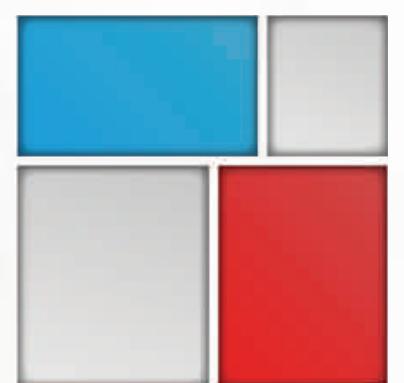


THE #RDATABLE PACKAGE

for fast, flexible and memory efficient data wrangling



Arun
Srinivasan
UseR'15



OPEN ANALYTICS

WHO AM I?

- Bioinformatician, Comp. Biologist
- Started using R in mid 2011



WHO AM I?

- Tried plyr and plyr in parallel
- Discovered data.table. Never looked back



WHO AM I?

- data.table developer since late 2013
- Data analyst at Open Analytics since Feb 2015



COME VISIT US



OVERVIEW

- Perform operations **straightforward**,
without compromising in **efficiency**



1. FREAD

```
fread('file.csv')
```

1.8.8, MARCH 2013
67 ISSUES CLOSED
- 26 FEATURES
- 17 BUG FIXES

- sep, colClasses, rows automatically detected



#Rdatatable



BENCHMARK

23GB .csv, 500 million rows, 9 columns

Method	Run Time	Threadedness
h2o.importFile	50s	Multiple
fread	5m	Single
readr::read_csv	12m	Single



FWRITE?

[R-Forge #2622] Add command "fwrite" to faster save csv files #580

① Open

arunsrinivasan opened this issue on Jun 8, 2014 · 6 comments



arunsrinivasan commented on Jun 8, 2014

Owner

Labels

feature request

Submitted by: Ma Jom; Assigned to: Nobody; R-Forge link

Milestone

No milestone

I really love the new command "fread" as it really speeds up the work with large files. It would be great to see a corresponding command for writing CSV files.

Assignee

No one—assign yourself

Currently, I use a combination of write.csv and as.matrix to speed up this process a bit:

<http://stackoverflow.com/questions/10505605/speeding-up-the-performance-of-write-table>

robbyjo referenced this issue on Jun 10, 2014

Feature request: fwrite for data.table #690

② Closed

Notifications

◀ Unsubscribe



2. GROUPED AGGREGATIONS

year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6



year	val
2014	5
2015	12

2. GROUPED AGGREGATIONS

year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6

+



+

year	val
2014	5
2015	12

2. GROUPED AGGREGATIONS

```
x[year %in% 2014:2015, .(val = sum(val)), by = year]
```

year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6

+



year	val
2014	5
2015	12

+

2. GROUPED AGGREGATIONS

```
x[year %in% 2014:2015, .(val = sum(val)), by = year]
```

NO MORE X\$

EXPRESSION

GROUP BY

2. GROUPED AGGREGATIONS

```
x[year %in% 2014:2015, .(val = sum(val)), by = year]
```

I, ON WHICH ROWS?

J, WHAT TO DO?

BY, GROUPED BY WHAT?

VIGNETTES

<https://github.com/Rdatatable/data.table/wiki/Getting-started>

The screenshot shows a GitHub page for the 'Getting started' guide of the 'Rdatatable / data.table' repository. The page has a light gray background with dark gray text. At the top, there's a header bar with the repository name, a 'Unwatch' button, an 'Unstar' button (with 377 stars), a 'Fork' button (with 271 forks), and a 'New Page' button. Below the header, the title 'Getting started' is displayed in large bold letters. A small note below it says 'arunsrinivasan edited this page 8 days ago · 8 revisions'. The main content area contains three sections: 'HTML vignettes (NEW)', 'PDF Vignettes', and 'Documentation and examples'. Each section lists several links. To the right of the main content, there's a sidebar with navigation links: 'Pages', 'Home', 'Installation', 'How to file a bug report', and 'Getting started'. The bottom of the page features a footer with social media icons for GitHub, LinkedIn, and Twitter.

Rdatatable / data.table

Unwatch 88 Unstar 377 Fork 271

Getting started

arunsrinivasan edited this page 8 days ago · 8 revisions

HTML vignettes (NEW): i) [Introduction to data.table](#), ii) [Reference semantics](#), iii) [Keys and fast binary search based subsets](#), iv) [Efficient reshaping using data.tables](#)

PDF Vignettes: i) [Quick start guide](#), ii) [Frequently asked questions](#)

Documentation and examples: i) [?data.table](#), ii) [?fread](#)

Pages

Home

Installation

How to file a bug report

Getting started



3. AGGREGATIONS ON JOINS

year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6

year	mul
2014	20
2015	10

3. AGGREGATIONS ON JOINS

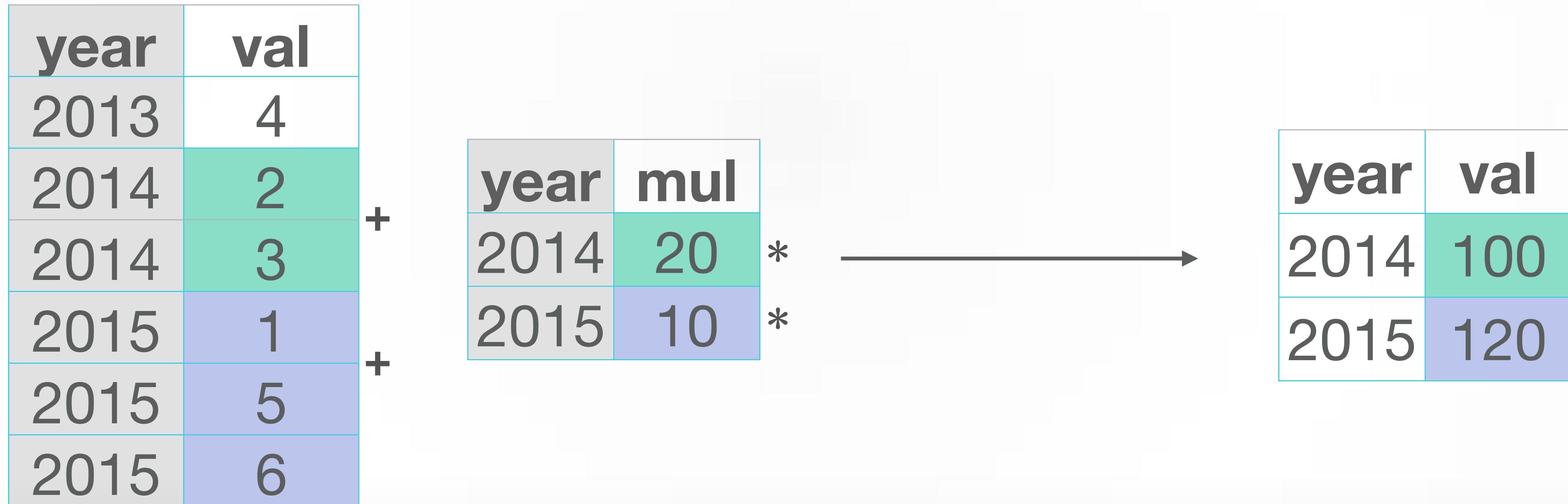
year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6

year	mul
2014	20
2015	10



year	val
2014	100
2015	120

3. AGGREGATIONS ON JOINS



3. AGGREGATIONS ON JOINS

```
X[Y, .(val = sum(val) * mul), by = .EACHI]
```

year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6

+

year	mul
2014	20
2015	10

*



year	val
2014	100
2015	120

3. AGGREGATIONS ON JOINS

```
X[Y, .(val = sum(val) * mul), by = .EACHI]
```

year	val
2013	4
2014	2
2014	3
2015	1
2015	5
2015	6

+

year	mul
2014	20
2015	10

+

*



year	val
2014	100
2015	120

3. AGGREGATIONS ON JOINS

WHAT TO DO?

```
x[Y, .(val = sum(val) * mul), by = .EACHI]
```

ON WHICH ROWS?

GROUPED BY WHAT?

3. AGGREGATIONS ON JOINS

WHAT TO DO?

```
x[Y, .(val = sum(val) * mul), by = .EACHI]
```

ON WHICH ROWS?

GROUPED BY WHAT?

Joins as Subsets



4. RESHAPING

id	A1	A2	B1	B2
a	1	3	q	s
b	2	4	r	t

id	num	A	B
a	1	1	q
b	1	2	r
a	2	3	s
b	2	4	t

4. RESHAPING

id	A1	A2	B1	B2
a	1	3	q	s
b	2	4	r	t

COERCED TO CHAR

melt

id	var	val

split
var

UNCLEAR

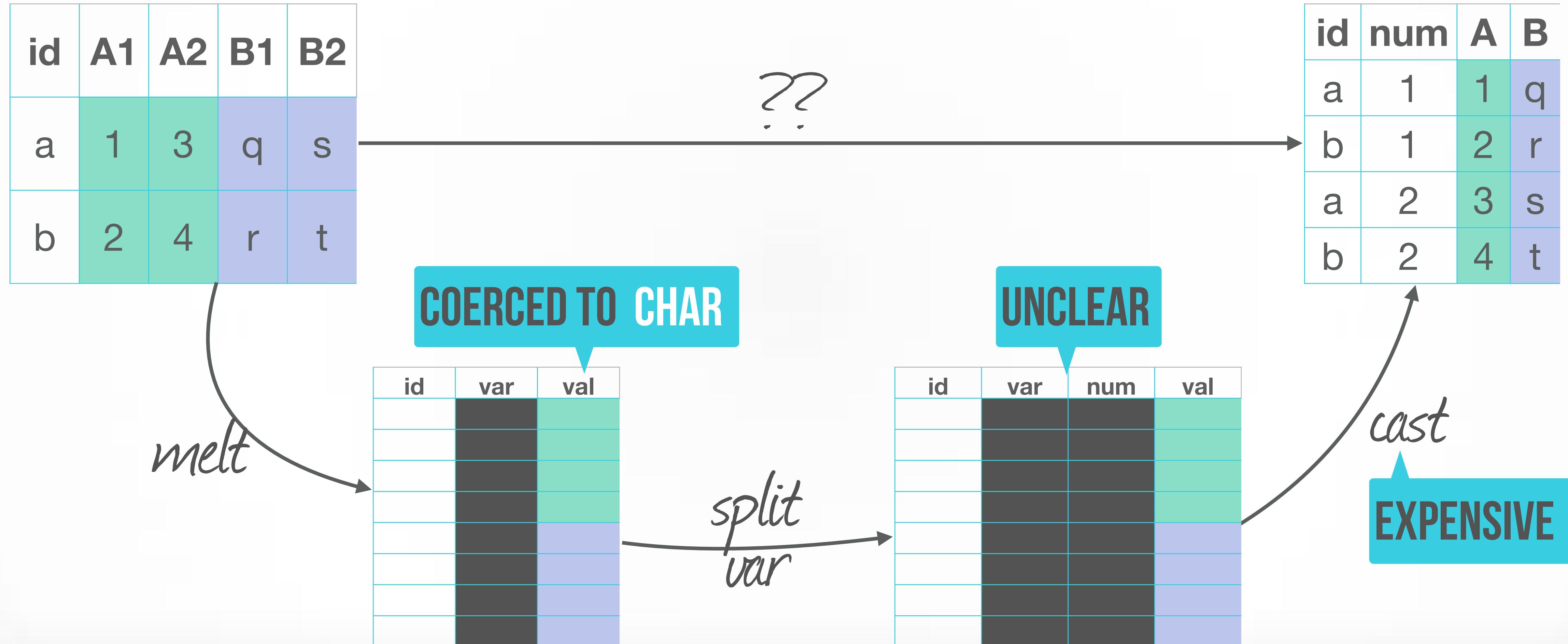
id	var	num	val

id	num	A	B
a	1	1	q
b	1	2	r
a	2	3	s
b	2	4	t

cast

EXPENSIVE

4. RESHAPING



4. RESHAPING

id	A1	A2	B1	B2
a	1	3	q	s
b	2	4	r	t

```
melt(DT, measure = patterns("^A", "^B"))
```

NEW FEATURE IN 1.9.6

id	num	A	B
a	1	1	q
b	1	2	r
a	2	3	s
b	2	4	t

BENCHMARK

27MB .csv, 500 thousand rows, 8 cols

Method	Run Time
base::reshape	4.48s
melt (v1.9.6)	0.05s
tidyR	9.41s



4. RESHAPING

id	A1	A2	B1	B2
a	1	3	q	s
b	2	4	r	t

```
dcast(DT, id ~ num, value.var = c("A", "B"))
```

NEW FEATURE IN 1.9.6

id	num	A	B
a	1	1	q
b	1	2	r
a	2	3	s
b	2	4	t

THANKS TO

ANANDA MAHTO for ideas on reshaping enhancements.
Also check out [SPLITSTACKSHAPE](#).



5. OVERLAPPING RANGE JOINS

A			
	chr	start	end
1:	1	5	11
2:	1	10	20
3:	2	1	4
4:	2	25	52
5:	2	50	60

B			
	chr	start	end
1:	1	1	4
2:	1	15	18
3:	2	1	55

| foverlaps(A, B, which=TRUE)

SINCE V1.9.4. BUILT
USING ROLLING JOINS.

RESULT	
xid	yid
1	NA
2	2
3	3
4	3
5	3

5. OVERLAPPING RANGE JOINS

A			
	chr	start	end
1:	1	5	11
2:	1	10	20
3:	2	1	4
4:	2	25	52
5:	2	50	60

B			
	chr	start	end
1:	1	1	4
2:	1	15	18
3:	2	1	55

| foverlaps(A, B, which=TRUE)

SINCE V1.9.4. BUILT
USING ROLLING JOINS.

RESULT	
xid	yid
1	NA
2	2
3	3
4	3
5	3

5. OVERLAPPING RANGE JOINS

A			
	chr	start	end
1:	1	5	11
2:	1	10	20
3:	2	1	4
4:	2	25	52
5:	2	50	60

B			
	chr	start	end
1:	1	1	4
2:	1	15	18
3:	2	1	55

| foverlaps(A, B, which=TRUE)

SINCE V1.9.4. BUILT
USING ROLLING JOINS.

RESULT	
xid	yid
1	NA
2	2
3	3
4	3
5	3

5. OVERLAPPING RANGE JOINS

A			
	id	start	end
1:	1	0.5	1.1
2:	1	1	2
3:	2	0.1	0.4
4:	2	2.5	5.2
5:	2	5	6

B			
	id	start	end
1:	1	0.1	0.4
2:	1	1.5	1.8
3:	2	0.1	5.5

```
| foverlaps(A, B, which=TRUE)
```



RESULT	
xid	yid
1	NA
2	2
3	3
4	3
5	3

Not limited to integer ranges, but numeric, POSIXct,
Date ranges work just the same.

6. AUTO-INDEXING

SINCE V1.9.4

1.2GB .csv, 100 million rows, 2 columns

		Run 1	4.7s
		Run 2	4.8s
data frame	DF[DF\$id %in% c("KIC", "HEJ"),]		
data table	DT[id %in% c("KIC", "HEJ")]		
		Run 1	3.7s
		Run 2	0.002s

INDEXING +
SUBSET

VERSION 1.9.6

- ~80 bug fixes and >20 new features
- many new functions : `rleid()`,
`tstrsplit()`, `shift()`, `frank()`, `na.omit()`
- `melt()` and `dcast()`

ACKNOWLEDGEMENTS

- Thanks to users and contributors
- Package authors who use `data.table`
(CRAN - 84, bioconductor - 32)
- My colleagues and Matt
- And you for listening! :-)

QUESTIONS?

