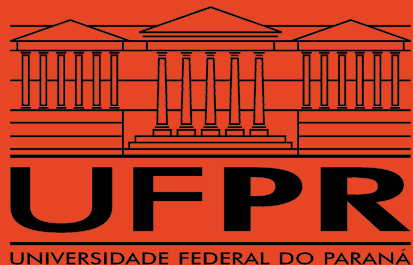


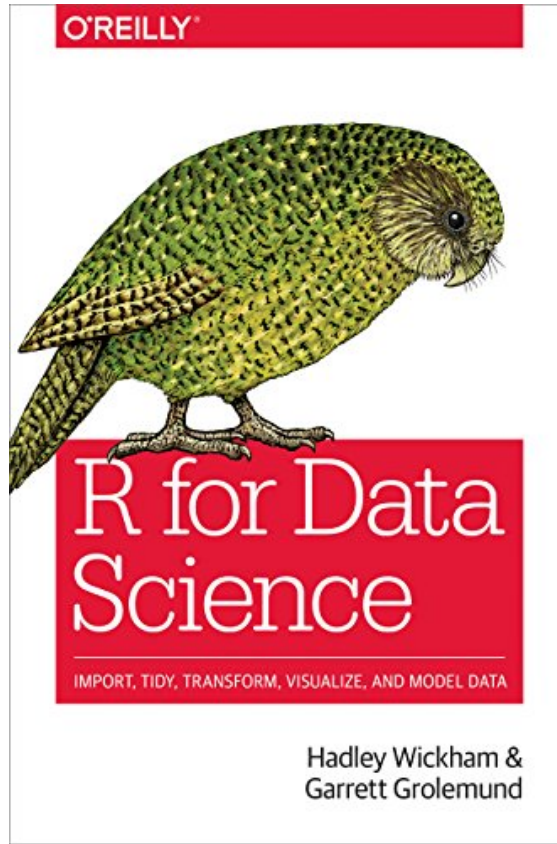
# Conhecendo o Tidyverse

*dplyr*

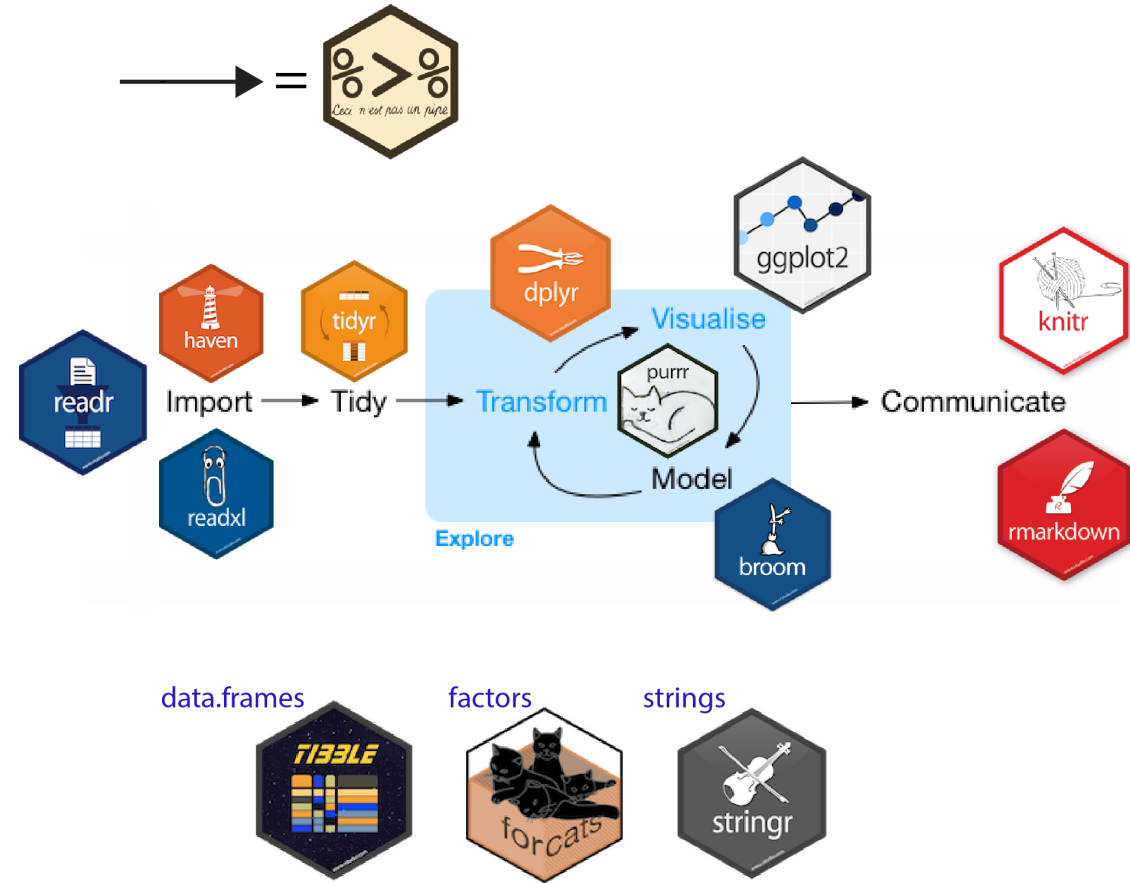
Fernando de Pol Mayer (LEG/DEST/UFPR)  
2022-03-22



# R for Data Science



R for Data Science, a principal referência sobre o emprego da linguagem R em ciência de dados.



Workflow de ciência de dados com o {tidyverse}. Fonte:  
[https://oliviergimenez.github.io/intro\\_tidyverse/#7](https://oliviergimenez.github.io/intro_tidyverse/#7)

{dplyr}

## Um overview do {dplyr}

- ▶ Depois dos dados arrumados, é a hora começar conhecê-los!
- ▶ Começa a fase de **análise exploratória de dados** (AED).
- ▶ Os dados são explorados para:
  - ▶ Conhecer as (propriedades das) variáveis.
  - ▶ Determinar medidas descritivas.
  - ▶ Comparar grupos.
  - ▶ Quantificar relações entre variáveis.
  - ▶ Extrair padrões.
  - ▶ Detectar ameaças e corrigir problemas.
- ▶ AED envolve inúmeras operações.
- ▶ É preciso conhecê-las e ser criativo para aplicar da melhor forma.

# Detalhes do dplyr

- ▶ O dplyr é a **gramática** para manipulação de dados.
- ▶ Tem um conjunto **consistente** de verbos para atuar sobre tabelas.
  - ▶ Verbos: `mutate()`, `select()`, `filter()`, `arrange()`, `summarise()`, `slice()`, `rename()`, etc.
  - ▶ Agrupamento: `group_by()` e `ungroup()`.
  - ▶ Junções: `inner_join()`, `full_join()`, `left_join()` e `right_join()`.
  - ▶ Funções resumo: `n()`, `n_distinct()`, `first()`, `last()`, `nth()`, etc.
  - ▶ E muito mais no cartão de referência: <https://github.com/rstudio/cheatsheets/blob/main/data-transformation.pdf>
- ▶ Documentação:
  - ▶ <https://dplyr.tidyverse.org/>.
  - ▶ <https://r4ds.had.co.nz/relational-data.html>.
  - ▶ <https://cran.r-project.org/package=dplyr>

## Criação de um tibble

	matrícula	nome	curso	prova1	prova2	prova3	faltas
1	256	João	Mat	80	90	80	4
2	487	Vanessa	Mat	75	75	75	4
3	965	Tiago	Est	95	80	75	0
4	125	Luana	Est	70	85	50	8
5	458	Gisele	Est	45	50		16
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30		30	20

Uma tabela com dados fictícios sobre alunos e seus desempenhos.

# Criação de um tibble

## Criação por colunas

```
library(tidyverse)
```

```
# Tabela com alunos do curso de
```

```
# Matemática e de Estatística.
```

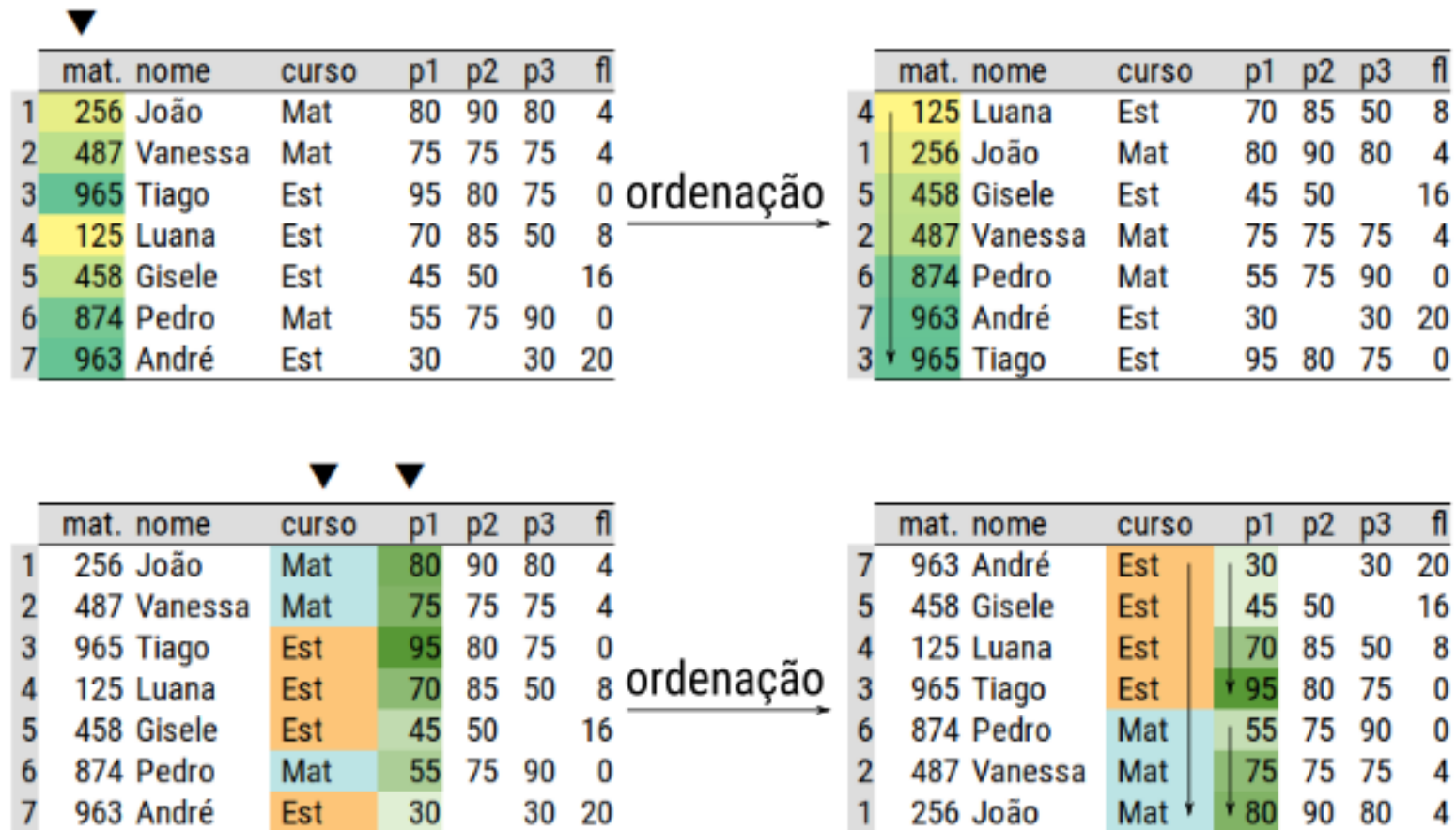
```
df1 <- tibble(  
  matricula = c(256, 487, 965,  
                125, 458, 874, 963),  
  nome = c("João", "Vanessa", "Tiago",  
           "Luana", "Gisele", "Pedro",  
           "André"),  
  curso = c("Mat", "Mat", "Est", "Est",  
            "Est", "Mat", "Est"),  
  prova1 = c(80, 75, 95, 70, 45, 55, 30),  
  prova2 = c(90, 75, 80, 85, 50, 75, NA),  
  prova3 = c(80, 75, 75, 50, NA, 90, 30),  
  faltas = c(4, 4, 0, 8, 16, 0, 20))
```

```
df1
```

```
# # A tibble: 7 × 7
```

#	matricula	nome	curso	prova1	prova2	prova3	faltas
#	<dbl>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
# 1	256	João	Mat	80	90	80	4
# 2	487	Vanessa	Mat	75	75	75	4
# 3	965	Tiago	Est	95	80	75	0
# 4	125	Luana	Est	70	85	50	8
# 5	458	Gisele	Est	45	50	NA	16
# 6	874	Pedro	Mat	55	75	90	0
# 7	963	André	Est	30	NA	30	20

# Ordenação



Ordenação dos registros de uma tabela.



# Ordenação

## Por uma variável

```
df1 %>%  
  arrange(matricula)
```

```
# # A tibble: 7 × 7  
#   matricula nome      curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>  
# 1      125 Luana     Est      70      85      50      8  
# 2      256 João      Mat      80      90      80      4  
# 3      458 Gisele    Est      45      50     NA     16  
# 4      487 Vanessa  Mat      75      75      75      4  
# 5      874 Pedro    Mat      55      75      90      0  
# 6      963 André    Est      30     NA      30     20  
# 7      965 Tiago    Est      95      80      75      0
```

## Por duas ou mais variáveis

```
df1 %>%  
  arrange(curso, desc(prova1))
```

```
# # A tibble: 7 × 7  
#   matricula nome      curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>  
# 1      965 Tiago     Est      95      80      75      0  
# 2      125 Luana     Est      70      85      50      8  
# 3      458 Gisele    Est      45      50     NA     16  
# 4      963 André     Est      30     NA      30     20  
# 5      256 João      Mat      80      90      80      4  
# 6      487 Vanessa  Mat      75      75      75      4  
# 7      874 Pedro    Mat      55      75      90      0
```

`desc()`: ordenação **decrecente** da variável.

# Seleção das variáveis

## Seleção pelos nomes

```
df1 %>%  
  select(nome, prova1, prova2, prova3)
```

```
# # A tibble: 7 × 4  
#   nome      prova1 prova2 prova3  
#   <chr>    <dbl>  <dbl>  <dbl>  
# 1 João      80      90      80  
# 2 Vanessa   75      75      75  
# 3 Tiago     95      80      75  
# 4 Luana     70      85      50  
# 5 Gisele    45      50      NA  
# 6 Pedro     55      75      90  
# 7 André     30      NA      30
```

```
df1 %>% select(c("nome", "prova1"))  
df1 %>% select(-nome, -faltas)  
df1 %>% select(prova1:prova3)
```

## Seleção pela posição

```
df1 %>%  
  select(1:3)
```

```
# # A tibble: 7 × 3  
#   matricula nome      curso  
#   <dbl> <chr>    <chr>  
# 1      256 João      Mat  
# 2      487 Vanessa  Mat  
# 3      965 Tiago     Est  
# 4      125 Luana     Est  
# 5      458 Gisele    Est  
# 6      874 Pedro     Mat  
# 7      963 André     Est
```

```
df1 %>% select(1, 4)  
df1 %>% select(-1, -4)
```

# Seleção das variáveis

## Seleção de variáveis por condição

```
df1 %>%  
  select(where(is.numeric))
```

```
# # A tibble: 7 × 5  
#   matricula prova1 prova2 prova3 faltas  
#   <dbl> <dbl> <dbl> <dbl> <dbl>  
# 1      256      80      90      80      4  
# 2      487      75      75      75      4  
# 3      965      95      80      75      0  
# 4      125      70      85      50      8  
# 5      458      45      50      NA     16  
# 6      874      55      75      90      0  
# 7      963      30      NA      30     20
```

```
df1 %>%  
  select(!where(is.numeric))
```

## Seleção por expressão regular

```
df1 %>%  
  select(matches("^prova"))
```

```
# # A tibble: 7 × 3  
#   prova1 prova2 prova3  
#   <dbl> <dbl> <dbl>  
# 1      80      90      80  
# 2      75      75      75  
# 3      95      80      75  
# 4      70      85      50  
# 5      45      50      NA  
# 6      55      75      90  
# 7      30      NA      30
```

```
## Nomes que terminam em dígitos  
df1 %>%  
  select(matches("\\d$"))  
## Todos os nomes com 6 caracteres  
df1 %>%  
  select(matches("^.{6}$"))
```

```
## Outros  
df1 %>%  
  select(starts_with("pr"))  
df1 %>%  
  select(ends_with("a"))  
df1 %>%  
  select(contains("urso"))
```

# Fatiamento por linhas

## Fatiamento

```
df1 %>%  
  slice(3:5)
```

```
# # A tibble: 3 × 7  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     965 Tiago   Est    95    80    75     0  
# 2     125 Luana   Est    70    85    50     8  
# 3     458 Gisele  Est    45    50    NA    16
```

```
df1 %>%  
  slice(-(3:5))  
df1 %>%  
  slice(c(3:4, 1:2))
```

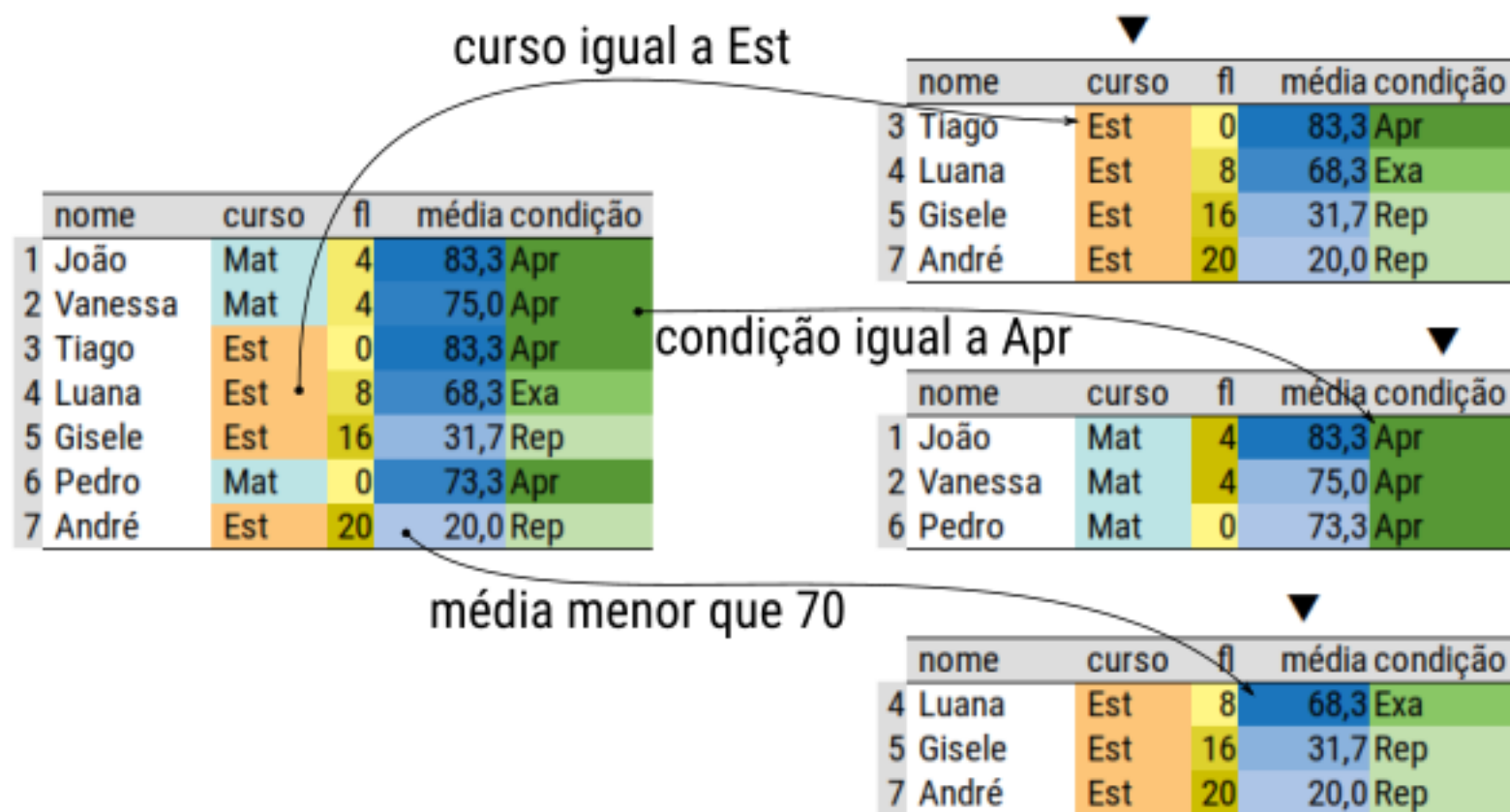
## Extremidades

```
# Primeiras linhas (cabeça).  
df1 %>%  
  head(n = 4)
```

```
# # A tibble: 4 × 7  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     256 João   Mat    80    90    80     4  
# 2     487 Vanessa Mat    75    75    75     4  
# 3     965 Tiago   Est    95    80    75     0  
# 4     125 Luana   Est    70    85    50     8
```

```
# Últimas linhas (cauda).  
df1 %>%  
  tail(n = 4)
```

# Filtros



Filtro dos registros de uma tabela.

# Filtros

## Usando uma variável

```
df1 %>%  
  filter(curso == "Est")
```

```
# # A tibble: 4 × 7  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>   <chr>   <dbl>   <dbl>   <dbl>   <dbl>  
# 1     965 Tiago   Est      95      80      75      0  
# 2     125 Luana   Est      70      85      50      8  
# 3     458 Gisele  Est      45      50      NA     16  
# 4     963 André  Est      30      NA      30     20
```

```
df1 %>%  
  filter(faltas == 0)  
df1 %>%  
  filter(faltas != 0)  
df1 %>%  
  filter(nome %in% c("Luana", "Vanessa"))  
## Seleciona colunas que possuem pelo menos um NA  
df1 %>%  
  select(function(x) any(is.na(x)))
```

## Usando duas ou mais

```
df1 %>%  
  ## select(curso, prova1:prova3) %>%  
  filter(curso == "Est",  
         (prova1 + prova2 + prova3)/3 > 70)
```

```
# # A tibble: 1 × 7  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>   <chr>   <dbl>   <dbl>   <dbl>   <dbl>  
# 1     965 Tiago   Est      95      80      75      0
```

```
## Note a diferença  
df1 %>%  
  filter(faltas == 0 & curso == "Est")  
df1 %>%  
  filter(faltas == 0, curso == "Est")
```

```
## Para reproduzir a figura  
df2 <- df1 %>%  
  mutate(media = (prova1 + prova2 + prova3)/3,  
         condicao = ifelse(media >= 70, "Apr", "Rep"))  
  
df2  
df2 %>%  
  filter(condicao == "Apr")  
df2 %>%  
  filter(media < 70)  
df2 %>%  
  filter(condicao == "Apr" & curso == "Est")
```

# Amostragem das observações

## Amostra de $n$ elementos

```
# Amostra aleatória das linhas.  
df1 %>%  
  sample_n(size = 3, replace = FALSE)
```

```
# # A tibble: 3 × 7  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     256 João    Mat     80     90     80     4  
# 2     874 Pedro   Mat     55     75     90     0  
# 3     458 Gisele Est      45     50     NA    16
```

## Amostra de uma fração $p$

```
# Amostra aleatória das linhas.  
df1 %>%  
  sample_frac(size = 0.5, replace = FALSE)
```

```
# # A tibble: 4 × 7  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     487 Vanessa Mat      75     75     75     4  
# 2     874 Pedro   Mat     55     75     90     0  
# 3     256 João    Mat     80     90     80     4  
# 4     125 Luana   Est      70     85     50     8
```

## Renomeação

	matrícula	nome	curso	prova1	prova2	prova3	faltas
1	256	João	Mat	80	90	80	4
2	487	Vanessa	Mat	75	75	75	4
3	965	Tiago	Est	95	80	75	0
4	125	Luana	Est	70	85	50	8
5	458	Gisele	Est	45	50		16
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30		30	20

encurtar      substituir      abreviar

	mat.	nome	curso	p1	p2	p3	fl
1	256	João	Mat	80	90	80	4
2	487	Vanessa	Mat	75	75	75	4
3	965	Tiago	Est	95	80	75	0
4	125	Luana	Est	70	85	50	8
5	458	Gisele	Est	45	50		16
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30		30	20

Formas de renomear as colunas de uma tabela.



# Renomeação

## Via pares de substituição

```
names(df1)
```

```
# [1] "matricula" "nome"      "curso"     "prova1"
# [5] "prova2"    "prova3"    "faltas"
```

```
# Renomeia nomes de colunas (variáveis).
```

```
df1 %>%
  rename("mat." = "matricula",
         "fl" = "faltas",
         "nm" = 2)
```

```
# # A tibble: 7 × 7
#   mat. nm      curso prova1 prova2 prova3 fl
#   <dbl> <chr>   <chr>   <dbl> <dbl> <dbl> <dbl>
# 1   256 João    Mat       80     90     80     4
# 2   487 Vanessa Mat       75     75     75     4
# 3   965 Tiago   Est       95     80     75     0
# 4   125 Luana   Est       70     85     50     8
# 5   458 Gisele  Est       45     50    NA    16
# 6   874 Pedro   Mat       55     75     90     0
# 7   963 André   Est       30    NA     30    20
```

## Função de transformação de *strings*

```
df1 %>%
  rename_with(.fn = str_to_upper,
             .cols = prova1:prova3) %>%
  head(n = 2)
```

```
# # A tibble: 2 × 7
#   matricula nome      curso PROVA1 PROVA2 PROVA3 faltas
#   <dbl> <chr>   <chr>   <dbl> <dbl> <dbl> <dbl>
# 1     256 João    Mat       80     90     80     4
# 2     487 Vanessa Mat       75     75     75     4
```

```
df1 %>%
  rename_with(.fn = abbreviate, minlength = 2,
             .cols = prova1:faltas) %>%
  head(n = 2)
```

```
# # A tibble: 2 × 7
#   matricula nome      curso  p1    p2    p3    fl
#   <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <dbl>
# 1     256 João    Mat     80    90    80     4
# 2     487 Vanessa Mat     75    75    75     4
```

```
apropos("rename_")
```

```
# [1] "rename_"      "rename_all"   "rename_at"
# [4] "rename_if"    "rename_vars"  "rename_vars_"
# [7] "rename_with"
```

# Realocação de colunas

## Realocação pelos nomes

```
df1 %>%  
  relocate(curso) %>%  
  head(n = 2)
```

```
# # A tibble: 2 × 7  
#   curso matricula nome   prova1 prova2 prova3 faltas  
#   <chr>      <dbl> <chr>   <dbl>  <dbl>  <dbl>  <dbl>  
# 1 Mat         256 João     80     90     80     4  
# 2 Mat         487 Vanessa  75     75     75     4
```

```
df1 %>%  
  relocate(prova1:faltas) %>%  
  head(n = 2)
```

```
# # A tibble: 2 × 7  
#   prova1 prova2 prova3 faltas matricula nome   curso  
#   <dbl>  <dbl>  <dbl>  <dbl>      <dbl> <chr>  <chr>  
# 1     80     90     80     4         256 João   Mat  
# 2     75     75     75     4         487 Vanessa Mat
```

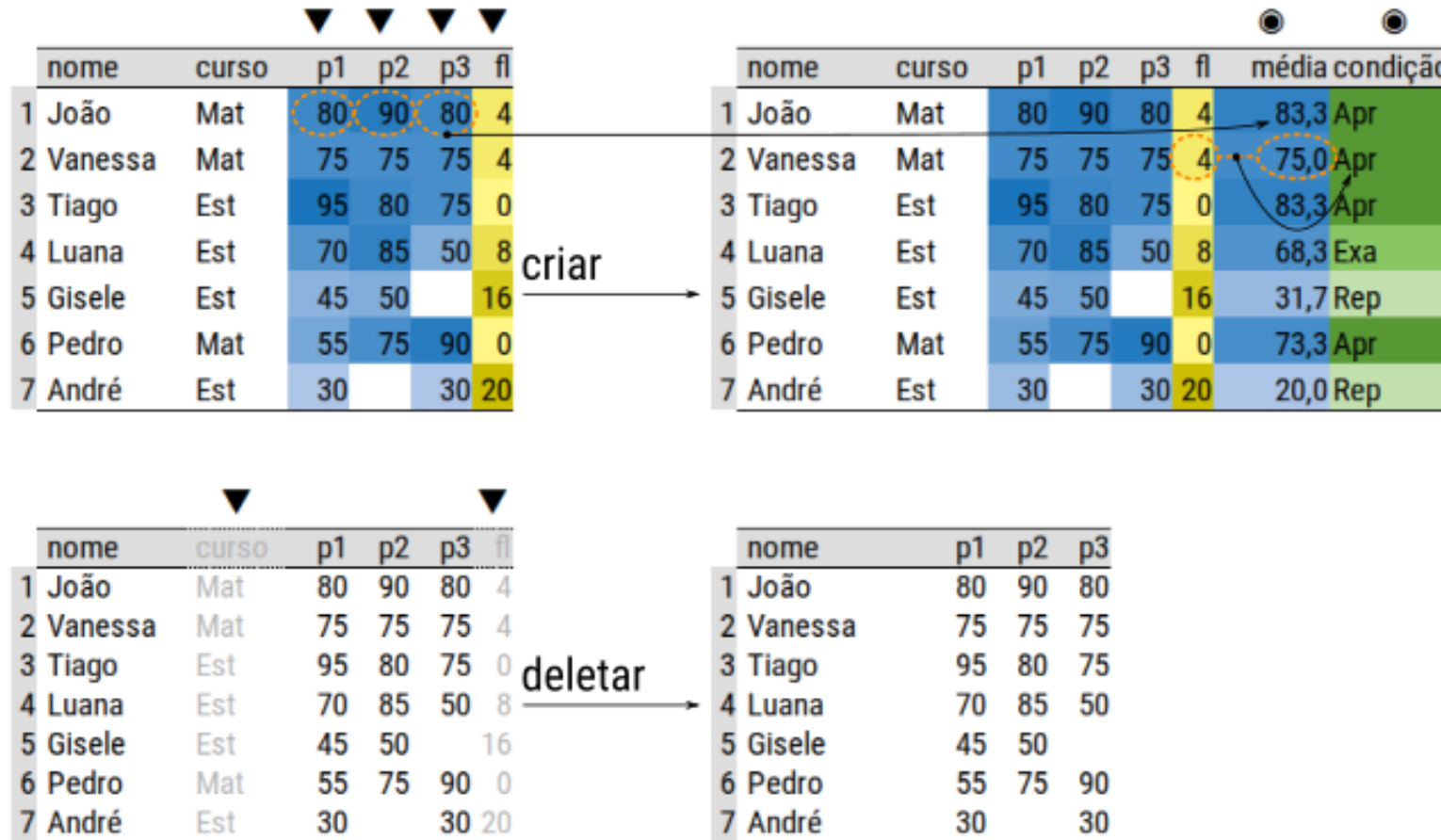
```
df1 %>%  
  relocate(matricula, nome,  
            .after = curso) %>%  
  head(n = 2)
```

```
# # A tibble: 2 × 7  
#   curso matricula nome   prova1 prova2 prova3 faltas  
#   <chr>      <dbl> <chr>   <dbl>  <dbl>  <dbl>  <dbl>  
# 1 Mat         256 João     80     90     80     4  
# 2 Mat         487 Vanessa  75     75     75     4
```

```
df1 %>%  
  relocate(matricula, nome,  
            .after = last_col()) %>%  
  head(n = 2)
```

```
# # A tibble: 2 × 7  
#   curso prova1 prova2 prova3 faltas matricula nome  
#   <chr>  <dbl>  <dbl>  <dbl>  <dbl>      <dbl> <chr>  
# 1 Mat     80     90     80     4         256 João  
# 2 Mat     75     75     75     4         487 Vanessa
```

# Transformações



Criação e deleção de variáveis em uma tabela.

# Transformações

As operações podem modificar a tabela com a:

1. **Criação** de novas variáveis.
2. **Remoção** de variáveis.
3. **Transformação** de variáveis.

As operações de criação/transformação podem ser:

1. **Matemáticas**: aritméticas, potência, logarítmicas, trigonométricas, etc.
2. **Compartimentação** (*binning*): agrupar em classes.
3. **Conversão** de tipo de valor: i.e. de `int` → `str`.
4. **Substituição**: i.e. preencher um valor ausente.

# Transformações

## Criação de variável

```
df1 %>%  
  mutate(media = (prova1 + prova2 + prova3)/3) %>%  
  select(prova1:prova3, media)
```

```
# # A tibble: 7 × 4  
#   prova1 prova2 prova3 media  
#   <dbl> <dbl> <dbl> <dbl>  
# 1     80     90     80  83.3  
# 2     75     75     75   75  
# 3     95     80     75  83.3  
# 4     70     85     50  68.3  
# 5     45     50    NA   NA  
# 6     55     75     90  73.3  
# 7     30    NA     30   NA
```

*## Usando operações por LINHA (rowwise)*

```
df1 %>%  
  rowwise() %>%  
  mutate(media = mean(c(prova1, prova2, prova3))) %>%  
  select(prova1:prova3, media)
```

*## Usando c\_across*

```
df1 %>%  
  rowwise() %>%  
  mutate(media = mean(c_across(prova1:prova3))) %>%  
  select(prova1:prova3, media)
```

## Transformações em várias variáveis

```
df1 %>%  
  mutate(across(prova1:prova3,  
    ~replace_na(., 0)))
```

```
# # A tibble: 7 × 7  
#   matricula nome      curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     256 João      Mat      80     90     80     4  
# 2     487 Vanessa  Mat      75     75     75     4  
# 3     965 Tiago    Est      95     80     75     0  
# 4     125 Luana    Est      70     85     50     8  
# 5     458 Gisele   Est      45     50      0    16  
# 6     874 Pedro    Mat      55     75     90     0  
# 7     963 André    Est      30      0     30    20
```

*## São abreviações de*

```
df1 %>%  
  replace_na(list(prova1 = 0, prova2 = 0, prova3 = 0))
```

# Transformações

## Transformações dado uma condição

```
## Passa para caixa alta.
```

```
df1 %>%  
  mutate(across(where(is.character), str_to_upper))
```

```
# # A tibble: 7 × 7
```

#	matricula	nome	curso	prova1	prova2	prova3	faltas
#	<dbl>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
# 1	256	JOÃO	MAT	80	90	80	4
# 2	487	VANESSA	MAT	75	75	75	4
# 3	965	TIAGO	EST	95	80	75	0
# 4	125	LUANA	EST	70	85	50	8
# 5	458	GISELE	EST	45	50	NA	16
# 6	874	PEDRO	MAT	55	75	90	0
# 7	963	ANDRÉ	EST	30	NA	30	20

```
df1 %>% mutate(across(where(is.numeric), sqrt))  
df1 %>% mutate(across(where(is.numeric), log))  
df1 %>% mutate(across(where(is.character), as.factor))  
df1 %>% mutate(across(where(is.numeric), as.integer))
```

## Transformações dado uma expressão regular

```
## Divide a nota por 10.
```

```
df1 %>%  
  mutate(across(starts_with("prova"), ~ ./10))
```

```
# # A tibble: 7 × 7
```

#	matricula	nome	curso	prova1	prova2	prova3	faltas
#	<dbl>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
# 1	256	João	Mat	8	9	8	4
# 2	487	Vanessa	Mat	7.5	7.5	7.5	4
# 3	965	Tiago	Est	9.5	8	7.5	0
# 4	125	Luana	Est	7	8.5	5	8
# 5	458	Gisele	Est	4.5	5	NA	16
# 6	874	Pedro	Mat	5.5	7.5	9	0
# 7	963	André	Est	3	NA	3	20

# Transformações

## Cortar valores em classe

```
# Intervalos para corte e rótulos.
inter <- c(0, 40, 70, 100)
condi <- c("reprovado", "exame", "aprovado")

tb_final <- df1 %>%
  mutate(across(starts_with("prova"),
    ~replace_na(., 0))) %>%
  mutate(media = (prova1 + prova2 + prova3)/3,
    result= cut(media,
      breaks = inter,
      labels = condi,
      right = FALSE,
      include.lowest = TRUE))
```

```
tb_final %>%
  select(nome, curso,
    media, result)
```

# # A tibble: 7 × 4

#	nome	curso	media	result
#	<chr>	<chr>	<dbl>	<fct>
# 1	João	Mat	83.3	aprovado
# 2	Vanessa	Mat	75	aprovado
# 3	Tiago	Est	83.3	aprovado
# 4	Luana	Est	68.3	exame
# 5	Gisele	Est	31.7	reprovado
# 6	Pedro	Mat	73.3	aprovado
# 7	André	Est	20	reprovado

# Transformações

## Remoção de variáveis

```
tb_final %>%  
  mutate(media = NULL,  
         result = NULL)
```

```
# # A tibble: 7 × 7  
#   matricula nome      curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     256 João      Mat      80     90     80     4  
# 2     487 Vanessa  Mat      75     75     75     4  
# 3     965 Tiago    Est      95     80     75     0  
# 4     125 Luana    Est      70     85     50     8  
# 5     458 Gisele   Est      45     50      0    16  
# 6     874 Pedro    Mat      55     75     90     0  
# 7     963 André    Est      30      0     30    20
```

```
tb_final$media <- NULL  
tb_final$result <- NULL  
tb_final
```

```
# # A tibble: 7 × 7  
#   matricula nome      curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     256 João      Mat      80     90     80     4  
# 2     487 Vanessa  Mat      75     75     75     4  
# 3     965 Tiago    Est      95     80     75     0  
# 4     125 Luana    Est      70     85     50     8  
# 5     458 Gisele   Est      45     50      0    16  
# 6     874 Pedro    Mat      55     75     90     0  
# 7     963 André    Est      30      0     30    20
```



# Rearranjo

- ▶ São operações de *reshaping* da tabela.
- ▶ Modificam a disposição dos registros.
  - ▶ Empilhar ou amontoar um conjunto de variáveis.
  - ▶ Desempilhar ou esparramar os níveis de uma variável.

	nome	curso	p1	p2	p3	fi
1	João	Mat	80	90	80	4
2	Vanessa	Mat	75	75	75	4
3	Tiago	Est	95	80	75	0
4	Luana	Est	70	85	50	8
5	Gisele	Est	45	50		16
6	Pedro	Mat	55	75	90	0
7	André	Est	30		30	20

	nome	curso	exame	nota
1	João	Mat	p1	80
2	Vanessa	Mat	p1	75
3	Tiago	Est	p1	95
4	Luana	Est	p1	70
5	Gisele	Est	p1	45
6	Pedro	Mat	p1	55
7	André	Est	p1	30

	nome	curso	exame	nota
1	João	Mat	p2	90
2	Vanessa	Mat	p2	75
3	Tiago	Est	p2	80
4	Luana	Est	p2	85
5	Gisele	Est	p2	50
6	Pedro	Mat	p2	75
7	André	Est	p2	

	nome	curso	exame	nota
1	João	Mat	p3	80
2	Vanessa	Mat	p3	75
3	Tiago	Est	p3	75
4	Luana	Est	p3	50
5	Gisele	Est	p3	
6	Pedro	Mat	p3	90
7	André	Est	p3	30

Modificação da disposição com empilhamento.

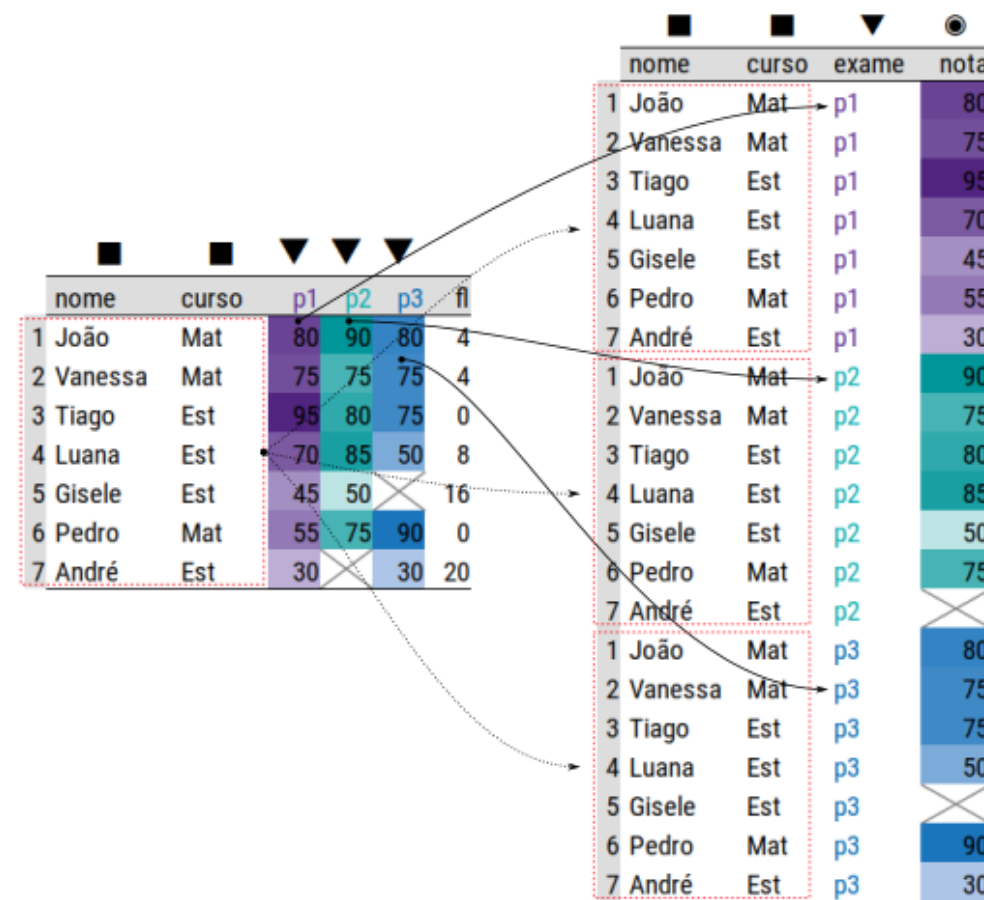
# Rearranjo · Empilhar

```
# Empilhar.  
u <- df1 %>%  
  pivot_longer(cols = prova1:prova3,  
               names_to = "exame",  
               values_to = "nota")  
  
head(u) %>%  
  select(nome, curso, exame, nota)
```

```
# # A tibble: 6 × 4  
#   nome    curso exame  nota  
#   <chr>  <chr> <chr> <dbl>  
# 1 João   Mat   prova1    80  
# 2 João   Mat   prova2    90  
# 3 João   Mat   prova3    80  
# 4 Vanessa Mat   prova1    75  
# 5 Vanessa Mat   prova2    75  
# 6 Vanessa Mat   prova3    75
```

```
dim(u)
```

```
# [1] 21  6
```



Modificação da disposição com empilhamento.

# Rearranjo · Esparramar

```
# Spread = esparramar.
```

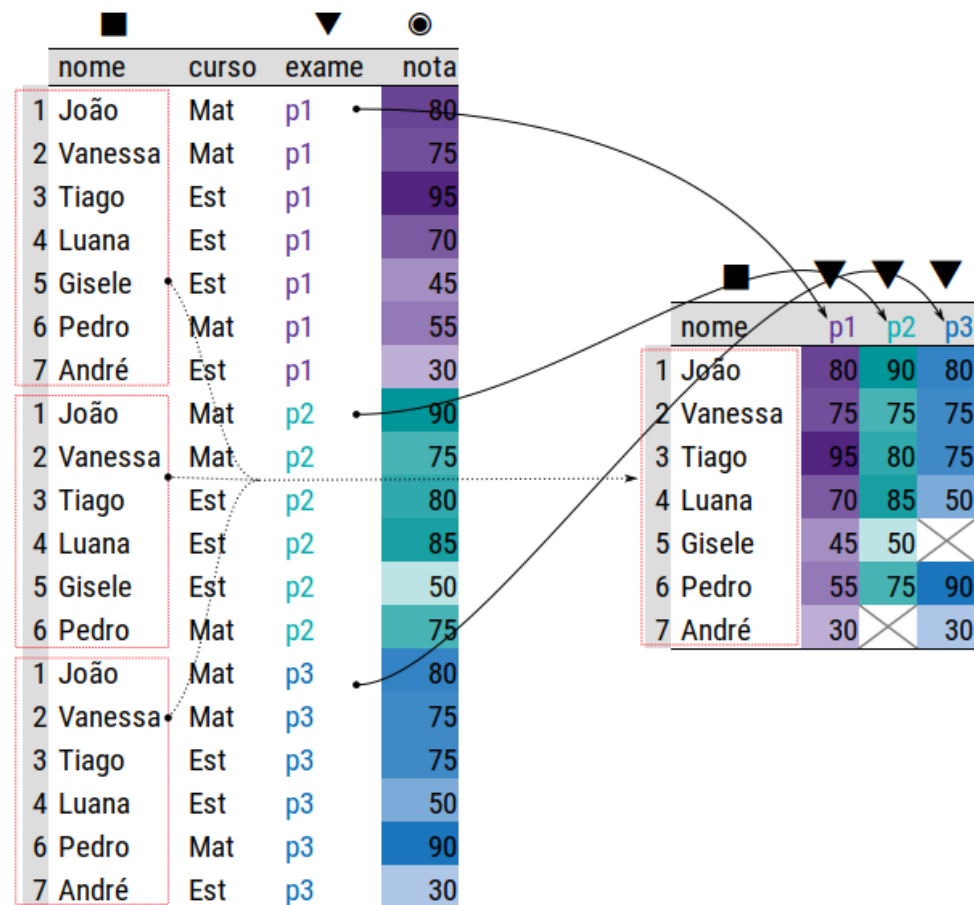
```
v <- u %>%
```

```
  pivot_wider(names_from = "exame",  
              values_from = "nota")
```

```
head(v)
```

```
# # A tibble: 6 × 7
```

```
#   matricula nome      curso faltas prova1 prova2 prova3  
#   <dbl> <chr>    <chr>  <dbl>  <dbl>  <dbl>  <dbl>  
# 1     256 João      Mat      4      80     90     80  
# 2     487 Vanessa    Mat      4      75     75     75  
# 3     965 Tiago      Est      0      95     80     75  
# 4     125 Luana      Est      8      70     85     50  
# 5     458 Gisele     Est     16      45     50     NA  
# 6     874 Pedro      Mat      0      55     75     90
```



Modificação da disposição com desempilhamento.

# Medidas resumo

- ▶ Operações para determinar estatísticas descritivas.
  - ▶ Soma, média, mediana, quartis, quantis, etc.
  - ▶ Variância, desvio-padrão, amplitude, desvio absoluto da mediana, coeficiente de variação, etc.
  - ▶ Número de níveis distintos, frequências absolutas/relativas, etc.
- ▶ Elas podem ser marginais ou considerar a estratificação conforme uma ou mais variáveis categóricas.
- ▶ Podem ser aplicadas em todas as variáveis de um mesmo tipo (homogêneo).

		▼ ▼ ▼					
	nome	curso	p1	p2	p3	fl	
1	João	Mat	80	90	80	4	média 83,3 desvio 5,8
2	Vanessa	Mat	75	75	75	4	75,0 0,0
3	Tiago	Est	95	80	75	0	83,3 10,4
4	Luana	Est	70	85	50	8	68,3 17,6
5	Gisele	Est	45	50		16	47,5 3,5
6	Pedro	Mat	55	75	90	0	73,3 17,6
7	André	Est	30		30	20	30,0 0,0

	p1	p2	p3
média	64,3	75,8	66,7
desvio	22,3	13,9	22,3

Cálculo de medidas resumo.

# Medidas resumo

## Uma estatística

```
# Medidas resumo de uma estatística.  
df1 %>%  
  summarise(soma = sum(prova1),  
            media = mean(prova1),  
            max = max(prova1),  
            sd = sd(prova1))
```

```
# # A tibble: 1 × 4  
#   soma media   max    sd  
#   <dbl> <dbl> <dbl> <dbl>  
# 1   450  64.3    95  22.3
```

## Um vetor de estatísticas

```
# Medidas resumo de um vetor de estatísticas.  
quantile(df1$prova1, probs = c(0.25, 0.5, 0.75))
```

```
# 25% 50% 75%  
# 50.0 70.0 77.5
```

```
fivenum(df1$prova1)
```

```
# [1] 30.0 50.0 70.0 77.5 95.0
```

```
table(df1$curso)
```

```
#  
# Est Mat  
#    4    3
```

# Medidas resumo

## Estatísticas definidas pelo usuário

```
df1 %>%  
  summarise(CV = 100 * sd(prova1)/mean(prova1))
```

```
# Define uma função para facilitar.
```

```
CV <- function(x, ...) {  
  100 * sd(x, ...)/mean(x, ...)  
}
```

```
df1 %>%  
  summarise(across(starts_with("prova"),  
                    CV, na.rm = TRUE))
```

```
# # A tibble: 1 × 3  
#   prova1 prova2 prova3  
#   <dbl> <dbl> <dbl>  
# 1   34.6   18.4   33.4
```

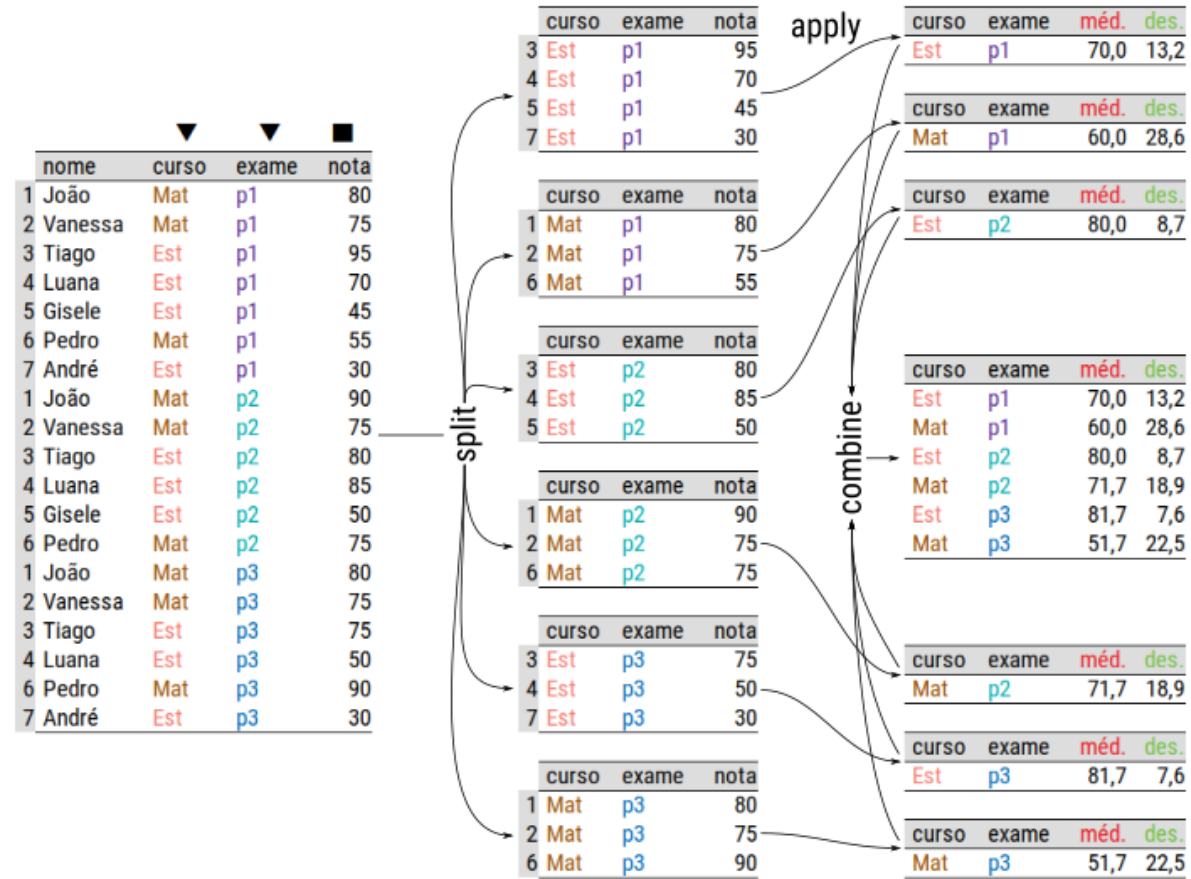
## Várias estatísticas para várias variáveis

```
df1 %>%  
  summarise(across(prova1:prova3,  
                    c(mean, sd), na.rm = TRUE))
```

```
# # A tibble: 1 × 6  
#   prova1_1 prova1_2 prova2_1 prova2_2 prova3_1 prova3_2  
#   <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>  
# 1    64.3    22.3    75.8    13.9    66.7    22.3
```

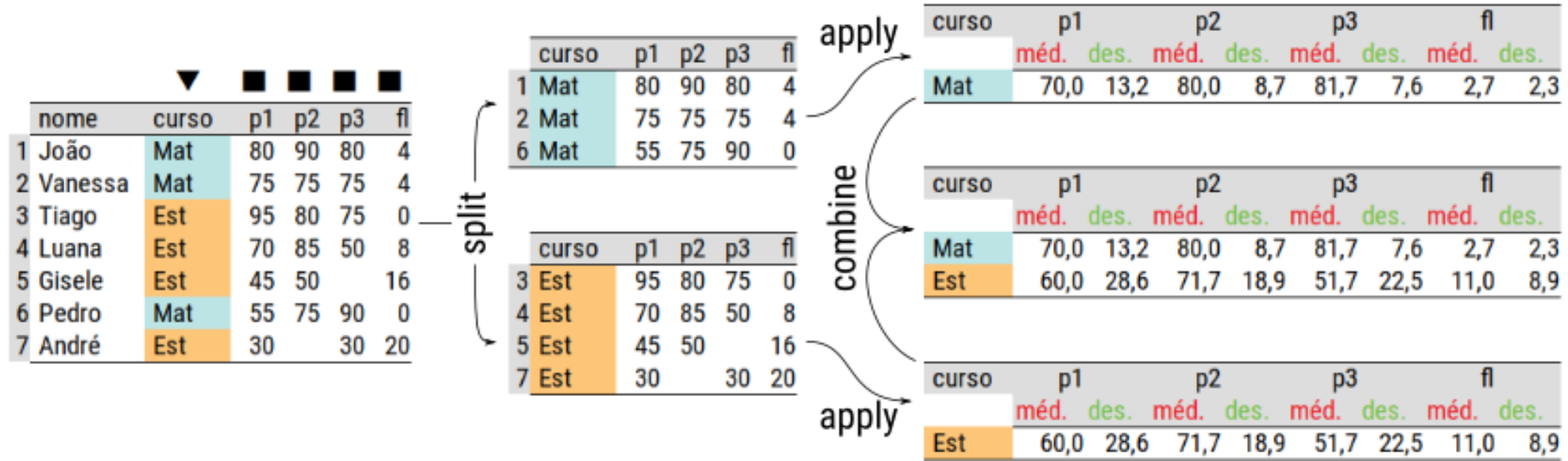
# Agregação

- Consiste em aplicar estatísticas em variáveis fazendo a extratificação por outras variáveis.
- São tarefas conhecidas como *split-apply-combine*.
- Ou também chamadas de **GROUP BY**.



Agregação de uma tabela.

# Agregação



Agregação de uma tabela.



# Agregação

## Cálculo de frequências

```
# Registros por curso.
```

```
df1 %>%  
  count(curso)
```

```
# # A tibble: 2 × 2
```

```
#   curso      n
```

```
#   <chr> <int>
```

```
# 1 Est      4
```

```
# 2 Mat      3
```

```
## Aprovados/reprovados por curso
```

```
df1 %>%  
  mutate(across(prova1:prova3, ~replace_na(., 0))) %>%  
  mutate(media = (prova1 + prova2 + prova3)/3,  
         cond = ifelse(media >= 70, "Apr", "Rep")) %>%  
  count(curso, cond)
```

```
# # A tibble: 3 × 3
```

```
#   curso cond      n
```

```
#   <chr> <chr> <int>
```

```
# 1 Est   Apr      1
```

```
# 2 Est   Rep      3
```

```
# 3 Mat   Apr      3
```

# Agregação

## Medidas descritivas resumo

```
## Nota média por curso em cada avaliação.  
df1 %>%  
  group_by(curso) %>%  
  summarise(across(prova1:prova3, mean, na.rm = TRUE))
```

```
# # A tibble: 2 × 4  
#   curso prova1 prova2 prova3  
#   <chr>   <dbl>   <dbl>   <dbl>  
# 1 Est      60    71.7    51.7  
# 2 Mat      70    80      81.7
```

```
## Média final por curso.  
df1 %>%  
  mutate(across(prova1:prova3, ~replace_na(., 0))) %>%  
  mutate(media = (prova1 + prova2 + prova3)/3) %>%  
  group_by(curso) %>%  
  summarise(med = mean(media), sd = sd(media))
```

```
# # A tibble: 2 × 3  
#   curso med    sd  
#   <chr> <dbl> <dbl>  
# 1 Est   50.8  29.9  
# 2 Mat   77.2   5.36
```

# Agregação

## Usando o formato longo

```
df1 <- df1 %>%
  pivot_longer(cols = prova1:prova3,
               names_to = "prova",
               values_to = "nota") %>%
  replace_na(list(nota = 0))
head(df1, n = 4)
```

```
# # A tibble: 4 × 6
#   matricula nome      curso faltas prova  nota
#   <dbl> <chr>    <chr>   <dbl> <chr>  <dbl>
# 1     256 João      Mat        4 prova1    80
# 2     256 João      Mat        4 prova2    90
# 3     256 João      Mat        4 prova3    80
# 4     487 Vanessa Mat        4 prova1    75
```

```
df1 %>%
  group_by(prova, curso) %>%
  summarise(media = mean(nota), dp = sd(nota))
```

```
# # A tibble: 6 × 4
# # Groups:   prova [3]
#   prova curso media  dp
#   <chr> <chr> <dbl> <dbl>
# 1 prova1 Est    60  28.6
# 2 prova1 Mat    70  13.2
# 3 prova2 Est   53.8 39.0
# 4 prova2 Mat    80   8.66
# 5 prova3 Est   38.8 31.7
# 6 prova3 Mat   81.7  7.64
```

```
df1 %>%
  group_by(nome) %>%
  summarise(media = mean(nota), dp = sd(nota),
            faltas = unique(faltas),
            cond = ifelse(media > 70, "Apr", "Rep"))
```

```
# # A tibble: 7 × 5
#   nome      media  dp faltas cond
#   <chr>    <dbl> <dbl> <dbl> <chr>
# 1 André      20  17.3     20 Rep
# 2 Gisele    31.7 27.5     16 Rep
# 3 João     83.3  5.77      4 Apr
# 4 Luana     68.3 17.6      8 Rep
# 5 Pedro     73.3 17.6      0 Apr
# 6 Tiago     83.3 10.4      0 Apr
# 7 Vanessa    75   0        4 Apr
```

# Agregação

- ▶ Uma vez que uma tabela é agrupada, várias informações e métodos estão disponíveis.
- ▶ Isso permite criar uma lista de tabelas para operar com programação funcional.

```
## Cria uma tabela de dados agrupados.
```

```
u <- df1 %>%  
  group_by(curso)  
u
```

```
# # A tibble: 7 × 7  
# # Groups:   curso [2]  
#   matricula nome    curso prova1 prova2 prova3 faltas  
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>  
# 1      256 João     Mat      80     90     80     4  
# 2      487 Vanessa  Mat      75     75     75     4  
# 3      965 Tiago    Est      95     80     75     0  
# 4      125 Luana    Est      70     85     50     8  
# 5      458 Gisele   Est      45     50    NA     16  
# 6      874 Pedro    Mat      55     75     90     0  
# 7      963 André    Est      30    NA     30    20
```

```
## Inspecciona os métodos disponíveis.
```

```
class(u)  
methods(class = "grouped_df")  
## Usa alguns dos métodos.  
n_groups(u)  
group_vars(u)  
group_size(u)  
group_indices(u)
```

```
## Cria uma lista de tabelas.
```

```
u <- df1 %>%  
  group_split(curso)  
glimpse(u)
```

```
# list<tibble[,7]> [1:2]  
# $ : tibble [4 × 7] (S3: tbl_df/tbl/data.frame)  
# $ : tibble [3 × 7] (S3: tbl_df/tbl/data.frame)  
# @ ptype: tibble [0 × 7] (S3: tbl_df/tbl/data.frame)
```

# Concatenação

- ▶ A concatenação permite adicionar novas observações a uma tabela ou novas variáveis.
- ▶ Seja por linha ou colunas, entradas com NA são criadas para os índices que não foram especificados.

	mat.	nome	p1	p2	p3	fl
1	256	João	80	90	80	4
2	487	Vanessa	75	75	75	4
3	965	Tiago	95	80	75	0
4	125	Luana	70	85	50	8
5	458	Gisele	45	50		16
6	874	Pedro	55	75	90	0
7	963	André	30		30	20

	mat.	nome	p1	p2	fl
1	505	Bia	65	85	0
2	658	Carlos	75	80	2
3	713	Cris	75	90	2

	mat.	nome	p1	p2	p3	fl
1	256	João	80	90	80	4
2	487	Vanessa	75	75	75	4
3	965	Tiago	95	80	75	0
4	125	Luana	70	85	50	8
5	458	Gisele	45	50		16
6	874	Pedro	55	75	90	0
7	963	André	30		30	20
8	505	Bia	65	85		0
9	658	Carlos	75	80		2
10	713	Cris	75	90		2

Concatenação de duas tabelas.

# Concatenação

## De linhas (vertical)

```
## Concatenação na vertical (pilha)
df2 <- tibble(
  matricula = c(505, 658, 713),
  nome = c("Bia", "Carlos", "Cris"),
  prova1 = c(65, 75, 75),
  prova2 = c(85, 80, 90),
  faltas = c(0, 2, 2))
bind_rows(df1, df2)
```

```
# # A tibble: 10 × 7
#   matricula nome      curso prova1 prova2 prova3 faltas
#   <dbl> <chr>    <chr> <dbl> <dbl> <dbl> <dbl>
# 1     256 João      Mat      80     90     80     4
# 2     487 Vanessa    Mat      75     75     75     4
# 3     965 Tiago      Est      95     80     75     0
# 4     125 Luana      Est      70     85     50     8
# 5     458 Gisele     Est      45     50     NA    16
# 6     874 Pedro      Mat      55     75     90     0
# 7     963 André      Est      30     NA     30    20
# 8     505 Bia        <NA>     65     85     NA     0
# 9     658 Carlos     <NA>     75     80     NA     2
# 10    713 Cris       <NA>     75     90     NA     2
```

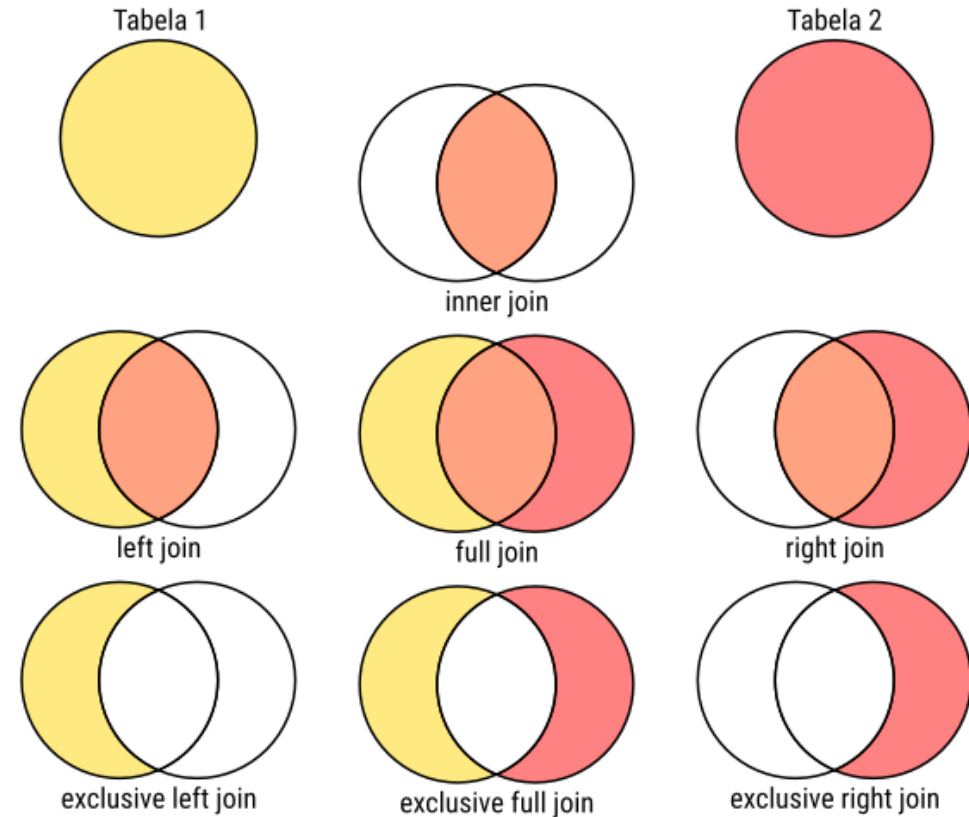
## De colunas (horizontal)

```
## Concatenação na horizontal (fila)
bind_cols(df1[, c(1:3)],
          df1[, c(6:7)])
```

```
# # A tibble: 7 × 5
#   matricula nome      curso prova3 faltas
#   <dbl> <chr>    <chr> <dbl> <dbl>
# 1     256 João      Mat      80     4
# 2     487 Vanessa    Mat      75     4
# 3     965 Tiago      Est      75     0
# 4     125 Luana      Est      50     8
# 5     458 Gisele     Est      NA    16
# 6     874 Pedro      Mat      90     0
# 7     963 André      Est      30    20
```

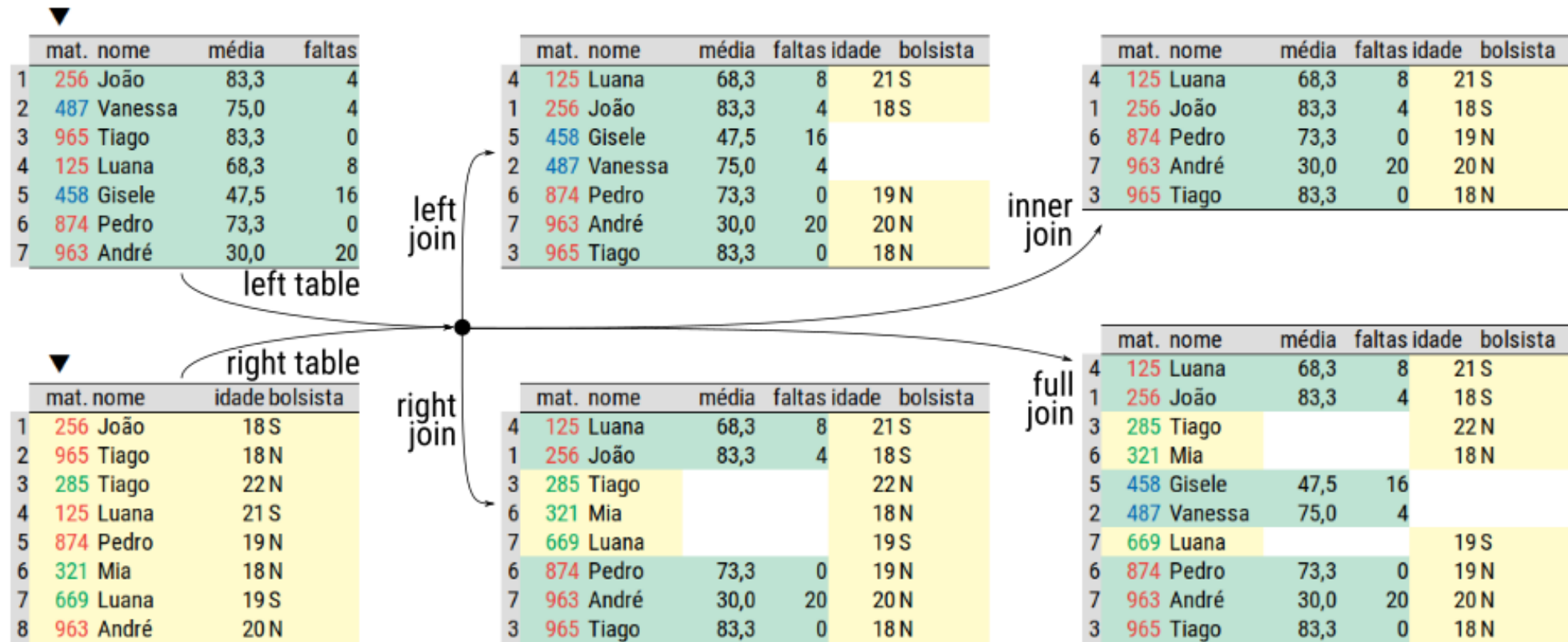
# Junções

- ▶ Junções permitem parear dados de tabelas separadas quando elas possuem uma chave (ou chave primária).
- ▶ As operações de junção podem ser inicialmente de 4 tipos:
  - ▶ Junção por interseção (*inner join*).
  - ▶ Junção por união (*full join*).
  - ▶ Junção à esquerda (*left join*).
  - ▶ Junção à direita (*right join*).
  - ▶ Existe também os *exclusive joins*.



Tipos de junções de tabelas ilustrado com diagramas de Veen.

# Junções



Junções de tabelas do tipo inclusivas.



# Junções

```
## Informações de cadastro dos alunos
## em outra base de dados.
df_extra <- tribble(
  ~mat, ~nome, ~idade, ~bolsista,
  256, 'João', 18, "S",
  965, 'Tiago', 18, "N",
  285, 'Tiago', 22, "N",
  125, 'Luana', 21, "S",
  874, 'Pedro', 19, "N",
  321, 'Mia', 18, "N",
  669, 'Luana', 19, "S",
  963, 'André', 20, "N",
)
```

```
## Full join = união.
full_join(df1, df_extra,
  by = c("matricula" = "mat", "nome")) %>%
  arrange(matricula)
```

```
# # A tibble: 10 × 9
#   matricula nome curso prova1 prova2 prova3 faltas idade bolsista
#   <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
# 1 125 Luana Est 70 85 50 8 21 S
# 2 256 João Mat 80 90 80 4 18 S
# 3 285 Tiago <NA> NA NA NA NA 22 N
# 4 321 Mia <NA> NA NA NA NA 18 N
# 5 458 Gisele Est 45 50 NA 16 NA <NA>
# 6 487 Vanessa Mat 75 75 75 4 NA <NA>
# 7 669 Luana <NA> NA NA NA NA 19 S
# 8 874 Pedro Mat 55 75 90 0 19 N
# 9 963 André Est 30 NA 30 20 20 N
# 10 965 Tiago Est 95 80 75 0 18 N
```

# Junções

```
## Inner join = intersecção.  
inner_join(df1, df_extra,  
           by = c("matricula" = "mat", "nome")) %>%  
  arrange(matricula)
```

```
# # A tibble: 5 × 9  
#   matricula nome   curso prova1 prova2 prova3 faltas idade bolsista  
#   <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>  
# 1     125 Luana Est      70     85     50     8     21 S  
# 2     256 João  Mat      80     90     80     4     18 S  
# 3     874 Pedro Mat      55     75     90     0     19 N  
# 4     963 André Est      30    NA     30    20     20 N  
# 5     965 Tiago Est      95     80     75     0     18 N
```

```
# Os da 2ª que não aparecem na 1ª.  
anti_join(df1, df_extra,  
          by = c("matricula" = "mat", "nome")) %>%  
  arrange(matricula)
```

```
# # A tibble: 2 × 7  
#   matricula nome   curso prova1 prova2 prova3 faltas  
#   <dbl> <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
# 1     458 Gisele Est      45     50    NA     16  
# 2     487 Vanessa Mat      75     75     75     4
```

# Junções

```
## Todos os que estão na 1ª tabela
left_join(df1, df_extra, by = c("matricula" = "mat", "nome")) %>%
  arrange(matricula)
```

```
# # A tibble: 7 × 9
#   matricula nome      curso prova1 prova2 prova3 faltas idade bolsista
#   <dbl> <chr>    <chr>   <dbl>   <dbl>   <dbl>   <dbl> <dbl> <chr>
# 1      125 Luana     Est       70      85      50      8     21 S
# 2      256 João      Mat       80      90      80      4     18 S
# 3      458 Gisele    Est       45      50      NA     16     NA <NA>
# 4      487 Vanessa  Mat       75      75      75      4     NA <NA>
# 5      874 Pedro    Mat       55      75      90      0     19 N
# 6      963 André     Est       30      NA      30     20     20 N
# 7      965 Tiago    Est       95      80      75      0     18 N
```

```
## Todos os que estão na 2ª tabela
right_join(df1, df_extra, by = c("matricula" = "mat", "nome")) %>%
  arrange(matricula)
```

```
# # A tibble: 8 × 9
#   matricula nome      curso prova1 prova2 prova3 faltas idade bolsista
#   <dbl> <chr>    <chr>   <dbl>   <dbl>   <dbl>   <dbl> <dbl> <chr>
# 1      125 Luana     Est       70      85      50      8     21 S
# 2      256 João      Mat       80      90      80      4     18 S
# 3      285 Tiago    <NA>      NA      NA      NA     NA     22 N
# 4      321 Mia      <NA>      NA      NA      NA     NA     18 N
# 5      669 Luana    <NA>      NA      NA      NA     NA     19 S
# 6      874 Pedro    Mat       55      75      90      0     19 N
# 7      963 André     Est       30      NA      30     20     20 N
# 8      965 Tiago    Est       95      80      75      0     18 N
```

## Extra · Funções para descrição geral dos dados

```
Hmisc::describe(iris)
psych::describe(iris)
skimr::skim(iris)
summarytools::descr(iris)
summarytools::dfSummary(iris)
```

```
Hmisc::describe(df1)

# df1
#
# 7 Variables      7 Observations
# -----
# matricula
#      n missing distinct      Info      Mean      Gmd
#      7         0         7         1     589.7     414.3
#
# lowest : 125 256 458 487 874, highest: 458 487 874 963 965
#
# Value      125    256    458    487    874    963    965
# Frequency      1      1      1      1      1      1      1
# Proportion 0.143 0.143 0.143 0.143 0.143 0.143 0.143
# -----
# nome
#      n missing distinct
#      7         0         7
#
# lowest : André  Gisele  João   Luana  Pedro
# highest: João   Luana  Pedro  Tiago  Vanessa
#
# Value      André  Gisele  João   Luana  Pedro  Tiago  Vanessa
# Frequency      1      1      1      1      1      1      1
# Proportion 0.143 0.143 0.143 0.143 0.143 0.143 0.143
# -----
# curso
#      n missing distinct
#      7         0         2
#
```