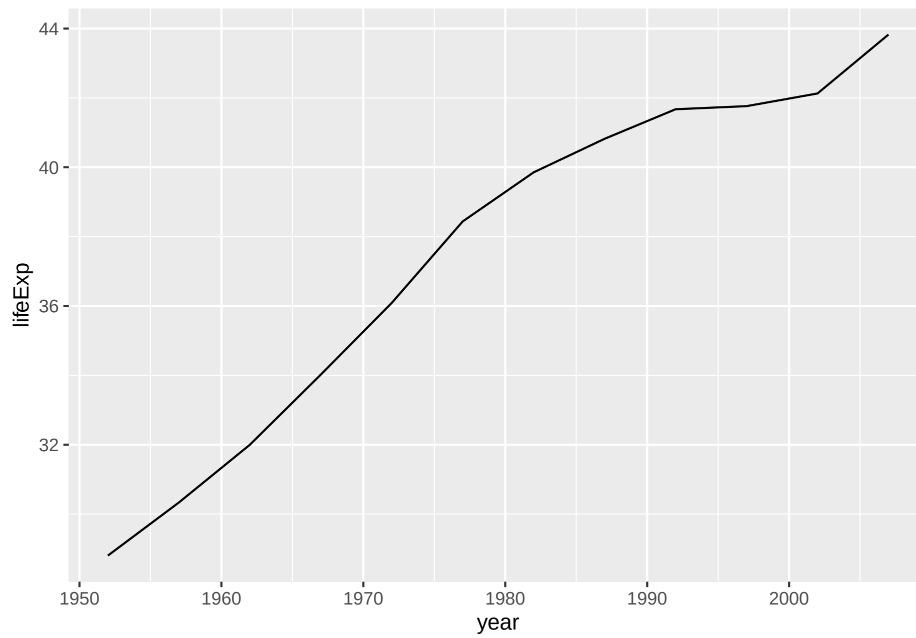

```
ggplot(df, aes(x = as_factor(year), y = lifeExp)) + geom_boxplot()
```



6.10.1.3.2 Box Plot

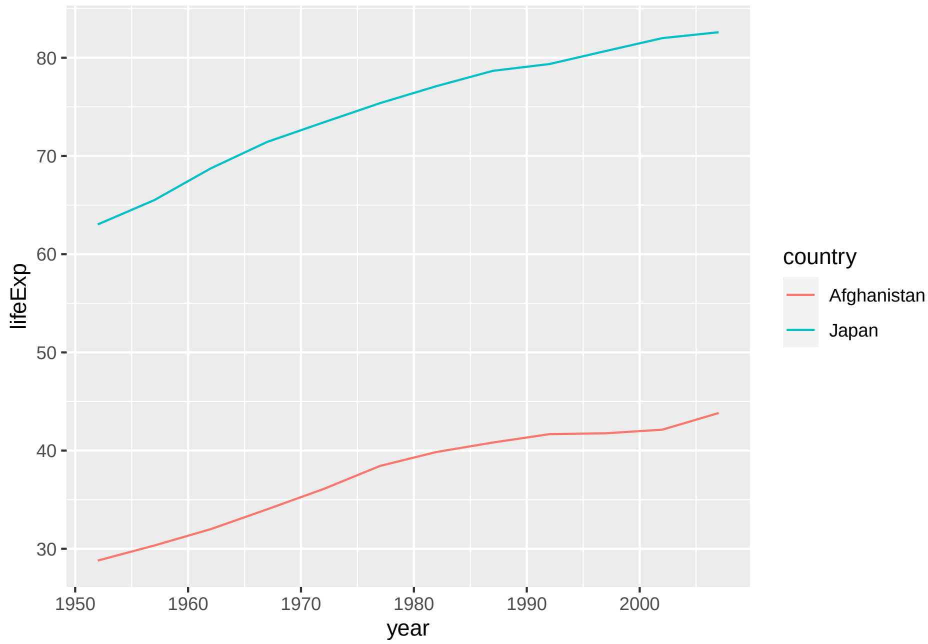
```
df %>% filter(country == "Afghanistan") %>%
  ggplot(aes(x = year, y = lifeExp)) + geom_line()
```


6.10.1.4 Applications of dplyr



6.10.1.4.1 filter

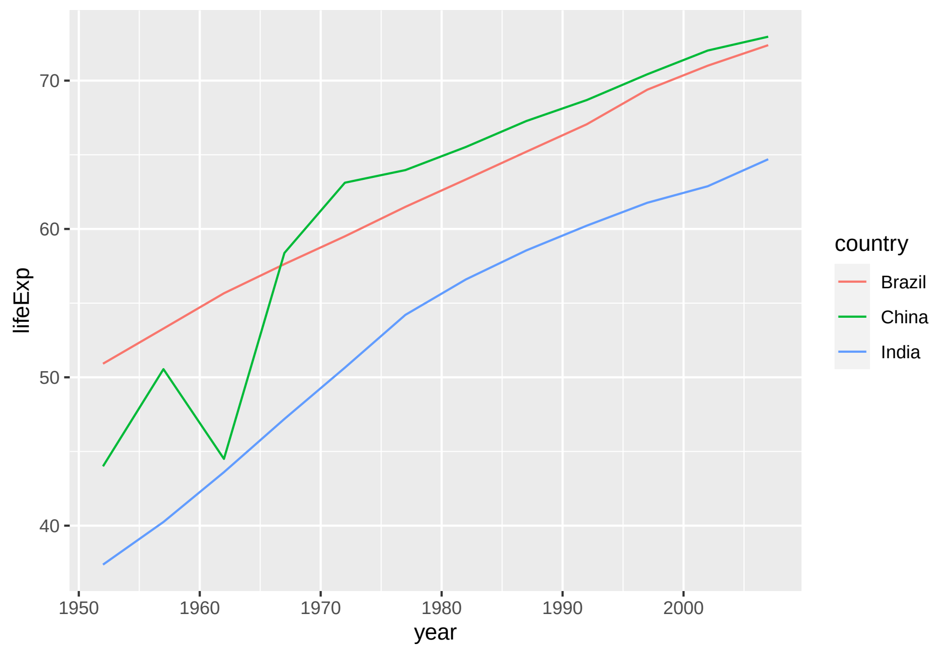
```
df %>% filter(country %in% c("Afghanistan", "Japan")) %>%
  ggplot(aes(x = year, y = lifeExp, color = country)) + geom_line()
```




```
df %>% distinct(country) %>% pull()
#>   [1] Afghanistan              Albania
#>   [3] Algeria                  Angola
#>   [5] Argentina                Australia
#>   [7] Austria                  Bahrain
#>   [9] Bangladesh               Belgium
#>  [11] Benin                    Bolivia
#>  [13] Bosnia and Herzegovina   Botswana
#>  [15] Brazil                   Bulgaria
#>  [17] Burkina Faso             Burundi
#>  [19] Cambodia                 Cameroon
#>  [21] Canada                   Central African Republic
#>  [23] Chad                     Chile
#>  [25] China                    Colombia
#>  [27] Comoros                  Congo, Dem. Rep.
#>  [29] Congo, Rep.              Costa Rica
#>  [31] Cote d'Ivoire            Croatia
#>  [33] Cuba                     Czech Republic
#>  [35] Denmark                  Djibouti
#>  [37] Dominican Republic       Ecuador
#>  [39] Egypt                    El Salvador
#>  [41] Equatorial Guinea        Eritrea
#>  [43] Ethiopia                 Finland
#>  [45] France                   Gabon
#>  [47] Gambia                   Germany
#>  [49] Ghana                    Greece
#>  [51] Guatemala                Guinea
#>  [53] Guinea-Bissau            Haiti
#>  [55] Honduras                 Hong Kong, China
#>  [57] Hungary                  Iceland
#>  [59] India                    Indonesia
#>  [61] Iran                     Iraq
#>  [63] Ireland                  Israel
#>  [65] Italy                    Jamaica
#>  [67] Japan                    Jordan
#>  [69] Kenya                    Korea, Dem. Rep.
```

```
#>  [71] Korea, Rep.              Kuwait
#>  [73] Lebanon                  Lesotho
#>  [75] Liberia                  Libya
#>  [77] Madagascar               Malawi
#>  [79] Malaysia                 Mali
#>  [81] Mauritania               Mauritius
#>  [83] Mexico                   Mongolia
#>  [85] Montenegro               Morocco
#>  [87] Mozambique               Myanmar
#>  [89] Namibia                  Nepal
#>  [91] Netherlands              New Zealand
#>  [93] Nicaragua                Niger
#>  [95] Nigeria                  Norway
#>  [97] Oman                     Pakistan
#>  [99] Panama                   Paraguay
#> [101] Peru                     Philippines
#> [103] Poland                   Portugal
#> [105] Puerto Rico              Reunion
#> [107] Romania                  Rwanda
#> [109] Sao Tome and Principe    Saudi Arabia
#> [111] Senegal                  Serbia
#> [113] Sierra Leone             Singapore
#> [115] Slovak Republic          Slovenia
#> [117] Somalia                  South Africa
#> [119] Spain                    Sri Lanka
#> [121] Sudan                    Swaziland
#> [123] Sweden                   Switzerland
#> [125] Syria                    Taiwan
#> [127] Tanzania                 Thailand
#> [129] Togo                     Trinidad and Tobago
#> [131] Tunisia                  Turkey
#> [133] Uganda                   United Kingdom
#> [135] United States            Uruguay
#> [137] Venezuela                Vietnam
#> [139] West Bank and Gaza       Yemen, Rep.
#> [141] Zambia                   Zimbabwe
#> 142 Levels: Afghanistan Albania Algeria Angola ... Zimbabwe
```

```
df %>% filter(country %in% c("Brazil", "Russia", "India", "China")) %>%
  ggplot(aes(x = year, y = lifeExp, color = country)) + geom_line()
```



Russian data is missing.

6.10.2 Exercises

1. Change `lifeExp` to `pop` and `gdpPercap` and do the same.
2. Choose ASEAN countries and do the similar investigations.

- Brunei, Cambodia, Indonesia, Laos, Malaysia, Myanmar, Philippines, Singapore.

3. Choose several countries by yourself and do the similar investigations.

6.10.3 `group_by` and `summarize`

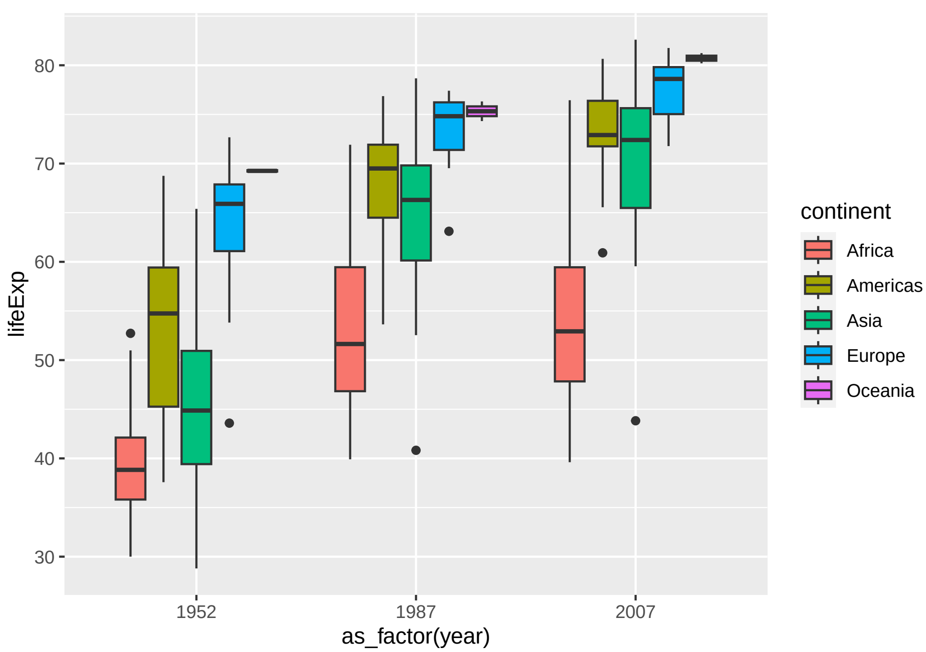
Let us use the variable `continent` and summarize the data.

```
df_lifeExp <- df %>% group_by(continent, year) %>%
  summarize(mean_lifeExp = mean(lifeExp), median_lifeExp = median(lifeExp), max_lifeExp = max(lifeExp), min_lifeExp = min(lifeExp), .grou
```

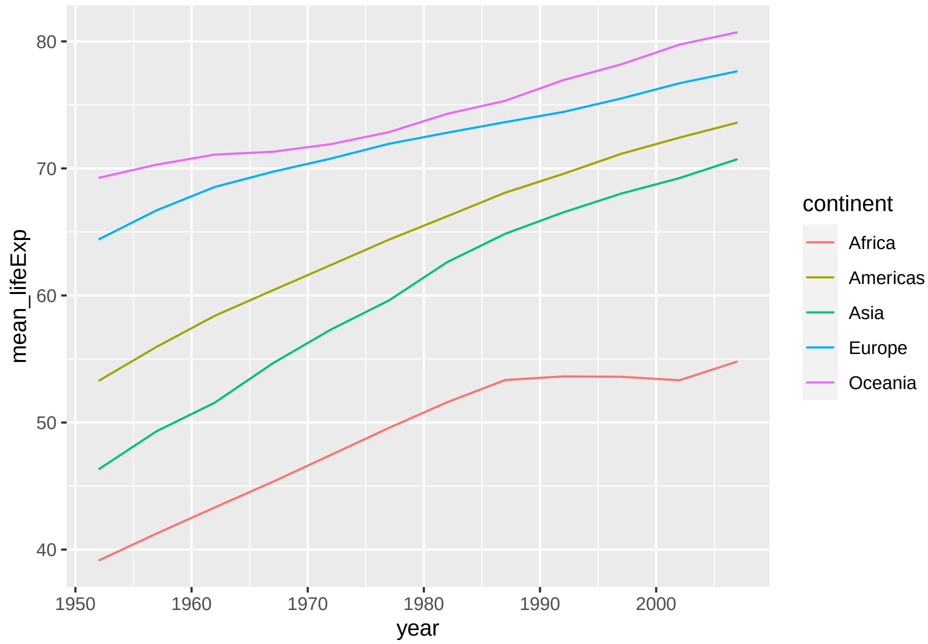
```
df_lifeExp %>% slice(1:10)
#> # A tibble: 60 x 6
#> # Groups:   continent, year [60]
#>    continent  year mean_lifeExp median_lif~1 max_l~2 min_l~3
#>    <fct>     <int>        <dbl>        <dbl>   <dbl>   <dbl>
#>  1 Africa     1952         39.1         38.8    52.7    30
#>  2 Africa     1957         41.3         40.6    58.1    31.6
#>  3 Africa     1962         43.3         42.6    60.2    32.8
#>  4 Africa     1967         45.3         44.7    61.6    34.1
#>  5 Africa     1972         47.5         47.0    64.3    35.4
#>  6 Africa     1977         49.6         49.3    67.1    36.8
#>  7 Africa     1982         51.6         50.8    69.9    38.4
#>  8 Africa     1987         53.3         51.6    71.9    39.9
#>  9 Africa     1992         53.6         52.4    73.6    23.6
```

```
#> 10 Africa     1997         53.6         52.8     74.8    36.1
#> # ... with 50 more rows, and abbreviated variable names
#> #   1: median_lifeExp, 2: max_lifeExp, 3: min_lifeExp
```

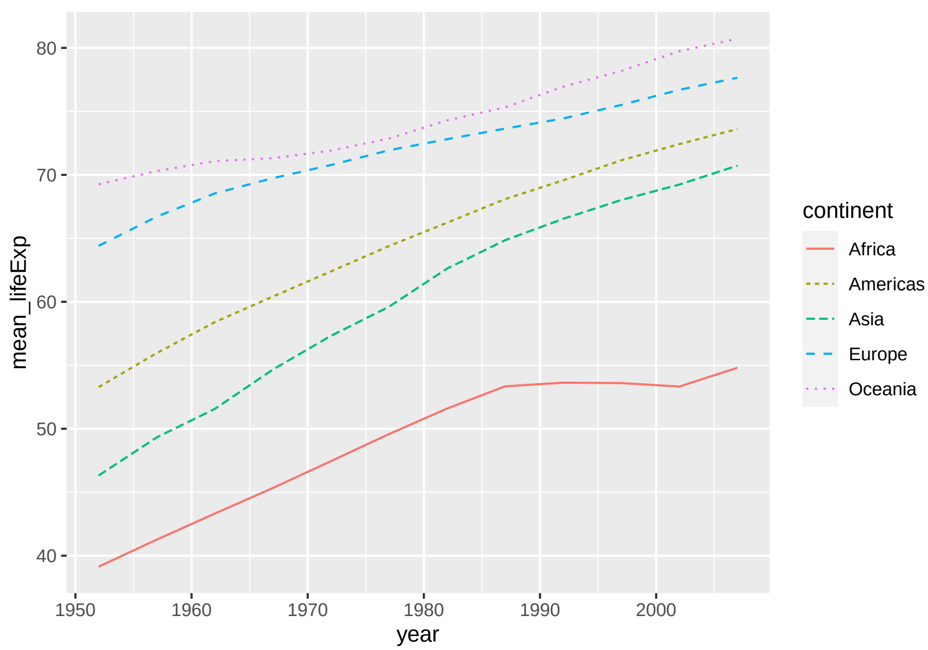
```
df %>% filter(year %in% c(1952, 1987, 2007)) %>%
  ggplot(aes(x=as_factor(year), y = lifeExp, fill = continent)) +
  geom_boxplot()
```



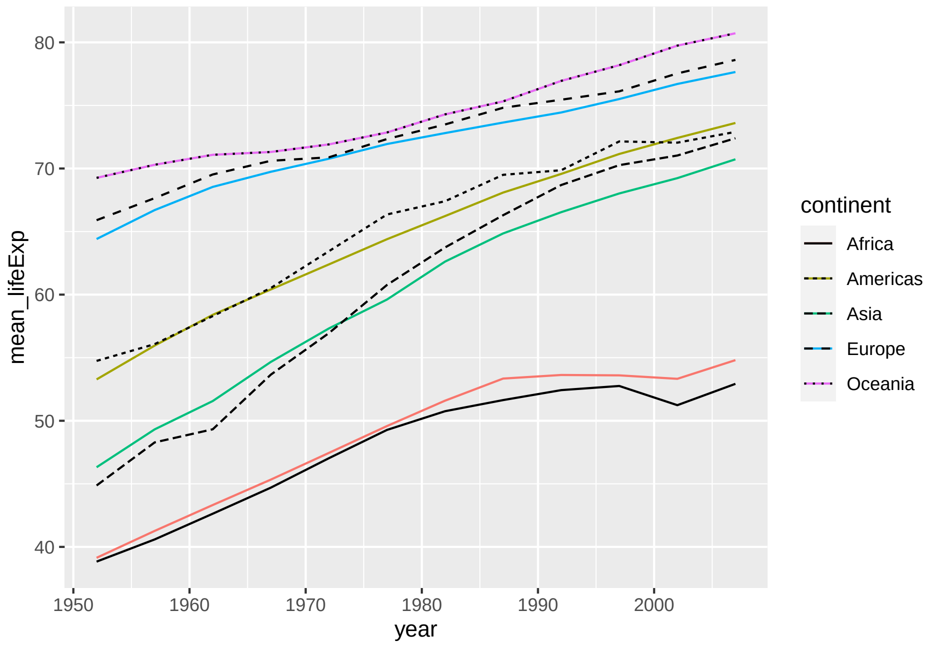
```
df_lifeExp %>% ggplot(aes(x = year, y = mean_lifeExp, color = continent)) +
  geom_line()
```




```
df_lifeExp %>% ggplot(aes(x = year, y = mean_lifeExp, color = continent, linetype = continent)) +
  geom_line()
```



```
df_lifeExp %>% ggplot() +
  geom_line(aes(x = year, y = mean_lifeExp, color = continent)) +
  geom_line(aes(x = year, y = median_lifeExp, linetype = continent))
```



6.11 The Week Two Assignment (in Moodle)

R Markdown and dplyr

- Create an R Notebook of a Data Analysis containing the following and submit the rendered HTML file (eg. `a2_123456.nb.html`)
  1. create an R Notebook using the R Notebook Template in Moodle, save as `a2_123456.Rmd`,
  2. write your name and ID and the contents,
  3. run each code block,
  4. preview to create `a2_123456.nb.html`,
  5. submit `a2_123456.nb.html` to Moodle.

1. Pick data from the built-in datasets besides `cars`. (`library(help = "datasets")` or go to the site The R Datasets Package)

   - Information of the data: Name, Description, Usage, Format, Source, References (Hint: ?cars)
   - Use `head()`, `str()`, …, and create at least one chart using `ggplot2` - Code Chunk.
     - Don't forget to add `library(tidyverse)` in the first code chunk.
   - An observation of the chart - in your own words.

---

2. Load `gapminder` by `library(gapminder)`.

   - Choose `pop` or `gdpPercap`, or both, one country in the data, a group of countries in the data.
   - Create charts using ggplot2 with geom_line and the variables and countries chosen in 1. (See examples of the charts for `lifeExp`.)
   - Study the data as you like.
   - Observations and difficulties encountered.

Due: 2023-01-09 23:59:00. Submit your R Notebook file in Moodle (The Second Assignment). Due on Monday!

6.11.1 Original Data? WDI?

```
gapminder %>% slice(1:10)
#> # A tibble: 10 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
```

6.11.1.1 WDI

- SP.DYN.LE00.IN: Life expectancy at birth, total (years)
- NY.GDP.PCAP.KD: GDP per capita (constant 2015 US$)
- SP.POP.TOTL: Population, total

```
df_wdi <- WDI(
  country = "all",
  indicator = c(lifeExp = "SP.DYN.LE00.IN", pop = "SP.POP.TOTL", gdpPercap = "NY.GDP.PCAP.KD")
)
```

```
#> Rows: 16492 Columns: 7
#> -- Column specification ------------------------------------
#> Delimiter: ","
#> chr (3): country, iso2c, iso3c
#> dbl (4): year, lifeExp, pop, gdpPercap
#>
#> i Use `spec()` to retrieve the full column specification for this data.
#> i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_wdi %>% slice(1:10)
#> # A tibble: 10 x 7
#>    country     iso2c iso3c  year lifeExp      pop gdpPercap
#>    <chr>       <chr> <chr> <dbl>   <dbl>    <dbl>     <dbl>
#>  1 Afghanistan AF    AFG    1960    32.5  8622466        NA
#>  2 Afghanistan AF    AFG    1961    33.1  8790140        NA
#>  3 Afghanistan AF    AFG    1962    33.5  8969047        NA
#>  4 Afghanistan AF    AFG    1963    34.0  9157465        NA
#>  5 Afghanistan AF    AFG    1964    34.5  9355514        NA
#>  6 Afghanistan AF    AFG    1965    35.0  9565147        NA
#>  7 Afghanistan AF    AFG    1966    35.5  9783147        NA
#>  8 Afghanistan AF    AFG    1967    35.9 10010030        NA
#>  9 Afghanistan AF    AFG    1968    36.4 10247780        NA
#> 10 Afghanistan AF    AFG    1969    36.9 10494489        NA
```

```
df_wdi_extra <- WDI(
  country = "all",
  indicator = c(lifeExp = "SP.DYN.LE00.IN", pop = "SP.POP.TOTL", gdpPercap = "NY.GDP.PCAP.KD"),
  extra = TRUE
)
```

```
#> Rows: 16492 Columns: 15
#> -- Column specification ------------------------------------
#> Delimiter: ","
#> chr  (7): country, iso2c, iso3c, region, capital, income...
#> dbl  (6): year, lifeExp, pop, gdpPercap, longitude, lati...
#> lgl  (1): status
#> date (1): lastupdated
#>
#> i Use `spec()` to retrieve the full column specification for this data.
#> i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_wdi_extra
#> # A tibble: 16,492 x 15
#>    country     iso2c iso3c  year status lastupdated lifeExp
#>    <chr>       <chr> <chr> <dbl> <lgl>  <date>        <dbl>
#>  1 Afghanistan AF    AFG    1993 NA     2022-12-22     51.5
#>  2 Afghanistan AF    AFG    1997 NA     2022-12-22     53.6
#>  3 Afghanistan AF    AFG    1994 NA     2022-12-22     51.5
#>  4 Afghanistan AF    AFG    1995 NA     2022-12-22     52.5
#>  5 Afghanistan AF    AFG    2001 NA     2022-12-22     55.8
#>  6 Afghanistan AF    AFG    1998 NA     2022-12-22     52.9
#>  7 Afghanistan AF    AFG    1999 NA     2022-12-22     54.8
#>  8 Afghanistan AF    AFG    2007 NA     2022-12-22     59.1
#>  9 Afghanistan AF    AFG    2008 NA     2022-12-22     59.9
#> 10 Afghanistan AF    AFG    1980 NA     2022-12-22     39.6
#> # ... with 16,482 more rows, and 8 more variables:
#> #   pop <dbl>, gdpPercap <dbl>, region <chr>,
#> #   capital <chr>, longitude <dbl>, latitude <dbl>,
#> #   income <chr>, lending <chr>
```

Chapter 7

Responses to Assignment Two

1. You are supposed to submit an R Notebook File with a file name a2_YourID.nb.html.

- Some submitted an HTML file, such as a2_YourID.html. You need to create an R Notebook. Use the template in Moodle. It creates a file with *.nb.html at the end automatically.
- Some did not run each code chunk. You should run each code or select 'Run all' under 'Run' button. If some code chunk has a problem or an error, run each code chunk or use Run all chunk above or Run all chunk below, so the result appear in your R Notebook file.

2. You are supposed to write observations.

- Writing codes seem to be challenging, however, we are learning 'data analysis' not 'programming'. Do not forget to write explanations of the data, questions and observations.

3. Cheat Sheets, Posit Primers, and the textbook 'R for Data Science' are the first set of references you should look at together wih my lecture materials.

Chapter 8

Set up

```
library(tidyverse)
#> -- Attaching packages ------------------- tidyverse 1.3.2 --
#> v ggplot2 3.4.0      v purrr   1.0.0
#> v tibble  3.1.8      v dplyr   1.0.10
#> v tidyr   1.2.1      v stringr 1.5.0
#> v readr   2.1.3      v forcats 0.5.2
#> -- Conflicts ---------------------- tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
library(gapminder)
```

The following `(df <- gapminder)` is a short-hand of

```
df <- gapminder
df
```

```
(df <- gapminder)
#> # A tibble: 1,704 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
#> # ... with 1,694 more rows
```

Chapter 9

General Comments

9.1 Varibles

We should know first about the variables. At least you must know if each of the variables is a categorical variable or a numerical variable.

For example, in the `gapminder` data, `country`, `continent` are categorical variables, and `year`, `lifeExp`, `pop`, `gdpPercap` are numerical variables. It is possible to treat `year` as a categorical variable.

9.2 Example: `datasets::CO2`

9.2.1 The first step

You can obtain basic information of the data by the following or typing CO2 in the search box under Help tab. You can see the same at: https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html

```
help(CO2) # or ? CO2
```

- **Description**: The CO2 data frame has 84 rows and 5 columns of data from an experiment on the cold tolerance of the grass species Echinochloa crus-galli.
- **Usage**: CO2
- **Format**
  - An object of class c("nfnGroupedData", "nfGroupedData", "groupedData", "data.frame") containing the following columns:
  - Plant: an ordered factor with levels Qn1 < Qn2 < Qn3 < … < Mc1 giving a unique identifier for each plant.
  - Type: a factor with levels Quebec Mississippi giving the origin of the plant
  - Treatment: a factor with levels nonchilled chilled
  - conc: a numeric vector of ambient carbon dioxide concentrations (mL/L).
  - uptake: a numeric vector of carbon dioxide uptake rates (mol/m^2 mol/m 2 sec).
- **Details**: The CO_2 uptake of six plants from Quebec and six plants from Mississippi was measured at several levels of ambient CO_2 concentration. Half the plants of each type were chilled overnight before the experiment was conducted.
  - This dataset was originally part of package nlme, and that has methods (including for [, as.data.frame, plot and print) for its grouped-data classes.
- **Source**: Potvin, C., Lechowicz, M. J. and Tardif, S. (1990) "The statistical analysis of ecophysiological response curves obtained from experiments involving repeated measures", Ecology, 71, 1389–1400.
  - Pinheiro, J. C. and Bates, D. M. (2000) Mixed-effects Models in S and S-PLUS, Springer.

```
df_co2 <- as_tibble(datasets::CO2) # what happens if simply `df_co2 <- datasets::CO2`
df_co2
#> # A tibble: 84 x 5
#>    Plant Type   Treatment   conc uptake
#>    <ord> <fct>  <fct>      <dbl>  <dbl>
#>  1 Qn1   Quebec nonchilled    95   16
#>  2 Qn1   Quebec nonchilled   175   30.4
#>  3 Qn1   Quebec nonchilled   250   34.8
#>  4 Qn1   Quebec nonchilled   350   37.2
#>  5 Qn1   Quebec nonchilled   500   35.3
#>  6 Qn1   Quebec nonchilled   675   39.2
#>  7 Qn1   Quebec nonchilled  1000   39.7
#>  8 Qn2   Quebec nonchilled    95   13.6
#>  9 Qn2   Quebec nonchilled   175   27.3
#> 10 Qn2   Quebec nonchilled   250   37.1
#> # ... with 74 more rows
```

You can use `head(CO2)` if you set `df_co2 <-CO2` or `df_co2 <- datasets::CO2`.

```
glimpse(df_co2)
#> Rows: 84
#> Columns: 5
#> $ Plant     <ord> Qn1, Qn1, Qn1, Qn1, Qn1, Qn1, Qn1, Qn2, ~
#> $ Type      <fct> Quebec, Quebec, Quebec, Quebec, Quebec, ~
#> $ Treatment <fct> nonchilled, nonchilled, nonchilled, nonc~
#> $ conc      <dbl> 95, 175, 250, 350, 500, 675, 1000, 95, 1~
#> $ uptake    <dbl> 16.0, 30.4, 34.8, 37.2, 35.3, 39.2, 39.7~
```

"factor" is a categorical data, and "double" is a numerical data.

```
class(df_co2$Plant)
#> [1] "ordered" "factor"
class(df_co2$Type)
#> [1] "factor"
class(df_co2$Treatment)
#> [1] "factor"
class(df_co2$conc)
#> [1] "numeric"
class(df_co2$uptake)
#> [1] "numeric"
```

```
summary(df_co2)
#>      Plant              Type         Treatment
#>  Qn1    : 7   Quebec     :42   nonchilled:42
#>  Qn2    : 7   Mississippi:42   chilled   :42
#>  Qn3    : 7
#>  Qc1    : 7
#>  Qc3    : 7
#>  Qc2    : 7
#>  (Other):42
#>       conc          uptake
#>  Min.   :  95   Min.   : 7.70
#>  1st Qu.: 175   1st Qu.:17.90
#>  Median : 350   Median :28.30
#>  Mean   : 435   Mean   :27.21
#>  3rd Qu.: 675   3rd Qu.:37.12
#>  Max.   :1000   Max.   :45.50
#>
```

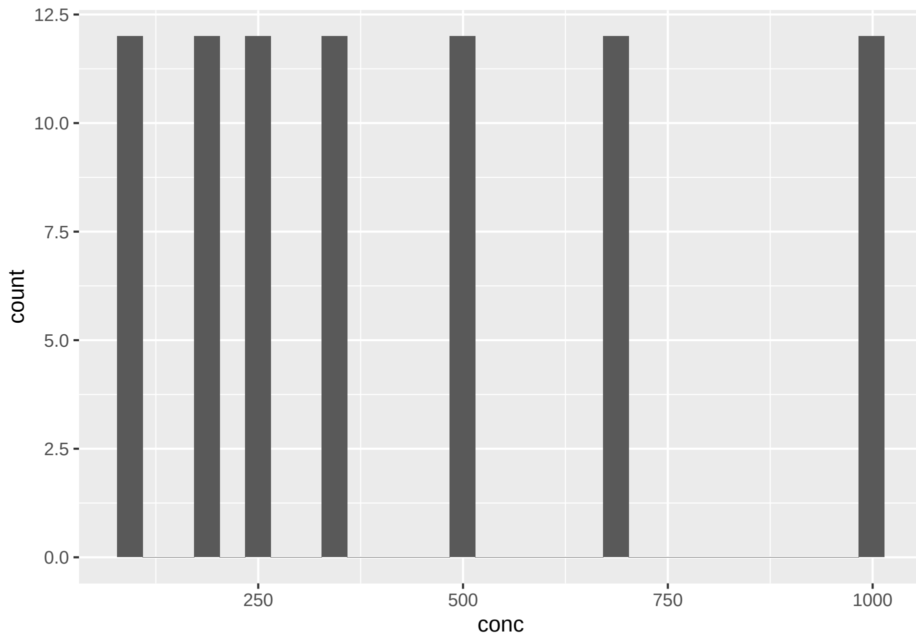
9.2.2 Try as many visualizations as possible

Then you can choose appropriate ones later in your research.

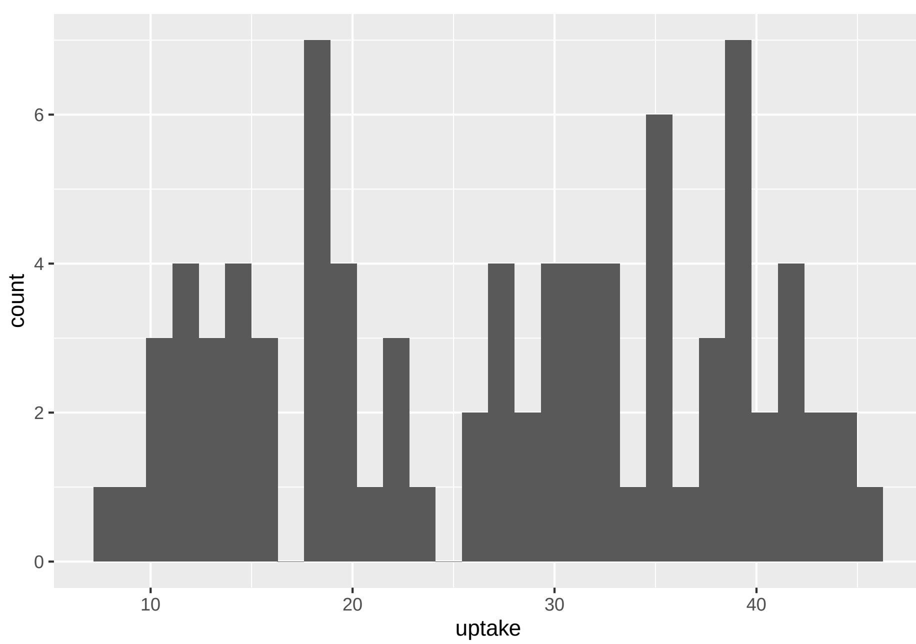
9.2.2.1 Histogram

```
df_co2 %>% ggplot(aes(x = conc)) + geom_histogram()
#> `stat_bin()` using `bins = 30`. Pick better value with
```

```
#> `binwidth`.
```

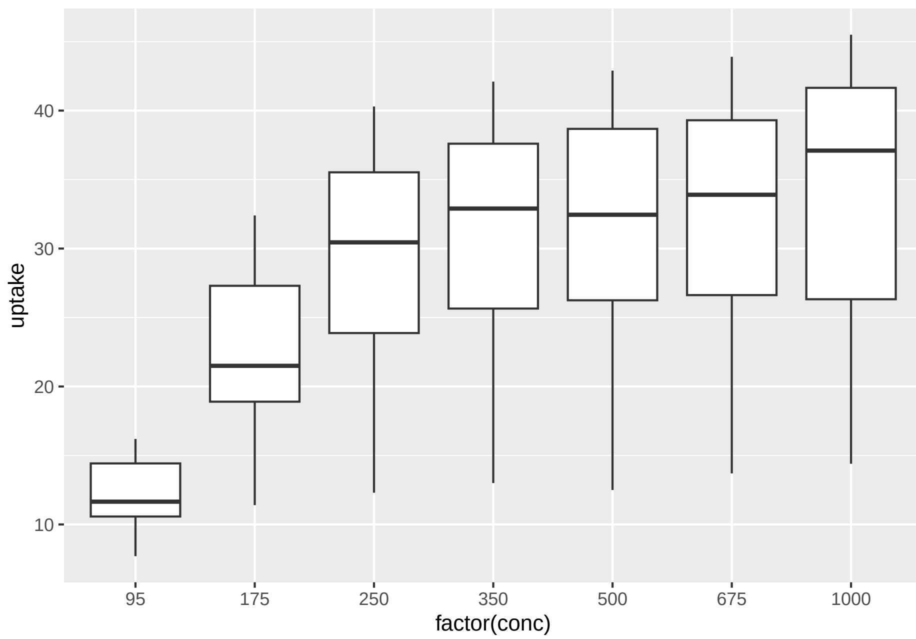



```
df_co2 %>% ggplot(aes(x = uptake)) + geom_histogram()
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
```

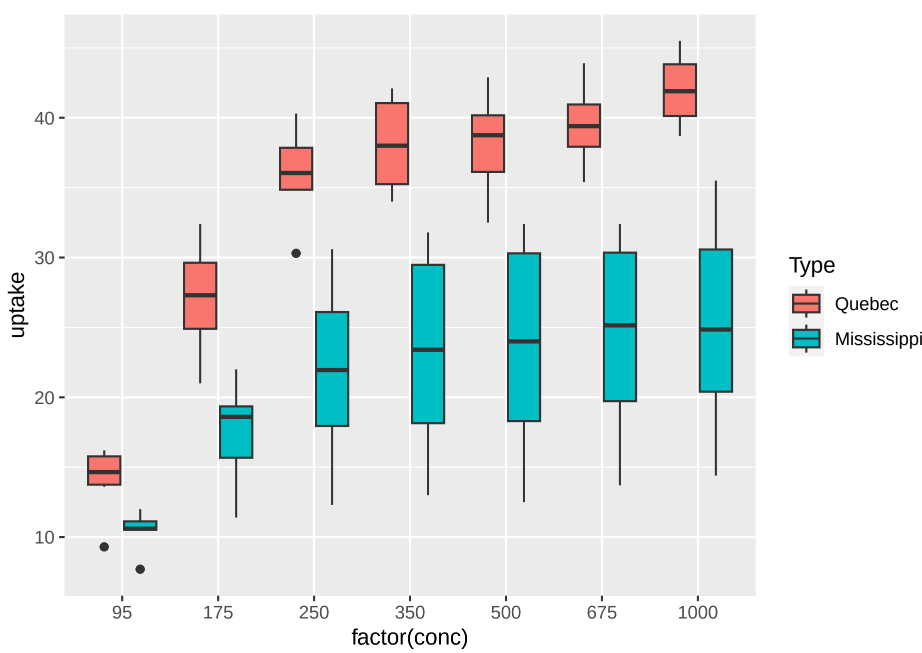


9.2.2.2 Box Plots

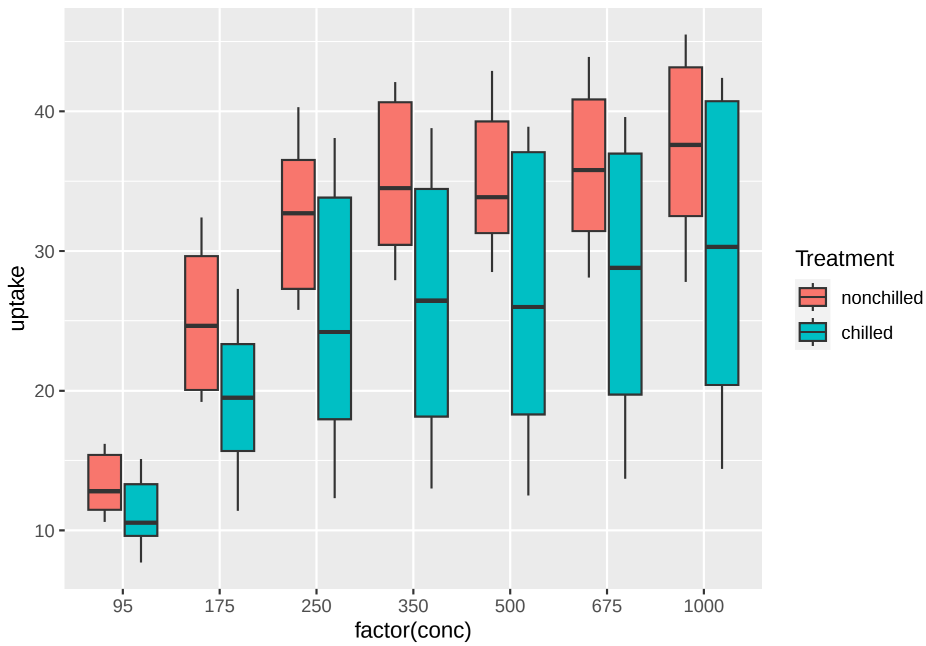
```
df_co2 %>% ggplot(aes(x = factor(conc), y = uptake)) + geom_boxplot()
```



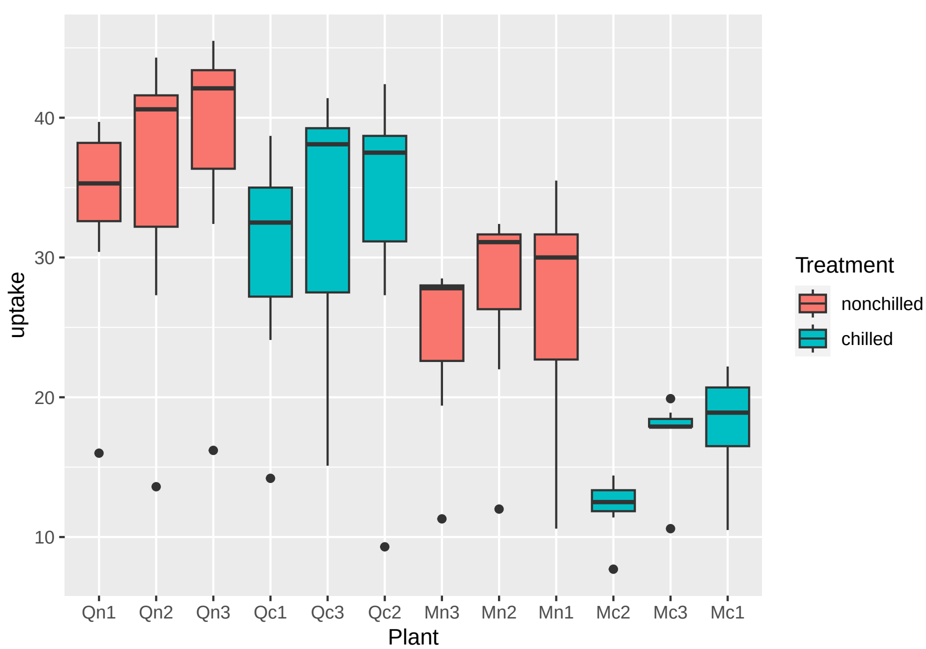

```
df_co2 %>% ggplot(aes(x = factor(conc), y = uptake, fill = Type)) + geom_boxplot()
```



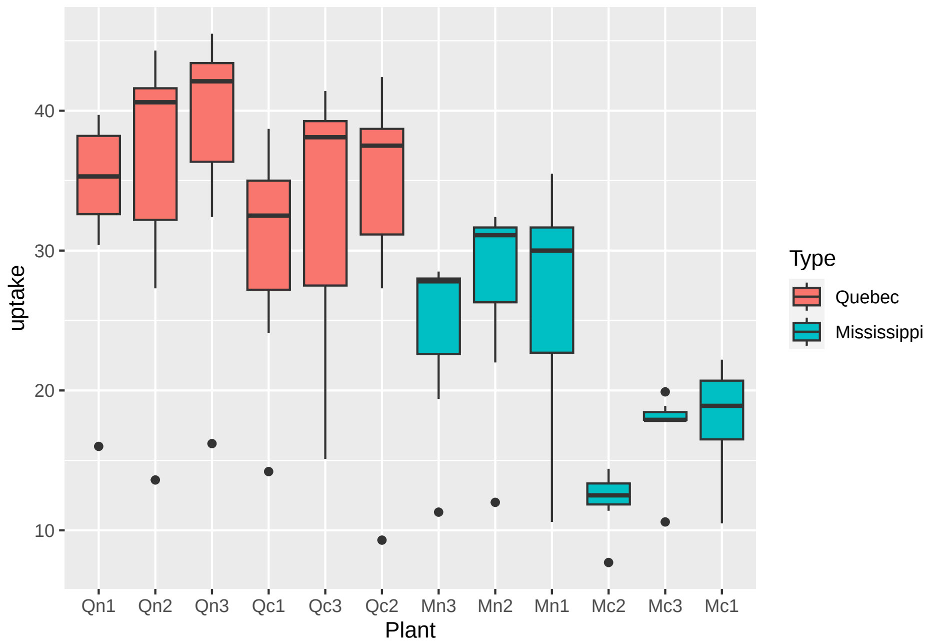
```
df_co2 %>% ggplot(aes(x = factor(conc), y = uptake, fill = Treatment)) + geom_boxplot()
```



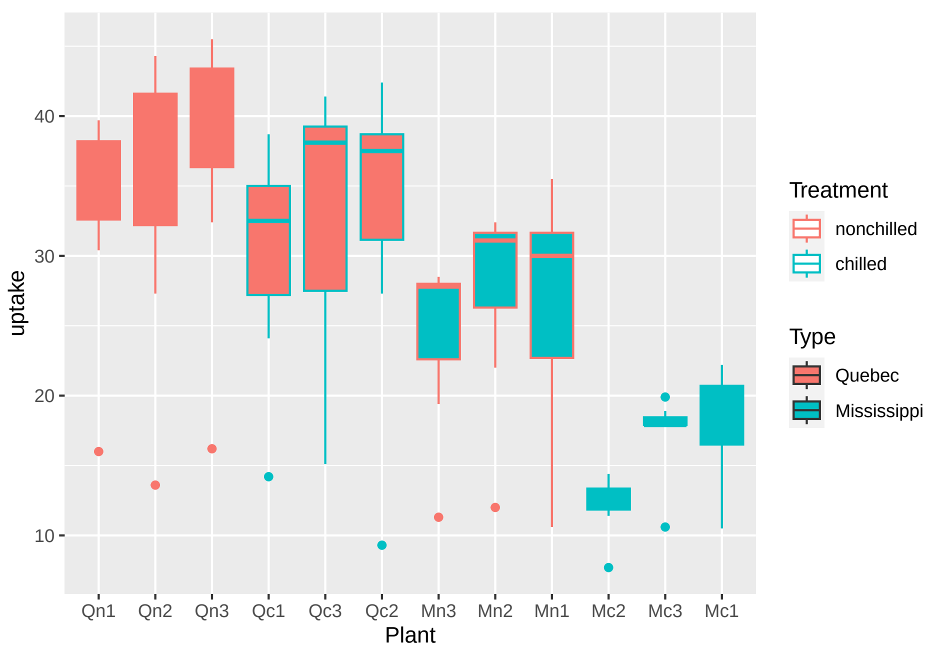

```
df_co2 %>% ggplot(aes(x = Plant, y = uptake, fill = Treatment)) + geom_boxplot()
```




```
df_co2 %>% ggplot(aes(x = Plant, y = uptake, fill = Type)) + geom_boxplot()
```



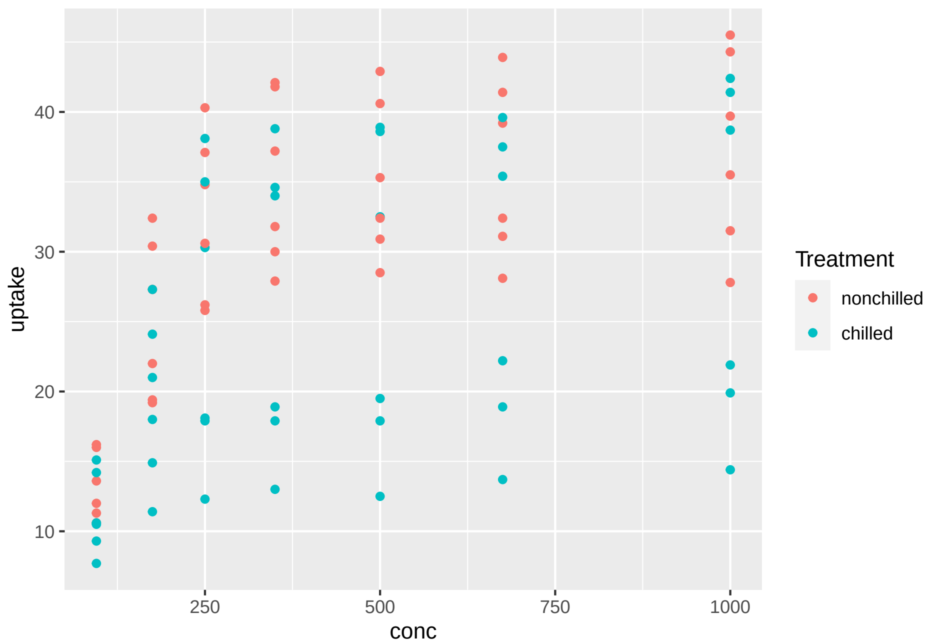

```
df_co2 %>% ggplot(aes(x = Plant, y = uptake, fill = Type, color = Treatment)) + geom_boxplot()
```



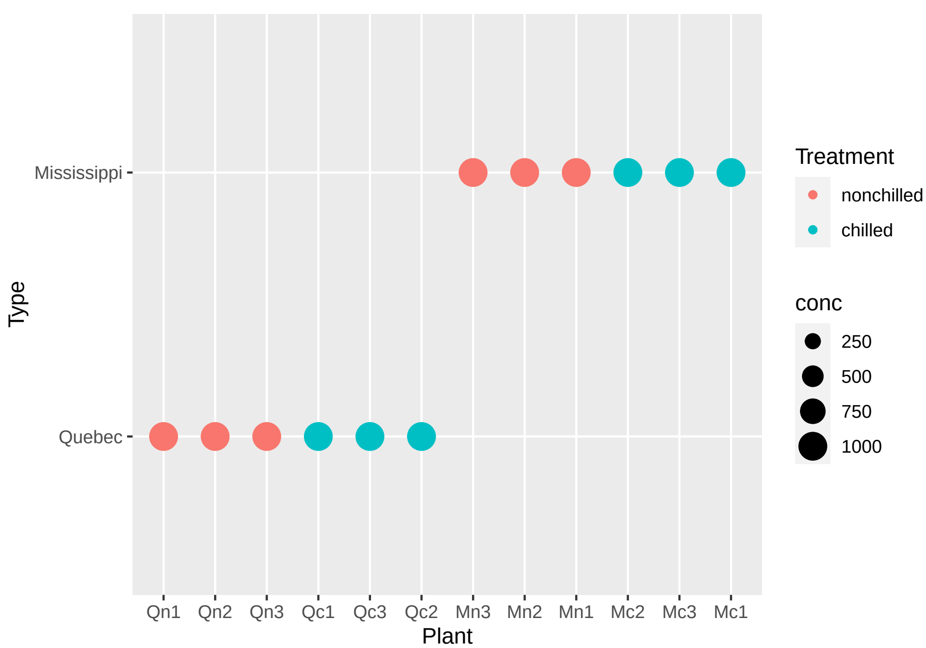
What can you see? Write your observations.

9.2.2.3 Scatter Plots

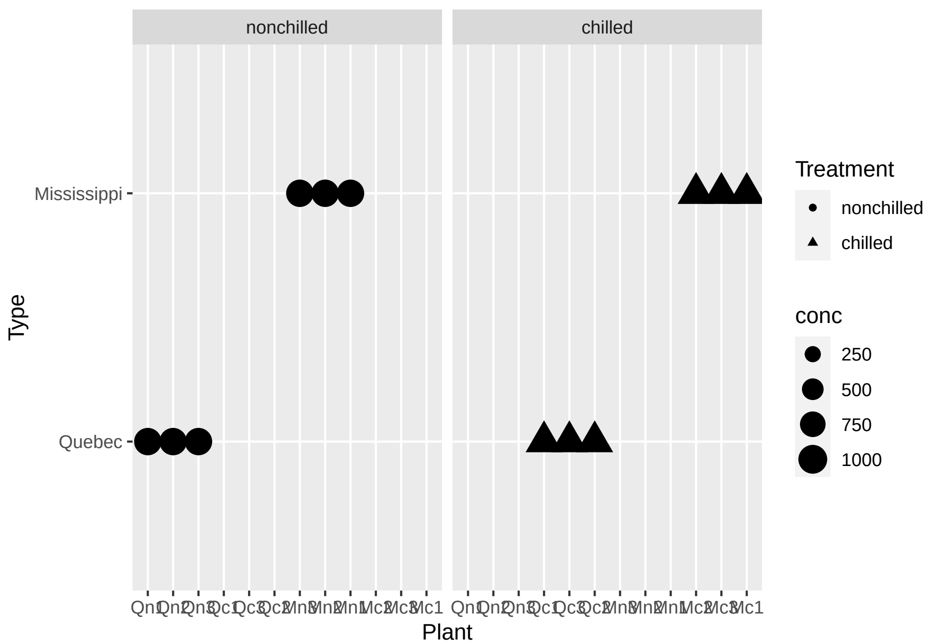
```
df_co2 %>% ggplot(aes(x = conc, y = uptake, color = Treatment)) + geom_point()
```



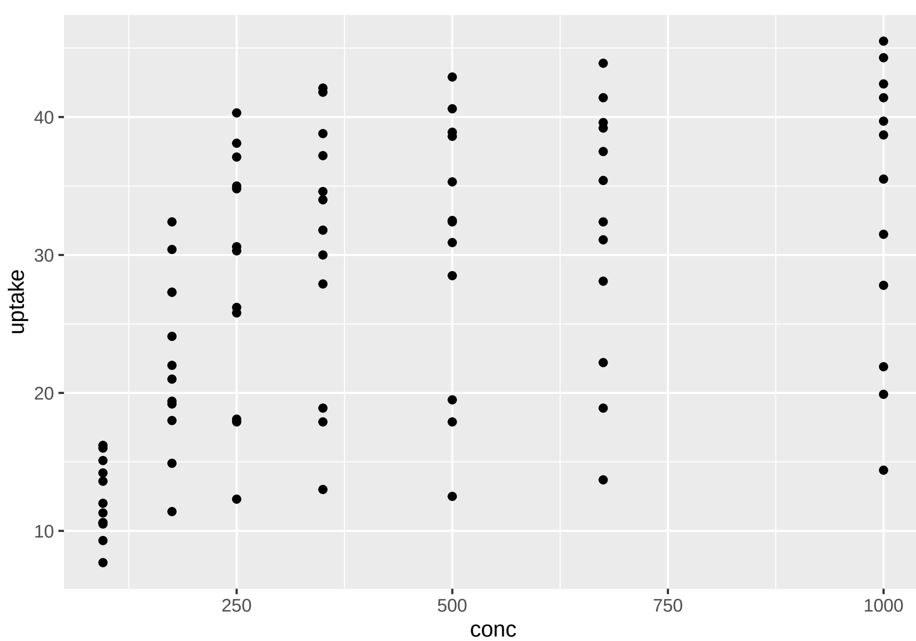

```
df_co2 %>% ggplot(aes(x = Plant, y = Type, color = Treatment, size = conc)) + geom_point()
```



```
df_co2 %>% ggplot(aes(x = Plant, y = Type, size = conc, shape = Treatment)) + geom_point() + facet_wrap(vars(Treatment))
```




```
ggplot(data = df_co2) +
  geom_point(aes(x = conc, y = uptake))
```



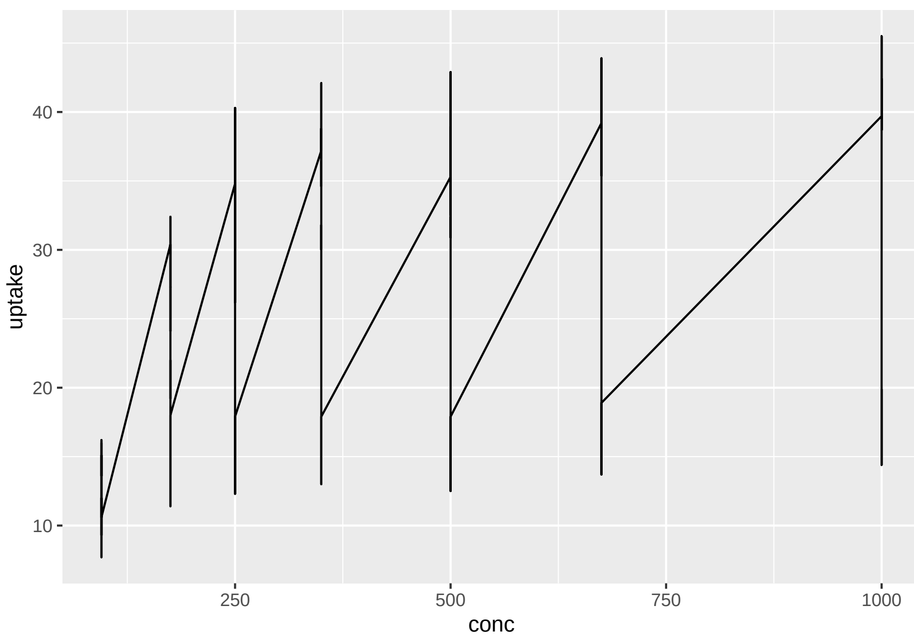
The following prints a vector.

```
df_co2 %>% distinct(conc) %>% pull()
#> [1]   95  175  250  350  500  675 1000
```

The following code generates a data frame.

```
df_co2 %>% distinct(conc)
#> # A tibble: 7 x 1
#>    conc
#>   <dbl>
#> 1    95
#> 2   175
#> 3   250
#> 4   350
#> 5   500
#> 6   675
#> 7  1000
```

```
ggplot(data = CO2) +
  geom_line(aes(x = conc, y = uptake))
```



The code above did not work, and the line graph is not appropriate in this case. There are so many `update` values at the same `conc`.

9.2.3 Example. `datasets::Seatbelts`

Search the data information.

Road Casualties in Great Britain 1969–84

- Seatbelts is a multiple time series, with columns
- DriversKilled: car drivers killed.
- drivers: same as UKDriverDeaths.
- front: front-seat passengers killed or seriously injured.
- rear: rear-seat passengers killed or seriously injured.
- kms: distance driven.
- PetrolPrice: petrol price.
- VanKilled: number of van ('light goods vehicle') drivers.
- law: 0/1: was the law in effect that month?

References Harvey, A. C. and Durbin, J. (1986). The effects of seat belt legislation on British road casualties: A case study in structural time series modelling. Journal of the Royal Statistical Society series A, 149, 187–227. doi:10.2307/2981553.

The paper is available as you log-in to ICU Library > E-Databases > JSTOR

Can you see the difference of the following two codes?

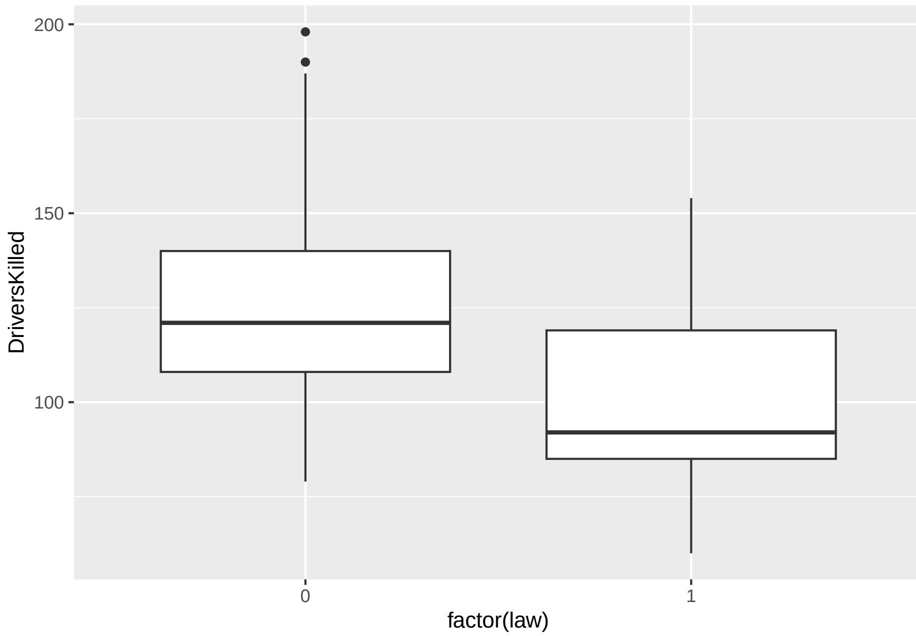
```
head(Seatbelts)
#>      DriversKilled drivers front rear   kms PetrolPrice
#> [1,]           107    1687   867  269  9059   0.1029718
#> [2,]            97    1508   825  265  7685   0.1023630
#> [3,]           102    1507   806  319  9963   0.1020625
#> [4,]            87    1385   814  407 10955   0.1008733
#> [5,]           119    1632   991  454 11823   0.1010197
#> [6,]           106    1511   945  427 12391   0.1005812
#>      VanKilled law
#> [1,]        12   0
#> [2,]         6   0
#> [3,]        12   0
#> [4,]         8   0
#> [5,]        10   0
#> [6,]        13   0
```

```
df_sb <- as_tibble(datasets::Seatbelts)
df_sb
#> # A tibble: 192 x 8
#>    Drivers~1 drivers front  rear   kms Petro~2 VanKi~3   law
#>        <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl> <dbl>
#>  1       107    1687   867   269  9059   0.103      12     0
#>  2        97    1508   825   265  7685   0.102       6     0
#>  3       102    1507   806   319  9963   0.102      12     0
#>  4        87    1385   814   407 10955   0.101       8     0
#>  5       119    1632   991   454 11823   0.101      10     0
#>  6       106    1511   945   427 12391   0.101      13     0
#>  7       110    1559  1004   522 13460   0.104      11     0
#>  8       106    1630  1091   536 14055   0.104       6     0
#>  9       107    1579   958   405 12106   0.104      10     0
#> 10       134    1653   850   437 11372   0.103      16     0
#> # ... with 182 more rows, and abbreviated variable names
#> #   1: DriversKilled, 2: PetrolPrice, 3: VanKilled
```

```
summary(df_sb)
#>  DriversKilled      drivers         front
#>  Min.   : 60.0   Min.   :1057   Min.   : 426.0
#>  1st Qu.:104.8   1st Qu.:1462   1st Qu.: 715.5
#>  Median :118.5   Median :1631   Median : 828.5
#>  Mean   :122.8   Mean   :1670   Mean   : 837.2
#>  3rd Qu.:138.0   3rd Qu.:1851   3rd Qu.: 950.8
#>  Max.   :198.0   Max.   :2654   Max.   :1299.0
#>       rear            kms          PetrolPrice
#>  Min.   :224.0   Min.   : 7685   Min.   :0.08118
#>  1st Qu.:344.8   1st Qu.:12685   1st Qu.:0.09258
#>  Median :401.5   Median :14987   Median :0.10448
#>  Mean   :401.2   Mean   :14994   Mean   :0.10362
#>  3rd Qu.:456.2   3rd Qu.:17202   3rd Qu.:0.11406
#>  Max.   :646.0   Max.   :21626   Max.   :0.13303
#>    VanKilled           law
#>  Min.   : 2.000   Min.   :0.0000
#>  1st Qu.: 6.000   1st Qu.:0.0000
#>  Median : 8.000   Median :0.0000
#>  Mean   : 9.057   Mean   :0.1198
#>  3rd Qu.:12.000   3rd Qu.:0.0000
#>  Max.   :17.000   Max.   :1.0000
```

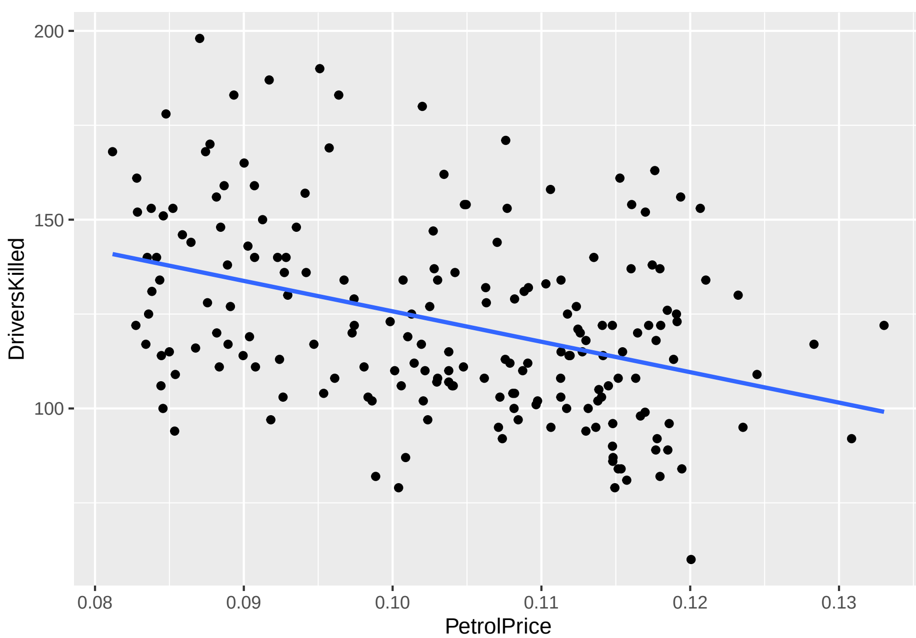
Which visualization do you apply?

```
df_sb %>% ggplot(aes(x = factor(law), y = DriversKilled)) + geom_boxplot()
```



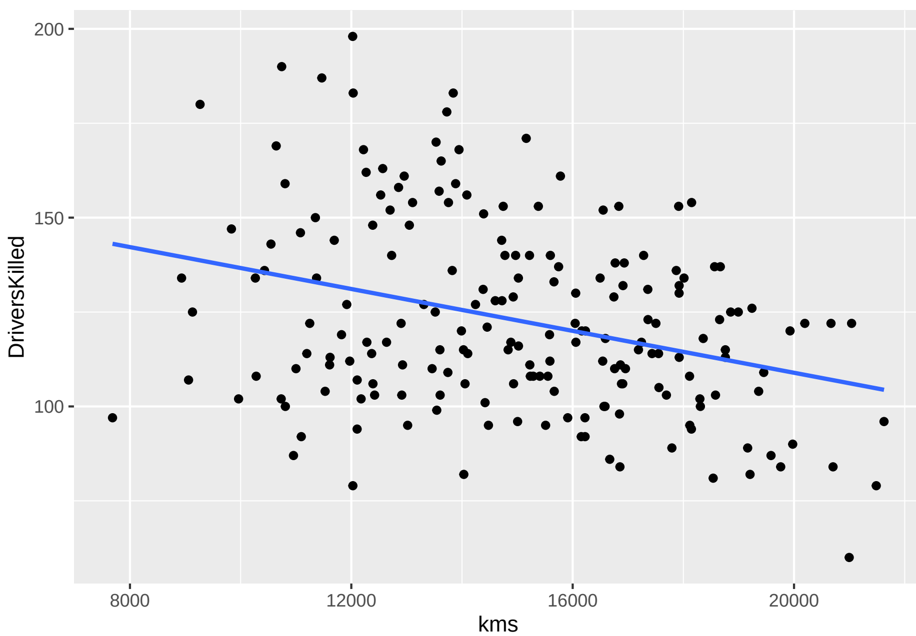
What do you observe above?

```
df_sb %>% ggplot(aes(x = PetrolPrice, y = DriversKilled)) + geom_point() +
  geom_smooth(formula = y~x, method = "lm", se = FALSE)
```



What can you see above?

```
df_sb %>% ggplot(aes(x = kms, y = DriversKilled)) + geom_point() +
  geom_smooth(formula = y~x, method = "lm", se = FALSE)
```

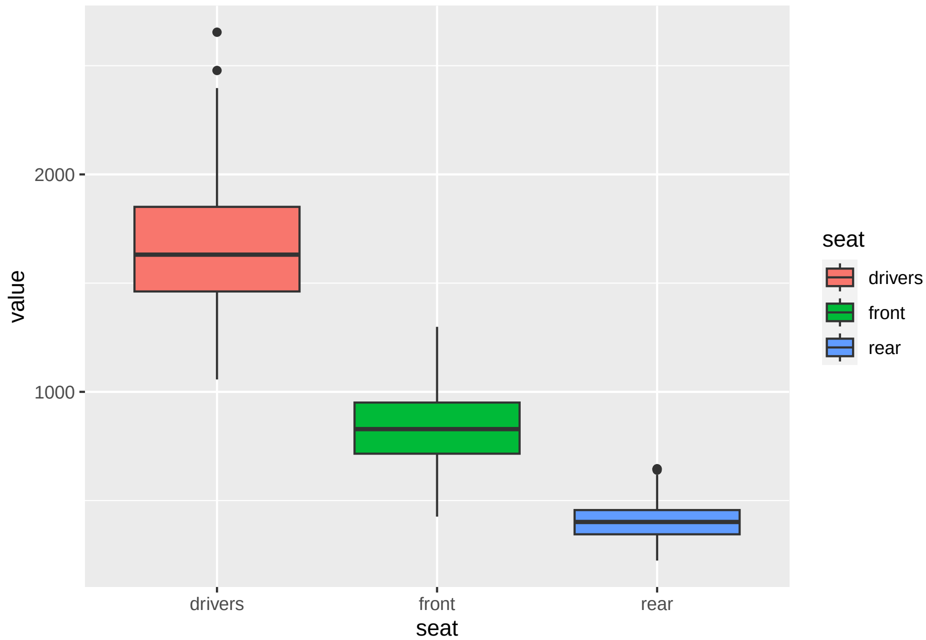


What can you see above?

We will learn how to use `pivot_longer` and `pivot_wider` in EDA4.

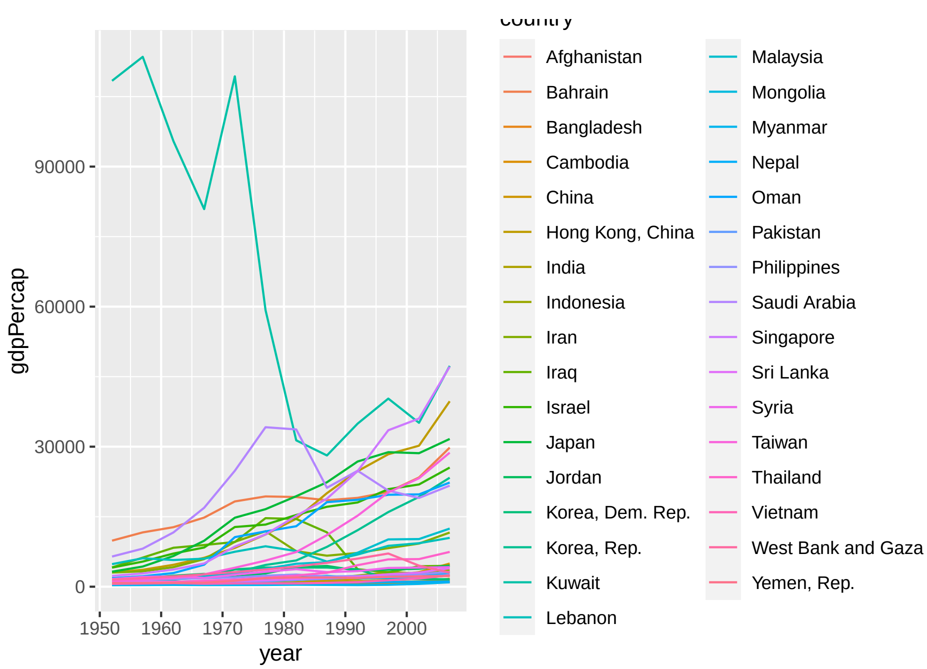
```
df_sb %>%
  pivot_longer(cols = 2:4, names_to = "seat", values_to = "value")
#> # A tibble: 576 x 7
#>    DriversKilled   kms PetrolPrice VanKi~1   law seat  value
#>            <dbl> <dbl>       <dbl>   <dbl> <dbl> <chr> <dbl>
#>  1           107  9059       0.103      12     0 driv~  1687
#>  2           107  9059       0.103      12     0 front   867
#>  3           107  9059       0.103      12     0 rear    269
#>  4            97  7685       0.102       6     0 driv~  1508
#>  5            97  7685       0.102       6     0 front   825
#>  6            97  7685       0.102       6     0 rear    265
#>  7           102  9963       0.102      12     0 driv~  1507
#>  8           102  9963       0.102      12     0 front   806
#>  9           102  9963       0.102      12     0 rear    319
#> 10            87 10955       0.101       8     0 driv~  1385
#> # ... with 566 more rows, and abbreviated variable name
#> #   1: VanKilled
```

```
df_sb %>%
  pivot_longer(cols = 2:4, names_to = "seat", values_to = "value") %>%
  ggplot() + geom_boxplot(aes(x = seat, y = value, fill = seat))
```



What can you observe?

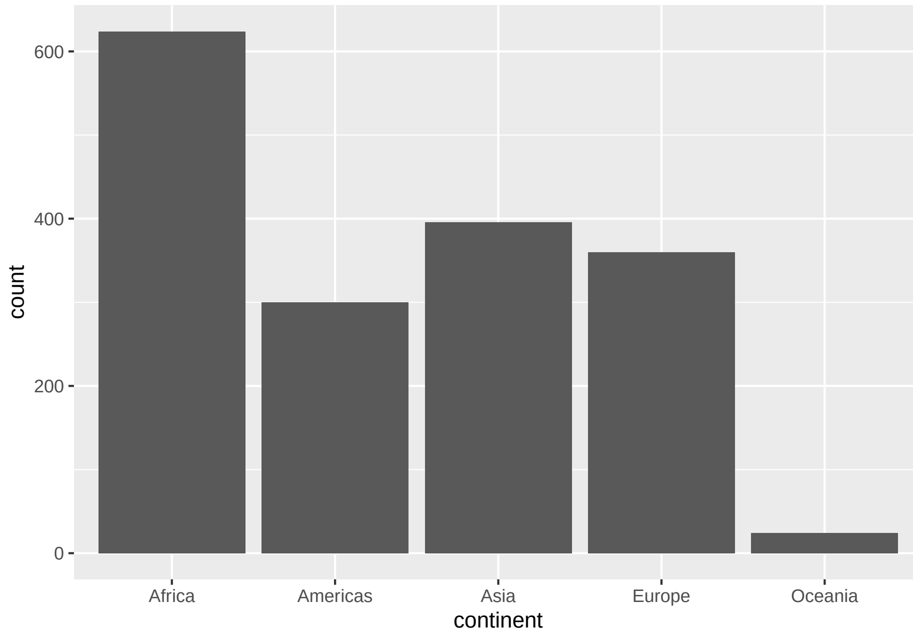
```
df %>% filter(continent == "Asia") %>%
  ggplot(aes(x = year, y = gdpPercap, color = country)) + geom_line()
```



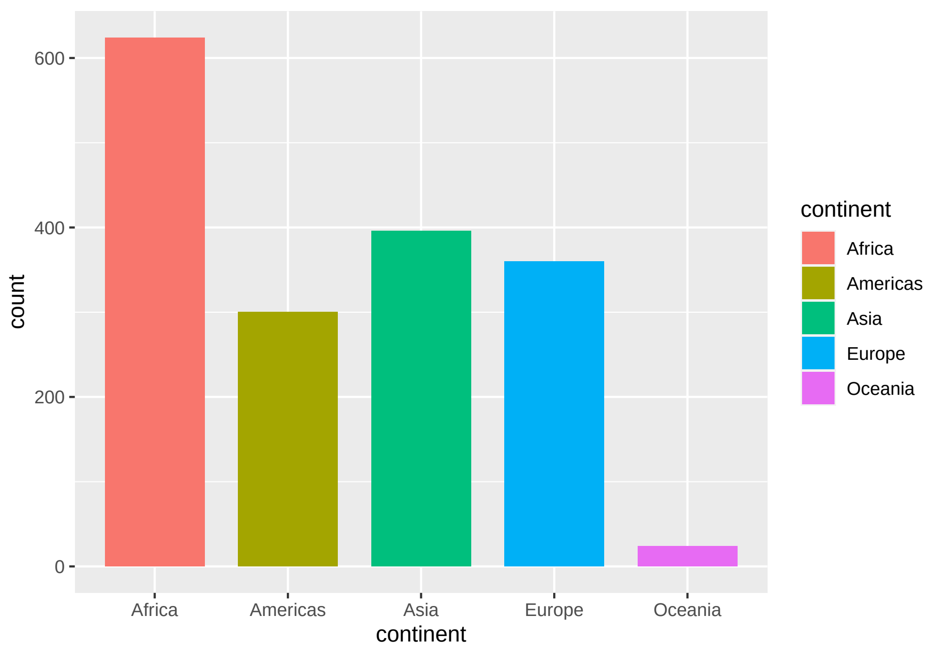
Appropriate graph?

9.3 Gapminder

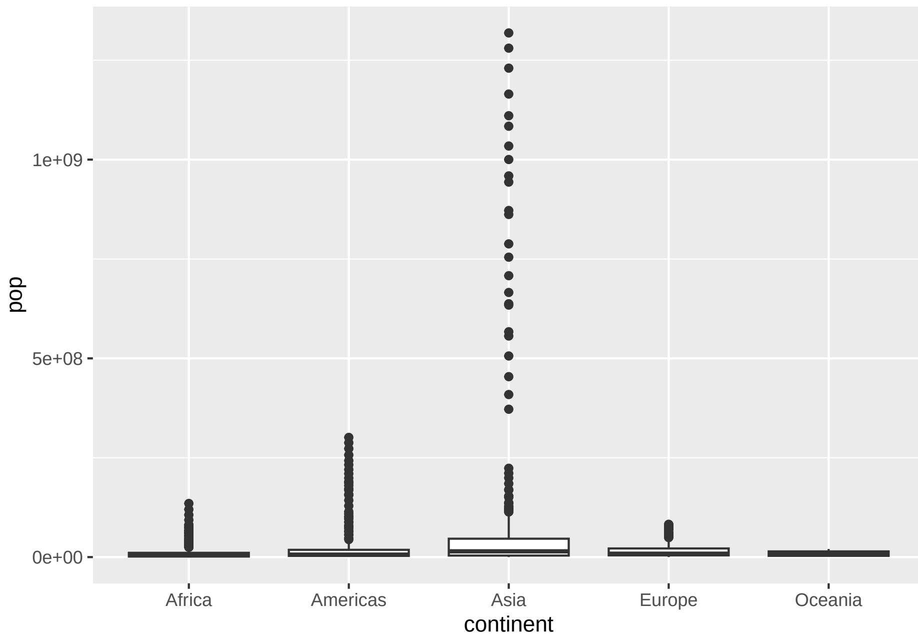
```
ggplot(df, aes(x = continent)) + geom_bar()
```



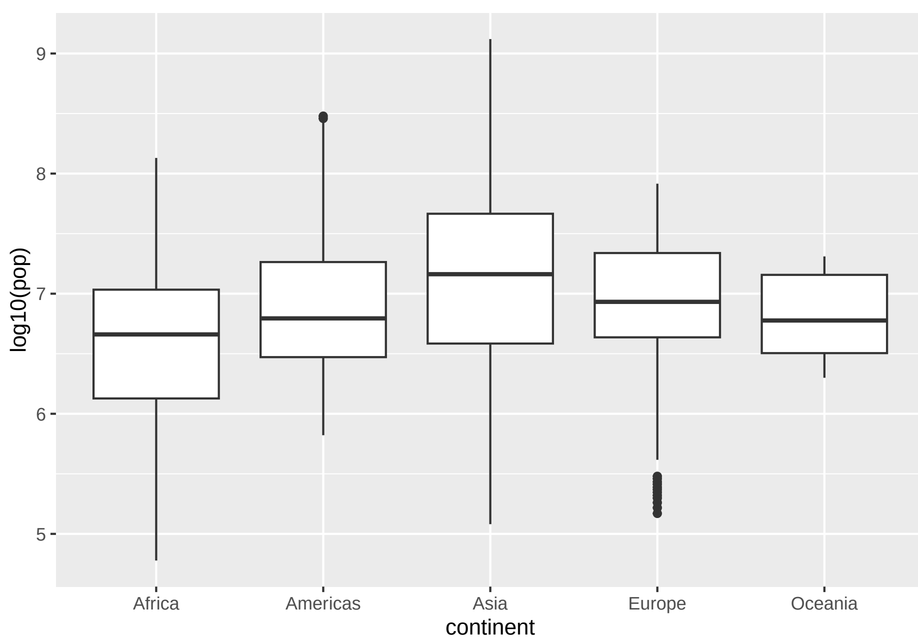

```
ggplot(df, aes(x = continent, fill = continent)) + geom_bar(width = 0.75)
```



```
ggplot(df, aes(x = continent, y = pop)) + geom_boxplot()
```

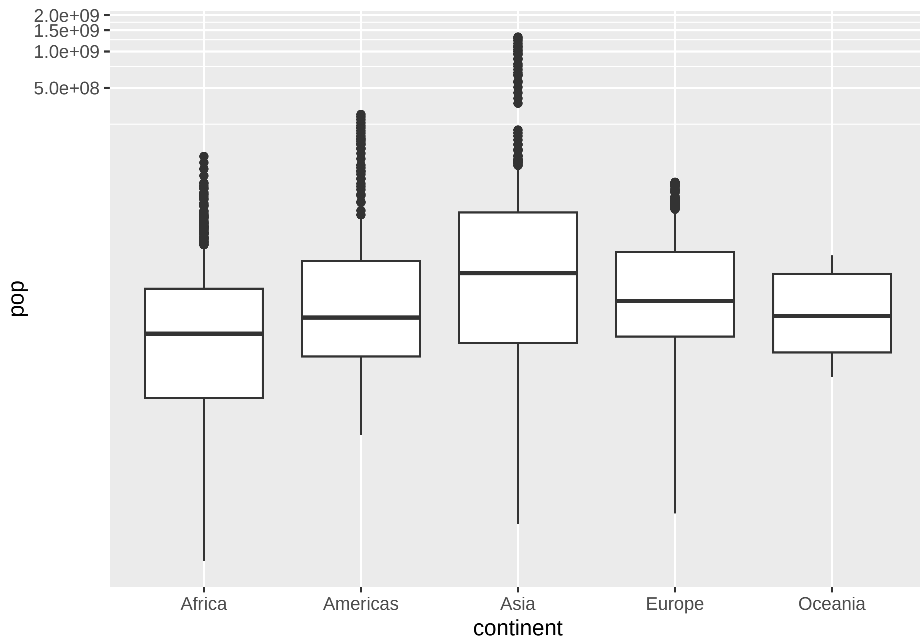



```
ggplot(df, aes(x = continent, y = log10(pop))) + geom_boxplot()
```

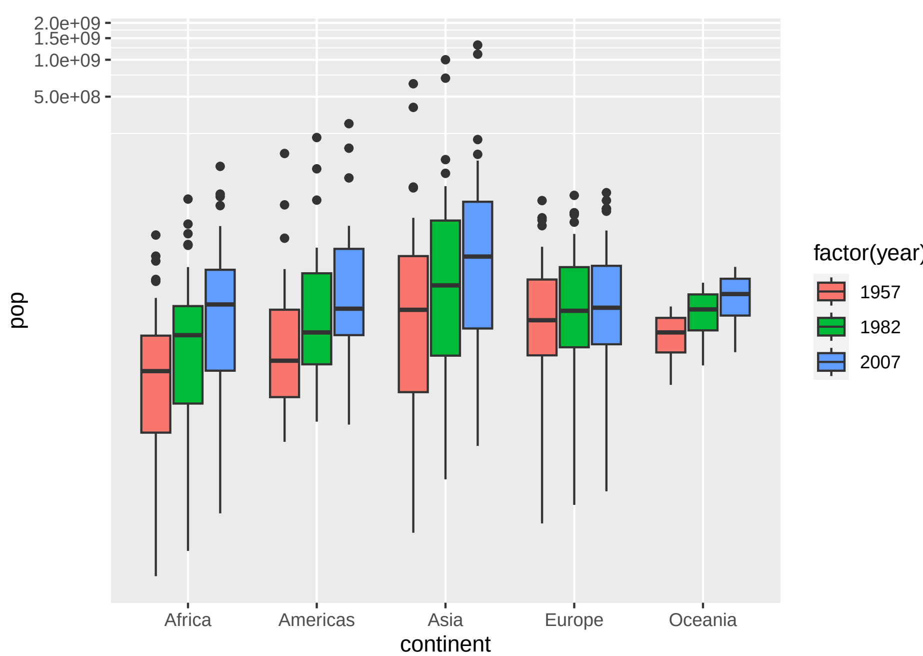


Alternately, you can use `coord_trans(x = "identity", y = "log10")` in stead of `y = log10(pop)`. Can you see the difference?

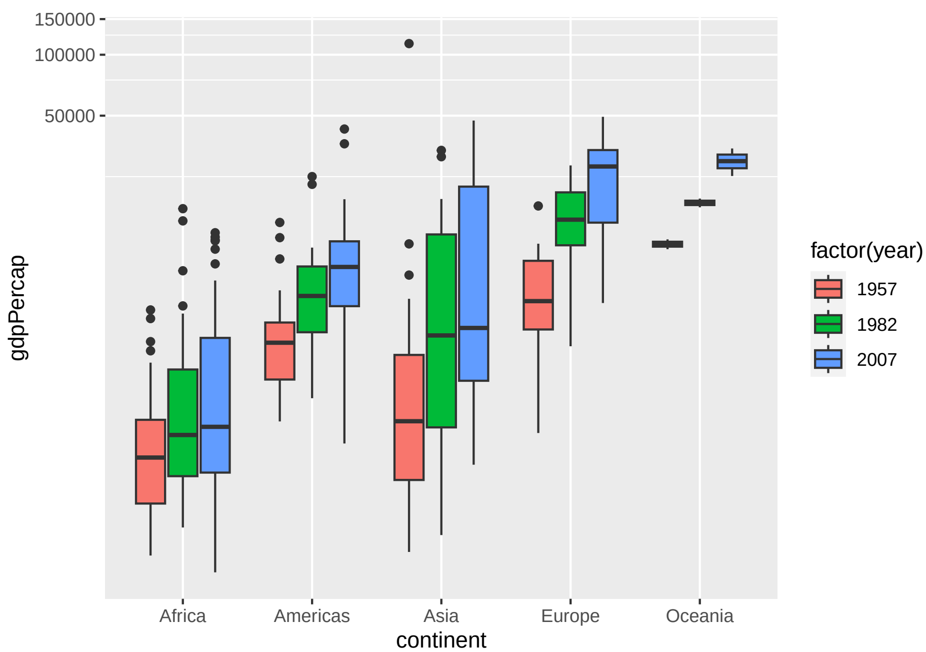
```
ggplot(df, aes(x = continent, y = pop)) + geom_boxplot() +
  coord_trans(x = "identity", y = "log10")
```



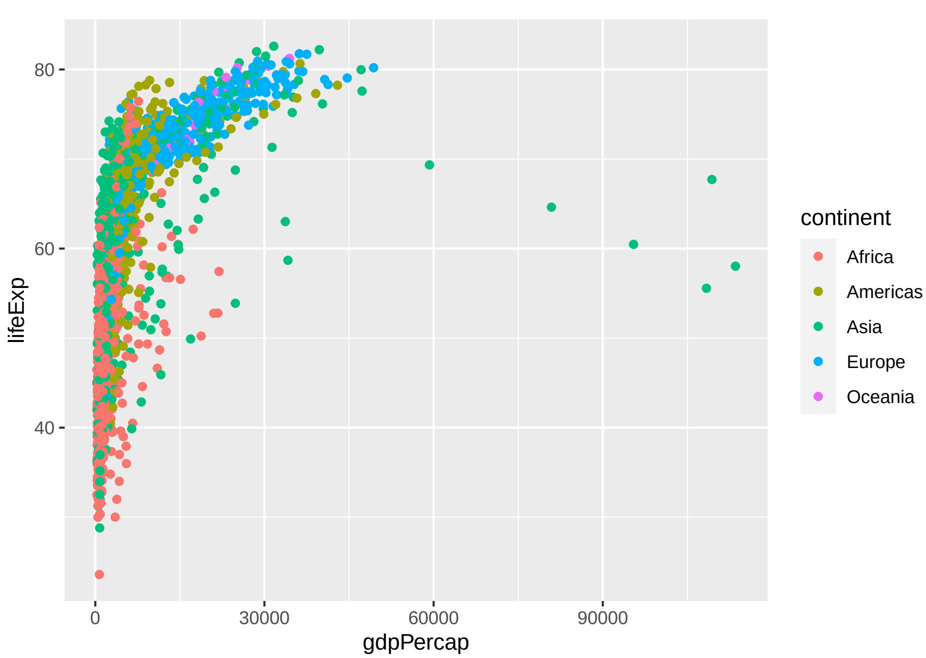

```
df %>% filter(year %in% c(1957, 1982, 2007)) %>%
  ggplot() + geom_boxplot(aes(x = continent, y = pop, fill = factor(year))) +
    coord_trans(x = "identity", y = "log10")
```



```
df %>% filter(year %in% c(1957, 1982, 2007)) %>%
  ggplot() + geom_boxplot(aes(x = continent, y = gdpPercap, fill = factor(year))) +
  coord_trans(x = "identity", y = "log10")
```

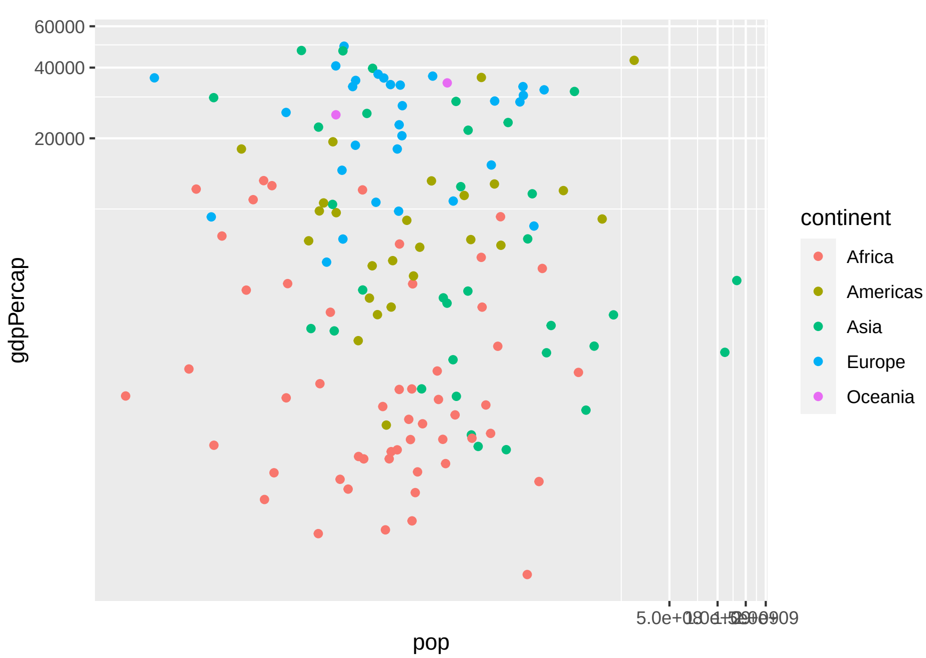



```
ggplot(df, aes(gdpPercap, lifeExp)) + geom_point(aes(color=continent))
```

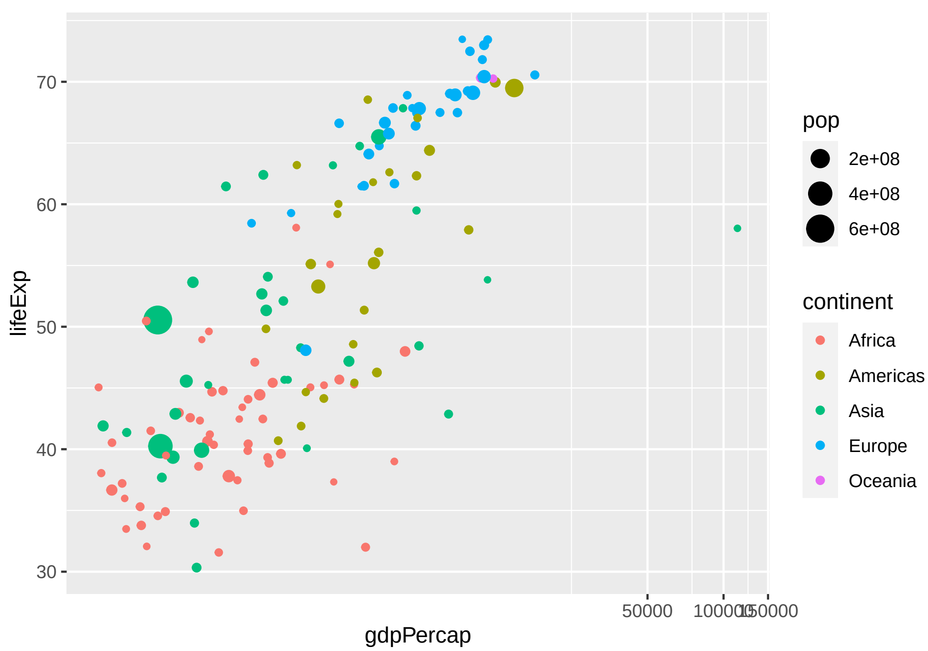


```
df %>% ggplot(aes(gdpPercap, lifeExp, color = continent)) + geom_point() +
  coord_trans(x = "log10", y = "identity")
```

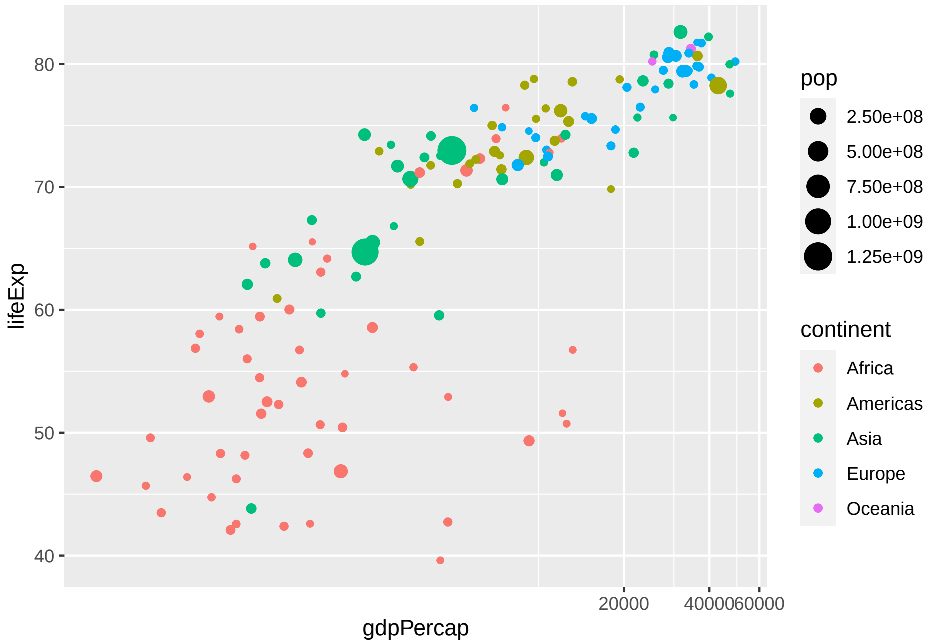

```
df %>% filter(year %in% c(2007)) %>%
  ggplot() + geom_point(aes(x = pop, y = gdpPercap, col = continent)) +
  coord_trans(x = "log10", y = "log10")
```



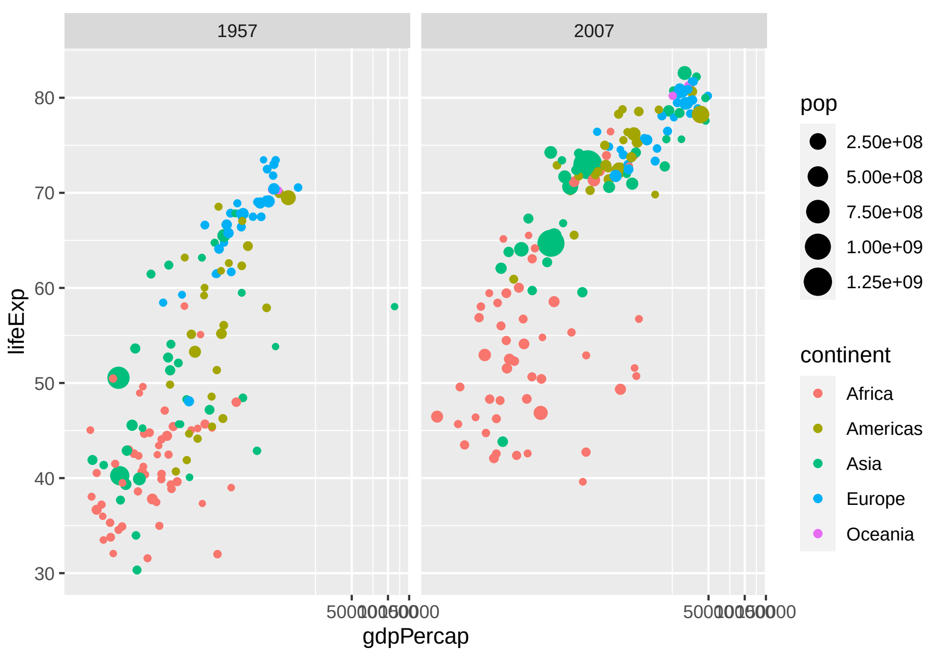
```
df %>% filter(year %in% c(1957)) %>%
  ggplot() + geom_point(aes(x = gdpPercap, y = lifeExp, col = continent, size = pop)) +
  coord_trans(x = "log10", y = "identity")
```




```
df %>% filter(year %in% c(2007)) %>%
  ggplot() + geom_point(aes(x = gdpPercap, y = lifeExp, col = continent, size = pop)) +
  coord_trans(x = "log10", y = "identity")
```




```
df %>% filter(year %in% c(1957, 2007)) %>%
  ggplot() + geom_point(aes(x = gdpPercap, y = lifeExp, col = continent, size = pop)) +
  coord_trans(x = "log10", y = "identity") +
  facet_wrap(vars(year))
```

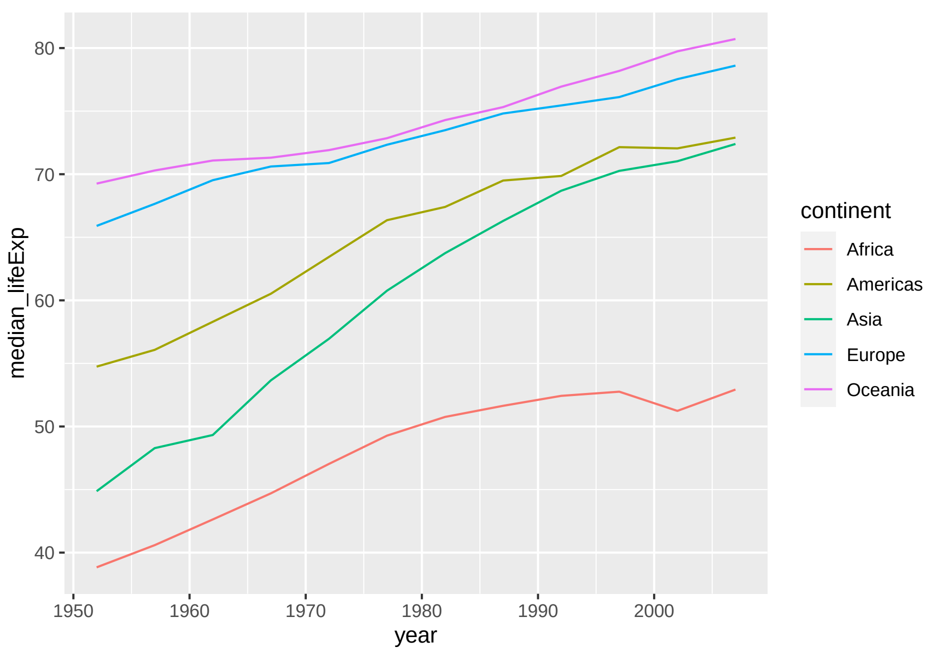



```
df_lifeExp <- df %>% group_by(continent, year) %>%
  summarize(mean_lifeExp = mean(lifeExp), median_lifeExp = median(lifeExp), max_lifeExp = max(lifeExp), min_lifeExp = min(
```

```
#> `summarise()` has grouped output by 'continent'. You can
#> override using the `.groups` argument.
```

The code above gives a message, but it works.

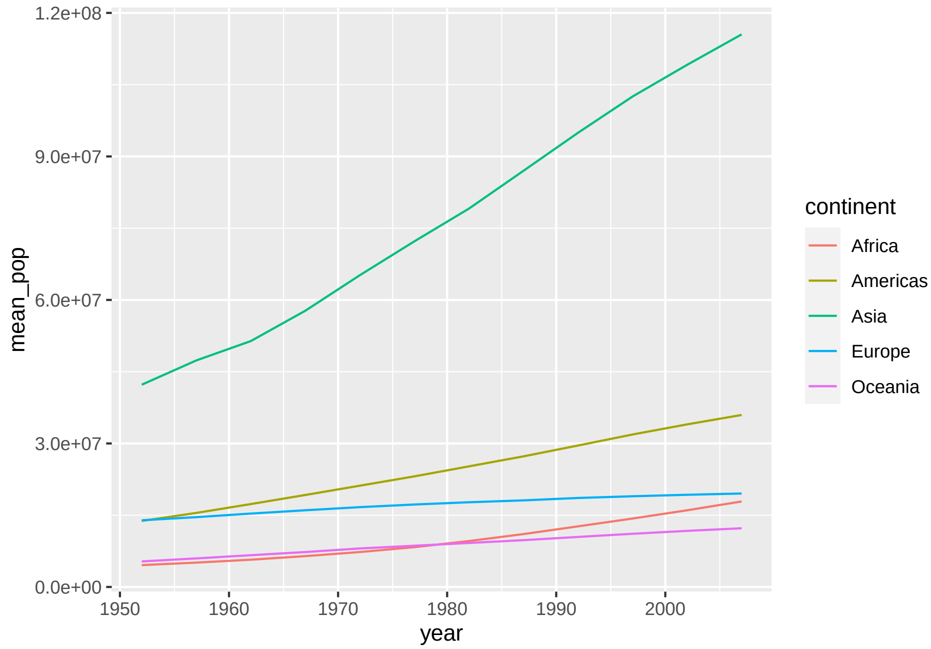
```
df_lifeExp %>% ggplot(aes(x = year, y = median_lifeExp, color = continent)) +
  geom_line()
```



If you do not want to have a message, the following is an option. Otherwise, grouping is kept and you can get the original data back by `ungroup()`.

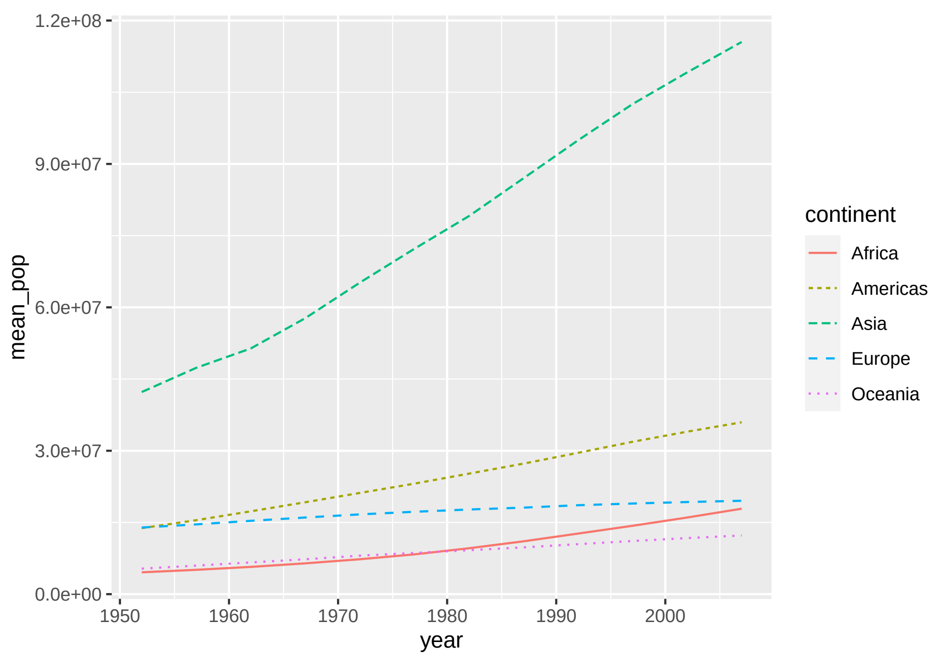
```
df_pop <- df %>% group_by(continent, year) %>%
  summarize(mean_pop = mean(pop), median_pop = median(pop), max_pop = max(pop), min_pop = min(pop), .groups = "drop")
```

```
df_pop %>% ggplot(aes(x = year, y = mean_pop, color = continent)) +
  geom_line()
```

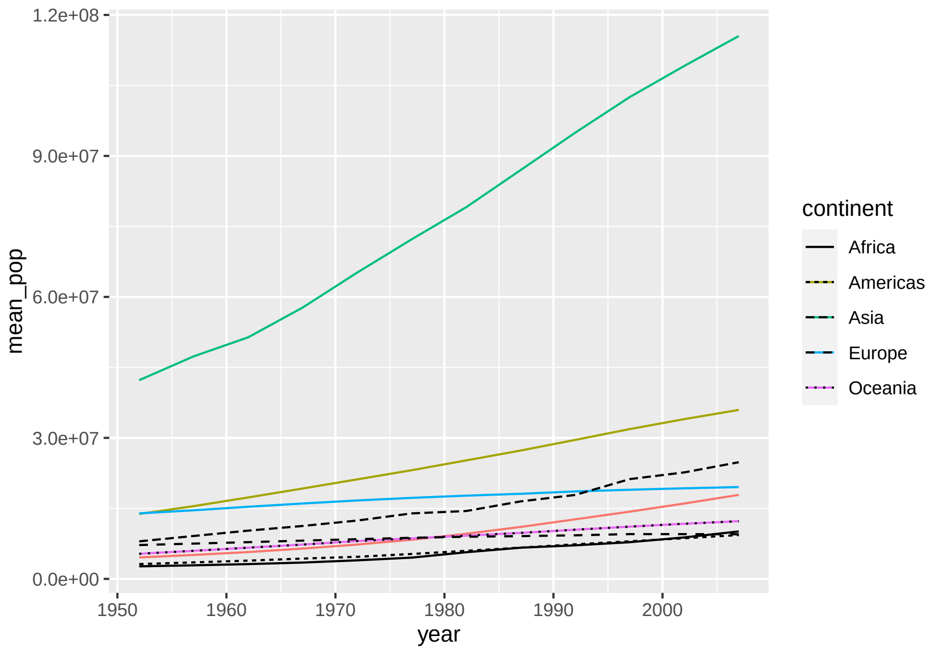


The following two codes create the same chart.

```
df_pop %>% ggplot(aes(x = year, y = mean_pop, color = continent, linetype = continent)) +
  geom_line()
```




```
df_pop %>% ggplot() +
  geom_line(aes(x = year, y = mean_pop, color = continent)) +
  geom_line(aes(x = year, y = median_pop, linetype = continent))
```



1.2e+08
9.0e+07
6.0e+07
3.0e+07
0.0e+00
mean_pop
1950
1960
1970
1980
1990
2000
year
continent
Africa
Americas
Asia
Europe
Oceania

Chapter 10

Responses to Questions

10.1 Q1. Two categorical variables and one numerical variables

Eg. Smoking, Alcohol and (O)esophageal Cancer

```
(df_esoph <- as_tibble(esoph))
#> # A tibble: 88 x 5
#>    agegp alcgp     tobgp    ncases ncontrols
#>    <ord> <ord>     <ord>     <dbl>     <dbl>
#>  1 25-34 0-39g/day 0-9g/day      0        40
#>  2 25-34 0-39g/day 10-19         0        10
#>  3 25-34 0-39g/day 20-29         0         6
#>  4 25-34 0-39g/day 30+           0         5
#>  5 25-34 40-79     0-9g/day      0        27
#>  6 25-34 40-79     10-19         0         7
#>  7 25-34 40-79     20-29         0         4
#>  8 25-34 40-79     30+           0         7
#>  9 25-34 80-119    0-9g/day      0         2
#> 10 25-34 80-119    10-19         0         1
#> # ... with 78 more rows
```

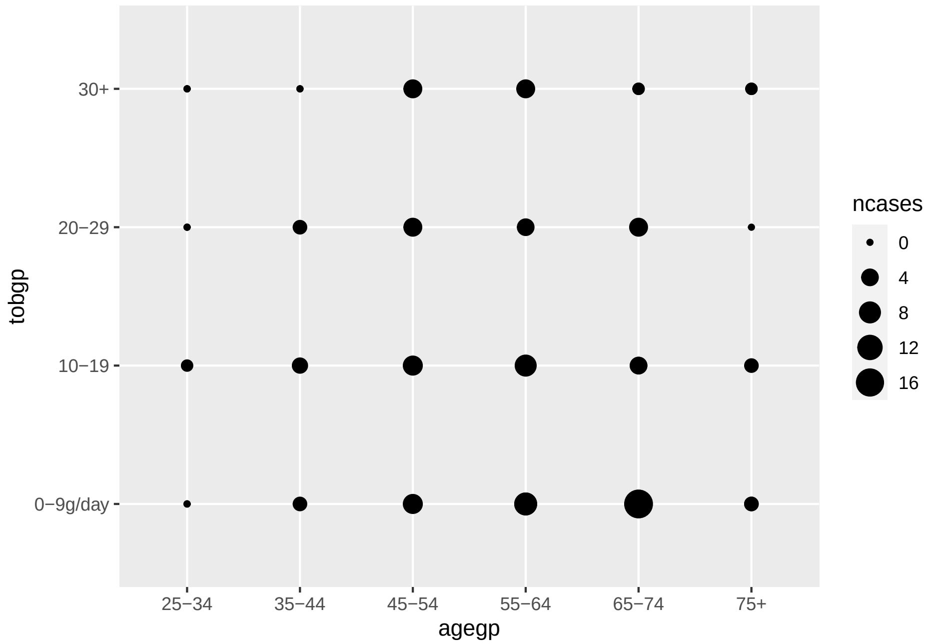
`df_esoph` has three categorical variables and one numerical variable `ncases` to investigate.

Comments: I wanted to include three variables in the first exercise to be able to compare tobacco consumption, number of cases of cancer, and age in the same graph but I was not able to do it.

Solutions: There are various ways you can choose from.

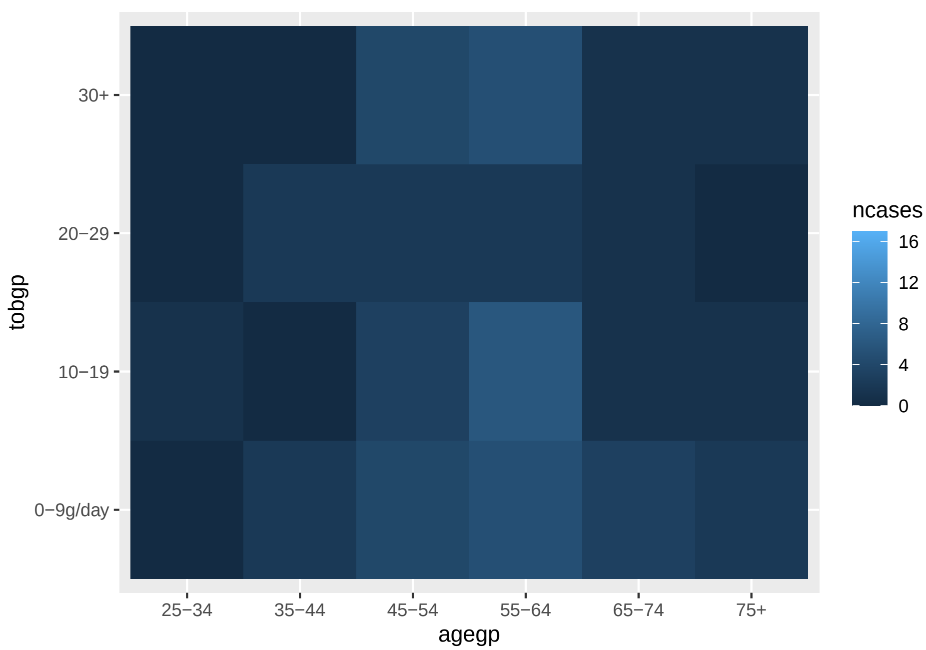
Scatter plot with size by `geom_point()`.

```
ggplot(df_esoph) + geom_point(aes(agegp, tobgp, size=ncases))
```



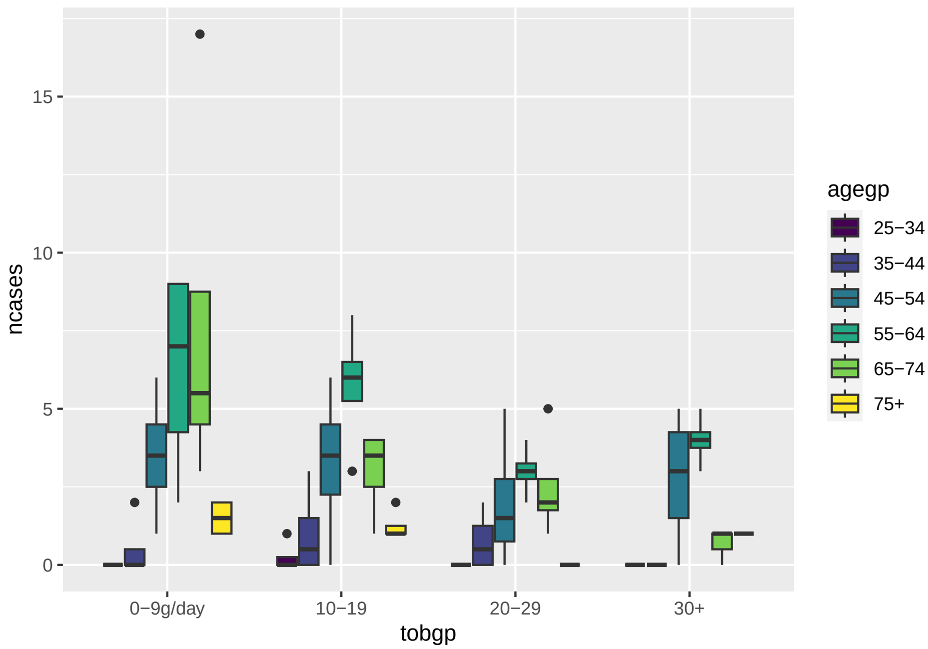
Heatmap with geom_tile()

```
ggplot(df_esoph) + geom_tile(mapping = aes(x = agegp, y = tobgp, fill = ncases))
```

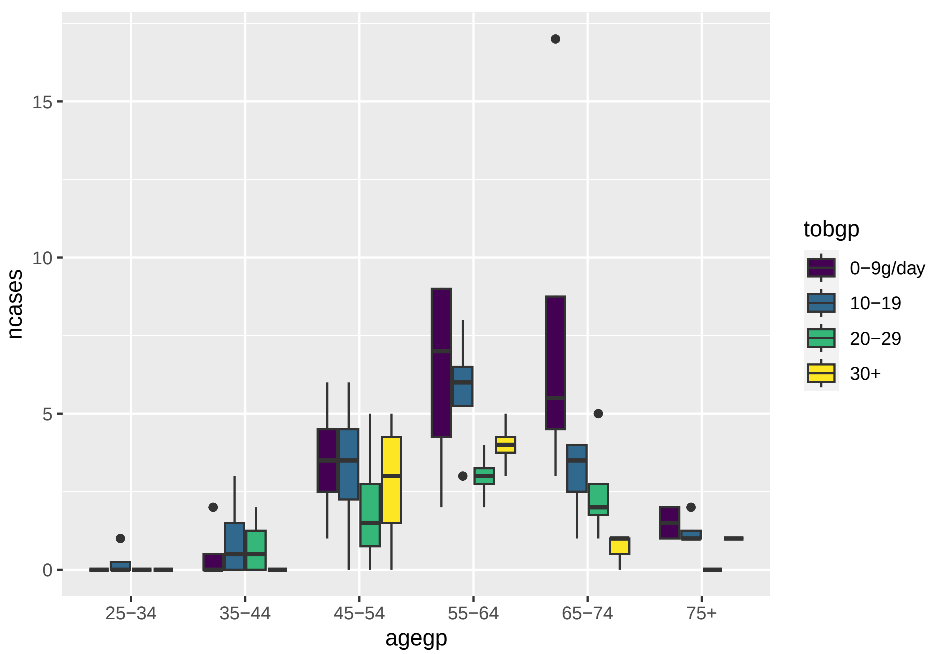


geom_boxplot()

```
ggplot(df_esoph, aes(x= tobgp, y=ncases, fill=agegp))+geom_boxplot()
```

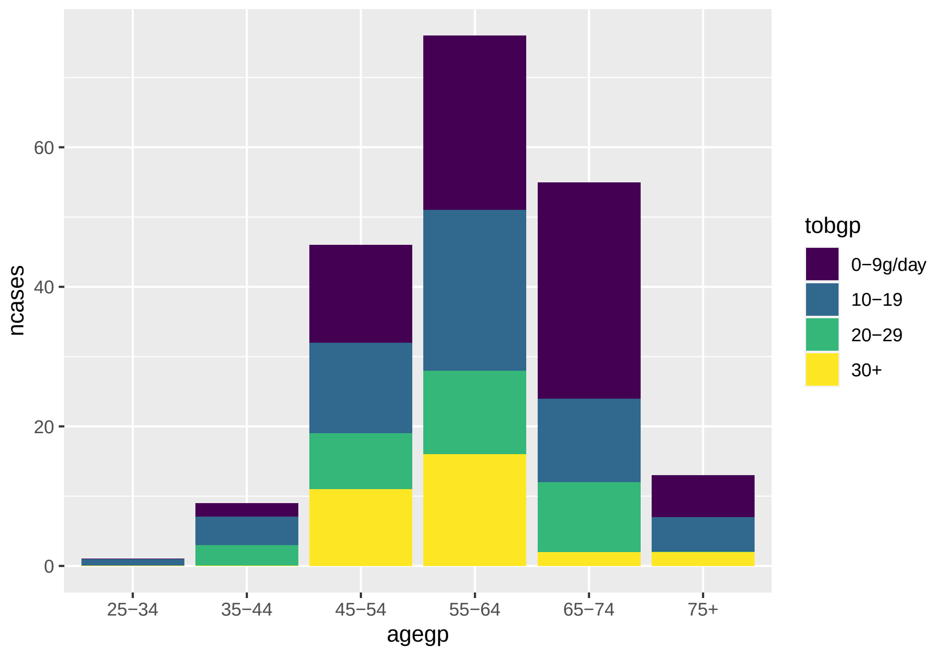



```
ggplot(df_esoph, aes(x= agegp, y=ncases, fill=tobgp))+geom_boxplot()
```



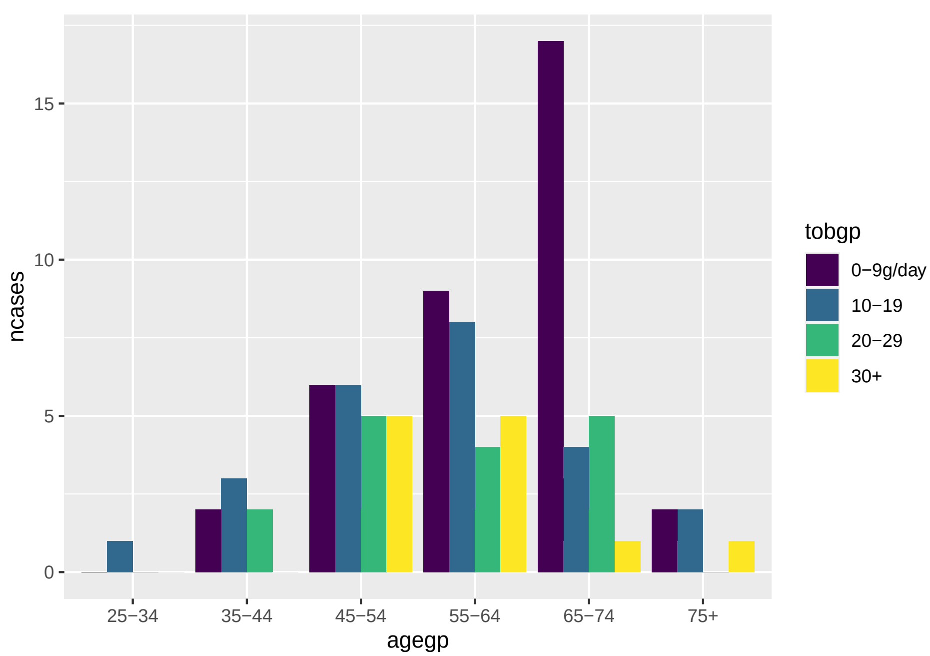

```
geom_col()
```

```
ggplot(df_esoph, aes(x= agegp, y=ncases, fill=tobgp))+geom_col()
```

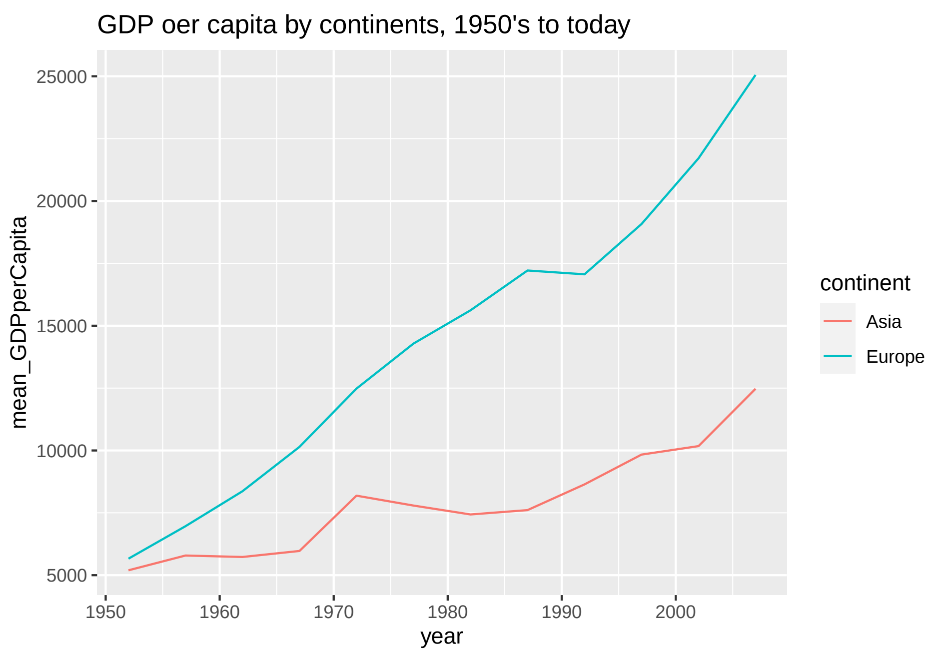


Default position is "stack".

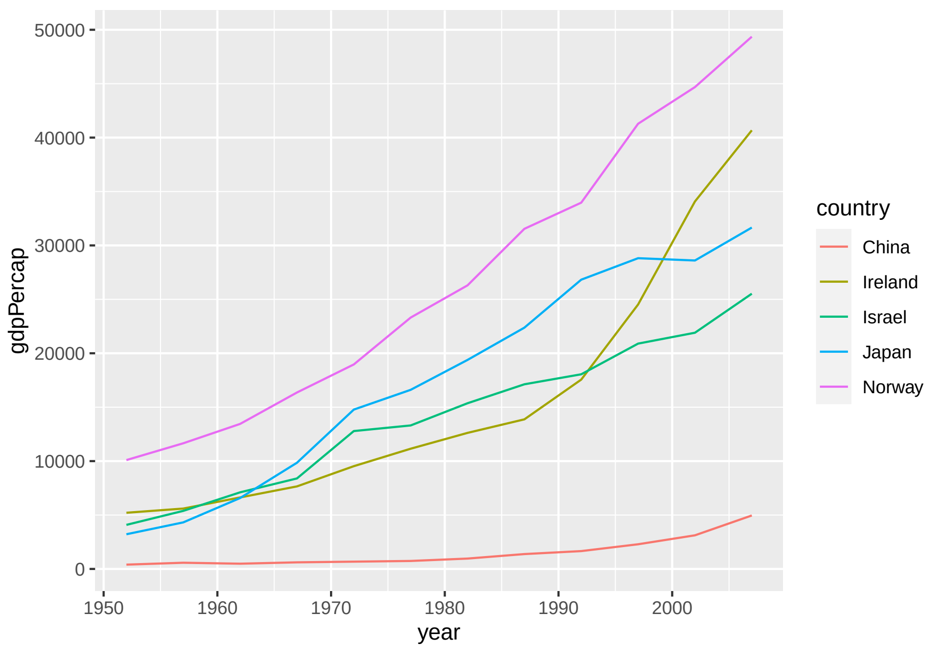
```
ggplot(df_esoph, aes(x= agegp, y=ncases, fill=tobgp))+geom_col(position = "dodge")
```



10.2 Q2. Combine two charts




```
df %>%
  select(country, year, gdpPercap) %>%
  filter(country %in% c("Israel", "Japan", "Norway", "China", "Ireland")) %>%
  ggplot(aes(x=year)) +
  geom_line(aes(y=gdpPercap, color=country))
```

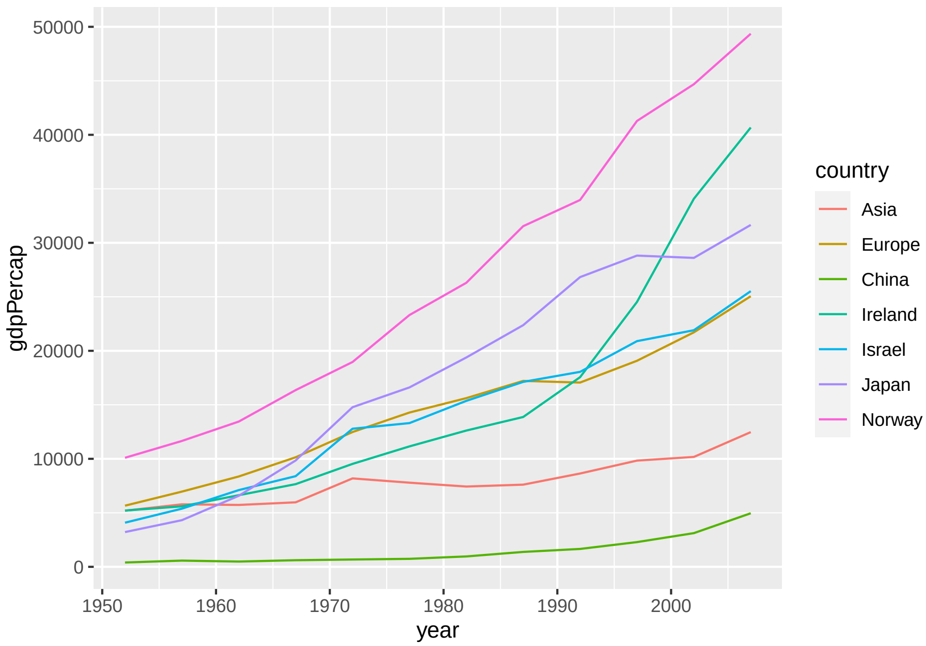


Question. I have not managed to add on the same graph of the continents the data for the individual countries, as I would have liked:

Solution. Construct two data sets and combine them into one.

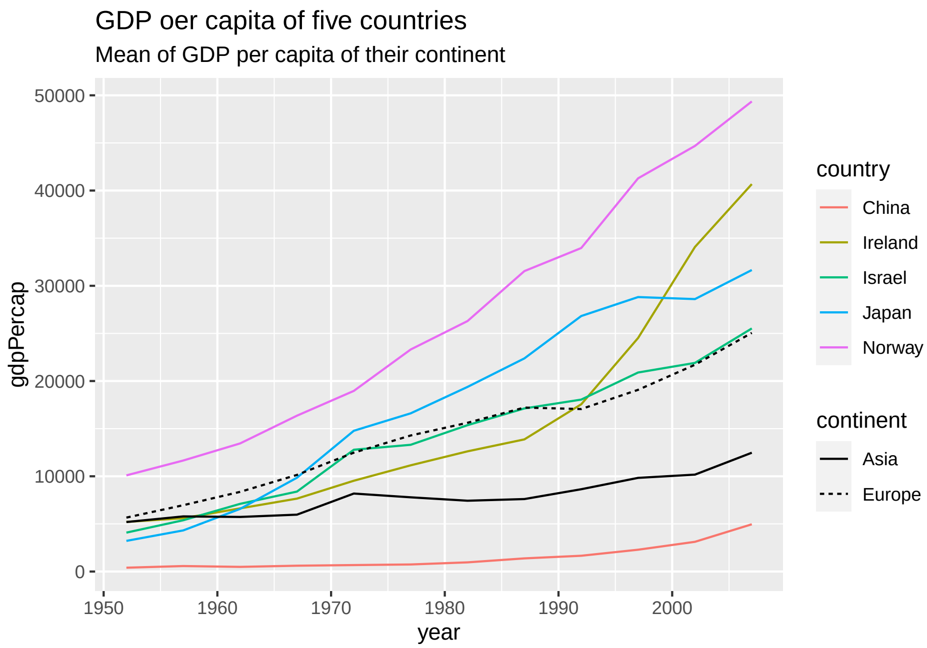
`ggplot2` starts with one data.

```
df_2c <- df %>%
  select(continent, year, gdpPercap) %>%
  filter(continent %in% c("Asia", "Europe")) %>%
  group_by(continent, year) %>%
  summarise(gdpPercap = mean(gdpPercap), .groups = 'drop') %>%
  select(country = continent, year, gdpPercap)

df_5c <- df %>%
  select(country, year, gdpPercap) %>%
  filter(country %in% c("Israel", "Japan", "Norway", "China", "Ireland"))

df_2c %>% bind_rows(df_5c) %>%
  ggplot(aes(x = year, y = gdpPercap, color = country)) + geom_line()
```



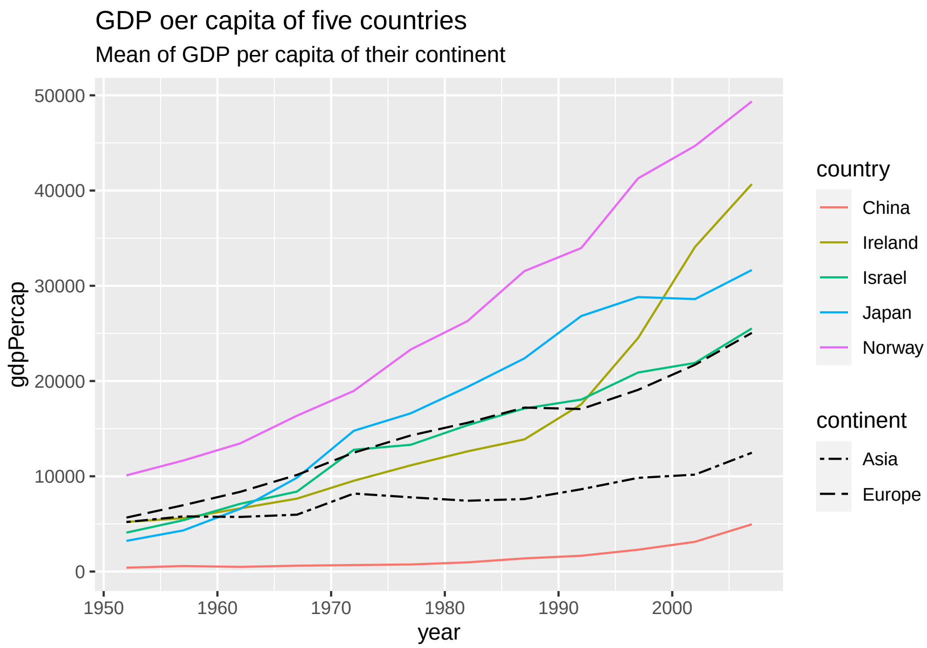
Use `mutate`.

```
df %>%
  group_by(continent, year) %>%
  mutate(mean_by_continent = mean(gdpPercap)) %>%
  ungroup() %>%
  filter(country %in% c("Israel", "Japan", "Norway", "China", "Ireland")) %>%
  ggplot(aes(x = year)) +
    geom_line(aes(y = gdpPercap, color=country)) +
    geom_line(aes(y = mean_by_continent, linetype=continent)) +
    labs(title = "GDP oer capita of five countries", subtitle = "Mean of GDP per capita of their continent")
```



When you want to change the linetype manually, use `scale_linetype_manual()`.

```
df %>%
  group_by(continent, year) %>%
  mutate(mean_by_continent = mean(gdpPercap)) %>%
  ungroup() %>%
  filter(country %in% c("Israel", "Japan", "Norway", "China", "Ireland")) %>%
  ggplot(aes(x = year)) +
    geom_line(aes(y = gdpPercap, color=country)) +
    geom_line(aes(y = mean_by_continent, linetype=continent)) +
    scale_linetype_manual(values = c("Asia" = "twodash", "Europe" = "longdash")) +
    labs(title = "GDP oer capita of five countries", subtitle = "Mean of GDP per capita of their continent")
```

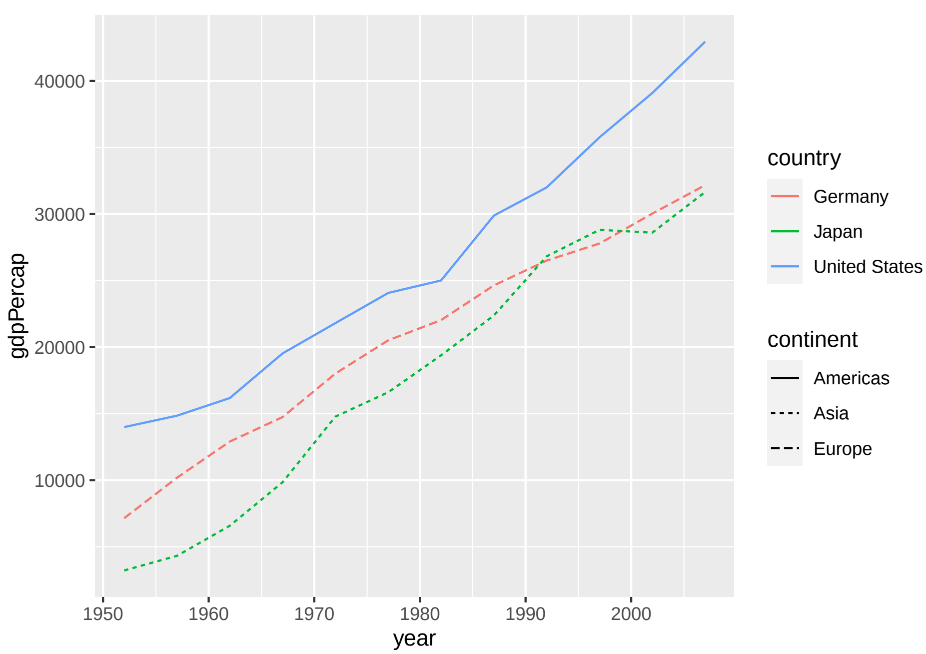

GDP oer capita of five countries
Mean of GDP per capita of their continent
gdpPercap
50000
40000
30000
20000
10000
0
1950
1960
1970
1980
1990
2000
year
country
China
Ireland
Israel
Japan
Norway
continent
Asia
Europe

Chapter 11

Appendix: Change colors, shapes, linetypes, etc. manually

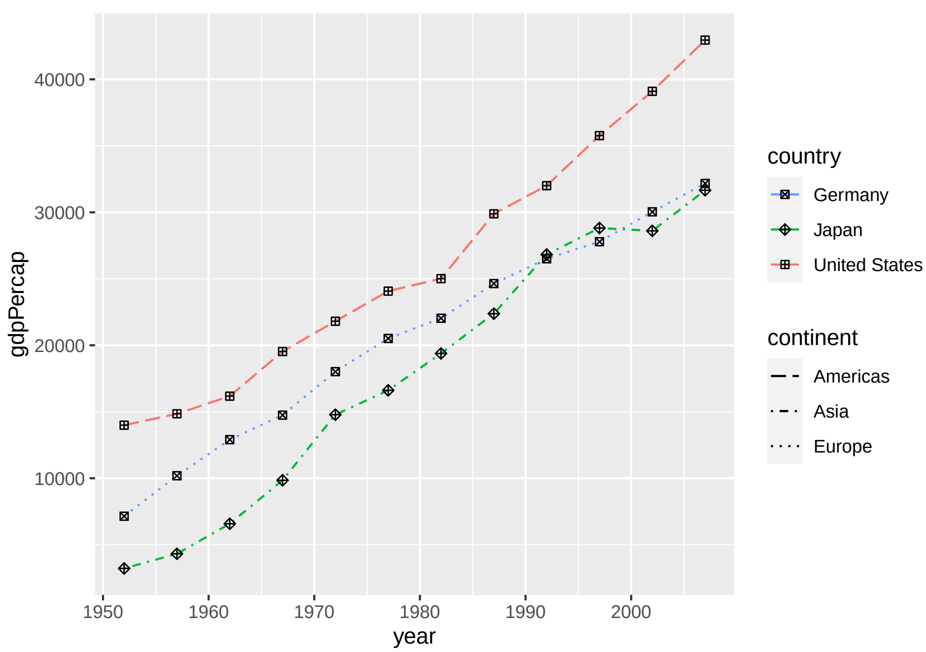
Example: Default

```
df %>%
  filter(country %in% c("Germany", "Japan", "United States")) %>%
  ggplot() +
    geom_line(aes(x = year, y = gdpPercap, color=country, linetype=continent))
```



- `scale_color_manual`: https://ggplot2-book.org/scale-colour.html
  - eg1: scale_colour_manual(values = c("red", "blue", "green"))
  - eg2: scale_colour_manual(values = c("China" = "red", "Japan" = "blue", "Norway" = "green"))
  - eg3: scale_colour_manual(values = scales::hue_pal()(3)) # default
  - eg4: scale_colour_manual(values = scales::hue_pal(direction = -1)(3)) # reverse order
- `scale_fill_manual`: similar to `scale_color_manual`
- `scale_linetype_manual`: https://ggplot2-book.org/scale-other.html?q=linetype#scale-linetype
- 'scale_shape_manual: https://ggplot2-book.org/scale-other.html?q=scale_shape_manual#scale-shape
- 'scale_size: https://ggplot2-book.org/scale-other.html?q=size#scale-size

```
df %>%
  filter(country %in% c("Germany", "Japan", "United States")) %>%
  ggplot(aes(x = year, y = gdpPercap)) +
    geom_line(aes(color=country, linetype=continent)) +
    geom_point(aes(shape = country)) +
    scale_colour_manual(values = scales::hue_pal(direction = -1)(3)) +
    scale_linetype_manual(values = c("Europe" = "dotted", "Asia" = "dotdash", "Americas" = "longdash")) +
    scale_shape_manual(values = c("Germany" = 7, "Japan" = 9, "United States" = 12))
```



Chapter 12

Importing Public Data, WDI

12.1 Reviews and Previews

```
library(tidyverse)
library(gapminder)
library(maps)
library(WDI)
library(readxl)
library(ggrepel)
```

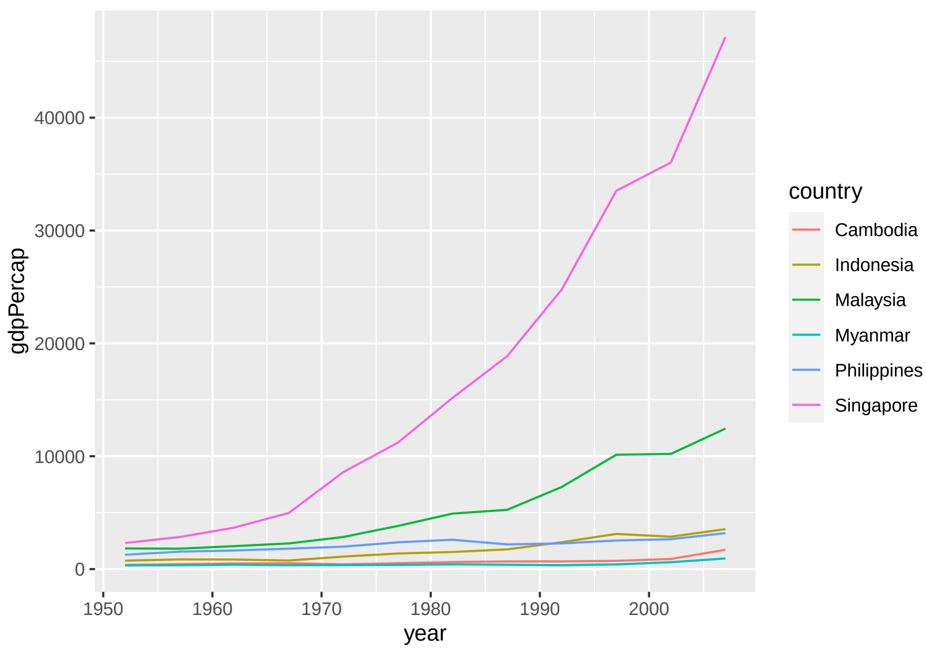
- We have used `tidyverse` and `gapminder` already.
- If you have not installed `WDI`, install it.
- We will not use `ggrepel` but if you want to use it, install it.
- `maps` and `readxl` are bundled in `tidyverse` but need to be attached by `library`.

12.1.1 Gapminder Package Data

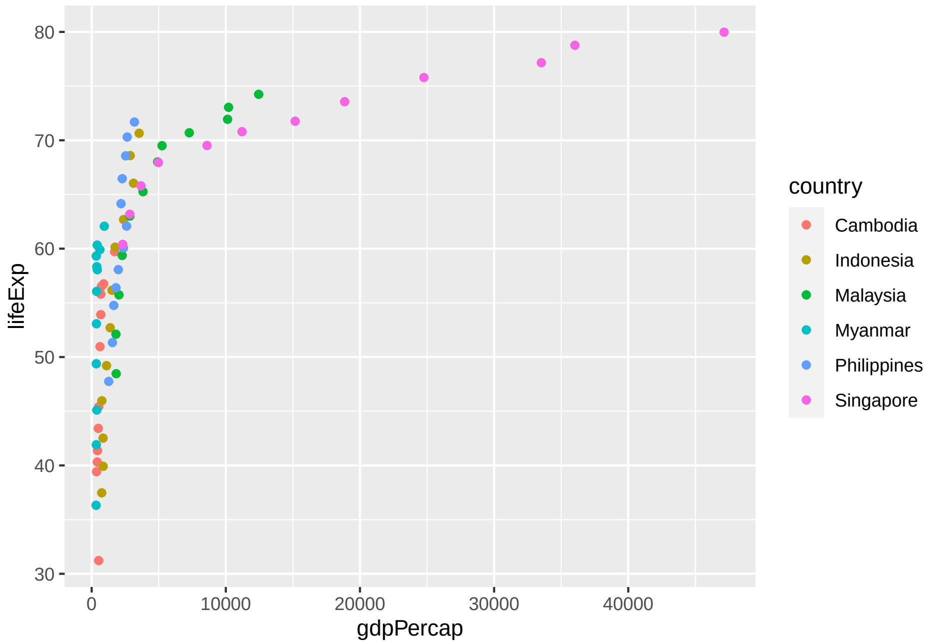
```
df <- gapminder
df
#> # A tibble: 1,704 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
#> # ... with 1,694 more rows
```

```r
asean <- c("Brunei", "Cambodia", "Laos", "Myanmar",
           "Philippines", "Indonesia", "Malaysia", "Singapore")
df %>% filter(country %in% asean) %>%
  ggplot(aes(x = year, y = gdpPercap, col = country)) + geom_line()
```

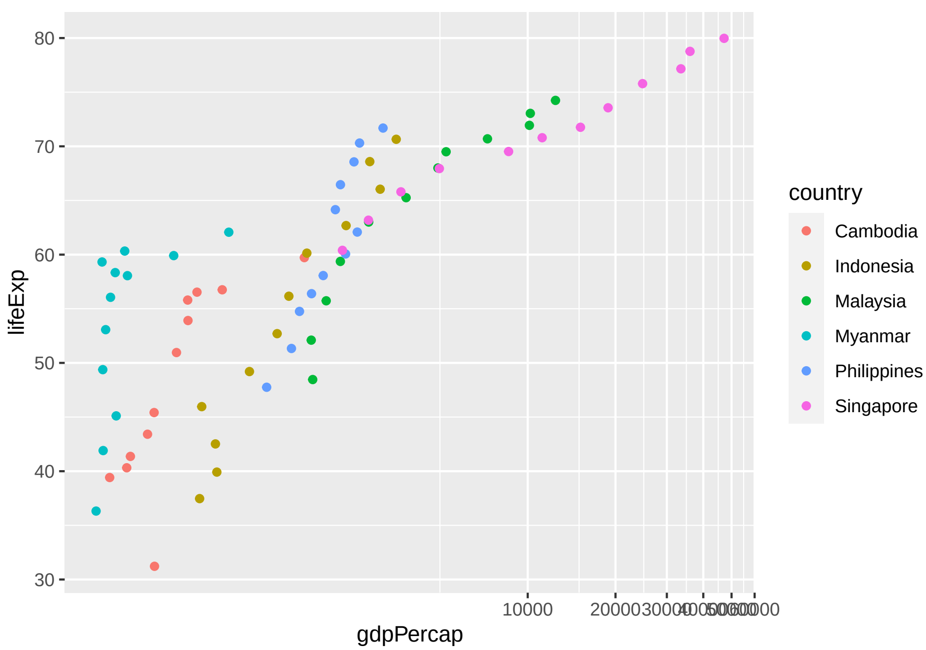
12.1.2 gdpPercap of ASEAN countries




```r
df %>% filter(country %in% asean) %>%
  ggplot(aes(x = gdpPercap, y = lifeExp, col = country)) + geom_point()
```

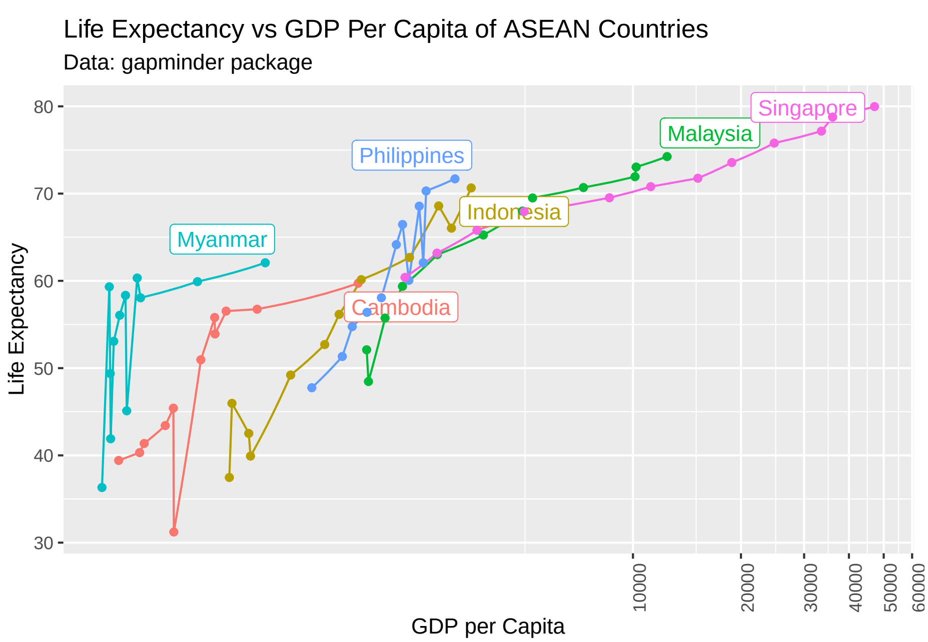


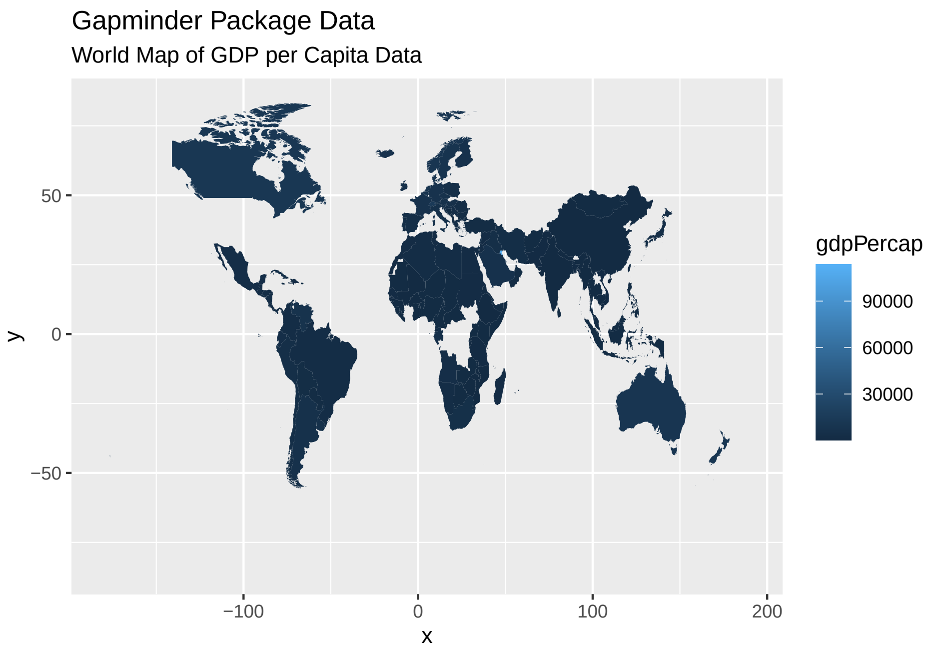
```
df %>% filter(country %in% asean) %>%
  ggplot(aes(x = gdpPercap, y = lifeExp, col = country)) +
  geom_point() + coord_trans(x = "log10", y = "identity")
```



$\log_{10} 100 = 2$, $\log_{10} 1000 = 3$, $\log_{10} 10000 = 4$

```
library(ggrepel)
df2007 <- df %>% filter(country %in% asean, year == 2007)
df %>% filter(country %in% asean) %>%
  ggplot(aes(x = gdpPercap, y = lifeExp, col = country))+
  geom_line() + geom_label_repel(data = df2007, aes(label = country)) + geom_point()  +
  coord_trans(x = "log10", y = "identity") +
  theme(axis.text.x = element_text(angle = 90, vjust = 1, hjust=1), legend.position = "none") +
  labs(title = "Life Expectancy vs GDP Per Capita of ASEAN Countries",
       subtitle = "Data: gapminder package", x = "GDP per Capita", y = "Life Expectancy")
```





12.1.3 World Bank: World Development Indicators (WDI)

- SP.DYN.LE00.IN: Life expectancy at birth, total (years)
- NY.GDP.PCAP.KD: GDP per capita (constant 2015 US$)
- SP.POP.TOTL: Population, total

```
df_wdi <- WDI(
  country = "all",
  indicator = c(lifeExp = "SP.DYN.LE00.IN", pop = "SP.POP.TOTL", gdpPercap = "NY.GDP.PCAP.KD")
)
```

```
df_wdi
#> # A tibble: 16,492 x 7
#>    country     iso2c iso3c  year lifeExp      pop gdpPercap
#>    <chr>       <chr> <chr> <dbl>   <dbl>    <dbl>     <dbl>
#>  1 Afghanistan AF    AFG    1960    32.5  8622466        NA
#>  2 Afghanistan AF    AFG    1961    33.1  8790140        NA
#>  3 Afghanistan AF    AFG    1962    33.5  8969047        NA
#>  4 Afghanistan AF    AFG    1963    34.0  9157465        NA
#>  5 Afghanistan AF    AFG    1964    34.5  9355514        NA
#>  6 Afghanistan AF    AFG    1965    35.0  9565147        NA
#>  7 Afghanistan AF    AFG    1966    35.5  9783147        NA
#>  8 Afghanistan AF    AFG    1967    35.9 10010030        NA
#>  9 Afghanistan AF    AFG    1968    36.4 10247780        NA
#> 10 Afghanistan AF    AFG    1969    36.9 10494489        NA
#> # ... with 16,482 more rows
```

```
df_wdi_extra <- WDI(
  country = "all",
  indicator = c(lifeExp = "SP.DYN.LE00.IN", pop = "SP.POP.TOTL", gdpPercap = "NY.GDP.PCAP.KD"),
```

```
  extra = TRUE
)
```

```
df_wdi_extra
#> # A tibble: 16,492 x 15
#>    country     iso2c iso3c  year status lastupdated lifeExp
#>    <chr>       <chr> <chr> <dbl> <lgl>  <date>        <dbl>
#>  1 Afghanistan AF    AFG    1993 NA     2022-12-22     51.5
#>  2 Afghanistan AF    AFG    1997 NA     2022-12-22     53.6
#>  3 Afghanistan AF    AFG    1994 NA     2022-12-22     51.5
#>  4 Afghanistan AF    AFG    1995 NA     2022-12-22     52.5
#>  5 Afghanistan AF    AFG    2001 NA     2022-12-22     55.8
#>  6 Afghanistan AF    AFG    1998 NA     2022-12-22     52.9
#>  7 Afghanistan AF    AFG    1999 NA     2022-12-22     54.8
#>  8 Afghanistan AF    AFG    2007 NA     2022-12-22     59.1
#>  9 Afghanistan AF    AFG    2008 NA     2022-12-22     59.9
#> 10 Afghanistan AF    AFG    1980 NA     2022-12-22     39.6
#> # ... with 16,482 more rows, and 8 more variables:
#> #   pop <dbl>, gdpPercap <dbl>, region <chr>,
#> #   capital <chr>, longitude <dbl>, latitude <dbl>,
#> #   income <chr>, lending <chr>
```

12.2 Exploratory Data Analysis

12.2.1 What is EDA (Posit Primers: Visualise Data)

1. EDA is an iterative cycle that helps you understand what your data says. When you do EDA, you:

2. Generate questions about your data

3. Search for answers by visualising, transforming, and/or modeling your data

Use what you learn to refine your questions and/or generate new questions

EDA is an important part of any data analysis. You can use EDA to make discoveries about the world; or you can use EDA to ensure the quality of your data, asking questions about whether the data meets your standards or not.

12.3 Open and Public Data, World Bank

12.3.1 Open Government Data Toolkit: Open Data Defined

The term **Open Data** has a very precise meaning. Data or content is open if anyone is free to use, re-use or redistribute it, subject at most to measures that preserve provenance and openness.

1. The data must be *legally open*, which means they must be placed in the public domain or under liberal terms of use with minimal restrictions.
2. The data must be *technically open*, which means they must be published in electronic formats that are machine readable and non-proprietary, so that anyone can access and use the data using common, freely available software tools. Data must also be publicly available and accessible on a public server, without password or firewall restrictions. To make Open Data easier to find, most organizations create and manage Open Data catalogs.

12.4 World Bank: WDI - World Development Indicaters

- World Bank: https://www.worldbank.org
- Who we are:
  - To end extreme poverty: By reducing the share of the global population that lives in extreme poverty to 3 percent by 2030.
  - To promote shared prosperity: By increasing the incomes of the poorest 40 percent of people in every country.
- World Bank Open Data: https://data.worldbank.org
  - Data Bank, World Development Indicators, etc.
- World Development Indicators (WDI) : the World Bank's premier compilation of cross-country comparable data on development; 1400 time series indicators
  - Themes: Poverty and Inequality, People, Environment, Economy, States and Markets, Global Links
  - Open Data & DataBank: Explore data, Query database
  - Bulk Download: Excel, CSV
  - API Documentation

12.5 R Package WDI

- WDI: World Development Indicators and Other World Bank Data
- Search and download data from over 40 databases hosted by the World Bank, including the World Development Indicators ('WDI'), International Debt Statistics, Doing Business, Human Capital Index, and Sub-national Poverty indicators.
- Version: 2.7.4
- Materials: README - *usage*
  - NEWS - *version history*
- Published: 2021-04-06
- README: https://cran.r-project.org/web/packages/WDI/readme/README.html
- Reference manual: WDI.pdf

12.6 Function WDI

- **Usage**

```
WDI(country = "all",
    indicator = "NY.GDP.PCAP.KD",
    start = 1960,
    end = 2020,
    extra = FALSE,
    cache = NULL)
```

- **Arguments** See Help!
  - country: Vector of countries (ISO-2 character codes, e.g. "BR", "US", "CA", or "all")
  - indicator: If you supply a named vector, the indicators will be automatically renamed: `c('women_private_sector' = 'BI.PWK.PRVS.FE.ZS')`

12.7 Function WDIsearch

```
library(WDI)
```

```
WDIsearch(string = "NY.GDP.PCAP.KD",
          field = "indicator", cache = NULL)
#>                 indicator                                 name
#> 11431    NY.GDP.PCAP.KD GDP per capita (constant 2015 US$)
#> 11432 NY.GDP.PCAP.KD.ZG   GDP per capita growth (annual %)
```