Modificare le stringhe e i vettori carattere

All'interno di un dataframe, una variabile testuale — in quanto vettore — può essere gestita come le altre:

```
library(tidyverse)
```

```
starwars %>%
  rename("nome" = name) %>%
  head()
```

```
# A tibble: 6 x 14
  nome     height  mass hair_color  skin_color eye_color birth_year
sex   gender
  <chr>     <int> <dbl> <chr>       <chr>      <chr>          <dbl>
<chr> <chr>
1 Luke Sk~    172    77 blond       fair       blue            19
male  mascu~
2 C-3PO       167    75 NA          gold       yellow         112
none  mascu~
3 R2-D2        96    32 NA          white, bl~ red             33
none  mascu~
4 Darth V~    202   136 none        white      yellow          41.9
male  mascu~
5 Leia Or~    150    49 brown       light      brown           19
fema~ femin~
6 Owen La~    178   120 brown, grey light      blue            52
male  mascu~
# ... with 5 more variables: homeworld <chr>, species <chr>, films
<list>,
#   vehicles <list>, starships <list>
```

Un po' più complicato invece è trasformare i valori.

Trasformazioni di base

Tab. 1: Funzioni dei pacchetti di base

`tolower()`	tutti caratteri minuscoli
`toupper()`	TUTTI CARATTERI MAIUSCOLI
`paste()`, `paste0()`	incollare, unire stringhe
`strsplit()`	dividere stringhe

Tab. 2: Funzioni di *stringr* (tidyverse)

`str_to_lower()`	tutti caratteri minuscoli
`str_to_upper()`	TUTTI CARATTERI MAIUSCOLI
`str_to_title()`	Tutte Le Iniziali Maiuscole
`str_to_sentence()`	La prima parola di ogni frase in maiuscolo
`str_trim()`	elimina gli spazi all'inizio e/o alla fine
`str_squish()`	elimina gli spazi all'inizio, alla fine, e quelli ripetuti in mezzo
`str_pad()`	aggiunge spazi o altri caratteri a destra e/o a sinistra
`str_split()`	dividere stringhe

Il pacchetto *stringr*, è caricato con `library(tidyverse)`.

A meno che non venga diversamente specificato nel seguito, tutte queste funzioni sono applicabili ad una variabile nei modi noti.

Comando base:

```
dataset$variabile <- tolower(dataset$variabile)
```

Con `mutate()`:

```
mutate(starwars, name = tolower(name))
```

o

```
starwars %>%
  mutate(name = tolower(name))
```

```
# A tibble: 87 x 14
   name       height  mass hair_color  skin_color eye_color
birth_year sex   gender
   <chr>       <int> <dbl> <chr>       <chr>      <chr>
<dbl> <chr> <chr>
 1 luke skyw~    172    77 blond       fair       blue            19
male  mascul~
 2 c-3po         167    75 NA          gold       yellow         112
none  mascul~
 3 r2-d2          96    32 NA          white, bl~ red             33
none  mascul~
 4 darth vad~    202   136 none        white      yellow
41.9 male  mascul~
 5 leia orga~    150    49 brown       light      brown           19
fema~ femini~
 6 owen lars     178   120 brown, grey light      blue            52
```

```
male  mascul~
 7 beru whit~    165    75 brown       light      blue            47
fema~ femini~
 8 r5-d4          97    32 NA          white, red red             NA
none  mascul~
 9 biggs dar~    183    84 black       light      brown           24
male  mascul~
10 obi-wan k~    182    77 auburn, wh~ fair       blue-gray       57
male  mascul~
# ... with 77 more rows, and 5 more variables: homeworld <chr>,
species <chr>,
#   films <list>, vehicles <list>, starships <list>
```

senza dimenticare, eventualmente, di salvare il risultato:

```
# non eseguire
starwars <- starwars %>%
              mutate(name = tolower(name))
```

Maiuscole e minuscole

```
frase <- "Sempre caro mi fu quest\'ermo colle,
e questa siepe, che da tanta parte
dell\'ultimo orizzonte il guardo esclude."
```

Stringa tutta in minuscolo

```
tolower(frase)
```

oppure

```
# stringr
str_to_lower(frase)
```

```
[1] "sempre caro mi fu quest'ermo colle,\ne questa siepe, che da
tanta parte\ndell'
ultimo orizzonte il guardo esclude."
```

Stringa tutta in maiuscolo

```
toupper(frase)
```

oppure

```
# stringr
```

```
str_to_upper(frase)
```

```
[1] "SEMPRE CARO MI FU QUEST'ERMO COLLE,\nE QUESTA SIEPE, CHE DA
TANTA PARTE\nDELL'
ULTIMO ORIZZONTE IL GUARDO ESCLUDE."
```

Iniziali tutte in maiuscolo (title)

```
str_to_title("ricerca SOCIALE con R")
```

```
[1] "Ricerca Sociale Con R"
```

Prima parola con iniziale in maiuscolo (sentence)

```
str_to_sentence("ricerca SOCIALE con R")
```

```
[1] "Ricerca sociale con r"
```

Spazi bianchi

```
# stringr: inizio e fine
str_trim(" Mela  rossa ")
```

```
[1] "Mela  rossa"
```

```
# stringr: inizio, fine, e in mezzo
str_squish(" Mela  rossa ")
```

```
[1] "Mela rossa"
```

```
# stringr
str_pad("Mela", 10, side = "both", pad = "-")
```

```
[1] "---Mela---"
```

Unire, incollare stringhe

Per unire i valori (le stringhe) di due o più variabili di testo, vedi unite, separate (tidyr)

Queste funzioni sono utili per rinominare le variabili, o per gestire elementi di testo quali le etichette o i titoli dei grafici (come nell'esempio riportato in Grafici a torta).

```
paste("Mela", "Rossa")
```

```
[1] "Mela Rossa"
```

Possiamo scegliere il carattere che unisce le stringhe con `sep =`:

```
paste("Mela", "Rossa", sep = "_")
```

```
[1] "Mela_Rossa"
```

paste0() e str_c()

`paste0(x)` equivale a `paste(x, sep = "")`

```
paste0("Mela", "Rossa")
```

```
[1] "MelaRossa"
```

ed equivale alla funzione `str_c()` di *stringr*:

```
str_c("Mela", "Rossa")
```

```
[1] "MelaRossa"
```

Questa funzione è comoda quando vogliamo usare diversi elementi di separazione fra le stringhe:

```
# le stringhe sono separate da spazi
paste(starwars$name[1:5], starwars$eye_color[1:5], "eyes")
```

```
[1] "Luke Skywalker blue eyes" "C-3PO yellow eyes"
[3] "R2-D2 red eyes"           "Darth Vader yellow eyes"
[5] "Leia Organa brown eyes"
```

```
# le stringhe non sono separate
paste0(starwars$name[1:5], ": ", starwars$eye_color[1:5], " eyes")
```

```
[1] "Luke Skywalker: blue eyes" "C-3PO: yellow eyes"
[3] "R2-D2: red eyes"           "Darth Vader: yellow eyes"
[5] "Leia Organa: brown eyes"
```

collapse

Con l'argomento `collapse`, infine, possiamo ridurre questo vettore di lunghezza 5 a un vettore di lunghezza 1 (un'unica stringa di testo, ad esempio per creare un titolo):

```
paste0(starwars$name[1:5], ": ", starwars$eye_color[1:5], " eyes",
       collapse = "; ")
```

o anche

```
str_c(starwars$name[1:5], ": ", starwars$eye_color[1:5], " eyes",
       collapse = "; ")
```

```
[1] "Luke Skywalker: blue eyes; C-3PO: yellow eyes; R2-D2: red eyes;
Darth
Vader: yellow eyes; Leia Organa: brown eyes"
```

Dividere stringhe

Per dividere i valori (le stringhe) di variabili di testo in due o più altre variabili, vedi unite, separate (tidyr)

Queste funzioni consentono di usare espressioni regolari (regexp).

```
strsplit("text mining", split = " ")
```

```
# stringr
str_split("text mining", pattern = " ")
```

In entrambi i casi, il risultato è una lista:

```
# il risultato è una lista
[[1]]
[1] "text"   "mining"
```

Ciò si rivela utile quando si applichino tali funzioni a un vettore carattere composto di più elementi:

```
strsplit(starwars$name[1:5], split = " ")
```

```
[[1]]
[1] "Luke"      "Skywalker"
```

```
[[2]]
[1] "C-3PO"

[[3]]
[1] "R2-D2"

[[4]]
[1] "Darth" "Vader"

[[5]]
[1] "Leia"   "Organa"
```

Cercare e sostituire testo

Vedi Ricerca e sostituzione di testo

Formattare il testo

Vedi

- formattare_testi_numeri
- format()

Controlli ortografici

Vedi Correzione ortografica dei testi

Script di esempio

E' possibile scaricare ed eseguire lo script dell'esempio:

nome_file.R

Gestione dei dati, Analisi testuale, tidyverse