Manipulação de Dados

Jairo Nicolau, inspirado por Claus O. Wilke

Atualização: 2022-05-18

Vamos usar a library `palmerpenguins`

```r
penguins %>%
  glimpse
```

```
Rows: 344
Columns: 8
$ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adel…
$ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torgerse…
$ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1, …
$ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1, …
$ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 186…
$ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475, …
$ sex               <fct> male, female, female, NA, female, male, female, male…
$ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007…
```

Manipulações de dados elementares

- Selecionar linhas: `filter()`
- Selecionar colunas: `select()`
- Ordenar linhas: `arrange()`
- Contar coisas: `count()`
- Criar novas colunas: `mutate()`
- Analisar segmentos: `group_by()` and `summarize()`

Mas, primeiro: o operador pipe %>%

%>% é pronunciado "e então"

O pipe %>% alimenta os dados em funções

```
head(penguins)
```

```
# A tibble: 6 × 8
  species island bill_length_mm bill_depth_mm flipper_length_… body_mass_g sex  
  <fct>   <fct>           <dbl>         <dbl>            <int>       <int> <fct>
1 Adelie  Torge…           39.1          18.7              181        3750 male 
2 Adelie  Torge…           39.5          17.4              186        3800 fema…
3 Adelie  Torge…           40.3          18                195        3250 fema…
4 Adelie  Torge…           NA            NA                 NA          NA <NA> 
5 Adelie  Torge…           36.7          19.3              193        3450 fema…
6 Adelie  Torge…           39.3          20.6              190        3650 male 
# … with 1 more variable: year <int>
```

O pipe %>% alimenta os dados em funções

```r
# head(penguins)

penguins %>%
  head()
```

```
# A tibble: 6 × 8
  species island bill_length_mm bill_depth_mm flipper_length_… body_mass_g sex
  <fct>   <fct>           <dbl>         <dbl>            <int>       <int> <fct>
1 Adelie  Torge…           39.1          18.7              181        3750 male
2 Adelie  Torge…           39.5          17.4              186        3800 fema…
3 Adelie  Torge…           40.3          18                195        3250 fema…
4 Adelie  Torge…           NA            NA                 NA          NA <NA>
5 Adelie  Torge…           36.7          19.3              193        3450 fema…
6 Adelie  Torge…           39.3          20.6              190        3650 male
# … with 1 more variable: year <int>
```

O pipe %>% alimenta os dados em funções

```
ggplot(penguins, aes(bill_length_mm, bill_depth_mm, color = species)) + geom_point()
```

O pipe %>% alimenta os dados em funções

```r
# ggplot(penguins, aes(bill_length_mm, bill_depth_mm, color = species)) + geom_point()

penguins %>%
  ggplot(aes(bill_length_mm, bill_depth_mm, color = species)) + geom_point()
```

Desde R 4.1: O pipe nativo: |>

```r
penguins |>
  ggplot(aes(bill_length_mm, bill_depth_mm, color = species)) + geom_point()
```

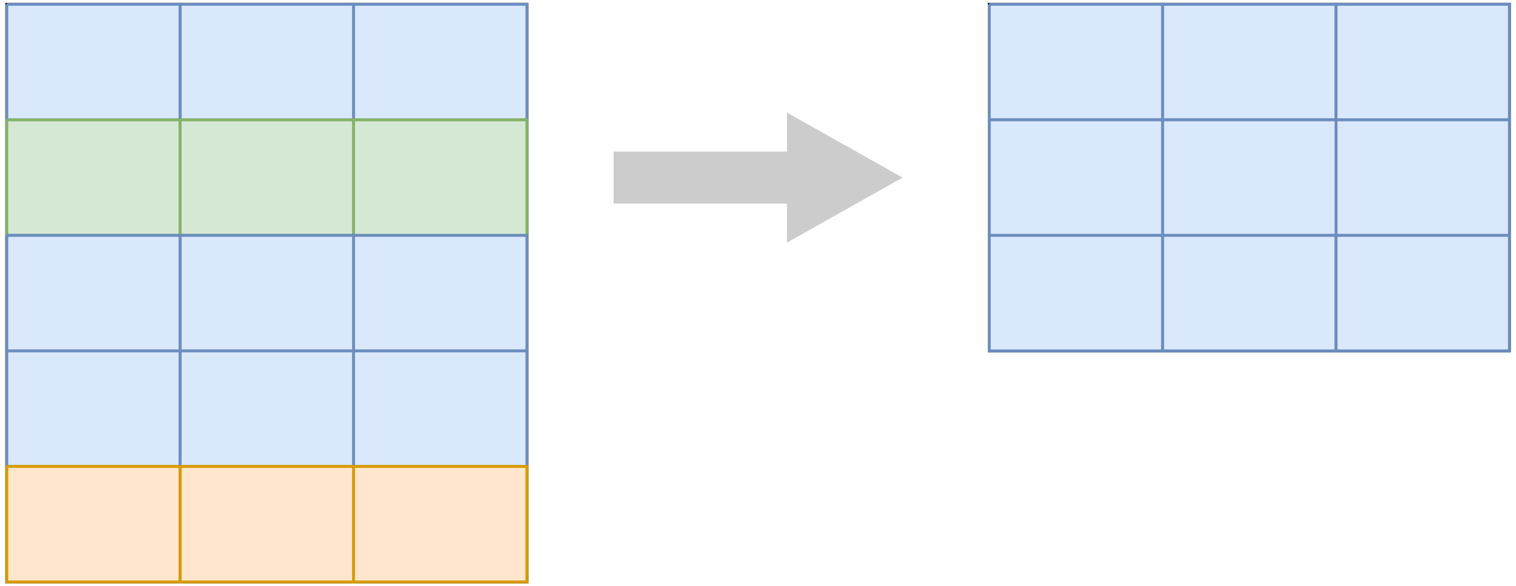
Qual usar? O pipe nativo ou o pipe tradicional?

- `|>` é o futuro. Se puder, use-o.
- `%>%` trabalha em versões omais antigas. É masi seguro por agora

Nós usamos `%>%` porque muitas pessoas ainda usam o R 3.x ou 4.0.

Selecione linhas ou colunas, e ordene

Selecione linhas de uma tabela: `filter()`



Selecione pinguins da espécie Gentoo

```r
penguins %>%
  filter(species == "Gentoo")
```

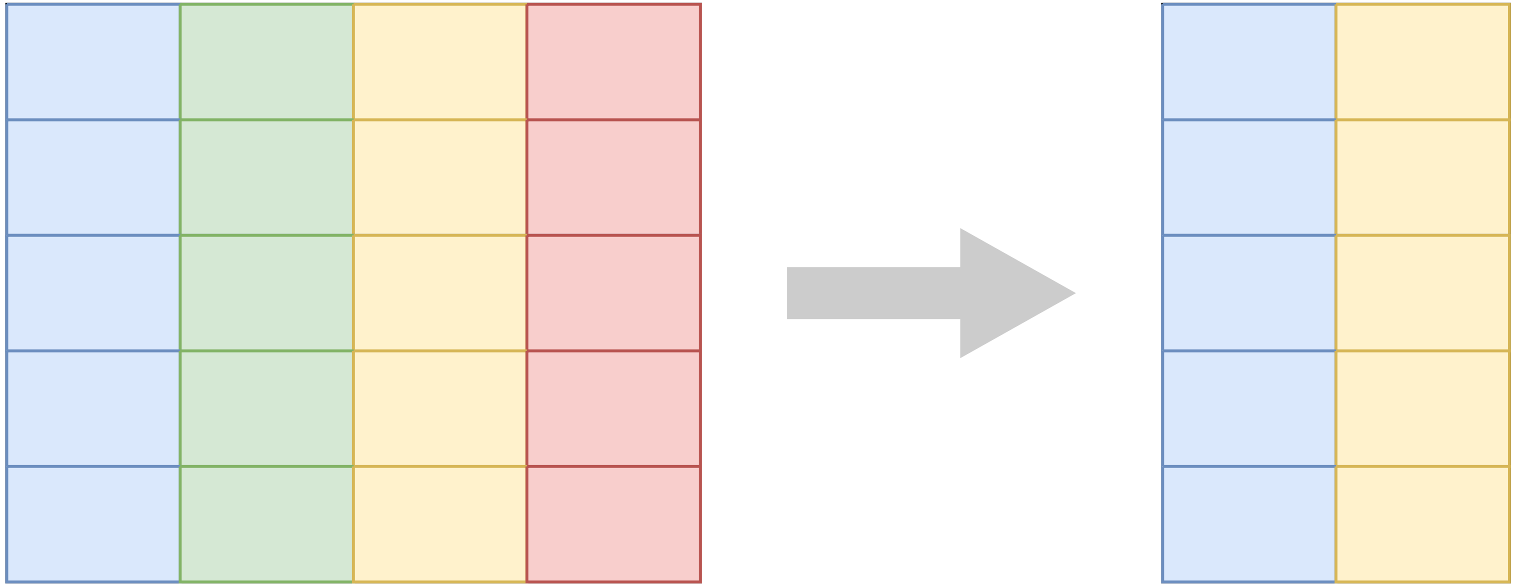
```
# A tibble: 124 × 8
   species island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>           <dbl>         <dbl>             <int>       <int>
 1 Gentoo  Biscoe           46.1          13.2               211        4500
 2 Gentoo  Biscoe           50            16.3               230        5700
 3 Gentoo  Biscoe           48.7          14.1               210        4450
 4 Gentoo  Biscoe           50            15.2               218        5700
 5 Gentoo  Biscoe           47.6          14.5               215        5400
 6 Gentoo  Biscoe           46.5          13.5               210        4550
 7 Gentoo  Biscoe           45.4          14.6               211        4800
 8 Gentoo  Biscoe           46.7          15.3               219        5200
 9 Gentoo  Biscoe           43.3          13.4               209        4400
10 Gentoo  Biscoe           46.8          15.4               215        5150
# … with 114 more rows, and 2 more variables: sex <fct>, year <int>
```

Filtre pinguins com comprimento de bico > 50 mm

```r
penguins %>%
  filter(bill_length_mm > 50)
```

```
# A tibble: 52 × 8
   species island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>           <dbl>         <dbl>             <int>       <int>
 1 Gentoo  Biscoe           50.2          14.3               218        5700
 2 Gentoo  Biscoe           59.6          17                 230        6050
 3 Gentoo  Biscoe           50.5          15.9               222        5550
 4 Gentoo  Biscoe           50.5          15.9               225        5400
 5 Gentoo  Biscoe           50.1          15                 225        5000
 6 Gentoo  Biscoe           50.4          15.3               224        5550
 7 Gentoo  Biscoe           54.3          15.7               231        5650
 8 Gentoo  Biscoe           50.7          15                 223        5550
 9 Gentoo  Biscoe           51.1          16.3               220        6000
10 Gentoo  Biscoe           52.5          15.6               221        5450
# … with 42 more rows, and 2 more variables: sex <fct>, year <int>
```

Selecione colunas de uma tabela: `select()`

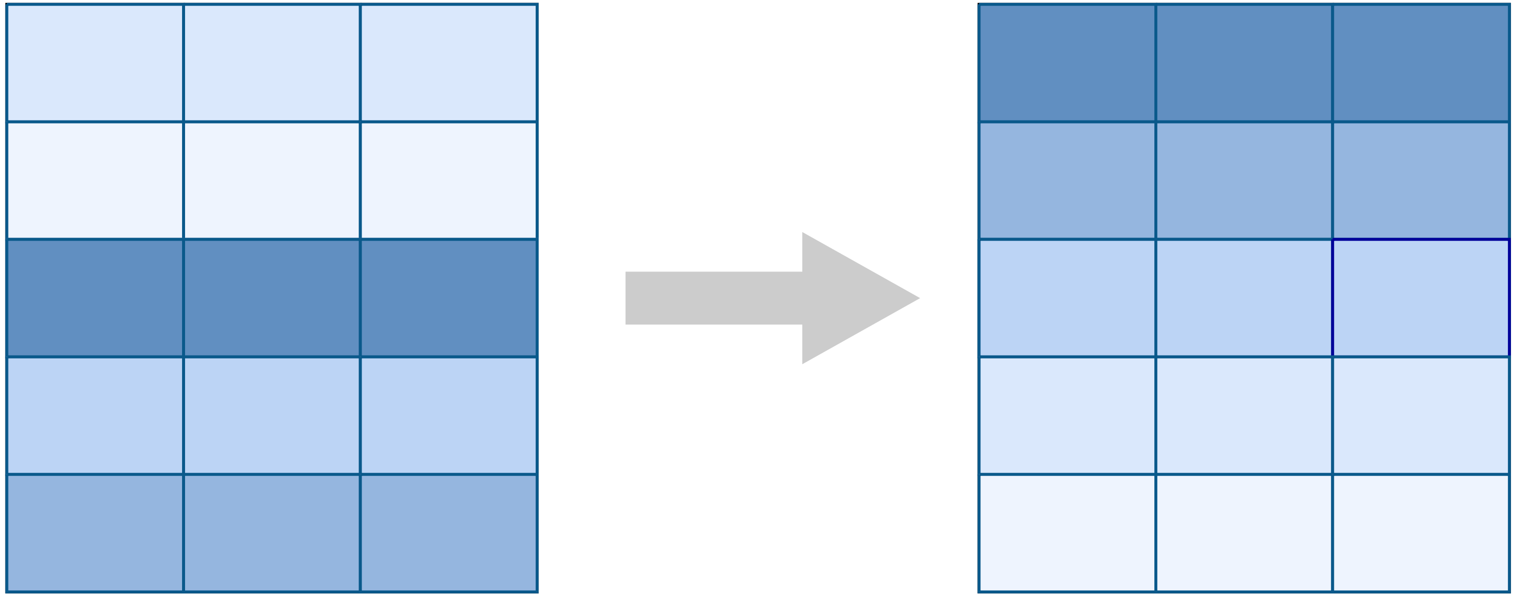


Selecione as colunas species, island, and sex

```
penguins %>%
  select(species, island, sex)
```

```
# A tibble: 344 × 3
   species island    sex   
   <fct>   <fct>     <fct> 
 1 Adelie  Torgersen male  
 2 Adelie  Torgersen female
 3 Adelie  Torgersen female
 4 Adelie  Torgersen <NA>  
 5 Adelie  Torgersen female
 6 Adelie  Torgersen male  
 7 Adelie  Torgersen female
 8 Adelie  Torgersen male  
 9 Adelie  Torgersen <NA>  
10 Adelie  Torgersen <NA>  
# … with 334 more rows
```

Ordene as linhas em uma tabela: `arrange()`



Ordene os pinguins pelo comprimento do bico, crescente

```r
penguins %>%
  arrange(bill_length_mm)
```

```
# A tibble: 344 × 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Dream               32.1          15.5               188        3050
 2 Adelie  Dream               33.1          16.1               178        2900
 3 Adelie  Torgersen           33.5          19                 190        3600
 4 Adelie  Dream               34            17.1               185        3400
 5 Adelie  Torgersen           34.1          18.1               193        3475
 6 Adelie  Torgersen           34.4          18.4               184        3325
 7 Adelie  Biscoe              34.5          18.1               187        2900
 8 Adelie  Torgersen           34.6          21.1               198        4400
 9 Adelie  Torgersen           34.6          17.2               189        3200
10 Adelie  Biscoe              35            17.9               190        3450
# … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Classifique os pinguins pelo comprimento do bico, decrescente

```
penguins %>%
  arrange(desc(bill_length_mm))
```

```
# A tibble: 344 × 8
   species   island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>     <fct>           <dbl>         <dbl>             <int>       <int>
 1 Gentoo    Biscoe           59.6          17                 230        6050
 2 Chinstrap Dream            58            17.8               181        3700
 3 Gentoo    Biscoe           55.9          17                 228        5600
 4 Chinstrap Dream            55.8          19.8               207        4000
 5 Gentoo    Biscoe           55.1          16                 230        5850
 6 Gentoo    Biscoe           54.3          15.7               231        5650
 7 Chinstrap Dream            54.2          20.8               201        4300
 8 Chinstrap Dream            53.5          19.9               205        4500
 9 Gentoo    Biscoe           53.4          15.8               219        5500
10 Chinstrap Dream            52.8          20                 205        4550
# … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Contando coisas

Contando coisas

```
penguins
```

```
# A tibble: 344 × 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Contando coisas

```
penguins %>%
  count(species)
```

```
# A tibble: 3 × 2
  species       n
  <fct>     <int>
1 Adelie      152
2 Chinstrap    68
3 Gentoo      124
```

Contando coisas

```
penguins %>%
  count(species, island)
```

```
# A tibble: 5 × 3
  species   island        n
  <fct>     <fct>     <int>
1 Adelie    Biscoe       44
2 Adelie    Dream        56
3 Adelie    Torgersen    52
4 Chinstrap Dream        68
5 Gentoo    Biscoe      124
```

Use o pipe para construir pipelines (tubulações) de dados

```r
penguins %>%
  filter(species == "Adelie")
```

```
# A tibble: 152 × 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# … with 142 more rows, and 2 more variables: sex <fct>, year <int>
```

Use o pipe para construir pipelines (tubulações) de dados

```r
penguins %>%
  filter(species == "Adelie") %>%
  select(island, sex)
```

```
# A tibble: 152 × 2
   island    sex   
   <fct>     <fct> 
 1 Torgersen male  
 2 Torgersen female
 3 Torgersen female
 4 Torgersen <NA>  
 5 Torgersen female
 6 Torgersen male  
 7 Torgersen female
 8 Torgersen male  
 9 Torgersen <NA>  
10 Torgersen <NA>  
# … with 142 more rows
```

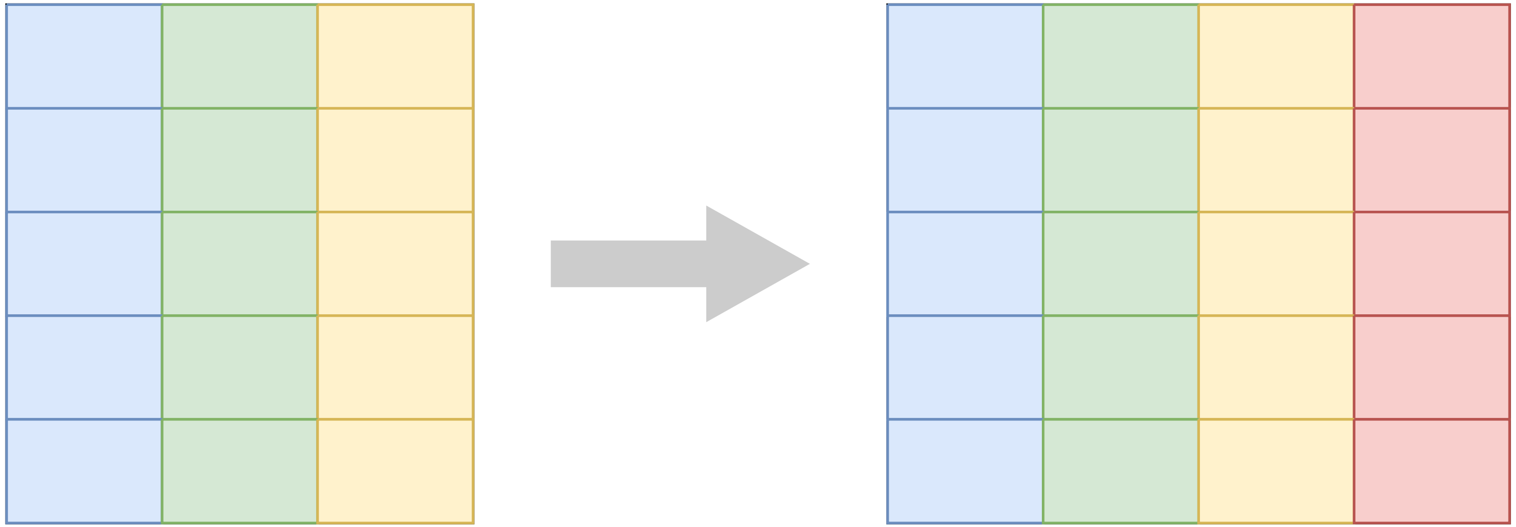
Use o pipe para construir pipelines (tubulações) de dados

```r
penguins %>%
  filter(species == "Adelie") %>%
  select(island, sex) %>%
  count(island, sex)
```

```
# A tibble: 8 × 3
  island    sex        n
  <fct>     <fct>  <int>
1 Biscoe    female    22
2 Biscoe    male      22
3 Dream     female    27
4 Dream     male      28
5 Dream     <NA>       1
6 Torgersen female    24
7 Torgersen male      23
8 Torgersen <NA>       5
```

Acrescente novas colunas ao banco de dados

Construa uma nova coluna: `mutate()`



Exemplo: comprimento da nadadeira em cm

```r
penguins %>%
  select(species, island, flipper_length_mm)
```

```
# A tibble: 344 × 3
   species island    flipper_length_mm
   <fct>   <fct>                 <int>
 1 Adelie  Torgersen               181
 2 Adelie  Torgersen               186
 3 Adelie  Torgersen               195
 4 Adelie  Torgersen                NA
 5 Adelie  Torgersen               193
 6 Adelie  Torgersen               190
 7 Adelie  Torgersen               181
 8 Adelie  Torgersen               195
 9 Adelie  Torgersen               193
10 Adelie  Torgersen               190
# … with 334 more rows
```

Exemplo: comprimento do bico em cm

```
penguins %>%
  select(species, island, flipper_length_mm) %>%
  mutate(flipper_length_cm = flipper_length_mm / 10)
```

```
# A tibble: 344 × 4
   species island    flipper_length_mm flipper_length_cm
   <fct>   <fct>                 <int>             <dbl>
 1 Adelie  Torgersen               181              18.1
 2 Adelie  Torgersen               186              18.6
 3 Adelie  Torgersen               195              19.5
 4 Adelie  Torgersen                NA              NA  
 5 Adelie  Torgersen               193              19.3
 6 Adelie  Torgersen               190              19  
 7 Adelie  Torgersen               181              18.1
 8 Adelie  Torgersen               195              19.5
 9 Adelie  Torgersen               193              19.3
10 Adelie  Torgersen               190              19  
# … with 334 more rows
```

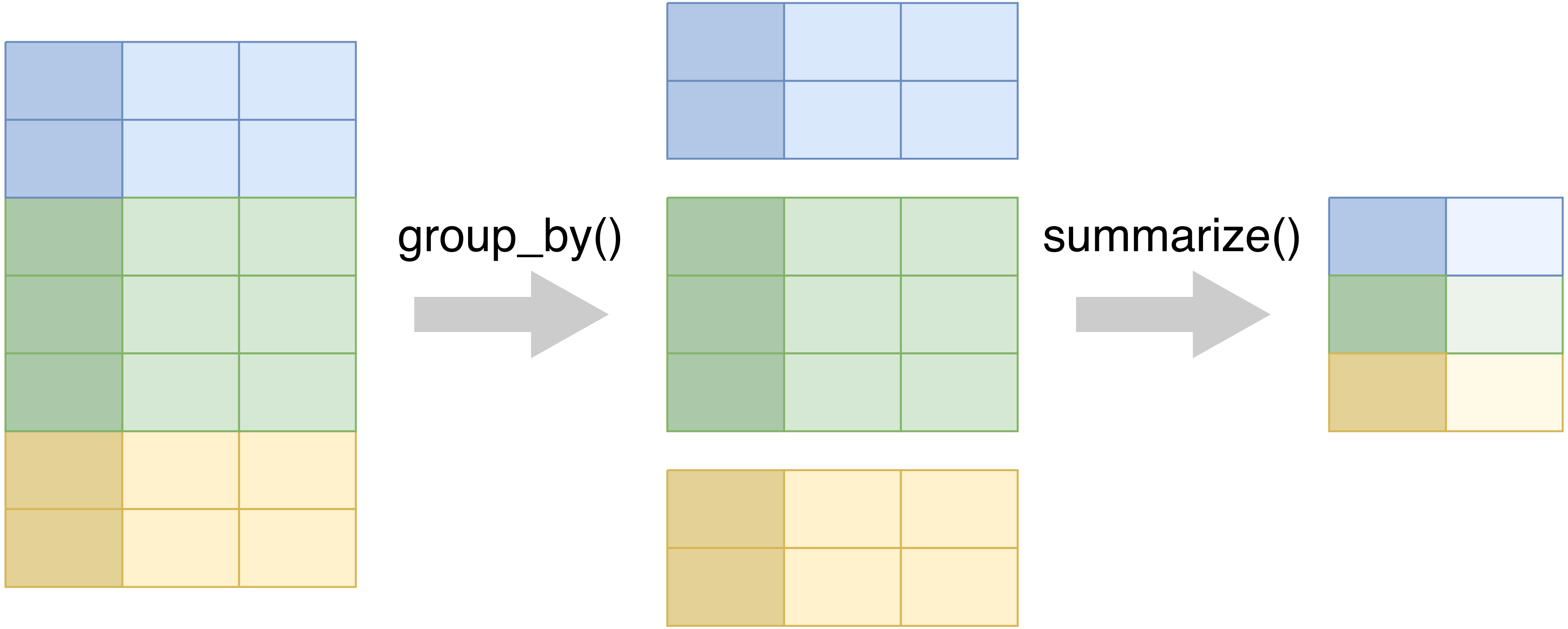
Faça várias colunas de uma só vez

```
penguins %>%
  select(species, island, flipper_length_mm) %>%
  mutate(
    flipper_length_cm = flipper_length_mm / 10,  # <- observe a virgula
    flipper_length_in = flipper_length_mm / 25.4
  )
```

```
# A tibble: 344 × 5
   species island    flipper_length_mm flipper_length_cm flipper_length_in
   <fct>   <fct>                 <int>             <dbl>             <dbl>
 1 Adelie  Torgersen               181              18.1              7.13
 2 Adelie  Torgersen               186              18.6              7.32
 3 Adelie  Torgersen               195              19.5              7.68
 4 Adelie  Torgersen                NA              NA               NA   
 5 Adelie  Torgersen               193              19.3              7.60
 6 Adelie  Torgersen               190              19                7.48
 7 Adelie  Torgersen               181              18.1              7.13
 8 Adelie  Torgersen               195              19.5              7.68
 9 Adelie  Torgersen               193              19.3              7.60
10 Adelie  Torgersen               190              19                7.48
# … with 334 more rows
```

Analise segmentos: `group_by()` and `summarize()`

Analise segmentos: `group_by()` and `summarize()`



Exemplo de aplicação de agrupamento: Contagem

Anteriormente, contamos assim:

```
penguins %>%
  count(species)
```

```
# A tibble: 3 × 2
  species       n
  <fct>     <int>
1 Adelie      152
2 Chinstrap    68
3 Gentoo      124
```

Agora vamos fazer da maneira mais difícil

Exemplo de aplicação de agrupamento: Contagem

Vamos voltar aos dados brutos:

```
penguins
```

```
# A tibble: 344 × 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Exemplo de aplicação de agrupamento: Contagem

vamos agrupá-lo:

```
penguins %>%
  group_by(species)
```

```
# A tibble: 344 × 8
# Groups:   species [3]
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Exemplo de aplicação de agrupamento: Contagem

e sumarize:

```
penguins %>%
  group_by(species) %>%
  summarize(
    n = n()  # n() returns the number of observations per group
  )
```

```
# A tibble: 3 × 2
  species       n
  <fct>     <int>
1 Adelie      152
2 Chinstrap    68
3 Gentoo      124
```

Exemplo de aplicação de agrupamento: Contagem

agora vamos agrupar por multiplas variáveis:

```
penguins %>%
  group_by(species, island)
```

```
# A tibble: 344 × 8
# Groups:   species, island [5]
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

Exemplo de aplicação de agrupamento: Contagem

e sumarize:

```
penguins %>%
  group_by(species, island) %>%
  summarize(
    n = n()  # n() returns the number of observations per group
  )
```

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

```
# A tibble: 5 × 3
# Groups:   species [3]
  species   island        n
  <fct>     <fct>     <int>
1 Adelie    Biscoe       44
2 Adelie    Dream        56
3 Adelie    Torgersen    52
4 Chinstrap Dream        68
5 Gentoo    Biscoe      124
```

Exemplo de aplicação de agrupamento: Contagem

`count(...)` is a short-cut for `group_by(...) %>% summarize(n = n())`

```
penguins %>%
  count(species)
```

```
# A tibble: 3 × 2
  species       n
  <fct>     <int>
1 Adelie      152
2 Chinstrap    68
3 Gentoo      124
```

```
penguins %>%
  group_by(species) %>%
  summarize(
    n = n()
  )
```

```
# A tibble: 3 × 2
  species       n
  <fct>     <int>
1 Adelie      152
2 Chinstrap    68
3 Gentoo      124
```

o output é exatamente o mesmo

Realizando múltiplos sumários de uma vez

```
penguins %>%
  group_by(species) %>%
  summarize(
    n = n(),                                      # number of penguins
    mean_mass = mean(body_mass_g),                # mean body mass
    max_flipper_length = max(flipper_length_mm),  # max flipper length
    percent_female = sum(sex == "female")/n()     # percent of female penguins
  )
```

```
# A tibble: 3 × 5
  species       n mean_mass max_flipper_length percent_female
  <fct>     <int>     <dbl>              <int>          <dbl>
1 Adelie      152       NA                  NA         NA    
2 Chinstrap    68     3733.                212          0.5  
3 Gentoo      124       NA                  NA         NA    
```

Cada etapa do `summarize()` cria uma nova coluna

Mas por que os `NA`s?

Realizando múltiplos sumários de uma vez

```
penguins %>%
  group_by(species) %>%
  summarize(
    n = n(),
    mean_mass = mean(body_mass_g, na.rm = TRUE),
    max_flipper_length = max(flipper_length_mm, na.rm = TRUE),
    percent_female = sum(sex == "female", na.rm = TRUE)/sum(!is.na(sex))
  )
```

```
# A tibble: 3 × 5
  species       n mean_mass max_flipper_length percent_female
  <fct>     <int>     <dbl>              <int>          <dbl>
1 Adelie      152     3701.                210          0.5
2 Chinstrap    68     3733.                212          0.5
3 Gentoo      124     5076.                231          0.487
```

Nós necessitamos dizer para o R exatamente como os NAs devem ser tratados