# R For Data Science *Cheat Sheet*

Simple | Smart | Speed

## data.table

**data.table** is an R package that provides a high-performance version of base R's data.frame with syntax and feature enhancements for ease of use, convenience and programming speed.

Load the package:

```
> library(data.table)
```

## Creating A data.table

```
> set.seed(45L)
> DT <- data.table(V1=c(1L,2L),
V2=LETTERS[1:3],
 V3=round(rnorm(4),4),
 V4=1:12)
```

Create a data.table and call it DT

## Subsetting Rows Using i

```
> DT[3:5,]
> DT[3:5]
> DT[V2=="A"]
> DT[V2 %in% c("A","C")]
```

Select 3rd to 5th row
Select 3rd to 5th row
Select all rows that have A in column V2 value Select all rows that A or C in column V2 have value

## Manipulating on Columns in j

```
> DT[,V2]
[1] "A" "B" "C" "A" "B" "C"
> DT[,.(V2,V3)]
> DT[,sum(V1)]
[1] 18
> DT[,.(sum(V1),sd(V3))]
  V1 V2    1: 18
0.4546055
> DT[,.(Aggregate=sum(V1),
Sd.V3=sd(V3))]
   Aggregate Sd.V3   1: 18
0.4546055
> DT[,.(V1,Sd.V3=sd(V3))]
> DT[,.(print(V2),
plot(V3),
NULL)]
```

Return as a vector
Return V2 and V3 as a data.table
Return the sum of all elements of V1 in a vector
Return the sum of all elements of V1 and the std. dev. of V3 in a data.table
The same as the above, with new names
Select column V2 and compute std. dev. of V3, which returns a single value and gets recycled
Print column V2 and plot V3

## Doing j by Group

```
> DT[,.(V4.Sum=sum(V4)),by=V1]
   V1  V4.Sum
1: 1 36
 2: 2 42

> DT[,.(V4.Sum=sum(V4)),
by=.(V1,V2)]
> DT[,.(V4.Sum=sum(V4)),
          by=sign(V1-1)]
  sign  V4.Sum
1: 0 36
2: 1 42
```

Calculate sum of V4 for every group in V1
Calculate sum of V4 for every group in V1 and V2
Calculate sum of V4 for every group in sign(V1-1)

> DT[,.(V4.Sum=sum(V4)), The same as the above, with new name by=.(V1.01=sign(V1-1))] for the variable you're grouping by
> DT[1:5,.(V4.Sum=sum(V4)), Calculate sum of V4 for every group in V1 by=V1] after subsetting on the first 5 rows
> DT[,.N,by=V1] Count number of rows for every group in V1

**General form:** `DT[i, j, by]`

"Take j, DT, subset rows using i, then calculate grouped by by"

## Adding/Updating Columns By Reference in j Using :=

```
> DT[,V1:=round(exp(V1),2)]
> DT
 V1 V2 V3 V4  1: 2.72 A
-0.1107 1
 2: 7.39 B -0.1427 2
 3: 2.72 C -1.8893 3
 4: 7.39 A -0.3571 4
 ...

> DT[,c("V1","V2"):=list(round(exp(V1),2),
LETTERS[4:6])]
> DT[,':='(V1=round(exp(V1),2),
V2=LETTERS[4:6])][]
 V1 V2 V3 V4   1: 15.18 D
-0.1107 1
 2: 1619.71 E -0.1427 2
 3: 15.18 F -1.8893 3
 4: 1619.71 D -0.3571 4
> DT[,V1:=NULL] Remove V1
> DT[,c("V1","V2"):=NULL] Remove columns V1 and V2
> Cols.chosen=c("A","B")
> DT[,Cols.Chosen:=NULL] Delete the column with column name
Cols.chosen
> DT[,(Cols.Chosen):=NULL] Delete the columns specified in the variable
Cols.chosen
```

V1 is updated by what is after := Return the result by calling DT

Columns V1 and V2 are updated by what is after := Alternative to the above one. With [], you print the result to the screen

## Indexing And Keys

```
> setkey(DT,V2)
> DT["A"]
 V1 V2 V3 V4
1: 1 A -0.2392 1
 2: 2 A -1.6148 4
 3: 1 A 1.0498 7
 4: 2 A 0.3262 10
> DT[c("A","C")] Return all rows where the key column (V2) has value A or C
> DT["A",mult="first"] Return first row of all rows that match value A in key column V2
> DT["A",mult="last"] Return last row of all rows that match value A in key column V2
> DT[c("A","D")] Return all rows where key column V2 has value A or D
 V1 V2 V3 V4
 1: 1 A -0.2392 1
 2: 2 A -1.6148 4
 3: 1 A 1.0498 7
 4: 2 A 0.3262 10
 5: NA D NA NA

> DT[c("A","D"),nomatch=0]
 V1 V2 V3 V4   1: 1 A
-0.2392 1    2:  2  A
-1.6148  4    3:  1  A
1.0498  7    4:  2  A
0.3262 10

> DT[c("A","C"),sum(V4)]
> DT[c("A","C"),
sum(V4),
by=.EACHI]
 V2 V1   1:
A 22   2: C
30
> setkey(DT,V1,V2)
> DT[.(2,"C")]
          V1 V2 V3 V4
 1: 2 C 0.3262 6   2:
2 C -1.6148 12

> DT[.(2,c("A","C"))]
          V1 V2 V3 V4
 1: 2 A -1.6148 4
 2: 2 A 0.3262 10
 3: 2 C 0.3262 6
 4: 2 C -1.6148 12
```

A key is set on V2; output is returned invisibly Return all rows where the key column (set to the value A V2) has

Return all rows where key column V2 has value A or D

Return total sum of V4, for rows of key column V2 that have values A or C
Return sum of column V4 for rows of V2 that have value A, and another sum for rows of V2 that have value C

Sort by V1 and then by V2 within each group of V1 (invisible)
Select rows that have value 2 for the first key ( V1) and the value C for the second key (V2)

Select rows that have value 2 for the first key ( V1) and within those rows the value A or C for the second key ( V2)

## Advanced Data Table Operations

```
> DT[.N-1]
> DT[,.N]
> DT[,.(V2,V3)]
> DT[,list(V2,V3)]
> DT[,mean(V3),by=.(V1,V2)]
          V1 V2 V1
 1: 1 A 0.4053  2:
1 B 0.4053  3: 1 C
0.4053    4:  2  A
-0.6443    5:  2  B
-0.6443    6:  2  C
-0.6443
```

Return the penultimate row of the DT
Return the number of rows
Return V2 and V3 as a data.table
Return V2 and V3 as a data.table
Return the result of j, grouped by all possible combinations of groups specified in by

### .SD & .SDcols

```
> DT[,print(.SD),by=V2]
> DT[,.SD[c(1,.N)],by=V2]
> DT[,lapply(.SD,sum),by=V2]

> DT[,lapply(.SD,sum),by=V2,
.SDcols=c("V3","V4")]
 V2 V3 V4   1: A
-0.478 22
 2: B -0.478 26
 3: C -0.478 30

> DT[,lapply(.SD,sum),by=V2,
      .SDcols=paste0("V",3:4)]
```

Look at what .SD contains
Select the first and last row grouped by V2
Calculate sum of columns in .SD grouped by V2
Calculate sum of V3 and V4 in .SD grouped by V2
Calculate sum of V3 and V4 in .SD grouped by V2

## Chaining

```
> DT <- DT[,.(V4.Sum=sum(V4)),
by=V1]
 V1 V4.Sum  1:
1 36  2: 2 42

> DT[V4.Sum>40]
> DT[,.(V4.Sum=sum(V4)),
by=V1][V4.Sum>40]
 V1 V4.Sum  1:
2 42

> DT[,.(V4.Sum=sum(V4)),
by=V1][order(-V1)]
 V1 V4.Sum  1:
2 42  2: 1 36
```

Calculate sum of V4, grouped by V1

Select that group of which the sum is >40 Select that group of which the sum is >40 (chaining)

Calculate sum of V4, grouped by V1, ordered on V1

## set()-Family

### set()

Syntax: `for (i in from:to) set(DT, row, column, new value)`

```
> rows <- list(3:4,5:6)
> cols <- 1:2
> for(i in seq_along(rows))
{set(DT,
i=rows[[i]],
j=cols[i],
value=NA)}
```

Sequence along the values of rows, and for the values of cols, set the values of those elements equal to NA (invisible)

### setnames()

Syntax: `setnames(DT,"old","new")[]`

```
> setnames(DT,"V2","Rating") V2
> setnames(DT,
c("V2","V3"),
 c("V2.rating","V3.DC"))
```

Set name of to Rating (invisible)
Change 2 column names (invisible)

### setnames()

Syntax: `setcolorder(DT,"neworder")`

```
> setcolorder(DT,
c("V2","V1","V4","V3"))
```

Change column ordering to contents of the specified vector (invisible)