

Das ist ein Quarto-Dokument

Hier kommt dein Name

Einleitung

Dieses Quarto-Dokument könnt ihr als PDF anzeigen lassen.

Um aus einem Quarto-Dokument ein PDF erstellen zu können, muss eine LaTeX-Distribution installiert sein. Am einfachsten geht das mit dem Paket TinyTex. Dazu muss man lediglich diese zwei Zeilen Code in dieser Reihenfolge ausführen. Diese Installation kann eine Weile dauern.

```
# install.packages("tinytex") # den «#» vor dem Befehl bei euch entfernen  
# tinytex::install_tinytex() # den «#» vor dem Befehl bei euch entfernen
```

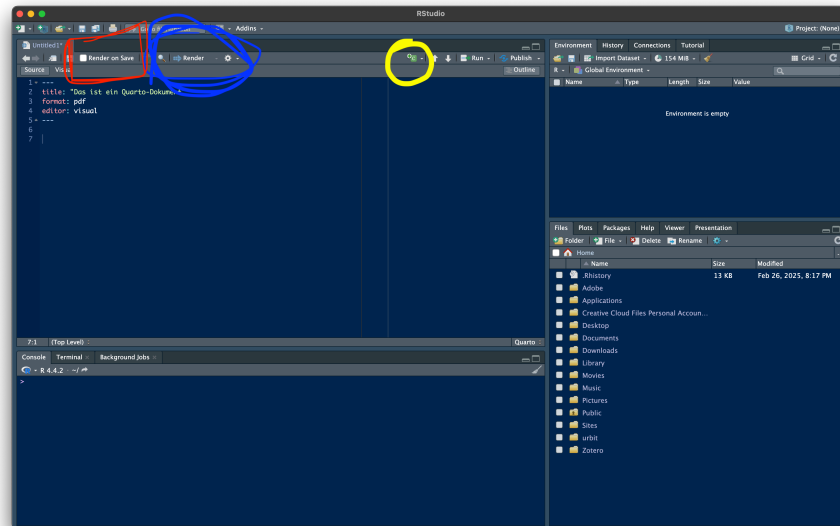


Figure 1: Ein geöffnetes Quarto-Dokument.

Setzt man nun oben links im Editor das Häkchen bei «Render on Save» (rot), wird beim Speichern aus einem Quarto-Dokument ein PDF erzeugt - oder man klickt einfach auf den blauen Pfeil (blau), der mit «Render» beschriftet ist, vgl. Figure 1.

Die Abbildungen (vgl. Figure 1.) sind noch auf Englisch beschriftet, wir wollen das aber auf Deutsch. Dazu geht ihr in die YAML (das ist das Ding ganz oben, wo `title` usw. steht) und ändert bei `lang` «en» zu «de».

Code-Chunks

Das hier ist ein Code-Chunk:

```
# Diese Bereiche sind sog. Code-Chunks. Hier kommt Code. In allen anderen
# Bereichen kommt normaler Text. In einem Quarto-Dokument kann man also
# normalen Fliesstext schreiben. In einem R-Skript kann man nur Text
# schreiben, wenn vorher ein «#» steht. Auch in den Code-Chunks in einem
# Quarto Dokument (also genau dieser Bereich hier) muss man einen «#»
# verwenden, wenn man Text schreiben will.
```

Mit dem Button (gelb) in Figure 1 erstellt ihr einen neuen Code-Chunk.

Aufgabe 1

1. Erstellt einen neuen Code-Chunk. Ladet dort das `tidyverse`. Ihr kennt bereits zwei Befehle, um Pakete zu laden.
2. Gebt dem Code-Chunk das `label` «Pakete laden».
3. Erstellt unter dem `label` den `include`-Tag und den `message`-Tag.
4. Rendert das Dokument einmal mit (1) `include` auf «true» und `message` auf «true», dann mit (2) `include` auf «true» und `message` auf «false» und zum Schluss mit (3) `include` auf «false» und `message` auf «false». Schaut, was sich im gerenderten Dokument zwischen diesen drei Varianten verändert.

Beispiel

```
library(dplyr)
```

Aufgabe 2

R hat den Beispieldatensatz `mtcars`. Dieser Datensatz muss daher nicht extra eingelesen werden, man kann ihn einfach verwenden.

```
mtcars |>
  head()
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Erklärt, was `head()` macht und was sich verändert, wenn man dort «3» in die Klammer schreibt.

Antwort: *hier*

Wie heisst das Zeichen nach `mtcars`? Welches Zeichen gibt es noch, welches dasselbe macht und auch gleich heisst? Wie unterscheiden sie sich?

Antwort: *hier*

Aufgabe 3

Erklärt, was in den folgenden Code-Chunks passiert.

```
mtcars |>
  select(cyl, hp)|>
  head()
```

	cyl	hp
Mazda RX4	6	110
Mazda RX4 Wag	6	110
Datsun 710	4	93
Hornet 4 Drive	6	110
Hornet Sportabout	8	175
Valiant	6	105

Antwort: *hier*

```
mtcars |>
  filter(hp > 180)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Duster 360	14.3	8	360	245	3.21	3.570	15.84	0	0	3	4
Cadillac Fleetwood	10.4	8	472	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440	230	3.23	5.345	17.42	0	0	3	4
Camaro Z28	13.3	8	350	245	3.73	3.840	15.41	0	0	3	4
Ford Pantera L	15.8	8	351	264	4.22	3.170	14.50	0	1	5	4
Maserati Bora	15.0	8	301	335	3.54	3.570	14.60	0	1	5	8

Antwort: *hier*

```
mtcars |>
  group_by(cyl) |>
  summarise(durschn_hp = mean(hp))
```

```
# A tibble: 3 x 2
  cyl durschn_hp
<dbl> <dbl>
1     4      82.6
2     6     122.
3     8     209.
```

Antwort: *hier*

```
mtcars |>
  group_by(cyl) |>
  summarise(durschn_hp = mean(hp)) |>
  mutate(ueber100hp = if_else(durschn_hp > 100, "Jep", "Nö"))
```

```
# A tibble: 3 x 3
  cyl durschn_hp ueber100hp
<dbl> <dbl> <chr>
1     4      82.6 Nö
2     6     122. Jep
3     8     209. Jep
```

Antwort: *hier*

```
mtcars_hp_tbl <- mtcars |>
  group_by(cyl)|>
  summarise(durschn_hp = mean(hp)) |>
  mutate(ueber100h = if_else(durschn_hp > 100, "Jep", "Nö"))
```

Antwort: *hier*

```
library(ggplot2)
mtcars |>
  ggplot(aes(x = mpg, y = hp)) +
  geom_point(size = 2, color = "#002fa7") +
  theme_bw() +
  labs(
    x = "Verbrauch in mpg",
    y = "Pferdestärke (PS)",
    title = "Verbrauch und PS") +
  geom_hline(
    yintercept = mean(mtcars$hp),
    linetype = "dotted",
    color = "blue") +
  annotate("text", x = max(mtcars$mpg) - 5, y = mean(mtcars$hp) + 10,
    label = paste("Durchschnitt:", round(mean(mtcars$hp)), "PS"),
    color = "midnightblue", hjust = 1)
```

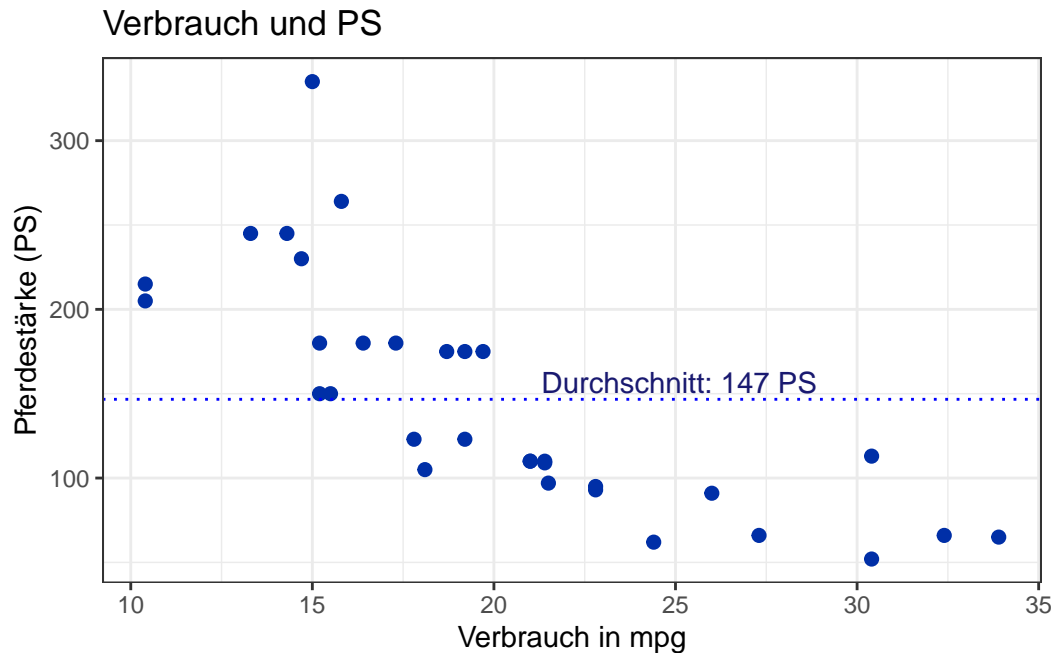


Figure 2: Was ist das für ein Diagramm?

Antwort: *hier*

Aufgabe 4

Gehe in die YAML (das ist das Ding ganz oben, wo `title` usw. steht) und ändere `toc` auf `«true»`. Was verändert sich nun im gerenderten Dokument?

Antwort: *hier*

Aufgabe 5

Den Beispieldatensatz `mtcars` muss man nicht extra einlesen. Andere Datensätze müssen in R importiert werden. Im Unterordner `«data»` findet ihr den Datensatz zur Schweizer Nachwahlbefragung (Selects) aus dem Jahr 2019.

```
df_selects <- readr::read_csv("data/Selects2019.csv")
```

Erkläre, was hier passiert.

Antwort: *hier*

Was macht `readr::?`? Braucht es das?

Antwort: *hier*

Aufgabe 6

Der Datensatz ist nun in «df_selects» gespeichert. Anstatt wie oben «mtcars» zu schreiben, schreibst du nun «df_selects».

1. Berechne das Durchschnittsalter der Befragten. Tipp: Du brauchst dafür einen Pipe-Operator und zwei Funktionen, `mean()` und `summarise()`.
2. Berechne, wie viele Männer und Frauen befragt wurden. Tipp: Du brauchst dafür einen Pipe-Operator und eine Funktion, `count()`.
3. Berechne, wie viele Frauen unter 30 befragt wurden. Tipp: Du brauchst dafür zwei Pipe-Operatoren und zwei Funktionen, `filter()` und `count()`.
4. Berechne das Durchschnittsalter für die Wähler jeder Partei. Tipp: Du brauchst dafür zwei Pipe-Operatoren und drei Funktionen, `group_by()`, `summarise()` und `mean()`.
5. Wähle mit `select()` die Spalten für das Alter, das Geschlecht und die Religion aus. Erstelle mit `mutate()` eine neue Spalte/Variable, die den Namen «religioesTRUE» hat. Die Variable «religioesTRUE» soll 1 sein, wenn die Person irgendeine Religion hat und 0, wenn sie keine hat. Verwende dazu `if_else()`.