Folha de Exercícios — Manipulação de Dados em R

dplyr • tidyr • joins

17-10-2025

Contents

Exercício 1 — Aquecimento com `select`, `filter`, `arrange`, `mutate`

1. A partir de `mtcars`, crie uma tibble com coluna `modelo` (vinda do rowname) e selecione apenas `modelo`, `mpg`, `cyl`, `hp`.

```
mtcars |>
  as_tibble(rownames= "modelo") |>
  select(modelo,mpg,cyl,hp) |>
  arrange(desc(modelo))
```

```
## # A tibble: 32 x 4
##    modelo             mpg   cyl    hp
##    <chr>            <dbl> <dbl> <dbl>
##  1 Volvo 142E        21.4     4   109
##  2 Valiant           18.1     6   105
##  3 Toyota Corona     21.5     4    97
##  4 Toyota Corolla    33.9     4    65
##  5 Porsche 914-2     26       4    91
##  6 Pontiac Firebird  19.2     8   175
##  7 Merc 450SLC       15.2     8   180
##  8 Merc 450SL        17.3     8   180
##  9 Merc 450SE        16.4     8   180
## 10 Merc 280C         17.8     6   123
## # i 22 more rows
```

2. Filtre carros com `mpg > 25` e `cyl %in% c(4, 6)`.

```
mtcars |>
  filter(mpg>25,cyl %in% c(4,6))|>
  arrange(desc(cyl))
```

```
##                 mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Fiat X1-9      27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
```

3. Crie `kmpl = mpg * 0.4251` e uma categoria de potência:

   - `hp < 100` → "baixa", `100 hp < 150` → "média", `150` → "alta".

```
mtcars |>
  as_tibble(rownames = "modelo") |>
  mutate(kmpl = mpg * 0.4251,
         potencia_cat = case_when(
           hp < 100 ~ "baixa",
           hp < 150 ~ "média",
           TRUE     ~ "alta"
         )) |>
  select(modelo, mpg, kmpl, hp, potencia_cat)
```

```
## # A tibble: 32 x 5
##    modelo              mpg  kmpl    hp potencia_cat
##    <chr>             <dbl> <dbl> <dbl> <chr>
##  1 Mazda RX4          21    8.93   110 média
##  2 Mazda RX4 Wag      21    8.93   110 média
##  3 Datsun 710         22.8  9.69    93 baixa
##  4 Hornet 4 Drive     21.4  9.10   110 média
##  5 Hornet Sportabout  18.7  7.95   175 alta
##  6 Valiant            18.1  7.69   105 média
##  7 Duster 360         14.3  6.08   245 alta
##  8 Merc 240D          24.4 10.4     62 baixa
##  9 Merc 230           22.8  9.69    95 baixa
## 10 Merc 280           19.2  8.16   123 média
## # i 22 more rows
```

4. Ordene por `desc(kmpl)` e mostre as 8 primeiras linhas.

```
mtcars |>
  as_tibble(rownames = "modelo") |>
  mutate(kmpl = mpg * 0.4251,
         potencia_cat = case_when(
           hp < 100 ~ "baixa",
           hp < 150 ~ "média",
           TRUE     ~ "alta"
         )) |>
  select(modelo, mpg, kmpl, hp, potencia_cat) |>
  arrange(desc(kmpl))
```

```
## # A tibble: 32 x 5
##    modelo           mpg  kmpl    hp potencia_cat
##    <chr>          <dbl> <dbl> <dbl> <chr>
##  1 Toyota Corolla  33.9 14.4     65 baixa
##  2 Fiat 128        32.4 13.8     66 baixa
##  3 Honda Civic     30.4 12.9     52 baixa
##  4 Lotus Europa    30.4 12.9    113 média
##  5 Fiat X1-9       27.3 11.6     66 baixa
##  6 Porsche 914-2   26   11.1     91 baixa
##  7 Merc 240D       24.4 10.4     62 baixa
##  8 Datsun 710      22.8  9.69    93 baixa
##  9 Merc 230        22.8  9.69    95 baixa
## 10 Toyota Corona   21.5  9.14    97 baixa
## # i 22 more rows
```

Exercício 2 — `summarise` + `group_by` + `across`

Com o dataset `nycflights13::flights`:

1. Calcule, por `carrier`, o número de voos (`n()`), o atraso médio de partida (`dep_delay`) e de chegada (`arr_delay`) ignorando `NA`.

```
flights <- nycflights13::flights
flights |>
  group_by("carrier")
```

```
## # A tibble: 336,776 x 20
## # Groups:   "carrier" [1]
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 336,766 more rows
## # i 12 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>, `"carrier"` <chr>
```

2. Use `across()` para obter **média** e **desvio-padrão** de `dep_delay` e `arr_delay` por `carrier`.

```
resumo<-
  flights |>
  group_by(carrier) |>
  summarize(
    across(
      c(dep_delay,arr_delay),
      list(media = ~mean(.x, na.rm=TRUE),
           dP = ~sd(.x, na.rm = TRUE))
      ),
    n=n(),
    .groups = "drop"
    )
resumo
```

3. Ordene do maior para o menor número de voos e mostre o top-10.

```
top10 <-
  flights |>
  group_by(carrier) |>
  summarize(n_voos=n()) |>
  arrange(desc(n_voos)) |> head(10)

top10
```

```
## # A tibble: 10 x 2
##    carrier n_voos
##    <chr>    <int>
##  1 UA       58665
##  2 B6       54635
##  3 EV       54173
##  4 DL       48110
##  5 AA       32729
##  6 MQ       26397
##  7 US       20536
##  8 9E       18460
##  9 WN       12275
## 10 VX        5162
```

Exercício 3 — `pivot_longer` / `pivot_wider` e `separate` / `unite`

1. Crie a tibble:

```
vendas <- tibble(
 id = 1:4,
 Jan = c(10, 12, 9, 11),
 Fev = c(11, 8, 13, 10),
 Mar = c(9, 12, 7, 15),
 Abr = c(12, 4, 13 ,8),
 Mai = c(4,3,2,1),
```

```
  Jun= c(12,15,8,7)
   )
   vendas
```

```
## # A tibble: 4 x 7
##      id   Jan   Fev   Mar   Abr   Mai   Jun
##   <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     1    10    11     9    12     4    12
## 2     2    12     8    12     4     3    15
## 3     3     9    13     7    13     2     8
## 4     4    11    10    15     8     1     7
```

2. Converta de **largo** → **longo** (`mes`, `valor`) com `pivot_longer`.

```
vendas_long <- vendas |> pivot_longer(
  cols = Jan:Jun,
  names_to = "mes",
  values_to = "valor"
)

vendas_long
```

```
## # A tibble: 24 x 3
##       id mes   valor
##    <int> <chr> <dbl>
##  1     1 Jan      10
##  2     1 Fev      11
##  3     1 Mar       9
##  4     1 Abr      12
##  5     1 Mai       4
##  6     1 Jun      12
##  7     2 Jan      12
##  8     2 Fev       8
##  9     2 Mar      12
## 10     2 Abr       4
## # i 14 more rows
```

3. Volte de **longo** → **largo** com `pivot_wider`.

```
vendas_largo <- vendas_long |>
  pivot_wider(
    names_from = mes,
    values_from = valor
  )
vendas_largo
```

```
## # A tibble: 4 x 7
##      id   Jan   Fev   Mar   Abr   Mai   Jun
##   <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     1    10    11     9    12     4    12
## 2     2    12     8    12     4     3    15
## 3     3     9    13     7    13     2     8
## 4     4    11    10    15     8     1     7
```

4. Crie `nomes <- tibble(nome_completo = c("Ana Silva","Bruno Costa","C. Rocha"))`. Use `separate` para `primeiro, apelido` e depois `unite` para `nome_compacto = "primeiro_apelido"`.

```
nomes <- tibble(nome_completo = c("Ana Silva","Bruno Costa","C. Rocha"))

separados <- nomes |> separate(nome_completo, into=c("primeiro","apelido"),sep= " ")
separados
```

```
## # A tibble: 3 x 2
##   primeiro apelido
##   <chr>    <chr>
## 1 Ana      Silva
## 2 Bruno    Costa
## 3 C.       Rocha
```

```
juntos <- separados |> unite("nome_compacto",apelido,primeiro,sep="/")
juntos
```

```
## # A tibble: 3 x 1
##   nome_compacto
##   <chr>
## 1 Silva/Ana
## 2 Costa/Bruno
## 3 Rocha/C.
```

Exercício 4 — Junções I (left/inner/full)

1. Com `flights`, obtenha uma tabela de contagem por destino: `voos_dest <- flights |> count(dest, name = "n_voos")`.

```
voos_dest <- flights |> count(dest, name = "n_voos")
```

2. Faça `left_join` com `airports` (chave: `dest  faa`) para obter o nome e coordenadas dos aeroportos.
3. Faça `inner_join` e compare o número de linhas com o `left_join`.
4. Use `full_join` para verificar quantos destinos não têm metadados em `airports` e quantos aeroportos não aparecem em `flights`.