

3 Tipos de Dados Compostos 3.5 DataFrames

# **Data Frames** · São como tabelas do SQL ou Planilhas do Excel · Geralmente é como seus dados serão representados · Listas especiais onde todos os elementos possuem o mesmo comprimento · Cada elemento é uma coluna, e seu comprimento é a quantidade de linhas · Então é uma lista de colunas · Cada coluna pode ter tipos diferentes

SEPT / UFPR

Data Frames: Criação • Transformação de Lista em Data Frame > lista <- list(x=c(1, 2, 3), y=c("a", "b", "c")) > lista \$x [1] 1 2 3 \$у [1] "a" "b" "c" > df <- as.data.frame(lista) > df х у 1 1 a 2 2 b 3 3 c SEPT / UFPR Prof. Dr. Razer A N R Montaño

Prof. Dr. Razer A N R Montaño

```
Data Frames: Criação
          • Criação de Data Frame Vazio: data.frame()
                  > df <- data.frame()
                  > df
                      data frame with 0 columns and 0 rows
                                                                                      Nota

    Adição de colunas: operador $

                                                                              integer (x) cria um vetor de
                  > df$x <- integer()</pre>
                                                                             inteiros de comprimento x
                  > df$nome <- character()
                                                                             (default 0). Todos recebem o valor
                  > df
                                                                             character(x) cria um vetor
                  [1] x
                             nome
                                                                             de caracteres de comprimento x
                                                                             (default 0). Todos recebem o valor
                  <0 linhas> (ou row.names de comprimento 0)

    Adição de linhas

                  > df[nrow(df)+1,] <- c(10, "razer")
                         nome
                  1 10 razer
          Prof. Dr. Razer A N R Montaño
                                                                                SEPT / UFPR
5
```

```
Data Frames: Criação
• Criação de Data Frame Vazio com Colunas: data.frame()
       > df <- data.frame(x=integer(), nome=character())</pre>
       > df
       [1] x
                nome
       <0 linhas> (ou row.names de comprimento 0)
• Função str () mostra a estrutura de um objeto qualquer
       > str(df)
       'data.frame': 0 obs. of 2 variables:
        $ x : int
        $ nome: Factor w/ 0 levels:
                                                             SEPT / UFPR
Prof. Dr. Razer A N R Montaño
```

```
Data Frames: Criação
• Criação de Data Frame com Dados: data.frame()
      > df <- data.frame(id = letters[1:10], x = 1:10, y = 11:20)
      > df
         id x y
          a 1 11
          b 2 12
          c 3 13
      3
          d 4 14
      5
          e 5 15
      6
          f 6 16
          g 7 17
          h 8 18
      9
         i 9 19
      10 ј 10 20
Prof. Dr. Razer A N R Montaño
                                                        SEPT / UFPR
```

```
Data Frames: Criação
• Criação de Data Frame com Dados: data.frame()
      > df <- data.frame(nome=c("Razer", "Anthom", "Nizer", "Rojas",</pre>
"Montaño"), idade=c(20, 18, 26, 32, 15))
      > df
           nome idade
                    20
      1
          Razer
      2 Anthom
                  18
      3 Nizer
                   26
         Rojas
                   32
                   15
      5 Montaño
                                                          SEPT / UFPR
Prof. Dr. Razer A N R Montaño
```

```
Data Frames: Criação
        • Criação de Data Frame com Dados, Sem Factors: data.frame()
            Parâmetro booleano: stringsAsFactors
               > df <- data.frame(nome=c("Razer", "Anthom", "Nizer", "Rojas",
        "Montaño"), idade=c(20, 18, 26, 32, 15))
               > str(df)
                'data.frame': 5 obs. of 2 variables:
                $ nome : Factor w/ 5 levels "Anthom", "Montaño", ...: 4 1 3 5 2
                $ idade: num 20 18 26 32 15
               > df <- data.frame(nome=c("Razer", "Anthom", "Nizer", "Rojas",
        "Montaño"), idade=c(20, 18, 26, 32, 15), stringsAsFactors=FALSE)
               > str(df)
               'data.frame': 5 obs. of 2 variables:
                $ nome : chr "Razer" "Anthom" "Nizer" "Rojas" ...
                $ idade: num 20 18 26 32 15
        Prof. Dr. Razer A N R Montaño
                                                                   SEPT / UFPR
9
```

```
Data Frames: Criação.
                                                "nome","idade"
                                               "Razer",18
• Leitura de um arquivo CSV: read.csv()
                                               "Anthom",20
  • Seja o seguinte arquivo : exemplo.csv
                                               "Nizer",15
                                               "Rojas",30
                                               "Montaño",26

    Carrega-se com: read.csv

       > df <- read.csv("exemplo.csv", fileEncoding="UTF-8")</pre>
       > str(df)
       'data.frame': 5 obs. of 2 variables:
        $ nome : Factor w/ 5 levels "Anthom", "Montaño", ..: 4 1 3 5 2
        $ idade: int 18 20 15 30 26
       > df
             nome idade
                     18
           Razer
       2 Anthom
                     20
                     1.5
          Nizer
                     30
           Rojas
                     26
       5 Montaño
Prof. Dr. Razer A N R Montaño
                                                                SEPT / UFPR
```

```
Data Frames: Funções Úteis

> df <- data.frame(id = letters[1:10], x = 1:10, y = 11:20)

• Funções úteis

• dim():número de linhas e colunas

> dim(df)

[1] 10 3

• nrow():número de linhas

> nrow(df)

[1] 10

• ncol():número de colunas

> ncol(df)

[1] 3
```

```
Data Frames: Funções Úteis
         > df <- data.frame(id = letters[1:10], x = 1:10, y = 11:20)

    Funções úteis

           • names () : mostra o nome das colunas
                > names(df)
                [1] "id" "x" "y"
           • rownames () : mostra o nome das linhas
                > rownames(df)
                [1] "1" "2" "3" "4" "5" "6" "7" "8" "9" "10"
           • str () : mostra a estrutura do data frame, amostra de dados, e a classe de cada coluna
                > str(df)
                'data.frame': 10 obs. of 3 variables:
                 $ id: Factor w/ 10 levels "a", "b", "c", "d", ..: 1 2 3 4 5 6 7 8 9 10
                 $ x : int 1 2 3 4 5 6 7 8 9 10
                 $ y : int 11 12 13 14 15 16 17 18 19 20
        Prof. Dr. Razer A N R Montaño
                                                                     SEPT / UFPR
13
```

```
Data Frames: Funções Úteis
> df <- data.frame(id = letters[1:10], x = 1:10, y = 11:20)
> df2 <- data.frame(z = 111:120)

    Funções úteis

    cbind(): acopla dois data frames lado a lado (adiciona colunas)

      > cbind(df, df2)
          id x y z.
                                                        Coluna Adicionada
          a 1 11 111
          b 2 12 112
          c 3 13 113
          d 4 14 114
      4
      5
           e 5 15 115
          f 6 16 116
          q 7 17 117
      7
      8
          h 8 18 118
          i 9 19 119
          j 10 20 120
Prof. Dr. Razer A N R Montaño
                                                          SEPT / UFPR
```

```
Data Frames: Funções Úteis
        > df <- data.frame(id = letters[1:10], x = 1:10, y = 11:20)
        > df3 <- data.frame(id="r", x=200, y=300),

    Funções úteis

           • rbind(): empilha duas tabelas (adiciona linhas)
               > rbind(df, df3)
                  id
                       Х
                            У
                                                                Linha Adicionada
                       1
               1
                   а
                          11
               2
                        2 12
                   b
               3
                       3 13
                   С
               4
                   d
                       4 14
               5
                   е
                       5 15
               6
                  f
                       6 16
               7
                       7
                          17
                   g
                          18
                         19
                      10 20
               10
                   j
                  r 200 300
        Prof. Dr. Razer A N R Montaño
                                                                  SEPT / UFPR
15
```

```
Data Frames: Funções Úteis.
> lista <- list(nome=c("TADS", "TGQ", "TCI"), alunos=c(45, 50, 40))
> lista
$nome
[1] "TADS" "TGQ" "TCI"
$alunos
[1] 45 50 40
· Funções úteis
  • as.data.frame(): transforma uma lista em data frame
      > as.data.frame(lista)
           nome alunos
       1 TADS
                   45
                   50
         TGQ
          TCI
                   40
                                                         SEPT / UFPR
Prof. Dr. Razer A N R Montaño
```

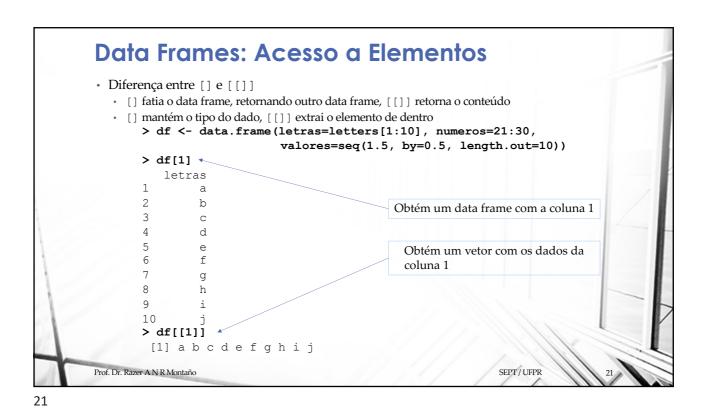
### Data Frames: Acesso a Elementos Acesso a elementos como em uma matriz > lista <- list(x=c(1, 2, 3), y=c("a", "b", "c")) > df <- as.data.frame(lista)</pre> > str(df) 'data.frame': 3 obs. of 2 variables: \$ x: num 1 2 3 \$ y: Factor w/ 3 levels "a", "b", "c": 1 2 3 > df х у 1 1 a 2 2 b 3 3 c > df[3, 2][1] c Levels: a b c > df[3, 1][1] 3 Prof. Dr. Razer A N R Montaño SEPT / UFPR

```
Data Frames: Acesso a Elementos
· Retornar uma coluna individualmente
· Retorna um Data Frame com a coluna escolhida
       > df[1]
          Х
       1 1
2 2
3 3
> df[2]
          У
       1 a
        2 b
   Retorna um Vetor com a coluna escolhida (dados do mesmo tipo, [ , ] retorna vetor
   quando é uma coluna)
       > df[,1]
[1] 1 2 3
> df[,2]
        [1] a b c
        Levels: a b c
                                                                    SEPT / UFPR
Prof. Dr. Razer A N R Montaño
```

18

```
Data Frames: Acesso a Elementos
· Retornar uma linha individualmente
· Retorna um Data Frame com a linha escolhida
        > df[1,]
        x y
1 1 a
        > df[2,]
          х у
        2 2 b
• Retorna uma Lista com a linha escolhida (dados diferentes, precisa ser lista, usar drop)
        > df[1, ,drop=TRUE]
$x
[1] 1
        $y
[1] a
        Levels: a b c
        > df[2, ,drop=TRUE]
        $x
[1] 2
        $y
[1] b
Levels: a b c
Prof. Dr. Razer A N R Montaño
                                                                           SEPT / UFPR
```

```
Data Frames: Acesso a Elementos
· Acesso a elementos pelo nome da coluna
      > lista <- list(x=c(1, 2, 3), y=c("a", "b", "c"))</pre>
      > df <- as.data.frame(lista)</pre>
      > df
        х у
      1 1 a
      2 2 b
      3 3 с
      > df$x
      [1] 1 2 3
      > df$y
      [1] "a" "b" "c"
      > df$y[3]
       [1] c
                                                         SEPT / UFPR
Prof. Dr. Razer A N R Montaño
```

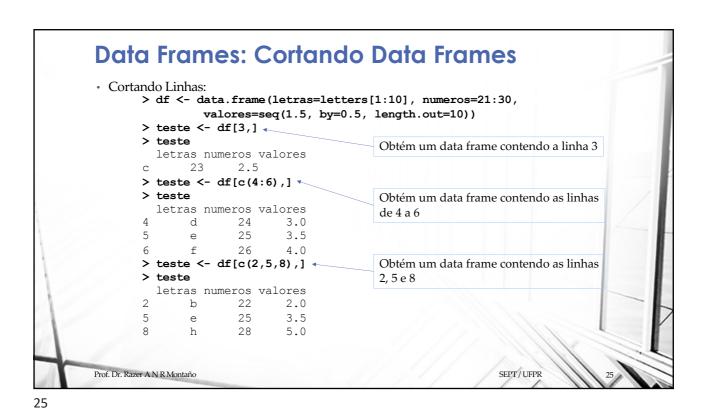


```
Data Frames: Acesso a Elementos
• Diferença entre [] e [[]]
   • Vetores: Retornam o mesmo tipo de dado
                                             · Matrizes: Como é um vetor com dimensões,
       > vet <- 1:10
                                               retornam o mesmo tipo de dado
       > vet
                                                  > mat <- matrix(1:10, nrow=5)
        [1] 1 2 3 4 5 6 7 8 9
                                                  > mat
10
                                                       [,1] [,2]
       > vet[2]
                                                          1
                                                               6
                                                  [1,]
       [1] 2
                                                          2
                                                  [2,]
       > vet[[2]]
                                                               8
                                                  [3,]
                                                          3
       [1] 2
                                                               9
                                                  [4,]
                                                          4
                                                              10
                                                  [5,]
                                                  > mat[5]
                                                  [1] 5
                                                  > mat[[5]]
                                                  [1] 5
                                                                 SEPT / UFPR
Prof. Dr. Razer A N R Montaño
```

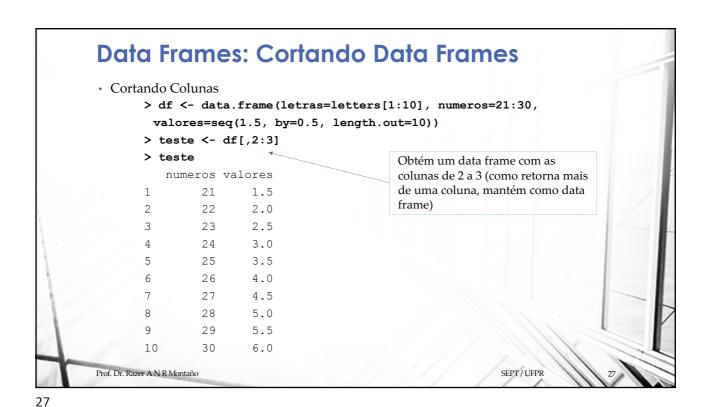
```
Data Frames: Acesso a Elementos
• Diferença entre [] e [[]]
  • Listas: Retornam tipos diferentes. [] retorna uma sublista e [[]] um vetor
       > lista <- list(letras=letters[1:5], numeros=10:1, valores=rnorm(3))</pre>
       > lista
       $letras
       [1] "a" "b" "c" "d" "e"
       $numeros
        [1] 10 9 8 7 6 5 4 3 2 1
       $valores
       [1] -0.4997171 -0.7824937 -0.5532524
       > lista[2]
       $numeros
        [1] 10 9 8 7 6 5 4 3 2 1
       > lista[[2]]
        [1] 10 9 8 7 6 5 4 3 2 1
Prof. Dr. Razer A N R Montaño
                                                              SEPT / UFPR
```

Data Frames: Acesso a Elementos. • Diferença entre [] e [[]] • Data Frames: Como são listas, retornam tipos diferentes. [] retorna um sub-data-frame e [[]] um vetor > df <- data.frame(numero=1:5, letras=letters[1:5]) > df numero letras 1 1 2 2 b 3 3 С 4 4 d 5 5 > df[2] letras b 3 С d > df[[2]] [1] a b c d e Levels: a b c d e SEPT / UFPR Prof. Dr. Razer A N R Montaño

24



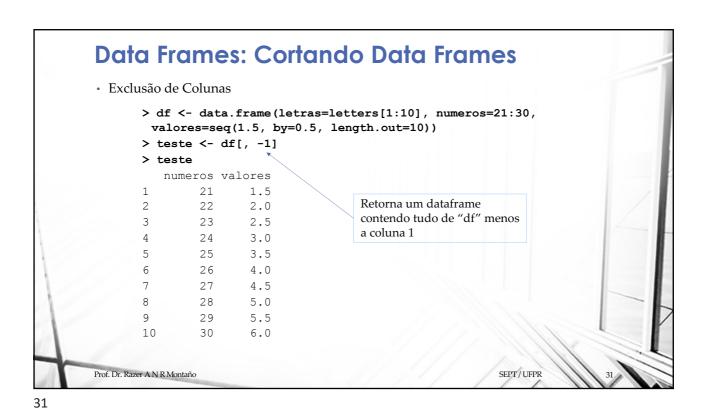
**Data Frames: Cortando Data Frames**  Cortando Colunas > df <- data.frame(letras=letters[1:10], numeros=21:30, valores=seq(1.5, by=0.5, length.out=10)) > teste <- df[,2] -Obtém um vetor com a coluna 2 > teste (como é só uma coluna transforma [1] 21 22 23 24 25 26 27 28 29 30 > teste <- df[,2, drop=FALSE]</pre> em vetor) > teste numeros Obtém um data frame contendo a 1 21 coluna 2 (drop=FALSE) 2 22 3 23 4 24 5 25 6 26 7 27 8 28 9 29 10 30 Prof. Dr. Razer A N R Montaño SEPT / UFPR



**Data Frames: Cortando Data Frames**  Cortando Colunas > df <- data.frame(letras=letters[1:10], numeros=21:30,</pre> valores=seq(1.5, by=0.5, length.out=10)) > teste <- df[,c(1,3)] > teste Obtém um data frame com as letras valores colunas de 1 e 3 (como retorna mais 1.5 de uma coluna, mantém como data 1 а frame) b 2.0 2.5 С 4 d 3.0 3.5 6 f 4.0 4.5 g 5.0 h i 5.5 6.0 j SEPT / UFPR Prof. Dr. Razer A N R Montaño

```
Data Frames: Cortando Data Frames
· Cortando Linhas e Colunas
      > df <- data.frame(letras=letters[1:10], numeros=21:30,
        valores=seq(1.5, by=0.5, length.out=10))
      > teste <- df[c(1,6,10), c("letras", "valores")]
      > teste
                                        Obtém um data frame contendo as linhas
         letras valores
                                        de 1, 6 e 10, e as colunas "letras" e "valores"
             a 1.5
             f
                   4.0
      10
                   6.0
             j
Prof. Dr. Razer A N R Montaño
                                                         SEPT / UFPR
```

```
Data Frames: Cortando Data Frames
· Exclusão de Linhas
      > df <- data.frame(letras=letters[1:10], numeros=21:30,
       valores=seq(1.5, by=0.5, length.out=10))
      > teste <- df[-c(2:9),]
      > teste
                                              Obtém um data frame
         letras numeros valores
                                              contendo tudo menos as
            a 21 1.5
                                              linhas de 2 a 9
      10
                    30
              j
      > teste <- df[-c(1,3,5,7,9),]
      > teste
                                              Obtém um data frame
         letras numeros valores
                                              contendo tudo menos as
      2 b 22 2
                                              linhas de 1, 3, 5, 7 e 9
             d
                   24
             f
                    26
              h
                    28
      10
                     30
Prof. Dr. Razer A N R Montaño
                                                       SEPT / UFPR
```



```
Data Frames: Cortando Data Frames

    Exclusão de Colunas

       > df <- data.frame(letras=letters[1:10], numeros=21:30,
        valores=seq(1.5, by=0.5, length.out=10))
       > teste <- df[, -c(1:2)]
        [1] 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0
                                                          Obtém um vetor contendo
                                                          tudo menos as colunas de 1 a
       > teste <- df[, -c(1,3)]
                                                          2 (ficam os dados da coluna
       > teste
                                                          3). Como sobra só uma coluna
        [1] 21 22 23 24 25 26 27 28 29 30
                                                          retorna um vetor.
                                    Obtém um vetor contendo
                                    tudo menos as colunas de 1 e
                                    3 (sobram os dados da coluna
                                    2). Como sobra só uma
                                    coluna, retorna um vetor.
Prof. Dr. Razer A N R Montaño
                                                              SEPT / UFPR
```

```
Data Frames: Cortando Data Frames.
· Exclusão de Colunas
       > df <- data.frame(letras=letters[1:10], numeros=21:30,</pre>
        valores=seq(1.5, by=0.5, length.out=10))
       > teste <- df[, -c(1:2), drop=FALSE]</pre>
       > teste
        valores
               1.5
                                                Nesse caso o "drop=FALSE"
               2.0
                                                indica que o retorno não será
       3
               2.5
                                                 transformado em vetor.
       4
               3.0
                                                 mantendo como dataframe
       5
               3.5
               4.0
       6
       7
               4.5
               5.0
       8
       9
               5.5
       10
               6.0
Prof. Dr. Razer A N R Montaño
                                                               SEPT / UFPR
```

```
Data Frames: Seleção de Elementos
• Seleção condicional de dados com which ()
    which (): recebe um condicional e retorna a posição dos elementos que atendem esta condição (Verdadeiro-
    TRUE)
       > lista <- c(TRUE, FALSE, FALSE, TRUE)
       > which(lista)
       [1] 1 4
       > lista <- c(10:1)
       > lista
       [1] 10 9 8 7 6 5 4 3 2 1
       > lista==3
       [1] FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE
       > which( lista==3 )
       [1] 8
       > which( lista%%2==0 )
       [1] 1 3 5 7 9
       > which(lista%2==0 & lista>5)
       [1] 1 3 5
Prof. Dr. Razer A N R Montaño
                                                              SEPT / UFPR
```

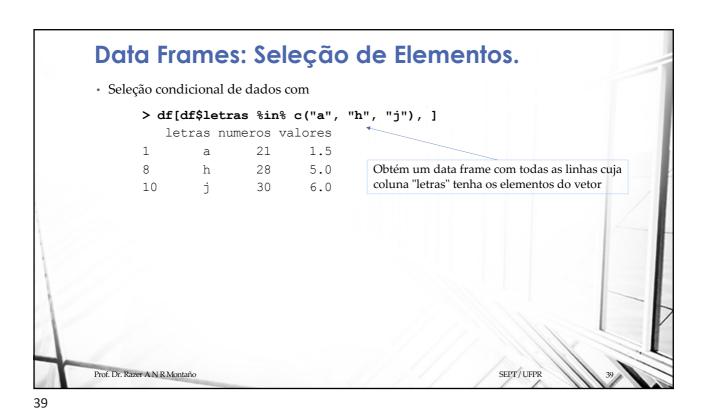
```
Data Frames: Seleção de Elementos
        • Seleção condicional de dados com which ()
               > df <- data.frame(letras=letters[1:10], numeros=21:30,
                 valores=seq(1.5, by=0.5, length.out=10))
               > which (df$numeros>=28)
                                                        Obtém a posição dos elementos
               [1] 8 9 10
                                                        cujos números são maiores ou
               > df[ which(df$numeros>=28),
                                                        iguais a 28
                  letras numeros valores
                                                         Retorna um data frame com as
                        h
                                28
                                        5.0
                                                         linhas cujos números são
                                29
                                        5.5
                                                         maiores ou iguais a 28
                                30
                                        6.0
               > teste <- df[ which(df$numeros>=28),
               > df[ which(df$numeros>=28 & df$numero%%2==1),
                 letras numeros valores
                       i
                              29
                                       5.5
        Prof. Dr. Razer A N R Montaño
                                                                 SEPT / UFPR
35
```

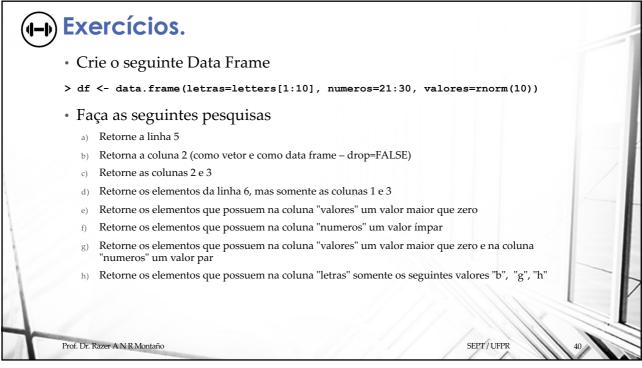
```
Data Frames: Seleção de Elementos
• Seleção condicional de dados SEM which ()
      > df <- data.frame(letras=letters[1:10], numeros=21:30,</pre>
        valores=seq(1.5, by=0.5, length.out=10))
      > df[ df$numeros>=28,
         letras numeros valores
               h
                      28
      9
               i
                      29
                              5.5
                      30
                              6.0
      > teste <- df[ df$numeros>=28,
      > df[ df$numeros>=28 & df$numero%%2==1,
        letras numeros valores
              i
                     29
                             5.5
Prof. Dr. Razer A N R Montaño
                                                      SEPT / UFPR
```

# Data Frames: Seleção de Elementos Seleção condicional de dados com %in% %in% verifica se um elemento está dentro do vetor > 10 %in% c(5:50) [1] TRUE > 1 %in% c(5:50) [1] FALSE Quando aplicado a um vetor, verifica cada um dos elementos Retorna um vetor com o resultado de cada teste, para cada elemento > c(1:10) %in% c(5:50) [1] FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE Prof. Dr. Razer ANR Montaño SELT/UFFR 37

```
Data Frames: Seleção de Elementos
· Quando aplicado a um data frame, também verifica cada um dos elementos
       > df <- data.frame(letras=letters[1:10], numeros=21:30,</pre>
        valores=seq(1.5, by=0.5, length.out=10))
          letras numeros valores
      1
                      21
                            1.5
               b
                      22
                              2.0
                             2.5
       3
                      23
               С
               d
                      24
                              3.0
       5
                      25
                              3.5
               е
       6
                      26
                              4.0
               f
                      27
                              4.5
               g
               h
                      28
                              5.0
               i
                      29
                              5.5
               j
                      30
                              6.0
       > df$letras %in% c("a", "d", "i")
             TRUE FALSE FALSE TRUE FALSE FALSE FALSE
                                                               TRUE FALSE
        [1]
Prof. Dr. Razer A N R Montaño
                                                           SEPT / UFPR
```

38





### **Data Frames: Merge**

- · Usado para unir data frames de forma horizontal
  - · CROSS JOIN: produto cartesiano
  - · INNER JOIN
    - · Une fazendo o casamento de atributos chave
    - · Só une se todos os atributos casarem
  - · OUTER JOIN
    - · Une fazendo o casamento de atributos chave
    - · Une os dados mesmo se o atributo chave não casar nos dois data frames
  - · LEFT OUTER JOIN
    - · Une fazendo o casamento de atributos chave
    - Une todos os dados do data frame esquerdo (primeiro), mesmo se não houver correspondência no segundo
  - · RIGHT OUTER JOIN
    - · Une fazendo o casamento de atributos chave
    - Une todos os dados do data frame direito (segundo), mesmo se não houver correspondência no primeiro

Prof. Dr. Razer A N R Montaño

SEPT / UFPR

41

41

### **Data Frames: Merge**

· Sejam 2 data frames: df1 e cidades

df1	
<nome></nome>	<cidadeid></cidadeid>
"Razer"	3
"Anthom"	10
"Nizer"	2
"Rojas"	3
"Montaño"	1

cidades	
<cidadeid></cidadeid>	<cidade></cidade>
1	"Curitiba"
2	"SJP"
3	"Pinhais"
4	"Colombo"

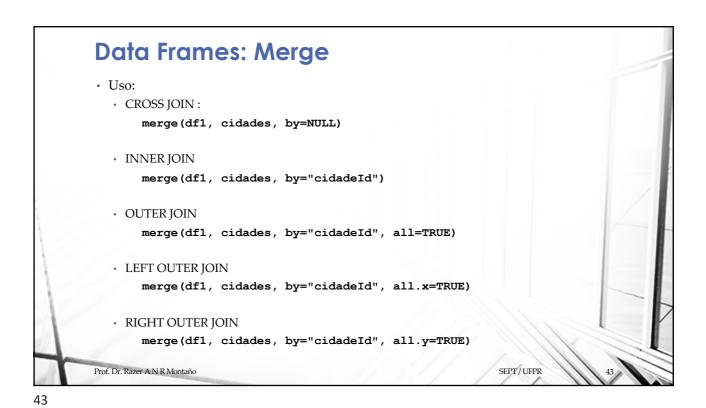
> df1 <- data.frame(nome=c("Razer", "Anthom", "Nizer", "Rojas", "Montaño"),
cidadeId=c(3, 10, 2, 3, 1))</pre>

> cidades <- data.frame(cidadeId=c(1, 2, 3, 4), cidade=c("Curitiba", "SJP",
"Pinhais", "Colombo"))</pre>

Prof. Dr. Razer ANR Montaño

SEPT / UFPR

12



### **Data Frames: Merge**

- INNER JOIN
  - · Une fazendo o casamento de atributos chave
  - · Só une se todos os atributos casarem

```
df1 <- data.frame(nome=c("Razer", "Anthom", "Nizer", "Rojas", "Montaño"), cidadeId=c(3, 10, 2, 3, 1))
df1
     nome cidadeId
     Razer
   Anthom
Nizer
                  10
     Rojas
  Montaño
               data.frame(cidadeId=c(1, 2, 3, 4), cidade=c("Curitiba", "SJP", "Pinhais", "Colombo"))
  cidades <-
   cidadeId
               cidade
            Curitiba
SJP
              Pinhais
              Colombo
  merge(df1, cidades, by="cidadeId")
cidadeId nome cidade
   cidadeId
                nome
           1 Montaño Curitiba
               Nizer
                      Pinhais
                       Pinhais
Prof. Dr. Razer A N R Montaño
                                                                                           SEPT / UFPR
```

45

### **Data Frames: Merge**

- OUTER JOIN
  - · Une fazendo o casamento de atributos chave
  - · Une os dados mesmo se o atributo chave não casar nos dois data frames

```
| Solution | Sept | Sep
```

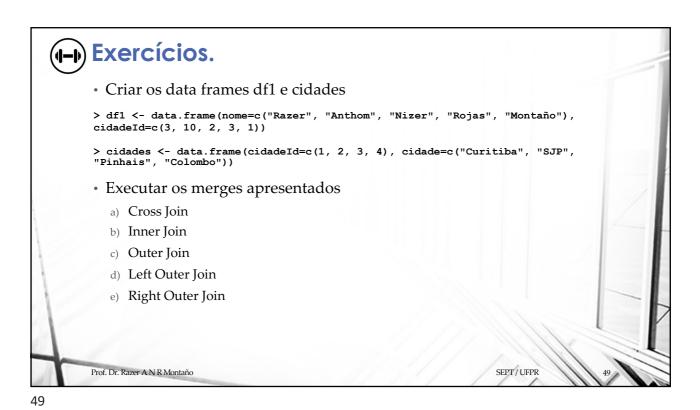
### **Data Frames: Merge**

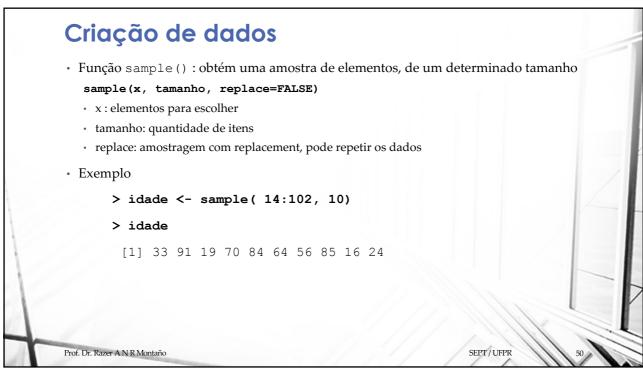
- LEFT OUTER JOIN
  - · Une fazendo o casamento de atributos chave
  - Une todos os dados do data frame esquerdo (primeiro), mesmo se não houver correspondência no segundo

47

## Data Frames: Merge.

- RIGHT OUTER JOIN
  - · Une fazendo o casamento de atributos chave
  - Une todos os dados do data frame direito (segundo), mesmo se não houver correspondência no primeiro





```
Criação de dados
• Exemplo de criação de dados de pessoas usando sample() e data.frame()
  > n <- 10
  > sexo <- sample(c("masculino", "feminino"), n, replace=TRUE)
  > idade <- sample( 14:102, n, replace=TRUE)</pre>
  > peso <- sample( 50:90, n, replace=TRUE)
  > menor <- idade<18
  > pessoas <- data.frame(sexo=sexo, idade=idade, peso=peso, menor=menor)
  > pessoas
           sexo idade peso menor
                       71 FALSE
   1
      feminino
                  44
                   81
                        70 FALSE
      feminino
   3 masculino
                       62 TRUE
                   14
                 22
                       79 FALSE
     feminino
   5 masculino
                49
                       61 FALSE
                16
   6 masculino
                       82 TRUE
      feminino
                   72
                       83 FALSE
   8 masculino
                        79 FALSE
                   80
   9 masculino
                 31
                       81 FALSE
   10 feminino
                 57
                        66 FALSE
Prof. Dr. Razer A N R Montaño
                                                            SEPT / UFPR
```

```
Criação de dados

• Ao criar o data frame, automaticamente o sexo é transformado em Factor

• Factor são valores categóricos

> str(pessoas)

'data.frame': 10 obs. of 4 variables:

$ sexo : Factor w/ 2 levels "feminino", "masculino": 1 1 2 1 2 2 1 2 2 1

$ idade: int 44 81 14 22 49 16 72 80 31 57

$ peso : int 71 70 62 79 61 82 83 79 81 66

$ menor: logi FALSE FALSE TRUE FALSE FALSE TRUE ...
```

```
Criação de dados.

Para evitar este comportamento use o parâmetro stringsAsFactors

> pessoas <- data.frame(sexo=sexo, idade=idade, peso=peso, menor=menor, stringsAsFactors=FALSE)
> str(pessoas)
'data.frame': 10 obs. of 4 variables:
$ sexo: chr "feminino" "feminino" "masculino" "feminino" ...
$ idade: int 44 81 14 22 49 16 72 80 31 57
$ peso: int 71 70 62 79 61 82 83 79 81 66
$ menor: logi FALSE FALSE TRUE FALSE TRUE ...

Prof.Dr. Rozer ANR.Montaño

SEET/UFFR 33
```

```
Ordenação do Data Frame

Ordenação do data frame: sort() e order()

sort() retorna o vetor com os valores ordenados. Default ordem crescente (decreasing=TRUE para ordem decrescente)

sort(pessoas$idade)

[1] 14 16 22 31 44 49 57 72 80 81

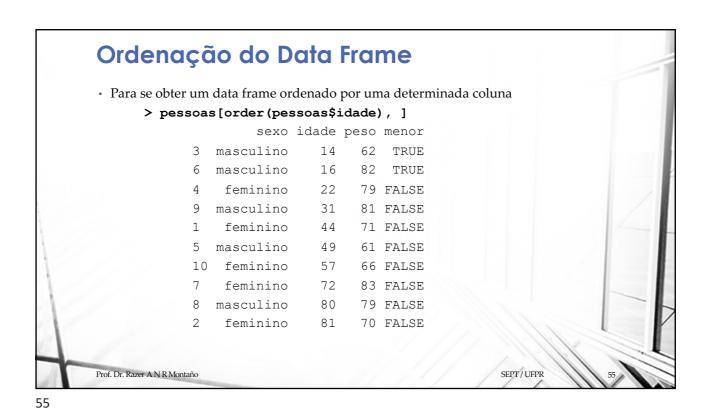
sort(pessoas$idade, decreasing=TRUE)

[1] 81 80 72 57 49 44 31 22 16 14

order() retorna um vetor com as posições dos elementos ordenados, usado para se criar um data frame ordenado. Default ordem crescente (decreasing=TRUE para ordem decrescente)

order(pessoas$idade)

[1] 3 6 4 9 1 5 10 7 8 2
```



Ordenação do Data Frame · Ordenação por duas ou mais colunas > pessoas[order(pessoas\$sexo, pessoas\$idade), ] sexo idade peso menor 22 feminino 79 FALSE feminino 44 71 FALSE 57 10 feminino 66 FALSE 7 feminino 72 83 FALSE 2 81 feminino 70 FALSE masculino 14 62 TRUE 6 masculino 16 82 TRUE masculino 31 81 FALSE 49 masculino 61 FALSE masculino 80 79 FALSE SEPT / UFPR Prof. Dr. Razer A N R Montaño

