

# R para Ciência de Dados I

Manipulando bases de dados



# Manipulando bases de dados

# O pacote dplyr

O `dplyr` é o pacote mais útil para realizar manipulação de dados, pois possui funções para fazer virtualmente qualquer tipo de transformação nas linhas e colunas da base.

As principais funções do `dplyr` são:

- `filter()`: filtra linhas
- `select()`: seleciona colunas
- `arrange()`: ordena as linhas conforme os valores de uma coluna
- `mutate()`: modifica ou cria novas colunas
- `group_by()`: agrupa a base conforme uma coluna
- `summarise()`: sumariza colunas

Todas essas funções seguem as seguintes características:

- A função sempre recebe uma `tibble` e sempre devolve uma `tibble`.
- Colocamos o `tibble` no primeiro argumento e o que queremos fazer nos demais argumentos.

Os exemplos apresentados aqui continuarão a usar a base IMDB. Não se esqueça de carregar o pacote `tidyverse` e carregar a base de dados.

```
library(tidyverse)
```

```
imdb <- read_rds("dados/imdb.rds")
```

# Selecionando colunas

Para selecionar colunas, utilizamos a função `select()`.

O primeiro argumento da função é a base de dados e os demais argumentos são os nomes das colunas que você gostaria de selecionar. Repare que você não precisa colocar o nome da coluna entre aspas.

```
select(imdb, titulo)
```

```
## # A tibble: 28,490 × 1
##   titulo
##   <chr>
## 1 Prestige
## 2 Nob Hill
## 3 The Shade
## 4 Viewer Discretion Advised
## 5 Broadcast News
## 6 Murder, He Says
## 7 Mother Is a Freshman
## 8 On the Threshold of Space
## 9 Her Name Was Torment
## 10 It Lives in the Attic
## # i 28,480 more rows
```

Você também pode selecionar várias colunas.

```
select(imdb, titulo, ano, orcamento)
```

```
## # A tibble: 28,490 × 3
##   titulo          ano orcamento
##   <chr>          <dbl>     <dbl>
## 1 Prestige        1931         NA
## 2 Nob Hill        1945         NA
## 3 The Shade       1999     400000
## 4 Viewer Discretion Advised 1998         NA
## 5 Broadcast News  1987    20000000
## 6 Murder, He Says 1945         NA
## 7 Mother Is a Freshman 1949         NA
## 8 On the Threshold of Space 1956     1505000
## 9 Her Name Was Torment 2014         500
## 10 It Lives in the Attic 2016     1000
## # i 28,480 more rows
```

O operador : é muito útil para selecionar colunas consecutivas.

```
select(imdb, titulo:generos)
```

```
## # A tibble: 28,490 × 4
##   titulo          ano data_lancamento generos
##   <chr>          <dbl> <chr>             <chr>
## 1 Prestige       1931 1932-01-22        Adventure, Drama
## 2 Nob Hill       1945 1945-11-15        Drama, Musical
## 3 The Shade     1999 2000-03-01        Drama
## 4 Viewer Discretion Advised 1998 2012-05-01        Comedy, Horror
## 5 Broadcast News 1987 1988-04-01        Comedy, Drama, Romance
## 6 Murder, He Says 1945 1945-06-23        Comedy, Crime, Mystery
## 7 Mother Is a Freshman 1949 1949-06-10        Comedy
## 8 On the Threshold of Space 1956 1956-04-30        Drama
## 9 Her Name Was Torment 2014 2014-04-29        Horror
## 10 It Lives in the Attic 2016 2016-11-12        Thriller
## # i 28,480 more rows
```

O dplyr possui o conjunto de funções auxiliares muito úteis para seleção de colunas. As principais são:

- `starts_with()`: para colunas que começam com um texto padrão
- `ends_with()`: para colunas que terminam com um texto padrão
- `contains()`: para colunas que contêm um texto padrão

Selecionamos a seguir todas as colunas que começam com o texto "num".

```
select(imdb, starts_with("num"))
```

```
## # A tibble: 28,490 × 3
##   num_avaliacoes num_criticas_publico num_criticas_critica
##   <dbl>          <dbl>          <dbl>
## 1             240             12             2
## 2             246             11             2
## 3             102              1             1
## 4             111              5             3
## 5          26257          142            62
## 6          1639           35            10
## 7             310              8             5
## 8             162              9            NA
## 9             115              4             5
## 10            197             12             7
## # i 28,480 more rows
```



# Ordenando linhas

Para ordenar linhas, utilizamos a função `arrange()`. O primeiro argumento é a base de dados. Os demais argumentos são as colunas pelas quais queremos ordenar as linhas. No exemplo a seguir, ordenamos as linhas da base por ordem crescente de orçamento.

```
arrange(imdb, orcamento)
```

```
## # A tibble: 28,490 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>          <chr>    <dbl> <chr> <chr>    <dbl>
## 1 tt5345298 Patie... 2016 2016-10-11 Horror      116 USA Icela...      0
## 2 tt7692822 Driven  2019 2019-02-09 Comedy...    90 USA Engli...      0
## 3 tt117773... Murde... 2020 2020-03-21 Crime,...    80 USA Engli...      0
## 4 tt4540326 Buried  2011 2011-06-25 Thrill...    95 USA Engli...      0
## 5 tt4655630 Drift... 2016 2017-02-24 Crime,...    86 USA Engli...      0
## 6 tt7604948 Gutte... 2019 2019-04-28 Crime,...   101 USA Engli...      0
## 7 tt2226440 Dead ... 2018 2018-01-02 Action...   180 USA Engli...      0
## 8 tt3889450 Ameri... 2017 2017-02-01 Comedy...    96 USA Engli...      0
## 9 tt5252440 Hot W... 2005 2005-10-01 Animat...    60 USA Engli...      0
## 10 tt5229754 She W... 2016 2016-06-24 Horror       76 USA Engli...      0
## # i 28,480 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```

Também podemos ordenar de forma decrescente usando a função `desc()`.

```
arrange(imdb, desc(orcamento))
```

```
## # A tibble: 28,490 × 20
##   id_filme  titulo    ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr>  <dbl> <chr>              <chr>   <dbl> <chr> <chr>    <dbl>
## 1 tt4154796 Aveng... 2019 2019-04-24      Action...   181 USA  Engli... 356000000
## 2 tt4154756 Aveng... 2018 2018-04-25      Action...   149 USA  Engli... 321000000
## 3 tt2527336 Star ... 2017 2017-12-13      Action...   152 USA  Engli... 317000000
## 4 tt0449088 Pirat... 2007 2007-05-23      Action...   169 USA  Engli... 300000000
## 5 tt2527338 Star ... 2019 2019-12-18      Action...   141 USA  Engli... 275000000
## 6 tt3778644 Solo:... 2018 2018-05-23      Action...   135 USA  Engli... 275000000
## 7 tt0348150 Super... 2006 2006-09-01      Action...   154 USA  Engli... 270000000
## 8 tt0398286 Tangl... 2010 2010-11-26      Animat...   100 USA  Engli... 260000000
## 9 tt0413300 Spide... 2007 2007-05-01      Action...   139 USA  Engli... 258000000
## 10 tt2975590 Batma... 2016 2016-03-23      Action...   152 USA  Engli... 250000000
## # i 28,480 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```

E claro, ordenar segundo duas ou mais colunas.

```
arrange(imdb, desc(ano), desc(orcamento))
```

```
## # A tibble: 28,490 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>                <chr>    <dbl> <chr> <chr>    <dbl>
## 1 tt6587640 Troll... 2020 2020-04-10          Animat...    90 USA  Engli... 90000000
## 2 tt7713068 Birds... 2020 2020-02-06          Action...   109 USA  Engli... 84500000
## 3 tt5774060 Under... 2020 2020-01-30          Action...    95 USA  Engli... 80000000
## 4 tt6820324 Timmy... 2020 2020-03-24          Advent...    99 USA  Engli... 45000000
## 5 tt1634106 Blood... 2020 2020-03-27          Action...   109 USA  Engli... 45000000
## 6 tt100595... Unhin... 2020 2020-09-24          Action...    90 USA  Engli... 33000000
## 7 tt8461224 The T... 2020 2020-08-07          Action...    95 USA  Engli... 30000000
## 8 tt7939428 10 Th... 2020 2020-02-21          Romance     74 USA  Engli... 25000000
## 9 tt103089... Force... 2020 2020-06-30          Action...    91 USA  Engli... 23000000
## 10 tt4411584 The S... 2020 2020-07-31          Drama,...  107 USA  Engli... 21000000
## # i 28,480 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```

# Aplicando mais de uma operação

Na grande maioria dos casos, vamos aplicar mais de uma função de manipulação em uma base para obtermos a tabela que desejamos. Poderíamos, por exemplo, querer uma tabela apenas com o título e ano dos filmes, ordenada de forma crescente de lançamento. Para fazer isso, poderíamos aninhar as funções

```
arrange(select(imdb, titulo, ano), ano)
```

ou criar um objeto intermediário

```
tab_titulo_ano <- select(imdb, titulo, ano)
arrange(tab_titulo_ano, ano)
```

Os dois códigos funcionam e levam ao mesmo resultado, mas não são eficientes.

A primeira alternativa é ruim de escrever, já que precisamos escrever primeiro a função que roda por último, e de ler, pois é difícil identificar qual argumento pertence a qual função.

A segunda alternativa é ruim pois exige a criação de objetos auxiliares. Se quiséssimos aplicar 10 operações na base, precisaríamos criar 9 objetos intermediários.

A solução para aplicar diversas operações de manipulação em uma base de dados é aplicar o operador pipe: `|>`.

# O operador pipe |>

A ideia do operador pipe é a seguinte: ele vai aplicar a função do lado direito ao objeto do lado esquerdo.

No exemplo a seguir, estamos aplicando a função `sum()` (lado direito) no objeto `vetor` (lado esquerdo).

```
vetor <- c(1, 2, 3)
vetor |> sum()
```

```
## [1] 6
```

O código acima é equivalente a:

```
sum(vetor)
```

```
## [1] 6
```

Na prática, o pipe coloca o objeto do lado esquerdo no primeiro argumento da função no lado direito. Se precisarmos passar mais argumentos para a função, podemos fazer isso normalmente. É como se estivéssemos escrevendo a função, omitindo o primeiro argumento.

```
vetor <- c(1, 2, 3, NA)
vetor |> sum(na.rm = TRUE)
```

```
## [1] 6
```

O código acima é equivalente a

```
sum(vetor, na.rm = TRUE)
```

```
## [1] 6
```

Quando estamos aplicando apenas uma função, o pipe não parece trazer vantagens. Mas vamos ver como fica o nosso exemplo do imdb utilizando esse operador:

```
# Sem pipe
arrange(select(imdb, titulo, ano), ano)

# Com pipe
imdb |>
  select(titulo, ano) |>
  arrange(ano)
```

O que está sendo feito no código com pipe? Da primeira para a segunda linha, estamos aplicando a função `select()` à base `imdb`. Da segunda para a terceira, estamos aplicando a função `arrange()` à base resultante da função `select()`.

O resultado desse código é idêntico às tentativas sem pipe, com a vantagem de termos escrito o código na ordem em que as funções são aplicadas, de termos um código muito mais legível e de não precisarmos utilizar objetos intermediários.



# Filtrando linhas

Para filtrar valores de uma coluna da base, utilizamos a função `filter()`.

```
imdb |> filter(nota_imdb > 9)
```

```
## # A tibble: 8 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>              <chr>   <dbl> <chr> <chr>      <dbl>
## 1 tt10218912 As I ... 2019 2019-12-06      Drama,...    62 USA  Engli...    10000
## 2 tt6735740 Love ... 2019 2019-06-23      Comedy      100 USA  Engli...   3000000
## 3 tt0111161 The S... 1994 1995-02-10      Drama       142 USA  Engli...  25000000
## 4 tt0068646 The G... 1972 1972-09-21      Crime,...   175 USA  Engli...   6000000
## 5 tt1508669 Hopef... 2010 2010-12-15      Drama       94 USA  Engli...   1500000
## 6 tt11164090 The M... 2020 2020-05-11      Comedy      85 USA  Engli...      NA
## 7 tt5980638 The T... 2018 2020-06-19      Music,...   96 USA  Engli...    90000
## 8 tt6074834 Delaw... 2018 2018-12-21      Drama       98 USA  Engli...      NA
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```

Podemos seleccionar apenas as colunas título e nota para visualizarmos as notas:

```
imdb |>
  filter(nota_imdb > 9) |>
  select(título, nota_imdb)
```

```
## # A tibble: 8 × 2
##   título                nota_imdb
##   <chr>                 <dbl>
## 1 As I Am                9.3
## 2 Love in Kilnerry      9.3
## 3 The Shawshank Redemption 9.3
## 4 The Godfather         9.2
## 5 Hopeful Notes         9.7
## 6 The Moving on Phase    9.5
## 7 The Transcendents     9.2
## 8 Delaware Shore        9.1
```

Podemos estender o filtro para duas ou mais colunas. Para isso, separamos cada operação por uma vírgula.

```
imdb |> filter(ano > 2010,  
              nota_imdb > 8.5,  
              num_avaliacoes > 5000)
```

```
## # A tibble: 3 × 20  
##   id_filme  titulo      ano data_lancamento generos duracao pais idioma orcamento  
##   <chr>    <chr>    <dbl> <chr>              <chr>    <dbl> <chr> <chr>    <dbl>  
## 1 tt8503618 Hamilt... 2020 2020-07-03      Biogra...    160 USA  Engli...    NA  
## 2 tt6019206 Kill B... 2011 2011-03-27      Action...    247 USA  <NA>        NA  
## 3 tt2170667 Wheels   2014 2017-02-01      Drama        115 USA  Engli...    NA  
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,  
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,  
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,  
## #   num_criticas_critica <dbl>
```

Também podemos fazer operações com as colunas da base dentro da função filter. O código abaixo devolve uma tabela apenas com os filmes que lucraram.

```
imdb |> filter(receita - orcamento > 0)
```

```
## # A tibble: 2,584 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>              <chr>   <dbl> <chr> <chr>    <dbl>
## 1 tt0092699 Broad... 1987 1988-04-01      Comedy...   133 USA  Engli... 20000000
## 2 tt0183505 Me, M... 2000 2000-09-08      Comedy     116 USA  Engli... 51000000
## 3 tt0093640 No Wa... 1987 1987-12-11      Action...   114 USA  Engli... 15000000
## 4 tt0494652 Welco... 2008 2008-02-08      Comedy...   104 USA  Engli... 35000000
## 5 tt0285869 Pando... 2002 2002-09-27      Drama,...   103 USA  Engli...   800000
## 6 tt1488555 The C... 2011 2011-12-09      Comedy...   112 USA  Engli... 52000000
## 7 tt0090022 Silve... 1985 1986-01-16      Action...   133 USA  Engli... 26000000
## 8 tt0120434 Vegas... 1997 1997-02-14      Comedy     93 USA  Engli... 25000000
## 9 tt1086772 Blend... 2014 2014-07-02      Comedy...   117 USA  Engli... 40000000
## 10 tt0064115 Butch... 1969 1969-09-26      Biogra...   110 USA  Engli... 6000000
## # i 2,574 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```

Naturalmente, podemos filtrar colunas categóricas. O exemplo abaixo retorna uma tabela apenas com os filmes dirigidos por Quentin Tarantino e George Lucas.

```
imdb |>
  filter(direcao %in% c('Quentin Tarantino', 'George Lucas'))
```

```
## # A tibble: 15 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>          <chr>    <dbl> <chr> <chr>    <dbl>
## 1 tt0076759 Star ... 1977 1977-10-20    Action...    121 USA  Engli... 11000000
## 2 tt0121766 Star ... 2005 2005-05-20    Action...    140 USA  Engli... 113000000
## 3 tt0069704 Ameri... 1973 1974-04-24    Comedy...    110 USA  Engli...   777000
## 4 tt0121765 Star ... 2002 2002-05-16    Action...    142 USA  Engli... 115000000
## 5 tt0110912 Pulp ... 1994 1994-10-28    Crime,...    154 USA  Engli...   8000000
## 6 tt0359715 My Be... 1987 1987          Comedy        90 USA  Engli...    5000
## 7 tt3460252 The H... 2015 2016-02-04    Crime,...    168 USA  Engli... 44000000
## 8 tt6019206 Kill ... 2011 2011-03-27    Action...    247 USA  <NA>      NA
## 9 tt0105236 Reser... 1992 1992-10-09    Crime,...    99 USA  Engli...  1200000
## 10 tt0119396 Jacki... 1997 1998-03-27    Crime,...    154 USA  Engli... 12000000
## 11 tt1028528 Death... 2007 2007-06-01    Action...    127 USA  Engli...    NA
## 12 tt0378194 Kill ... 2004 2004-04-23    Action...    137 USA  Engli... 30000000
## 13 tt0066434 THX 1... 1971 1976-10-29    Drama,...    86 USA  Engli...   777000
## 14 tt1853728 Djang... 2012 2013-01-17    Drama,...    165 USA  Engli... 100000000
## 15 tt0120915 Star ... 1999 1999-09-17    Action...    136 USA  Engli... 115000000
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
```

Para filtrar textos sem correspondência exata, podemos utilizar a função auxiliar `str_detect()`. Ela serve para verificar se cada string de um vetor contém um determinado padrão de texto.

```
str_detect(  
  string = c("a", "aa", "abc", "bc", "A", NA),  
  pattern = "a"  
)
```

```
## [1] TRUE TRUE TRUE FALSE FALSE NA
```

Podemos utilizá-la para filtrar apenas os filmes que contêm o gênero ação.

```
# A coluna gêneros apresenta todos os gêneros dos filmes concatenados  
imdb$generos[1:6]
```

```
## [1] "Adventure, Drama"      "Drama, Musical"      "Drama"  
## [4] "Comedy, Horror"       "Comedy, Drama, Romance" "Comedy, Crime, Mystery"
```

```
# Podemos detectar se o gênero Drama aparece na string
str_detect(
  string = imdb$generos[1:6],
  pattern = "Drama"
)
```

```
## [1] TRUE TRUE TRUE FALSE TRUE FALSE
```

```
# Aplicamos essa lógica dentro da função filter, para a coluna completa
imdb |> filter(str_detect(generos, "Drama"))
```

```
## # A tibble: 13,816 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>          <chr>   <dbl> <chr> <chr>    <dbl>
## 1 tt0023352 Prest... 1931 1932-01-22    Advent...    71 USA  Engli...    NA
## 2 tt0037946 Nob H... 1945 1945-11-15    Drama,...    95 USA  Engli...    NA
## 3 tt0216204 The S... 1999 2000-03-01    Drama        83 USA  Engli...  400000
## 4 tt0092699 Broad... 1987 1988-04-01    Comedy...   133 USA  Engli... 20000000
## 5 tt0049571 On th... 1956 1956-04-30    Drama        98 USA  Engli... 1505000
## 6 tt0421539 9/Ten... 2006 2008-08-28    Drama,...    98 USA  Engli... 1000000
## 7 tt0027200 Westw... 1935 1935-08-19    Action...    61 USA  Engli...   35000
## 8 tt2822280 Confe... 2015 2015-03-24    Drama        90 USA  Engli...    NA
## 9 tt0372122 Adam ... 2005 2007-05-17    Comedy...    99 USA  Engli...    NA
## 10 tt3703836 Henry... 2015 2016-01-08    Drama        87 USA  Engli...    NA
## # i 13,806 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
```

# Criando novas colunas

Para modificar uma coluna existente ou criar uma nova coluna, utilizamos a função `mutate()`. O código abaixo divide os valores da coluna duração por 60, mudando a unidade de medida dessa variável de minutos para horas.

```
imdb |> mutate(duracao = duracao/60)
```

```
## # A tibble: 28,490 × 20
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>          <chr>   <dbl> <chr> <chr>    <dbl>
## 1 tt0023352 Prest... 1931 1932-01-22    Advent... 1.18 USA Engli...    NA
## 2 tt0037946 Nob H... 1945 1945-11-15    Drama,... 1.58 USA Engli...    NA
## 3 tt0216204 The S... 1999 2000-03-01    Drama     1.38 USA Engli... 400000
## 4 tt0171889 Viewe... 1998 2012-05-01    Comedy... 1.75 USA Engli...    NA
## 5 tt0092699 Broad... 1987 1988-04-01    Comedy... 2.22 USA Engli... 20000000
## 6 tt0037931 Murde... 1945 1945-06-23    Comedy... 1.52 USA Engli...    NA
## 7 tt0041659 Mothe... 1949 1949-06-10    Comedy    1.35 USA Engli...    NA
## 8 tt0049571 On th... 1956 1956-04-30    Drama     1.63 USA Engli... 1505000
## 9 tt3772198 Her N... 2014 2014-04-29    Horror    0.833 USA <NA>      500
## 10 tt6212020 It Li... 2016 2016-11-12    Thrill... 1.42 USA Engli... 1000
## # i 28,480 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```



Também poderíamos ter criado essa variável em uma nova coluna. Repare que a nova coluna duracao\_horas é colocada no final da tabela.

```
imdb |> mutate(duracao_horas = duracao/60)
```

```
## # A tibble: 28,490 × 21
##   id_filme  titulo    ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr>  <dbl> <chr>              <chr>   <dbl> <chr> <chr>      <dbl>
## 1 tt0023352 Prest... 1931 1932-01-22      Advent...    71 USA  Engli...    NA
## 2 tt0037946 Nob H... 1945 1945-11-15      Drama,...    95 USA  Engli...    NA
## 3 tt0216204 The S... 1999 2000-03-01      Drama        83 USA  Engli...  400000
## 4 tt0171889 Viewe... 1998 2012-05-01      Comedy...   105 USA  Engli...    NA
## 5 tt0092699 Broad... 1987 1988-04-01      Comedy...   133 USA  Engli... 20000000
## 6 tt0037931 Murde... 1945 1945-06-23      Comedy...    91 USA  Engli...    NA
## 7 tt0041659 Mothe... 1949 1949-06-10      Comedy       81 USA  Engli...    NA
## 8 tt0049571 On th... 1956 1956-04-30      Drama        98 USA  Engli... 1505000
## 9 tt3772198 Her N... 2014 2014-04-29      Horror        50 USA  <NA>       500
## 10 tt6212020 It Li... 2016 2016-11-12      Thrill...    85 USA  Engli...   1000
## # i 28,480 more rows
## # i 12 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>, duracao_horas <dbl>
```

Podemos fazer qualquer operação com uma ou mais colunas. A única regra é que o resultado da operação retorne um vetor com comprimento igual ao número de linhas da base (ou com comprimento 1 para distribuir um mesmo valor em todas as linhas). Você também pode criar/modificar quantas colunas quiser dentro de um mesmo mutate.

```
imdb |>
  mutate(lucro = receita - orcamento, pais = "Estados Unidos") |>
  select(titulo, lucro, pais)
```

```
## # A tibble: 28,490 × 3
##   titulo                lucro pais
##   <chr>                 <dbl> <chr>
## 1 Prestige                NA Estados Unidos
## 2 Nob Hill                NA Estados Unidos
## 3 The Shade              NA Estados Unidos
## 4 Viewer Discretion Advised NA Estados Unidos
## 5 Broadcast News         47331309 Estados Unidos
## 6 Murder, He Says        NA Estados Unidos
## 7 Mother Is a Freshman    NA Estados Unidos
## 8 On the Threshold of Space NA Estados Unidos
## 9 Her Name Was Torment    NA Estados Unidos
## 10 It Lives in the Attic   NA Estados Unidos
## # i 28,480 more rows
```

# Sumarizando colunas

Sumarização é a técnica de se resumir um conjunto de dados utilizando alguma métrica de interesse. A média, a mediana, a variância, a frequência, a proporção, por exemplo, são tipos de sumarização que trazem diferentes informações sobre uma variável.

Para sumarizar uma coluna da base, utilizamos a função `summarize()`. O código abaixo resume a coluna `orcamento` pela sua média.

```
imdb |> summarize(media_orcamento = mean(orcamento, na.rm = TRUE))
```

```
## # A tibble: 1 × 1  
##   media_orcamento  
##             <dbl>  
## 1             12135607.
```

Repare que a saída da função continua sendo uma tibble.

Podemos calcular diversas sumarizações diferentes em um mesmo `summarize`. Cada sumarização será uma coluna da nova base.

```
imdb |> summarise(  
  media_orcamento = mean(orcamento, na.rm = TRUE),  
  mediana_orcamento = median(orcamento, na.rm = TRUE),  
  variancia_orcamento = var(orcamento, na.rm = TRUE)  
)
```

```
## # A tibble: 1 × 3  
##   media_orcamento mediana_orcamento variancia_orcamento  
##   <dbl>           <dbl>           <dbl>  
## 1      12135607.         2000000          7.17e14
```

E também sumarizar diversas colunas.

```
imdb |> summarize(  
  media_orcamento = mean(orcamento, na.rm = TRUE),  
  media_receita = mean(receita, na.rm = TRUE),  
  media_lucro = mean(receita - orcamento, na.rm = TRUE)  
)
```

```
## # A tibble: 1 × 3  
##   media_orcamento media_receita media_lucro  
##           <dbl>           <dbl>           <dbl>  
## 1      12135607.      45375563.      45786607.
```

# Sumarizando colunas agrupadas

Muitas vezes queremos sumarizar uma coluna agrupada pelas categorias de uma segunda coluna. Para isso, além do `summarize`, utilizamos também a função `group_by()`.

O código a seguir calcula a receita média dos filmes para cada categoria da coluna "direcao".

```
imdb |>
  group_by(direcao) |>
  summarise(receita_media = mean(receita, na.rm = TRUE)) |>
  drop_na(receita_media) |>
  arrange(desc(receita_media))
```

```
## # A tibble: 4,032 × 2
##   direcao                receita_media
##   <chr>                  <dbl>
## 1 Anthony Russo, Joe Russo 1368908569.
## 2 Chris Buck, Jennifer Lee 1365415143
## 3 J.A. Bayona             1331958159
## 4 Kyle Balda, Pierre Coffin 1097121269
## 5 James Cameron           1086440534.
## 6 Josh Cooley              1073394593
## 7 Lee Unkrich              1066969703
## 8 Gareth Edwards           1056057720
## 9 Andrew Stanton, Angus MacLane 1028570889
## 10 Paul Verhoeven, Rink M... 1022700075
```

A única alteração que a função `group_by()` faz na base é a marcação de que a base está agrupada.

```
imdb |> group_by(direcao)
```

```
## # A tibble: 28,490 × 20
## # Groups:   direcao [12,459]
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>          <chr>   <dbl> <chr> <chr>    <dbl>
## 1 tt0023352 Prest... 1931 1932-01-22    Advent...   71 USA  Engli...    NA
## 2 tt0037946 Nob H... 1945 1945-11-15    Drama,...   95 USA  Engli...    NA
## 3 tt0216204 The S... 1999 2000-03-01    Drama       83 USA  Engli...  400000
## 4 tt0171889 Viewe... 1998 2012-05-01    Comedy...  105 USA  Engli...    NA
## 5 tt0092699 Broad... 1987 1988-04-01    Comedy...  133 USA  Engli... 20000000
## 6 tt0037931 Murde... 1945 1945-06-23    Comedy...   91 USA  Engli...    NA
## 7 tt0041659 Mothe... 1949 1949-06-10    Comedy     81 USA  Engli...    NA
## 8 tt0049571 On th... 1956 1956-04-30    Drama      98 USA  Engli...  1505000
## 9 tt3772198 Her N... 2014 2014-04-29    Horror     50 USA  <NA>      500
## 10 tt6212020 It Li... 2016 2016-11-12    Thrill...  85 USA  Engli...   1000
## # i 28,480 more rows
## # i 11 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>
```

# Juntando bases

Podemos juntar duas tabelas a partir de uma (coluna) chave utilizando a função `left_join()`. Como exemplo, vamos inicialmente calcular o lucro médio dos filmes de cada pessoa que dirige filme e salvar no objeto `tab_lucro_direcao`.

```
tab_lucro_direcao <- imdb |>
  group_by(direcao) |>
  summarise(lucro_medio = mean(receita - orcamento, na.rm = TRUE))
```

```
tab_lucro_direcao
```

```
## # A tibble: 12,459 × 2
##   direcao                                lucro_medio
##   <chr>                                   <dbl>
## 1 'Evil' Ted Smith                        NaN
## 2 'Philthy' Phil Phillips                 NaN
## 3 A. Dean Bell                            NaN
## 4 A. Edward Sutherland                   NaN
## 5 A. Edward Sutherland, John Rawlins     NaN
## 6 A. Edward Sutherland, Wesley Ruggles  NaN
## 7 A. Raven Cruz                          -996300
## 8 A.D. Calvo                              NaN
## 9 A.J. Edwards                            NaN
## 10 A.J. Kparr                              NaN
## # i 12,449 more rows
```



E se quisermos colocar essa informação na base original? Basta usar a função `left_join()` utilizando a coluna `direcao` como chave. Observe que a coluna `lucro_medio` aparece agora no fim da tabela.

```
imdb_com_lucro_medio <- left_join(imdb, tab_lucro_direcao, by = "direcao")
```

```
imdb_com_lucro_medio
```

```
## # A tibble: 28,490 × 21
##   id_filme  titulo  ano data_lancamento generos duracao pais idioma orcamento
##   <chr>    <chr> <dbl> <chr>             <chr>   <dbl> <chr> <chr>      <dbl>
## 1 tt0023352 Prest... 1931 1932-01-22      Advent...    71 USA  Engli...    NA
## 2 tt0037946 Nob H... 1945 1945-11-15      Drama,...    95 USA  Engli...    NA
## 3 tt0216204 The S... 1999 2000-03-01      Drama        83 USA  Engli...  400000
## 4 tt0171889 Viewe... 1998 2012-05-01      Comedy...   105 USA  Engli...    NA
## 5 tt0092699 Broad... 1987 1988-04-01      Comedy...   133 USA  Engli... 20000000
## 6 tt0037931 Murde... 1945 1945-06-23      Comedy...    91 USA  Engli...    NA
## 7 tt0041659 Mothe... 1949 1949-06-10      Comedy      81 USA  Engli...    NA
## 8 tt0049571 On th... 1956 1956-04-30      Drama        98 USA  Engli... 1505000
## 9 tt3772198 Her N... 2014 2014-04-29      Horror       50 USA  <NA>      500
## 10 tt6212020 It Li... 2016 2016-11-12      Thrill...    85 USA  Engli...   1000
## # i 28,480 more rows
## # i 12 more variables: receita <dbl>, receita_eua <dbl>, nota_imdb <dbl>,
## #   num_avaliacoes <dbl>, direcao <chr>, roteiro <chr>, producao <chr>,
## #   elenco <chr>, descricao <chr>, num_criticas_publico <dbl>,
## #   num_criticas_critica <dbl>, lucro_medio <dbl>
```

Na tabela `imdb_com_lucro_medio`, como na tabela `imdb`, cada linha continua a representar um filme diferente, mas agora temos também a informação do lucro médio da direção de cada filme.

A primeira linha, por exemplo, traz as informações do filme Avatar. O valor do `lucro_medio` nessa linha representa o lucro médio de todos os filmes do James Cameron, que é o diretor de Avatar. Com essa informação, podemos calcular o quanto o lucro do Avatar se afasta do lucro médio do James Cameron.

```
imdb_com_lucro_medio |>
  mutate(
    lucro = receita - orcamento,
    lucro_relativo = (lucro - lucro_medio)/lucro_medio,
    lucro_relativo = scales::percent(lucro_relativo)
  ) |>
  select(titulo, direcao, lucro, lucro_medio, lucro_relativo)
```

```
## # A tibble: 28,490 × 5
##   titulo                direcao          lucro lucro_medio lucro_relativo
##   <chr>                 <chr>          <dbl>     <dbl> <chr>
## 1 Prestige              Tay Garnett     NA         NaN <NA>
## 2 Nob Hill               Henry Hathaway NA        -2966627 <NA>
## 3 The Shade             Raphaël Nadjari NA         NaN <NA>
## 4 Viewer Discretion Advised Eddie Beverly J... NA         NaN <NA>
## 5 Broadcast News        James L. Brooks  4.73e7    47749419. -0.87563%
## 6 Murder, He Says       George Marshall NA         NaN <NA>
## 7 Mother Is a Freshman   Lloyd Bacon     NA        -437400 <NA>
## 8 On the Threshold of Space Robert D. Webb   NA         NaN <NA>
## 9 Her Name Was Torment   Dustin Mills     NA         NaN <NA>
## 10 It Lives in the Attic  Steve Hudgins   NA         NaN <NA>
## # i 28,480 more rows
```

Observamos então que o Avatar obteve um lucro aproximadamente 169% maior que a média dos filmes do James Cameron.