DATA MANIPULATION IN R WITH DATA.TABLE

Adding and updating columns by reference

Matt Dowle, Arun Srinivasan
Instructors, DataCamp

data.frame internals

Let's say we would like to change the 2nd row of column "y" to 10

```
df <- data.frame(x = 1:5, y = 6:10)
df
  x  y
1 1  6
2 2  7
 ...
```

```
df$y[2] <- 10
```

data.frame internals

In R < v3.1.0, this operation resulted in *deep* copying the entire data.frame

```
# what happens internally prior to R v3.1.0
tmp <- <deep copy of "df">
tmp$y[2] <- 10
df <- tmp
```

- What happens if you would like to do the same operation on a 10GB data.frame?

data.frame internals

- In v3.1.0, improvements were made to deep copy *only* the column that is updated
- In this case, just columns `a` and `b` are deep copied in the operation performed on `df` below

```
df <- data.frame(a = 1:3, b = 4:6, c = 7:9, d = 10:12)
df[1:2] <- lapply(df[1:2], function(x) ifelse(x%%2, x, NA))
df
   a  b c  d
1  1 NA 7 10
2 NA  5 8 11
3  3 NA 9 12
```

data.table internals

- `data.table` updates columns *in place*, i.e., by reference
- This means, you don't need the assign the result back to a variable
- No copy of any column is made while their values are changed
- `data.table` uses a new operator `:=` to add/update/delete columns *by reference*

LHS := RHS form

```
batrips[, c("is_dur_gt_1hour", "week_day") := list(duration > 3600,
                                                    wday(start_date))]

# When adding a single column quotes aren't necessary
batrips[, is_dur_gt_1hour := duration > 3600]
```

Functional form

```
batrips[, `:=`(is_dur_gt_1hour = NULL,
               start_station = toupper(start_station))]
```

DATA MANIPULATION IN R WITH DATA.TABLE

Let's practice!

DATA MANIPULATION IN R WITH DATA.TABLE

Grouped aggregations

Matt Dowle, Arun Srinivasan
Instructors, DataCamp

Combining := with by

```
ncol(batrips)
[1] 11

batrips[, n_zip_code := .N, by = zip_code]

ncol(batrips)
[1] 12

batrips[, n_zip_code := .N, by = zip_code][]
   trip_id duration ... zip_code  n_zip_code
1:  139545      435 ...    94612        1228
2:  139546      432 ...    94107       36061
3:  139547     1523 ...    94112        2168
...
```

Combining := with by

```
batrips[, n_zip_code := .N, by = zip_code][]
   trip_id duration ... zip_code  n_zip_code
1:  139545      435 ...    94612        1228
2:  139546      432 ...    94107       36061
3:  139547     1523 ...    94112        2168

batrips[n_zip_code > 1000]

          ...  bike_id subscription_type zip_code n_zip_code
     1:            473        Subscriber    94612       1228
     2:            395        Subscriber    94107      36061
     3:            331        Subscriber    94112       2168
     4:            335          Customer    94109       6980
     5:            580          Customer                1541
    ---
248267:            677        Subscriber    94107      36061
248268:            604        Subscriber    94133      15687
248269:            480          Customer    94109       6980
248270:            277          Customer    94109       6980
248271:             56        Subscriber    94105      19899
```

Combining := with by

```
batrips[, n_zip_code := .N, by = zip_code]

zip_1000 <- batrips[n_zip_code > 1000][, n_zip_code := NULL]
```

```
# Same as
zip_1000 <- batrips[, n_zip_code := .N,
                    by = zip_code][n_zip_code > 1000][, n_zip_code := NULL]
```

DATA MANIPULATION IN R WITH DATA.TABLE

Let's practice!

DATA MANIPULATION IN R WITH DATA.TABLE

Advanced aggregations

Matt Dowle, Arun Srinivasan
Instructors, DataCamp

Recap

```
# Same example as seen before
## LHS := RHS Form
batrips[, c("is_dur_gt_1hour", "week_day") :=
            .(duration > 3600, wday(start_date)]
```

```
# Same as above, but in `:=`() functional form
batrips[, `:=`(is_dur_gt_1hour = duration > 3600,
               week_day = wday(start_date))]
```

```
# Update by reference with by
batrips[, n_zip_code := .N, by = zip_code]
```

Adding multiple columns by reference by group

```
# Functional form
batrips[, `:=`(end_dur_first = duration[1],
               end_dur_last  = duration[.N]),
        by = end_station]

# LHS := RHS form
batrips[, c("end_dur_first",
            "end_dur_last") := list(duration[1], duration[.N]),
        by = end_station]

batrips[1:5]
   trip_id duration ...        end_station ... end_dur_first end_dur_last
1:  139545      435 ...    Townsend at 7th ...           435          660
2:  139546      432 ...    Townsend at 7th ...           435          660
3:  139547     1523 ...    Beale at Market ...          1523          229
4:  139549     1620 ... Powell Street BART ...          1620          540
5:  139550     1617 ... Powell Street BART ...          1620          540
```

Binning values

For each unique combination of `start_station` and `end_station`, if median duration:

- less than 600, "short"
- between 600 and 1800, "medium"
- "long", otherwise

Multi-line expressions in j

```
batrips[, trip_category := {
             med_dur = median(duration, na.rm = TRUE)
             if (med_dur < 600) "short"
             else if (med_dur >= 600 & med_dur <= 1800) "medium"
             else "long"
           },
        by = .(start_station, end_station)]
batrips[1:3]
   trip_id duration ... zip_code trip_category
1:  139545      435 ...    94612         short
2:  139546      432 ...    94107         short
3:  139547     1523 ...    94112         short
```

Alternative way

```r
bin_median_duration <- function(dur) {
  med_dur <- median(dur, na.rm = TRUE)
  if (med_dur < 600) "short"
  else if (med_dur >= 600 & med_dur <= 1800) "medium"
  else "long"
}

batrips[, trip_category := bin_median_duration(duration),
          by = .(start_station, end_station)]
```

All together - i, j and by

```
batrips[duration > 500, min_dur_gt_500 := min(duration),
           by = .(start_station, end_station)]
batrips[1:3]
   trip_id duration ... zip_code min_dur_gt_500
1:  139545      435 ...    94612             NA
2:  139546      432 ...    94107             NA
3:  139547     1523 ...    94112            502
```

DATA MANIPULATION IN R WITH DATA.TABLE

Let's practice!