

Data

R for Data Science
Basel R Bootcamp



February 2019

Data

In this session you will get to know...

- R's 3 main **data types**
- a little more about **functions**
- R's **Import/Export** functions



from cloudtweaks.com

3 Object types for data

R has 3 main data objects...

list - R's multi-purpose container

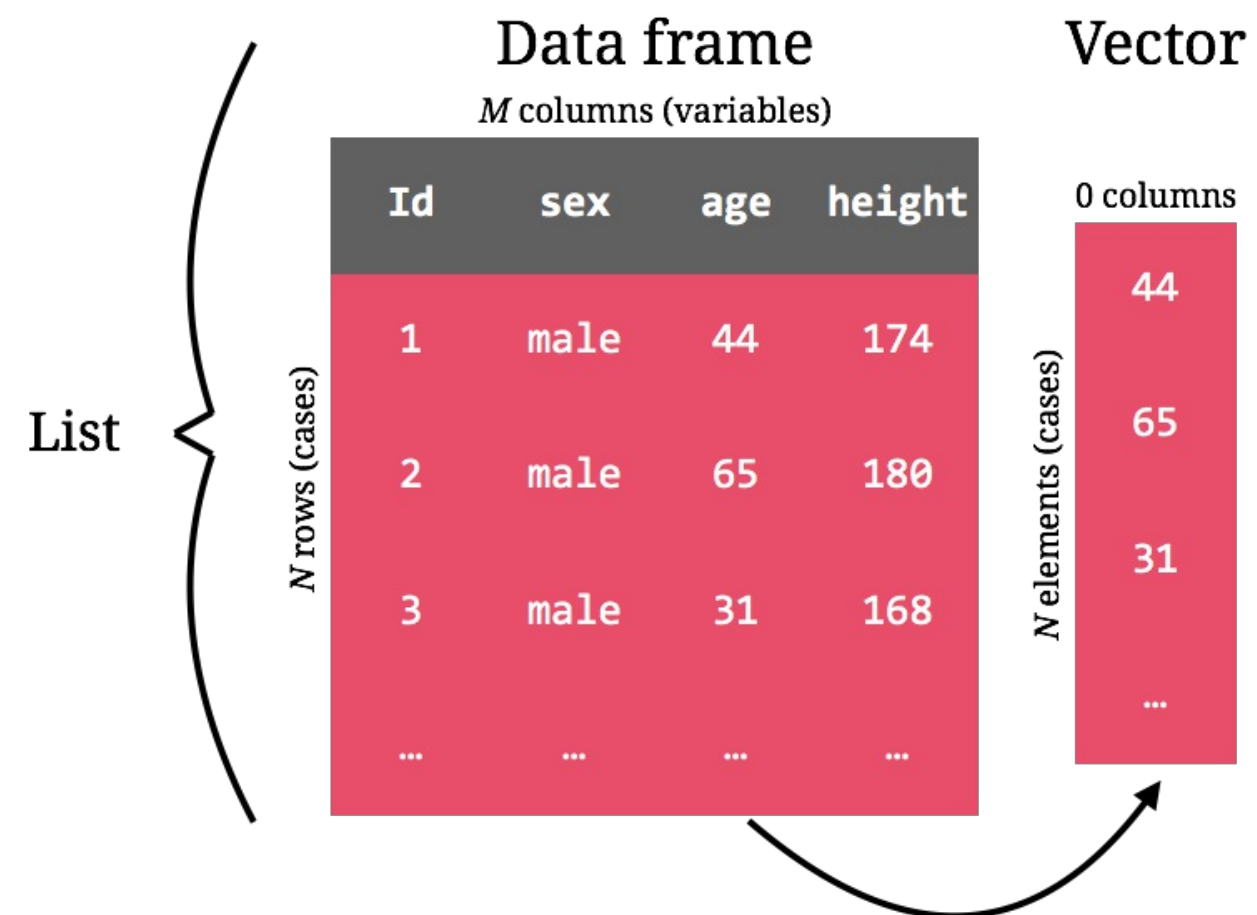
- Can carry any data, incl. lists
- Often used for function outputs

data_frame - R's spreadsheet

- Specific type of list
- Typical data format
- For multi-variable data sets

vectors - R's data container

- Actually carries the data
- Contain data of 1 of many types



list

- 1 - Can **carry any data**, incl. lists, data_frames, vectors, etc.
- 2 - Are often used for **function outputs**
- 3 - Have **named elements**.
- 4 - Elements can be **inspected** via `names()` or `str()`.
- 5 - Elements are (typically) **selected** by `$`.

List

N **named** Elements (objects)

coefficients			df	resid- uals	r square
var	est.	T	99	1.74 1.42 8.21 4.24 0.45 43.1 2.1 ...	0.37
x1	1.17	1.86	99		
x2	3.32	2.65			

List: Select element using \$

```
# regression
reg_model <- lm(height ~ sex + age,
               data = baselers)
reg_results <- summary(reg_model)

# get element names
names(reg_results)
```

```
## [1] "call"      "terms"
## [3] "residuals" "coefficients"
## [5] "aliased"   "sigma"
## [7] "df"        "r.squared"
```

```
# select element using $
reg_results$coefficients
```

```
##           Estimate t value
## (Intercept) 164.171266 499.5339
## sexmale      13.993699  66.4724
## age          -0.003753  -0.5819
```

List

N named Elements (objects)

coefficients			df	residuals	r square
var	est.	T	99	1.74	0.37
				1.42	
			99	8.21	
x1	1.17	1.86		4.24	
				0.45	
				43.1	
x2	3.32	2.65		2.1	
				...	

data_frame

1 - Are lists containing **vectors of equal length** representing the variables.

2 - Contain vectors of different types: numeric, character, etc.

3 - Have named elements.

4 - Elements can be **inspected** via `names()`, `str()`, `print()`, `View()`, or `skimr::skim()`.

5 - Elements are (typically) **selected** by `$`.

6 - Come in different flavors: `data.frame()`, `data.table()`, `tibble()`.

Data frame
4 columns (variables)

N rows (cases)	Id	sex	age	height
	1	male	44	174
	2	male	65	180
	3	male	31	168

Inspect content

```
# inspect baselers via print
baselers
```

```
## # A tibble: 10,000 x 20
##       id sex    age height weight income
##   <dbl> <chr> <dbl>  <dbl>  <dbl>  <dbl>
## 1     1 male    44   174.   113.   6300
## 2     2 male    65   180.    75.2  10900
## 3     3 fema...  31   168.    55.5   5100
## 4     4 male    27   209    93.8   4200
## 5     5 male    24   177.    NA    4000
##   education confession children
##   <chr>      <chr>      <dbl>
## 1 SEK_III   catholic      2
## 2 obligato... confessio...    2
## 3 SEK_III   <NA>          2
## 4 SEK_III   catholic      2
## 5 SEK_III   catholic      1
## # ... with 9,995 more rows, and 11 more
## #   variables
```

Data frame

4 columns (variables)

N rows (cases)	Id	sex	age	height
	1	male	44	174
	2	male	65	180
	3	male	31	168

Inspect content

```
# View dataframe in a new window  
View(baselers)
```

	id	sex	age	height	weight	income
1	1	male	44	174.3	113.4	6300
2	2	male	65	180.3	75.2	10900
3	3	female	31	168.3	55.5	5100
4	4	male	27	209.0	93.8	4200
5	5	male	24	176.7	NA	4000
6	6	male	63	186.6	67.4	11400
7	7	male	71	151.6	83.3	12000
8	8	female	41	155.7	67.8	7600
9	9	male	43	176.1	69.3	8500
10	10	female	31	166.1	66.3	6100
11	11	female	42	157.8	51.9	8000

Data frame

4 columns (variables)

N rows (cases)	Id	sex	age	height
	1	male	44	174
	2	male	65	180
	3	male	31	168

Select via \$

```
# Access age column from baselers
baselers$age
```

```
## [1] 44 65 31 27 24 63 71 41 43 31 42 31
## [13] 38 49 39 54 78 62 88 74
```

```
# Access education column from baselers
baselers$education
```

```
## [1] "SEK_III"
## [2] "obligatory_school"
## [3] "SEK_III"
## [4] "SEK_III"
## [5] "SEK_III"
## [6] "SEK_III"
## [7] "SEK_III"
## [8] "SEK_III"
## [9] "apprenticeship"
## [10] "SEK_II"
```

Data frame

4 columns (variables)

N rows (cases)	Id	sex	age	height
	1	male	44	174
	2	male	65	180
	3	male	31	168

Change/Add via \$

```
# Divide income by 1000
baselers$income <- baselers$income / 1000

# inspect baselers
baselers
```

```
## # A tibble: 10,000 x 20
##       id sex    age height weight income
##   <dbl> <chr> <dbl> <dbl> <dbl> <dbl>
## 1     1 male    44   174.  113.    6.3
## 2     2 male    65   180.   75.2  10.9
## 3     3 fema...  31   168.   55.5   5.1
## 4     4 male    27   209   93.8   4.2
## 5     5 male    24   177.   NA     4
##   education confession children
##   <chr>      <chr>      <dbl>
## 1 SEK_III   catholic      2
## 2 obligato... confessio...    2
## 3 SEK_III   <NA>          2
## 4 SEK_III   catholic      2
## 5 SEK_III   catholic      1
## # ... with 9,995 more rows, and 11 more
```

Data frame

4 columns (variables)

N rows (cases)	Id	sex	age	height
	1	male	44	174
	2	male	65	180
	3	male	31	168

vector

1 - R's **basic and, in a way, only data container**.

2 - Can contain only a **single type of data** and missing values.

3 - Data types

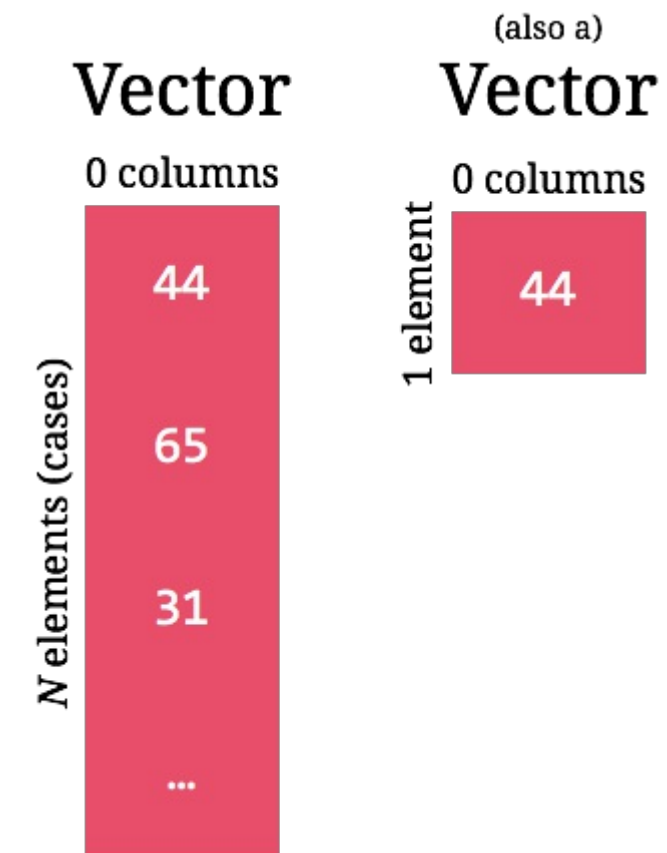
numeric - All numbers

character - All characters (e.g., names)

logical - TRUE or FALSE

...

NA - missing values



Select/Change/(Add) via []

```
# extract vector containing age
age <- baselers$age
age
```

```
## [1] 44 65 31 27 24 63 71 41 43
```

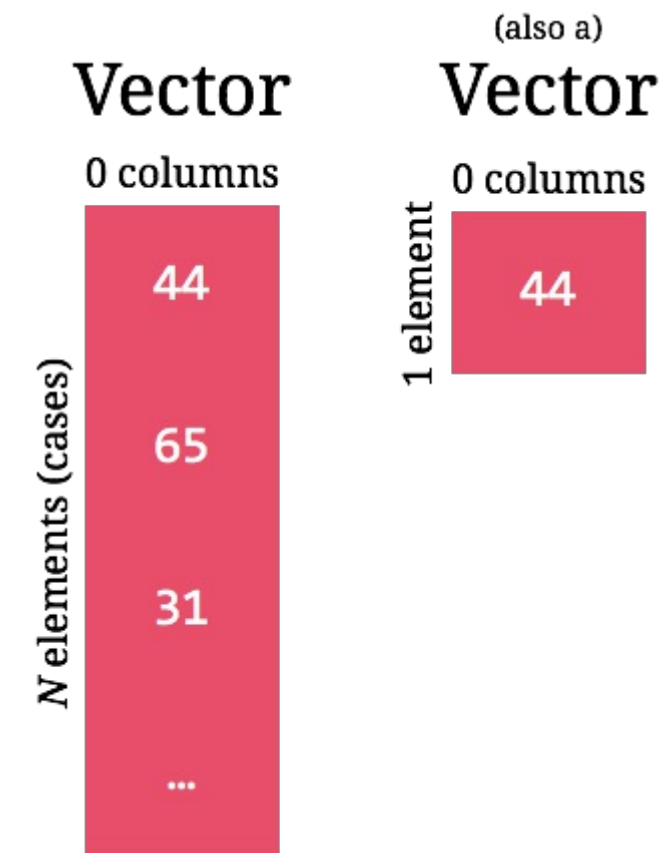
```
# select value
age[2]
```

```
## [1] 65
```

```
# change value
age[2] <- 100
age
```

```
## [1] 44 100 31 27 24 63 71 41 43
```

Find more info on indexing [here](#).



Data types: numeric

numeric vectors are used to store numbers and only numbers.

```
baselers$age

## [1] 44 65 31 27 24 63 71 41 43

# evaluate class
class(baselers$age)

## [1] "numeric"

# is age numeric?
is.numeric(baselers$age)

## [1] TRUE
```

numeric Vector	character Vector	logical Vector
.\$age	.\$sex	.\$sex=="male"
44	"male"	TRUE
65	"female"	FALSE
31	"male"	TRUE
...

Data types: character

character vectors are used to store data represented by **letters and symbols, and all other data**.

You can always recognise character vectors by **quotation marks " "**

```
baselers$sex
```

```
## [1] "male"  "male"  "female" "male"
## [5] "male"  "male"  "male"   "female"
```

```
baselers$education
```

```
## [1] "SEK_III"
## [2] "obligatory_school"
## [3] "SEK_III"
## [4] "SEK_III"
```

numeric Vector	character Vector	logical Vector
.\$age	.\$sex	.\$sex=="male"
44	"male"	TRUE
65	"female"	FALSE
31	"male"	TRUE
...

Data types: character

character vectors are used to store data represented by **letters and symbols, and all other data**.

You can always recognise character vectors by **quotation marks " "**

```
baselers$age
```

```
## [1] 44 65 31 27 24 63 71 41
```

```
# convert age to character  
as.character(baselers$age)
```

```
## [1] "44" "65" "31" "27" "24" "63" "71"  
## [8] "41" "43"
```

numeric Vector	character Vector	logical Vector
.\$age	.\$sex	.\$sex=="male"
44	"male"	TRUE
65	"female"	FALSE
31	"male"	TRUE
...

Data types: Logical

logical vector are used to **data** aka to select elements or rows. logical are typically created from other vectors via **logical comparisons**.

```
# which sex values are male?  
baselers$sex == "male"
```

```
## [1] TRUE TRUE FALSE TRUE TRUE TRUE  
## [7] TRUE FALSE
```

```
# which ages are less than 30?  
baselers$age < 30
```

```
## [1] FALSE FALSE FALSE TRUE TRUE FALSE  
## [7] FALSE FALSE FALSE
```

numeric Vector	character Vector	logical Vector
.\$age	.\$sex	.\$sex=="male"
44	"male"	TRUE
65	"female"	FALSE
31	"male"	TRUE
...

Data types: Logical

logical vector are used to **data** aka to select elements or rows. logical are typically created from other vectors via **logical comparisons**.

Logical operators

== - is equal to

<, > - smaller/greater than

<=, >= - smaller/greater than or equal

&, && - logical AND

|, || - logical OR

numeric Vector	character Vector	logical Vector
.\$age	.\$sex	.\$sex=="male"
44	"male"	TRUE
65	"female"	FALSE
31	"male"	TRUE
...

Data I/O

Raw (structured) Data

delim-separated data

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,95
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```

markup data

```
<!doctype html>
<html lang="en" class="gr__therbootcamp_github_io">
  <head>...</head>
  <body data-gr-c-s-loaded="true">
    <script async src="https://www.google-analytics.com/analytics.js">
    <script type="text/javascript" async src="https://snap.licdn.com/li
    insight.min.js"></script>
    <script type="text/javascript">
      _linkedin_data_partner_id = "111419";
    </script>
    <script type="text/javascript">...</script>
    <div id="particles-js">
      <div class="content">
        <h1>
          <span class="site-title">TheRBootcamp</span>
          <span class="site-description">Learn Data Science in R</
        <a class="link" href="#upcoming" data-scroll>
          <font size="6" color="#FF3A2A">Basel July 21 22 28 29
        </a>
      </h1>
```

Delim-separated data

- 1 - Most typical file format.
- 2 - Requires **delimiter** to separate entries.



delim-separated data

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,97
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```

readr

readr is a tidyverse package that provides convenient functions to **read in** (non-nested) data files into data frames (tibbles to be precise):



```
# Importing data from a file
```

```
data <- read_csv(file, ...) # comma-delimited  
data <- read_csv2(file, ...) # semicolon-delimited  
data <- read_delim(file, ...) # arbitrary-delimited
```

```
# Writing a data frame to a file
```

```
write_csv(data_object, path, ...) # comma-delimited  
write_delim(data_object, path, ...) # arbitrary-delimited
```

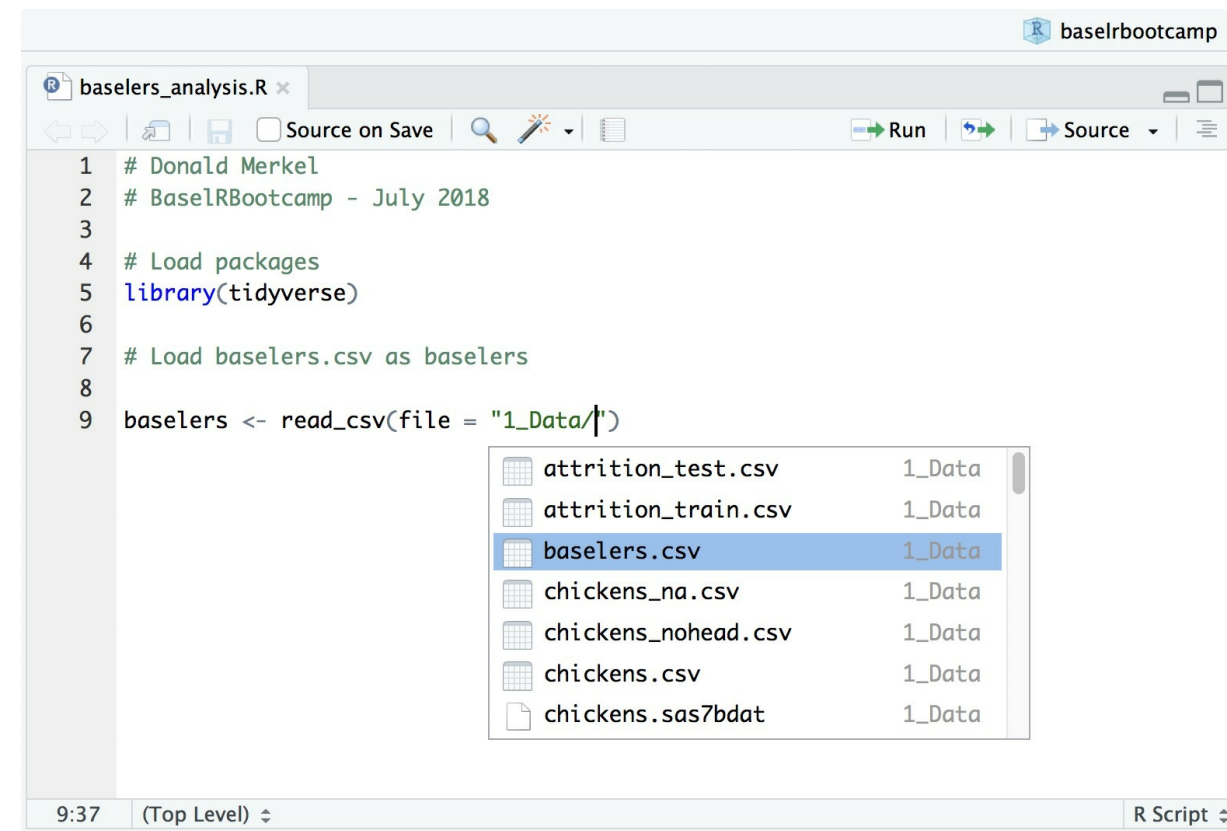

Finding the file path

1 - Identify the file path using the **auto-complete**.

2 - Initiate auto-complete and browse through the folder structure by placing the cursor between two quotation marks and using the **tab key**.



3 - Auto-complete begins with the project folder - **place your data inside your project folder!**

A screenshot of the RStudio IDE. The editor window shows a script named 'baselers_analysis.R' with the following code:

```
1 # Donald Merkel
2 # BaselRBootcamp - July 2018
3
4 # Load packages
5 library(tidyverse)
6
7 # Load baselers.csv as baselers
8
9 baselers <- read_csv(file = "1_Data/|")
```

The cursor is at the end of the string "1_Data/". An auto-complete dropdown menu is visible, listing several files in the '1_Data' directory: 'attrition_test.csv', 'attrition_train.csv', 'baselers.csv' (which is highlighted), 'chickens_na.csv', 'chickens_nohead.csv', 'chickens.csv', and 'chickens.sas7bdat'. The status bar at the bottom shows '9:37 (Top Level) R Script'.

Identifying the delimiter

- 1 - **Find the file** on your hard drive. Should be in your data folder inside your project.
- 2 - **Open the file** in RStudio (right-click on the file in the pane) a text viewer, e.g., `code editor` (Mac), `Notepad++` (Mac), `Notepad` (Windows).



baselers.csv

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,97
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```

Identifying the delimiter

- 1 - **Find the file** on your hard drive. Should be in your data folder inside your project.
- 2 - **Open the file** in RStudio (right-click on the file in the pane) a text viewer, e.g., `vim` (Mac), `notepad` (Mac), `notepad++` (Windows).

```
# Read with explicit column names
baselers <- read_delim(file = ".../baselers.csv",
                      delim = c(",",""))
```

baselers.csv

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,97
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```


Handling headers

1 - readr- functions typically expect the **column names** in the first line.

2 - If no column names are available, use the **col_names-argument** to provide them.

```
# Read with explicit column names
baselers <- read_csv(file = ".../baselers.csv",
                     col_names = c("id",
                                   "age",
                                   ...))
```

baselers.csv

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,97
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```

Handling data types

Reading in data, **readr infers the type of data** for each column.

```
# Read baselers
read_csv(file = "1_Data/baselers.csv")

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   sex = col_character(),
##   education = col_character(),
##   confession = col_character(),
##   fasnacht = col_character(),
##   eyecor = col_character()
## )

## See spec(...) for full column specifications.

## # A tibble: 10,000 x 20
##       id sex    age height weight income
##   <dbl> <chr> <dbl>  <dbl>  <dbl>  <dbl>
## 1      1 1 male    44    174.   113.   6300
```

baselers.csv

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,97
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```


Handling data types

Incorrect data types can be fixed. Typically this involves:

- 1 - **removing character elements** from otherwise numeric variables.
- 2 - Setting **explicit NA strings** using the `na`-argument.
- 3 - Re-running `type_convert`.

```
# Read baselers
baselers <- read_csv(file = ".../baselers.csv",
                     na = c('NA'))

# Try to fix incorrect data types
baselers <- type_convert(baselers)
```

baselers.csv

```
id,sex,age,height,weight,income,education,confession,children
1,male,44,174.3,113.4,6300,SEK_III,catholic,2,5,7,610,40,6,4
2,male,65,180.3,75.2,10900,obligatory_school,confessionless,
3,female,31,168.3,55.5,5100,SEK_III,NA,2,7,6,720,14,3,6,102,
4,male,27,209,93.8,4200,SEK_III,catholic,2,7,8,680,39,6,0,11
5,male,24,176.7,NA,4e3,SEK_III,catholic,1,5,4,260,19,0,1,82,
6,male,63,186.6,67.4,11400,SEK_III,evangelical-reformed,0,7,
7,male,71,151.6,83.3,12e3,SEK_III,evangelical-reformed,2,8,5
8,female,41,155.7,67.8,7600,SEK_III,confessionless,1,7,2,135
9,male,43,176.1,69.3,8500,apprenticeship,catholic,2,7,5,150,
10,female,31,166.1,66.3,6100,SEK_II,catholic,1,6,7,700,0,0,3
11,female,42,157.8,51.9,8e3,obligatory_school,catholic,2,9,7
12,male,31,165.9,66,5900,apprenticeship,evangelical-reforme
13,female,38,162.5,73.4,6200,apprenticeship,confessionless,2
14,female,49,182.8,46.9,NA,SEK_III,evangelical-reformed,1,6,
15,female,39,160,NA,5600,SEK_III,other,2,7,4,540,35,7,4,122,
16,female,54,139.7,50.3,10900,SEK_III,evangelical-reformed,3
17,female,78,153.1,64.1,11e3,SEK_III,confessionless,2,7,2,97
18,female,62,174.6,63.8,11500,SEK_III,confessionless,2,9,7,1
19,male,88,191.4,99.8,14200,SEK_III,confessionless,2,7,3,121
20,male,74,183.8,78.1,12100,apprenticeship,catholic,2,5,7,11
```

Other data

R provides **read and write functions** for practically all data file formats. See [rio](#).

readr



```
# read fixed width files (can be fast)
data <- read_fwf(file, ...)

# read Apache style log files
data <- read_log(file, ...)
```

haven



```
# read SAS's .sas7bat and sas7bcat files
data <- read_sas(file, ...)

# read SPSS's .sav files
data <- read_sav(file, ...)

# etc
```

readxl



```
# read Excel's .xls and xlsx files
data <- read_excel(file, ...)
```

Other

```
# Read Matlab .mat files
data <- R.matlab::readMat(file, ...)

# Read and wrangle .xml and .html
data <- XML::xmlParseParse(file, ...)

# from package jsonlite: read .json files
data <- jsonlite::read_json(file, ...)
```

Practical