

# R For Data Science Cheat Sheet

## data.table

Learn R for data science [Interactively at www.DataCamp.com](https://www.datacamp.com)



### 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 value A in column v2  
 Select all rows that have value A or C in column v2

### 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))]
  v1      v2
1: 18 0.4546055
> DT[, .(print(V2),
         plot(V3),
         NULL)]
```

Return v2 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
> DT[, .(V4.Sum=sum(V4)), by=. (V1.01=sign(V1-1))]
> DT[1:5, .(V4.Sum=sum(V4)), by=V1]
> DT[, .N, by=V1]
```

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)  
 The same as the above, with new name for the variable you're grouping by  
 Calculate sum of v4 for every group in v1 after subsetting on the first 5 rows  
 Count number of rows for every group in v1

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

"Take DT, subset rows using i, then calculate j 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]
> DT[, c("V1", "V2") :=NULL]
> Cols.Chosen=c("A", "B")
> DT[, Cols.Chosen:=NULL]
> DT[, (Cols.Chosen) :=NULL]
```

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  
 Remove v1  
 Remove columns v1 and v2  
 Delete the column with column name Cols.Chosen  
 Delete the columns specified in the variable Cols.Chosen

### 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")]
> DT["A", mult="first"]
> DT["A", mult="last"]
> DT[c("A", "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 v2) has the value a  
 Return all rows where the key column (v2) has value A or C  
 Return first row of all rows that match value A in key column v2  
 Return last row of all rows that match value A in key column v2  
 Return all rows where key column v2 has value A or D  
 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
2:  1      36
> 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")
> setnames(DT,
           c("V2", "V3"),
           c("V2.rating", "V3.DC"))
```

Set name of v2 to Rating (invisible)  
 Change 2 column names (invisible)

### setcolorder()

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

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