



# Using data.table

Jeffrey Leek

Johns Hopkins Bloomberg School of Public Health

# data.table

- Inherits from data.frame
  - All functions that accept data.frame work on data.table
- Written in C so it is much faster
- Much, much faster at subsetting, group, and updating

# Create data tables just like data frames

```
library(data.table)
DF = data.frame(x=rnorm(9),y=rep(c("a","b","c"),each=3),z=rnorm(9))
head(DF,3)
```

	x	y	z
1	0.4159	a	-0.05855
2	0.8433	a	0.13732
3	1.0585	a	2.16448

```
DT = data.table(x=rnorm(9),y=rep(c("a","b","c"),each=3),z=rnorm(9))
head(DT,3)
```

	x	y	z
1:	-0.27721	a	0.2530
2:	1.00158	a	1.5093
3:	-0.03382	a	0.4844

# See all the data tables in memory

```
tables()
```

```
      NAME NROW MB COLS  KEY  
[1,] DT      9 1  x,y,z  
Total: 1MB
```

# Subsetting rows

```
DT[2, ]
```

```
      x y      z  
1: 1.002 a 1.509
```

```
DT[DT$y=="a", ]
```

```
      x y      z  
1: -0.27721 a 0.2530  
2:  1.00158 a 1.5093  
3: -0.03382 a 0.4844
```

# Subsetting rows

```
DT[c(2,3)]
```



	x	y	z
1:	1.00158	a	1.5093
2:	-0.03382	a	0.4844

# Subsetting columns!?

```
DT[,c(2,3)]
```

```
[1] 2 3
```

# Column subsetting in data.table

- The subsetting function is modified for data.table
- The argument you pass after the comma is called an "expression"
- In R an expression is a collection of statements enclosed in curly brackets

```
{  
  x = 1  
  y = 2  
}  
k = {print(10); 5}
```

```
[1] 10
```

```
print(k)
```

```
[1] 5
```



# Calculating values for variables with expressions

```
DT[,list(mean(x),sum(z))]
```

```
      V1      V2  
1: 0.05637 0.5815
```

```
DT[,table(y)]
```

```
y  
a b c  
3 3 3
```

# Adding new columns

```
DT[,w:=z^2]
```

	x	y	z	w
1:	-0.27721	a	0.25300	0.064009
2:	1.00158	a	1.50933	2.278091
3:	-0.03382	a	0.48437	0.234619
4:	-0.70493	b	-1.22755	1.506885
5:	-1.36402	b	-0.64624	0.417631
6:	-0.26224	b	-0.51427	0.264475
7:	-0.10929	c	1.21445	1.474901
8:	1.40234	c	0.07493	0.005614
9:	0.85494	c	-0.56652	0.320948

# Adding new columns

```
DT2 <- DT
```



```
DT[, y:= 2]
```

	x	y	z	w
1:	-0.27721	2	0.25300	0.064009
2:	1.00158	2	1.50933	2.278091
3:	-0.03382	2	0.48437	0.234619
4:	-0.70493	2	-1.22755	1.506885
5:	-1.36402	2	-0.64624	0.417631
6:	-0.26224	2	-0.51427	0.264475
7:	-0.10929	2	1.21445	1.474901
8:	1.40234	2	0.07493	0.005614
9:	0.85494	2	-0.56652	0.320948

# Careful

```
head(DT,n=3)
```

```
      x y      z      w  
1: -0.27721 2 0.2530 0.06401  
2:  1.00158 2 1.5093 2.27809  
3: -0.03382 2 0.4844 0.23462
```

```
head(DT2,n=3)
```



```
      x y      z      w  
1: -0.27721 2 0.2530 0.06401  
2:  1.00158 2 1.5093 2.27809  
3: -0.03382 2 0.4844 0.23462
```

# Multiple operations

```
DT[,m:= {tmp <- (x+z); log2(tmp+5)}]
```

	x	y	z	w	m
1:	-0.27721	2	0.25300	0.064009	2.315
2:	1.00158	2	1.50933	2.278091	2.909
3:	-0.03382	2	0.48437	0.234619	2.446
4:	-0.70493	2	-1.22755	1.506885	1.617
5:	-1.36402	2	-0.64624	0.417631	1.580
6:	-0.26224	2	-0.51427	0.264475	2.078
7:	-0.10929	2	1.21445	1.474901	2.610
8:	1.40234	2	0.07493	0.005614	2.695
9:	0.85494	2	-0.56652	0.320948	2.403

# plyr like operations

```
DT[, a:=x>0]
```

	x	y	z	w	m	a
1:	-0.27721	2	0.25300	0.064009	2.315	FALSE
2:	1.00158	2	1.50933	2.278091	2.909	TRUE
3:	-0.03382	2	0.48437	0.234619	2.446	FALSE
4:	-0.70493	2	-1.22755	1.506885	1.617	FALSE
5:	-1.36402	2	-0.64624	0.417631	1.580	FALSE
6:	-0.26224	2	-0.51427	0.264475	2.078	FALSE
7:	-0.10929	2	1.21445	1.474901	2.610	FALSE
8:	1.40234	2	0.07493	0.005614	2.695	TRUE
9:	0.85494	2	-0.56652	0.320948	2.403	TRUE

# plyr like operations

```
DT[,b:= mean(x+w),by=a]
```

	x	y	z	w	m	a	b
1:	-0.27721	2	0.25300	0.064009	2.315	FALSE	0.2018
2:	1.00158	2	1.50933	2.278091	2.909	TRUE	1.9545
3:	-0.03382	2	0.48437	0.234619	2.446	FALSE	0.2018
4:	-0.70493	2	-1.22755	1.506885	1.617	FALSE	0.2018
5:	-1.36402	2	-0.64624	0.417631	1.580	FALSE	0.2018
6:	-0.26224	2	-0.51427	0.264475	2.078	FALSE	0.2018
7:	-0.10929	2	1.21445	1.474901	2.610	FALSE	0.2018
8:	1.40234	2	0.07493	0.005614	2.695	TRUE	1.9545
9:	0.85494	2	-0.56652	0.320948	2.403	TRUE	1.9545

# Special variables

`.N` An integer, length 1, containing the number



```
set.seed(123);  
DT <- data.table(x=sample(letters[1:3], 1E5, TRUE))  
DT[, .N, by=x]
```

```
      x      N  
1: a 33387  
2: c 33201  
3: b 33412
```



# Keys

```
DT <- data.table(x=rep(c("a", "b", "c"), each=100), y=rnorm(300))  
setkey(DT, x)  
DT[ 'a' ]
```

	x	y
1:	a	0.25959
2:	a	0.91751
3:	a	-0.72232
4:	a	-0.80828
5:	a	-0.14135
6:	a	2.25701
7:	a	-2.37955
8:	a	-0.45425
9:	a	-0.06007
10:	a	0.86090
11:	a	-1.78466
12:	a	-0.13074
13:	a	-0.36984
14:	a	-0.18066
15:	a	-1.04973

# Joins

```
DT1 <- data.table(x=c('a', 'a', 'b', 'dt1'), y=1:4)
DT2 <- data.table(x=c('a', 'b', 'dt2'), z=5:7)
setkey(DT1, x); setkey(DT2, x)
merge(DT1, DT2)
```

```
   x y z
1: a 1 5
2: a 2 5
3: b 3 6
```

# Fast reading

```
big_df <- data.frame(x=rnorm(1E6), y=rnorm(1E6))  
file <- tempfile()  
write.table(big_df, file=file, row.names=FALSE, col.names=TRUE, sep="\t", quote=FALSE)  
system.time(fread(file))
```





user	system	elapsed
0.312	0.015	0.326

```
system.time(read.table(file, header=TRUE, sep="\t"))
```



user	system	elapsed
5.702	0.048	5.755

# Summary and further reading

- The latest development version contains new functions like `melt` and `dcast` for `data.tables`
  - <https://r-forge.r-project.org/scm/viewvc.php/pkg/NEWS?view=markup&root=datatable> 
- Here is a list of differences between `data.table` and `data.frame`
  - <http://stackoverflow.com/questions/13618488/what-you-can-do-with-data-frame-that-you-cant-in-data-table> 
- Notes based on Raphael Gottardo's notes [https://github.com/raphg/Biostat-578/blob/master/Advanced\\_data\\_manipulation.Rpres](https://github.com/raphg/Biostat-578/blob/master/Advanced_data_manipulation.Rpres), who got them from Kevin Ushey.