Data Transformation with data.table : : CHEAT SHEET

Basics

data.table is an extremely fast and memory efficient package for transforming data in R. It works by converting R's native data frame objects into data.tables with new and enhanced functionality. The basics of working with data.tables are:

dt[i, j, by]

Take data.table **dt**,
subset rows using **i**,
and manipulate columns with **j**,
grouped according to **by**.

data.tables are also data frames – functions that work with data frames therefore also work with data.tables.

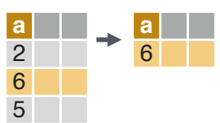
Create a data.table

data.table(a = c(1, 2), b = c("a", "b")**)** – create a data.table from scratch. Analogous to data.frame().

setDT(df**)*** or **as.data.table(**df**)** – convert a data frame or a list to a data.table.

Subset rows using i

dt[**1:2**,] – subset rows based on row numbers.



dt[**a > 5**,] – subset rows based on values in one or more columns.

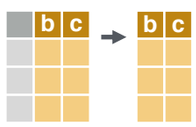
LOGICAL OPERATORS TO USE IN i

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

Manipulate columns with j

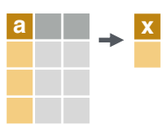
EXTRACT

dt[, **c(2)**] – extract column(s) by number. Prefix column numbers with "-" to drop.



dt[, **.(b, c)**] – extract column(s) by name.

SUMMARIZE

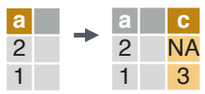


dt[, **.(x = sum(a))**] – create a data.table with new columns based on the summarized values of rows.

Summary functions like mean(), median(), min(), max(), etc. may be used to summarize rows.

COMPUTE COLUMNS*

dt[, **c := 1 + 2**] – compute a column based on an expression.



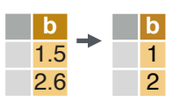
dt[**a == 1, c := 1 + 2**] – compute a column based on an expression but only for a subset of rows.

dt[, **`:=`(c = 1 , d = 2)**] – compute multiple columns based on separate expressions.

DELETE COLUMN

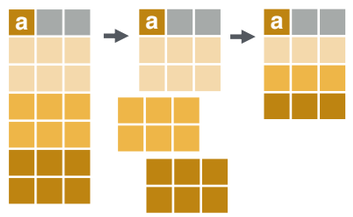
dt[, **c := NULL**] – delete a column.

CONVERT COLUMN TYPE



dt[, **b := as.integer(b)**] – convert the type of a column using as.integer(), as.numeric(), as.character(), as.Date(), etc..

Group according to by



dt[, j, **by = .(a)**] – group rows by values in specified column(s).

dt[, j, **keyby = .(a)**] – group *and simultaneously sort* rows according to values in specified column(s).

COMMON GROUPED OPERATIONS

dt[, **.(c = sum(b))**, **by = a**] – summarize rows within groups.

dt[, **c := sum(b)**, **by = a**] – create a new column and compute rows within groups.

dt[, **.SD[1]**, **by = a**] – extract first row of groups.

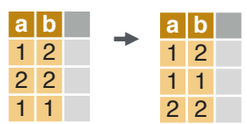
dt[, **.SD[.N]**, **by = a**] – extract last row of groups.

Chaining

dt[…**][**…**]** – perform a sequence of data.table operations by *chaining* multiple "[]".

Functions for data.tables

REORDER

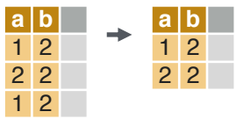


setorder(dt, a, **-**b**)** – reorder a data.table according to specified columns. Prefix column names with "**-**" for descending order.

* SET FUNCTIONS AND :=

data.table's functions prefixed with "set" and the operator ":=" work without "<-" to alter data without making copies in memory. E.g. the more efficient "setDT(df)" is analogous to "df <- as.data.table(df)".

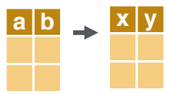
UNIQUE ROWS



unique(dt, by = c("a", "b")**)** – extract unique rows based on columns specified in "by". Leave out "by" to use all columns.

uniqueN(dt, by = c("a", "b")**)** – return the number of unique rows based on columns specified in "by".

RENAME COLUMNS



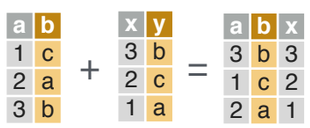
setnames(dt, c("a", "b"), c("x", "y")**)** – rename column(s).

SET KEYS

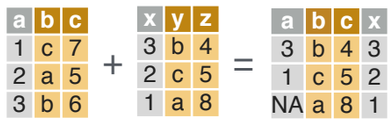
setkey(dt, a, b**)** – set keys in a data.table to enable faster repeated lookups in specified column(s) using "dt[.(value),]" or for merging without specifying merging columns "dt_a[dt_b]".

Combine data.tables

JOIN



dt_a[dt_b, **on = .(b = y)**] – join two data.tables based on rows with equal values. You can leave out "on" if keys are already set.



dt_a[dt_b, **on = .(b = y, c > z)**] – join two data.tables based on rows with equal and unequal values.

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

+

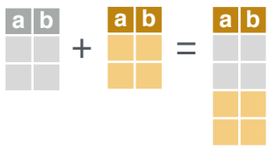
b	id	date
1	A	01-01-2013
1	B	01-01-2013

=

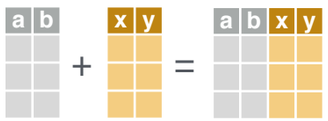
a	id	date	b
2	A	01-01-2013	1
2	B	01-01-2013	1

dt_a[dt_b, **on = .(id = id, date = date), roll = TRUE**] – by default, a rolling join matches rows, according to id columns, but only keeps the most recent preceding match with the left table, according to date columns. Use "roll = -Inf" to reverse direction.

BIND



rbind(dt_a, dt_b**)** – combine rows of two data.tables.



cbind(dt_a, dt_b**)** – combine columns of two data.tables.

Reshape a data.table

RESHAPE TO WIDE FORMAT

id	y	a	b
A	X	1	3
A	Z	2	4
B	X	1	3
B	Z	2	4

→

id	a_X	a_Z	b_X	b_Z
A	1	2	3	4
B	1	2	3	4

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

Reshape a data.table from long to wide format.

dt	A data.table.
id ~ y	Formula with a LHS: id column(s) containing id(s) for multiple entries. And a RHS: column(s) with value(s) to spread in column headers.
value.var	Column(s) containing values to fill into cells.

RESHAPE TO LONG FORMAT

id	a_X	a_Z	b_X	b_Z
A	1	2	3	4
B	1	2	3	4

→

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

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

Reshape a data.table from wide to long format.

dt	A data.table.
id.vars	Id column(s) with id(s) for multiple entries.
measure	Column(s) containing values to fill into cells (often in pattern form).
variable.name, value.name	Name(s) of new column(s) for variables and values derived from old headers.

.SD

Refer to a **S**ubset of the **D**ata with **.SD**.

MULTIPLE COLUMN TYPE CONVERSION

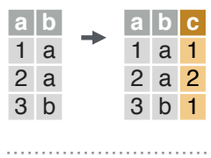
dt[, **lapply(.SD, as.character)**, **.SDcols = c("a", "b")**] – convert the type of designated columns.

GROUP OPTIMA

dt[, **.SD[which.max(a)]**, **by = b**] – within groups, extract rows with the maximum value in a specified column. Also works with which.min() and which(). Similar to ".SD[.N]" and ".SD[1]".

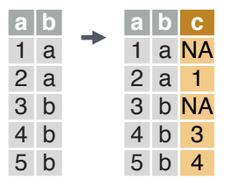
Sequential rows

ROW IDS



dt[, **c := 1:.N**, **by = b**] – within groups, compute a column with sequential row IDs.

LAG & LEAD



dt[, **c := shift(a, 1)**, **by = b**] – within groups, duplicate a column with rows *lagged* by specified amount.

dt[, **c := shift(a, 1, type = "lead")**, **by = b**] – within groups, duplicate a column with rows *leading* by specified amount.

fread & fwrite

IMPORT

fread("file.csv"**)** – read data from a flat file such as .csv or .tsv into R.

fread("file.csv", select = c("a", "b")**)** – read specified column(s) from a flat file into R.

EXPORT

fwrite(dt, file = "file.csv"**)** – write data to a flat file from R.