# Daten Import & Daten Management & Kollaboratives Arbeiten mit GitHub II

rstatsZH - Data Science mit R

Lars Schöbitz

Oct 15, 2024

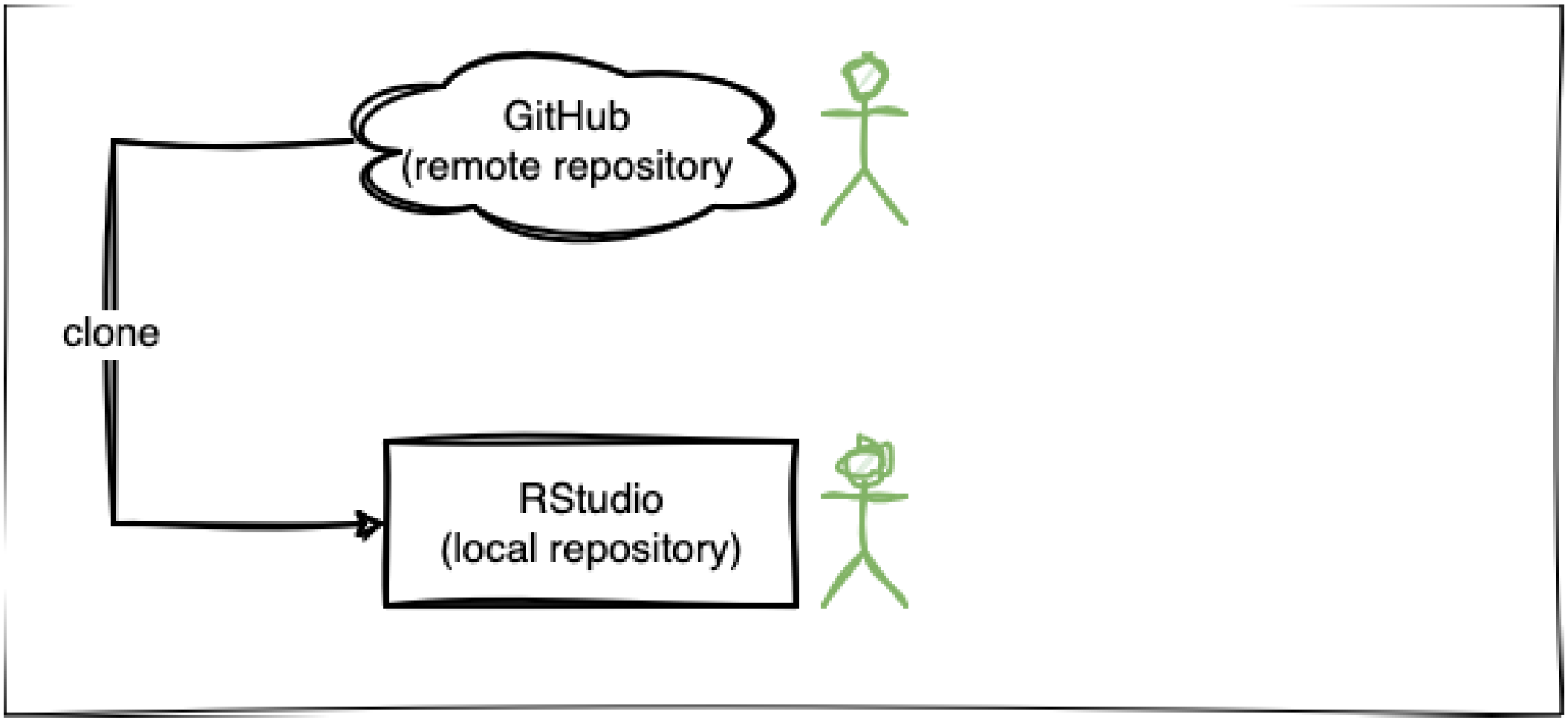
rstatszh-k009.github.io/website/

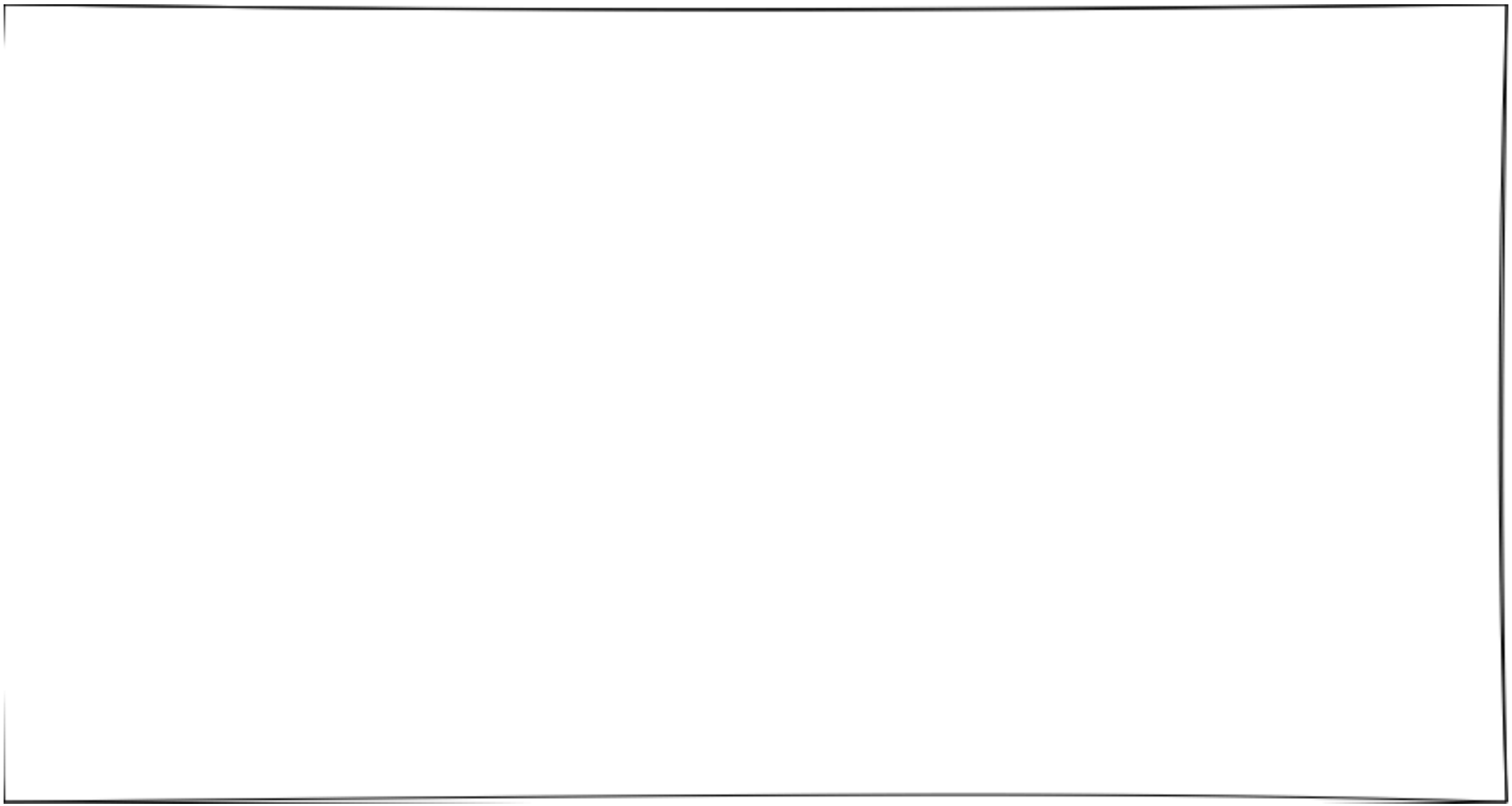
# Lernziele (für diese Woche)

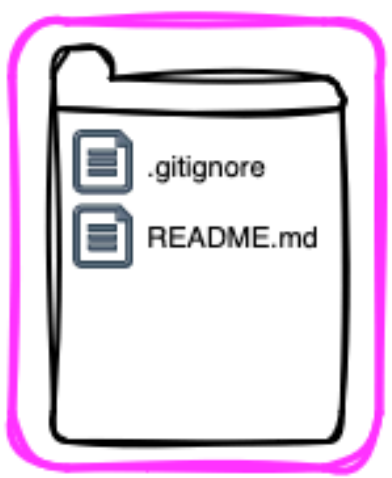
1. Die Lernenden können Daten aus Dateien im CSV und XLSX-Format importieren, die sich in Unterverzeichnissen des Stammverzeichnisses, und auf GitHub, befinden.
2. Die Lernenden können den Unterschied zwischen drei Arten von Daten erörten: (1) unverarbeitete Rohdaten; (2) verarbeitete, analysefähige Daten, und (3) Daten, die einer Veröffentlichung zugrunde liegen.
3. Die Lernenden können die Anwendung der Git Befehle clone, commit, push beschreiben.
4. Die Lernenden können die Begriffe local und remote Repository unterscheiden.

# Git Befehle

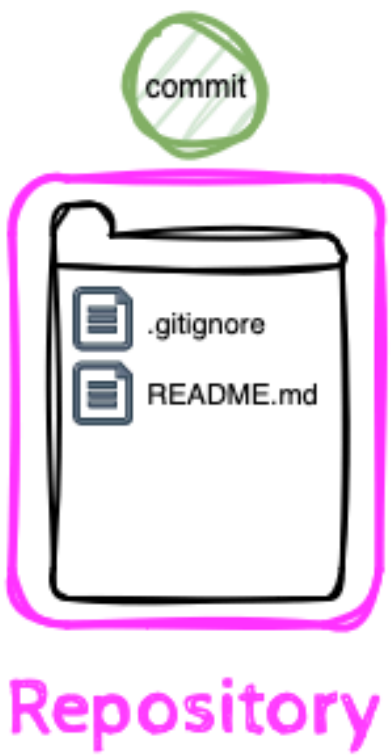

GitHub
(remote repository


GitHub
(remote repository
clone
RStudio
(local repository)





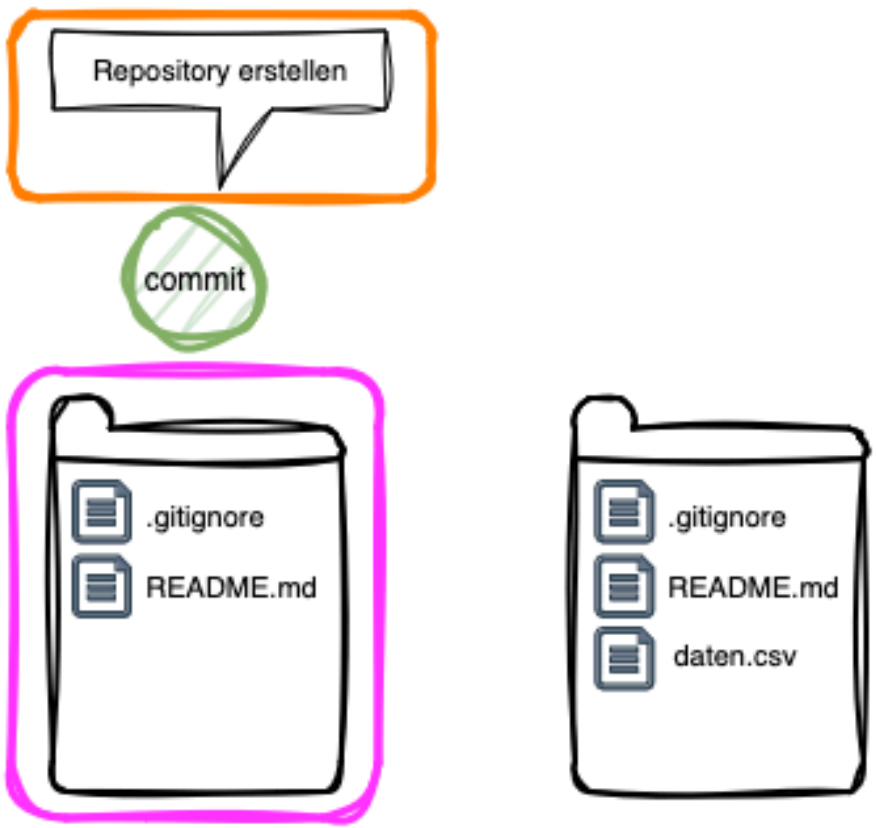

Repository


commit
.gitignore
README.md
Repository

# Commit message

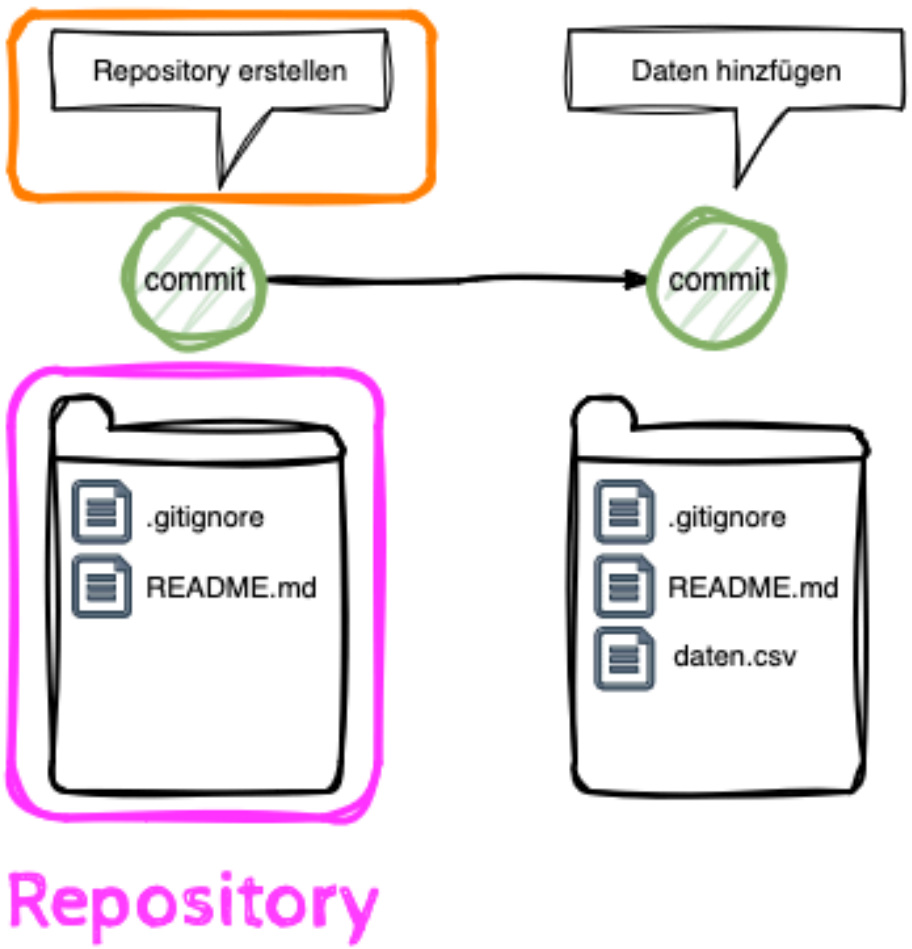

Repository

# Commit message



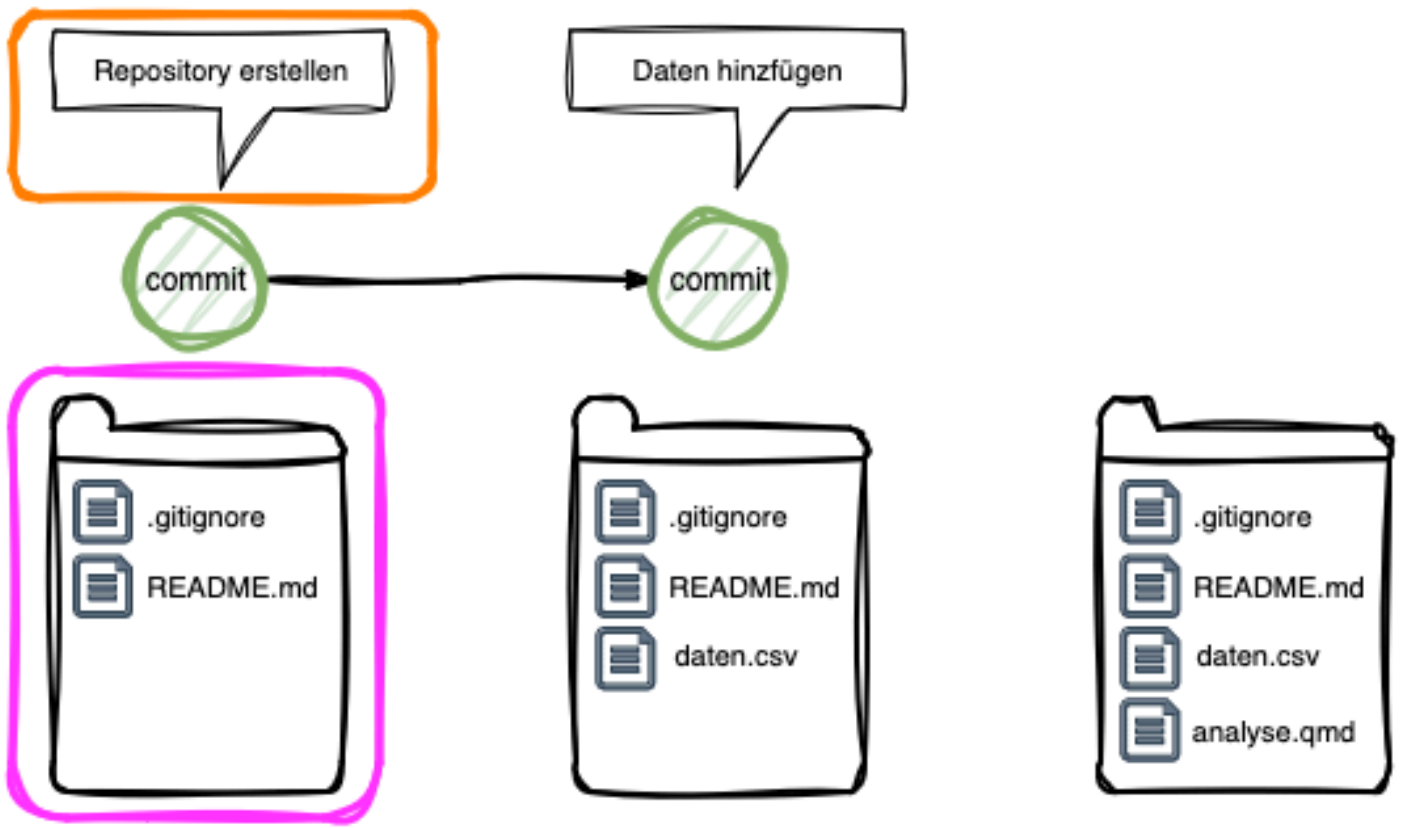

Repository

# Commit message



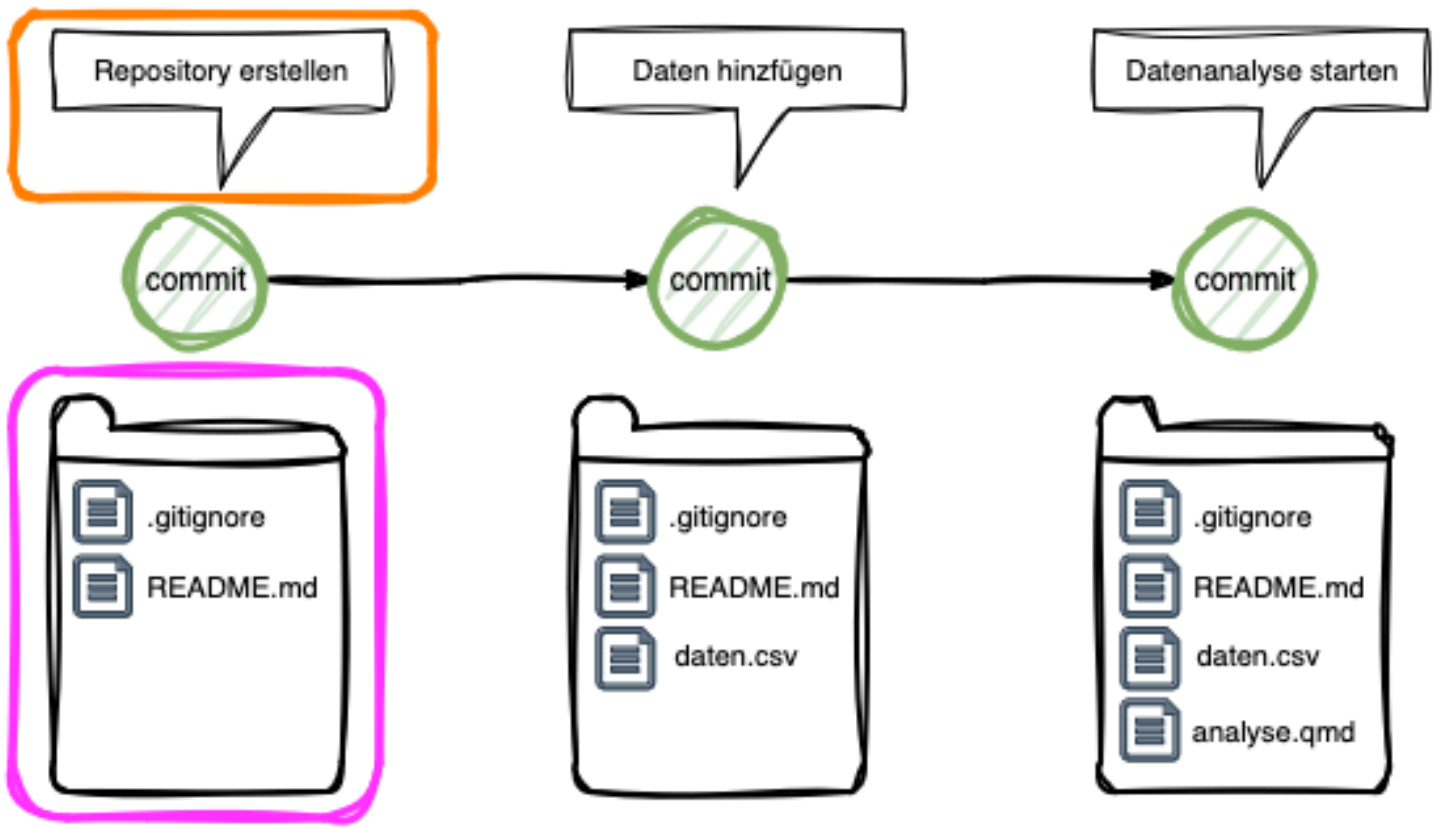

Repository

# Commit message



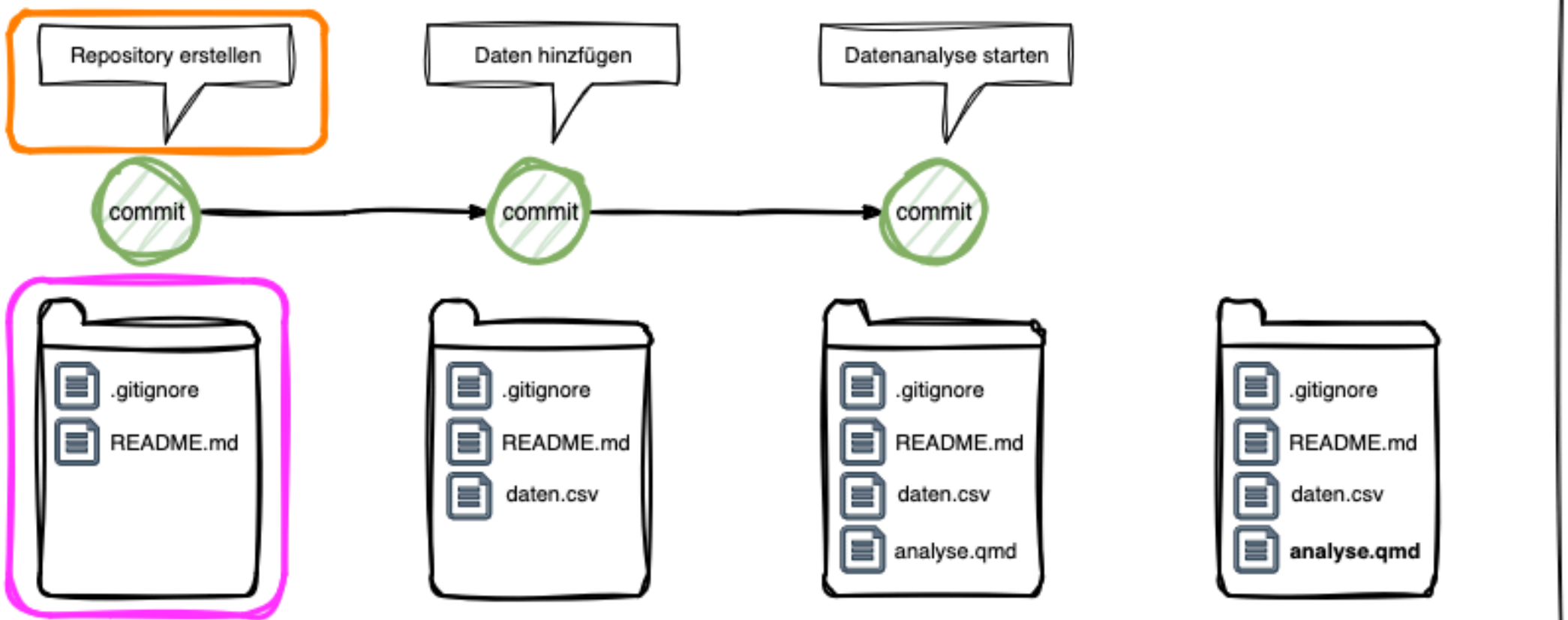

Repository

# Commit message

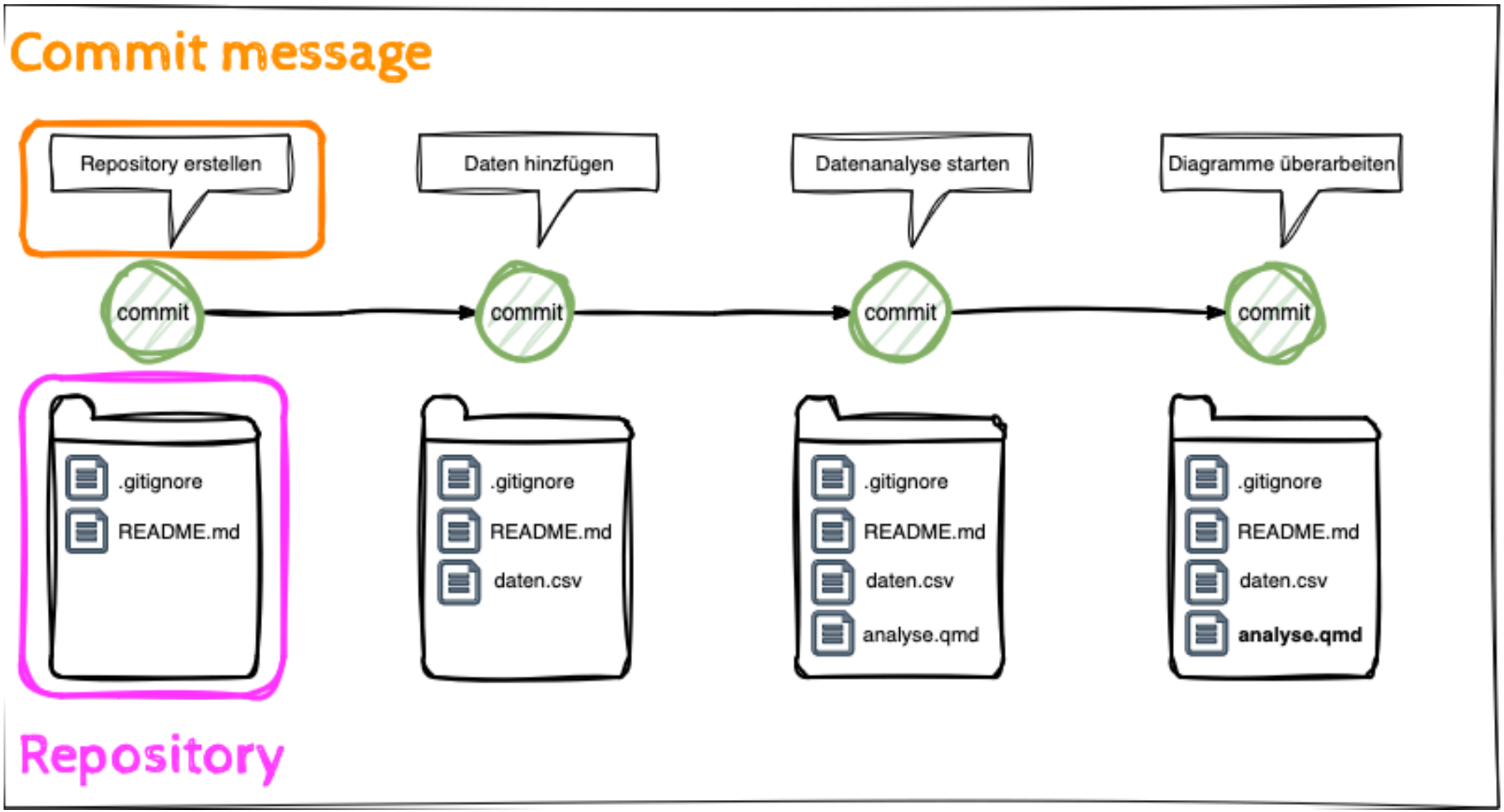


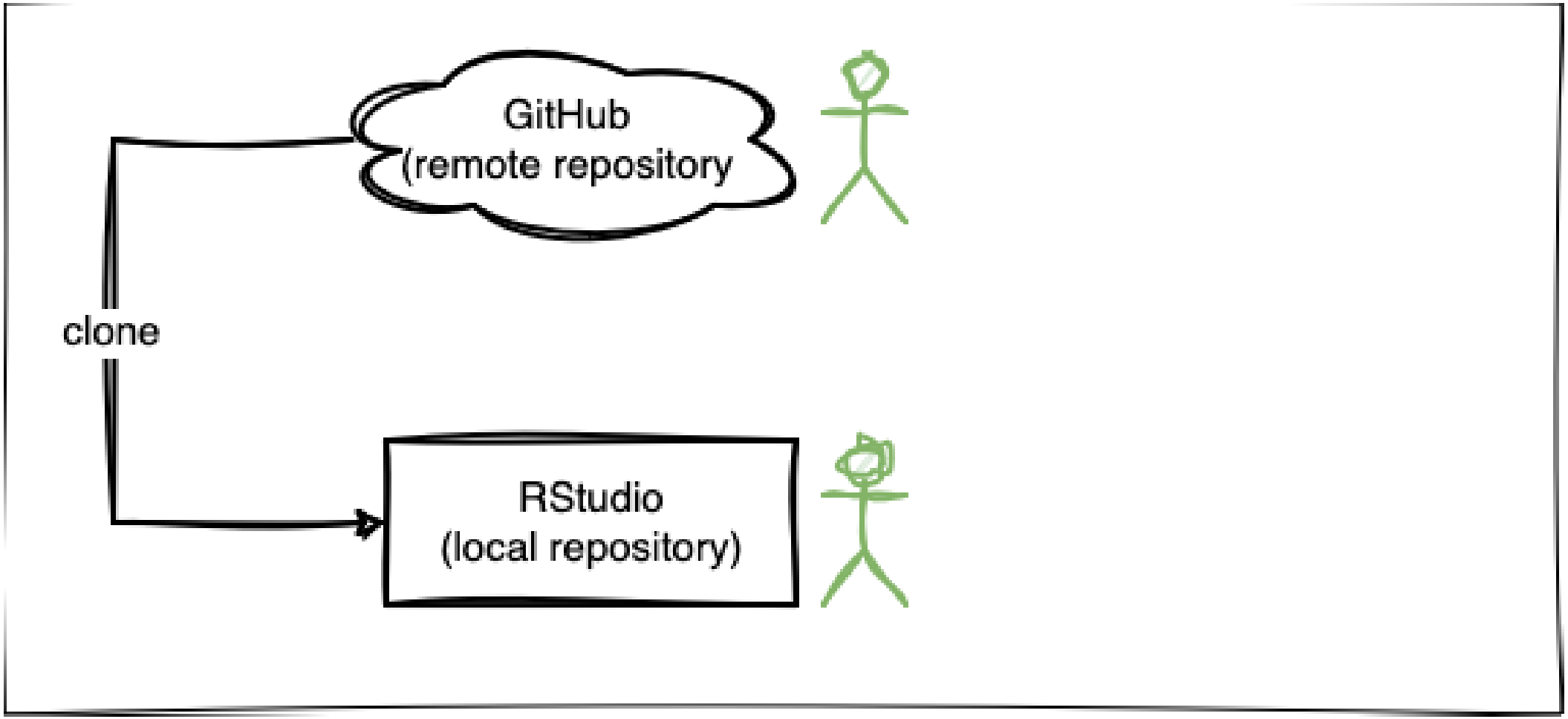

# Repository

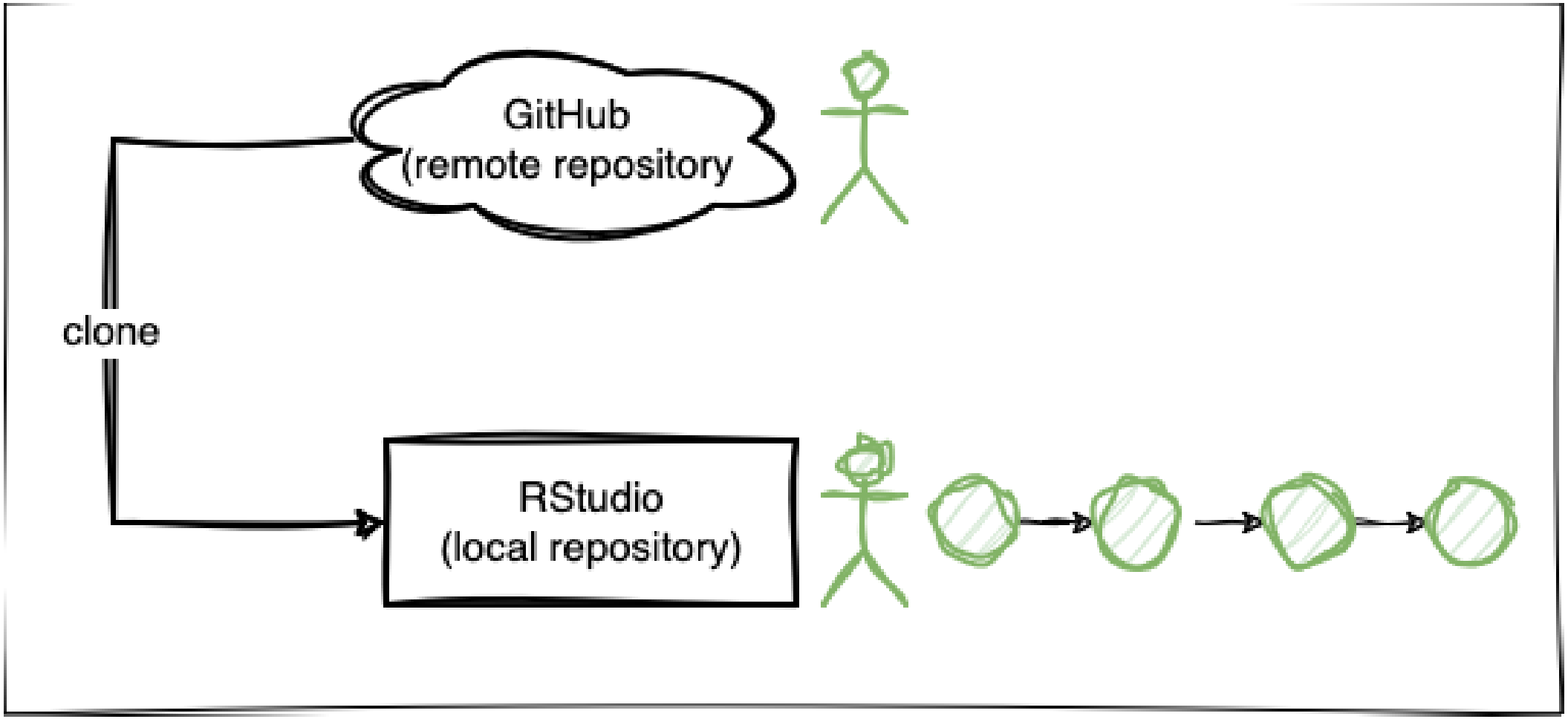
# Commit message

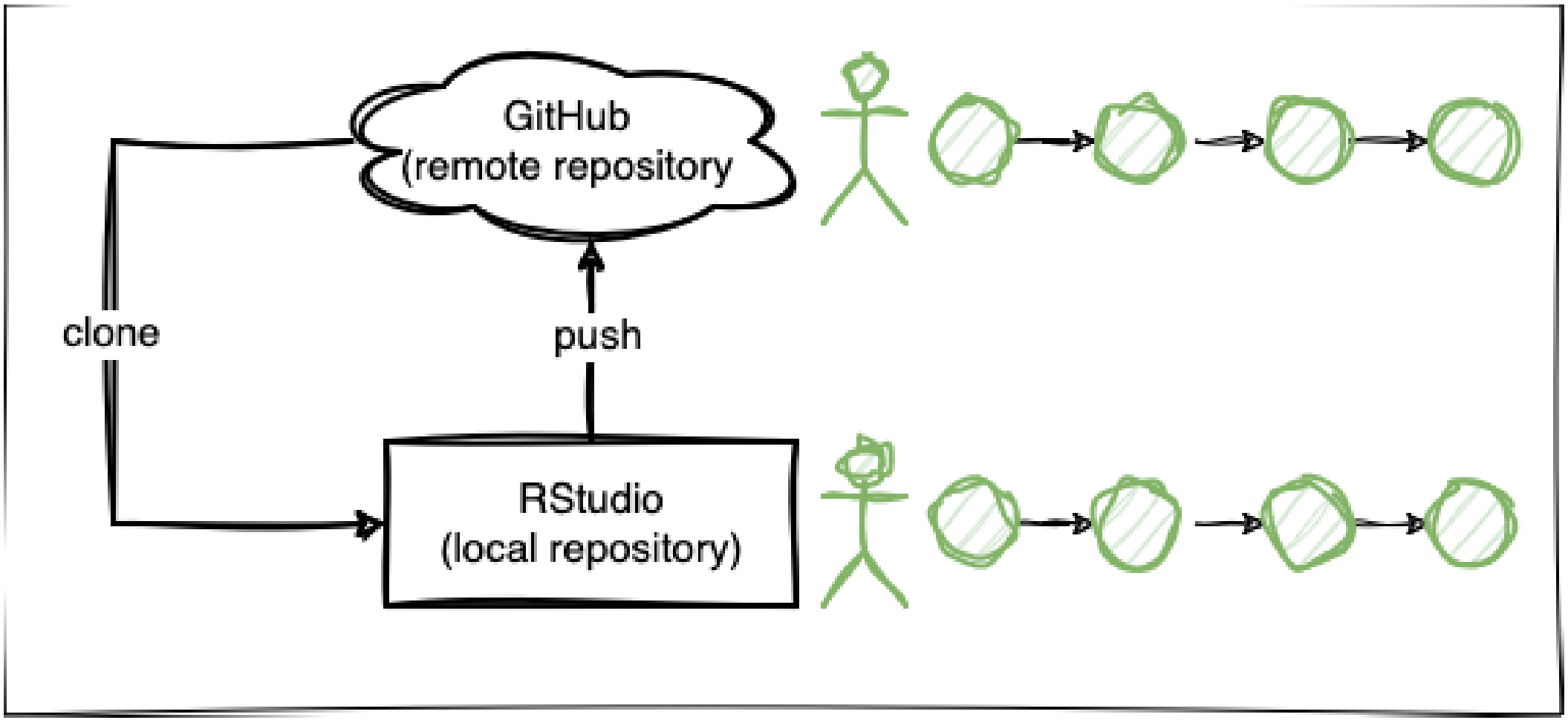



Repository


Commit message
Repository erstellen
Daten hinzfügen
Datenanalyse starten
Diagramme überarbeiten
commit
commit
commit
commit
.gitignore
README.md
.gitignore
README.md
daten.csv
.gitignore
README.md
daten.csv
analyse.qmd
.gitignore
README.md
daten.csv
analyse.qmd
Repository


GitHub
(remote repository
clone
RStudio
(local repository)


GitHub
(remote repository
clone
RStudio
(local repository)


GitHub
(remote repository
clone
push
RStudio
(local repository)

# Faktoren in R

# Variablen Typen

## Numerisch

### Diskrete Variablen

- nicht negative
- zählbare
- ganze Zahlen
- z.B. Anzahl Schüler, Würfelwurf

### Stetige (kontinuierliche) Variablen

- unendliche Anzahl von Werten
- zwischen zwei Werten
- auch Datums/Uhrzeitwerte
- z.B. Länge, Gewicht, Grösse

## Nicht numerisch

### Kategoriale Variablen

- endliche Anzahl von Werten
- eindeutige Gruppen (z.B. EU Länder)
- ordinal, wenn diese eine logische Reihenfolge/Rangordnung aufweisen (z.B. Wochentage)

# ordinal skalierte Daten in R

- ordinal skalierte Daten sind kategoriale Daten, die eine logische Reihenfolge aufweisen
- in R werden Text-Daten standardmässig als `character` gespeichert
- Beurteilungen: sehr gut, gut, mittel, schlecht, sehr schlecht
- die Reihenfolge von Text Daten ist aplhabetisch

```
1 df |>
2     arrange(beurteilung)
```

```
# A tibble: 5 × 2
  name    beurteilung
  <chr>   <chr>
1 Bob     gut
2 Charlie mittel
3 Diana   schlecht
4 Alice   sehr gut
5 Eve     sehr schlecht
```

# ordinal skalierte Daten in R

- in R können wir ordinal skalierte Daten mit dem `factor` Datentyp speichern
- die Level geben die Reihenfolge der Kategorien an
- die Umwandlung beeinflusst das Verhalten der Daten in Tabellen und Diagrammen

```r
# Faktor Level werden in einem Vektor definiert
beurteilung_level <- c("sehr schlecht", "schlecht", "mittel", "gut", "sehr gut")

df |>
    # Die Spalte wird in einen Faktor umgewandelt
    mutate(beurteilung = factor(beurteilung, levels = beurteilung_level)) |>
    # Die Tabelle wird nach der Reihenfolge sortiert
    arrange(beurteilung)
```

```
# A tibble: 5 × 2
  name    beurteilung
  <chr>   <fct>
1 Eve     sehr schlecht
2 Diana   schlecht
3 Charlie mittel
4 Bob     gut
5 Alice   sehr gut
```

# Ich bin dran: Faktoren in R

Zurücklehnen und genießen!

# Pause machen

Bitte steh auf und beweg dich. Lasst eure E-Mails in Frieden ruhen.

# Ihr seid dran: 02-faktoren-ihr.qmd

1. Öffne posit.cloud in deinem Browser (verwende dein Lesezeichen).
2. Öffne den rstatszh-k009 Arbeitsbereich (Workspace) für den Kurs.
3. Klicke auf `Start` neben `md-04-uebungen`.
4. Suche im Dateimanager im Fenster unten rechts die Datei `02-faktoren-ihr.qmd` und klicke darauf, um sie im Fenster oben links zu öffnen.
5. Folge den Anweisungen in der Datei.

# Daten einlesen

# Rechteckige Daten in R einlesen


# CSV & XLSX

## readr

- `read_csv()` - durch Kommas getrennte Werte
- `read_csv2()` - durch Semicolon getrennte Werte (Tipp für das Konvertieren von xlsx als csv)
- `read_tsv()` - durch Tab getrennte Werte
- `read_delim()` - liest Dateien mit beliebigem Trennzeichen

## readxl

- `read_excel()` - liest xls oder xlsx Dateien

# Daten aus CSV-Dateien lesen

- Import von unbearbeiteten Rohdaten

```
1 befragung <- read_csv("raw/KTZH_00001341_00002759_frage7a1.csv")
```

```
1 befragung
```

```
# A tibble: 1,213 × 5
   geschlecht alter     gemeinde_groesse bezirk_name antwort
   <chr>      <chr>     <chr>            <chr>       <chr>
 1 weiblich   55 bis 59 Winterthur       Winterthur  viel zu hoch
 2 männlich   70 bis 74 Winterthur       Winterthur  eher zu hoch
 3 weiblich   55 bis 59 10001 bis 20000  Hinwil      eher zu hoch
 4 weiblich   35 bis 39 20001 bis 50000  Hinwil      eher zu hoch
 5 weiblich   50 bis 54 5001 bis 10000   Meilen      eher zu hoch
 6 männlich   35 bis 39 <1000            Andelfingen gerade angemessen
 7 weiblich   45 bis 49 10001 bis 20000  Pfäffikon   viel zu hoch
 8 männlich   30 bis 34 <1000            Winterthur  eher zu hoch
 9 weiblich   50 bis 54 5001 bis 10000   Winterthur  eher zu hoch
10 weiblich   80+       20001 bis 50000  Uster       eher zu hoch
# ℹ 1,203 more rows
```

# Daten als CSV-Datei schreiben

- transformiere Daten
- exportiere verarbeitete, analysereife Daten

```
antwort_levels <- c("viel zu hoch", "eher zu hoch", "gerade angemessen",
                    "eher zu tief", "viel zu tief")

befragung_fct <- befragung |>
    mutate(antwort = factor(antwort, levels = antwort_levels))
```

```
befragung_fct
```

```
# A tibble: 1,213 × 5
   geschlecht alter     gemeinde_groesse bezirk_name antwort
   <chr>      <chr>     <chr>            <chr>       <fct>
 1 weiblich   55 bis 59 Winterthur       Winterthur  viel zu hoch
 2 männlich   70 bis 74 Winterthur       Winterthur  eher zu hoch
 3 weiblich   55 bis 59 10001 bis 20000  Hinwil      eher zu hoch
 4 weiblich   35 bis 39 20001 bis 50000  Hinwil      eher zu hoch
 5 weiblich   50 bis 54 5001 bis 10000   Meilen      eher zu hoch
 6 männlich   35 bis 39 <1000            Andelfingen gerade angemessen
 7 weiblich   45 bis 49 10001 bis 20000  Pfäffikon   viel zu hoch
 8 männlich   30 bis 34 <1000            Winterthur  eher zu hoch
 9 weiblich   50 bis 54 5001 bis 10000   Winterthur  eher zu hoch
10 weiblich   80+       20001 bis 50000  Uster       eher zu hoch
# ℹ 1,203 more rows
```

```
write_csv(befragung_fct, "daten/processed/ktzh-befragung-zufriedenheit.csv")
```

# Analysefertige Daten einlesen

- Was ist aus unserem Faktor geworden?

```
1 befragung_fct <- read_csv("daten/processed/ktzh-befragung-zufriedenheit.csv")
```

```
1 befragung_fct
```

```
# A tibble: 1,213 × 5
   geschlecht alter     gemeinde_groesse bezirk_name antwort
   <chr>      <chr>     <chr>            <chr>       <chr>
 1 weiblich   55 bis 59 Winterthur       Winterthur  viel zu hoch
 2 männlich   70 bis 74 Winterthur       Winterthur  eher zu hoch
 3 weiblich   55 bis 59 10001 bis 20000  Hinwil      eher zu hoch
 4 weiblich   35 bis 39 20001 bis 50000  Hinwil      eher zu hoch
 5 weiblich   50 bis 54 5001 bis 10000   Meilen      eher zu hoch
 6 männlich   35 bis 39 <1000            Andelfingen gerade angemessen
 7 weiblich   45 bis 49 10001 bis 20000  Pfäffikon   viel zu hoch
 8 männlich   30 bis 34 <1000            Winterthur  eher zu hoch
 9 weiblich   50 bis 54 5001 bis 10000   Winterthur  eher zu hoch
10 weiblich   80+       20001 bis 50000  Uster       eher zu hoch
# ℹ 1,203 more rows
```

# Wo ist der Faktor?

# Wie speichern wir Faktoren?

- In R können Daten als `.rds` Datei gespeichert werden
- `.rds` Dateien speichern die Struktur der Daten
- Faktoren und andere Datenstrukturen bleiben erhalten

```
1 write_rds(befragung_fct, "folien/daten/processed/ktzh-befragung-zufriedenheit.rds")
```

```
1 befragung_rds <- read_rds("daten/processed/ktzh-befragung-zufriedenheit.rds")
```

```
1 befragung_rds
```

```
# A tibble: 1,213 × 5
   geschlecht alter     gemeinde_groesse bezirk_name antwort
   <chr>      <chr>     <chr>            <chr>       <fct>
 1 weiblich   55 bis 59 Winterthur       Winterthur  viel zu hoch
 2 männlich   70 bis 74 Winterthur       Winterthur  eher zu hoch
 3 weiblich   55 bis 59 10001 bis 20000  Hinwil      eher zu hoch
 4 weiblich   35 bis 39 20001 bis 50000  Hinwil      eher zu hoch
 5 weiblich   50 bis 54 5001 bis 10000   Meilen      eher zu hoch
 6 männlich   35 bis 39 <1000            Andelfingen gerade angemessen
 7 weiblich   45 bis 49 10001 bis 20000  Pfäffikon   viel zu hoch
 8 männlich   30 bis 34 <1000            Winterthur  eher zu hoch
 9 weiblich   50 bis 54 5001 bis 10000   Winterthur  eher zu hoch
10 weiblich   80+       20001 bis 50000  Uster       eher zu hoch
# ℹ 1,203 more rows
```

# Daten zusammenfassen

- für eine Visualisierung oder Tabelle in einer veröffentlichten Arbeit

```r
befragung_sum_alter <- befragung_rds |>
    group_by(alter, antwort) |>
    summarise(antwort_anzahl = n()) |>
    mutate(antwort_prozent = antwort_anzahl / sum(antwort_anzahl) * 100)
```

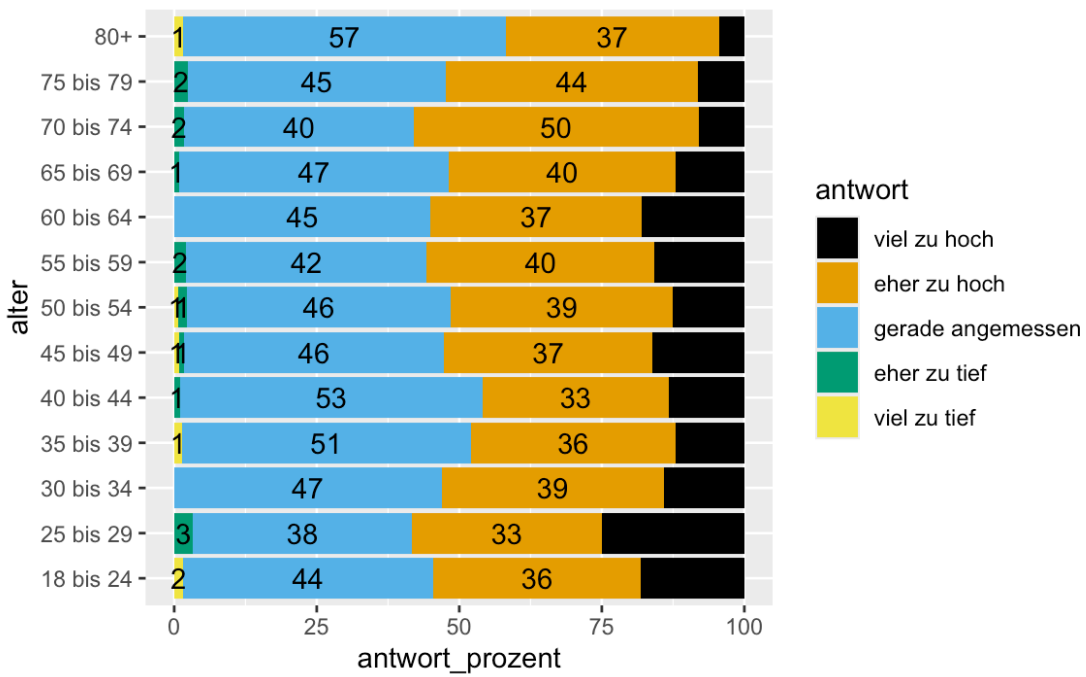
```r
befragung_sum_alter
```

```
# A tibble: 52 × 4
# Groups:   alter [13]
   alter     antwort           antwort_anzahl antwort_prozent
   <chr>     <fct>                      <int>           <dbl>
 1 18 bis 24 viel zu hoch                  12           18.2
 2 18 bis 24 eher zu hoch                  24           36.4
 3 18 bis 24 gerade angemessen             29           43.9
 4 18 bis 24 viel zu tief                   1            1.52
 5 25 bis 29 viel zu hoch                  15           25
 6 25 bis 29 eher zu hoch                  20           33.3
 7 25 bis 29 gerade angemessen             23           38.3
 8 25 bis 29 eher zu tief                   2            3.33
 9 30 bis 34 viel zu hoch                   9           14.1
10 30 bis 34 eher zu hoch                  25           39.1
# ℹ 42 more rows
```

# Daten visualisieren

- in einer Veröffentlichung

```
ggplot(data = befragung_sum_alter,
       mapping = aes(x = antwort_prozent,
                     y = alter,
                     fill = antwort)) +
    geom_col() +
    geom_text(aes(label = round(antwort_prozent, 0)),
              position = position_stack(vjust = 0.5)) +
    scale_fill_colorblind()
```




rstatszh-k009.github.io/website/

# Daten exportieren

- Daten, die einer Veröffentlichung zugrunde liegen
- als CSV-Datei
- erhöht die Wiederverwendbarkeit

```
write_csv(befragung_sum_alter,
          "daten/final/ktzh-befragung-zufriedenheit-sum.csv")
```

# Daten Management

Beispiele für Begriffe, die bei der Datenverwaltung verwendet werden.

| Begriff | Ordnername | Erklärung | Dateiformat |
|---|---|---|---|
| unbearbeitete Rohdaten | data/raw | Daten, die nicht bearbeitet wurden und in ihrer ursprünglichen Form und Datei bleiben | often XLSX, also CSV, JSON, and others |

# Daten Management

Beispiele für Begriffe, die bei der Datenverwaltung verwendet werden.

| Begriff | Ordnername | Erklärung | Dateiformat |
|---|---|---|---|
| unbearbeitete Rohdaten | data/raw | Daten, die nicht bearbeitet wurden und in ihrer ursprünglichen Form und Datei bleiben | often XLSX, also CSV, JSON, and others |
| verarbeitete, analysefähige Daten | data/processed | Daten, die zur Vorbereitung einer Analyse verarbeitet werden und in ihrer neuen Form als neue Datei gespeichert werden | CSV, RDS, JSON |

# Daten Management

Beispiele für Begriffe, die bei der Datenverwaltung verwendet werden.

| Begriff | Ordnername | Erklärung | Dateiformat |
|---|---|---|---|
| unbearbeitete Rohdaten | data/raw | Daten, die nicht bearbeitet wurden und in ihrer ursprünglichen Form und Datei bleiben | often XLSX, also CSV, JSON, and others |
| verarbeitete, analysefähige Daten | data/processed | Daten, die zur Vorbereitung einer Analyse verarbeitet werden und in ihrer neuen Form als neue Datei gespeichert werden | CSV, RDS, JSON |
| Daten, die einer Veröffentlichung zugrunde liegen | data/final | Daten, die das Ergebnis einer Analyse sind (z. B. deskriptive Statistik oder Datenvisualisierung) und in einem Bericht angezeigt werden, dann aber auch in ihrer neuen Form als neue Datei exportiert werden | CSV |

# Pause machen

Bitte steh auf und beweg dich. Lasst eure E-Mails in Frieden ruhen.

# Ihr seid dran: 03-daten-import-ihr.qmd

1. Öffne posit.cloud in deinem Browser (verwende dein Lesezeichen).
2. Öffne den rstatszh-k009 Arbeitsbereich (Workspace) für den Kurs.
3. Klicke auf `Continue` neben `md-04-uebungen`.
4. Suche im Dateimanager im Fenster unten rechts die Datei `03-daten-import-ihr.qmd` und klicke darauf, um sie im Fenster oben links zu öffnen.
5. Folge den Anweisungen in der Datei.

# Zeitpuffer: Modul 4

1. Die Lernenden können Daten aus Dateien im CSV und XLSX-Format importieren, die sich in Unterverzeichnissen des Stammverzeichnisses, und auf GitHub, befinden.
2. Die Lernenden können den Unterschied zwischen drei Arten von Daten erörten: (1) unverarbeitete Rohdaten; (2) verarbeitete, analysefähige Daten, und (3) Daten, die einer Veröffentlichung zugrunde liegen.
3. Die Lernenden können die Anwendung der Git Befehle clone, commit, push beschreiben.
4. Die Lernenden können die Begriffe local und remote Repository unterscheiden.

## Welche Konzepte kann ich nochmals erklären?

# Zusatzaufgaben Modul 4

# Modul 4 Dokumentation

rstatszh-k009.github.io/website/module/md-04.html

# Zusatzaufgaben Abgabedatum

- Abgabedatum: Montag, 21. Oktober
- Korrektur- und Feedbackphase bis zu: Donnerstag, 24. Oktober

# Danke

rstatszh-k009.github.io/website/

# Danke! 🌻

Folien erstellt mit revealjs und Quarto: https://quarto.org/docs/presentations/revealjs/ Access slides als PDF auf GitHub

Alle Materialien sind lizenziert unter Creative Commons Attribution Share Alike 4.0 International.