Introduction to R: **Data Tables**

Session 2, Part A

Nick Graetz[1]

[1] University of Pennsylvania, Population Studies Center

9/4/2020

In this lecture

1. Data frames
2. data.table package
3. Creating a data table
4. Viewing a data table
5. Subsetting rows
6. Removing rows
7. Selecting columns
8. Manipulating columns
9. Removing columns

Data frames

So far we've covered three data structures for ordered collections of values:

- Vectors (1 dimension)
- Matrices (2 dimensions)
- Arrays (N dimensions)

These three data structures require that all values be of the same class (e.g., character, numeric, integer, or logical).

Data frames

So far we've covered three data structures for ordered collections of values:

- Vectors (1 dimension)
- Matrices (2 dimensions)
- Arrays (N dimensions)

These three data structures require that all values be of the same class (e.g., character, numeric, integer, or logical).

For spreadsheet-like data, R uses data frames (`data.frame`).

Data frames are essentially collections of vectors of the same length where each vector forms a column of a table; these vectors *do not need to be the same class*.

Creating a data frame from vectors

A data frame can be constructed by combining several related vectors:

```
> iso3 <- c("CAN", "USA", "MEX")
> pop <- c(35.16, 318.9, 122.3)
> admin1 <- c(13L, 51L, 31L)
> spanish <- c(FALSE, FALSE, TRUE)
```

```
> df <- data.frame(iso3, pop, admin1, spanish)
> df
  iso3    pop admin1 spanish
1  CAN  35.16     13   FALSE
2  USA 318.90     51   FALSE
3  MEX 122.30     31    TRUE
```

Data frame structure

You can check if an object is a data frame using `is.data.frame()`:

```
> is.data.frame(df)
[1] TRUE
```

And you can see what class each column is using the `str()` function:

```
> str(df)
'data.frame':   3 obs. of  4 variables:
 $ iso3   : chr  "CAN" "USA" "MEX"
 $ pop    : num  35.2 318.9 122.3
 $ admin1 : int  13 51 31
 $ spanish: logi  FALSE FALSE TRUE
```

DATA.TABLE PACKAGE

The `data.table` package extends data frames to allow for faster operations on large datasets

```
> # install.packages("data.table")
> library(data.table)
```

Creating a data table

A data table can be created from vectors…

```
> dt <- data.table(iso3, pop, admin1, spanish)
> str(dt)
Classes 'data.table' and 'data.frame':  3 obs. of  4 variables
 $ iso3   : chr  "CAN" "USA" "MEX"
 $ pop    : num  35.2 318.9 122.3
 $ admin1 : int  13 51 31
 $ spanish: logi  FALSE FALSE TRUE
 - attr(*, ".internal.selfref")=<externalptr>
> rm(iso3, pop, admin1, spanish) # clean up the work space
```

Creating a data table

or by converting a data frame to a data table

```
> dt <- as.data.table(df)
> str(dt)
Classes 'data.table' and 'data.frame':  3 obs. of  4 variables
 $ iso3   : chr  "CAN" "USA" "MEX"
 $ pop    : num  35.2 318.9 122.3
 $ admin1 : int  13 51 31
 $ spanish: logi  FALSE FALSE TRUE
 - attr(*, ".internal.selfref")=<externalptr>
> rm(df)
```

Viewing a data table

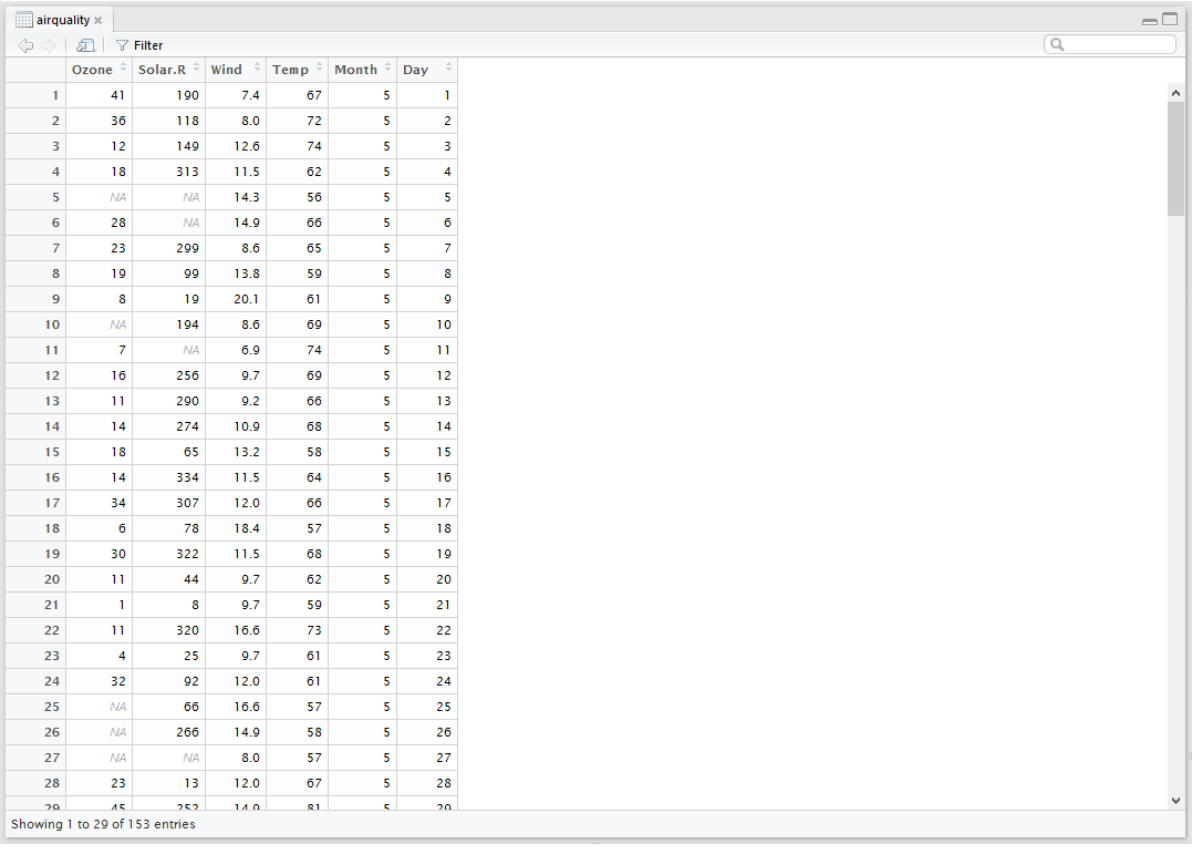
Including just the name of the data table in a command will print the first and last five lines

```
> data(airquality) # load the airquality data
> air.dt <- as.data.table(airquality)
> air.dt
     Ozone Solar.R Wind Temp Month Day
  1:    41     190  7.4   67     5   1
  2:    36     118  8.0   72     5   2
  3:    12     149 12.6   74     5   3
  4:    18     313 11.5   62     5   4
  5:    NA      NA 14.3   56     5   5
 ---
149:    30     193  6.9   70     9  26
150:    NA     145 13.2   77     9  27
151:    14     191 14.3   75     9  28
152:    18     131  8.0   76     9  29
153:    20     223 11.5   68     9  30
```

Viewing a data table

R has a built-in interface for viewing whole data sets

```
> View(air.dt)
```



airquality

Filter

	Ozone	Solar.R	Wind	Temp	Month	Day
1	41	190	7.4	67	5	1
2	36	118	8.0	72	5	2
3	12	149	12.6	74	5	3
4	18	313	11.5	62	5	4
5	NA	NA	14.3	56	5	5
6	28	NA	14.9	66	5	6
7	23	299	8.6	65	5	7
8	19	99	13.8	59	5	8
9	8	19	20.1	61	5	9
10	NA	194	8.6	69	5	10
11	7	NA	6.9	74	5	11
12	16	256	9.7	69	5	12
13	11	290	9.2	66	5	13
14	14	274	10.9	68	5	14
15	18	65	13.2	58	5	15
16	14	334	11.5	64	5	16
17	34	307	12.0	66	5	17
18	6	78	18.4	57	5	18
19	30	322	11.5	68	5	19
20	11	44	9.7	62	5	20
21	1	8	9.7	59	5	21
22	11	320	16.6	73	5	22
23	4	25	9.7	61	5	23
24	32	92	12.0	61	5	24
25	NA	66	16.6	57	5	25
26	NA	266	14.9	58	5	26
27	NA	NA	8.0	57	5	27
28	23	13	12.0	67	5	28
29	45	252	14.9	81	5	29

Showing 1 to 29 of 153 entries

Viewing a data table

The `summary()` command is useful for summarizing a data table's contents:

```
> summary(air.dt)
     Ozone           Solar.R           Wind      
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700  
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400  
 Median : 31.50   Median :205.0   Median : 9.700  
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958  
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500  
 Max.   :168.00   Max.   :334.0   Max.   :20.700  
 NA's   :37       NA's   :7                       
      Temp           Month            Day      
 Min.   :56.00   Min.   :5.000   Min.   : 1.0  
 1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0  
 Median :79.00   Median :7.000   Median :16.0  
 Mean   :77.88   Mean   :6.993   Mean   :15.8  
 3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0  
 Max.   :97.00   Max.   :9.000   Max.   :31.0  
```

Viewing a data table

`nrow()`, `ncol()`, and `dim()` are useful for figuring out a data table's size:

```
> nrow(air.dt)
[1] 153
> ncol(air.dt)
[1] 6
> dim(air.dt)
[1] 153   6
```

Viewing a data table

`nrow()`, `ncol()`, and `dim()` are useful for figuring out a data table's size:

```
> nrow(air.dt)
[1] 153
> ncol(air.dt)
[1] 6
> dim(air.dt)
[1] 153   6
```

and you can get a list of columns using `names()`:

```
> names(air.dt)
[1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"
[6] "Day"
```

Subsetting rows

Rows of a data table can be subset using a vector of integers or booleans:

```
> dt[1:2,]
   iso3    pop admin1 spanish
1:  CAN  35.16     13   FALSE
2:  USA 318.90     51   FALSE
> dt[pop > 200] # The comma is not required
   iso3   pop admin1 spanish
1:  USA 318.9     51   FALSE
> dt[grepl("A", iso3)]
   iso3    pop admin1 spanish
1:  CAN  35.16     13   FALSE
2:  USA 318.90     51   FALSE
```

Note that the column names are treated as objects while within the bracket notation of a data table

Removing rows

You can remove rows entirely by selecting just the ones you want to keep and then assigning or reassigning to an object:

```
> subset.dt <- dt[pop < 150,]
> subset.dt
   iso3    pop admin1 spanish
1:  CAN  35.16     13   FALSE
2:  MEX 122.30     31    TRUE
```

```
> air.dt.nomiss <- air.dt[!is.na(Ozone),]
> nrow(air.dt)
[1] 153
> nrow(air.dt.nomiss)
[1] 116
```

Selecting columns

A single column can be selected *by number* or *by name*. These methods return a data table…

```
> dt[, 2]
      pop
1:  35.16
2: 318.90
3: 122.30
> dt[, "pop"]
      pop
1:  35.16
2: 318.90
3: 122.30
```

Selecting columns

while these return a vector:

```
> dt[, pop]
[1]  35.16 318.90 122.30
> dt$pop # data frame syntax
[1]  35.16 318.90 122.30
```

Selecting columns

Multiple columns from a data table can be selected at once:

```
> dt[, c("iso3", "pop")]
   iso3    pop
1:  CAN  35.16
2:  USA 318.90
3:  MEX 122.30
> dt[, .(iso3, pop)]
   iso3    pop
1:  CAN  35.16
2:  USA 318.90
3:  MEX 122.30
```

Manipulating columns

Because a data table column is a vector, all of the operations that are valid for vectors are also valid for data table columns:

```
> mean(dt$pop)
[1] 158.7867
> log(dt$pop)
[1] 3.559909 5.764878 4.806477
> dt$pop / dt$admin1
[1] 2.704615 6.252941 3.945161
> dt$iso3 == "USA"
[1] FALSE  TRUE FALSE
```

Manipulating columns

You can create a new column or update an existing one using the `:=` operator:

```
> dt[, gdp := c(1827, 16770, 1261)]
> dt[, pop := pop * 1e6]
> dt
   iso3       pop admin1 spanish   gdp
1:  CAN  35160000     13   FALSE  1827
2:  USA 318900000     51   FALSE 16770
3:  MEX 122300000     31    TRUE  1261
```

Manipulating columns

Similarly, you can create new columns that are functions of existing columns:

```
> dt[, log_gdp := log(gdp)]
> dt[, gdp_pc := (1e9 * gdp) / (pop)]
> str(dt)
Classes 'data.table' and 'data.frame':  3 obs. of  7 variables
 $ iso3   : chr  "CAN" "USA" "MEX"
 $ pop    : num  3.52e+07 3.19e+08 1.22e+08
 $ admin1 : int  13 51 31
 $ spanish: logi  FALSE FALSE TRUE
 $ gdp    : num  1827 16770 1261
 $ log_gdp: num  7.51 9.73 7.14
 $ gdp_pc : num  51962 52587 10311
 - attr(*, ".internal.selfref")=<externalptr>
```

Removing columns

You can remove one or multiple columns from a data frame by assigning it to `NULL`:

```
> names(dt)
[1] "iso3"    "pop"     "admin1"  "spanish" "gdp"
[6] "log_gdp" "gdp_pc"
> dt[, log_gdp := NULL]
> names(dt)
[1] "iso3"    "pop"     "admin1"  "spanish" "gdp"
[6] "gdp_pc"
> dt[, c("admin1", "spanish") := NULL]
> names(dt)
[1] "iso3"   "pop"    "gdp"    "gdp_pc"
```

Removing columns

Alternatively, you can subset to just the columns you want to keep and then reassign the object:

```
> air.dt <- air.dt[, c("Ozone", "Wind", "Month", "Day")]
```

```
> air.dt
     Ozone Wind Month Day
  1:    41  7.4     5   1
  2:    36  8.0     5   2
  3:    12 12.6     5   3
  4:    18 11.5     5   4
  5:    NA 14.3     5   5
 ---
149:    30  6.9     9  26
150:    NA 13.2     9  27
151:    14 14.3     9  28
152:    18  8.0     9  29
153:    20 11.5     9  30
```

Things to keep in mind...

Both rows and columns can be selected using a `dt[i, j]` notation.

In this framework, everything *before* the comma tells R what row(s) you want...

```
> dt[pop > 1e8,]
   iso3       pop   gdp   gdp_pc
1:  USA 318900000 16770 52587.02
2:  MEX 122300000  1261 10310.71
```

and everything *after* the comma tells R what column(s) you want:

```
> dt[, "iso3"]
   iso3
1:  CAN
2:  USA
3:  MEX
```

Things to keep in mind...

You can also select rows and columns at the same time using `dt[i, j]` notation:

```
> dt[pop > 1e8, .(iso3, gdp_pc)]
   iso3   gdp_pc
1:  USA 52587.02
2:  MEX 10310.71
```

Things to keep in mind...

If you want to store the output of these selections you must assign them to an object:

```
> big.pop <- dt[pop > 1e8, iso3]
> big.pop
[1] "USA" "MEX"
```

```
> air.dt.may <- air.dt[Month == 5,]
> head(air.dt.may)
   Ozone Wind Month Day
1:    41  7.4     5   1
2:    36  8.0     5   2
3:    12 12.6     5   3
4:    18 11.5     5   4
5:    NA 14.3     5   5
6:    28 14.9     5   6
```

Things to keep in mind...

Note that the object you store your selection in might be either a vector or a data.frame, depending on the nature of your selection.

```
> class(big.pop)
[1] "character"
> class(air.dt.may)
[1] "data.table" "data.frame"
```

THINGS TO KEEP IN MIND...

(Re)assignment can also be used to store the output of your selections using the original object name:

```
> nrow(air.dt)
[1] 153
> range(air.dt$Month)
[1] 5 9
```

```
> air.dt <- air.dt[Month == 5,]
```

```
> nrow(air.dt)
[1] 31
> range(air.dt$Month)
[1] 5 5
```