Aula 3
Terça à tarde

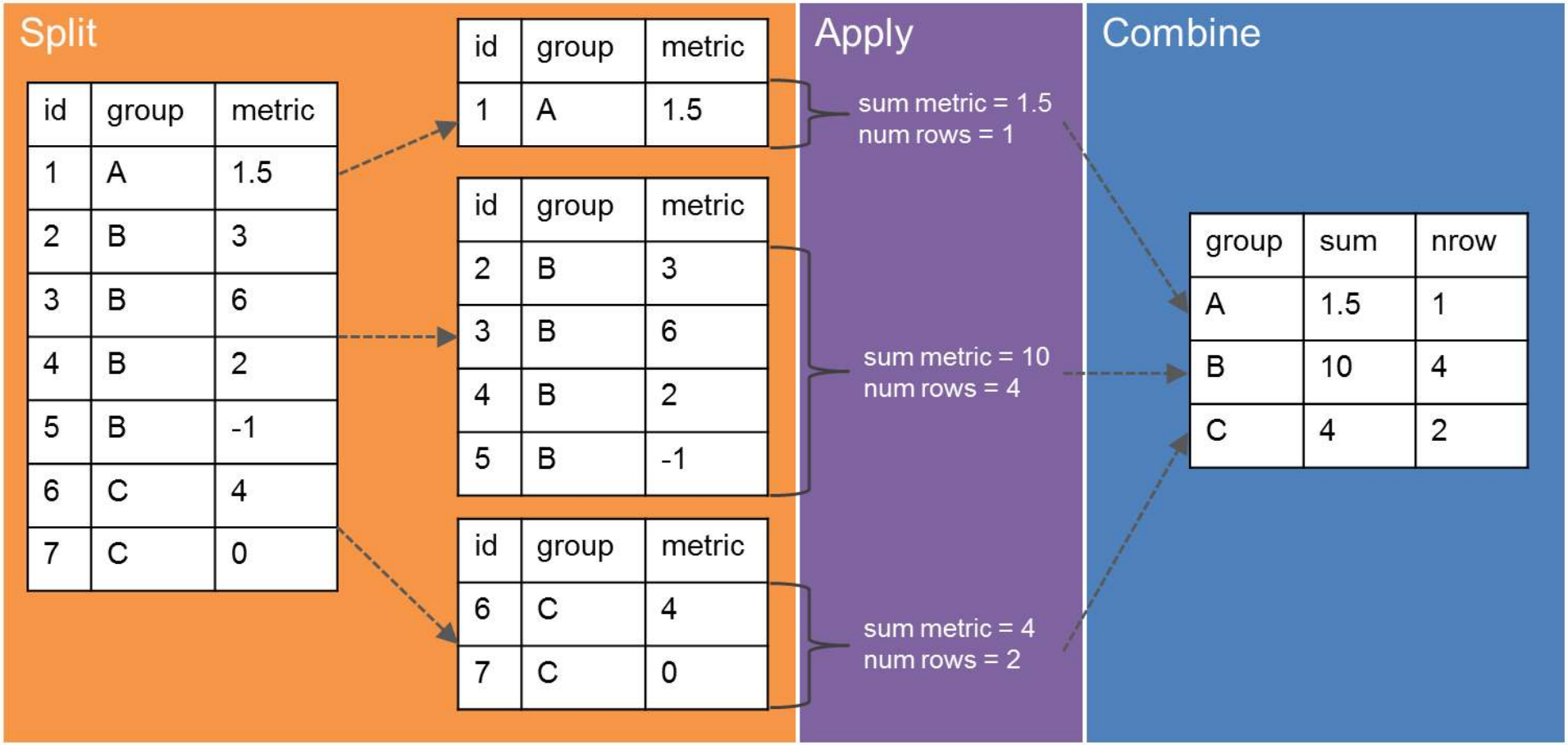
Introdução ao
** Tidyverse **
Manuseio de dados

Journal of Statistical Software

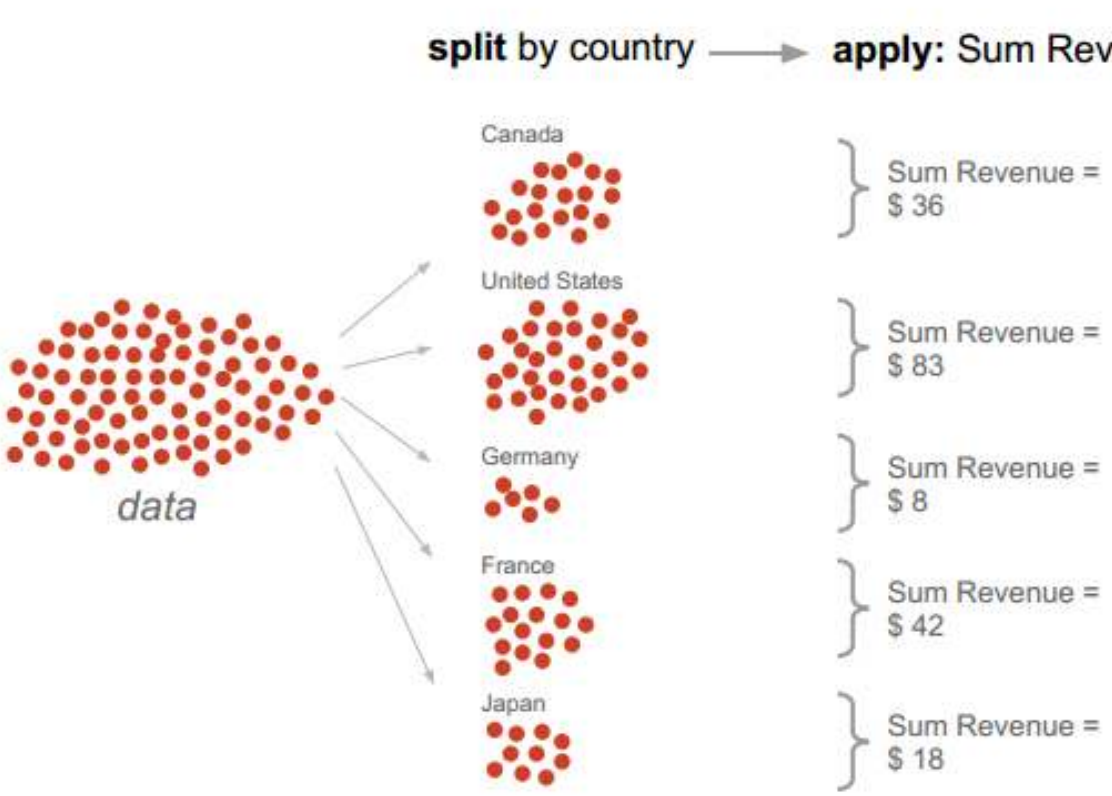
April 2011, Volume 40, Issue 1. *http://www.jstatsoft.org/*

The Split-Apply-Combine Strategy for Data Analysis

Hadley Wickham


Split
id group metric
1 A 1.5
2 B 3
3 B 6
4 B 2
5 B -1
6 C 4
7 C 0
id group metric
1 A 1.5
id group metric
2 B 3
3 B 6
4 B 2
5 B -1
id group metric
6 C 4
7 C 0
Apply
sum metric = 1.5
num rows = 1
sum metric = 10
num rows = 4
sum metric = 4
num rows = 2
Combine
group sum nrow
A 1.5 1
B 10 4
C 4 2

The basics of **split-apply-combine**



Country	Sum Revenue
United States	$ 83
France	$ 42
Canada	$ 36
Japan	$ 18

Tidy Data Tutor visualizes how your R and Tidyverse code transforms your data (for Python try Pandas Tutor)

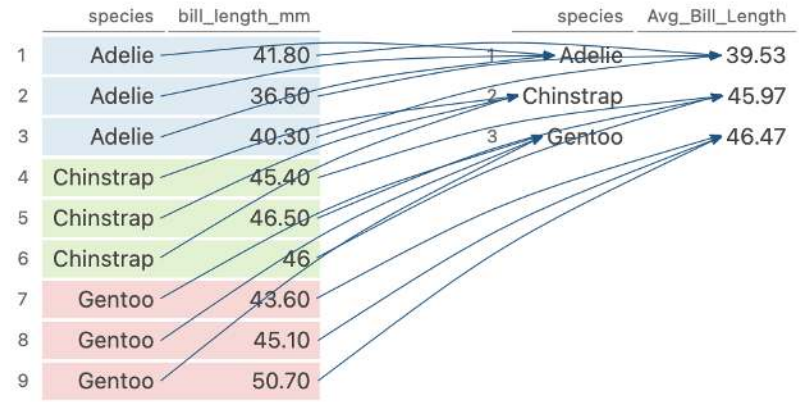
Join our private mailing list if you're a data science instructor who wants to help guide this tool's development

Examples: penguins, csv, arrange, filter, select, mutate | group arrange, filter, slice, mutate, summarise, add

```
1  library(dplyr)
2  library(palmerpenguins)
3
4  set.seed(2021-12-03)
5
6  sample_penguins <- penguins %>%
7    group_by(species) %>%
8    sample_n(3) %>%
9    select(species, bill_length_mm)
10
11 sample_penguins %>%
12   summarise(Avg_Bill_Length = mean(bill_length_mm))
```

Visualize %>% pipeline on last line

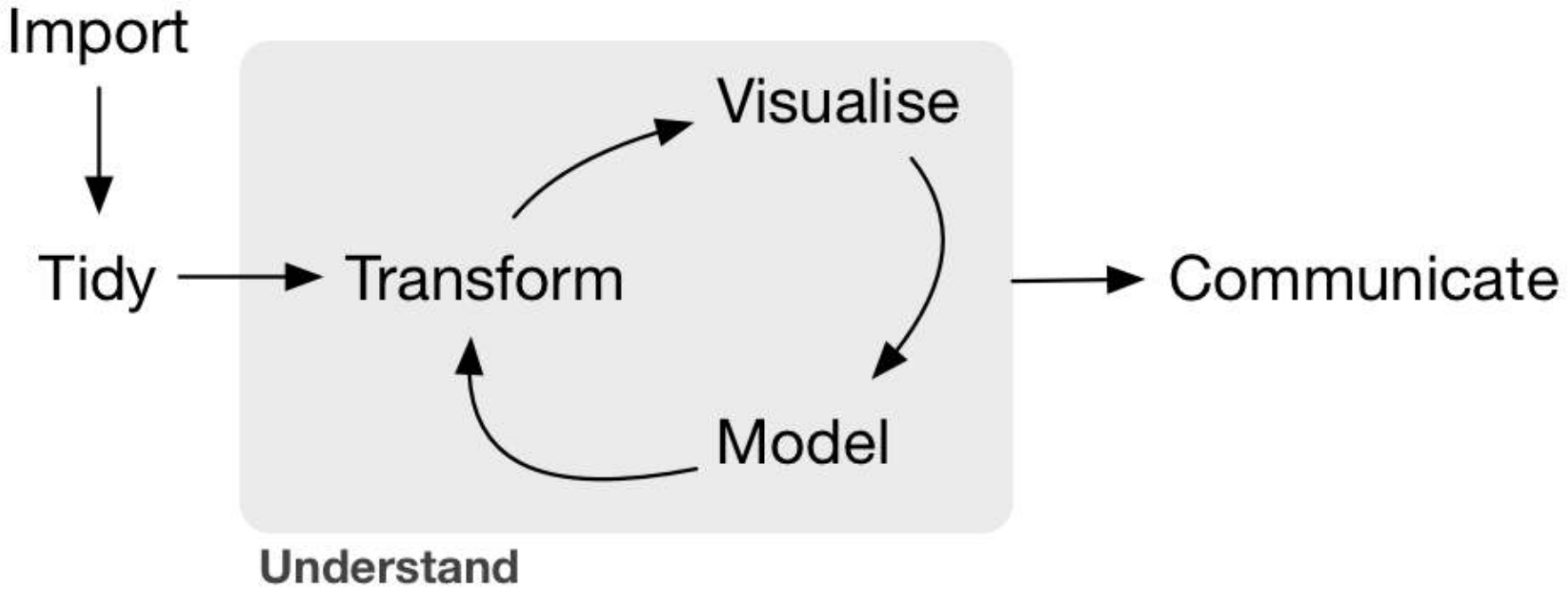
`summarise(Avg_Bill_Length = mean(bill_length_mm))`

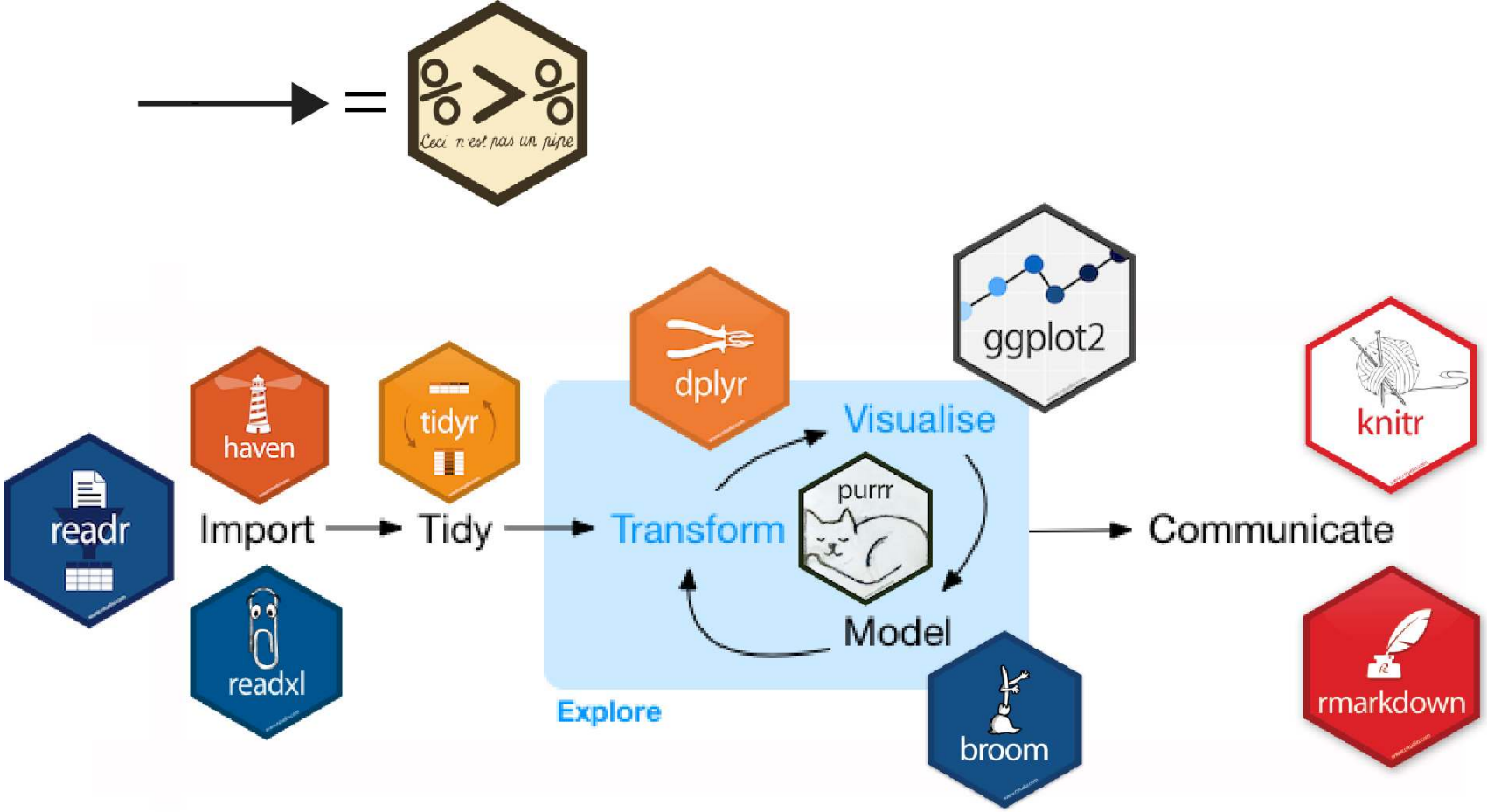


suggest improvement

pin no-hover URL: https://tidydatatutor.com/vis.html#code=library%28dplyr%29%0Alibrary%28palmerpenguins

https://tidydatatutor.com


Import
Tidy
Transform
Visualise
Model
Communicate
Understand

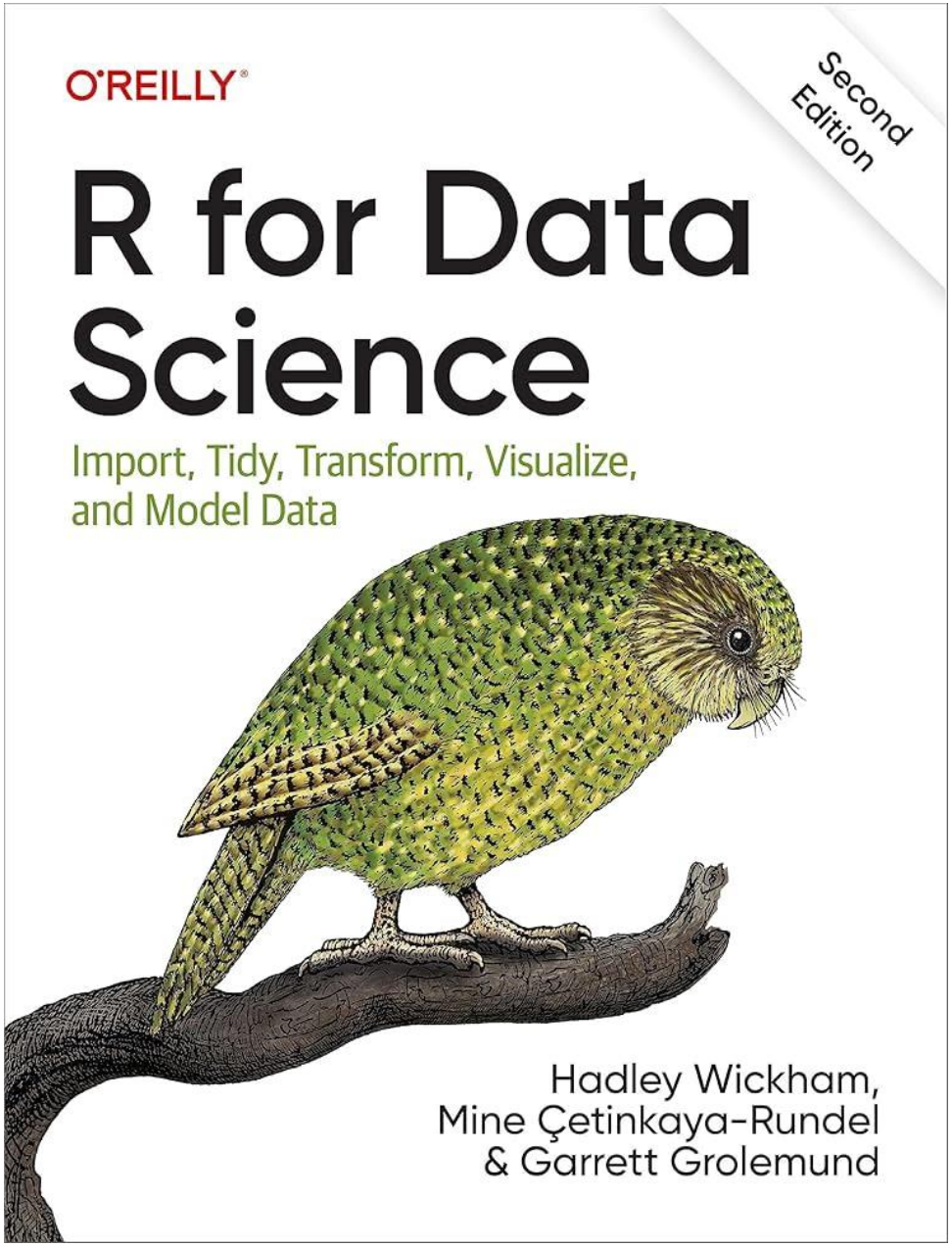


data.frames

factors

strings

https://www.tidyverse.org



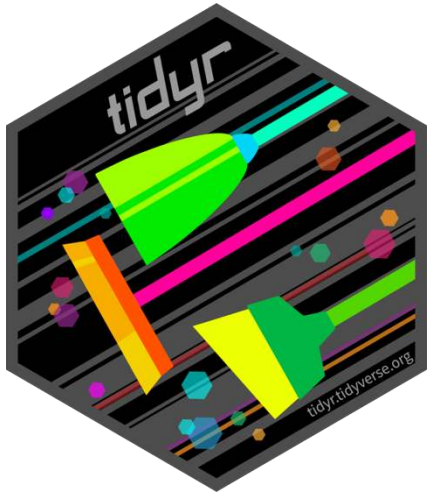
https://r4ds.hadley.nz

Instalando e carregando pacotes

- install.packages(“tidyverse”)

Lógica do tidyverse

- Principais verbos do **dplyr e tidyr**
- O que é um conjunto de dados *tidy*?
- Como converter dados em formatos não padrão para tidy?
- Trabalhar com Data frame



Verbos do dplyr

As principais funções do dplyr são:

- **mutate( )** cria novas varíaveis que são funções de variáveis já existentes.
- **select( )** permite selecionar as variáveis que se deseja trabalhar e assim criar um novo dataset.
- **filter( )** realiza a mesma função dos filtros do Excel, selecionar as linhas por um critério definido.
- **summarise( )** gera valores resumo de um agrupamento dos dados (ex: pela média, por quantis, etc.).
- **arrange( )** edita a ordem das linhas de acordo com um critério.

Verbos do tidyr

Reshape Data - Pivot data to reorganize values into a new layout.

table4a

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K

country	year	cases
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

pivot_longer(data, cols, names_to = "name", values_to = "value", values_drop_na = FALSE)

"Lengthen" data by collapsing several columns into two. Column names move to a new names_to column and values to a new values_to column.

```
pivot_longer(table4a, cols = 2:3, names_to ="year",
  values_to = "cases")
```

table2

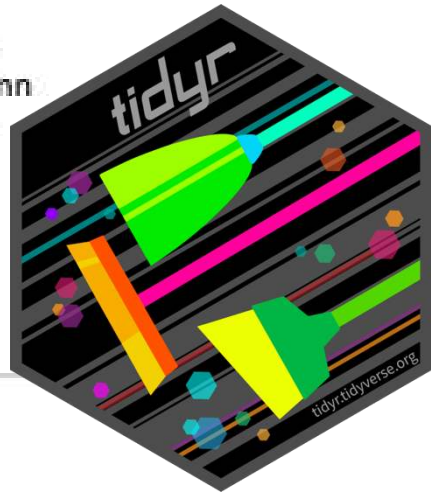
country	year	type	count
A	1999	cases	0.7K
A	1999	pop	19M
A	2000	cases	2K
A	2000	pop	20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

pivot_wider(data, names_from = "name", values_from = "value")

The inverse of pivot_longer(). "Widen" data by expanding two columns into several. One column provides the new column names, the other the values.

```
pivot_wider(table2, names_from = type,
  values_from = count)
```



Data tidying with tidyr : : CHEAT SHEET

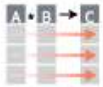
Tidy data is a way to organize tabular data in a consistent data structure across packages.
A table is tidy if:

Each **variable** is in its own **column**

&

Each **observation**, or **case**, is in its own row

Access **variables** as **vectors**



Preserve **cases** in vectorized operations

Tibbles

AN ENHANCED DATA FRAME

Tibbles are a table format provided by the **tibble** package. They inherit the data frame class, but have improved behaviors:

- **Subset** a new tibble with], a vector with [[and $.
- **No partial matching** when subsetting columns.
- **Display** concise views of the data on one screen.

options(tibble.print_max = n, tibble.print_min = m, tibble.width = Inf) Control default display settings.

View() or **glimpse()** View the entire data set.

CONSTRUCT A TIBBLE

tibble(…) Construct by columns.
tibble(x = 1:3, y = c("a", "b", "c"))

tribble(…) Construct by rows.
tribble(~x, ~y,
1, "a",
2, "b",
3, "c")

Both make this tibble

```
A tibble: 3 × 2
      x     y
  <int> <chr>
1     1     a
2     2     b
3     3     c
```

as_tibble(x, …) Convert a data frame to a tibble.

enframe(x, name = "name", value = "value") Convert a named vector to a tibble. Also **deframe()**.

is_tibble(x) Test whether x is a tibble.

Reshape Data - Pivot data to reorganize values into a new layout.

table4a

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K

→

country	year	cases
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

pivot_longer(data, cols, names_to = "name", values_to = "value", values_drop_na = FALSE)

"Lengthen" data by collapsing several columns into two. Column names move to a new names_to column and values to a new values_to column.

pivot_longer(table4a, cols = 2:3, names_to = "year", values_to = "cases")

table2

country	year	type	count
A	1999	cases	0.7K
A	1999	pop	19M
A	2000	cases	2K
A	2000	pop	20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

→

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

pivot_wider(data, names_from = "name", values_from = "value")

The inverse of pivot_longer(). "Widen" data by expanding two columns into several. One column provides the new column names, the other the values.

pivot_wider(table2, names_from = type, values_from = count)

Split Cells - Use these functions to split or combine cells into individual, isolated values.

table5

country	century	year
A	19	99
A	20	00
B	19	99
B	20	00

→

country	year
A	1999
A	2000
B	1999
B	2000

unite(data, col, …, sep = "_", remove = TRUE, na.rm = FALSE) Collapse cells across several columns into a single column.

unite(table5, century, year, col = "year", sep = "")

table3

country	year	rate
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M

→

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172
B	2000	80K	174

separate(data, col, into, sep = "[^[:alnum:]]+", remove = TRUE, convert = FALSE, extra = "warn", fill = "warn", …) Separate each cell in a column into several columns. Also **extract()**.

separate(table3, rate, sep = "/", into = c("cases", "pop"))

table3

country	year	rate
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M

→

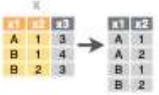
country	year	rate
A	1999	0.7K
A	1999	19M
A	2000	2K
A	2000	20M
B	1999	37K
B	1999	172M
B	2000	80K
B	2000	174M

separate_rows(data, …, sep = "[^[:alnum:].]+", convert = FALSE) Separate each cell in a column into several rows.

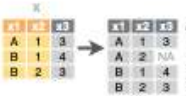
separate_rows(table3, rate, sep = "/")

Expand Tables

Create new combinations of variables or identify implicit missing values (combinations of variables not present in the data).



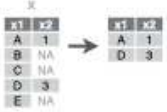
expand(data, …) Create a new tibble with all possible combinations of the values of the variables listed in … Drop other variables.
expand(mtcars, cyl, gear, carb)



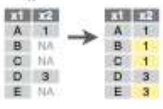
complete(data, …, fill = list()) Add missing possible combinations of values of variables listed in … Fill remaining variables with NA.
complete(mtcars, cyl, gear, carb)

Handle Missing Values

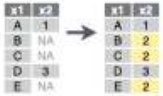
Drop or replace explicit missing values (NA).



drop_na(data, …) Drop rows containing NA's in … columns.
drop_na(x, x2)



fill(data, …, .direction = "down") Fill in NA's in … columns using the next or previous value.
fill(x, x2)



replace_na(data, replace) Specify a value to replace NA in selected columns.
replace_na(x, list(x2 = 2))

Tidy Data - A foundation for wrangling in R

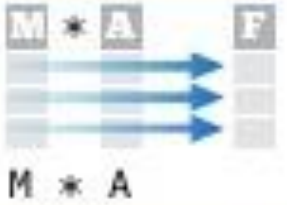
In a tidy data set:

Each **variable** is saved in its own **column**

&

Each **observation** is saved in its own **row**

Tidy data complements R's **vectorized operations**. R will automatically preserve observations as you manipulate variables. No other format works as intuitively with R.



Exercícios

https://analises-ecologicas.com/cap5.html#pivot_longer-e-pivot_wider