FDS::**CHEATSHEETS**["project 1"]

EPFL EXTENSION SCHOOL

Hints

How to start?

- Create an R Markdown file with the name "Report.Rmd". This R Markdown file will contain your report and is the file you knit when you finish the report. Knitting it will create the "Report.html" file.
- Create a folder called "data". This folder should contain any source files used in your code, such as any .csv files you import in your report.

- **Part 1**

- Import and clean the dataset.
- Use dplyr::**rename()** to change column names, if necessary.
- Store the dataset in a variable.
- Focus on the dplyr functions to answer the questions.

Part 2

- Use the dataset created in Part 1 to make a bar plot.
- Use ggplot2::**labs()** to add a title and a subtitle.

Part 3

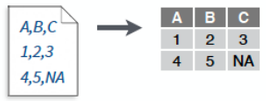
- Correct the column number_of_employees by recreating it using dplyr::**mutate()** and dplyr::**case_when()**.
- Store the dataset in a new variable.
- Use the dataset with the corrected column to create a bar plot.

Part 4

- Use dplyr::**mutate()** to add the columns percent_of_phd, percent_of_postdocs, and percent_of_prof and store the dataset in a new variable.
- Select the columns survey_id, percent_of_phd, percent_of_postdocs, and percent_of_prof using dplyr::**select()**.
- Use tidyr::**pivot_longer()** to reshape the dataset. Use the help to get more information by typing `?pivot_longer()` in the console. Alternatively, download the provided dataset which you can find in the "Resources" tab on the platform.
- Create a bar plot using ggplot2::**geom_col()**. Remember to add labels.

Data import

readr::**read_csv("file.csv")** Allows you to read a comma delimited file.



Data cleaning

janitor::**clean_names()** Creates columns names in `snake_case`.

dplyr::**rename(**.data, ...**)** Renames columns.

Data exploring

dplyr::**glimpse(**.data**)** Prints an overview of your dataset.

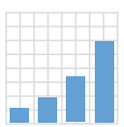
Data visualisation

ggplot2::**ggplot(**.data = mpg, aes(x = cty, y = hwy)**)** Begins a plot that you finish by adding layers to. Add one geom function per layer with a "+".

ggplot2::**labs()** Labels the elements of your plot.

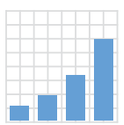
```
labs(x = "New x axis label",
     y = "New y axis label",
     title ="Add a title above the plot",
     subtitle = "Add a subtitle below title",
     caption = "Add a caption below plot", …)
```

PLOT ONE DISCRETE VARIABLE



ggplot2::**ggplot**(mpg, aes(fl)) **+**
ggplot2::**geom_bar()**

PLOT ONE DISCRETE AND ONE CONTINUOUS VARIABLE



ggplot2::**ggplot**(mpg, aes(class, hwy)) **+**
ggplot2::**geom_col()**

Data wrangling

SUMMARISING

dplyr::**summarise(**.data, ... **)** Applies summary functions to columns to create a new table. Summary functions take vectors as input and return single values as output.

dplyr::**count(**.data, ..., wt = NULL, sort = FALSE, name = NULL**)** Counts number of rows in each group defined by the variables in ... .

TO USE WITH SUMMARISE()

dplyr::**n()** - number of values/rows

ARRANGING

dplyr::**arrange(**.data, ...**)** Orders rows by values of a column or columns (low to high), use with **desc()** to order from high to low.

EXTRACTING COLUMNS OR ROWS

dplyr::**select**(.data, ...) Extracts column(s) as a table.

dplyr::**pull**(.data, var=-1, ...) Extracts column values as a vector, by name or index.

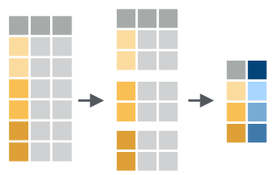
dplyr::**slice_max(**.data, order_by, ..., n, ... **)** Select rows with the highest values.

CREATING VARIABLES

dplyr::**mutate**(.data, ...) Computes new column(s).

GROUPING

dplyr::**group_by(**.data, ..., .add = FALSE, .drop = TRUE**)** Creates a "grouped" copy of a table grouped by columns in ... . dplyr functions manipulate each "group" separately and combine the results.



```
mtcars  %>%
   group_by(cyl) %>%
   summarise(n =n ())
```

dplyr::**ungroup**(.data, ...) Returns ungrouped copy of table.