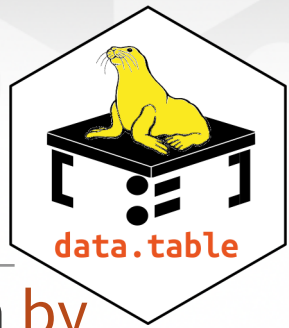


# Transformação de dados com data.table :: CHEAT SHEET



## Informação básica

“data.table” é um pacote extremamente rápido e de uso eficiente de memória usado na transformação de dados em R. Ele funciona por converter o conjunto de dados nativo do R (data.frame) em data.table que apresenta funcionalidades novas e melhoradas. O uso básico de data.tables segue:

**dt[i, j, by]**

pegue data.table **dt**,  
subdivida linhas usando **i**  
e manipule colunas por **j**,  
agrupadas de acordo com **by**

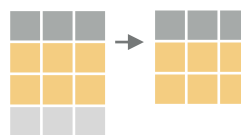
data.tables também são data.frames, portanto funções que se aplicam a data.frames também funcionam sobre data.tables


## Crie um data.table

**data.table(a = c(1, 2), b = c("a", "b"))**  
cria um data.table a partir do zero. Análogo ao data.frame().

**setDT(df)\*** ou **as.data.table(df)**  
converte um data.frame ou lista a um data.table.

## Subdivida linhas usando i

 **dt[1:2, ]**  
subdivide linhas baseado nos números (índices) das linhas.


 **dt[a > 5, ]**  
subdivide linhas baseado em valores em uma ou mais colunas.

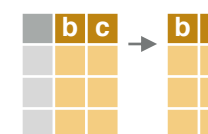
### OPERADORES LÓGICOS PARA USAR EM i

<	<=	is.na()	%in%		%like%
>	>=	!is.na()	!	&	%between%


## Manipule colunas por j

### EXTRAIR

 **dt[, c(2)]**  
extraí colunas por número; anteceda números de colunas com “-” para remover colunas.

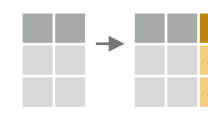
 **dt[, .(b, c)]**  
extraí colunas por nome.

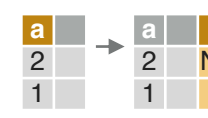
### SUMARIZAR

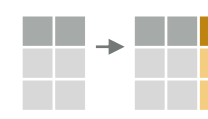
 **dt[, .(x = sum(a))]**  
cria um data.table com novas colunas baseado no valores sumarizados das linhas.

funções de sumário como mean(), median(), min(), max(), etc. podem ser usadas para sumarizar linhas


### COMPUTAR COLUNAS\*

 **dt[, c := 1 + 2]**  
calcula e cria uma coluna baseado numa expressão

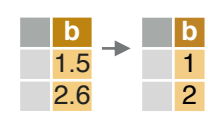
 **dt[a == 1, c := 1 + 2]**  
computa uma coluna baseado numa expressão, mas apenas para um grupo de linhas

 **dt[, `:=`(c = 1, d = 2)]**  
computa múltiplas colunas baseado em diferentes expressões

### DELETAR COLUNAS

 **dt[, c := NULL]**  
remove uma coluna.

### CONVERTER TIPOS DE COLUNAS

 **dt[, b := as.integer(b)]**  
converte o tipo de coluna usando as.integer(), as.numeric(), as.character(), as.Date(), etc..

## Agrupe de acordo com by

 **dt[, j, by = .(a)]** – agrupa linhas por valores em colunas específicas.

**dt[, j, keyby = .(a)]** – agrupa e simultaneamente e ordena linhas por valores em colunas específicas.

### OPERAÇÕES AGRUPADAS COMUNS

**dt[, .(c = sum(b)), by = a]** – sumariza linhas dentro de grupos.

**dt[, c := sum(b), by = a]** – cria uma nova coluna e computa linhas dentro de grupos.

**dt[, .SD[1], by = a]** – extraí primeira linha de grupos.

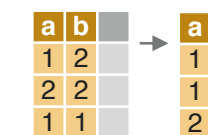
**dt[, .SD[N], by = a]** – extraí ultima linha de grupos.

## Encadeamento

**dt[...][...]** – realiza sequência de operações em data.table por encadeamento de múltiplos “[ ]”.

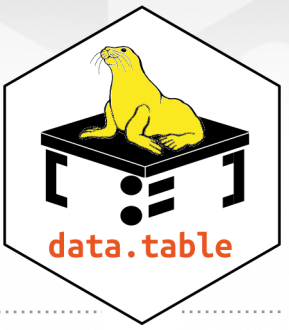
## Funções para data.tables

### REORDENAR

 **setorder(dt, a, -b)**  
reordena um data.table de acordo com colunas especificadas; anteceda nomes de colunas com “-” para ordens decrescentes.

### \* SET FUNCTIONS AND :=

funções em data.table prefixadas com “set” e o operador “:=” funcionam sem “<-” para alterar os dados sem fazer cópias na memória. Por exemplo, o mais eficiente “setDT(df)” é análogo a “df <- as.data.table(df)”.



## LINHAS ÚNICAS

a	b
1	2
2	2
1	2

**unique**(dt, by = c("a", "b")) – extrai linhas únicas baseado em colunas especificadas por “by”. Omite “by” para usar todas as colunas.

**uniqueN**(dt, by = c("a", "b")) – conta o número de linhas únicas baseado em colunas especificadas por “by”.

## RENOMEAR COLUNAS

a	b
1	2
2	2
1	2

**setnames**(dt, c("a", "b"), c("x", "y"))  
renomeia colunas.

## DEFINA PALAVRAS-CHAVE

**setkey**(dt, a, b) – define palavras-chave que permitem pesquisa rápida e repetida em colunas especificadas usando "dt[(valor),]" ou fundir dados sem especificar colunas usando "dt\_a[dt\_b]".

# Combinar data.tables

## JUNTAR (join)

a	b
1	c
2	a
3	b

**dt\_a[dt\_b, on = .(b = y)]** – junte data.tables em linhas com valores iguais

a	b	c
1	c	7
2	a	5
3	b	6

**dt\_a[dt\_b, on = .(b = y, c > z)]** – junte data.tables em linhas com valores iguais e *desiguais*

## JUNÇÃO POR “ROLAMENTO” (Rolling join)

a	id	date
1	A	01-01-2010
2	A	01-01-2012
3	A	01-01-2014
1	B	01-01-2010
2	B	01-01-2012

**dt\_a[dt\_b, on = .(id = id, date = date), roll = TRUE]** – junte data.tables em linhas correspondentes a colunas de IDs, mas mantendo apenas as correspondências mais recentes com o data.table a esquerda de acordo com colunas de datas.

“roll = -Inf” reverte a direção.

## LIGAR

a	b
1	2
2	2
1	2

**rbind**(dt\_a, dt\_b) – combinar linhas de dois data.tables

a	b
1	2
2	2
1	2

**cbind**(dt\_a, dt\_b) – combinar colunas de dois data.tables

# Reformate um data.table

## TRANSFORMAR PARA FORMATO AMPLO

id	y	a	b
A	x	1	3
A	z	2	4
B	x	1	3
B	z	2	4

**dcast**(dt,  
id ~ y,  
value.var = c("a", "b"))

Transforme um data.table de um formato longo a amplo.

dt Um data.table.  
id ~ y Formula com um “LHS”: colunas de IDs contendo IDs de múltiplas entradas; e um “RHS”: colunas com valores para distribuir nos cabeçalhos das colunas.  
value.var Colunas contendo valores para preencher células.

## TRANSFORMAR PARA FORMATO LONGO

id	a	x	a	z	b	x	b	z
A	1	2	3	4				
B	1	2	3	4				

**melt**(dt,  
id.vars = c("id"),  
measure.vars = patterns("^a", "^b"),  
variable.name = "y",  
value.name = c("a", "b"))

Transforme um data.table de um formato amplo a longo.

dt Um data.table.  
id.vars Colunas de IDs com IDs para múltiplas entradas.  
measure.vars Colunas contendo valores para preencher em células (geralmente em forma de padrões).  
variable.name, value.name Nomes das novas colunas para variáveis e valores derivados das colunas antigas.

# Aplique funções a colunas

## APLICAR UMA FUNÇÃO A MÚLTIPLAS COLUNAS

a	b
1	4
2	5
3	6

**dt[, lapply(.SD, mean), .SDcols = c("a", "b")]** – aplicar uma função – e.g. mean(), as.character(), which.max() – a colunas especificadas em .SDcols com lapply() e o símbolo .SD. Também funciona com grupos.

a	b
1	2
2	2
3	2

**cols <- c("a")**  
**dt[, paste0(cols, "\_m") := lapply(.SD, mean), .SDcols = cols]** – aplicar uma função a colunas especificadas e assignar o resultado com variáveis sufixadas aos dados originais.

# Linhas sequenciais

## IDs DE LINHAS

a	b
1	a
2	a
3	b

**dt[, c := 1:N, by = b]** – dentro de grupos, compute uma coluna com linhas sequenciais de IDs.

## DEFASAGEM (LAG) & INCREMENTO (LEAD)

a	b
1	a
2	a
3	b
4	b
5	b

**dt[, c := shift(a, 1), by = b]** – dentro de grupos, duplicar uma coluna com linhas *defasadas* por um valor especificado.

**dt[, c := shift(a, 1, type = "lead"), by = b]** – dentro de grupos, duplicar uma coluna com linhas sendo *incrementadas* por um valor especificado.

# carregar & salvar arquivos

## IMPORTAR

**fread**("file.csv")  
carregar dados desde um arquivo plano, como \*.csv ou \*.tsv em R.  
**fread**("file.csv", select = c("a", "b"))  
carregar colunas especificadas de um arquivo plano em R.

## EXPORTAR

**fwrite**(dt, "file.csv") – salvar dados em um arquivo plano desde R.