Data types and structures

AUTHOR
Ralf Ansorg

PUBLISHED
26 September 2025

Exercise 1

Suppose you got the following data or data frame:

```
a <- c(1:5) # mouse ID
b <- c(5, 6, 3, 10, 7) # licks per hours
c <- c("17.18", "16.03", "15.9", "17.99", "14") # length in cm
d <- c("Female", "Male", "Male", "Female", "Maile") # sex of mouse
e <- c(TRUE, FALSE, TRUE, TRUE, FALSE) # healthy mouse
```

```
df <- data.frame(a=a, b=b, c=c, d=d, e=e)
```

Question 1

Which vectors (or columns of the data frame) need some work-over? If so, please write the correct code. *Hint*: try to figure this out by looking at its structure; recall coercion and accessing elements of vectors, and factors. Think about on how to assign new a value to a particular element of a vector, e.g. `my_example_vector[1]`.

Solution

Vector `c` has quotation marks and looks suspicious (the code coloring above gives it already away). The length should be a decimal number. For convenience, we are using the vectors directly since we already have them. Otherwise, with the `data.frame`, it's `df$c` for accessing or manipulation. What about vector `d`? Isn't it a category (factor in R)?

Let's check the data type of `c` to be sure first.

Length vector

```
class(c)
```

```
[1] "character"
```

```
c
```

```
[1] "17.18" "16.03" "15.9"  "17.99" "14"
```

As assumed, it is of type `character`. We need to transform the values to decimal numbers, `numeric` in R.

Length vector (2)

```
c <- as.numeric(c)
c
```

```
[1] 17.18 16.03 15.90 17.99 14.00
```

```
class(c)
```

```
[1] "numeric"
```

Perfect! Our `length` vector is now of type `numeric`. We can now apply mathemical or statistical functions to it.

Vector `d` is a category with Female and Male and (two) possible values. Since categories are factors in R, we coerce using the `as.factor()` function. Note that this is, strictly speaking, *not* necessary in our example. However, in some cases, factors behave differently then characters or strings in computations and plotting.

Sex vector

```
d <- as.factor(d)
d
```

```
[1] Female Male   Male   Female Maile
Levels: Female Maile Male
```

Wait, the levels indicate all available categories. There are three of them, obviously due to a spelling mistake. Let's correct it and run it again.

Sex vector (2)

```
d[5] <- "Male" # was misspelled as "Maile"
d <- as.factor(d)
d
```

```
[1] Female Male   Male   Female Male
Levels: Female Maile Male
```

The levels haven't changed although we corrected the spelling. What happened? Since `d` already had levels assigned, we also need to re-assign the levels.

Sex vector (3)

```
d <- factor(d, levels = c("Female", "Male"))
d
```

```
[1] Female Male   Male   Female Male
Levels: Female Male
```

It would have been easier to first correct the spelling (cleaning the data) and then make it a factor (i.e. just running (2)). Note that we *must* use the `factor()` function to set the levels.

If you have an *ordered* factor, you can also change the order like this (there is, however, an entire package for it, `forcats`, which is part of the `tidyverse` package we will use later).

Question 2

Add all vectors to a new data frame `df_clean` with proper/meaningful headers (instead of `a` ... `e`).

Solution

You can simply use column names you want to use and assign the vectors `a` ... `e`. Best practice is to avoid spaces. Use a single word or combine words with _ (underscore) or - (dash, *not very*

common!): `length_cm` or `licks_hr`. In this example, adding the unit may enhance the readability of your data (for humans).

If you get data with messy headers, the `janitor` package helps.

Clean data frame

```
df_clean <- data.frame(ID=a, licks_hr=b, length_cm=c, sex=d, healthy=e)
df_clean
```

```
  ID licks_hr length_cm    sex healthy
1  1        5     17.18 Female    TRUE
2  2        6     16.03   Male   FALSE
3  3        3     15.90   Male    TRUE
4  4       10     17.99 Female    TRUE
5  5        7     14.00   Male   FALSE
```

Question 3

What is the mean of the length of the mice? *Hint*: you could use the `mean()` function. What is the median?

Solution

We can simply run the `mean()` and `median()` functions on the `length` column.

```
mean(df_clean$length_cm)
```

```
[1] 16.22
```

```
median(df_clean$length_cm)
```

```
[1] 16.03
```

Inline computations are also possible if you want to insert values or the result of a function in a text:

> The mean length of the mice is 16.22 and the median is 16.03.[1]

Question 4

Look at the structure **and** summary if your cleaned-up data is reasonable and briefly explain why. Bullet points only (use the * or - symbol), no essay.

Solution

Review the structure and summary from Question 1 and compare it with our cleaned data frame (Question 2). If you want to check for data types, `str()` is probably the better and more compact choice. When we start working with the `tidyverse` package, there is an advanced data.frame (`tibble`) which prints the data types below the column names.

```
str(df)
```

```
'data.frame':   5 obs. of  5 variables:
 $ a: int  1 2 3 4 5
 $ b: num  5 6 3 10 7
```

[1] Check the .qmd file on how to have inline code!

```
 $ c: chr  "17.18" "16.03" "15.9" "17.99" ...
 $ d: chr  "Female" "Male" "Male" "Female" ...
 $ e: logi  TRUE FALSE TRUE TRUE FALSE
```

```
summary(df)
```

```
      a           b             c                  d
 Min.   :1   Min.   : 3.0   Length:5           Length:5
 1st Qu.:2   1st Qu.: 5.0   Class :character   Class :character
 Median :3   Median : 6.0   Mode  :character   Mode  :character
 Mean   :3   Mean   : 6.2
 3rd Qu.:4   3rd Qu.: 7.0
 Max.   :5   Max.   :10.0
    e
 Mode :logical
 FALSE:2
 TRUE :3
```

```
str(df_clean)
```

```
'data.frame':   5 obs. of  5 variables:
 $ ID       : int  1 2 3 4 5
 $ licks_hr : num  5 6 3 10 7
 $ length_cm: num  17.2 16 15.9 18 14
 $ sex      : Factor w/ 2 levels "Female","Male": 1 2 2 1 2
 $ healthy  : logi  TRUE FALSE TRUE TRUE FALSE
```

```
summary(df_clean)
```

```
      ID       licks_hr      length_cm         sex     healthy
 Min.   :1   Min.   : 3.0   Min.   :14.00   Female:2   Mode :logical
 1st Qu.:2   1st Qu.: 5.0   1st Qu.:15.90   Male  :3   FALSE:2
 Median :3   Median : 6.0   Median :16.03              TRUE :3
 Mean   :3   Mean   : 6.2   Mean   :16.22
 3rd Qu.:4   3rd Qu.: 7.0   3rd Qu.:17.18
 Max.   :5   Max.   :10.0   Max.   :17.99
```

Comments/Bullet points

- Variable/vector `length` had to be converted to numerical (from character)
- Variable/vector `sex` is categorical, hence a factor (without any order)
- Factors and logicals are displayed differently: counts instead of the descriptive statistics[2]

Bonus

If you know that `FALSE` and `TRUE` are internally represented as `0` and `1`. What is the number of healthy mice (computed)?

Use the `sum()` function to count all `TRUE`s in column `healthly` which returns 3 (compare it to the `summary()` output above).

[2] Min, 25% Qrt, Median, Mean, 75% Qrt, Max