what you can do with data.frame that you can't in data.table

I just started using R, and came across data.table. I found it brilliant.

A very naive question: Can I ignore data.frame to use data.table to avoid syntax confusion between two packages?

r data.frame data.table

edited Nov 29 '12 at 3:56
mnel
51.5k ●6 ●85 ●116

asked Nov 29 '12 at 3:46
AdamNYC
3,121 ●6 ●32 ●83

9 See the data.table faq specifically 1.8 and 2.17. There will be cases where other packages may rely some strange coding that falls down, but I haven't come across any (that haven't been fixed already). – mnel Nov 29 '12 at 3:51

@mnel I think your comment is the answer. – Brandon Bertelsen Nov 29 '12 at 7:24

5 The `data.table` package is relatively new (originated in 2006). I suspect that evolutionary pressures will lead to (mostly) abandonment of `data.frame` rather as `ggplot` will replace `lattice` . – Carl Witthoft Nov 29 '12 at 12:20

If it comes to plotting, you might have to use `as.data.frame()` in order to use your data with ggplot2. So no - you can't (yet) ignore it completely. – jakob r Nov 29 '12 at 20:36

2 You don't need as.data.frame to work with the most recent versions of ggplot2. – mnel Nov 29 '12 at 21:27

add a comment

1 Answer

From the data.table FAQ

FAQ 1.8 OK, I'm starting to see what data.table is about, but why didn't you enhance data.frame in R? Why does it have to be a new package?

As FAQ 1.1 highlights, `j` in `[.data.table` is fundamentally different from `j` in `[.data.frame` . Even something as simple as `DF[,1]` would break existing code in many packages and user code. This is by design, and we want it to work this way for more complicated syntax to work. There are other differences, too (see FAQ 2.17).

Furthermore, `data.table` inherits from `data.frame` . It is a `data.frame` , too. A `data.table` can be passed to any package that only accepts `data.frame` and that package can use `[.data.frame` syntax on the `data.table` .

We have proposed enhancements to R wherever possible, too. One of these was accepted as a new feature in R 2.12.0 :

> `unique()` and `match()` are now faster on character vectors where all elements are in the global `CHARSXP` cache and have unmarked encoding (ASCII). Thanks to Matthew Dowle for suggesting improvements to the way the hash code is generated in `unique.c` .

A second proposal was to use `memcpy` in `duplicate.c` , which is much faster than a for loop in C. This would improve the way that R copies data internally (on some measures by 13 times). The thread on r-devel is here : http://tolstoy.newcastle.edu.au/R/e10/devel/10/04/0148.html.

2.17 What are the smaller syntax differences between data.frame and

data.table?

- `DT[3]` refers to the 3rd row, but `DF[3]` refers to the 3rd column
- `DT[3,] == DT[3],` but `DF[,3] == DF[3]` (somewhat confusingly)
- For this reason we say the comma is optional in DT, but not optional in DF
- `DT[[3]] == DF[3] == DF[[3]]`
- `DT[i,]` where i is a single integer returns a single row, just like `DF[i,]` , but unlike a matrix single row subset which returns a vector.
- `DT[,j,with=FALSE]` where j is a single integer returns a one column data.table, unlike `DF[,j]` which returns a vector by default
- `DT[,"colA",with=FALSE][[1]] == DF[,"colA"]` .
- `DT[,colA] == DF[,"colA"]`
- `DT[,list(colA)] == DF[,"colA",drop=FALSE]`
- `DT[NA]` returns 1 row of NA, but `DF[NA]` returns a copy of DF containing NA throughout.
- The symbol `NA` is type logical in R, and is therefore recycled by `[.data.frame` . Intention wasprobably `DF[NA_integer_]` . `[.data.table` does this automatically for convenience.
- `DT[c(TRUE,NA,FALSE)]` treats the NA as FALSE, but `DF[c(TRUE,NA,FALSE)]` returns NA rows for each `NA`
- `DT[ColA==ColB]` is simpler than `DF[!is.na(ColA) & !is.na(ColB) & ColA==ColB,]`
- `data.frame(list(1:2,"k",1:4))` creates 3 columns, `data.table` creates one list column.
- `check.names` is by default `TRUE` in `data.frame` but `FALSE` in `data.table` , for convenience.
- `stringsAsFactors` is by default TRUE in `data.frame` but FALSE in `data.table` , for efficiency.
- Since a global string cache was added to R, characters items are a pointer to the single cached string and there is no longer a performance benefit of coverting to factor.
- Atomic vectors in list columns are collapsed when printed using ", " in data.frame, but "," in data.table with a trailing comma after the 6th item to avoid accidental printing of large embedded objects.
- In `[.data.frame` we very often set `drop=FALSE` . When we forget, bugs can arise in edge cases where single columns are selected and all of a sudden a vector is returned rather than a single column data.frame. In `[.data.table` we took the opportunity to make it consistent and drop drop.
- When a data.table is passed to a data.table-unaware package, that package it not concerned with any of these differences; it just works

Small caveat

There will possibly be cases where some packages use code that falls down when given a data.frame, however, given that `data.table` is constantly being maintained to avoid such problems, any problems that may arise will be fixed promptly.

For example

- see this question and prompt response
- From the NEWS for v 1.8.2

- base::unname(DT) now works again, as needed by plyr::melt(). Thanks to Christoph Jaeckel for reporting. Test added.
- An as.data.frame method has been added for ITime, so that ITime can be passed to ggplot2 without error, #1713. Thanks to Farrel Buchinsky for reporting. Tests added. ITime axis labels are still displayed as integer seconds from midnight; we don't know why ggplot2 doesn't invoke ITime's as.character method. Convert ITime to POSIXct for ggplot2, is one approach.

edited Nov 30 '12 at 0:11

community wiki
3 revs, 2 users 91%
mnel

add a comment

Not the answer you're looking for? Browse other questions tagged r data.frame data.table **or ask your own question.**