Exercícios Aula 05

Pílula 1

1. Vamos comparar o `dplyr::mutate()` com o `dplyr::summarise()`.

A) Inspecione a base de dados `da` (abaixo) com o `dplyr::glimpse()`. Quantas colunas ela possui? E quantas linhas? Qual é o tipo de cada coluna?

B) **Apenas olhe** o seguinte código. Utilizamos a função do `baseR` chamada `sqrt()` (*square root*), que serve para fazer a raiz quadrada de uma variável numérica. Sabendo disso, o que você **espera** que essa função faça no `mutate()`? Haverá exclusão de coluna? E modificação? E adição? E no `summarise()`? Neste caso, haverá exclusão de coluna? E modificação? E adição?

C) Agora rode o código. O que aconteceu em cada caso? Isso foi de acordo com a sua expectativa?

D) Faz sentido usar o `sqrt()` junto do `summarise()`?

```
set.seed(3)
da <- tibble::tibble(
  a = sample(letters, 1000, replace=T),
  b = sample(20:80, 1000, replace=T)
)

da |>
  dplyr::mutate(
    b2 = sqrt(b)
  )

da |>
  dplyr::summarise(
    b2 = sqrt(b)
  )
```

2. Vamos agora olhar para os comparadores do `baseR` e entender sua aplicação para o `mutate()` e para o `summarise()`.

A) O que faz o comparador `==` ?

```
set.seed(3)
vetor <- sample(letters, 100, replace=T)

vetor == "a"
```

B) O que faz a função `all()` ?

```
set.seed(3)
vetor <- sample(letters, 100, replace=T)

all(vetor == "a")
```

C) O que faz a função `any()` ?

```
set.seed(3)
vetor <- sample(letters, 100, replace=T)

any(vetor == "a")
```

D) Qual das três opções acima você acha mais apropriado utilizar no `summarise()`? E no `mutate()`? Crie exemplos utilizando a base `cursoESMP::dados`.

3. É possível usar o `group_by()` junto de `filter()` também.

A) Qual é a diferença entre o Código A e o Código B abaixo?

```
set.seed(3)
base <- tibble::tibble(
  letras = sample(letters, 100, replace=T),
  valores = sample(30:80, 100, replace=T)
)

# Código A
base |>
  dplyr::filter(valores >= 60)

# Código B
base |>
```

```
  dplyr::group_by(letras) |>
  dplyr::filter(valores >= 60) |>
  dplyr::ungroup()
```

B) E agora? Qual é a diferença entre o Código A e o Código B abaixo?

```
set.seed(3)
base <- tibble::tibble(
  letras = sample(letters, 100, replace=T),
  valores = sample(30:80, 100, replace=T)
)

# Código A
base |>
  dplyr::filter(all(valores >= 60))

# Código B
base |>
  dplyr::group_by(letras) |>
  dplyr::filter(all(valores >= 60)) |>
  dplyr::ungroup()
```

C) E o que acontece se eu usar o `dplyr::group_by`, mas sem usar o `all()` dentro do `dplyr::filter()` ? Existe diferença entre o Código C e o Código D?

```
set.seed(3)
base <- tibble::tibble(
  letras = sample(letters, 100, replace=T),
  valores = sample(30:80, 100, replace=T)
)

# Código C
base |>
  dplyr::group_by(letras) |>
  dplyr::filter(valores >= 60) |>
  dplyr::ungroup()

# Código D
base |>
  dplyr::filter(valores >= 60)
```

Pílula 2

1. Identifique todos os textos que se referem ao Marco Civil da Internet (MCI). Dicas: (a) crie um vetor único contendo todos os textos; (b) crie uma regex em separado; (c) crie um pipeline, iniciado pelo vetor criado em (a) e que utilize uma ou mais funções do `stringr` para detectar quais textos mencionam o MCI.

```
txt1 <- "Este texto trata do MCI"
txt2 <- "Vamos olhar para a LEI Nº 12.965, DE 23 DE ABRIL DE 2014"
txt3 <- "É um grande marco da internet a promulgação da LGPD"
txt4 <- "Quando falamos de internet, é importante olharmos para a lei n. 12.965/2014 e não pa
txt5 <- "Comparemos a Lei nº 12965/2014 com a Lei Nº 13.709/2018"
txt6 <- "O MarcO CIVil da InterNET é importante"
```

2. Ajuste o código criado no exercício anterior para identificar os textos que se referem APENAS ao marco civil da internet

3. A partir da base `cursoESMP::movimentacao`, crie uma base contendo apenas as movimentações de sentença, chamada `movimentacoes_sentenca`. Para isso, crie a coluna `eh_sentenca`, para identificar cada movimentação que se refere a uma sentença.

DICA 1: Faça essa análise usando apenas a coluna `movimento`.

DICA 2: Se vocês querem ver todas as possibilidades de `movimento`, usem a função `dplyr::distinct(base, coluna)` e atribuam para um objeto, para que vocês possam visualizá-lo.

4. A partir da base de movimentações de sentenças criada no exercício anterior, determine o resultado de cada sentença. Há 4 resultados possíveis: a) extinto sem julgamento de mérito, b) favorável, c) parcialmente favorável, d) desfavorável. Crie a coluna `sentenca_resultado` (variável do tipo `character`, indicando se o resultado foi "Extinto sem mérito", "Favorável", "Desfavorável" ou "Parcialmente favorável").

DICA 3: Para a variável `sentenca_resultado`, existe uma função muito útil chamada `dplyr::case_when()`. Vocês não precisam usar, caso achem difícil a sintaxe dessa função. Abaixo segue um exemplo explicando.

```
tibble::tibble(
  id = letters[1:10],
  numeros = 1:10
) |>
  dplyr::mutate(
    tipo_numero = dplyr::case_when(
      numeros %% 2 == 0 ~ "par",
```

```
      numeros %% 2 == 1 ~ "impar",
    )
  )
```

Algumas notas sobre essa função;

- Ela é usada DENTRO do `dplyr::mutate()`.
- Como argumentos ela recebe (1) uma condição lógica e (2) o que acontece se essa condição for verdadeira
- A sintaxe é "(1) ~ (2)", ou seja **"[condição lógica] ~ 'efeito se a condição for verdadeira'"**
- Vários pares de 1-2 (condição-efeito) podem ser enumerados, separados por uma vírgula ao final. O último par não pode ter ","

EXERCÍCIO DESAFIO (Dificuldade: Altíssima). A partir da base criada no exercício anterior, crie uma nova base, com uma unidade amostral diferente. Ao invés de ser uma base cuja unidade observacional seja processo-movimento, queremos criar uma base em que cada linha seja apenas um processo. Essa nova base terá apenas 3 colunas: (1) `processo` (variável do tipo `character`, contendo o número CNJ de 20 dígitos), (2) `houve_sentenca` (variável do tipo `logical`, indicando se para aquele processo houve ou não sentença) e (3) `sentenca_resultado` (variável do tipo `character` indicando um dos 4 resultados possíveis, para os casos em que houve sentença).

DICA 1: Você vai precisar criar uma base intermediária.

DICA 2: Existem processos que vão ter mais de uma sentença. E cada sentença pode ter uma resposta diferente para a variável `sentenca_resultado` ! Então você vai precisar tomar uma decisão: você quer SEMPRE a primeria decisão ou quer SEMPRE a última decisão ?

Pílula 3

1. Arrume a coluna `distribuicao` da base de dados `cursoESMP::dados`, transformando-a de `character` para `date`, no formato YYYY-MM-dd (excluindo, portanto, horas e minutos).