Descriptive Statistics And tabulation

Pavan Kumar A
Senior Project Engineer
Big Data Analytics Team
CDAC-KP

INTRODUCTION

- Important elements in data analysis are understanding Summary and Descriptive Statistics
- This helps in correct understanding of data.
- There are three main ways to describe or summarize the data
  1. Summary Statistics
  2. Tabulation
  3. Graphical

Summary Commands

- Summary commands are used to get the overview of the data we are dealing with.
- To know the datasets available (in-built) in the base R, use `data()` command

```
Data sets in package 'datasets':

AirPassengers           Monthly Airline Passenger Numbers 1949-1960
BJsales                 Sales Data with Leading Indicator
BJsales.lead (BJsales)
                        Sales Data with Leading Indicator
BOD                     Biochemical Oxygen Demand
CO2                     Carbon Dioxide Uptake in Grass Plants
ChickWeight             Weight versus age of chicks on different diets
DNase                   Elisa assay of DNase
EuStockMarkets          Daily Closing Prices of Major European Stock
                        Indices, 1991-1998
Formaldehyde            Determination of Formaldehyde
```

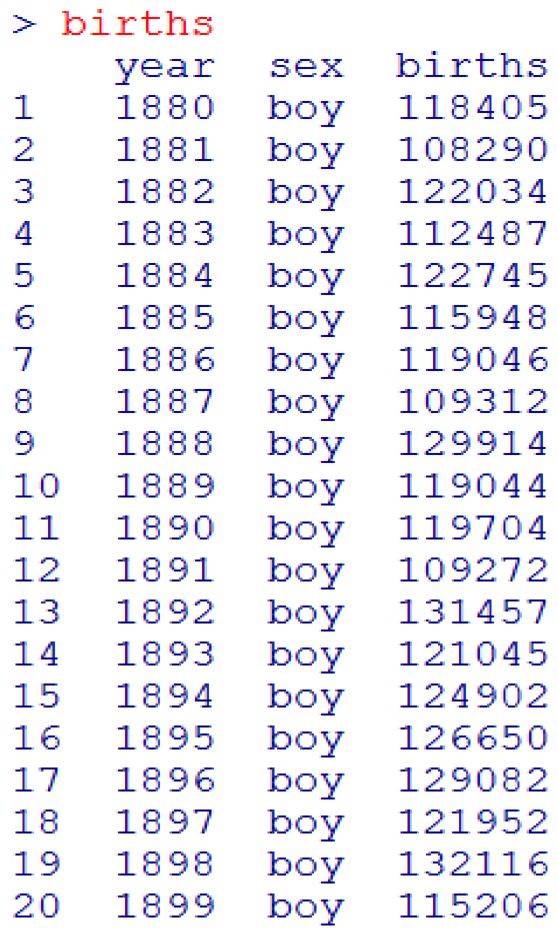
Summary Commands

- Here, we are seeing the contents of 'births' object

  It shows three columns ("year", "sex", births) and 20 rows

  Some data contains hundreds of rows and columns. In such cases, going though whole data on R console is difficult.

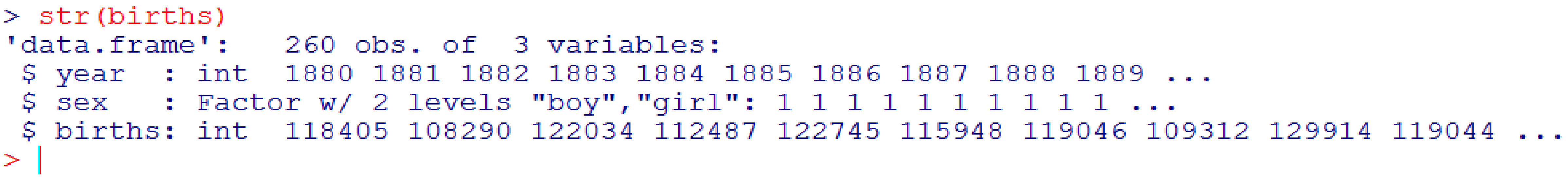
  To view overall structure of any data str() command is used



```
> births
     year  sex  births
1    1880  boy  118405
2    1881  boy  108290
3    1882  boy  122034
4    1883  boy  112487
5    1884  boy  122745
6    1885  boy  115948
7    1886  boy  119046
8    1887  boy  109312
9    1888  boy  129914
10   1889  boy  119044
11   1890  boy  119704
12   1891  boy  109272
13   1892  boy  131457
14   1893  boy  121045
15   1894  boy  124902
16   1895  boy  126650
17   1896  boy  129082
18   1897  boy  121952
19   1898  boy  132116
20   1899  boy  115206
```

Summary Commands – str() Command

- The str() command:



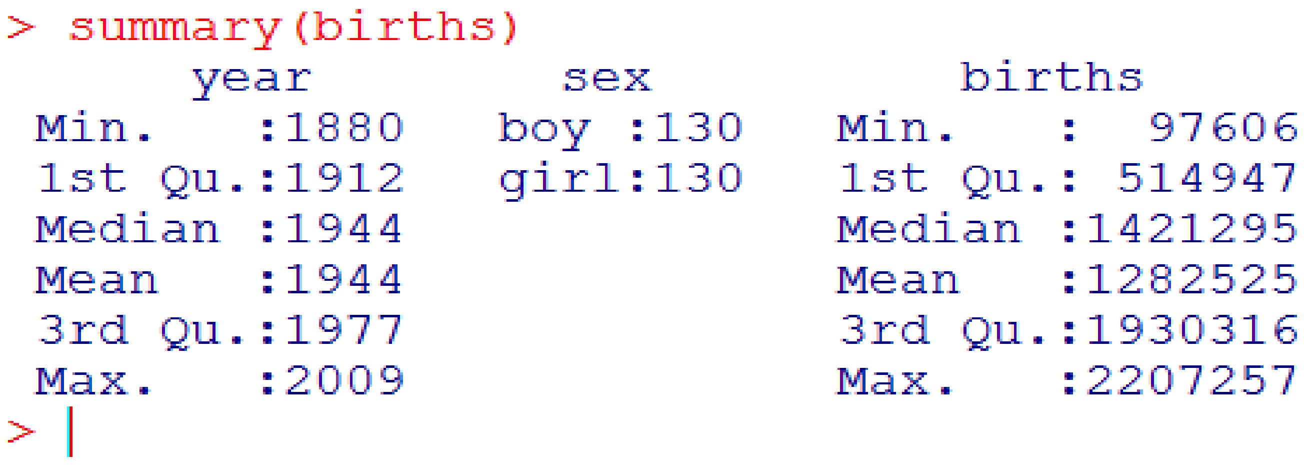
```
> str(births)
'data.frame':   260 obs. of  3 variables:
 $ year  : int  1880 1881 1882 1883 1884 1885 1886 1887 1888 1889 ...
 $ sex   : Factor w/ 2 levels "boy","girl": 1 1 1 1 1 1 1 1 1 1 ...
 $ births: int  118405 108290 122034 112487 122745 115948 119046 109312 129914 119044 ...
> |
```

- First line of the output tells that: It is a data frame with 260 observations (rows) of 3 variables (columns)
- Following three lines are names of the columns and their type.
  - $year is a type of int (Integer)
  - $sex is a type of Factor and 2 levels (Boy/Girl)
  - $births is of type int (Integer)

Summary Commands – summary() Command

- The `str()` command is used to get only structure of data object.
- The `summary()` command is used to get the summary of data object.
- Following is the summary statistics of “births” data object.

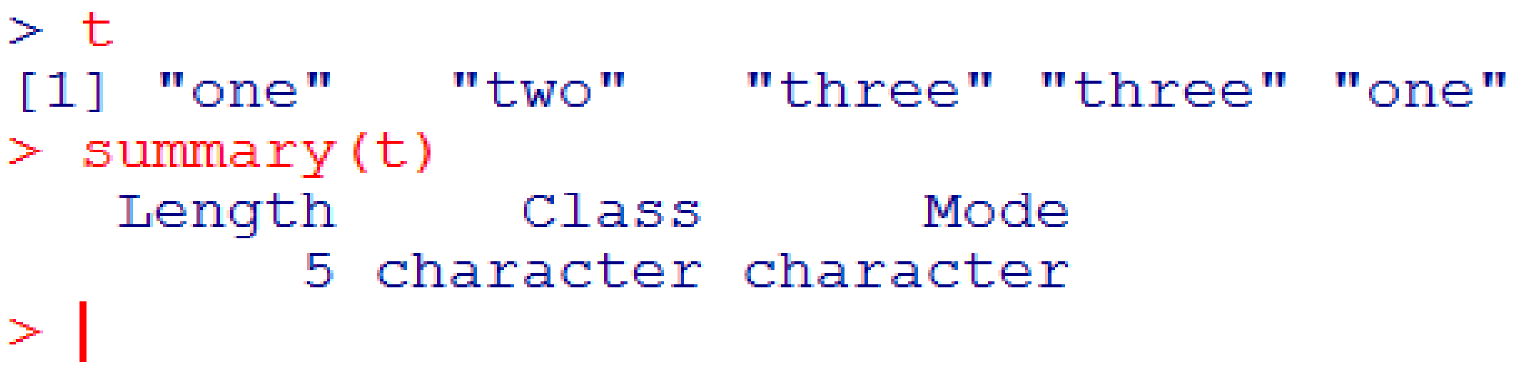
1. It describes the simple statistics of three columns “year”, “sex” and “births” like **Minimum, Median, Mean and Max values**.
2. In addition to this, it also gives 1st Quartile and 3rd Quartile



```
> summary(births)
      year           sex           births
 Min.   :1880   boy :130   Min.   :  97606
 1st Qu.:1912   girl:130   1st Qu.: 514947
 Median :1944              Median :1421295
 Mean   :1944              Mean   :1282525
 3rd Qu.:1977              3rd Qu.:1930316
 Max.   :2009              Max.   :2207257
> |
```

Summary Commands – summary() Command

- Applying summary() command on the character data object.



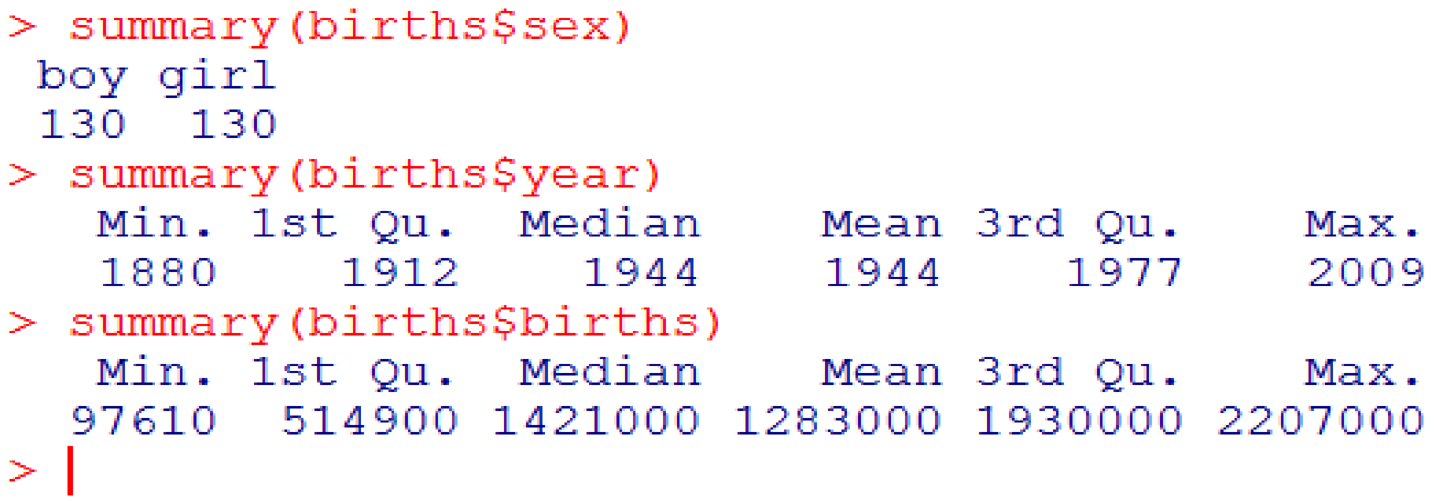
```
> t
[1] "one"   "two"   "three" "three" "one"
> summary(t)
   Length     Class      Mode
        5 character character
> 
```

- Here, data object “t” contain character items that are in quotes, they are treated as standard characters rather than factors.
- It shows, length of the data object and type of the values in the data object.

Summary Commands – summary() Command

- The summary() function can be applied separately on each column of the data frame.
- For example: In the data frame “births”, there are three columns “year”, “sex”, “births”. Applying summary on each of these columns are shown

Column sex is of factorial data type and year and births columns are of integer data type and their corresponding statistics are given



```
> summary(births$sex)
 boy girl 
 130  130 
> summary(births$year)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   1880    1912    1944    1944    1977    2009 
> summary(births$births)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  97610  514900 1421000 1283000 1930000 2207000 
> 
```

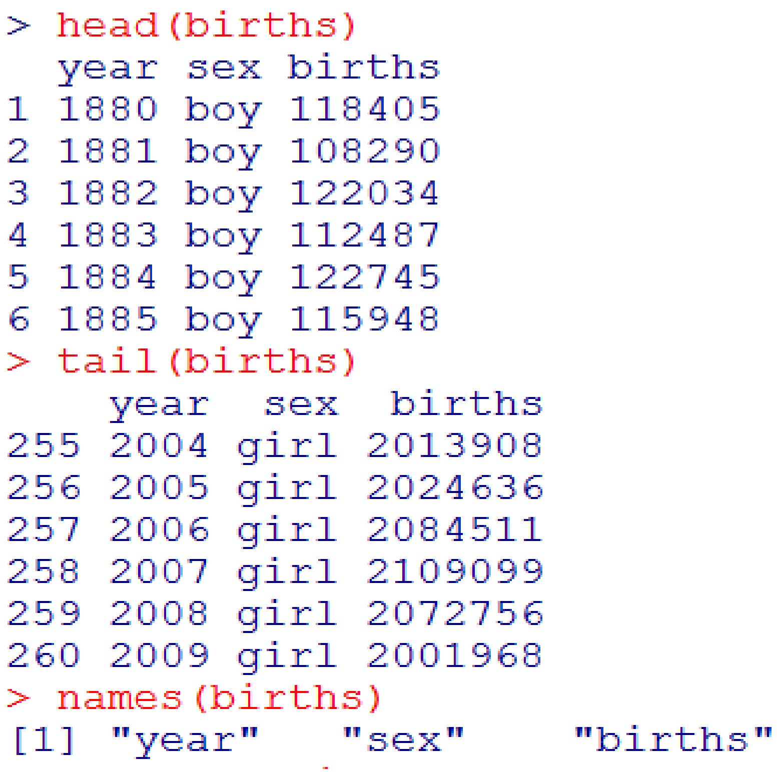
Summary Commands

- In the previous example, $ (Dollar Sign) is used to extract an item from the data frame.
- births$year : Gives the "year" column from the data frame called "births".
- **Few more commands:**

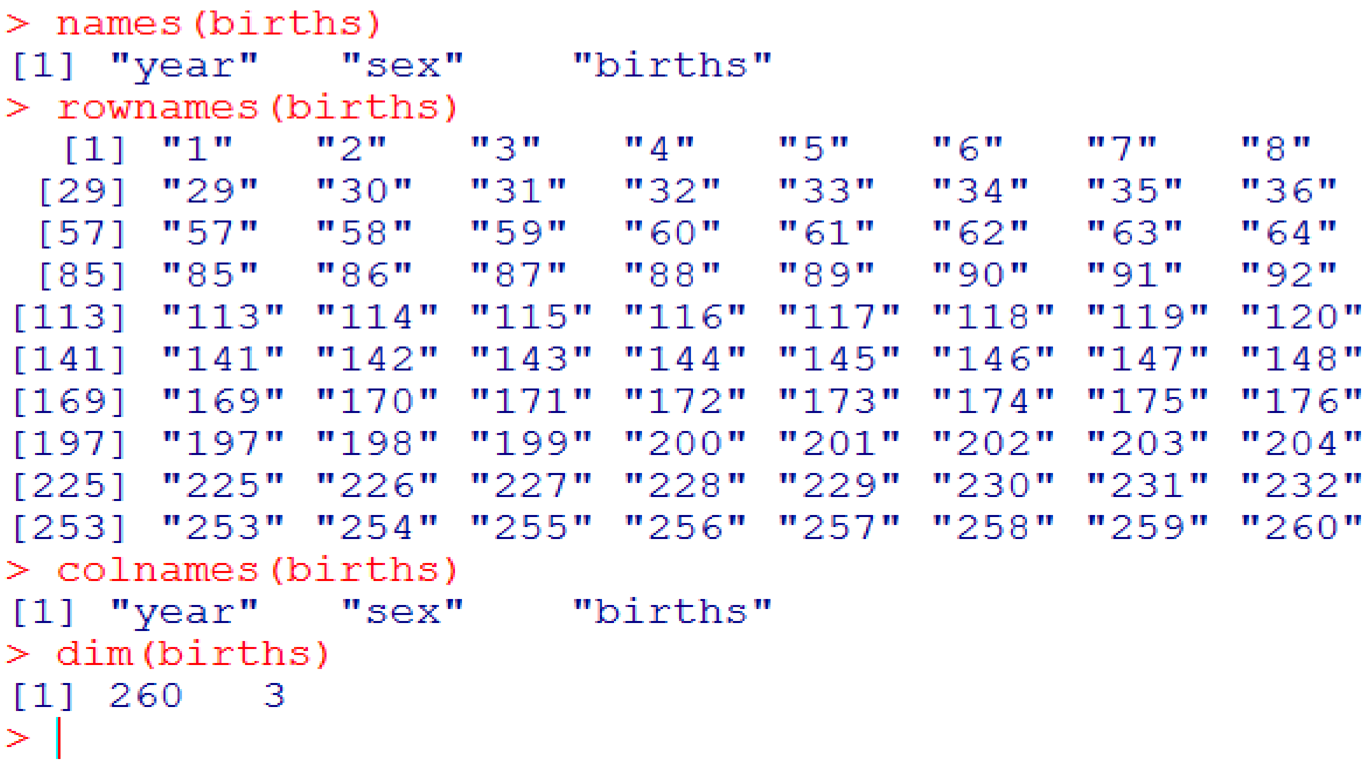
Command	Explanation
names()	Works on list and data frames. Gets the names of the columns of the data frame
rownames()	Gets the row names of the data frame or matrix
colnames()	Gets the column names of the data frame or matrix
dimnames()	Gets the row and column names for matrix or data frame objects
head()	Gives the first few lines of the data frame
tail()	Gives the last few lines of the data frame

Summary Commands

- Examples of summary commands:



```
> head(births)
  year sex births
1 1880 boy 118405
2 1881 boy 108290
3 1882 boy 122034
4 1883 boy 112487
5 1884 boy 122745
6 1885 boy 115948
> tail(births)
    year  sex  births
255 2004 girl 2013908
256 2005 girl 2024636
257 2006 girl 2084511
258 2007 girl 2109099
259 2008 girl 2072756
260 2009 girl 2001968
> names(births)
[1] "year"   "sex"    "births"
```



```
> names(births)
[1] "year"   "sex"    "births"
> rownames(births)
  [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"
 [29] "29"  "30"  "31"  "32"  "33"  "34"  "35"  "36"
 [57] "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"
 [85] "85"  "86"  "87"  "88"  "89"  "90"  "91"  "92"
[113] "113" "114" "115" "116" "117" "118" "119" "120"
[141] "141" "142" "143" "144" "145" "146" "147" "148"
[169] "169" "170" "171" "172" "173" "174" "175" "176"
[197] "197" "198" "199" "200" "201" "202" "203" "204"
[225] "225" "226" "227" "228" "229" "230" "231" "232"
[253] "253" "254" "255" "256" "257" "258" "259" "260"
> colnames(births)
[1] "year"   "sex"    "births"
> dim(births)
[1] 260   3
> |
```

Summary Statistics For Vectors

- Simplest data object is vector (Single dimensional representation of values).
- There are variety of summary statistics can be applied on vector of numbers.
- Two kinds of summary commands that can be applied
  1. Commands that produce single values as a result
  2. Commands that produces multiple values as a result

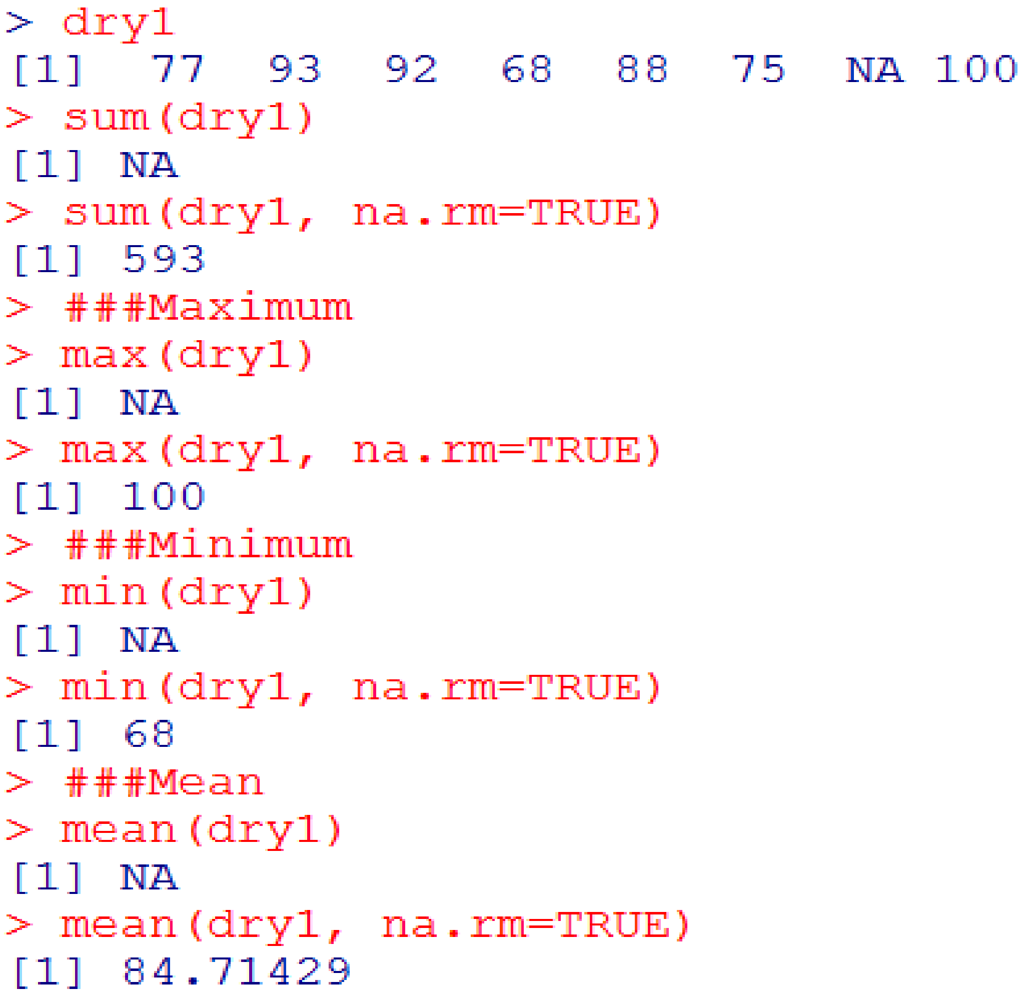
Summary Commands – Single value Result

- Some of the summary commands which produce single value result

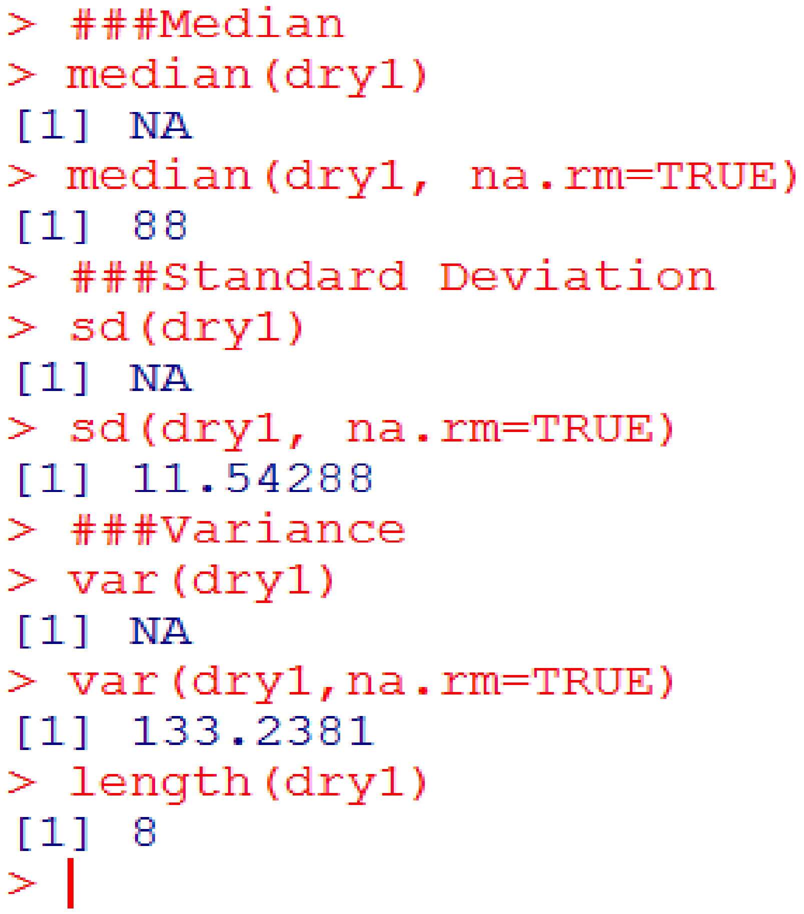
Command	Explanation
max(x, na.rm=TRUE)	Shows the maximum value in the numeric vector. To remove null values `na.rm=TRUE` is used
min(x, na.rm=TRUE)	Shows the minimum value in the numeric vector. To remove null values `na.rm=TRUE` is used
length(x)	Gives the length of the vector
sum(x, na.rm=TRUE)	Summation of the vector after removing null values, if any
mean(x, na.rm=TRUE)	Gives the mean value of the vector after removing null values, if any.
median(x, na.rm=TRUE)	Gives the median value of the vector after removing null values, if any.
sd(x, na.rm=TRUE)	Shows the Standard Deviation of the vector
var(x, na.rm=TRUE)	Shows the Variance of the vector

Summary Commands – Single value Result

- Applying single value result commands on the vector called “dry1”.
- The dry1 vector is having null value.



```
> dry1
[1]  77  93  92  68  88  75  NA 100
> sum(dry1)
[1] NA
> sum(dry1, na.rm=TRUE)
[1] 593
> ###Maximum
> max(dry1)
[1] NA
> max(dry1, na.rm=TRUE)
[1] 100
> ###Minimum
> min(dry1)
[1] NA
> min(dry1, na.rm=TRUE)
[1] 68
> ###Mean
> mean(dry1)
[1] NA
> mean(dry1, na.rm=TRUE)
[1] 84.71429
```



```
> ###Median
> median(dry1)
[1] NA
> median(dry1, na.rm=TRUE)
[1] 88
> ###Standard Deviation
> sd(dry1)
[1] NA
> sd(dry1, na.rm=TRUE)
[1] 11.54288
> ###Variance
> var(dry1)
[1] NA
> var(dry1,na.rm=TRUE)
[1] 133.2381
> length(dry1)
[1] 8
>
```

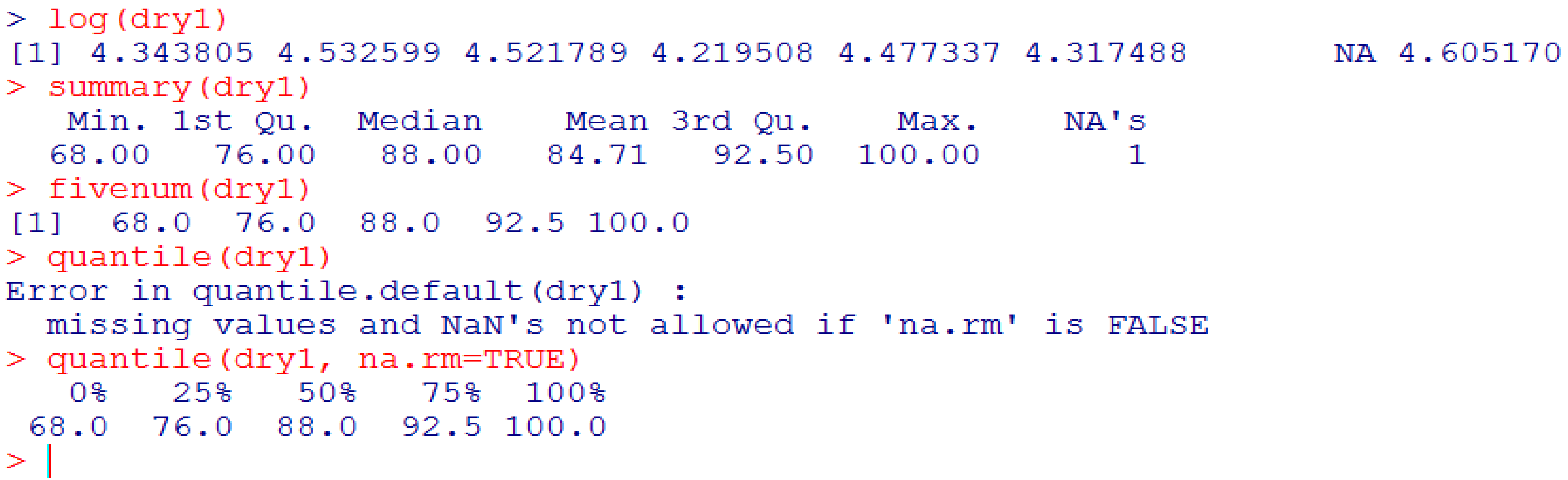
Summary Commands – Multiple value Results

- Some of the Summary commands that produce multiple value results are given

Command	Explanation
log()	Gives the logarithmic values of all entries in the vector
summary()	Gives the summary of the data frames or matrix
quantile()	Gives sample quantiles corresponding to the given probabilities
fivenum()	Gives five number summary for the input data

Summary Commands – Multiple value Results

- Applying multiple value result commands on the vector called “dry1”.
- The dry1 vector is having null value.



```
> log(dry1)
[1] 4.343805 4.532599 4.521789 4.219508 4.477337 4.317488       NA 4.605170
> summary(dry1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
  68.00   76.00   88.00   84.71   92.50  100.00       1 
> fivenum(dry1)
[1]  68.0  76.0  88.0  92.5 100.0
> quantile(dry1)
Error in quantile.default(dry1) : 
  missing values and NaN's not allowed if 'na.rm' is FALSE
> quantile(dry1, na.rm=TRUE)
   0%   25%   50%   75%  100% 
 68.0  76.0  88.0  92.5 100.0 
> 
```

Cumulative Statistics

- Cumulative statistics are those that are applied sequentially to a series of values.

- Two kinds of Cumulative statistics

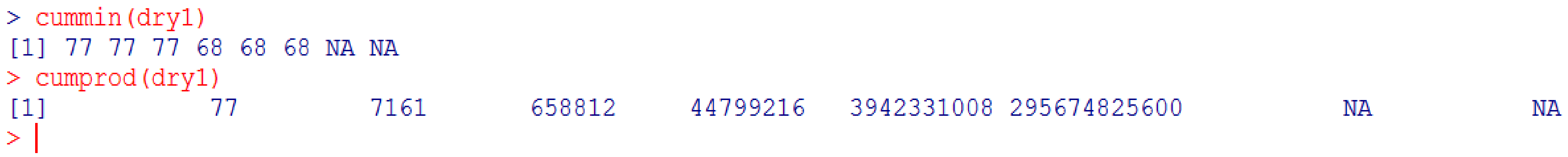
1. Simple Cumulative Statistics
2. Complex Cumulative Statistics

- For simple commands, require only the name of the data
- For complex commands, we have to create more complicated instructions to produce the results.

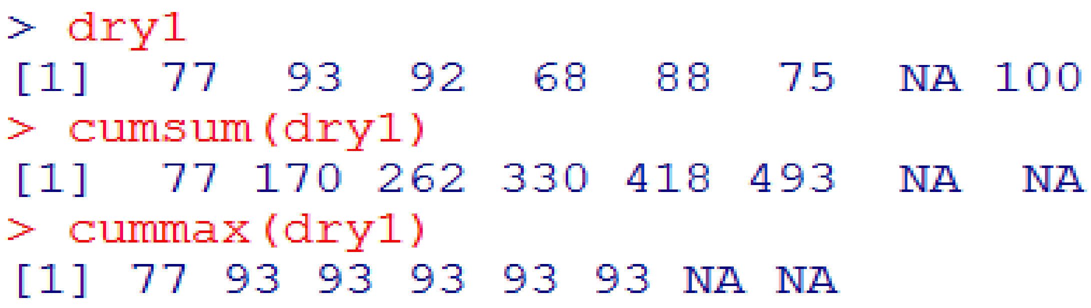
Cumulative Statistics – Simple Commands

- Simple cumulative statistics commands are shown as follows

Command	Explanation
cumsum(x)	The cumulative sum of vector
cummax(x)	The cumulative maximum value
cummin(x)	The cumulative minimum value
cumprd(x)	The cumulative product value



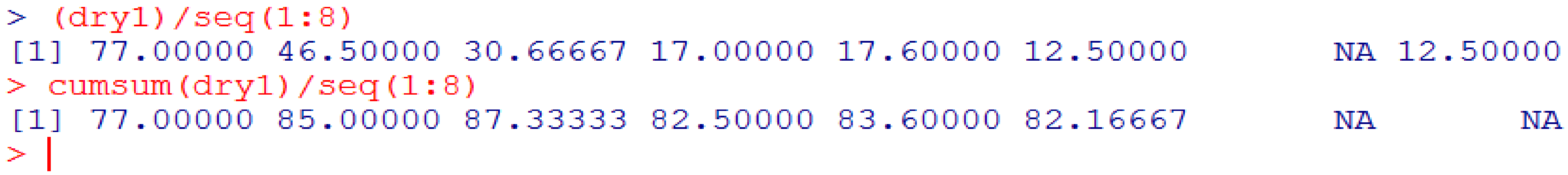
```
> cummin(dry1)
[1] 77 77 77 68 68 68 NA NA
> cumprod(dry1)
[1]           77         7161       658812     44799216   3942331008 295674825600           NA           NA
> |
```



```
> dry1
[1]  77  93  92  68  88  75  NA 100
> cumsum(dry1)
[1]  77 170 262 330 418 493  NA  NA
> cummax(dry1)
[1] 77 93 93 93 93 93 NA NA
```

Cumulative Statistics – Complex Commands

- Cumulative statistics command can be used in combination with other commands to produce additional use measures.
- For example
  - Data1 object divided by series of numbers



```
> (dry1)/seq(1:8)
[1] 77.00000 46.50000 30.66667 17.00000 17.60000 12.50000       NA 12.50000
> cumsum(dry1)/seq(1:8)
[1] 77.00000 85.00000 87.33333 82.50000 83.60000 82.16667       NA       NA
> |
```

Summary Tables - Introduction

- The table() command is used to produce table objects.
- The table() command is also used to create a few special kinds of table objects, including contingency tables.

- Contingency Table?
  - It is a type of table in a matrix format that displays the frequency distribution of the variables.
  - They provide the interrelation between 2 variables.

	Right-handed	Left-handed	Total
Males	43	9	52
Females	44	4	48
Total	87	13	100

Making Contingency Tables

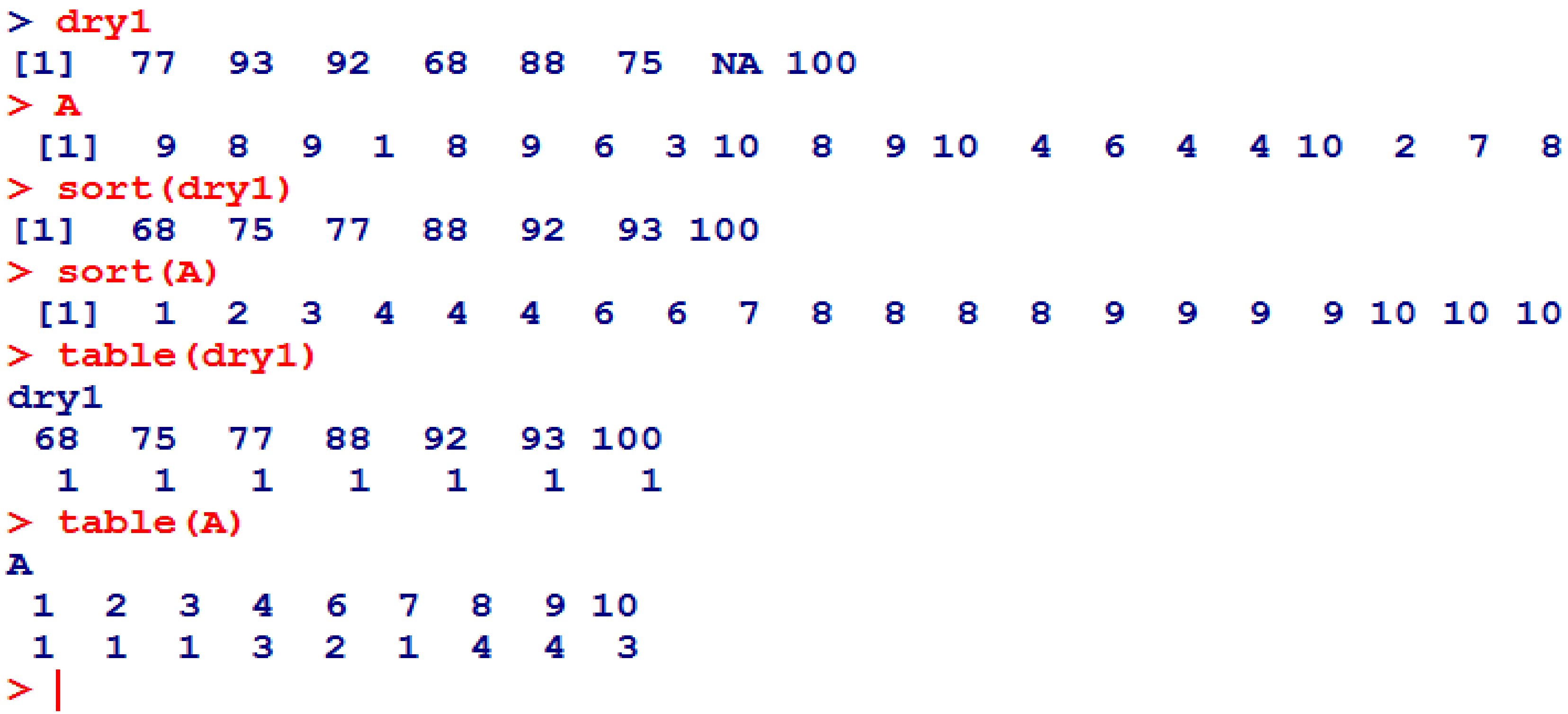
- **Definition :** A contingency table is a way to redraw data and assemble it into a table that shows the layout of the original data in a manner that allows the reader to gain overall summary of the original data.
- The table() command is used to create the table objects.

- **Creating Contingency Tables from vectors**
- The simplest data object from which you can create a contingency table is vector
- **Syntax is as follows**

```
table(x)          ## Where x is integer vector
```

Making Contingency Tables – Integer Vector

- Executing table() command on the integer vectors

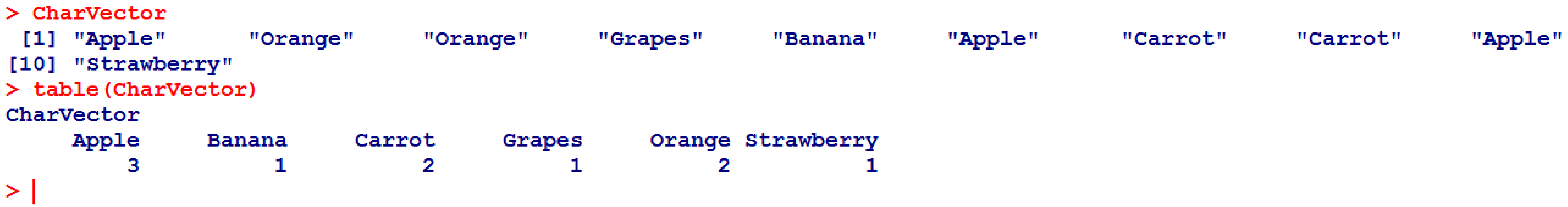


```
> dry1
[1]  77  93  92  68  88  75  NA 100
> A
 [1]  9  8  9  1  8  9  6  3 10  8  9 10  4  6  4  4 10  2  7  8
> sort(dry1)
[1]  68  75  77  88  92  93 100
> sort(A)
 [1]  1  2  3  4  4  4  6  6  7  8  8  8  8  9  9  9  9 10 10 10
> table(dry1)
dry1
 68  75  77  88  92  93 100
  1   1   1   1   1   1   1
> table(A)
A
 1  2  3  4  6  7  8  9 10
 1  1  1  3  2  1  4  4  3
> 
```

- Here, we have applied on 2 vectors (“dry1” and “A”)
- Output shows the frequency of the values in the vector

Making Contingency Tables – Character Vector

- The table() command can also used on character data too.
- Example is shown below



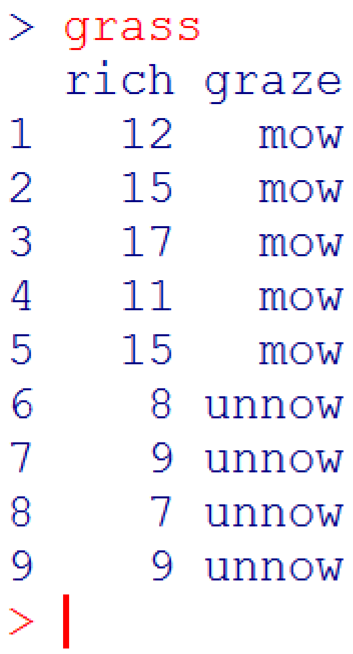
```
> CharVector
 [1] "Apple"      "Orange"     "Orange"     "Grapes"     "Banana"     "Apple"      "Carrot"     "Carrot"     "Apple"
[10] "Strawberry"
> table(CharVector)
CharVector
     Apple     Banana     Carrot     Grapes     Orange Strawberry
         3          1          2          1          2          1
> |
```

- Here, charVector is data object with character values.
- We are applying table() command on charVector to create contingency table.
- It shows, Apple appeared 3 times, Banana 1s, Carrot 2s…

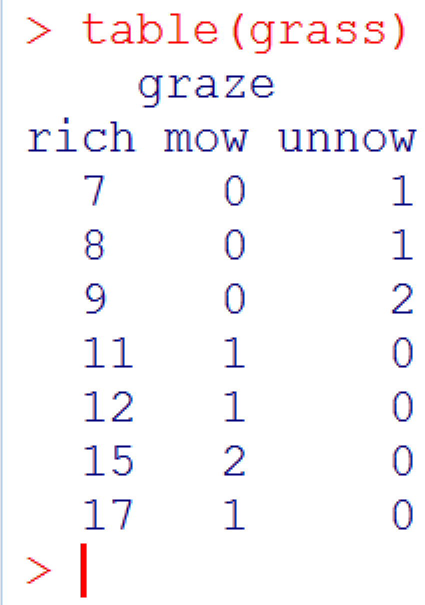
Creating Contingency Tables From Complicated Data

- Applying a `table()` command on the data frames.
- Let us apply on “grass” data frame which has 2 columns (“rich” and graze)

1. You can see the numerical data in the first column, followed by a column for each of `graze` treatments
2. The table shows- how many times a particular numerical value cropped up in each of the `graze` treatments



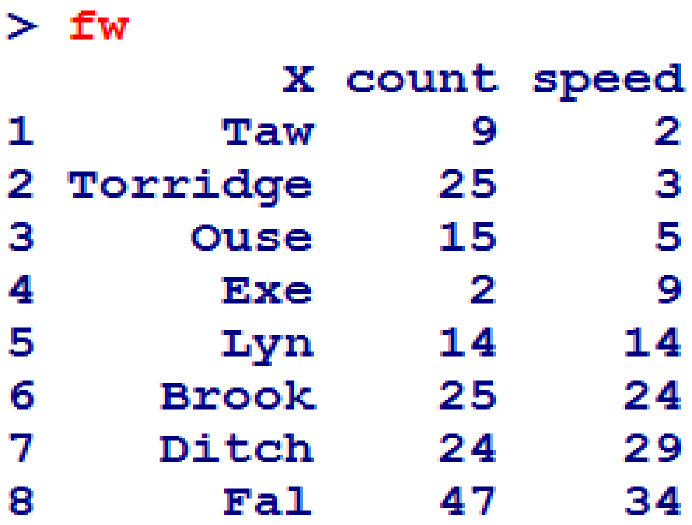
```
> grass
  rich graze
1   12   mow
2   15   mow
3   17   mow
4   11   mow
5   15   mow
6    8 unmow
7    9 unmow
8    7 unmow
9    9 unmow
> 
```



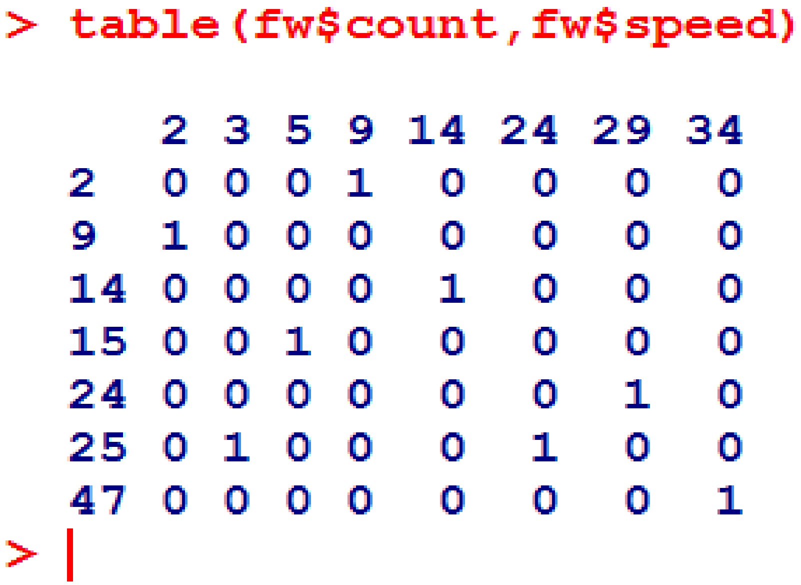
```
> table(grass)
    graze
rich mow unmow
  7    0     1
  8    0     1
  9    0     2
  11   1     0
  12   1     0
  15   2     0
  17   1     0
> 
```

Creating Contingency Tables From Complicated Data

- If the data frame is more columns unlike previous example, the contingency table will be more complex.
- Applying table() command on fw data frame.
- Data Frame “fw” contains 3 columns, out of which first column is Characters values and other 2 columns are integer values



```
> fw
         X count speed
1      Taw     9     2
2 Torridge    25     3
3     Ouse    15     5
4      Exe     2     9
5      Lyn    14    14
6    Brook    25    24
7    Ditch    24    29
8      Fal    47    34
```



```
> table(fw$count,fw$speed)

     2  3  5  9 14 24 29 34
  2  0  0  0  1  0  0  0  0
  9  1  0  0  0  0  0  0  0
  14 0  0  0  0  1  0  0  0
  15 0  0  1  0  0  0  0  0
  24 0  0  0  0  0  0  1  0
  25 0  1  0  0  0  1  0  0
  47 0  0  0  0  0  0  0  1
> |
```

THANK YOU !!!