Programación en R para el análisis de datos

Visualizacion de datos con ggplot2

Nicolás Schmidt

mail::nschmidt@cienciassociales.edu.uy
GitHub::@Nicolas-Schmidt

Departamento de Ciencia Poltica
Facultad de Ciencias Sociales

Ruta

Visualización

ggplot2

Histogramas y densidades

Barras

Boxplot

Lineas

Combinar gráficos

Visualización

Cuarteto de Anscombe

```
anscombe
```

```
##    x1 x2 x3 x4    y1   y2    y3    y4
## 1  10 10 10  8  8.04 9.14  7.46  6.58
## 2   8  8  8  8  6.95 8.14  6.77  5.76
## 3  13 13 13  8  7.58 8.74 12.74  7.71
## 4   9  9  9  8  8.81 8.77  7.11  8.84
## 5  11 11 11  8  8.33 9.26  7.81  8.47
## 6  14 14 14  8  9.96 8.10  8.84  7.04
## 7   6  6  6  8  7.24 6.13  6.08  5.25
## 8   4  4  4 19  4.26 3.10  5.39 12.50
## 9  12 12 12  8 10.84 9.13  8.15  5.56
## 10  7  7  7  8  4.82 7.26  6.42  7.91
## 11  5  5  5  8  5.68 4.74  5.73  6.89
```

```
apply(anscombe, 2, mean)
```

```
##       x1       x2       x3       x4       y1       y2       y3       y4
## 9.000000 9.000000 9.000000 9.000000 7.500909 7.500909 7.500000 7.500909
```

```
apply(anscombe, 2, sd)
```

```
##       x1       x2       x3       x4       y1       y2       y3       y4
## 3.316625 3.316625 3.316625 3.316625 2.031568 2.031657 2.030424 2.030579
```

Cuarteto de Anscombe

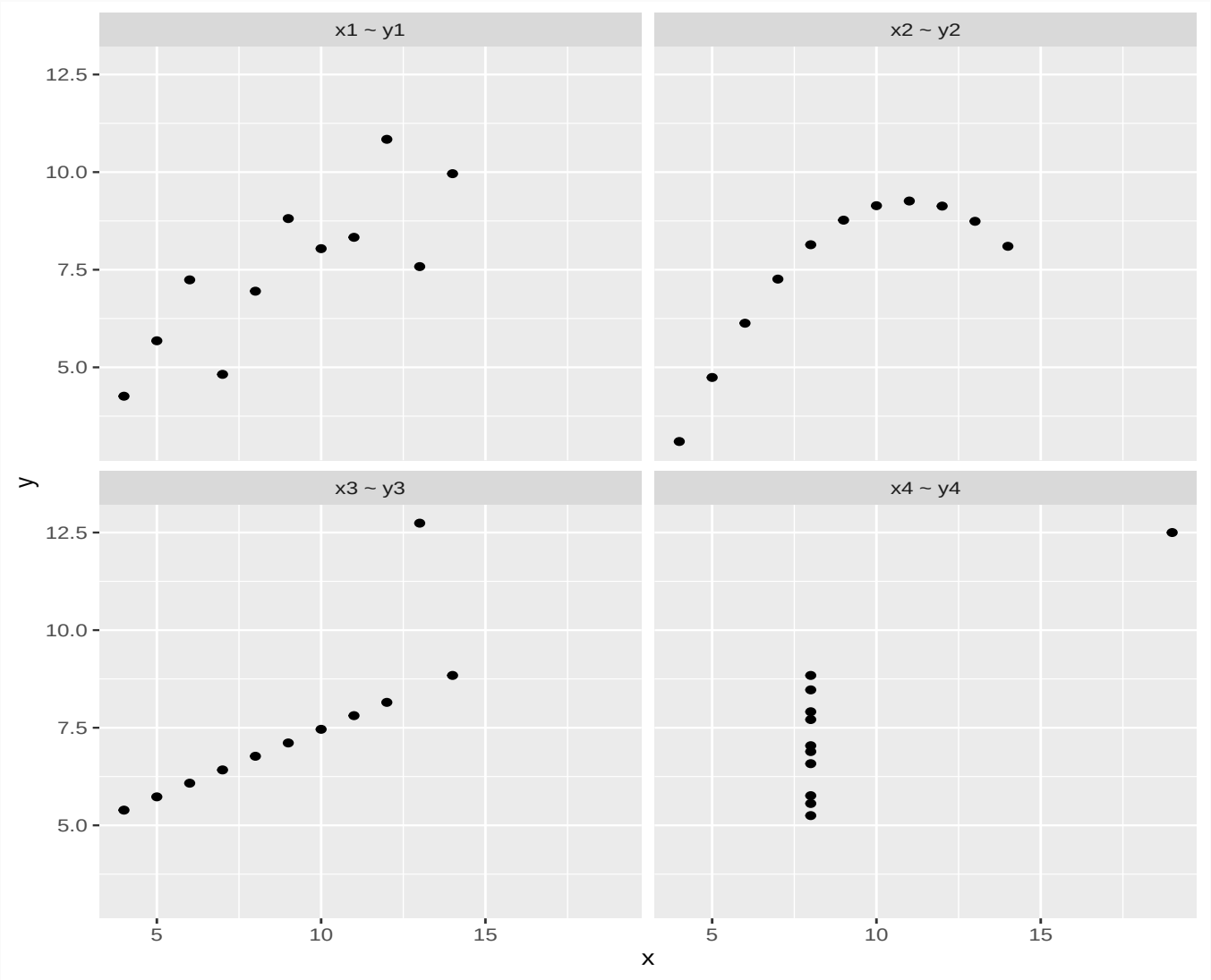
```r
ans <- data.frame(stack(anscombe), mod = rep(c('x', 'y'), each = 44))
ans <- unstack(ans[, -2])
ans$model <- rep(c("x1 ~ y1", "x2 ~ y2", "x3 ~ y3", "x4 ~ y4") , each = 11)

ans %>%
  split(.$model) %>%
  purrr::map(~ lm(x ~ y, data = .)) %>%
  purrr::map(summary) %>%
  purrr::map_dbl("r.squared") %>%
  round(digits = 4) %>%
  as.list()
```

```
## $`x1 ~ y1`
## [1] 0.6665
##
## $`x2 ~ y2`
## [1] 0.6662
##
## $`x3 ~ y3`
## [1] 0.6663
##
## $`x4 ~ y4`
## [1] 0.6667
```

Cuarteto de Anscombe

```
ggplot(ans, aes(x = x, y = y)) +
    geom_point() +
    facet_wrap(~model, ncol = 2)
```



datasauRus

```r
wow <- datasaurus_dozen %>% filter(dataset %in% c('dino', 'star', 'x_shape', 'circle'))

wow
```

```
## # A tibble: 568 x 3
##    dataset     x     y
##    <chr>   <dbl> <dbl>
##  1 dino     55.4  97.2
##  2 dino     51.5  96.0
##  3 dino     46.2  94.5
##  4 dino     42.8  91.4
##  5 dino     40.8  88.3
##  6 dino     38.7  84.9
##  7 dino     35.6  79.9
##  8 dino     33.1  77.6
##  9 dino     29.0  74.5
## 10 dino     26.2  71.4
## # ... with 558 more rows
```

```r
wow %>%
    group_by(dataset) %>%
    summarise(media_x = mean(x), media_y = mean(y), sd_x = sd(x), sd_y = sd(y))
```

```
## # A tibble: 4 x 5
##   dataset media_x media_y  sd_x  sd_y
##   <chr>     <dbl>   <dbl> <dbl> <dbl>
## 1 circle     54.3    47.8  16.8  26.9
## 2 dino       54.3    47.8  16.8  26.9
## 3 star       54.3    47.8  16.8  26.9
## 4 x_shape    54.3    47.8  16.8  26.9
```

datasauRus

```
wow %>%
    split(.$dataset) %>%
    purrr::map(~ lm(x ~ y, data = .)) %>%
    purrr::map(summary) %>%
    purrr::map_dbl("r.squared") %>%
    round(digits = 4)
```

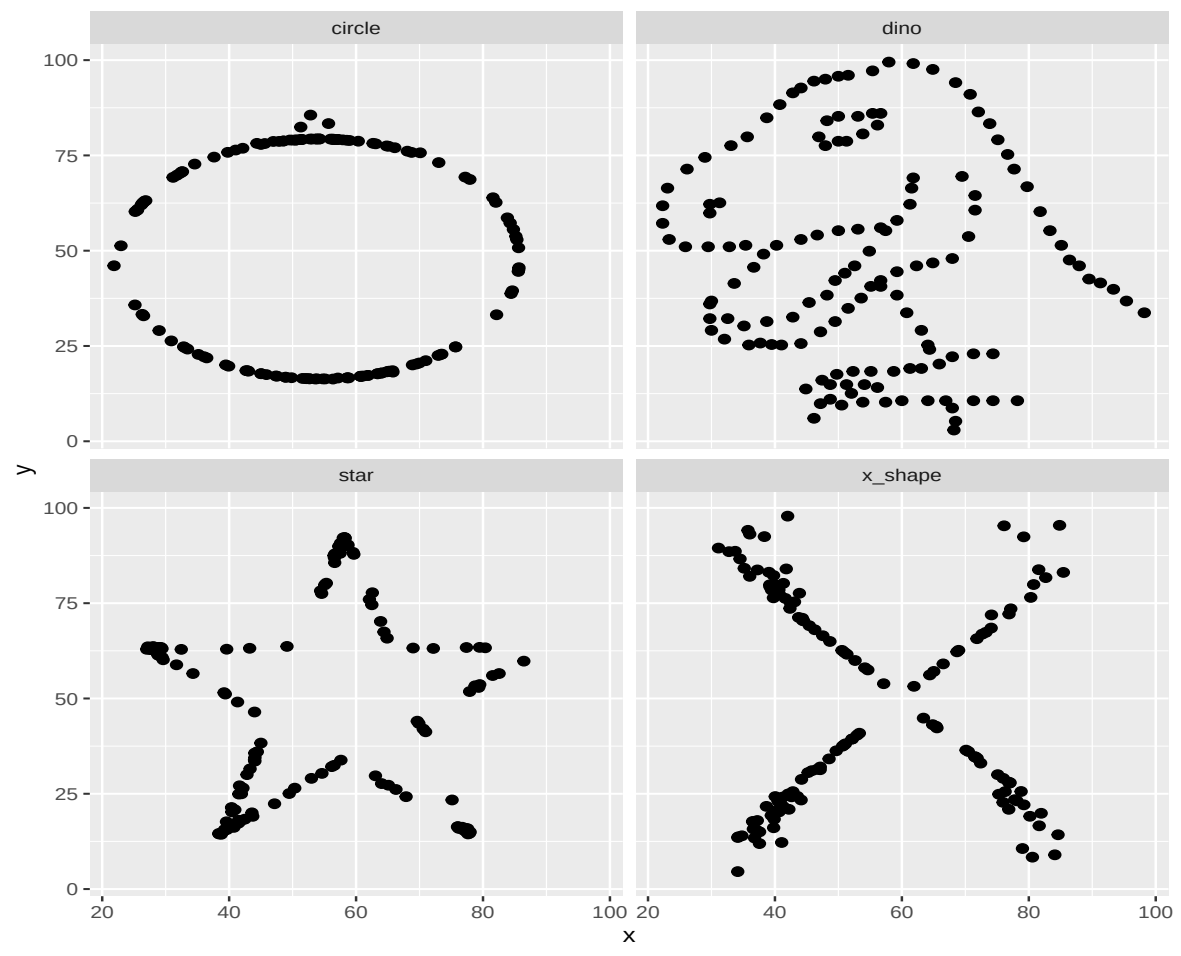
```
##  circle    dino    star x_shape
##  0.0047  0.0042  0.0040  0.0043
```

```
wow %>%
    split(.$dataset) %>%
    purrr::map_dbl(~ cor(.$x, .$y)) %>%
    round(digits = 4)
```

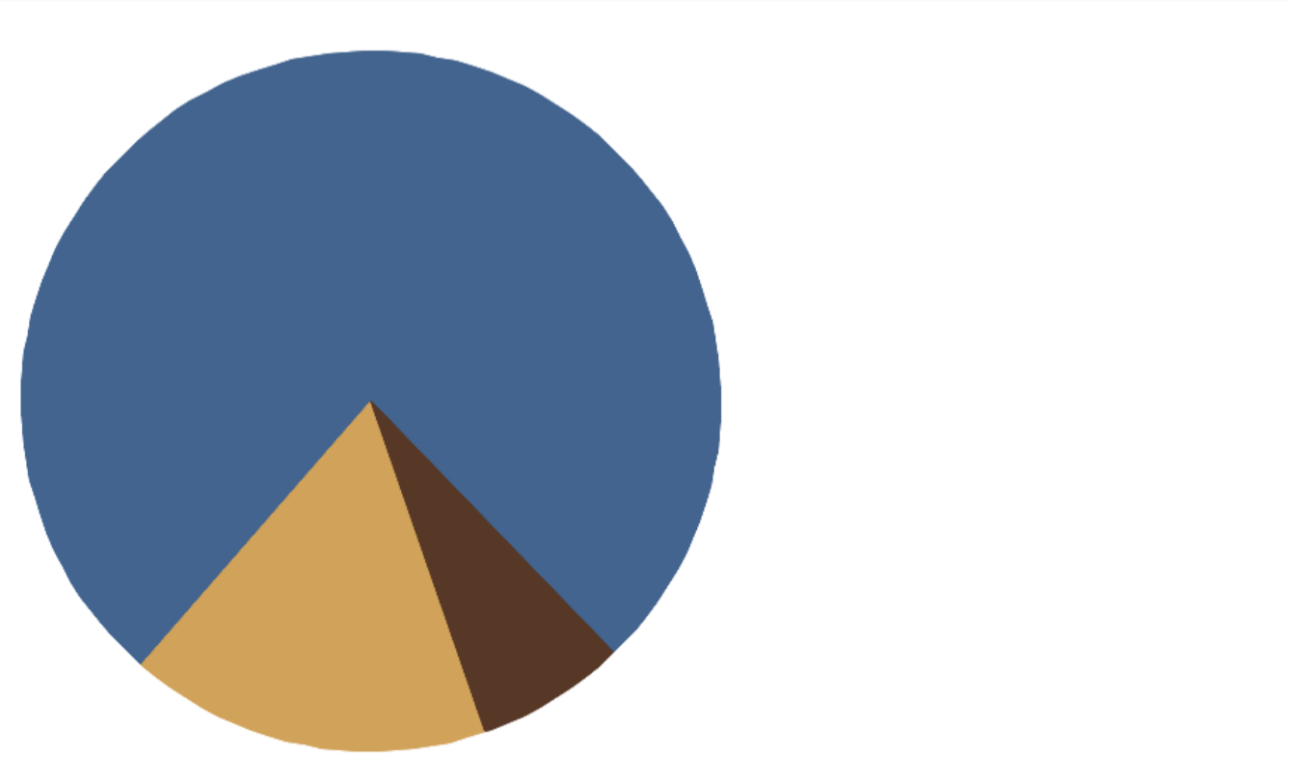
```
##  circle    dino    star x_shape
## -0.0683 -0.0645 -0.0630 -0.0656
```

datasauRus

```
ggplot(wow, aes(x = x, y = y)) +
    geom_point(size = 2) +
    facet_wrap(~dataset)
```

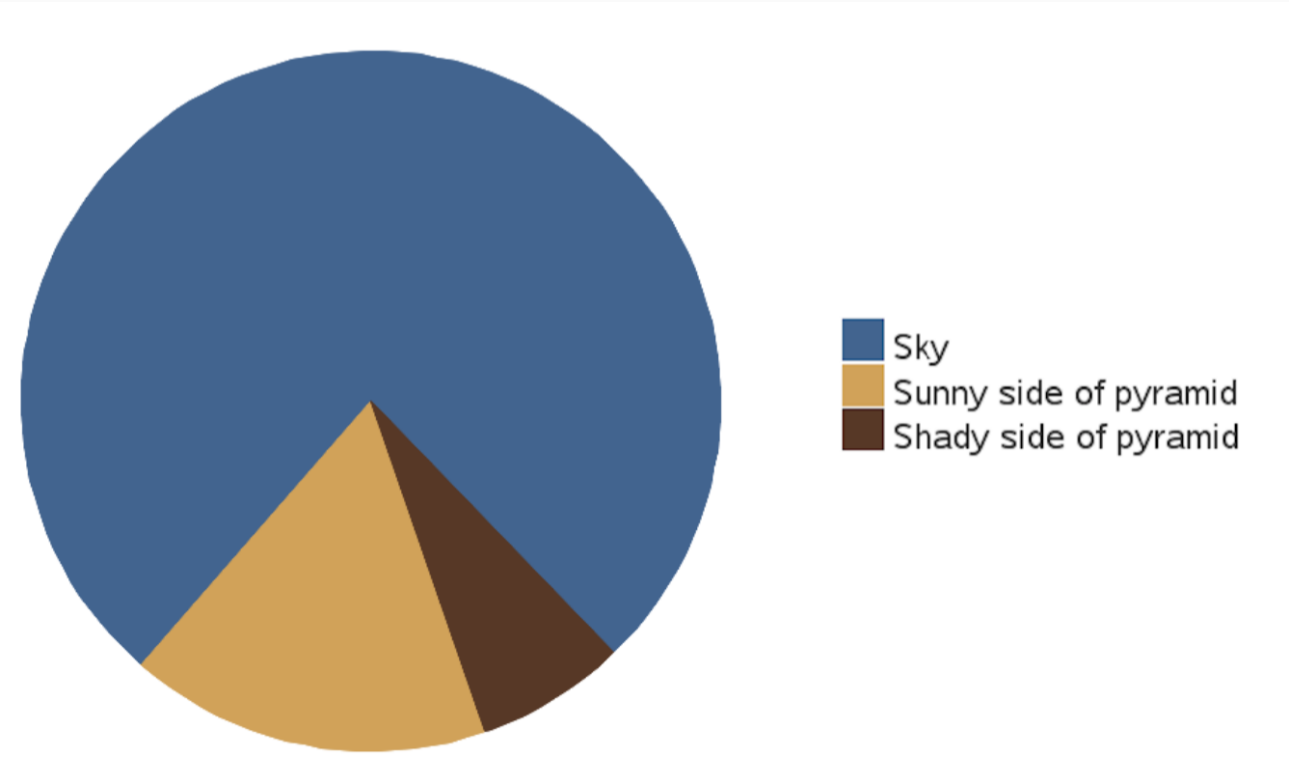


Tortas



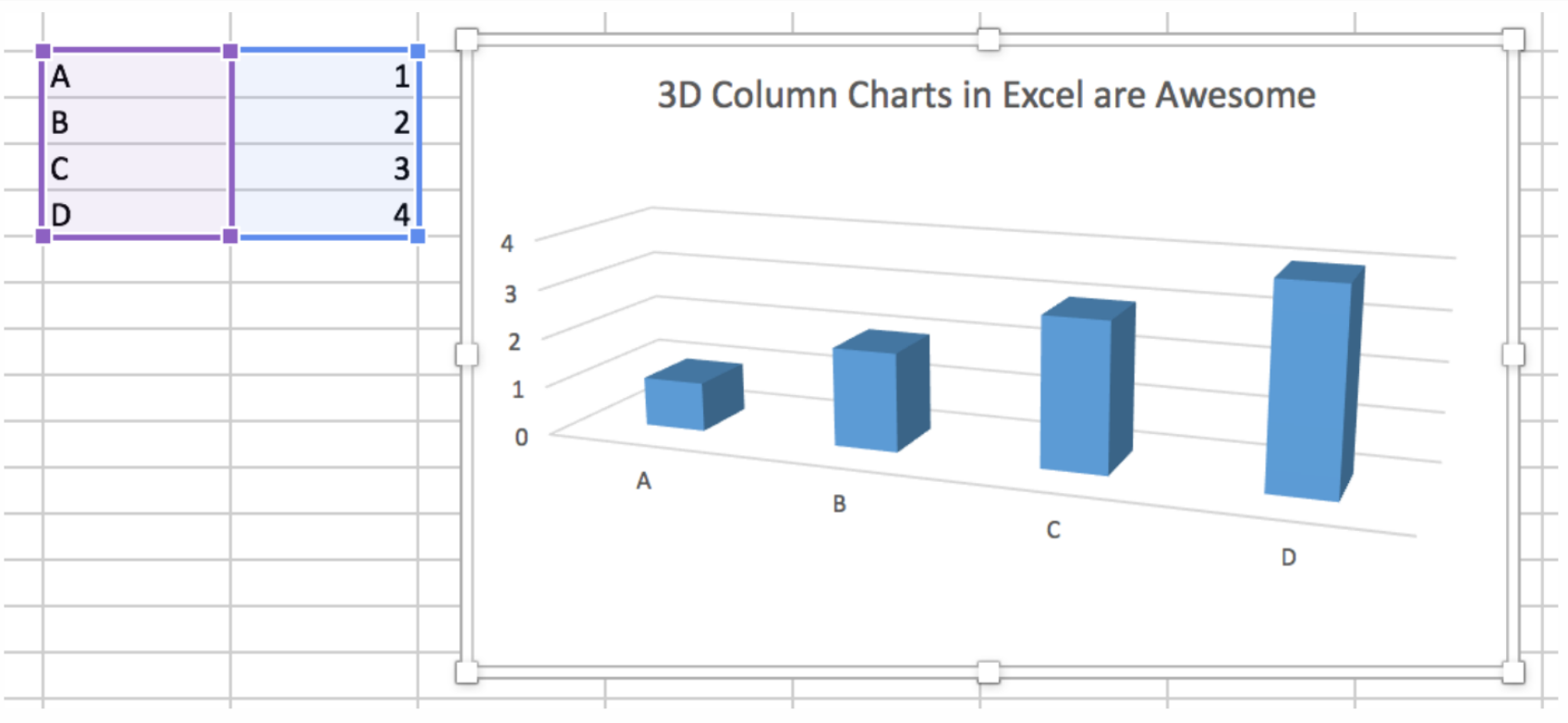
Fuente: Geofaceting Argentina: @TuQmano

Tortas



Fuente: Geofaceting Argentina: @TuQmano

3D



Fuente: Kieran Healy

ggplot2

Gramática de gráficos

El paquete ggplot2 desarrollado por Hadley Wickham se basa en la teora de visualización de datos desarrollada en *The Grammar of Graphics*, por Leland Wilkinson.

La existencia de una gramática supone cierto orden en la composición de un gráfico. Igual que una gramática un de un lenguaje, para construir una oración en algn idioma hay que respetar cierto orden (verbo, predicado, sujeto...)

Gráficos estadsticos

In brief, the grammar tells us that a statistical graphic is a mapping from data to aesthetic attributes (colour, shape, size) of geometric objects (points, lines, bars). The plot may also contain statistical transformations of the data and is drawn on a specific coordinate system. Faceting can be used to generate the same plot for different subsets of the dataset. It is the combination of these independent components that make up a graphic. (Wickham)

Principales componentes (capas)

`data`	conjunto de datos (data.frame)
`aes`	atributos estéticos que son asignados a variables (posición (x, c(x, y),), tamao, color
`geom`	el objeto geométrico a plotear en una parcela
`coord`	sistema de coordenadas
`stats`	transformaciones estadsticas para visualizar determinado geom_*
`facets`	el mismo gráfico pero por algn subconjunto de datos
`labels`	etiquetas de los ejes...
`theme`	Todos los aspectos relativos al fondo...

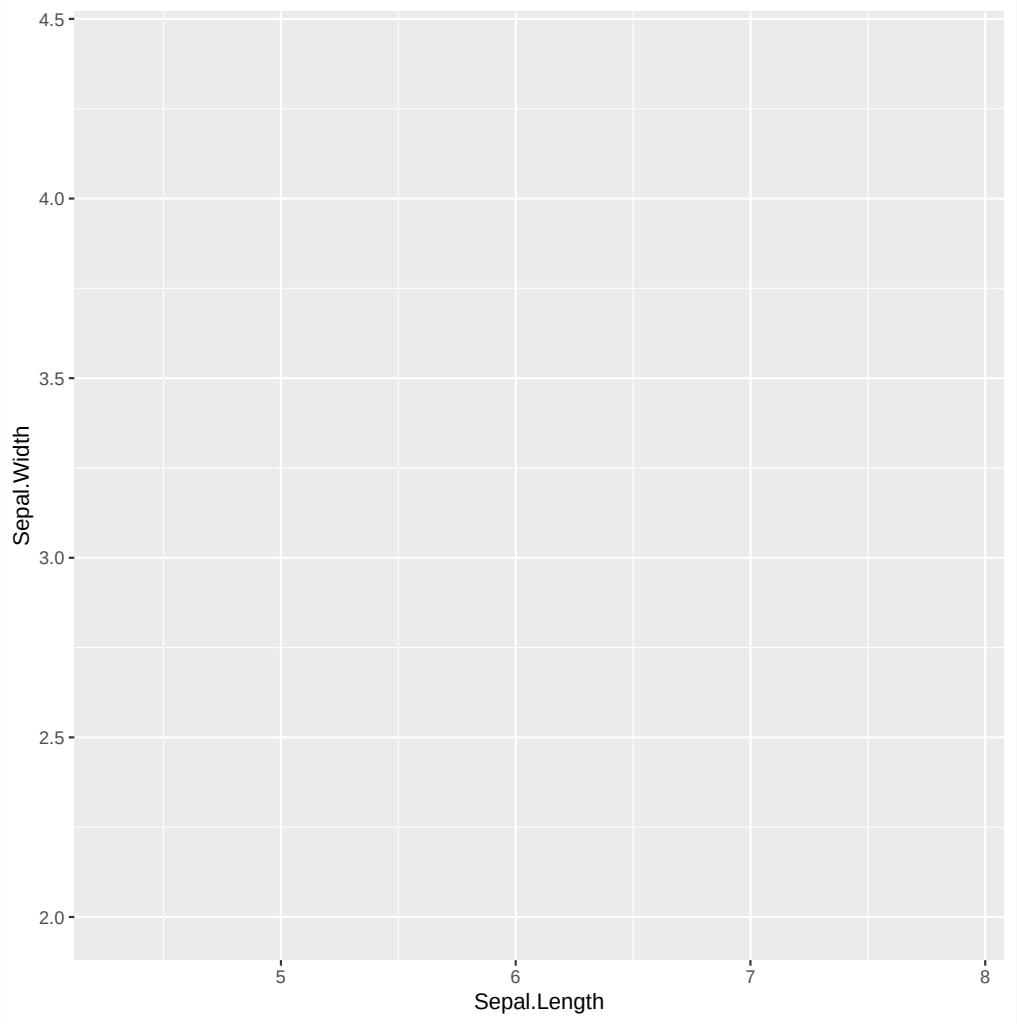
Principales componentes (capas)

Código básico

```
library(ggplot2)
ggplot()
```

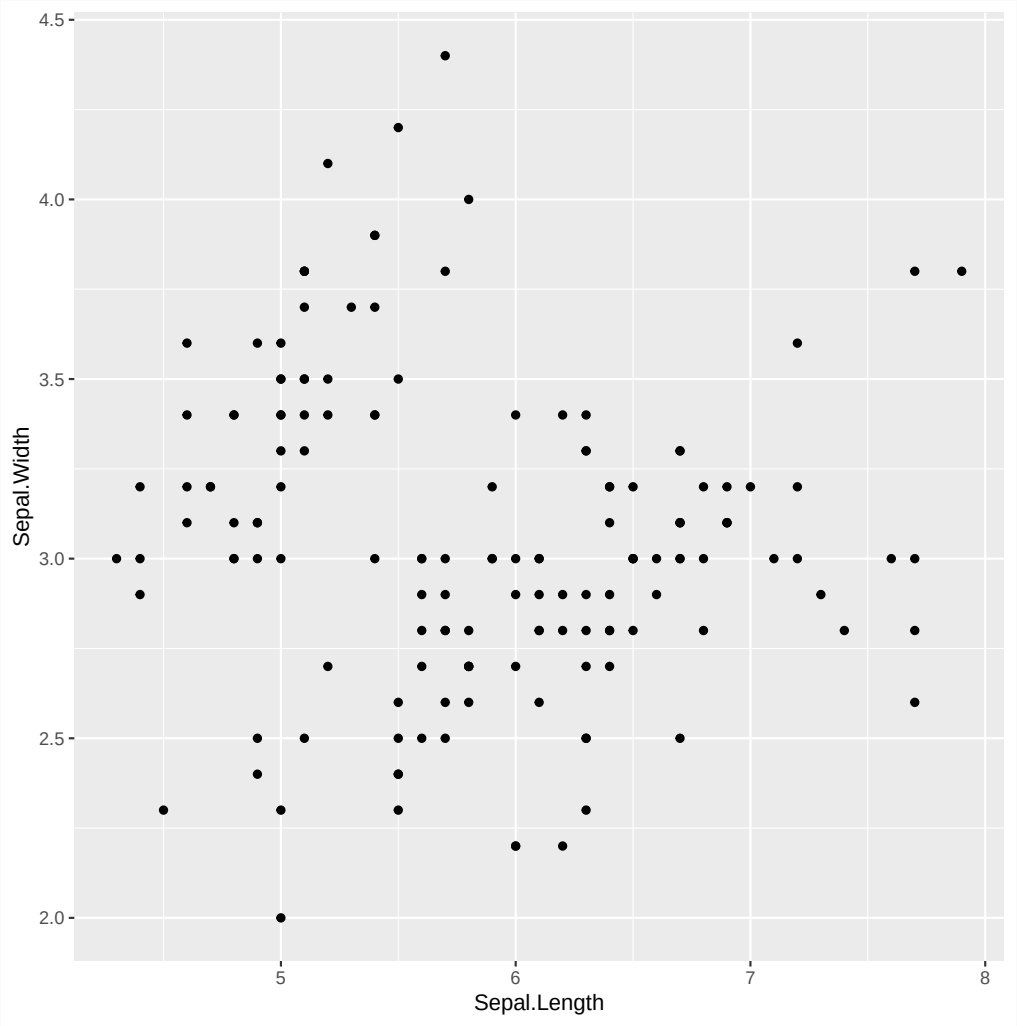
Código básico: primer capa

```
ggplot(data = iris, mapping = aes(x = Sepal.Length, y = Sepal.Width))
```



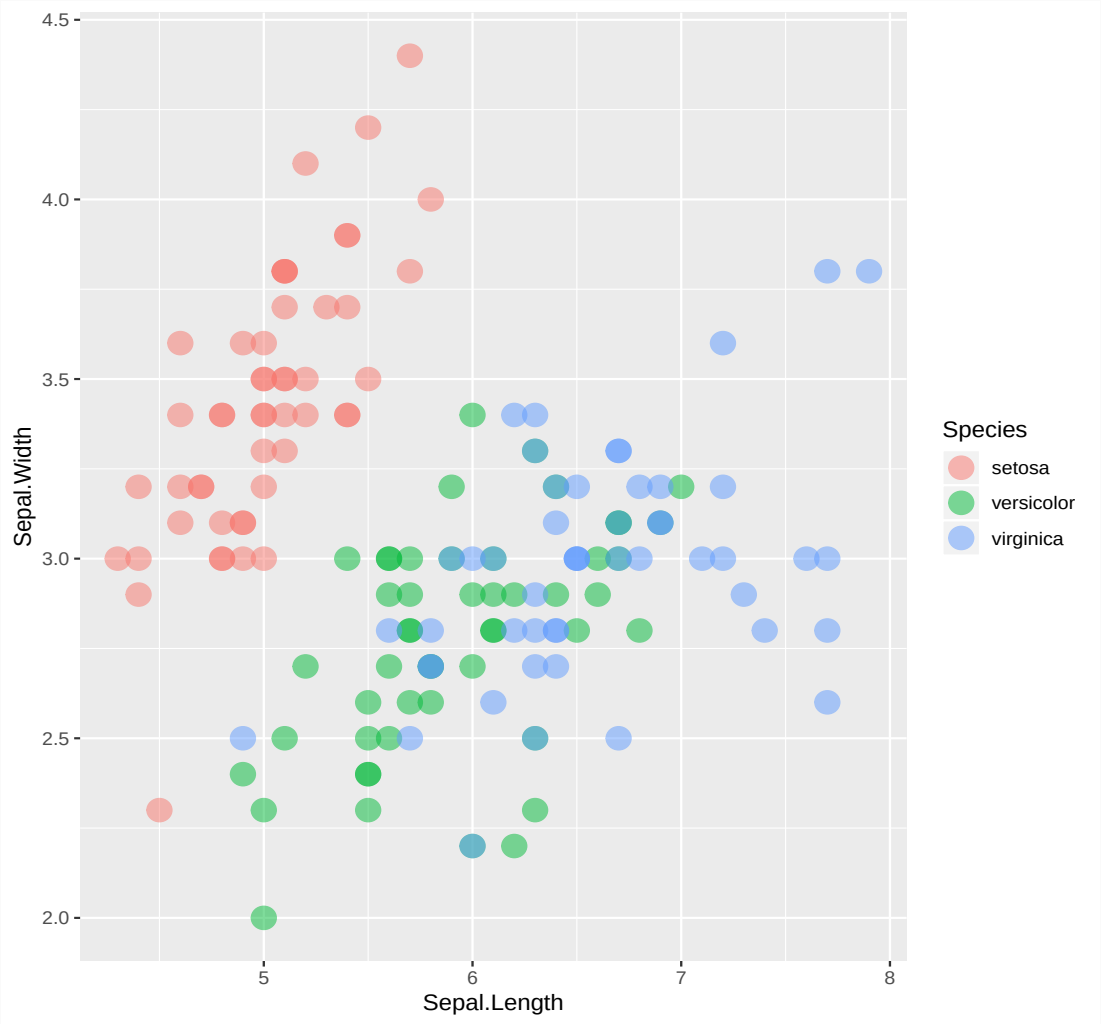
Código básico: geom

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



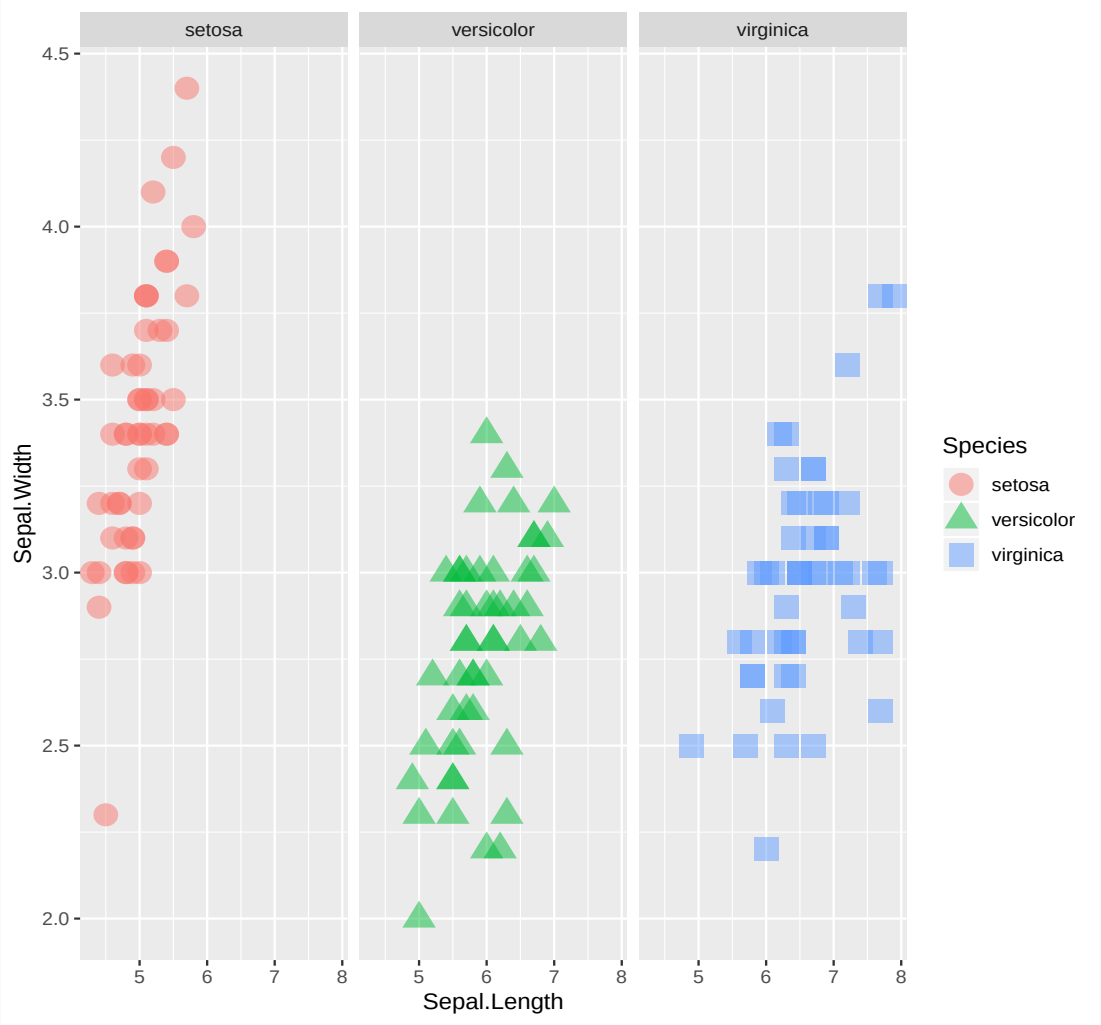
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
    geom_point(size = 5, alpha = .5, )
```



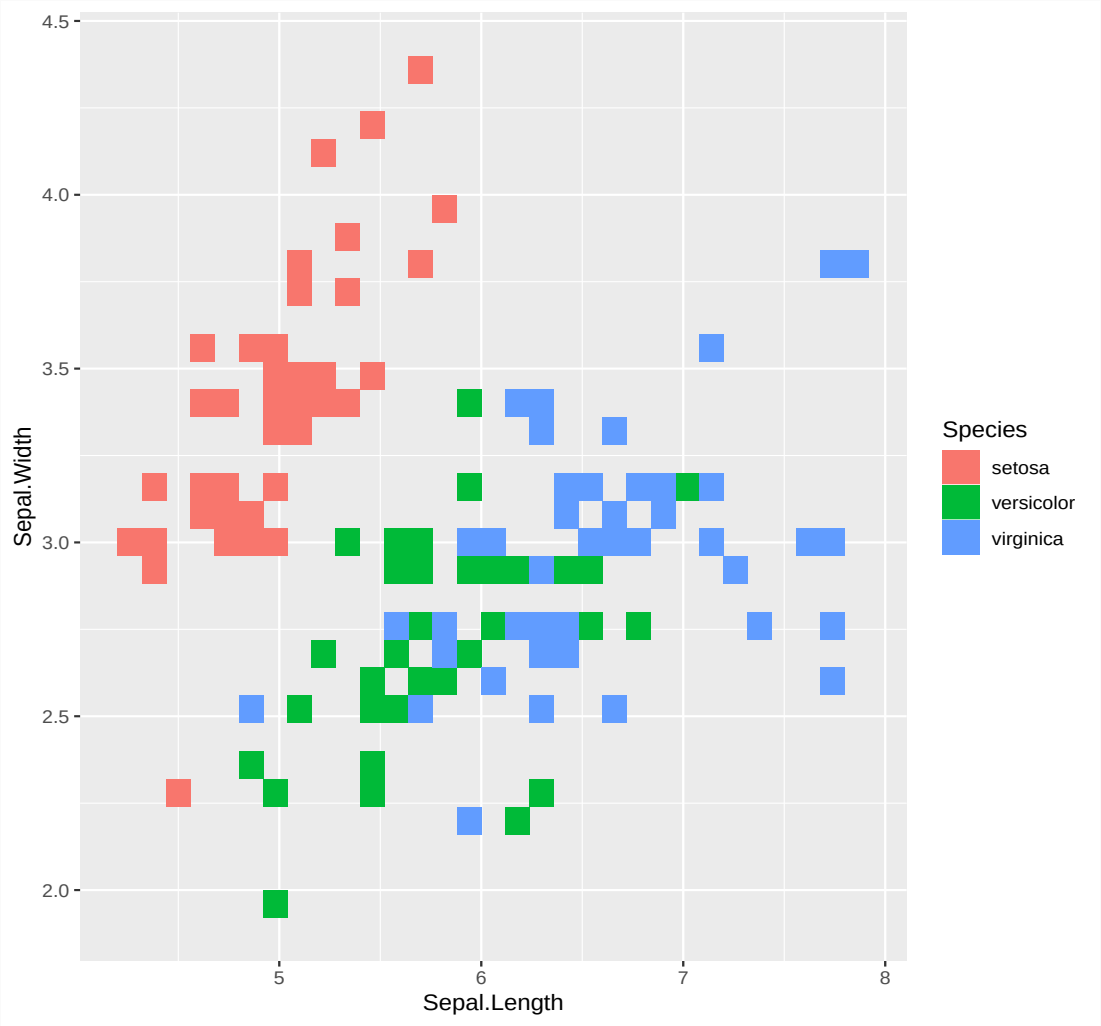
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species, shape = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    facet_wrap(Species~.)
```



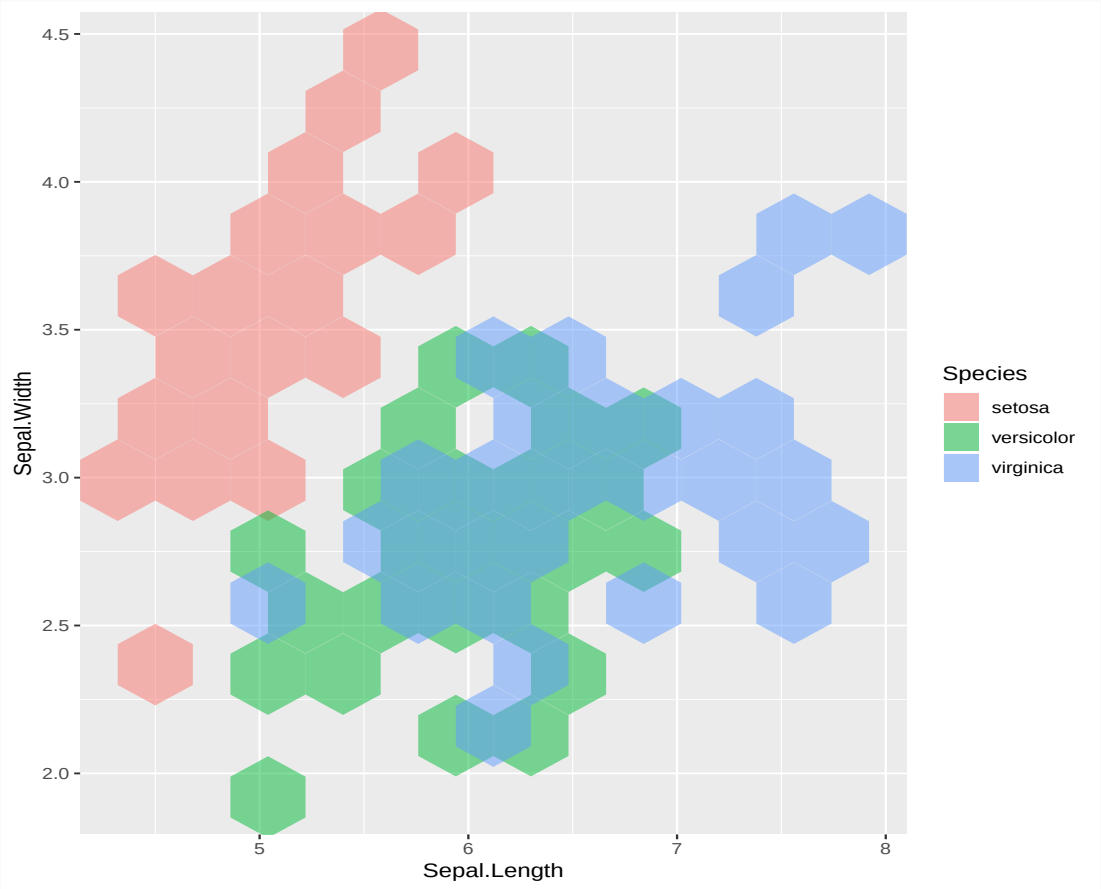
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, fill = Species)) +
    geom_bin2d()
```



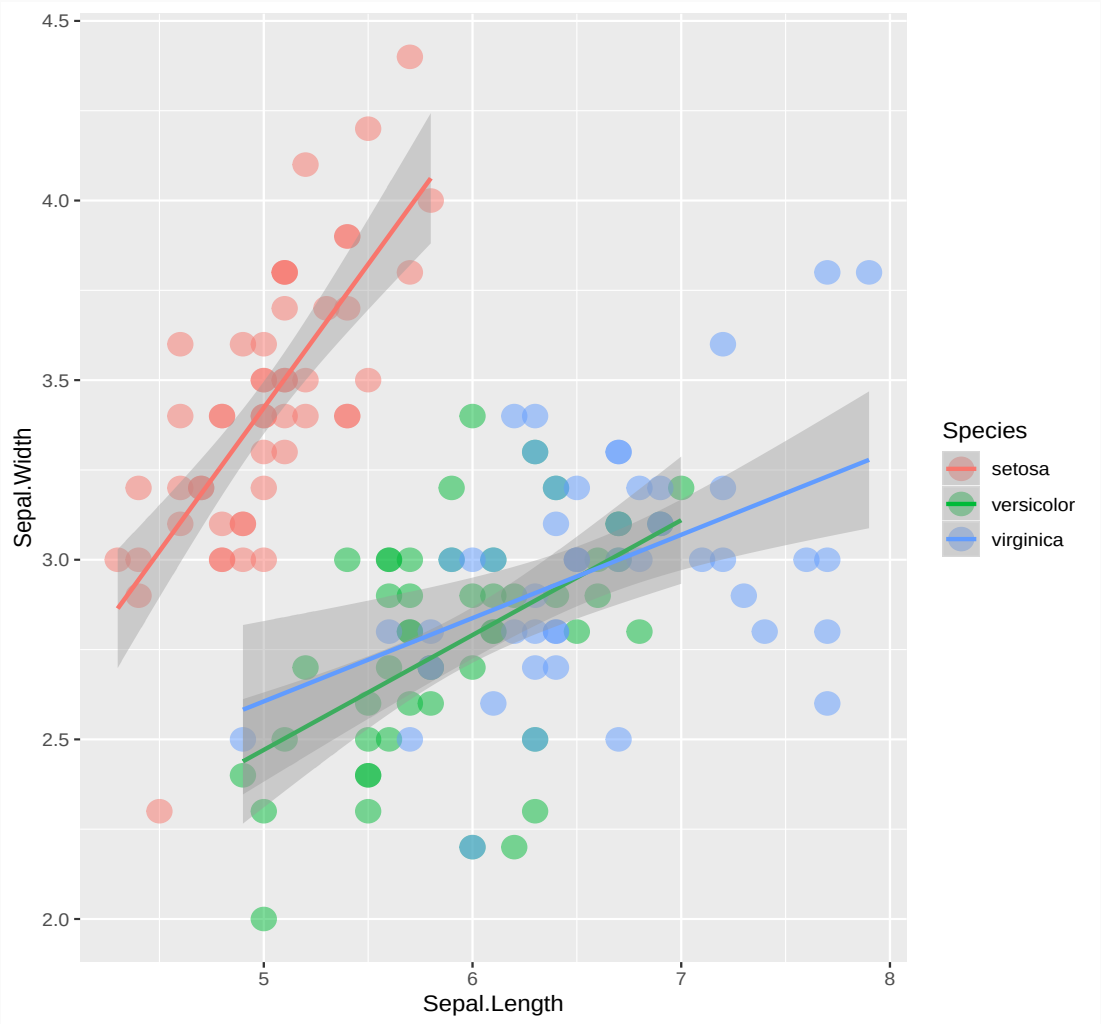
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, fill = Species)) +
    geom_hex(bins  =  10, alpha = .5)
```



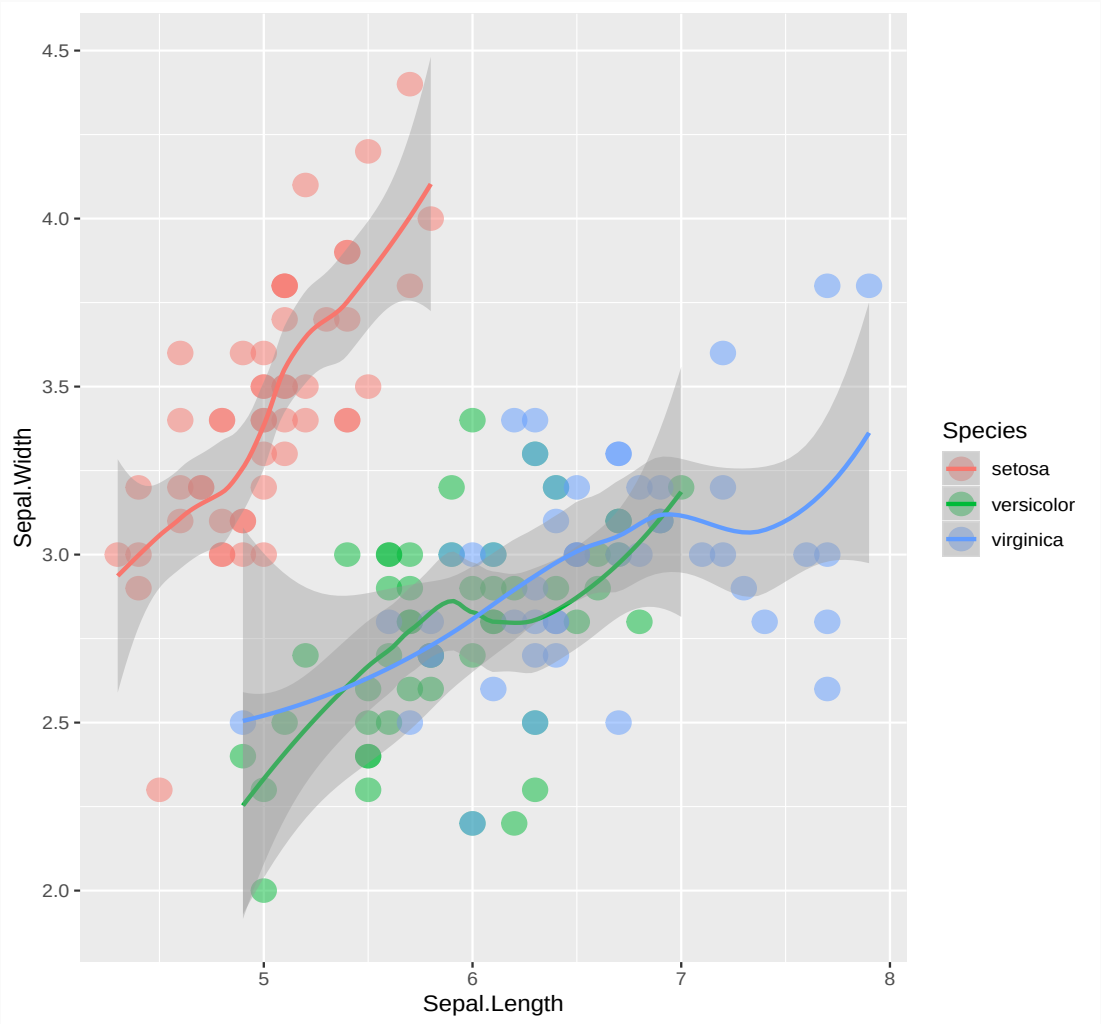
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    geom_smooth(method="lm")
```



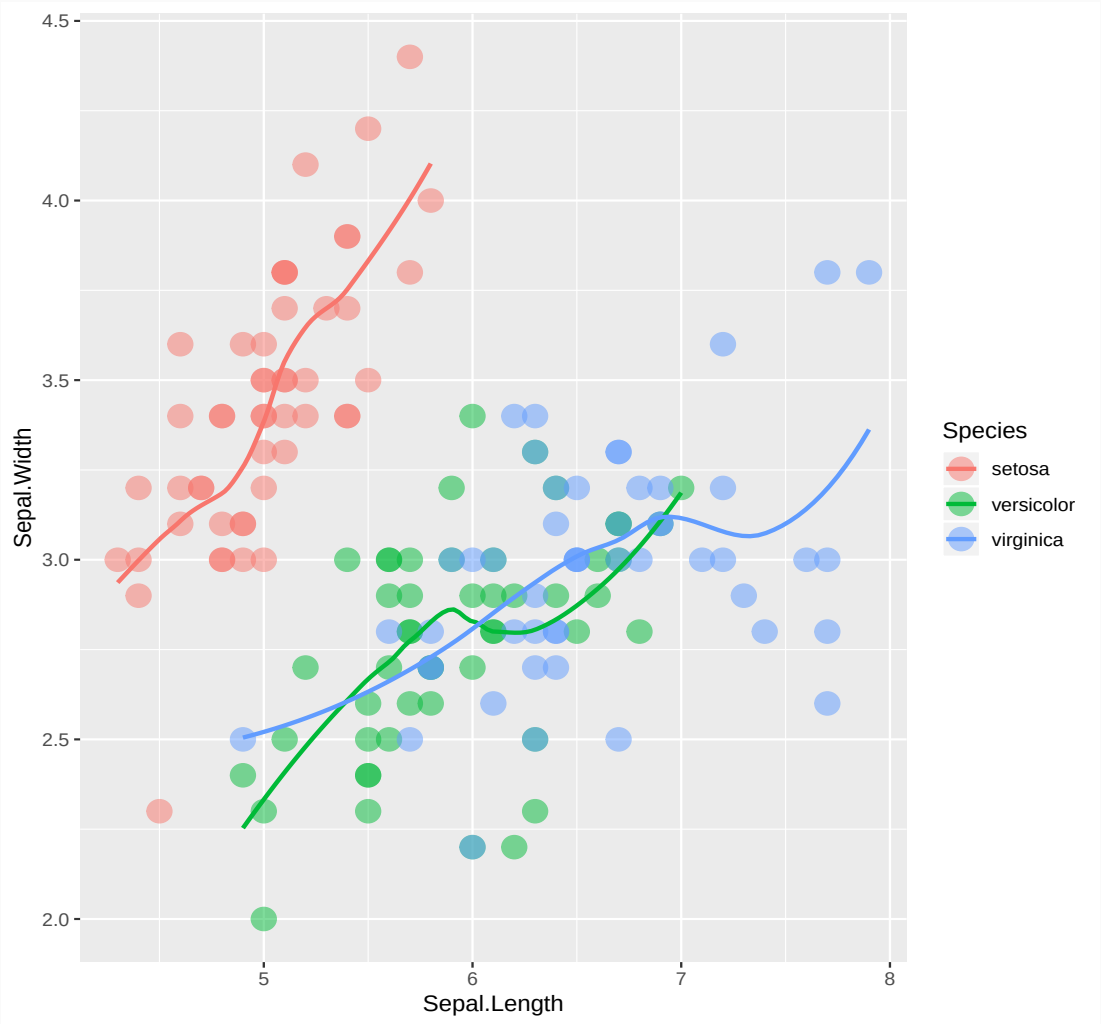
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    geom_smooth(method="loess")
```



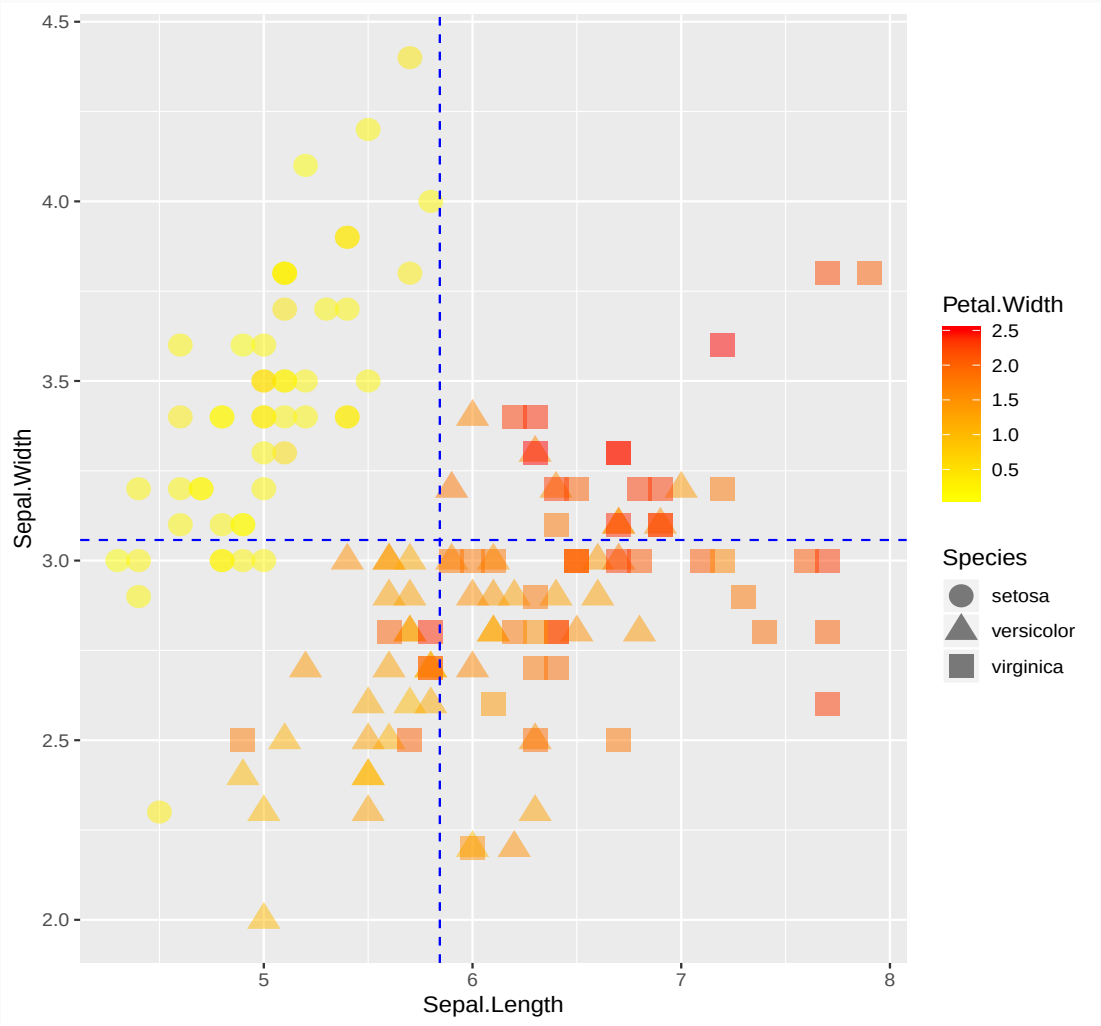
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    geom_smooth(method="loess", se = FALSE)
```



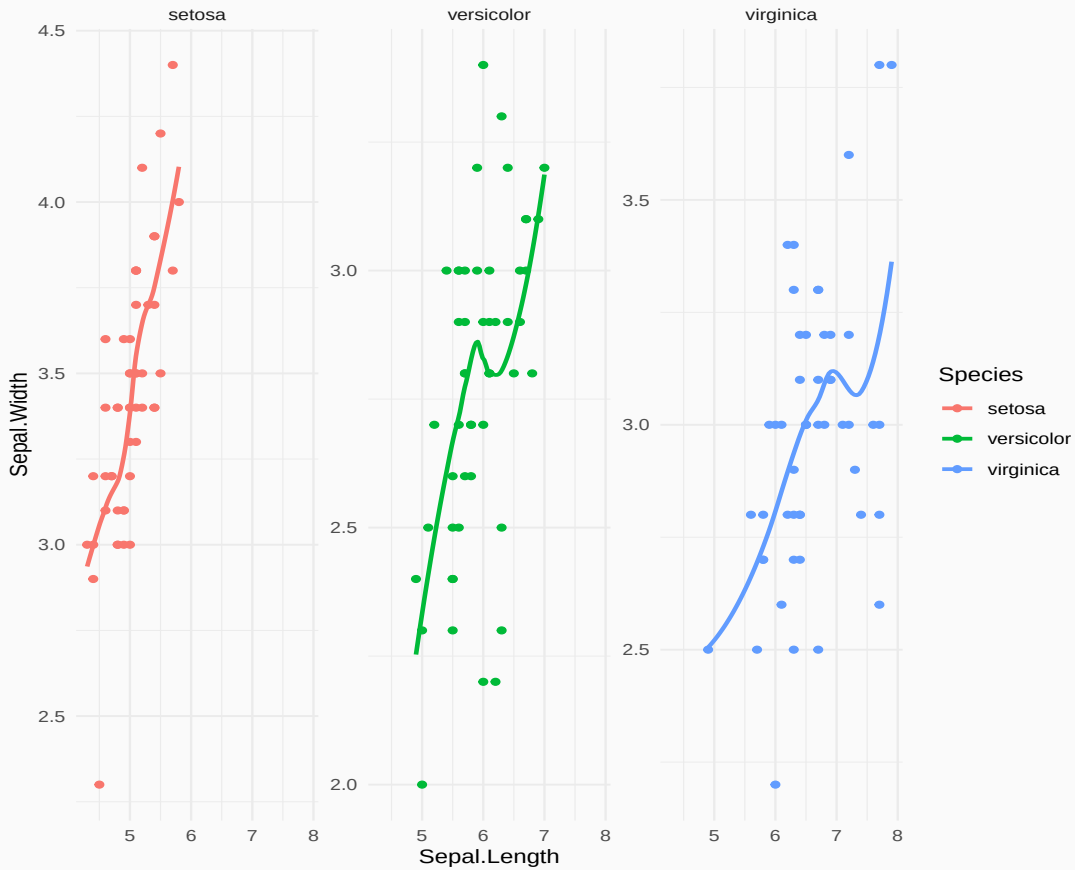
Código básico: geom_*

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Petal.Width, shape = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    geom_vline(aes(xintercept = mean(Sepal.Length)), color = "blue", linetype = "dashed") +
    geom_hline(aes(yintercept = mean(Sepal.Width)),  color = "blue", linetype = "dashed") +
    scale_color_gradient(low = "yellow", high = "red")
```



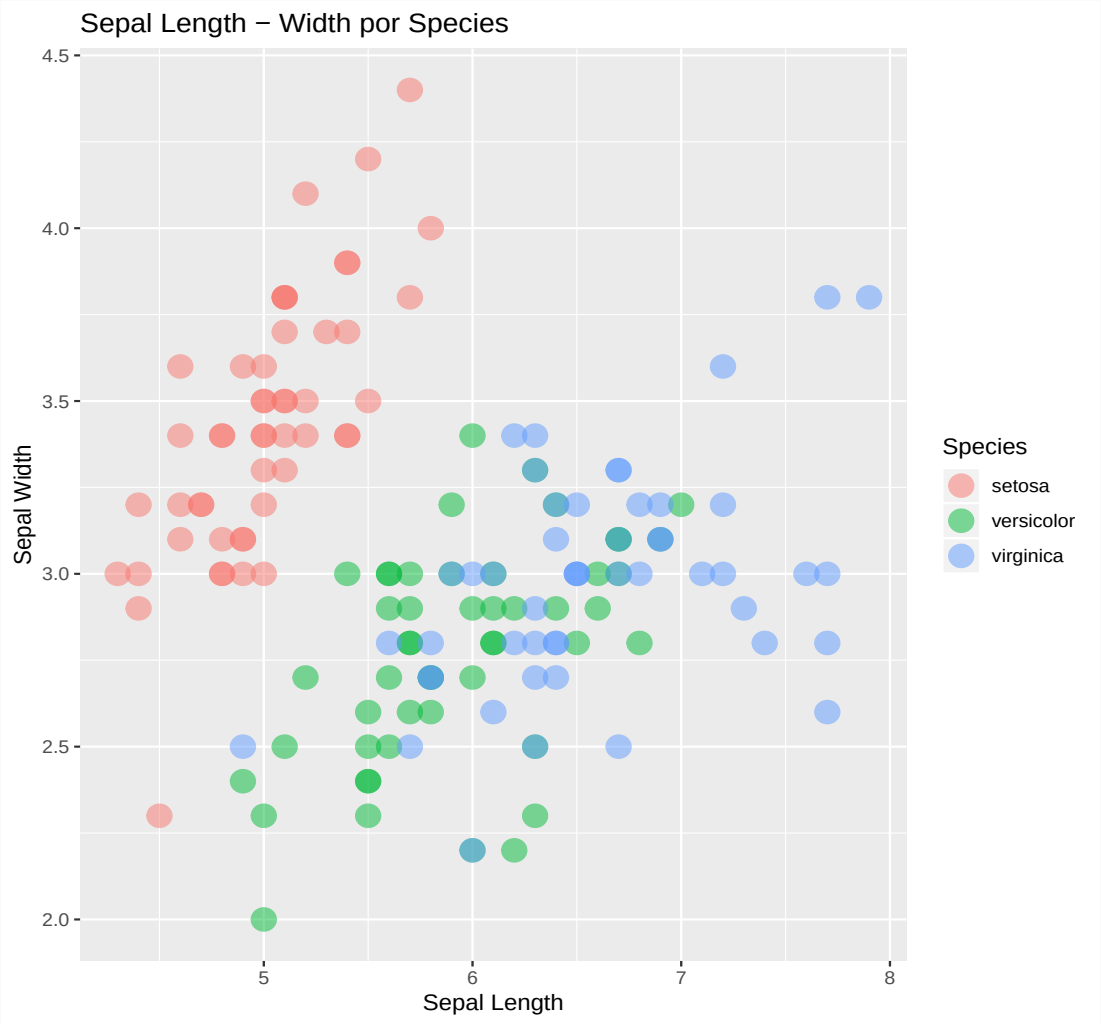
Código básico: geom_*

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width ,color = Species)) +
    geom_point() +
    geom_smooth(method = "loess", se = FALSE) +
    facet_wrap(~Species, scale = 'free_y') +
    theme_minimal()
```



Código básico: geom_*

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width ,color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    xlab("Sepal Length") +
    ylab("Sepal Width") +
    ggtitle("Sepal Length - Width por Species")
```

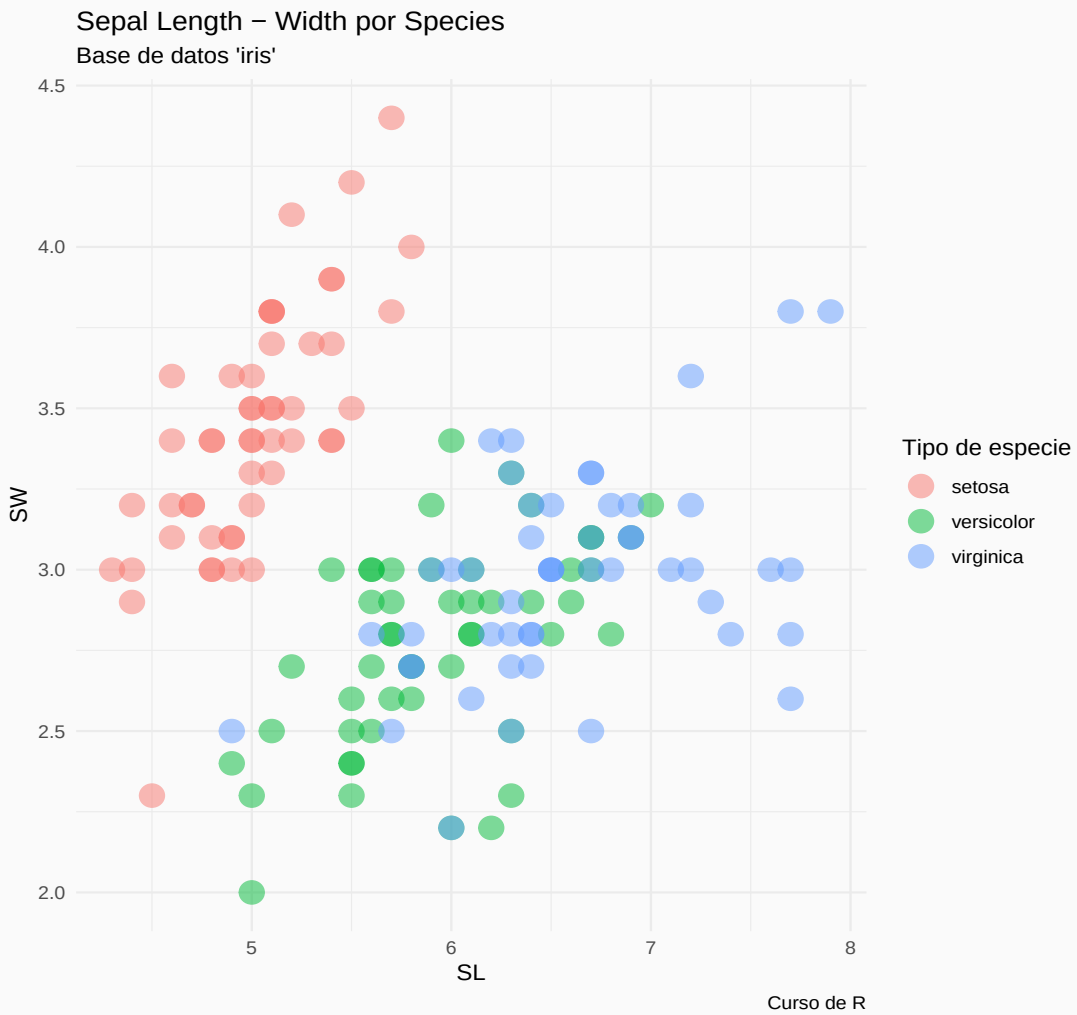


Código básico: labs

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width ,color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    labs(title = "Sepal Length - Width por Species",
        subtitle = "Base de datos 'iris'",
        color = "Tipo de especie", caption = "Curso de R", x = "SL", y = "SW")
```

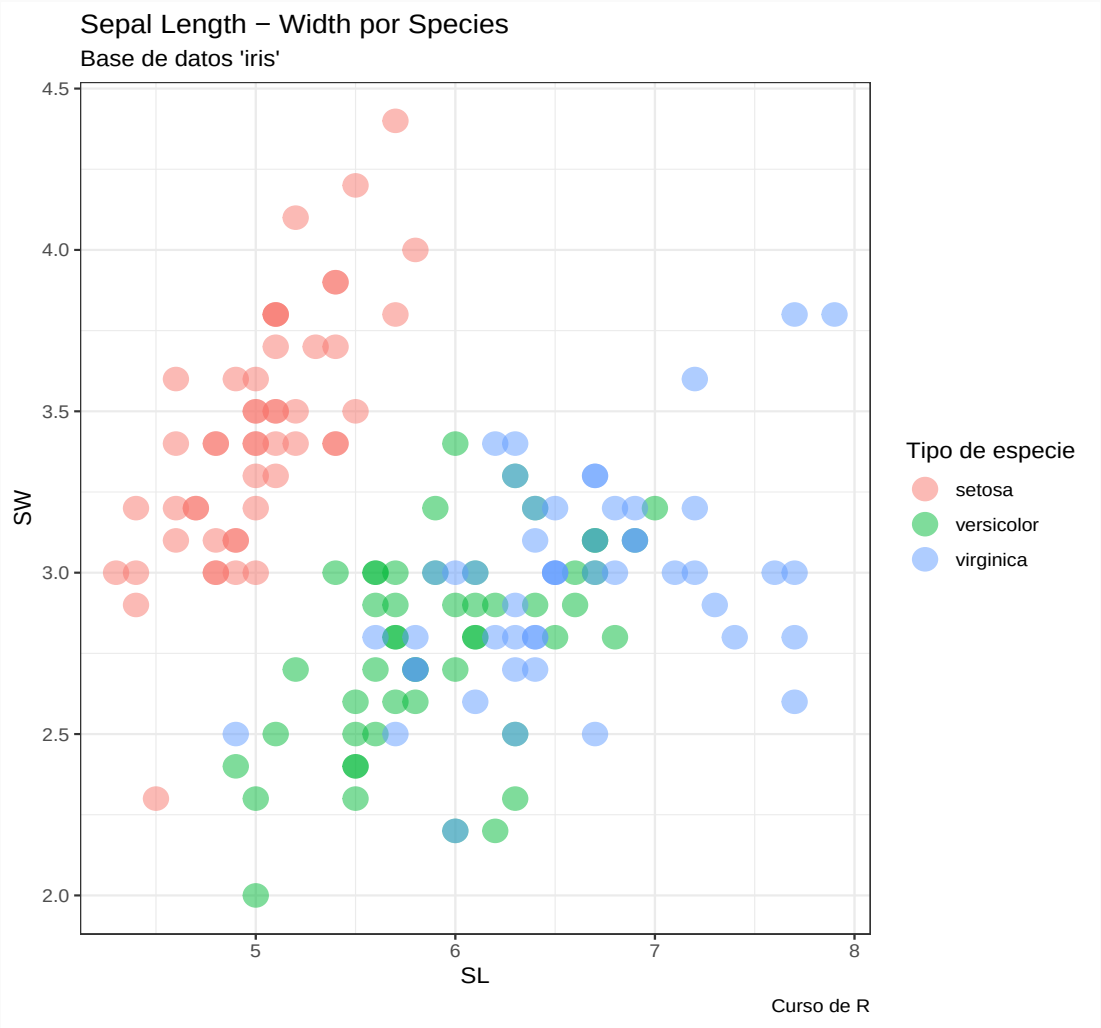
Código básico: theme

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width ,color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    labs(title = "Sepal Length - Width por Species",
        subtitle = "Base de datos 'iris'",
        color = "Tipo de especie", caption = "Curso de R", x = "SL", y = "SW") +
        theme_minimal()
```



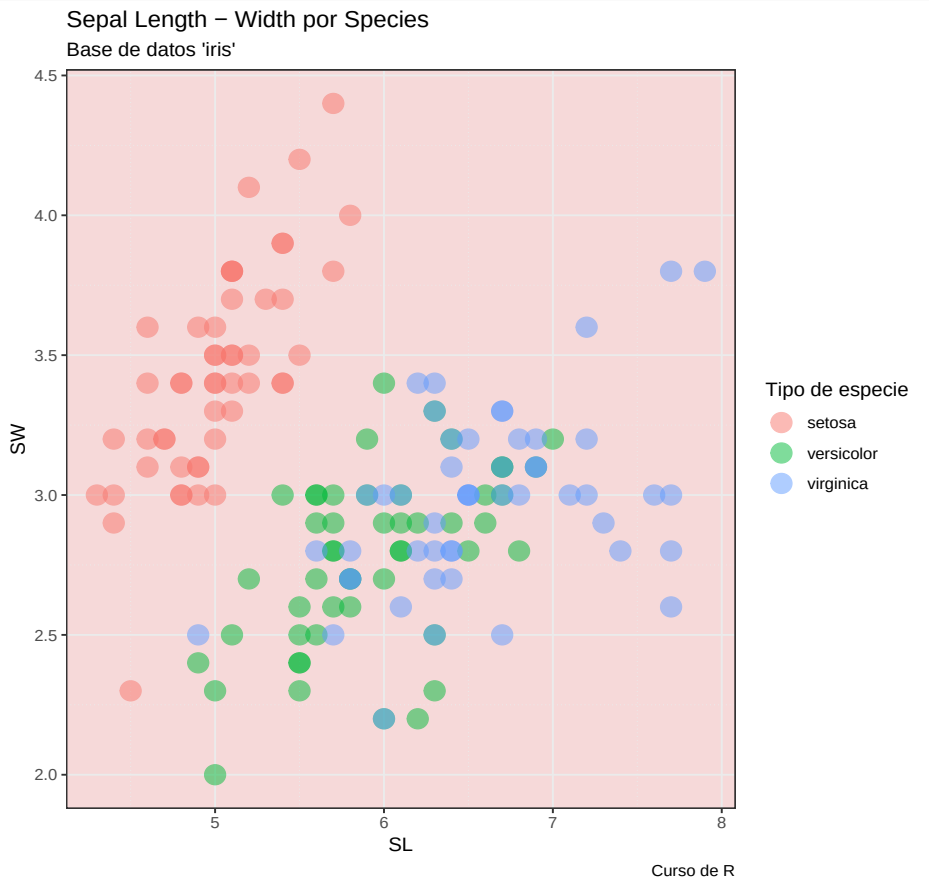
Código básico: theme

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width ,color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    labs(title = "Sepal Length - Width por Species",
        subtitle = "Base de datos 'iris'",
        color = "Tipo de especie", caption = "Curso de R", x = "SL", y = "SW") +
        theme_bw()
```



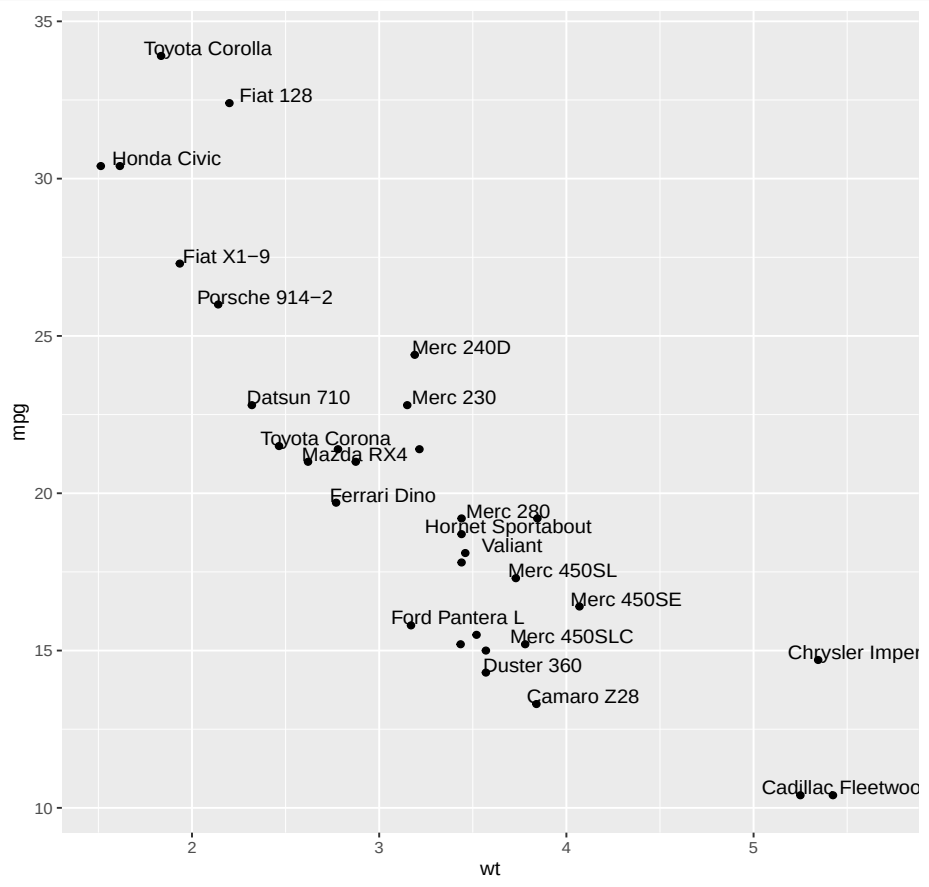
Código básico: theme

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width ,color = Species)) +
    geom_point(size = 5, alpha = .5, ) +
    labs(title = "Sepal Length - Width por Species", subtitle = "Base de datos 'iris'",
        color = "Tipo de especie", caption = "Curso de R", x = "SL", y = "SW") +
        theme_bw()  +
            theme(panel.background = element_rect(fill = "#F6D9D9"),
                  panel.grid.minor = element_line(linetype = "dotted"))
```



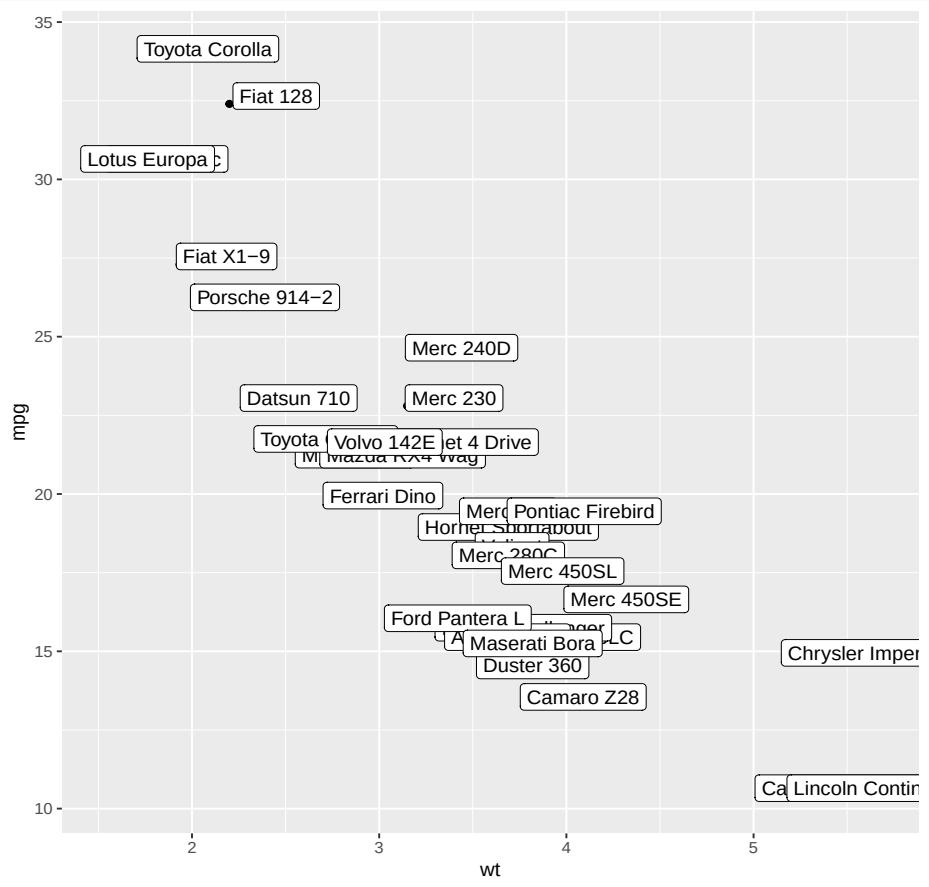
Código básico: text

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point() +
    geom_text(
        label=rownames(mtcars),
        nudge_x = 0.25, nudge_y = 0.25,
        check_overlap = TRUE
    )
```



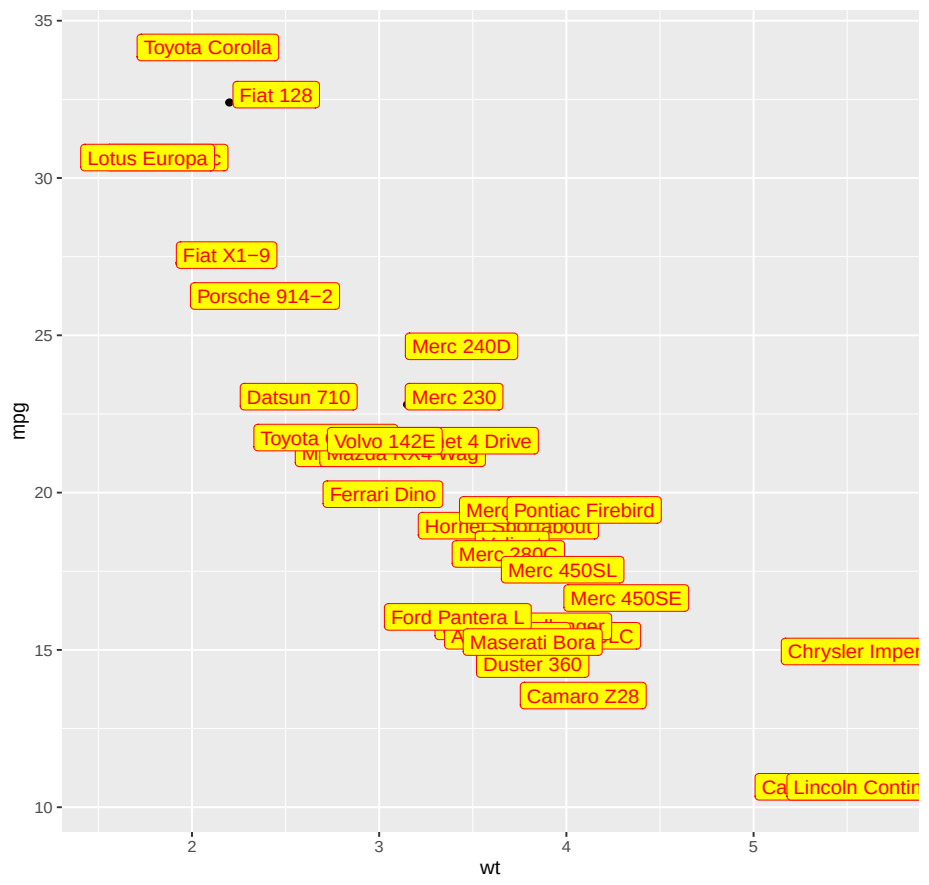
Código básico: label

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point() +
    geom_label(
        label=rownames(mtcars),
        nudge_x = 0.25, nudge_y = 0.25
    )
```



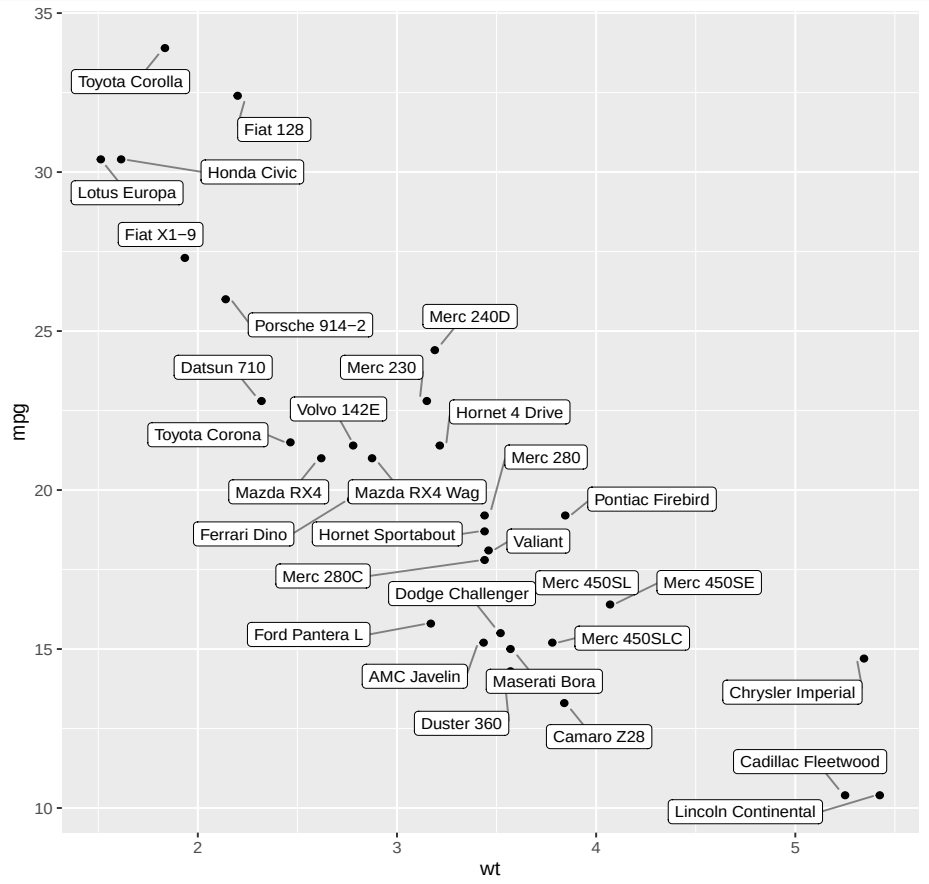
Código básico: label

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point() +
    geom_label(
        label=rownames(mtcars),
        nudge_x = 0.25, nudge_y = 0.25,
        fill = "yellow",
        color = "red"
    )
```



Código básico: label (ggrepel)

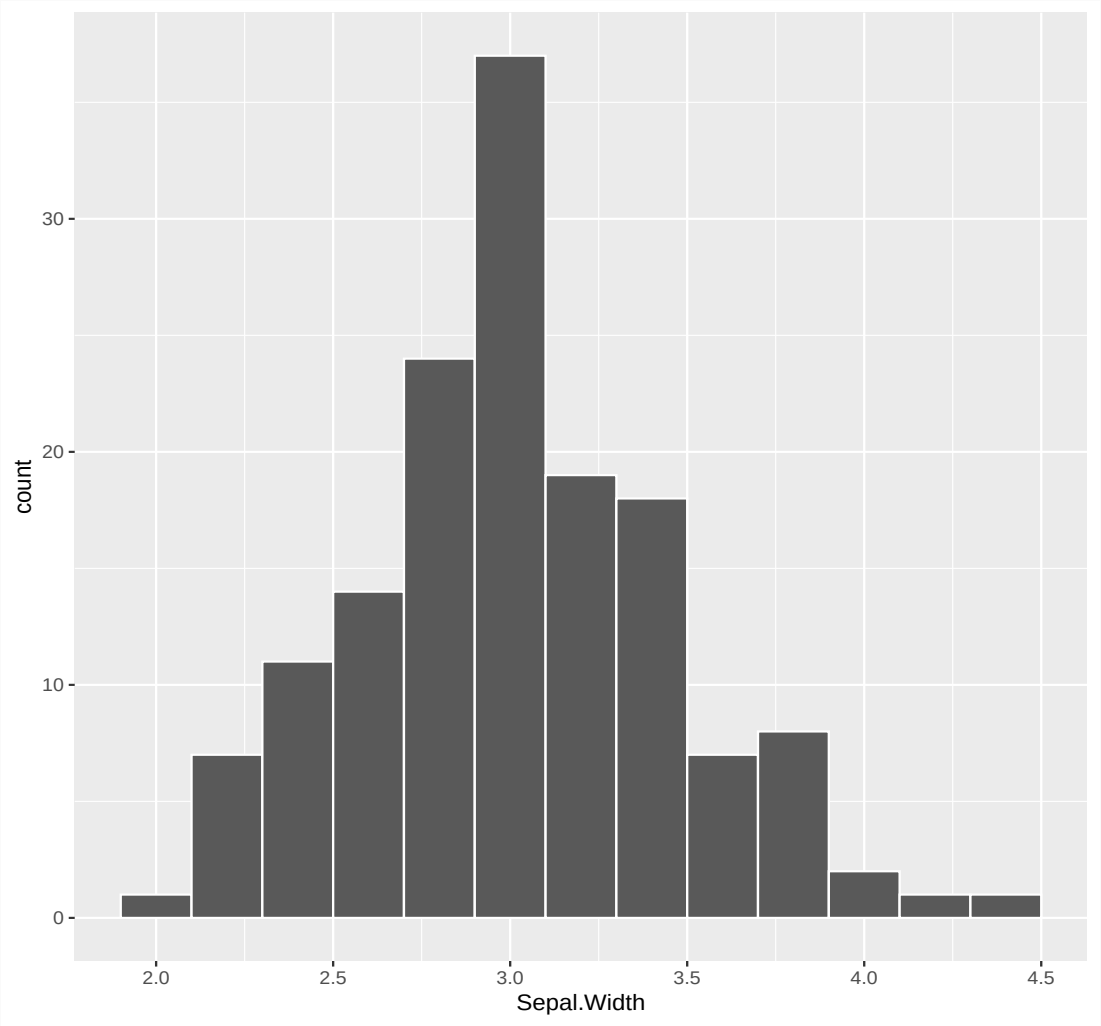
```
library(ggrepel)
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point() +
    geom_label_repel(aes(label=rownames(mtcars)),
                     box.padding   = 0.35,
                     point.padding = 0.5,
                     segment.color = 'grey50', size = 3)
```



Histograma

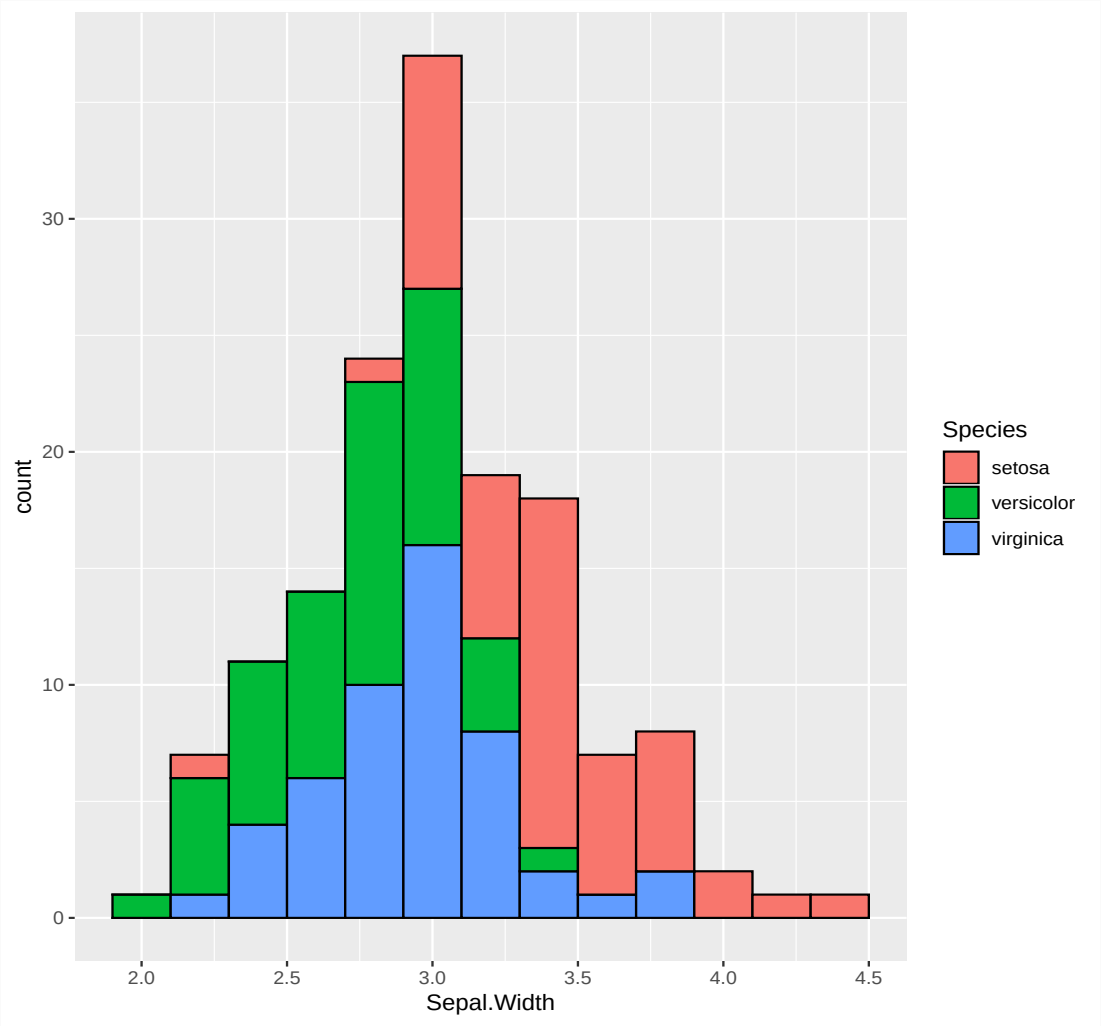
Código básico: histogram

```
ggplot(data = iris, aes(x = Sepal.Width)) +
    geom_histogram(binwidth = 0.2, color = "white")
```



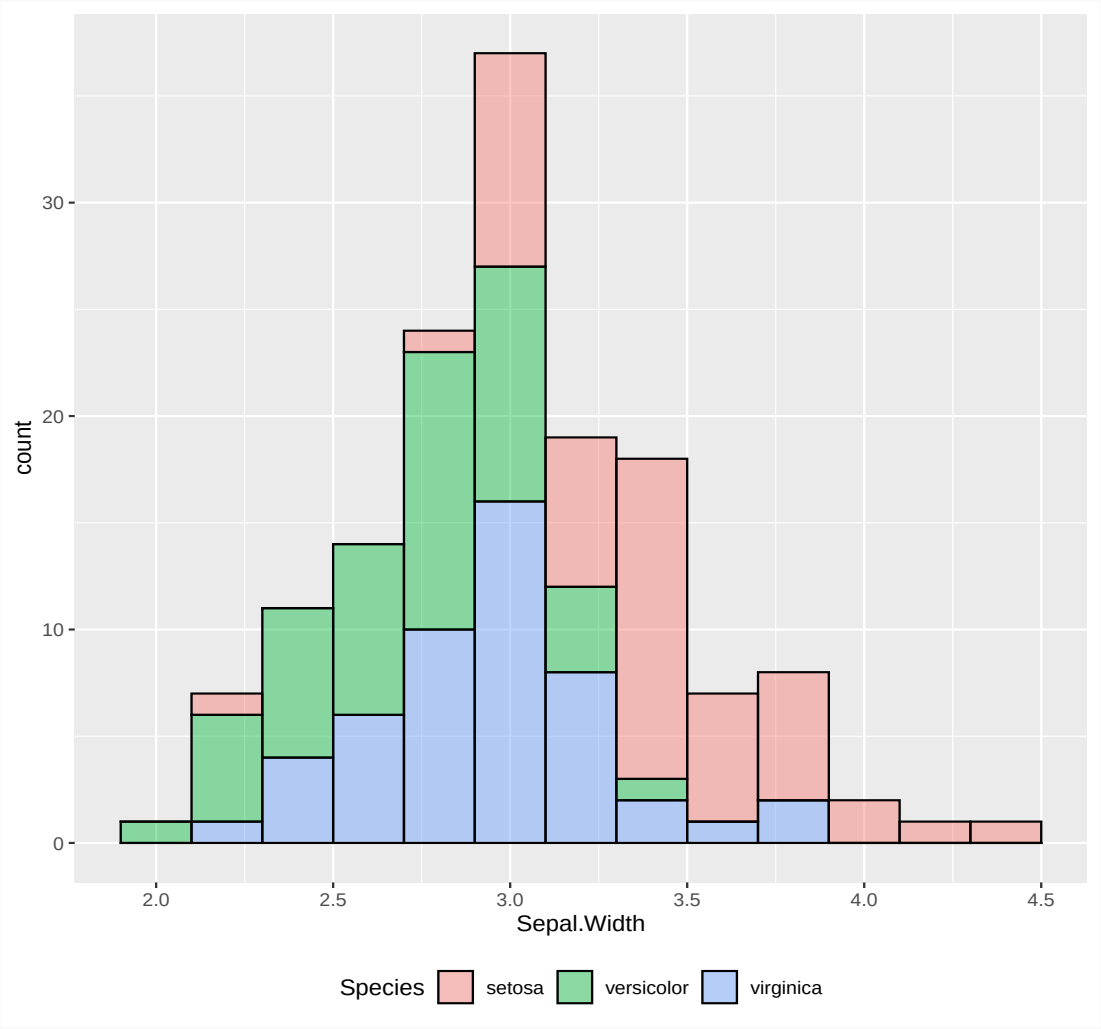
Código básico: histogram

```
ggplot(data = iris, aes(x = Sepal.Width)) +
    geom_histogram(binwidth = 0.2, color = "black", aes(fill = Species))
```



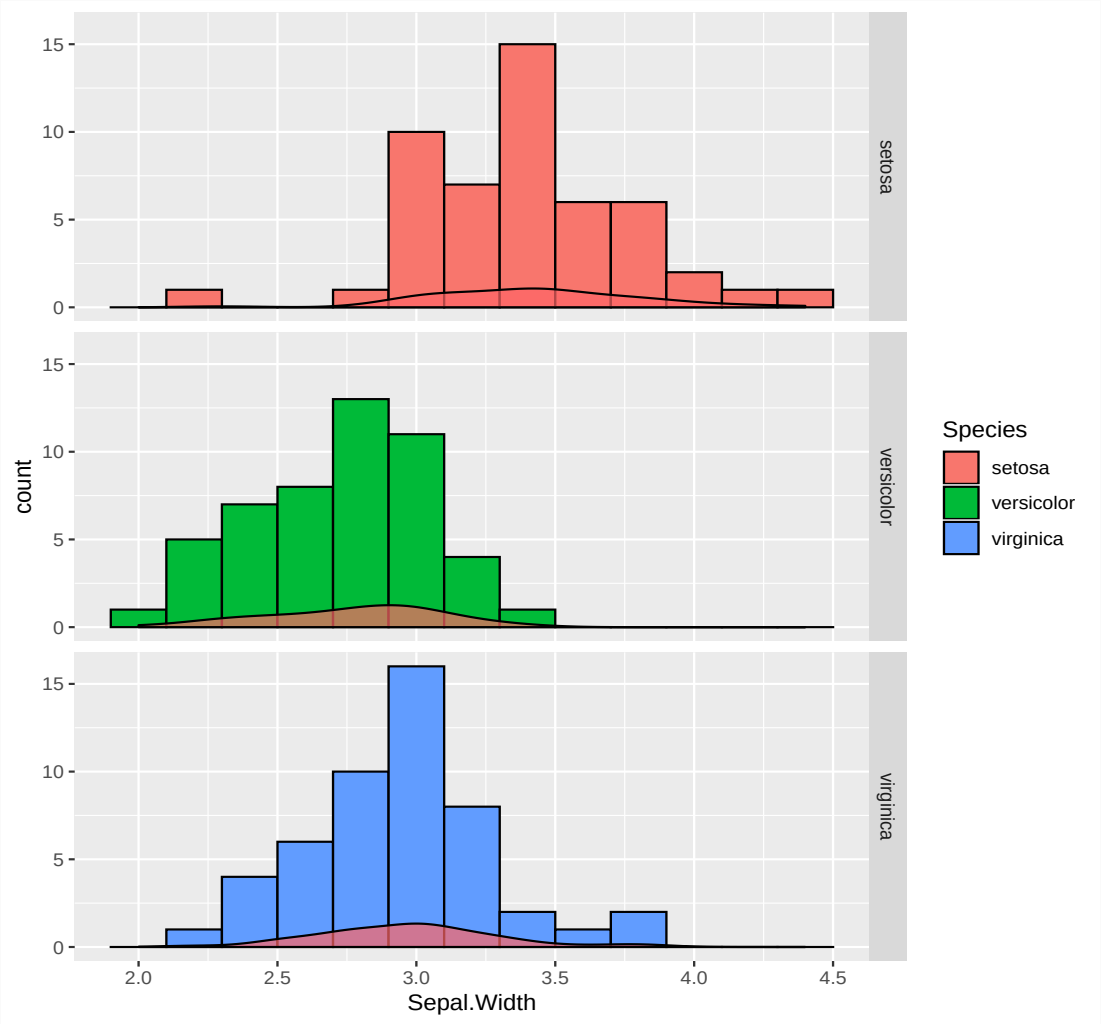
Código básico: histogram

```
ggplot(data = iris, aes(x = Sepal.Width)) +
    geom_histogram(binwidth = 0.2, color = "black",alpha = .42, aes(fill = Species)) +
    theme(legend.position="bottom") #"top", "none"..
```



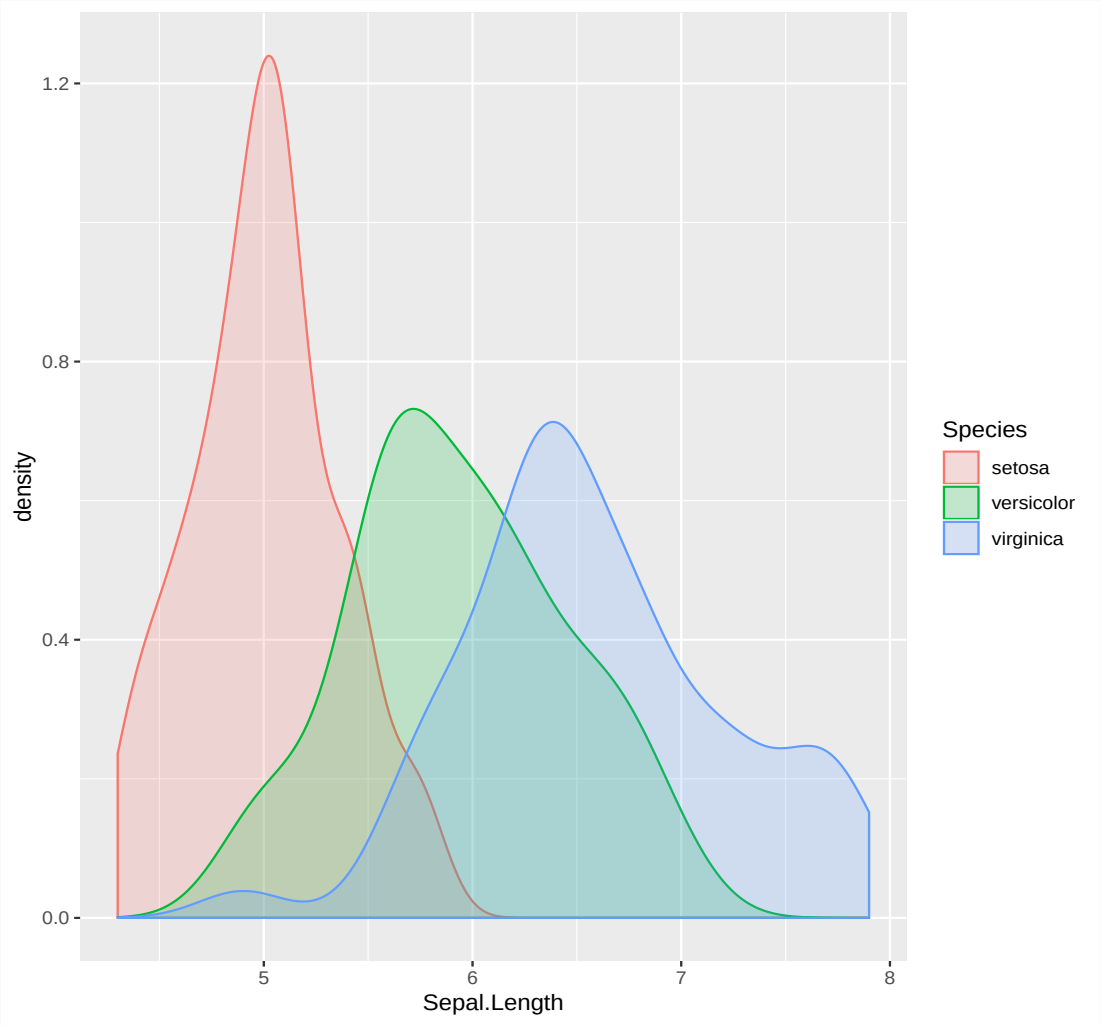
Código básico: histogram

```
ggplot(data = iris, aes(x = Sepal.Width)) +
    geom_histogram(binwidth = 0.2, color = "black", aes(fill = Species)) +
    facet_grid(Species ~.) +
    geom_density(alpha=.7, fill="#FF6666")
```



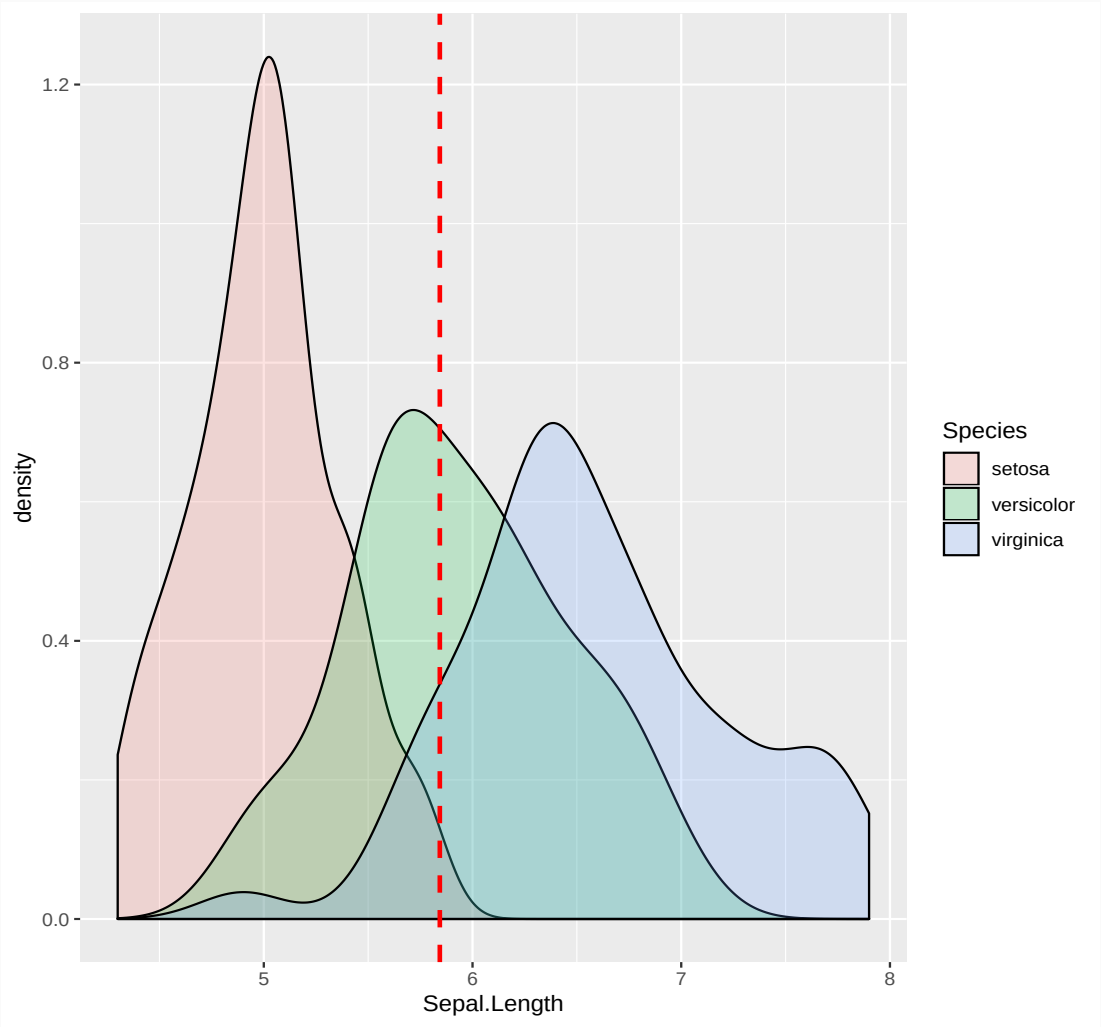
Código básico: density

```
ggplot(data = iris,aes(x = Sepal.Length, color = Species, fill = Species)) +
    geom_density(alpha = 0.2)
```



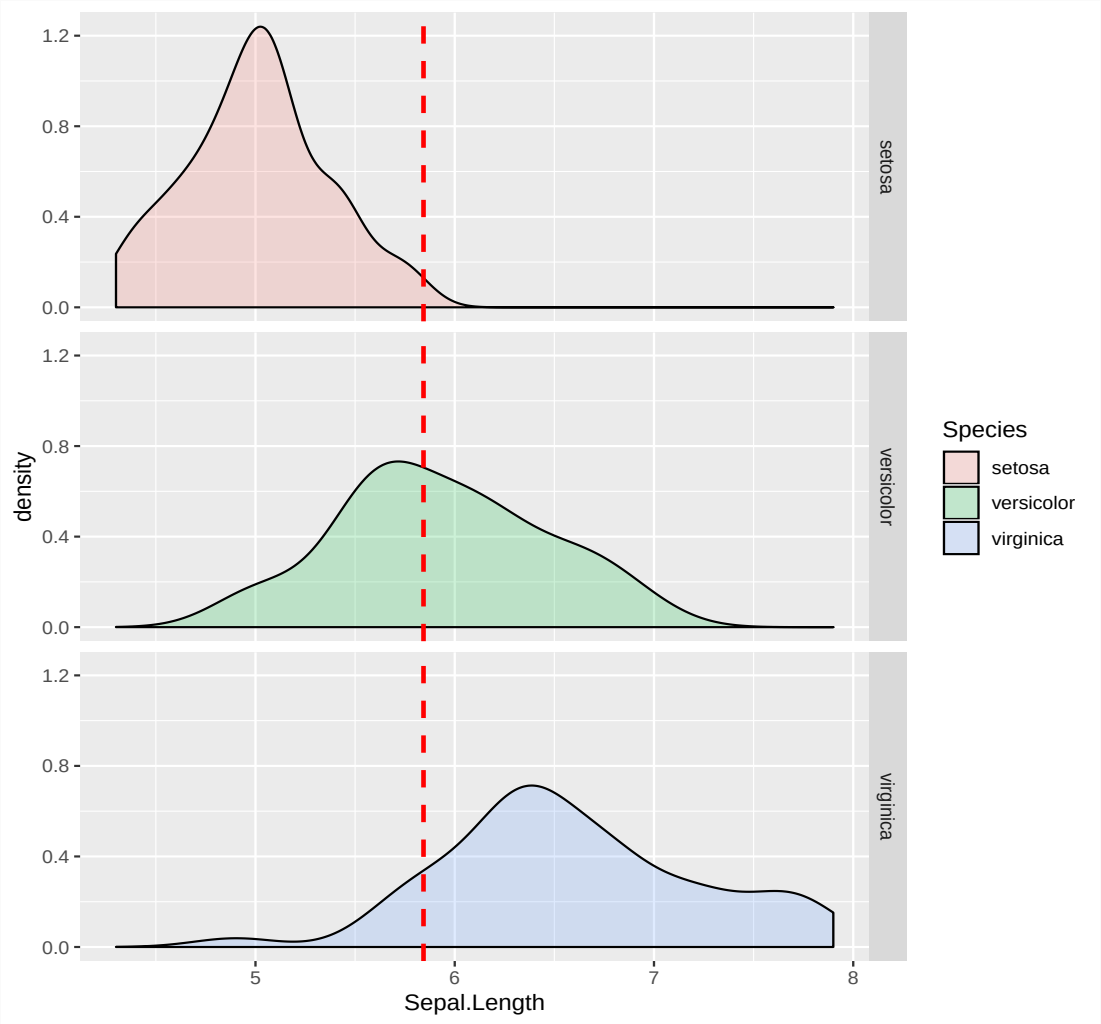
Código básico: density

```
ggplot(data = iris,aes(x = Sepal.Length, fill = Species)) +
    geom_density(alpha = 0.2) +
    geom_vline(aes(xintercept = mean(Sepal.Length)), color="red", linetype = "dashed", size = 1)
```



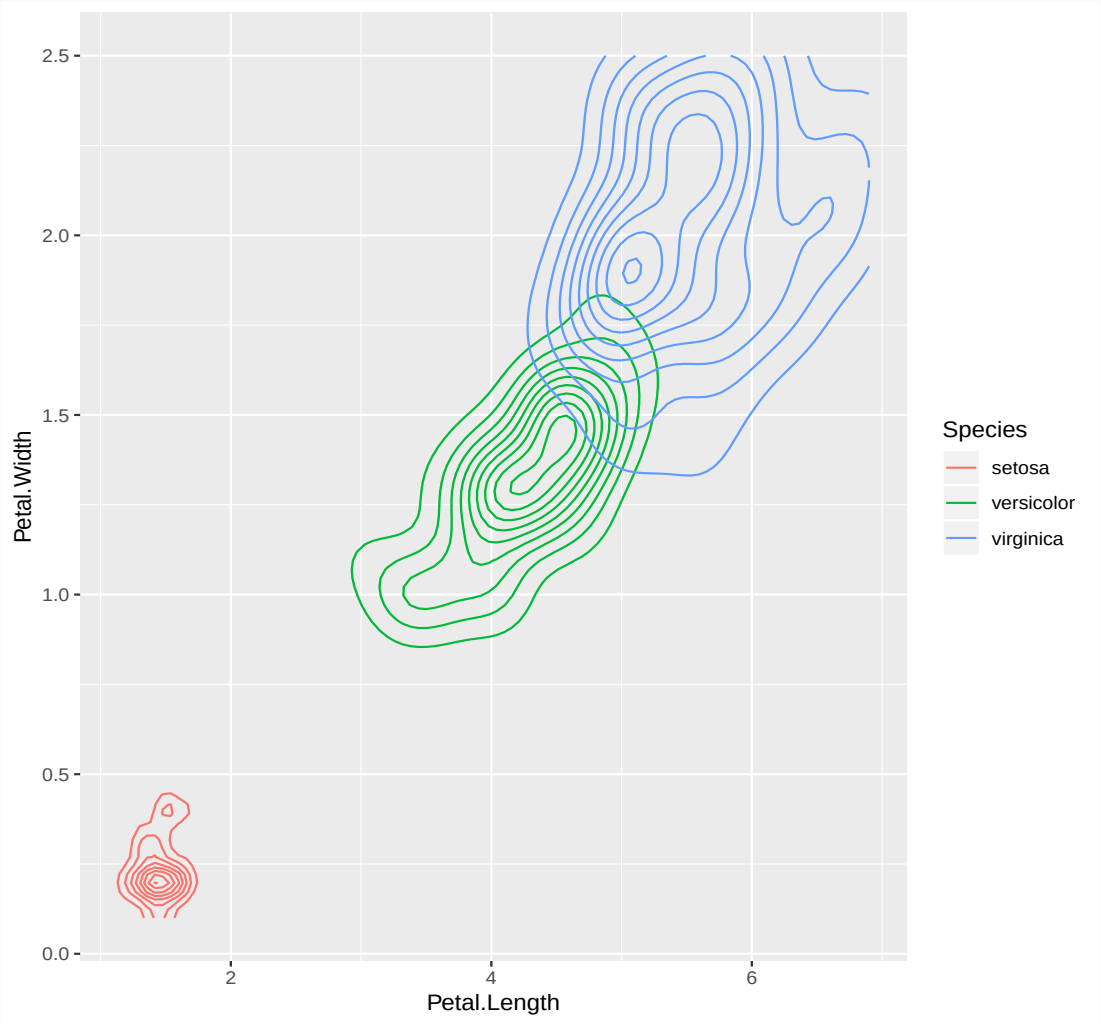
Código básico: density

```
ggplot(data = iris,aes(x = Sepal.Length, fill = Species)) +
    geom_density(alpha = 0.2) +
    geom_vline(aes(xintercept = mean(Sepal.Length)), color="red", linetype = "dashed", size = 1) +
    facet_grid(Species~.)
```



Código básico: density

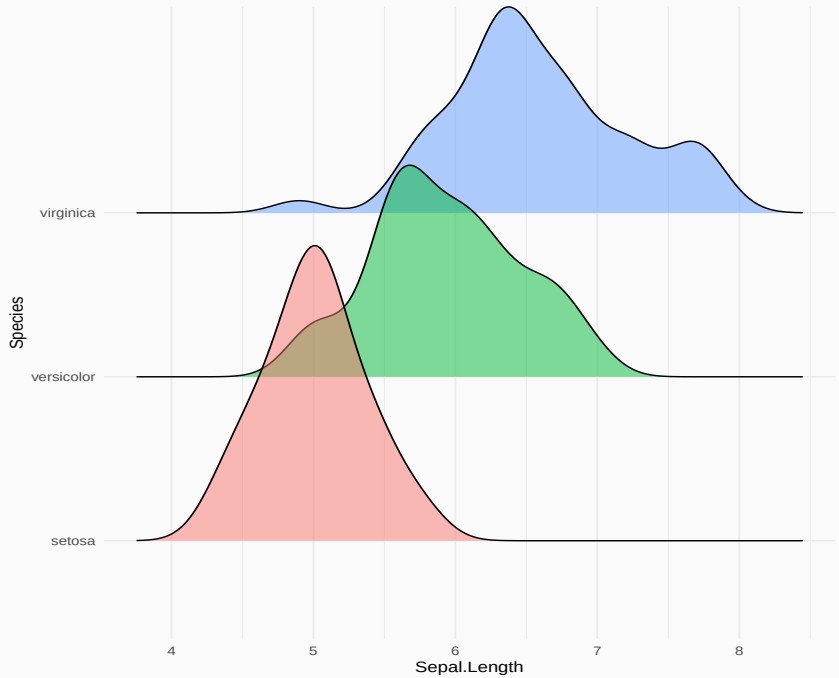
```
ggplot(data = iris,aes(x = Petal.Length, Petal.Width, color = Species)) +
   geom_density2d()
```



Código básico: density

```
library(ggridges)
ggplot(iris, aes(x = Sepal.Length, y = Species, fill = Species)) +
  geom_density_ridges(alpha = 0.5) +
  theme_minimal() +
  theme(
      legend.position="none",
      panel.spacing = unit(0.1, "lines"),
      strip.text.x = element_text(size = 8)
  )
```

```
## Picking joint bandwidth of 0.181
```

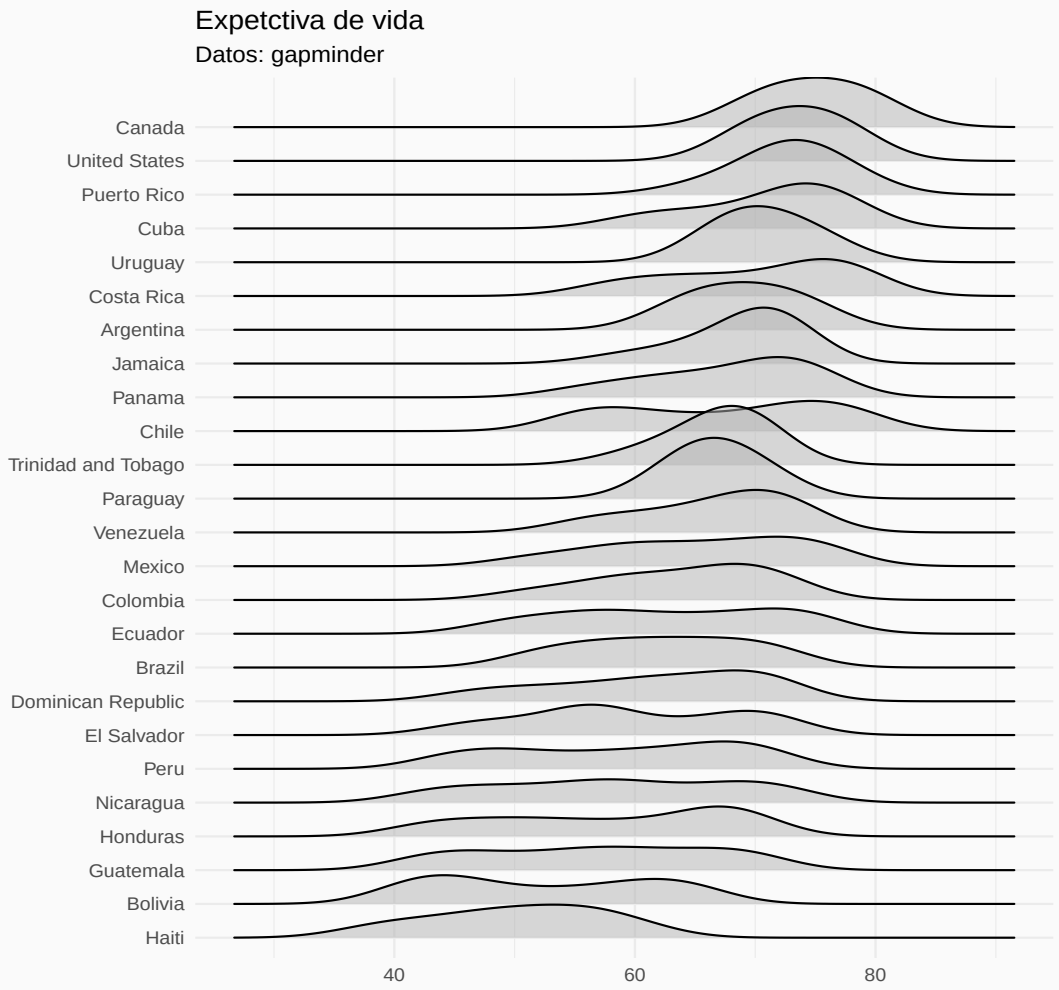


Código básico: density

```
gapminder %>%
  filter(continent == 'Americas') %>%
  group_by(country) %>%
  mutate(media = mean(lifeExp)) %>%
  ggplot(aes(x = lifeExp, y = reorder(country, media))) +
      geom_density_ridges(alpha = 0.5) +
      theme_minimal() +
      theme(
          legend.position="none",
          panel.spacing = unit(0.1, "lines"),
          strip.text.x = element_text(size = 8)
           ) +
      labs(x = "",
           y = "",
           title = "Expetctiva de vida",
           subtitle = "Datos: gapminder"
           )
```

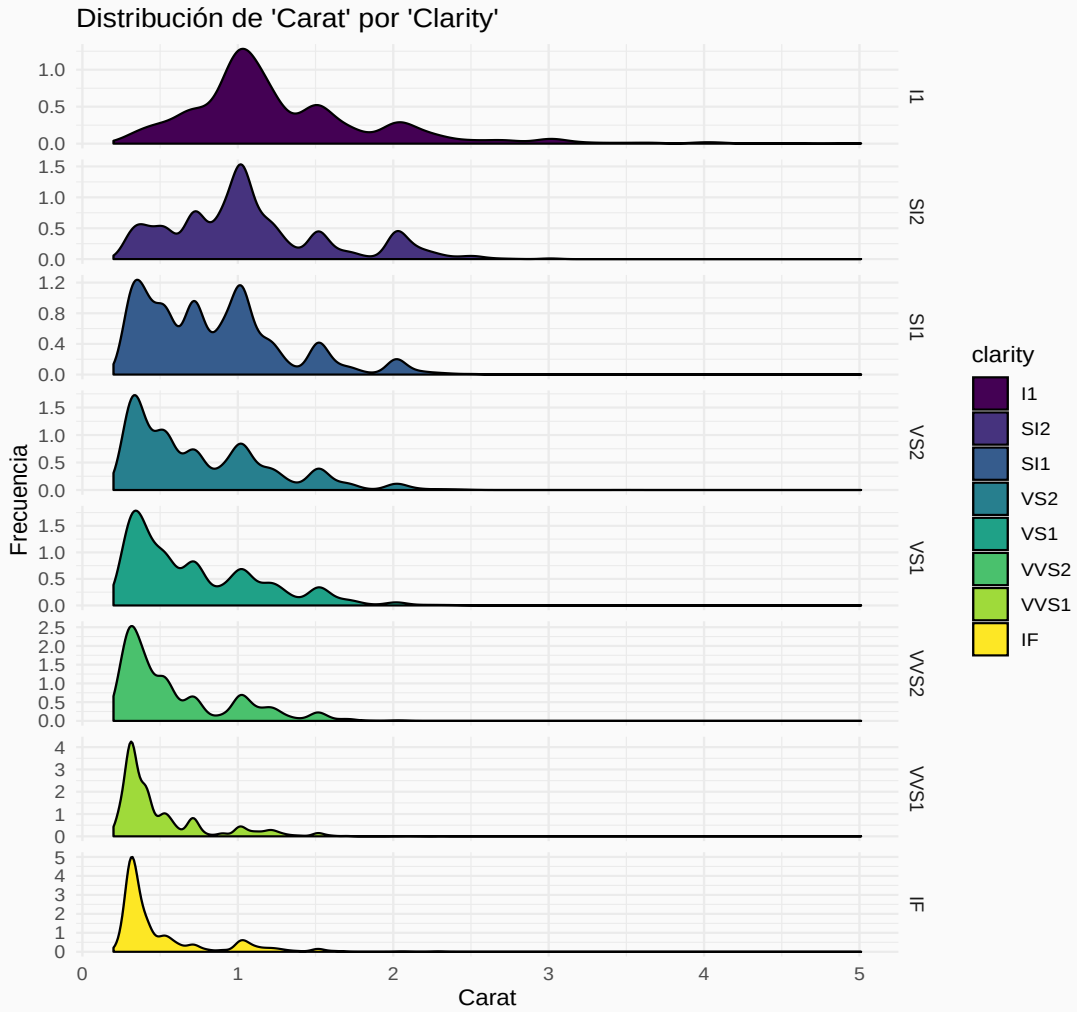
Código básico: density

```
## Picking joint bandwidth of 3.63
```



Código básico: density

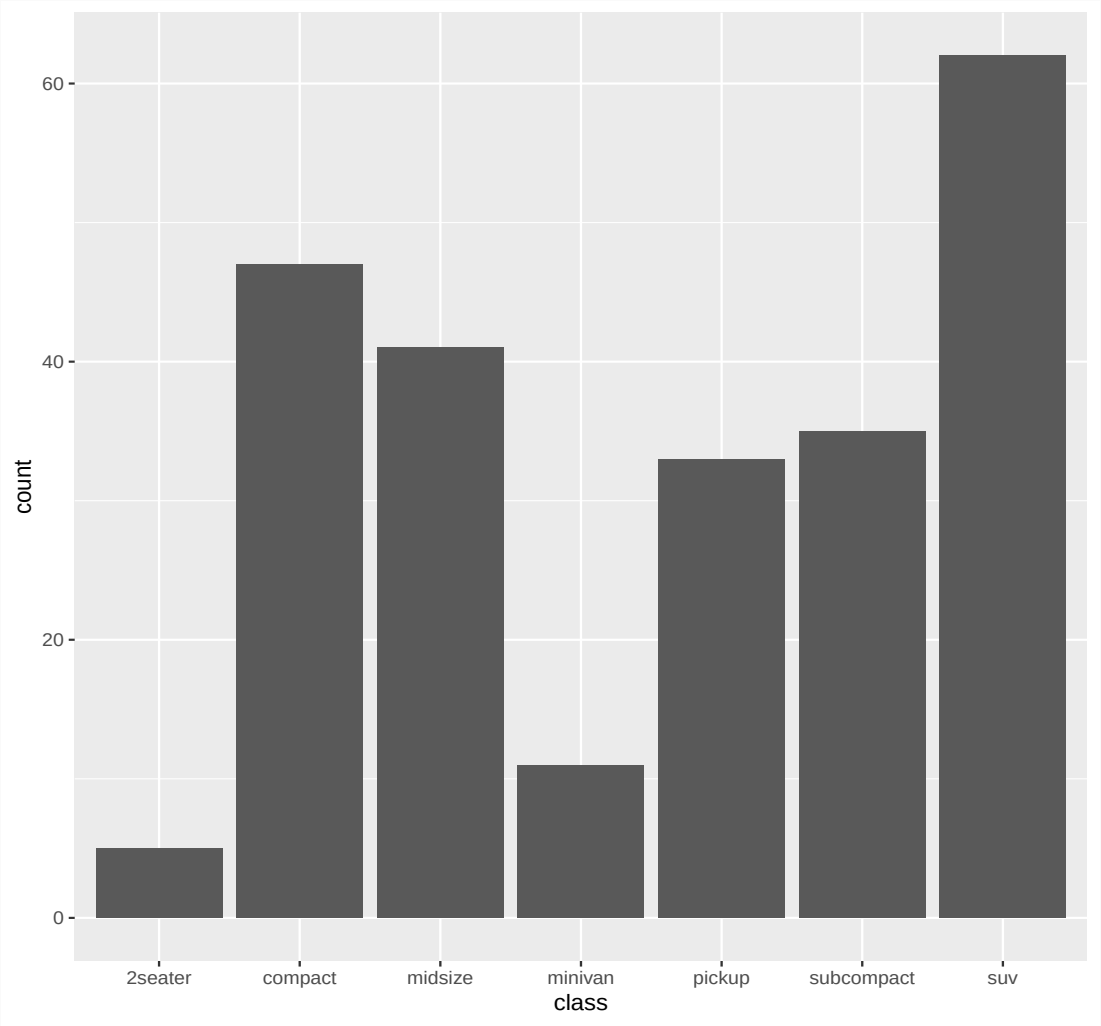
```
ggplot(diamonds) +
    geom_density(aes(x = carat, fill = clarity), position = 'stack') +
    facet_grid(clarity ~., scales = 'free') +
    labs(x = "Carat", y = "Frecuencia", title = "Distribucin de 'Carat' por 'Clarity'") +
    theme_minimal()
```



Barras

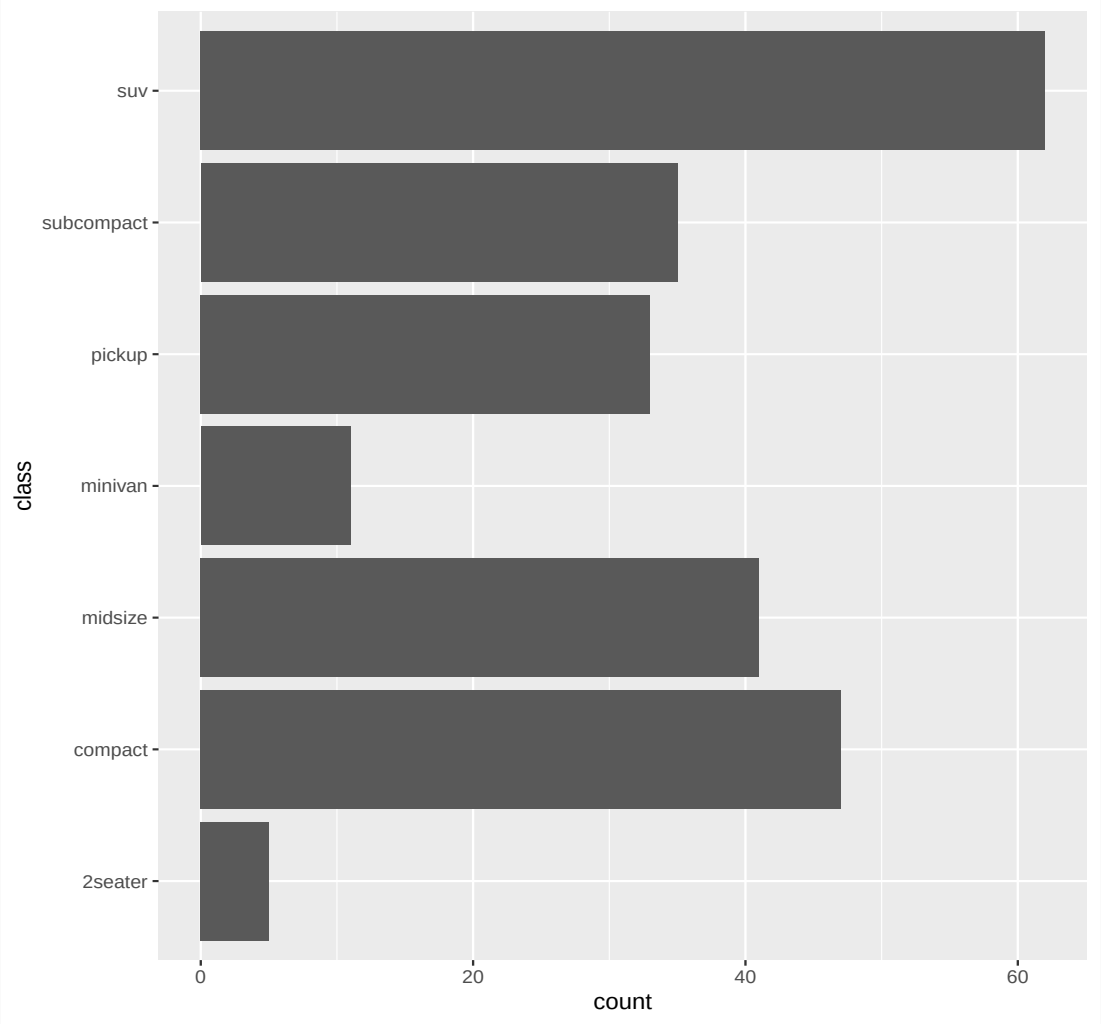
Código básico: barplot

```
ggplot(mpg, aes(x = class)) +
  geom_bar()
```



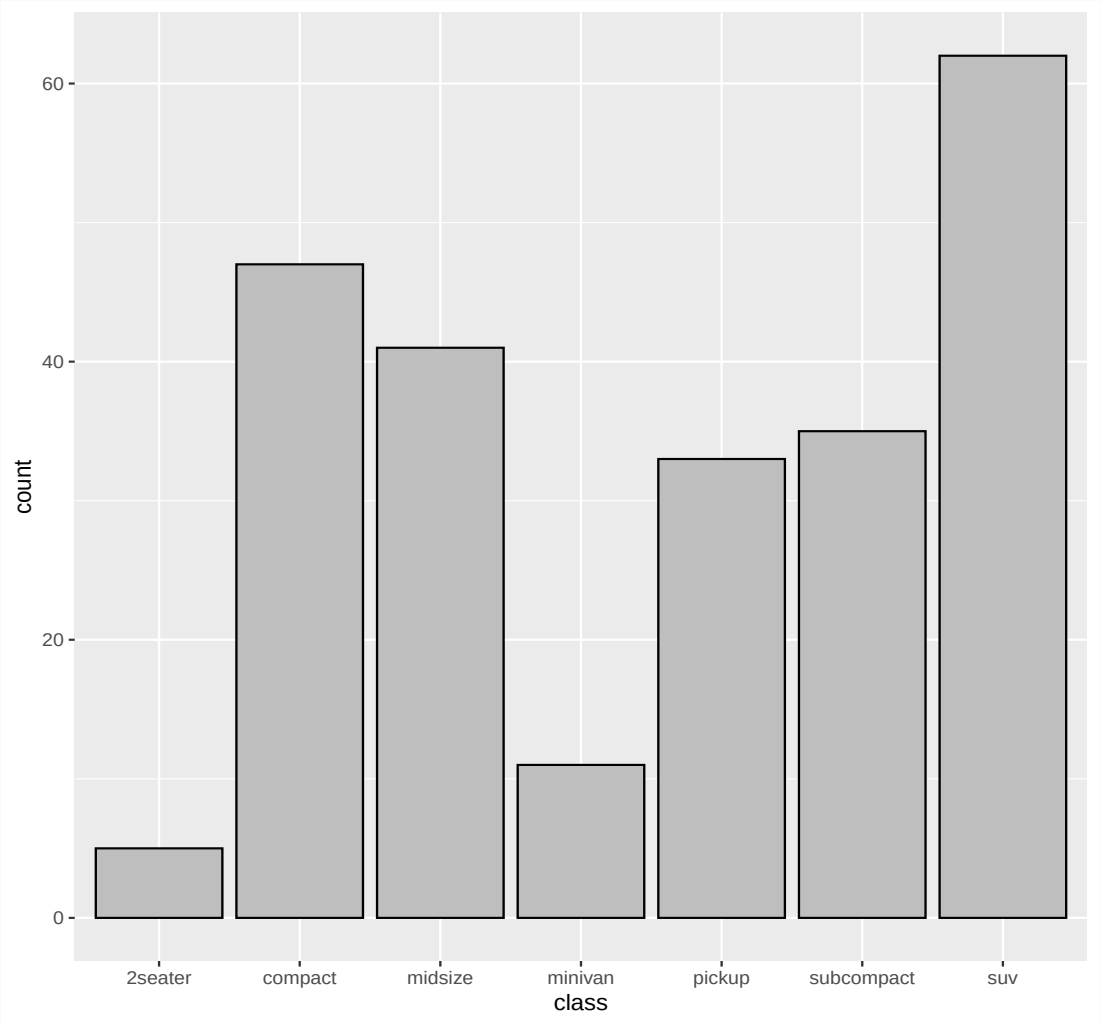
Código básico: barplot

```
ggplot(mpg, aes(x = class)) +
  geom_bar() +
  coord_flip()
```



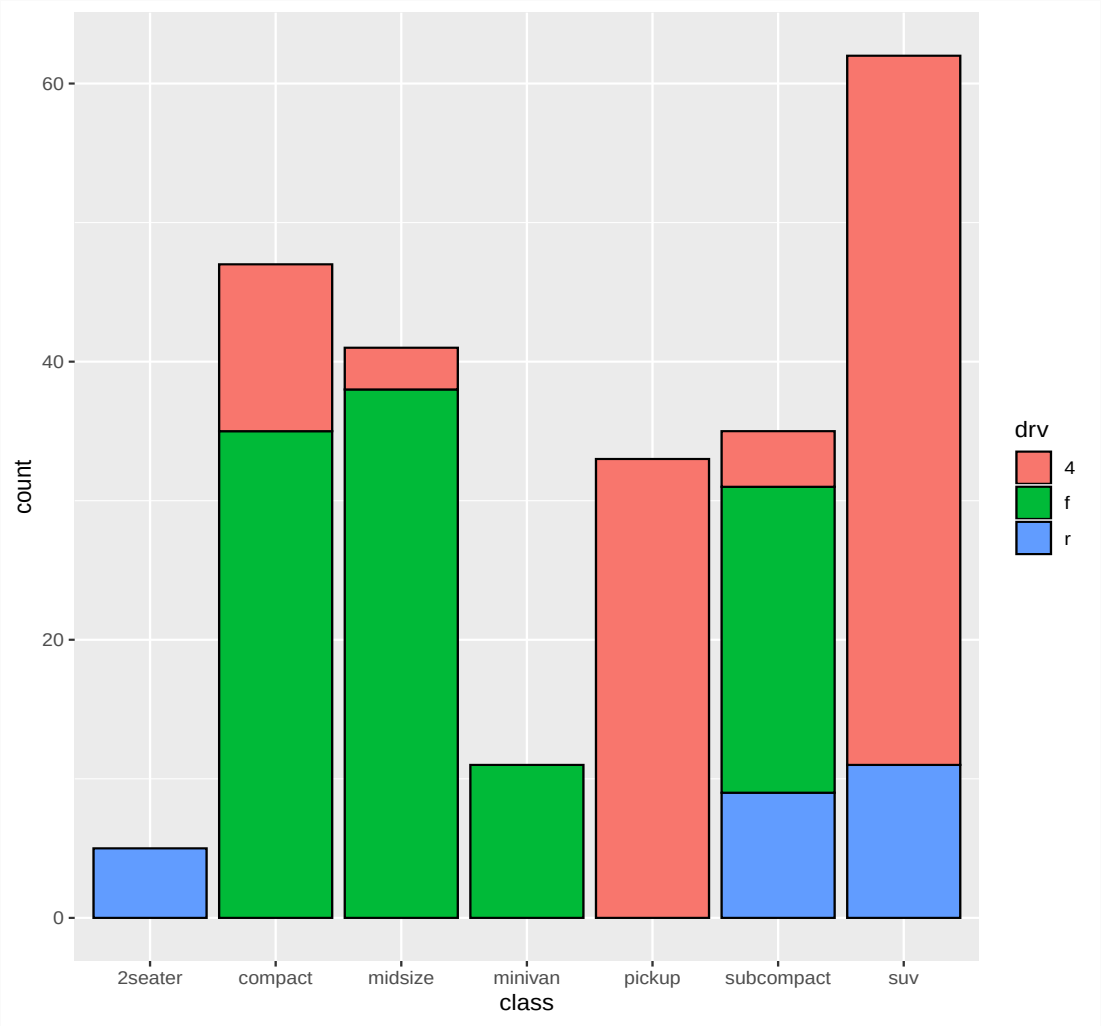
Código básico: barplot

```
ggplot(mpg, aes(x = class)) +
  geom_bar(fill = 'gray', color = 'black')
```



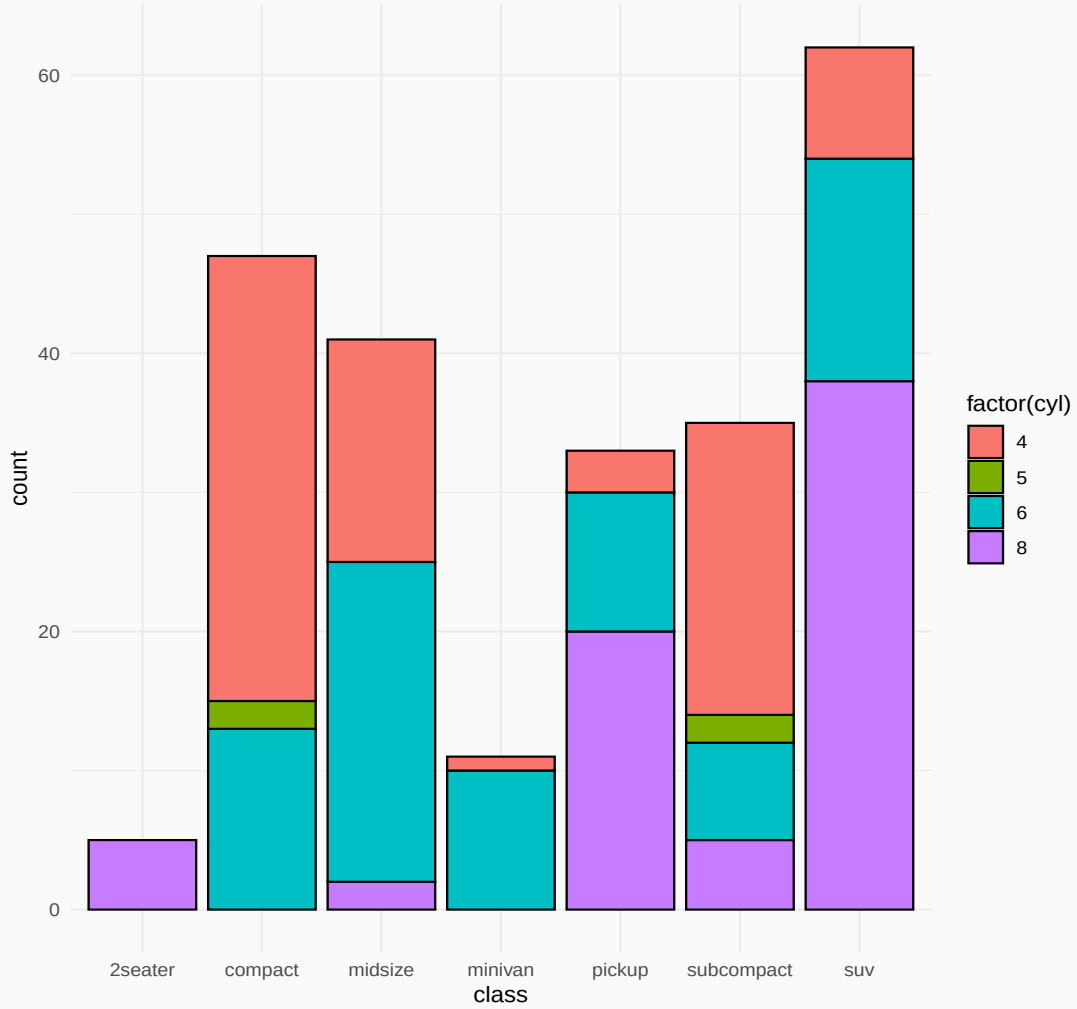
Código básico: barplot

```
ggplot(mpg, aes(x = class, fill = drv)) +
  geom_bar( color = 'black')
```



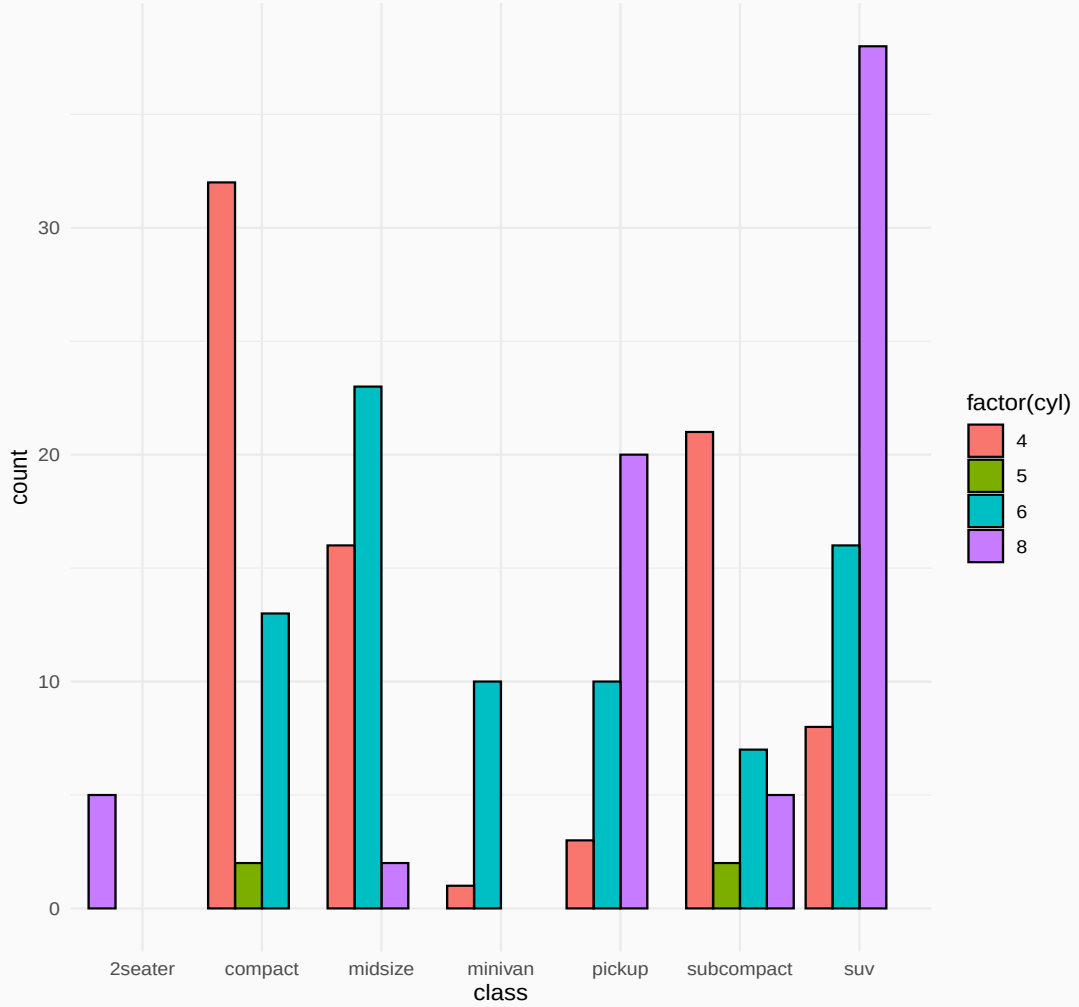
Código básico: barplot

```
ggplot(mpg, aes(x = class, fill = factor(cyl))) +
  geom_bar( color = 'black') +
  theme_minimal()
```



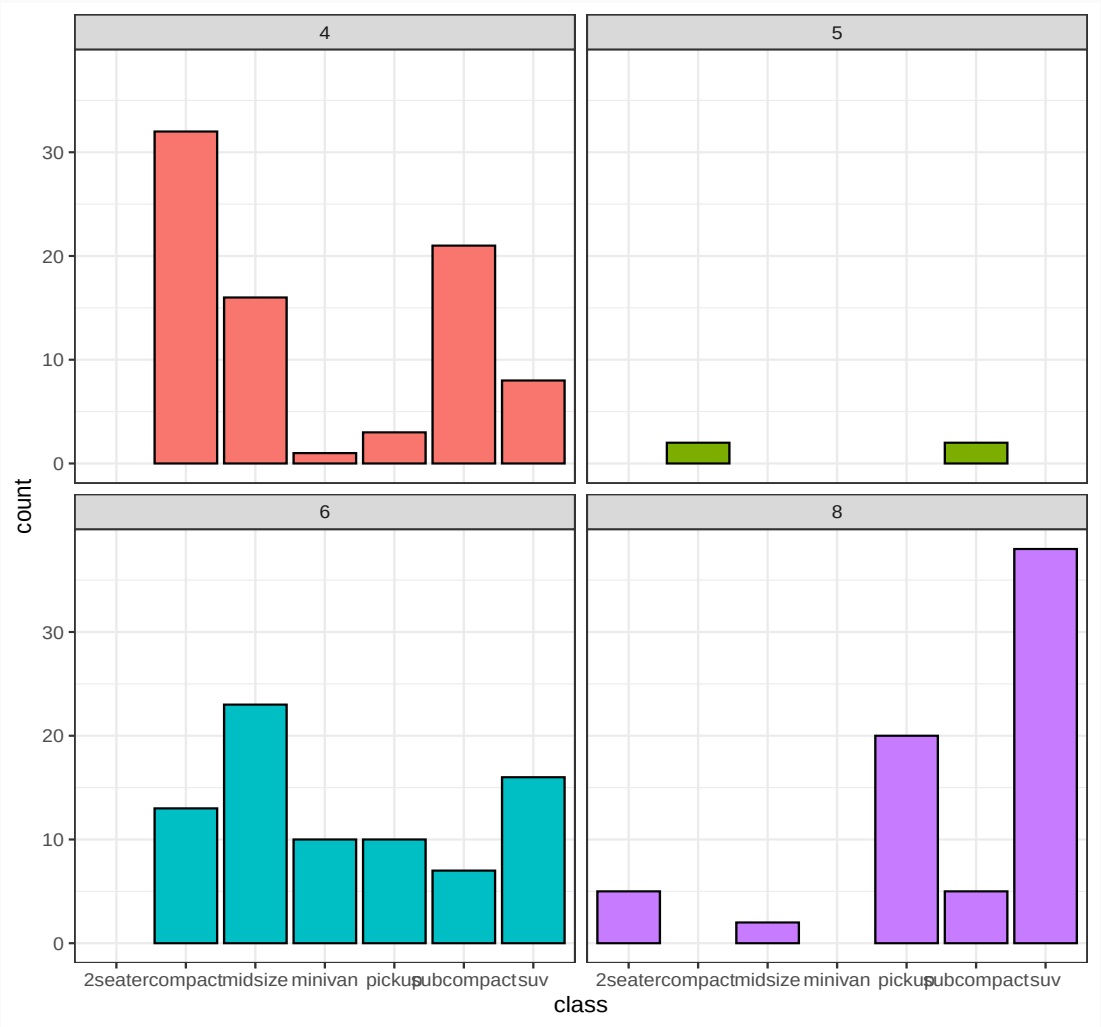
Código básico: barplot

```
ggplot(mpg, aes(x = class, fill = factor(cyl))) +
  geom_bar(color = 'black',
           position = position_dodge(preserve = 'single')
          ) +
  theme_minimal()
```



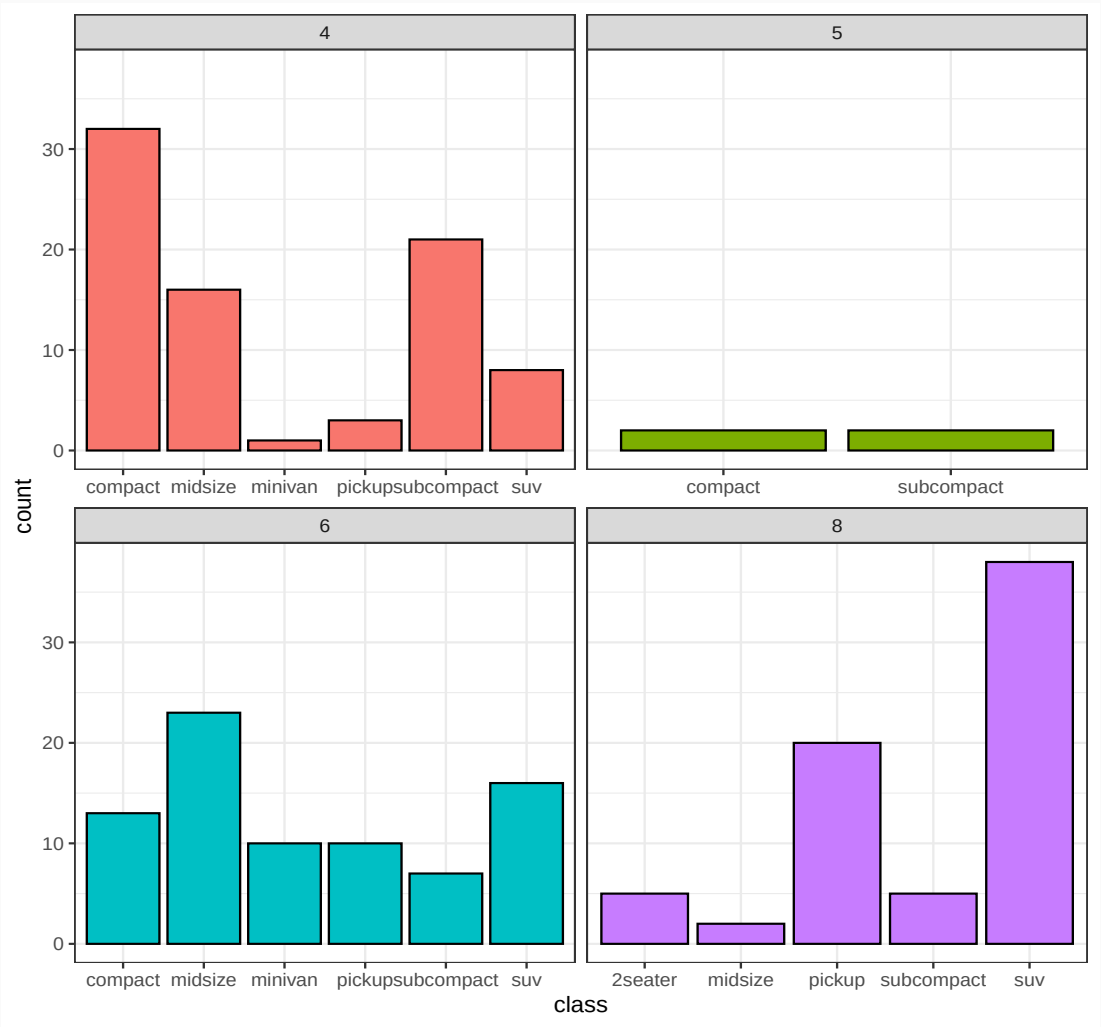
Código básico: barplot

```
ggplot(mpg, aes(x = class, fill = factor(cyl))) +
  geom_bar(color = 'black') +
  theme_bw() +
  facet_wrap( ~ factor(cyl)) +
  theme(legend.position = "none")
```



Código básico: barplot

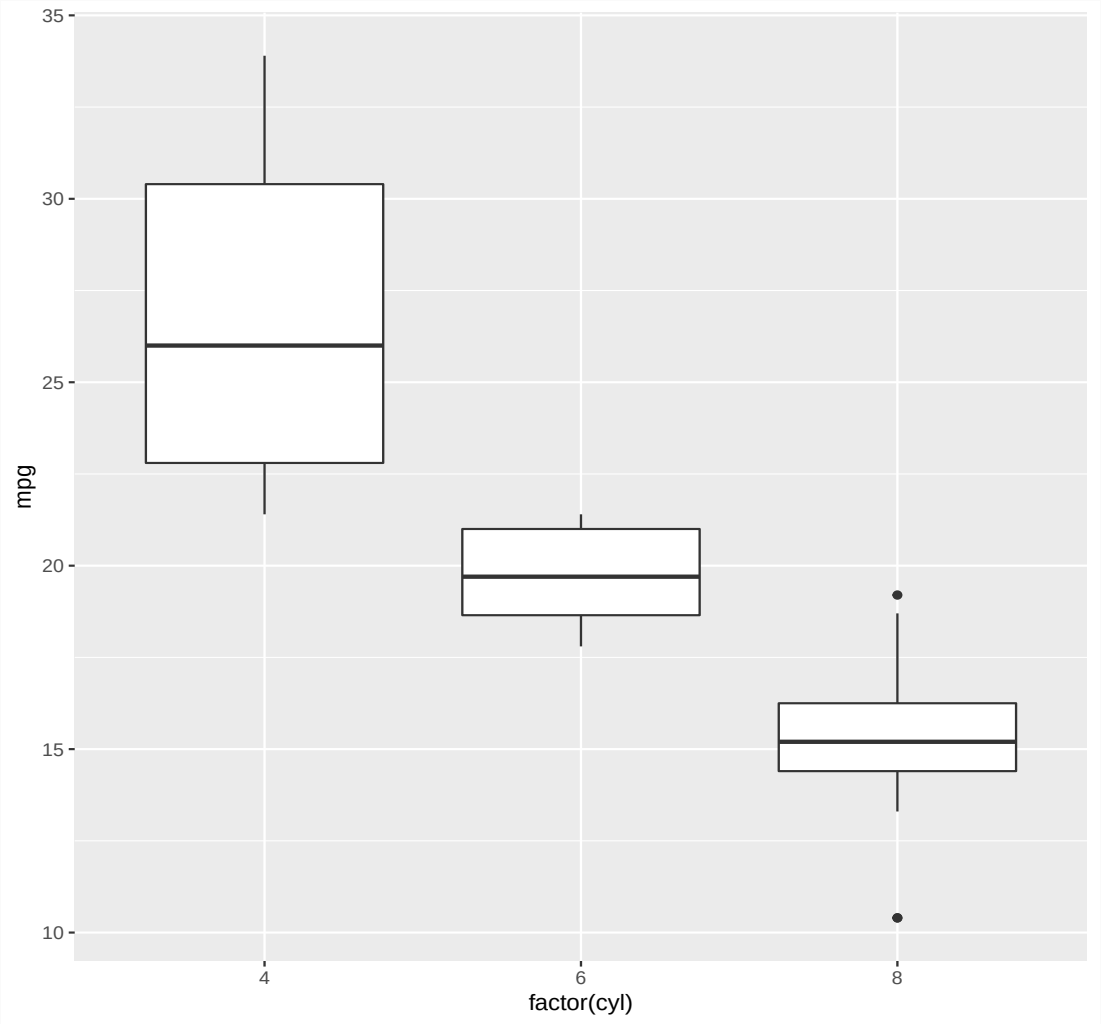
```
ggplot(mpg, aes(x = class, fill = factor(cyl))) +
  geom_bar(color = 'black') +
  theme_bw() +
  facet_wrap( ~ factor(cyl), scales = "free_x") +
  theme(legend.position = "none")
```



Boxplot

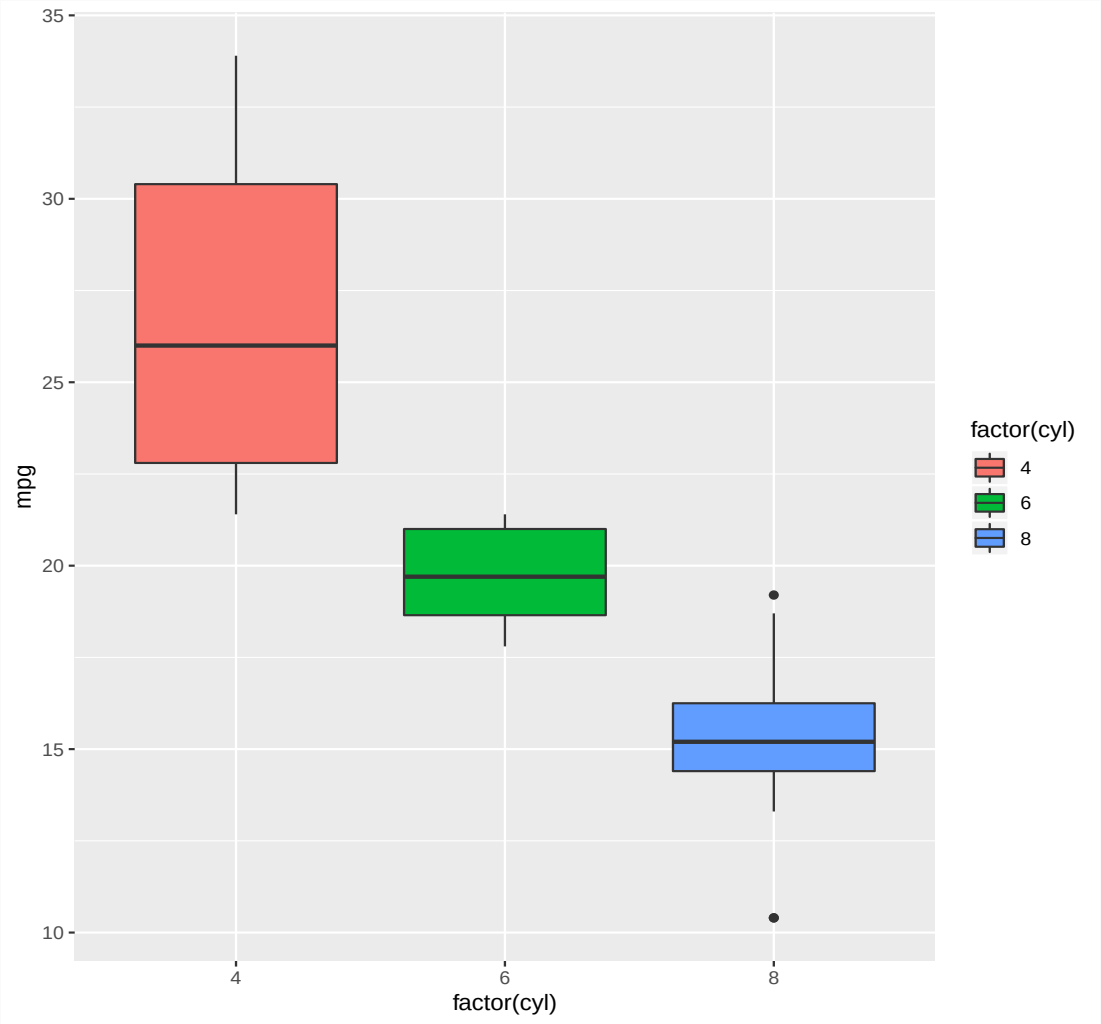
Código básico: boxplot

```
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
    geom_boxplot()
```



Código básico: boxplot

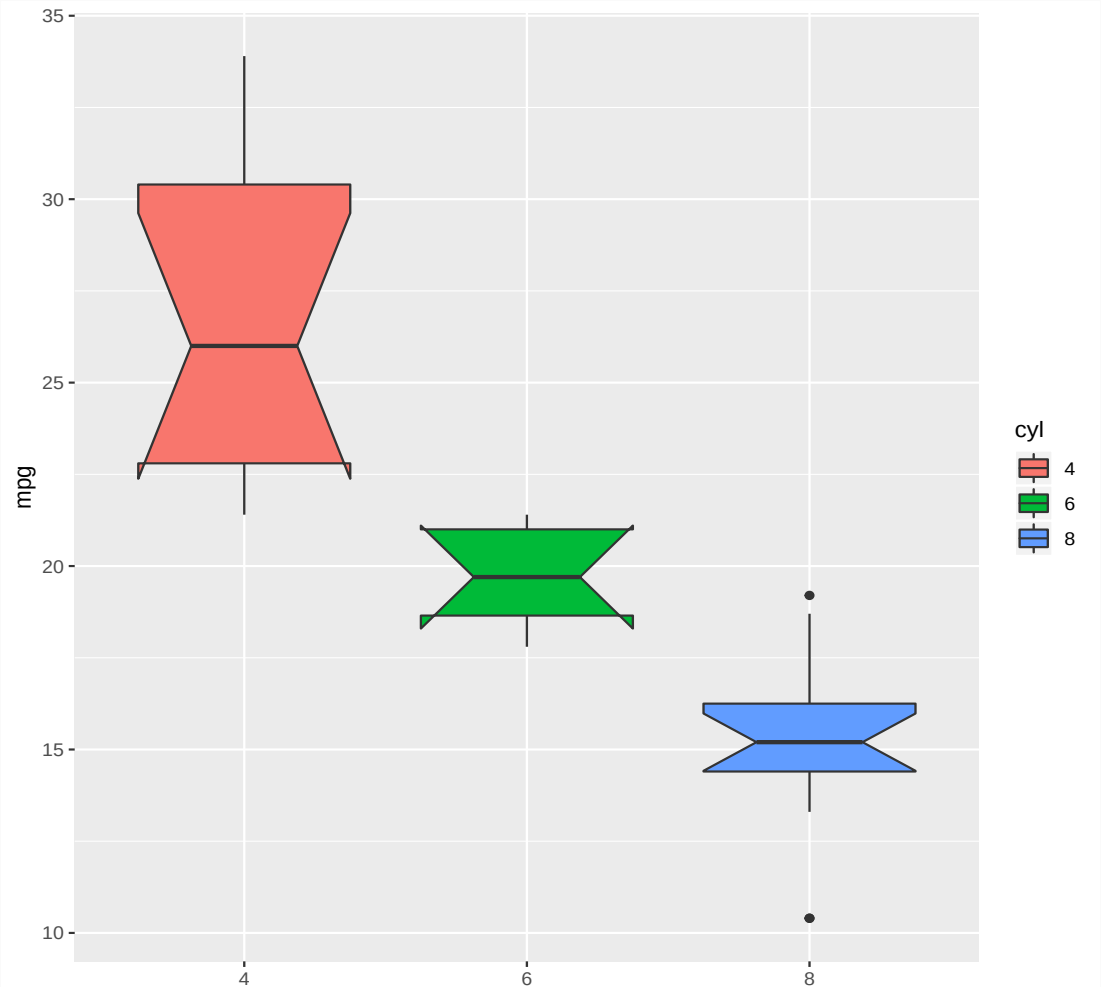
```
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
    geom_boxplot(aes(fill = factor(cyl)))
```



Código básico: boxplot

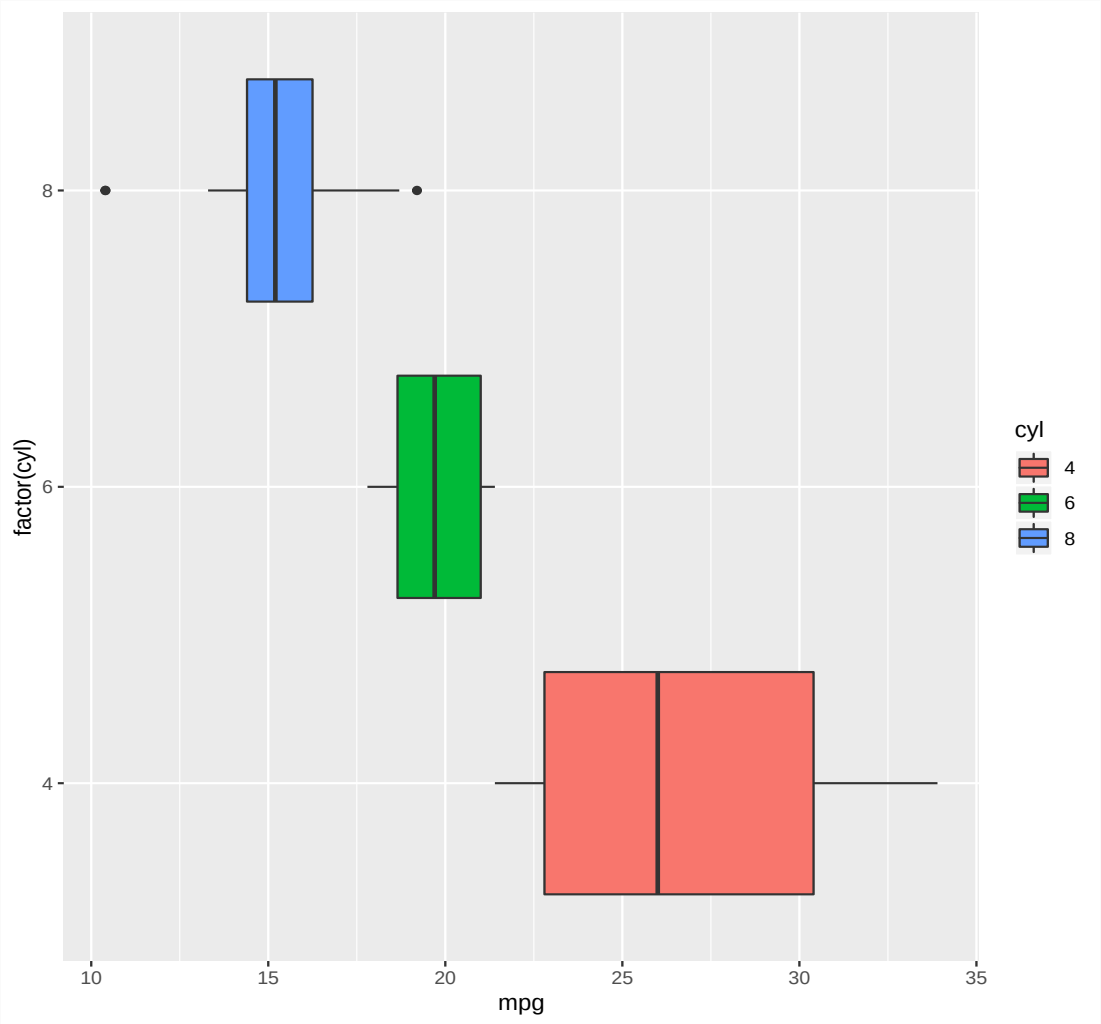
```
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
    geom_boxplot(aes(fill = factor(cyl)), notch = TRUE) +
    labs(fill = "cyl")
```

```
## notch went outside hinges. Try setting notch=FALSE.
## notch went outside hinges. Try setting notch=FALSE.
```



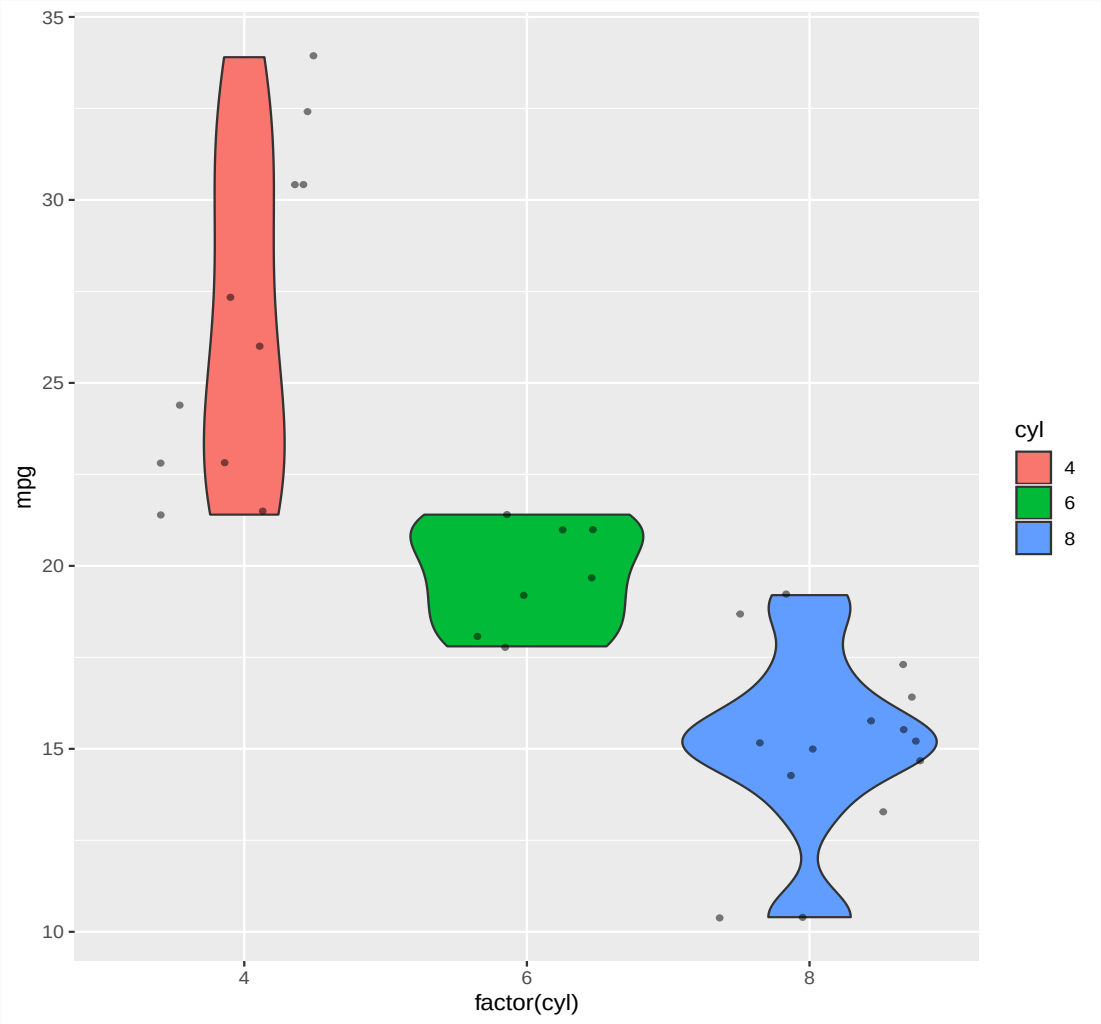
Código básico: boxplot

```
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
    geom_boxplot(aes(fill = factor(cyl))) +
    coord_flip() +
    labs(fill = "cyl")
```



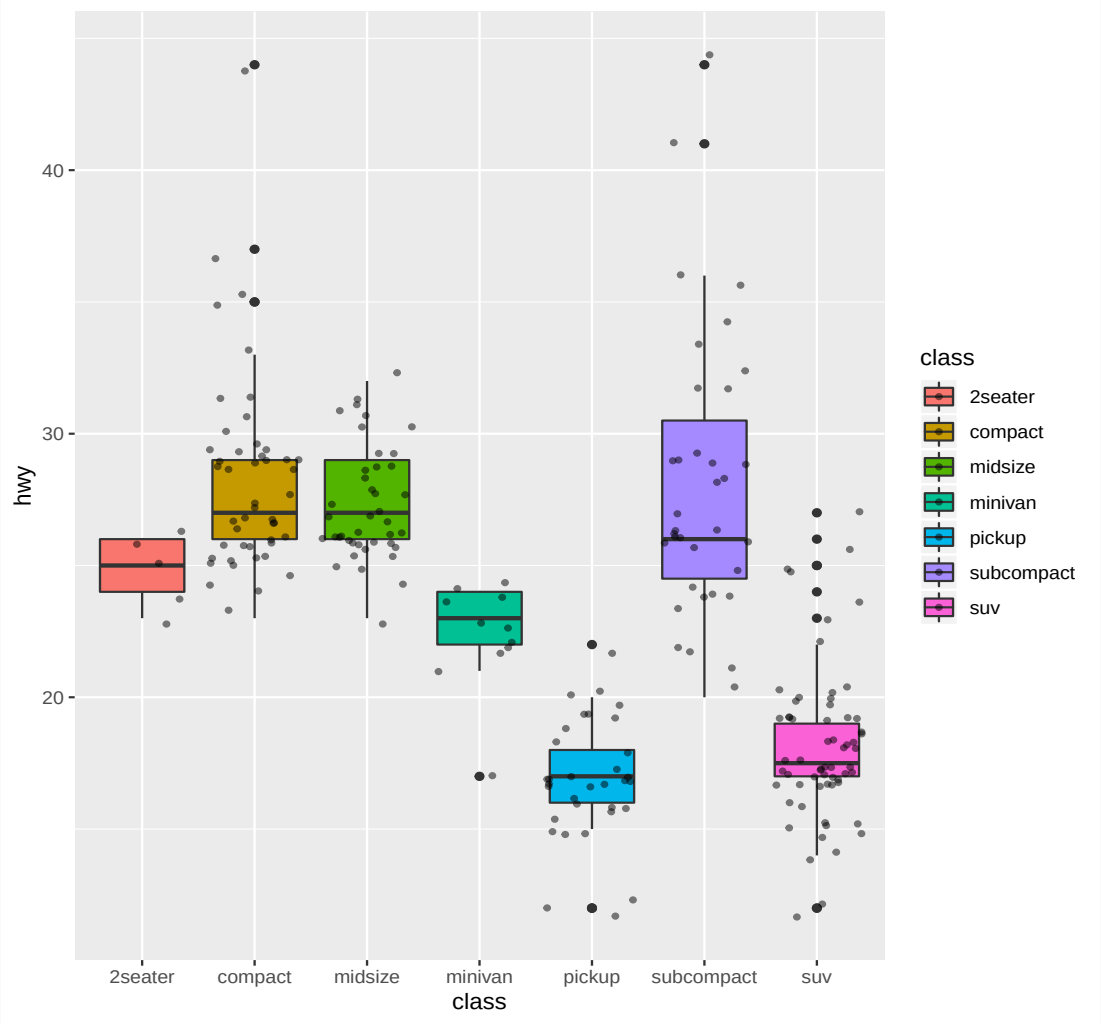
Código básico: boxplot

```
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
    geom_violin(aes(fill = factor(cyl))) +
    geom_jitter(color = "black", size = 1, alpha = 0.5) +
    labs(fill = "cyl")
```



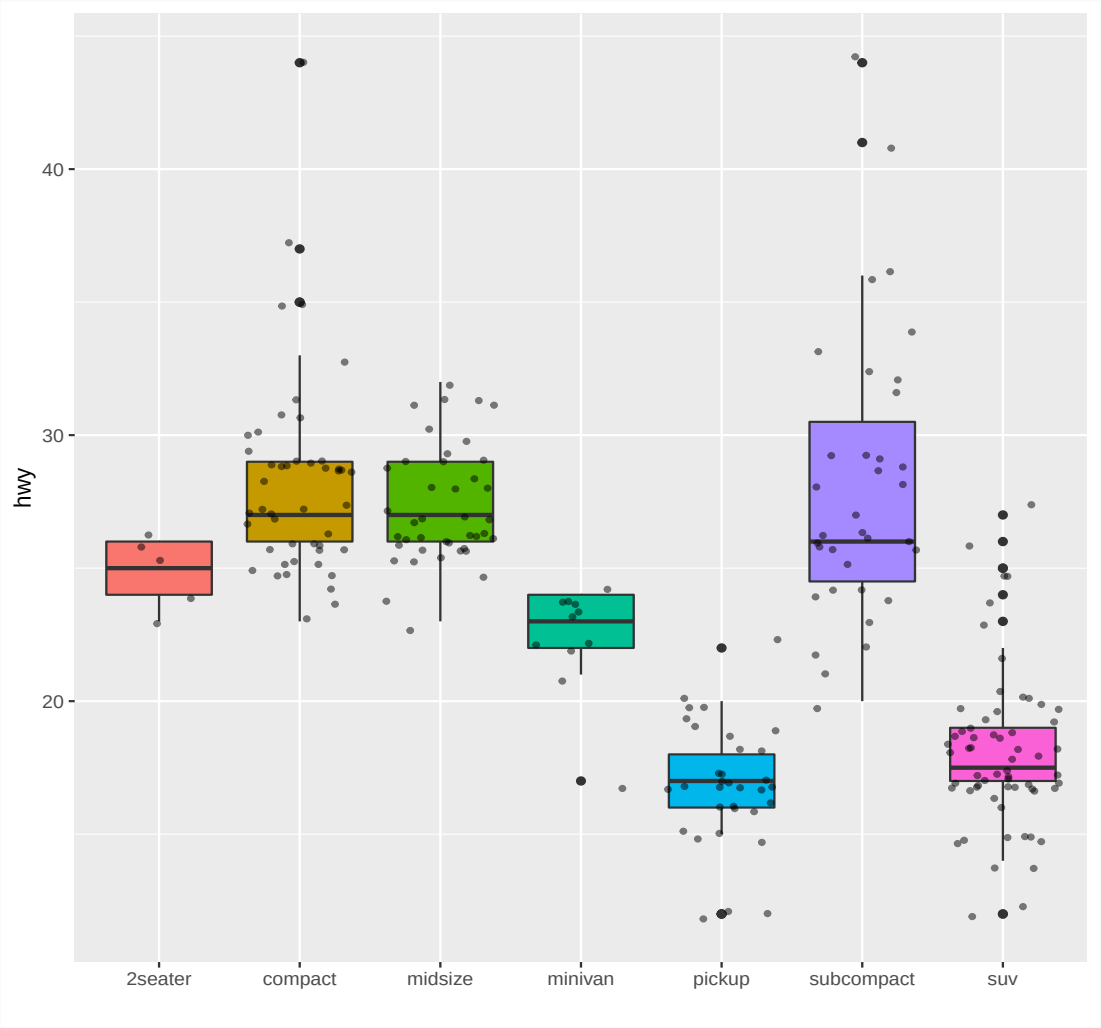
Código básico: boxplot

```
ggplot(mpg,  aes(x=class, y=hwy, fill=class)) +
    geom_boxplot() +
    geom_jitter(color = "black", size = 1, alpha = 0.5)
```



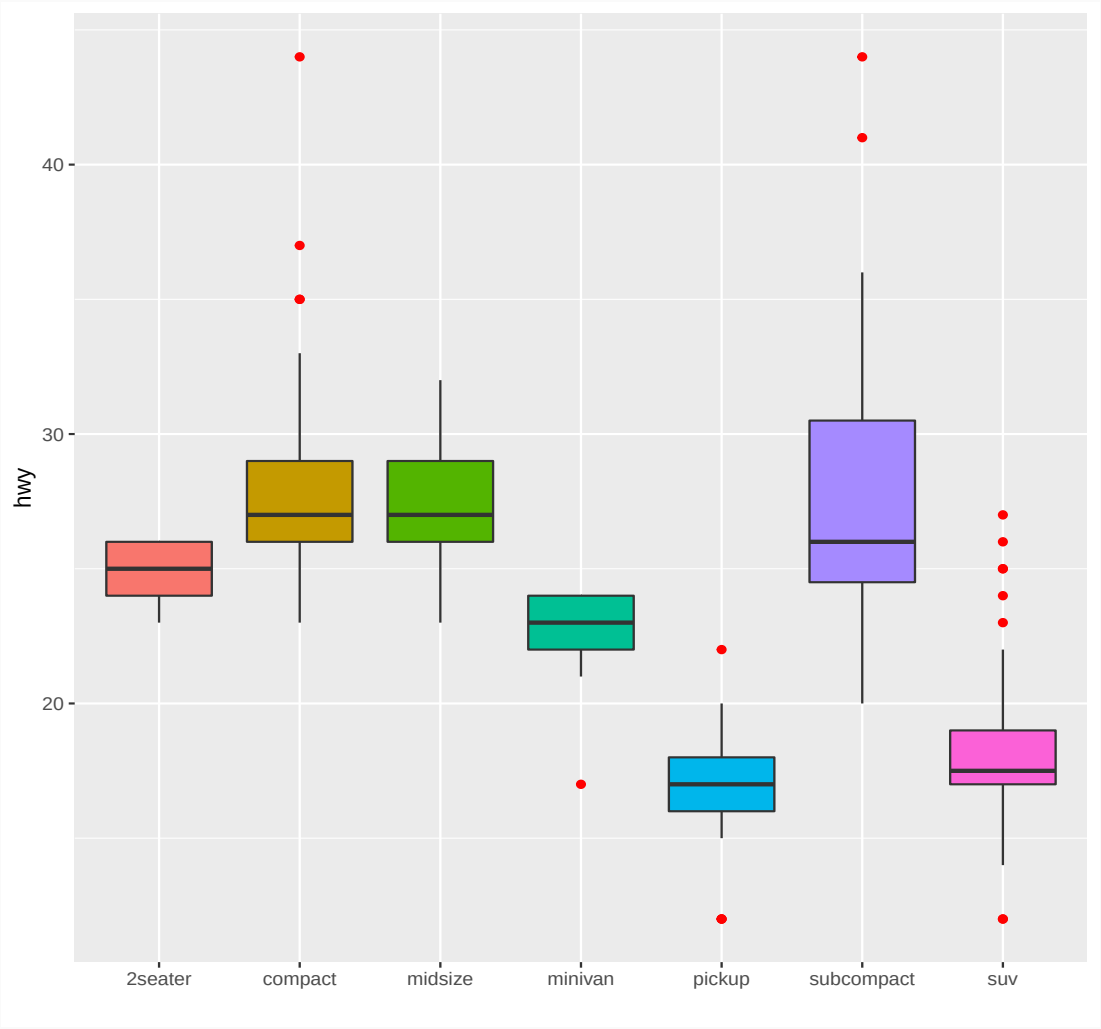
Código básico: boxplot

```
ggplot(mpg,  aes(x=class, y=hwy, fill=class)) +
    geom_boxplot() +
    geom_jitter(color = "black", size = 1, alpha = 0.5) +
    theme(legend.position="none") +
    xlab("") +
    xlab("")
```



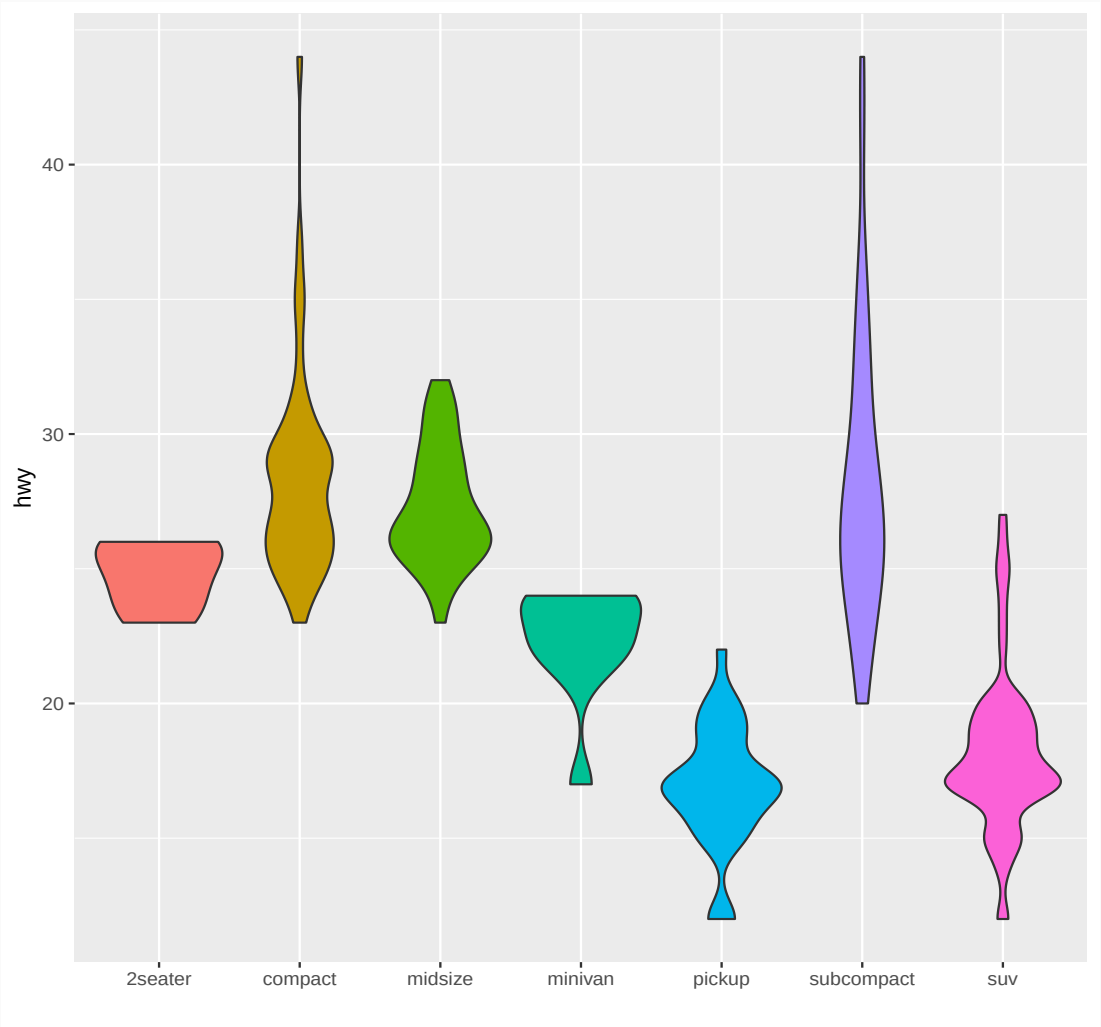
Código básico: boxplot

```
ggplot(mpg,  aes(x=class, y=hwy, fill=class)) +
    geom_boxplot(outlier.colour  =  "red") +
    theme(legend.position="none") +
    xlab("") +
    xlab("")
```



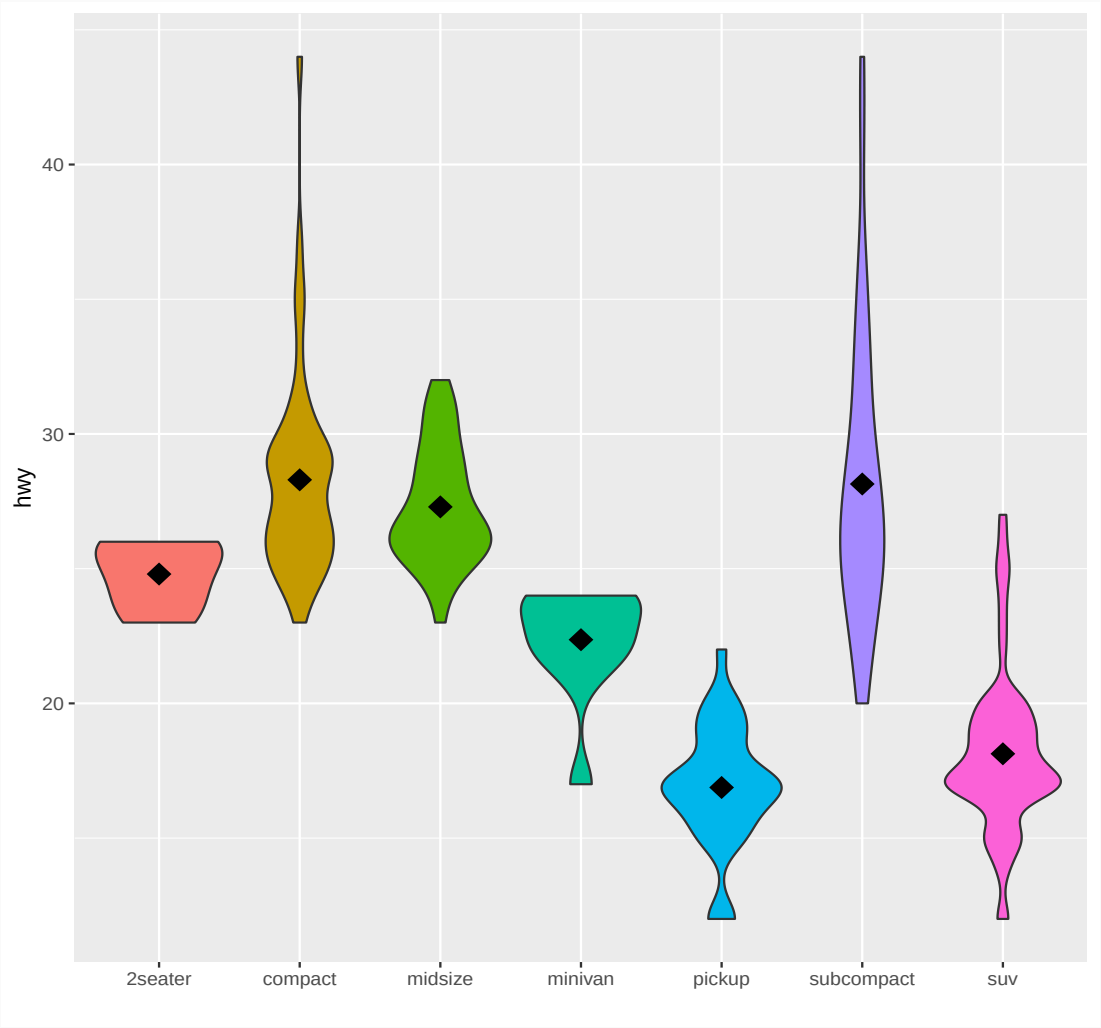
Código básico: boxplot

```
ggplot(mpg,  aes(x=class, y=hwy, fill=class)) +
    geom_violin() +
    theme(legend.position="none") +
    xlab("") +
    xlab("")
```



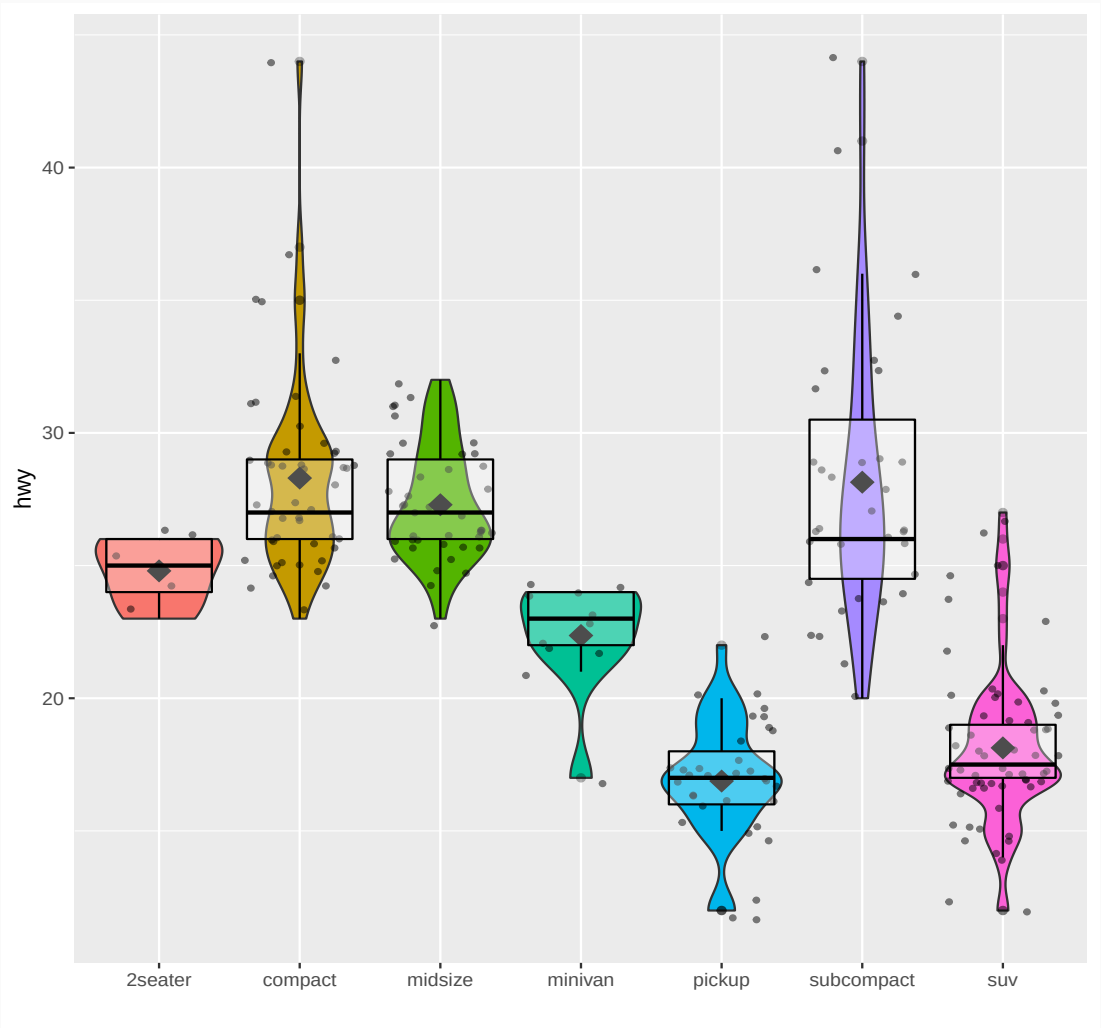
Código básico: boxplot

```
ggplot(mpg,  aes(x=class, y=hwy, fill=class)) +
  geom_violin() +
  theme(legend.position="none") +
  stat_summary(fun.y = mean, geom = "point", shape = 18, size=5, color="black") +
  xlab("") +
  xlab("")
```



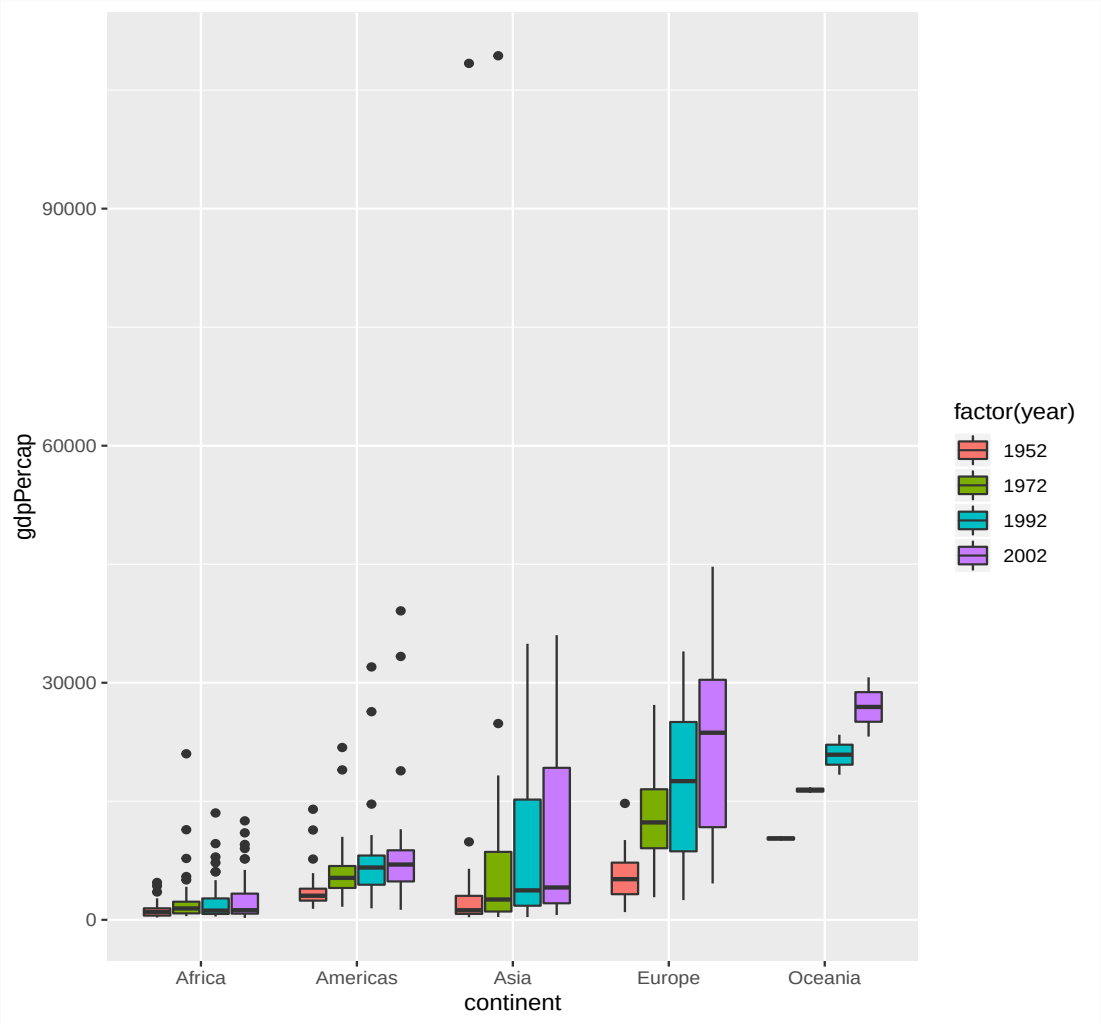
Código básico: boxplot

```
ggplot(mpg,  aes(x=class, y=hwy, fill=class)) +
    geom_violin() +
    geom_jitter(size = 1, color = 'black', alpha = 0.5) +
    stat_summary(fun.y = mean, geom = "point", shape = 18, size=5, color='black') +
    geom_boxplot(color = 'black', fill = 'white', alpha = 0.3) +
    xlab("") + xlab("") + theme(legend.position="none")
```



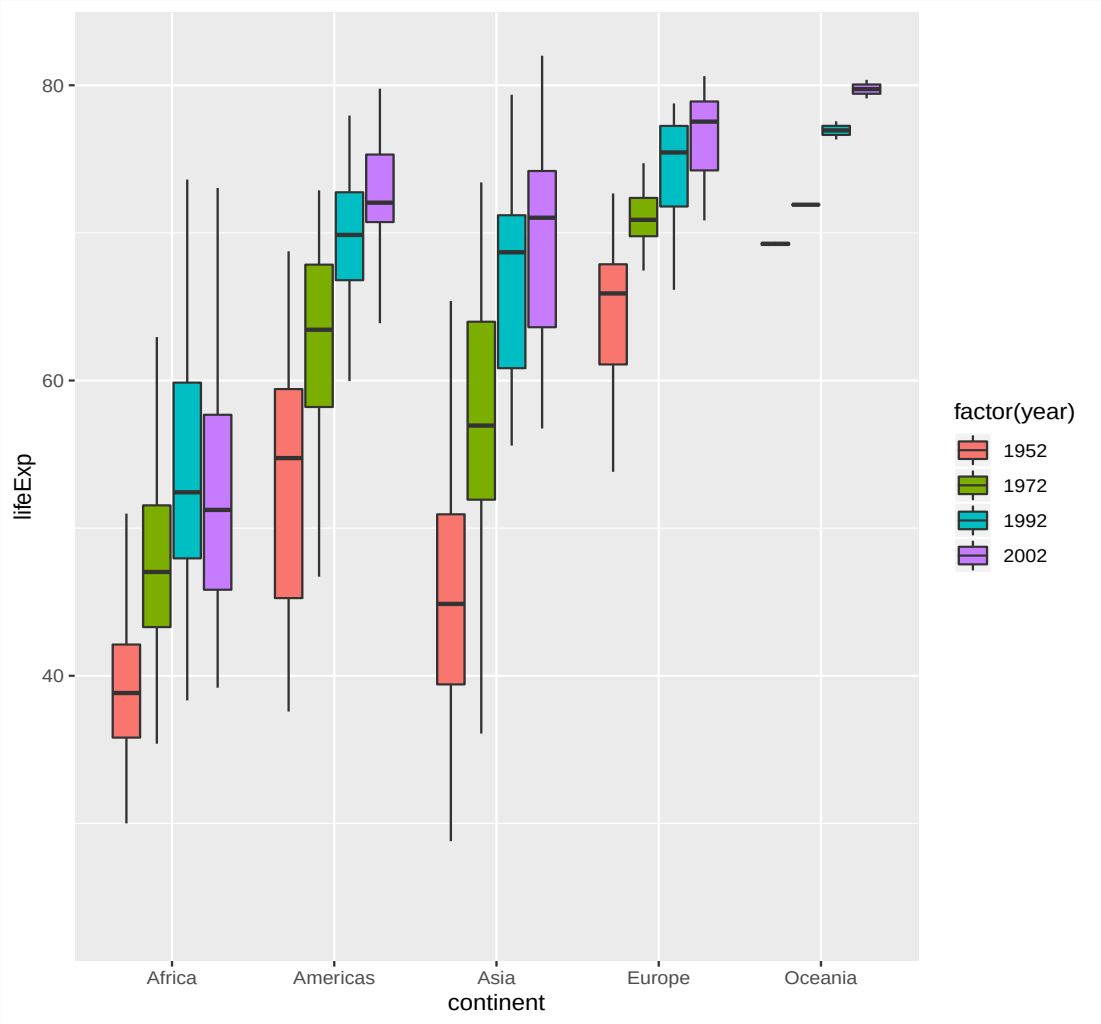
Código básico: boxplot

```
gapminder %>%
    filter(year %in% c(1952,1972,1992, 2002)) %>%
    ggplot(aes(x = continent, y = gdpPercap, fill = factor(year))) +
    geom_boxplot()
```



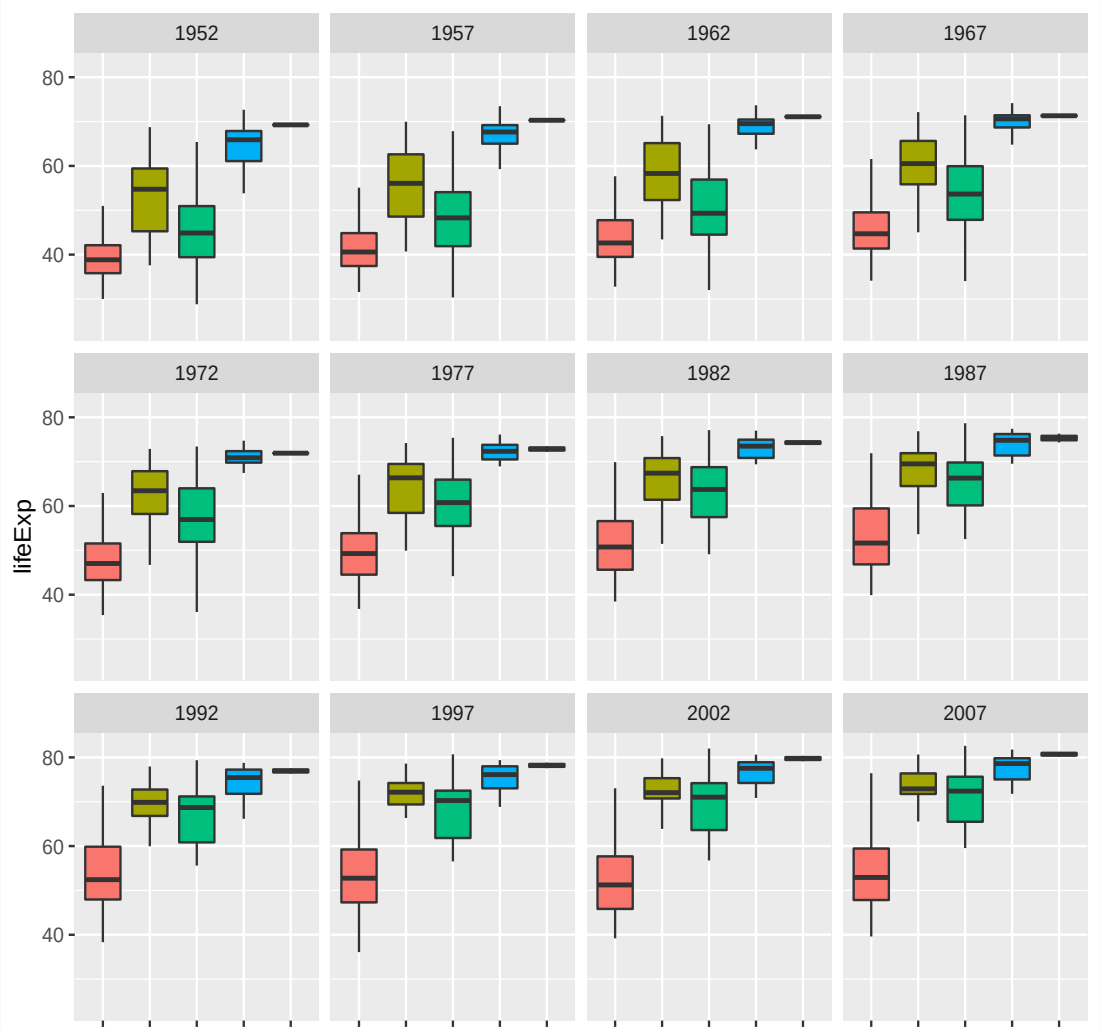
Código básico: boxplot

```
gapminder %>%
    filter(year %in% c(1952,1972,1992, 2002)) %>%
    ggplot(aes(x = continent, y = lifeExp, fill = factor(year))) +
    geom_boxplot(outlier.shape = NA)
```



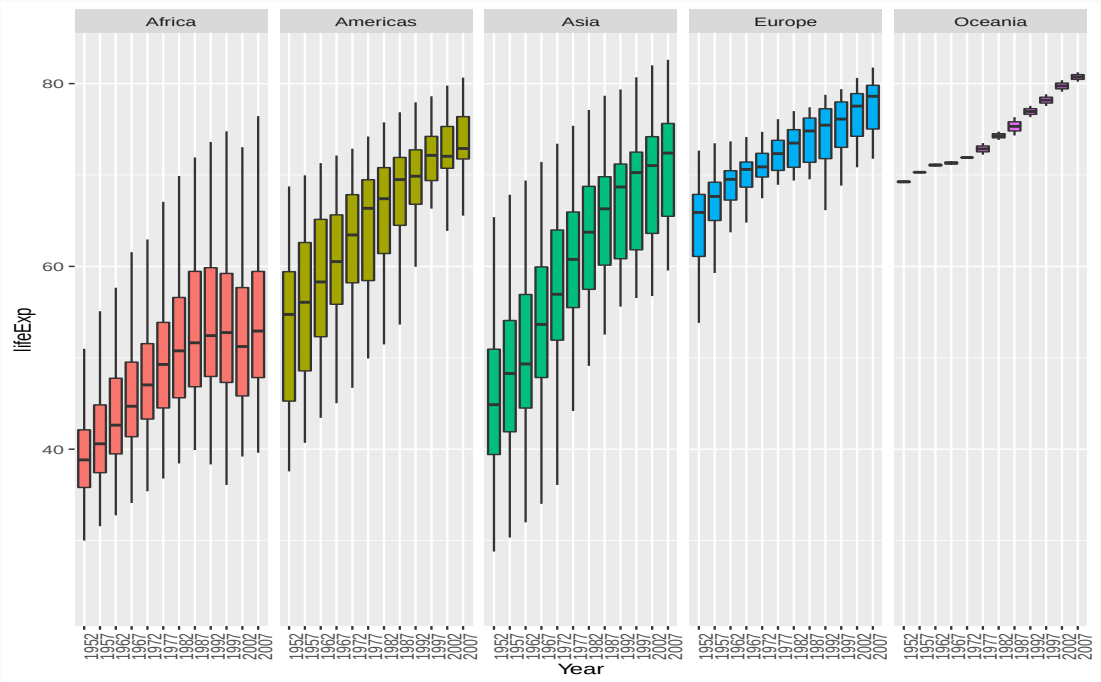
Código básico: boxplot

```
gapminder %>%
    ggplot(aes(x = continent, y = lifeExp, fill = factor(continent))) +
    geom_boxplot(outlier.shape = NA) +
    facet_wrap(~year) +
    theme(legend.position="none") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Código básico: boxplot

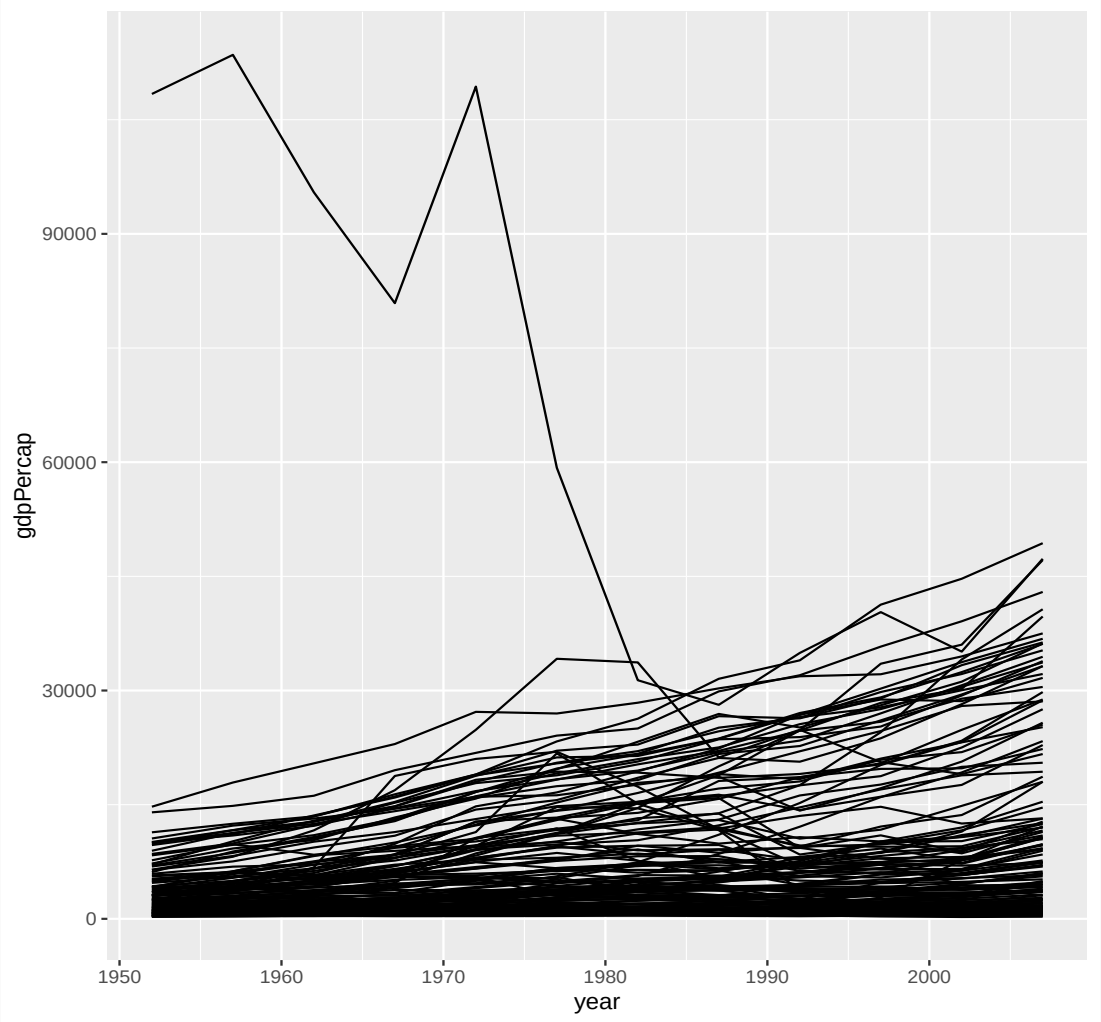
```
gapminder%>%
    ggplot(aes(x=factor(year),y = lifeExp, fill = continent)) +
    geom_boxplot(outlier.shape = NA) +
    xlab("Year")+
    facet_wrap(~continent, ncol = 5) +
    theme(legend.position="none") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Lineas

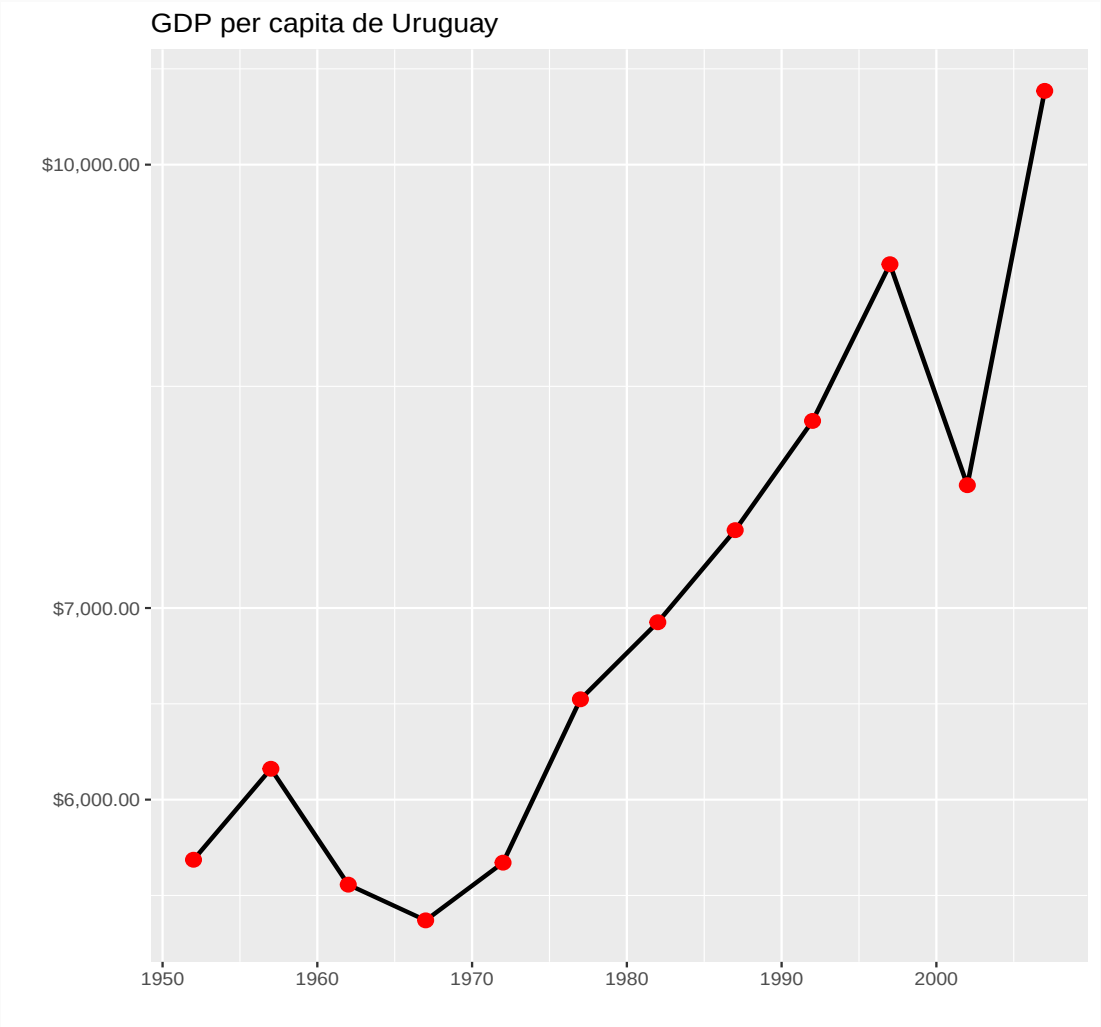
Código básico: line

```
library(gapminder)
ggplot(gapminder, aes(x = year, y = gdpPercap)) +
    geom_line(aes(group = country))
```



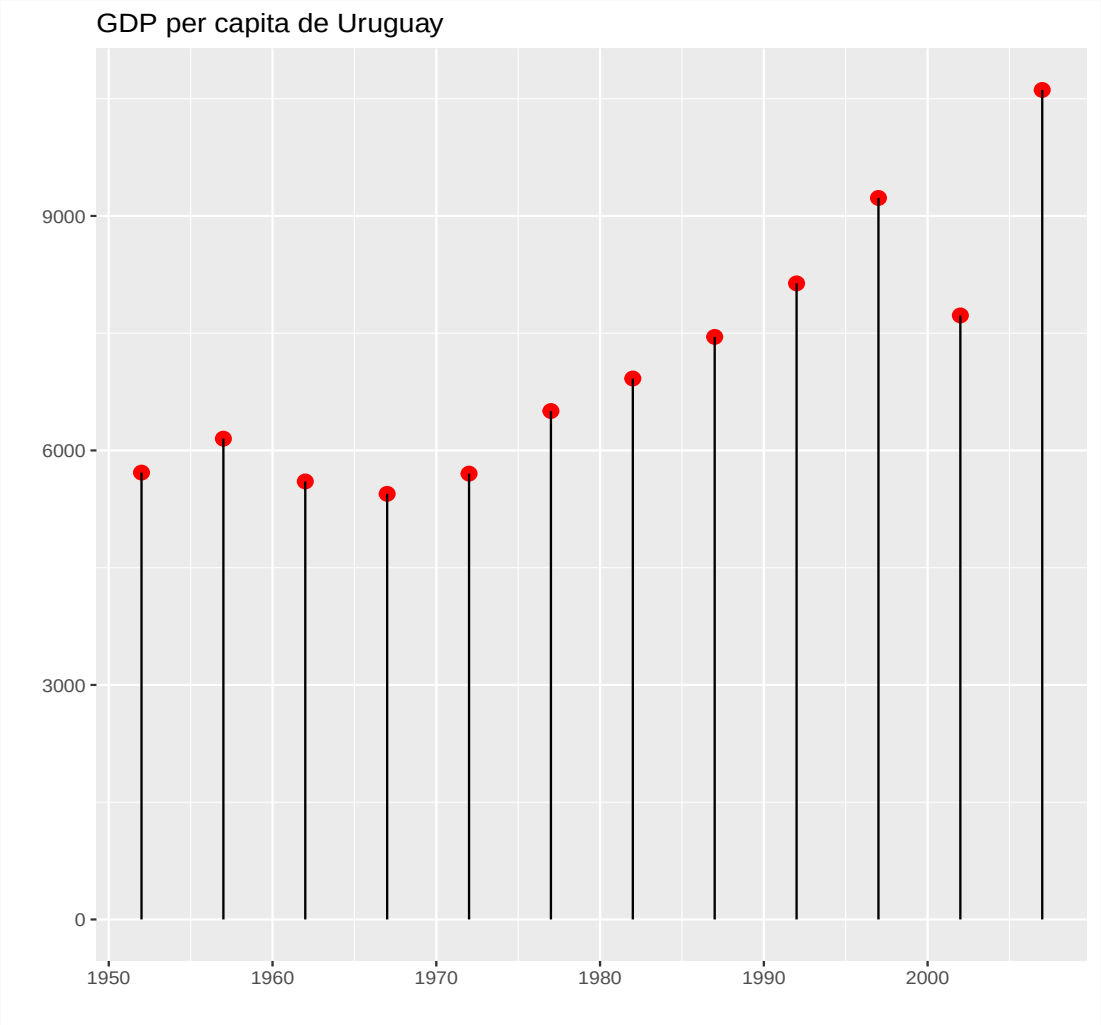
Código básico: line

```
gapminder %>% dplyr::filter(country == 'Uruguay') %>%
ggplot(aes(x = year, y = gdpPercap)) +
    geom_line(size = 1) +
    geom_point(color = 'red', size = 3) +
    scale_y_log10(labels = scales::dollar) +
    labs(x = "", y = "", title = "GDP per capita de Uruguay")
```



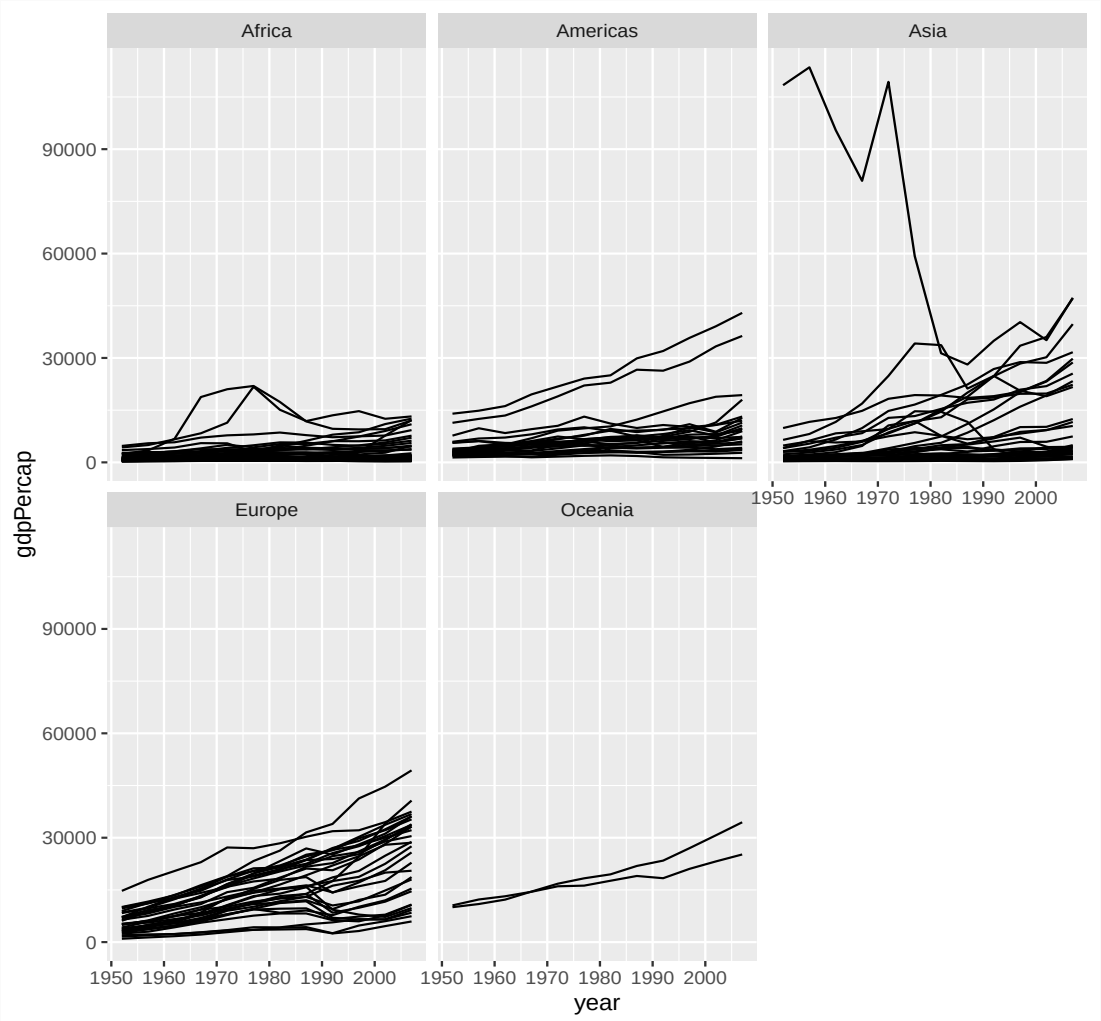
Código básico: segment

```
gapminder %>% dplyr::filter(country == 'Uruguay') %>%
ggplot(aes(x = year, y = gdpPercap)) +
    geom_point(color = 'red', size = 3) +
    geom_segment(aes(x = year, xend = year, y = 0, yend = gdpPercap)) +
    labs(x = "", y = "", title = "GDP per capita de Uruguay")
```



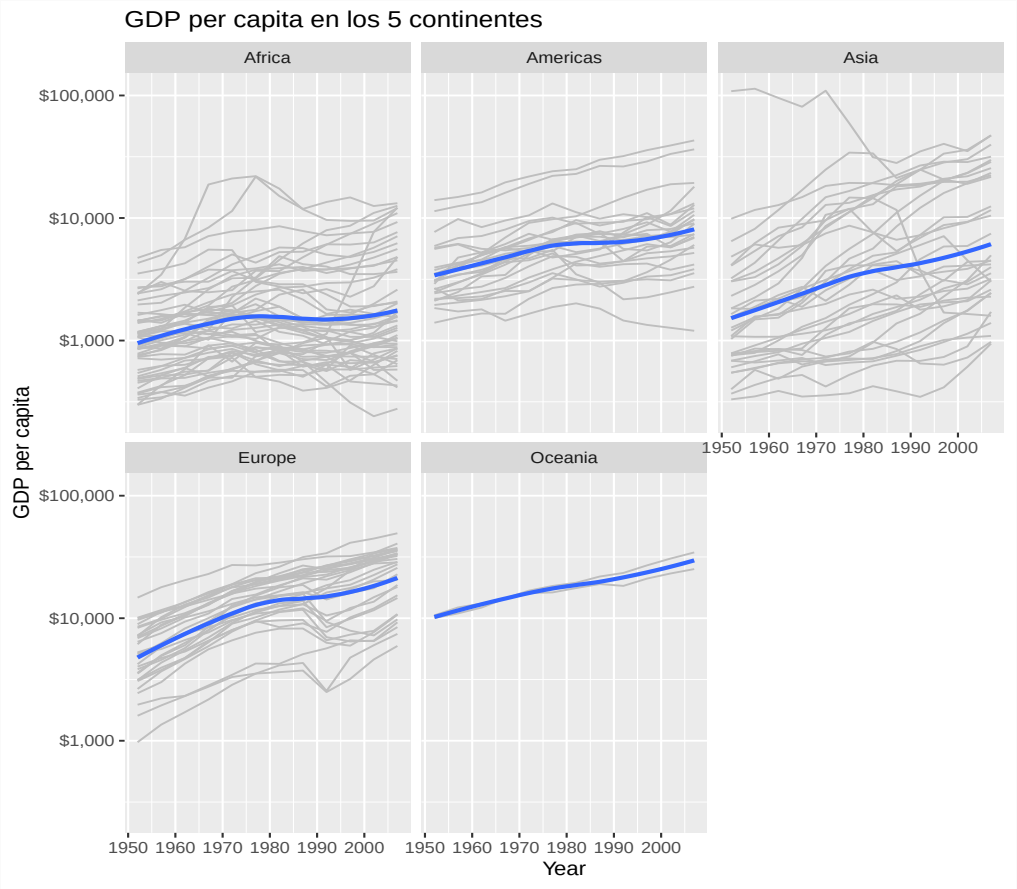
Código básico: line

```
ggplot(gapminder, aes(x = year, y = gdpPercap)) +
    geom_line(aes(group = country)) + facet_wrap(~ continent)
```



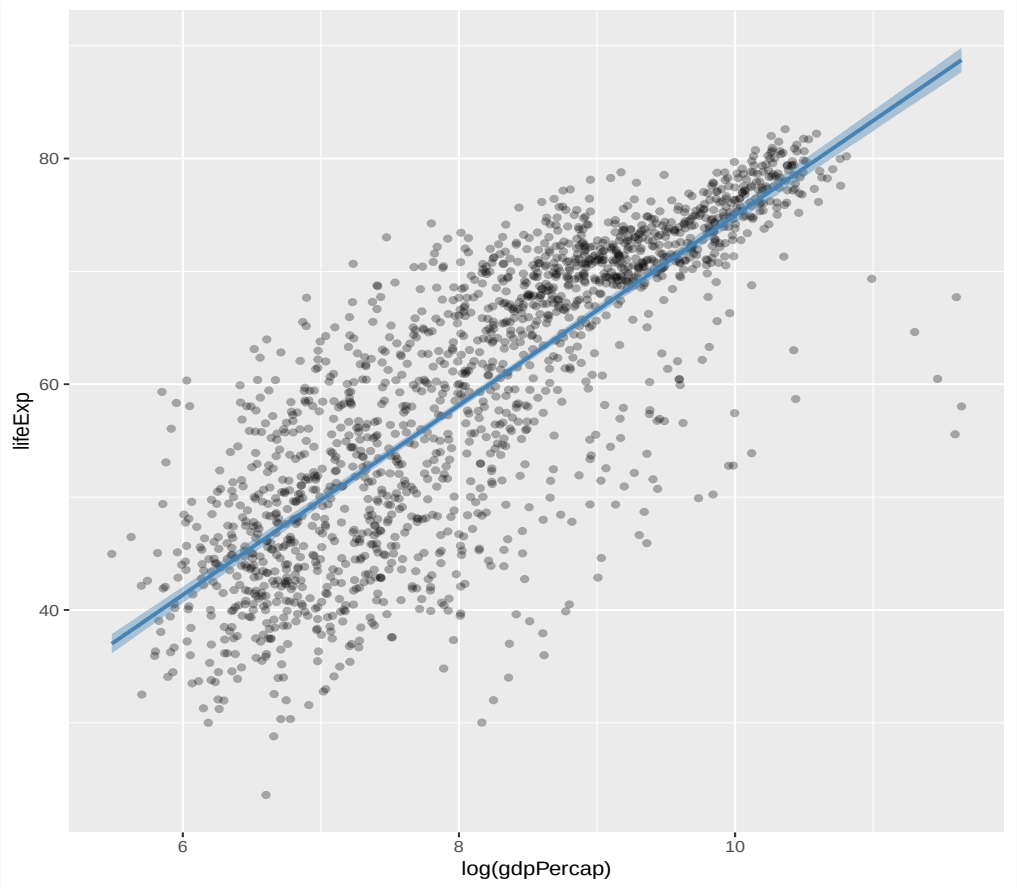
Código básico: line

```
ggplot(gapminder, aes(x = year, y = gdpPercap)) +
    geom_line(color="gray", aes(group = country)) +
    geom_smooth(size = 1.1, method = "loess", se = FALSE) +
    scale_y_log10(labels = scales::dollar) +
    facet_wrap(~ continent) +
    labs(x = "Year", y = "GDP per capita",
         title = "GDP per capita en los 5 continentes")
```



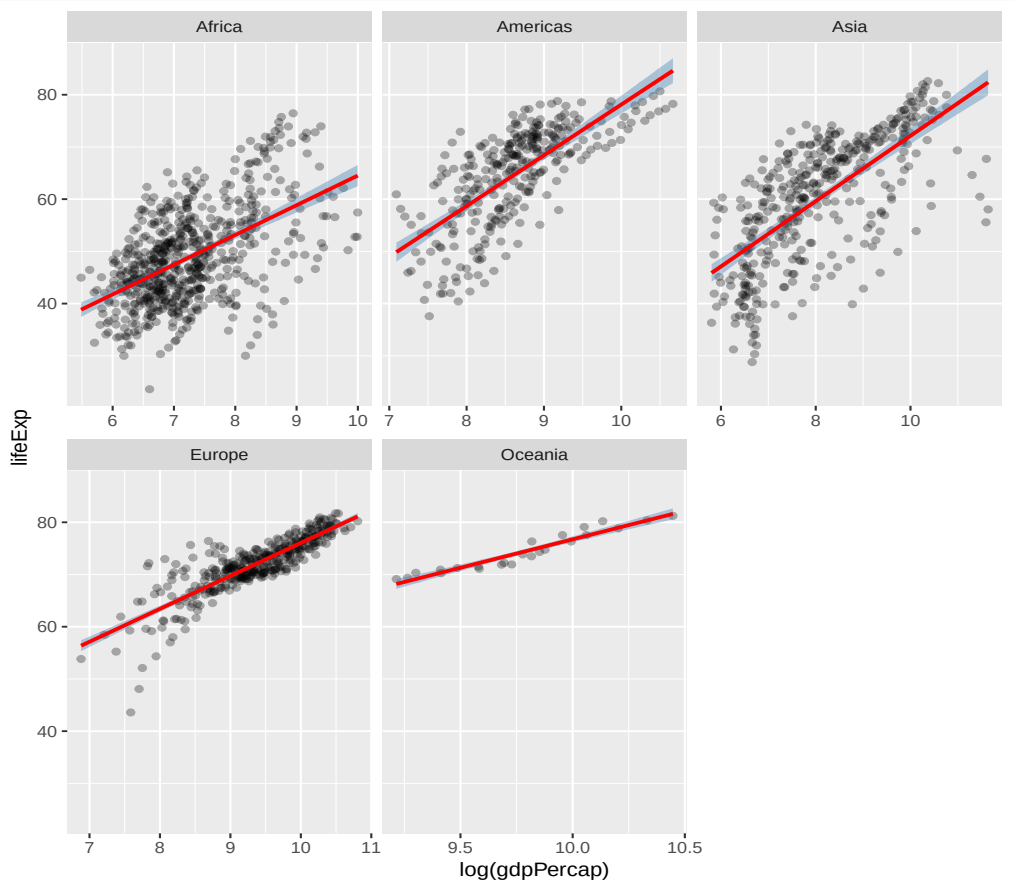
Código básico: smooth

```
ggplot(gapminder, mapping = aes(x = log(gdpPercap), y = lifeExp)) +
  geom_point(alpha=0.3) +
  geom_smooth(color = "steelblue", fill = "steelblue", method = "lm")
```



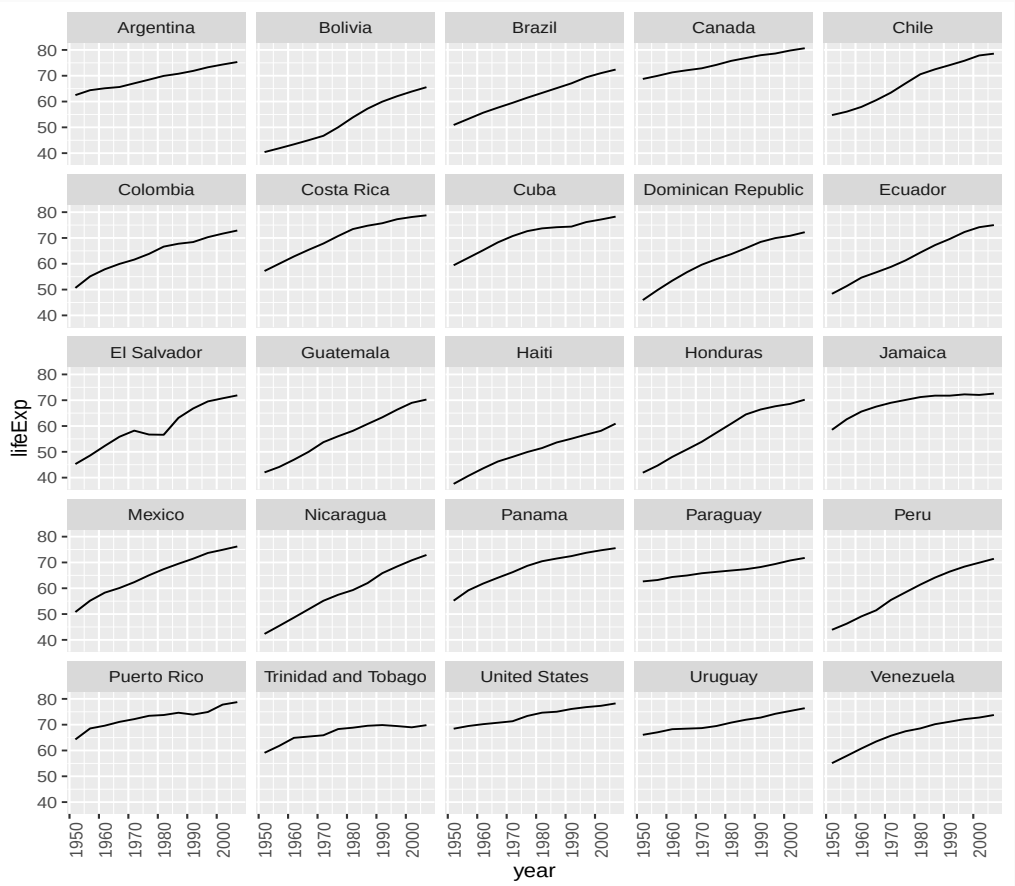
Código básico: smooth

```
ggplot(gapminder, mapping = aes(x = log(gdpPercap), y = lifeExp)) +
  geom_point(alpha=0.3) +
  geom_smooth(color = "red", fill = "steelblue", method = "lm") +
  facet_wrap(~continent, scale = 'free_x')
```



Código básico: smooth

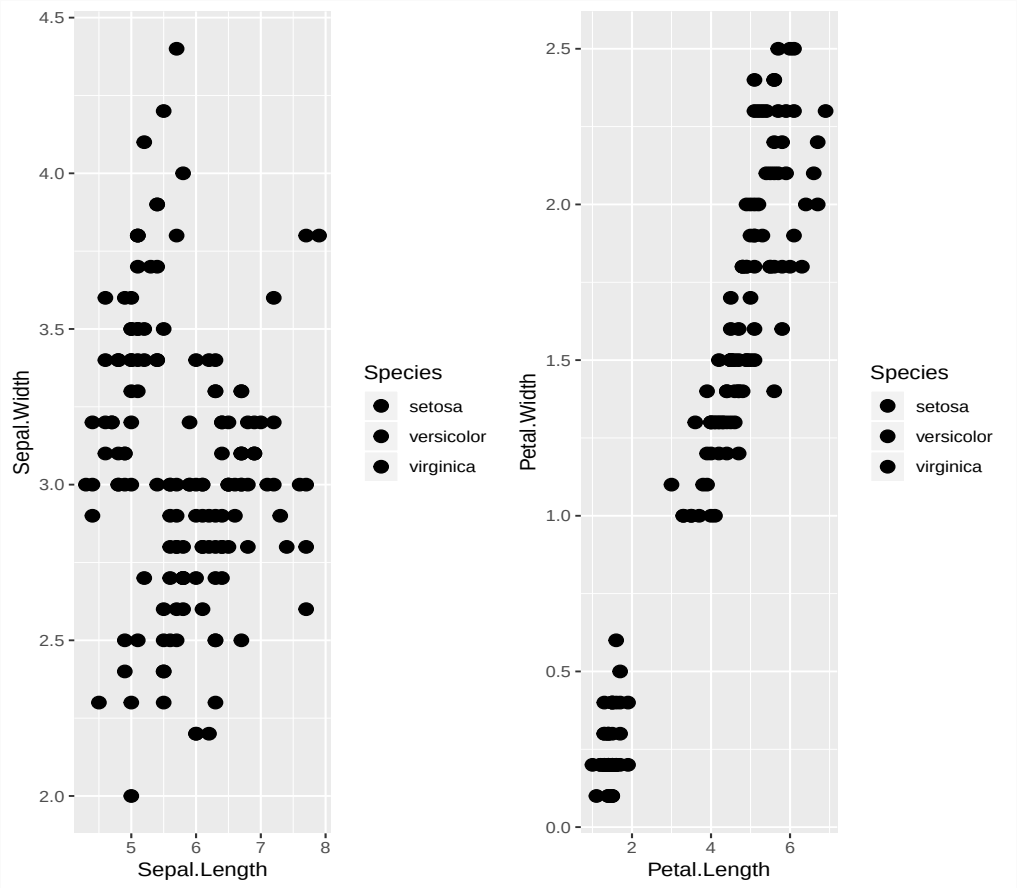
```
gapminder %>%  filter(continent == 'Americas') %>%
  ggplot(aes(x = year, y = lifeExp)) +
     geom_line() +
     facet_wrap(~country) +
     theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Combinar gráficos

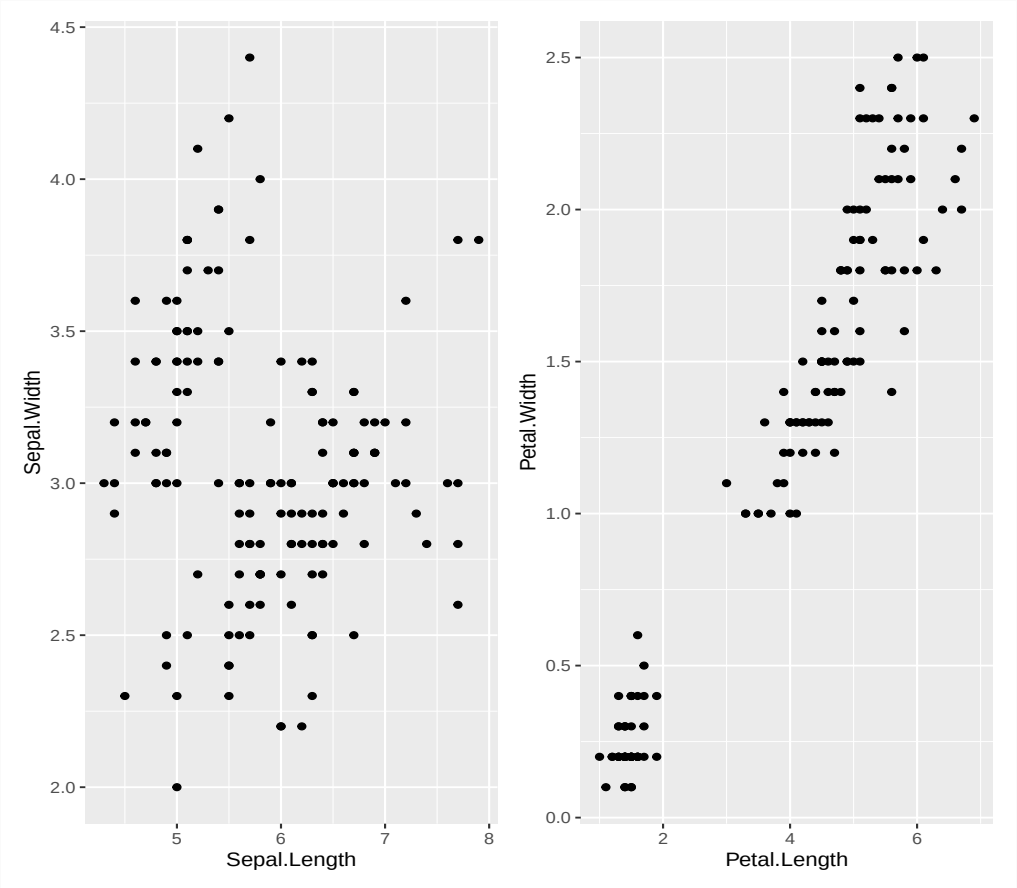
gridExtra

```
library(gridExtra)
g1 <- ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, fill = Species)) + geom_point(size = 3)
g2 <- ggplot(iris, aes(x = Petal.Length, y = Petal.Width, fill = Species)) + geom_point(size = 3)
grid.arrange(g1, g2, ncol = 2)
```



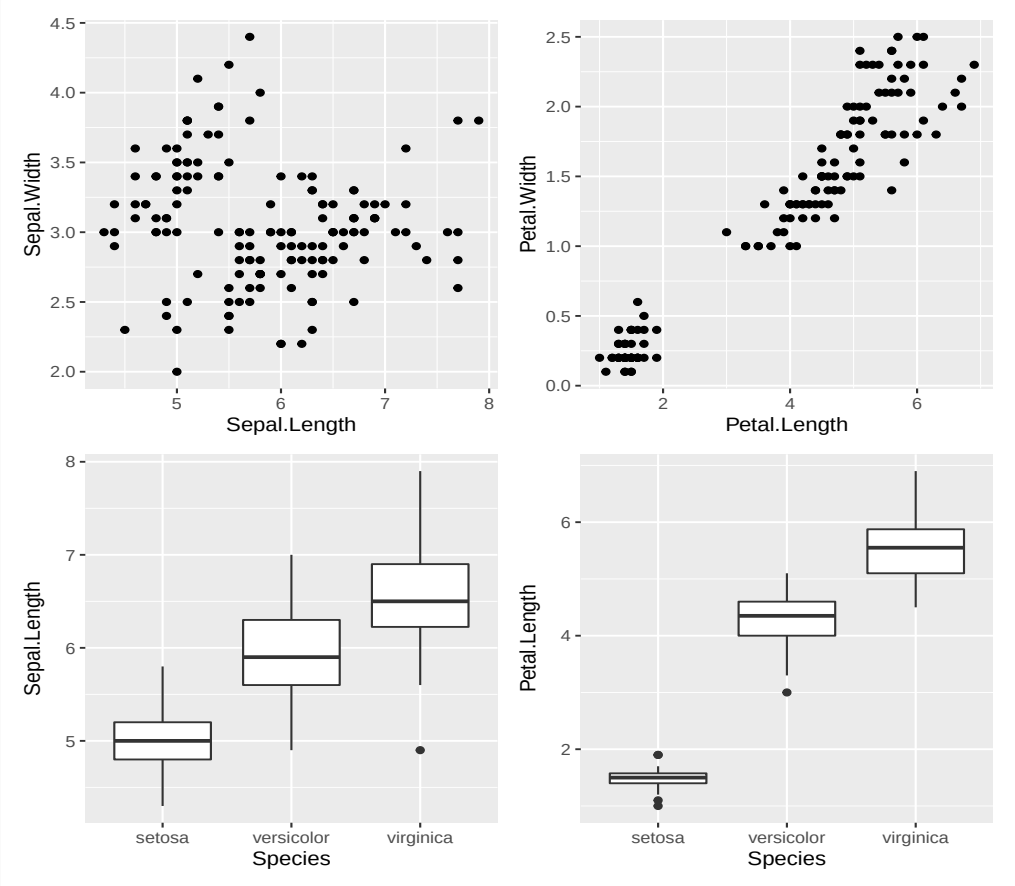
Código básico: patchwork

```
library(patchwork)
graf1 <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) + geom_point()
graf2 <- ggplot(iris, aes(Petal.Length, Petal.Width)) + geom_point()
graf1 + graf2
```



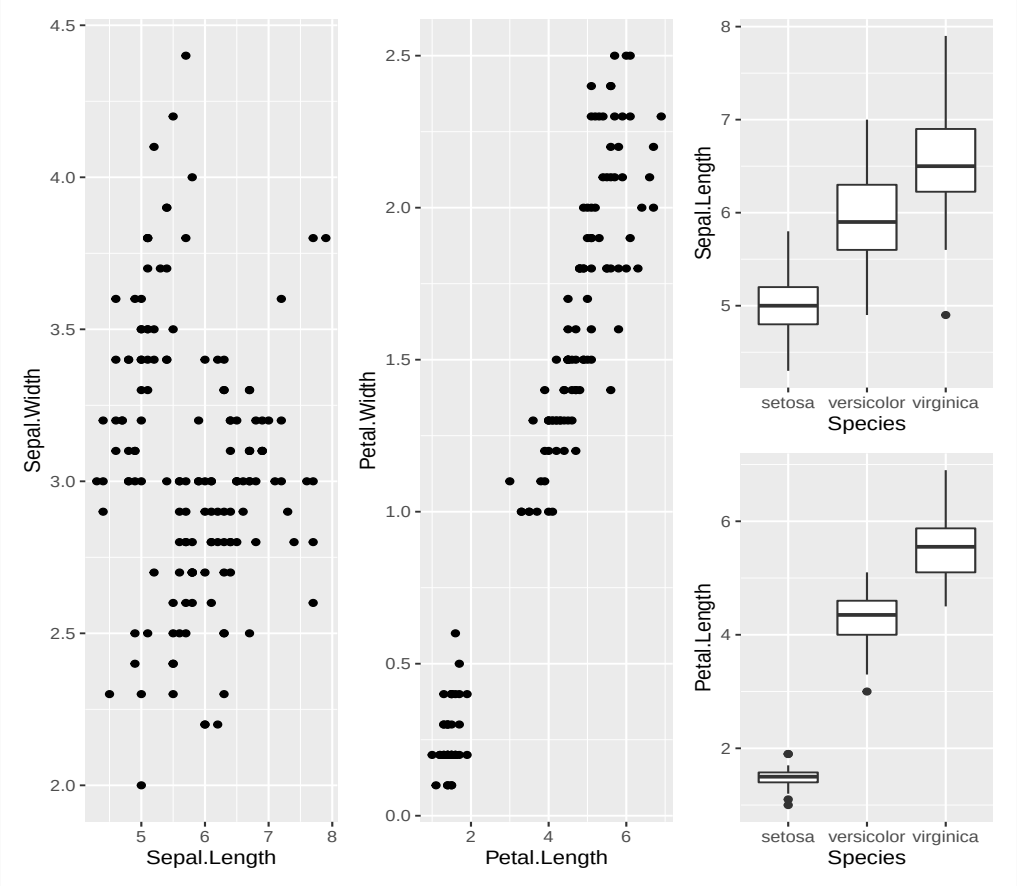
Código básico: patchwork

```
graf3 <- ggplot(iris, aes(Species, Sepal.Length)) + geom_boxplot()
graf4 <- ggplot(iris, aes(Species, Petal.Length)) + geom_boxplot()
graf1 + graf2 + graf3 + graf4
```



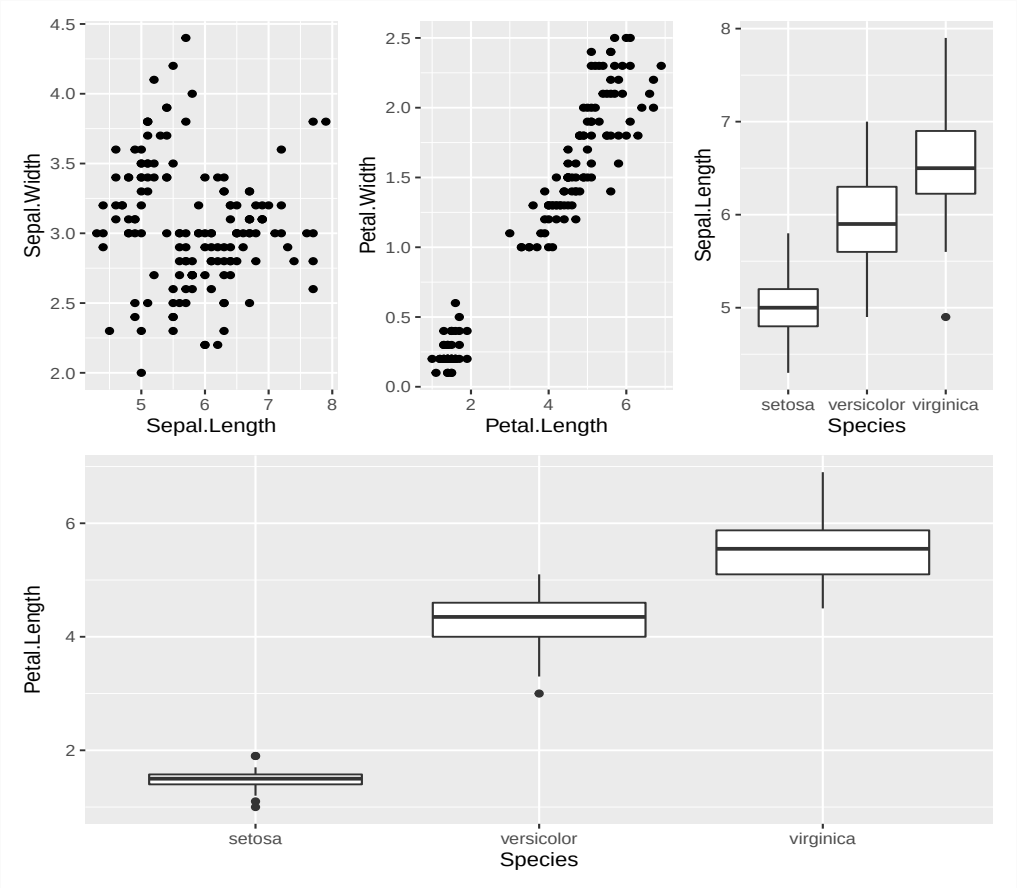
Código básico: patchwork

```
graf1 + graf2 + graf3 / graf4
```



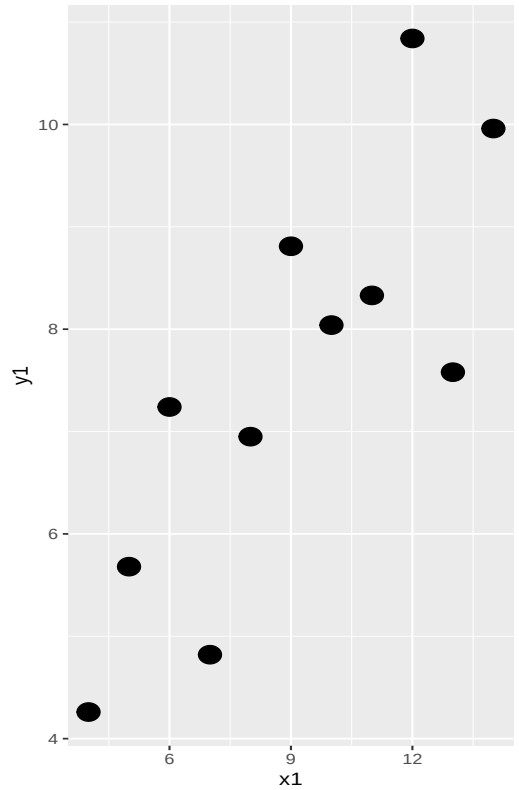
Código básico: patchwork

```
(graf1 + graf2 + graf3) / graf4
```



Código básico: patchwork

```
graf5 <- ggplot(anscombe) + geom_point(aes(x = x1, y = y1), size = 5)
graf5 + gridExtra::tableGrob(anscombe[, c('x1', 'y1')])
```



	x1	y1
1	10	8.04
2	8	6.95
3	13	7.58
4	9	8.81
5	11	8.33
6	14	9.96
7	6	7.24
8	4	4.26
9	12	10.84
10	7	4.82
11	5	5.68

Código básico: patchwork

```
graf5 + grid::textGrob('Primero del cuarteto de Anscombe!')
```