Introdução à programação com R

Tabelas

curso-r

Tópicos desta aula

- Pacotes
- Caminhos absolutos e relativos
- Importação de bases de dados
- Tabelas: Data.frames

Pacotes

Pacotes

Um pacote no R é um conjunto de funções que visam resolver um problema em específico. O R já vem com alguns pacotes instalados. Geralmente chamamos esses pacotes de *base R*.

Mas a força do R está na gigantesca variedade de pacotes desenvolvidos pela comunidade, em especial, pelos criadores do `tidyverse`.

Instalando e carregando pacotes

Para instalar um pacote, usamos a função `install.packages`.

```
# Instalando um pacote
install.packages("tidyverse")

# Instalando vários pacotes de uma vez
install.packages(c("tidyverse", "rmarkdown", "devtools"))
```

Para usar as funções de um pacote, precisamos carregá-lo. Fazemos isso usando a função `library()`.

```
library(tidyverse)
```

Instale uma vez, carregue várias vezes!

Caminhos absolutos e relativos

Caminhos

Um passo importante na tarefa de importação de dados para o R é saber onde está o arquivo que queremos importar.

Toda função de importação vai exigir um **caminho**, uma string que representa o endereço do arquivo no computador.

Há duas formas de passarmos o caminho de arquivo: usar o **caminho absoluto** ou usar o **caminho relativo**.

Antes de falarmos sobre a diferença dos dois, precisamos definir o que é o **diretório de trabalho**.

Diretório de trabalho

O diretório de trabalho (*working directory*) é a pasta em que o R vai procurar arquivos na hora de ler informações ou gravar arquivos na hora de salvar objetos.

Se você está usando um projeto, o diretório de trabalho da sua sessão será, por padrão, a pasta raiz do seu projeto (é a pasta que contém o arquivo com extensão `.Rproj`).

Se você não estiver usando um projeto ou não souber qual é o seu diretório de trabalho, você pode descobri-lo usando a seguinte função `getwd()`.

Ela vai devolver uma string com o caminho do seu diretório de trabalho.

A função `setwd()` pode ser utilizada para mudar o diretório de trabalho. Como argumento, ela recebe o caminho para o novo diretório.

Caminhos absolutos

Caminhos absolutos são aqueles que tem início na pasta raiz do seu computador/usuário. Por exemplo:

```
/Users/beatrizmilz/Documents/Curso-R/cursos/introducao-programacao/main-intro-programacao/slides
```

Esse é o caminho absoluto para a pasta onde esses slides foram produzidos.

Na grande maioria dos casos, caminhos absolutos são uma **má prática**, pois deixam o código irreprodutível. Se você trocar de computador ou passar o script para outra pessoa rodar, o código não vai funcionar, pois o caminho absoluto para o arquivo muito provavelmente será diferente.

Caminhos relativos

Caminhos relativos são aqueles que tem início no diretório de trabalho da sua sessão.

O diretório de trabalho da sessão utilizada para produzir esses slides é a pasta `intro-programacao-em-r-mestre`. Veja o caminho absoluto no slide anterior. Então, o caminho relativo para a pasta onde esses slides foram produzidos seria apenas `slides/`.

Trabalhar com projetos no RStudio ajuda bastante o uso de caminhos relativos, pois nos incentiva a colocar todos os arquivos da análise dentro da pasta do projeto.

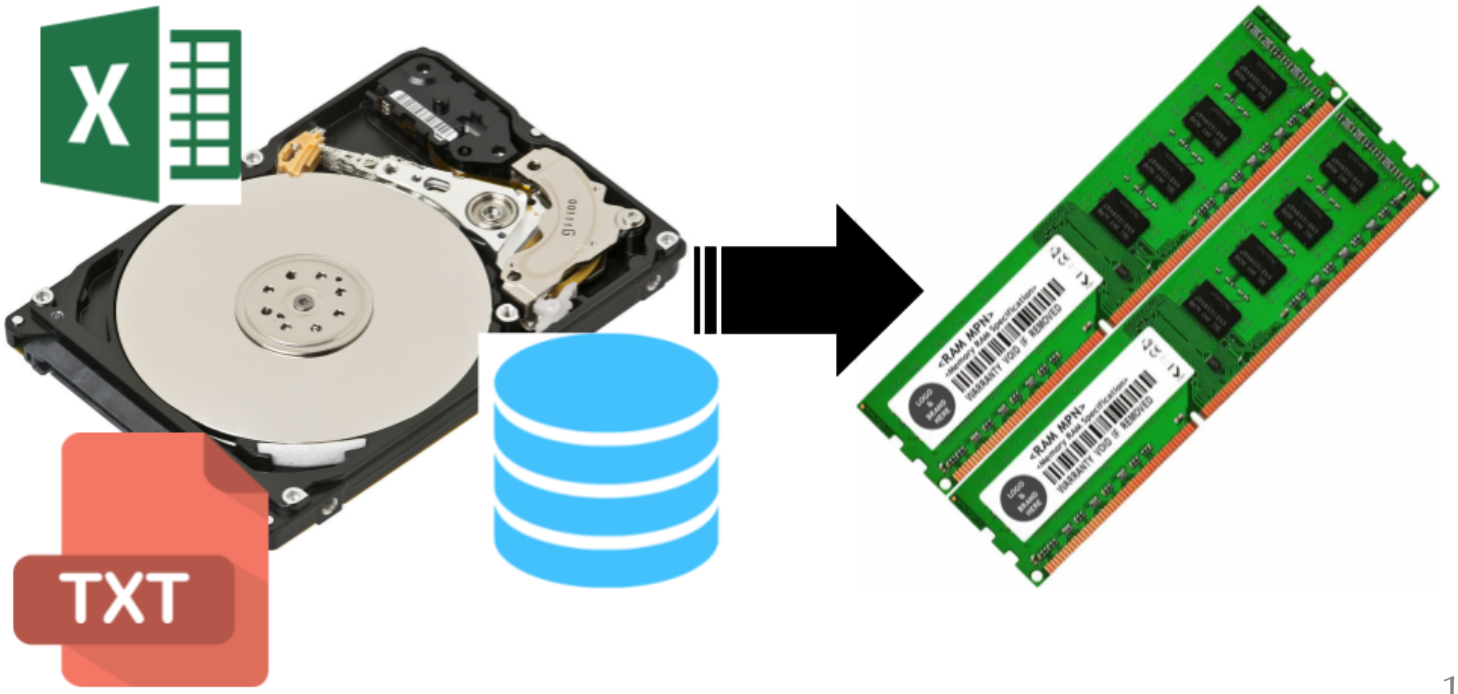
Assim, se você usar apenas caminhos relativos e compartilhar a pasta do projeto com alguém, todos os caminhos existentes nos códigos continuarão a funcionar em qualquer computador!

Importação de bases de dados

Importação de bases de dados

O que é?

Importar uma base de dados para o R significa levar a informação contida no disco rígido (HD) para a mé moria RAM.



curso-r

Lendo tabelas

Para ler tabelas, como arquivos `.csv`, utilizaremos funções do pacote `readr`.

Para isso, utilizamos a função `read_csv()` ou `read_csv2()`. Se o arquivo estiver bem formatado, a função só precisa do caminho até o arquivo para funcionar.

A mensagem devolvida pela função indica qual classe foi atribuída para cada coluna da base.

```
## ℹ Using '\',\'' as decimal and '\'.\'' as grouping mark. Use `read_delim()` for more control.

##
## ── Column specification ────────────────────────────────────────────────────────
## cols(
##   ano = col_double(),
##   mes = col_double(),
##   dia = col_double(),
##   horario_saida = col_double(),
##   saida_programada = col_double(),
##   atraso_saida = col_double(),
##   horario_chegada = col_double(),
##   chegada_prevista = col_double(),
##   atraso_chegada = col_double(),
##   companhia_aerea = col_character(),
##   voo = col_double(),
##   cauda = col_character(),
##   origem = col_character(),
##   destino = col_character(),
##   tempo_voo = col_double(),
##   distancia = col_double(),
##   hora = col_double(),
##   minuto = col_double(),
##   data_hora = col_datetime(format = "")
## )
```

- Em alguns países, como o Brasil, as vírgulas são utilizadas para separar as casas decimais dos números, inviabilizando os arquivos `.csv`. Nesses casos, os arquivos `.csv` são na verdade separados por ponto-e-vírgula. Para ler bases separadas por ponto-e-vírgula no R, utilize a função `read_csv2()`.

```
voos_csv <- readr::read_csv2("../dados/voos_de_janeiro.csv")
```

- Arquivos `.txt` podem ser lidos com a função `read_delim()`. Além do caminho até o arquivo, você também precisa indicar qual é o caractere utilizado para separar as colunas da base. Um arquivo separado por tabulação, por exemplo, pode ser lido utilizando a o código abaixo. O código `\t` é uma forma textual de representar a tecla TAB.

- Para ler planilhas do Excel (arquivos `.xlsx` ou `.xls`), basta utilizarmos a função `read_excel()` do pacote `readxl`.

Tabelas no R: Data frames

Tabelas no r: Data frames

O objeto mais importante para o cientista de dados é, claro, a base de dados. No R, uma base de dados é representa por objetos chamados de *data frames*. Eles são equivalentes a uma tabela do SQL ou uma planilha do Excel.

A principal característica de um *data frame* é possuir linhas e colunas:

```
mtcars
```

```
##                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```

O `mtcars` é um *data frame* nativo do R que contém informações sobre diversos modelos de carros. Ele possui 32 linhas e 11 colunas (só estamos vendo as primeiras 10 linhas no slide anterior).

A primeira "coluna" representa apenas o *nome* das linhas (modelo do carro), não é uma coluna da base. Repare que ela não possui um nome, como as outras. Essa estrutura de nome de linha é própria de *data frames* no R. Se exportássemos essa base para o Excel, por exemplo, essa coluna não apareceria.

Se você quiser saber mais sobre o `mtcars`, veja a documentação dele rodando `?mtcars` no **Console**.

Para entender melhor sobre *data frames*, precisamos estudar um pouco sobre classes, vetores e testes lógicos.

Classes

A classe de um objeto é muito importante dentro do R. É a partir dela que as funções e operadores conseguem saber exatamente o que fazer com um objeto.

Por exemplo, podemos somar dois números, mas não conseguimos somar duas letras (texto):

```
1 + 1
```

```
## [1] 2
```

```
"a" + "b"
```

```
## Error in "a" + "b": argumento não-numérico para operador binário
```

O operador `+` verifica que `"a"` e `"b"` não são números (ou que a classe deles não é numérica) e devolve uma mensagem de erro informando isso.

Texto

Observe que para criar texto no R, colocamos os caracteres entre aspas. As aspas servem para diferenciar *nomes* (objetos, funções, pacotes) de *textos* (letras e palavras). Os textos são muito comuns em variáveis categóricas.

```
a <- 10
# O objeto `a`, sem aspas
a
```

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

```
# A letra (texto) `a`, com aspas
"a"
```

```
## [1] "a"
```

A classe de um objeto

Para saber a classe de um objeto, basta rodarmos `class(nome-do-objeto)`.

```
x <- 1
class(x)
```

```
## [1] "numeric"
```

```
y <- "a"
class(y)
```

```
## [1] "character"
```

```
class(mtcars)
```

```
## [1] "data.frame"
```