Practica 2 - Visualizacion de datos

Carlos Santiago Martinez Torres

2025-05-24

Contents

Configuracion inicial

```
# Instalacion de librerias con pacman

# Asegurarse de que el paquete 'pacman' está instalado
if (!require("pacman")) install.packages("pacman")

# Poner las librerias necesarias
pacman::p_load(tidyverse, knitr, datasets, RColorBrewer)

# Con tidyverse: ggplot2, dplyr, tidyr, readr, purrr, tibble, stringr, forcats
```

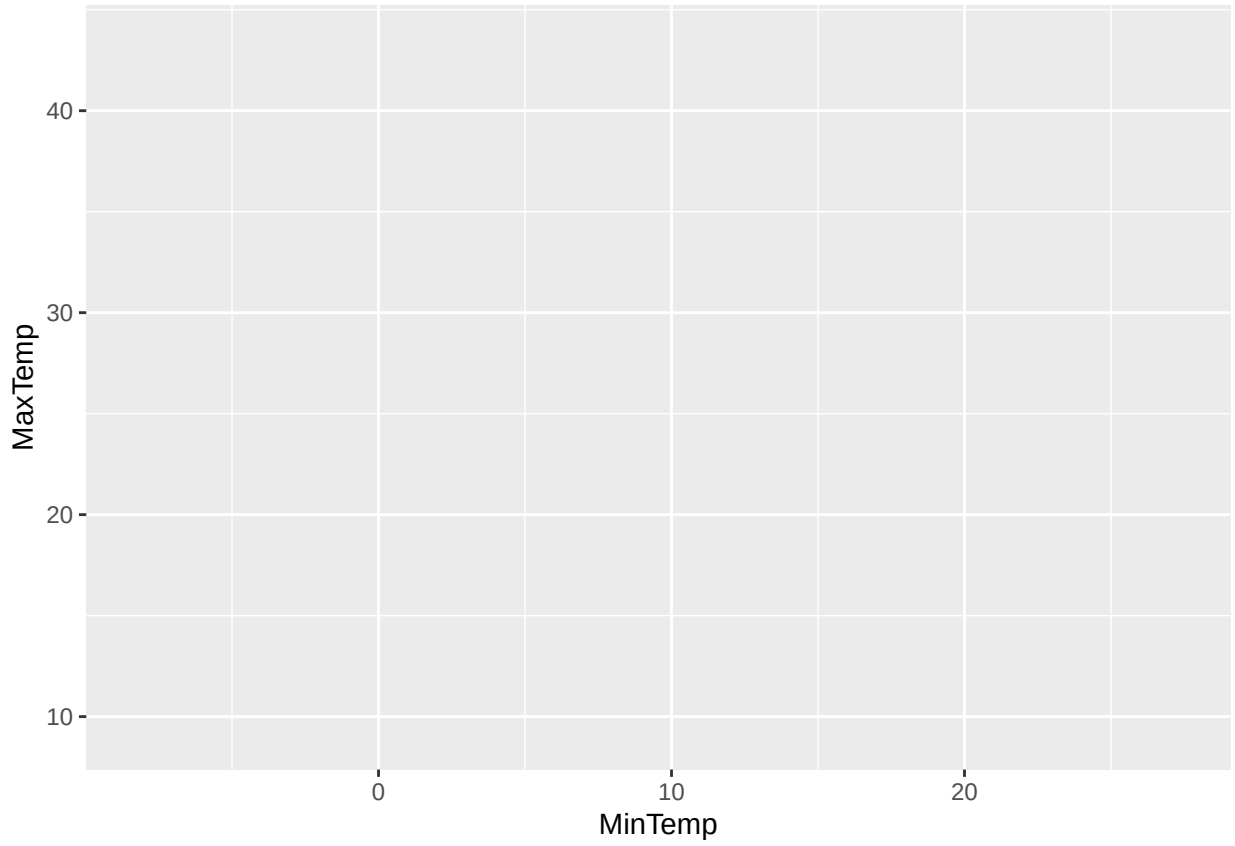
Ejercicio 1

- Carga el fichero `weatherAUS.RData` utilizando lo aprendido en la práctica 1.

```r
# Directorio que contiene los ficheros
carpeta <- 'data/'

f1 <- 'weatherAUS.RData'
f1 <- paste0(carpeta, f1)

load(file = f1)
```

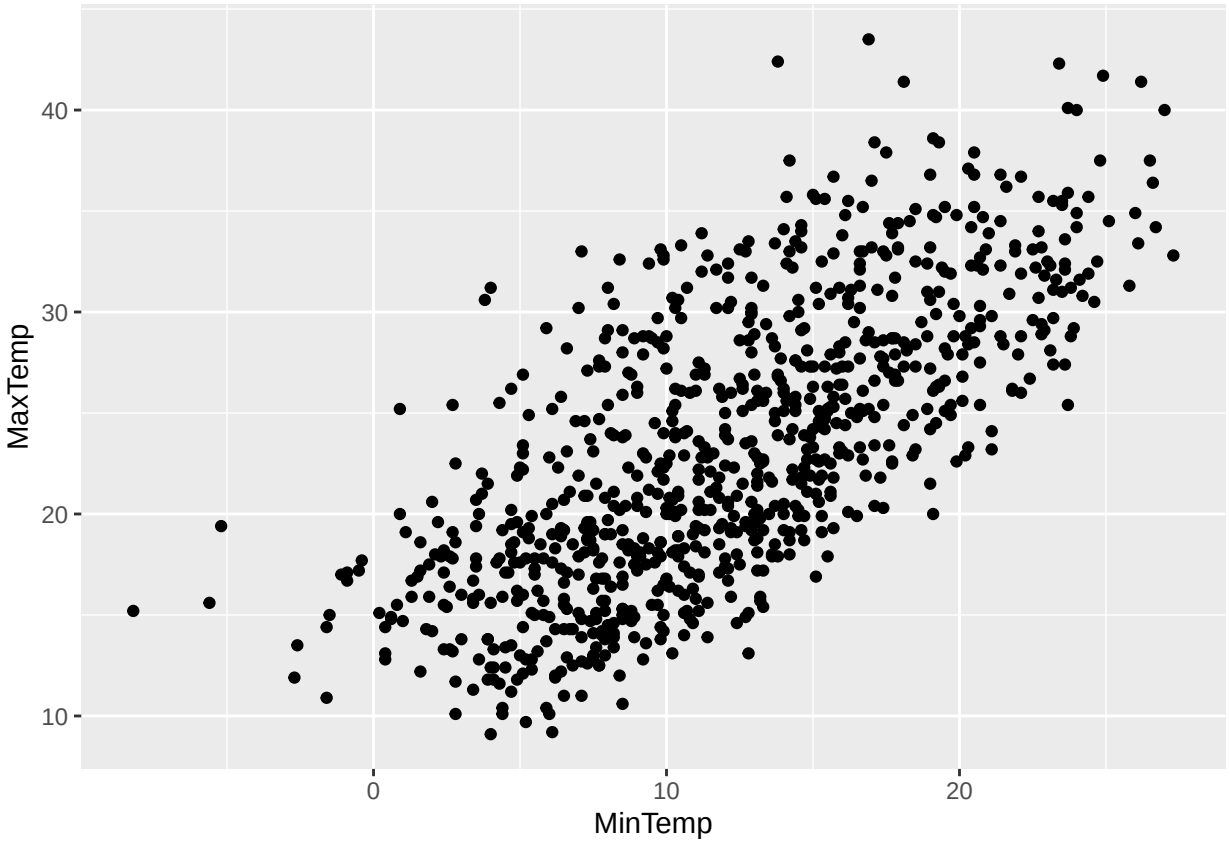
- Crea un objeto ggplot denominado `grafico1` en el que esté definido el conjunto de datos `ds` alojado en el fichero `weatherAUS.RData` y un *aesthetics* en el que en el eje x esté definida la variable `MinTemp` y en el eje y esté definida la variable `MaxTemp`.

```r
grafico1 <- ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp))
grafico1
```



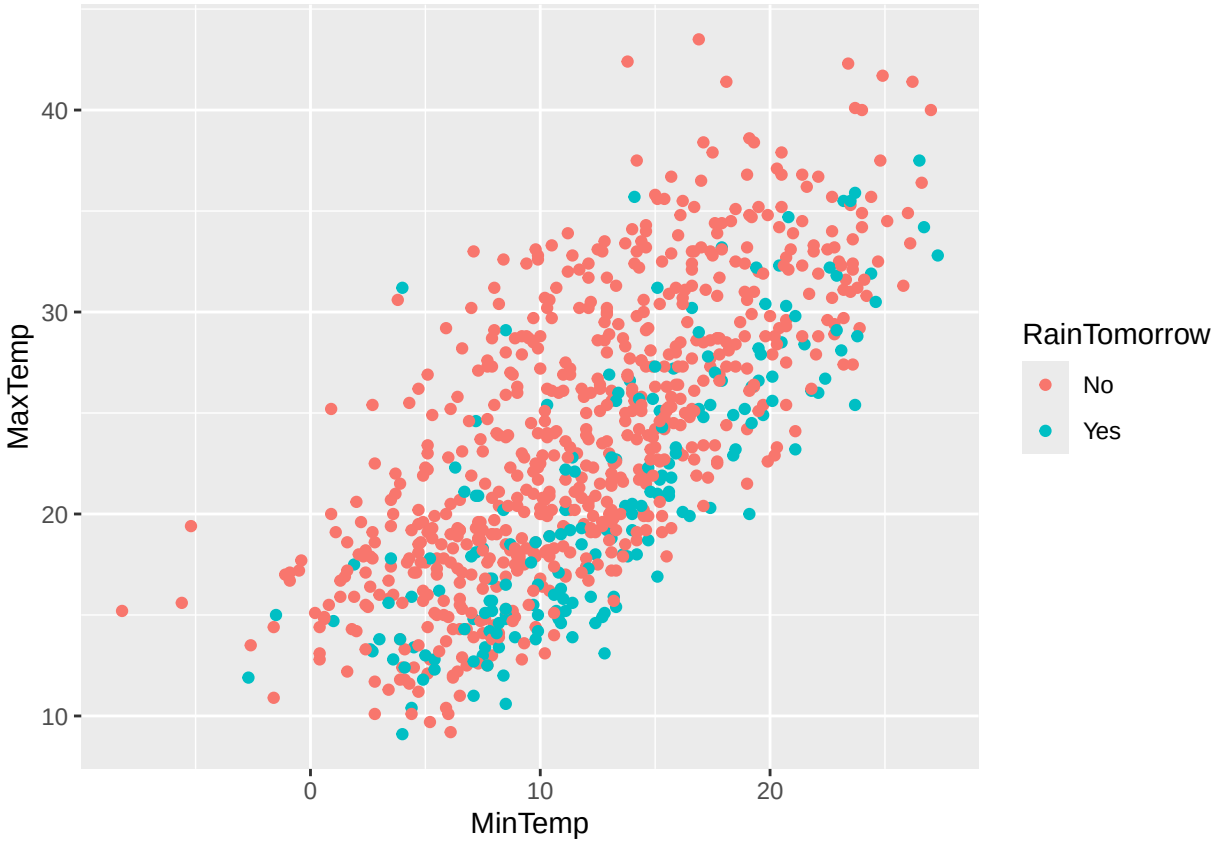
- Añade una capa que defina el objeto geométrico necesario para construir un gráfico de dispersión de las variables `MinTemp` y `MaxTemp`.

```r
grafico1 <- grafico1 + geom_point()
grafico1
```

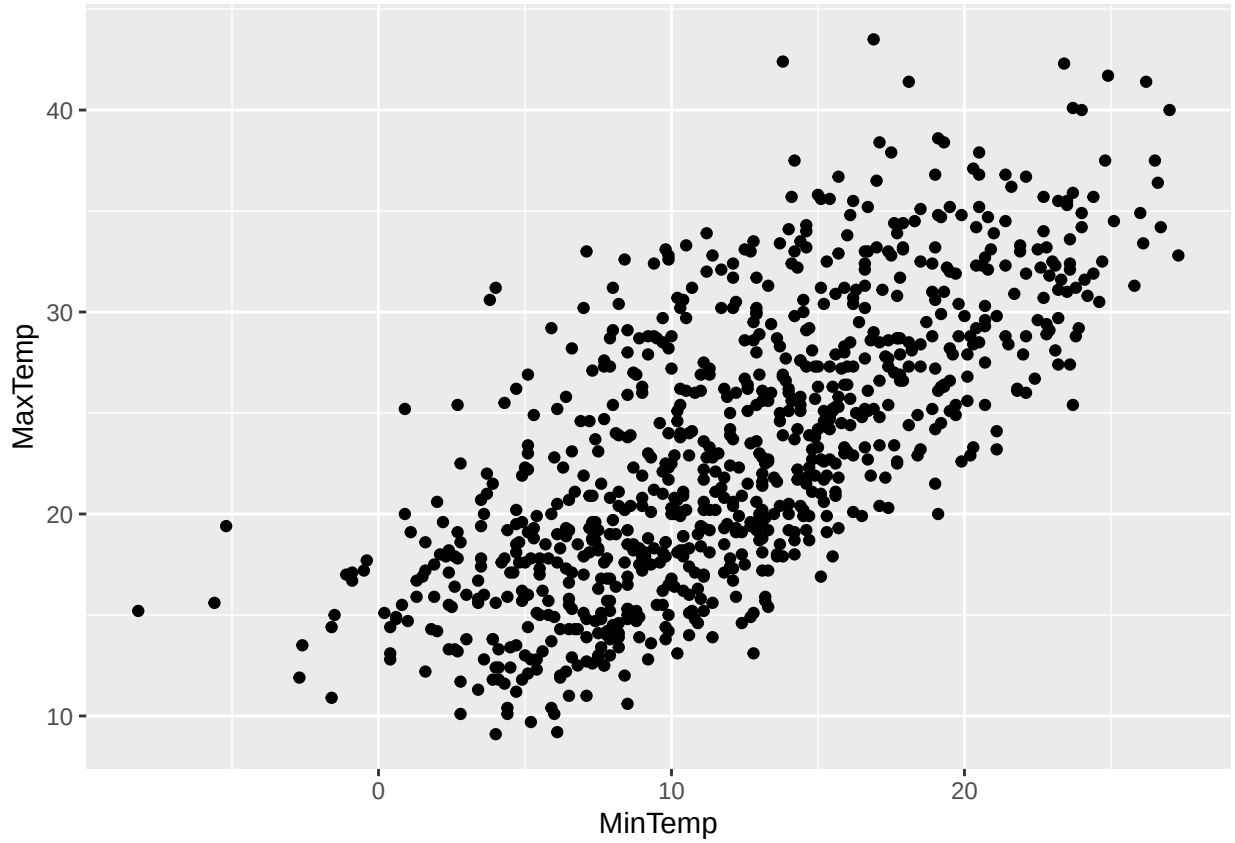


- Genera ahora el mismo gráfico y almacénalo en `grafico2` pero define el *aesthetics* para visualizar las variables `MinTemp` y `MaxTemp` en los ejes de la gráfica y la variable categórica `RainTomorrow` como el color. A partir del objeto `grafico2`, representa un gráfico de dispersión en el que se muestren los puntos con el color del punto correspondiente a la variable `RainTomorrow`.

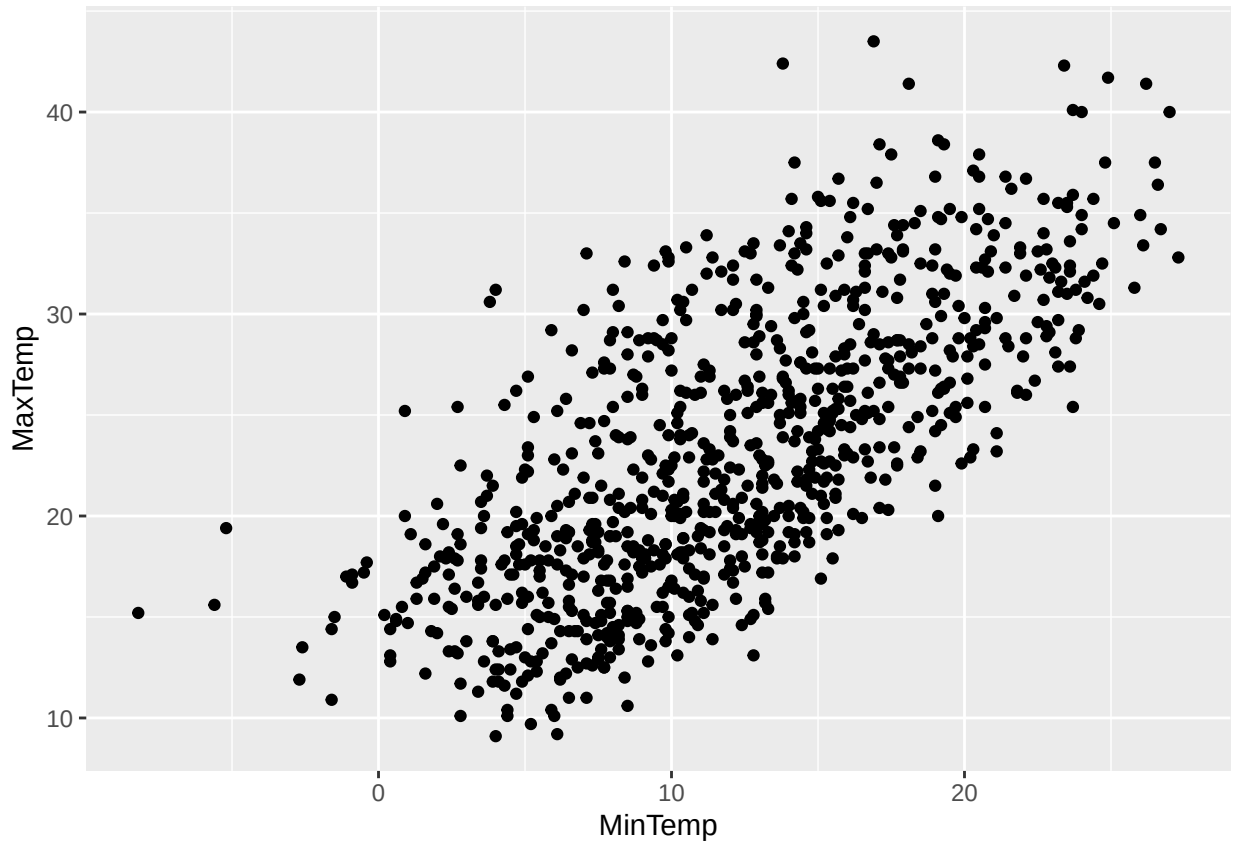
```
# Las lineas comentadas generan el mismo grafico

# ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp)) + geom_point(mapping = aes(color = RainTom
ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp, color = RainTomorrow)) + geom_point()
```



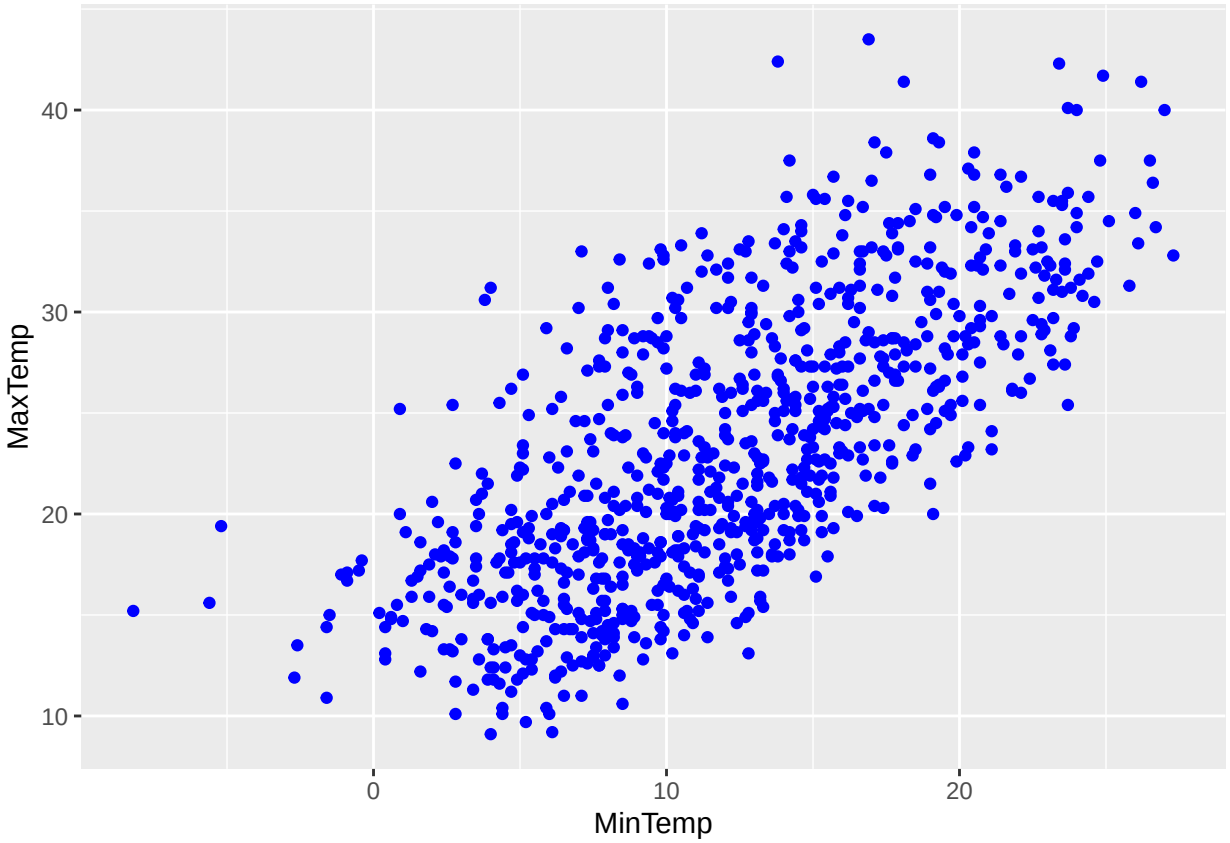

```
# Este elemento es diferente, porque color (aesthetic) no está asignado a una variable
ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp), color = RainTomorrow) + geom_point()
```



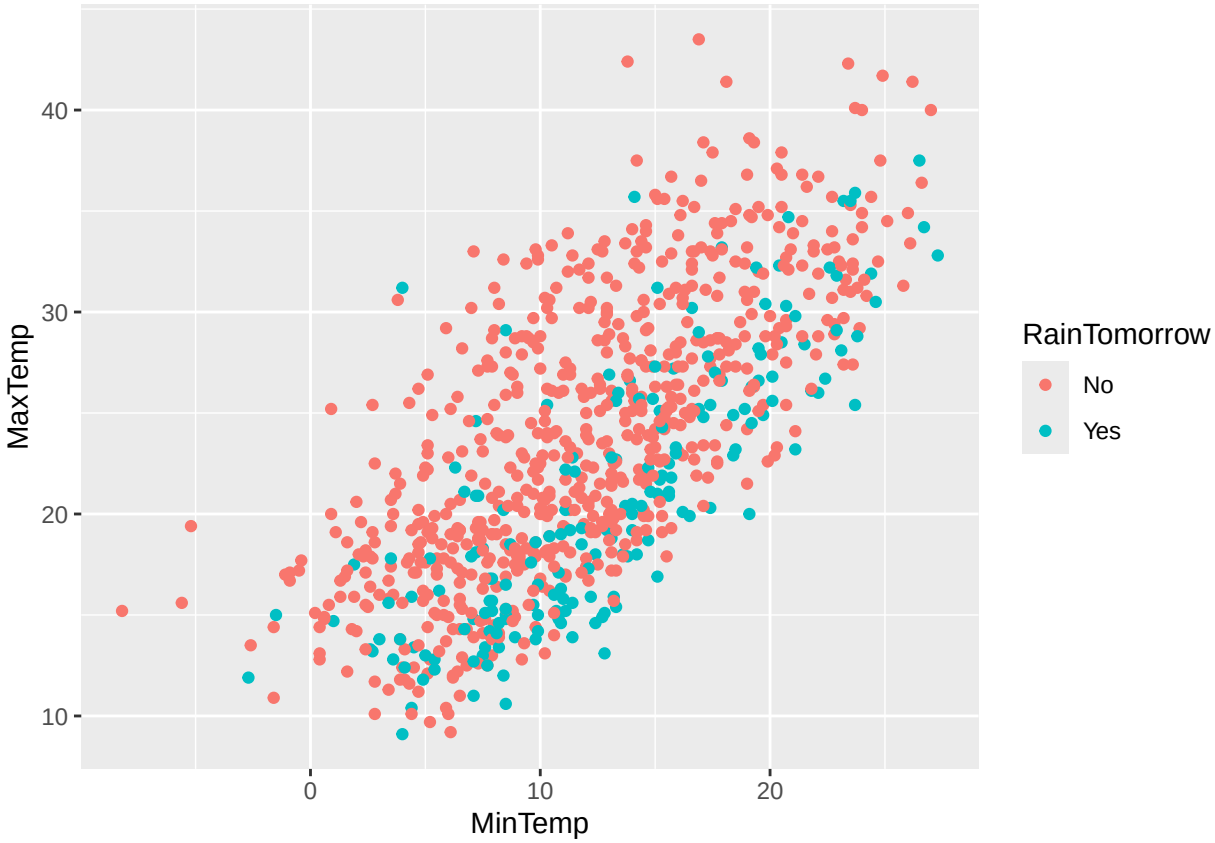

```
# No tiene a quien asignar el color azul
ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp), color = 'blue') + geom_point()
```




```
# El color queda 'estatico'
ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp), color = RainTomorrow) + geom_point(color = '
```

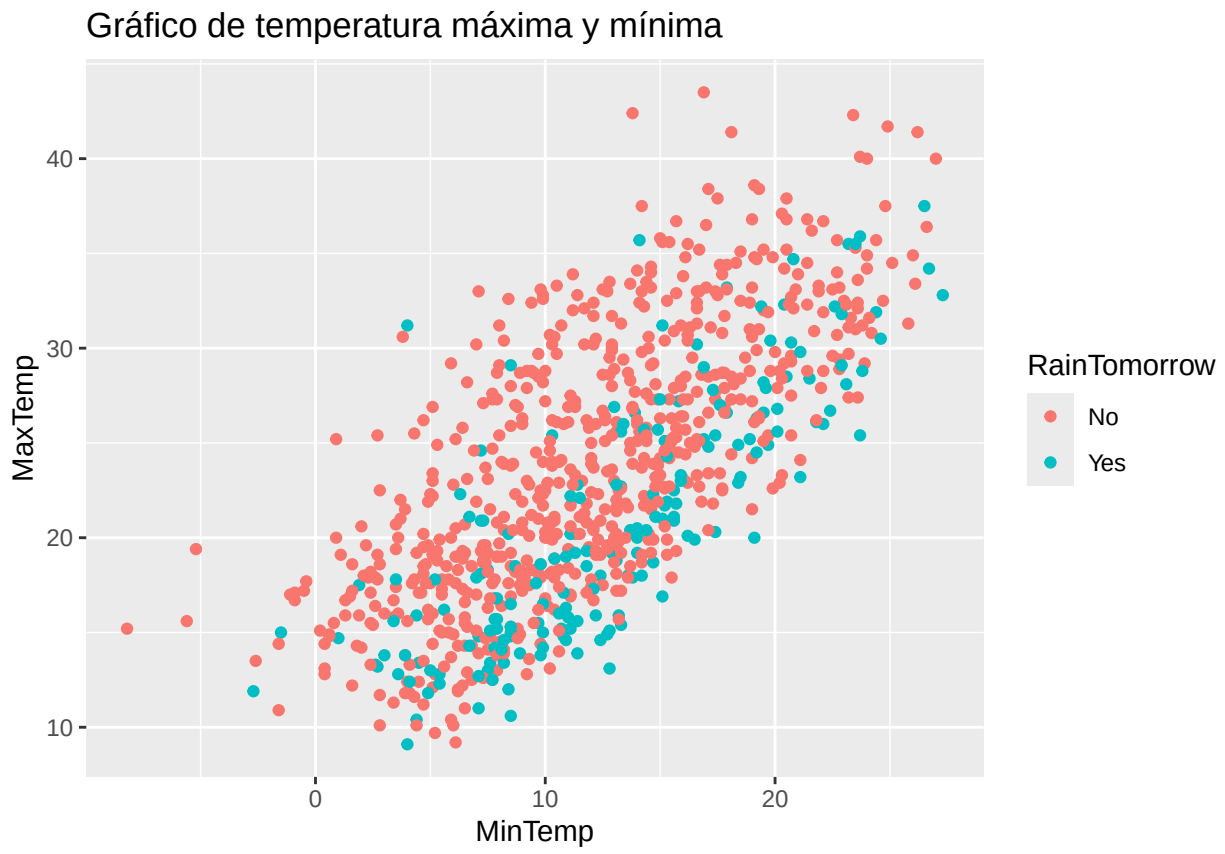



```
grafico2 <- ggplot(data = ds, mapping = aes(x = MinTemp, y = MaxTemp, color = RainTomorrow)) + geom_poi
grafico2
```



- Añade a `grafico2` una capa para incluir un título con **ggtitle()** y almacénalo en '`grafico3`.

```
grafico3 <- grafico2 + ggtitle(label = 'Gráfico de temperatura máxima y mínima')
grafico3
```



- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio1.pdf`.

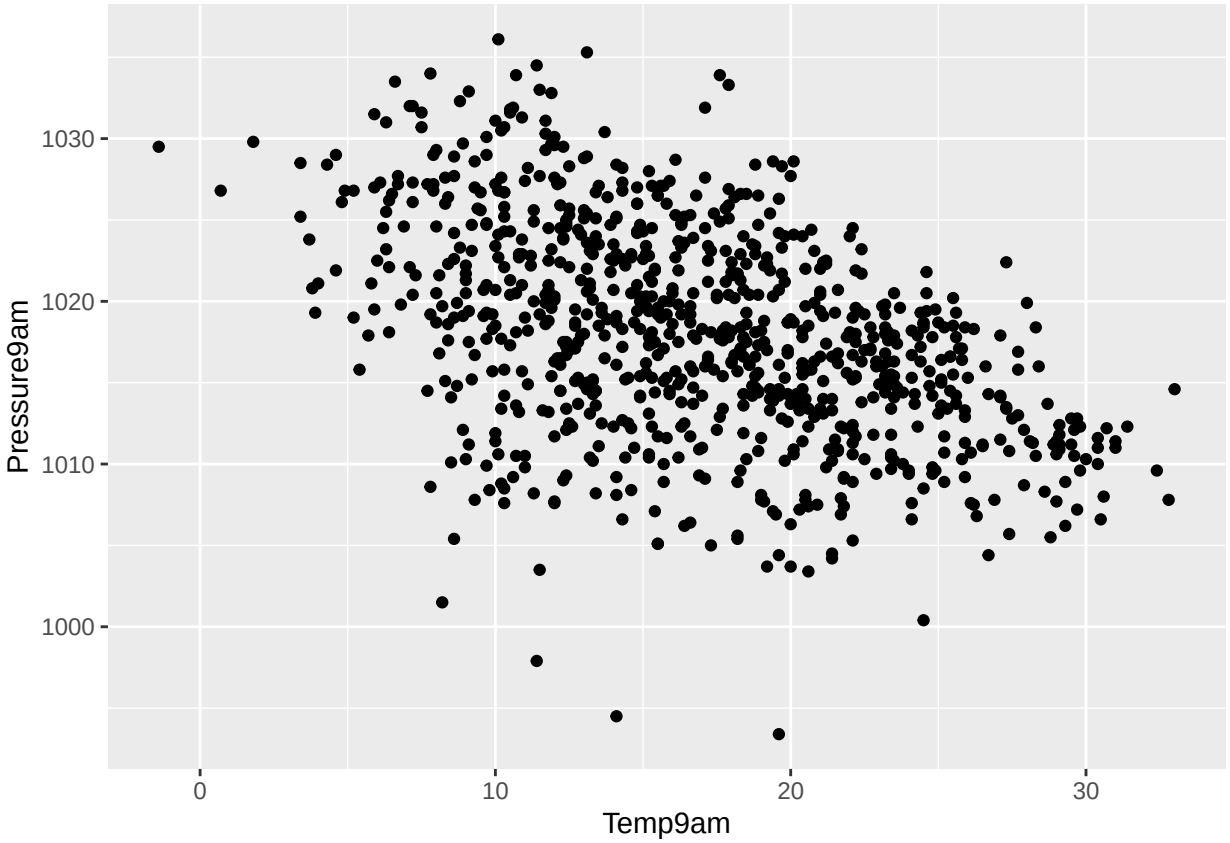
```
# Directorio que contiene las figuras
figuras <- 'figure/'
n_salida <- '1'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
grafico3
dev.off()
```

```
## pdf
##   2
```

Ejercicio 2

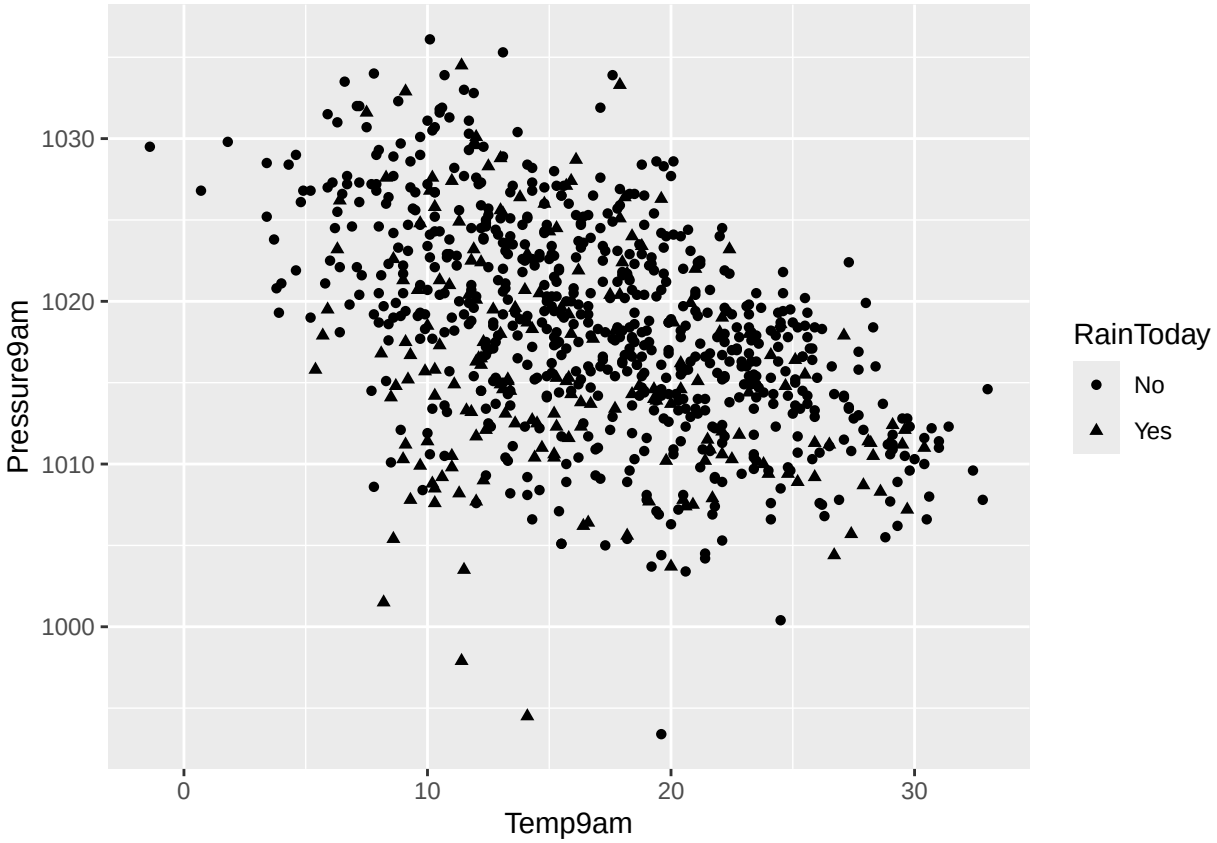
- Utilizando ggplot haz un grafico de dispersión en el que representes las variables `Temp9am` y `Pressure9am` en los ejes x e y respectivamente.

```
ggplot(data = ds, mapping = aes(x = Temp9am, y = Pressure9am)) + geom_point()
```



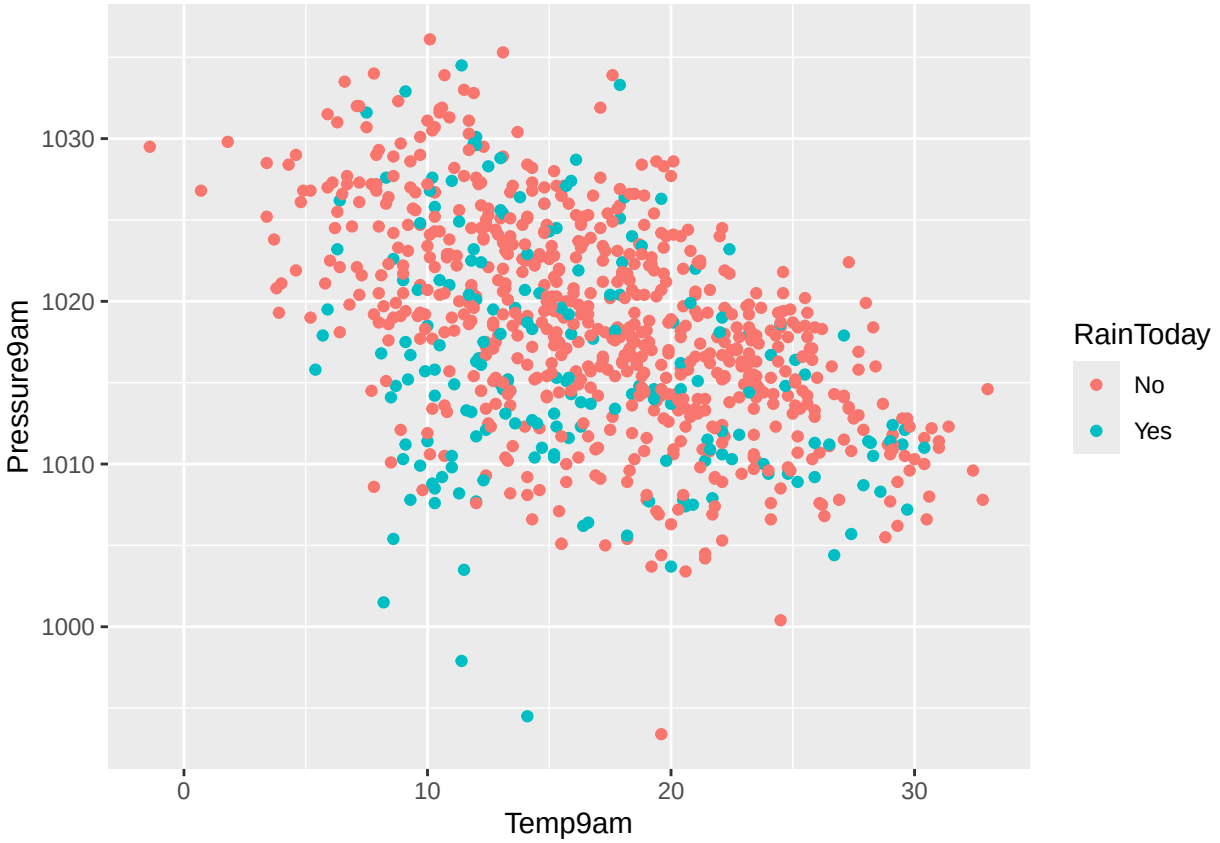
- Modifica el gráfico anterior para incluir información categórica proporcionada por la variable `RainToday` con la forma del punto.

```
# ggplot(data = ds, mapping = aes(x = Temp9am, y = Pressure9am, shape = RainToday)) + geom_point()
ggplot(data = ds, mapping = aes(x = Temp9am, y = Pressure9am)) + geom_point(aes(shape = RainToday))
```



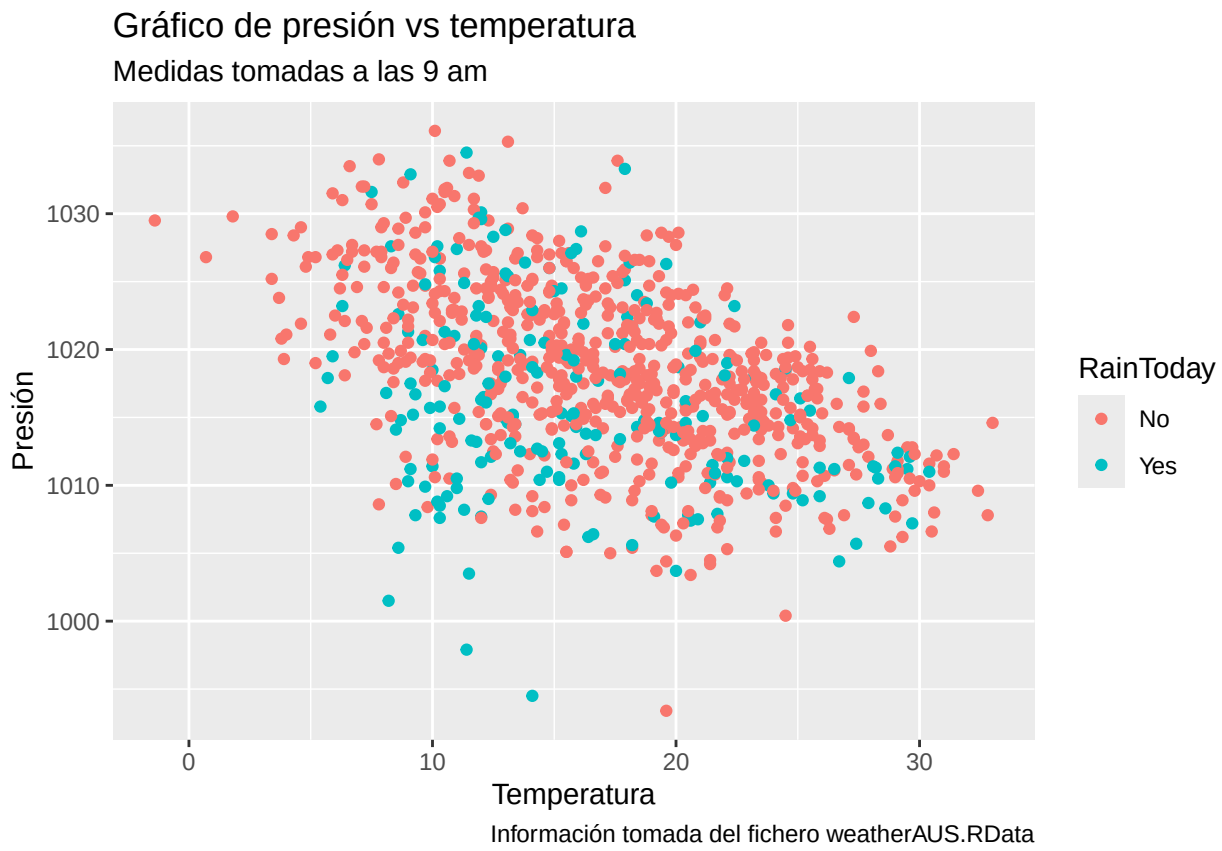
- Modifica el gráfico anterior para incluir información categórica proporcionada por la variable `RainToday` con el color del punto. ¿Con qué representación (color o forma) se observa mejor la información que introduce la variable `RainToday`?

```
ggplot(data = ds, mapping = aes(x = Temp9am, y = Pressure9am)) + geom_point(aes(color = RainToday))
```



- Añade títulos explicativos al gráfico anterior usando la función (debe aparece un título, subtítulo, leyendas e información de las variables representadas en los ejes).

```
g2 <- ggplot(data = ds, mapping = aes(x = Temp9am, y = Pressure9am)) + geom_point(aes(color = RainToday
  labs(title = 'Gráfico de presión vs temperatura',
       subtitle = 'Medidas tomadas a las 9 am',
       x = 'Temperatura',
       y = 'Presión',
       caption = 'Información tomada del fichero weatherAUS.RData')

g2
```



- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio2.pdf`.

```
n_salida <- '2'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
g2
dev.off()
```

```
## pdf
##   2
```

Ejercicio 3

- Carga el fichero `iris.tidy.Rdata` utilizando lo aprendido en la práctica 1.

```
f2 <- paste0(carpeta, 'iris_tidy.Rdata')
load(f2)
```

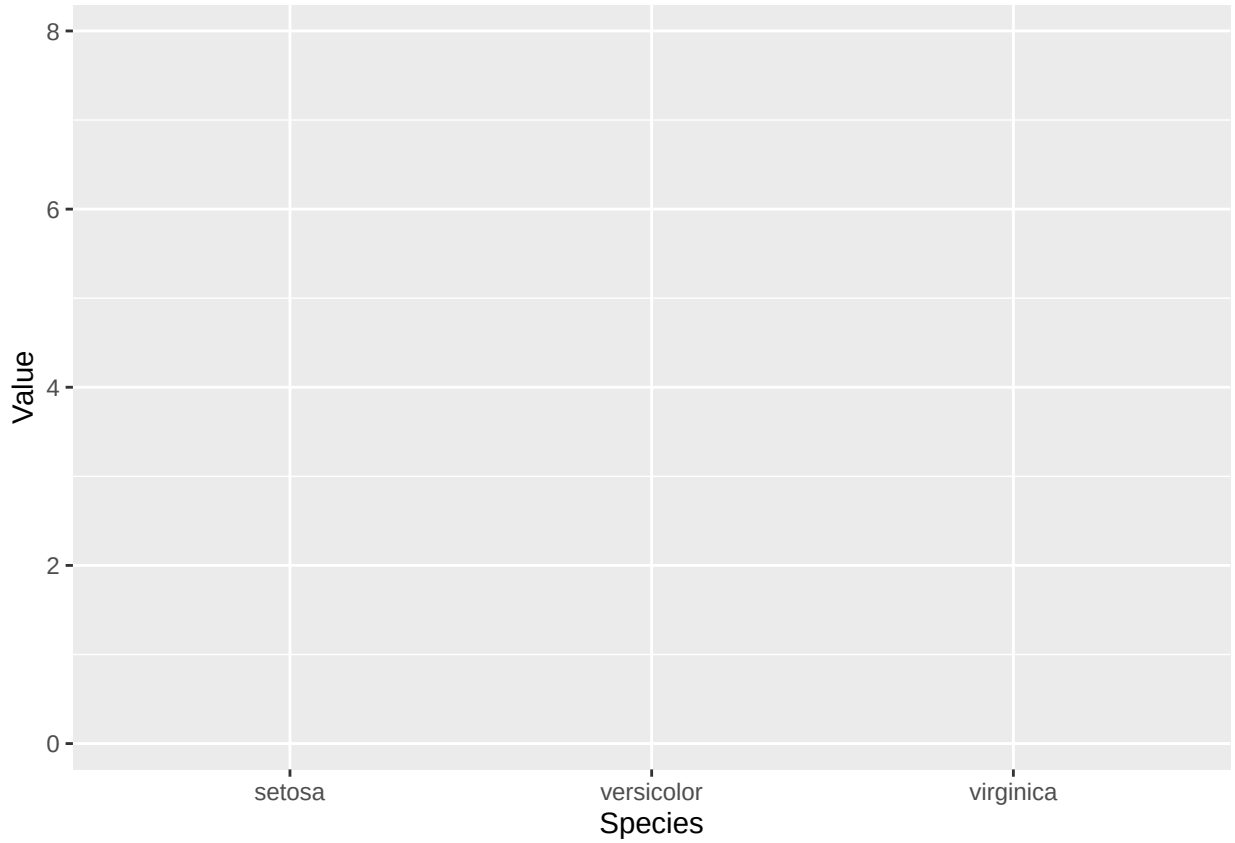
- Utiliza la función **str()** para ver qué variables contiene el dataset.

```
str(iris.tidy)
```

```
## 'data.frame':    600 obs. of  4 variables:
##  $ Species: Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Part   : Factor w/ 2 levels "Petal","Sepal": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Measure: chr  "Length" "Length" "Length" "Length" ...
##  $ Value  : num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
```

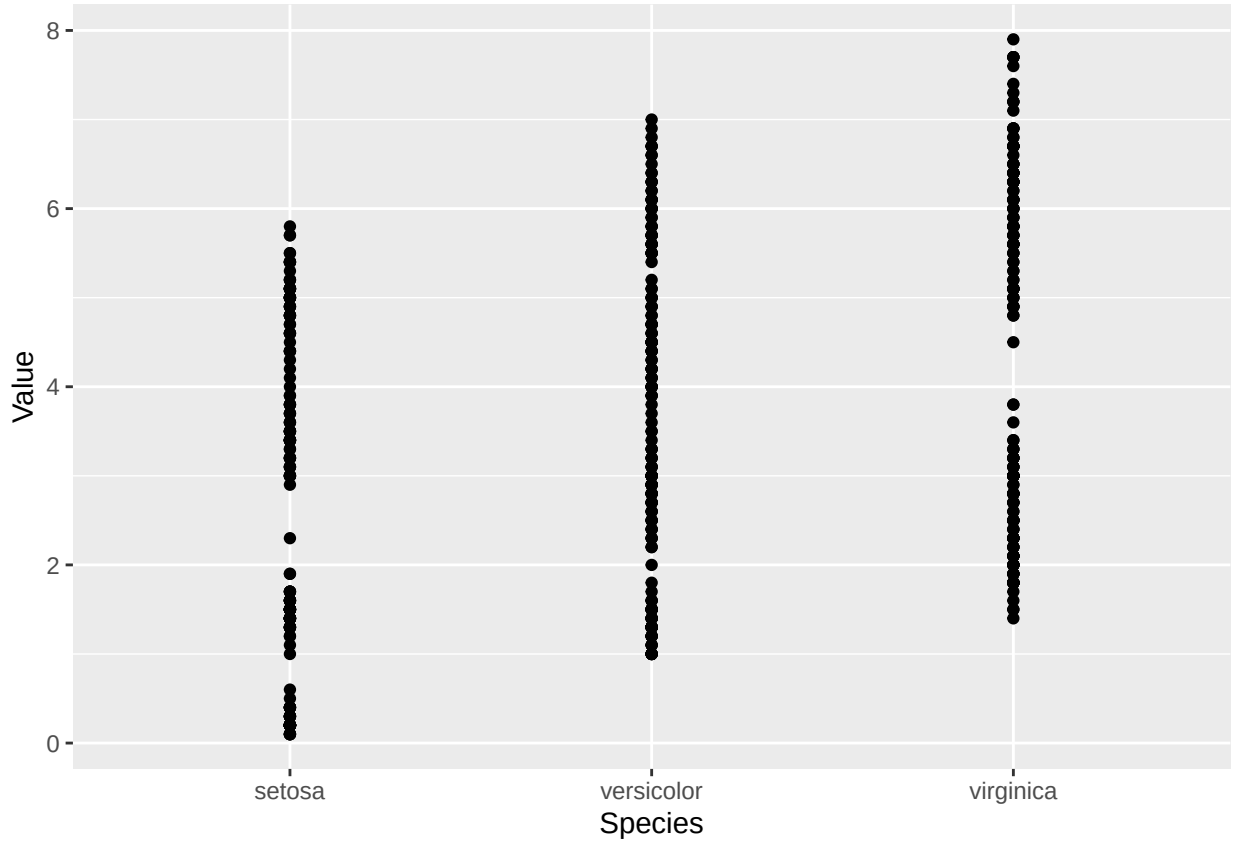
- Crea un objeto ggplot denominado `iris.grafico1` en el que esté definido el conjunto de datos `iris.tidy` alojado en el fichero `iris.tidy.RData` y un *aesthetics* en el que en el eje x esté definida la variable `Species` y en el eje y esté definida la variable `Value`.

```
iris.grafico1 <- ggplot(data = iris.tidy, mapping = aes(x = Species, y = Value))
iris.grafico1
```

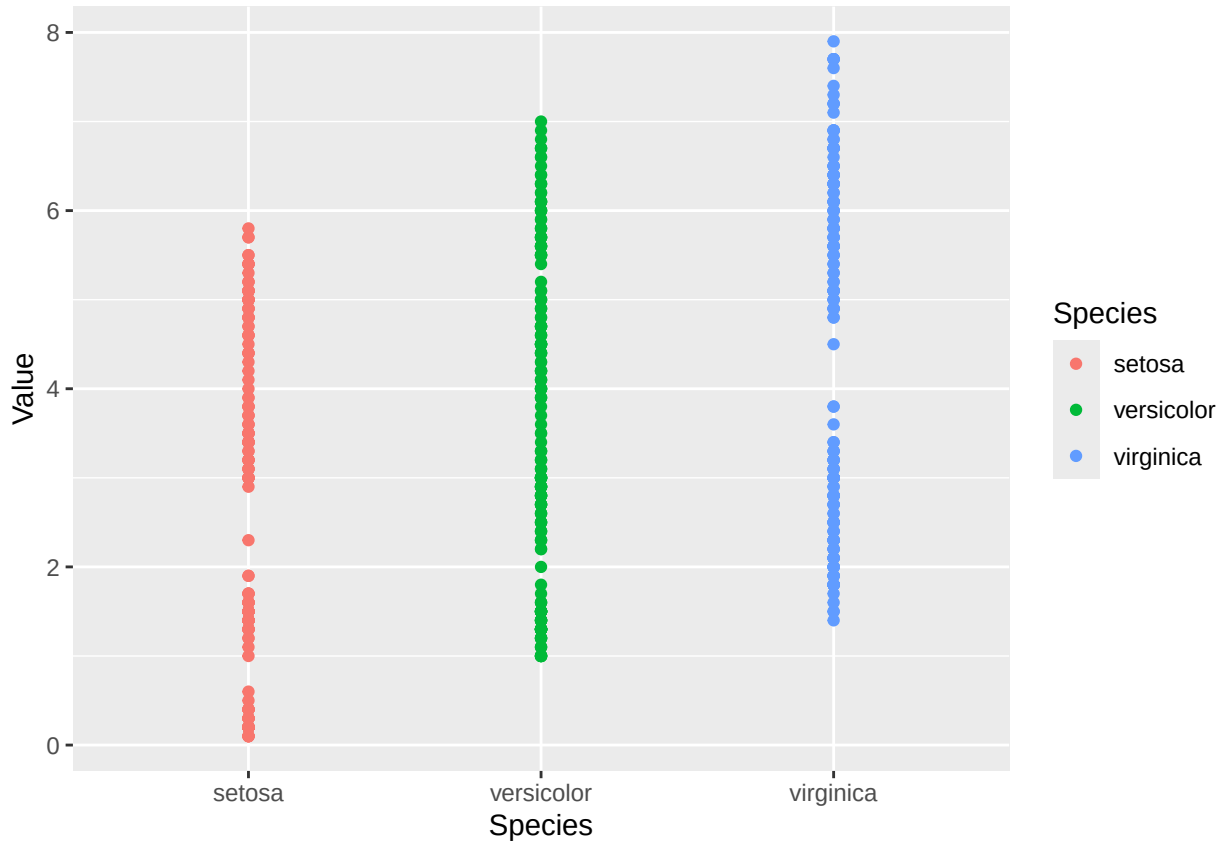


- Añade una capa que defina el objeto geométrico necesario para construir un gráfico de dispersión de las variables `Species` y `Value`.

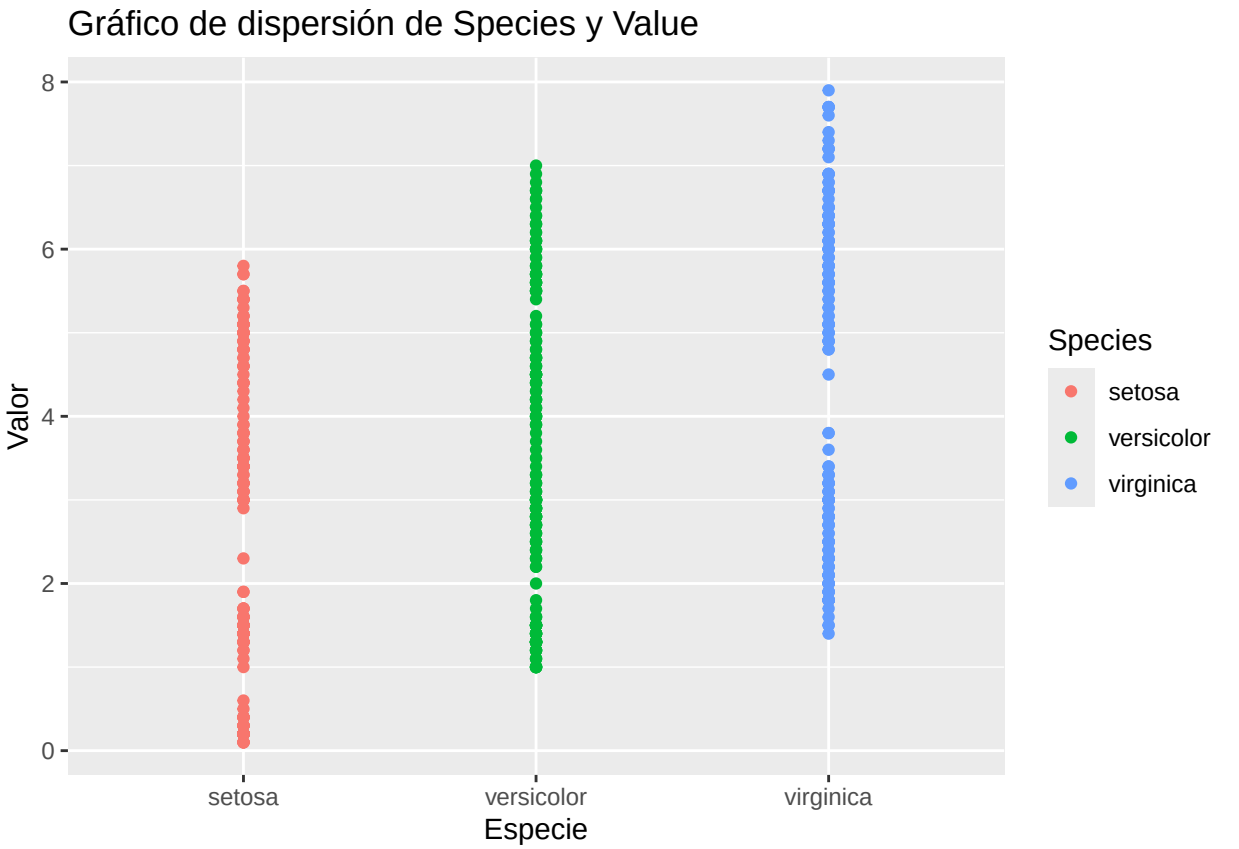
```
iris.grafico1 + geom_point()
```




```
iris.grafico1 + geom_point(aes(color = Species))
```

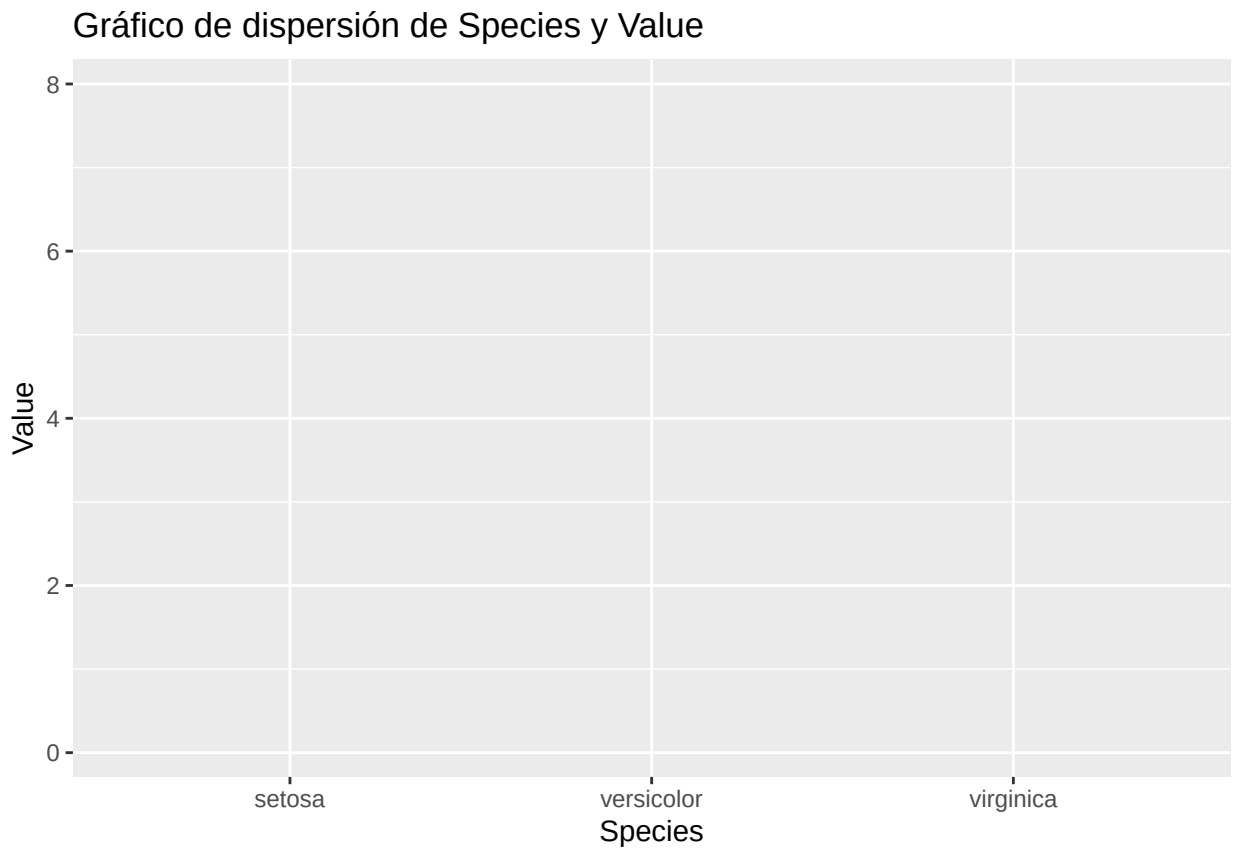



```
iris.grafico1 + geom_point(aes(color = Species)) + labs(title = 'Gráfico de dispersión de Species y Valu
                                                        x = 'Especie',
                                                        y = 'Valor')
```

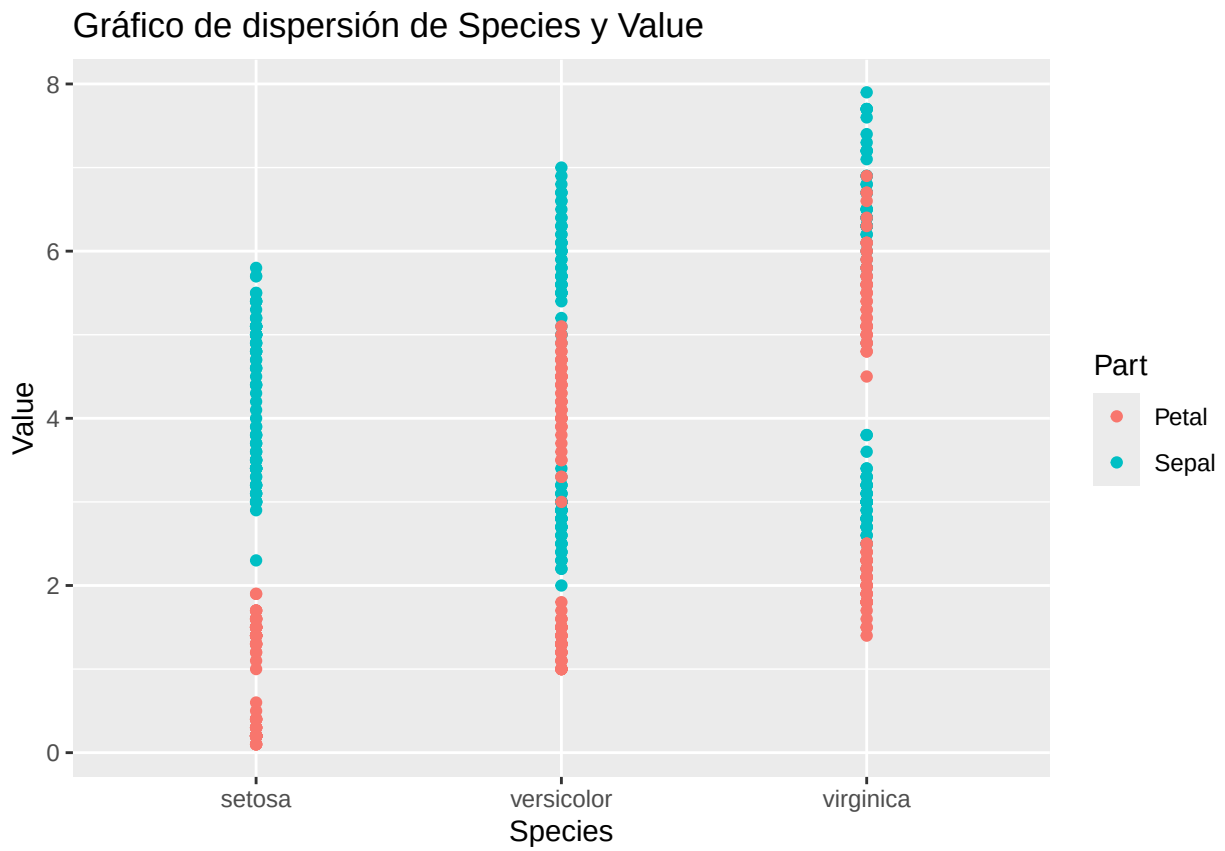


- Modifica el *aesthetics* de `iris.grafico1` para visualizar la variable categórica `Part` y almacénalo en `iris.grafico2`. A partir del objeto `iris.grafico2`, representa un gráfico de dispersión en el que se muestren los puntos dados por las variables `Species` y `Value`, y el color del punto se corresponda con el color de la variable `Part`.

```
iris.grafico2 <- ggplot(data = iris.tidy,
                        mapping = aes(x = Species, y = Value, color = Part)) +
                  ggtitle('Gráfico de dispersión de Species y Value')
iris.grafico2
```

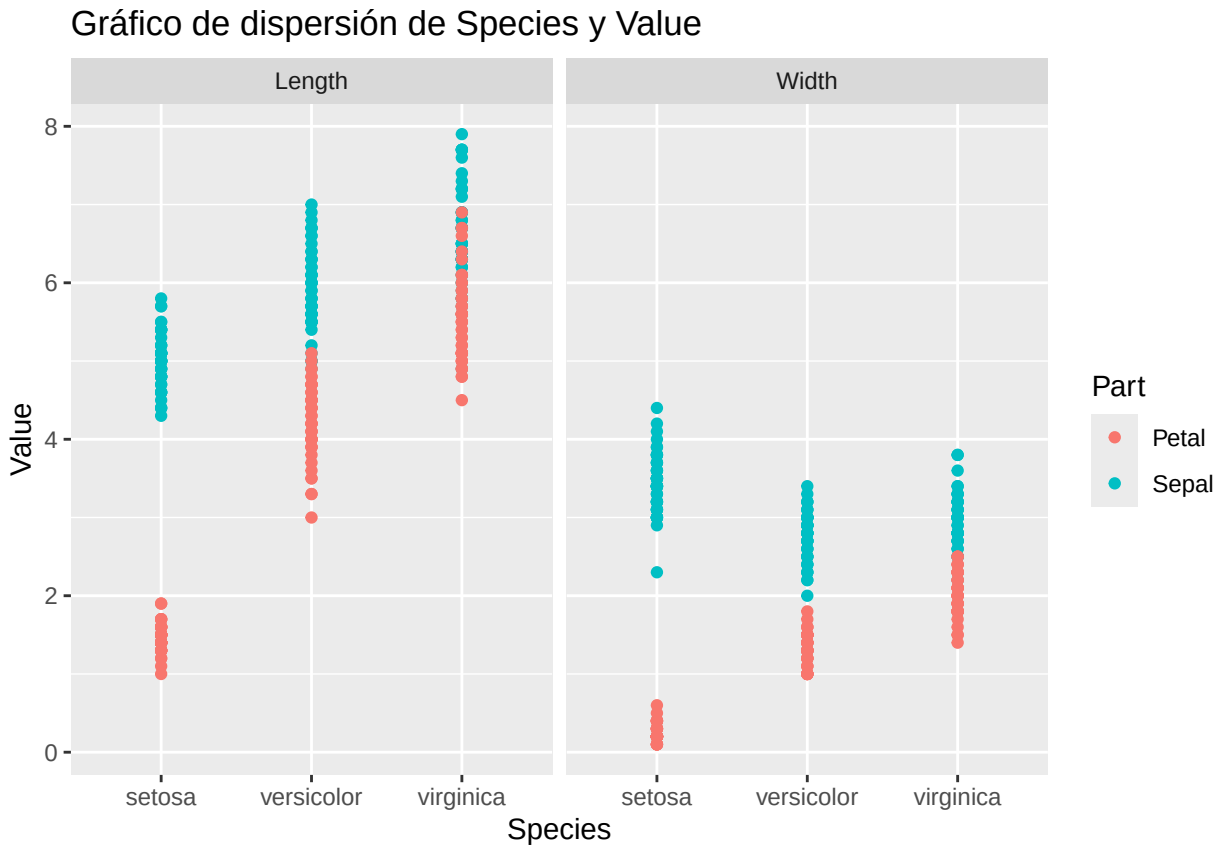



```
iris.grafico2 + geom_point()
```

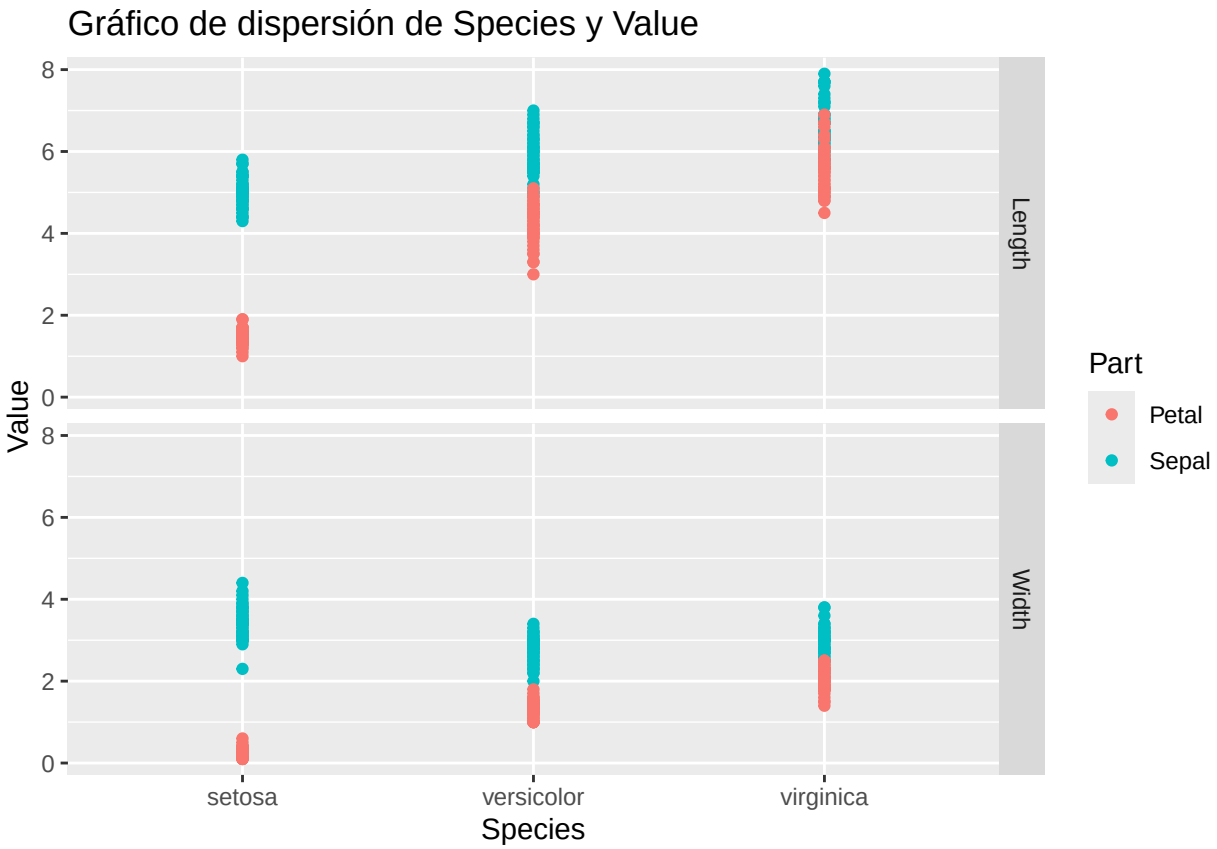


- Incluye información de la variable `Measure` haciendo dos gráficos de dispersión (dispuestos en filas) con **facet_grid()** como en el apartado de introducción para los dos valores que toma la variable `Measure`. Prueba a cambiar la disposición de los gráficos para que sea en columnas ¿Qué disposición, en filas o en columnas, permite comparar mejor los valores de las dos variables?

```
# iris.grafico2 + facet_grid(~ Measure)
iris.grafico2 + geom_point() + facet_grid(.~Measure) # Forma vertical
```

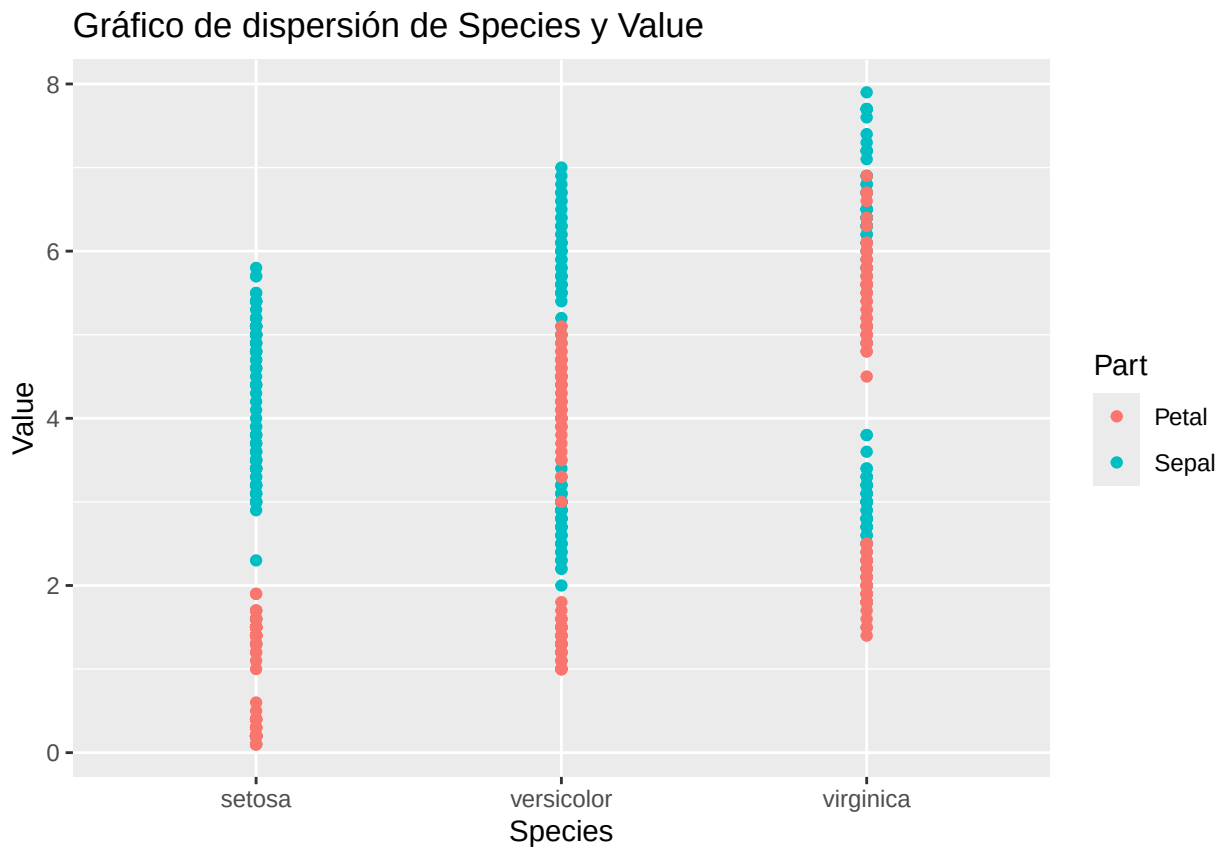



```
iris.grafico2 + geom_point() + facet_grid(Measure~.) # Forma horizontal
```

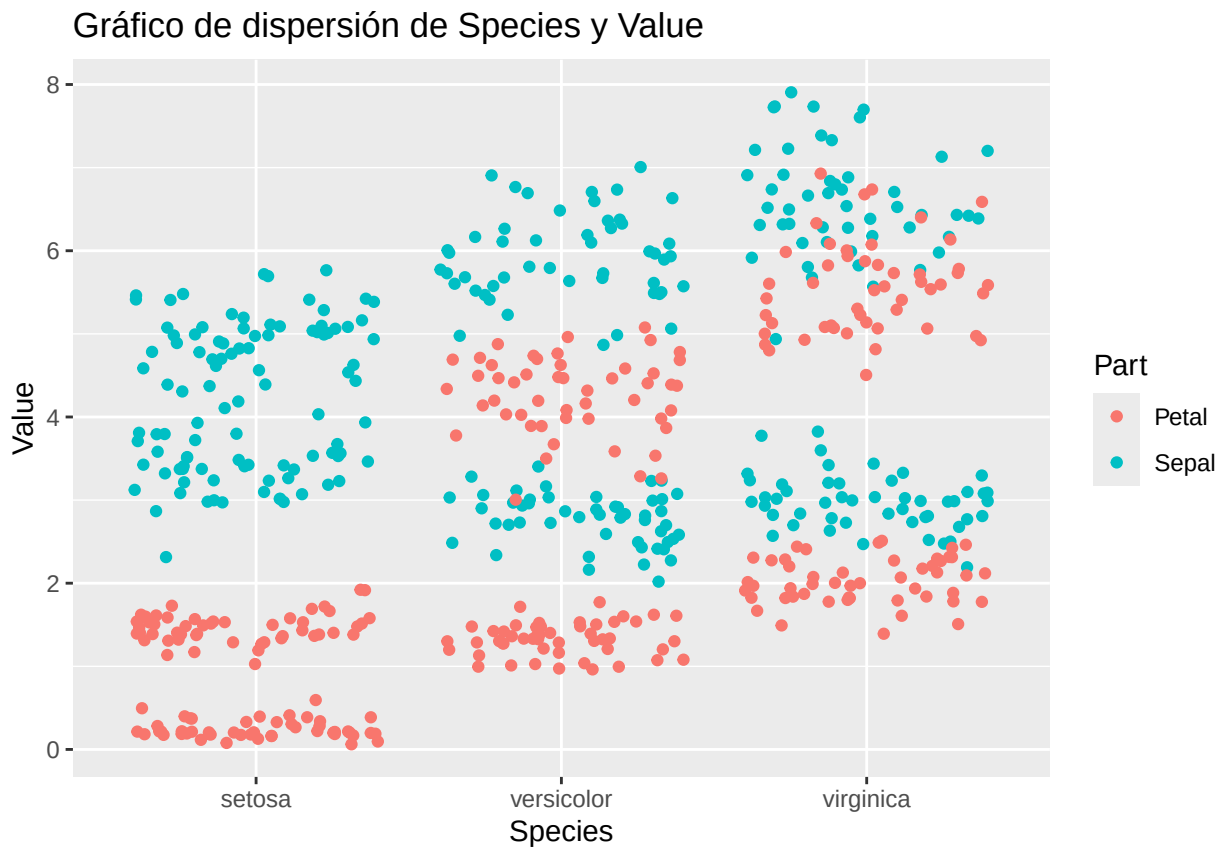


- Cambia ahora la capa **geom_point()** del gráfico anterior por la capa **geom_jitter()**. ¿Qué diferencia observas?

```
iris.grafico2 + geom_point()
```




```
g3 <- iris.grafico2 + geom_jitter()
g3
```



- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio3.pdf`.

```
n_salida <- '3'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
g3
dev.off()
```

```
## pdf
##   2
```

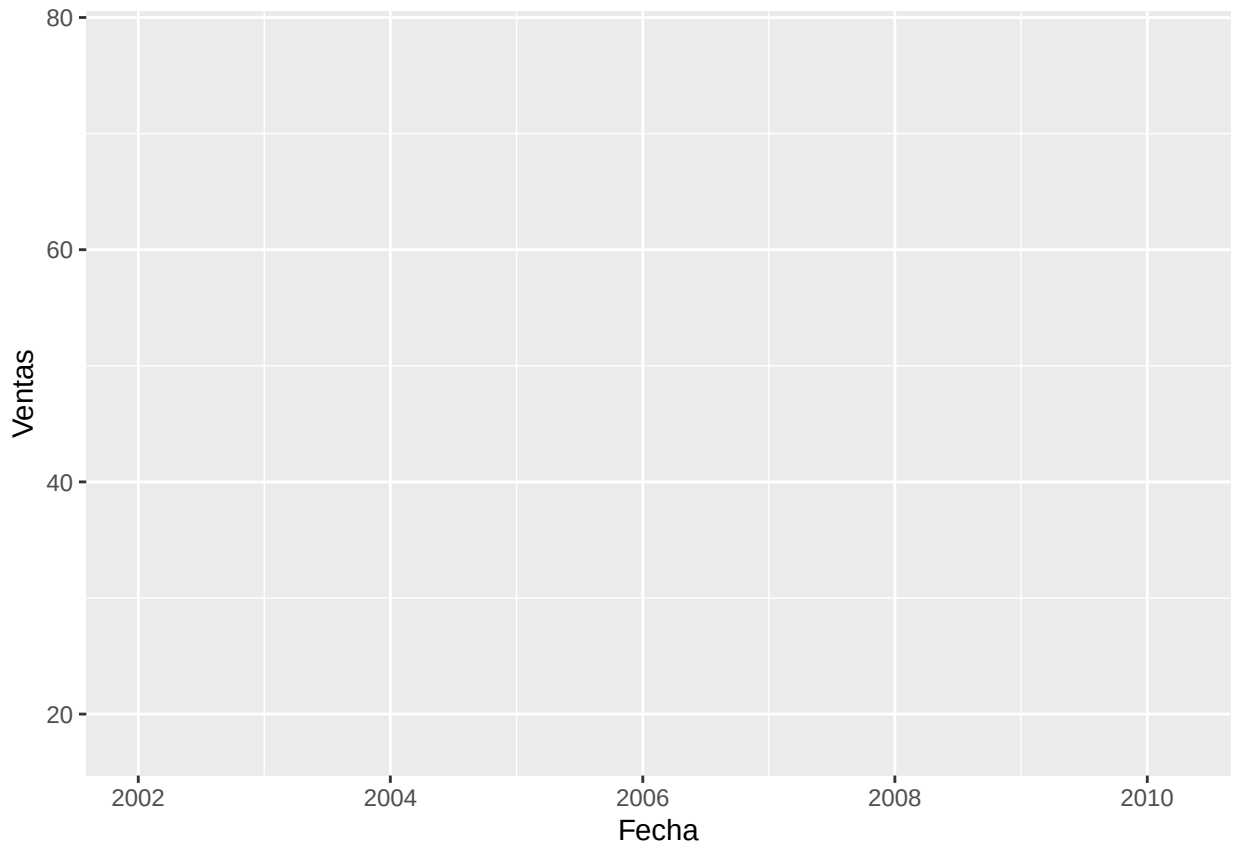
Ejercicio 4

- Carga el fichero `Ejercicio4_Data.Rdata`. Examina las variables que tiene dicho dataset.

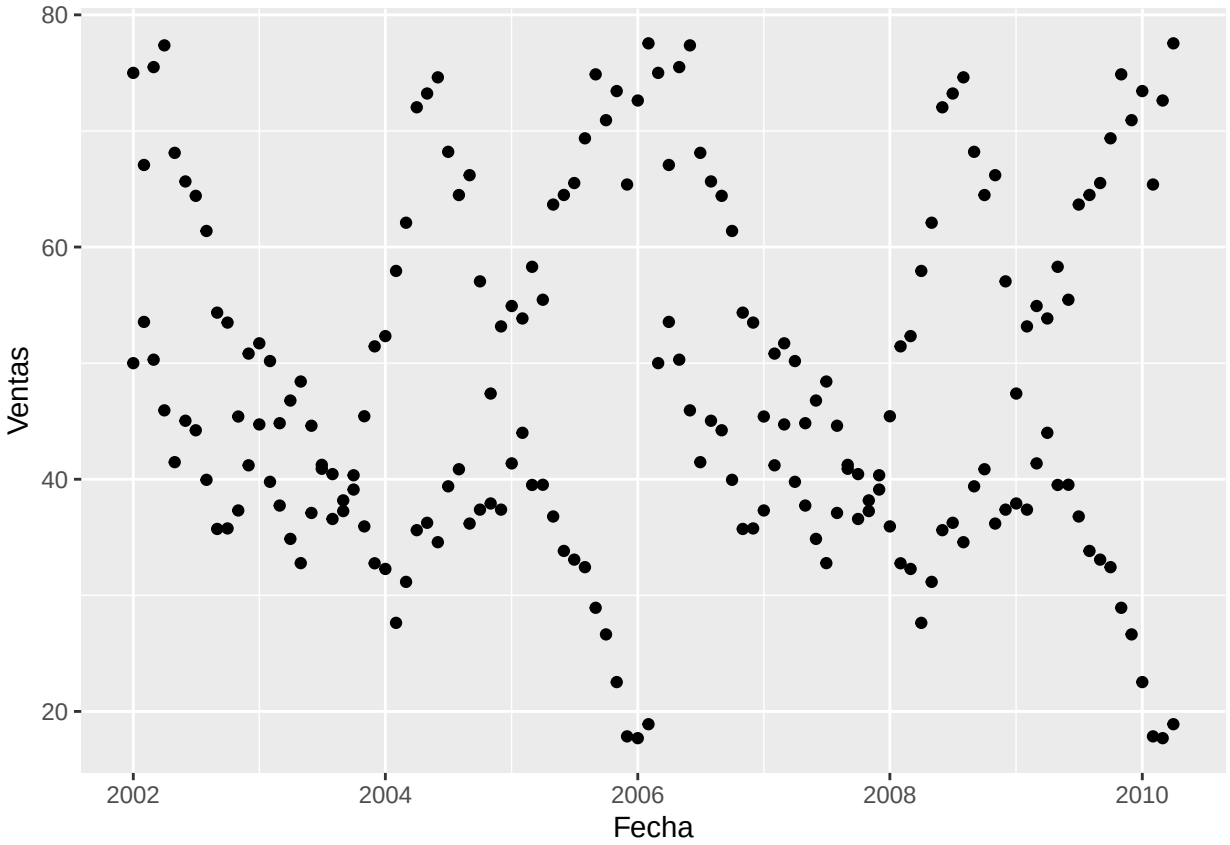
```
f3 <- paste0(carpeta, 'Ejercicio4_Data.Rdata')
load (f3)
```

- Representa un gráfico de puntos, en el que se muestre la fecha en el eje x y las ventas en el eje y. Queremos que los datos de cada fabricante sean de un color distinto. ¿Es este gráfico valido para evaluar la evolución de las ventas de cada fabricante?

```r
g4 <- ggplot(data = DatosEjer4, mapping = aes(x = Fecha, y = Ventas))
g4
```

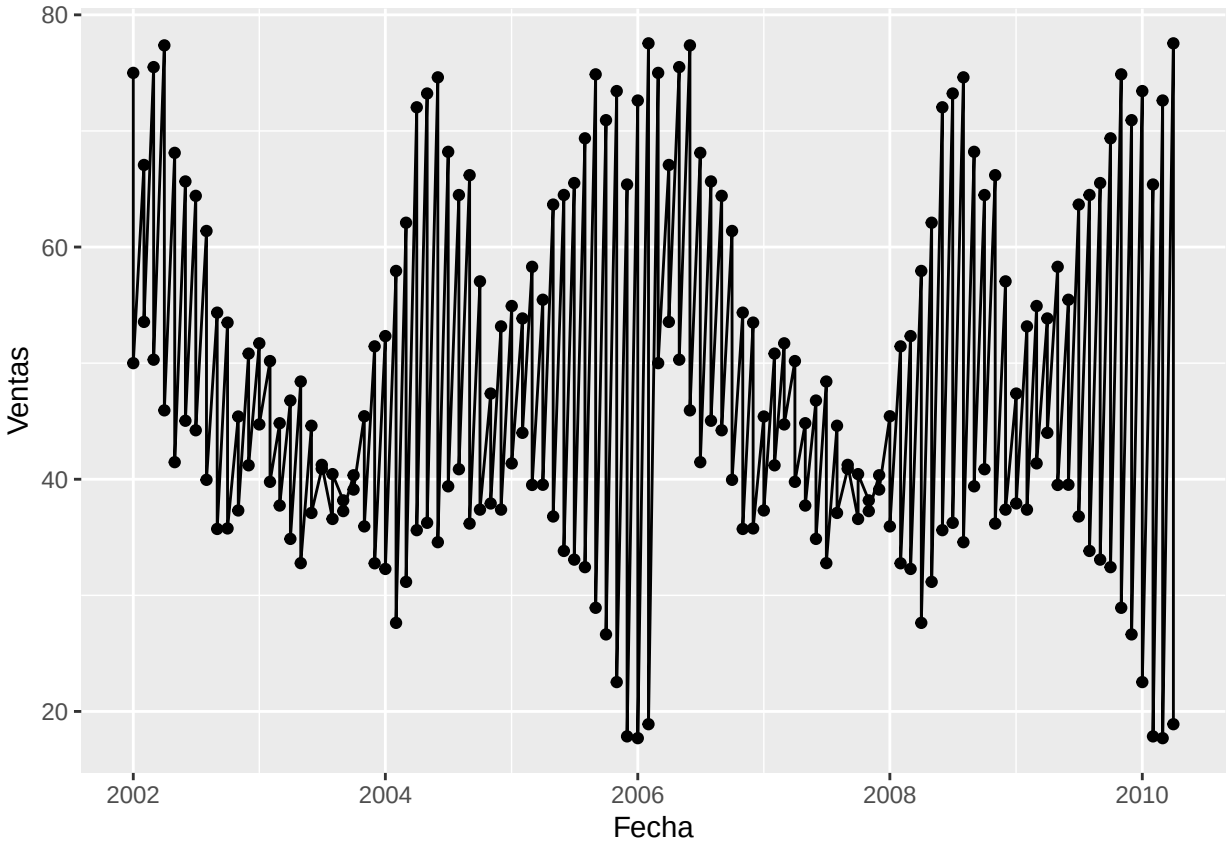



```r
g4  + geom_point()
```



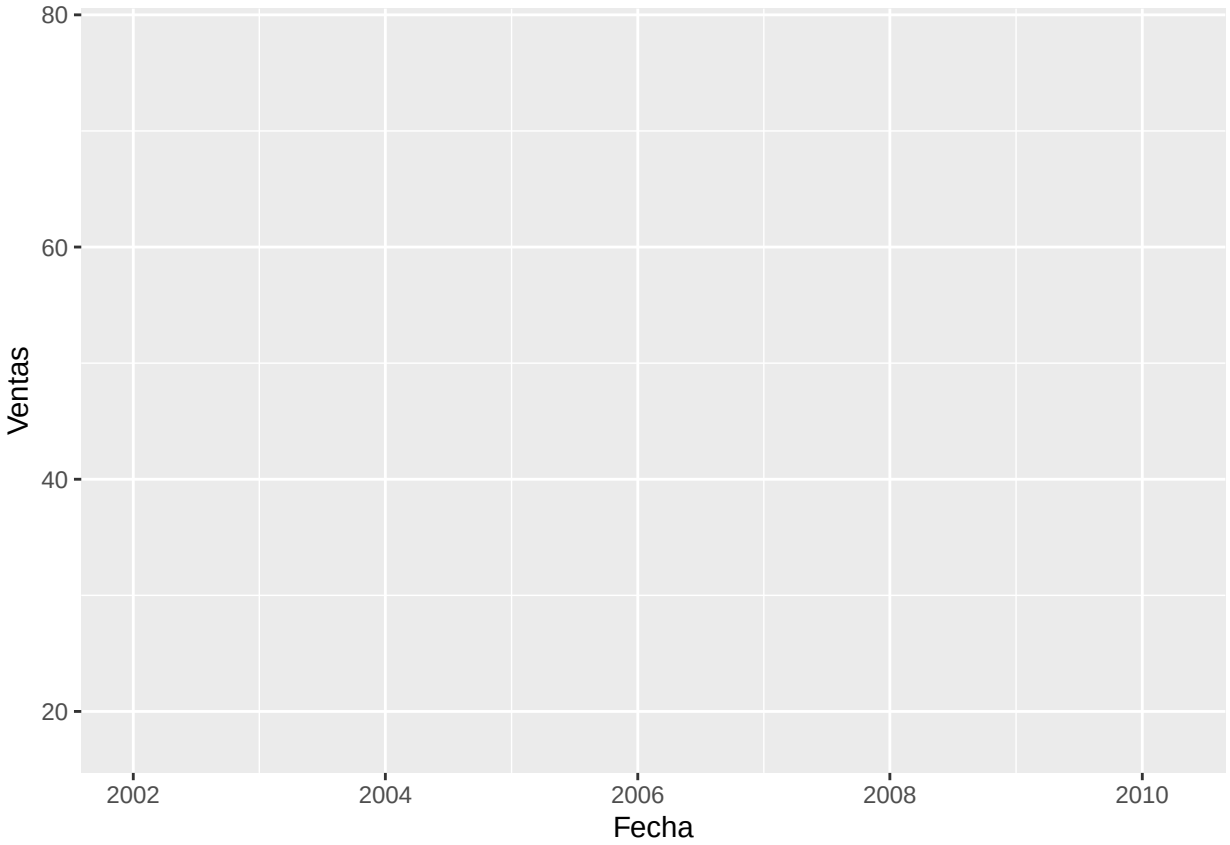
- Representa un gráfico como el anterior en el que se muestre la línea que une los puntos que antes has representado.

```
g4 + geom_point() + geom_line()
```

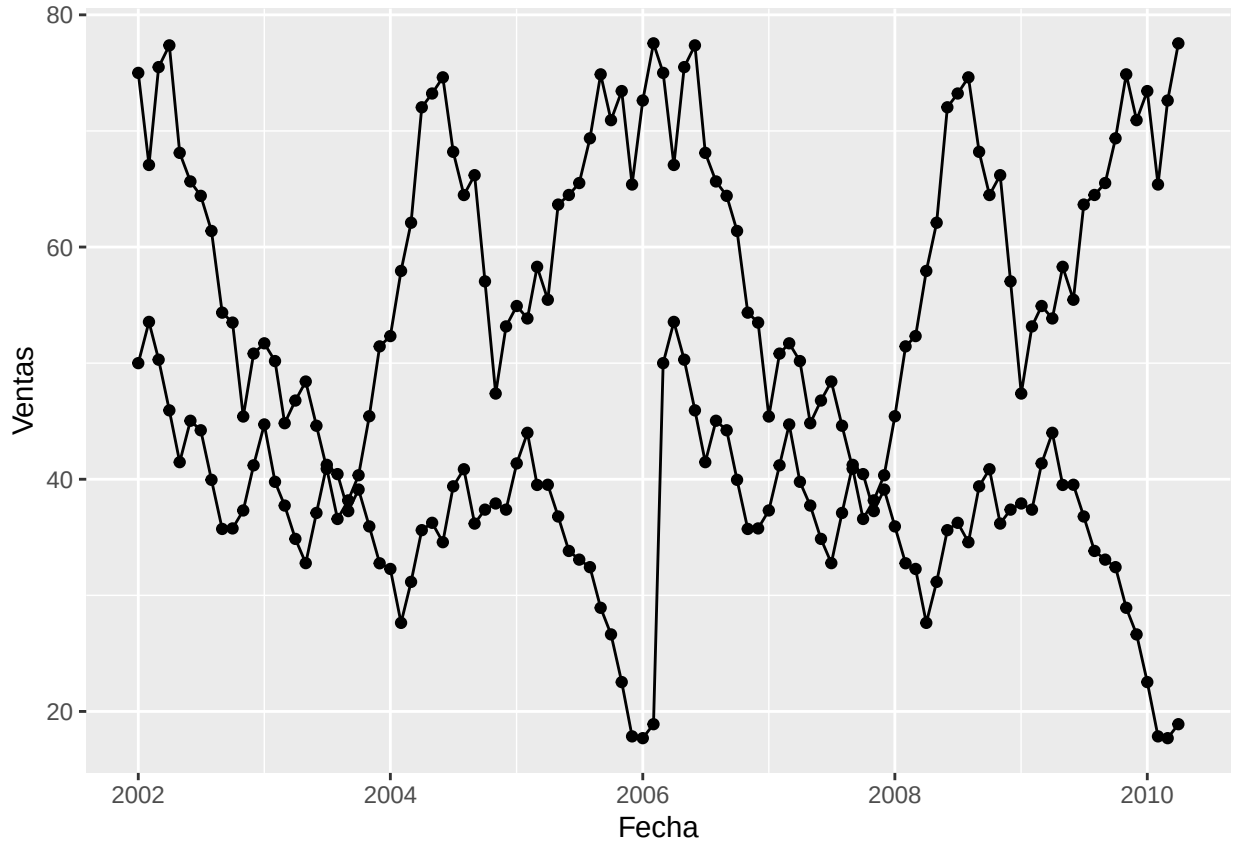


- Modifica el gráfico anterior para que la tendencia de cada fabricante este trazada con un tipo de línea diferente.

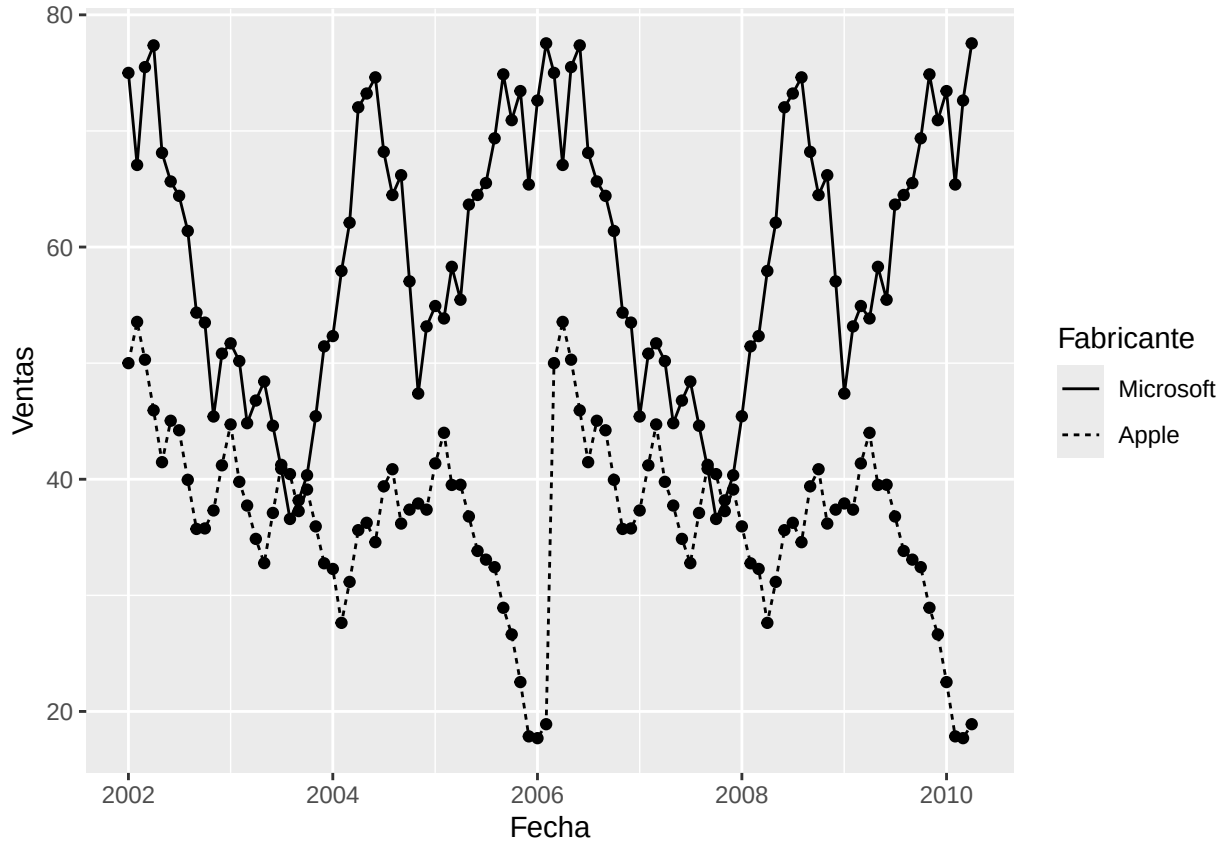
```
g4 <- ggplot(data = DatosEjer4, mapping = aes(x = Fecha, y = Ventas, group = Fabricante))
g4 # Ahora está agrupado por Fabricante
```



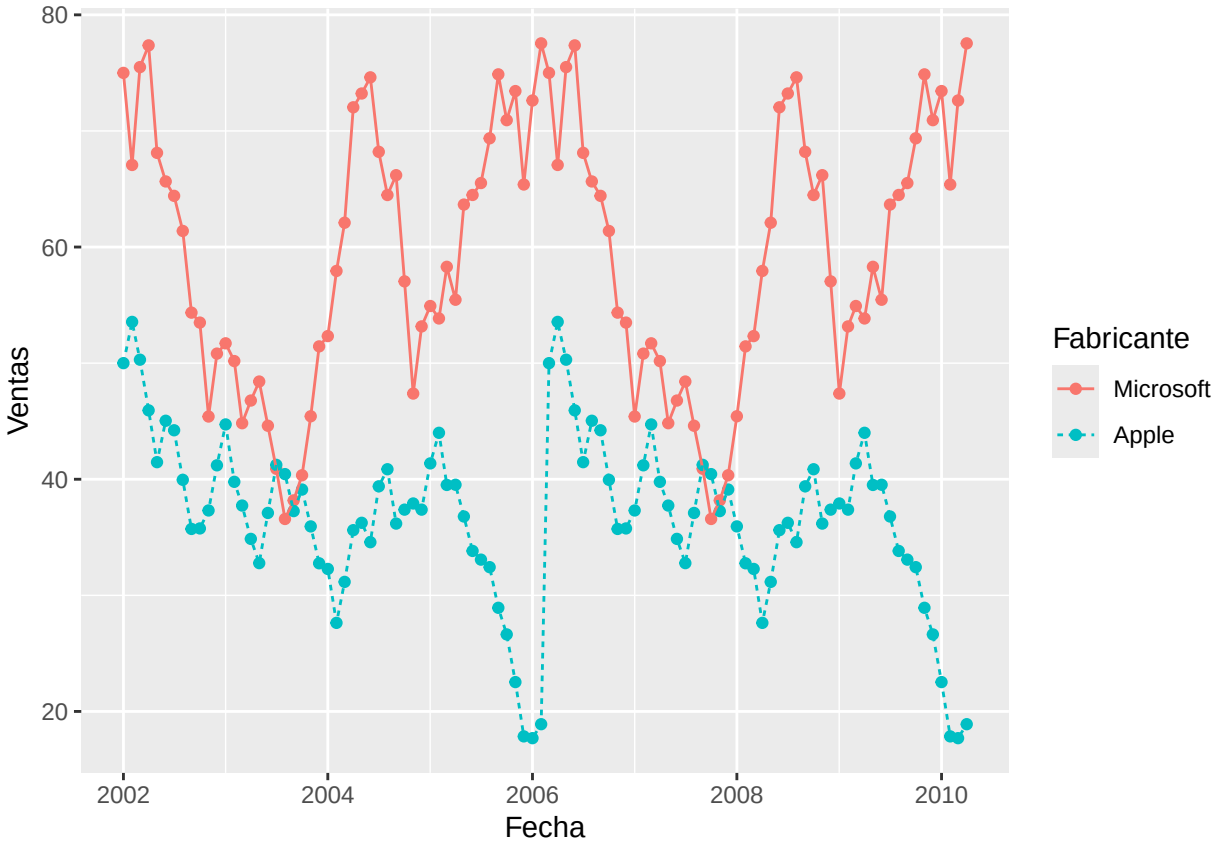

```
g4 + geom_point() + geom_line()
```




```
g4 + geom_point() + geom_line(aes(linetype = Fabricante))
```

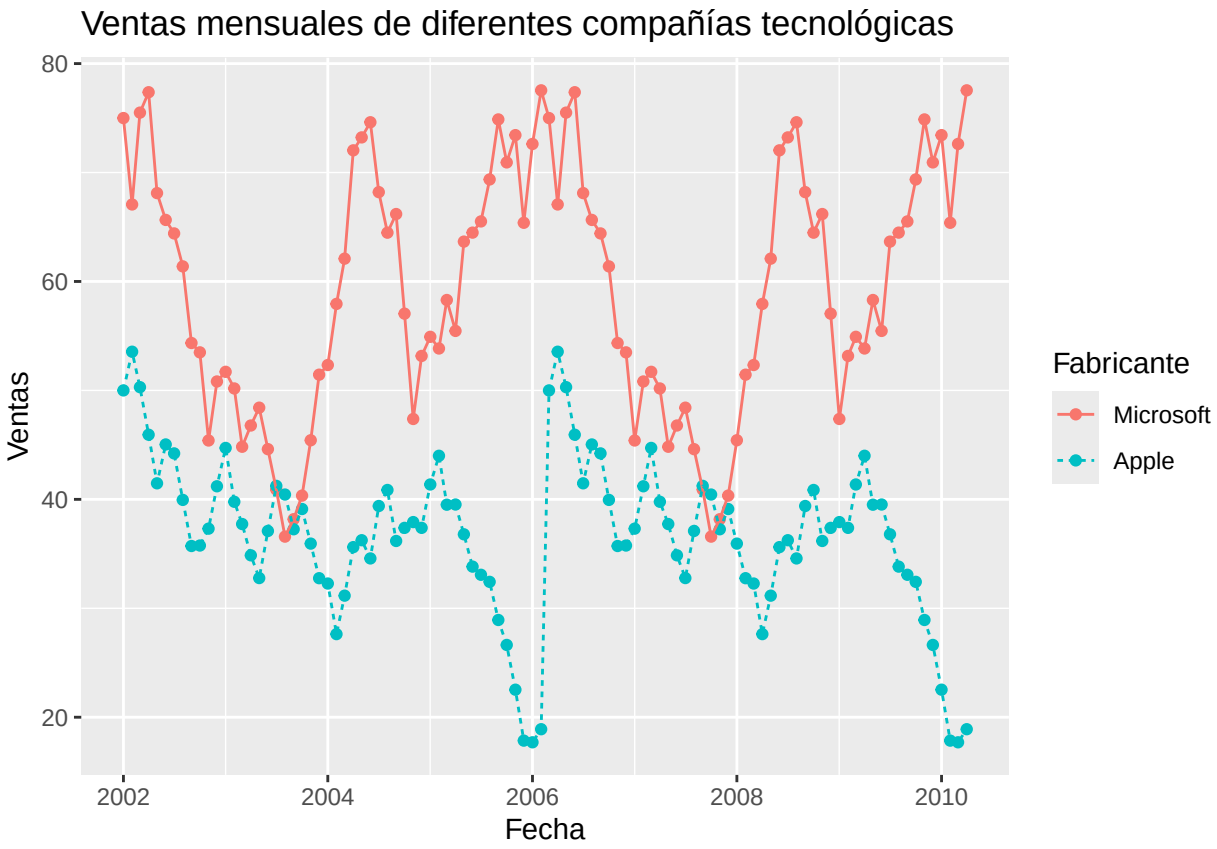



```
g4 + geom_point(aes(color = Fabricante)) + geom_line(aes(linetype = Fabricante, color = Fabricante))
```



- Añade el siguiente título al gráfico *"Ventas mensuales de diferentes compañías tecnológicas"*.

```
g4 <- g4 + geom_point(aes(color = Fabricante)) + geom_line(aes(linetype = Fabricante, color = Fabricante
  ggtitle(label = 'Ventas mensuales de diferentes compañías tecnológicas')
g4
```



- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio4.pdf`.

```
n_salida <- '4'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
g4
dev.off()
```

```
## pdf
##   2
```

Ejercicio 5

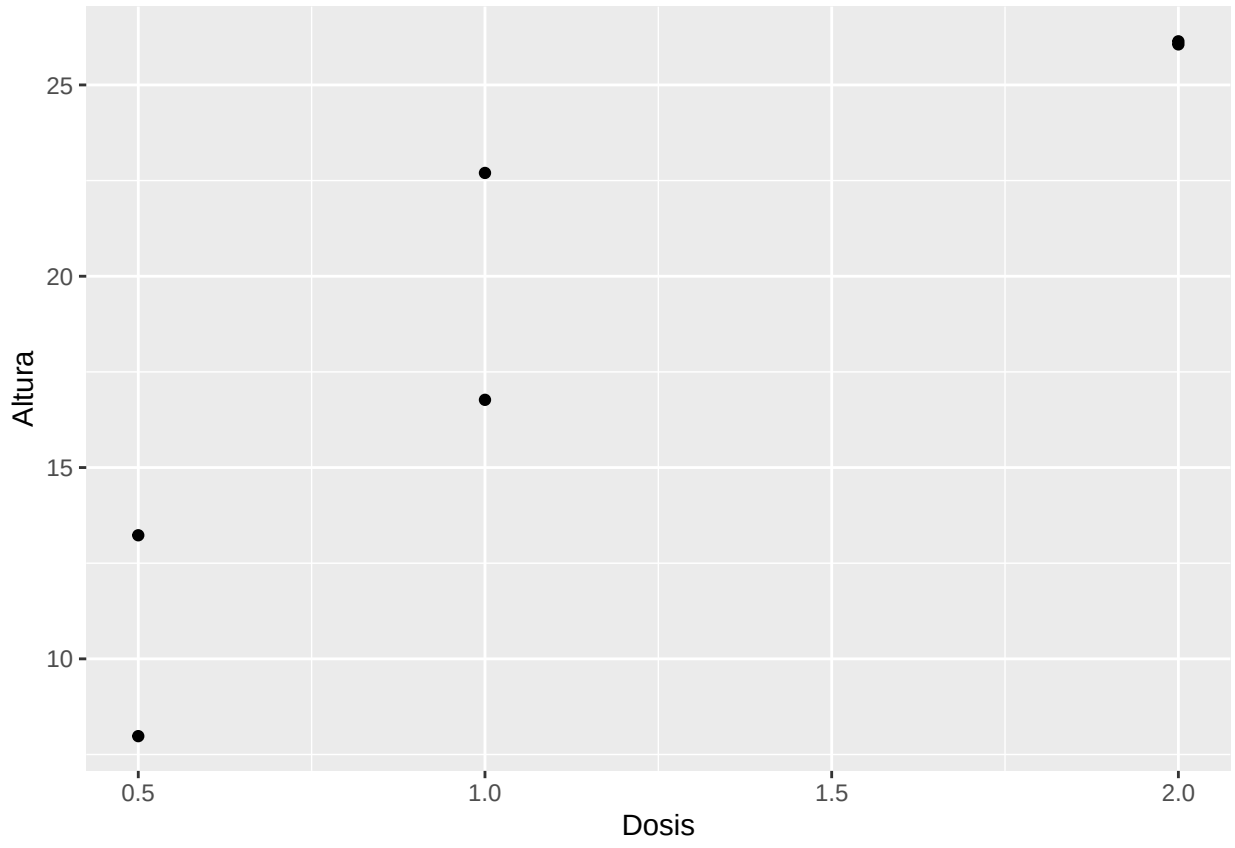
Añadir barras de error a un gráfico de línea.

- Carga el fichero `Experimento.Rdata`. Examina las variables que tiene dicho dataset.

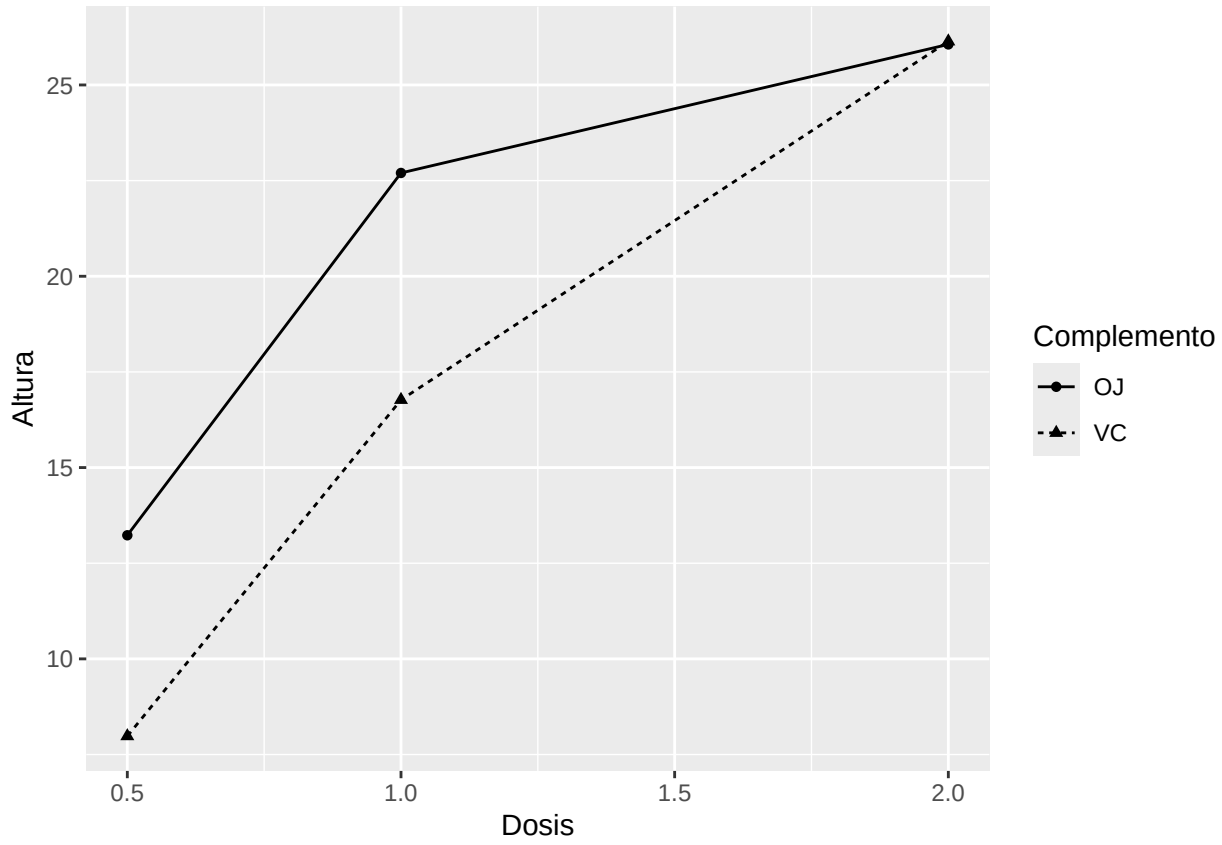
```
f4 <- paste0(carpeta, 'Experimento.Rdata')
load(f4)
```

- Representa un gráfico de puntos, en la que se muestre la `Dosis` en el eje x y la `Altura` en el eje y. Queremos que los datos de cada `Complemento` se represente en una línea diferente con diferentes trazos. Hazlo incluyendo las opciones de tipo de línea y forma del punto dentro del **aes()** de la capa **ggplot**.

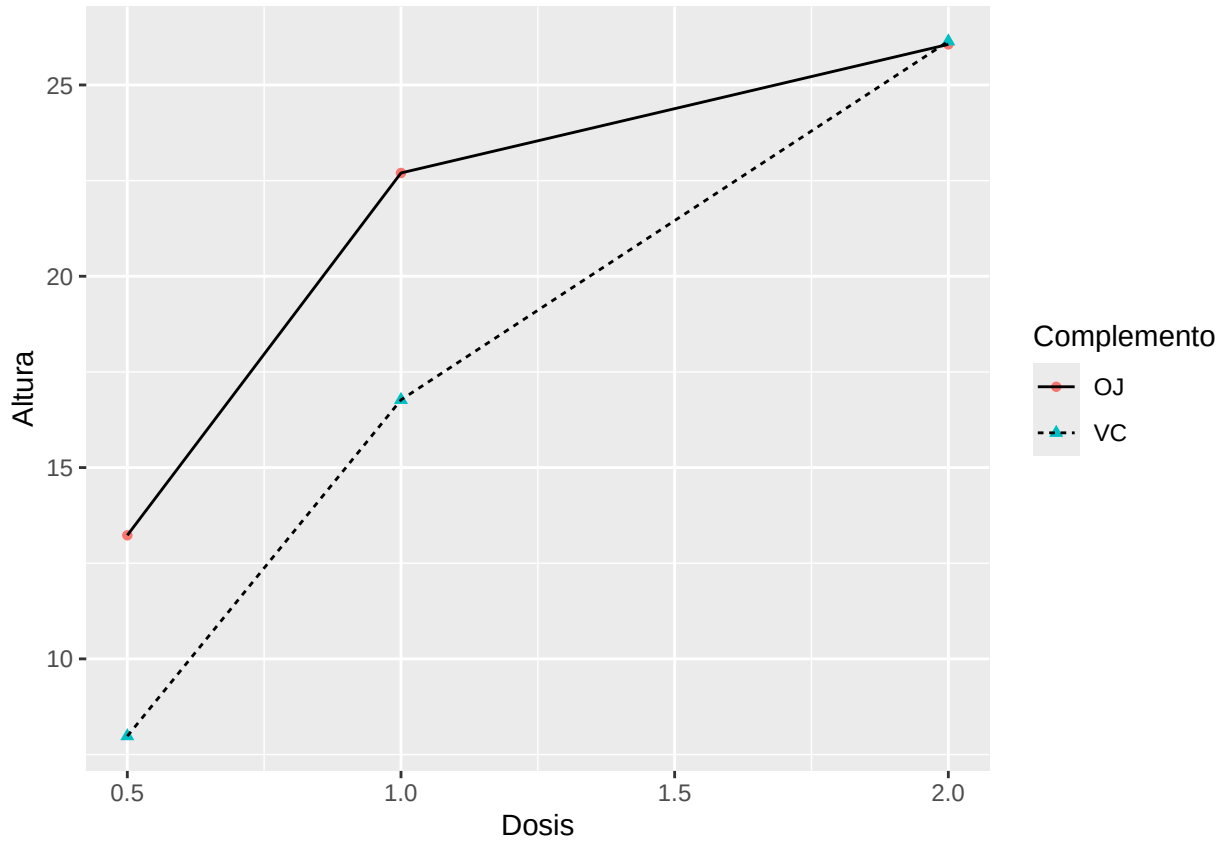
```
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento)) + geom_point()
```



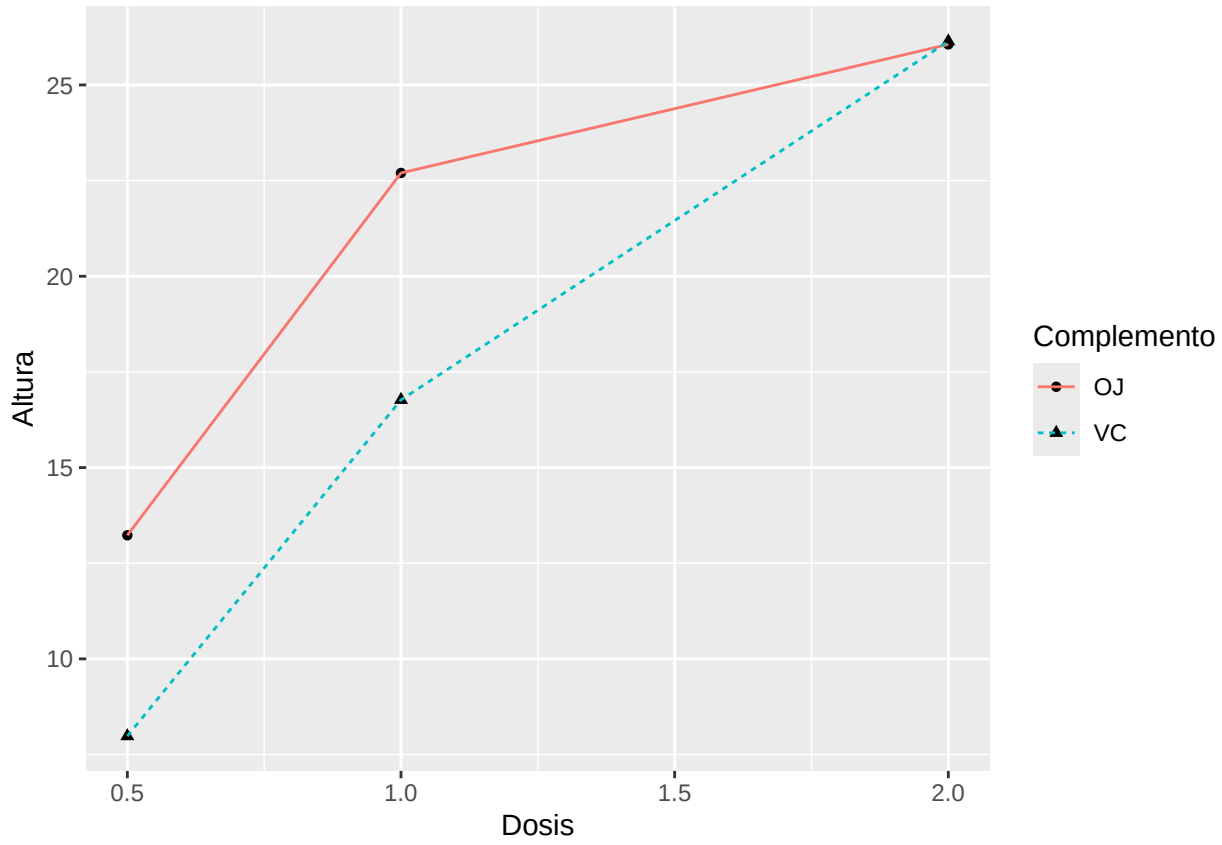

```
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento)) +
  geom_point(aes(shape = Complemento)) + geom_line(aes(linetype = Complemento))
```



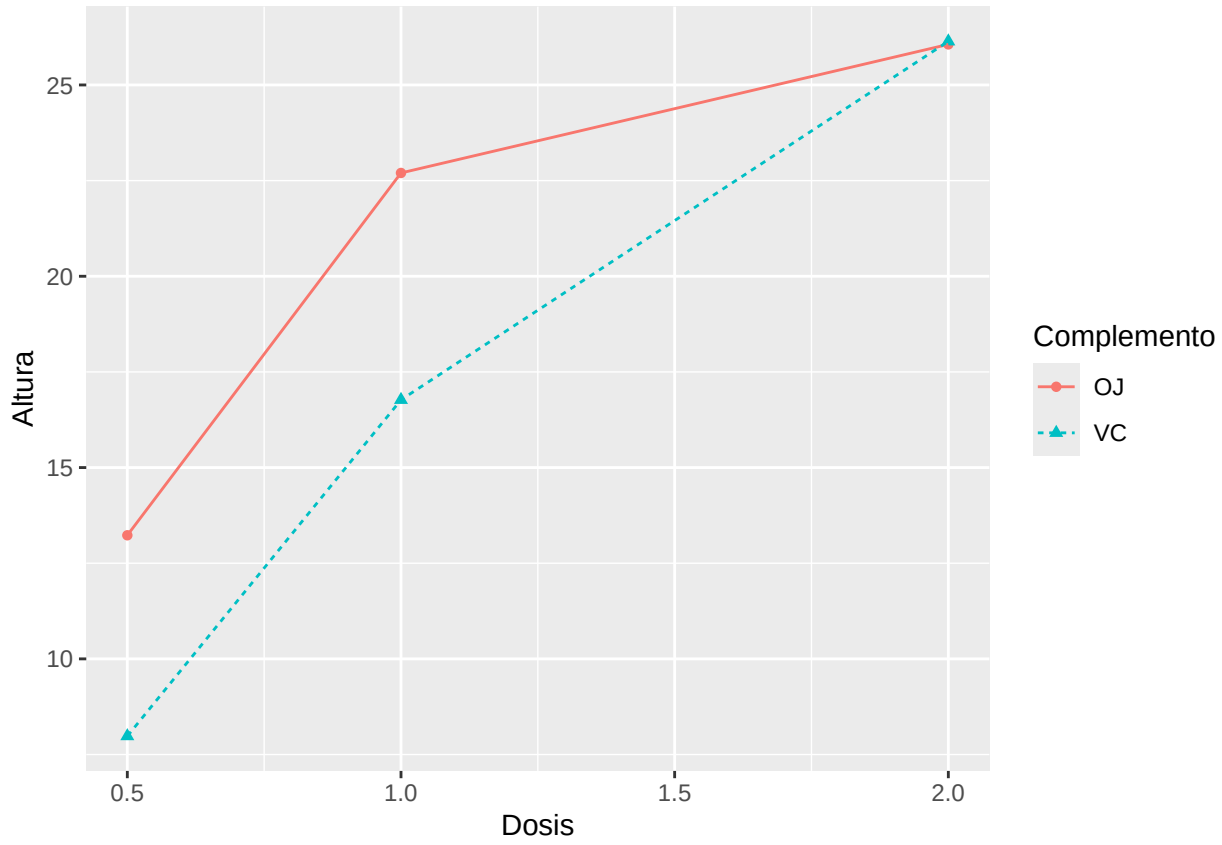

```
# Color se puede pasar a cada geom_function, o bien, en el 'general' de ggplot
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento)) +
  geom_point(aes(shape = Complemento, color = Complemento)) + geom_line(aes(linetype = Complemento))
```



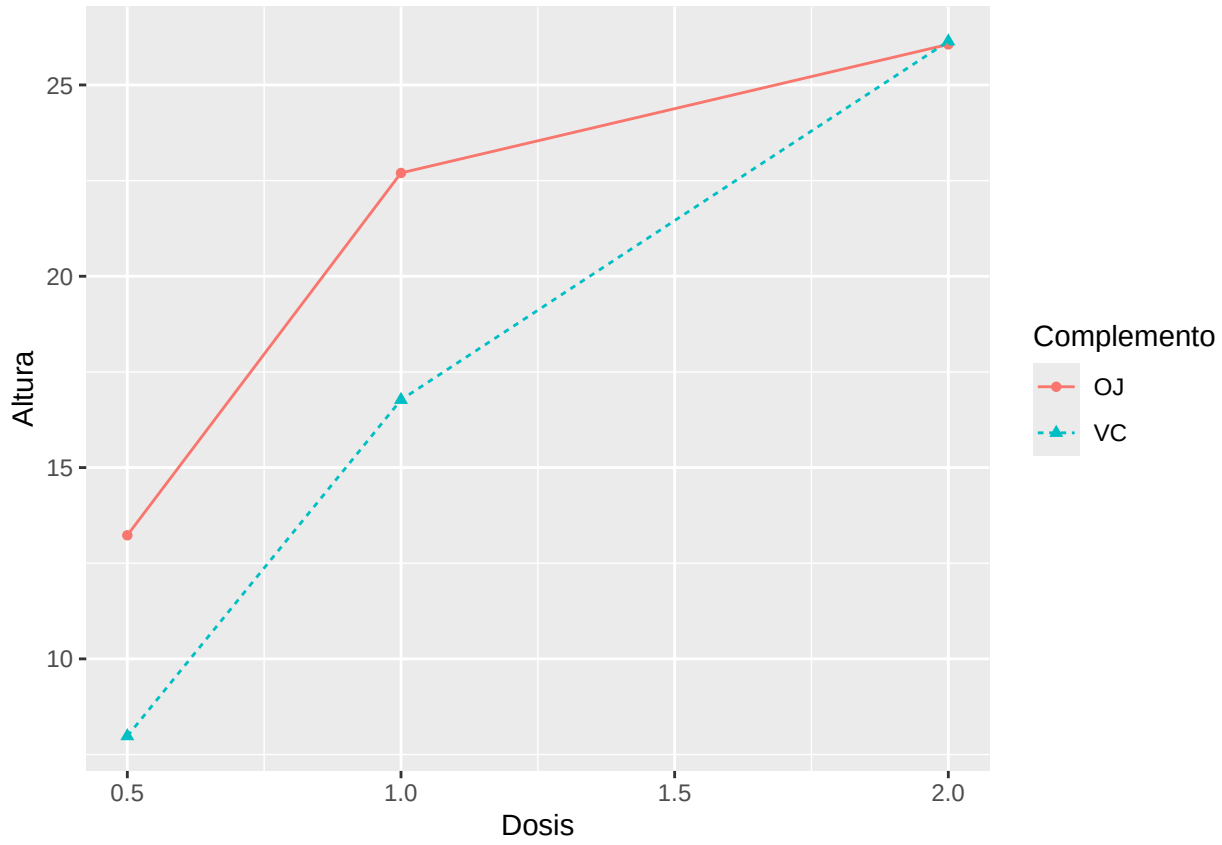

```
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento)) +
  geom_point(aes(shape = Complemento)) + geom_line(aes(linetype = Complemento, color = Complemento))
```




```
# Estos son los finales, y representan lo mismo
ggplot(data = Experimento,
       mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento, shape = Complemen
  geom_point() + geom_line()
```

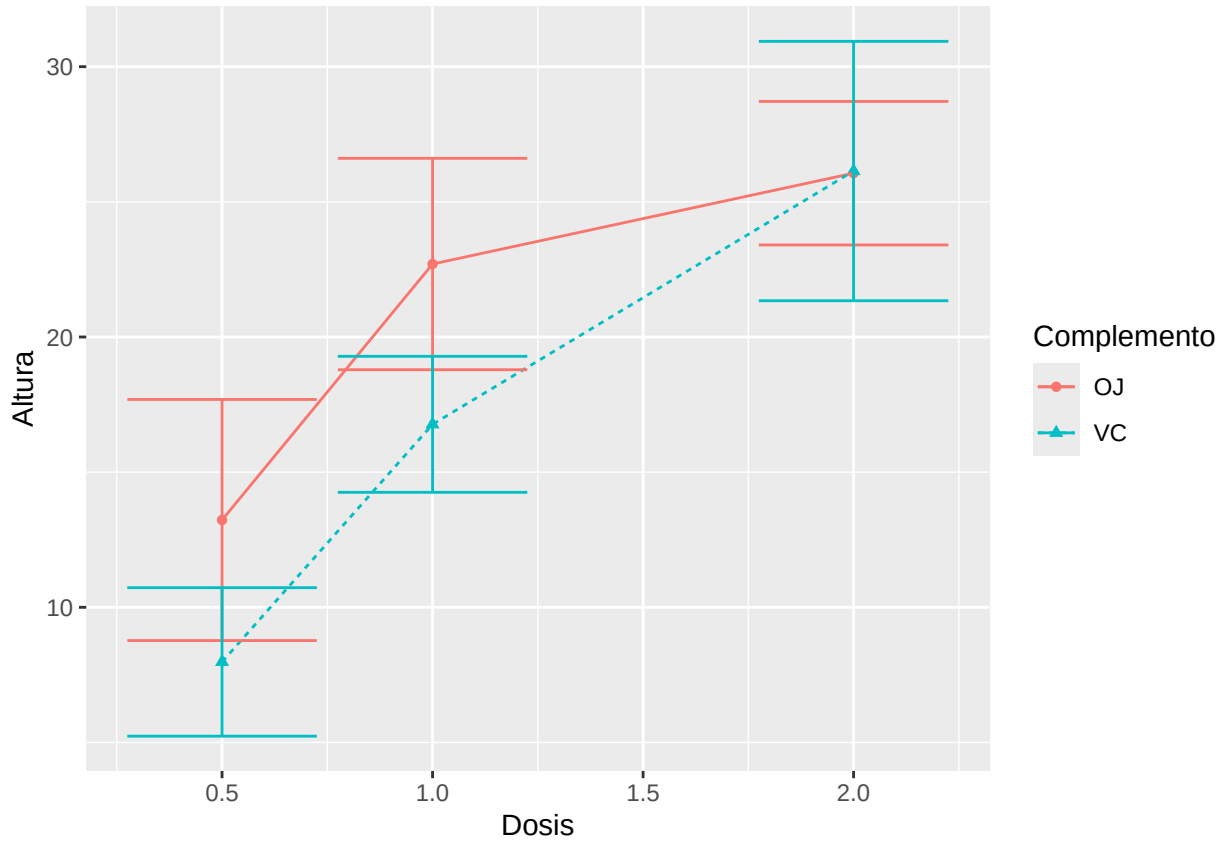



```
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complement
  geom_point(aes(shape = Complemento)) + geom_line(aes(linetype = Complemento))
```

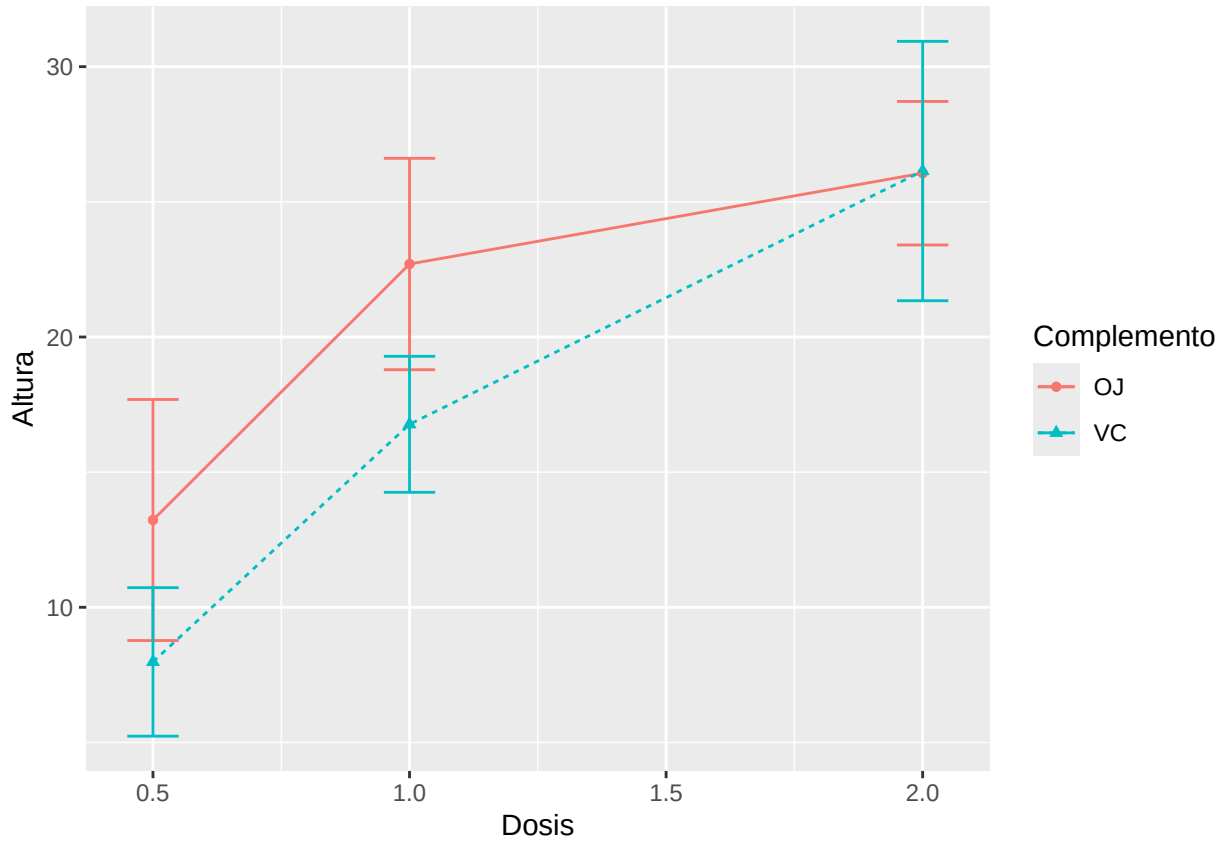


- Crea otro gráfico a partir de añadir al gráfico anterior barras de error a cada punto. El error viene dado por la variable `Error_Altura`.

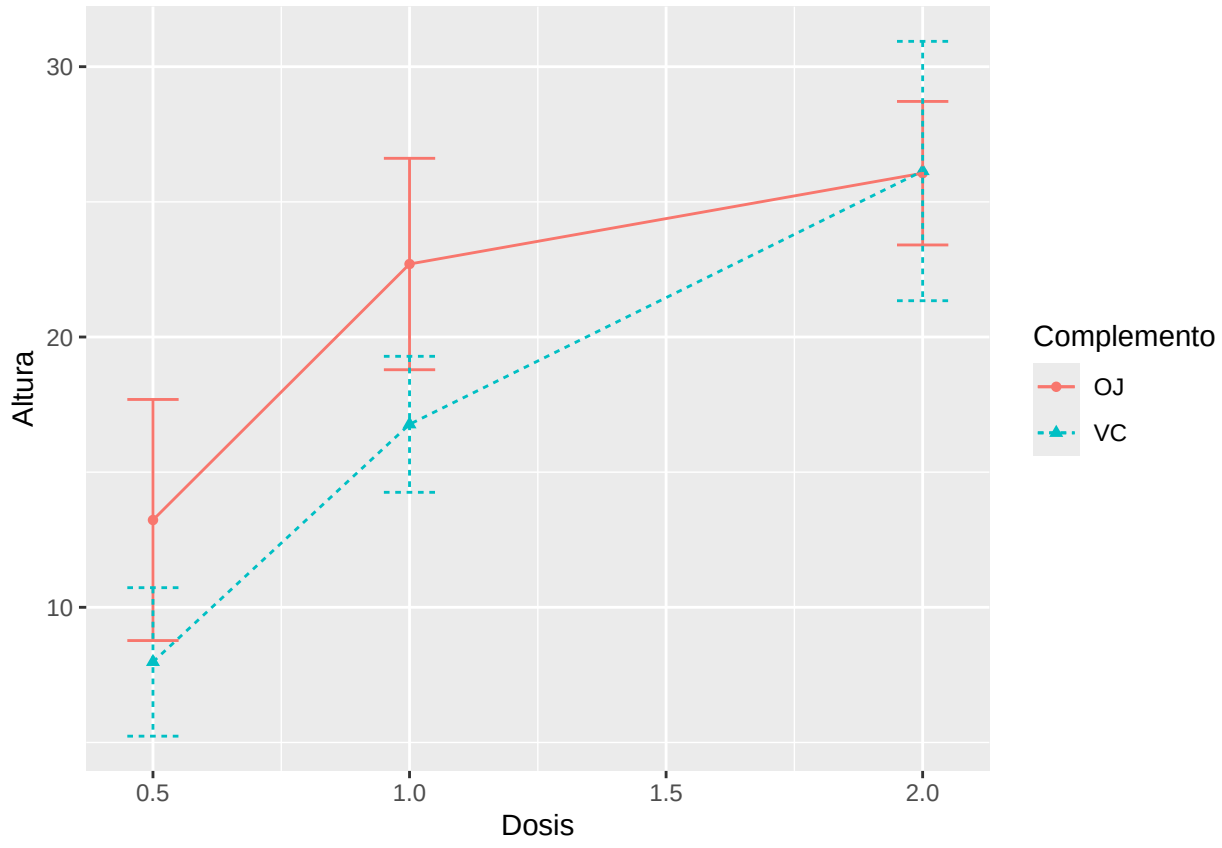
```
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complement
  geom_point(aes(shape = Complemento)) + geom_line(aes(linetype = Complemento)) +
  geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura))
```




```
ggplot(data = Experimento, mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento
  geom_point(aes(shape = Complemento)) + geom_line(aes(linetype = Complemento)) +
  geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura), width = 0.1)
```

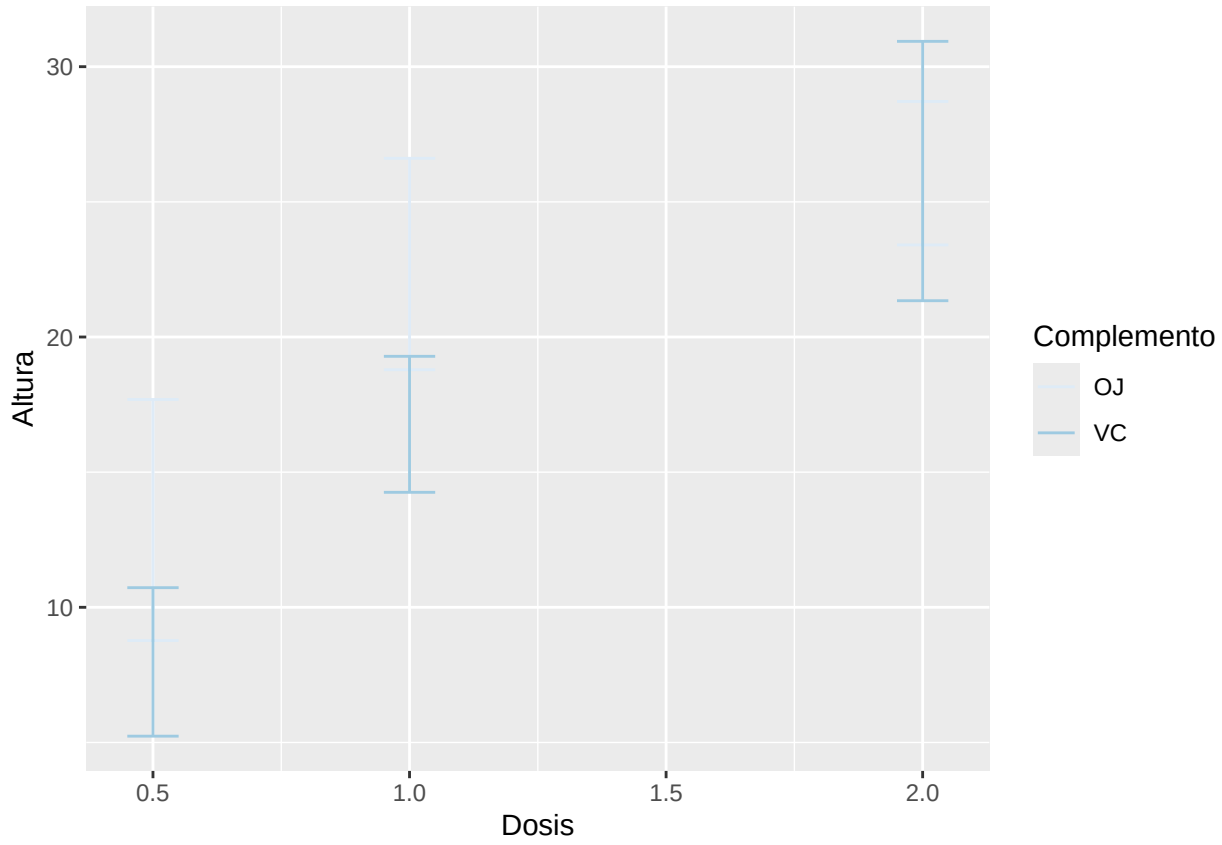



```
# En este caso, cambia la linea de las barras de error, ya que es el general 'ggplot', que tiene la con
ggplot(data = Experimento,
       mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento, shape = Complemen
  geom_point() +
  geom_line() +
  geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura), width = 0.1)
```

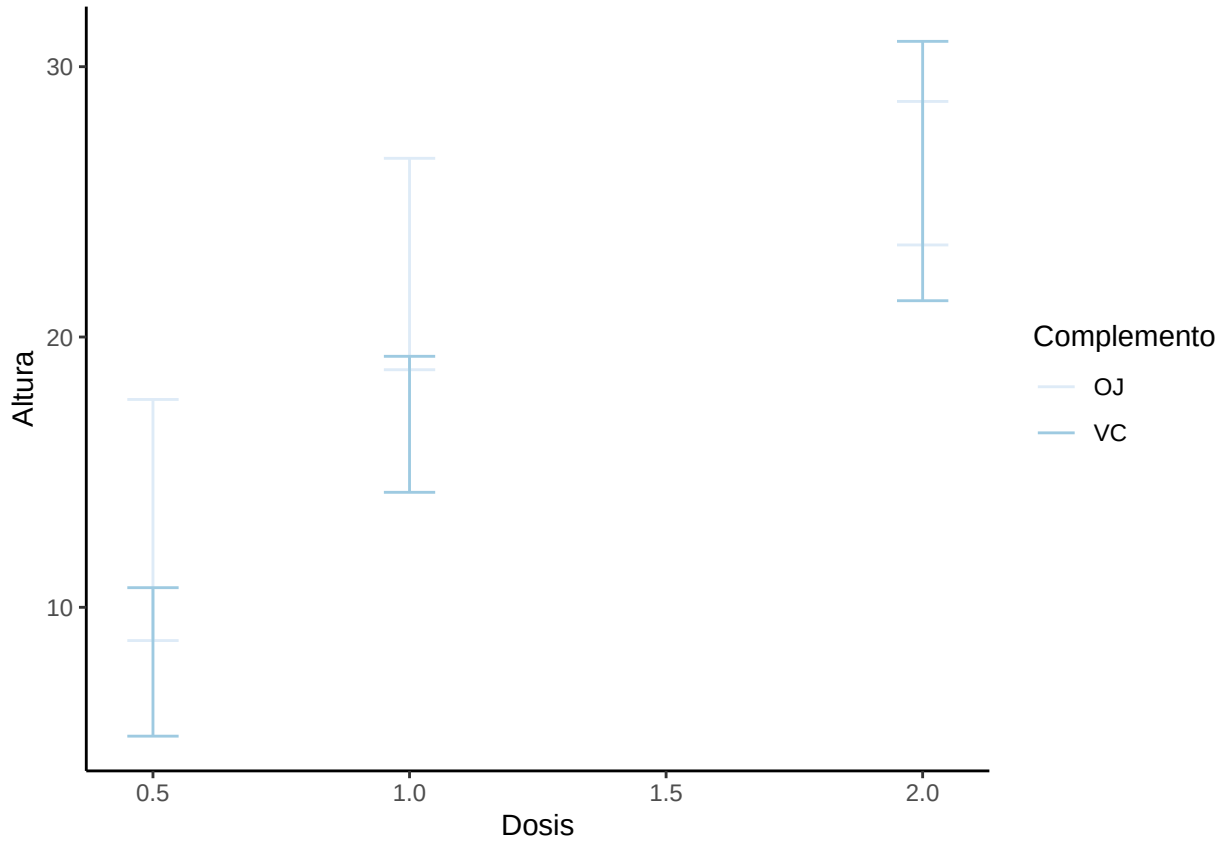


- Crea otro gráfico, como el anterior (**idéntico**), pero en este caso no puedes incluir el tipo de línea y el tipo de punto en el aes() de la capa ggplot. Además queremos que la paleta de colores sea en tonos de azules.

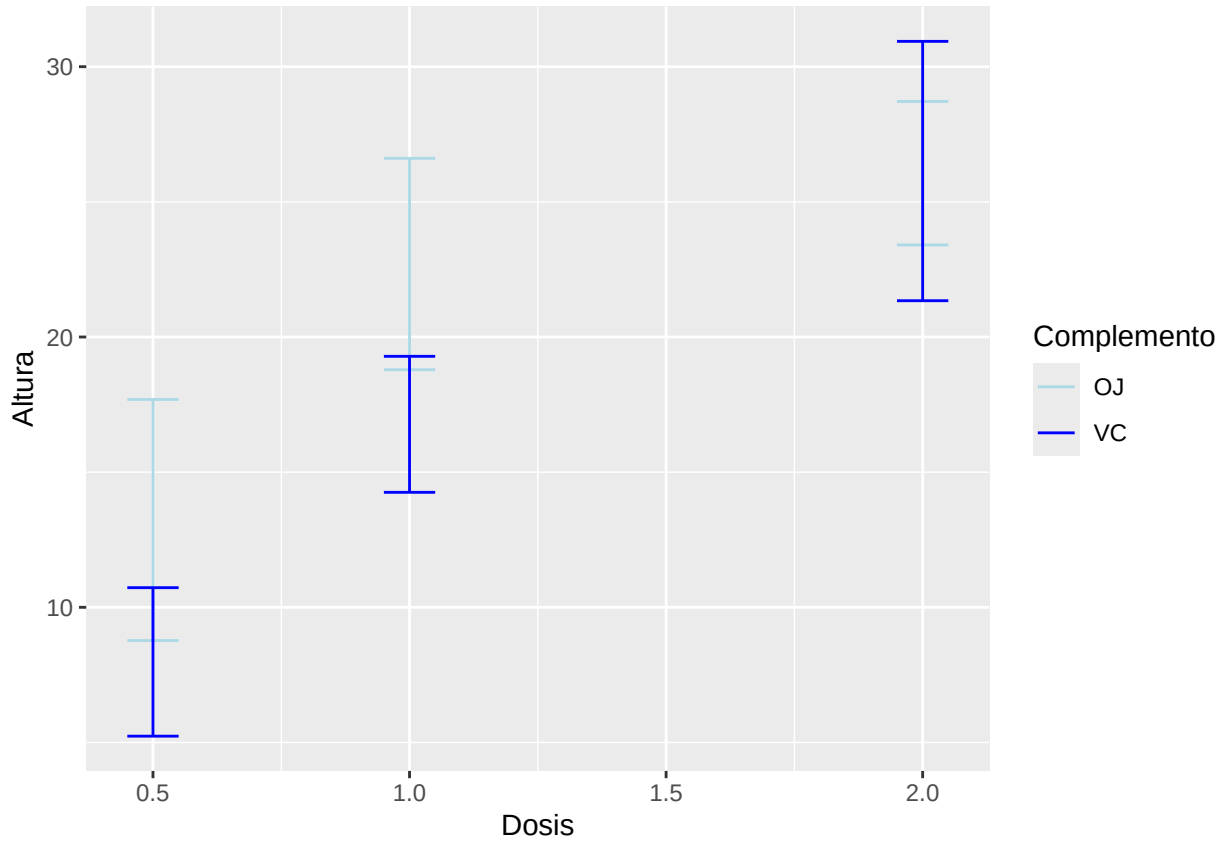
```
ggplot(data = Experimento,
       mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento)) +
  geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura), width = 0.1) +
  # theme_classic() +
  scale_color_brewer(palette = 'Blues') # Colores muy claros
```




```
# Haciendo uso de theme, se evidencia mejor
ggplot(data = Experimento,
       mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento)) +
  geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura), width = 0.1) +
  theme_classic() +
  scale_color_brewer(palette = 'Blues')
```




```
# También se puede
g5 <- ggplot(data = Experimento,
       mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento)) +
  geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura), width = 0.1) +
  # theme_classic() +
  # scale_color_brewer(palette = 'Blues')
  scale_color_manual(values = c("lightblue", "blue"))
g5
```




```
# ggplot(data = Experimento,
#        mapping = aes(x = Dosis, y = Altura, group = Complemento, color = Complemento)) +
#   geom_errorbar(aes(ymin = Altura - Error_Altura, ymax = Altura + Error_Altura), width = 0.1) +
#   # theme_classic() +
#   # scale_color_brewer(palette = 'Blues')
#   # scale_color_manual(values = c("lightblue", "blue"))
#   scale_color_viridis_d() # la letra _d, es para valores discretos
```

- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio5.pdf`.

```
n_salida <- '5'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
g5
dev.off()
```

```
## pdf
##   2
```

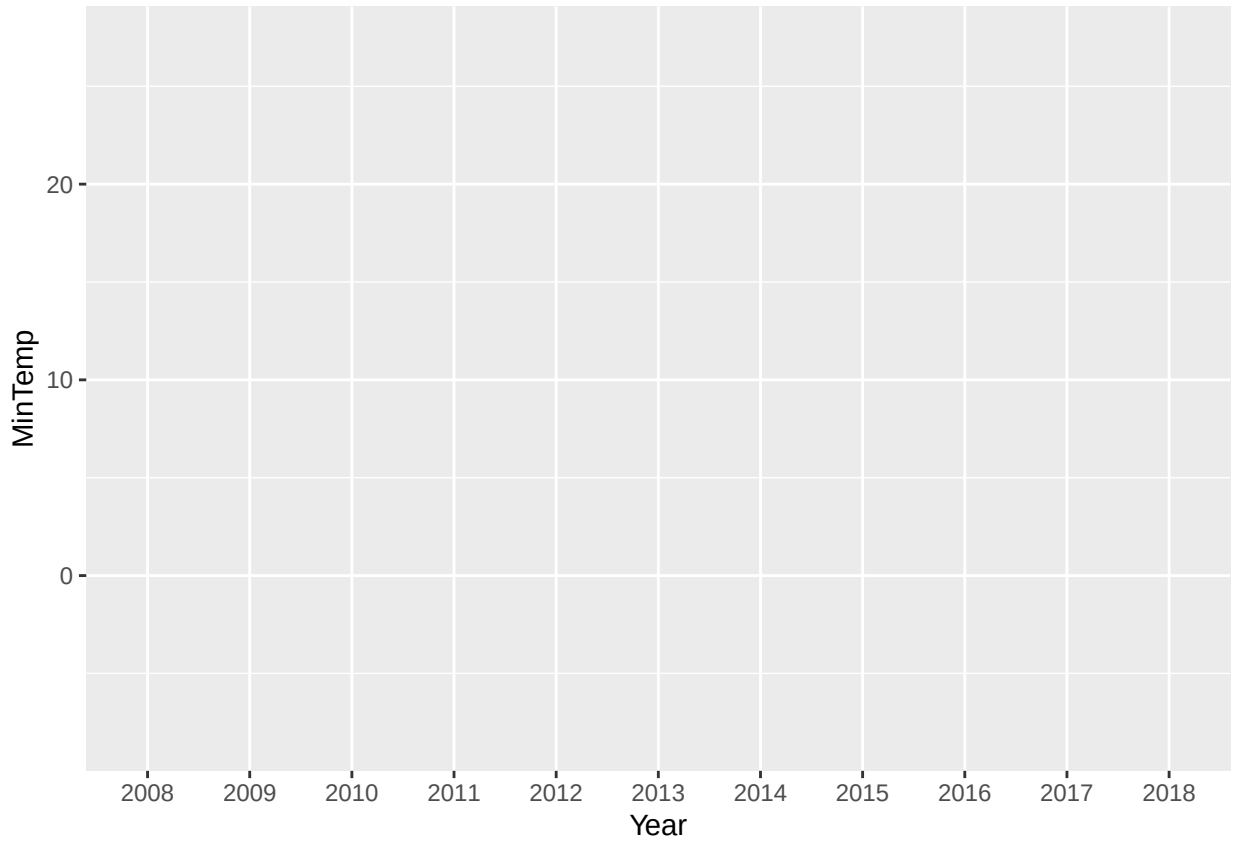
Ejercicio 6

- Carga el fichero `weatherAUS_distributions.RData`.

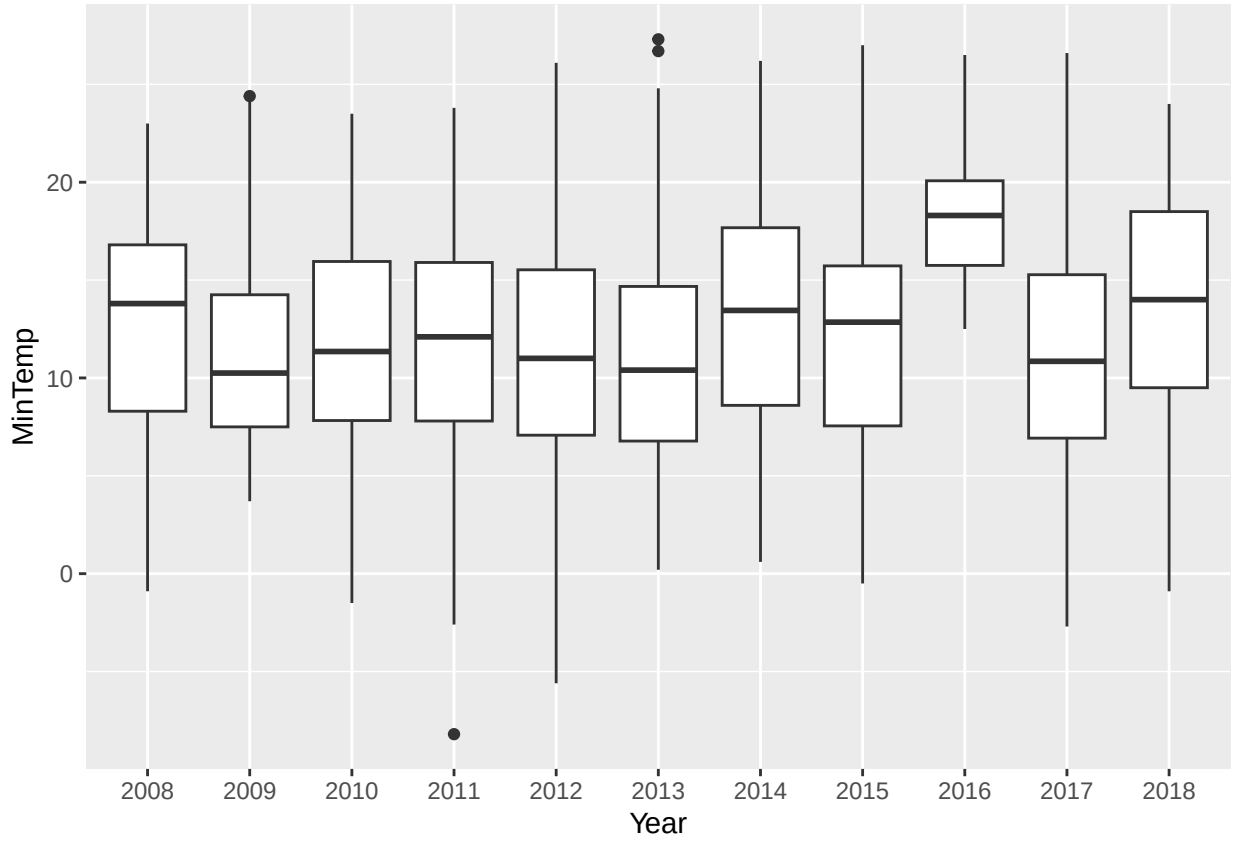
```
f5 <- paste0(carpeta, 'weatherAUS_distributions.RData')
load(f5)
```

- Crea un gráfico en el que estén los boxplots de la variable `MinTemp` para cada uno de los años presentes en la variable `Year` del dataset.

```
g6 <- ggplot(data = ds, mapping = aes(x = Year, y = MinTemp))
g6
```

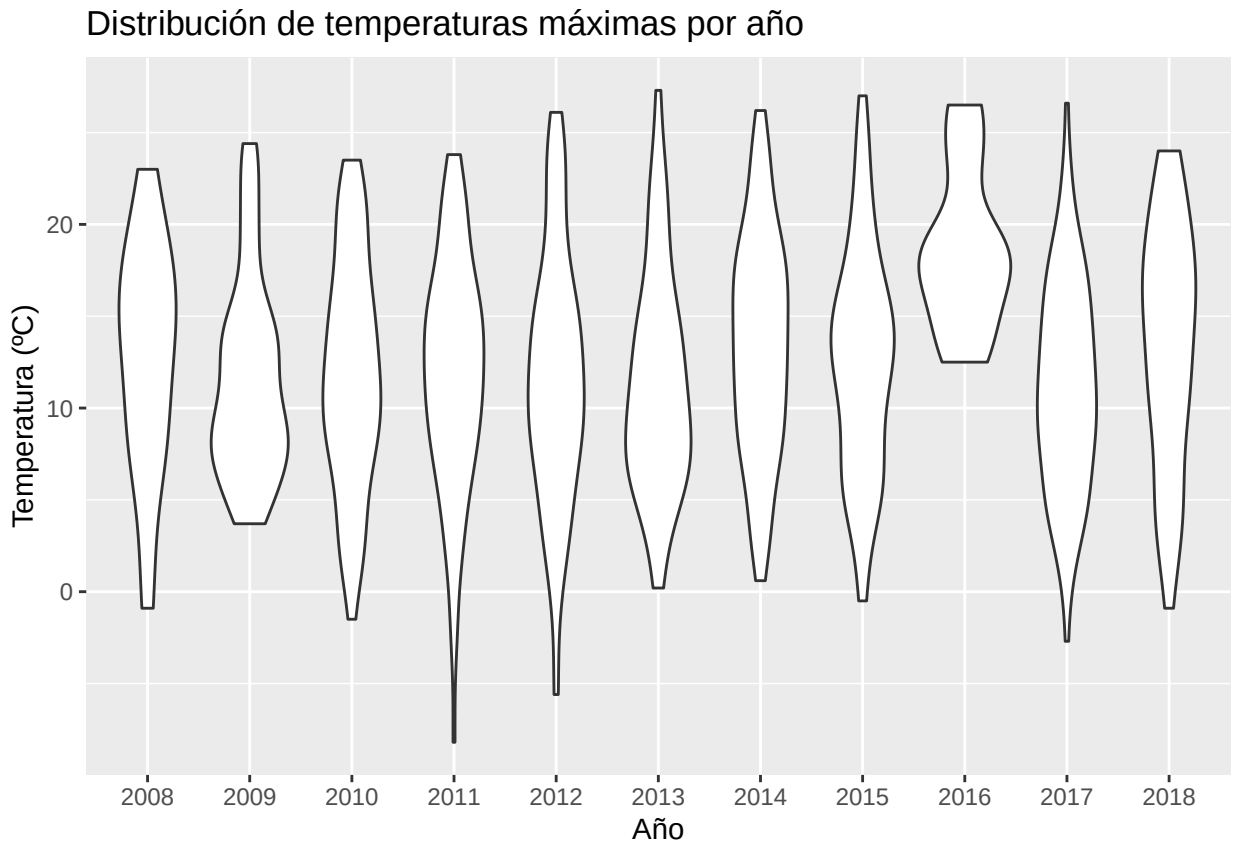



```
g6 + geom_boxplot()
```

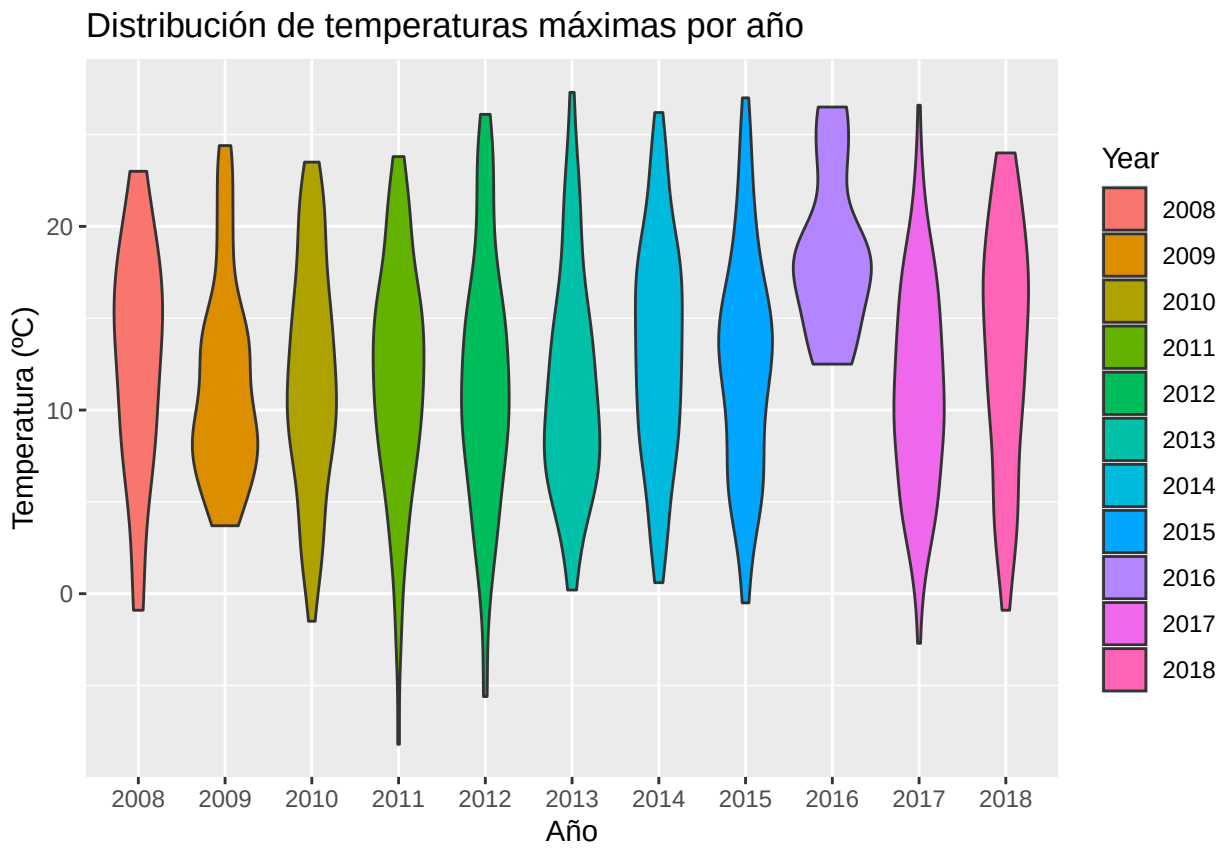


- Crea un gráfico en el que estén los gráficos de violín de la variable `MaxTemp` para cada uno de los años presentes en la variable `Year` del dataset. Añade en el mismo gráfico una capa de etiquetas de título (Distribución de temperaturas máximas por año), eje x (Año) y eje y (Temperatura (ºC)). Colorea el interior de los violines según la la variable `Year`.

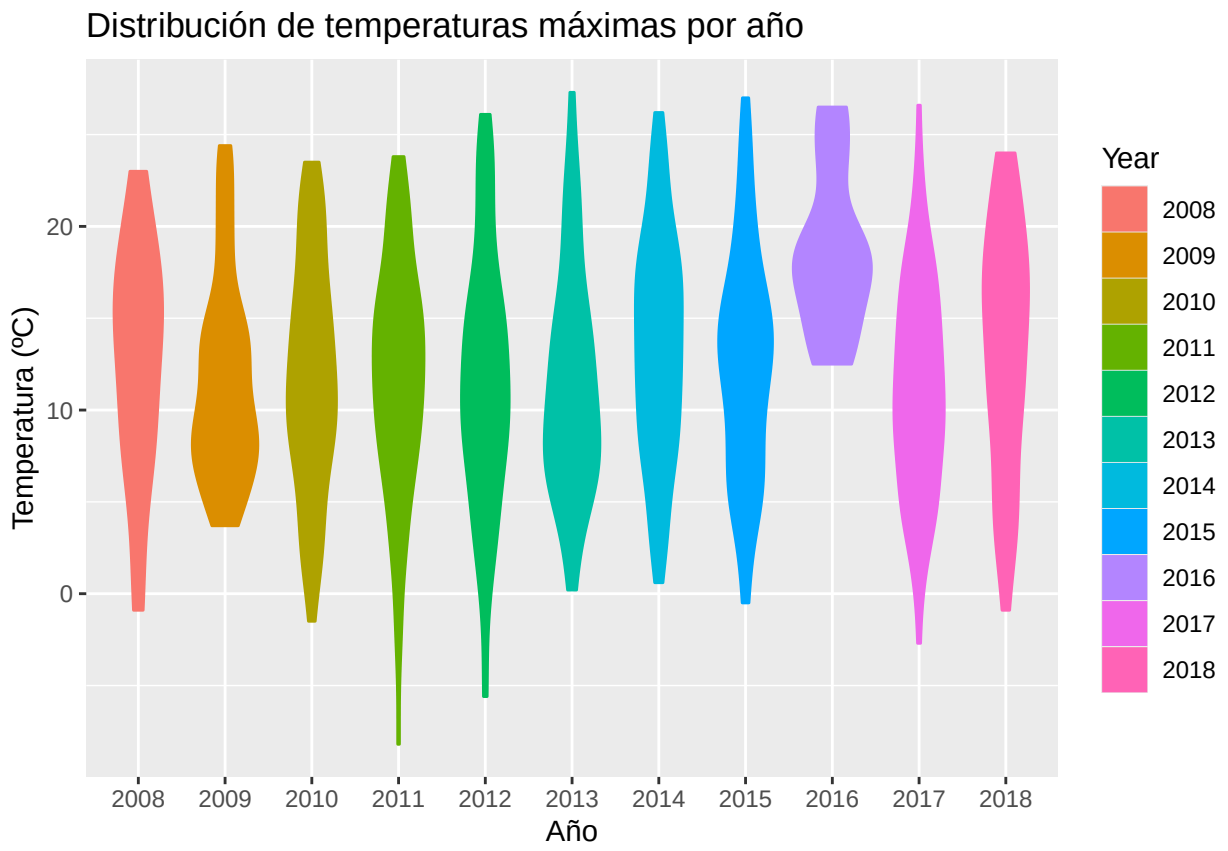
```
# Si el color no esta en ggplot y geom_violin no tiene fill
ggplot(data = ds, mapping = aes(x = Year, y = MinTemp)) +
  geom_violin() +
  ggtitle(label = 'Distribución de temperaturas máximas por año') +
  xlab(label = 'Año') + ylab(label = 'Temperatura (ºC)')
```



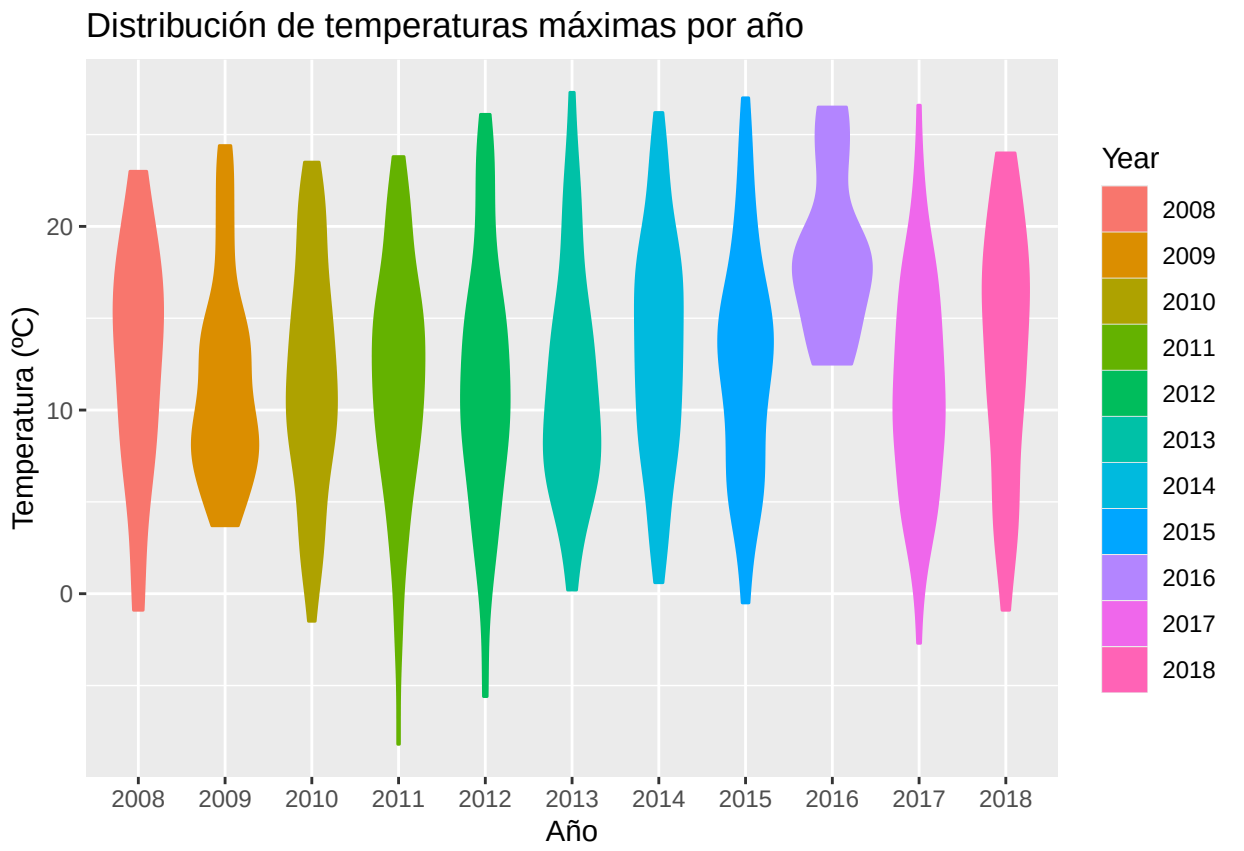

```
# Si ggplot no tiene el color, pero violin si tiene fill
ggplot(data = ds, mapping = aes(x = Year, y = MinTemp)) +
  geom_violin(aes(fill = Year)) +
  ggtitle(label = 'Distribución de temperaturas máximas por año') +
  xlab(label = 'Año') + ylab(label = 'Temperatura (ºC)')
```




```
# Colo esta en ggplot y violin
ggplot(data = ds, mapping = aes(x = Year, y = MinTemp, color = Year)) +
  geom_violin(aes(fill = Year)) +
  ggtitle(label = 'Distribución de temperaturas máximas por año') +
  xlab(label = 'Año') + ylab(label = 'Temperatura (ºC)')
```

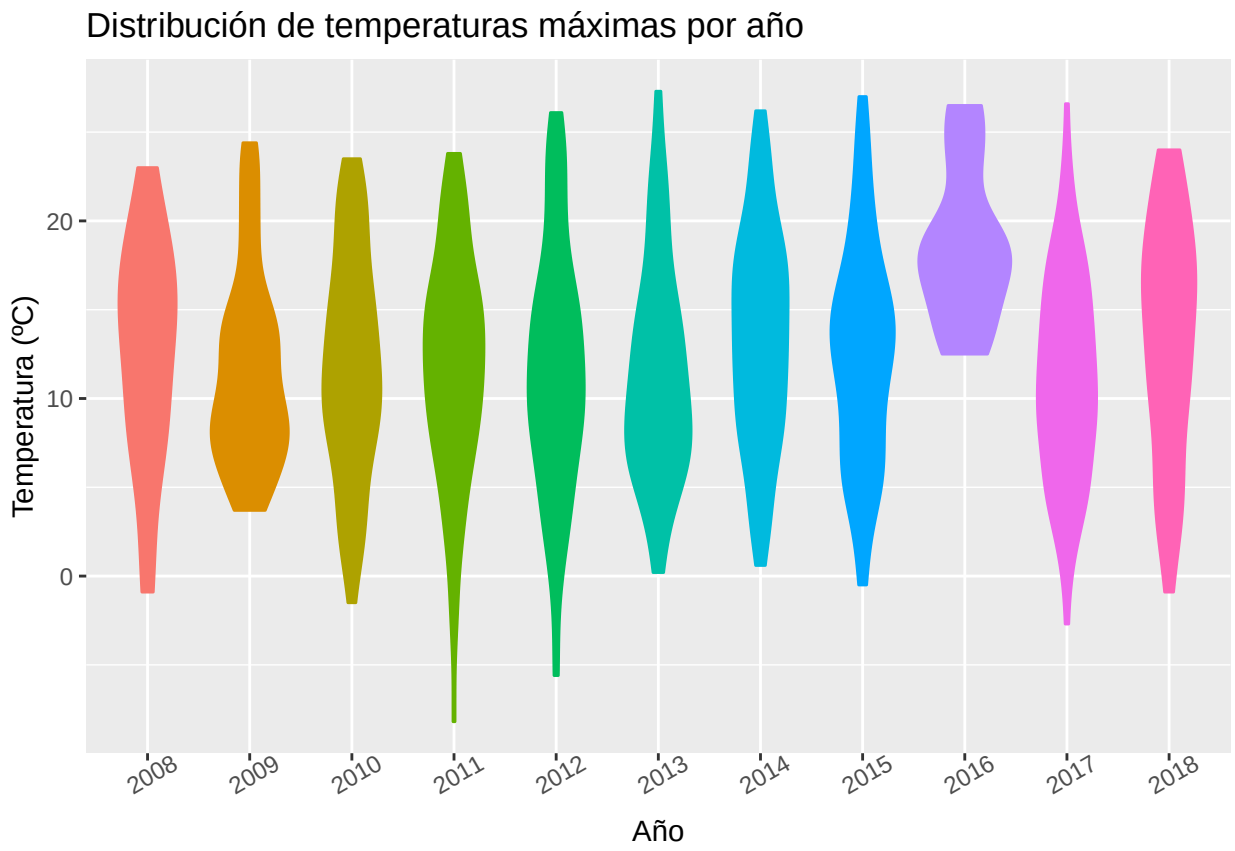



```
# Esto es lo mismo al anterior. Cambia el llamado de Labs
ggplot(data = ds, mapping = aes(x = Year, y = MinTemp, color = Year)) +
  geom_violin(aes(fill = Year)) +
  labs(title = 'Distribución de temperaturas máximas por año',
       x = 'Año',
       y = 'Temperatura (ºC)')
```



- En el gráfico anterior elimina la leyenda y coloca los años en el eje x a 45 grados de inclinación. Busca información en la red sobre cómo hacer esto.

```
g6 <- ggplot(data = ds, mapping = aes(x = Year, y = MinTemp, color = Year)) +
  geom_violin(aes(fill = Year)) +
  labs(title = 'Distribución de temperaturas máximas por año',
       x = 'Año',
       y = 'Temperatura (ºC)') +
  theme(legend.position = 'none',
        axis.text.x = element_text(angle = 30))
g6
```



- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio6.pdf`.

```
n_salida <- '6'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
g6
dev.off()
```

```
## pdf
##   2
```

Ejercicio 7

- Descarga de la url "http://assets.datacamp.com/blog_assets/chol.txt" el dataset de datacamp que utilizarás para este ejercicio utilizando la función **read.table()**.

```
# URL donde se descargara el fichero
url_ej <- 'http://assets.datacamp.com/blog_assets/chol.txt'

# Almacenamiento del fichero
fichero <- paste0(carpeta, 'chol.txt')
```

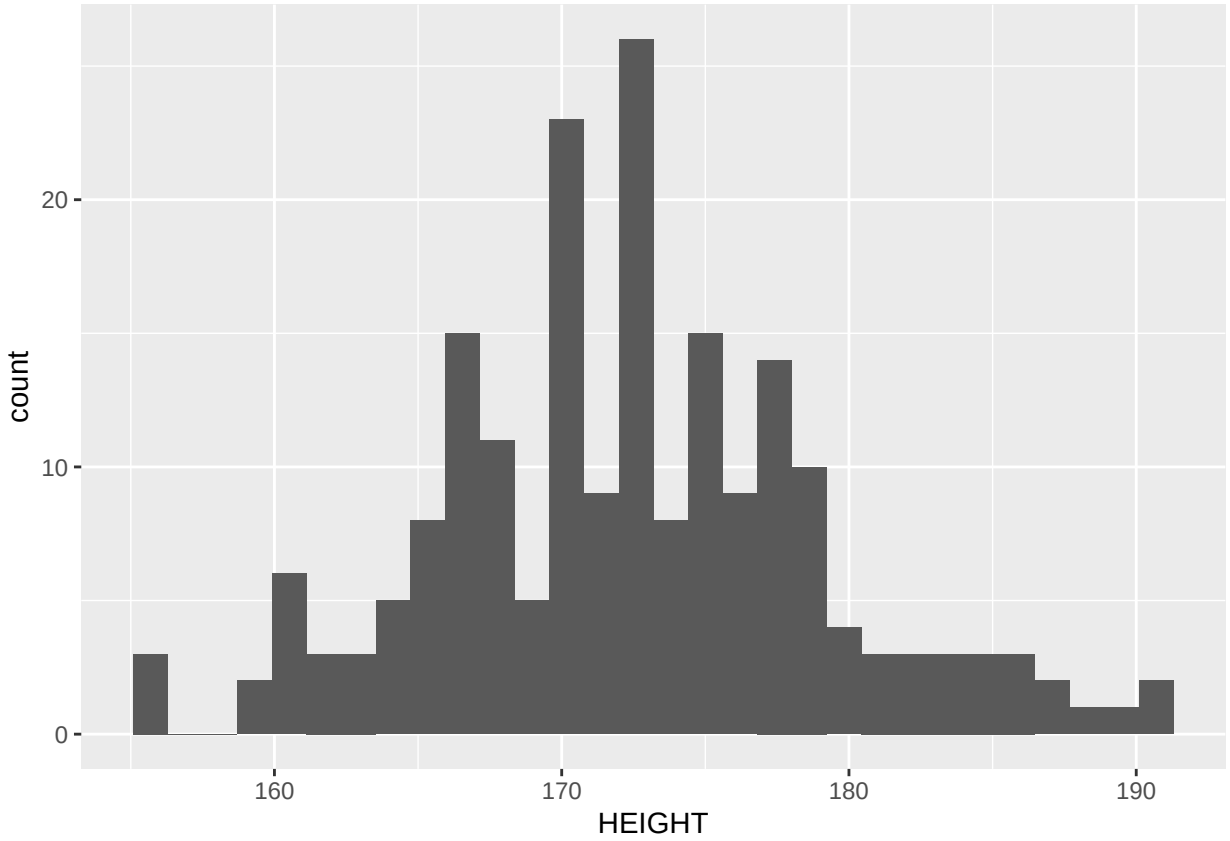
```r
# Descarga
download.file(url = url_ej, destfile = fichero)

# Leemos el fichero
df <- read.table(file = fichero, header = TRUE)
head(df)
```

```
##   AGE HEIGHT WEIGHT CHOL  SMOKE BLOOD  MORT
## 1  20    176     77  195 nonsmo     b alive
## 2  53    167     56  250 sigare     o  dead
## 3  44    170     80  304 sigare     a  dead
## 4  37    173     89  178 nonsmo     o alive
## 5  26    170     71  206 sigare     o alive
## 6  41    165     62  284 sigare     o alive
```

- Representa el histograma de la variable `HEIGHT` del dataframe cargado del fichero anterior.

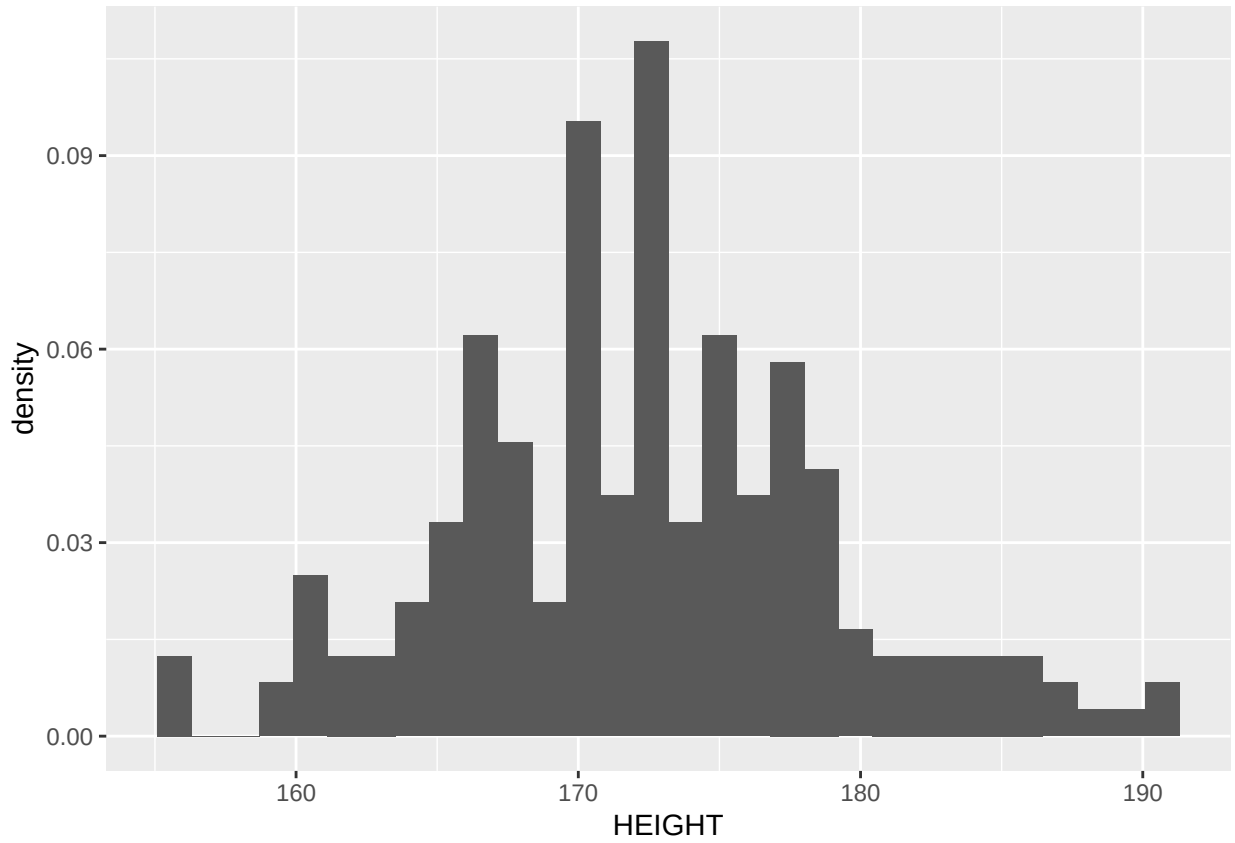
```r
# Sale un mensaje, indicando que por defecto, bins = 30
# Esto es porque no se especifica bins o binwidth

# count - por defecto
# Muestra cuántos datos caen dentro de cada intervalo (bin)
# Es una frecuencia absoluta
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram()
```

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

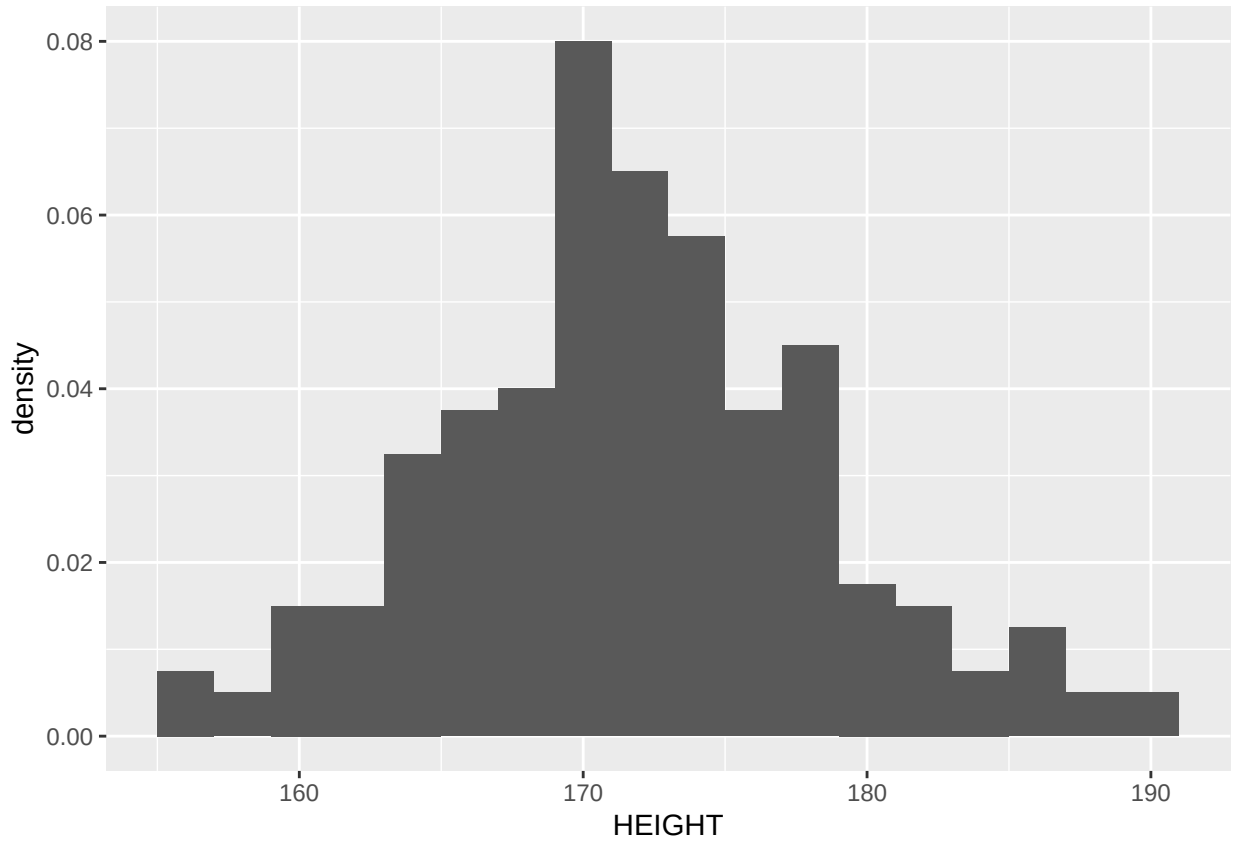



```
# density
# Muestra la densidad de probabilidad
# Las áreas de las barras suman 1, no los valores en el eje y
# Se usa para comparar con una función de densidad (como una curva normal).
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(aes(y = after_stat(density))) # Eje y muestre la densidad, en lugar del conteo
```

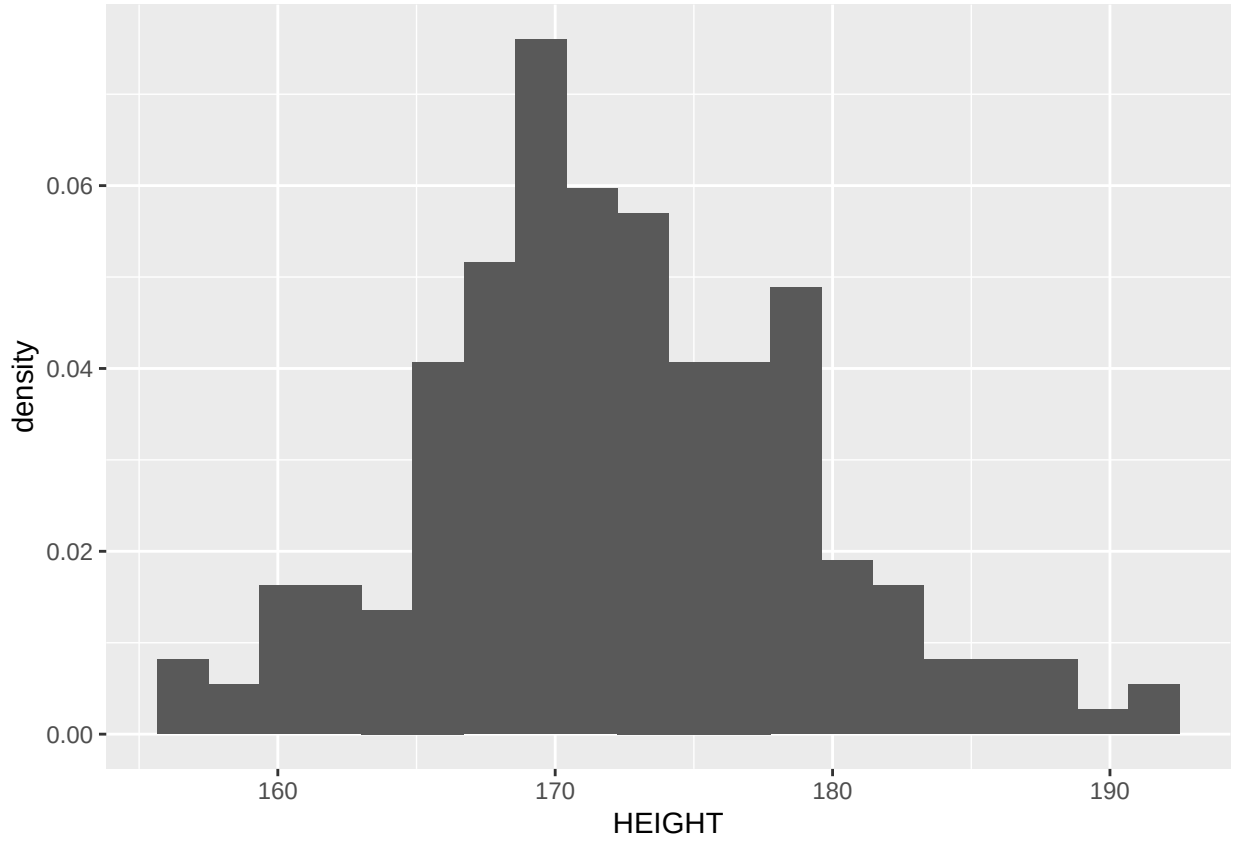
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



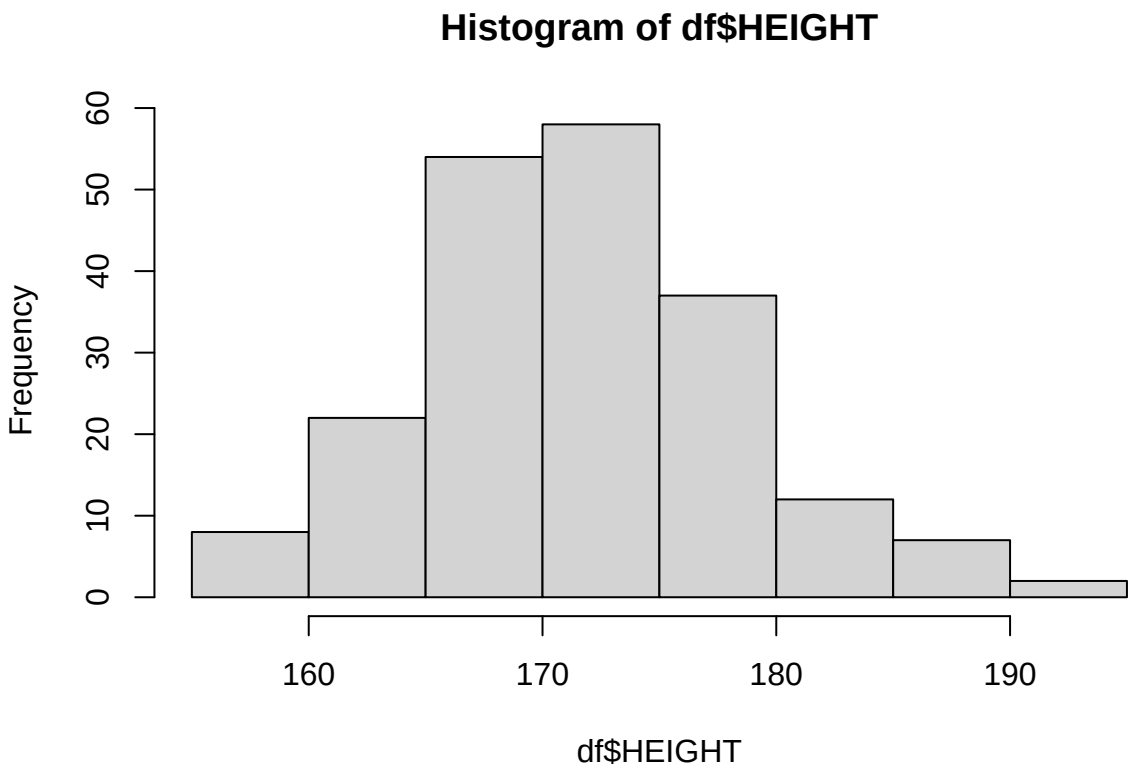

```
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(aes(y = after_stat(density)), binwidth = 2) # binwidth = 2 define el ancho de la barra
```




```
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(aes(y = after_stat(density)), bins = 20) # bins = 20 define el total de barras
```



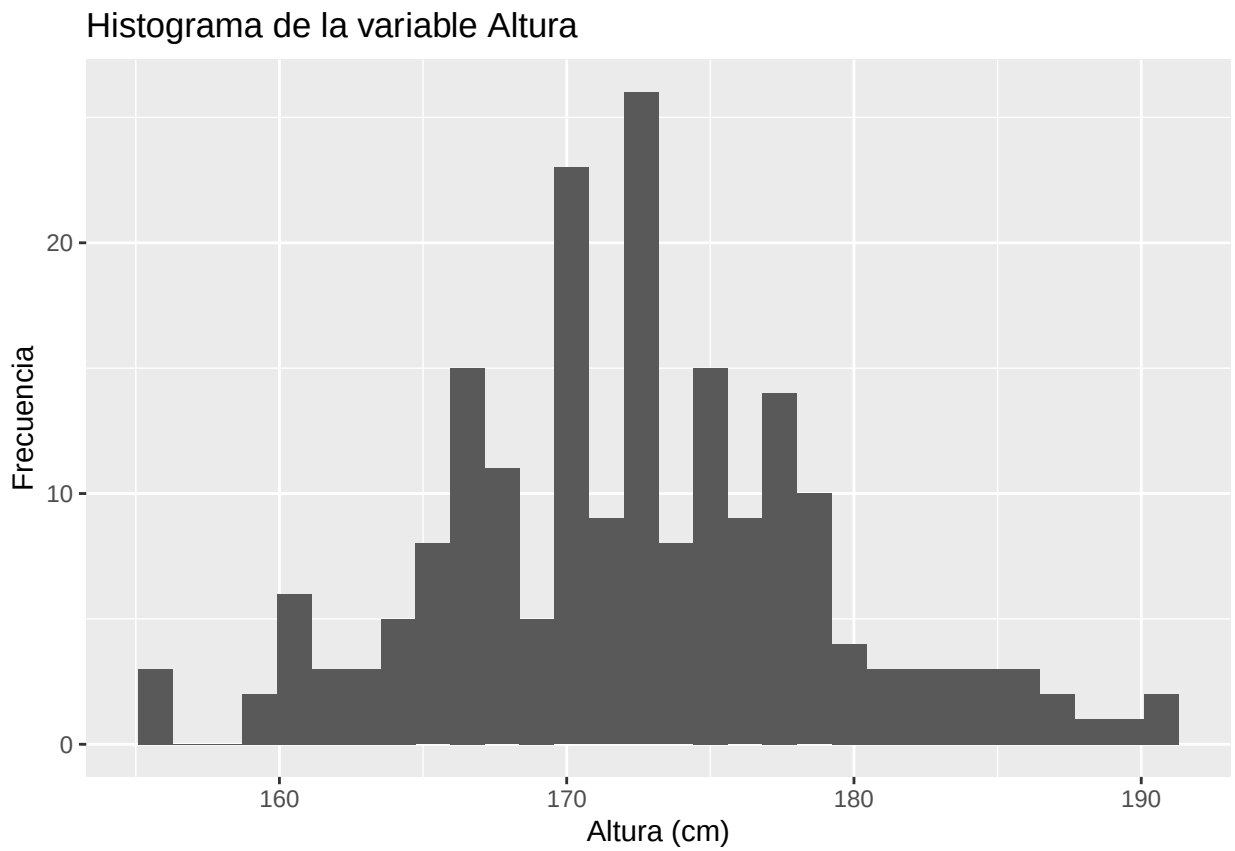

```r
# Otro ejemplo de histograma - sin ggplot
# breaks: número o posición de los cortes.
# freq: TRUE (frecuencia absoluta) o FALSE (densidad).
# col: color de las barras.
# border: color del borde.
hist(x = df$HEIGHT)
```



- Añade título ("Histograma de la variable Altura"), una etiqueta en el eje x ("Altura (cm)") y una etiqueta en el eje y ("Frecuencia").

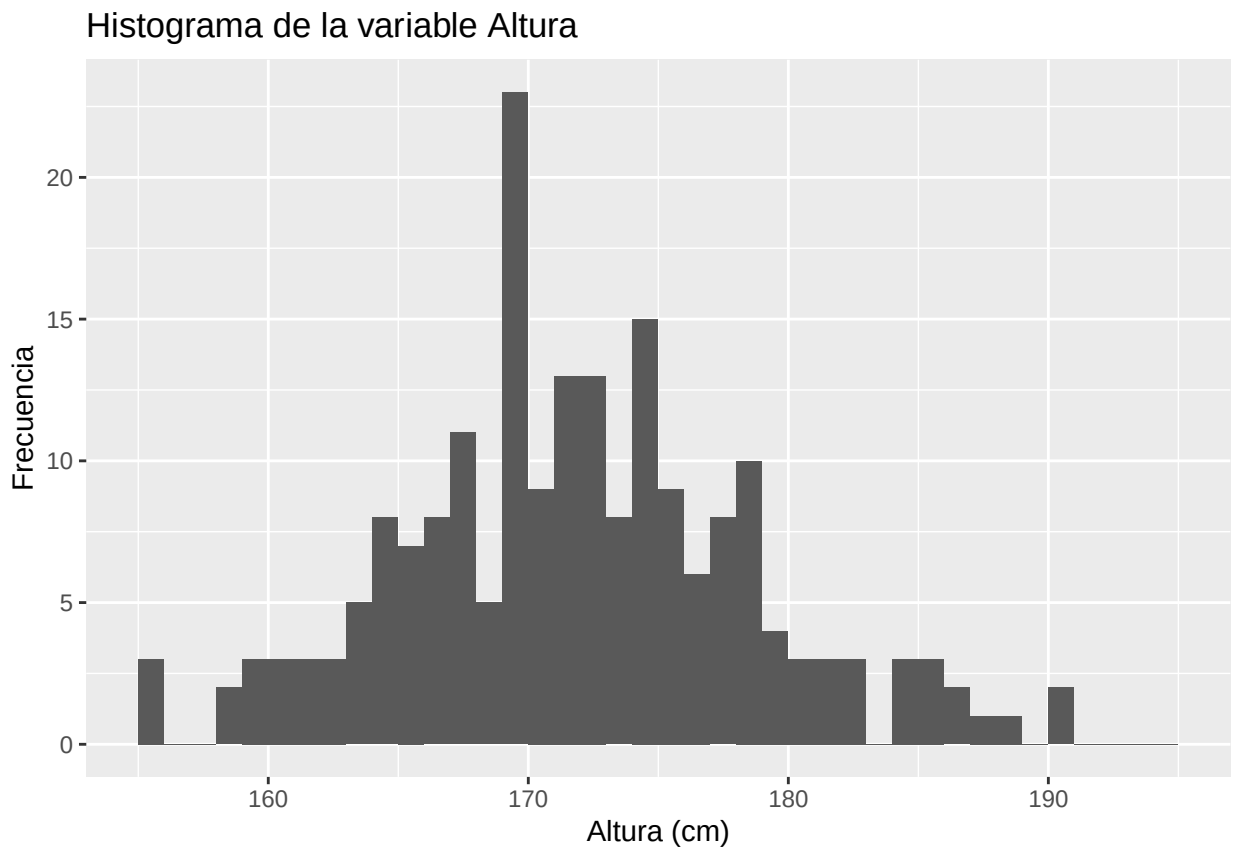
```
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram() +
  labs(title = 'Histograma de la variable Altura',
       x = 'Altura (cm)',
       y = 'Frecuencia')
```

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



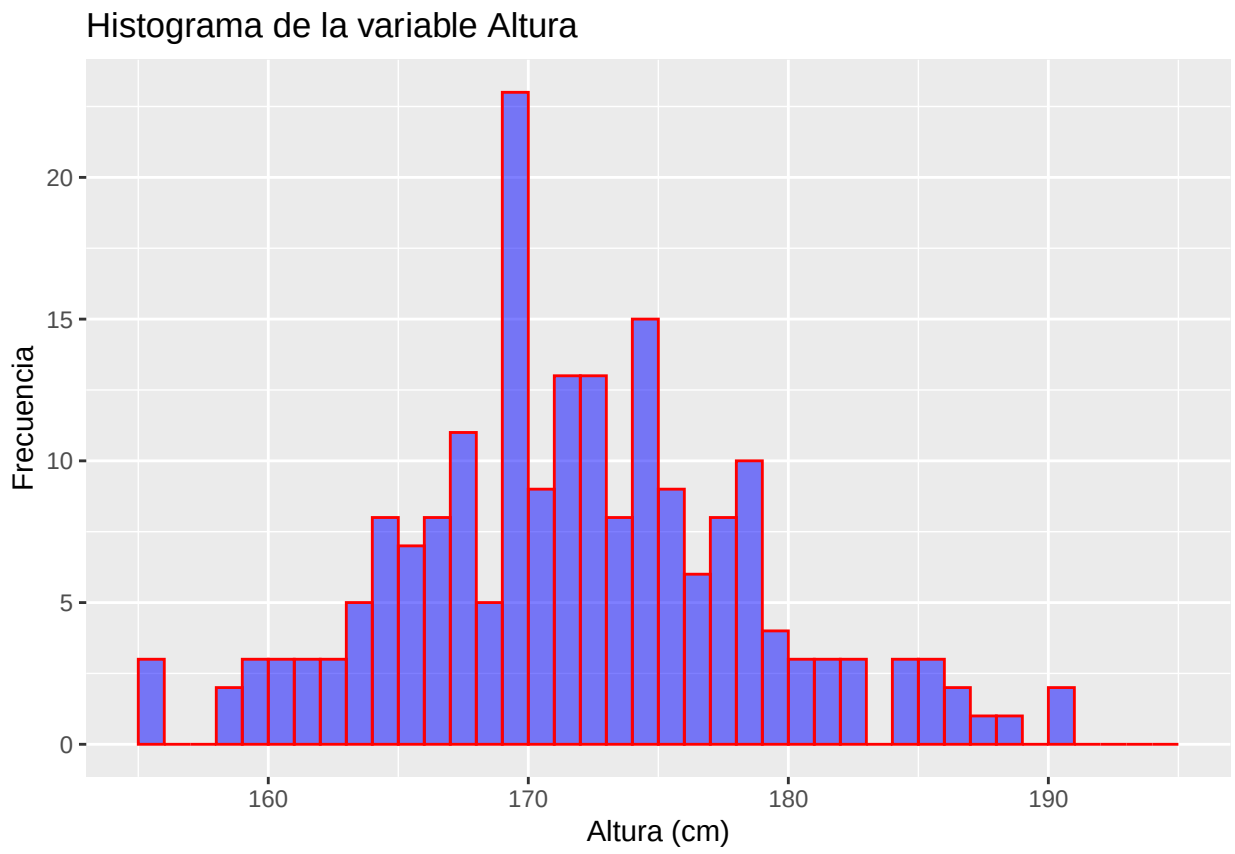
- Ajusta el rango de representación entre 155 y 195, además consigue que el histograma tenga 20 *bins*.

```
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(breaks = seq(155, 195),
                 bins = 20) +
  labs(title = 'Histograma de la variable Altura',
       x = 'Altura (cm)',
       y = 'Frecuencia')
```



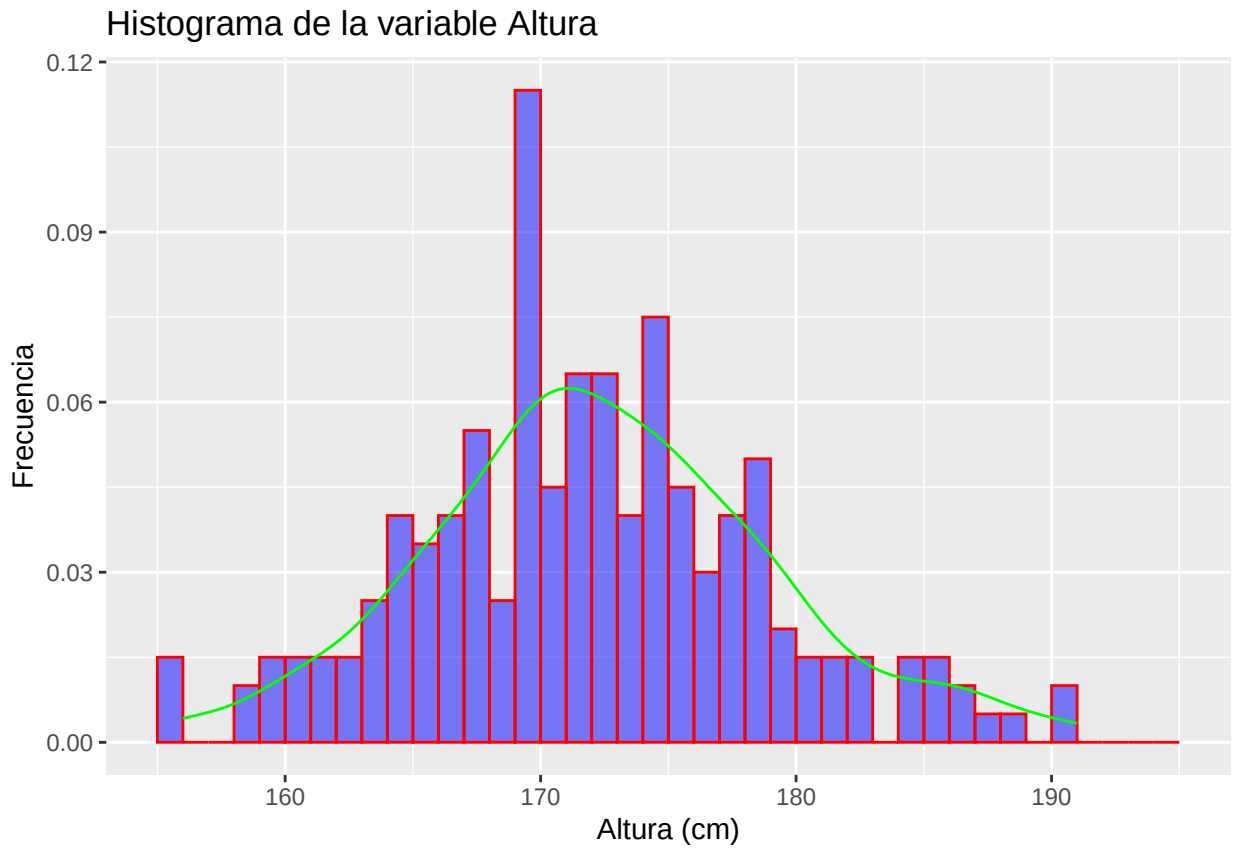
- Haz que las barras del histograma sean de color azul con una grado de transparencia del 50% y el contorno de color rojo.

```
# Los parametros están fuera de aes, porque es 'general'. No se asigna a una variable especifica
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(breaks = seq(155, 195),
                 bins = 20,
                 color = 'red',
                 fill = 'blue',
                 alpha = 0.5) +
  labs(title = 'Histograma de la variable Altura',
       x = 'Altura (cm)',
       y = 'Frecuencia')
```



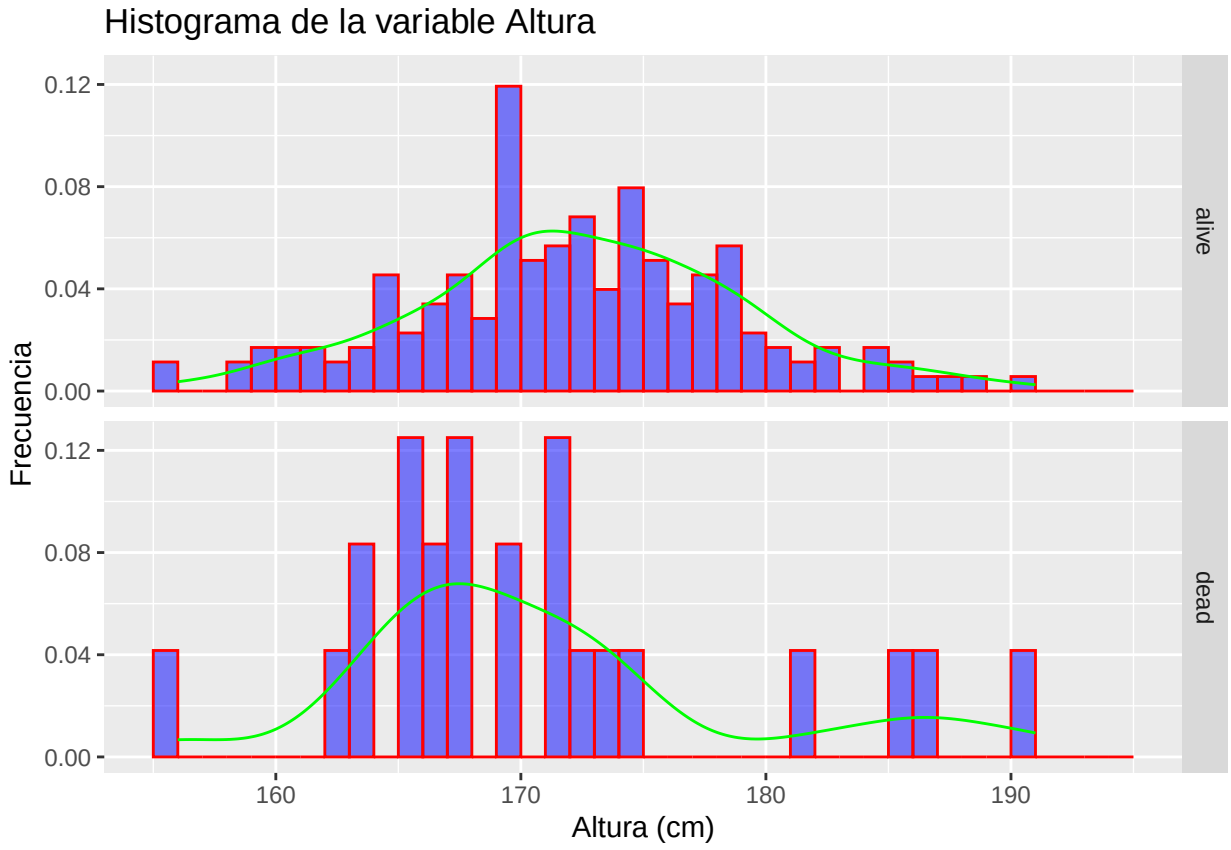
- Añade la curva de distribución estimada de color verde.

```
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(aes(y = after_stat(density)),
                 breaks = seq(155, 195),
                 bins = 20,
                 color = 'red',
                 fill = 'blue',
                 alpha = 0.5) +
  labs(title = 'Histograma de la variable Altura',
       x = 'Altura (cm)',
       y = 'Frecuencia') +
  geom_density(color = 'green') # Para usar geom_density, debe usarse density en aes
```

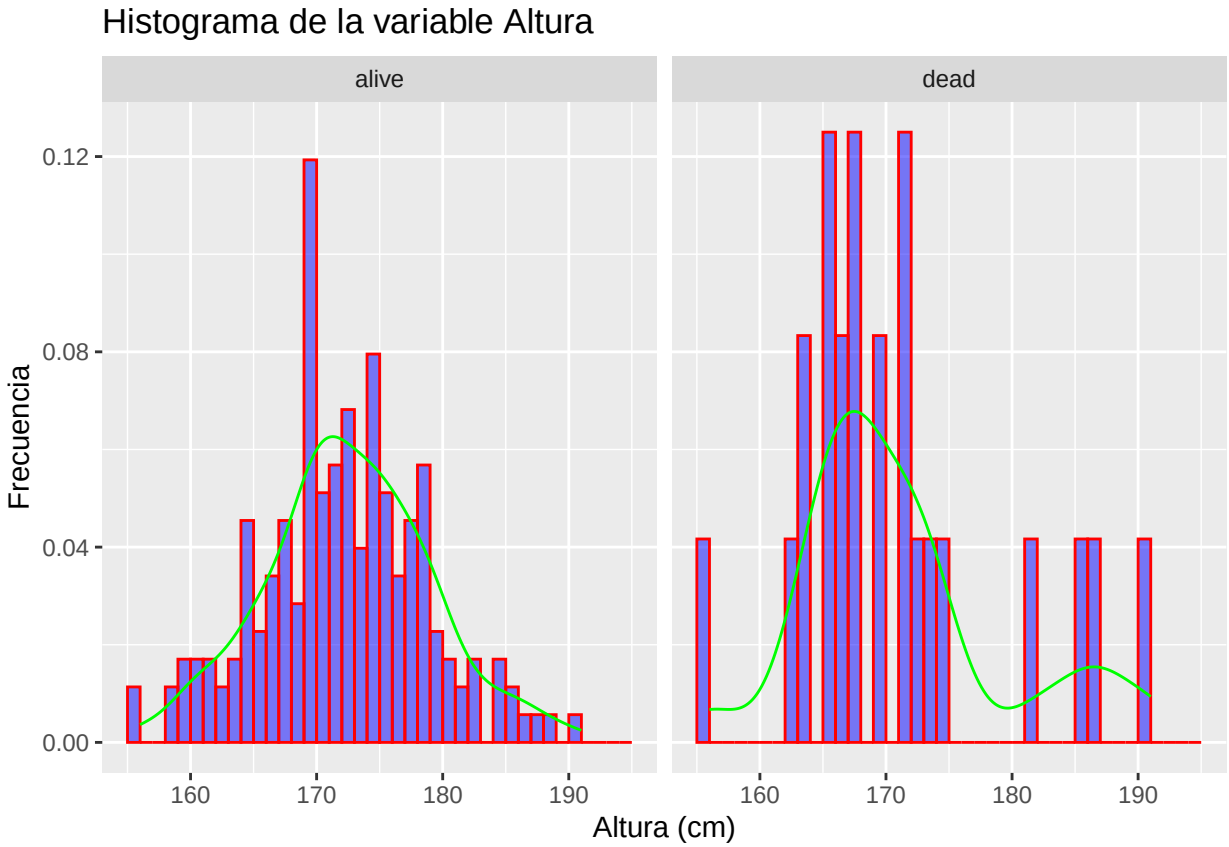


- Representa el mismo histograma del apartado anterior para cada valor de la variable `MORT`, recuerda utilizar la función **facet_grid()**.

```
ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(aes(y = after_stat(density)),
                 breaks = seq(155, 195),
                 bins = 20,
                 color = 'red',
                 fill = 'blue',
                 alpha = 0.5) +
  labs(title = 'Histograma de la variable Altura',
       x = 'Altura (cm)',
       y = 'Frecuencia') +
  geom_density(color = 'green') +
  facet_grid(MORT~.) # Horizontal
```




```
g7 <- ggplot(data = df, aes(x = HEIGHT)) +
  geom_histogram(aes(y = after_stat(density)),
                 breaks = seq(155, 195),
                 bins = 20,
                 color = 'red',
                 fill = 'blue',
                 alpha = 0.5) +
  labs(title = 'Histograma de la variable Altura',
       x = 'Altura (cm)',
       y = 'Frecuencia') +
  geom_density(color = 'green') +
  facet_grid(.~MORT) # vertical
g7
```



- Almacena en formato **pdf** el gráfico resultante en un fichero denominado `Ejercicio7.pdf`.

```
n_salida <- '7'
f_salida <- paste0(figuras, 'Ejercicio', n_salida, '.pdf')

# Grafico resultante
pdf(f_salida)
g7
dev.off()
```

```
## pdf
##   2
```