Atividade 3. Manipulação de dados no R

Pedro H. G. Ferreira de Souza

2022-11-10

Instruções

Para responder aos exercícios abaixo, simplesmente substituam as reticências (...) pelas funções e expressões adequadas e, de preferência, adicionem comentários na mesma linha, após o símbolo "#", explicando o objetivo de cada linha.

Se não souberem responder a algum tópico, por favor acrescentem um comentário explicando sua dúvida.

Pontuação

- Parte 1 → até 10 pontos
- Parte 2 → até 50 pontos
- Parte 3 → até 40 pontos

Parte 1. Preparação

1. Para iniciar, limparemos o workspace, apagando os objetos carregados

Exercício:

```
rm(list = ...)
```

Resposta:

```
rm(list = ls())
```

2. Crie um vetor com a forma abaixo para identificar seu nome e email

Exercício:

```
aluno <- c(... , ...)   #
names(aluno) <- c('nome', 'email')
print(aluno)
```

Resposta:

```
aluno <- c("Pedro Souza", "pedro.ferreira@ipea.gov.br")
names(aluno) <- c('nome', 'email')
print(aluno)
```

```
##                         nome                        email
##               "Pedro Souza" "pedro.ferreira@ipea.gov.br"
```

3. Instale (se necessário) e carregue os pacotes "tidyverse" e "gapminder"

Para facilitar, carrego também o `summarytools`. Exercício:

```
#install.packages(c('tidyverse', 'gapminder'))
library(...)    #
library(...)    #
```

Resposta:

```
#install.packages(c('tidyverse', 'gapminder'))
library(tidyverse)
library(gapminder)
library(summarytools)
```

Parte 2. Manipulação de dados do Gapminder

1. Crie o data frame abaixo a partir do pacote gapminder

```
gapminder.df <- gapminder
```

2. Verifique as características de "gapminder.df" com str, class e dim

Exercício:

```
...(...)      #
class(...)    #
...(...)      #
```

Resposta:

```
str(gapminder.df)
class(gapminder.df)
dim(gapminder.df)
```

```
## tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
##  $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ year     : int [1:1704] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
##  $ lifeExp  : num [1:1704] 28.8 30.3 32 34 36.1 ...
##  $ pop      : int [1:1704] 8425333 9240934 10267083 11537966 13079460 14880372 12881816 13867957 163
##  $ gdpPercap: num [1:1704] 779 821 853 836 740 ...
## [1] "tbl_df"     "tbl"        "data.frame"
## [1] 1704    6
```

3. Imprima no console as 15 primeiras linhas do data frame

Exercício:

```
...(..., ...)
```

Resposta:

```
head(gapminder.df, n = 15)
```

```
## # A tibble: 15 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## 11 Afghanistan Asia       2002    42.1 25268405      727.
## 12 Afghanistan Asia       2007    43.8 31889923      975.
## 13 Albania     Europe     1952    55.2  1282697     1601.
## 14 Albania     Europe     1957    59.3  1476505     1942.
## 15 Albania     Europe     1962    64.8  1728137     2313.
```

4. Adicione uma coluna ao gapminder.df com o PIB total em bilhões de dólares

Ou seja, multiplique o PIB per capita (`gdpPercap`) pela população (`pop`) e divida o resultado por 1 bilhão.

Exercício:

```
gapminder.df <-
  gapminder.df %>%
    mutate(gdpTotal = ...)
```

Resposta:

```
gapminder.df <-
  gapminder.df %>%
  mutate(gdpTotal = pop * gdpPercap / 10^9)
descr(gapminder.df$gdpTotal)
```

```
## Descriptive Statistics
## gapminder.df$gdpTotal
## N: 1704
##
##                     gdpTotal
## ----------------- ----------
##              Mean     186.81
##           Std.Dev     714.03
##               Min       0.05
##                Q1       5.89
##            Median      22.34
##                Q3     105.79
```

```
##                Max   12934.46
##                MAD      29.92
##                IQR      99.85
##                 CV       3.82
##           Skewness      10.23
##        SE.Skewness       0.06
##           Kurtosis     136.12
##            N.Valid    1704.00
##          Pct.Valid     100.00
```

5. Use summarise() e group_by() para ver a soma do PIB total por continente em 2007

```
gapminder.df %>%
  filter(year == 2007) %>%
    group_by(continent) %>%
      summarise(pib_continental = sum(gdpTotal))
```

```
## # A tibble: 5 x 2
##   continent pib_continental
##   <fct>               <dbl>
## 1 Africa              2380.
## 2 Americas           19418.
## 3 Asia               20708.
## 4 Europe             14795.
## 5 Oceania              807.
```

6. Crie um objeto continentes.df, que será um data frame com uma linha por continente e colunas para anos, população total (em milhões) e PIB total (em bilhões)

*Dica: use o `group_by` por continente, não use o comando `filter`.

Exercício:

```
continentes.df <-
  ... %>%     #
    group_by(...) %>%
      summarise(popTotal = ...,   #
                gdpTotal = ...)   #
```

Resposta:

```
continentes.df <-
  gapminder.df %>%
    group_by(continent, year) %>%
      summarise(popTotal = sum(pop / 10^6),
                gdpTotal = sum(pop * gdpPercap / 10^9))
glimpse(continentes.df)
head(continentes.df)
```

```
## Rows: 60
## Columns: 4
## Groups: continent [5]
## $ continent <fct> Africa, Africa, Africa, Africa, Africa, Africa, Africa, Afri~
## $ year      <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 1997, ~
## $ popTotal  <dbl> 237.6405, 264.8377, 296.5169, 335.2895, 379.8795, 433.0610, ~
```

```
## $ gdpTotal  <dbl> 311.5993, 382.6778, 456.8136, 595.0877, 783.7566, 972.1347, ~
## # A tibble: 6 x 4
## # Groups:   continent [1]
##   continent  year popTotal gdpTotal
##   <fct>     <int>    <dbl>    <dbl>
## 1 Africa     1952     238.     312.
## 2 Africa     1957     265.     383.
## 3 Africa     1962     297.     457.
## 4 Africa     1967     335.     595.
## 5 Africa     1972     380.     784.
## 6 Africa     1977     433.     972.
```

7. Crie uma variável em continentes.df com o PIB per capita continental

Dica: não se esqueça de converter população e PIB total de volta para as unidades originais.

Exercício:

```
continentes.df <-
  ... %>%     #
    ...(gdpPercap = ... / ...)    #
```

Resposta:

```
continentes.df <-
  continentes.df %>%
    mutate(gdpPercap = (gdpTotal * 10^9) / (popTotal * 10^6))
descr(continentes.df$gdpPercap)
```

```
## Descriptive Statistics
## continentes.df$gdpPercap
## N: 60
##
##                      gdpPercap
## ----------------- -----------
##              Mean    10692.94
##           Std.Dev     8470.51
##               Min      806.36
##                Q1     2228.18
##            Median     9948.18
##                Q3    17068.57
##               Max    32884.56
##               MAD    11383.67
##               IQR    14624.59
##                CV        0.79
##          Skewness        0.54
##       SE.Skewness        0.31
##          Kurtosis       -0.73
##           N.Valid       60.00
##         Pct.Valid      100.00
```

8. Crie uma variável em continentes.df do tipo LOGICAL, com valor TRUE se o PIB per capita em um dado ano for o maior da série para cada continente e FALSE caso não seja

Dica: use `group_by` *e* `mutate` *com a função* `max`*.*

Exercício:

```
continentes.df <-
  continentes.df %>%
    ...(continent) %>%    #
      mutate(maior_gdpPercap = ...)   #
```

Resposta:

```
continentes.df <-
  continentes.df %>%
    group_by(continent) %>%
      mutate(maior_gdpPercap = gdpPercap == max(gdpPercap))
freq(continentes.df$maior_gdpPercap)
```

```
## Frequencies
## continentes.df$maior_gdpPercap
## Type: Logical
##
##               Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
## ----------- ------ --------- -------------- --------- --------------
##       FALSE     55     91.67          91.67     91.67          91.67
##        TRUE      5      8.33         100.00      8.33         100.00
##        <NA>      0                                0.00         100.00
##       Total     60    100.00         100.00    100.00         100.00
```

9. Repita o exercício anterior, mas agora crie uma variável LOGICAL para o menor PIB per capita da série

Exercício:

```
continentes.df <-
  ... %>%   #
    ...(...) %>%    #
      ...(menor_gdpPercap = ...)    #
```

Resposta:

```
continentes.df <-
  continentes.df %>%
    group_by(continent) %>%
      mutate(menor_gdpPercap = gdpPercap == min(gdpPercap))
freq(continentes.df$menor_gdpPercap)
```

```
## Frequencies
## continentes.df$menor_gdpPercap
## Type: Logical
##
##               Freq   % Valid   % Valid Cum.   % Total   % Total Cum.
## ----------- ------ --------- -------------- --------- --------------
##       FALSE     55     91.67          91.67     91.67          91.67
##        TRUE      5      8.33         100.00      8.33         100.00
##        <NA>      0                                0.00         100.00
##       Total     60    100.00         100.00    100.00         100.00
```

10. Crie um novo data frame incluindo somente as linhas de continentes.df com o maior ou menor PIB per capita da série

O data frame deve ter apenas três colunas: continent, year e gdpPercap.

Dica: use a função `filter` e as variáveis que criamos acima, e depois a função `select`.

Exercício:

```
... <-      #
  ... %>%   #
    filter(...) %>%   #
      ...(..., ..., ...)    #
```

Resposta:

```
resumo.df <-
  continentes.df %>%
    filter(maior_gdpPercap == TRUE | menor_gdpPercap == TRUE) %>%
      select(continent, year, gdpPercap)
glimpse(resumo.df)
```

```
## Rows: 10
## Columns: 3
## Groups: continent [5]
## $ continent <fct> Africa, Africa, Americas, Americas, Asia, Asia, Europe, Euro~
## $ year      <int> 1952, 2007, 1952, 2007, 1952, 2007, 1952, 2007, 1952, 2007
## $ gdpPercap <dbl> 1311.2214, 2560.9296, 8528.0431, 21602.7458, 806.3599, 5432.~
```

11. Imprima o objeto no console

Como se pode ver, em todos os casos o PIB per capita mínimo foi em 1952 e o máximo, em 2007.

Exercício:

```
print(...)    #
```

Resposta:

```
print(resumo.df)
```

```
## # A tibble: 10 x 3
## # Groups:   continent [5]
##    continent  year gdpPercap
##    <fct>     <int>     <dbl>
##  1 Africa     1952     1311.
##  2 Africa     2007     2561.
##  3 Americas   1952     8528.
##  4 Americas   2007    21603.
##  5 Asia       1952      806.
##  6 Asia       2007     5432.
##  7 Europe     1952     6097.
##  8 Europe     2007    25244.
##  9 Oceania    1952    10136.
## 10 Oceania    2007    32885.
```

12. Qual continente teve a maior taxa de crescimento?

Para saber, vamos criar novo data frame com cinco linhas (uma por continente) e apenas duas colunas: continente e taxa de crescimento.

Como calcular essa taxa? É fácil. Seja `X = (renda final / renda inicial)` e `Y = (ano final - ano inicial)`, a fórmula é `TX = X^(1/Y)`. Para expressá-la em porcentagem é só fazer `TX = (TX - 1)100`.

Dica: use as funções `min` e `max` no `mutate` para obter valores iniciais e finais.

Exercício:

```
taxascresc.df <-
  ... %>%     #
    group_by(...) %>%   #
      ...(renda_final_sobre_inicial = ... / ...),     #
             ano_final_menos_inicial = ... - ...,     #
             taxacresc = ...,     #
             taxacresc_pct = (... - 1)*100) %>%       #
      summarise(taxacresc_pct = ...)      #
```

Resposta:

```
taxascresc.df <-
  resumo.df %>%
    group_by(continent) %>%
      mutate(renda_final_sobre_inicial = max(gdpPercap) / min(gdpPercap),
             ano_final_menos_inicial = max(year) - min(year),
             taxacresc = renda_final_sobre_inicial^(1/(ano_final_menos_inicial)),
             taxacresc_pct = (taxacresc - 1)*100) %>%
      summarise( taxacresc_pct = mean(taxacresc_pct))
```

13. Vamos ordenar o último objeto em ordem decrescente de crescimento

Dica: use o `arrange`.

Exercício:

```
taxascresc.df <- taxascresc.df %>% ...    #
```

Resposta:

```
taxascresc.df <- taxascresc.df %>% arrange(desc(taxacresc_pct))
```

14. Imprima o objeto no console.

Exercício:

```
...   #
```

Resposta:

```
print(taxascresc.df)
```

```
## # A tibble: 5 x 2
##   continent taxacresc_pct
##   <fct>             <dbl>
## 1 Asia               3.53
## 2 Europe             2.62
```

```
## 3 Oceania            2.16
## 4 Americas           1.70
## 5 Africa             1.22
```

15. Qual continente cresceu mais rápido?

Preencha o objeto abaixo com a resposta.

Exercício:

```
quem_cresceu_mais_rapido <- as.character(...$continent[...])   #
```

Resposta:

```
quem_cresceu_mais_rapido <- as.character(taxascresc.df$continent[1])
print(quem_cresceu_mais_rapido)
```

```
## [1] "Asia"
```

Parte 3. Manipulação de dados dos estados americanos

1. O R vem com dados pré-instalados

```
data(state)
```

Observe que o comando acima criou vários objetos: cinco vetores separados e uma matriz.

2. Transformando a matriz e um dos vetores em um data frame

```
states.df <- data.frame(state.x77)
states.df <- states.df %>% cbind(region = state.region)
```

O data frame states.df tem informações sobre os 50 estados americanos nos anos 1970.

3. Imprime as últimas 12 linhas do data frame

```
tail(states.df, n = 12)
```

```
##                Population Income Illiteracy Life.Exp Murder HS.Grad Frost
## Rhode Island          931   4558        1.3    71.90    2.4    46.4   127
## South Carolina       2816   3635        2.3    67.96   11.6    37.8    65
## South Dakota          681   4167        0.5    72.08    1.7    53.3   172
## Tennessee            4173   3821        1.7    70.11   11.0    41.8    70
## Texas               12237   4188        2.2    70.90   12.2    47.4    35
## Utah                 1203   4022        0.6    72.90    4.5    67.3   137
## Vermont               472   3907        0.6    71.64    5.5    57.1   168
## Virginia             4981   4701        1.4    70.08    9.5    47.8    85
## Washington           3559   4864        0.6    71.72    4.3    63.5    32
## West Virginia        1799   3617        1.4    69.48    6.7    41.6   100
## Wisconsin            4589   4468        0.7    72.48    3.0    54.5   149
## Wyoming               376   4566        0.6    70.29    6.9    62.9   173
##                  Area        region
## Rhode Island     1049     Northeast
```

```
## South Carolina  30225         South
## South Dakota    75955 North Central
## Tennessee       41328         South
## Texas          262134         South
## Utah            82096          West
## Vermont          9267     Northeast
## Virginia        39780         South
## Washington      66570          West
## West Virginia   24070         South
## Wisconsin       54464 North Central
## Wyoming         97203          West
```

4. Verifique as características do data frame

```
glimpse(states.df)
class(states.df)
dim(states.df)
```

```
## Rows: 50
## Columns: 9
## $ Population <dbl> 3615, 365, 2212, 2110, 21198, 2541, 3100, 579, 8277, 4931, ~
## $ Income     <dbl> 3624, 6315, 4530, 3378, 5114, 4884, 5348, 4809, 4815, 4091,~
## $ Illiteracy <dbl> 2.1, 1.5, 1.8, 1.9, 1.1, 0.7, 1.1, 0.9, 1.3, 2.0, 1.9, 0.6,~
## $ Life.Exp   <dbl> 69.05, 69.31, 70.55, 70.66, 71.71, 72.06, 72.48, 70.06, 70.~
## $ Murder     <dbl> 15.1, 11.3, 7.8, 10.1, 10.3, 6.8, 3.1, 6.2, 10.7, 13.9, 6.2~
## $ HS.Grad    <dbl> 41.3, 66.7, 58.1, 39.9, 62.6, 63.9, 56.0, 54.6, 52.6, 40.6,~
## $ Frost      <dbl> 20, 152, 15, 65, 20, 166, 139, 103, 11, 60, 0, 126, 127, 12~
## $ Area       <dbl> 50708, 566432, 113417, 51945, 156361, 103766, 4862, 1982, 5~
## $ region     <fct> South, West, West, South, West, West, Northeast, South, Sou~
## [1] "data.frame"
## [1] 50  9
```

5. Vamos criar um banco de dados agregados por região, com quatro linhas e quatro colunas

As colunas serão:

- ***pct_da_pop_total**: percentual da população americana em cada região*
- ***densidade**: população total da região dividida pela área (em km2)*
- ***expvida**: expectativa de vida média em cada região (em anos)*
- ***estados**: número de estados em cada região*

Não é tão simples quanto parece. Observe que:

- *A variável **Area** está em milhas quadradas e, portanto, precisa ser multiplicada para 2.59 para ser convertida em km2.*
- *A variável **Population** está em milhares e, portanto, precisa ser multiplicada por mil.*
- *É preciso calcular a média ponderada da expectativa de vida por regiões, pois as populações dos estados são diferentes. Para isso, podemos usar a função **weighted;.mean**.*
- *Para obter o número de linhas por região, podemos usar a função **n**.*

*É possível fazer o procedimento em um único comando, encadeado instruções por meio do **pipe**. Observe que, no final, vamos ordenar o banco pelo número de estados. Preencha as lacunas abaixo:*

Exercício:

```
regioes.df <-
    ... %>%
      mutate(pct_da_pop_total = Population / ...) %>%
        group_by(...) %>%
          summarise(pct_da_pop_total = ...,
                    densidade = ...,
                    expvida = ...,
                    estados = ...) %>%
            arrange(...)
```

Resposta:

```
regioes.df <-
    states.df %>%
      mutate(pct_da_pop_total = Population / sum(Population)) %>%
        group_by(region) %>%
          summarise(pct_da_pop_total = sum(pct_da_pop_total),
                    densidade = sum(Population * 1000) / sum(Area * 2.59),
                    expvida = weighted.mean(Life.Exp, Population),
                    estados = n()) %>%
            arrange(estados)
print(regioes.df)
```

```
## # A tibble: 4 x 5
##   region        pct_da_pop_total densidade expvida estados
##   <fct>                    <dbl>     <dbl>   <dbl>   <int>
## 1 Northeast                0.233    117.      70.9       9
## 2 North Central            0.271     29.6     71.2      12
## 3 West                     0.178      8.37    71.6      13
## 4 South                    0.317     29.8     69.9      16
```

6. Responda preenchendo as lacunas abaixo:

Exercício:

```
regiao_com_menor_densidade <- filter(..., densidade == ...)$region[1]
print(regiao_com_menor_densidade)
```

Resposta:

```
regiao_com_menor_densidade <- filter(regioes.df, densidade == min(densidade))$region[1]
print(regiao_com_menor_densidade)
```

```
## [1] West
## Levels: Northeast South North Central West
```