Barcelona R

Introduction to Data Manipulation

Nicolas Attalides

Workshop Setup:

Wi-Fi

- Network Name: N/A
- Password: N/A

Resources

- R (version 4.0.1)
- RStudio (version 1.3.959)

Packages

- tidyverse

What is tidyverse?

Tidyverse is a collection of R packages that are designed for data science tasks, more specifically for data manipulation, transformation, exploration and visualisation.

These packages share a common design philosophy and contain functions that are consistent and uniform in coding style.

You can read more at https://www.tidyverse.org/

Topics

- Workshop aim:

Learn how to do data manipulations using tidyverse packages.

- Topics:
  - Learn the “verbs” with 

  - Improve your workflow with 

  - Simple string manipulation using 

Learn the “verbs” with 

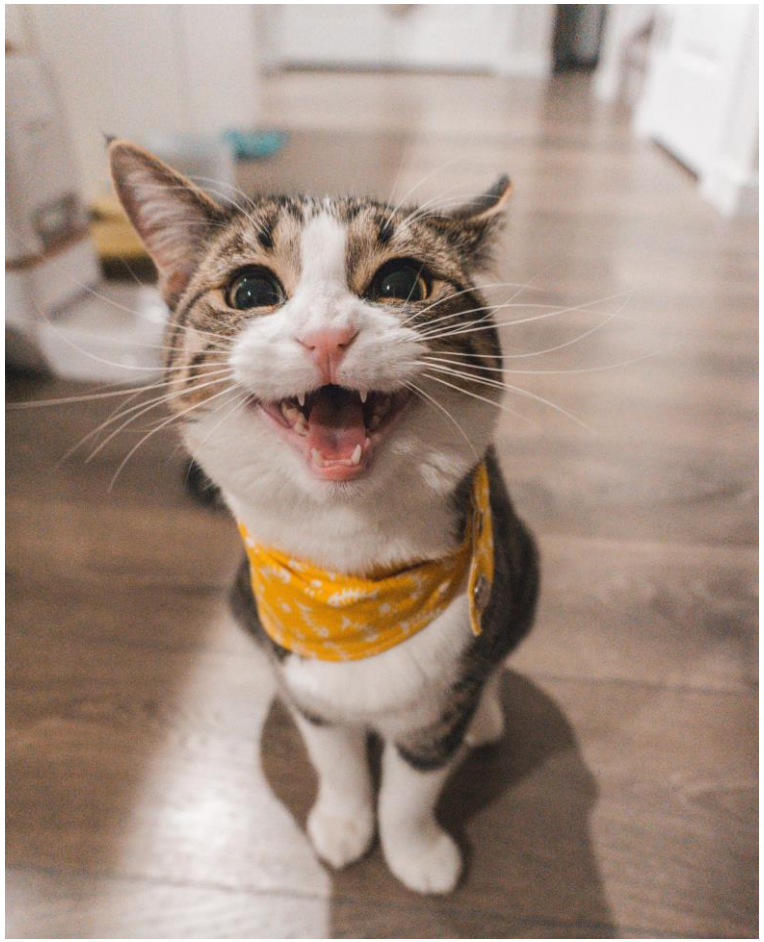
One of the most commonly used R packages when dealing with data manipulations is **{dplyr}**. It is very powerful in handling tabular data such as data frames and is easy to use through “verb” functions. You can use **{dplyr}** to:

- **Select** columns from your data
- **Filter** your data to keep the rows that meet some conditions
- **Arrange** your data in some order
- **Mutate** your data and create new columns
- **Group** and **summarise** your data

```
library(tidyverse)

view(starwars)
```

Live Coding Example 1



Use the starwars dataset from the **{dplyr}** package to:

1. Select the columns: “name”, “height”, “mass”, “species”.
2. Filter the rows to keep only those characters that are greater than or equal to 175cm.
3. Filter the rows to keep only the “Human” characters.
4. Arrange the rows according to descending “mass” values.
5. Who is the character on the first row?

Live Coding Example 1

selected columns

arranged rows

	name	height	mass	species
1	Darth Vader	202	136.0	Human
2	Owen Lars	178	120.0	Human
3	Jek Tono Porkins	180	110.0	Human
4	Qui-Gon Jinn	193	89.0	Human
5	Gregar Typho	185	85.0	Human
6	Biggs Darklighter	183	84.0	Human
7	Anakin Skywalker	188	84.0	Human
8	Mace Windu	188	84.0	Human
9	Han Solo	180	80.0	Human
10	Dooku	193	80.0	Human
11	Lando Calrissian	177	79.0	Human
12	Lobot	175	79.0	Human
13	Jango Fett	183	79.0	Human
14	Raymus Antilles	188	79.0	Human
15	Boba Fett	183	78.2	Human
16	Obi-Wan Kenobi	182	77.0	Human
17	Wilhuff Tarkin	180	*NA*	Human
18	Cliegg Lars	183	*NA*	Human
19	Bail Prestor Organa	191	*NA*	Human

filtered rows

Who is the character on the first row?

Live Coding Example 1

```r
# Select columns
df <- select(starwars, name, height, mass, species)

# Filter rows by height condition
df <- filter(df, height >= 175)

# Filter rows by species condition
df <- filter(df, species == "Human")

# Arrange rows by descending mass
df <- arrange(df, desc(mass))
```

Mutate your data

Very often you will want to create new columns from your existing data. The function **`mutate()`** in the **{dplyr}** package can be used to do exactly this task.

You can actually create multiple columns in a single function call.

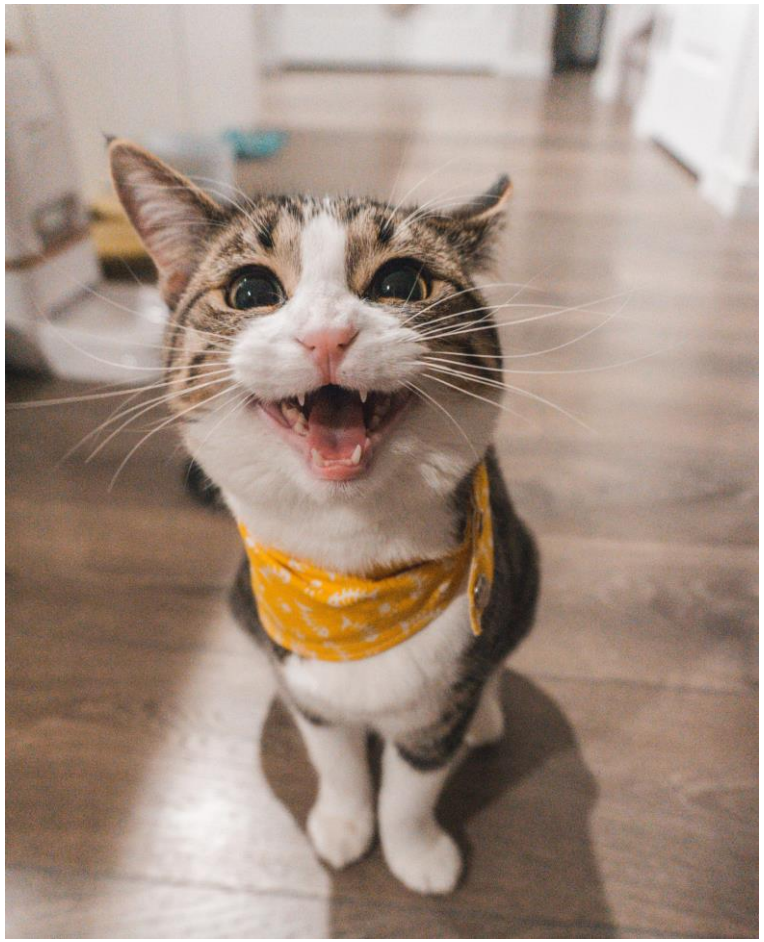
```
# Create a column for height in metres
df <- mutate(starwars, height_m = height/100)
```

Group and summarise your data

Another very common task is to group your data by a column (or more than one column) and then create summarised values for the grouped data. The functions **`group_by()`** and **`summarise()`** in the **{dplyr}** package make it very easy to do these transformations.

```
# Find the min/mean/max mass value for each species category
df <- summarise(group_by(starwars, species),
                min_mass = min(mass, na.rm = TRUE),
                mean_mass = mean(mass, na.rm = TRUE),
                max_mass = max(mass, na.rm = TRUE))
```

Live Coding Example 2



Use the starwars dataset to:

1. Remove the columns “films”, “vehicles”, “starships” from the data.
2. Remove rows that have missing mass values.
3. Calculate the Body Mass Index (BMI) for each character*.
4. Arrange the rows by descending BMI … who do you think is at the top?
5. Find the median BMI value for each gender category.

*BMI = weight (kg) / height^2 (m)

Use minus sign “-” to remove columns

Live Coding Example 2

Who is the character with the highest BMI?

	name	height	mass	hair_color	skin_color	eye_color	birth_year	gender	homeworld	species	height_m	BMI
1	Jabba Desilijic Tiure	175	1358.0	*NA*	green-tan, brown	orange	600.0	hermaphrodite	Nal Hutta	Hutt	1.75	443.42857
2	Dud Bolt	94	45.0	none	blue, grey	yellow	*NA*	male	Vulpter	Vulptereen	0.94	50.92802
3	Yoda	66	17.0	white	green	brown	896.0	male	*NA*	Yoda's species	0.66	39.02663
4	Owen Lars	178	120.0	brown, grey	light	blue	52.0	male	Tatooine	Human	1.78	37.87401
5	IG-88	200	140.0	none	metal	red	15.0	none	*NA*	Droid	2.00	35.00000
6	R2-D2	96	32.0	*NA*	white, blue	red	33.0	*NA*	Naboo	Droid	0.96	34.72222

	gender	median_BMI
1	female	18.06751
2	hermaphrodite	443.42857
3	male	24.70827
4	none	35.00000
5	*NA*	34.00999

Investigate data quality

Live Coding Example 2

```
# Select columns
df <- select(starwars, -films, -vehicles, -starships)

# Filter rows that have missing mass
df <- filter(df, !is.na(mass))

# Create columns: height in metres and the Body Mass Index (BMI)
df <- mutate(df, height_m = height/100, BMI = mass / (height_m)^2)

# Arrange rows according to descending “BMI” values
df <- arrange(df, desc(BMI))

# Calculate the median BMI value for each gender
df <- summarise(group_by(df, gender), median_BMI = median(BMI))
```

Summary of {dplyr} "verb" functions

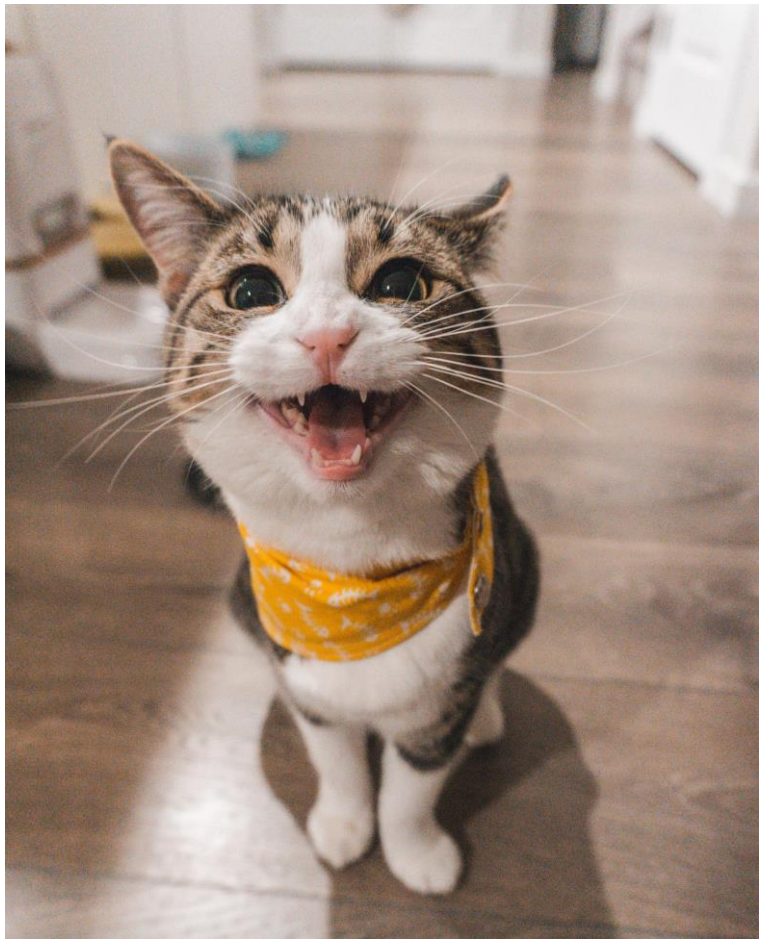
Function	Description
select	Select columns by name
filter	Filter rows that meet a condition
arrange	Arrange rows to some order
mutate	Mutate data to create new columns
group_by	Group data by columns
summarise	Summarise data to values

Improve your workflow with

A package that has changed the way we write R code is called **{magrittr}**. It has significantly improved the readability and workflow of code by introducing the "pipe" operator. It acts as a "then" operation where we can pass data from one function to another function very easily.

Fun fact: The package name is inspired by the famous artist René Magritte. One of his work, a pipe, has the text "this is not a pipe" as a caption … this is where the **{magrittr}** package gets its image.

Live Coding Example 3



Repeat Example 1 using the pipe operator from the **{magrittr}** package.

1. Select the columns: “name”, “height”, “mass”, “species” THEN filter the rows to keep only those characters that are greater than or equal to 175cm THEN filter the rows to keep only the human characters THEN arrange the rows according to descending “mass” values.

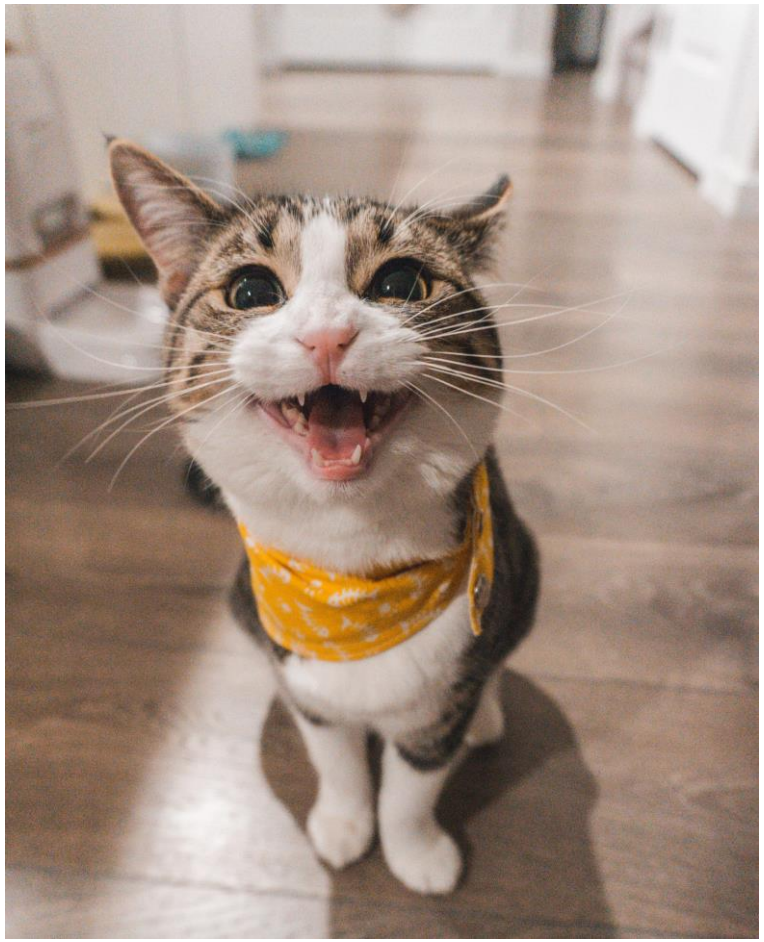
Live Coding Example 3

```
library(magrittr)


# Pipe each data manipulation operation to the next one
df <- starwars %>%
  select(name, height, mass, species) %>%
  filter(height >= 175) %>%
  filter(species == "Human") %>%
  arrange(desc(mass))
```

Try **CTRL+SHIFT+M** (Windows) **CMD+SHIFT+M** (Mac) and see what happens

Live Coding Example 4



Repeat Example 2 using the pipe operator from the **{magrittr}** package.

1. Remove the columns “films”, “vehicles”, “starships” from the data THEN remove rows that have missing mass values THEN calculate the Body Mass Index (BMI) for each character THEN arrange the rows by descending BMI THEN find the median BMI value for each gender category.

Live Coding Example 4

```
# Pipe each data manipulation operation to the next one
df <- starwars %>%
  select(-films, -vehicles, -starships) %>%
  filter(!is.na(mass)) %>%
  mutate(height_m = height/100,
         BMI = mass / (height_m)^2) %>%
  arrange(desc(BMI)) %>%
  group_by(gender) %>%
  summarise(median_BMI = median(BMI))
```

Simple string manipulation using 

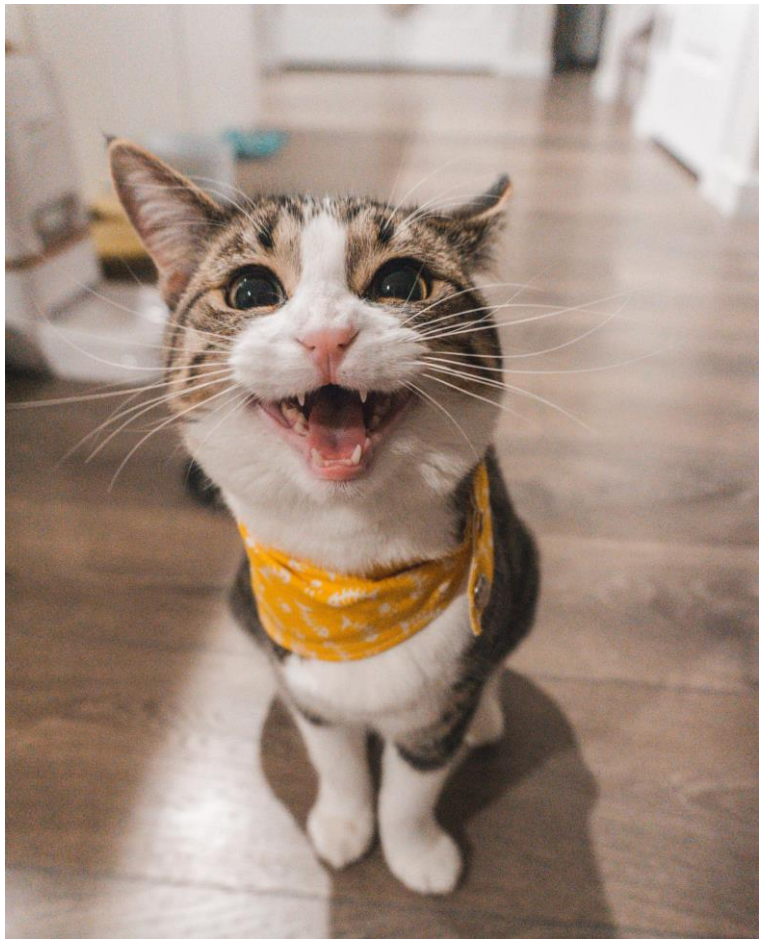
The package in the tidyverse collection that helps us do data manipulations involving strings is called **{stringr}**. String manipulation is another common task, especially in data cleaning and pre-processing. Here are some examples:

```
# Make all character names as lower case
string <- str_to_lower(starwars$name)

# Combine the name, hair colour & eye colour of characters in a sentence
string <- str_c(starwars$name, " has ",
                starwars$hair_color, " hair and ",
                starwars$eye_color, " eyes.")

# Create an indicator where the specific pattern matches
ind <- str_detect(string = starwars$name, pattern = "Skywalker")
```

Live Coding Example 5



Use the starwars dataset to:

1. Transform the character names to upper case.
2. Combine the “name” and the “homeworld” to create a sentence, for example: “Luke Skywalker is from Tatooine”.
3. Create an indicator for the rows where characters have green skin.

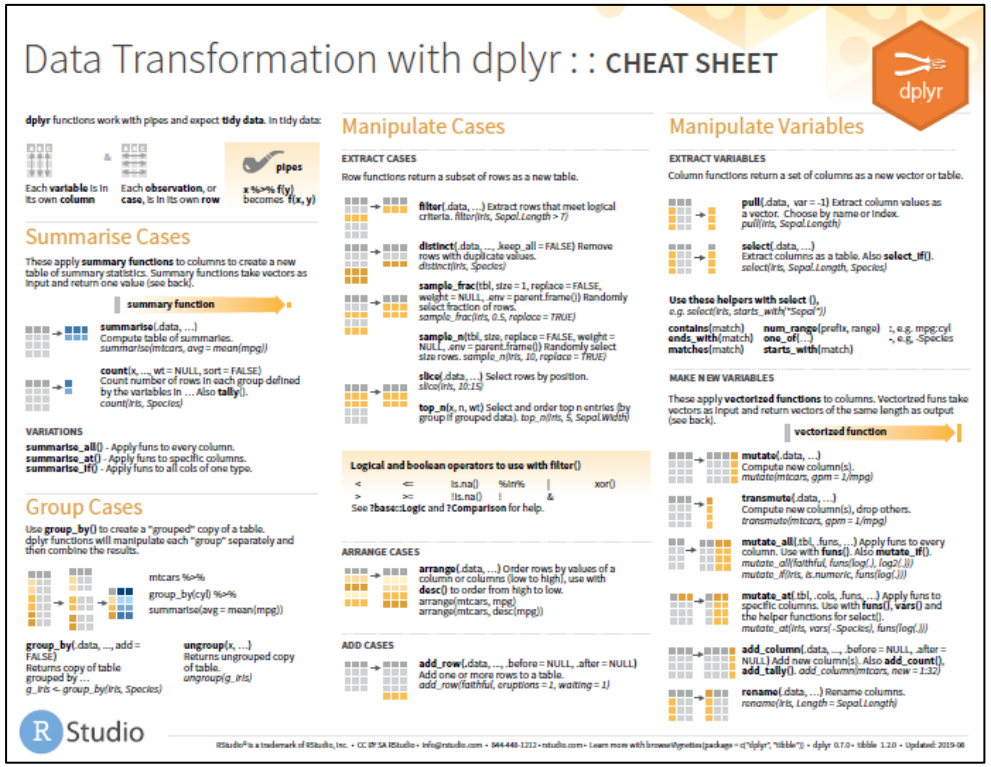
Live Coding Example 5

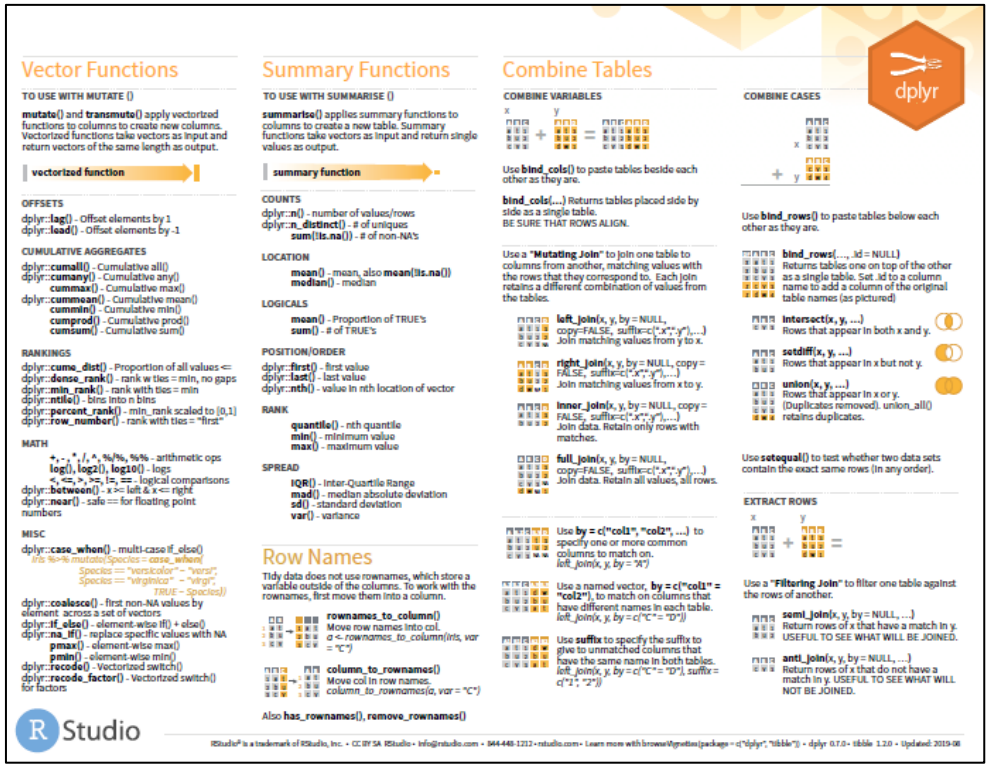
```
# Make all character names as upper case
string <- str_to_upper(starwars$name)

# Combine the name, hair colour & eye colour of characters in a sentence
string <- str_c(starwars$name, " is from ",
                starwars$homeworld, ".")

# Create an indicator where the specific pattern matches
ind <- str_detect(string = starwars$skin_color, pattern = "green")
```

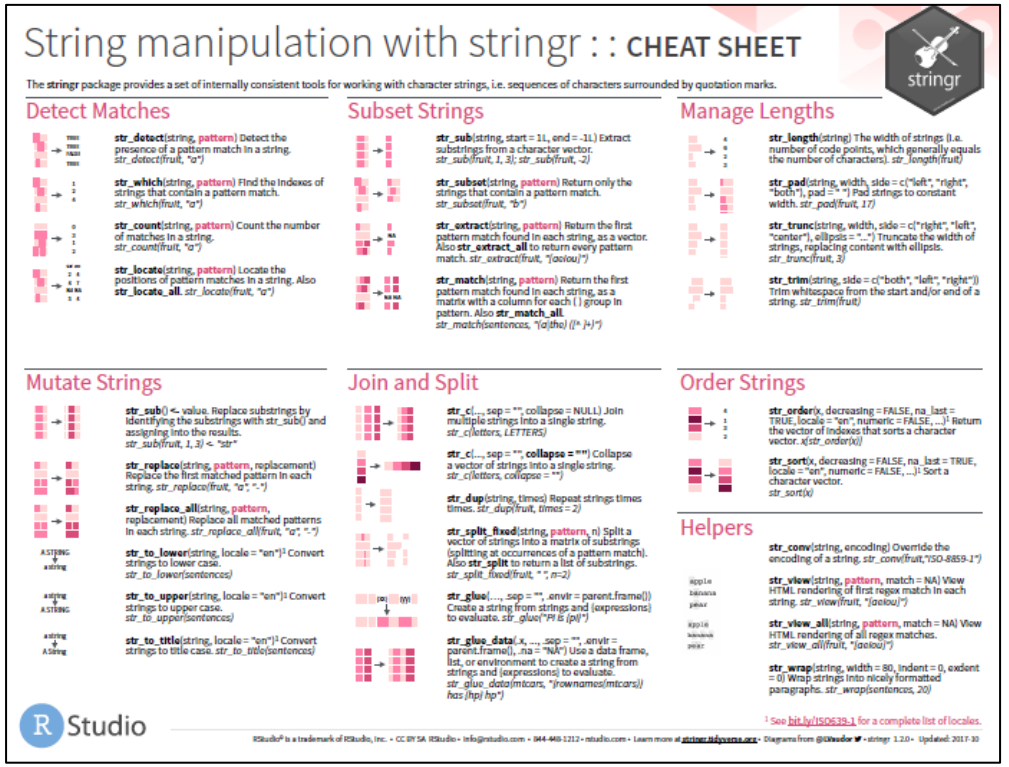
Other resources – {dplyr} cheat sheet

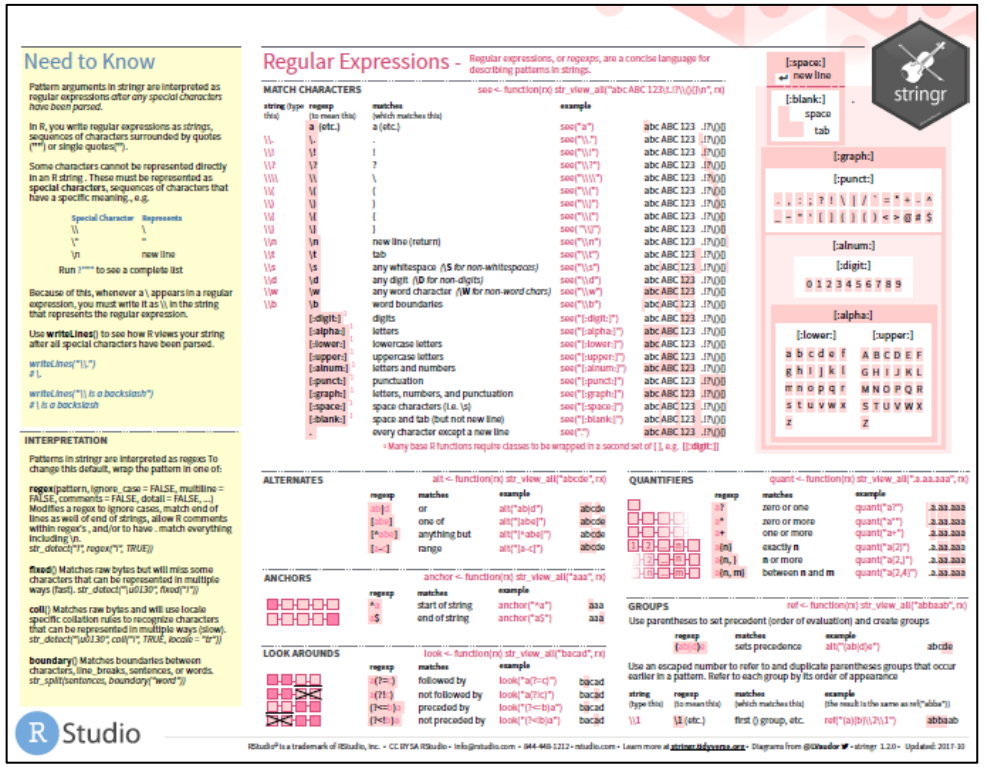




Get the cheat sheet at: https://rstudio.com/resources/cheatsheets/

Other resources – {stringr} cheat sheet





Get the cheat sheet at: https://rstudio.com/resources/cheatsheets/

Barcelona R

Thank you to our sponsors and partners!