Dataframes

Marcella Pedro

01/11/2021

Dataframes

Dataframe é uma estrutura de dados conhecida como tabular, pois é organizada em linhas e colunas que lembra muito a estrutura de um excel por exemplo.

Dados tabulares podem ser armazenados de muitas formas, **xlsx**, extensão utilizada pelo Excel conforme já mencionado, ou Google Sheets. Também podemos usar a extensão **csv** *(Comma-separated Values)* que como o nome indica é separado por vírgulas, também podemos separar por ponto e vírgula, pipeline, espaçamentos definidos e etc.

No fim conseguimos visualizar todas as estruturas acima em um formato de tabela, onde as colunas armazenam uma informação em comum com um rótulo e cada linha possui um índice geralmente começado em 1.

No `tidyverse` temos um tipo especial de *dataframe* chamado *tibble*.

Importar Bibliotecas para manipular dataframes

`readr`

Para ler um arquivo csv, também muito conhecido como *flat file data*, e armazenar em um formato tabular, em outras palavras em um dataframe, usamos a função `read_csv` do pacote `readr`

Como resultado temos um tibble.

```
library(readr)
```

Lendo dataset separado por **vírgulas**.

```
df_virg <- read_csv("~/Documentos/Dataquest/dados_virgula.csv")
```

```
## Rows: 3 Columns: 4

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): nome, genero, profissao
## dbl (1): idade

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_virg
```

```
## # A tibble: 3 x 4
##   nome        idade genero    profissao
##   <chr>       <dbl> <chr>     <chr>
## 1 marcella       26 feminino  analista de dados
## 2 rafael         28 masculino analista de sistemas
## 3 ana vitoria    24 feminino  analista contabil
```

Lendo dataset separado por **ponto e vírgulas**.

```
df_pont_virg <- read_delim("~/Documentos/Dataquest/dados_ponto_virgula.csv", delim=";")
```

```
## Rows: 3 Columns: 4
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (3): nome, genero, profissao
## dbl (1): idade
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_pont_virg
```

```
## # A tibble: 3 x 4
##   nome        idade genero    profissao
##   <chr>       <dbl> <chr>     <chr>
## 1 marcella       26 feminino  analista de dados
## 2 rafael         28 masculino analista de sistemas
## 3 ana vitoria    24 feminino  analista contabil
```

Lendo dataset separado por **pipeline**.

```
df_pipe <- read_delim("~/Documentos/Dataquest/dados_pipeline.csv", delim="|")
```

```
## Rows: 3 Columns: 4
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "|"
## chr (3): nome, genero, profissao
## dbl (1): idade
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_pipe
```

```
## # A tibble: 3 x 4
##   nome        idade genero    profissao
##   <chr>       <dbl> <chr>     <chr>
## 1 marcella       26 feminino  analista de dados
## 2 rafael         28 masculino analista de sistemas
## 3 ana vitoria    24 feminino  analista contabil
```

Lendo dataset separado por **espaços**.

```
df_space <- read_delim("~/Documentos/Dataquest/dados_espaco.csv", delim="\t")
```

```
## Rows: 3 Columns: 4
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (3): nome, genero, profissao
## dbl (1): idade
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_space
```

```
## # A tibble: 3 x 4
##   nome        idade genero    profissao
##   <chr>       <dbl> <chr>     <chr>
## 1 marcella       26 feminino  analista de dados
## 2 rafael         28 masculino analista de sistemas
## 3 ana vitoria    24 feminino  analista contabil
```

readxl

Biblioteca para leitura de Excel **(xlsx)**

```
library(readxl)
```

```
df_excel <- read_xlsx("~/Documentos/Dataquest/dados_excel.xlsx")
df_excel
```

```
## # A tibble: 3 x 4
##   nome        idade genero    profissao
##   <chr>       <dbl> <chr>     <chr>
## 1 marcella       26 feminino  analista de dados
## 2 rafael         28 masculino analista de sistemas
## 3 ana vitoria    24 feminino  analista contabil
```

Coletar informações de um dataframe

`View()`

Para visualizar um dataframe por completo, basta apenas executar o nome do dataframe, mas esta função abre uma janela dedicada

```
View(df_virg)
```

`head()` ou `tail()`

Para visualizar as primeiras linhas ou últimas.

```
head(df_virg)
```

```
## # A tibble: 3 x 4
##   nome        idade genero    profissao
##   <chr>       <dbl> <chr>     <chr>
## 1 marcella       26 feminino  analista de dados
## 2 rafael         28 masculino analista de sistemas
## 3 ana vitoria    24 feminino  analista contabil
```

`glimpse()`

Para um compilado de informações básicas.

```
library(tibble)
glimpse(df_virg)
```

```
## Rows: 3
## Columns: 4
## $ nome      <chr> "marcella", "rafael", "ana vitoria"
## $ idade     <dbl> 26, 28, 24
## $ genero    <chr> "feminino", "masculino", "feminino"
## $ profissao <chr> "analista de dados", "analista de sistemas", "analista conta~
```

Número de linhas

```
nrow(df_virg)
```

```
## [1] 3
```

Número de colunas

```
ncol(df_virg)
```

```
## [1] 4
```

Nome das colunas

```
names(df_virg)
```

```
## [1] "nome"      "idade"     "genero"    "profissao"
```

Selecionar ou filtrar parte do Dataframe

`select()`

Seleciona uma ou mais colunas.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
select(df_virg, nome)
```

```
## # A tibble: 3 x 1
##   nome
##   <chr>
## 1 marcella
## 2 rafael
## 3 ana vitoria
```

Ou seleciona todas colunas exceto as destacadas.

```
select(df_virg, -nome)
```

```
## # A tibble: 3 x 3
##   idade genero    profissao
##   <dbl> <chr>     <chr>
## 1    26 feminino  analista de dados
## 2    28 masculino analista de sistemas
## 3    24 feminino  analista contabil
```

`filter()`

Filtra o dataframe de acordo com uma condição.

```
filter(df_virg, idade >= 26)
```

```
## # A tibble: 2 x 4
##   nome     idade genero    profissao
##   <chr>    <dbl> <chr>     <chr>
## 1 marcella    26 feminino  analista de dados
## 2 rafael      28 masculino analista de sistemas
```

É possível filtrar várias condições em conjunto.

```
filter(df_virg,
       idade > 24,
       idade < 28)
```

```
## # A tibble: 1 x 4
##   nome     idade genero   profissao
##   <chr>    <dbl> <chr>    <chr>
## 1 marcella    26 feminino analista de dados
```

Pipe operator %>%

O pipe %>% é uma forma de aninhar várias operações em um tibble sem necessidade de ficar criando variáveis temporárias para armazenar os valores até chegar no resultado desejado. Assim o resultado de uma operação já se torna o input da função seguinte.

No exemplo abaixo foi possível em uma única operação filtrar resultados onde a idade é maior que 24 e apenas trazer os nomes.

```
df_final <- df_virg %>%
              filter(idade > 24) %>%
              select(nome)

df_final
```

```
## # A tibble: 2 x 1
##   nome
##   <chr>
## 1 marcella
## 2 rafael
```

Esse operador também funciona com outras estruturas apesar de ser comum em dataframes.

```
c(1,2,3,4) %>% sum()
```

```
## [1] 10
```

Alterar um dataframe

mutate

Para criar novas colunas no dataframe.

```r
df_mutate <- df_virg %>% mutate(
                          Teste = idade * 100
                        )

df_mutate
```

```
## # A tibble: 3 x 5
##   nome        idade genero    profissao            Teste
##   <chr>       <dbl> <chr>     <chr>                <dbl>
## 1 marcella       26 feminino  analista de dados     2600
## 2 rafael         28 masculino analista de sistemas  2800
## 3 ana vitoria    24 feminino  analista contabil     2400
```

Criando coluna diretamente

```r
df_mutate$NovoTeste <- 0
df_mutate
```

```
## # A tibble: 3 x 6
##   nome        idade genero    profissao            Teste NovoTeste
##   <chr>       <dbl> <chr>     <chr>                <dbl>     <dbl>
## 1 marcella       26 feminino  analista de dados     2600         0
## 2 rafael         28 masculino analista de sistemas  2800         0
## 3 ana vitoria    24 feminino  analista contabil     2400         0
```

Alterando coluna existente

```r
df_mutate$Teste <- "lalala"
df_mutate
```

```
## # A tibble: 3 x 6
##   nome        idade genero    profissao            Teste   NovoTeste
##   <chr>       <dbl> <chr>     <chr>                <chr>       <dbl>
## 1 marcella       26 feminino  analista de dados    lalala          0
## 2 rafael         28 masculino analista de sistemas lalala          0
## 3 ana vitoria    24 feminino  analista contabil    lalala          0
```

Alterando coluna existente com filtro

```r
df_mutate$Teste[df_mutate$genero == "feminino"] <- "lalala123"
df_mutate
```

```
## # A tibble: 3 x 6
##   nome        idade genero    profissao            Teste      NovoTeste
##   <chr>       <dbl> <chr>     <chr>                <chr>          <dbl>
## 1 marcella       26 feminino  analista de dados    lalala123          0
## 2 rafael         28 masculino analista de sistemas lalala             0
## 3 ana vitoria    24 feminino  analista contabil    lalala123          0
```

Ordenar um dataframe

arrange

Ordenando em ordem crescente

```
df_mutate %>% arrange(idade)
```

```
## # A tibble: 3 x 6
##   nome        idade genero    profissao            Teste     NovoTeste
##   <chr>       <dbl> <chr>     <chr>                <chr>         <dbl>
## 1 ana vitoria    24 feminino  analista contabil    lalala123         0
## 2 marcella       26 feminino  analista de dados    lalala123         0
## 3 rafael         28 masculino analista de sistemas lalala            0
```

Ordenando em ordem decrescente

```
df_mutate %>% arrange(-idade)
```

```
## # A tibble: 3 x 6
##   nome        idade genero    profissao            Teste     NovoTeste
##   <chr>       <dbl> <chr>     <chr>                <chr>         <dbl>
## 1 rafael         28 masculino analista de sistemas lalala            0
## 2 marcella       26 feminino  analista de dados    lalala123         0
## 3 ana vitoria    24 feminino  analista contabil    lalala123         0
```

Extrair dados descritivos (Agregação)

summarize

Usando apenas o summarize, os dados são reduzidos a uma única linha, então a média, mínimo e máximo são calculados considerando todos os registros.

```
df_mutate %>% summarize(
                media = mean(idade),
                max   = max(idade),
                min   = min(idade)
              )
```

```
## # A tibble: 1 x 3
##   media   max   min
##   <dbl> <dbl> <dbl>
## 1    26    28    24
```

group_by

Já com o group_by o cálculo é feito de acordo com categorias específicas, então aqui vemos a média, mínimo e máximo de idade por gênero.

```
df_mutate %>% group_by(genero) %>%
              summarize(
                media = mean(idade),
                max   = max(idade),
                min   = min(idade)
              )
```

```
## # A tibble: 2 x 4
##   genero    media   max   min
##   <chr>     <dbl> <dbl> <dbl>
## 1 feminino     25    26    24
## 2 masculino    28    28    28
```

Transformar uma coluna em vetor para efetuar cálculos

Primeira forma $

Ao utilizar `dataframe$coluna` convertemos o resultado a um vetor.

```
df_mutate$idade %>% sum
```

```
## [1] 78
```

Segunda forma []

Ao utilizar colchetes duplos convertemos o resultado a um vetor.

```
df_mutate[["idade"]] %>% sum
```

```
## [1] 78
```

Terceira forma `pull`

Com a função pull também temos como resultado um vetor.

```
df_mutate %>% pull(idade) %>% sum
```

```
## [1] 78
```

Salvar um dataframe em um arquivo

Ainda através da biblioteca `readr`, da mesma forma que fizemos a leitura de um arquivo, podemos salvar um arquivo em csv.

```
write_csv(df_mutate, "~/Documentos/exemplo.csv")
```