tutorial uRos 2018

Jaap Walhout

September 12, 2018

Overview

1. Introduction
2. Fast read & write
3. Syntax
4. Basic operations (filtering rows & selecting columns)
5. Summarizing
6. Adding / updating variables
7. Joining datasets
8. Reshaping data

Special symbols: .N + .SD + .I

Special operator: :=

Introduction

Developers: Matt Dowle, Arun Srinivasan, Jan Gorecki, Michael Chirico,
Pasha Stetsenko, Tom Short, Steve Lianoglou, Eduard Antonyan,
Markus Bonsch, Hugh Parsonage

Since 2006 on CRAN, > 35 releases so far

678 packages import/depend/suggest **data.table** (543 CRAN + 135 Bioconductor)

Homepage: http://r-datatable.com

Introduction

Why use data.table?

Pros:

- speed
- memory efficiency
- coding flexibility
- non-equi joins

Cons:

- ‘different’ syntax

Fast read & write

50 million rows / 10 columns / ± 4GB

fread("datafile.csv")

```
             expr      time
data.table_fread      15.6
  readr_read_csv      92.6
   base_read.csv     559.9
```

fwrite(DT, "datafile.csv")

```
             expr      time
data.table_fread      32.6
  readr_read_csv     102.2
   base_read.csv     201.9
```

times in seconds

Syntax: data.table == enhanced data.frame

Three main enhancements:

1. Column names can be used as variables inside [….]
2. Because they are variables, we can use column names to calculate stuff inside [….]
3. An additional grouping argument: by

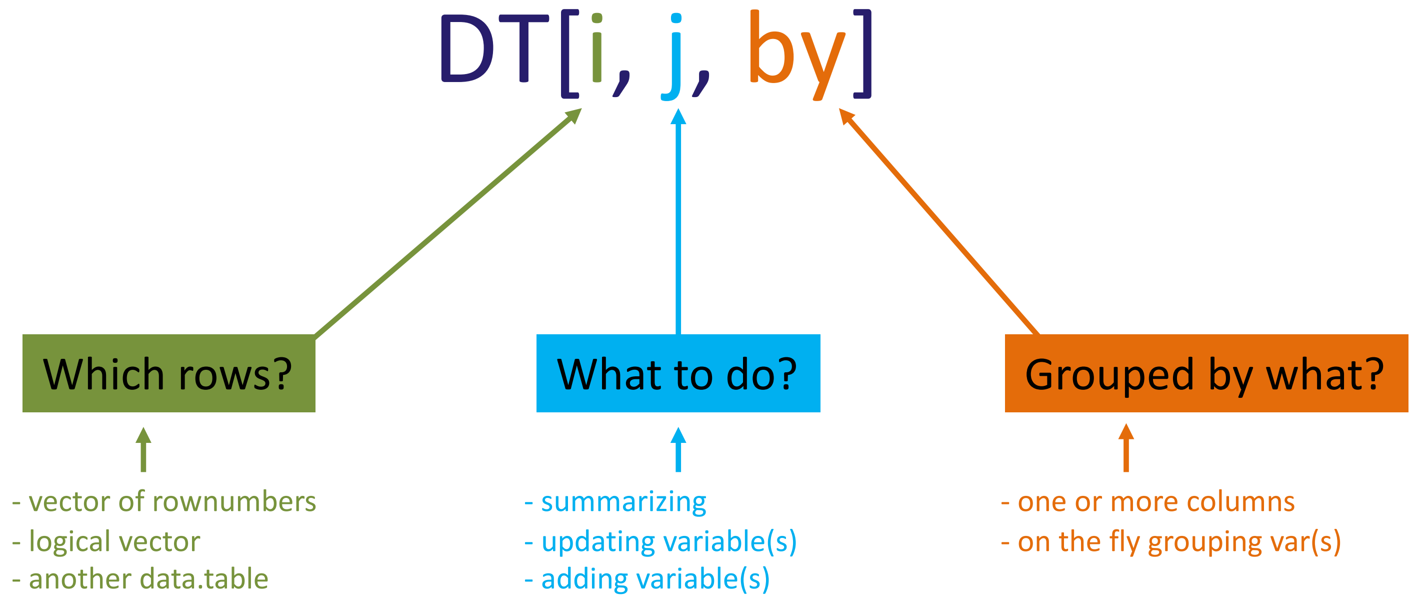
Syntax: dataframe refresher

Columnar data structure: 2D – rows and columns

- subset rows	`df[df$id == "01", ]`
- select columns	`df[, "val1"]`
- subset rows & select columns	`df[df$id == "01", "val1"]`

- that’s about it ….

Syntax: general form



Syntax: general form

DT[i, j, by]

```
data.table:     i                j          by
       SQL: where   select | update   group by
```

Example data

build in **iris** dataset:

```
irisDT <- as.data.table(iris)
```

Filtering rows & selecting columns

syntax: DT[**i**, **j**, **by**]

subset rows	`irisDT[Species == "setosa", ]`
select columns	`irisDT[, Petal.Width]`
	`irisDT[, .(Petal.Width)]`
subset rows & select columns	`irisDT[Species == "setosa", Petal.Width]`
	`irisDT[Species == "setosa", .(Petal.Width)]`

Filtering rows & selecting columns

subset rows	`irisDT[between(Petal.Width, 1, 1)]`
	`irisDT[Petal.Width %between% c(1, 2)]`
select columns	`irisDT[, .(Species, Sepal.Length)]`

Exercise 1

Open the file **ex1.R**

subset rows	:	get only the rows with a day lower than or equal to 10
select columns	:	select only the Month column and make sure you get a data.table back
subset rows & select columns	:	get only the Wind & Temp columns for the rows with a day higher than 5 and lower than or equal to 10

Summarizing

1. Counts
2. Aggregating
3. Group by

Counts

syntax: DT[**i**, **j**, **by**]

count	`irisDT[Species == "setosa", .N]`
count distinct	`irisDT[, .uniqueN(Species)]`
	`irisDT[Petal.Width < 0.9, uniqueN(Species)]`
	`uniqueN(irisDT, by = "Species")`

Aggregating

syntax: DT[**i**, **j**, **by**]

Simple aggregation: irisDT[, .(count = .N, average = mean(Petal.Width))]

Including filtering: irisDT[Petal.Width < 0.9, .(count = .N, average = mean(Petal.Width))]

Group by

syntax: DT[**i**, **j**, **by**]

```
irisDT[, .N, by = Species]

irisDT[, .(average = mean(Petal.Width)), by = Species]

irisDT[Sepal.Length < 5.3, .(average = mean(Petal.Width)), by = Species]

irisDT[, .(average = mean(Petal.Width)), by = .(Species, logi = Sepal.Length < 5.3)]
```

Group by

special symbol: **.SD**

SD = **S**ubset of **D**ata

- a data.table by itself
- holds data of current goup as defined in by
- when no by, .SD applies to whole data.table
- allows for calculations on multiple columns

Group by

special symbol: **.SD**

```
irisDT[, lapply(.SD, mean), by = Species]

irisDT[Sepal.Length < 5.3, lapply(.SD, mean), by = Species]
```

Group by

special symbol: **.SD**

special symbol: **.SDcols**

```
irisDT[, lapply(.SD, mean), by = Species, .SDcols = 1:2]

irisDT[, lapply(.SD, mean), by = Species, .SDcols = grep("Length", names(irisDT))]
```

Order of execution

DT[i, j, by]

DT[1, 3, 2]

Exercise 2

Open the file **ex2.R**

- Count the number of days per month
- Calculate the average Wind speed by month for only those days that have an ozone value
- Calculate the mean temperature for the odd and even days for each month

Updating, adding & deleting variables

special operator: **:=**

- updates a data.table in place (by reference)
- can be used to:
    - update existing column(s)
    - add new column(s)
    - delete column(s)

Updating variables

special operator: **:=**

```
irisDT[, Sepal.Length := Sepal.Length * 2]
```

```
irisDT[, `:=` (Sepal.Length = Sepal.Length * 2,
               Petal.Width = Petal.Width / 2)]
```

Updating variables by group

special operator: **:=**

```
irisDT[, Sepal.Length := Sepal.Length * uniqueN(Sepal.Width) / .N, by = Species]
```

```
irisDT[, `:=` (Sepal.Length = Sepal.Length * uniqueN(Sepal.Width),
               Petal.Width = Petal.Width / .N)
        , by = Species]
```

Adding variables

special operator: **:=**

special symbol: **.I**

```
irisDT[, rownumber := .I]

irisDT[, Sepal.Area := Sepal.Length * Sepal.Width]

irisDT[, `:=` (Sepal.Area = Sepal.Length * Sepal.Width,
               Petal.Area = Petal.Length * Petal.Width)]
```

Adding variables by group

special operator: **:=**

```
irisDT[, Total.Sepal.Area := sum(Sepal.Area), by = Species]
```

```
irisDT[, `:=` (Total.Sepal.Area = sum(Sepal.Area),
               Total.Petal.Area = sum(Petal.Area))
       , by = Species]
```

Deleting variables

special operator: **:=**

```
irisDT[, Sepal.Length := NULL]

irisDT[, (1:4) := NULL]

irisDT[, grep("Length", names(irisDT)) := NULL]
```

Exercise 3

Open the file **ex3.R**

- Change the Wind column from miles per hour to kilometers per hour

  (1 mph = 1.6 kmh)
- Calculate a new **chill** variable (Wind * Temperature)
- Calculate the average chill by month and add that as a new variable
- Remove the **Ozone** and **Solar.R** columns

Joining datasets

- subset rows	DT[id == “01”,]
- select columns	DT[, val1]
- subset rows & select columns	DT[id == “01”, val1]

Joining datasets

DT[i, j, by]

Which rows?

- vector of rownumbers
- logical vector
- another data.table

What to do?

- summarizing
- updating variable(s)
- adding variable(s)

Grouped by what?

- one or more columns
- on the fly grouping var(s)

Joining datasets

Example data

```
irisDT <- copy(iris)
setDT(irisDT)

irisH <- data.table(Species = c("setosa","versicolor","virginica"),
                    Species.full = c("Iris setosa","Iris versicolor","Iris virginica"),
                    height = 1:3,
                    soil = c("mud","rock","sand"))
```

Joining datasets

syntax: DT[**i**, **on**, **j**, **by**]

```
irisDT[irisH, on = .(Species)]

irisDT[irisH, on = "Species"]

irisDT[irisH, on = .(Species = Spec, other_col)]
```

Joining datasets

syntax: DT[**i**, **on**, **j**, **by**]

```
irisDT[irisH, on = .(Species), Species.full := Species.full]

irisDT[irisH
       , on = .(Species)
       , `:=` (Species.full = Species.full, height = height, soil = soil)]

irisDT[irisH
       , on = .(Species)
       , `:=` (Species.full = i.Species.full, height = i.height, soil = i.soil)]
```

Joining & chaining

syntax: DT[**i**, **on**, **j**, **by**]

like %>% from the tidyverse, you can also chain data.table operations together

```
irisDT[ … ][ … ][ … ]
```

```
irisDT[irisH, on = .(Species), Species.full := Species.full
         ][, median(Sepal.Length), by = Species.full]
```

Exercise 4

Open the file **ex4.R**

- Use a join to add the month name from 'airmonths' to 'air'
- Use a join to add both the month name and the month abbreviation from 'airmonths' to 'air'
- Use a join to add the month name from 'airmonths' to 'air'; then use chaining to calculate the median Wind speed for each month name

Reshaping data

From wide to long: irisMelted <- melt(irisDT, id = "Species")

```
melt(data, id.vars, measure.vars,
        variable.name = "variable",
        value.name = "value",
        na.rm = FALSE,
        variable.factor = TRUE,
        value.factor = FALSE)
```

See also: ?melt

Reshaping data

From long to wide: dcast(irisMelted, Species ~ variable)

```
dcast(data, formula,
        fun.aggregate = NULL, sep = "_", ...,
        margins = NULL, subset = NULL,
        fill = NULL, drop = TRUE,
        value.var = guess(data))
```

See also: ?dcast

What else is there to discover?

more joins:	non-equi joins + rolling joins
more special symbols:	.BY + .GRP
special grouping functions:	rowid + rleid
set* functions:	setkey + setorder + setcolorder + setnames + …..
and even more:	frank + shift + CJ + tstrsplit + …..

Want to learn more?

Overview of getting started vignettes

Datacamp’s data.table course (paid)

StackOverflow [data.table] tag (> 7700 questions)

The End

Thank you for your attention!