

# Modificação de Valores

Aprenda técnicas essenciais para carregar, salvar e modificar estruturas de dados em R, dominando operações fundamentais com data frames.



Eduardo Ogasawara

[eduardo.ogasawara@cefet-rj.br](mailto:eduardo.ogasawara@cefet-rj.br)

<https://eic.cefet-rj.br/~eogasawara>

## Atribuição e Estado do Programa

### O que é atribuição?

Atribuição é o processo de associar um valor a um nome de variável. Quando fazemos isso, alteramos o **estado do programa**.



#### Associação

Liga um nome a um valor na memória



#### Mudança de Estado

Quando um valor muda, o programa evolui

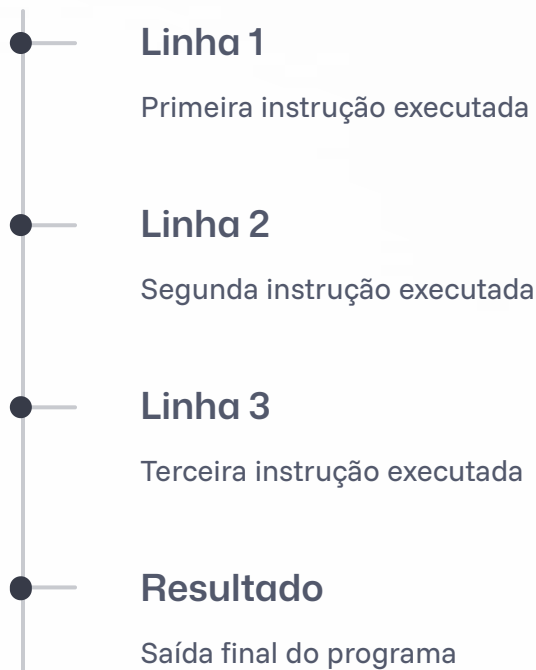


#### Sem Memória Automática

O valor antigo não é lembrado automaticamente

O programa evolui à medida que as instruções são executadas. Cada atribuição representa uma mudança no estado atual.

## Execução Sequencial de um Programa



📄 **Consequência prática:** Trocar a ordem das linhas pode mudar completamente o resultado. A sequência importa!

Um programa é executado **linha por linha**, de cima para baixo. A ordem das instruções é fundamental, pois o resultado final depende da sequência e dos valores atuais das variáveis.

## Carregando Baralho de uma URL

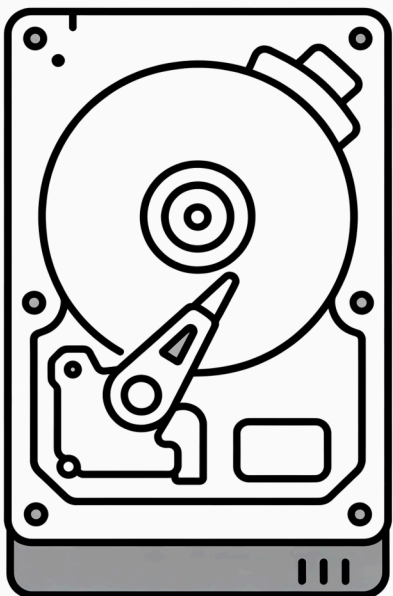
O R permite carregar dados diretamente de URLs remotas usando a função `load()` combinada com `url()`. Esta técnica é especialmente útil para acessar datasets compartilhados online.

O comando carrega o objeto `baralho` na memória, tornando-o imediatamente disponível para análise.

```
u <- "https://raw.githubusercontent.com/
eogasawara/tutorial-r/main/
examples/baralho.rda"
```

```
load(url(u))
head(baralho)
```

**Resultado:** Visualização das primeiras 6 linhas do baralho, mostrando colunas `face` e `naipe`.



 SALVAMENTO

## Salvando Baralho em Arquivo Local

01

### Salvar o objeto

Use `save()` para persistir o data frame em formato `.rda`

02

### Remover da memória

O comando `rm()` limpa o objeto do ambiente de trabalho atual

03

### Arquivo criado

O arquivo `baralho.rda` fica disponível no diretório de trabalho

```
save(baralho, file="baralho.rda")  
rm(baralho)
```

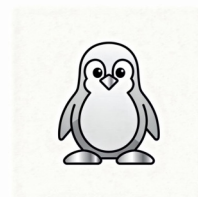
# Recarregando Dados do Arquivo

## Carregamento local

Após salvar, você pode recarregar os dados a qualquer momento usando `load()` com o caminho do arquivo.

```
load(file="baralho.rda")  
head(baralho)
```

O data frame retorna exatamente como estava antes de ser removido da memória.



### Importante no Linux

Em sistemas Linux, adicione o parâmetro `compress=TRUE` para otimizar o armazenamento:

```
save(baralho, file="baralho.rda", compress=TRUE)
```

## Adicionando Colunas à Tabela

Expandir um data frame com novas colunas é simples no R. Use o operador `$` para criar e atribuir valores a uma nova coluna.



### Criação da coluna

Use `baralho$idx` para adicionar uma nova coluna chamada `idx`



### Atribuição de valores

A sequência `1:52` cria índices sequenciais para as 52 cartas



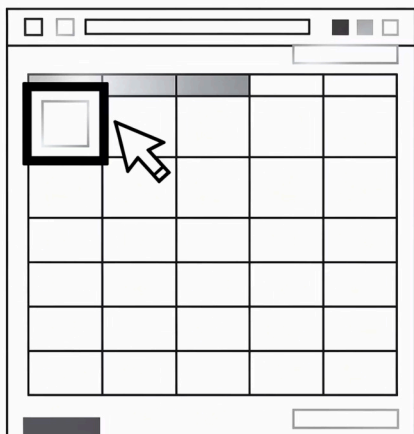
### Resultado imediato

A coluna aparece instantaneamente no data frame

```
baralho$idx <- 1:52  
head(baralho)
```

# Alterando Valores em Vetores ou Colunas

O R permite modificar valores específicos em colunas usando indexação. Você pode alterar elementos individuais ou realizar operações em subconjuntos.



1

## Modificação por índice

Altere valores específicos usando vetores de posições

```
baralho$idx[c(1,3,5)] <- 1
```

2

## Modificação por intervalo

Aplique operações matemáticas a intervalos de valores

```
baralho$idx[4:6] <- baralho$idx[4:6] + 1
```

Essas técnicas são fundamentais para limpeza e transformação de dados.

## Filtrando Dados do Data Frame

### Criando vetores lógicos

Use operações de comparação para criar vetores booleanos que identificam quais linhas atendem aos critérios.

```
vec <- (baralho$idx %% 2 == 1)
idx <- baralho$idx[vec]
```

O operador `%%` retorna o resto da divisão, identificando números ímpares.

### Aplicando o filtro

Use o vetor lógico para extrair apenas as linhas desejadas:

```
cartas <- baralho[vec,]
```

Resultado: apenas cartas com índices ímpares (1, 3, 5, 7...).

# Operações de Comparação

O R oferece operadores poderosos para comparar valores e criar filtros lógicos. Domine estes operadores para manipular dados eficientemente.



## Comparação simples

```
1 > 2 # FALSE
1 > c(0, 1, 2) # TRUE FALSE
FALSE
```



## Igualdade vetorial

```
c(1,2,3) == c(3,2,1) # FALSE
TRUE FALSE
```



## Operador %in%

```
1 %in% c(3,4,5) # FALSE
c(1,2,3) %in% c(3,4,5) # FALSE
FALSE TRUE
```



**Operadores disponíveis:** `>`, `>=`, `<`, `<=`, `==`, `!=`

# Operadores Lógicos

Combine múltiplas condições usando operadores lógicos para criar filtros complexos e precisos.

## Operador AND (&)

Ambas condições devem ser verdadeiras

```
x <- baralho$face == "dama" &  
  baralho$naipe == "espadas"  
head(baralho[x,])
```

Resultado: apenas a dama de espadas

## Operador OR (|)

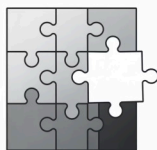
Pelo menos uma condição deve ser verdadeira

```
x <- baralho$face == "dama" |  
  baralho$naipe == "espadas"  
head(baralho[x,])
```

Resultado: todas as damas e todas as cartas de espadas

**Outros operadores:** `&`, `|`, `xor`, `!`, `any`, `all`

# Tratamento de Ausência de Dados (NA)



Valores `NA` (Not Available) representam dados ausentes no R. É crucial entender como eles se comportam em operações.

## Comportamento do NA

```
1 + NA # NA
NA == 1 # NA
mean(c(NA, 1:50)) # NA
```

Operações com `NA` geralmente retornam `NA`.

1

## Remover NAs

Use `na.rm = TRUE` em funções estatísticas

```
mean(c(NA, 1:50), na.rm = TRUE) # 25.5
```

2

## Filtrar com cuidado

Comparações com colunas inexistentes retornam dados vazios

```
baralho[baralho$valor < 3, ] # 0 rows
```

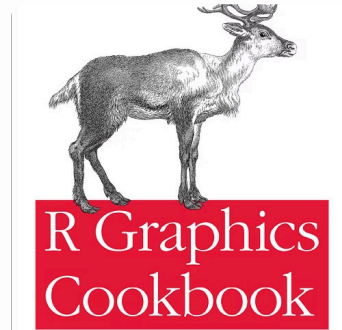
# Referências



## Hands-on Programming

Aprenda R criando suas próprias funções e simulações

<https://rstudio-education.github.io/hopr/index.html>



## R Graphics Cookbook

Domine visualizações de dados em R

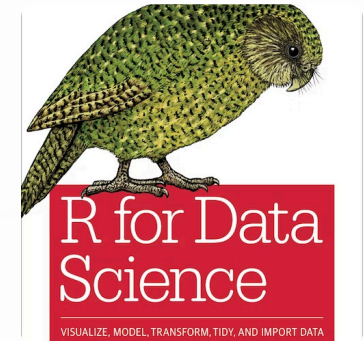
<https://r-graphics.org>



## R Packages

Desenvolva seus próprios pacotes R

<https://r-pkgs.org/index.html>



## R for Data Science

Guia completo para ciência de dados

<https://r4ds.had.co.nz>