

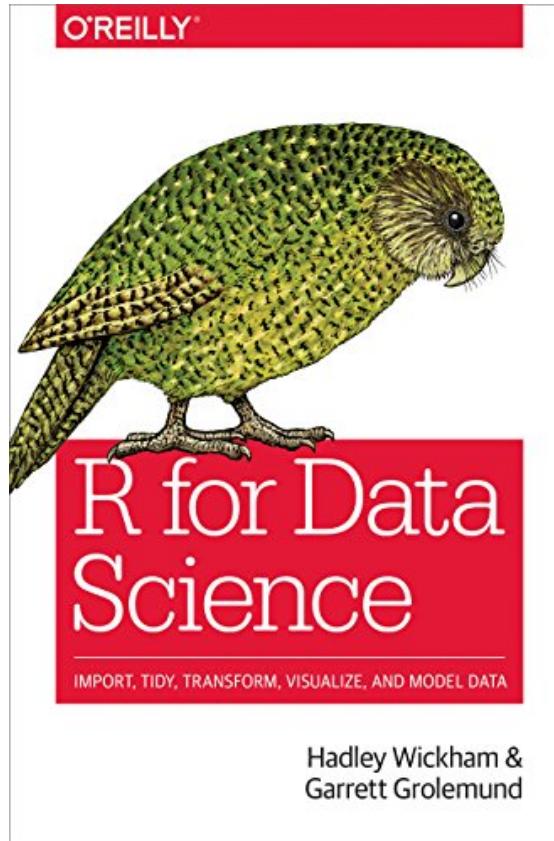
# Conhecendo o Tidyverse

*tidyverse*

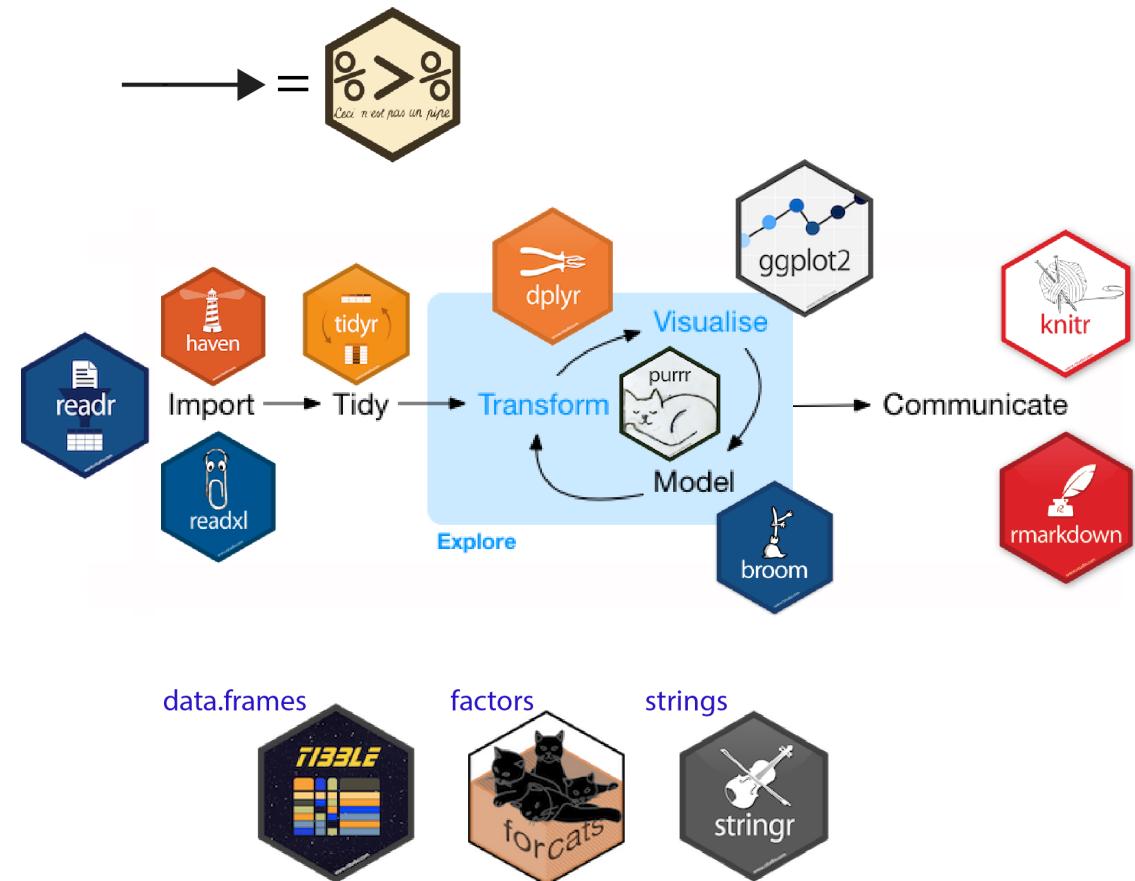
Fernando de Pol Mayer (LEG/DEST/UFPR)  
2022-03-15



# R for Data Science



R for Data Science, a principal referência sobre o emprego da linguagem R em ciência de dados.



Workflow de ciência de dados com o {tidyverse}. Fonte: [https://oliviergimenez.github.io/intro\\_tidyverse/#7](https://oliviergimenez.github.io/intro_tidyverse/#7)

{tidyr}

# Um overview do {tidyr}

- ▶ A maior parte das etapas de análise de dados assume que os dados estão arrumados:
  - ▶ Cada **coluna** é uma variável/atributo/campo.
  - ▶ Cada **linha** é uma observação/caso/instância/tupla.
  - ▶ Cada **cédula** é o registro de uma variável de uma observação.
- ▶ Situações que fogem a regra:
  - ▶ Disposição no **formato** longo ou amplo.
  - ▶ Colunas com valores **concatenados/separados**.
  - ▶ Registros com valores **ausentes**.
- ▶ O **tidyr** contém recursos para arrumação dos dados.
  - ▶ Mudança de disposição dos dados.
  - ▶ Substituição de missings.
  - ▶ Separação e união de campos.
- ▶ Documentação:
  - ▶ <https://tidyr.tidyverse.org/>.
  - ▶ <https://r4ds.had.co.nz/tidy-data.html>.
  - ▶ <https://cran.r-project.org/package=tidyr>

# Funções do pacote

```
library(tidyverse)
ls("package:tidyR")
```

# [1] "%>%"	"all_of"	"any_of"	"as_tibble"
# [5] "billboard"	"build_longer_spec"	"build_wider_spec"	"check_pivot_spec"
# [9] "chop"	"complete"	"complete_"	"construction"
# [13] "contains"	"crossing"	"crossing_"	"drop_na"
# [17] "drop_na_"	"ends_with"	"everything"	"expand"
# [21] "expand_"	"expand_grid"	"extract"	"extract_"
# [25] "extract_numeric"	"fill"	"fill_"	"fish_encounters"
# [29] "full_seq"	"gather"	"gather_"	"hoist"
# [33] "last_col"	"matches"	"nest"	"nest_"
# [37] "nest_legacy"	"nesting"	"nesting_"	"num_range"
# [41] "one_of"	"pack"	"pivot_longer"	"pivot_longer_spec"
# [45] "pivot_wider"	"pivot_wider_spec"	"population"	"relig_income"
# [49] "replace_na"	"separate"	"separate_"	"separate_rows"
# [53] "separate_rows_"	"smiths"	"spread"	"spread_"
# [57] "starts_with"	"table1"	"table2"	"table3"
# [61] "table4a"	"table4b"	"table5"	"tibble"
# [65] "tidyr_legacy"	"tribble"	"unchop"	"uncount"
# [69] "unite"	"unite_"	"unnest"	"unnest_"
# [73] "unnest_auto"	"unnest_legacy"	"unnest_longer"	"unnest_wider"
# [77] "unpack"	"us_rent_income"	"who"	"world_bank_pop"

# Empilhar variáveis

Situação comum quando:

- ▶ são feitas medidas ao longo do tempo.
- ▶ dados de painel e/ou questionário.

```
n <- 3
tb1 <- tibble(prod = LETTERS[1:n],
               aval1 = rpois(n, 4),
               aval2 = rpois(n, 4),
               aval3 = rpois(n, 4))
tb1
```

```
# # A tibble: 3 × 4
#   prod  aval1  aval2  aval3
#   <chr> <int> <int> <int>
# 1 A        7      4      5
# 2 B        5      5      3
# 3 C        2      1      5
```

```
tb2 <- tb1 %>%
  pivot_longer(names_to = "aval",
               values_to = "nota",
               cols = aval1:aval3)
tb2
```

```
# # A tibble: 9 × 3
#   prod  aval  nota
#   <chr> <chr> <int>
# 1 A     aval1    7
# 2 A     aval2    4
# 3 A     aval3    5
# 4 B     aval1    5
# 5 B     aval2    5
# 6 B     aval3    3
# 7 C     aval1    2
# 8 C     aval2    1
# 9 C     aval3    5
```

## Desempilhar variável

- É a operação inversa de empilhar.
- Dados nessa disposição são menos comuns.

```
tb2 %>%  
  pivot_wider(names_from = "aval",  
             values_from = "nota")  
  
# # A tibble: 3 × 4  
#   prod  aval1  aval2  aval3  
#   <chr> <int> <int> <int>  
# 1 A        7     4     5  
# 2 B        5     5     3  
# 3 C        2     1     5
```

# Separar variável

- Muito comum quando um campo de texto é a união de várias informações.
- Ex: datas, horas, endereços, etc.

```
tb <- tibble(  
  veiculo = c("Celta", "Gol", "Uno"),  
  ano_mod = c("2011/2012", "2012/2012", "2015/2016"),  
  local = c("Curitiba-PR", "Santos-SP", "Viçosa-MG"))  
tb  
  
# # A tibble: 3 × 3  
#   veiculo  ano_mod    local  
#   <chr>     <chr>      <chr>  
# 1 Celta    2011/2012 Curitiba-PR  
# 2 Gol      2012/2012 Santos-SP  
# 3 Uno      2015/2016 Viçosa-MG
```

```
## Tudo junto  
tb %>%  
  separate(col = "ano_mod",  
           into = c("ano", "modelo"),  
           sep = "/",  
           convert = TRUE) %>%  
  separate(col = "local",  
           into = c("cidade", "estado"),  
           sep = "-")  
  
# # A tibble: 3 × 5  
#   veiculo  ano modelo cidade estado  
#   <chr>     <int> <int> <chr>   <chr>  
# 1 Celta      2011    2012 Curitiba PR  
# 2 Gol        2012    2012 Santos    SP  
# 3 Uno        2015    2016 Viçosa    MG
```

## Unir variáveis

- Quando vários campos precisam ser combinados para gerar uma informação.
- Ex: datas, horas, endereços, nomes.

```
tb <- tibble(dia = c(1, 5, 23, 16),
             mes = c(3, 6, 2, 9),
             ano = 2018)

## Cria data e converte para a classe date
tb %>%
  unite(col = "data", ano, mes, dia, sep = "-", remove = FALSE) %>%
  mutate(data = parse_date(data, format = "%Y-%m-%d"))

# # A tibble: 4 × 4
#   data      dia   mes   ano
#   <date>     <dbl> <dbl> <dbl>
# 1 2018-03-01     1     3  2018
# 2 2018-06-05     5     6  2018
# 3 2018-02-23    23     2  2018
# 4 2018-09-16    16     9  2018
```

# Manuseio de valores ausentes

```
tb <- tibble(jogador = 1:5,
             jogos = c(0, 1, 3, 1, 2),
             gols = c(NA, 0, 0, 2, 1),
             faltas = c(NA, 1, 1, 0, 0))
tb
```

```
# # A tibble: 5 × 4
#   jogador jogos  gols faltas
#   <int>    <dbl> <dbl> <dbl>
# 1       1      0     NA     NA
# 2       2      1      0      1
# 3       3      3      0      1
# 4       4      1      2      0
# 5       5      2      1      0
```

```
## Remove todas as linhas com NA
tb %>%
  drop_na()
```

```
# # A tibble: 4 × 4
#   jogador jogos  gols faltas
#   <int>    <dbl> <dbl> <dbl>
# 1       2      1      0      1
# 2       3      3      0      1
# 3       4      1      2      0
# 4       5      2      1      0
```

```
## Substitui os NAs por algum valor
tb %>%
  replace_na(list(gols = 0, faltas = 0))
```

```
# # A tibble: 5 × 4
#   jogador jogos  gols faltas
#   <int>    <dbl> <dbl> <dbl>
# 1       1      0     0     0
# 2       2      1     0     1
# 3       3      3     0     1
# 4       4      1     2     0
# 5       5      2     1     0
```

## Expandir combinações

Expandir combinações entre variáveis é necessário:

- ▶ Para criar o desenho experimental de experimentos fatoriais.
- ▶ Criar o grid para predição da resposta combinando todas as variáveis de entrada.

```
tb <- crossing(blc = as.character(as.roman(1:2)),  
                 trt = LETTERS[1:3])  
tb$prod <- rexp(nrow(tb))  
tb
```

```
# # A tibble: 6 × 3  
#   blc    trt    prod  
#   <chr> <chr> <dbl>  
# 1 I      A     0.248  
# 2 I      B     0.588  
# 3 I      C     1.01  
# 4 II     A     0.880  
# 5 II     B     1.49  
# 6 II     C     3.46
```

# Exercícios para usar o {tidyR}

- ▶ Ninfas em soja.
  1. Ler os dados em  
<http://leg.ufpr.br/~walmes/data/ninfas.txt>.
  2. Empilhar nos terços da planta (superior, médio, inferior)
- ▶ Óleos essenciais.
  1. Ler o dados em  
<http://leg.ufpr.br/~walmes/data/oleos.txt>.
  2. Criar uma variável indicadora do registro:  
`1:nrow(tb)`.
  3. Empilhar nas avaliações, i.e, os 5 campos começados com a.
- ▶ Futebol.
  1. Ler os dados em  
[http://leg.ufpr.br/~walmes/data/euro\\_football\\_players.txt](http://leg.ufpr.br/~walmes/data/euro_football_players.txt).
  2. Substituir os missings em `goal`, `red` e `yel` por 0.
- ▶ Avaliação de veículos.
  1. Ler os dados em  
[http://leg.ufpr.br/~walmes/data/aval\\_carros\\_nota.txt](http://leg.ufpr.br/~walmes/data/aval_carros_nota.txt).
  2. Desempilhar na variável `item` os valores de `nota`.
- ▶ Carros à venda.
  1. Ler os dados em  
[http://leg.ufpr.br/~walmes/data/duster\\_venda\\_260314.txt](http://leg.ufpr.br/~walmes/data/duster_venda_260314.txt).
  2. Separar os campos `ano` e `modelo` na variável `ano`. Ex: `2012/2013`.
  3. Substituir o `NA` em `km` percorrido pela média de `km` percorrido: `mean(..., na.rm = TRUE)`.
- ▶ Condição domiciliar dos municípios.
  1. Ler os dados em  
[http://leg.ufpr.br/~walmes/data/ipea\\_habitacao.csv](http://leg.ufpr.br/~walmes/data/ipea_habitacao.csv).
  2. Concatenar o nome do município com o nome do estado.

{ggplot2}

# Visão geral

O [landscape de recursos para visualização](#) de dados no R pode ser representado por uma divisão em 4 territórios.

1. O pacote `graphics` e derivados.
2. O pacote `lattice` e derivados.
3. O pacote `ggplot2` e derivados.
4. Pacotes para gráficos interativos.

## O pacote `graphics`

- ▶ Contém os recursos mais primitivos: com funções de alto e baixo nível.
- ▶ Vários pacotes que complementam suas funcionalidades: `plotrix` e `gplots`.
- ▶ Usado em métodos gráficos de saídas de análises: dendrogramas, biplots, etc.

## O pacote `lattice`

- ▶ Plotagem multi-painel
- ▶ Já vem com a instalação básica do R.
- ▶ Também é utilizado na implementação de métodos gráficos.

## O pacote `ggplot2`

- ▶ Baseado na [\*Grammar of Graphics\*](#).
- ▶ Plotagem multi-painel
- ▶ A importância da `ggplot2` está no modelo mental mais claro.

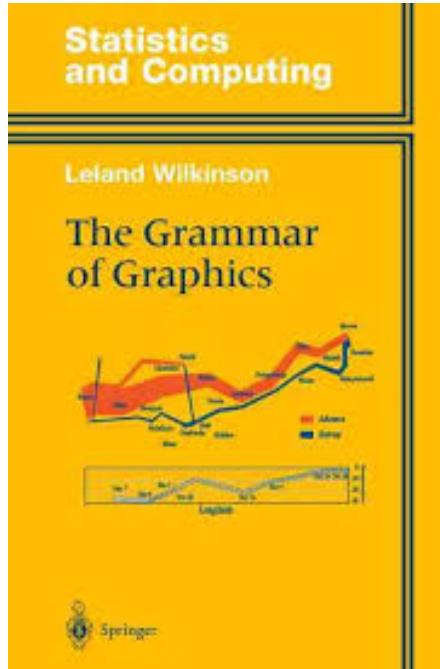
# Um overview do {ggplot2}

*A implementação da gramática dos gráficos*

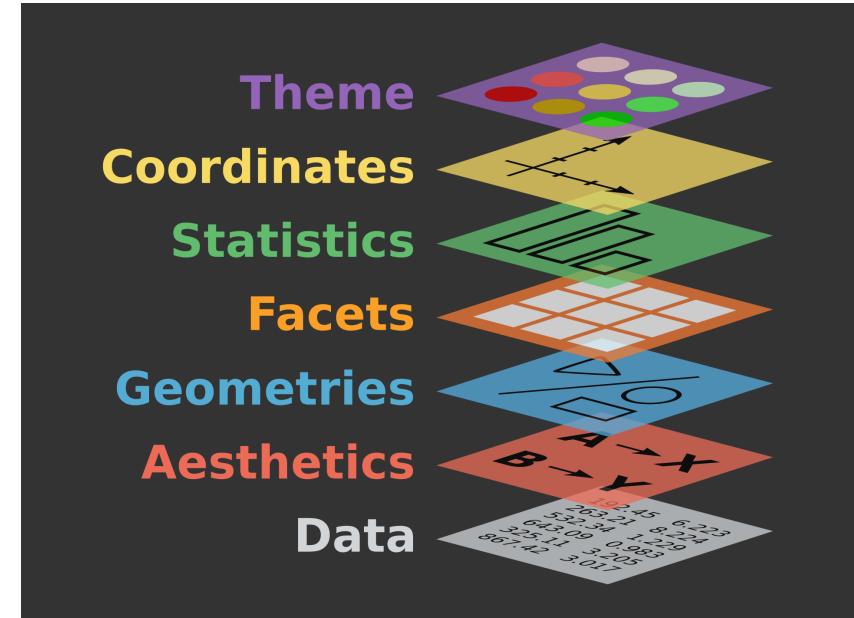
# Gramática dos gráficos



Leland Wilkinson.

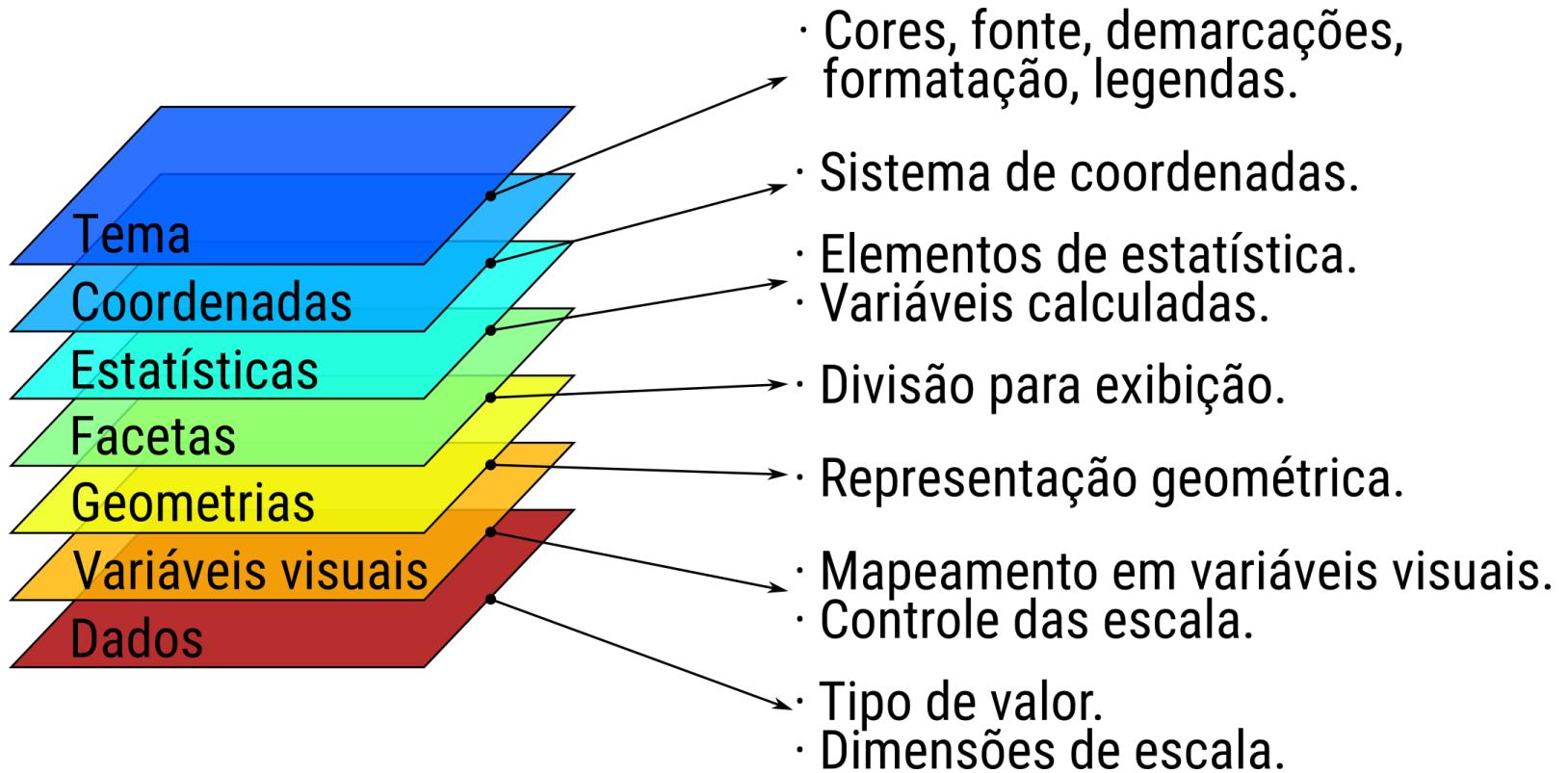


A capa do livro.



As camadas da gramática dos gráficos.

## Gramática dos gráficos

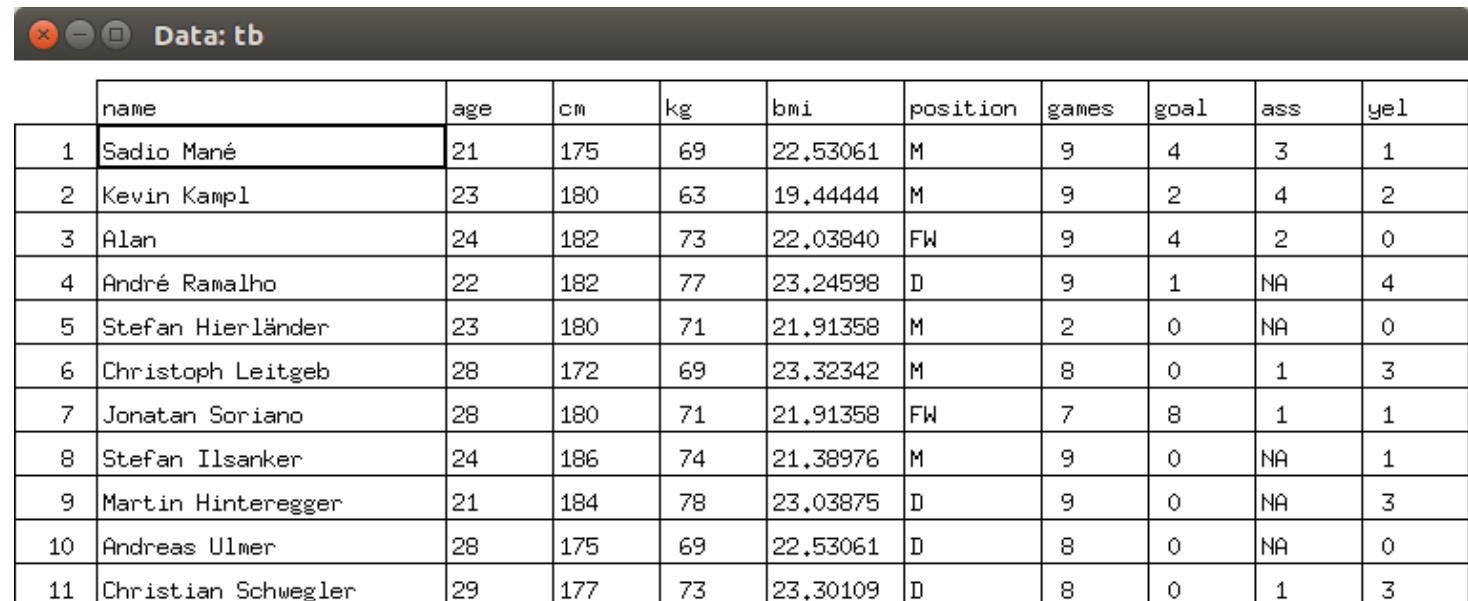


As camadas na gramática dos gráficos.

# Camada 1: dados

Deve se estar atento ao tipo de valor/objeto.

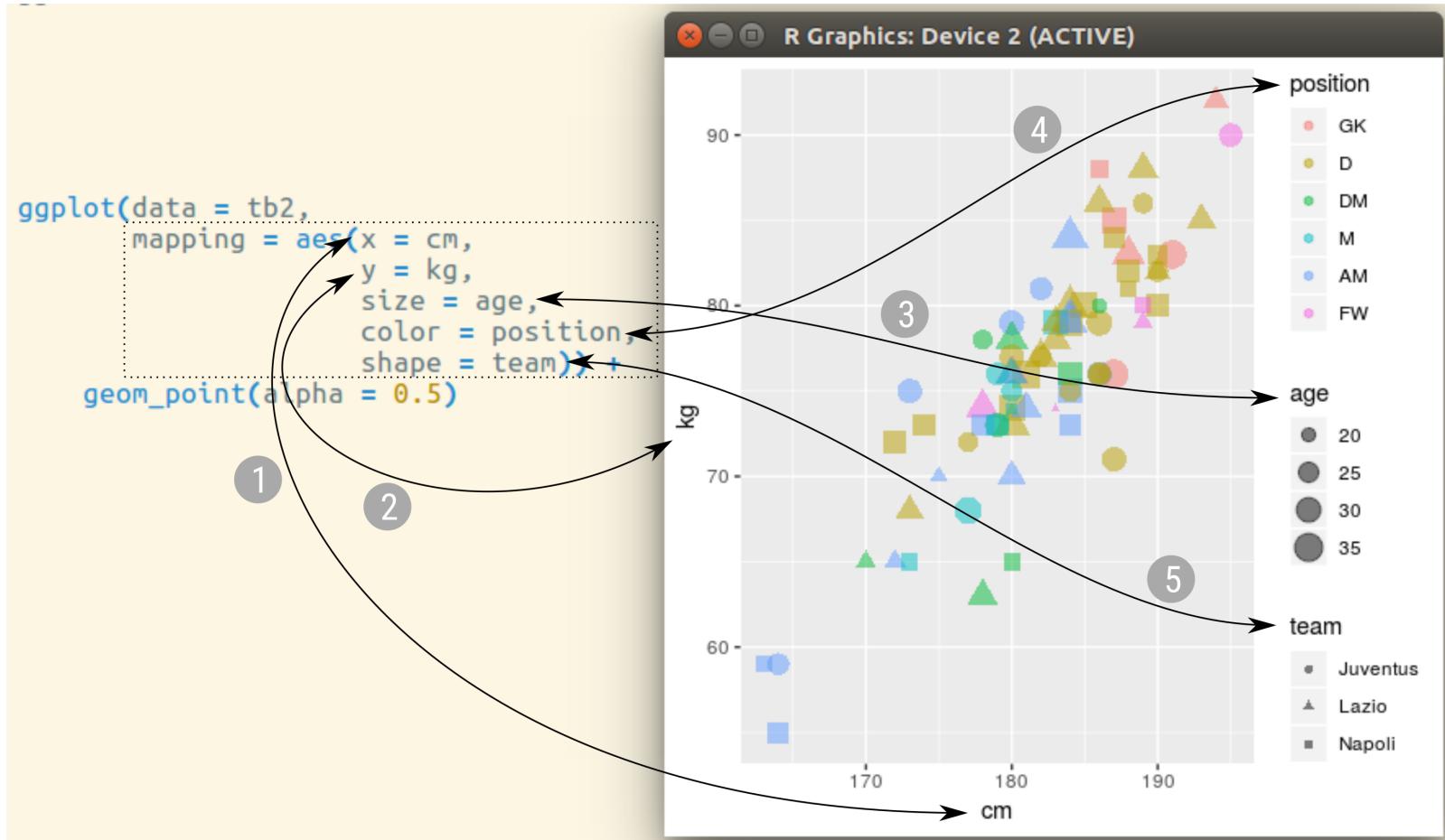
- ▶ **Quantitativa:** variável numérica discreta ou contínua.
- ▶ **Qualitativa:** variável nominal ou ordinal.
- ▶ **Cronológica:** variável de data ou data-tempo.
- ▶ **Geográfica:** objeto como polígonos, por exemplo.



	name	age	cm	kg	bmi	position	games	goal	ass	yel
1	Sadio Mané	21	175	69	22.53061	M	9	4	3	1
2	Kevin Kampl	23	180	63	19.44444	M	9	2	4	2
3	Alan	24	182	73	22.03840	FW	9	4	2	0
4	André Ramalho	22	182	77	23.24598	D	9	1	NA	4
5	Stefan Hierländer	23	180	71	21.91358	M	2	0	NA	0
6	Christoph Leitgeb	28	172	69	23.32342	M	8	0	1	3
7	Jonatan Soriano	28	180	71	21.91358	FW	7	8	1	1
8	Stefan Ilsanker	24	186	74	21.38976	M	9	0	NA	1
9	Martin Hinteregger	21	184	78	23.03875	D	9	0	NA	3
10	Andreas Ulmer	28	175	69	22.53061	D	8	0	NA	0
11	Christian Schwegler	29	177	73	23.30109	D	8	0	1	3

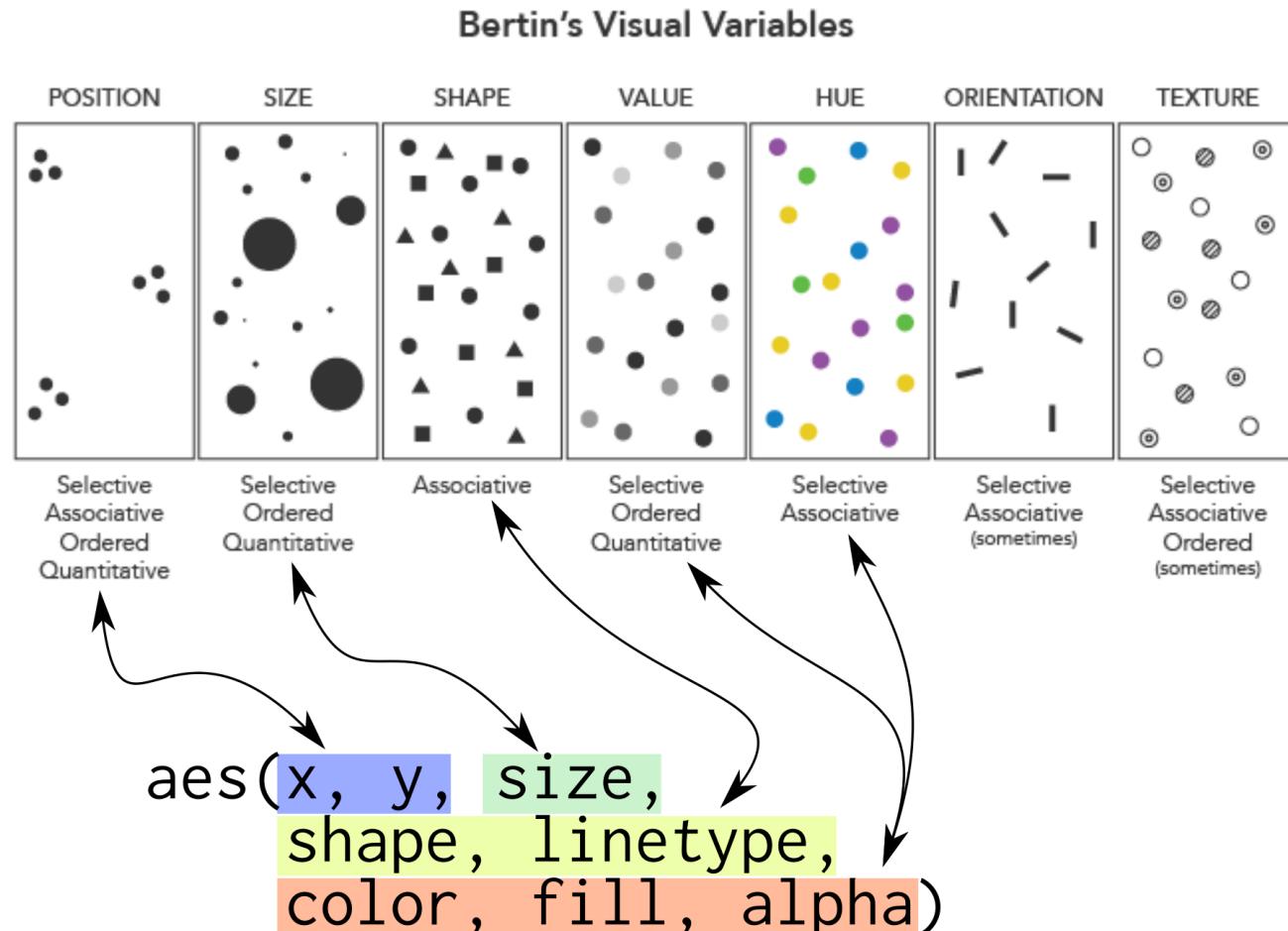
A camada dos dados.

## Camada 2: mapeamento em variáveis visuais



A camada de mapeamento dos valores em variáveis visuais.

## Camada 2: mapeamento em variáveis visuais



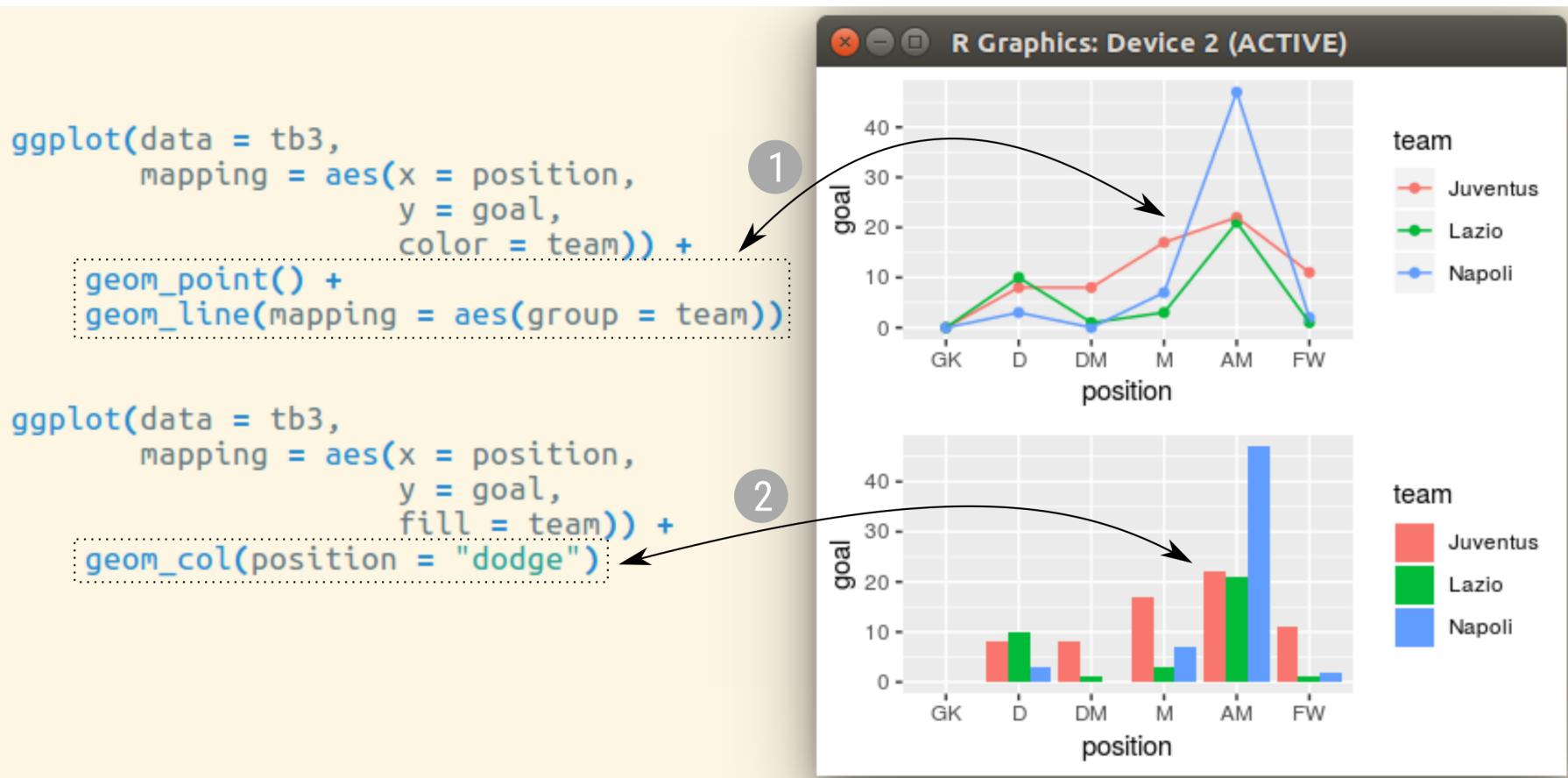
Variáveis visuais disponíveis no {ggplot2}.

## Camada 2: mapeamento em variáveis visuais

Funções para controle de escala:

```
scale_alpha_:
  binned, continuous, date, datetime, discrete, identity, manual, ordinal
scale_colour_:
  binned, brewer, continuous, date, datetime, discrete, distiller, fermenter, gradient,
  gradient2, gradientn, grey, hue, identity, manual, ordinal, steps, steps2, stepsn
scale_continuous_:
  identity
scale_discrete_:
  identity, manual
scale_fill_:
  binned, brewer, continuous, date, datetime, discrete, distiller, fermenter, gradient,
  gradient2, gradientn, grey, hue, identity, manual, ordinal, steps, steps2, stepsn
scale_linetype_:
  binned, continuous, discrete, identity, manual
scale_shape_:
  binned, continuous, discrete, identity, manual, ordinal
scale_size_:
  area, binned, continuous, date, datetime, discrete, identity, manual, ordinal
scale_x_:
  binned, continuous, date, datetime, discrete, log10, reverse, sqrt, time
scale_y_:
  binned, continuous, date, datetime, discrete, log10, reverse, sqrt, time
```

## Camada 3: geometrias



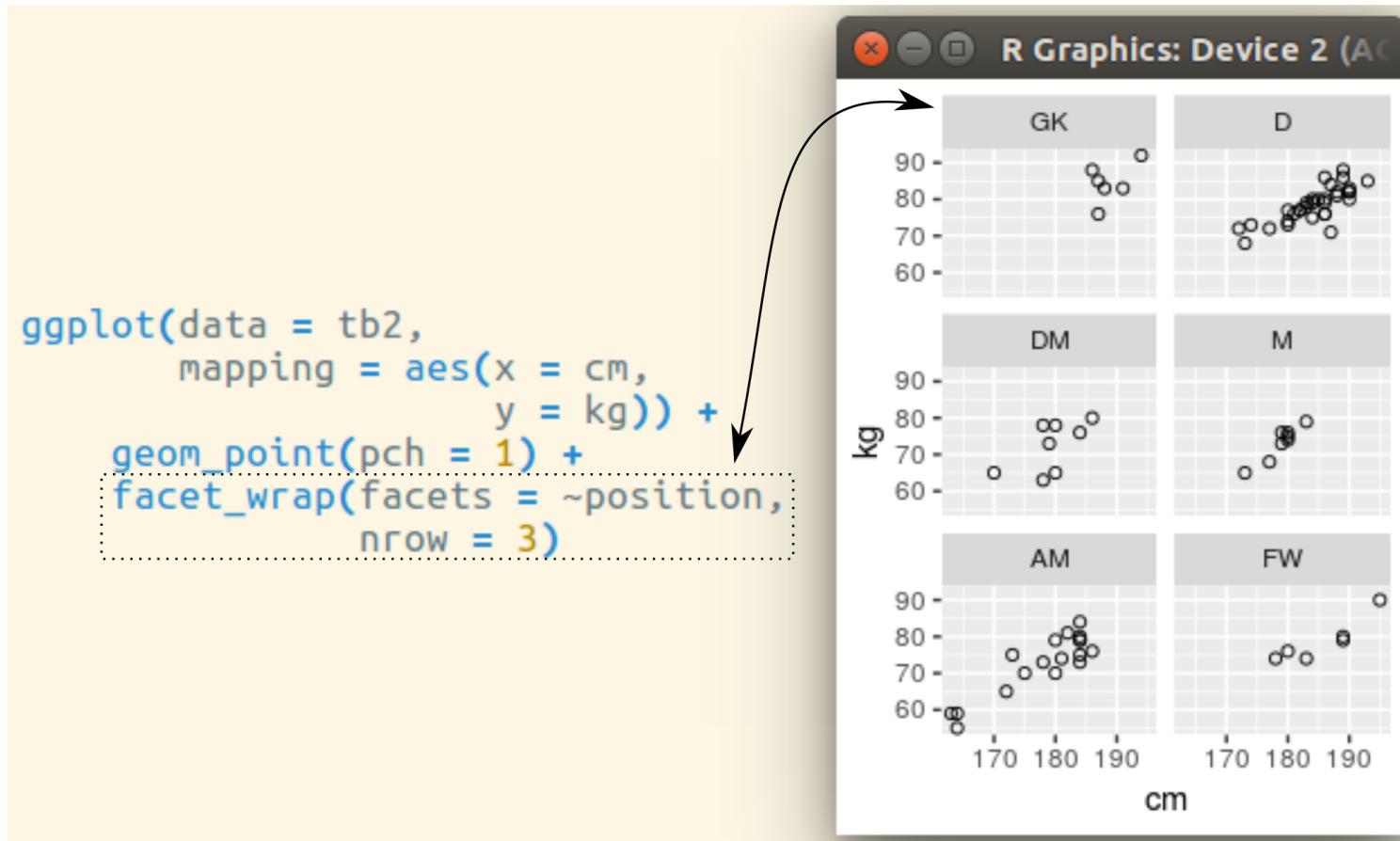
A camada de elementos geométricos.

## Camada 3: geometrias

Funções disponíveis para a camada de geometria:

geom_abline	geom_dotplot	geom_qq_line
geom_area	geom_errorbar	geom_quantile
geom_bar	geom_errorbarh	geom_raster
geom_bin_2d	geom_freqpoly	geom_rect
geom_bin2d	geom_function	geom_ribbon
geom_blank	geom_hex	geom_rug
geom_boxplot	geom_histogram	geom_segment
geom_col	geom_hline	geom_sf
geom_contour	geom_jitter	geom_sf_label
geom_contour_filled	geom_label	geom_sf_text
geom_count	geom_line	geom_smooth
geom_crossbar	geom_linerange	geom_spoke
geom_curve	geom_map	geom_step
geom_density	geom_path	geom_text
geom_density_2d	geom_point	geom_tile
geom_density_2d_filled	geom_pointrange	geom_violin
geom_density2d	geom_polygon	geom_vline
geom_density2d_filled	geom_qq	

## Camada 4: divisão em facetas



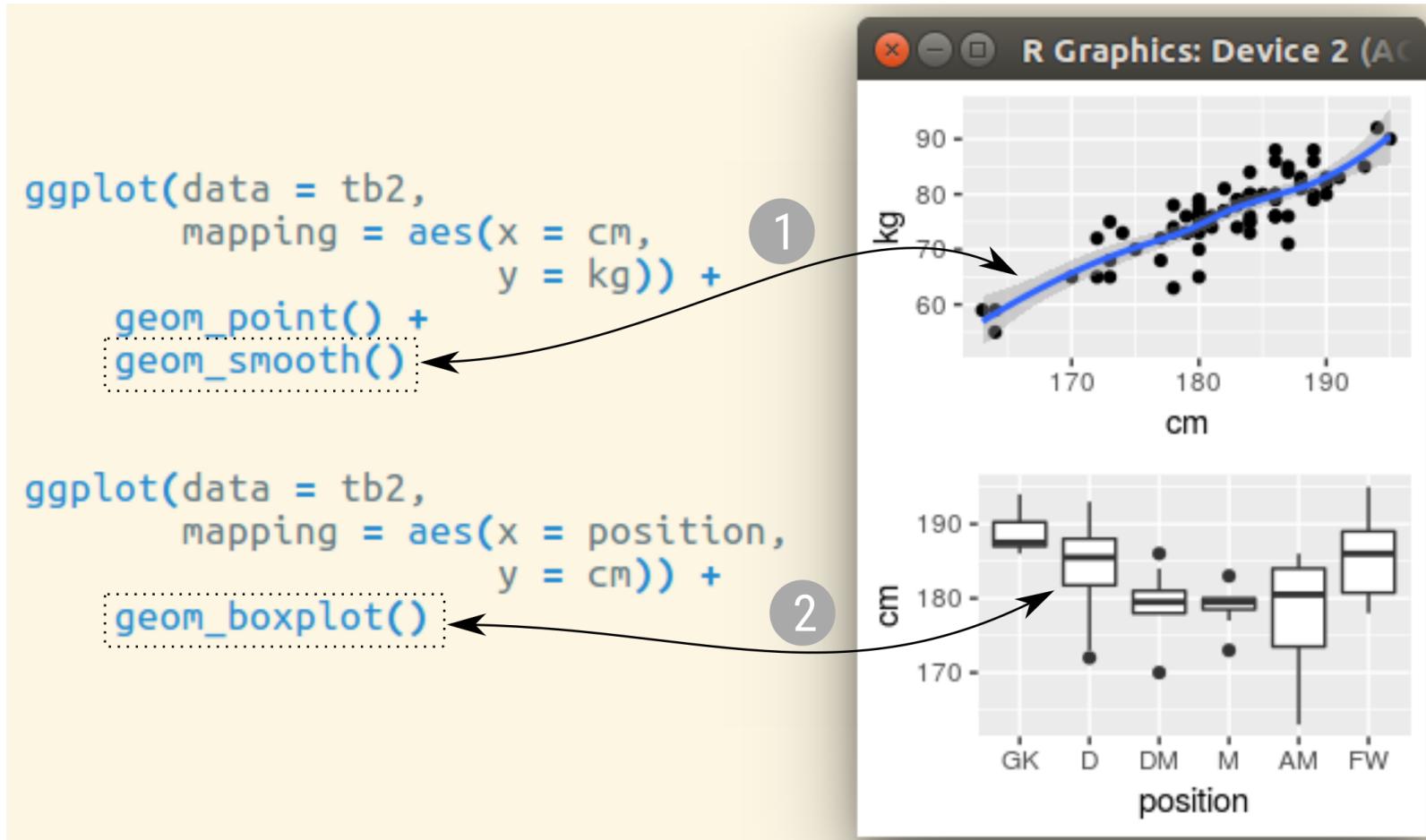
A camada da divisão em facetas.

## Camada 4: divisão em facetas

Funções disponíveis para divisão em facetas:

```
facet_grid  
facet_null  
facet_wrap
```

## Camada 5: estatística



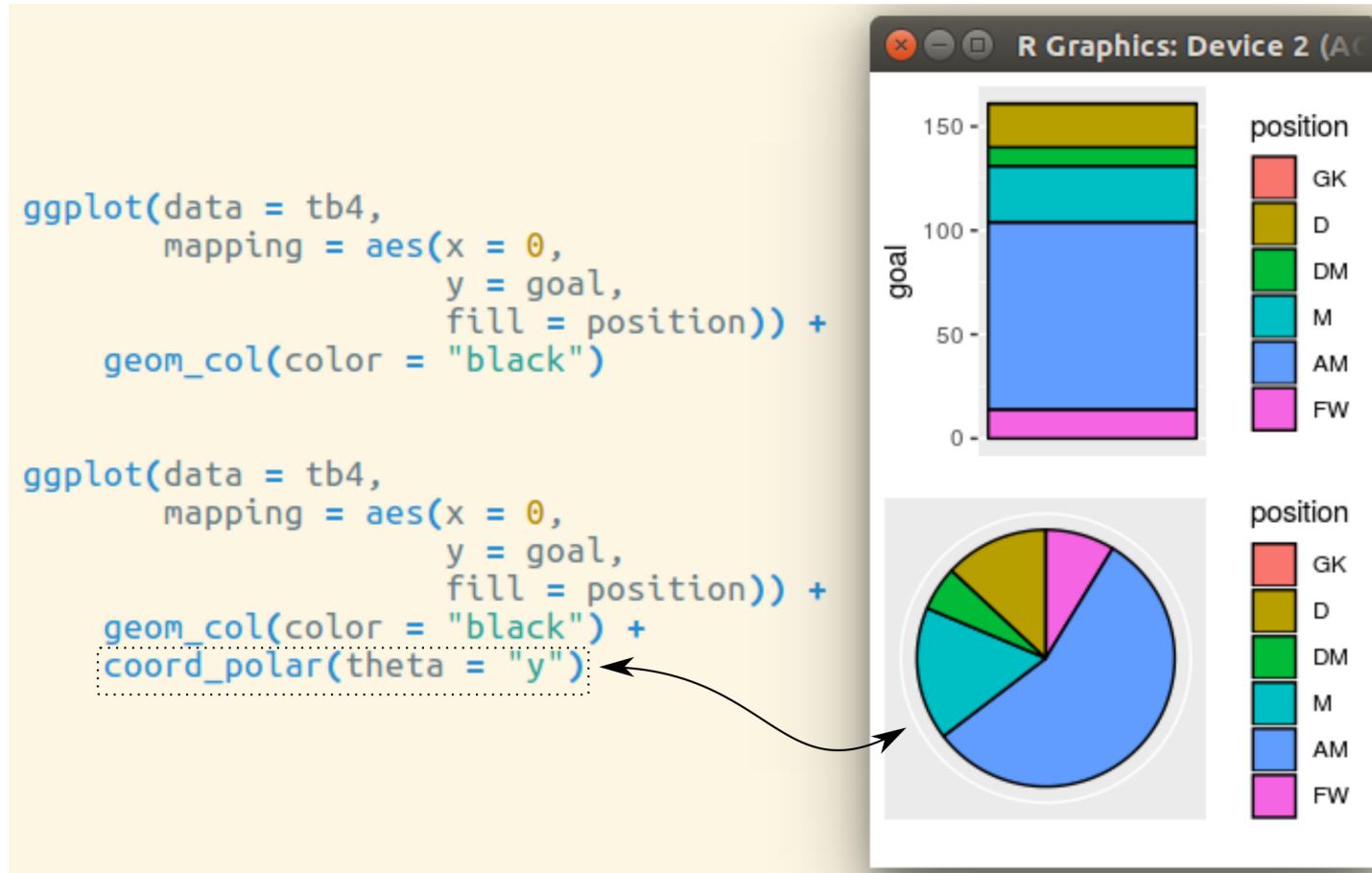
A camada de elementos de estatística.

## Camada 5: estatística

Funções disponíveis para a camada estatística:

```
stat_bin           stat_density_2d_filled  stat_sf_coordinates
stat_bin_2d        stat_density2d       stat_smooth
stat_bin_hex       stat_density2d_filled  stat_spoke
stat_bin2d         stat_ecdf            stat_sum
stat_binhex        stat_ellipse          stat_summary
stat_boxplot       stat_function         stat_summary_2d
stat_contour       stat_identity         stat_summary_bin
stat_contour_filled stat_qq              stat_summary_hex
stat_count          stat_qq_line          stat_summary2d
stat_density        stat_quantile        stat_unique
stat_density_2d     stat_sf              stat_ydensity
```

## Camada 6: coordenadas



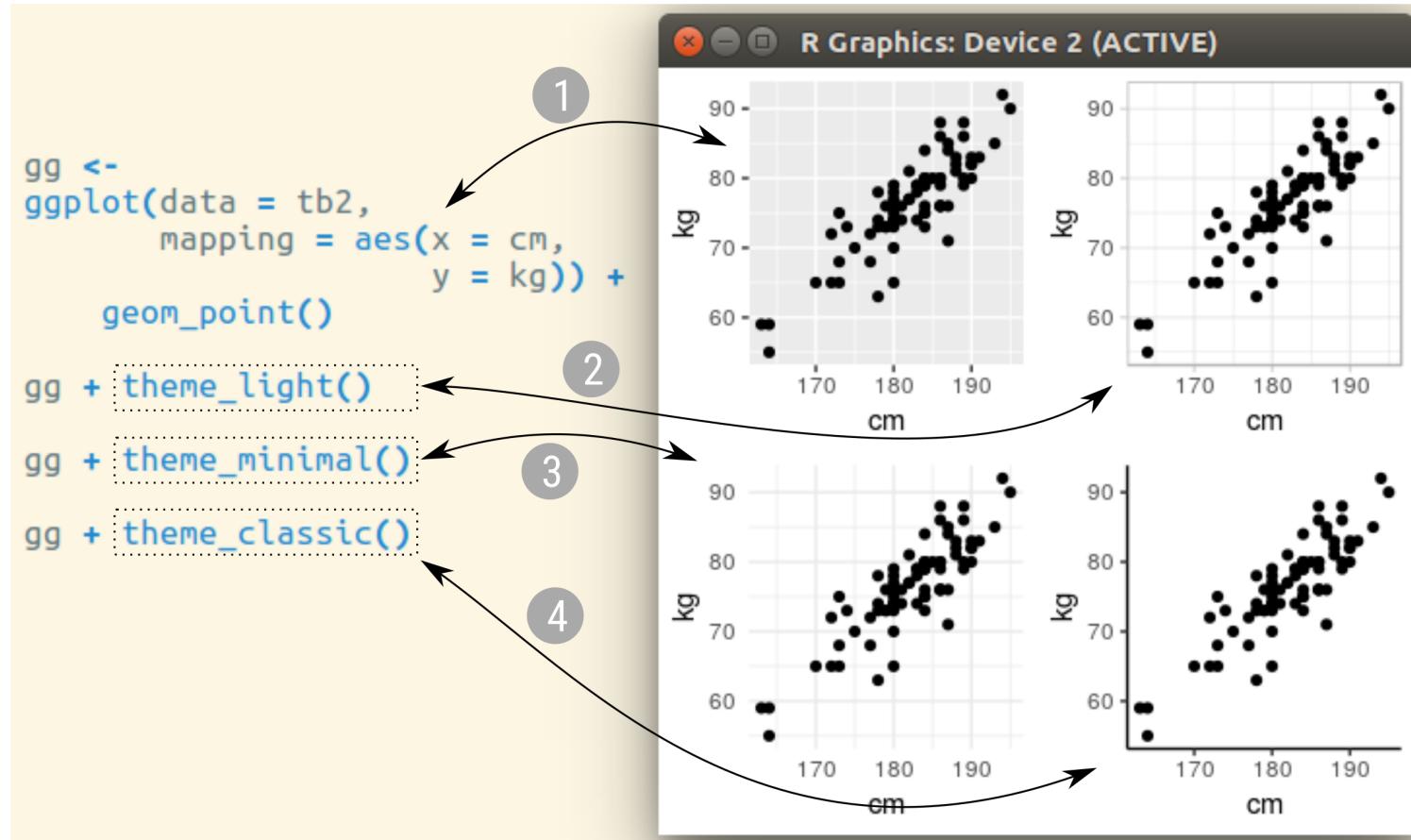
A camada do sistema de coordenadas.

## Camada 6: coordenadas

Funções disponíveis para a camada de coordenadas:

```
coord_cartesian  
coord_equal  
coord_fixed  
coord_flip  
coord_map  
coord_munch  
coord_polar  
coord_quickmap  
coord_sf  
coord_trans
```

## Camada 7: tema



A camada de tema.

## Camada 7: tema

Funções disponíveis para a camada de tema:

```
element_blank  
element_grob  
element_line  
element_rect  
element_render  
element_text  
theme_bw  
theme_classic  
theme_dark  
theme_get  
theme_gray  
theme_grey  
theme_light  
theme_linedraw  
theme_minimal  
theme_replace  
theme_set  
theme_test  
theme_update  
theme_void
```

## Galerias

### Gráficos

- ▶ <https://www.r-graph-gallery.com/portfolio/ggplot2-package/.>
- ▶ <http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html.>
- ▶ <https://r4stats.com/examples/graphics-ggplot2/>.
- ▶ <http://girke.bioinformatics.ucr.edu/GEN242/pages/mydoc/Rgraphics.html.>

### Mapas

- ▶ [https://rstudio-pubs-static.s3.amazonaws.com/176768\\_ec7fb4801e3a4772886d61e65885fbdd.html.](https://rstudio-pubs-static.s3.amazonaws.com/176768_ec7fb4801e3a4772886d61e65885fbdd.html.)
- ▶ <http://girke.bioinformatics.ucr.edu/GEN242/pages/mydoc/Rgraphics.html.>
- ▶ [https://cran.r-project.org/web/packages/geobr/vignettes/intro\\_to\\_geobr.html.](https://cran.r-project.org/web/packages/geobr/vignettes/intro_to_geobr.html.)

## Pacotes para recursos interativos

A visualização interativa é voltada para exibição na WEB. Alguns dos pacotes para isso são estes:

- ▶ `plotly`.
- ▶ `highcharter`.
- ▶ `googleVis`.
- ▶ `echarts4r`.
- ▶ `leaflet`.
- ▶ `iplots`.
- ▶ `rgl`.
- ▶ `animation`.

Para mais detalhes sobre os recursos gráficos, siga esse link:

[https://www.stat.ubc.ca/~jenny/STAT545A/block90\\_baseLatticeGgplot2.html](https://www.stat.ubc.ca/~jenny/STAT545A/block90_baseLatticeGgplot2.html).

Para uma descrição completa da comparação entre `lattice` e `ggplot2`, siga esse link:

<https://learnr.wordpress.com/2009/08/26/ggplot2-version-of-figures-in-lattice-multivariate-data-visualization-with-r-final-part/>.

Confira no **R Graph Gallery** a variedade de gráficos confeccionados com o R.

## Considerações finais

- ▶ A análise exploratória de dados está presente em todos os projetos de ciência de dados.
- ▶ Ela serve para tirar impressões iniciais dos dados como a distribuição das variáveis, a relação entre elas, problemas com valores atípicos, etc.
- ▶ Uma boa análise exploratória subsidia o levantamento de hipóteses, as decisões de negócio, a engenharia de características e a construção e validação de modelos.
- ▶ Dominar ferramentas para uma análise exploratória ágil é importante para celeridade dos projetos e ganho de conhecimento.
- ▶ O R é extraordinário em termos de visualização com vários pacotes e frameworks para visualização de dados.
- ▶ Dispõe ainda de inúmeros recursos para a visualização interativa.

Voltando aos dados

## Dados da aula passada

```
## Com read.table
da <- read.table("dados/JF_Secao_25_Ago_2020.csv", sep = ";",
                  dec = ",", header = TRUE, encoding = "latin1")
str(da)
## Com read.csv
da <- read.csv2("dados/JF_Secao_25_Ago_2020.csv",
                 encoding = "latin1")
str(da)
## Com readr::read_csv2
da <- read_csv2("dados/JF_Secao_25_Ago_2020.csv")
str(da)
## Especificando o encoding
da <- read_csv2("dados/JF_Secao_25_Ago_2020.csv",
                locale = locale(encoding = "latin1"))
str(da)
## Para tirar o spec e problems
attr(da, "spec") <- NULL
attr(da, "problems") <- NULL
## OU, de uma só vez (um subset vazio)
da <- da[]
str(da)
```

## Dados novos

```
library(readxl)
url <- "dados/Base de Dados Justiça Estadual 2020 CNJ.xlsx"
da <- read_excel(url)
## Abra o arquivo para ver que a planilha não tem apenas a tabela
str(da)

## Especifica o range de linhas e colunas a serem lidas
da <- read_excel(url, range = "A3:Q31")
str(da)
```